Министерство образования и науки РФ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Ульяновский государственный университет»
Факультет математики и информационных технологий

УЧЕНЫЕ ЗАПИСКИ

Ульяновского государственного университета

Серия МАТЕМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Выпуск 1(4)

Под ред. проф. А. А. Смагина

Ульяновск, 2012

ББК 32.97
У91

Печатается по решению Ученого совета факультета математики и информационных технологий Ульяновского государственного университета

У91 Ученые записки Ульяновского государственного университета. Сер. Математика и информационные технологии. Вып. 1(4) / Под ред. проф. А. А. Смагина. – Ульяновск: УлГУ, 2012. - 286 с.

Научное издание

УЧЕНЫЕ ЗАПИСКИ
Ульяновского государственного университета

Серия
МАТЕМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Выпуск 1(4)

Под ред. проф. ***А.А. Смагина***

Печатается в авторской редакции

Директор Издательского центра – *Т.В. Филиппова*

Подписано в печать 25.12.2012.
Формат 60x84/8. Гарнитура Times New Roman.
Усл. п.л. 33,2. Уч.-изд. л. 28,4. Тираж 100 экз.
Заказ №228/

Отпечатано в Издательском центре
Ульяновского государственного университета
432000, г. Ульяновск, ул. Л.Толстого, 42

Содержание

ИССЛЕДОВАНИЕ УГЛОВОГО ДВИЖЕНИЯ РАЗГОННОГО БЛОКА С ТОРОИДАЛЬНЫМ БАКОМ, ЦЕЛИКОМ ЗАПОЛНЕННЫМ ТОПЛИВОМ[1]

А.В.Алексеев, В.С.Красников

Самарский государственный аэрокосмический университет имени академика С.П.Королёва (национальный исследовательский университет)

1. Состояние проблемы.

Проблема исследования движения твердых тел с полостями, частично или полностью заполненными жидкостью, была и остается одной из важных проблем теоретической механики, имеющей большое практическое значение для задач современной механики космического полета. Среди первых научных исследований в рамках указанной проблемы следует указать работу Жуковского Н.Е. [11], в которой, в основном, рассматривалось движение тел с идеальной невязкой жидкостью, целиком заполняющей полость. В работе [11] приведены уравнения пространственного движения твердого тела с идеальной жидкостью:

$$\begin{cases} L = A\dfrac{dp}{dt} + (C-B)qr + Rq - Qr; \\ M = B\dfrac{dq}{dt} + (A-C)rp + Pr - Rp; \\ N = C\dfrac{dr}{dt} + (B-A)pq + Qp - Pq, \end{cases}$$

где A, B, C – моменты инерции относительно осей связанной с телом системы координат, L, M, N – проекции моментов внешних сил, ω_i – проекции вектора угловой скорости тела, P, Q, R – проекции главного момента количеств движения жидкости на оси связанной с телом системы координат.

Движение тел с жидкостью, имеющей свободную поверхность, рассматривалась в работе Моисеева Н.Н. и Румянцева В.В. [15]. В данной работе получены уравнения движения твердого тела с жидкостью:

$$\frac{d\mathbf{Q}}{dt} + \boldsymbol{\omega} \times \mathbf{Q} = \mathbf{K},$$

$$\frac{d\mathbf{G}}{dt} + \boldsymbol{\omega} \times \mathbf{G} + \boldsymbol{\upsilon}_0 \times \mathbf{Q} = \mathbf{L},$$

$$\frac{d\boldsymbol{\upsilon}}{dt} + \boldsymbol{\omega} \times \boldsymbol{\upsilon} = \mathbf{F} - \frac{1}{\rho}\operatorname{grad} p,$$

где $\mathbf{Q}$ – вектор количества движения системы, $\boldsymbol{\omega}$ – вектор угловой скорости тела, $\mathbf{K}$ и $\mathbf{L}$ – главный вектор и главный момент всех приложенных к системе активных сил, $\mathbf{G}$ – кинетический момент системы, $\boldsymbol{\upsilon}_0$ – скорость центра подвижной системы отсчета, $\boldsymbol{\upsilon}$ – скорость какой-либо точки системы, $\mathbf{F}$ – вектор массовой силы, отнесенный к единице массы, p – гидродинамическое давление, ρ – плотность жидкости, кроме того, к данным уравнениям необходимо добавить уравнение несжимаемости (уравнение связи) и граничные условия.

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0230 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

В труде [15] Румянцевым В.В. и в [16] Соболевым С.Л. рассматриваются вращательные движения твердых тел с полостями, заполненными жидкостью, и исследуется устойчивость таких движений.

Работа Черноусько Ф.Л. [17] посвящена исследованиям движений твердых тел с неидеальной жидкостью при больших и малых числах Рейнольдса. При этом методы составления уравнений пространственного движения тел с жидкостью разделены в зависимости от величины кинематической вязкости. При большой вязкости уравнения движения сводятся к обыкновенным дифференциальным, а при малой – к интегро-дифференциальным.

Как видно из вышесказанного, к настоящему времени решено много задач, касающихся вопросов пространственного движения твердых тел с жидкостью. В указанных работах приводятся упрощённые математические модели, исследуются твердые тела, содержащие полости наполненные жидкостью, производится анализ устойчивости стационарных движений, однако, мало внимания уделяется получению аналитических зависимостей параметров движения от времени, которые позволяют проводить непосредственный анализ влияния характеристик системы на движение.

2. Постановка задачи.

Осуществленный обзор научных достижений показывает, что недостаточно широко исследовано угловое движение РБ, как системы твердого тела с тороидальной полостью. Мало работ посвящено получению аналитических зависимостей параметров движения от времени. В связи с этим выделяются следующие конкретные задачи:

- произвести обзор РБ;
- построить математическую модель движения РБ вокруг центра масс;
- провести численное интегрирование дифференциальных уравнений движения;
- получить аналитическое решение динамических уравнений движения;
- получить аналитические зависимости параметров Кэли-Клейна от времени;
- выразить углы Эйлера через параметры Кэли-Клейна.

3. Математическая модель движения относительно центра масс разгонного блока с тороидальным баком.

В настоящей главе на основе теоремы об изменении кинетического момента составляются уравнения движения системы относительно центра масс, которые применяется для анализа углового движения РБ с полостью с жидкостью. Приводятся уравнения движения твердого тела с полостью заполненной жидкостью переменного состава. Интегрируются кинематические и динамические уравнения, по результатам которых находятся численные зависимости параметров движения от времени.

3.1. Построение динамических уравнений движения.

Под простейшими случаями движения твердого тела (ТТ) с полостями, наполненными жидкостью, мы будем подразумевать случаи, когда движение жидкости в полости можно полностью охарактеризовать конечным числом переменных. Очевидно, это возможно лишь при полном заполнении жидкостью полости, когда отсутствует свободная поверхность. Известно [11], что если при этом движение жидкости является потенциальным или же однородным вихревым, то движение системы описывается обыкновенными дифференциальными уравнениями (ОДУ) вместе с уравнением Лапласа. Исследование движения системы в этих случаях значительно упрощается.

В данном пункте результаты Н.Е. Жуковского по исследованию движения ТТ с полостями, целиком заполненными идеальной жидкостью, адаптируются к задаче движения РБ с жидким топливом в тороидальном баке.

Приведем лишь основные уравнения, необходимые в нашей работе [15]. Векторное уравнение движения свободного твердого тела с жидкостью имеет вид:

$$\frac{d\bar{G}}{dt} + \bar{\omega} \times \bar{G} + \bar{\vartheta}_0 \times \bar{Q} = \bar{L}, \tag{1}$$

где G – вектор момента количества движения относительно точки O, Q – количество движения, являющиеся результатом операции сложения соответствующих величин для твердого тела и жидкости. L – момент активных сил относительно точки O всех приложенных к системе активных сил. ϑ_0 - вектор скорости движения точки O.

Отметим также, что в случае, когда полость целиком заполнена жидкостью и за начало подвижных осей принят центр инерции системы, уравнения движения (1) принимают более простой вид:

$$\frac{d\bar{G}}{dt} + \bar{\omega} \times \bar{G} = \bar{L}. \tag{2}$$

Суммарный момент количеств движения твердого тела и жидкости в его полости равен:

$$\bar{G} = \bar{G}_1 + \bar{G}^* + \bar{R},$$

где вектор $\bar{R}$ представляет собой линейную функцию главных циркуляций и является постоянным в связанной системе координат.

Уравнения движения системы относительно центра масс в векторном выражении имеет вид:

$$\frac{d}{dt}(\bar{G}_1 + \bar{G}^*) + \bar{\omega} \times (\bar{G}_1 + \bar{G}^* + \bar{R}) = \bar{L} \tag{3}$$

Такой же вид имеет уравнение движения системы около неподвижной точки.

Согласно Жуковскому, следует, что, заменив жидкость эквивалентным телом, мы вполне заменяем ее механический эффект, каково бы ни было движение ТТ. Так как эллипсоид инерции эквивалентного тела заключает внутри себя эллипсоид инерции жидкости, эквивалентное тело имеет относительно какой-либо оси меньший момент инерции, чем жидкость.

Теорема Жуковского: эффект жидких масс, не имеющих начальных скоростей, тождествен эффектам некоторых эквивалентных ТТ.

Твердое тело с присоединенным к нему эквивалентным телом будем называть, следуя Н.Е. Жуковскому, преобразованным телом. А эквивалентное ТТ, согласно Стоксу, это такое заменяющее жидкость тело, масса которого равна массе жидкости, центр масс совпадает с центром масс жидкости, а эллипсоид инерции для точки O имеет уравнение:

$$r\theta^* r = 1,$$

где r – радиус вектор, θ^* – тензор инерции.

Рассматривая случай, когда R неравен нулю, видно, что уравнение (3) одинаково с уравнением движения вокруг неподвижной точки ТТ с присоединенным к нему вращающимся ротором. Ось вращения этого ротора образует с осями x_1, x_2, x_3 углы, косинусы которых находятся в отношении $R_1 : R_2 : R_3$, а произведение начальной скорости ротора (при неподвижном твердом теле) на момент инерции его относительно оси вращения равно $\left|\bar{R}\right|$.Такая механическая система называется гиростатом.

Если оси абсолютной системы координат направить по главным осям эллипсоида $r(\theta^* + \theta^{(1)})r = 1$, то векторное уравнение (3) в проекциях на оси жестко связной с телом системы координат дает следующие 3 скалярных ОДУ:

$$\begin{cases} A\dot{p} + (C - B)qr + R_3 q - R_2 r = L; \\ B\dot{q} + (A - C)pr + R_1 r - R_3 p = M; \\ C\dot{r} + (B - A)pq + R_2 p - R_1 q = N, \end{cases} \tag{4}$$

полученных Н.Е. Жуковским, где A, B, C обозначают главные моменты преобразованного тела для точки O, равные суммам моментов инерции твердого тела и эквивалентного тела, относительно соответствующих осей координат, направленных по главным осям эллипсоида

инерции для точки *O*. *L, M, N* –проекции внешнего момента на связанные оси координат. *p, q, r* –проекции вектора мгновенной угловой скорости ω на оси связанной системы координат.

В дополнение стоит сказать, что если эллипсоид инерции преобразованного тела является эллипсоидом вращения (*A=B*), проекции момента циклического движения R_1, R_2, R_3 на оси x_1, x_2 отсутствуют $(R_1 = R_2 = 0)$ и момент внешних сил относительно оси симметрии *N*=0, то интеграл вида:

$$\omega_3 = const$$

существует и в том случае, когда полость не имеет формы тела вращения вокруг оси x_3.

Заметим также, что если начальное движение жидкости отсутствует, то *R* равно нулю и уравнения (4) превращаются в уравнения Эйлера движения ТТ около неподвижной точки.

Укажем только, что для всех кольцевидных полостей в форме тел вращения вокруг оси x_3 потенциал циклического движения жидкости:

$$\chi = \frac{\chi}{2\pi}\vartheta,\ \vartheta = arctg\frac{x_2}{x_1}. \tag{5}$$

Проекции вектора *R* момента количеств циклического движения жидкости:

$$R_1 = R_2 = 0,\ R_3 = \rho\int_\tau \frac{d\chi}{d\vartheta}d\tau = \frac{M_{2\chi}}{2\pi}, \tag{6}$$

т.е. для кольцевидных полостей вращения главный момент количеств начального движения равен произведению массы жидкости на циркуляцию скорости, деленную на 2π.

4. Получение аналитических зависимостей параметров движения от времени.

В данной главе производится аналитическое исследование движения твердого тела, содержащего полость с жидкостью. С помощью методов решения дифференциальных уравнений находятся аналитические зависимости проекций угловой скорости на оси связанной системы координат. Используя кинематические дифференциальные уравнения в параметрах Кэли-Клейна, а так же связь между этими параметрами и углами Эйлера получены их аналитические зависимости от времени. С помощью данных зависимостей построены соответствующие графики и дана соответствующая интерпретация полученным зависимостям. Так же было приведено графическое сравнение результатов, полученных численным и аналитическим способами. Рассмотрены предельные случаи ориентации в углах Эйлера, показано преимущество использования кинематических дифференциальных уравнений в параметрах Кэли-Клейна.

4.1. Решение системы динамических уравнений.

Перепишем систему (4) с учетом введенных допущений в пункте 2.3:

$$\begin{cases} A\dot{p} + (C - A)qr + R_3 q = 0; \\ B\dot{q} + (A - C)pr - R_3 p = 0; \\ C\dot{r} = 0. \end{cases} \tag{7}$$

Получим, для свободного разгонного блока, аналитические зависимости параметров движения, со следующими начальными условиями:

$$\begin{aligned} p_0(t) &= p_0; \\ q_0(t) &= q_0; \\ r_0(t) &= r_0, \end{aligned} \tag{8}$$

по которым, в дальнейшем, можно проводить анализ влияния инерционно-массовых характеристик РБ и его начальных условий на движение относительно центра масс.

Для этого решим систему дифференциальных уравнений первого порядка (7).

Из третьего уравнения системы (7), очевидно, что:

$$r(t) = r_0. \tag{9}$$

Подставляя (9) в оставшиеся уравнения системы (7), получим:

$$\begin{cases} A\dot{p} + (C - A)qr_0 + R_3 q = 0; \\ A\dot{q} + (A - C)pr_0 - R_3 p = 0. \end{cases} \quad (10)$$

Введем замену:

$$k = \frac{1}{A}[r_0(C - A) + R_3]. \quad (11)$$

С учетом замены (11) уравнения (10) перепишутся в следующем виде:

$$\begin{cases} \dot{p} + kq = 0; \\ \dot{q} - kp = 0. \end{cases} \quad (12)$$

Решая систему (12) и добавляя решение (9) получим общее решение (7):

$$\begin{cases} p(t) = P\cos kt - K\sin kt; \\ q(t) = K\cos kt + P\sin kt; \\ r(t) = r_0, \end{cases} \quad (13)$$

где константы *K, P,* r_0 определяются из начальных условий (8).

Определяя постоянные интегрирования в (13) по начальным условиям, получим решение системы (7):

$$\begin{cases} p(t) = p_0\cos kt - q_0\sin kt; \\ q(t) = p_0\sin kt + q_0\cos kt; \\ r(t) = r_0. \end{cases} \quad (14)$$

Выражения (14) являются аналитическим решением динамических уравнений, по которым можно анализировать изменение проекции угловых скоростей на оси связанной системы координат. Подставляя данное решение в кинематические уравнения в той или иной форме, мы можем решить задачу Дарбу. Решив задачу Дарбу, мы получим полную картину движения твердого тела вокруг неподвижной точки.

На рисунках 1–3 представлены графики зависимостей компонент угловой скорости твердого тела от времени на оси связанной системы координат (14), для следующих начальных условий:

$$A = B = 1,\ C = 2;$$
$$p_0 = 1, q_0 = 0, r_0 = 5; k = 6$$
$$R_1 = R_2 = 0,\ R_3 = 1;$$
$$t = 0..5.$$

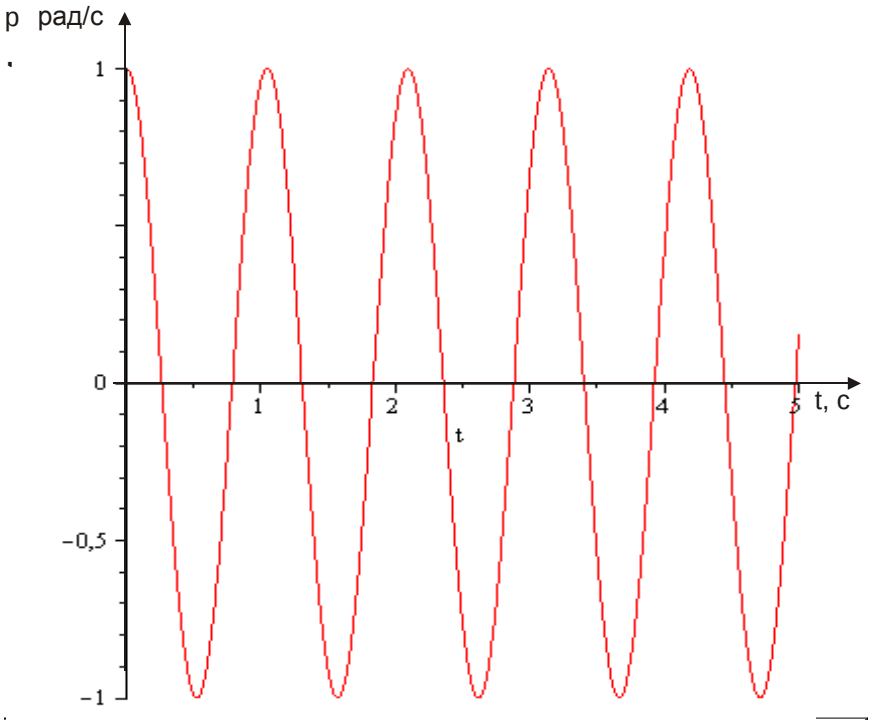


Рис. 1. Зависимость *p(t).*

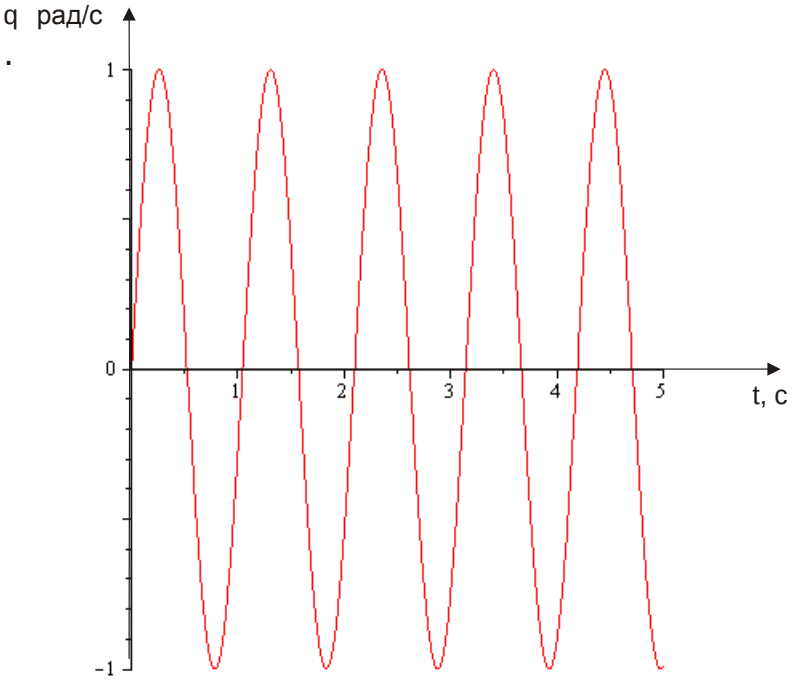


Рис. 2. Зависимость *q(t).*

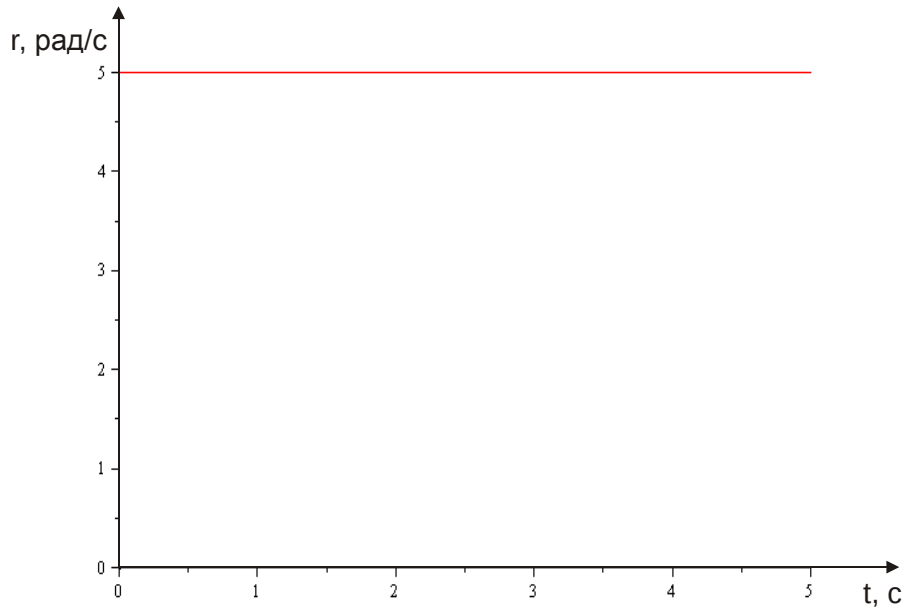


Рис. 3. Зависимость *r(t).*

4.2. Решение кинематических уравнений в параметрах Кэли-Клейна.

Связь между параметрами Кэли-Клейна и углами Эйлера имеет следующий вид [10, 13]:

$$\begin{cases} \theta = \arccos(\alpha\delta + \beta\gamma); \\ \varphi = \dfrac{1}{2}\arccos\left(\operatorname{Re}\left(\dfrac{\alpha\gamma}{\beta\delta}\right)\right); \\ \psi = \dfrac{1}{2}\arccos\left(\operatorname{Re}\left(\dfrac{\alpha\beta}{\delta\gamma}\right)\right). \end{cases} \tag{15}$$

Параметры Кэли-Клейна находятся из следующей системы кинематических дифференциальных уравнений:

$$\begin{cases} \alpha = \dfrac{r}{2}\alpha i + \dfrac{i}{2}(p - qi)\beta; \\ \beta = -\dfrac{r}{2}\beta i + \dfrac{i}{2}(p + qi)\alpha; \\ \gamma = \dfrac{r}{2}\gamma i + \dfrac{i}{2}(p - qi)\delta; \\ \delta = -\dfrac{r}{2}\delta i + \dfrac{i}{2}(p + qi)\gamma. \end{cases} \tag{16}$$

Зная начальную ориентацию разгонного блока, т.е. соответствующие углы $\theta_0,\ \varphi_0,\ \psi_0$, мы можем найти начальные условия для системы (31) из следующей системы:

$$\begin{cases} \alpha_0 = \cos\left(\frac{\theta_0}{2}\right) e^{i\left(\frac{\psi_0+\varphi_0}{2}\right)}; \\ \beta_0 = i\sin\left(\frac{\theta_0}{2}\right) e^{i\left(\frac{\psi_0-\varphi_0}{2}\right)}; \\ \delta_0 = \cos\left(\frac{\theta_0}{2}\right) e^{-i\left(\frac{\psi_0+\varphi_0}{2}\right)}; \\ \gamma_0 = i\sin\left(\frac{\theta_0}{2}\right) e^{-i\left(\frac{\psi_0-\varphi_0}{2}\right)}. \end{cases} \tag{17}$$

Для проверки можно использовать уравнение связи:

$$\det\begin{vmatrix} \alpha_0 & \beta_0 \\ \gamma_0 & \delta_0 \end{vmatrix} = 1 \tag{18}$$

Подробно рассмотрев систему (16) легко увидеть, что первые два уравнения не зависят от двух оставшихся. Решение первых двух уравнений системы (16) также будет являться решением оставшихся двух уравнений, только в комплексно-сопряженном виде.

Выберем следующие начальные условия, удовлетворяющие соотношению (18):

$$\begin{aligned} \alpha(0) &= \alpha_0; \\ \beta(0) &= \beta_0; \\ \gamma(0) &= \gamma_0; \\ \delta(0) &= \delta_0. \end{aligned} \tag{19}$$

Введем замену:

$$\mu = \sqrt{-k^2 - r_0{}^2 - 2kr_0 - q_0{}^2 - p_0{}^2} \tag{20}$$

Далее, подставляя в систему дифференциальных уравнений первого порядка (16) аналитические выражения для угловых скоростей (14) и решая задачу Коши для этой системы с начальными условиями (19), получим следующие аналитические выражения для параметров Кэли-Клейна:

$$\begin{aligned} \alpha(t) = &\frac{\alpha_0 kI + \beta_0 q_0 + Ir_0\alpha_0 + \alpha_0\mu + I\beta_0 p_0}{2\mu} e^{-0.5(Ik-\mu)t} - \\ &\frac{I(\alpha_0 k - I\beta_0 q_0 + r_0\alpha_0 + I\alpha_0\mu + \beta_0 p_0)}{2\mu} e^{-0.5(Ik+\mu)t} \end{aligned} \tag{21}$$

$$\begin{aligned} \beta(t) = \frac{1}{2}(&Ie^{-\frac{1}{2}(Ik+\mu)t}\mu\beta_0 q_0 - Ie^{-\frac{1}{2}(Ik-\mu)t}\alpha_0 q_0^2 + Ir_0 e^{-\frac{1}{2}(Ik-\mu)t}\beta_0 p_0 + Ie^{-\frac{1}{2}(Ik-\mu)t}k\beta_0 p_0 + Ie^{-\frac{1}{2}(Ik+\mu)t}\mu\alpha_0 q_0^2 - \\ &Ie^{-\frac{1}{2}(Ik-\mu)t}\mu\alpha_0 p_0^2 + e^{-\frac{1}{2}(Ik-\mu)t}k\beta_0 q_0 - Ie^{-\frac{1}{2}(Ik+\mu)t}\mu\beta_0 p_0 + r_0 e^{-\frac{1}{2}(Ik-\mu)t}\beta_0 q_0 - e^{-\frac{1}{2}(Ik-\mu)t}\mu\beta_0 p_0 - \\ &e^{-\frac{1}{2}(Ik+\mu)t}\mu\beta_0 q_0 + Ie^{-\frac{1}{2}(Ik-\mu)t}\mu\beta_0 q_0 - Ir_0 e^{-\frac{1}{2}(Ik+\mu)t}\beta_0 p_0 - \\ &Ie^{-\frac{1}{2}(Ik+\mu)t}k\beta_0 p_0)\frac{1}{\mu(I\cos(kt)q_0 - p_0\cos(kt) + q_0\sin(kt) + I\sin(kt)p_0)} \end{aligned}$$

Как было отмечено выше, аналитическое решение для δ(t), γ(t) являются комплексным сопряжением α(t), β(t) и, наоборот, с точностью до знака:

$$\begin{aligned} \gamma(t) &= -\overline{\beta}(t); \\ \delta(t) &= \overline{\alpha}(t). \end{aligned} \tag{22}$$

Так как параметры Кэли-Клейна являются комплексными комбинациями параметров Родрига-Гамильтона, то с помощью зависимости между этими параметрами:

$$\begin{cases} \lambda_0 = \frac{1}{2}(\alpha + \delta);\ \lambda_2 = -\frac{i}{2}(\beta + \gamma); \\ \lambda_1 = \frac{1}{2}(\beta - \gamma);\ \lambda_3 = -\frac{i}{2}(\alpha - \delta), \end{cases} \tag{23}$$

мы с легкостью можем получить аналитическое решение кинематических уравнений для параметров Родрига-Гамильтона.

Заключение.

В настоящей работе проведено исследование движения твердого тела с полостью, заполненной жидкостью. Решен ряд самостоятельных задач с использованием классических методов механики, алгебры кватернионов, решения дифференциальных уравнений, а также проведено численное моделирование движения.

Основные результаты работы заключаются в следующем:

- Проведен обзор используемых разгонных блоков, а так же опубликованных к настоящему времени трудов, посвященных движению твердого тела, содержащего полости с жидкостью;
- На основании классических методов теоретической механики построена математическая модель движения РБ с жидким топливом;
- Проведено численное интегрирование динамических и кинематических уравнений Эйлера, по результатам которого построены графики;
- Получено аналитическое решение динамических уравнений движения в проекциях угловой скорости тела на оси связанной системы координат;
- Получено аналитическое решение кинематических уравнений в параметрах Кэли-Клейна;
- Получены аналитические зависимости углов Эйлера от времени, а так же исследованы 2 предельных случая.

Полученные результаты позволяют производить исследование движения РБ, имеющих на борту запас жидкого топлива, осуществлять выбор начальных условий движения, инерционно-массовых, кинематических и других параметров, обеспечивающих реализацию тех или иных режимов движения, а также, могут быть использованы при проектировании новых РБ и совершенствовании старых.

Результаты настоящей научно-исследовательской работы получили признание как на городских, областных, так и международных конференциях, о чем свидетельствуют дипломы, грамоты, а так же благодарность за предоставленный доклад. По результатам работы было опубликовано две статьи в журналах, имеющих аккредитацию ВАК. Так же данная работа была оценена губернатором Самарской области и отмечена сертификатом о назначении стипендии губернатора Самарской области, кроме того, данная конкурсная работа выиграла в конкурсе «Молодой ученый» и конкурсе Программы поддержки технического образования на получение именной стипендии для студентов и преподавателей, финансируемой Фондом Alcoa.

Литература

1. Жуковский Н.Е. О движении твердого тела, имеющего полости, наполненные однородной капельной жидкостью [Текст] // Собрание сочинений. Т. 2. Гидродинамика. М.: Гостехиздат, 1949.

2. Алексеев А.В., Дорошин А.В. Приведение спутника-гиростата с полостью с жидкостью к системам твердых тел с вязким трением [Текст] // Общероссийский научно-технический журнал «Полёт», 2007, № 9, С. 26-33.

3. Лурье А.И. Аналитическая механика [Текст] // М.:ГИФМЛ, 1961.

4. Голдстейн Г., Классическая механика, пер. с англ., 2 изд. [Текст] //М.: Гостехиздат, 1975.
5. Черноусько Ф.Л. Движение твердого тела с полостями, содержащими вязкую жидкость [Текст] // М.: ВЦ АН СССР, 1968.
6. Моисеев Н.Н., Румянцев В.В. Динамика тела с полостями, содержащими жидкость [Текст] // М.: Наука, 1965.
7. Алексеев А.В., Красников В.С. Исследование движения разгонного блока с тороидальным топливным баком относительно центра масс в параметрах Кэли-Клейна [Текст] // «Вестник СГАУ», 2011, № 6, С. 9-14.
8. Алексеев А.В., Красников В.С. Угловое движение разгонного блока с жидким топливом [Текст] // «Вестник ИжГТУ», 2011, № 4, С. 42-45.

МЕТОД ВЕКТОРНОЙ ФУНКЦИИ ЛЯПУНОВА В ЗАДАЧЕ ОБ УПРАВЛЕНИИ СИСТЕМ С МГНОВЕННОЙ ОБРАТНОЙ СВЯЗЬЮ[1]

А.С.Андреев, А.О.Артемова

Ульяновский государственный университет

Рассмотрим управляемую систему, движение которой описывается векторным уравнением

$$\dot{\mathbf{x}} = \mathbf{X}(t, \mathbf{x}, \mathbf{u}) \tag{1}$$

где $x \in R^n$ – n-мерный вектор фазовых координат, $u \in R^m$ – m-мерный вектор управления, функция f ограничена и непрерывна в области $R^+ \times D \times R^m$, $D \subset R^n$ – некоторая область.

Пусть $f = \{\mathbf{x} : [t_0, +\infty) \to D\}$ $(t_0 \geq 0)$, есть класс программных движений (1) (представляющих собой абсолютные непрерывные функции, ограниченные некоторым компактом $K \subset D$), обеспечивающихся управлениями $\mathbf{u} \in U$, где U есть некоторое множество кусочно-непрерывных управлений.

В такой постановке для каждого $\mathbf{u} \in U$ найдем значения $\mathbf{X} = \mathbf{X}(t, \mathbf{x}, \mathbf{u}(t, \mathbf{x}))$ в области непрерывности $\mathbf{u} = \mathbf{u}(t, \mathbf{x})$ и доопределим во всей области $R^+ \times D$ уравнение (1) до включения

$$\dot{\mathbf{x}} \in \mathbf{F}(t, \mathbf{x}) \tag{2}$$

в соответствии с одним из доопределений [1].

Рассмотрим задачу об управлении системой (1) состоящую в нахождении для каждого $\mathbf{x}^0(t) \in f$ управления $\mathbf{u} \in U$ в виде зависимости $\mathbf{u} = \mathbf{u}^0(t, \mathbf{x})$, при котором движение $\mathbf{x}^0(t)$ являлось бы асимптотически устойчивым в силу уравнения (2).

Поставленную задачу можно отнести к задаче о стабилизации, или к задаче синтеза управления на бесконечном интервале времени [2, 3].

Из постановки следует, что программное движение $\mathbf{x}^0(t)$ для почти всех $t \geq t_0$ должно удовлетворять соотношению

$$\frac{d\mathbf{x}^0(t)}{dt} \in \mathbf{F}(t, \mathbf{x}^0(t), \mathbf{u}^0(t, \mathbf{x}^0(t))). \tag{3}$$

Будем полагать, что движение $\mathbf{x} = \mathbf{x}^0(t)$ при выбранном $\mathbf{u} = \mathbf{u}^0(t, \mathbf{x})$, $\mathbf{u}^0 \in U$, является единственным для включения (2) с правой частью $\mathbf{F}^0(t, \mathbf{x}) = \mathbf{F}(t, \mathbf{x}, \mathbf{u}^0(t, \mathbf{x}))$.

Примем, что по отношению к уравнению (2) выполнены условия существования решений соответствующего ему включения с $\mathbf{F} = \mathbf{F}^0(t, \mathbf{x})$, а также условия существования предельных включений

$$\frac{d\mathbf{x}}{dt} \in \mathbf{F}^*(t, \mathbf{x}) \tag{4}$$

Полагаем, что предельное к $\mathbf{x}^0(t)$ решение $\mathbf{x} = \mathbf{x}^*(t)$ также является единственным для (4).

Без ограничения общности можно считать, что $\mathbf{x}^0(t) = \mathbf{0}$ и $\mathbf{F}^*(t, 0) \equiv 0$.

Рассмотрим непрерывно дифференцируемую вектор-функцию $\mathbf{V}(t, \mathbf{x})$, $\mathbf{V}(t, \mathbf{x}) = (v^1(t, \mathbf{x}), \ldots, v^k(t, \mathbf{x}))^/$, $\mathbf{V} : R^+ \times D_H \to R^k$. Определим для неё верхнюю производную по времени в силу системы (4)

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0373 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

$$\dot{\mathbf{V}}^* = \left(\frac{d\mathbf{V}}{dt}\right)^* = \left(\frac{dv_1}{dt}^*, ..., \frac{dv_k}{dt}^*\right)^/,$$

где $\frac{dv_i}{dt}^* = \sup_{y \in F(t,x)} \left(\frac{\partial v_i}{\partial t} + \sum_{j=1}^{n} \frac{\partial v_i}{\partial x_j} y_j\right)$, $i = 1, 2, ..., k$.

Пусть K_1 – класс векторных функций $\mathbf{V} = (V^1, V^2, \ldots, V^k)^/$, $\mathbf{V}: \Gamma \to R^k$ ограниченных, равномерно непрерывных на каждом множестве $R \times K$, где $K \subset D$ – компакт. Пусть также K_2 и K_3 – аналогичные классы векторных функций $\mathbf{Y}: R \times R^k \to R^k$ и $\mathbf{W}: R \times D \times R^k \to R^k$ ограниченных и равномерно непрерывных по $(t, \mathbf{y}) \in R \times K_2$ и $(t, \mathbf{x}, \mathbf{y}) \in R \times K_1 \times K_2$ для любых компактных множеств $K_1 \subset D$ и $K_2 \subset R^k$.

Для каждой функции $\mathbf{V} \in \mathsf{K}_1$ семейство сдвигов

$$\{\mathbf{V}_\tau(t, \mathbf{x}) = \mathbf{V}(t + \tau, \mathbf{x}), \tau \in R^+\}$$

будет предкомпактно в некотором функциональном метризуемом пространстве F_V непрерывных функций $\mathbf{V}: \Gamma \to R^k$ с открыто-компактной топологией. Отсюда следует, что для любой последовательности $t_l \to +\infty$ найдутся подпоследовательность $t_{l_j} \to +\infty$ и функция $\mathbf{V}^* \in F_v$, такие, что последовательность сдвигов $\{\mathbf{V}_j(t, \mathbf{x}) = \mathbf{V}(t_{l_j} + t, \mathbf{x})\}$ будет сходиться к предельной функции $\mathbf{V}^*(t, \mathbf{x})$ в пространстве F_V, а именно: сходимость будет равномерной по $(t, \mathbf{x}) \in [-\beta, \beta] \times K$ для каждого числа $\beta > 0$ и каждого компактного множества $K \subset D$. Тем самым, для функции $\mathbf{V}$ можно определить семейство предельных функций $\{\mathbf{V}^*\}$.

Аналогично, для функций $Y \in \mathsf{K}_2$ и $W \in \mathsf{K}_3$ можно построить соответственно семейства $\{\mathbf{Y}^*\}$ и $\{\mathbf{W}^*\}$ предельных функций.

Пусть существует непрерывно дифференцируемая функция $\mathbf{V} \in \mathsf{K}_1$, верхняя производная которой в силу включения (4) представима в виде

$$\begin{gathered} \dot{\mathbf{V}}^*(t, \mathbf{x}) = \mathbf{Y}(t, \mathbf{V}(t, \mathbf{x})) + \mathbf{W}(t, \mathbf{x}, \mathbf{V}(t, \mathbf{x})), \\ \mathbf{Y}(t, \mathbf{0}) \equiv \mathbf{0}, \quad \mathbf{W}(t, \mathbf{0}, \mathbf{V}(t, \mathbf{0})) \equiv \mathbf{0}, \end{gathered} \tag{5}$$

где функция $\mathbf{Y} = \mathbf{Y}(t, \mathbf{y})$ принадлежит классу K_2, $\mathbf{Y} \in \mathsf{K}_2$, и является квазимонотонной и непрерывно дифференцируемой по $\mathbf{y} \in R^k$, $\partial\mathbf{Y}/\partial\mathbf{y} \in \mathsf{K}_2$, функция $\mathbf{W} = \mathbf{W}(t, \mathbf{x}, \mathbf{y})$ принадлежит классу K_3, $\mathbf{W} \in \mathsf{K}_3$, и имеет место неравенство $\mathbf{W}(t, \mathbf{x}, \mathbf{y}) \leq \mathbf{0}$ для любых $(t, \mathbf{x}, \mathbf{y}) \in R \times D \times R^k$.

Из представления (5) следует, что функция $\mathbf{V}(t, \mathbf{x})$ является вектор-функцией сравнения, а система

$$\dot{\mathbf{y}} = \mathbf{Y}(t, \mathbf{y}) \tag{6}$$

– системой сравнения.

Если $\mathbf{V} = \mathbf{V}(t, \mathbf{x})$ есть функция, удовлетворяющая уравнению (5), при этом $\mathbf{V}(t_0, \mathbf{x}_0) = \mathbf{V}_0$, а $\mathbf{y} = \mathbf{y}(t, t_0, \mathbf{V}_0)$ есть решение (6), определённое на интервале $[t_0, t_0 + \beta)$, $\beta > 0$, то для всех $t \in [t_0, t_0 + \beta)$ на решении $\mathbf{x} = \mathbf{x}(t, t_0, \mathbf{x}_0)$ системы (4) выполняется неравенство

$$\mathbf{V}(t, \mathbf{x}(t, t_0, x_0)) \leq \mathbf{y}(t, t_0, \mathbf{V}_0).$$

Из условия $\mathbf{Y} \in \mathsf{K}_2$ следует, что система (6) предкомпактна и для неё можно определить семейство предельных систем сравнения

$$\dot{\mathbf{y}} = \mathbf{Y}^*(t,\mathbf{y}), \qquad \mathbf{Y}^* \in F_Y. \tag{7}$$

Из условий относительно правой части $\mathbf{Y} = \mathbf{Y}(t,\mathbf{y})$ системы (7) следует, что решения этой системы $\mathbf{y} = \mathbf{y}(t,t_0,\mathbf{y}_0)$ непрерывно дифференцируемы по $(t_0,\mathbf{y}_0) \in R^+ \times R^k$. Из свойства неубывания зависимости $\mathbf{y}(t,t_0,\mathbf{y}_0)$ по $\mathbf{y}_0$ следует, что матрица

$$\mathbf{\Phi}(t,t_0,\mathbf{y}_0) = \frac{\partial \mathbf{y}(t,t_0,\mathbf{y}_0)}{\partial \mathbf{y}_0}$$

является неотрицательной, нормированной, $\mathbf{\Phi}(t_0,t_0,\mathbf{y}_0) = \mathbf{I}$ ($\mathbf{I} \in R^{n\times n}$ – единичная матрица) фундаментальной матрицей для линейной системы в вариациях

$$\dot{\mathbf{z}} = \mathbf{H}(t,t_0,\mathbf{y}_0)\mathbf{z}, \quad \mathbf{H} = \left.\frac{\partial \mathbf{Y}(t,\mathbf{y})}{\partial \mathbf{y}}\right|_{\mathbf{y}=\mathbf{y}(t,t_0,\mathbf{y}_0)}.$$

Предположим, что для любого компакта $K \in R^k$ существуют числа $M(K)$ и $\alpha(K) > 0$, такие, что матрица $\mathbf{\Phi}$ для любых $(t,t_0,\mathbf{y}_0) \in R^+ \times R^+ \times K$ удовлетворяет условиям

$$\mathrm{P}\mathbf{\Phi}(t,t_0,\mathbf{y}_0)\mathrm{P}\ldots M(K), \quad \det \mathbf{\Phi}(t,t_0,\mathbf{y}_0) \ldots \alpha(K).$$

Теорема 1. *Предположим, что существует вектор-функция Ляпунова* $\mathbf{V} = \mathbf{V}(t,\mathbf{x})$, $\mathbf{V} \in \mathrm{K}_1$, *такая, что: соответствующая скалярная функция* $\overline{V}(t,\mathbf{x})$, *определяемая в виде*

$$\overline{V}(t,\mathbf{x}) = \sum_{i=1}^{k} V^i(t,\mathbf{x}) \wedge \overline{V}(t,\mathbf{x}) = \max_{i=1,\ldots,k} V^i(t,\mathbf{x}),$$

является определённо-положительной; выполняются условия:

1) существует вектор-функция Ляпунова $\mathbf{V} = \mathbf{V}(t,\mathbf{x})$, $\mathbf{V} \in \mathrm{K}_1$, *удовлетворяющая дифференциальному равенству*

$$\dot{\mathbf{V}}^*(t,\mathbf{x}) = \mathbf{Y}(t,\mathbf{V}(t,\mathbf{x})) + \mathbf{W}(t,\mathbf{x},\mathbf{V}(t,\mathbf{x})),$$
$$\mathbf{Y}(t,\mathbf{0}) \equiv \mathbf{0}, \quad \mathbf{W}(t,\mathbf{0},\mathbf{V}(t,\mathbf{0})) \equiv \mathbf{0};$$

2) решения системы сравнения $\dot{\mathbf{y}} = \mathbf{Y}(t,\mathbf{y})$ *удовлетворяют условию* $\mathrm{P}\mathbf{\Phi}(t,t_0,\mathbf{y}_0)\mathrm{P}\ldots M(K), \quad \det \mathbf{\Phi}(t,t_0,\mathbf{y}_0) \ldots \alpha(K)$;

3) нулевое решение $\mathbf{y} = \mathbf{0}$ *системы сравнения (6) равномерно устойчиво;*

4) для любой предельной совокупности $(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)$ *и каждого ограниченного решения* $\mathbf{y} = \mathbf{y}^*(t) \neq \mathbf{0}$ *предельной системы сравнения (7) множество* $\{\mathbf{V}^*(t,\mathbf{x}) = \mathbf{y}^*(t)\} \bigcap \{\mathbf{W}^*(t,\mathbf{x},\mathbf{y}^*(t)) = 0\}$ *не содержит решений предельного включения (4).*

Тогда управление $\mathbf{u} = \mathbf{u}^0(t,\mathbf{x})$ *является стабилизирующим для движения* $\mathbf{x} = \mathbf{x}^0(t)$.

Для обоснования применения знакопостоянных вектор-функций Ляпунова в поставленной задаче стабилизации введём следующие определения.

Определение 1. *Нулевое решение* $\mathbf{x} = \mathbf{0}$ *устойчиво относительно множества* $\{\overline{V}^*(t,\mathbf{x}) = 0\}$ *и выбранной предельной совокупности* $(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)$, *если для любого числа* $\varepsilon > 0$ *найдётся* $\delta = \delta(\varepsilon) > 0$, *такое, что для любого решения* $\mathbf{x} = \mathbf{x}^*(t,\mathbf{x}_0)$ *системы (4), такого, что*

$$\mathbf{x}^*(0,\mathbf{x}_0) = \mathbf{x}_0, \quad \mathbf{x}_0 \in \{|\,\mathbf{x}\,| < \delta\} \bigcap \{\overline{V}^*(0,\mathbf{x}) = 0\} \bigcap \{\mathbf{W}^*(0,\mathbf{x},\mathbf{0}) = \mathbf{0}\},$$

для всех $t \ldots 0$ *выполняется неравенство* $|\,\mathbf{x}^*(t,\mathbf{x}_0)\,| < \varepsilon$.

Нулевое решение $\mathbf{x} = \mathbf{0}$ *асимптотически устойчиво относительно множества* $\{\overline{V}^*(t,\mathbf{x}) = 0\}$ *и выбранной предельной совокупности* $(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)$, *если оно устойчиво, а также существует число* $\Delta > 0$ *и для любого* $\varepsilon > 0$ *найдётся* $T = T(\varepsilon) > 0$, *такие, что для любого решения* $\mathbf{x} = \mathbf{x}^*(t,\mathbf{x}_0)$ *системы (4), такого, что*

$$\mathbf{x}^*(0,\mathbf{x}_0)=\mathbf{x}_0,\quad \mathbf{x}_0\in\{|\mathbf{x}|<\Delta\}\bigcap\{\overline{V}^*(0,\mathbf{x})=0\}\bigcap\{\mathbf{W}^*(0,\mathbf{x},\mathbf{0})=\mathbf{0}\},$$

для всех $t\ldots T$ *выполняется неравенство* $|\mathbf{x}^*(t,\mathbf{x}_0)|<\varepsilon$.

Определение 2. *Нулевое решение* $\mathbf{x}=\mathbf{0}$ *равномерно устойчиво относительно множества* $\{\overline{V}^*(t,\mathbf{x})=0\}$ *и семейства предельных совокупностей* $\{(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)\}$, *если число* $\delta=\delta(\varepsilon)>0$ *в определении 1 не зависит от выбора* $(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)$.

Нулевое решение $\mathbf{x}=\mathbf{0}$ *равномерно асимптотически устойчиво относительно множества* $\{\overline{V}^*(t,\mathbf{x})=0\}$ *и семейства предельных совокупностей* $\{(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)\}$, *если числа* $\Delta>0$ *и* $T=T(\varepsilon)>0$ *в определении 1 не зависят от выбора* $(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)$.

Теорема 2. *Предположим, что существует вектор-функция Ляпунова* $\mathbf{V}=\mathbf{V}(t,\mathbf{x})\ldots\mathbf{0}$, $\mathbf{V}\in\mathsf{K}_1$, *такая, что выполнены условия 1—3 теоремы 1, а также условия:*

4) нулевое решение $\mathbf{x}=\mathbf{0}$ *равномерно асимптотически устойчиво относительно множества* $\{\overline{V}^*(t,\mathbf{x})=0\}$ *и семейства предельных совокупностей* $\{(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)\}$.

5) множество $\{\mathbf{V}^*(t,\mathbf{x})=\mathbf{y}^*(t)\}\bigcap\{\mathbf{W}^*(t,\mathbf{x},\mathbf{y}^*(t))=\mathbf{0}\}$ *не содержит решений предельного включения (4). (Здесь* $\mathbf{y}^*(t)\neq\mathbf{0}$ *— произвольное ограниченное решение предельной системы сравнения (7)).*

Тогда управление $\mathbf{u}=\mathbf{u}^0(t,\mathbf{x})$ *является стабилизирующим для движения* $\mathbf{x}=\mathbf{x}^0(t)$.

В решении конкретных задач представляются важными следующие достаточные условия обеспечения заданного программного движения $x=x^0(t)$ системы (1).

Пусть управление $u(t,x)=(u_1(t,x),u_2(t,x),\ldots,u_m(t,x))'$ является кусочно-непрерывным, при этом функции $u_j(t,x)$ $(j=\overline{1,m})$ терпят разрыв на поверхности $\{\psi_j(t,x)=0\}$, где $\psi_j: R^+\times D\to R$ $(j=\overline{1,m})$ есть функции ограниченные и равномерно непрерывные по $(t,x)\in R^+\times K$, $K\subset D$.

Теорема 3. *Допустим, что можно найти вектор-функцию Ляпунова* $\mathbf{V}=\mathbf{V}(t,\mathbf{x})\ldots\mathbf{0}$, $\mathbf{V}\in\mathsf{K}_1$, *такую, что:* $\overline{V}(t,\mathbf{x})\ldots a_1(|\psi(t,\mathbf{x})|)$, *выполнены условия 1—3 теоремы 2, а также условия:*

4) нулевое решение $\mathbf{x}=\mathbf{0}$ *равномерно асимптотически устойчиво относительно множества* $\{\psi^*(t,\mathbf{x})=0\}$ *и семейства предельных совокупностей* $\{(\mathbf{F}^*,\mathbf{V}^*,\mathbf{Y}^*,\mathbf{W}^*)\}$.

5) множество $\{|\mathbf{V}^*(t,\mathbf{x})|=|\psi^*(t,\mathbf{x})|\}\bigcap\{\mathbf{W}^*(t,\mathbf{x},\mathbf{y}^*(t))=\mathbf{0}\}$ *не содержит решений предельного включения (4). (Здесь* $\mathbf{y}^*(t)\neq\mathbf{0}$ *— произвольное ограниченное решение предельной системы сравнения (7)).*

Тогда управление $\mathbf{u}=\mathbf{u}^0(t,\mathbf{x})$ *является стабилизирующим для движения* $\mathbf{x}=\mathbf{x}^0(t)$.

Следствием изложенных результатов являются следующие теоремы об управлении системой (1).

Теорема 4. *Допустим, что для системы (1) можно найти управляющее воздействие* $\mathbf{u}=\mathbf{u}^0(t,\mathbf{x})$ *и вектор функцию Ляпунова* $\mathbf{V}=\mathbf{V}(t,\mathbf{x})$, *такие, что:*

1) $a_1(|\psi(t,\mathbf{x})|)\le\overline{\mathbf{V}}(t,\mathbf{x}),\quad V_i(t,\mathbf{x})\le a_2(|\psi(t,\mathbf{x})|), i=1,2,...,k$;

2) производная вектор-функции Ляпунова в силу (4) удовлетворяет неравенству

$$\dot{\overline{\mathbf{V}}}^*(t,\mathbf{x})\le -a_3(|\psi(t,\mathbf{x})|);$$

3) невозмущенное движение $\mathbf{x}=0$ *системы (4) равномерно асимптотически устойчиво относительно множества* $\{\psi(t,\mathbf{x})=0\}$.

Тогда управляющее воздействие $\mathbf{u}=\mathbf{u}^0(t,\mathbf{x})$ *решает поставленную задачу о стабилизации.*

Теорема 5. *Если в условиях теоремы 4 решения системы сравнения (6) попадают в точку* $\mathbf{y} = \mathbf{0}$ *за конечный промежуток времени, тогда при* $\mathbf{u} = \mathbf{u}^0(t, \mathbf{x})$ *движения (4) достигают множества* $\{\overline{\mathbf{V}}(t, \mathbf{x}) = 0\}$ *за конечный промежуток времени.*

Если в дополнение движения с множества $\{\overline{\mathbf{V}}(t, \mathbf{x}) = 0\}$ *достигают также положения* $\mathbf{x} = 0$ *за конечный промежуток времени, тогда управляющее воздействие* $\mathbf{u} = \mathbf{u}^0(t, \mathbf{x})$ *решает задачу конечного синтеза.*

Литература

1. Филиппов А.Ф. Дифференциальные уравнения с разрывной правой частью. М.: Физматгиз. 1985. 224 с.

2. Красовский Н.Н. Проблемы стабилизации управляемых движений / Н.Н. Красовский // Малкин И.Г. Теория устойчивости движения. Доп.4 – М.: Наука, 1966. – С. 475—514.

3. Летов А. М. Аналитическое конструирование регуляторов / А.М. Летов // Автоматика и телемеханика. – 1960, № 4. – С. 436—441; – № 5. – С. 561—568; – № 6. – С. 661—665. – 1961, № 4. – С. 425—435; – 1962, № 11. – С. 1405—1413.

О НЕЛИНЕЙНОЙ ЗАДАЧЕ СО СМЕЩЕНИЕМ ПО ТРАЕКТОРИЯМ[1]

А.Н.Андронов

Мордовский гуманитарный институт

Рассматривается задача применения методов Ляпунова к качественному анализу систем, описываемых нелинейными уравнениями в частных производных. Проведено исследование задачи со смещениям в производных для нелинейно возмущенного оператора Лапласа на s-мерной сфере ($s \geq 2$). Вычислены собственные значения и построены собственные и присоединенные функции для случая $s = 2$ и $s > 2$.

Рассмотрим случай $s = 2$ в цилиндрических координатах

$$(\Delta + \lambda)\Phi^{(1)} = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial \Phi^{(1)}}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 \Phi^{(1)}}{\partial \varphi^2} + \lambda\Phi^{(1)} = 0,$$
$$\Phi^{(1)} \in C^{2+\alpha}(\Omega), \quad \Omega = r \mid r < 1 \tag{1}$$
$$\frac{\partial \Phi^{(1)}}{\partial r}\Big|_{r=r_0} = \frac{\partial \Phi^{(1)}}{\partial r}\Big|_{r=r_1}$$

Разделяя переменные $\Phi^{(1)}(r,\varphi) = X^{(1)}(r)Y(\varphi)$ и используя условия периодичности $\Phi^{(1)}(r,\varphi + 2\pi) = \Phi^{(1)}(r,\varphi)$ и ограниченности в нуле, находим

$$Y(\varphi) = C_1 \cos n\varphi + C_2 \sin n\varphi,$$
$$X^{(1)}{}'' + \frac{1}{r}X^{(1)}{}' + \left(\lambda - \frac{n^2}{r^2}\right)X^{(1)} = 0, \text{ (уравнение Бесселя)}$$
$$\left\|X^{(1)}(0)\right\| < \infty, \; X_r^{(1)}{}'(r_0) = X_r^{(1)}(1).$$

Собственные функции, определяемые с помощью $X^{(1)}(r) = J_n(\alpha r)$ ($J_n(\alpha r)$ — функции Бесселя) отвечают собственным значениям, являющимся корнями уравнения

$$f(\alpha) \equiv -J_n'(\alpha r_0) + J_n'(\alpha) = 0. \tag{2}$$

Сопряженная к (1) по Лагранжу задача

$$(\Delta + \lambda)\Psi^{(1)} = 0, \; \Psi^{(1)} \in C^{2+\alpha}(\Omega_1) \cup C^{2+\alpha}(\Omega_2),$$
$$\Phi_r^{(1)}{}'(r_0 - 0, \varphi) = \Phi_r^{(1)}{}'(r_0 + 0, \varphi), \; \Phi_r^{(1)}{}'(1,\varphi) = 0, \tag{3}$$
$$\Phi^{(1)}(1,\varphi) + r_0[\Phi^{(1)}(r_0 - 0,\varphi) - \Phi^{(1)}(r_0 + 0,\varphi)].$$

Так как $\Phi^{(1)}$ ограничена, решение (3) $\Phi^{(1)}(r,\varphi) = \chi^{(1)}(r)[C_1 \cos n\varphi + C_2 \sin n\varphi]$ имеет вид

$$\chi^{(1)}(r) = \begin{cases} C_{11}J_n(\alpha r), 0 \leq r < r_0, \\ C_{21}J_n(\alpha r) + C_{22}N_n(\alpha r), r_0 \leq r \leq 1. \end{cases}$$

($N_n(\alpha r)$ - функции Неймана). Из условий (3) находим систему для определения C_{jk}

$$\begin{cases} C_{11}J_n'(\alpha r_0) - C_{21}J_n'(\alpha r_0) - C_{22}N_n'(\alpha r_0) = 0, \\ C_{21}J_n(\alpha r) + C_{22}N_n(\alpha r) = 0, \\ C_{11}r_0 J_n(\alpha r_0) + C_{21}[J_n(\alpha) - r_0 J_n(\alpha r_0)] + C_{22}[N_n(\alpha) - r_0 N_n(\alpha r_0)] = 0, \end{cases} \tag{4}$$

затем находим ее определитель

$$\Delta_0 = J_n'(\alpha r_0)[J_n'(\alpha)N_n(\alpha) - J_n(\alpha)N_n'(\alpha)] + r_0 J_n'(\alpha)[J_n(\alpha r_0)N_n'(\alpha r_0) - J_n'(\alpha r_0)N_n(\alpha r_0)]$$

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0230 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

Из известного соотношения $J_\nu(z)N_\nu'(z)-N_\nu(z)J_\nu'(z)=\frac{2}{\pi z}$ получаем то же уравнение (2) для определения собственных значений $\lambda=\alpha^2$: $\frac{2}{\pi\alpha}f(\alpha)=\frac{2}{\pi\alpha}[J_n'(\alpha)-J_n'(\alpha r_0)]=0$. В силу $J_n'(\alpha)J_n'(\alpha r_0)\neq 0$ и из первого уравнения системы (4)

$$C_{11}=\frac{1}{J_n'(\alpha)}[C_{21}J_n'(\alpha r_0)+C_{22}N_n'(\alpha r_0)].$$

Тогда

$$\chi^{(1)}(r)=C\begin{cases}[N_n'(\alpha r_0)-N_n'(\alpha)]J_n(\alpha r), 0\le r<r_0,\\ J_n'(\alpha)N_n(\alpha r)-N_n'(\alpha r)J_n(\alpha r), r_0\le r\le 1.\end{cases} \tag{5}$$

Теорема 1. Задача (1) имеет собственные значения $\lambda=\alpha^2(n)$, определяемые равенством (2) с собственными функциями $\Phi_m^{(1)}=J_n(\alpha r)[C_{n1}\cos n\varphi+C_{n2}\sin n\varphi]$. Ей отвечает сопряженная задача (3) с теми же собственными значениями, которым соответствуют собственные функции (5).

Условие отсутствия присоединенных элементов $\Phi^{(2)}$ имеет вид

$$I_n^1(\alpha)=\frac{1}{\pi r_0\alpha^3}[(n^2-\alpha^2)r_0J_n(\alpha)+(r_0^2\alpha^2-n^2)J_n(\alpha r_0)]=\frac{1}{\alpha\pi}f'(\alpha)\neq 0. \tag{6}$$

Действительно, $I_n(\alpha)=\int_0^1 X^{(1)}(r)\chi^{(1)}(r)rdr$ дает первое равенство (6). Так как

$$J_n''(\alpha)=-\frac{1}{\alpha}J_n'(\alpha)-\left(1-\frac{n^2}{\alpha^2}\right)J_n(\alpha),\ J_n''(\alpha r_0)=-\frac{1}{\alpha r_0}J_n'(\alpha r_0)-\left(1-\frac{n^2}{\alpha^2 r_0^2}\right)J_n(\alpha r_0),$$

то

$$f'(\alpha)=\frac{1}{r_0\alpha^2}[(n^2-\alpha^2)r_0J_n(\alpha)+(r_0^2\alpha^2-n^2)J_n(\alpha r_0)].$$

Соответственно неоднородная задача (уравнение Бесселя) с правой частью $J_n(\alpha r)$ имеет решение при $0<r<1$

$$X^{(2)}(r)=C_1J_n(\alpha r)+\frac{r}{2\alpha}J_n'(\alpha r)+\frac{N_n(\alpha r)}{2\alpha(N_n'(\alpha r_0)-N_n'(\alpha))}\cdot\left[r_0\left(1-\frac{n^2}{\alpha^2r_0^2}\right)J_n(\alpha r_0)-\left(1-\frac{n^2}{\alpha^2}\right)J_n(\alpha)\right]$$

.

При этом условие отсутствия присоединенных элементов $\Phi^{(3)}$ определяется интегралом $I_n^2(\alpha)=\int_0^1 X^{(2)}(r)\chi^{(1)}rdr\neq 0$ и может быть также выражено неравенством $f''(\alpha)\neq 0$.

Отдельно могут быть найдены условия отсутствия присоединенных элементов высших порядков.

Исследуем более общий случай, т. е. $s>2$. Имеем [1,2]

$$\begin{gathered}(\Delta+\lambda)\Phi^{(1)}=\frac{1}{r}\frac{\partial}{\partial r^{s-1}}\left(r^{s-1}\frac{\partial\Phi^{(1)}}{\partial r}\right)+\frac{1}{r^2}\Delta_\theta\Phi^{(1)}+\lambda\Phi^{(1)}=0,\\ \Phi^{(1)}\in C^{2+\alpha}(\Omega),\quad \Omega=r,\theta\,|\,r<1,\theta=(\theta_1,\ldots,\theta_{n-1})\\ \frac{\partial\Phi^{(1)}(r_0,\theta)}{\partial r}=\frac{\partial\Phi^{(1)}(1,\theta)}{\partial r}\end{gathered} \tag{7}$$

где Δ_θ — оператор Лапласа на единичной сфере в R^s. Разделяя переменные

$$\Phi^{(1)}=X^{(1)}(r)Y(\theta),$$

получаем уравнение для полисферических функций

$$\Delta_\theta Y_{s.n} - n(n+s-2)Y_{s,n} = 0,$$

и, после подстановки $X^{(1)}(r) = r^{-\frac{s}{2}+1}x(r)$ — уравнение Бесселя

$$x'' + \frac{1}{r}x' + \left[\lambda - \frac{\left(n+\frac{s}{2}-1\right)}{r^2}\right]x = 0 \qquad (8)$$

При этом смещение (7) дает в предположении ограниченности решения условие, определяющее собственные значения $\lambda = \alpha^2$ как корни уравнения

$$f(\alpha) \equiv \alpha\left[r_0^{-\frac{s}{2}+1} J'_{n+\frac{s}{2}-1}(\alpha r_0) - J'_{n+\frac{s}{2}-1}\right] + \left(1-\frac{s}{2}\right)\left[r_0^{-\frac{s}{2}} J_{n+\frac{s}{2}-1}(\alpha r_0) - J_{n+\frac{s}{2}-1}(\alpha)\right] \qquad (9)$$

Для определения сопряженной задачи рассматривается интеграл

$$\int_0^1\int_{S_r} \Delta u v r^{s-1} dr dS = \int_{S_1}\left(\int_0^{r_0-0} + \int_{r_0-1}^1\right)\left[\frac{1}{r^{s-1}}\frac{\partial}{\partial r}\left(r^{s-1}\frac{\partial u}{\partial r}\right) + \frac{1}{r^2}\Delta_\theta u\right] v(r,\theta) r^{s-1} dr dS_1 =$$

$$= \int_{S_1}\left(\int_0^{r_0-0} + \int_{r_0-0}^1\right)\left[\frac{\partial}{\partial r}\left(r^{s-1}\frac{\partial u}{\partial r}\right) + r^{s-3}\Delta_\theta u\right] v(r,\theta) dr dS_1 = \int_{S_1}\left[r^{s-1}\frac{\partial u}{\partial r}v(r,\theta)\right]\left(\Big|_0^{r_0-0} + \Big|_{r_0+0}^1\right) -$$

$$-\int_{S_1}\left(\int_0^{r_0-0} + \int_{r_0+0}^1\right)\left(r^{s-1}\frac{\partial u}{\partial r}\frac{\partial v}{\partial r}\right) dr dS_1 + \int_0^1 r^{s-3}\int_{S_1} u\Delta_\theta v ds dr =$$

$$= \int_{S_1}\left\{r_0^{s-1}\frac{\partial u(r_0-0,\theta)}{\partial r}v(r_0-0,\theta) + \frac{\partial u(1,\theta)}{\partial r}v(1,\theta) - r_0^{s-1}\frac{\partial u(r_0+0,\theta)}{\partial r}v(r_0+0,\theta) -\right.$$

$$\left. -u(r_0-0,\theta)r_0^{s-1}\frac{\partial v(r_0-0,\theta)}{\partial r} - u(1,\theta)\frac{\partial v(1,\theta)}{\partial r} + u(r_0+0,\theta)r_0^{s-1}\frac{\partial v(r_0+0,\theta)}{\partial r}\right\} dS_1 +$$

$$+\int_{S_1}\int_0^1 u\frac{\partial}{\partial r}\left(r^{s-1}\frac{\partial v}{\partial r}\right) dr dS_1 + \int_0^1 r^{s-3}\int_{S_1} u\Delta_\theta v ds dr =$$

$$= \int_{S_1}\int_0^1 u\left[\frac{1}{r^{s-1}}\frac{\partial}{\partial r}\left(r^{s-1}\frac{dv}{dr}\right) + \frac{1}{r^2}\Delta_\theta v\right] r^{s-1} dr dS_1 + \int_{S_1}\left[v(r_0-0,\theta)r_0^{s-1}\frac{\partial u(r_0-0,\theta)}{\partial r} -\right.$$

$$-v(r_0+0,\theta)r_0^{s-1}\frac{\partial u(r_0+0,\theta)}{\partial r} - u(r_0-0,\theta)r_0^{s-1}\frac{\partial v(r_0-0,\theta)}{\partial r} + u(r_0+0.\theta)r_0^{s-1}\frac{\partial v(r_0+0.\theta)}{\partial r} +$$

$$\left. +v(1,\theta)\frac{\partial u(1,\theta)}{\partial r} - u(1,\theta)\frac{\partial v(1,\theta)}{\partial r}\right] dS_1.$$

Т.к. $u(r_0-0,\theta) = u(r_0+0,\theta)$, то

$$\frac{\partial v(r_0-0,\theta)}{\partial r} = \frac{\partial v(r_0+0,\theta)}{\partial r}.$$

Поскольку $\frac{\partial u(r_0-0,\theta)}{\partial r} = \frac{\partial u(1,\theta)}{\partial r}$ (смещение в производной), то

$$r_0^{s-1}\left[v(r_0-0,\theta) - v(r_0+0,\theta)\right] + v(1,\theta) = 0,$$

$$\frac{\partial v(1,\theta)}{\partial r} = 0.$$

При этом использовано следующее равенство:

$$\int_{S_1}\left(\int_0^{r_0-0}+\int_{r_0+0}^{1}\right)u\left[\frac{\partial u}{\partial r}\left(r^{s-1}\frac{\partial v}{\partial r}\right)\right]drdS_1=\int_{S_1}\left[\left(ur^{s-1}\frac{\partial u}{\partial r}\right)\left(\Big|_0^{r_0-0}+\Big|_{r_0+0}^{1}\right)\right]dS_1-$$

$$-\int_{S_1}\left(\int_0^{r_0-0}+\int_{r_0+0}^{1}\right)\left(r^{s-1}\frac{\partial v}{\partial r}\frac{\partial u}{\partial r}\right)drdS_1$$

Сопряженная задача:

$$\begin{gathered}(\Delta+\lambda)\Psi^{(1)}=0,\quad \Omega_1=\{r\mid r<r_0\}\cup\Omega_2=\{r\mid r_0<r<1\},\\ \Psi^{(1)}{}'_r(r_0-0,\theta)=\Psi^{(1)}{}'_r(r_0+0,\theta),\Psi^{(1)}{}'_r(1,\theta)=0,\\ r_0^{s-1}\left[\Psi^{(1)}(r_0-0,\theta)-\Psi^{(1)}(r_0+0,\theta)\right]+\Psi^{(1)}(r_0-0,\theta)=0,\\ \Psi^{(1)}(r,\theta)=\chi_{s,n}^{(1)}(r)Y_{s,n}(\theta).\end{gathered}\tag{10}$$

Решение сопряженной задачи имеет вид:

$$\chi_{s,n}^{(1)}(r)=r^{-\frac{s}{2}+1}\begin{cases}C_{11}J_{n+\frac{s}{2}-1}(\alpha r),0\le r<r_0,\\ C_{21}J_{n+\frac{s}{2}-1}(\alpha r)+C_{22}N_{n+\frac{s}{2}-1}(\alpha r),r_0\le r\le 1.\end{cases}$$

Для определения постоянных в решении получаем систему

$$\begin{cases}C_{21}\left[\left(1-\frac{s}{2}\right)J_{n+\frac{s}{2}-1}(\alpha)+\alpha J'_{n+\frac{s}{2}-1}(\alpha)\right]+C_{22}\left[\left(1-\frac{s}{2}\right)N_{n+\frac{s}{2}-1}(\alpha)+\alpha N'_{n+\frac{s}{2}-1}(\alpha)\right]=0.\\ -C_{11}\left[\left(1-\frac{s}{2}\right)J_{n+\frac{s}{2}-1}(\alpha r_0)+\alpha J'_{n+\frac{s}{2}-1}(\alpha r_0)\right]+C_{21}\left[\left(1-\frac{s}{2}\right)J_{n+\frac{s}{2}-1}(\alpha r_0)+\alpha J'_{n+\frac{s}{2}-1}(\alpha r_0)\right]+\\ +C_{22}\left[\left(1-\frac{s}{2}\right)N_{n+\frac{s}{2}-1}(\alpha r_0)+\alpha N'_{n+\frac{s}{2}-1}(\alpha r_0)\right]=0,\\ C_{11}r_0^{s/2}J_{n+\frac{s}{2}-1}(\alpha r_0)+C_{21}\left[-r_0^{s/2}J_{n+\frac{s}{2}-1}(\alpha r_0)+J_{n+\frac{s}{2}-1}(\alpha)\right]+C_{22}\left[-r_0^{s/2}N_{n+\frac{s}{2}-1}(\alpha r_0)+N_{n+\frac{s}{2}-1}(\alpha)\right]=0.\end{cases}\tag{11}$$

с определителем $-\frac{2}{\pi}r_0^{s/2}f(\alpha)$.

Находя постоянные, определяем собственные функции сопряженной задачи.

$$\chi_{s,n}^{(1)}(r)=\begin{cases}\left\{\left[\left(1-\frac{s}{2}\right)N_{n+\frac{s}{2}-1}(\alpha r_0)+\alpha r_0N'_{n+\frac{s}{2}-1}(\alpha r_0)\right]r_0^{-s/2}-\left(1-\frac{s}{2}\right)J_{n+\frac{s}{2}-1}(\alpha r_0)+\alpha r_0J'_{n+\frac{s}{2}-1}(\alpha r_0)\right\}\times\\ \times J_{n+\frac{s}{2}-1}(\alpha r),\quad 0\le r<r_0,\\ -\left[\left(1-\frac{s}{2}\right)N_{n+\frac{s}{2}-1}(\alpha)+\alpha N'_{n+\frac{s}{2}-1}(\alpha)\right]J_{n+\frac{s}{2}-1}(\alpha r)+N_{n+\frac{s}{2}-1}(\alpha r),\quad r_0<r\le 1.\end{cases}$$

Теорема 2. Задача (7) имеет собственные значения $\lambda=\alpha^2(n)$, определяемые равенством (9) с собственными функциями $\Phi_n^{(1)}(r,\theta)=J_n(\alpha r)Y_{s,n}(\theta)$. Ей отвечает сопряженная задача (10) с теми же собственными значениями, которым соответствуют собственные функции $\Psi^{(1)}(r,\theta)=\chi_{s,n}^{(1)}(r)Y_{s,n}(\theta)$.

Условие отсутствия присоединенных элементов $\Phi^{(2)}$ мы можем получить, вычисляя интеграл $I_{s,n}(\alpha)=\int_0^1 X_{s,n}^{(1)}(r)\chi_{s,n}^{(1)}(r)r^{s-1}dr$. Как и в частном случае, оно связано с условием

$f'(\alpha)=0$. Условиями отсутствия присоединенных элементов последовательно более высоких порядков являются $f''(\alpha)\neq 0$, $f'''(\alpha)\neq 0$, причем на присоединенных элементах третьего порядка жордановы цепочки обрываются.

Литература

1. Логинов Б. В., Коноплева И. В. Бифуркация, симметрия и кососимметрия в ДУ, не разрешенных относительно производной с вариационными уравениями разветвления // Доклады РАН. Математика. – 2009. – Т. 427, № 4. –С. 452—457.

2. Логинов Б. В., Коноплева И. В., Русак Ю. Б. Симметрия и потенциальность УРК в неявно заданных стационарных и динамических бифуркационных задачах // Известия ВУЗов Северо-Кавказский регион. Спецвыпуск памяти В. И. Юдовича. Ростов-на-Дону Ун-т. – 2009. – С. 115—124.

УДК 531.36

О МОДЕЛИРОВАНИИ УПРАВЛЯЕМОЙ ЛАГРАНЖЕВОЙ СИСТЕМЫ[1]

А.О.Артемова

Ульяновский государственный университет

Рассматривается робототехническая система, моделируемая уравнениями Лагранжа второго рода. Определяются алгоритмы построения непрерывного и релейного управлений, решающих задачу о стабилизации спектра заданных программных движений системы.

Пусть положение робототехнической системы определяется n обобщенными координатами $q_1, q_2, ..., q_n$, $q^T = (q_1, q_2, ..., q_n)$ – соответствующий вектор (здесь и далее $(\cdot)^T$ – операция транспонирования) и соответственно кинетическая энергия системы представима в виде

$$T = T_2 + T_1 + T_0,$$
$$T_2(t,q,\dot{q}) = \tfrac{1}{2}\dot{q}^T A(t,q)\dot{q},$$
$$T_1(t,q,\dot{q}) = B^T(t,q)\dot{q}, \quad T_0(t,q) = C(t,q),$$

где $A(t,q) \in R^{n\times n}$ — матрица размерности $n \times n$ является в общем случае положительно определенной, $B(t,q) \in R^n$ — матрица-столбец или вектор, $C(t,q)$ — скалярная функция. Будем полагать для определенности, что матрицы $A(t,q), B(t,q)$ и функция $C(t,q)$ является непрерывно-дифференцируемыми, $A, B, C \in C^1$.

Составим уравнения управляемого движения механической системы в виде уравнений Лагранжа второго рода [1]

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}}\right) - \frac{\partial T}{\partial q} = Q + U, \tag{1}$$

где $U = (U_1, U_2, \ldots, U_n)^T$ – совокупность управляющих сил, подлежащих определению, $U \in F_u$, $Q = (Q_1, Q_2, \ldots, Q_n)^T$, $Q = Q(t,q,\dot{q})$ – совокупность обобщенных сил, включающих в себя все остальные внешние и внутренние силы, в том числе, неконтролируемые возмущения.

В рассматриваемой модели $U_1, U_2, \ldots, U_n$ есть непосредственно управляющие силы и моменты, они определяются в виде зависимостей $U = U(t,w)$, где $w = (w_1, \ldots, w_r)$ есть совокупность управляющих сигналов, подающихся на исполнительные механизмы.

Допустим, что для системы (1) задан некоторый спектр программных движений

$$L = \{(q(t), \dot{q}(t)), t_0 \le t < +\infty\}. \tag{2}$$

Исследуем задачу о построении управления, стабилизирующего каждое из движений (2) с тем, чтобы закон управления в максимальной степени не зависел от этого движения, а также от инерционных параметров системы, а при возможности величина управления была ограничена.

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0373 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

Положим, что $\left(q^0(t),\dot{q}^0(t)\right)\in L$ – есть какое-либо движение, описываемое уравнениями (1). Перепишем эти уравнения в виде

$$A\ddot{q}=\left\{\dot{q}^T S_1\dot{q}\right\}-\frac{\partial A}{\partial t}\dot{q}+G\dot{q}+\frac{\partial T_0}{\partial q}-\frac{\partial B}{\partial t}+Q+U. \tag{3}$$

В силу предположения должно иметь место тождество

$$\begin{gathered}A\left(t,q^0(t)\right)\ddot{q}^0(t)\equiv\left\{\left(\dot{q}^0(t)\right)^T S_1\left(t,q^0(t)\right)\dot{q}^0(t)\right\}-\\ -\frac{\partial A}{\partial t}\left(t,q^0(t)\right)\dot{q}^0(t)+G\left(t,q^0(t)\right)\dot{q}^0(t)+\frac{\partial T}{\partial q}\left(t,q^0(t)\right)- \\ -\frac{\partial B}{\partial t}\left(t,q^0(t)\right)+Q\left(t,q^0(t),\dot{q}^0(t)\right)+U^0(t),\end{gathered} \tag{4}$$

которым определяется требуемое управление $U^0(t)$, обеспечивающее движение $\left(q^0(t),\dot{q}^0(t)\right)$.

Введем возмущения $x=q-q_0(t)$ и управляющие воздействия $U\left(t,x,\dot{x}\right)=U-U^0(t)$. Уравнения возмущенного движения запишутся в виде:

$$\begin{gathered}A\left(t,q^0(t)+x\right)\ddot{x}+\left(A\left(t,q^0(t)+x\right)-A\left(t,q^0(t)\right)\right)\ddot{q}^0(t)=\\ =\left\{\dot{x}^T S_1\left(t,q^0(t)+x\right)\dot{x}\right\}+2\left\{\left(\dot{q}^0(t)\right)^T S_1\left(t,q^0(t)+x\right)\dot{x}\right\}+\\ +\left\{\left(\dot{q}^0(t)\right)^T S_1\left(t,q^0(t)+x\right)-S_1\left(t,q^0(t)\right)\dot{q}^0(t)\right\}+G\left(t,q^0(t)+x\right)\dot{x}+\left(G\left(t,q^0(t)+x\right)-\right.\\ \left.-G\left(t,q^0(t)\right)\right)\dot{q}^0(t)-\frac{\partial A}{\partial t}\left(t,q^0(t)+x\right)\dot{x}-\left(\frac{\partial A}{\partial t}\left(t,q^0(t)+x\right)-\frac{\partial A}{\partial t}\left(t,q^0(t)\right)\right)\dot{q}^0(t)+\\ +\frac{\partial T_0}{\partial q}\left(t,q^0(t)+x\right)-\frac{\partial T_0}{\partial q}\left(t,q^0(t)\right)-\frac{\partial B}{\partial t}\left(t,q^0(t)+x\right)-\frac{\partial B}{\partial t}\left(t,q^0(t)\right)+\\ +Q\left(t,q^0(t)+x,\dot{q}^0(t)+\dot{x}\right)-Q\left(t,q^0(t),\dot{q}^0(t)\right)+U\left(t,x,\dot{x}\right)+\mu(t),\end{gathered}$$

где $\mu(t)\equiv 0$ при выполнении условия (4) и $\mu(t)\neq 0$ в противном случае.

Выделив члены, линейные по $\dot{x}$, представим эти уравнения в следующей форме

$$\begin{gathered}A_1\left(t,x\right)\ddot{x}=\left\{\dot{x}^T S_1\left(t,x\right)\dot{x}\right\}+D_1\left(t,x\right)\dot{x}+G_1\left(t,x\right)\dot{x}+\\ +R\left(t,x\right)+Q_1^x\left(t,x,\dot{x}\right)+U\left(t,x,\dot{x}\right)+\mu(t),\end{gathered}$$

где матрицы D_1 и G_1 соответственно симметричная и кососимметричная:

$$D_1^T=D_1,\quad G_1^T=-G_1,$$

функции $R\left(t,x\right)$ и Q_1 определяются изменением составляющих параметров и внешних сил системы на возмущенном движении $\left(x(t),\dot{x}(t)\right)$:

$$\begin{gathered}R\left(t,0\right)=0,\\ Q_1^x\left(t,x,\dot{x}\right)=Q\left(t,q^0(t)+x,\dot{q}^0(t)+\dot{x}\right)-Q\left(t,q^0(t),\dot{q}^0(t)\right)=Q_x\left(t,x\right)+Q_1\left(t,x\right)\dot{x}+\\ +\left\{\dot{x}^T Q_2^x\left(t,x\right)\dot{x}\right\}+Q_3^x\left(t,x,\dot{x}\right).\end{gathered}$$

Выразим эти уравнения относительно $\ddot{x}$:

$$\ddot{x} = A_1^{-1}\left(\left\{\dot{x}^T S_1 \dot{x}\right\} + D_1\dot{x} + G_1\dot{x} + R_x f + Q_1\dot{x} + \left\{\dot{x}^T Q_2 \dot{x}\right\} + Q_3 + U + \mu\right). \qquad (5)$$

Пусть $f = f(t,x)$ – есть некоторая вектор-функция, определяемая структурой системы, $f(t,0) \equiv 0$. Построим управляющее воздействие U, обеспечивающее стабилизацию невозмущенного движения $x = \dot{x} = 0$ системы (5), в виде

$$U = \Psi\left(t, H(t)\dot{x} + f(t,x)\right), \qquad (6)$$

где $\Psi(t,\gamma)$ – есть некоторая вектор–функция, $\Psi(t,0) = 0, \Psi \in C^1$, H – матрица, $H \in R^{n\times n}$.

Рассмотрим наиболее простой вид для U

$$U = B\left(H\dot{x} + f(t,x)\right), \quad B, H \in R^{n\times n} = const. \qquad (7)$$

Следуя [2] введем функцию Ляпунова

$$V = \tfrac{1}{2}\left(H\dot{x} + f\right)^T M\left(H\dot{x} + f\right) + \Pi(t,x),$$

где положительно-определенная матрица $M \in R^{n\times n}$, $M = const$ и определенно-положительная, допускающая бесконечно малый высший предел функция $\Pi = \Pi(t,x)$ находятся из условия $\dot{V} \le 0$, в предположении $\mu(t) \equiv 0$ (или при существовании программного движения $\left(\dot{q}_0(t), q_0(t)\right)$).

Находим производную $\dot{V}$ в силу (5)

$$\dot{V} = \left(H\dot{x} + f\right)^T M\left(HA_3^{-1}\left(\left\{\dot{x}^T S_1 \dot{x}\right\} + D_1\dot{x} + G_1\dot{x} + \right.\right.$$

$$\left.\left. + R_x f + Q_1\dot{x} + \left\{\dot{x}^T Q_2 \dot{x}\right\} + Q_3 + Q_x f + B\left(H\dot{x} + f\right)\right) + F_t + F_x\dot{x}\right) + \Pi_t + \Pi_x\dot{x} =$$

$$= \dot{x}^t\left[H^T MHA_3^{-1}\left(D_1 + G_1 + Q_1 + BH + F_x\right)\right]\dot{x} + \left(H\dot{x} + f\right)^T MHA_3^{-1}\left(\left\{\dot{x}^T S_1\dot{x}\right\} + \left\{\dot{x}^T Q_2\dot{x}\right\} + Q_3\right) +$$

$$+\left[f^T MHA_3^{-1}\left(D_1 + G_1 + Q_1 + F_x\right) + \Pi_x + f^T\left(Q_x^T + R_x^T + B^T\right)A_3^{-1}H^T MH\right]\dot{x} +$$

$$+ f^T MHA_3^{-1}\left(Bf + F_t\right) + \Pi_t.$$

При малых $\|x\|$ и $\|\dot{x}\|$ для $\dot{V}$ имеем оценку $\dot{V} \le -\beta_0\|\dot{x}\|^2$, если выполнены условия

$$\dot{x}^T N_1\dot{x} \equiv \dot{x}^T H^T MHA_3^{-1}\left(D_1 + G_1 + Q_1 + BH + F_x\right)\dot{x} \le -2\beta_0\|\dot{x}\|^2,$$

$$N_2 = f^T MHA_3^{-1}\left(D_1 + G_1 + Q_{11} + F_x\right) + \Pi_x + \qquad (8)$$

$$+ f^T\left(Q_x^T + R_x^T + B^T\right)A_3^{-1}H^T MH,$$

$$N_3 - N_2^T N_1^{-1} N_2 \le 0, \; N_3 = f^T MHA_3^{-1}\left(Bf + F_t\right) + \Pi_t.$$

Множество положений равновесия $\dot{x} = x = 0$ системы (5) является изолированным, если при малых $\|x\|$

$$\Phi(t,x) = R(t,x) + Q_{12}(t,x) + Bf(t,x) = 0 \Leftrightarrow x = 0, \qquad (9)$$

при этом это свойство является невырожденным.

Соответственно [2] имеем следующий алгоритм построения управляющего воздействия, обеспечивающего стабилизацию любого программного движения $\left(\dot{q}^0(t), q^0(t)\right) \in L$ при условии (4).

Результат 1. Пусть матрица H управляющего сигнала и матрица B коэффициентов усиления таковы, что можно подобрать матрицу M, при которой для любых

$\left(\dot{q}(t), q(t)\right) \in L$ имеют место соотношения (8) и (9). Тогда управляющее воздействие (7) решает поставленную задачу о стабилизации.

Это воздействие будет также решать задачу о стабилизации в целом или глобально, если первое условие (8) в выборе B заменить на условие

$$\dot{x}^T N_1 \dot{x} + \left(H\dot{x} + f\right)^T MHA_3^{-1}\left(\left\{\dot{x}^T S_1 \dot{x}\right\} + Q_2\right) \le -2\beta_0 \left\|\dot{x}\right\|^2, \tag{10}$$

и потребовать невырожденность свойства (9) при любых $x \in R^n$:

$$\left(\forall \varepsilon > 0\right)\left(\exists \delta = \delta(\varepsilon) > 0\right)\left(\forall x : \left\|x\right\| \ge \varepsilon\right)\left\|\Phi\left(t, x\right)\right\| \ge \delta. \tag{11}$$

При невыполнении условия (4), при действии неопределенных сил, когда не достигается строгое существование заданного движения $\left(\dot{q}^0(t), q^0(t)\right)$, задача о стабилизации этого движения может быть решена на основе релейного управления. Такое управление может быть эффективно использовано и при ограничении на управления.

Введем нелинейное управление вида

$$U = B\Delta\left(H\dot{x} + f(t,x)\right), \tag{12}$$

$$\Delta = \begin{pmatrix} \Delta_1 \\ \Delta_2 \\ \vdots \\ \Delta_n \end{pmatrix} = \begin{pmatrix} sign\left(\left(Hx + f\right)_1\right) \\ sign\left(\left(Hx + f\right)_2\right) \\ \cdots \\ sign\left(\left(Hx + f\right)_n\right) \end{pmatrix}.$$

Допустим, что матрица коэффициентов усиления B подобрана таким образом, что для некоторой матрицы $M \in R^{n\times n}$, $M = const$ имеет место оценка

$$\left(H\dot{x} + f\right)^T MHA_3 B\Delta \le -2\eta_1 \left\|H\dot{x} + f\right\|, \quad \eta_1 = const > 0. \tag{13}$$

Тогда при достаточно малых $\left\|\dot{x}\right\| + \left\|x\right\|$, таких, что $\left(x, \dot{x}\right) \notin \left\{H\dot{x} + f = 0\right\}$ для производной функции V будем иметь оценку $\dot{V} \le -\eta_1 \left\|H\dot{x} + f\right\| \le 0$. Соответственно [3] имеем следующий результат.

Результат 2. Пусть матрица H и функция в структуре обратной связи с коэффициентами усиления – матрицей B таковы, что имеет место неравенство (11). Тогда релейное управление (12) решает поставленную задачу о стабилизации.

Рассмотрим задачу о стабилизации программного движения пространственного трёхзвенного манипулятора.

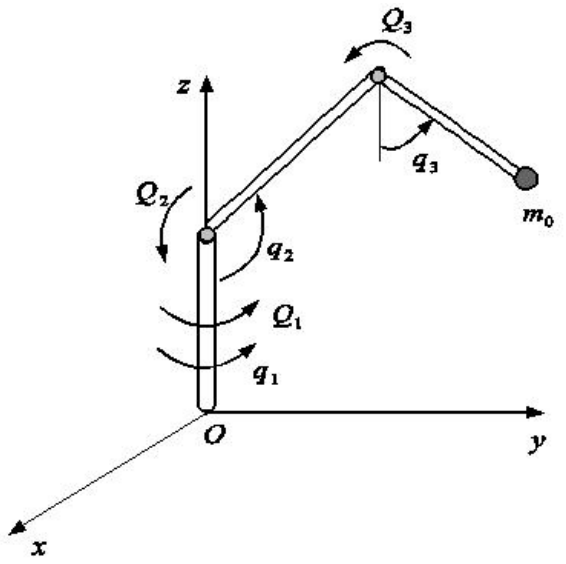


Рис. 1. Трёхзвенный манипулятор

Манипулятор состоит из трёх звеньев (рис. 1). Первое звено (вертикальная колонка) опирается на основание в точке O, второе и третье звенья расположены в вертикальной плоскости. В схвате третьего звена находится перемещаемый груз.

Обозначим: $q_i\ (i=1,2,3)$ – обобщённые координаты системы – углы поворотов звеньев манипулятора, l_i – длина звена, m_i – масса i-го звена, m_0 – масса груза, $m_{30}=m_3+m_0$, J_{01} – момент инерции первого звена относительно оси вращения, r_2 и r_3 – расстояния от центра тяжести второго звена и третьего звена с грузом соответственно до оси вращения этого звена, $k_i\ (i=1,2,3)$ – коэффициенты моментов сил вязкого трения, действующих в шарнирах, $Q_i\ (i=1,2,3)$ – управляющие моменты.

Уравнения движения системы будут следующими

$$\begin{cases}\left(J_{01}+m_2r_2^2\sin^2 q_2+m_{30}\left(l_2\sin q_2+r_3\sin q_3\right)^2\right)\ddot{q}_1+\\ +2\left(m_2r_2^2\sin q_2\cos q_2+m_{30}\left(l_2\sin q_2+r_3\sin q_3\right)\cdot l_2\cos q_2\right)\dot{q}_1\dot{q}_2+\\ +2m_{30}\left(l_2\sin q_2+r_3\sin q_3\right)\cdot r_3\cos q_3\dot{q}_1\dot{q}_3+k_1\dot{q}_1=Q_1,\\ \left(m_2r_2^2+m_3l_2^2\right)\ddot{q}_2+\frac{1}{2}m_{30}l_2r_3\cos\left(q_2-q_3\right)\ddot{q}_3+\\ +\frac{1}{2}m_{30}l_2r_3\sin\left(q_2-q_3\right)\dot{q}_3^2-\left(m_{30}\left(l_2\sin q_2+r_3\sin q_3\right)\cdot l_2+m_2r_2^2\sin q_2\right)\cos q_2\dot{q}_1^2+\\ +\left(m_2r_2^2+m_{30}l_2\right)g\sin q_2+k_2\dot{q}_2=Q_2,\\ \frac{1}{2}m_{30}l_2r_3\cos\left(q_2-q_3\right)\ddot{q}_2+m_{30}r_3^2\ddot{q}_3-\frac{1}{2}m_{30}l_2r_3\cdot\sin\left(q_2-q_3\right)\dot{q}_2^2-\\ -m_{30}\left(l_2\sin q_2+r_3\sin q_3\right)r_3\cos q_3\dot{q}_1^2+m_{30}gr_3\sin q_3+k_3\left(\dot{q}_3-\dot{q}_2\right)=Q_3.\end{cases}$$

Пусть выбрано программное движение $q_1=q_{10}(t),\ q_2=q_{20}(t),\ q_3=q_{30}(t)$ в виде

$$q_{10}(t)=0.2t\ \text{рад},\ \ q_{20}(t)=1{,}5+0.5\sin t\ \text{рад},\ \ q_{30}(t)=0.5\sin 0.5t\ \text{рад}.$$

Переменные q_1, q_2, q_3, описывающие движение системы, являются периодическими $\left(\mathrm{mod}\,2\pi\right)$. Поэтому в качестве нелинейных функций, определяющих состояние системы вблизи программного, удобно выбрать функции

$$f_1(q_1)=\sin q_1,\ f_2(q_2)=\sin q_2,\ f_3(q_3)=\sin q_3.$$

Матрицы H и B примем диагональными. Численно-аналитическое решение согласно результату 2 определило закон управления в виде релейного управления

$$\begin{cases} Q_1=\mu_1 sign\left(\sin\left(q_1-q_{10}(t)\right)+\alpha\left(\dot q_1-\dot q_{10}(t)\right)\right),\\ Q_2=\mu_2 sign\left(\sin\left(q_2-q_{20}(t)\right)+\alpha\left(\dot q_2-\dot q_{20}(t)\right)\right)+\mu_3 sign\left(q_3-q_{30}(t)+\alpha\left(\dot q_3-\dot q_{30}(t)\right)\right),\\ Q_3=\mu_3 sign\left(\sin\left(q_2-q_{20}(t)\right)+\alpha\left(\dot q_2-\dot q_{20}(t)\right)\right)+\mu_4 sign\left(q_3-q_{30}(t)+\alpha\left(\dot q_3-\dot q_{30}(t)\right)\right). \end{cases}$$

Ниже представлены результаты численного моделирования при следующих значениях параметров системы и программной траектории

$$J_{01}=0.1\,\text{кг}\times\text{м}^2,\ m_2=15\,\text{кг},\ m_3=2.5\,\text{кг},\ m_0=2\,\text{кг},$$
$$l_2=1\,\text{м},\ r_2=r_3=0.5\,\text{м},\ k_1=k_2=k_3=0.12\,\text{Н}\times\text{с}\times\text{м}.$$

Найдены следующие параметры управления $\mu_1=\mu_2=\mu_4=-50,\ \mu_3=10,\ \alpha=1$. На рисунке 2 представлены результаты моделирования при найденном управлении.

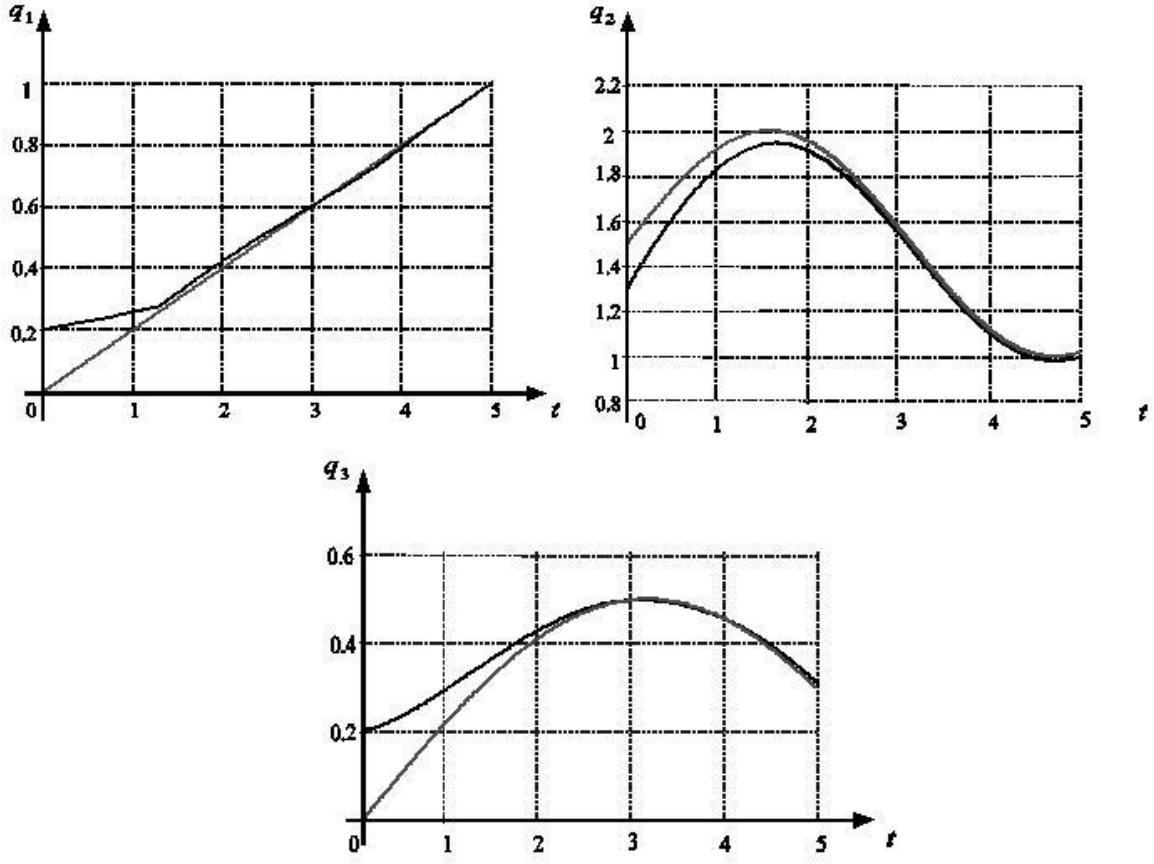


Рис. 2. Зависимость координат манипулятора от времени при релейном управлении

Для задачи стабилизации программного движения $q_1=0.2t$ рад, $q_2=\frac{\pi}{2}$, $q_3=\frac{\pi}{4}$ манипулятора найдено непрерывное управление вида

$$\begin{cases} Q_1=\mu_{11}\left(\sin\left(q_1-q_{10}(t)\right)+\alpha\left(\dot q_1-\dot q_{10}(t)\right)\right),\\ Q_2=\mu_{22}\left(\sin\left(q_2-q_{20}\right)+\alpha\dot q_2\right)+\mu_{23}\left(\sin\left(q_3-q_{30}\right)+\alpha\dot q_3\right),\\ Q_3=\mu_{32}\left(\sin\left(q_2-q_{20}\right)+\alpha\dot q_2\right)+\mu_{33}\left(\sin\left(q_3-q_{30}\right)+\alpha\dot q_3\right). \end{cases}$$

где $\mu_{ij}=const,\ \alpha=const.$

На рисунке 3 представлены результаты моделирования при найденном управлении.

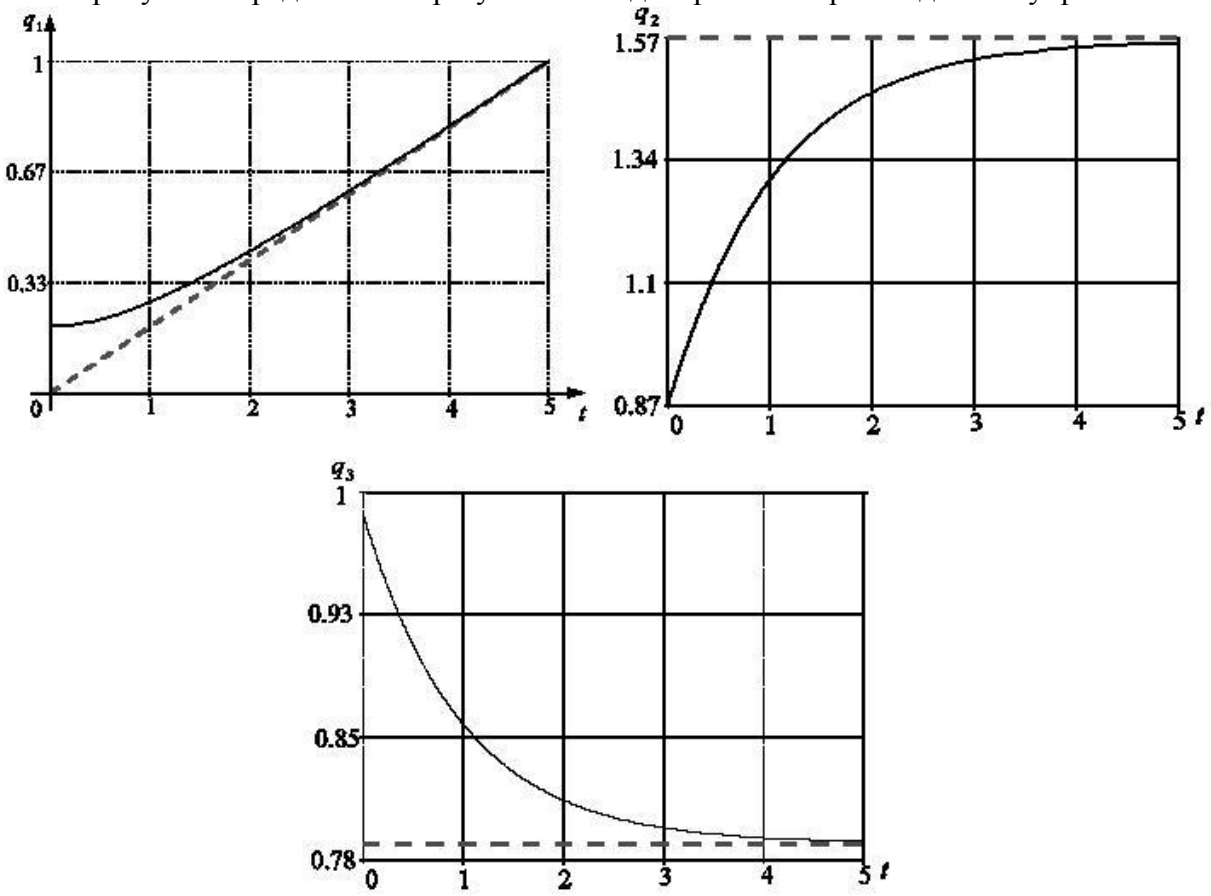


Рис. 3. Зависимость координат манипулятора от времени при непрерывном управлении

Преимущество релейного управления состоит в сходимости к заданному движению за меньший промежуток времени, недостаток — в возникновении эффекта чаттера (биения).

Работа выполнена при финансовой поддержке РФФИ (проект № 11-01-00541).

Литература

1. Маркеев А.П. Теоретическая механика – М.: ЧеРо, 1999. – 569 с.

2. Андреев А.С., Бойкова Т.А. Об устойчивости неустановившегося движения механической системы // ПММ, 2004 – Т.68, вып. 4. – С. 678-686.

3. Андреев А.С., Дмитриева О.Г, Петровичева Ю.В. Об устойчивости нулевого решения системы с разрывной правой частью // Научно-технический вестник Поволжья, 2011. – № 1. – С. 15-21.

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ДЕФОРМИРУЕМОСТИ УПРУГОГО ТЕЛА[1]

Д.М.Бодунов

Московский государственный машиностроительный университет (МАМИ)

Аннотация

Процессы пластического течения в тонком слое металла, поверхности движущегося тела обладают рядом особенностей, в частности, для них характерны высокие удельные давления, на порядок превышающие величины сдвиговых напряжений. Важная в практических приложениях проблема течения тонких пластических слоев в условиях анизотропии свойств материала и контактного трения на сегодня практически не исследована.

В работе приводится постановка и решение тестовой задачи течения тонкого слоя пластически анизотропного материала по границе упругой плоскости.

Формулировка классов новых задач, параметрический анализ их решений и классификация, построение новой математической модели деформируемости упругого тела при обтекании

Пусть тонкий слой пластического материала располагается между двумя параллельными поверхностями упругих тел инструментов, которые сближаются по направлению друг к другу [1–3]. В результате заданного движения инструментов сжимаемый слой растекается. В некоторый момент $t=0$, принимаемый за начальный, слой ограничен произвольным в плане xy кусочно-гладким контуром Г (рис. 1). Уравнение контура можно записать либо в явном виде:

$$y_0=\varphi(x_0),$$

либо параметрическими уравнениями:

$$x_0=x_0(\tau),\ y_0=y_0(\tau).$$

Расстояние $H(x,y,t)$ между рабочими поверхностями тел инструментов определяет толщину пластического слоя и полагается весьма малым сравнительно с протяжённостью слоя вдоль осей x и y. Размеры тел инструментов в целом больше размеров слоя.

Будем считать, что слой сжимается между поверхностями, уравнения которых в недеформированном состоянии имеют вид:

$$z=f_1(x,y,t),$$
$$z=f_2(x,y,t),$$

поэтому разность

$$f_1(x,y,t)-f_2(x,y,t)=H(x,y,t),$$

была бы известной толщиной слоя, сжимаемого жёсткими поверхностями.

В процессах течения тонких слоёв развиваются высокие удельные давления на поверхностях контакта с инструментом ($\sigma_n\sim\sigma_s\,\ell_0^2/h_0^2$), что приводит к их заметным упругим деформациям. Поэтому толщина слоя становится функцией процесса:

$$h(x,y,t)=H(x,y,t)+w(\sigma_n).$$

Если w_1, w_2 нормальные перемещения точек поверхностей f_1 и f_2, то толщина слоя запишется в виде:

$$h(x,y,t)=H(x,y,t)+w_1(x,y,t)+w_2(x,y,t).$$

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0230 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

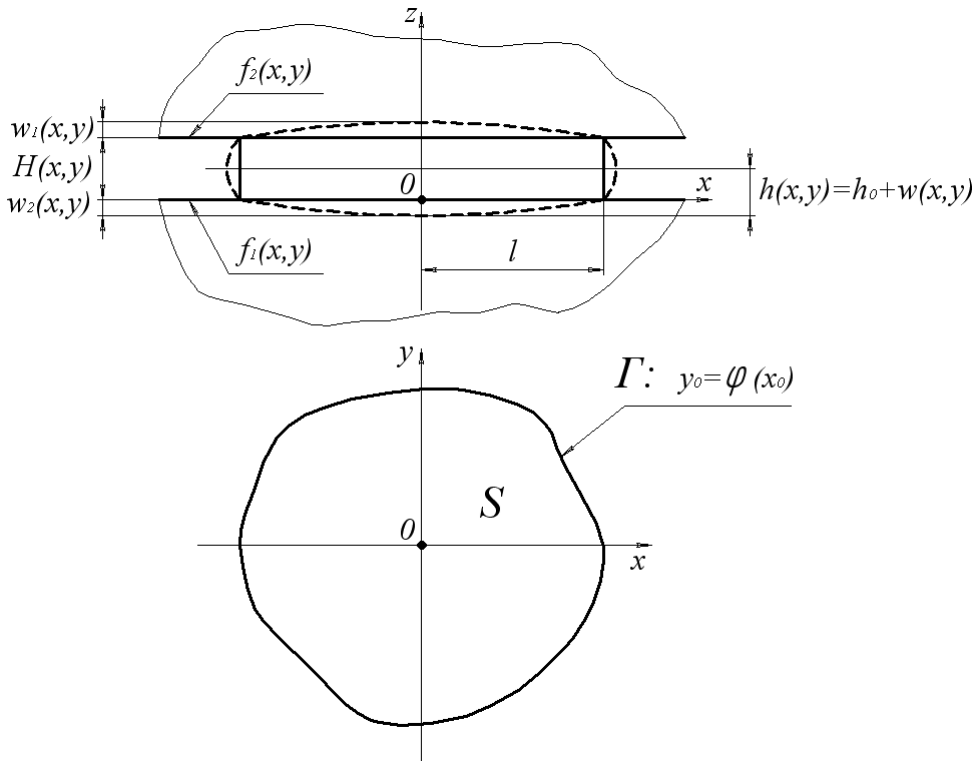


Рис. 1. Общая схема процесса

здесь $H(x,y,t)$ — функция изменяющейся толщины слоя в предположении того, что течение происходит по абсолютно жёстким поверхностям. В данной задаче представленный процесс течения будем рассматривать в относительно небольшой промежуток времени Δt, в течение которого контур $\Gamma(x,y,t)$ практически не изменяется, но степень деформации достаточная для того, чтобы весь слой перешел в состояние пластичности и начался процесс течения. В этом случае за $H(x,y,t)$ можно принять начальную толщину слоя, а в силу того, что время здесь — параметр несущественный и производных по нему нет (кинематика не рассматривается), то переменную t можно опустить при написании аргументов функций [1-3].

Предположим, что оба инструмента обладают одинаковыми физическими свойствами и геометрией, тогда можно допустить, что при сближении инструментов срединная плоскость слоя остаётся неизменной, что позволяет вследствие симметрии процесса рассматривать течение слоя вдвое меньшей толщины, ограниченного снизу упругим инструментом, а сверху — абсолютно жёстким телом. Если принять начальную толщину слоя h_0 за постоянную величину, фактическая толщина слоя с учетом упругих деформаций будет определяться (рис. 1):

$$h(x,y) = h_0 + w(x,y) \tag{1}$$

Таким образом, нахождение неизвестной толщины пластического слоя, определяемой (1), а также распределение давления в нём, при условии, что область контакта — фиксирована, представляется основной целью исследования.

Перейдём к математической формулировке задачи пластического течения в тонком слое. Обозначим p — давление со стороны слоя на поверхности контакта; $\overline{V}\{u,v\}$ — осреднённый по толщине вектор скорости частиц слоя; θ — угол наклона вектора скорости с осью x; σ_s — предел текучести материала при известных условиях процесса течения (температура, скорости и степени деформации и т.д.). Приняв все те гипотезы и упрощения, изложенные в [1–3], запишем систему основных уравнений движения для выделенного элемента слоя $hdxdy$ в виде:

$$\begin{cases} \dfrac{\partial p}{\partial x} = -\dfrac{2\tau_s}{h}\dfrac{u}{\sqrt{u^2+v^2}} = -\dfrac{2\tau_s}{h}\cos\theta \\ \dfrac{\partial p}{\partial y} = -\dfrac{2\tau_s}{h}\dfrac{v}{\sqrt{u^2+v^2}} = -\dfrac{2\tau_s}{h}\sin\theta \end{cases} \tag{2}$$

Возводя в квадрат левые и правые части (4.2) и складывая полученные уравнения, запишем соотношение, определяющее контактное давление p:

$$\left(\frac{\partial p}{\partial x}\right)^2 + \left(\frac{\partial p}{\partial y}\right)^2 = \frac{4\tau_s^2}{h^2} \text{ или } \left|grad\ p\right|^2 = \frac{4\tau_s^2}{h^2}, \tag{3}$$

где τ_s — предел текучести материала пластического слоя на сдвиг, $\sigma_s = \sqrt{3}\tau_s$.

Граничные условия для (3) определяются из следующих соображений: если контур Г свободен от внешних воздействий ($\sigma_{xx} = \sigma_{yy} = 0$), то $p(x_0; y_0) = \sigma_s$; если граница является пазом, так что в него может свободно втекать металл слоя, а ширина паза порядка или меньше толщины слоя h, то приближённым условием свободного втекания будет $p(x_0; y_0) = 2\sigma_s$. В общем случае запишем:

$$p(x_0; y_0) = \lambda\sigma_s, \tag{4}$$

где λ — множитель порядка единицы, $(x_0; y_0)$ — точки, принадлежащие фиксированному контуру Г.

Толщина слоя $h(x, y)$ в выражении (3) определяется при общей постановке задачи в процессе решения; в простейшем варианте, когда известна функция жёсткости $K(x, y, x', y')$ тела инструмента, для w имеем:

$$w(x, y) = \iint_S K(x, y, x', y')\, p(x', y')\, dx'dy'. \tag{5}$$

Определение функции влияния K для произвольного упругого трёхмерного тела представляет собой самостоятельную трудную задачу, поэтому в рамках данной работы мы остановимся на использовании известной функции Буссинеска, описывающей упругое полупространство [4]:

$$K = \frac{1}{\sqrt{(x-x')^2 + (y-y')^2}}.$$

Далее, после подстановки (1) в (3), получим

$$\left|grad\ p\right|^2 = \frac{4\tau_s^2}{\left(h_0 + w(x, y)\right)^2}. \tag{6}$$

Если область S фиксирована (что в дальнейшем предполагается), уравнения (5), (6) вместе с граничным условием (4) составляют краевую задачу для определения двух неизвестных функций p и h.

Далее приведем полученную систему уравнений, а также граничные условия к безразмерному виду, который в дальнейшем будем использовать. Координаты x, y, а также x', y' отнесем к l, перемещения w к h_0 (которую считаем постоянной) и оставим за ними прежние обозначения. Введем функцию давления по формуле $z = \dfrac{(p - \lambda\sigma_s)h_0}{2\tau_s l}$, тогда уравнение (6) и граничное условие (4) перепишутся в виде

$$\left|grad\ z\right|^2 = (1+w)^{-2}, \tag{7}$$

$$z(x_0, y_0) = 0 \tag{8}$$

Из (5) найдем

$$w = \delta_1 \iint_S K(x, y, x', y')\, dx'dy' + \delta_2 \iint_S K(x, y, x', y')\, z(x', y')\, dx'dy'. \tag{9}$$

В (9) приняты следующие обозначения:

$$\delta_1 = \delta\lambda \frac{l}{h_0}\sigma_s,\ \delta_2 = 2\delta\lambda \frac{l^2}{h_0^2}\tau_s.$$

Отметим, что параметр $\delta = \dfrac{1-\mu^2}{\pi E}$ характеризует размерность функции жесткости $K(x, y, x', y')$, здесь μ — коэффициент Пуассона, E — модуль Юнга.

Разработка численно-аналитического метода последовательных приближений.

Поскольку система включает нелинейное дифференциальное уравнение в частных производных и интегральное соотношение, применим метод последовательных приближений [5]. Полагая $w = 0$ в первом приближении (в этом случае инструмент представляет собой абсолютно жесткое тело), решаем дифференциальное уравнение (7). Решением будет функция z в первом приближении. При известном z из (9) находим w во втором приближении. Далее процесс повторяется. Алгоритм метода можно представить так:

$$\left| grad\ z^{(k)} \right|^2 = \left(1 + w^{(k)}\right)^{-2},$$

$$z^{(k)}(x_0, y_0) = 0,$$

$$w^{(k+1)} = \delta_1 \iint_S K(x, y, x', y')\, dx'dy' + \delta_2 \iint_S K(x, y, x', y')\, z^{(k)}(x', y')\, dx'dy',$$

где k — номер итерации.

Задача о течении тонкого слоя идеально пластического материала по поверхности, ограничивающей упругое полупространство в указанной постановке исследована в работах [6, 7].

Скорость сходимости метода от показателей анизотропии

В работе [8], которая стала обобщением теории течения тонкого пластического слоя на случай анизотропии материала и контактного трения, было показано, что уравнение, связывающее давление в слое ортотропного материала с его толщиной (а, значит, и упругими деформациями инструмента) математически тождественно уравнению (3) и имеет вид:

$$\left(\frac{\partial p}{\partial x}\right)^2 + \beta^2 \left(\frac{\partial p}{\partial y}\right)^2 = \frac{4\tau_s^2}{h^2}, \tag{10}$$

где β — отношение пределов текучести материала вдоль осей x и y соответственно, т.е. показатель анизотропии. После подстановки $y = \beta\eta$ в (10), получаем

$$\left(\frac{\partial p}{\partial x}\right)^2 + \left(\frac{\partial p}{\partial \eta}\right)^2 = \frac{4\tau_s^2}{h^2}, \tag{11}$$

что дает возможность решить задачу (11), (4), (5) с помощью уже известного метода приближений [5].

Следует отметить, что каждая итерация метода приближений приводит к необходимости перехода от одной системы координат к другой. Задача для определения давления решается в плоскости (x, η), а задача по определению прогибов в плоскости (x, y).

Наличие большого числа параметров задачи позволило провести достаточно глубокий анализ, целью которого является оценка влияния параметра анизотропии на скорость сходимости метода последовательных приближений.

Решение модельных, тестовых и новых задач; параметрический анализ и классификация процессов обтекания с приложением к исследованию управляемых движений упруго-твердого тела в вязкой среде

Ниже (рис 2, 3) представлены некоторые результаты исследования задачи течения тонкого пластически ортотропного слоя, занимающего в плане прямоугольную область с соотношением сторон 2:1, относительной толщиной 1/20 и значениями параметра анизотропии 0,7 и 1,3 (исследования проводились в диапазоне значений 0,7…1,3 с шагом 0,2, для областей с соотношением сторон 2:1 и 3:1, толщин 1/20, 1/30, 1/40). Вычислительные процедуры проводились с помощью пакета Maple 13.

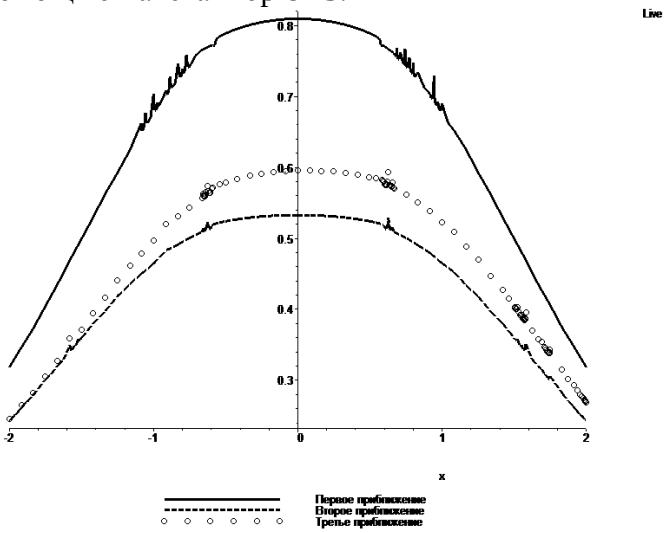


Рис. 2. Сечение поверхности $w(x, y)$ плоскостью $y = 0$. $\beta = 0,7$

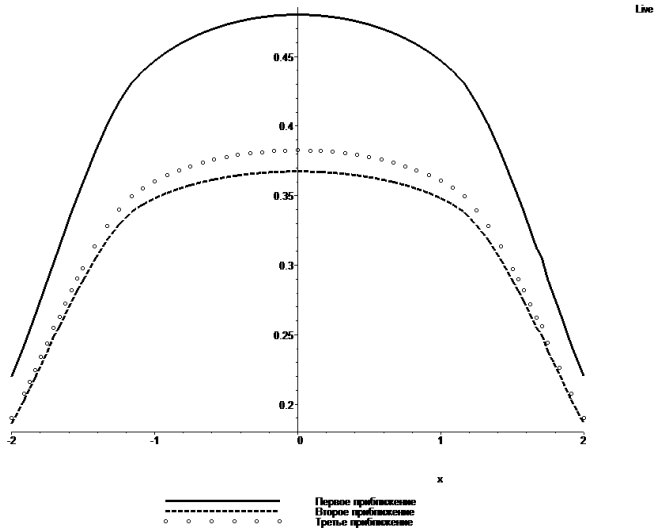


Рис. 3. Сечение поверхности $w(x, y)$ плоскостью $y = 0$. $\beta = 1,3$

В качестве основных выводов можно установить следующее:

1. Установлена фактическая сходимость метода последовательных приближений, предложенного для решения системы интегро-дифференциальных уравнений;

2. Характер поверхности перемещений качественным образом не зависит от параметров задачи;

3. Значение параметра анизотропии может оказать существенное влияние на скорость сходимости метода, поскольку влияет на распределение давления в слое, тем самым подтверждаются полученные ранее результаты исследований – увеличение давления приводит к снижению скорости сходимости метода. Очевидно, первое приближение является грубым, но второе, для ряда значений параметров задачи, может дать довольно точную оценку величины упругих перемещений контактных поверхностей инструмента.

Литература

1. Ильюшин А.А. Вопросы теории течения пластического вещества по поверхностям // Прикл. матем. и мех. – 1954, Т. 18, № 3. – С. 265—288.

2. Ильюшин А.А. Полная пластичность в процессах течения между жесткими поверхностями, аналогия с песчаной насыпью и некоторые приложения // Прикл. матем. и мех. – 1955, т. 19, № 6. – с. 693-713.

3. Кийко И.А. Теория пластического течения в тонком слое металла. – М.: Инст. мех. МГУ, 1971. – 66 с.

4. Качанов Л.М. Основы теории пластичности // М., Наука. – 1969. – 420 с.

5. Кийко И.А. Вариационный принцип в задачах течения тонкого слоя пластического вещества // ДАН СССР. – 1964, т. 157, № 3. – с. 551-553.

6. Бодунов Д.М. Течение тонкого слоя идеально-пластического материала по деформируемым поверхностям: Дис... канд. физ.-мат.н. – М., МГТУ МАМИ, 2004. – 163 с.

7. Кийко И.А., Анизотропия в процессах течения тонкого пластического слоя//ПММ, 2006, т.70, вып.2, с. 344-351.

8. Бодунов Д.М., Бодунов М.А., Коваленко П.В., Задача о течении пластического вещества в фиксированной области, имеющей форму равнобокой трапеции. Инструмент – упругое полупространство. Известия МГТУ «МАМИ», 2007, №2(4), с.229-239.

ПОДХОДЫ К ОЦЕНИВАНИЮ МОМЕНТОВ РАЗЛАДОК ТОЧЕЧНОГО ПРОЦЕССА

А.А.Волков

Ульяновский государственный университет

В прикладных исследованиях часто возникает необходимость определять, в какой момент времени произошла смена характеристик изучаемого объекта. Решение такого рода задач сводится к обнаружению моментов разладок случайного процесса.

Пусть задано число T $\left(0<T<+\infty\right)$ и определен стохастический базис $\mathrm{B}_T=\left(\Omega, F, \mathbf{F}=(F_t)_{t\in[o;T]}, \mathbf{P}\right)$ с обычными условиями Деллашери [1], на котором задан точечный процесс $A=\left(A_t\right)_{t\geq 0}$. В случайный момент времени τ происходит разладка с вероятностью $P\{\tau\leq x\}=\begin{cases}1-e^{-\lambda x}, x\in[0;T]\\ 0, x\leq 0\end{cases}$ и с вероятностью $e^{-\lambda T}$ разладка не происходит. Предположим, что точечный процесс $A=\left(A_t\right)_{t\geq 0}$ имеет компенсатор

$$\tilde{A}_t=\int_0^t\left(\gamma+\alpha\cdot I\left(\tau\leq s\right)\right)ds. \tag{1}$$

Для решения задачи оценивания момента разладки сформулируем и докажем Теорему 1 и Теорему 2.

Теорема 1. Приближение момента разладки τ процесса $A=\left(A_t\right)_{t\geq 0}$ при $t\in[0,T]$, определяемое формулой (2):

$$\hat{\tau}_1=\frac{(\gamma+\alpha)T-A_T}{\alpha},\ \hat{\tau}_1=\hat{\tau}_1(T) \tag{2}$$

является асимптотически несмещенным при $T\to\infty$, т.е. выполняется условие: $E\hat{\tau}_1-E\tau\xrightarrow{T\to\infty}0$.

Доказательство теоремы 1.

Обозначим τ - момент разладки. По формуле (1) до момента τ процесс $A=\left(A_t\right)_{t\geq 0}$ в среднем достигает значения: $A_\tau=\gamma\left(\tau\wedge T\right)+m_{(\tau\wedge T)}$, где $m=\left(m_t\right)_{t\geq 0}$ - мартингал на стохастическом базисе B и, таким образом, процесс с $Em_\tau=Em_t\equiv 0$ для всех (как детерминированных, так и случайных) моментов остановки $t\in[0,\tau]$. Тогда как после момента τ процесс возрастает на $\left(T-\left(\tau\wedge T\right)\right)\left(\gamma+\alpha\right)$: $A_\tau=\tilde{A}_t+m_t$, но

$$\begin{aligned}\tilde{A}_T&=\int_0^T\left\{\gamma+\alpha\cdot I\left(\tau\leq s\right)\right\}ds=\gamma\cdot T+\alpha\cdot\left(T-\tau\right)^+=\\ &=\gamma\cdot T+\alpha\cdot\left(T-\tau\right)\cdot\left\{I\left(\tau\leq T\right)+I\left(\tau>T\right)-I\left(\tau>T\right)\right\}=\\ &=\gamma\cdot T+\alpha\cdot\left(T-\tau\right)-\alpha\cdot\left(T-\tau\right)\cdot I\left(\tau>T\right).\end{aligned} \tag{3}$$

Выразим из выражения (3) момент разладки τ:

$$\tau=T-\frac{A_T-\gamma\cdot T}{\alpha}+\frac{m_T}{\alpha}-\left(T-\tau\right)\cdot I\left(\tau>T\right). \tag{4}$$

Обозначим $\hat{\tau}_1=T-\frac{A_T-\gamma\cdot T}{\alpha}=\frac{(\gamma+\alpha)T-A_T}{\alpha}$, $\hat{\tau}_1=\hat{\tau}_1(T)$, тогда выражение (4) примет следующий вид:

$$\tau = \hat{\tau}_1 + \frac{m_T}{\alpha} - (T - \tau) \cdot I(\tau > T).$$

Выразим приближение $\hat{\tau}_1$ момента разладки τ:

$$\hat{\tau}_1 = \tau - \frac{m_T}{\alpha} + (T - \tau) \cdot I(\tau > T)$$

Найдем математическое ожидание приближения $\hat{\tau}_1$:

$$E\hat{\tau}_1 = E\left\{\tau - \frac{m_T}{\alpha} + (T - \tau) \cdot I(\tau > T)\right\} =$$

$$= E\tau - E\frac{m_T}{\alpha} + E((T - \tau) \cdot I(\tau > T)). \quad (5)$$

$E\frac{m_T}{\alpha} = \frac{1}{\alpha} Em_T = \frac{1}{\alpha} \cdot 0 = 0$, так как $m = (m_t)_{t \geq 0}$ - мартингал на стохастическом базисе B.
Подставим полученный результат в (5), тогда:

$$E\hat{\tau}_1 = E\tau - E((\tau - T) \cdot I(\tau > T)) = E\tau - \frac{e^{-\lambda T}}{\lambda},$$

но в таком случае:

$$E\hat{\tau}_1 - E\tau = \frac{e^{-\lambda T}}{\lambda} \xrightarrow{T \to \infty} 0.$$

Следовательно, приближение $\hat{\tau}_1 = \frac{(\gamma + \alpha)T - A_T}{\alpha}$ момента разладки τ процесса $A = (A_t)_{t \geq 0}$ является асимптотически несмещенным при $T \to \infty$.
Теорема 1 доказана.

Следствие 1. Приближение момента разладки τ процесса $A = (A_t)_{t \geq 0}$ при $t \in [0, T]$, определяемое формулой (6):

$$\hat{\tau}_2 = \hat{\tau}_1 + \frac{e^{-\lambda T}}{\lambda} = \frac{(\gamma + \alpha)T - A_T}{\alpha} + \frac{e^{-\lambda T}}{\lambda} \quad (6)$$

является несмещенным, т.е. выполняется условие: $E\hat{\tau}_2 = E\tau$.

Доказательство следствия 1.

Из Теоремы 1 следует, что:

$$E\hat{\tau}_1 - E\tau = \frac{e^{-\lambda T}}{\lambda} \xrightarrow{T \to \infty} 0, \text{ где } \hat{\tau}_1 = \frac{(\gamma + \alpha)T - A_T}{\alpha}.$$

Тогда:

$$\hat{\tau}_2 = \hat{\tau}_1 + \frac{e^{-\lambda T}}{\lambda} = \frac{(\gamma + \alpha)T - A_T}{\alpha} + \frac{e^{-\lambda T}}{\lambda}$$

является несмещенным приближением момента разладки τ в силу:

$$E\hat{\tau}_2 = E\hat{\tau}_1 + \frac{e^{-\lambda T}}{\lambda} = E\tau.$$

Следствие 1 доказано.

Построим оптимальную в среднеквадратическом смысле оценку момента разладки.

Теорема 2. (об общем виде оценки момента разладки). Условное математическое ожидание момента разладки τ является оптимальной в среднеквадратическом смысле оценкой для τ процесса $A = (A_t)_{t \geq 0}$ при $t \in [0, T]$ определяется формулой (7):

$$\hat{\tau} = E\left(\tau \middle| F_t^A\right) = \frac{(\gamma+\alpha)t - A_t}{\alpha} \cdot \pi_t + \left(t + \frac{1}{\lambda}\right) \cdot (1 - \pi_t), \tag{7}$$

где $\pi_t = P\left(\tau \le t \middle| F_t^A\right)$.

Доказательство теоремы 2.

Компенсатор процесса $A = (A_t)_{t\ge 0}$ имеет вид (1), тогда:

$$A_t = \begin{cases} \gamma t + m_t & при\ t \le \tau \\ \gamma\tau + (t-\tau)(\gamma+\alpha) + m_t & при\ t > \tau \end{cases}$$

Но в то же время:

$$A_t = t(\gamma+\alpha) - \tau\alpha + m_t, \tag{8}$$

где $m = (m_t)_{t\ge 0}$ - мартингал на стохастическом базисе B.

Следовательно, из (8) получаем:

$$\tau = \frac{(\gamma+\alpha)t - A_t}{\alpha} + \frac{m_t}{\alpha}.$$

Введем процесс $X_t = I(\tau \le t)$, тогда:

$$E\left(\tau \mid \sigma\left(F_t^A, X_t\right)\right) = \frac{(\gamma+\alpha)t - A_t}{\alpha} \cdot I(t \ge \tau) + \left(t + \frac{1}{\lambda}\right) \cdot I(t < \tau).$$

По определению оптимальной в среднеквадратическом смысле оценки момента разладки τ процесса $A = (A_t)_{t\ge 0}$ получим:

$$\hat{\tau} = E\left(\tau \mid F_t^A\right),$$

но

$$E\left(\tau \mid F_t^A\right) = E\left(E\left(\tau \mid F_t^{A,X}\right) \mid F_t^A\right) =$$

$$= \frac{(\gamma+\alpha)t - A_t}{\alpha} \cdot P\left(t \ge \tau \mid F_t^A\right) + \left(t + \frac{1}{\lambda}\right) \cdot P\left(t < \tau \mid F_t^A\right). \tag{9}$$

Обозначим $\pi_t = P\left(\tau \le t \mid F_t^A\right)$, тогда (9) примет вид:

$$\hat{\tau} = E\left(\tau \mid F_t^A\right) = \frac{(\gamma+\alpha)t - A_t}{\alpha} \cdot \pi_t + \left(t + \frac{1}{\lambda}\right) \cdot (1 - \pi_t).$$

Теорема 2 доказана.

Теорема 2 дает представление об общем виде оценки момента разладки точечного процесса, но не о способе её вычисления в силу сложности точного подсчета $\pi_t = P\left(\tau \le t \mid F_t^A\right)$. Поэтому при численной оценке целесообразно использовать выведенные приближения.

В результате проведенного эксперимента было выявлено, что приближение $\hat{\tau}_2$ является более эффективным в среднеквадратическом смысле, чем приближение $\hat{\tau}_1$ при условии отсутствия информации о появлении разладки в рассматриваемый период. Однако, если известно, что $\tau \le T$, то оценка $\hat{\tau}_2$ является менее эффективной в среднеквадратическом смысле, чем $\hat{\tau}_1$.

Для достижения наибольшей точности данный результат позволяет в зависимости от полноты располагаемой информации об исследуемом объекте использовать либо одну из приведенных выше оценок, либо их комбинацию.

Случайный процесс может иметь не только одну разладку или ни одной, но также их может быть несколько, поэтому приведем следующую задачу. Дан процесс $A = (A_t)_{t\ge 0}$ со

множественными разладками на стохастическом базисе $\mathrm{B}_T = \left(\Omega, F, \mathbf{F} = (F_t)_{t\in[o;T]}, \mathbf{P}\right)$ с обычными условиями Деллашери [1]. Задан считающий процесс числа разладок $\rho = (\rho_t)_{t\geq 0}$, представляющий собой пуассоновский процесс с интенсивностью λ. Выбор интенсивности λ зависит от поставленной перед моделью задачи. Процесс $A = (A_t)_{t\geq 0}$ описан компенсатором вида:

$$\tilde{A}_t = \int_0^t \left(\gamma + \sum_{i=0}^{\infty} \alpha_i I(\tau_i \leq s < \tau_{i+1}) \right) ds, \tau_0 = 0,$$

где $\tau_i = \inf(t : \rho_t \geq i)$ - моменты разладок; $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_k$ - известные коэффициенты интенсивности.

Перед оцениванием моментов разладок, требуется оценить их количество. Пусть для $A = (A_t)_{t\in[0;T]}$ в эксперименте выпало ρ_T разладок за время $T > 0$. После того, как произошла разладка, процесс $A = (A_t)_{t\geq 0}$ меняет «угол наклона», что сопровождается сменой характеристики процесса – коэффициента α_i. При этом известны коэффициенты интенсивности $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_k$. Однако возникает вопрос – насколько должен измениться угол и как близко могут располагаться моменты, чтобы считать текущий момент времени разладкой.

Введем аппроксимирующую ломаную f_t для процесса $A = (A_t)_{t\geq 0}$, где каждому излому f_t будет соответствовать момент разладки, то есть ломаная с k изломами и прямыми под «углами» $x_0, x_1, x_2, \ldots, x_k$ такая, что $\min_{i\geq 1}(\hat{\tau}_{i+1} - \hat{\tau}_i) \geq \Delta$. Общий вид аппроксимирующей ломаной:

$$f_t(k; x_1, \ldots, x_k) = \sum_{i=0}^{k} x_i \int_0^t I(\hat{\tau}_i < s \leq \hat{\tau}_{i+1}) ds, \qquad (10)$$

где $\hat{\tau}_1, \hat{\tau}_2, \ldots, \hat{\tau}_k$ - оценочные моменты разладок, а $\hat{\tau}_0 = 0$, $\hat{\tau}_{k+1} = T$.

Введем функционал потерь, который будет учитывать ошибку в оценивании числа разладок и ошибку отклонения ломаной f_t от траектории процесса A. Обозначим Φ_1 - плата за неверное число разладок, а Φ_2 - плата за «плохую» аппроксимацию при заданном числе разладок. Тогда функционал потерь $\Phi(\varphi_1, \varphi_2; k)$ примет вид:

$$\Phi(\varphi_1, \varphi_2; k) = \varphi_1 \cdot \Phi_1 + \varphi_2 \cdot \Phi_2,$$

где φ_1, φ_2 - параметры управления; k - количество изломов для рассматриваемой аппроксимации $f_t(k; \alpha_0, \ldots, \alpha_k)$.

В итоге функционал потерь имеет вид:

$$\Phi(\varphi_1, \varphi_2; k) = \varphi_1 \cdot (\rho_T - k)^2 + \varphi_2 \cdot E \inf_{(f_t^k)_{0\leq t\leq T}} \int_0^T (A_t(\rho) - f_t)^2 dt. \qquad (11)$$

Следовательно, оценить число разладок процесса $A = (A_t)_{t\geq 0}$, значит решить задачу поиска такого значения $\hat{k}$, которое приводит функционал потерь (11) к наименьшему значению:

$$\Phi(\varphi_1, \varphi_2; k) \to \inf_k : \hat{k} = \arg\min_k \Phi(\varphi_1, \varphi_2; k). \qquad (12)$$

Рассмотрим второй подход к оценке числа разладок в задаче с известной группой параметров $\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_k$.

Введем точечный процесс $Y=(Y_t)_{t\geq 0}$, определяемый следующим стохастическим дифференциальным уравнением:

$$dY_t=\beta(\lambda t+m_t^{\rho})dt+dm_t^{Y},$$

где $\beta=\beta(x)=\beta_0-\beta_1\cdot x$, причем $\beta_0>\beta_1\cdot(\lambda T)$, $x\in[0;T]$; m_t^{ρ} - мартингал в разложении пуассоновского процесса; m_t^{Y} - мартингал процесса $Y=(Y_t)_{t\geq 0}$.

Функция $\beta=\beta(x)$ строится следующим образом. Пусть значения изначально известных коэффициентов интенсивности появления разладок $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$ убывают с ростом k. Тогда их можно аппроксимировать линейной функцией вида $\beta(t)=\beta_0-\beta_1\cdot t$. Но интенсивность α_i, $i=0..k$, не может быть отрицательной, т.е. $\alpha_i\geq 0$ для всех $i=0\ldots k$, поэтому необходимо выполнение условия, что $\beta(t)\geq 0\Rightarrow\beta_0\geq\beta_1\cdot t$.

Рассмотрим частично наблюдаемую схему, где $x^{(T)}=(x_t^{(T)})_{t\geq 0}$ ненаблюдаемая компонента, а наблюдение - $y^{(T)}=(y_t^{(T)})_{t\geq 0}$.

$$\begin{cases} dx_t^{(T)}=\dfrac{1}{\sqrt{T}}(d\rho_t-\lambda dt) \\ dy_t^{(T)}=\dfrac{1}{\sqrt{T}}\{\beta(\rho_t)-\beta(\lambda t)\}dt+\dfrac{1}{\sqrt{T}}dm_t^{Y} \end{cases}. \qquad (13)$$

Процесс $x=(x_t)_{t\geq 0}$ интерпретируется как ошибка оценки фактического числа разладок в сравнении со средним их числом. Следовательно:

$$\rho_T=\sqrt{T}\cdot x_T^{(T)}+\lambda T. \qquad (14)$$

Опишем схему Калмана-Бьюси для данной частично-наблюдаемой схемы:

$$\begin{cases} d\pi_t=-\alpha_1\gamma_t T(dy_t+\alpha_1\pi_t dt) \\ d\gamma_t=\left(-(\alpha_1\gamma_t)^2T+\dfrac{1}{T}\right)dt \end{cases},$$

где $\pi_t=E(x_t\mid F_t^{y})$, $\gamma_{ij}(t)=E\left[(x_i(t)-\pi_i(t))(x_j(t)-\pi_j(t))\right]$.

Таким образом, получили оценку в классе линейных для $x_t^{(T)}$:

$$\hat{x}_t^{(T)}=\pi_t=E(x_t^{(T)}\mid F_t^{y}). \qquad (15)$$

Подставим (15) в (14) и получим оценку числа разладок рассматриваемого точечного процесса $\hat{\rho}_T=\sqrt{T}\cdot\hat{x}_T^{(T)}+\lambda T$.

При найденном одним из выше описанных методов числе разладок $\hat{k}$ опишем алгоритм поиска самих моментов. Для его реализации требуется следующий вид аппроксимирующей ломаной (10):

$$f_t(\hat{k};\alpha_1,\ldots,\alpha_{\hat{k}})=\sum_{i=0}^{\hat{k}}\alpha_i\int_0^t I(\hat{\tau}_i<s\leq\hat{\tau}_{i+1})ds,$$

где $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_{\hat{k}}$ - известные коэффициенты интенсивности; $\hat{\tau}_1,\hat{\tau}_2,\ldots,\hat{\tau}_{\hat{k}}$ - оценочные моменты разладок.

Введем понятие эмпирической функции распределения $F_n^{(i)}(t)$, $i=1\ldots\hat{k}$. Построив эмпирическую функцию распределения для каждой $\hat{\tau}_i, i=1\ldots\hat{k}$, можно ранжировать оценки моментов разладок по наибольшему числу появления в фиксированных диапазонах.

Рассмотрим модель задачи со множественными разладками и неизвестной группой параметров. В большинстве случаев коэффициенты $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$ изначально неизвестны, т.к. сложно предсказать насколько сильно будет влияние на рассматриваемый процесс того или иного фактора. Таким образом, отсутствует возможность построения функции $\beta=\beta(x)=\beta_0-\beta_1\cdot x$, а, следовательно, можно использовать только первый способ оценки числа разладок с некоторыми модификациями.

Оценки самих моментов разладки, в отличие от точечного процесса с известными коэффициентами интенсивности, нельзя строить на основе второго метода оценивания самих моментов, т.к. $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$ - неизвестны. Поэтому для поиска оценок в данной модели воспользуемся понятием эмпирической функции распределения.

Рассмотрим процесс $A=(A_t)_{t\geq 0}$ с аналогичными условиями, описанными выше, и компенсатором:

$$\tilde{A}_t=\int_0^t\left(\gamma+\sum_{i=0}^{\infty}\alpha_i I(\tau_i\leq s<\tau_{i+1})\right)ds,\ \tau_0=0,$$

где $\tau_i=\inf(t:\rho_t\geq i)$; $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$ - неизвестные коэффициенты интенсивности.

Пусть для $A=(A_t)_{t\in[0;T]}$ в эксперименте произошло ρ_T разладок за время $T>0$. Введем аппроксимирующую ломаную f_t для процесса $A=(A_t)_{t\geq 0}$, где каждому излому f_t будет соответствовать момент разладки, то есть ломаная с k изломами и прямыми под изначально неизвестными «углами» $\hat{\alpha}_0,\hat{\alpha}_1,\hat{\alpha}_2,\ldots,\hat{\alpha}_k$ такая, что $\min_{i\geq 1}(\hat{\tau}_{i+1}-\hat{\tau}_i)\geq\Delta$.

По аналогии со случаем известных коэффициентов $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$ введем функционал потерь (11) с модифицированной ломаной $f_t=f_t(k;\hat{\alpha},\ldots,\hat{\alpha}_k)$. Следовательно, определить число разладок процесса $A=(A_t)_{t\geq 0}$, значит решить задачу вида (12).

При найденном числе разладок $\hat{k}$ рассмотрим аппроксимирующую ломаную $f_t=f_t(\hat{k};\hat{\alpha},\ldots,\hat{\alpha}_{\hat{k}})$ вида (10). Аналогично случаю с известными коэффициентами $\alpha_0,\alpha_1,\alpha_2,\ldots,\alpha_k$, построив эмпирическую функцию распределения для каждой $\hat{\tau}_i$, $i=1..\hat{k}$, можно ранжировать оценки моментов разладок по наибольшему числу появления в фиксированных диапазонах.

Автор выражает благодарность за внимание к работе профессору Бутову А.А.

Работа выполнена при поддержке грантов РФФИ: проекты 06-01-00338 и 08-01-97009.

Литература

1. Деллашери К. Емкости и случайные процессы. – М.: МИР, 1975.

МАТЕМАТИЧЕСКИЕ И КОМПЬЮТЕРНЫЕ ИМИТАЦИОННЫЕ МОДЕЛИ РЕГУЛЯЦИИ СИСТОЛИЧЕСКОГО АРТЕРИАЛЬНОГО ДАВЛЕНИЯ

М.С.Гаврилова, А.А.Бутов

Ульяновский государственный университет

В настоящее время артериальная гипертензия (АГ) является одним из наиболее распространенных неинфекционных заболеваний. В России около 40 млн. мужчин и женщин страдают АГ и ее осложнениями (ишемические и геморрагические инсульты, инфаркт миокарда, атеросклероз и другие). АГ представляет собой хроническое повышение артериального давления (АД) от 140/90 мм рт. ст. и выше. В связи с этим актуальным является построение компьютерных имитационных моделей регуляции АД у таких пациентов.

В большинстве работ математическое описание медико-биологических процессов дано в терминах обыкновенных дифференциальных уравнений [1–3]. В этом случае рассматривается поведение объекта без учета случайных воздействий внутренней и внешней среды. Организм человека — это сложная система с элементами хаотичной структуры, в которой не представляется возможным со стопроцентной вероятностью предсказать её состояние в каждый момент времени. Мы не можем учесть все параметры такой системы, поэтому детерминированные математические модели не могут адекватно описать ее поведение. При изучении динамики АД особый интерес представляет стохастическое имитационное моделирование [4–6].

В настоящей работе построены детерминированные модели циркадианного ритма систолического АД (САД) и стохастические модели регуляции САД в моменты стрессовых ситуаций.

Объекты и методы исследования

Проводилось исследование, в которое включены 71 женщина и 102 мужчины, из них 46 женщин и 71 мужчина страдали АГ. Пациенты находились под наблюдением в лаборатории артериальной гипертонии Ульяновского клинического госпиталя ветеранов войн в период с 2008–2010 гг. По результатам суточного мониторирования АД (СМАД) пациенты были разделены на 6 групп без учета возраста: лица без АГ (25 женщин и 31 мужчина), лица с АГ без терапии (32 женщины и 56 мужчин) и лица с АГ на терапии (14 женщин и 15 мужчин).

СМАД проводилось с использованием носимого монитора «BPLab МнСДП-3» (ООО «Петр Телегин», Нижний Новгород). Для статистической обработки данных использовался пакет прикладных программ BPLab v. 3.0 (ООО «Петр Телегин», Нижний Новгород). Для реализации компьютерной имитационной модели был выбран язык программирования Borland Delphi 7.0 (Borland Software Corporation, USA).

Математические модели циркадианного ритма САД

На основе анализа экспериментальных данных была сформулирована гипотеза о представлении суточного профиля САД в виде суммы медленных, устойчивых, периодических колебаний (циркадианный ритм) и резких, случайных колебаний, которые невозможно предсказать со стопроцентной вероятностью (вариабельность).

Циркадианный, или околосуточный ритм играет важнейшую роль для организма человека. Это понятие предложил американский ученый Франц Халберг в 1959 г. Циркадианный ритм САД представляет собой периодические двухфазные колебания «день-ночь» с периодом, близким к суткам (24 ± 4 часа).

В идеале у лиц без АГ и лиц с АГ на терапии гомеостатические механизмы должны поддерживать значения АД в определенных границах в течение некоторого периода времени (как днем, так и ночью), который мы назовем промежутком стабилизации. У таких пациентов предполагается наличие дневного и ночного промежутков стабилизации САД, что свидетельствует о нормальной работе гомеостатических механизмов регуляции АД.

У лиц с АГ без терапии работа системы саморегуляции АД нарушена, поэтому давление совершает значительные колебания, которые приводят к структурным изменениям его суточного профиля. У таких пациентов, как правило, отсутствуют промежутки стабилизации АД.

Данную гипотезу о структуре суточного профиля САД подтвердил анализ данных СМАД, который показал, что чаще всего промежутки стабилизации САД наблюдались у лиц без сердечно-сосудистых патологий и лиц с АГ на терапии.

Наличие промежутков стабилизации на графике суточного профиля САД является главной особенностью разработанных математических моделей циркадианного ритма САД.

Пусть случайный процесс $Y = (Y_t)_{t\geq 0}$ описывает динамику САД в мм рт. ст., время t измеряется в часах. Выделим у процесса Y детерминированную и стохастическую составляющие. Циркадианный ритм САД опишем детерминированным процессом $C(t)$, мм рт. ст. Тогда случайные изменения САД (вариабельность) представим в виде стохастического процесса.

В качестве приближения циркадианного ритма САД возьмем непрерывную функцию

$$C(t) = \lambda g(t) + \rho \qquad (1)$$

с периодом $T > 0$, где T — количество времени (в часах), в течение которого проводилось СМАД, $\lambda > 0$ — амплитуда колебаний САД, $\rho > 0$ — средний уровень САД по данным СМАД на $[0,T]$, мм рт. ст.

Пусть даны две числовые последовательности $\{t_i\}$ и $\{y_i\}$, i меняется от 0 до n–1. Число y_i — значение САД в мм рт. ст., полученное по результатам СМАД в момент времени $t_i \in [0,T]$ для любого i. Тогда параметры модели: $\rho = \frac{1}{n}\sum_{i=0}^{n-1} y_i$, $\lambda = \max\limits_{0\leq i\leq n-1} |y_i - \rho|$.

Модель 0

Классическим методом описания циркадианных ритмов САД является их аппроксимация синусоидой, поэтому в качестве функции $g(t)$ можно рассмотреть $g(t) = a\sin(\omega t)$, где $a > 0$, $\omega > 0$. Параметр $\omega = \frac{2\pi}{T}$, arcsin(мм рт. ст.)/час; параметр a вычисляется с помощью метода наименьших квадратов (МНК).

Данная модель, в виду простоты формул, удобна при решении прикладных задач, связанных с построением циркадианного ритма САД. Однако простота модели одновременно является и ее недостатком, поскольку в ней не учитываются некоторые важные регуляторные механизмы, формирующие суточный профиль САД. Одним из таких механизмов является процесс саморегуляции АД (гомеостаз).

В настоящей работе предложены 4 новые математические модели циркадианного ритма САД, основанные на анализе экспериментальных данных. Эти модели сохраняют достоинство классического подхода (простоту формул) и в то же время более детально описывают циркадианный ритм САД.

Модель 1

Пусть непрерывная функция $g(t)$, мм рт. ст., имеет вид:

$$g(t) = \begin{cases} a_1 \sin(\varphi(t)), 0 \leq t \leq t^0 \\ a_2 \sin(\varphi(t)), t^0 < t \leq T \end{cases},$$

где промежутки $[0, t^0]$ и $(t^0, T]$ рассматриваются как дневной и ночной периоды суток соответственно, $t^0 \in (0,T)$, часы. Параметры a_l, $l = 1,2$, определяются с помощью МНК. Для типов ночного снижения САД «non-dipper», «dipper» и «over-dipper» $a_l > 0$, $l = 1,2$. Для

типа ночного снижения САД «night-peaker» $a_1 > 0$, $a_2 < 0$. Непрерывная неубывающая функция $\varphi(t)$, arcsin(мм рт. ст.), — фаза САД, $\varphi(t) \in [0, 2\pi]$ при $t \in [0, T]$.

Определим $\varphi(t)$ следующим образом. Если по данным СМАД на графике суточного профиля САД:

1) Нет промежутка стабилизации при $t \in [0, t^0]$, то $\varphi(t) = k_1 t$, где параметр $k_1 = \frac{\pi}{t^0}$, arcsin(мм рт. ст.)/час.

2) Есть промежуток стабилизации при $t \in [0, t^0]$, то

$$\varphi(t) = \begin{cases} \frac{\varphi_1 t}{t^1}, 0 \le t < t^1 \\ \varphi_1, t^1 \le t \le t^2 \\ \frac{(\pi - \varphi_1)t}{t^0 - t^2} + \frac{\varphi_1 t^0 - \pi t^2}{t^0 - t^2}, t^2 < t \le t^0 \end{cases}$$

Здесь $[t^1, t^2]$ – дневной промежуток стабилизации САД, t^1, t^2 измеряются в часах, $0 < t^1 < t^2 < t^0$. Параметр c_1, мм рт. ст., найден как выборочное среднее значение последовательности $\{s_i\}$, где $s_i = \frac{y_i - \rho}{\lambda}$ (2). При этом все s_i, мм рт. ст., рассматриваются в моменты времени $t_i \in [t^1, t^2]$. Параметр $\varphi_1 = \arcsin\left(\frac{c_1}{a_1}\right)$. Если $\frac{c_1}{a_1} > 1$, примем $a_1 = 1$.

3) Нет промежутка стабилизации при $t \in (t^0, T]$, то $\varphi(t) = k_2 t + b_2$, где параметры $k_2 = \frac{\pi}{T - t^0}$, arcsin(мм рт. ст.)/час, $b_2 = \frac{\pi(T - 2t^0)}{T - t^0}$, arcsin(мм рт. ст.).

4) Есть промежуток стабилизации при $t \in (t^0, T]$, то

$$\varphi(t) = \begin{cases} \frac{(\varphi_2 - \pi)t}{t^3 - t^0} + \frac{\pi t^3 - \varphi_2 t^0}{t^3 - t^0}, t^0 < t < t^3 \\ \varphi_2, t^3 \le t \le t^4 \\ \frac{(2\pi - \varphi_2)t}{T - t^4} + \frac{\varphi_2 T - 2\pi t^4}{T - t^4}, t^4 < t \le T \end{cases}$$

Здесь $[t^3, t^4]$ – ночной промежуток стабилизации САД, t^3, t^4 измеряются в часах, $t^0 < t^3 < t^4 < T$. Параметр c_2, мм рт. ст., найден как выборочное среднее значение последовательности $\{s_i\}$, при этом все s_i, мм рт. ст., рассматриваются в моменты времени $t_i \in [t^3, t^4]$. Параметр $\varphi_2 = \pi - \arcsin\left(\frac{c_2}{a_2}\right)$. Если $\frac{c_2}{a_2} < -1$, примем $a_2 = 1$ для типов ночного снижения САД «non-dipper», «dipper» и «over-dipper» и $a_2 = -1$ для типа ночного снижения САД «night-peaker».

Функция $\varphi(t)\in[0,\pi]$ при $t\in[0,t^0]$, $\varphi(t)\in(\pi,2\pi]$ при $t\in(t^0,T]$ и является непрерывной на $[0,T]$.

При построении циркадианного ритма САД $C(t)$ на любом промежутке времени $[0,R]$, $R>T$, функция $g_R(t)=\begin{cases}a_1\sin(\varphi_R(t)), jT\le t\le t^0+jT\\ a_2\sin(\varphi_R(t)), t^0+jT<t\le(j+1)T\end{cases}$, фаза САД $\varphi_R(t)=\varphi(t)+2\pi j$, при $t\in[jT,(j+1)T]$, где $j=\overline{0,m}$, $m=\left[\frac{R}{T}\right]$ (функция $[L]$ — целая часть числа L). Тогда $C(t)=\lambda g_R(t)+\rho$.

В отличие от классической синусоиды, уравнения новой модели описывают процесс регуляции САД, учитывая такие особенности его суточного профиля как наличие промежутков стабилизации. В модели также учитывается изменение биологических параметров АД при переходе от дневного периода к ночному, поскольку функция $C(t)$ представляет собой совокупность двух синусоид, параметры которых изменяются в зависимости от периода суток. Модель 1 может быть построена для любого из четырех типов ночного снижения САД. Таким образом, математическая модель 1 описывает динамику циркадианного ритма САД более корректно, чем классическая синусоида.

Недостатком модели является то, что обе синусоиды, задающие функцию $C(t)$, стремятся достичь своих максимумов при $t\in[0,t^0]$ и $t\in(t^0,T]$ соответственно и делают скачок в своей области определения. Каждый скачок частично компенсируется умножением синусоиды на коэффициент a_1 (a_2) при условии $c_1\le a_1$ ($|c_2|\le|a_2|$).

Модель 2

Пусть непрерывная функция $g(t)$, мм рт. ст., имеет вид:

$$g(t)=\begin{cases}k_1t, 0\le t<t^1\\ c_1, t^1\le t<t^2\\ f_1(t), t^2\le t\le t^0\\ f_2(t), t^0<t\le t^3\\ c_2, t^3<t\le t^4\\ k_2t+b_2, t^4<t\le T\end{cases} \qquad (3)$$

В модели (2) промежутки $[0,t^0]$ и $(t^0,T]$ представляют собой дневной и ночной периоды суток, $t^0\in(0,T)$, часы. Отрезки $[t^1,t^2]$ и $[t^3,t^4]$ — дневной и ночной промежутки стабилизации САД, все t^k, $k=\overline{1,4}$, измеряются в часах, $0<t^1<t^2<t^0<t^3<t^4<T$. Параметры модели: $k_1=\frac{c_1}{t^1}$, (мм рт. ст.)/час, $k_2=\frac{-c_2}{T-t^4}$, (мм рт. ст.)/час, $b_2=\frac{c_2T}{T-t^4}$, мм рт. ст. Коэффициенты c_1 и c_2, мм рт. ст., рассчитываются как выборочные средние значения последовательности $\{s_i\}$ (2) на дневном и ночном промежутках стабилизации САД, т. е. при $t_i\in[t^1,t^2]$ и $t_i\in[t^3,t^4]$соответственно.

Функции $f_l(t)$, $l=1,2$, мм рт. ст., задаются следующим образом: $f_1(t)=a_1\sin(w_1t+d_1)+h_1$, $f_2(t)=a_2\sin(w_2t+d_2)+h_2$, где a_1, a_2 (безразмерные

величины), w_1, w_2 , arcsin(мм рт. ст.)/час, d_1, d_2, arcsin(мм рт. ст.), h_1, h_2, мм рт. ст., — параметры модели. Значения параметров a_l, w_l, d_l, h_l, $l=1,2$, вычисляются с помощью методов многомерной условной оптимизации (например, МНК в сочетании с методами барьерных функций и Хука-Дживса) при условиях непрерывности функции $g(t)$ на $[0,T]$: $f_1(t^2)=c_1$, $f_1(t^0)=0$, $f_2(t^0)=0$, $f_2(t^3)=c_2$.

Функцию $g(t)$ (а значит, и $C(t)$) можно продолжить на любой промежуток времени $[0,R]$, где $R>T$. Если $m=\left[\frac{R}{T}\right]$, то при $t\in[jT,(j+1)T]$, $j=\overline{0,m}$, $g(t)=g(t-jT)$, и, следовательно, $C(t)=C(t-jT)$.

Модель 2 обладает всеми достоинствами модели 1: наличие промежутков стабилизации САД, изменение параметров модели при переходе от дневного периода к ночному, возможность построения циркадианного ритма САД для любого типа ночного снижения. Кроме того, у нее отсутствует недостаток модели 1, поскольку все параметры синусоид $f_l(t)$, $l=1,2$, рассчитываются на основе экспериментальных данных с помощью методов многомерной оптимизации. В связи с этим данная математическая модель циркадианного ритма САД является более корректной, чем модель 1 и классическая синусоида.

Модель 3

Пусть непрерывная функция $g(t)$, мм рт. ст., имеет вид:

$$g(t)=\begin{cases} k_1 t, 0\le t<t^1 \\ c_1, t^1\le t<t^2 \\ f(t), t^2\le t\le t^3 \\ c_2, t^3<t\le t^4 \\ k_2 t+b_2, t^4<t\le T \end{cases} \qquad (4)$$

Здесь отрезки $[t^1,t^2]$ и $[t^3,t^4]$ рассматриваются как дневной и ночной промежутки стабилизации САД соответственно, все t^k, $k=\overline{1,4}$, измеряются в часах, $0<t^1<t^2<t^3<t^4<T$. Параметры модели: $k_1=\frac{c_1}{t^1}$, (мм рт. ст.)/час, $k_2=\frac{-c_2}{T-t^4}$, (мм рт. ст.)/час, $b_2=\frac{c_2 T}{T-t^4}$, мм рт. ст. Коэффициенты c_1 и c_2, мм рт. ст., рассчитываются как выборочные средние значения последовательности $\{s_i\}$ (2) при $t_i\in[t^1,t^2]$ и $t_i\in[t^3,t^4]$соответственно.

Функция $f(t)=a\sin(wt+d)+h$, где a (безразмерная величина), w, arcsin(мм рт. ст.)/час, d, arcsin(мм рт. ст.), h, мм рт. ст., — параметры модели. Значения параметров a, w, d, h определяются по данным СМАД с помощью методов многомерной условной оптимизации (например, МНК в сочетании с методами барьерных функций и Хука-Дживса) при условиях: 1) функция $g(t)$ непрерывна для любого $t\in[0,T]$: $f(t^2)=c_1$, $f(t^3)=c_2$; 2) уравнение $f(t)=0$ должно иметь ровно 1 корень $t=t^0\in(t^2,t^3)$, при этом $f(t^2)>0$, $f(t^3)<0$.

Функцию $g(t)$, а значит, и $C(t)$, можно продолжить на любой промежуток времени $[0,R]$, где $R>T$. Если $m=\left[\frac{R}{T}\right]$, то при $t\in[jT,(j+1)T]$, $j=\overline{0,m}$, $g(t)=g(t-jT)$, и, следовательно, $C(t)=C(t-jT)$.

Модель 3 так же, как и две предыдущие модели, учитывает наличие промежутков стабилизации САД и изменение биологических параметров АД при переходе из одного состояния в другое. В соответствии с экспериментальными данными в качестве функции перехода была выбрана синусоида $f(t)$.

Данная математическая модель является корректной для аппроксимации циркадианных ритмов САД у пациентов с типами ночного снижения САД «non-dipper», «dipper» и «over-dipper».

Модель 4

Пусть непрерывная функция $g(t)$, мм рт. ст., имеет вид:

$$g(t)=\begin{cases}k_1t, 0\le t<t^1\\ c_1, t^1\le t<t^2\\ q(t), t^2\le t\le t^3\\ c_2, t^3<t\le t^4\\ k_2t+b_2, t^4<t\le T\end{cases} \qquad (5)$$

Здесь отрезки $[t^1,t^2]$ и $[t^3,t^4]$ — дневной и ночной промежутки стабилизации САД соответственно, все t^k, $k=\overline{1,4}$, измеряются в часах, $0<t^1<t^2<t^3<t^4<T$. Параметры модели: $k_1=\frac{c_1}{t^1}$, (мм рт. ст.)/час, $k_2=\frac{-c_2}{T-t^4}$, (мм рт. ст.)/час, $b_2=\frac{c_2T}{T-t^4}$, мм рт. ст. Коэффициенты c_1 и c_2, мм рт. ст., рассчитываются как выборочные средние значения последовательности $\{s_i\}$ (2) при $t_i\in[t^1,t^2]$ и $t_i\in[t^3,t^4]$ соответственно.

Функция $q(t)=a\exp(wt+d)+h$, где $a>0$ (безразмерная величина), w, Ln(мм рт. ст.)/час, d, Ln(мм рт. ст.), $h<0$, мм рт. ст., — параметры модели. При этом $wt+d<0$ для любого $t\in(t^2,t^3)$. Значения параметров a, w, d, h определяются по данным СМАД с помощью методов многомерной условной оптимизации (например, МНК в сочетании с методами барьерных функций и Хука-Дживса) при условиях: 1) функция $g(t)$ непрерывна на $[0,T]$: $q(t^2)=c_1$, $q(t^3)=c_2$; 2) корень уравнения $q(t)=0$, $t=t^0$, должен принадлежать интервалу (t^2,t^3), $t^0=\frac{1}{w}\left(Ln\left(-\frac{h}{a}\right)-d\right)$.

Функцию $g(t)$, а значит, и $C(t)$, можно продолжить на любой промежуток времени $[0,R]$, где $R>T$. Если $m=\left[\frac{R}{T}\right]$, то при $t\in[jT,(j+1)T]$, $j=\overline{0,m}$, $g(t)=g(t-jT)$, и, следовательно, $C(t)=C(t-jT)$.

Модель 4 так же, как и модели 1–3, учитывает наличие промежутков стабилизации САД и изменение биологических параметров АД при переходе из одного состояния в другое.

В соответствии с данными СМАД в качестве функции перехода может быть рассмотрена экспонента $q(t)$.

Данная математическая модель, как и модель 3, является корректной для построения циркадианных ритмов САД у пациентов с типами ночного снижения САД «non-dipper», «dipper» и «over-dipper».

К недостаткам модели можно отнести чувствительность параметров экспоненты $q(t)$ к выбору начальных данных при проведении многомерной условной оптимизации.

Математическая модель регуляции САД

В современных условиях социальной среды человек испытывает состояние практически постоянного стресса. Как известно, стресс является частой причиной повышения АД. Чем сильнее стрессовое воздействие на организм, тем большие колебания совершает АД. Чрезмерное влияние стресса на сердечно-сосудистую систему может стать пусковым механизмом для развития АГ [7].

Стресс приводит к активации симпатоадреналовой системы, в результате которой в кровь выбрасывается большое количество катехоламинов (адреналин, норадреналин) из мозгового слоя надпочечников. Эти гормоны повышают уровень АД. Тогда организм запускает гомеостатические механизмы регуляции АД, которые понижают уровень катехоламинов в крови и оказывают сосудорасширяющее действие, тем самым снижая давление.

Пусть случайный процесс $X = (X_t)_{t\geq 0}$ — концентрация катехоламинов в крови в момент времени t, нмоль/л. Процесс выброса надпочечниками катехоламинов в моменты стресса имеет вид $\eta(t)dN_t$, где $N = (N_t)_{t\geq 0}$ — стандартный пуассоновский процесс с интенсивностью $\mu(t) > 0$, $\mu(t) = \begin{cases} \mu_1, jT \leq t \leq t^0 + jT \\ \mu_2, t^0 + jT < t \leq (j+1)T \end{cases}$, $j = \overline{0,m}$, $\mu_i > 0$, $i = 1,2$;

$\eta(t) > 0$ — коэффициент роста, нмоль/л, $\eta(t) = \begin{cases} \eta_1, jT \leq t \leq t^0 + jT \\ \eta_2, t^0 + jT < t \leq (j+1)T \end{cases}$, $j = \overline{0,m}$, $\eta_i > 0$, $i = 1,2$. Тогда $(-\alpha(t))X_t$ — гомеостатические механизмы, снижающие уровень катехоламинов в крови после стрессовых ситуаций, нмоль/л, $\alpha(t) > 0$ — коэффициент отрицательной обратной связи (коэффициент затухания), $\alpha(t) = \begin{cases} \alpha_1, jT \leq t \leq t^0 + jT \\ \alpha_2, t^0 + jT < t \leq (j+1)T \end{cases}$, $j = \overline{0,m}$, $\alpha_i > 0$, $i = 1,2$.

Процесс повышения САД под воздействием катехоламинов имеет вид $\beta(t)X_t$, где $\beta(t) > 0$ — коэффициент роста, (мм рт. ст.)·л/нмоль, $\beta(t) = \begin{cases} \beta_1, jT \leq t \leq t^0 + jT \\ \beta_2, t^0 + jT < t \leq (j+1)T \end{cases}$, $j = \overline{0,m}$, $\beta_i > 0$, $i = 1,2$. Чтобы нормализовать давление, организм запускает процесс саморегуляции $(-\gamma(t))Y_t$, мм рт. ст., который частично компенсирует стрессовое воздействие на уровень АД, $\gamma(t) > 0$ — коэффициент отрицательной обратной связи (коэффициент затухания), $\gamma(t) = \begin{cases} \gamma_1, jT \leq t \leq t^0 + jT \\ \gamma_2, t^0 + jT < t \leq (j+1)T \end{cases}$, $j = \overline{0,m}$, $\gamma_i > 0$, $i = 1,2$. Другие факторы внутренней и внешней среды, влияющие на уровень САД (гормоны и биологически активные вещества,

физические нагрузки, качество питания, метеоусловия, гиподинамия и т.д.), обозначим как $\sigma(t)Y_t dW_t$, мм рт. ст., где $\sigma(t) \neq 0$ — пропорциональный коэффициент, $\sigma(t) = \begin{cases} \sigma_1, jT \le t \le t^0 + jT \\ \sigma_2, t^0 + jT < t \le (j+1)T \end{cases}$, $j = \overline{0,m}$, $\sigma_i \neq 0$, $i = 1,2$; $W = (W_t)_{t \ge 0}$ — стандартный винеровский процесс. Процессы N и W независимы.

Математическая модель регуляции САД в моменты стрессовых ситуаций представляет собой систему стохастических дифференциальных уравнений

$$\begin{cases} dX_t = \eta(t)dN_t - \alpha(t)X_t dt \\ dY_t = \beta(t)X_t dt - \gamma(t)Y_t dt + \sigma(t)Y_t dW_t + C(t)dt \end{cases} \quad (6)$$

с начальными условиями $X_0 \ge 0$, $Y_0 > 0$. В качестве функции $C(t)$ можно выбрать любое приближение циркадианного ритма САД (например, модели 1–4).

Предполагается, что в системе (6) процесс X ненаблюдаемый, а наблюдать можно только процесс Y, который содержит неполную информацию о процессе X. Такая система называется частично-наблюдаемой.

В модели (6) случайный процесс X совершает скачки в моменты времени, совпадающие с моментами скачков пуассоновского процесса N. Процесс Y непрерывен, имеет множественные разладки. Разладкой наблюдаемого случайного процесса называется изменение его вероятностных характеристик (математического ожидания, дисперсии) в случайный момент времени (момент появления разладки) [8]. В настоящей работе разладкой является воздействие стресса на организм. Тогда моменты возникновения разладок — это моменты стрессовых ситуаций. У процесса Y они совпадают с моментами скачков пуассоновского процесса N.

Компьютерные модели регуляции САД

На рис. 1–5 изображены графики функций $C^k(t)$, $k = \overline{1,5}$, построенные на основе моделей 1–4 и классической модели (жирные линии). Тонкой линией обозначен график, построенный по экспериментальным данным. В приведенном примере показатели СМАД взяты у пациента из группы женщин с ГБ на терапии, тип ночного снижения САД «dipper». Имеются два промежутка стабилизации САД. Общие для всех моделей параметры: $R = T = 19.43$ часа, дискретность по времени $\Delta_t = 0.01$ часа, амплитуда колебаний циркадианного ритма САД $\lambda = 37.16$, средний уровень САД $\rho = 156.16$ мм рт. ст.

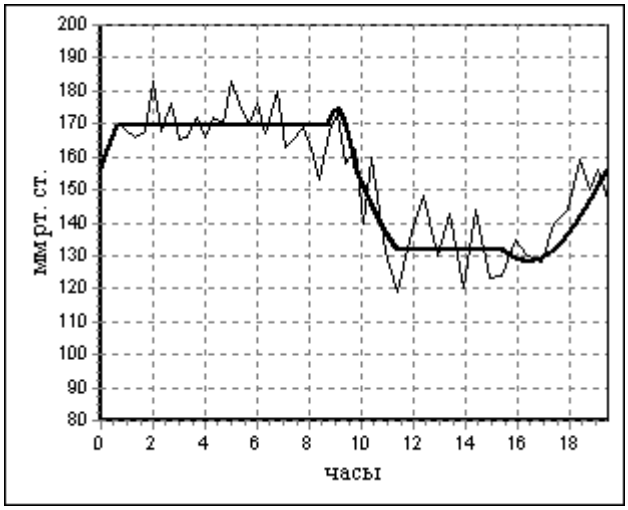


Рис. 1 График функции $C^1(t)$, модель 1.

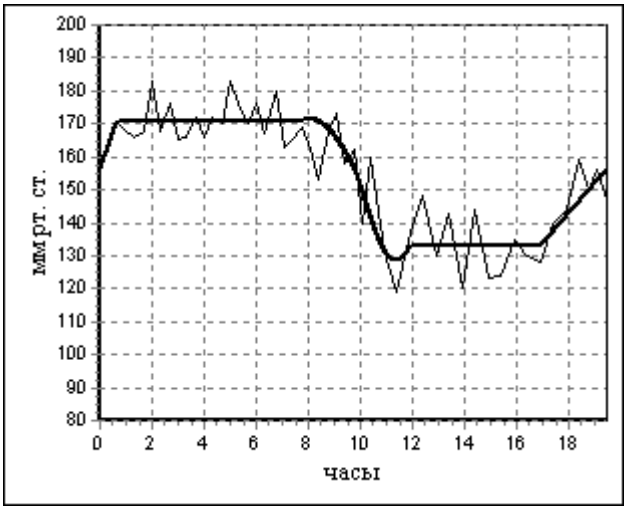


Рис. 2 График функции $C^2(t)$, модель 2.

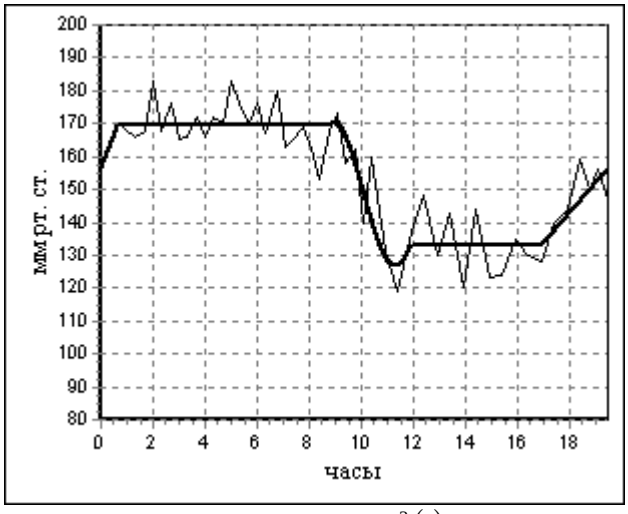


Рис. 3 График функции $C^3(t)$, модель 3.

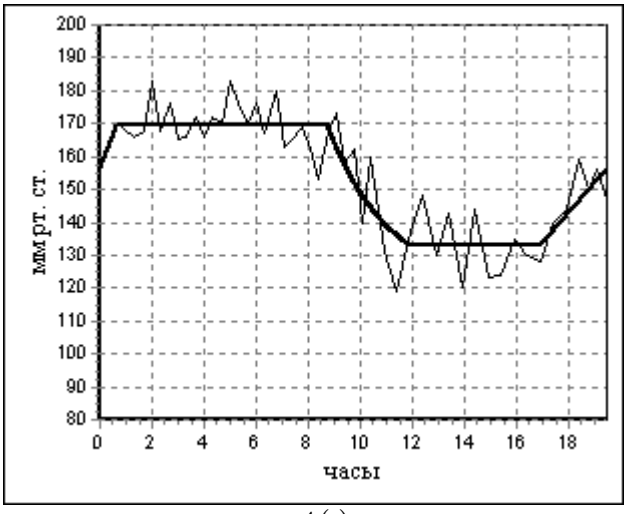


Рис. 4 График функции $C^4(t)$, модель 4.

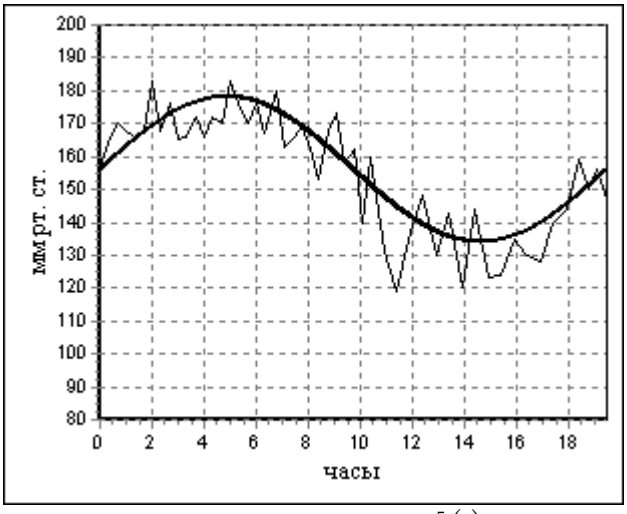


Рис. 5 График функции $C^5(t)$.

Как видно из рисунков, классическая модель описывает изменения циркадианного ритма САД менее адекватно, чем модели 1–4. Главным ее недостатком является отсутствие промежутков стабилизации САД. Параметры всех моделей приведены в таблицах 1–5.

Таблица 1. Параметры функции $C^1(t)$, модель 1.

t^0 = 9.8 часа	a_1 = 0.49
t^1 = 0.67 часа	c_1 = 0.36 мм рт. ст.
t^2 = 8.72 часа	k_2 = 0.33 arcsin(мм рт. ст.)/час
t^3 = 11.38 часа	b_2 = –0.055 arcsin(мм рт. ст.)
t^4 = 15.38 часа	a_2 = 0.75
k_1 = 0.32 arcsin(мм рт. ст.)/час	c_2 = –0.65 мм рт. ст.

Таблица 2. Параметры функции $C^2(t)$, модель 2.

t^0 = 9.8 часа	d_1 = 0.87 arcsin(мм рт. ст.)
t^1 = 0.67 часа	h_1 = -0.21 мм рт. ст.
t^2 = 7.72 часа	a_2 = 0.52
t^3 = 11.88 часа	w_2 = -1.3 arcsin(мм рт. ст.)/час
t^4 = 16.88 часа	d_2 = 0.62 arcsin(мм рт. ст.)
k_1 = 0.59 (мм рт. ст.)/час	h_2 = -0.22 мм рт. ст.
c_1 = 0.4 мм рт. ст.	c_2 = -0.63 мм рт. ст.
a_1 = 0.62	k_2 = 0.245 (мм рт. ст.)/час
w_1 = -0.7 arcsin(мм рт. ст.)/час	b_2 = -4.76 мм рт. ст.

Таблица 3. Параметры функции $C^3(t)$, модель 3.

t^0 = 9.83 часа	a = 0.59
t^1 = 0.67 часа	w = -1.32 arcsin(мм рт. ст.)/час
t^2 = 8.72 часа	d = 0.74 arcsin(мм рт. ст.)
t^3 = 11.88 часа	h = -0.2 мм рт. ст.
t^4 = 16.88 часа	c_2 = -0.63 мм рт. ст.
k_1 = 0.54 (мм рт. ст.)/час	k_2 = 0.245 (мм рт. ст.)/час
c_1 = 0.36 мм рт. ст.	b_2 = -4.76 мм рт. ст.

Таблица 4. Параметры функции $C^4(t)$, модель 4.

t^0 = 9.47 часа	a = 2.59
t^1 = 0.67 часа	w = -0.42 Ln(мм рт. ст.)/час
t^2 = 8.72 часа	d = 2.99 Ln(мм рт. ст.)
t^3 = 11.88 часа	h = -0.99 мм рт. ст.
t^4 = 16.88 часа	c_2 = -0.63 мм рт. ст.
k_1 = 0.54 (мм рт. ст.)/час	k_2 = 0.245 (мм рт. ст.)/час
c_1 = 0.36 мм рт. ст.	b_2 = -4.76 мм рт. ст.

Таблица 5. Параметры функции $C^5(t)$ модель 0.

t^0 = 9.7 часа	w = 0.32 (мм рт. ст.)/час
a = 0.59	

На рисунках 6–9 представлены траектории процессов Y^k, $k = \overline{1,4}$, построенных на основе различных приближений циркадианного ритма САД (модели 1–4 соответственно).

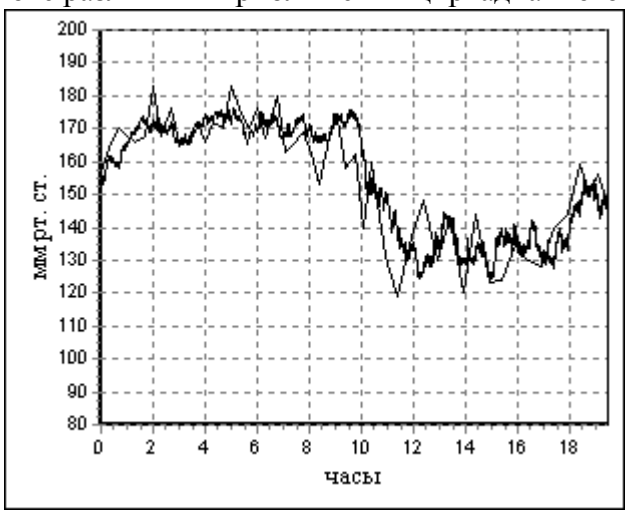


Рис. 6 Траектория процесса Y^1.

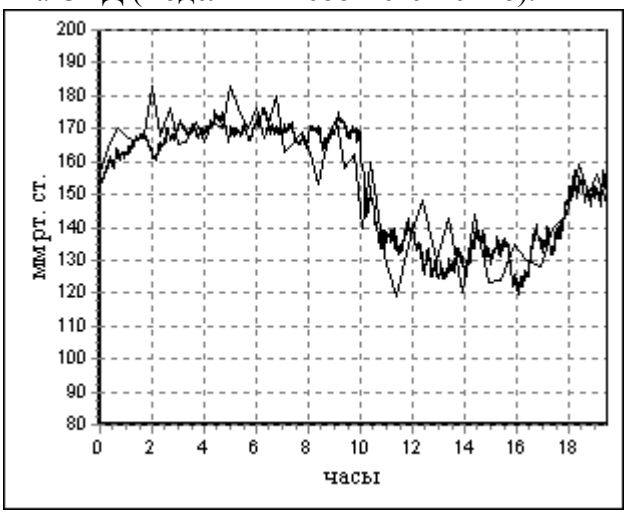


Рис. 7 Траектория процесса Y^2.

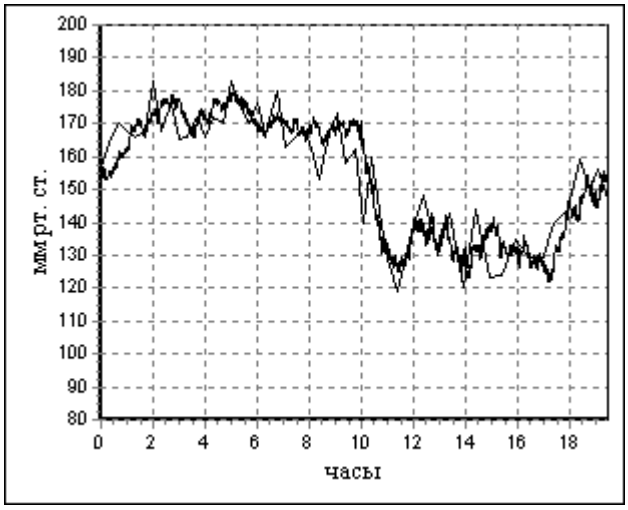


Рис. 8 Траектория процесса Y^3.

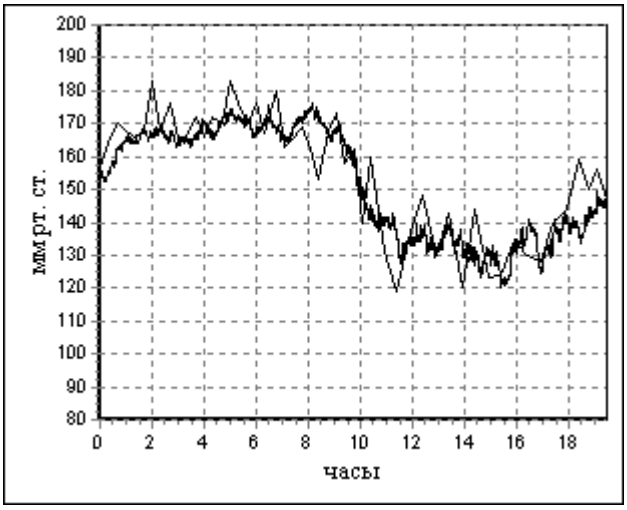


Рис. 9 Траектория процесса Y^4.

Значения параметров, общие для всех четырех моделей регуляции САД, приведены в таблице 6.

Таблица 6

X_0 = 0 нмоль/л	α_2 = 5
μ_1 = 3	Y_0 = 156.16 мм рт. ст.
μ_2 = 1.5	β_1 = 8 (мм рт. ст.)·л/нмоль
η_1 = 2 нмоль/л	β_2 = 4 (мм рт. ст.)·л/нмоль
η_2 = 1 нмоль/л	σ_1 = 0.035
α_1 = 10	σ_2 = 0.075

Параметры процессов Y^k, $k=\overline{1,4}$: $\gamma_1=1.027$, $\gamma_2=1.006$ (k = 1); $\gamma_1=1.026$, $\gamma_2=1.014$ (k = 2); $\gamma_1=1.027$, $\gamma_2=1.018$ (k = 3); $\gamma_1=1.028$, $\gamma_2=1.012$ (k = 4). В каждой из четырех моделей коэффициенты $\mu(t)$, $\eta(t)$, $\alpha(t)$, $\beta(t)$, $\gamma(t)$, $\sigma(t)$ могут быть найдены методами многомерной оптимизации (например, МНК).

С помощью компьютерного эксперимента был проведен анализ моделей регуляции САД. Динамика траекторий процессов Y^k, $k=\overline{1,4}$, (рис. 6–9) не имеет принципиальных различий за счет добавления к циркадианному ритму стохастических компонент. В связи с этим был сделан вывод о возможности применения в клинической практике любой из рассмотренных моделей динамики САД.

Выводы

Практическая значимость работы состоит в построении новых математических и компьютерных имитационных моделей циркадианного ритма САД. На их основе были построены математические и компьютерные имитационные модели регуляции САД в моменты стрессовых ситуаций.

Все рассмотренные в работе модели могут быть усложнены в зависимости от конкретной решаемой задачи.

Работа выполнена в рамках федеральной целевой программы «Научные и научно-педагогические кадры инновационной России» на 2009–2013, а также при поддержке Министерства образования и науки РФ в рамках постановления правительства РФ № 218.

Литература

1. Абакумов М.В. Методика математического моделирования сердечно-сосудистой системы / М.В. Абакумов, И.В. Ашметков, Н.Б. Есикова, В.Б. Кошелев, С.И. Мухин, Н.В. Соснин, В.Ф. Тишкин, А.П. Фаворский, А.Б. Хруленко // Математическое моделирование. 2000. Т. 12. № 2. С. 106–117.

2. McSharry P.E. Confronting a cardiovascular system model with heart rate and blood pressure data / P.E. McSharry, M.J. McGuinness, A.C. Fowler // Computers in Cardiology. 2005. Pp. 587–590.

3. Quarteroni A. What mathematics can do for the simulation of blood circulation / A. Quarteroni // Proceedings of the International Congress of Mathematicians. 2006. Pp. 110–144.

4. Подладчикова Т.В. Долгосрочное мониторирование и математическое моделирование хронобиологических изменений среднего артериального давления у различных возрастных групп / Т.В. Подладчикова, М.В. Рагульская, С.М. Чибисов, Д.Г. Стрелков // Успехи современного естествознания. 2008. № 2. С. 14–20.

5. Rao R.R. Multiple Model Predictive Control of Hemodynamic Variables: An Experimental Study / R.R. Rao, B. Aufderheide, B.W. Bequette // Proceedings of American Control Conference. 2-4 June, 1999. Vol. 2. Pp. 1253–1257.

6. Ahmed S.A. Multichannel Blind Deconvolution Using the Stochastic Calculus for the Estimation of the Central Arterial Pressure / S.A. Ahmed, M. El-S. Waheed, M.E. Nermeen // Math. Probl. Eng. 2010. Article ID 602373. 21 p.

7. Дильман В.М. Четыре модели медицины / В.М. Дильман. Ленинград: Медицина, 1987. 288 с.

8. Ширяев А.Н. Статистический последовательный анализ. Оптимальные правила остановки / А.Н. Ширяев. М.: Наука, 1976. 272 с.

ОПТИМАЛЬНЫЙ ВЫБОР МЕРОПРИЯТИЙ ПО СНИЖЕНИЮ РИСКА АВИАЦИОННЫХ ПРОИСШЕСТВИЙ

И.В. Круглова

Ульяновский государственный университет

Введение

Согласно статистике [1], Россия занимает второе место в мире по числу авиационных катастроф и погибших в них людей. Одной из мер по повышению безопасности полетов является автоматизация процедур прогнозирования авиапроисшествий и принятия управленческих решений по снижению авиационного риска. При прогнозировании авиационных происшествий и их риска часто используются диаграммы причинно-следственных связей – деревья событий [2]. Преимуществом такого подхода являются возможности проведения многофакторного анализа, оценки ситуации в реальном времени, оценки эффективности управленческого решения. Однако выбор оптимального управленческого решения (т. е. набора мероприятий по снижению риска происшествий) – вопрос, не достаточно изученный. Предметом данной статьи является выбор критерия оптимальности набора мероприятий по снижению риска авиационных происшествий (в стоимостной и вероятностной форме) и рекомендации по формированию этого набора.

Постановка задачи

Рассмотрим задачу выбора перечня рекомендуемых мероприятий, направленных на повышение безопасности полетов и предотвращение авиационных происшествий (АП).

Имеется база данных, которая содержит перечень мероприятий (например, «Ремонт двигателей», «Покупка новых двигателей», «Проведение дополнительной тренажерной подготовки летного состава», «Ввод в строй молодых пилотов и КВС» и т. д.) Каждое мероприятие снижает вероятности некоторых факторов опасности на определенный процент. При анализе мероприятий может использоваться следующая информация:

- Стоимость мероприятия C_i ($i \in \{1, \dots, M\}$, где M – число мероприятий)
- Показатели экономического ущерба от АП S_j, $j \in \{1,\dots,N\}$, где N – число происшествий
- Оценка эффективности управленческого решения (УР), т. е. степень его влияния на снижение вероятности возникновения авиационных происшествий P_j, $j \in \{1,\dots,N\}$.

Задачи о принятии УР обычно сводятся к оптимизационным задачам (для этого необходимо сформулировать критерий оптимальности набора мероприятий и задать условия, при которых набор мероприятий является допустимым). Для решения таких задач обычно используются методы математического программирования. Введем переменные, по которым осуществляется оптимизация в рассматриваемой задаче:

$$x_i = \begin{cases} 1, \textit{если выбрано мероприятие } i \\ 0, \textit{иначе} \end{cases},$$

$i \in \{1, \dots, M\}$, где M – число мероприятий.

Таким образом, задача об оптимальном УР – задача с булевыми переменными. Однако с определением целевой функции и условий допустимости, связаны некоторые трудности.

Критерии оптимальности:

1. Минимизировать стоимость риска+затраты на УР:

$$\min\left(\sum_{j=1}^{N} S_j P_{j,X} + \sum_{i=1}^{M} C_i x_i\right),$$

где $P_{j,X}$ – вероятность j-го АП при условии, что выбран набор мероприятий X, $X=\{i|x_i=1, i \in \{1\dots n\}\}$.

Недостатком такой постановки задачи является то, что она не учитывает следующий случай: при высокой стоимости мероприятий минимум этой функции может достигаться при

$X = \emptyset$. Однако снижение вероятности АП следует считать более приоритетной задачей, чем минимизацию расходов на мероприятия. Этот момент учитывается в следующей математической модели.

2. Минимизировать стоимость риска АП в рамках ограниченного бюджета (т. е. накладывается ограничение на стоимость УР):

$$min \sum_{j=1}^{N} S_j P_{j,X}$$, при условии что $\sum_{i=1}^{M} C_i x_i \leq Z.$

В данном случае необходимо обладать информацией о том, какие средства *Z* может выделить авиакомпания на проведение профилактических мероприятий.

3. Минимизировать расходы на УР при условии, что риск от происшествий должен быть снижен до определенного уровня.

$$min \sum_{i=1}^{M} C_i x_i$$, при условии что $\sum_{j=1}^{N} S_j P_{j,X} \leq R,$

где *R* – максимальное допустимое значение риска. Значение R может быть определено экспертами.

Недостаток вышеприведенных критериев заключается в том, что они не учитывают, какие происшествия влекут за собой высокий материальный ущерб, а какие низкий. В результате оптимальным может оказаться набор мероприятий, направленных в первую очередь на предотвращение АП с относительно низким значением ущерба (например, «столкновение с птицами», «столкновение с объектом на земле») ввиду высокой вероятности таких происшествий и низкой стоимости мероприятий по их предотвращению. Однако предпочтительнее снижать вероятности более опасных событий, таких как «столкновение воздушных судов в воздухе» или «пожар». В связи с этим можно использовать цветовые шкалы, которые обычно используются при оценивании вероятности АП. Такие шкалы организованы по светофорному принципу. Границы между зеленой и желтой зоной ($P_j^{з/ж}$) и между желтой и красной зоной ($P_j^{ж/к}$) определяются экспертами. Основываясь на этих шкалах, в первую очередь будем снижать вероятность событий, попавших в красную или желтую зону. Получаем следующие математические постановки задач:

4. Минимизировать расходы на УР при условии, что для тех событий, вероятность которых попала в красную зону, вероятность будет снижена до желтой зоны:

$$min \sum_{i=1}^{M} C_i x_i,$$

так чтобы $P_{j,X} < P_j^{ж/к}$, *j=1…N.*

5. Минимизировать расходы на УР при условии, что для тех событий, вероятность которых попала в красную или желтую зону, вероятность будет снижена до зеленой зоны:

$$min \sum_{i=1}^{M} C_i x_i,$$

так чтобы $P_{j,X} < P_j^{з/ж}$, *j=1…N.*

6. Минимизировать стоимостную оценку риска при условии, что для тех событий, вероятность которых попала в красную или желтую зону, вероятность будет снижена до зеленой зоны, а затраты на мероприятия ограничены бюджетом Z:

$$min \sum_{j=1}^{N} S_j P_{j,X},$$

так чтобы $P_{j,X} < P_j^{з/ж}$, *j=1…N* и $\sum_{i=1}^{M} C_i x_i \leq Z.$

7. Минимизировать расходы на УР при условии, что для тех событий, вероятность которых попала в красную или желтую зону, вероятность будет снижена до зеленой зоны, а суммарная стоимость риска будет снижена до определенного уровня R:

$$min \sum_{i=1}^{M} C_i x_i,$$

так чтобы $P_{j,X} < P_j^{э/эк}$, $j=1…N$ и $\sum_{j=1}^{N} S_j P_{j,X} \leq R$.

Постановки задач 6 и 7 содержат более строгие ограничения, но набор мероприятий, полученный в результате такой оптимизации, может получиться слишком объемным, требующим больших затрат денежных средств и времени. Поэтому целесообразно находить решения сразу нескольких задач и использовать полученные результаты для дальнейшего анализа.

Выбор мероприятий по предотвращению АП

В каждой из вышеприведенных математических постановок задачи используется вероятность АП, которая рассчитывается как функция от значений вероятностей исходных событий (или факторов опасностей) с помощью деревьев событий. Такая функция не является ни линейной, ни выпуклой. Точное решение рассматриваемой задачи оптимизации (в любой из ее постановок) можно найти методом перебора. Естественно, на практике такой подход не применим. Однако можно значительно сократить число мероприятий, по которым будет проводиться перебор, оставив только те мероприятия, которые снижают вероятности значимых факторов опасностей [2, с. 30-32].

Литература

1. http://www.aviasafety.ru/crash-stat
2. Белов П.Г. Моделирование опасных процессов в техносфере. – М: Издательство Академии гражданской защиты МЧС РФ, 1999. – 124 с.

ЗАДАЧА ОБ УПРАВЛЕНИИ МЕХАНИЧЕСКИМИ СИСТЕМАМИ. СИНТЕЗ НЕПРЕРЫВНОГО И КУСОЧНО-НЕПРЕРЫВНОГО УПРАВЛЕНИЯ[1]

Е.А.Кудашова

Ульяновский государственный университет

Аннотация

В силу конструктивных особенностей управляемой механической системы, информация между объектом управления и исполнительными органами может прерываться в дискретные моменты времени, откуда возникает необходимость обоснования дискретных моделей управления.

В данной статье рассматривается задача отыскания релейного управления, стабилизирующего невозмущенное движение механической системы с нестационарными, голономными, идеальными связями, положение которой определяется n обобщенными координатами $q^{'} = (q_1, q_2, ..., q_n)$.

Рассмотрим механическую систему, положение которой определяется n обобщенными координатами $q^{'} = (q_1, q_2, ..., q_n)$.

Кинетическая энергия системы представима в виде

$$\begin{gathered} T = T_2 + T_1 + T_0, \\ T_2(t,q,\dot{q}) = \tfrac{1}{2}\dot{q}^{'} A(t,q)\dot{q}, \\ T_1(t,q,\dot{q}) = B^{'}(t,q)\dot{q}, \quad T_0(t,q) = C(t,q), \end{gathered} \tag{1}$$

где $A(t,q)$ – матрица размерности $n \times n$ является положительно-определенной, $B(t,q)$ – матрица-столбец размерности $n \times 1$, $C(t,q)$ – скалярная функция, $(\ )^{'}$ – операция транспонирования. Предполагаем, что входящие в (4.1) функции переменных (t,q) определены и непрерывно-дифференцируемы в области $R^{+} \times R^{n}$.

При этом движение системы под действием управляющих сил U и других обобщенных сил Q описывается уравнениями Лагранжа,

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}}\right) - \frac{\partial T}{\partial q} = Q + U. \tag{2}$$

которые могут быть сведены к системе уравнений второго порядка

$$\begin{gathered} A(t,q)\ddot{q} = Q_1(t,q,\dot{q}) + U, \\ Q_1 = \frac{\partial T_0}{\partial q} + G^{'}\dot{q} - \frac{\partial B}{\partial t} - \frac{\partial A}{\partial t} + \left\{\dot{q}^{'} C \dot{q}\right\} + Q, \end{gathered} \tag{3}$$

Здесь $\left\{\dot{q}^{'} C \dot{q}\right\} = \left\{\dot{q}^{'} C_1 \dot{q}, \dot{q}^{'} C_2 \dot{q}, \ldots, \dot{q}^{'} C_n \dot{q}\right\}$ – n-мерный вектор соответственных квадратичных форм.

Пусть $X = \left\{\left(q^0(t), \dot{q}^0(t)\right) : [t, +\infty) \to R^n\right\}$ есть заданный спектр программных движений в виде ограниченных дважды непрерывно-дифференцируемых функций $q = q^0(t)$ с ограниченными производными при $t \in [t, +\infty)$.

[1] Исследование выполнено при поддержке Министерства образования и науки Российской Федерации, соглашение 14.B37.21.0373 «Развитие методов и алгоритмов исследования задач об управлении нелинейными механическими системами и компьютерное моделирование управляемого движения системы тел».

Пусть $\left(q_0(t),\dot{q}_0(t)\right)$ – какое-либо выбранное движение, реализуемое управлением $U=U^0(t)$, и согласно (3)

$$U^0(t)=A\left(t,q^0(t)\right)\ddot{q}^0(t)-Q_1\left(t,q^0(t),\dot{q}^0(t)\right). \tag{4}$$

Введем возмущения $x=q-q_0(t)$ и управляющие воздействия $U\left(t,x,\dot{x}\right)=U-U^0(t)$, формируемые на основе обратной связи, а так же вспомогательные функции $R\left(t,x\right)$ и Q_1 которые определяются изменением составляющих параметров и внешних сил системы на возмущенном движении $\left(x(t),\dot{x}(t)\right)$:

$$R\left(t,0\right)=0,\quad Q_1^x\left(t,x,\dot{x}\right)=Q\left(t,q^0(t)+x,\dot{q}^0(t)+\dot{x}\right)-Q\left(t,q^0(t),\dot{q}^0(t)\right)=$$
$$=Q_x\left(t,x\right)+Q_1\left(t,x\right)\dot{x}+\left\{\dot{x}^{'}Q_2^x\left(t,x\right)\dot{x}\right\}+Q_3^x\left(t,x,\dot{x}\right).$$

Выразим эти уравнения относительно $\ddot{x}$:

$$\ddot{x}=A_1^{-1}\left(\left\{\dot{x}^{'}S_1\dot{x}\right\}+D_1\dot{x}+G_1\dot{x}+R_x f+Q_1\dot{x}+\left\{\dot{x}^{'}Q_2\dot{x}\right\}+Q_3+U+\mu\right). \tag{5}$$

Согласно [1] можно построить непрерывное управляющее воздействие U, обеспечивающее стабилизацию невозмущенного движения $x=\dot{x}=0$ системы (5), в виде

$$U=\Psi\left(t,H(t)\dot{x}+f\left(t,x\right)\right), \tag{6}$$

где $\Psi\left(t,y\right)$ – есть некоторая вектор-функция, определяющая усиление в структуре обратной связи, $\Psi\left(t,0\right)=0,\ \Psi\in C^1$, H – матрица, $H\in R^{n\times n}$.

При невыполнении условия (4), задача о стабилизации движения $\left(q_0(t),\dot{q}_0(t)\right)$ может быть решена на основе релейного управления. Такой вид управления может быть эффективно использован и при ограничениях на управления.

Будем искать управляющее воздействие $U_1\left(t,x,\dot{x}\right)$ среди кусочно-непрерывных управлений, разрывных на поверхности вида

$$\dot{x}_i+\psi_i\left(t,x\right)=0,\quad \psi_i\left(t,0\right)=0\quad \left(i=1,\ldots,n\right),$$

где функции $\psi_i\in C^1\left(R^+\times D\right)$, при этом производные $\frac{\partial\psi_i}{\partial t},\frac{\partial\psi_i}{\partial x_j}\left(i,j=1,\ldots,n\right)$, являются ограниченными. Тем самым выполнены условия существования и единственности решения уравнения

$$\dot{x}=-\Psi\left(t,x\right),\quad \Psi=\left(\psi_1,\psi_2,\ldots,\psi_n\right)^{'}. \tag{7}$$

При этом будем предполагать, что решение $x=0$ системы (7) является равномерно асимптотически устойчивым с некоторой областью равномерного притяжения $\left\{\left\|q\right\|\le\Delta>0\right\}$.

Замечание 1 Можно полагать, что для каждого $\left(q^0(t),\dot{q}^0(t)\right)\in X$ строится отдельное управляющее воздействие $U=U^1\left(t,x,\dot{x}\right)$ со своей поверхностью разрыва $\dot{x}=-\Psi\left(t,x\right)$, а можно принять, что весь класс управлений U формируется в виде зависимостей $u=u\left(t,q-q^0(t),\dot{q}-\dot{q}^0(t)\right)$, разрывных на поверхностях $\dot{q}=\dot{q}^0(t)-\Psi\left(t,q-q^0(t)\right)$ с заданным законом разрывности $\dot{x}=-\Psi\left(t,x\right)$. На основании работы [2] получен следующий результат.

Результат 2 Пусть $\left(q_0(t),\dot{q}_0(t)\right)$ есть программное движение системы (2). Тогда управляющее воздействие

$$U=B\Delta\left(\dot{x}+\Psi\left(t,x\right)\right),\ \Delta^{'}=\left(sign\left(\left(\dot{x}+\Psi\left(t,x\right)\right)_1\right),\ldots,sign\left(\left(\dot{x}+\Psi\left(t,x\right)\right)_n\right)\right) \tag{8}$$

решает задачу о стабилизации этого программного движения. В частности, при управляющем воздействии

$$U^1 = (U_1,\ldots,U_n)', \quad U_i = -\mu \cdot sign(\dot{x}_i + \psi_i(t,x)), \quad (i = 1,\ldots,n),$$

где число μ удовлетворяет неравенству

$$\mu \geq 2\left(\left(a_{10} + na_{11}\Delta\right)\Delta^2 + na_0\left(\psi_{10} + \psi_{11}\Delta\right) + q_0\right),$$

$a_{10}, a_{11}, a_0, \psi_{10}, \psi_{11}, q_0$ есть нормы соответствующих векторов, матриц и семейств матриц $\frac{\partial A_1}{\partial t}, \left\{\frac{\partial A_1}{\partial x}\right\}, \frac{\partial \Psi}{\partial t}, \frac{\partial \Psi}{\partial x}, A_1, Q_2$.

Дискретные модели управления

Рассмотрим реализацию управлений (6) и (8) в виде дискретной модели.

Первый тип. Шаговое управление

Под таким управлением понимают управление, которое может принимать дискретные значения $p_k \in \square$, $h_k > 0$ - шаг задержки, $k = \overline{1,n}$.

Алгоритм построения такого управления состоит в следующем:

1. Задаем некоторый набор достаточно малых положительных значений $(\Delta_1, \Delta_2, ..., \Delta_n)$.

2. Для начального момента $t_0 \geq 0$ и начального положения системы $x_0 = q_0 - q(t_0)$ определяем начальные значения $\gamma_1^0, \gamma_2^0, ..., \gamma_n^0$ из неравенства

$$\left|U_k^g(0) - U_k(t_0, x_0)\right| \leq \frac{h_k}{2} + \Delta_k$$

3. Последующие дискретные значения времени $t_l \geq t_0, l = \overline{1,n}$ задаем из условия нарушения неравенства

$$\left|U_k^g(l-1) - U_k(t_l, \tilde{x}(t_l, t_0, x_0))\right| \leq \frac{h_k}{2} + \Delta_k,$$

Где $\tilde{x}(t_l, t_0, x_0)$ - движение системы.

4. Соответствующие значения дискретного времени определяем посредству равенств

5. $U_k^g(l) = U_k^g(l-1) + h_k \operatorname{sgn}(U_k[t_l] - U_k^g(l-1))$

Таким образом, пошаговое управление задается в виде

$$U_k(t) = U_k^g(l-1), \quad t_{l-1} \leq t < t_l \tag{9}$$

Основанием для того, чтобы это управление обеспечивало стабилизацию программного движения $(q^0(t), \dot{q}^0(t))$ системы (2) является следующее утверждение.

Утверждение 3 Существует такой набор шагов h_k задания обобщенных управлений $U_k^g = p_k h_k$, $k = \overline{1,n}$, при которых это управление обеспечивает асимптотическую устойчивость невозмущенного движения $\dot{x} = x = 0$ системы (8), или соответствующего программного движения $(q^0(t), \dot{q}^0(t))$.

Доказательство утверждения проводится с использованием той же функции Ляпунова, что и при выводе утверждения 1. Различие состоит в том, что используется не соответствующая общая теорема из [1], а непосредственно теорема Ляпунова об асимптотической устойчивости.

Второй тип управления. Управление, дискретное по времени с промежутком h

Для базовых управлений (9) и (8) можно определить две модели управления

$$\begin{aligned} U_l^{g1} &= U(t,x,\dot{x}): \quad x = x(t_l),\ \dot{x} = \dot{x}(t_l), \quad t_l = lh \\ U_l^{g2} &= U(t,x,\dot{x}): \quad x = x(t_l),\ \dot{x} = \dot{x}(t_l), \quad t_l = lh \end{aligned} \tag{10}$$

Первая модель управления представляет собой кусочно-непрерывную по t функцию; вторая- кусочно-постоянную функцию.

Используя ту же функцию Ляпунова, что и ранее, можно доказать схождение утверждения.

Утверждение 4 Существует число $h > 0$, такое, что управление (10) стабилизирует невозмущенное движение по $\dot{x} = x = 0$ системы (5) или заданное программное движение $(q^0(t), \dot{q}^0(t))$ системы (2).

Заключение

Исследована задача о стабилизации положения равновесия голономной механической системы с нестационарными, голономными, идеальными связями, положение которой определяется n обобщенными координатами. Обоснованы два новых типа дискретных управлений, обеспечивающих асимптотическую устойчивость программного движения рассматриваемой системы. На этой основе проведено аналитическое и численное решение задачи об управлении колесными системами.

Литература

1. Андреев А.С., Бойкова Т.А. Об устойчивости неустановившегося движения механической системы // ПММ, 2004 – Т.68, вып. 4. – С. 678-686.

2. Андреев А.С., Дмитриева О.Г, Петровичева Ю.В. Об устойчивости нулевого решения системы с разрывной правой частью // Научно-технический вестник Поволжья, 2011. – № 1. – С. 15-21.

ЛИЕВСКОЕ МНОГООБРАЗИЕ ДРОБНОЙ ЭКСПОНЕНТЫ

О.А.Малюшева

Ульяновский государственный университет

В теории многообразий линейных алгебр важнейшую роль играют их числовые характеристики. Например, такая характеристика как экспонента многообразия.

Цель работы: исследование экспоненты многообразий алгебр Ли над полем характеристики нуль и подсчет её приближенного значения.

Задача работы: построить новый пример лиевского многообразия дробной экспоненты.

Методы работы: для решения поставленной задачи использовалась техника диаграмм Юнга, теория представления симметрических групп, а также методы математического анализа.

Пусть V – многообразие линейных алгебр, а $F(V)$ – его относительно свободная алгебра счетного ранга, порожденная элементами $x_1, x_2, \ldots$. Обозначим через $P_n(V)$ подпространство полилинейных элементов от $x_1, \ldots, x_n$ в $F(V)$, а через $c_n(V) = dim\ P_n(V)$ – его размерность. Рост числовой последовательности $c_n(V)$ называют ростом многообразия V. Если последовательность $c_n(V)$ мажорируется экспонентой a^n для подходящего a, то существуют пределы

$$\mathrm{LEXP}(V) = \lim_{n\to\infty} \inf \sqrt[n]{c_n(V)}, \quad \mathrm{HEXP}(V) = \lim_{n\to\infty} \sup \sqrt[n]{c_n(V)},$$

которые называют нижней и верхней экспонентой многообразия V.

В случае ассоциативных алгебр любое многообразие имеет рост не выше экспоненциального [1] и более того его экспонента является натуральным числом [2]. В общем случае, как доказано в работе [3], для любого действительного $\alpha > 1$ существует такое многообразие V_α, что $\mathrm{EXP}(V_\alpha) = \alpha$.

В работе [4] был построен первый пример многообразия алгебр Ли с дробной экспонентой, а в статье [5] указано точное ее значение. В данной работе, используя аналогичные конструкции алгебр Ли, построен новый пример их многообразия с дробной экспонентой. Все неопределяемые понятия можно найти в монографиях [6] и [7].

Актуальность и научная новизна работы в том, что построенный пример является уникальным. В перспективе планируется обобщить уже имеющиеся примеры на общий случай семейства многообразий алгебр Ли с дробной экспонентой.

Договоримся опускать скобки в случае левонормированной записи произведений в алгебрах Ли, то есть, $(ab)c \equiv abc$. Будем использовать черту или волну над образующими для обозначения альтернирования. Заглавными латинскими буквами обозначим внутреннее дифференцирование алгебры, то есть $ad_y(x) = xY = xy$. Например,

$$y_1\overline{X_1[X_2,Y]} = 2(y_1x_1x_2y + y_1x_2yx_1 + y_1yx_1x_2 - y_1x_1yx_2 - y_1yx_2x_1 - y_1x_2x_1y).$$

$$\overline{X_1[X_2,X_3]}\big[[X_4,X_5],Y\big] = \sum_{p\in S_5} (-1)^p X_{p(1)}[X_{p(2)},X_{p(3)}]\Big[\big[X_{p(4)},X_{p(5)}\big],Y\Big],$$

где S_n – симметрическая группа, $(-1)^p$ – четность перестановки $p\in S_n$.

Пусть A^2 многообразие алгебр Ли, определяемое тождеством $(x_1x_2)(x_3x_4) \equiv 0$, а $M = F_4(A^2)$ является относительно свободной алгеброй этого многообразия с множеством свободных образующих $\{z_1,z_2,z_3,z_4\}$. Рассмотрим линейное преобразование d четырехмерного векторного пространства $\langle z_1,z_2,z_3,z_4\rangle$, определенное правилом $z_1d=z_2$, $z_2d=z_3$, $z_3d=z_4$, $z_4d=z_1$. В этом случае d может быть продолжено до дифференцирования алгебры M. Рассмотрим $D = \langle d\rangle$ – одномерную алгебру Ли с нулевым умножением. Можно построить полупрямое произведение $L = M \leftthreetimes D$ алгебр M и D. Многообразие, порожденное алгеброй L, обозначим через $V = var(L)$. Характер S_n – модуля $P_n(V)$ раскладывается в целочисленную комбинацию неприводимых характеров

$$\chi(P_n(V)) = \sum_{\lambda \vdash n} m_\lambda \chi_\lambda, \quad (1)$$

где m_λ – кратность, а χ_λ – характер неприводимого представления, соответствующего диаграмме Юнга, построенной по разбиению $\lambda = (\lambda_1, \dots, \lambda_k)$ числа n. Размерность неприводимого модуля, соответствующего разбиению $\lambda = (\lambda_1, \dots, \lambda_k)$ числа n, обозначим через d_λ. Тогда выполняется соотношение $c_n(V) = dim\ P_n(V) = \sum_{\lambda \vdash n} m_\lambda d_\lambda$.

Теорема. В случае поля нулевой характеристики для многообразия алгебр Ли $V = var(L)$ выполняется равенство $LEXP(V) = HEXP(V) \approx 3.83$.

Для доказательства теоремы потребуется следующее утверждение.

Лемма. В сумме (1) кратность $m_\lambda \neq 0$ может быть только в случае, когда разбиение $\lambda = (\lambda_1, \dots, \lambda_k)$ числа n соответствует диаграмме, имеющей вне первых пяти строк не более двух клеток, а длины строк удовлетворяют такому неравенству $\lambda_1 - \lambda_3 - 2\lambda_4 - 3\lambda_5 + 12 \geq 0$.

Доказательство.

Предположим противное. Пусть разбиение λ числа n такое, что в соответствующей диаграмме Юнга вне первых пяти строк расположено более двух клеток. Тогда рассмотрим произвольный элемент f, который порождает неприводимый модуль, соответствующий λ. Обозначим длины столбцов этого разбиения как $\lambda_1', \dots, \lambda_{l(\lambda)}'$.

Как было показано в работе [7] элемент f равен линейной комбинации слагаемых, каждое из которых содержит $l(\lambda)$ кососимметричных наборов с λ_i' переменными в i-ом наборе. Покажем, что в L тождественно равен нулю любой полилинейный лиевский полином, который либо кососимметричный по семи своим переменным, либо содержит два различных кососимметричных набора, каждый из которых длины 6. Для этого заметим, что алгебра L содержит абелев идеал M^2 коразмерности 5.

Рассмотрим первый случай. Полилинейный полином кососимметричный по семи своим переменным. Достаточно проверить его тождественность нулю на базисных элементах. Так как набор кососимметричный по семи переменным, а базисных элементов, не принадлежащих M^2, только пять, а именно: $\{d, z_1, z_2, z_3, z_4\}$, то придется подставить два элемента из M^2, а в этом идеале произведение любых двух элементов равно нулю.

Рассмотрим второй случай. Полилинейный полином содержит два различных кососимметричных набора длины шесть каждый. Тогда после подстановки базисных элементов, как и в предыдущем случае, вместо образующих каждого набора будет подставлен как минимум один элемент из идеала M^2. Следовательно, полином, соответствующий такому разбиению опять тождественно равен нулю. Итак, кратность $m_\lambda = 0$ для диаграммы с двумя и более клетками вне первых пяти строк.

Докажем второе утверждение о соотношении размеров строк диаграммы. Рассмотрим разбиение $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \dots)$ числа n, у которого $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 \geq n - 1$. Предположим обратное неравенство $\lambda_1 - \lambda_3 - 2\lambda_4 - 3\lambda_5 + 12 < 0$. Покажем, что такая диаграмма также задает тождество в алгебре L. Для этого достаточно убедиться в том, что любой полилинейный полином, зависящий от $l = \lambda_1$ кососимметричных наборов переменных, длины которых равны $\lambda_1', \dots, \lambda_l'$ соответственно, принимает только нулевые значения.

Пусть $d, z_1, z_2, \dots$ – базис алгебры L. Подставим некоторые из них в f. Элемент d можно подставить вместо не более одной переменной каждого кососимметричного набора, иначе получим нуль. Отождествим и обозначим буквой b те переменные в f, вместо которых мы подставляем d, а остальные переменные мы обозначим $y_1, \dots, y_k$. Продифференцируем элементом d необходимое количество раз. Элемент f перепишется в виде линейной комбинацию произведений:

$$(y_{s_1} b^{\alpha_1})(y_{s_2} b^{\alpha_2}) \dots (y_{s_k} b^{\alpha_k}), \quad (2)$$

у которых $\alpha_1, \alpha_2, \dots, \alpha_k \geq 0$. Причем $\alpha_1 + \dots + \alpha_k \leq \lambda_1$.

По новым переменным $y_{s_i} b^{\alpha_i}$ полином f уже не будет полилинеен, однако, его можно записать как сумму $f = f_1 + \dots + f_m$, в которой каждое слагаемое – полилинейный полином от части новых образующих. Если f кососимметричен по y_1 и y_2, тогда те из $f_1, \dots, f_m$, которые

зависят от y_1b, y_2b тоже по ним кососимметричны. Также сохраняется антисимметрия по y_1b^2, y_2b^2 и по y_1b^3, y_2b^3.

Докажем, что каждое слагаемое $f_1, \dots, f_m$ принимает нулевое значение. Рассмотрим, например, f_1. Зафиксируем индексы s_1, s_2 и покажем, что частичная сумма f_1' для f_1 с этими s_1, s_2 уже равна нулю. Пусть f был кососимметричен по y_1, y_2, y_3, y_4, причем $1,2,3,4 \neq s_1, s_2$. Если f_1' зависит одновременно или от y_1, y_2, или от y_1b, y_2b, или от y_1b^2, y_2b^2, или от y_1b^3, y_2b^3, то он равен нулю. Так как есть антисимметричность и все y_i принимают значения в метабелевой алгебре M. То есть, если f_1' зависит от $y_1b^{\alpha_1}, y_2b^{\alpha_2}, y_3b^{\alpha_3}, y_4b^{\alpha_4}$ и принимает ненулевое значение, то $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 \geq 6$, аналогично, если f_1' кососимметричен по переменным y_1, y_2, y_3 и принимает ненулевое значение, то $\alpha_1 + \alpha_2 + \alpha_3 \geq 3$. А для двух переменных – $\alpha_1 + \alpha_2 \geq 1$. Исходный полином f зависит от λ_5 кососимметричных наборов длины не менее 5, $\lambda_4 - \lambda_5$ кососимметричных наборов длины 4, $\lambda_3 - \lambda_4$ кососимметричных наборов длины 3. Переменные y_{s_1}, y_{s_2} входят не более чем в два из этих наборов, поэтому остается как минимум $\lambda_5 - 2$ кососимметричных набора длины 5, $\lambda_4 - \lambda_5$ кососимметричных наборов длины 4, $\lambda_3 - \lambda_4$ кососимметричных наборов длины 3. Только одну переменную из каждого набора можно заменить на d. При выражении f_1' через (2) этот элемент будет включать не менее $\lambda_5 - 2$ четверных, $\lambda_4 - \lambda_5$ тройных, $\lambda_3 - \lambda_4$ двойных кососимметричных наборов, а принимает он значение не равное нулю при условии $\alpha_1 + \dots + \alpha_k \geq 6(\lambda_5 - 2) + 3(\lambda_4 - \lambda_5) + (\lambda_3 - \lambda_4) = 3\lambda_5 + 2\lambda_4 + \lambda_3 - 12$.

Но $\alpha_1 + \dots + \alpha_k \leq \lambda_1$, откуда следует неравенство $\lambda_1 - \lambda_3 - 2\lambda_4 - 3\lambda_5 + 12 \geq 0$. Значит при $\lambda_1 - \lambda_3 - 2\lambda_4 - 3\lambda_5 + 12 < 0$ элемент f_1', а также f_1 и f принимают нулевые значения.

Лемма доказана.

Доказательство теоремы.

Из леммы следует, что если вне первых пяти строк диаграммы Юнга содержится более двух клеток либо $\lambda_1 - \lambda_3 - 2\lambda_4 - 3\lambda_5 + 12 < 0$, то кратность m_λ равна нулю. В частности, в многообразии выполнена система тождеств Капелли. Как доказано в работе [8], в этом случае кратности m_λ полиномиально ограничены. Понятно, что в случае выполнения системы тождеств Капелли количество слагаемых в сумме (1) также полиномиально ограничено. Поэтому верхнюю и нижнюю оценку экспонент можно находить, анализируя размерности неприводимых модулей симметрической группы.

Учитывая, что ограниченное количество клеток не влияет на числовые значения верхней и нижней экспонент многообразия, рассмотрим разбиения на не более, чем пять частей.

Для $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$ разбиения числа n определим числа $\alpha_i = \frac{\lambda_i}{n}$, $i=1,\dots,5$. Некоторые из последних λ_i могут быть равны нулю. Определим функцию $F(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = \frac{1}{\prod_{i=1}^{5} \alpha_i^{\alpha_i}}$, предполагая, что $\alpha_i^{\alpha_i} = 1$ в случае, когда $\alpha_i = 0$. Для любого натурального t рассмотрим разбиение $\lambda(t) = (\alpha_1 nt, \alpha_2 nt, \alpha_3 nt, \alpha_4 nt, \alpha_5 nt)$. Отметим, что $\lambda(1) = \lambda$. Пусть $d_{\lambda(t)}$ – размерность соответствующего разбиению $\lambda(t) \vdash nt$ модуля симметрической группы S_{nt}.

Используя формулу крюков для размерностей неприводимых представлений симметрической группы и формулу Стирлинга, получаем, что $\lim_{t\to\infty} \sqrt[nt]{d_{\lambda(t)}} = F(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$.

Определим область T_n пятимерного арифметического пространства следующим условием: точка $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$ принадлежит множеству T_n тогда и только тогда, когда существует такое разбиение $\lambda \vdash n$, что $m_\lambda \neq 0$ в сумме (1), где $\alpha_s = \frac{\lambda_s}{n}$, $s = 1, 2, \dots$. Заметим, что разбиение $\lambda \vdash n$ может состоять из более чем пяти частей.

Пусть T – область пятимерного пространства, определенная условиями

$$\begin{cases} \alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5=1 \\ \alpha_1-\alpha_3-2\alpha_4-3\alpha_5\geq 0 \\ \alpha_1\geq\alpha_2\geq\alpha_3\geq\alpha_4\geq\alpha_5\geq 0. \end{cases} \quad (3)$$

Во втором условии мы не учитываем слагаемое 12, так как оно не влияет на максимальное значение функции.

Так как функция $F(\alpha)$ непрерывна, то она достигает на компакте T своего максимального значения в некоторой точке $\alpha^{(0)} \in T$, $F_{\max} = F(\alpha^{(0)}) = \max_{\alpha\in T} F(\alpha)$. Из леммы получаем, что для любого $\varepsilon > 0$ существует такое натуральное число N, что множество T_n содержится в ε – окрестности компакта T. Отсюда, а также из того, что число слагаемых и кратности в сумме (1) полиномиально ограничено, получаем, что $\mathrm{HEXP}(V) \leq F_{\max}$.

Для доказательства неравенства $\mathrm{LEXP}(V) \geq F_{\max}$ достаточно доказать, что существует последовательность $\alpha^{(s)}$, s=1,2,… , такая, что $\lim_{s\to\infty}\alpha^{(s)} = \alpha^{(0)}$, $\lim_{s\to\infty} F(\alpha^{(s)}) = F_{\max}$, причем $m_{\lambda^{(s)}(t)} \neq 0$ в разложении (1) для любых натуральных s и t. Покажем, что таким свойством обладает произвольная точка множества T с рациональными $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5$.

Пусть $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5)$ некоторая произвольная рациональная точка множества T, $\lambda \vdash n$, $\lambda = (\alpha_1 n, \alpha_2 n, \alpha_3 n, \alpha_4 n, \alpha_5 n)$, где n – общий знаменатель рациональных чисел $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5$. Для любого натурального t определим разбиение $\lambda(t) = (\alpha_1 nt, \alpha_2 nt, \alpha_3 nt, \alpha_4 nt, \alpha_5 nt) \vdash nt$. Пусть $x_1, x_2, x_3, x_4, x_5, x_{01}$ и x_{02} – свободные образующие относительно свободной алгебры $F(V)$. Напоминаем, что заглавная буква обозначает соответствующее внутреннее дифференцирование алгебры.

Пусть $R_1 = X_{01}$, $R_2 = [\overline{X_1}, \overline{X_{01}}]$, $R_3 = [\overline{X_1}, \overline{X_{01}}]\,\overline{X_2}$, $R_4 = [[\overline{X_3}, X_{01}], X_{01}][\overline{X_1}, \overline{X_{01}}]\,\overline{X_2}$, $R_5 = [[[\overline{X_4}, X_{01}], X_{01}], X_{01}][[\overline{X_3}, X_{01}], X_{01}][\overline{X_1}, \overline{X_{01}}]\,\overline{X_2}$.

Рассмотрим элемент относительно свободной алгебры $F(V)$:

$$g_t = x_{02} R_1^{\alpha_1 nt-\alpha_3 nt-2\alpha_4 nt-3\alpha_5 nt} R_2^{\alpha_2 nt-\alpha_3 nt} R_3^{\alpha_3 nt-\alpha_4 nt} R_4^{\alpha_4 nt-\alpha_5 nt} R_5^{\alpha_5 nt}.$$

Степень g_t равна $m = nt+1$. Пусть f_t – полная линеаризация элемента g_t, а $R_t = \Phi S_{nt+1} f$ – подмодуль в $P_{n+1}(V)$, порожденный элементом f. Элемент g_t содержит $\alpha_5 nt$ альтернированных наборов по 5 переменных $\{x_{01}, x_1, x_2, x_3, x_4\}$ в каждом, $(\alpha_4 - \alpha_5)nt$ альтернированных наборов по 4 переменных $\{x_{01}, x_1, x_2, x_3\}$ в каждом, $(\alpha_3 - \alpha_4)nt$ альтернированных наборов по 3 переменных $\{x_{01}, x_1, x_2\}$ в каждом и $(\alpha_2 - \alpha_3)nt$ альтернированных наборов по 2 переменных $\{x_{01}, x_1\}$ в каждом. Все остальные переменные, кроме x_{02}, не входящие в альтернированные наборы, совпадают с x_{01}. Поэтому при разложении модуля R_t в прямую сумму неприводимых слагаемых возникают лишь модули, соответствующие диаграммам Юнга, которые содержат поддиаграмму, соответствующую разбиению $\lambda(t) \vdash nt$.

Докажем, что по крайней мере один из таких неприводимых модулей не равен нулю в полилинейной части $P_{nt+1}(V)$. Для этого рассмотрим такие элементы $h_s = x_{02} R_s$, $s=2,3,4,5$. Сделаем в h_2, h_3, h_4, h_5 следующую подстановку: $x_{02} = z_2 Z_1^{x_{01}}$, $x_1 = z_4$, $x_2 = z_1$, $x_3 = z_2$, $x_4 = z_3$, $x_{01} = d$. Если два элемента z_i, z_j в процессе суммирования одновременно попадают в коммутаторную скобку, то такое слагаемое равно нулю, так как M является метабелевым идеалом алгебры L. Таким образом, результат подстановки не равен нулю, так как в нем, например, присутствует такой ненулевой базисный элемент алгебры M, который ни с чем не сокращается: $z_2 z_1^{t_s}$, где $t_s = t + s - 1$. Другими словами, h_s при такой подстановке переходит в $\mu z_2 z_1^{t_s} + \ldots$, где μ – ненулевое целое число, а многоточием обозначена комбинация слагаемых вида $z_2 z_1^{t}$ с $t < t_s$ и базисных одночленов M, отличных от $z_2 z_1^{t}$.

Теперь понятно, что если в элемент g_t сделать такую подстановку элементов алгебры L: $x_{02} = z_2$, $x_1 = z_4$, $x_2 = z_1$, $x_3 = z_2$, $x_4 = z_3$, $x_{01} = d$, то результат подстановки будет не равен нулю, так как будет, например, содержать базисный элемент $z_2 z_1^{(\alpha_1+\alpha_3+\alpha_4-\alpha_5)nt}$, который ни с чем не сокращается.

Из полученных неравенств получаем, что $\mathrm{LEXP}(V) = \mathrm{HEXP}(V) = F_{\max}$.

Вычислим максимум функции $F(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5) = \frac{1}{\prod_{i=1}^{5}\alpha_i^{\alpha_i}}$ на области T при условиях (3).

В работе [5] был найден максимум функции $F(\gamma_1,\gamma_2,\gamma_3,\gamma_4) = \frac{1}{\prod_{i=1}^{4}\gamma_i^{\gamma_i}}$ при условиях:

$$\begin{cases} \gamma_1+\gamma_2+\gamma_3+\gamma_4=1 \\ \gamma_1-\gamma_3-2\gamma_4\geq 0 \\ \gamma_1\geq\gamma_2\geq\gamma_3\geq\gamma_4\geq 0. \end{cases}$$

Он оказался приблизительно равным 3.61. Сделаем замену переменных. $\gamma_0' = \frac{1}{2}$, $\gamma_1' = \frac{\gamma_1}{2}$, $\gamma_2' = \frac{\gamma_2}{2}$, $\gamma_3' = \frac{\gamma_3}{2}$, $\gamma_4' = \frac{\gamma_4}{2}$. Тогда $F(\gamma_0',\gamma_1',\gamma_2',\gamma_3',\gamma_4') = \frac{1}{(\frac{1}{2})^{\frac{1}{2}}(\frac{\gamma_1}{2})^{\frac{\gamma_1}{2}}(\frac{\gamma_2}{2})^{\frac{\gamma_2}{2}}(\frac{\gamma_3}{2})^{\frac{\gamma_3}{2}}(\frac{\gamma_4}{2})^{\frac{\gamma_4}{2}}}$. Причем $\gamma_0' + \gamma_1' + \gamma_2' + \gamma_3' + \gamma_4' = 1$. После преобразований $F(\gamma_0', \gamma_1', \gamma_2', \gamma_3', \gamma_4') = \frac{2}{(\prod_{i=1}^{4}\gamma_i^{\gamma_i})^{\frac{1}{2}}} = 2\sqrt{F(\gamma_1,\gamma_2,\gamma_3,\gamma_4)}$.

То есть $F(\gamma_0', \gamma_1', \gamma_2', \gamma_3', \gamma_4') > F(\gamma_1, \gamma_2, \gamma_3, \gamma_4)$. Но это возможно, только если добавленная таким образом переменная больше нуля. Поэтому можно считать, что $\alpha_5 > 0$.

Приведем задачу к стандартной форме: $\alpha_1^{-\alpha_1}\alpha_2^{-\alpha_2}\alpha_3^{-\alpha_3}\alpha_4^{-\alpha_4}\alpha_5^{-\alpha_5}$ достигает максимального значения тогда и только тогда, когда достигает максимального значения функция $-(\alpha_1\ln\alpha_1+\alpha_2\ln\alpha_2+\alpha_3\ln\alpha_3+\alpha_4\ln\alpha_4+\alpha_5\ln\alpha_5)$. Полагаем, что $0^0 = \lim_{u\to 0} exp^{u\ln u} = \lim_{u\to 0} u^u = 1$. Ограничения (3) задают компакт внутри $[0,1]^5 \subset R^5$. Функция $-(\alpha_1\ln\alpha_1+\alpha_2\ln\alpha_2+\alpha_3\ln\alpha_3+\alpha_4\ln\alpha_4+\alpha_5\ln\alpha_5)$ с учетом доопределения, такого, что

$$0\ln 0 = lim_{u\to 0+}\, u\ln u = lim_{u\to 0+}\frac{\ln u}{\frac{1}{u}} = lim_{u\to 0+}\frac{1/u}{-1/u^2} = 0,$$

представляет собой непрерывную функцию на компакте. Следовательно, она принимает наибольшее и наименьшее значение в этом множестве. Используя теорию функций Лагранжа, приводим ограничения к виду:

$$\begin{cases} \alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5=1 \\ \alpha_1-\alpha_3-2\alpha_4-3\alpha_5\geq 0 & (4.0) \\ \alpha_1-\alpha_2\geq 0 & (4.1) \\ \alpha_2-\alpha_3\geq 0 & (4.2) \\ \alpha_{31}-\alpha_4\geq 0 & (4.3) \\ \alpha_4-\alpha_5\geq 0 & (4.4) \\ \alpha_5\geq 0 & (4.5) \end{cases}$$

Функция Лагранжа имеет вид:

$$L(\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5,\beta_0,\beta_1,\beta_2,\beta_3,\beta_4,\beta_5,\mu) =$$
$$= -(\alpha_1\ln\alpha_1+\alpha_2\ln\alpha_2+\alpha_3\ln\alpha_3+\alpha_4\ln\alpha_4+\alpha_5\ln\alpha_5)+\beta_0(\alpha_1-\alpha_3-2\alpha_4-3\alpha_5)+$$
$$\beta_1(\alpha_1-\alpha_2)+\beta_2(\alpha_2-\alpha_3)+\beta_3(\alpha_3-\alpha_4)+\beta_4(\alpha_4-\alpha_5)+\beta_5(\alpha_5)+$$
$$+\mu(1-\alpha_1-\alpha_2-\alpha_3-\alpha_4-\alpha_5),$$

где β_i, μ – некоторые рациональные коэффициенты. Коэффициент β_i, соответствующий строгому неравенству, считается равным нулю. Так как неравенство (4.5) всегда строгое (как было показано выше), то $\beta_5 = 0$. Таким образом, $\alpha_1\geq\alpha_2\geq\alpha_3\geq\alpha_4\geq\alpha_5>0$, то есть все они строго положительны. Заметим, что $1 = \alpha_1+\alpha_2+\alpha_3+\alpha_4+\alpha_5\geq 5\alpha_5$. Условие $\alpha_5\in[0,\frac{1}{5}]$ является необходимым для ограничений (3). С учетом сказанного, запишем условия Лагранжа:

$$\begin{cases} \frac{\partial L(\alpha,\beta,\mu)}{\partial \alpha_1} = -\ln\alpha_1 - 1 + \beta_0 + \beta_1 - \mu = 0 \\ \frac{\partial L(\alpha,\beta,\mu)}{\partial \alpha_2} = -\ln\alpha_2 - 1 - \beta_1 + \beta_2 - \mu = 0 \\ \frac{\partial L(\alpha,\beta,\mu)}{\partial \alpha_3} = -\ln\alpha_3 - 1 - \beta_0 - \beta_2 + \beta_3 - \mu = 0 \\ \frac{\partial L(\alpha,\beta,\mu)}{\partial \alpha_4} = -\ln\alpha_4 - 1 - 2\beta_0 - \beta_3 + \beta_4 - \mu = 0 \\ \frac{\partial L(\alpha,\beta,\mu)}{\partial \alpha_5} = -\ln\alpha_5 - 1 - 3\beta_0 - \beta_4 - \mu = 0 \\ \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \alpha_1 - \alpha_3 - 2\alpha_4 - 3\alpha_5 \geq 0 \\ \alpha_1 \geq \alpha_2 \geq \alpha_3 \geq \alpha_4 \geq \alpha_5 > 0 \\ \beta_0, \beta_1, \beta_2, \beta_3, \beta_4 \geq 0 \end{cases} \quad (5)$$

Будем перебирать неравенства (4.1) – (4.4) на предмет их строгости.

Случай 1. Неравенства (4.1) – (4.4) строгие. То есть $\alpha_1 > \alpha_2 > \alpha_3 > \alpha_4 > \alpha_5 > 0$. Тогда $\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0$, $\beta_4 = 0$. Условия Лагранжа (5) примут вид:

$$\begin{cases} -\ln\alpha_1 - 1 + \beta_0 - \mu = 0 \\ -\ln\alpha_2 - 1 - \mu = 0 \\ -\ln\alpha_3 - 1 - \beta_0 - \mu = 0 \\ -\ln\alpha_4 - 1 - 2\beta_0 - \mu = 0 \\ -\ln\alpha_5 - 1 - 3\beta_0 - \mu = 0 \\ \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \alpha_1 - \alpha_3 - 2\alpha_4 - 3\alpha_5 \geq 0 \\ \alpha_1 > \alpha_2 > \alpha_3 > \alpha_4 > \alpha_5 > 0 \\ \beta_0 \geq 0 \end{cases} \Rightarrow \begin{cases} -\ln\alpha_2 - 1 = \mu \\ \ln\frac{\alpha_1}{\alpha_2} = \beta_0 \\ \ln\frac{\alpha_1^2}{\alpha_1\alpha_3} = 0 \\ \ln\frac{\alpha_2^3}{\alpha_1^2\alpha_4} = 0 \\ \ln\frac{\alpha_2^4}{\alpha_1^3\alpha_5} = 0 \\ \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \alpha_1 - \alpha_3 - 2\alpha_4 - 3\alpha_5 \geq 0 \\ \alpha_1 > \alpha_2 > \alpha_3 > \alpha_4 > \alpha_5 > 0 \\ \beta_0 \geq 0 \end{cases}$$

Так как $\alpha_1 > \alpha_2$, поэтому $\ln\frac{\alpha_1}{\alpha_2} = \beta_0 > 0$. Значит неравенство (4.0) не может быть строгим. Тогда это равенство. Будем последовательно исключать $\alpha_1, \alpha_2, \alpha_3$.

$$\begin{cases} \alpha_1 = \frac{\alpha_2^2}{\alpha_3} \\ \alpha_2\alpha_3 = \alpha_1\alpha_4 \\ \alpha_3^2 = \alpha_1\alpha_5 \\ \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \alpha_1 - \alpha_3 - 2\alpha_4 - 3\alpha_5 = 0 \\ \alpha_1 > \alpha_2 > \alpha_3 > \alpha_4 > \alpha_5 > 0 \end{cases} \Rightarrow \begin{cases} \alpha_2\alpha_3^2 = \alpha_2^2\alpha_4 \\ \alpha_3^3 = \alpha_2^2\alpha_5 \\ \frac{\alpha_2^3}{\alpha_3^2} + \alpha_2 + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \frac{\alpha_2^3}{\alpha_3^2} - \alpha_3 - 2\alpha_4 - 3\alpha_5 = 0 \\ \alpha_2 > \alpha_3 > \alpha_4 > \alpha_5 > 0 \end{cases} \Rightarrow$$

$$\begin{cases} \alpha_2 = \frac{\alpha_3^2}{\alpha_4} \\ \alpha_3^3 = \frac{\alpha_3^4}{\alpha_4^2}\alpha_5 \\ \frac{\alpha_3^6}{\alpha_4^3} + \frac{\alpha_3^3}{\alpha_4} + \alpha_3 + \alpha_4 + \alpha_5 = 1 \\ \frac{\alpha_3^6}{\alpha_4^3} - \alpha_3 - 2\alpha_4 - 3\alpha_5 = 0 \end{cases} \Rightarrow \begin{cases} \alpha_3 = \frac{\alpha_4^2}{\alpha_5} \\ \alpha_3^3 + \alpha_3^2\alpha_4 + \alpha_3\alpha_4^2 + \alpha_4^3 + \alpha_4^2\alpha_5 = \alpha_4^2 \\ \alpha_3^3 - \alpha_4^2\alpha_3 - 2\alpha_4^3 - 3\alpha_4^2\alpha_5 = 0 \end{cases} \Rightarrow$$

$$\begin{cases} \alpha_4^4 + \alpha_4^3\alpha_5 + \alpha_4^2\alpha_5^2 + \alpha_4\alpha_5^3 + \alpha_5^4 = \alpha_5^3 \\ \alpha_4^4 - \alpha_4^2\alpha_5^2 - 2\alpha_4\alpha_5^3 - 3\alpha_5^4 = 0 \end{cases} \Rightarrow \begin{cases} \alpha_4^3 + 2\alpha_4^2\alpha_5 + 3\alpha_4\alpha_5^3 + 4\alpha_5^3 - \alpha_5^2 = 0 \\ 2\alpha_5 - 5\alpha_5^2 = \alpha_4 \end{cases}.$$

То есть α_5 является решением уравнения: $125\alpha_5^4 - 200\alpha_5^3 + 115\alpha_5^2 - 26\alpha_5 + 1 = 0$. Исследуем функцию $f(x) = 125x^4 - 200x^3 + 115x^2 - 26x + 1$ методами математического анализа, при условии, что $x \in [0, \frac{1}{5}]$. Получаем единственный корень, принадлежащий

заданному интервалу. $\alpha_5 \approx 0{,}0477244$; $\alpha_4 \approx 0{,}08406076$; $\alpha_3 \approx 0{,}14806276$; $\alpha_2 \approx 0{,}26079446$; $\alpha_1 \approx 0{,}459357595$; $F(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) \approx 3{,}834437226$.

Случай 2. Пусть неравенства (4.1) – (4.3) строгие, а (4.4) является равенством. То есть $\alpha_4 = \alpha_5$, $\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0$. Условия Лагранжа (5) примут вид:

$$\begin{cases} -\ln\alpha_1 - 1 + \beta_0 - \mu = 0 \\ -\ln\alpha_2 - 1 - \mu = 0 \\ -\ln\alpha_3 - 1 - \beta_0 - \mu = 0 \\ -\ln\alpha_4 - 1 - 2\beta_0 + \beta_4 - \mu = 0 \\ -\ln\alpha_5 - 1 - 3\beta_0 - \beta_4 - \mu = 0 \\ \beta_0, \beta_4 \geq 0 \end{cases}$$

Вычтем из первого равенства второе. Тогда $\ln\frac{\alpha_2}{\alpha_1} = -\beta_0$. Но $\beta_0 \geq 0$ и

$\alpha_1 > \alpha_2$. Противоречие. Аналогично, случаи 3 - 16 сводятся к противоречиям.

Теорема полностью доказана.

Выражаю благодарность моему научному руководителю, профессору Мищенко Сергею Петровичу за постановку задачи, постоянное внимание и интерес к работе.

Литература

1. Regev A. Existence of polynomial identities in A $\otimes$ B// Bull. Amer. Math. Soc. 1971. V.77, N. 6. 1067-1069.

2. Giambruno A., Zaicev M. Exponential codimension growth of P.I. algebras: an exact estimate// Adv. Math.-1999.- 142.-221-243.

3. Giambruno A., Mishchenko S. P., Zaicev M. V. Codimensions of Algebras and Growth Functions// Adv. Math. 2008. 217. 1027-1052.

4. Mishchenko S. P., Zaicev M. V. An example of a variety of Lie algebras with a fractional exponent// J. Math. Sci. (New York). 1999. 93, N. 6. 977-982.

5. Веревкин А.Б., Зайцев М.В., Мищенко С.П. Достаточное условие совпадения нижней и верхней экспонент многообразия линейных алгебр// Вестник Моск. Унив. Сер. Мат. и мех. 2011, N. 2. 36-39.

6. Giambruno A. and Zaicev M. Polynomial Identities and Asymptotic Methods, Mathematical Surveys and Monographs// vol. 122, AMS, Providence, 2005.

7. Бахтурин Ю.А. Тождества в алгебрах Ли.// М.: Наука. 1984

8. Мищенко С.П. Цветные диаграммы Юнга // Вестник МГУ, 1993.-С. 90-91.

9. Зайцев М.В., Мищенко С.П. О полиномиальности роста кодлины многообразий алгебр Ли//Алгебра и логика. 1999. 38, N. 2. 161-175.

КОДЛИНА МНОГООБРАЗИЯ АЛГЕБРЫ ЛИ N_2A

С.П.Мищенко, Ю.Р.Фятхутдинова

Ульяновский государственный университет

На протяжении всей работы основное поле имеет нулевую характеристику. В данной статье речь пойдет о свойствах многообразия алгебр Ли $\mathbf{N_2A}$, определенного тождеством

$$(x_1x_2)(x_3x_4)(x_5x_6) \equiv 0.$$

Все необъясняемые понятия можно найти в монографиях [1], [2].

Изучение многообразия $\mathbf{N_2A}$ было начато в работе [3]. В частности, в ней доказано, что это многообразие имеет почти полиномиальный рост и его экспонента равна 2. В ней же доказано, что кратности этого многообразия для $n \geq 5$ вычисляются по формулам:

$$m_\lambda = \begin{cases} m(p,q) & \text{если } \lambda = (p+q+1, p+1, 1, 1) \text{ или } (p+q+2, p+2, 2); \\ m(p,q) & \text{если } \lambda = (p+q, p), \text{ где } p \geq 2; \\ n(p,q) & \text{если } \lambda = (p+q+1, p+1, 1); \\ 1 & \text{если } \lambda = (n-1, 1); \\ 0 & \text{в остальных случаях.} \end{cases}$$

Здесь, $m(p,q) = q/2$, если p, q – четные или $m(p,q) = [q/2] + 1$ в остальных случаях и $n(p,q) = \left[\frac{q+1}{2}\right]$, если $p = 0$, или $n(p,q) = q + 1$, если $p \neq 0$.

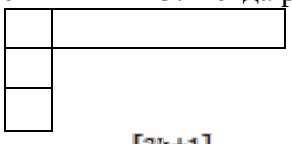
В данной статье продолжено исследование числовых характеристик этого многообразия. В частности, подсчитана так называемая кодлина $l_n(\mathbf{N_2A})$, $n=1,2, \ldots$. Напомним, что по определению кодлина равна сумме кратностей по всем разбиениям числа n.

Теорема. Кодлина многообразия $\mathbf{N_2A}$ вычисляется по следующим формулам:

$$l_n = \begin{cases} \dfrac{5n^2 - 24n + 32}{8}, & \text{если } n=4m; \\ \dfrac{5n^2 - 24n + 36}{8}, & \text{если } n=4m+2; \\ \dfrac{5n^2 - 24n + 35}{8}, & \text{если } n=4m+1 \text{ или } n=4m+3. \end{cases}$$

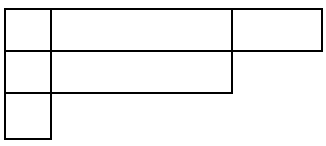
Доказательство. Формула кодлины получена непосредственным вычислением, используя формулы для кратности. Например, рассмотрим случай, когда разбиение имеет вид: $\lambda = (p + q + 1, p + 1, 1)$. Для этого типа разбиений кратности вычисляются по следующим формулам: $n(p,q) = \left[\frac{q+1}{2}\right]$, если $p = 0$, и $n(p,q) = q + 1$, если $p \neq 0$.

Пусть n – нечетное. Зафиксируем $n = 2k + 3$. Когда $p = 0$, диаграмма Юнга имеет вид:

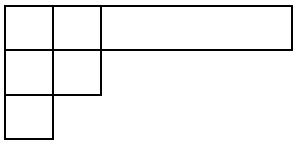
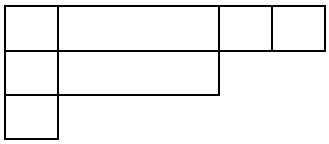
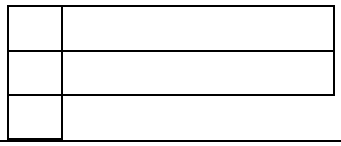


И в этом случае $q = 2k$. Тогда кратность $m_\lambda = \left[\frac{2k+1}{2}\right] = k$.

Когда $p \neq 0$, диаграмма Юнга имеет следующий вид:



Рассмотрим изменение диаграммы Юнга при увеличении числа *p* в виде таблицы:

p	Диаграмма Юнга	q	m_λ
1		2k - 2	2k - 1
. . .			
k – 1		2	3
k		0	1

Тогда по формуле суммы k – членов арифметической прогрессии имеем для случая

$$p \neq 0: \ \sum m_\lambda = \frac{(2k-1+1)k}{2} = k^2.$$

Так как мы фиксировали n = 2k + 3, выразим из него k и подставим в полученные нами формулы. Тогда сумма кратностей для данного типа разбиений будет равна

$$\sum m_\lambda = k^2 + k = \frac{n-3}{2}\left(\frac{n-3}{2}+1\right) = \frac{(n-3)(n-1)}{4}.$$

Пусть n – четное. Зафиксируем n = 2k + 4. Размышляя аналогичным способом, получим:

$$\sum m_\lambda = \frac{(n-2)^2}{4}.$$

В случае разбиений вида λ = (p + q + 1, p + 1, 1, 1) получаем следующую сумму кратностей:

$$\sum m_\lambda = \begin{cases} \frac{(n-4)n}{8}, & \text{если } n=4m; \\ \frac{(n-2)^2}{8}, & \text{если } n=4m+2; \\ \frac{(n-1)(n-3)}{8}, & \text{если } n=4m+1 \text{ или } n=4m+3. \end{cases}$$

Если рассмотрим разбиения вида λ = (p + q + 2, p + 2, 2), то сумма кратностей определится так:

$$\sum m_\lambda = \begin{cases} \frac{(n-4)^2}{8}, & \text{если } n=4m; \\ \frac{(n-6)(n-2)}{8}, & \text{если } n=4m+2; \\ \frac{(n-5)(n-3)}{8}, & \text{если } n=4m+1 \text{ или } n=4m+3. \end{cases}$$

А рассматривая разбиение вида λ = (p + q, p), где $p \geq 2$, получим

$$\sum m_\lambda = \begin{cases} \dfrac{(n-4)n}{8}, & \text{если n=4m;} \\ \dfrac{(n-2)^2}{8}, & \text{если n=4m+2;} \\ \dfrac{(n-1)(n-3)}{8}, & \text{если n=4m+1 или n=4m+3.} \end{cases}$$

Так как кодлина равна сумме кратностей по всем разбиениям числа n, то сложив полученные суммы кратностей и 1 (случай разбиения λ = (n – 1, 1)), получим формулу для вычисления кодлины многообразия $\mathbf{N_2A}$:

$$l_n = \begin{cases} \dfrac{5n^2-24n+32}{8}, & \text{если n=4m;} \\ \dfrac{5n^2-24n+36}{8}, & \text{если n=4m+2;} \\ \dfrac{5n^2-24n+35}{8}, & \text{если n=4m+1 или n=4m+3.} \end{cases}$$

Теорема доказана.

Литература:

1. Бахтурин Ю.А. Тождества в алгебрах Ли. - М.: Наука, 1985. 448 с.
2. Giambruno A., Zaicev M. Polynomial Identities and Asymptotic Methods. - Mathematical Surveys and Monographs Vol. 122, American Mathematical Society, Providence, RI,2005. 356.
3. Мищенко С.П. Многообразия алгебр Ли с двухстепенно нильпотентным коммутантом// Весці АН БССР №6. Сер. физ.–мат. навук, 1987. С. 39 - 43

О ДВИЖЕНИИ ШАРА С УДАРАМИ О ШЕРОХОВАТУЮ ПОВЕРХНОСТЬ

Л.С.Отраднова

Московский государственный университет им. Ломоносова

Аннотация. Рассматривается несколько задач о движении шара по инерции: между двумя параллельными плоскостями, внутри сферы и внутри кругового цилиндра. Считается что при ударе происходит мгновенное наложение и снятие связи, состоящее в том, что касательная составляющая скорости контактирующей точки шара равна нулю, т.е. выполняется условие качения без проскальзывания. Показывается, что во всех случаях движение в пределе выходит на установившийся режим по скорости: угловая скорость шара стремится к постоянному значению, а скорость его центра становится периодический, для плоскостей, и условно периодический, для сферы и цилиндра. В некоторых случаях на установившийся режим выходят и координаты, определяющие положение и ориентацию шара. Ключевые слова: движение шара по инерции, удары, качение без проскальзывания, неголономные связи.

1. Удар шара о шероховатую поверхность.

Рассмотрение ведется в рамках модели удара с вязким трением предложенной в [1]. Считается что в момент удара нормальная составляющая импульса меняет свой знак, а касательная составляющая разбивается на два слагаемых. Первое, отвечающее качению без проскальзывания, сохраняется, что соответствует абсолютно упругому столкновению. Второе, нормальное к первому, обращается в ноль. Это соответствует абсолютно неупругому столкновению. Эквивалентное описание такой модели соударения состоит в мгновенном наложении и снятии связей, отвечающих качению тела без проскальзывания. В работе [2] рассматривалась подобная модель ударного взаимодействия, но предполагалась, что полная энергия системы сохраняется при ударе. Соударение твердых тел с сухим трением и другие модели в общем случае подробно рассмотрены в [3–5].

Рассмотрим однородный шар радиуса a имеющий единичную массу $m=1$ и главные центральные моменты инерции J. Из физических соображений $J \le \frac{2}{3}a^2$. Движение шара происходит по инерции и ограничено некоторой неподвижной гладкой поверхностью. Пусть шар ударяется о поверхность в точке P. Введем обозначения: γ – единичная нормаль к поверхности в точке P, направленная внутрь области допустимой для движения шара; ω – угловая скорость шара; V_c – скорость его центра C.

Считается, что при ударе шара о поверхность происходит мгновенное наложение и снятие связи, состоящей в том, что касательная к поверхности составляющая скорости точки P шара равна нулю: $V_p^+ - \gamma < \gamma, V_p^+ \ge 0$ (условие качения без проскальзывания) или

$$V_c^+ - \left[\omega^+, a\gamma\right] - \gamma\left\langle V_c^+, \gamma\right\rangle = 0. \qquad (1)$$

Знаками $(\ )^-$ и $(\ )^+$ мы обозначаем параметры движения шара сразу до и после удара, (имея в виду $t-0$ и $t+0$ для удара в момент t).

Найдем, как связаны параметры движения шара до и после такого удара. Условие (1) допускает в момент удара любые виртуальные повороты шара вокруг точки P. Считаем, что моменты трения качения и верчения при ударе отсутствуют. Поэтому при ударе сохраняется кинетический момент шара относительно этой точки:

$$\left(a\left[\gamma, V_c\right] + J\omega\right)^- = \left(a\left[\gamma, V_c\right] + J\omega\right)^+. \qquad (2)$$

Условие (1) допускает в момент удара любое виртуальное перемещение вдоль $\vec{\gamma}$. Удар о поверхность считаем абсолютно упругим, т.е. нормальная к поверхностисоставляющая импульса шара при ударе меняет знак:

$$(\gamma, V_c)^- = -(\gamma, V_c)^+. \qquad (3)$$

Соотношения (1–3) позволяют определить параметры движения шара после удара по их значениям до удара. Перейдем к координатной форме записи. Введем неподвижную систему координат Oxyz так, чтобы ось Oz была параллельна вектору γ (но не обязательно сонаправлена). Тогда $\gamma = (0, 0, \gamma_z)$, $\gamma = \pm 1$, $\varpi = (\varpi_x, \varpi_y, \varpi_z)$, $V_c = (\dot{x}, \dot{y}, \dot{z})$ и соотношения (1 – 3) примут следующий вид:

$$\dot{x}^+ - a\gamma_z\omega_y^+ = 0\,,\ \dot{y}^+ + a\gamma_z\omega_x^+ = 0\,, \qquad (4)$$

$$\dot{z}^+ = -\dot{z}^-\,, \qquad (5)$$

$$\begin{aligned} -a\gamma_z\dot{y}^- + J\omega_x^- &= -a\gamma_z\dot{y}^+ + J\omega_x^+\,, \\ a\gamma_z\dot{x}^- + J\omega_y^- &= a\gamma_z\dot{x}^+ + J\omega_y^+\,, \end{aligned} \quad \omega_z^- = \omega_z^+\,. \qquad (6)$$

Подставив в (6) величины $\dot{x}^+$ и $\dot{y}^+$ найденные из (4), получим

$$\begin{aligned} -a\gamma_z\dot{y}^- + J\omega_x^- &= (a^2 + J)\omega_x^+\,, \\ a\gamma_z\dot{x}^- + J\omega_y^- &= (a^2 + J)\omega_y^+\,. \end{aligned}$$

Значит

$$\omega_x^+ = \frac{-a\gamma_z\dot{y}^- + J\omega_x^-}{J + a^2}\,,\ \omega_y^+ = \frac{a\gamma_z\dot{x}^- + J\omega_y^-}{J + a^2}\,,\ \omega_z^+ = \omega_z^-\,, \qquad (7)$$

$$\dot{x}^+ = a\gamma_z\omega_y^+\,,\ \dot{y}^+ = -a\gamma_z\omega_x^+\,, \qquad (8)$$

$$\dot{z}^+ = -\dot{z}^-\,. \qquad (9)$$

2. Движение шара между двумя параллельными плоскостями.

Пусть теперь шар движется по инерции между двумя шероховатыми плоскостями: $z = b$ и $z = -b$, $(b > a)$. Считаем, что в начальный момент $\dot{z} \neq 0$, т.е. шар движется соударяясь с плоскостями. Изменение параметров движения шара при ударе определяется соотношениями (4 – 6). При ударе о нижнюю плоскость $\gamma_z = 1$, а при ударе о верхнюю $\gamma_z = -1$. Рассмотрим как меняются параметры движения при последовательных ударах шара сначала о нижнюю плоскость, затем о верхнюю и т.д. После первого удара перед каждым ударом k о следующую плоскость параметры движения будут связаны соотношениями (4) для удара $k-1$ о предыдущую плоскость.

Поскольку $(\gamma_z)_{k-1} = -(\gamma_z)_k$, то

$$\left(\dot{x}^- + a\gamma_z\omega_y^-\right)_k = 0\,,\ \left(\dot{y}^- - a\gamma_z\omega_x^-\right)_k = 0\,.$$

С учетом этого соотношения (7–8) примут вид

$$\begin{aligned} \left(\dot{x}^+ = -\lambda\dot{x}^-\right)_k, \left(\omega_y^+ = \lambda\omega_y^-\right)_k, \\ \left(\dot{y}^+ = -\lambda\dot{y}^-\right)_k, \left(\omega_x^+ = \lambda\omega_x^-\right)_k, \end{aligned} \quad \lambda = \frac{J - a^2}{J + a^2}\,. \qquad (10)$$

Замечаем из (10), что $|\lambda| < 1$, поэтому при $k \to \infty$ имеем

$$\dot{x}_k \to 0\,,\ \dot{y}_k \to 0\,,\ (\omega_x)_k \to 0\,,\ (\omega_y)_k \to 0\,.$$

Эти величины убывают по модулю монотонно и после каждого удара меняют знак.

В целом движение шара подобно качению без проскальзывания по плоскости. Угловая скорость направлена вдоль постоянного вектора $(\omega_x, \omega_y, \omega_z)_1^+$. Проекция центра шара на горизонтальную плоскость движется по некоей неподвижной прямой L, направленной по вектору $(\dot{x}, \dot{y})_1^+$.

Интервал времени между ударами постоянен, поскольку из (9) имеем $|\dot{z}_k| = |\dot{z}_1|$. Поэтому точки удара стремятся к некоей точке на $L: (x_k, y_k) \to (x^*, y^*)$. В пределе движение выходит на периодический режим: шар попеременно ударяется о плоскости, его угловая

скорость постоянна и вертикальна, центр шара двигается по вертикальному отрезку, проходящему через точку $(x^*, y^*, 0)$.

3. Движение шара внутри сферы.

Пусть шар движется внутри сферы радиуса $R > a$ с центром в некоторой точке O. Изучим поведение параметров движения шара после второго, третьего и т.д. удара. Пусть на ударе k шар ударяется о точку сферы P_k и внутренняя нормаль сферы в этой точке обозначена γ_k, т.е. $P_k + R_{\gamma_k} = O$.

Скорость V_c центра шара при движении от удара $k-1$ до удара k коллинеарна вектору $\overline{P_{k-1}P_k}$.

Введем две системы координат $P_i xyz$ $(i = k-1, k)$ с началом в точках P_i. Ось $P_i z$ направим по γ_i. Пусть точки P_i не являются диаметрально противоположными. Оси $P_i y$ выберем ортогональными плоскости $OP_{k-1}P_k$ – обе эти оси коллинеарны.

Направим их так, чтобы тройка $\gamma_{k-1}, \gamma_k, P_i y$ была правой.

Если же точки P_i $(i = k-1, k)$ диаметрально противоположны ($\gamma_{k-1} = -\gamma_k$), то оси $P_{k-1}x$ и $P_{k-1}y$ развернем произвольным образом, ортогонально γ_{k-1}. Ось $P_k y$ выберем параллельной оси $P_{k-1}y$ и с таким же направлением. При этом ось $P_k x$ будет направлена противоположно оси $P_{k-1}x$.

В таких осях при движении от удара $k-1$ до удара k проекции угловой скорости шара ω на оси $P_i y$ совпадают. Вектор $\overline{P_{k-1}P_k}$ направлен по биссектрисе угла между направлениями осей $P_i x$, поэтому проекции вектора скорости V_c центра шара на оси $P_i x$ совпадают, а его проекции на оси $P_i y$ равны нулю. Отсюда получаем

$$\left(\dot{x}^+\right)_{k-1} = \left(\dot{x}^-\right)_k \geq 0,\ \left(\dot{y}^+\right)_{k-1} = \left(\dot{y}^-\right)_k = 0,$$
$$\left(\omega_x^+\right)_{k-1} = 0,\ \left(\omega_y^+\right)_{k-1} = \left(\omega_y^-\right)_k \geq 0,$$

и из (4) видим

$$\left(\dot{x}^+\right)_{k-1} - a\left(\omega_y^+\right)_{k-1} = 0,\ \left(\dot{x}^-\right)_k - a\left(\omega_y^-\right)_k = 0.$$

При ударе k выполняются соотношения (7–8), в которых $\gamma_z = 1$. Из них находим

$$\left(\omega_x^+\right)_k = \frac{J\left(\omega_x^-\right)_k}{J + a^2},\ \left(\omega_y^+\right)_k = \left(\omega_y^-\right)_k,\ \left(\omega_z^+\right)_k = \left(\omega_z^-\right)_k,$$
$$\left(\dot{x}^+\right)_k = \left(\dot{x}^-\right)_k \geq 0,\ \left(\dot{y}^+\right)_k = -a\left(\omega_x^+\right)_k. \qquad (11)$$

Вектор ω постоянен на участке движения между ударами, поэтому $\omega_{k-1}^+ = \omega_k^-$.

Поскольку $\frac{J}{J + a^2} < 1$, то $\left|\omega_{k-1}^+\right| = \left|\omega_k^-\right| \geq \left|\omega_k^+\right|$, т.е. последовательность $\left|\omega_k^+\right|$ монотонно невозрастает и, поэтому сходится к некоторому числу: $\left|\omega_k^+\right| \to g \geq 0$ при $k \to \infty$. Отсюда вытекает, что $\left(\omega_x^-\right)_k \to 0$, и, значит $\left(\omega_x^+\right)_k \to 0$, $\left(\dot{y}^+\right)_k \to 0$. Следовательно плоскость $OP_{k-1}P_k$ сходится к некоторой неподвижной плоскости (за исключением случаев, когда движение центра шара останавливается, или переходит в движение по диаметру сферы).

Оси $P_{k-1}y$ и $P_k y$ параллельны и система $P_k xyz$ повернута относительно системы $P_{k-1}xyz$ вокруг оси $P_k y$ на некий угол α_k. Поскольку $\left(\omega_x^+\right)_{k-1} = 0$, то $\left(\omega_z^-\right)_k = \left(\omega_z^+\right)_{k-1}\cos\alpha_k$, откуда $\left|\left(\omega_z^+\right)_k\right| = \left|\left(\omega_z^-\right)_k\right| \leq \left|\left(\omega_z^+\right)_{k-1}\right|$. Значит, последовательность $\left|\left(\omega_z^+\right)_k\right|$ монотонно невозрастает

и, следовательно, сходится к некоторому пределу: $\left|\left(\omega_z^+\right)_k\right| \to g_z \geq 0$. Но тогда и $\left|\left(\omega_y^+\right)_k\right| \to g_y = \sqrt{g^2 - g_z^2}$. Используя (11) окончательно получаем

$$\left(\omega_x^+\right)_k \to 0,\ \left(\omega_y^+\right)_k \to g_y,\ \left|\left(\omega_z^+\right)_k\right| \to g_z \qquad (12)$$

Поскольку модуль угловой скорости убывает и сходится, то из закона сохранения энергии (5) получаем, что монотонно возрастает и сходится модуль скорости центра шара: $\left|\left(V_c\right)_k^+\right| \to f \geq 0$. Используя (8) и (12), а также то, что $\left(\dot{z}^+\right)_k \geq 0$ получаем

$$\left(\dot{x}^+\right)_k \to f_x = a g_y,\ \left(\dot{y}^+\right)_k \to 0,\ \left(\dot{z}^+\right)_k \to f_z = \sqrt{f^2 - f_x^2}.$$

Рассмотрим случай $f = 0$, т.е. $f_x = f_z = g_y = 0$. Это значит, что в пределе движение центра шара останавливается, а его и он вращается вокруг оси направленной от центра шара к центру сферы.

Пусть теперь $f \neq 0$. Из геометрических соображений $\dfrac{\left(\dot{z}^+\right)_k}{\left|\left(V_c\right)_k^+\right|} = \sin\dfrac{\alpha_{k+1}}{2}$, $0 \leq \alpha_k \leq \pi$. Переходя к пределу, получаем

$$\alpha_k \to \alpha^*,\ \sin\frac{\alpha^*}{2} = \frac{f_z}{f}.$$

Если $f_z = 0$, то $\alpha^* = 0$, $f_x \neq 0$, $g_y \neq 0$. Предельное движение представляет собой качение шара по внутренности сферы без проскальзывания. Заметим, что в предельном движении угловая скорость ортогональна вектору γ. Однако может оказаться, что $g_z \neq 0$.

Если $0 < f_z < f$, то $0 < \alpha^* < \pi$. Поскольку $\cos\alpha_k < 1$, то переходя к пределу в равенстве $\left|\left(\omega_z^-\right)_k\right| = \left|\left(\omega_z^+\right)_{k-1}\right|\left|\cos\alpha_k\right|$ получаем $\left|\left(\omega_z^+\right)_k\right| \to 0$. В предельном движении шар движется аналогично точке математического бильярда в круге. Центр шара движется в плоскости по хордам окружности одинаковой длины. Шар периодически ударяется о сферу, вращаясь с постоянной угловой скоростью ортогональной плоскости, в которой движется центр шара.

Если $f_z = f$, то $\alpha^* = \pi$, $f_x = 0$, $g_y = 0$. В предельном движении центр шара движется по диаметру сферы. Шар периодически ударяется о диаметрально противоположные точки сферы. Угловая скорость направлена вдоль диаметра и постоянна.

В завершение раздела отметим, что все предельные движения шара возможны,при задании подходящих начальных условий. Однако открытым остается вопрос о том, какие из них являются действительно предельными, т.е. можно ли подобрать такие начальные условия, чтобы движение шара, которое вначале не совпадало с данным предельным движением, сошлось к нему.

4. Движение шара внутри цилиндра.

Пусть шар движется внутри кругового цилиндра радиуса $R > a$. Изучим поведение параметров движения шара после второго, третьего и т.д. удара. На ударе k шар ударяется о точку цилиндра P_k и внутренняя нормаль цилиндра в этой точке обозначена k, т.е. точка $P_k + R_{\gamma_k}$ лежит на оси цилиндра. Введем систему координат $P_k xyz$ с началом в точке P_k. Ось $P_k z$ направим по γ_i. Ось $P_k y$ направим по оси симметрии цилиндра Oz. Ось $P_k x$ направлена по касательной к направляющей окружности цилиндра, проходящей через точку P_k.

Скорость V_c центра шара при движении от удара $k-1$ до удара k коллинеарна вектору $\overline{P_{k-1}P_k}$. Из симметрии цилиндра и постоянства оси $P_k y$ находим

$$\left(\dot{x}\right)_{k-1}^+ = \dot{x}_k^-,\ \left(\dot{y}\right)_{k-1}^+ = \dot{y}_k^-,\ \left(\dot{z}\right)_{k-1}^+ = -\dot{z}_k^-,\ \left(\omega_y\right)_{k-1}^+ = \left(\omega_y\right)_k^-.$$

Соотношения (7–8): дают

$$\left(\omega_y\right)_k^+ = \frac{a\dot{x}_k^- + J\left(\omega_y\right)_k^-}{J + a^2},\ \dot{x}_{k-1}^+ = a\left(\omega_y\right)_{k-1}^+$$

и

$$\left(\omega_x\right)_k^+ = \frac{-a\dot{y}_k^- + J\left(\omega_x\right)_k^-}{J + a^2},\ \dot{y}_{k-1}^+ = -a\left(\omega_x\right)_{k-1}^+.$$

Отсюда получаем

$$\left(\omega_x\right)_k^+ = \frac{a^2\left(\omega_x\right)_{k-1}^+ + J\left(\omega_x\right)_k^-}{J + a^2},\ \left(\omega_y\right)_k^+ = \left(\omega_y\right)_k^-,\ \left(\omega_z\right)_k^+ = \left(\omega_z\right)_k^-.$$

Таким образом для всех ударов после первого будет

$$\left(\omega_y\right)_k^+ = \left(\omega_y\right)_k^- = const = g_1,\ \dot{x}_k^+ = \dot{x}_k^- = const = g_2.$$

Поскольку все параметры движения шара ограничены, то существуют предельные точки. выберем любую. Для предельных значения сохраняем обозначения, но не пишем номер удара:

$$(\ )_k^+ \to (\ )^+,\ (\ )_k^- \to (\ )^-,$$

тогда

$$\omega_x^+ = \frac{a^2\omega_x^+ + J\omega_x^-}{J + a^2},\ \omega_y^+ = \omega_y^- = \omega_y,\ \omega_z^+ = \omega_z^- = \omega_z.$$

или

$$\omega_x^+ = \omega_x^- = \omega_x,\ \omega_y^+ = \omega_y^- = \omega_y,\ \omega_z^+ = \omega_z^- = \omega_z.$$

Рассмотрим, какие возможны предельные точки

1. Пусть $g_2 \neq 0$. Тогда вектор ϖ остается неизменным при вращении вокруг оси Oy, и, следовательно параллелен ей:

$$\varpi = \left(0,\ g_1,\ 0\right).$$

Значит $\dot{y}^+ = \dot{y}^- = 0$.

Из соотношения $\left(\dot{z}\right)_{k-1}^+ = -\dot{z}_k^- = \dot{z}_k^+ - 1$ получаем

$$\dot{z}^+ = -\dot{z}^- = g_3 = \dot{z}_1^+,\ V_c^+ = \left(g_2,\ 0,\ g_3\right),\ V_c^- = \left(g_2,\ 0,\ -g_3\right).$$

Таким образом, если $g_2 \neq 0$, то есть всего одна предельная точка и, значит, параметры параметры скорости движения сходятся при $k \to \infty$ к предельному режиму движения. Такое движение аналогично движению точки математического бильярда в круге. Это может быть качение шара по направляющей окружности цилиндра (если $g_3 = 0$), или центр шара движется в плоскости по хордам этой окружности одинаковой длины. Шар периодически ударяется о цилиндр, вращаясь с постоянной угловой скоростью параллельной оси цилиндра.

2. Пусть $g_2 = 0$. Это означает, что движение центра шара происходит все время в вертикальной плоскости и мы имеем случай совпадающий с движением шара между двумя параллельными плоскостями.

Авторы весьма признательны А.В. Карапетяну за полезные обсуждения данной работы.

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проекты 10-01-00406, 12-01-00441 и 12-08-00591).

Литература

1. В.В. Козлов Об ударе с трением. // Механика твердого тела. 1989. 6. 54–60.

2. Березинская С.Н., Кугушев Е.И., Сорокина О.В. О движении механических систем с односторонними связями. // Вестн. Моск. ун-та. сер. 1. мат. мех. 2005. 3. 18–24.

3. Раус Э.Дж. Динамика системы твердых тел. т. 1. М.: Наука, 1983.

4. Маркеев А.П. Динамика тела, соприкасающегося с твердой поверхностью. М.: Наука, 1992.

5. Иванов А.П. Динамика систем с механическими соударениями. М.: Международная программа образования, 1997.

ПРИМЕНЕНИЕ ТЕОРИИ ФУНКЦИЙ КОМПЛЕКСНОГО ПЕРЕМЕННОГО ДЛЯ РЕШЕНИЯ ЗАДАЧ ВНЕШНЕГО ОБТЕКАНИЯ АЭРОДИНАМИЧЕСКОГО ПРОФИЛЯ С ИНТЕРЦЕПТОРОМ

К.В.Редькина, В.А.Фролов

Самарский государственный аэрокосмический университет имени академика С.П. Королёва (национальный исследовательский университет)

Введение.

Интерцепторы в настоящее время применяются для обеспечения высокой поперечной управляемости современных самолётов, а также для уменьшения подъёмной силы и увеличения силы торможения на режиме посадки и пробега самолёта по взлётно-посадочной полосе. Отклонение интерцепторов на крыле может вызывать нестационарные изменения его аэродинамических характеристик из-за развития на крыле отрывной зоны. Существенная необходимость в определении аэродинамических характеристик самолётов при нестационарном обтекании возникает в задачах аэроупругости и при разработке активной системы управления подъёмной силы крыла. В этой связи при проектировании интерцепторов необходимо правильно понимать аэродинамику интерцепторов, включая во внимание производимый ими вихревой след и его влияние на закрылки и горизонтальное оперение.

В ряде работ [1–3] рассматривались математические модели течений около аэродинамических профилей с интерцепторами. В работах [1, 2] стационарная рециркуляционная зона за интерцептором моделировалась методом конформных отображений. В работе [3] математическая модель основывалась на нестационарном подходе на основе использования метода дискретных вихрей (МДВ).

Целью работы является разработка математической модели построения квазиточных решений внешнего обтекания потенциальным потоком крылового профиля с интерцептором.

Научная новизна работы заключается в получении новых фундаментальных знаний о параметрах потенциальных течений с циркуляцией около аэродинамического профиля с интерцептором.

Практическая значимость – применение вычислительной программы для инженерных расчётов аэродинамических характеристик крыльев с механизацией в виде интерцепторов.

Основной задачей работы является построение новых математических моделей обтекания двумерных тел, основанных на сочетании аналитических преобразований (конформных отображений) и численных схем. В данной работе предлагается решение задачи применения численно-аналитического метода (ЧАМ) [4] для построения внешнего обтекания аэродинамического профиля со стационарной отрывной зоной за интерцептором.

Эффективным методом решения потенциальных задач обтекания тел является теория функций комплексного переменного (ТФКП), в которой широко используется конформное отображение физической области течения на вспомогательную плоскость комплексного переменного. На практике часто приходится иметь дело с телами сложной формы, для которых отыскание функции конформного преобразования представляет значительные математические трудности. В этом случае используют различные численные методы, например (ЧАМ).

В данной работе в первом приближении предлагается разработать математическую модель течения, опирающуюся на потенциальное течение с циркуляцией около профиля с интерцептором, и получить картину обтекания данной конфигурации при наличии стационарного вихря путём визуализации линий тока, провести расчёт распределения скорости и давления на поверхности этой комбинации. Провести параметрическое исследование коэффициента подъемной силы, сравнить полученные данные с вычислительным экспериментом в пакете ANSYS CFX.

В рамках данной работы была написана программа на языке Фортран [5], реализующая:

1. потенциальное течение с циркуляцией около профиля с интерцептором;
2. построение линий тока около комбинации;
3. поиск точки стационарности вихря, моделирующего рециркуляционную зону за интерцептором;
4. расчёт подъёмной силы конфигурации профиля с интерцептором;
5. вычисление распределённых характеристик потока на поверхности моделируемых тел;

Данная программа состоит из основной программы, 21 подпрограмм, 6 функций и 10 модулей.

Ввод данных происходит путем считывания текстовых файлов. Вывод данных осуществляется путём записи текстовых файлов, которые создаются в самой программе. Передача данных между подпрограммами и функциями осуществляется с помощью ассоциации между модулями. Возможность использования подпрограмм и функций в главной программе, а также между собой выполняется с помощью интерфейсов.

Основным преимуществом данной работы является быстрота и точность расчётов благодаря использованию ЧАМ.

Потенциальными потребителями данного программного продукта могут быть инженеры конструкторских бюро, занимающихся проектированием летательных аппаратов с механизированными крыльями.

Геометрия профиля.

Геометрия исследуемого профиля описана в работе [6]. Пусть профиль образован эллипсом с полуосями a_e и $c/2$ и дугами окружности радиуса r, касающимися контура эллипса в его максимальном поперечном сечении. Обозначим хорду такого профиля как b, тогда безразмерная полуось эллипса $\overline{a}_e = a_e/b$ будет всегда соответствовать безразмерной координате максимальной толщины профиля, т.е. $\overline{a}_e = \overline{x}_c = x_c/b$. При задании относительной толщины профиля $\overline{c} = c/b$ и относительной координаты максимальной толщины профиля $\overline{x}_c$ безразмерный радиус дуги окружности $\overline{r} = r/b$, принадлежащей профилю, вычисляется по формуле

$$\overline{r} = \frac{\left(1-\overline{x}_c\right)^2}{\overline{c}} + 0{,}25\cdot\overline{c}\,.$$

Формулы, определяющие верхнюю и нижнюю дужку такого симметричного профиля для безразмерных координат $\overline{x} = \frac{x}{b};\quad \overline{y} = \frac{y}{b}$, одинаковы и имеют следующий вид

$$\overline{y} = \frac{\overline{c}}{2\overline{x}_c}\cdot\sqrt{\overline{x}\left(2\overline{x}_c - \overline{x}\right)} \quad \overline{x} \le \overline{x}_c;$$

$$\overline{y} = \frac{\overline{c}}{2} - \overline{r} + \sqrt{\overline{r}^2 - \left(\overline{x} - \overline{x}_c\right)^2} \quad \overline{x} > \overline{x}_c.$$

Введем обозначение $E - \left[\overline{f}\right]\left[\overline{x}_c\right]\left[\overline{c}\right]$ серии профилей, рассмотренного типа, где введён параметр $\overline{f} = \frac{f}{b}$, характеризующий относительную вогнутость профиля. Так профиль с обозначением E-003015 имеет $\overline{f} = 0;\ \overline{x}_c = 30\%$ и $\overline{c} = 15\%$. На рисунке 1 показаны три симметричных профиля с $\overline{x}_c = 30\%$ и относительными толщинами $\overline{c} = 15\%,\ 20\%$ и 25%.

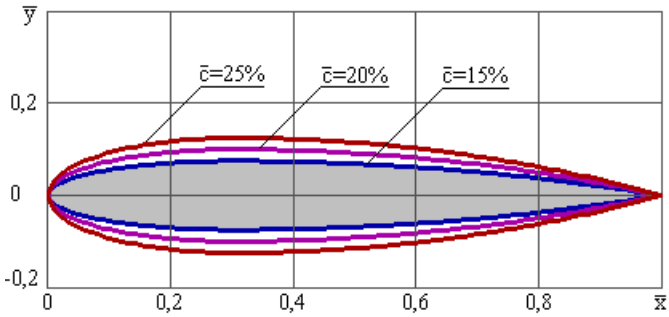


Рис. 1. Геометрия профилей Е-003025, Е-003020 и Е-003015.

Описание метода.

Рассматривается задача течения жидкости около симметричного аэродинамического профиля с эллиптической носовой частью при наличии стационарного вихря за интерцептором, моделирующего отрывную зону. Геометрическая схема задачи показана на рисунке 2.

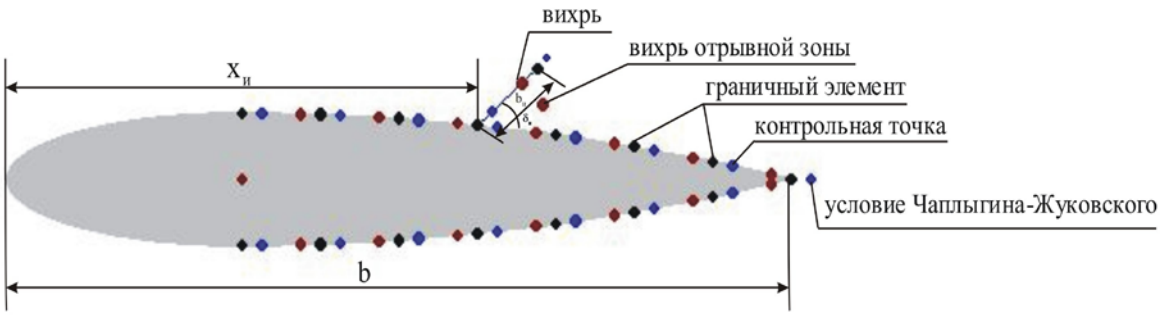


Рис. 2. Геометрическая схема задачи.

В рамках ЧАМ [4, 6–8] моделирование хвостовой части профиля и интерцептора выполняется с помощью набора точечных вихрей, равномерно распределённых на их поверхностях. Хвостовая часть профиля и интерцептор (контур *D* см. рисунок 3) разбиваются на граничные элементы, в пределах каждого помещается точечных вихрь и контрольная точка, используется численная схема метода дискретных вихрей «¼ - ¾». На ¼ граничного элемента располагается точечный вихрь, а на ¾ – контрольная точка.

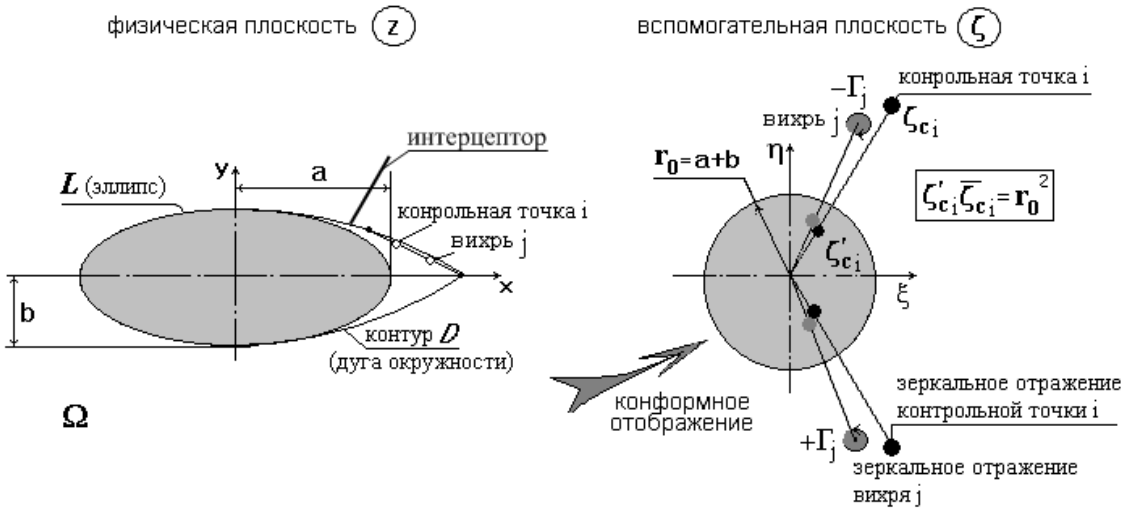


Рис. 3. Физическая и вспомогательная плоскости комплексного переменного.

Особенностью геометрической схемы является корректность расположения контрольных точек на хвостовой части профиля вблизи эллипса и в начале интерцептора. Это обеспечивает согласованность граничных условий и хорошую обусловленность системы линейных алгебраических уравнений (СЛАУ). Задача сводится к решению СЛАУ, физический смысл которой – удовлетворение условий непротекания в контрольных точках. Для обеспечения циркуляционного обтекания профиля в центр окружности во вспомогательной плоскости располагается вихрь.

В модели используется гипотеза стационарности внешнего вихря, расположенного за интерцептором, интенсивность которого находится из условия конечности скорости на задней кромке интерцептора, а координаты рассчитываются методом оптимизации – минимизации целевой функции скорости по двум проектным параметрам, в качестве которых принимаются координаты стационарного вихря.

Рассматривается обтекание аэродинамического профиля с интерцептором (рисунок 2) в присутствии стационарного дискретного вихря, расположенного за интерцептором. Среда считается НИЖ. Используется ТФКП, в рамках которой запись комплексного потенциала W определяет решение задачи.

Течение во вспомогательной плоскости на основании теоремы Милн-Томсона [9] можно описать комплексным потенциалом вида

$$W(\zeta)=\frac{1}{2}\left(\bar{V}_\infty\zeta+\frac{V_\infty a^2}{\zeta}\right)+\frac{1}{2\pi i}\left[\sum_{j=1}^{N+1}\Gamma_j\ln\frac{(\zeta-\zeta_{vj})\zeta}{\zeta-\frac{a^2}{\bar{\zeta}_{vj}}}+\Gamma_{N+2}\ln\zeta\right] \qquad (1)$$

где $\zeta=\xi+i\eta$ – комплексная переменная во вспомогательной плоскости (рисунок 3);

$i=\sqrt{-1}$ – мнимая единица;

$V_\infty,\bar{V}_\infty$ – комплексная и сопряжённая комплексная скорости набегающего потока, соответственно;

a – радиус окружности во вспомогательной плоскости, на которую осуществляется конформное преобразование эллипса по функции Н.Е. Жуковского;

$\zeta_{vj},\bar{\zeta}_{vj}$ – комплексная и сопряжённая комплексная переменная j-го дискретного вихря во вспомогательной плоскости;

Γ_j – интенсивность j-го дискретного вихря;

N – общее количество дискретных вихрей, размещенных на дугах хвостовой части профиля и на интерцепторе;

$N+1$ – номер стационарного вихря;

$N+2$ – номер вихря, отвечающего за условие Чаплыгина-Жуковского на задней кромке профиля.

Поле скоростей определяется через производную от комплексного потенциала (1)

$$\frac{dw}{d\zeta}\frac{d\zeta}{dz}=\bar{V}=\left\{\frac{1}{2}\left(\bar{V}_\infty-\frac{V_\infty a}{\zeta^2}\right)+\frac{1}{2\pi i}\left[\sum_{j=1}^{N+1}\Gamma_j\left(\frac{1}{\zeta-\zeta_{vj}}-\frac{1}{\zeta-\frac{a^2}{\bar{\zeta}_{vj}}}+\frac{1}{\zeta}\right)+\frac{\Gamma_{N+2}}{\zeta}\right]\right\}\frac{d\zeta}{dz}=u-iv, \qquad (2)$$

где $z=x+iy$ – комплексная переменная в физической плоскости;

u и v – компоненты скорости вдоль осей OX и OY, соответственно (рисунок 3).

Неизвестные интенсивности вихрей Γ_j вычисляются выполнением условий непротекания, а интенсивность стационарного вихря Γ_{N+2} находится из условия конечности скорости на задней кромке интерцептора.

Задача сводится к решению системы линейных алгебраических уравнений (СЛАУ), физический смысл которой состоит в удовлетворении условий непротекания в контрольных точках и выполнении условия Чаплыгина-Жуковского на задней кромке профиля и интерцептора. На основании комплексного потенциала (1) составляется СЛАУ вида

$$\mathbf{A}\cdot\mathbf{\Gamma}=\mathbf{R}, \qquad (3)$$

в которой матрица аэродинамического влияния $\mathbf{A}$ заполняется на основании коэффициентов при Γ_j формулы (2);

Γ – вектор-столбец неизвестных интенсивностей;
R – вектор-столбец правых частей, образованный коэффициентами, полученными из первых двух слагаемых формулы (2).

СЛАУ (3) может решаться стандартными методами. В данной работе был использован итерационный метод решения СЛАУ, в основе которого лежит метод триангулиаризации, который реализован в стандартной процедуре языка Фортран. После нахождения неизвестных интенсивностей Γ_j можно построить поле скоростей по формуле (2), в которой

$$u = \mathrm{Re}\left[\overline{V}\right]; \quad v = -\mathrm{Im}\left[\overline{V}\right].$$

Опираясь на ЧАМ, в разработанной вычислительной Фортран-программе написана подпрограмма, реализующая построение линий тока течения. Для построения линий тока проводится интегрирование дифференциального уравнения линий тока

$$\frac{dx}{u} = \frac{dy}{v} = \tau$$

где τ – произвольная константа, задающая шаг интегрирования.

Интегрирование этого уравнения производится методом Эйлера первого порядка

$$x_{n+1} = x_n + \tau u^{(n)}$$

$$y_{n+1} = y_n + \tau v^{(n)}$$

где $(x_{n+1}, y_{n+1}), (x_n, y_n)$ координаты (*n+1*)-ой и *n*-ой расчётных точек, соответственно;

$u^{(n)}$ è $v^{(n)}$ – компоненты скорости в *n*-ой расчётной точке.

В программе реализовано построение нулевых линий тока, которые проходят через точки торможения. Точки торможения найдены по методу Мюллера [10] (рисунок 4).

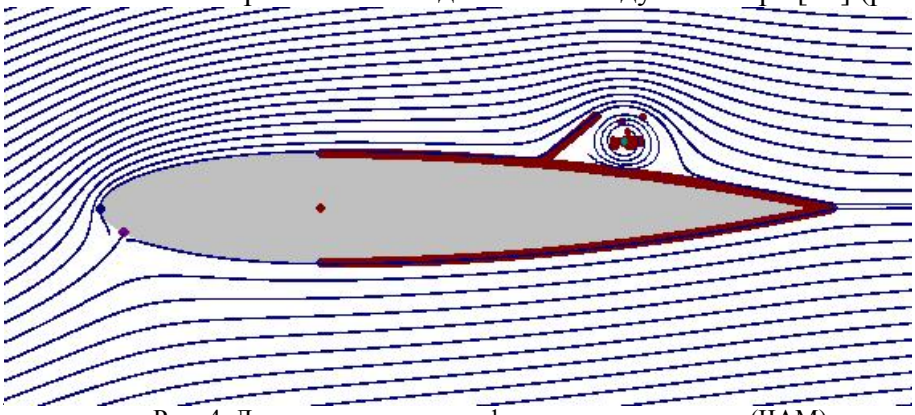


Рис. 4. Линии тока около профиля с интерцептором (ЧАМ).

В модели используется гипотеза стационарности внешнего вихря, расположенного за интерцептором, интенсивность которого находится из условия конечности скорости на задней кромке интерцептора, а координаты рассчитываются методом оптимизации – минимизации целевой функции скорости по двум проектным параметрам

$$\min\left(f(\mathbf{X})\right), \quad \mathbf{X} = \begin{Bmatrix} x_{\text{vortex point}} \\ y_{\text{vortex point}} \end{Bmatrix},$$

в качестве, которых принимаются координаты стационарного вихря.

В качестве целевой функции в методе оптимизации используется модуль полной скорости течения в действительной плоскости за исключением компонент скорости индуцируемых самим стационарным вихрем, т.е. исключена самоиндукция.

$$f = V_{\text{vortex point}} = \left|\overline{V}\right| - \left|\overline{V}_{\text{vortex}}\right|$$

Решается задача минимизации целевой функции с ограничениями, в качестве которых используются границы области течения за интерцептором. Вихрь принимается

стационарным, если модуль полной скорости течения в действительной плоскости за исключением компонент скорости индуцируемой самим вихрем не превышает 1% от скорости набегающего внешнего потока.

Оценка точности расчёта подъёмной силы пластины с интерцептором выполнена с помощью экстраполяции по Ричардсону и методу наименьших квадратов (МНК) (рисунок 5). Формулу экстраполяции по Ричардсону для значений коэффициентов подъёмной силы, рассчитанных при использовании N и $2N$ граничных элементов можно записать в виде

$$C_{ya(R)} = 2C_{ya(2N)} - C_{ya(N)}, \qquad (4)$$

где $C_{ya(R)}, C_{ya(2N)}, C_{ya(N)}$ – значения коэффициентов подъёмной силы, рассчитанное для экстраполяции по Ричардсону и для $2N$ и N граничных элементов, соответственно.

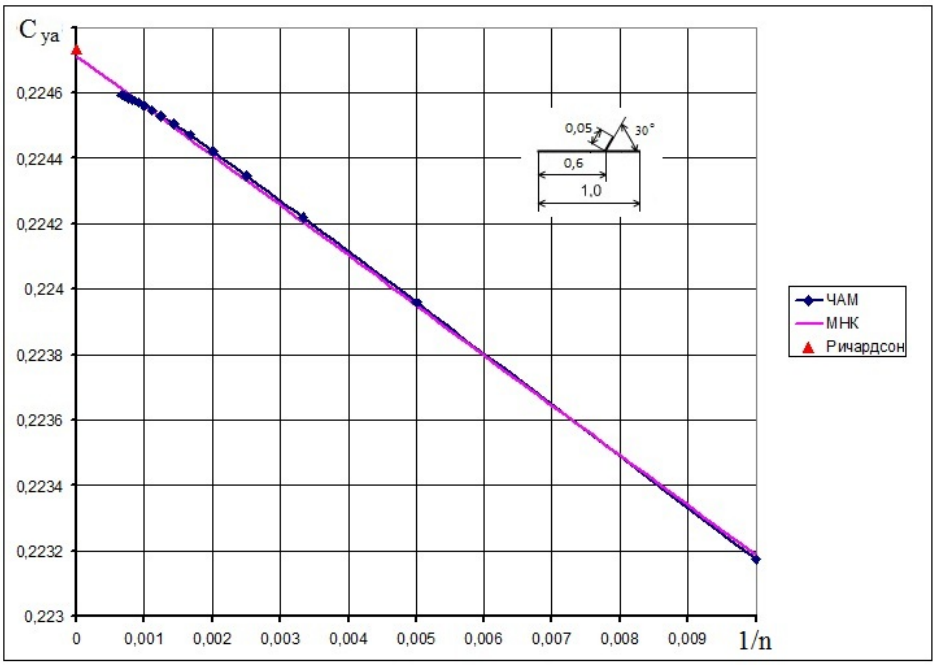


Рис. 5. Применение экстраполяции по Ричардсону для оценки точности расчёта коэффициента подъёмной силы пластины с интерцептором.

В таблице 1 приведены значения коэффициента подъёмной силы, рассчитанные для $N = 400$ и экстраполяционные значения, полученные по методу Ричардсона и МНК, а также значения среднего квадратического отклонения S_x и относительной погрешности δ, которые рассчитывались по формулам

$$S_x = \sqrt{\sum_{j=1}^{n} \frac{\left(C_{ya\,j} - C_{ya(MHK)\,j}\right)^2}{n-1}}; \quad \delta = \frac{C_{ya(R)} - C_{ya(MHK)|N=\infty}}{C_{ya(MHK)|N=\infty}} \cdot 100\%,$$

где использованы обозначения $C_{ya\,j}$, – расчётное значение коэффициента подъёмной силы для $N = const$; $C_{ya(MHK)\,j}$, $C_{ya(MHK)|N=\infty}$ – интерполяционное и аппроксимационное значение коэффициента подъёмной силы по МНК для $N = const$ и $N = \infty$, соответственно; n – общее количество расчётов при различных значениях N.

Таблица 1. Точность расчёта подъёмной силы пластины с интерцептором при наличии стационарного вихря в рециркуляционной зоне.

α, *град*	$C_{ya(N=400)}$	$C_{ya(R)}$	$C_{ya(MHK)}$	S_x	δ, %
-5	-0,83297	-0,83262	-0,83263	0,0010	0,0021
0	-0,30547	-0,30510	-0,30512	0,0011	0,0060
5	0,22435	0,22473	0,22472	0,0012	0,0085

Из таблицы 1 и рисунка 5 следует, что количество граничных элементов равное N=400 при применении экстраполяции по Ричардсону даёт высокую точность расчётов. На рисунке 6 показаны предельные углы отклонения интерцептора в модели стационарного вихря.

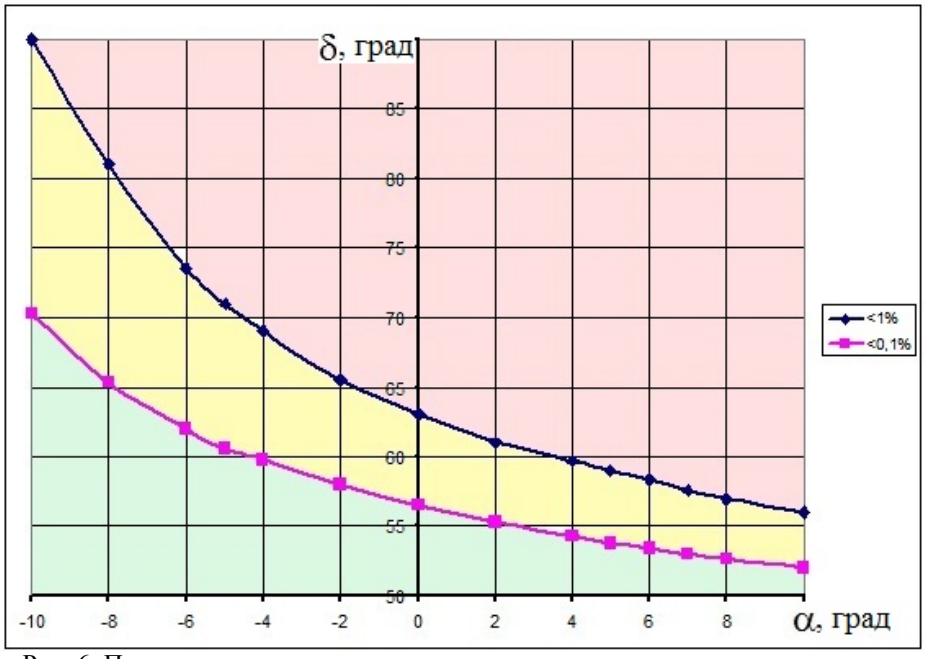


Рис. 6. Предельные углы отклонения интерцептора в модели стационарного вихря.

В рамках данной работы было исследовано изменения коэффициента подъёмной силы профиля с интерцептором в зависимости от различных геометрических параметров конфигурации. Как известно [6, 9], коэффициент подъёмной силы можно вычислить по следующей формуле

$$C_{ya}\frac{2Y_a}{\rho|V_\infty|^2 b} = -2\overline{\Gamma} \qquad (5)$$

В формуле (5) циркуляция определяется прямым интегрированием по замкнутому контуру вокруг конфигурации

$$\Gamma = \mathrm{Re}\left[\oint \bar{V}dz\right] - \Gamma_{N+2} = \oint\left(udx + vdy\right) - \Gamma_{N+2}$$

$$\overline{\Gamma} = \frac{\Gamma}{|V_\infty| b}.$$

На рисунке 7 представлены зависимость коэффициента подъёмной силы от угла атаки для различных углов отклонения интерцептора. Угол отклонения интерцептора существенно снижает подъёмную силу конфигурации особенно на интервале от 0 до 15 градусов.

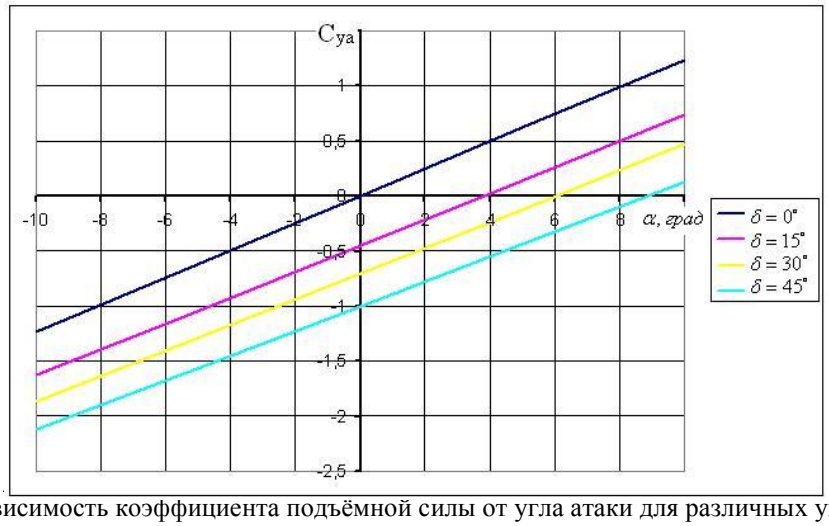


Рис. 7. Зависимость коэффициента подъёмной силы от угла атаки для различных углов отклонения интерцептора.

На рисунке 8 представлены зависимость коэффициента подъёмной силы от угла атаки для различных относительных хорд интерцептора. Длина интерцептора существенно снижает подъемную силу конфигурации.

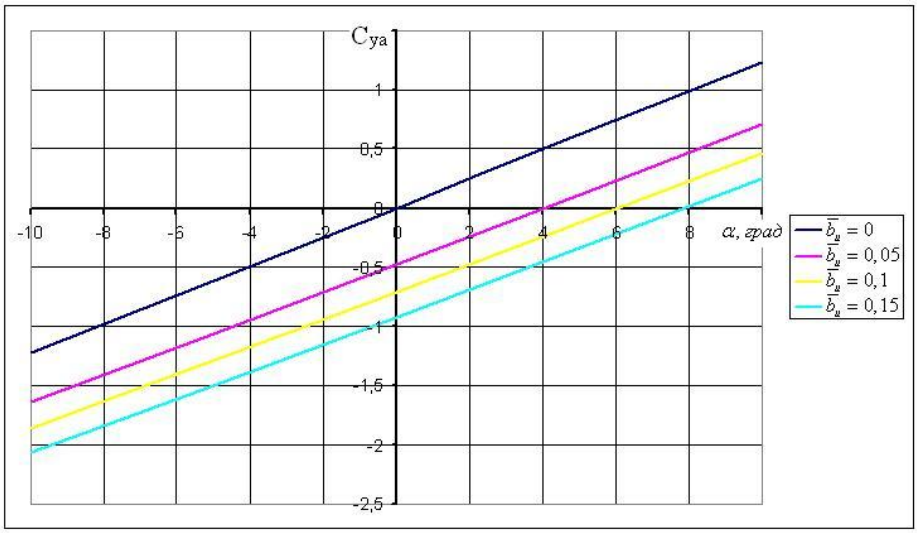


Рис. 8. Зависимость коэффициента подъёмной силы от угла атаки для различных относительных хорд интерцептора.

На рисунке 9 представлены зависимость коэффициента подъёмной силы от угла атаки для различных толщин профиля. Из рисунка видно, что с увеличением толщины профиля возрастает производная подъёмной силы.

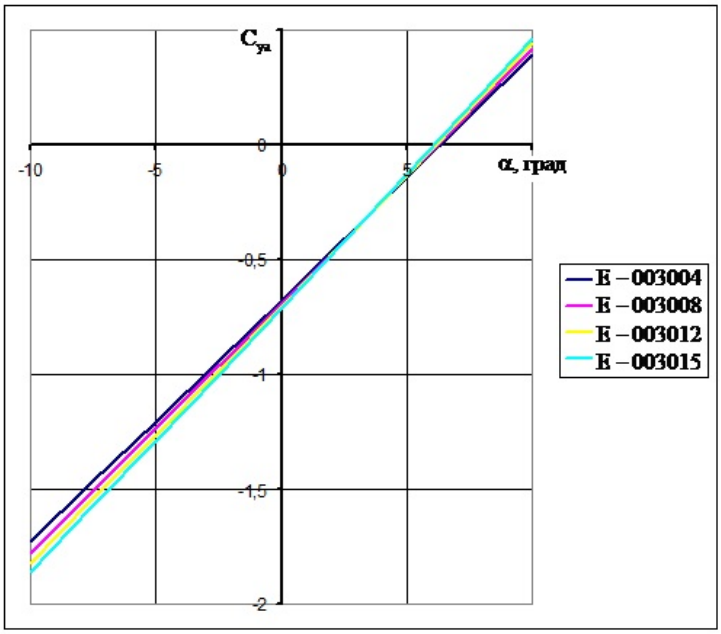


Рис. 9. Зависимость коэффициента подъёмной силы от угла атаки для толщин профиля.

В работе была проведена оценка влияния интерцептора на подъёмную силу комбинации, которая проводилась по следующей формуле

$$\Delta C_{ya} = C_{ya\ddot{i}\eth,\grave{e}} - C_{ya\ddot{i}\eth},$$

где $C_{ya\ddot{i}\eth,\grave{e}}, C_{ya\ddot{i}\eth}$ – коэффициенты подъёмной силы комбинации профиля с интерцептором и изолированного профиля, соответственно.

На рисунке 10 представлено влияние интерцептора на подъёмную силу комбинации в зависимости от угла атаки для различных углов отклонения интерцептора. Можно видеть, что угол отклонения интерцептора существенно снижает подъёмную силу конфигурации.

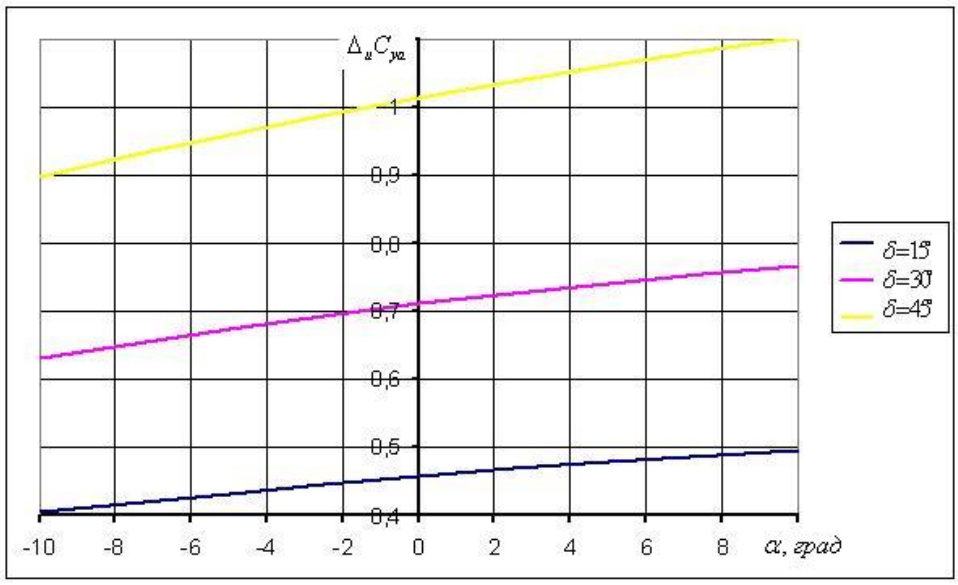


Рис. 10. Влияние интерцептора на подъёмную силу комбинации в зависимости от угла атаки для различных углов отклонения интерцептора.

На рисунке 11 представлено влияние относительной хорды интерцептора на подъёмную силу комбинации в зависимости от угла атаки. Длина интерцептора существенно снижает подъемную силу конфигурации.

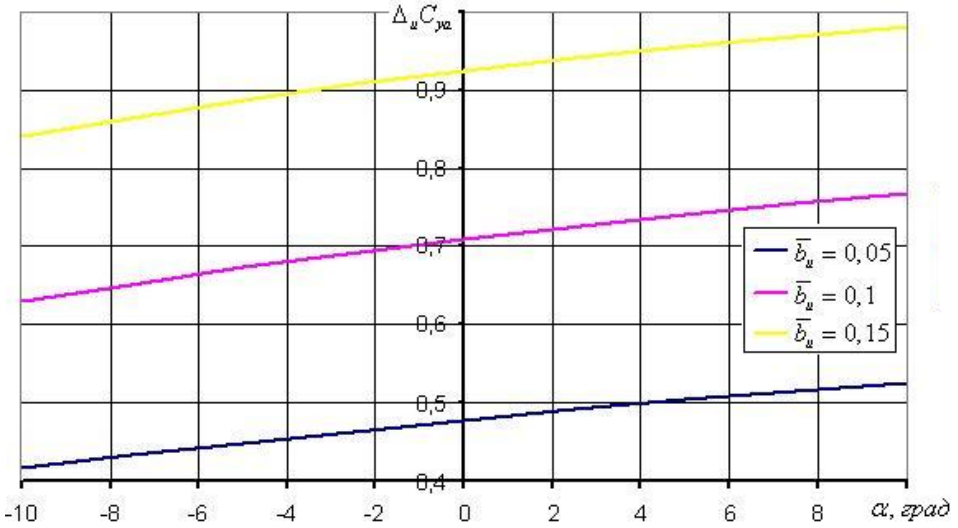


Рис. 11. Влияние относительной хорды интерцептора на подъёмную силу комбинации в зависимости от угла атаки для различных углов отклонения интерцептора.

Вычислительная программа также позволяет рассчитывать производную коэффициента подъёмной силы профиля с интерцептором по углу атаки. Результаты представлен на рисунках 12 и 13.

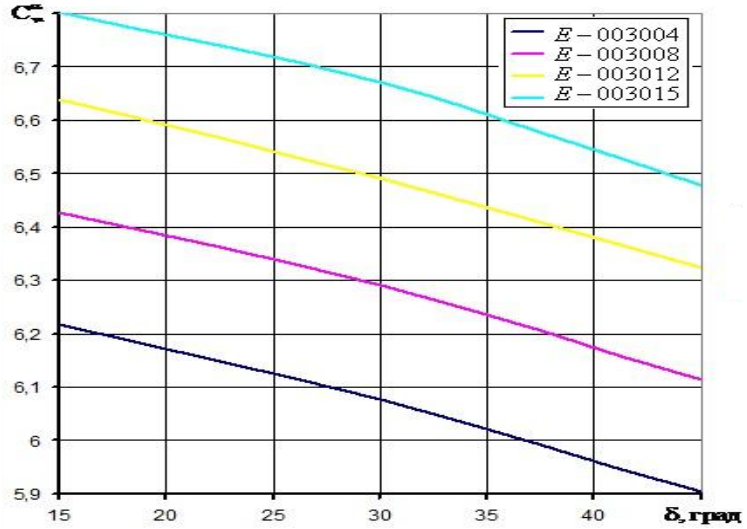


Рис. 12. Производная коэффициента подъемной силы профиля с интерцептором по углу атаки для различных толщин профиля.

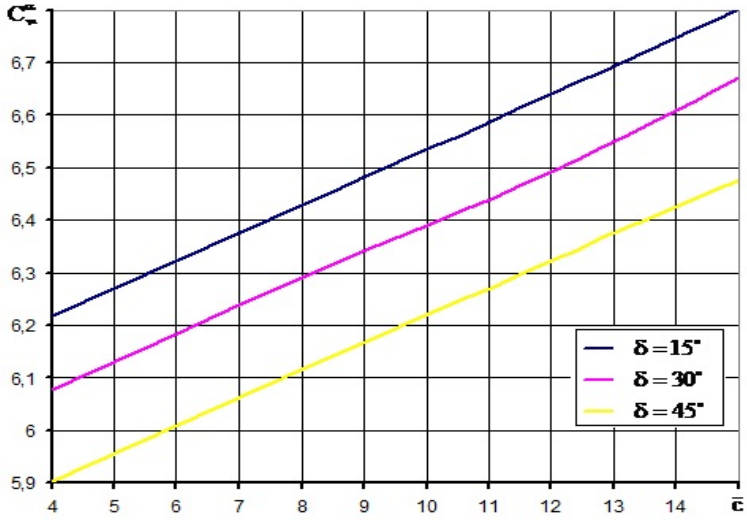


Рис. 13. Производная коэффициента подъемной силы профиля с интерцептором по углу атаки для различных углов отклонения интерцептора.

Из рисунков легко заметить, что наибольшая производная коэффициента подъемной силы по углу атаки реализуется для более толстого профиля, но снижается при отклонении интерцептора.

В рамках данной работы было исследовано распределение давления по поверхности профиля и интерцептора в присутствии стационарного вихря. Как известно, коэффициент давления можно вычислить по следующей формуле $C_p = 1 - \bar{V}^2$, где $\bar{V} = \frac{V}{V_\infty}$.

В соответствии с ЧАМ вычисление коэффициента давления производилось на расстоянии вычислительного радиуса от поверхности хвостовой части профиля и интерцептора.

На рисунке 14 представлены распределения давления по профилю и интерцептору в присутствии стационарного вихря. Получено, что в отрывной зоне находится область разряжения, где давление резко уменьшается.

В работе с целью верификации результатов, полученных на основании разработанной модели, были выполнены расчёты в вычислительном пакете ANSYS CFX. Так поле давлений, рассчитанное в пакете ANSYS CFX (Рисунок 15), также показало наличие области разряжения за интерцептором.

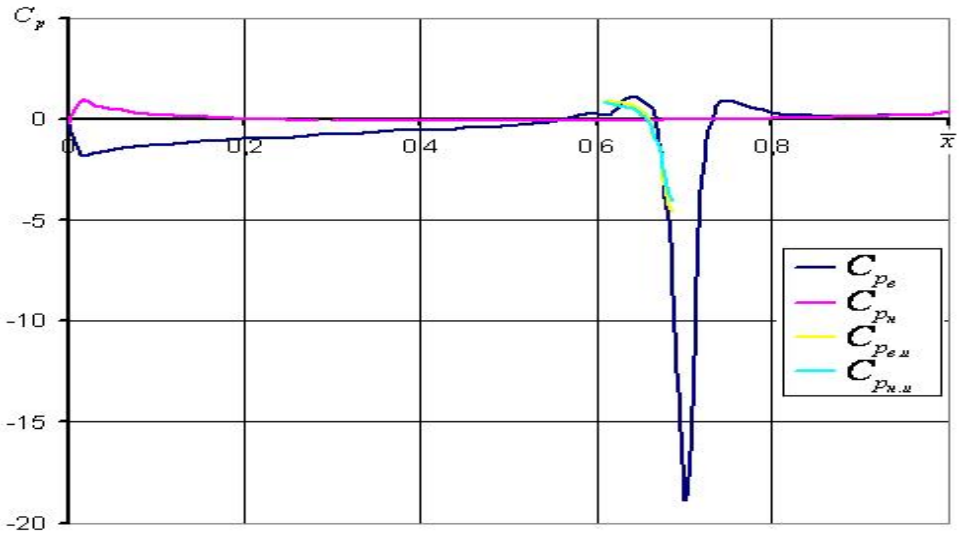


Рис. 14. Распределение давления по поверхности профиля и интерцептора в присутствии стационарного вихря.

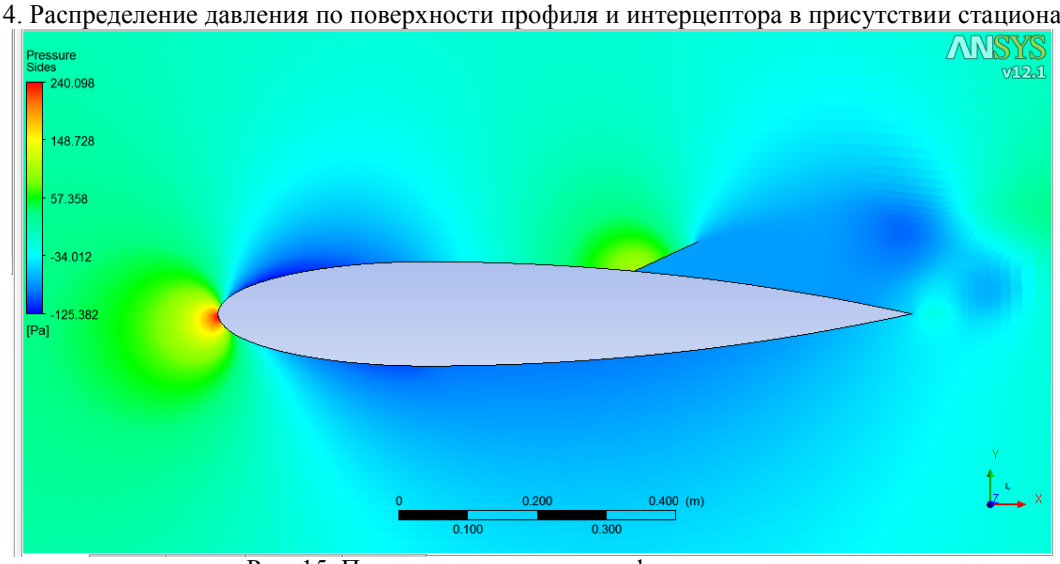


Рис. 15. Поле давлении около профиля с интерцептором.

На рисунке 16 представлены распределения скорости по профилю и интерцептору в присутствии стационарного вихря. Получено, что в отрывной зоне как и ожидалось, находится область заторможенного течения, где давление резко уменьшается. Так же было построено поле скоростей в пакете ANSYS CFX (рисунок 17), которое также показало наличие области заторможенного течения за интерцептором.

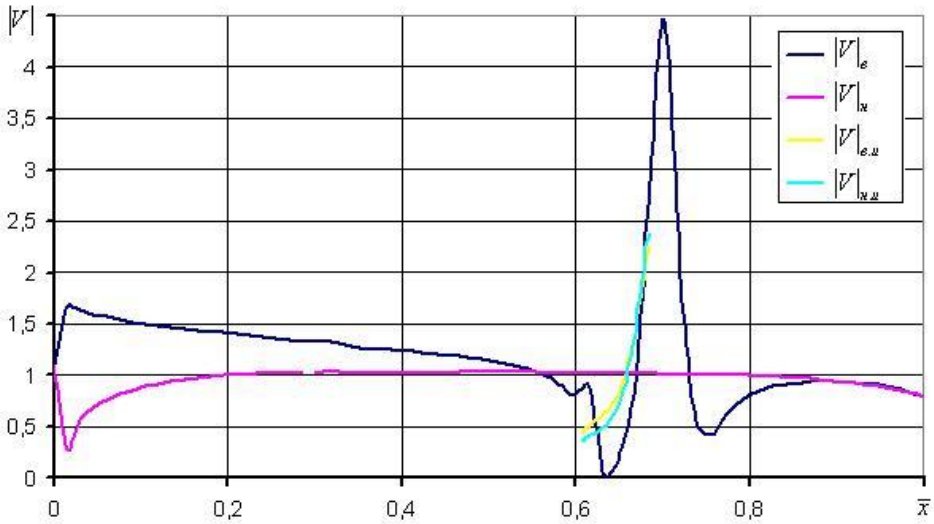


Рис. 16. Распределение скорости по поверхности профиля и интерцептора в присутствии стационарного вихря.

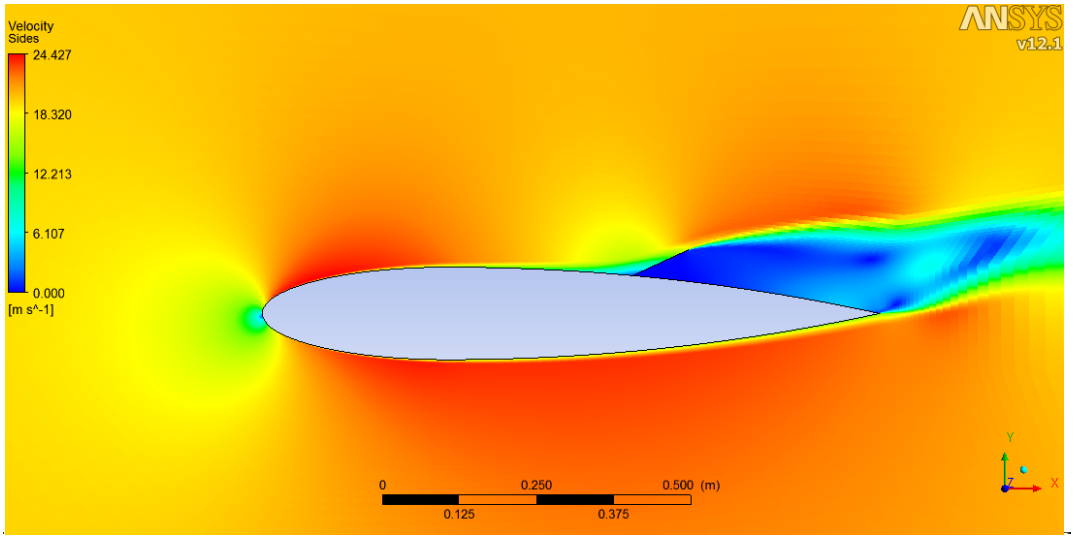


Рис. 17. Поле скоростей около профиля с интерцептором.

В работе было проведено сравнение результатов расчёта подъёмной сил профиля с интерцептором, полученных с помощью численно-аналитического метода и в пакете ANYS CFX (рисунок 18), которое дало

$$\alpha \in \left[0^{\circ};5^{\circ}\right] \quad \delta = \frac{\left|C_{ya\,ANSYS} - C_{ya\,ЧАМ}\right|}{\left|C_{ya\,ANSYS}\right|} < 10\% \cdot$$

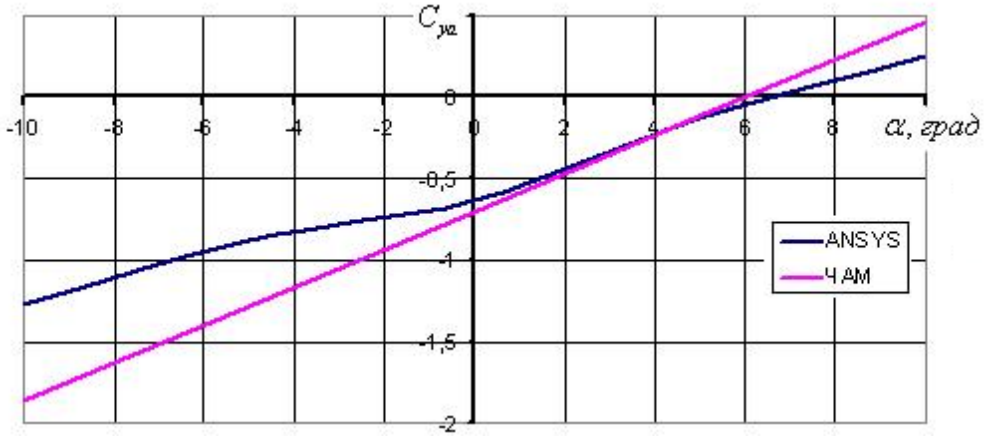


Рис. 18. Сравнение ЧАМ с ANSYS CFX.

В результате исследования была разработана математическая модель течения с циркуляцией вокруг профиля с интерцептором в присутствии стационарной отрывной зоны. Было получено, что подъёмная сила комбинации уменьшается при увеличении относительной длины интерцептора и его угла отклонения. Определены предельные углы отклонения интерцептора, для которых выполняется условие стационарности. Проведено сравнение результатов, полученных с помощью разработанной Фортран-программы, и результатами расчётов, полученных в пакете ANSYS CFX.

Результаты исследования могут быть использованы для инженерных расчётов подъёмной силы аэродинамических профилей с интерцептором.

Литература

1. Woods, L.C. Theory of Aerofoil Spoilers [Text] /L.C. Woods //ARC R&M, No.2969. – London, 1956. – 21 p.

2. Barnes, C.S. A Developed Theory of Spoilers on Aerofoils [Text] /C.S. Barnes //ARC CP, No.887. 1966. – 78 p.

3. Богатырёв, В.В. Метод расчёта нестационарного обтекания профиля с интерцептором и его аэродинамические характеристики [Текст] /В.В. Богатырёв //Учёные записки ЦАГИ, Т.29, №3-4. – 1998.

4. Редькина, К.В. Математическая модель течения около цилиндра с плоскими пластинами при наличии стационарных отрывных зон [Текст] /К.В. Редькина //Электронный журнал “Труды МАИ”, Вып. №45 от 7.06.2011.

5. Бартеньев, О.В. Фортран для профессионалов. Математическая библиотека IMSL: Ч. 2. [Текст] /О.В. Бартеньев. – М.: Диалог-МИФИ, 2001. – 320с.

6. Фролов, В.А. Метод построения потенциального течения около симметричного профиля, образованного дугами эллипса и окружности [Текст] /В.А. Фролов //Управление движением и навигации летат. аппаратов: Сб. тр. XII Всерос. науч.-техн. семинара по управлению движением и навигации летат. аппаратов. – Самара: СГАУ. 2006. – С.260-265.

7. Редькина, К.В. Математическая модель течения около пластины с интерцептором [Текст] /К.В. Редькина, В.А. Фролов //Тр. 54-й науч. конф. МФТИ «Проблемы фундаментальных и прикладных естественных и технических наук в современном информационном обществе»: Аэромеханика и летательная техника. М.: МФТИ – 2011. – С. 91-92.

8. Редькина, К.В. Численно-аналитический метод решения задачи обтекания пластины со щелевыми предкрылком и закрылком [Текст] /К.В. Редькина, В.А. Фролов //Тр. XV Междунар. симп. «Методы дискретных особенностей в задачах математической физики» (МДОЗМФ-2011), Харьков-Херсон, 2011. – С.378-381.

9. Милн-Томсон, Л. Теоретическая гидродинамика : Пер. с англ. [Текст] /Л.М. Милн-Томсон – М.: Мир, 1964. – 655 с.

10. Фролова, К.В. Применение метода Мюллера для нахождения точки торможения в циркуляционных потенциальных течениях [Текст] /К.В. Фролова //Тез. докл. XXXIII Самар. обл. студ. науч. конф., Самара, 2007. – Издательский дом «Федоров» – С.215.

УДК 621.9-05

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ БУРИЛЬНОЙ УСТАНОВКИ

Н.Ю.Санкин

Ульяновский государственный университет

Ключевые слова: математическая модель бурильной установки, переходный процесс в колонне труб

Аннотация. В статье рассматриваются вопросы, связанные с построением ма-тематической модели бурильной установки. Частотным методом получена оценка уровня колебаний упругих элементов бурильной установки. Рассматривается метод снижения уровня крутильных колебаний бурильной установки, включающие длинные стержни (например, в бурении − колонна труб, в глубиннонасосных установках – штанги и т. д.) как системы с распределенными параметрами.

Введение

Переходные процессы в электроприводах буровых и глубиннонасосных устано-вок, крутильные колебания стержней, представляющих ответственные части конст-рукции, определяются решениями дифференциальных уравнений в частных произ-водных гиперболического типа. Аналитические решения этих уравнений находятся с помощью метода Фурье или метода Даламбера. Решения получаются в виде беско-нечного ряда падающих и отраженных волн. Для получения решения, в соответствии с этими методами необходимо произвести сложные математические выкладки. Так-же трудоёмко получить решения в общем виде, удобном для анализа. Широкое рас-пространение получил операционный метод решения телеграфных и волновых уравнений с постоянными коэффициентами и некоторых типов уравнений в частных производных. Однако, при определении граничных условий для систем, включа-ющих нелинейные элементы, решение уравнений операционным методом не пред-ставляется возможным. Метод конечных разностей является наиболее распрост-раненным численным методом для решения дифференциальных уравнений. Подроб-ный анализ метода конечных разностей дан в литературе [1].

В работе [2] описывается численный метод, получивший название метод бегу-щей волны для расчета переходных процессов в линиях электропередачи. Предло-женный метод является вариантом определения переходных процессов, использовав-шимся при расчетах в [3] и [4]. Недостатком этого метода является то, что определе-ние переходного процесса сводится к решению самостоятельной задачи.

В [5] предлагается способ решения уравнений для линий электропередач на компьютере с использованием метода Рунге-Кутта. В [6] решение переходных процессов производится методом характеристик, в соответствии с которым процессы представляются в виде блуждающих волн.

В работе [7] изложен метод исследования систем с распределенными пара-метрами, явления в которых описываются дифференциальными уравнениями в част-ных производных гиперболического типа. Сущность разработанной методики заклю-чается в приведении систем с распределенными параметрами к импульсным систе-мам и использовании в качестве математического аппарата дискретного преоб-разования Лапласа. Благодаря работам [8] глубоко разработана теория как линейных, так и нелинейных импульсных систем. Приведение систем с распределенными параметрами к импульсным цепям упрощает математические выкладки, при этом происходит изменение структуры уравнений. Характеристические уравнения из трансцендентной формы переходят в алгебраическую. Для исследования переходных процессов в различных системах разработаны два метода [8].

Первый, аналитичес-кий, позволяет получить решение в общем виде, а так же проводить анализ и опреде-лять влияние отдельных параметров на ход переходного процесса. Второй, числен-ный, является универсальным, используется для сложных систем, включающих нелинейные элементы. Решения при этом получаются в виде рекуррентных соотно-шений, допускающих реализацию на ЭВМ. Однако в ряде случаев решения выглядят громоздкими и сложными, а полученные алгоритмы не рациональными, что не позволяет быстро решать ряд задач.

Разработанный в [9] метод применим к решению различных систем, опи-сываемых волновыми уравнениями. С помощью него быстро и эффективно произ-водятся расчеты переходных процессов в длинных линиях электропередач, электро-механических переходных процессов электропривода, включающего звено с распре-деленными параметрами. Метод состоит в том, что дифференциальные уравнения в частных производных преобразуются по Лапласу. В полученные уравнения входят начальные условия, которые наряду с заданными силами также являются возмущающими воздействиями. Решение преобразованных по Лапласу уравнений осуществляется при помощи метода конечных элементов (МКЭ), являющегося вариантом метода Бубнова-Галеркина-Ритца [10], в результате чего осуществляется построение амплитудно-фазовой частотной характеристики (АФЧХ). Построение АФЧХ, как правило, всегда возможно. Поскольку все особые точки соответству-ющих выражений, благодаря учету рассеяния энергии, лежат в левой полу-плоскости, то обратное преобразование Лапласа осуществляется путем использо-вания построенных АФЧХ. Кроме того, как смоделированные АФЧХ, так и АФЧХ снятые экспериментально, могут служить для построения линейных математических моделей различных систем. Особенностью построения вариантов МКЭ указанного выше класса задач является то, что за основу берется смешанный вариационный принцип для соответствующих величин, преобразованных по Лапласу, куда входят начальные условия [10]. В некоторых случаях, например для электрических линий или стержней, после преобразования по Лапласу получаются обыкновенные дифференциальные уравнения и при постоянных по длине параметрах, соответст-вующие формулы МКЭ получаются точным интегрированием. Первые два члена в разложениях этих формул по безразмерному частотному параметру, совпадают с соответствующими формулами, полученными из вариационных соображений. Ис-пользуя этот вариационный принцип, переходят к той или иной выбранной вели-чине, возможен также и смешанный подход. Выбирая соответствующий вариант МКЭ, исключаются ошибки и разного рода затруднения.

Спектральный метод решения задачи построения переходных процессов в сложных электромеханических системах с распределенными параметрами, основан на вариационном принципе, под которым понимается эквивалентность решения краевой задачи условию стационарности соответствующего функционала. Для пря-мого вариационного метода достаточно иметь в распоряжении функционал, которому искомое решение сообщает стационарное, а не обязательно экстремальное значение. Традиционно при реализации прямых методов объем вычислений резко возрастает с увеличением числа искомых, то есть с увеличением номера прибли-жений. В предлагаемом методе удалось достичь независимости объема вычислений от требуемой точности, что позволяет моделировать достаточно сложные электро-механические системы с распределенными параметрами.

Различные модификации МКЭ основаны на прямых методах типа Бубнова-Галеркина-Ритца [11]. Для решения нестационарных задач применяется дискретный метод Фурье. При этом имеют место трудности при расчете системы собственных функций, нахождении коэффициентов рядя Фурье и его суммы для системы в целом. Значительное уменьшение вычислительных затрат происходит за счет применения быстрого преобразования Фурье с учетом периодичности показательной функции мнимого аргумента $e^{i\omega t}$. Хотя это и не ведет к существенным упрощениям вычисли-тельных схем. Предлагаемый метод не требует проведения. Преобразования Фурье в целом, вычисления собственных векторов и собственных значений, а требует, как уже было отмечено, построения АФЧХ с учетом

начальных условий для отдельных точек системы с последующим численным преобразованием, что существенно упрощает алгоритм по сравнению с известными ранее методами.

Уравнения движения электропривода и крутильных колебаний стержня сплошного сечения преобразуются по Лапласу. Для преобразованного дифференци-ального уравнения крутильных колебаний стержня решается краевая задача, заклю-чающаяся в нахождении преобразованных по Лапласу краевых крутящих моментов, как функций краевых углов поворота. Затем составляются уравнения равновесия узлов, которые представляют собой систему уравнений для неизвестных узловых углов поворота. Поскольку соответствующие коэффициенты получаются точным интегрированием, то длина участков закручиваемого стержня неограниченна, а число участков может быть тоже любым. Решая полученную систему, при $p = j\omega$, где p – параметр преобразования Лапласа; ω – частота, строим АФЧХ для инте-ресующих сечений стержня. Если привод устойчив, то все особые точки соответ-ствующих передаточных функций лежат левее мнимой оси и обратное преобра-зование можно осуществить, полагая $p = j\omega$, т. е. используя построенные АФЧХ. АФЧХ строятся от воздействия возмущающих сил, затем численным интегри-рованием осуществляется обратное преобразование Лапласа. Между экстремальны-ми точками АФЧХ и коэффициентами соответствующих членов ряда существует однозначная связь, которая используется для осуществления обратного преобразова-ния Лапласа.

Построение расчетной модели

Уравнение движения электропривода, имеет следующий вид:

$$J\frac{d\omega}{dt} = M_{д} - M_{н0}, \tag{1}$$

где J – момент инерции электродвигателя; $M_{н}$ – момент нагрузки (сопротивления), т. е. момент в начале валопровода; ω – скорость вращения; $M_{д}$ – момент двигателя.

При линейной механической характеристике момент двигателя имеет вид:

$$M_{д} = a - b\omega, \tag{2}$$

где $a = M_0$ – начальный момент двигателя; b – угловой коэффициент механической характеристики.

Подставив (2) в (1), исключив постоянную составляющую a и преобразовав полученное уравнение по Лапласу, найдем:

$$J\left(p^2\varphi_{н} - p\omega_0\right) = -bp\varphi_{н} - M_{н}, \tag{3}$$

где $p\varphi_{н} = \omega$ – угловая скорость двигателя; $\varphi_{н}$ – угол закручивания вала в начале участка; ω_0 – угловая скорость двигателя в установившемся режиме; $M_{н} = M_{н0} - a$ – момент нагрузки на двигатель.

Момент нагрузки на двигатель действует со стороны вала. Рассмотрим опре-деление момента нагрузки. За начало отсчета берем начальные углы поворота $\varphi_0 = \varphi_0\left(x\right)$ в каждом сечении вала.

Рассмотрим крутильные колебания прямолинейного вала без учета рассеяния энергии, полагая заданным $M_{н}$. Вал (колонна труб) бурильной установки рассматри-вается как стержень прямолинейного сечения. Дифференциальное уравнение кру-тильных колебаний вала может быть записано в виде [47]:

$$\frac{\partial}{\partial x}\left(GJ_k\frac{\partial\varphi}{\partial x}\right) - J_{\rho}\frac{\partial^2\varphi}{\partial t^2} = m_k\left(s,t\right), \tag{4}$$

где φ – угол закручивания стержня; GJ_k – жесткость при кручении; J_ρ – момент инерции единицы длины стержня относительно центра кручения; $m_k(s,t)$ – распределенный по длине стержня крутящий момент.

Полагая жесткость на кручение GJ_k постоянной, из уравнения (4) получим следующее, преобразованное по Лапласу, дифференциальное уравнение:

$$GJ_k \frac{d^2\varphi}{dx^2} - p^2 J_\rho \varphi = m_k(s,p) + pJ_\rho \varphi_0 + J_\rho \omega_0,$$

где $m_k(s,p)$ – преобразованный по Лапласу распределенный крутящий момент.

Вначале рассмотрим однородное уравнение, когда $\varphi_0 = 0$, $\omega_0 = 0$, то есть уравнение

$$\frac{d^2\varphi}{d\zeta^2} + d\varphi = 0, \tag{5}$$

где $\zeta = \frac{x}{l}$ – новый аргумент; l – длина стержня.

Для построения АФЧХ положим $p = j\omega$, тогда $d = \frac{J_\rho \omega^2 l^2}{GJ_k}$. Следовательно, решение уравнения (5) запишется в виде:

$$\varphi = C_1 \cos\alpha\zeta + C_2 \sin\alpha\zeta, \tag{6}$$

где $\alpha = \sqrt{d}$. Постоянные интегрирования C_1 и C_2 находим из начальных условий $\varphi = \varphi_н$ и $M = M_н$.

Тогда:

$$C_1 = \varphi_н,\ C_2 = \frac{M_{кн} l}{GJ_k \alpha},$$

где $M_k = \frac{GJ_k}{l}\frac{d\varphi}{d\zeta}$ – крутящий момент.

Рассмотренному решению соответствует матрица фундаментальных решений или матрица переноса:

$$K = \begin{vmatrix} K_{\varphi\varphi} & \frac{l}{GJ_k} K_{\varphi M} \\ \frac{GJ_k}{l} K_{M\varphi} & K_{MM} \end{vmatrix}, \tag{7}$$

где $K_{\varphi\varphi} = \cos\alpha\zeta$; $K_{M\varphi} = -\alpha \sin\alpha\zeta$; $K_{\varphi M} = \frac{\sin\alpha\zeta}{\alpha}$; $K_{MM} = K_{\varphi\varphi}$.

Пользуясь матрицей переноса (7), запишем соотношения метода начальных параметров

$$V(\zeta) = K(\zeta)V(0) + \sum K(\zeta - \zeta_i)V(\zeta_i) + \sum l \int_{s_1}^{s_2} K(\zeta - s)V(s)ds, \tag{8}$$

где $V(0) = \left|\varphi_н, M_{кн}\right|^T$; $V(\zeta_i) = \left|0,\ M_{кi}\right|^T$; $V(s) = \left|0, m_к(s,p) + pJ_\rho\varphi_0 + J_\rho\omega_0\right|^T$;

s – переменная интегрирования.

Рассмотрим первую строку матричного соотношения (8), учитывая, что $\varphi_н$ – угол закручивания стержня в начале участка, $\varphi_к$ – угол закручивания стержня в конце участка,

$\varphi_0 = 0$, $m_к(s,p) = 0$. Тогда выражения для начального и конечного крутящего моментов будут таковы:

$$M_н = Q_{нк}\varphi_н - R_{нк}\varphi_к + R_{нк}[\varphi_к], \tag{9}$$

$$M_k = Q_{нк}\varphi_к - R_{нк}\varphi_н + R_{нк}[\varphi_н], \tag{10}$$

где $Q_{нк} = \phi\frac{K_{\varphi\varphi}(1)}{K_{\varphi M}(1)}$; $R_{нк} = \phi\frac{1}{K_{\varphi M}(1)}$; $\phi = \frac{GJ_к}{l}$;

$$[\varphi_n] = \frac{-J_\rho\omega_0 l^2}{GJ_к\alpha}\int_0^1 sin(\alpha s)\,ds = \frac{-J_\rho\omega_0 l^2}{GJ_к}\frac{1-cos\,\alpha}{\alpha^2};$$

$$[\varphi_n] = \frac{-J_\rho\omega_0 l^2}{GJ_к\alpha}\int_0^1 sin(\alpha(1-s))\,ds = \frac{-J_\rho\omega_0 l^2}{GJ_к}\frac{1-cos\,\alpha}{\alpha^2}.$$

Для того, что бы учесть рассеяние энергии, надо модуль сдвига умножить на величину $(1+j\omega\gamma)$, где γ – коэффициент рассеяния энергии [46].

Рассмотрим систему уравнений, состоящую из уравнений (3), (9), (10):

$$\left.\begin{aligned} J\left((j\omega)^2\varphi_н - j\omega\omega_0\right) &= -bp\varphi_н - M_н \\ M_н &= Q_{нк}\varphi_н - R_{нк}\varphi_к + R_{нк}[\varphi_к] \\ M_к &= Q_{нк}\varphi_к - R_{нк}\varphi_н + R_{нк}[\varphi_н] \end{aligned}\right\} \tag{11}$$

Если привод вращается с угловой скоростью ω_0, и концевое сечение внезапно заклинило, то есть $\varphi_к = 0$, то система уравнений (11) запишется в виде:

$$\left.\begin{aligned} J\left((j\omega)^2\varphi_н - i\omega\omega_0\right) &= -bj\omega\varphi_н - M_н \\ M_н &= Q_{нк}\varphi_н + R_{нк}[\varphi_к] \\ M_к &= -R_{нк}\varphi_н + R_{нк}[\varphi_н] \end{aligned}\right\} \tag{12}$$

Решая систему (12), найдем момент в начале вала:

$$M_н(j\omega) = j\omega\frac{Q_{нк}J\omega_0 + R_{нк}j\omega[\varphi_к]J + R_{нк}[\varphi_к]b}{\left(-J\omega^2 + bj\omega + Q_{нк}\right)}. \tag{13}$$

Момент в концевом сечении:

$$M_к(j\omega) = R_{нк}\frac{J\omega_0 j\omega + R_{нк}[\varphi_к] + Q_{нк}[\varphi_н] + J(j\omega)^2[\varphi_к] + bj\omega[\varphi_н]}{\left(-J\omega^2 + bj\omega + Q_{нк}\right)}. \tag{14}$$

Для случая, когда происходит срыв концевого сечения, то есть когда $M_к = 0$, система уравнений (11) примет вид:

$$\left.\begin{aligned} J\left(-\omega^2\varphi_н - j\omega\omega_0\right) &= -bj\omega\varphi_н - M_н, \\ M_н &= Q_{нк}\varphi_н - R_{нк}\varphi_к + R_{нк}[\varphi_к], \\ Q_{нк}\varphi_к - R_{нк}\varphi_н + R_{нк}[\varphi_к] &= 0. \end{aligned}\right\} \tag{15}$$

Момент в начале вала, согласно системе (15), будет:

$$M_н(j\omega) = -j\omega\left(\frac{Q_{нк}^2 J\omega_0 - R_{нк}^2 J\omega_0 + R_{нк}^2 Jj\omega[\varphi_к] + R_{нк}^2 b[\varphi_н]}{-J\omega^2 Q_{нк} + bj\omega Q_{нк} + Q_{нк}^2 - R_{нк}^2} + \right.$$
$$\left. + \frac{R_{нк} Jj\omega[\varphi_к] Jj\omega Q_{нк} + R_{нк}[\varphi_н] bQ_{нк}}{-J\omega^2 Q_{нк} + bj\omega Q_{нк} + Q_{нк}^2 - R_{нк}^2}\right). \quad (16)$$

Переходный процесс для момента и угла закручивания в начале и конце вала можно построить по формуле

$$u(x,t) = \frac{1}{\pi}\int_0^\infty \mathrm{Re}\left(W(j\omega)e^{j\omega t}\right)d\omega. \quad (17)$$

По приведенным формулам был произведен расчет электромеханического переходного процесса в системе бурового электропривода при заклинивании и срыве долота. Аналогичная задача рассматривалась в работе [7]. В работе [19] исполь-зовалось дискретное преобразование Лапласа [8], причем полагалось, что вал буро-вой установки имеет один участок. Недостатком подхода, описанного в [7], является ограничение на число участков вала, который, в предлагаемом здесь методе, отсутствует.

Результаты численного эксперимента

Четырехдюймовая колонна труб приводится во вращение электродвигателем типа МАД-128-8, $P_{ном}$=160 кВт, $n_{ном}$=735 об/мин, J=26 Н·с^2, $\omega_{нач}$=15,7 1/с, a=2540 Н·м, b=1290 Н·с.

Момент инерции сечения трубы J_k=324·10^{-8} м^4, модуль сдвига G=7,5·10^{10} Н/м^2, момент инерции трубы на 1 погонный метр J_ρ=258·10^{-4} Н^2, длинна колонны труб l=10 м.

Используя уравнение (17) для срыва долота, строим АФЧХ (рис. 1, кривая 1). Найдем экстремальные частоты АФЧХ ω_1=13,1 с$^{-1}$, ω_{1max}=10,1 с$^{-1}$ и амплитуду A_1=77818 $\frac{\text{Н}\cdot\text{м}}{\text{рад}}$.

Передаточная функция, для этого случая имеет вид:

$$W(p) = \frac{K_1 p}{T_{21}^2 p^2 + T_{11} p + 1}, \quad (18)$$

где $T_{21} = 1/\omega_1$; $T_{11} = \left(\left(\omega_1/\omega_{1max}\right) - \left(\omega_{1max}/\omega_1\right)\right)T_{21}$; $K_1 = A_1 T_{11}$.

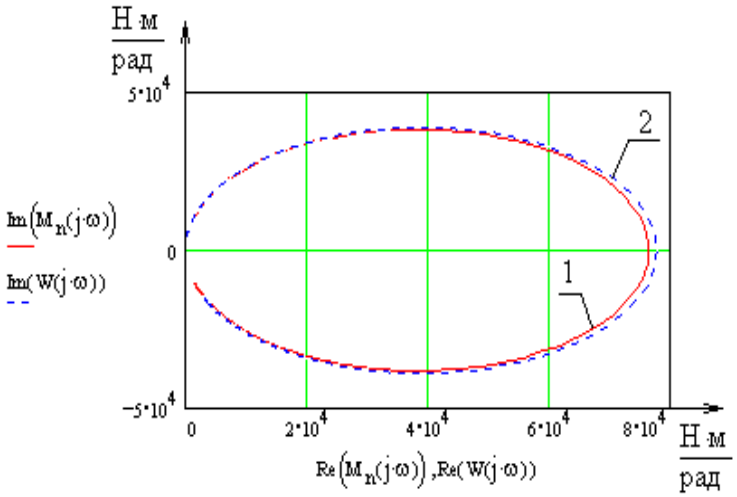


Рис. 1. АФЧХ для срыва долота

АФЧХ, построенная по формуле (18), приведена на рисунке 1, кривая 2. Пере-ходный процесс для этого случая в начале вала, согласно формуле (17), показан на рисунке 2.

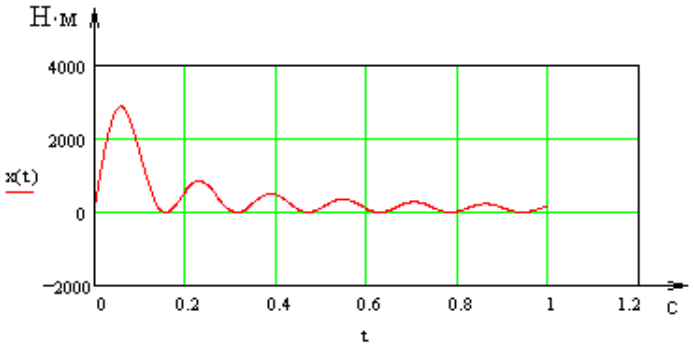


Рис. 2. Переходный процесс для $M_н=M_н(t)$ при срыве долота в начале вала

Переходный процесс в конце участка, согласно формуле (17) показан на рисунке 3.

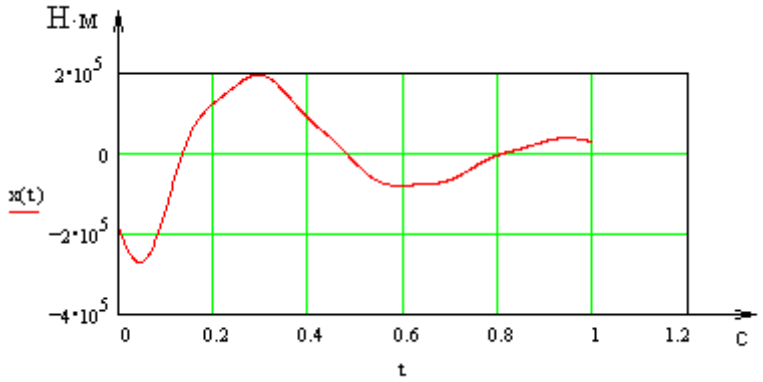


Рис. 3. Переходный про-цесс для $M_н=M_н(t)$ при срыве долота в конце вала

Переходный процесс при заклинивании долота в начале вала $M_н=M_н(t)$ выгля-дит следующим образом:

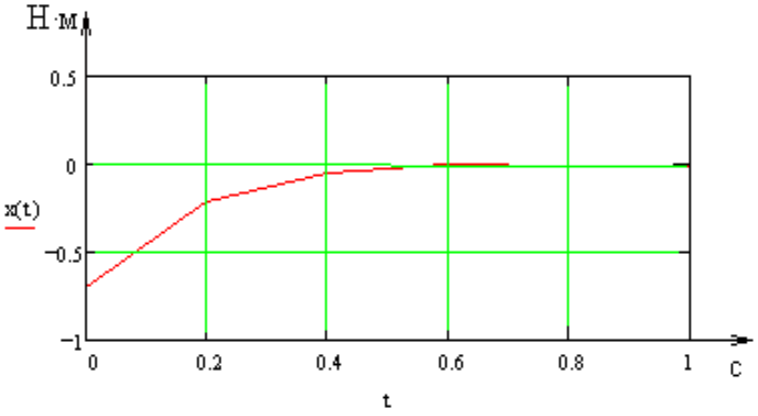


Рис. 4. Переходный процесс для $M_н=M_н(t)$ при заклини-вании долота

При наличии динамического гасителя колебаний система (12) будет иметь вид

$$\left.\begin{aligned} &J\left((j\omega)^2\varphi_н - j\omega\omega_0\right)+c_г\left(\varphi_н-\varphi_г\right)=-bj\omega\varphi_н - M_н \\ &M_н = Q_{нк}\varphi_н + R_{нк}\left[\varphi_к\right] \\ &M_к = -R_{нк}\varphi_н + R_{нк}\left[\varphi_к\right] \end{aligned}\right\} \qquad (19)$$

Дифференциальное уравнение динамического гасителя колебаний

$$J\left((j\omega)^2\right)\varphi_н - c_г\left(\varphi_н-\varphi_г\right)=0,$$

или

$$\varphi_г\left(J_г(j\omega)^2 + c_г\right) - c_г\varphi_н = 0.$$

Найдем угол закручивания $\varphi_г$

$$\varphi_г = \frac{c_г \varphi_н}{J_г (j\omega)^2 + c_г}.$$

Подставляя в (19), получаем

$$\left.\begin{aligned} J\left((j\omega)^2 \varphi_н - j\omega\omega_0\right) &= -b^* j\omega\varphi_н - M_н \\ M_н &= Q_{нк}\varphi_н + R_{нк}[\varphi_к] \\ M_к &= -R_{нк}\varphi_н + R_{нк}[\varphi_к] \end{aligned}\right\} \qquad (20)$$

где

$$b^* = \left[-b - \frac{c_г}{j\omega}\left(1 - \frac{c_г}{J_г (j\omega)^2 + c_г}\right)\right].$$

Таким образом

$$M_{пг} = \frac{Q_{нк}(j\omega) J\omega_0 + \left[R_{нк}(\omega)\left((j\omega)^2 J + j\omega b^*\right)\right]}{(j\omega)^2 J + (j\omega) b^* + Q_{нк}}. \qquad (21)$$

Переходный процесс, полученный с использованием (21) при $\omega_г = 0,25\,\text{с}^{-1}$, $J_г = 26$ Н·с^2, $c_г = \frac{J_г}{\omega_г^2} = 416\ \frac{\text{Н}\cdot\text{м}}{\text{рад}}$ показан на рисунке 5, где

$$x_г(t) = \frac{1}{\pi}\text{Re}\left(\int_\omega M_{пг}(\omega) e^{j\omega t} d\omega\right).$$

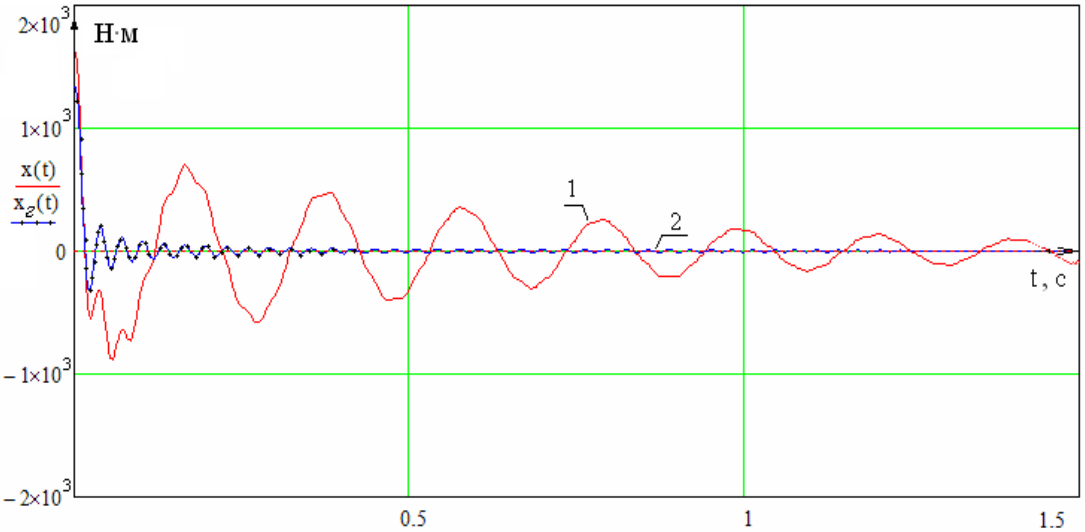


Рис. 5. Переходный процесс в колонне *x(t)* без гасителя колебаний – кривая 1 и при наличии гасителя – кривая 2.

Выводы

1. Разработана модификация МКЭ, основанная на узловых напряжениях, преобразованных по Лапласу.
2. Разработана математическая модель бурильной установки.
3. Рассмотрены переходные процессы в буровой установке при срыве и заклинивании долота с использованием изображений для краевых крутящих моментов при крутильных колебаниях вала, рассматриваемого как стержень с распределенными параметрами.
4. Рассмотрены тестовые примеры для определения колебаний колонны труб бурильной установки, подтверждающие эффективность предложенных мер по снижению

уровня колебаний. Наличие динамического гасителя колебаний в 2 раза сокращает уровень крутильных колебаний на опасных режимах бурения.

Литература

1. Ладыженская О. А. Метод конечных разностей в теории уравнений с частными производными. УМН, т. 12, вып. 5, 1957. С. 123 – 148.
2. Barthold L. O., Carter G. K., Digital Traveling Waves Solutions, PAS, 1961, v. 80, p. 812.
3. Бьюлей Л. В. Волновые процессы в линиях передачи и в трансформаторах. ОНТИ, 1938.
4. Костенко М. В. Атмосферные перенапряжения и грозозащита высоковольтных установок. М.: Госэнергоиздат, 1949. 330 с.
5. Кадомская К. П., Левинштейн М. Л., Штеренберг Г. П. О решении уравнений длинной линии электропередачи на математических машинах. М.: Изв. АН СССР, Энергетика и транспорт, №4, 1963. С. 587–592.
6. Караев Р. И. Переходные процессы в линиях большой протяженности. М.: Госэнергоиздат, 1963.
7. Кадымов Я. Б. Переходные процессы в системах с распределенными параметрами. М.: Наука, 1968, 192 с.
8. Цыпкин Я. З. Теория линейных импульсных систем / Я. З. Цыпкин. – М.: Физматгиз, 1963. 968 с.
9. Санкин Ю. Н. Метод конечных элементов в динамике вязкоупругих систем в пространстве преобразований Лапласа / Ю.Н. Санкин // Труды Средневолжского математического общества. – 2006. – Т.8. №2. С. 22–33.
10. Санкин Ю. Н. Спектральные методы исследования электромеханических систем, включающих звенья с распределенными параметрами / Ю.Н. Санкин, С.Л. Пирожков; под общ. ред. Ю.Н. Санкина. – Ульяновск: УлГТУ, 2009. 132 с.
11. Фридман В.М. Видоизменение метода Бубнова–Галеркина–Ритца, связанное со смешенным вариационным принципом в теории упругости / В. М. Фридман, В. С. Чернина // Изв. АН СССР, МТТ, 1969. – №1. С. 104 – 108.

ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ ВПОЛНЕ НЕУНИТАРНОГО ИЗОМЕТРИЧЕСКОГО ОПЕРАТОРА

Л.А. Штраус, И.В. Баринова

Ульяновский государственный университет

В настоящей работе завершается построение модели изометрического оператора, начатое в статье [11]. Фундаментальным результатом соответствующего круга проблем является модель сжатия в функциональном пространстве, полученная Б. Секефальви-Надем и Ч. Фояшем [1]. Модель симметрического оператора, связанная с голоморфными функциями, значениями которых являются операторы параллельного проектирования на дефектные пространства, развивалась в работах А.В. Штрауса сначала для частных случаев [2], а затем была получена им для произвольного симметрического вполне несамосопряжённого оператора в гильбертовом пространстве [3,4]. Вся реализуемая нами схема построения модели основана на методах А.В. Штрауса, которые в случае изометрического оператора приобретают особенно прозрачный и естественный вид.

Обобщённой резольвентой изометрического оператора T, действующего в гильбертовом пространстве H, порождённой его унитарным расширением $\tilde{T}$ в гильбертовом пространстве $\tilde{H}$, будем называть функцию $\lambda \mapsto R_\lambda,\ |\lambda| \neq 1$, заданную формулой $R_\lambda = P_{\mathrm{H}}\left(\tilde{T} - \lambda I\right)^{-1}|H$, где P_{H} — оператор ортогонального проектирования в $\tilde{H}$ на H. При $|\lambda| > 1$ имеет место представление [7]

$$R_\lambda = \left(\left(T \oplus \Phi(\lambda)\right) - \lambda I\right)^{-1}, \qquad (1)$$

где $\lambda \mapsto \Phi(\lambda),\ |\lambda| > 1$, голоморфная функция, значениями которой являются сжатия, действующие из пространства $N = H \ominus D_T$ в пространство $N_0 = H \ominus R_T$. Из (1) следует представление обобщённой резольвенты оператора T^{-1}, порождённой его унитарным расширением $\tilde{T}^{-1} = \tilde{T}^*$:

$$P_{\mathrm{H}}\left(\tilde{T}^{-1} - \lambda I\right)^{-1}|H = \left(\left(T^{-1} \oplus \Phi^*\left(\bar{\lambda}\right)\right) - \lambda I\right)^{-1}, \qquad |\lambda| > 1. \qquad (2)$$

Имеют место представления

$$(T - \lambda I)D_T \dot{+} N = H, \quad |\lambda| > 1,$$
$$(T - \lambda I)D_T \dot{+} N_0 = H, \quad |\lambda| < 1.$$

Обозначим через $P_{N,\lambda}$, $P_{N_0,\lambda}$ операторы проектирования в H на N и N_0 параллельно $(T - \lambda I)D_T$ при $|\lambda| > 1$ и $|\lambda| < 1$ соответственно. Также обозначим через $\tilde{P}_N$ и $\tilde{P}_{N_0}$ операторы ортогонального проектирования в $\tilde{H}$ на N и N_0 соответственно, $P_N = \tilde{P}_N|H,\ P_{N_0} = \tilde{P}_{N_0}|H$.

Получим представление функций $P_N R_\lambda = \tilde{P}_N\left(\tilde{T} - \lambda I\right)^{-1}|H$, $P_{N_0}R_\lambda = \tilde{P}_{N_0}\left(\tilde{T} - \lambda I\right)^{-1}|H$, использующее функцию $\Phi(\lambda)$ и операторы $P_{N,\lambda}$, $P_{N_0,\lambda}$.

Предложение 1. Имеют место формулы

$$P_N R_\lambda = \left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)^{-1} P_{N,\lambda}, \qquad (3)$$
$$P_{N_0}R_\lambda = \frac{1}{\lambda}\left(\Phi(\lambda)\left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)^{-1} P_{N,\lambda} - P_{N_0}\right), \qquad (4)$$

где $|\lambda| > 1$, и

$$P_N R_\lambda = \Phi^*\left(\frac{1}{\lambda}\right)\left(I - \lambda P_{N_0,\lambda}\,\Phi^*\left(\frac{1}{\lambda}\right)\right)^{-1} P_{N_0,\lambda}, \qquad (5)$$

$$P_{N_0} R_\lambda = \frac{1}{\lambda}\left(\left(I - \lambda P_{N_0,\lambda}\,\Phi^*\left(\frac{1}{\lambda}\right)\right)^{-1} P_{N_0,\lambda} - P_{N_0}\right), \qquad (6)$$

где $|\lambda| < 1$ (при $\lambda = 0$ выражения в правых частях равенств (5) и (6) заменяются их пределами).

Доказательство. Пусть $|\lambda| > 1$, $\left(\left(T \oplus \Phi(\lambda)\right) - \lambda I\right)(f + g) = h,$

где $f \in D_T,\ g \in N,\ h \in H.$ Тогда

$$(T - \lambda I)f + (\Phi(\lambda) - \lambda I)g = h. \qquad (7)$$

Применим к обеим частям (7) оператор $P_{N,\lambda}$:

$$\left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)g = P_{N,\lambda}h,$$

$$g = \left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)^{-1} P_{N,\lambda}h. \qquad (8)$$

Справедливость (3) установлена.

Применим к обеим частям (7) оператор P_{N_0}:

$$\Phi(\lambda)g - \lambda P_{N_0}(f + g) = P_{N_0}h,$$

$$P_{N_0}(f + g) = \frac{1}{\lambda}\left(\Phi(\lambda)g - P_{N_0}h\right).$$

Отсюда с учётом (8) получаем

$$P_{N_0}(f + g) = \frac{1}{\lambda}\left(\Phi(\lambda)\left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)^{-1} P_{N,\lambda} - P_{N_0}\right)h,$$

$$\tilde{P}_{N_0}\left(\tilde{T} - \lambda I\right)^{-1}|H = \frac{1}{\lambda}\left(\Phi(\lambda)\left(P_{N,\lambda}\Phi(\lambda) - \lambda I\right)^{-1} P_{N,\lambda} - P_{N_0}\right)$$

и тем самым справедливость (4) установлена. Для доказательства (5) и (6) получим представление обобщённой резольвенты R_λ при $|\lambda| < 1.$

Заметим, что

$$\left(\tilde{T} - \lambda I\right)^{-1} = \left(\tilde{T} - \lambda\tilde{T}^{-1}\tilde{T}\right)^{-1} = -\frac{1}{\lambda}\tilde{T}^{-1}\left(\tilde{T}^{-1} - \frac{1}{\lambda}I\right)^{-1} =$$

$$= -\frac{1}{\lambda}\left(\left(\tilde{T}^{-1} - \frac{1}{\lambda}I\right) + \frac{1}{\lambda}I\right)\left(\tilde{T}^{-1} - \frac{1}{\lambda}I\right)^{-1} = -\frac{1}{\lambda}\left(I + \frac{1}{\lambda}\left(\tilde{T}^{-1} - \frac{1}{\lambda}I\right)^{-1}\right).$$

Отсюда с учётом (1) получаем

$$R_\lambda = -\frac{1}{\lambda}\left(I + \frac{1}{\lambda}\left(\left(T^{-1} \oplus \Phi^*\left(\frac{1}{\lambda}\right)\right) - \frac{1}{\lambda}I\right)^{-1}\right) =$$

$$= \frac{\left(I - \lambda\left(T^{-1} \oplus \Phi^*\left(\frac{1}{\lambda}\right)\right)\right)^{-1} - I}{\lambda}. \qquad 9)$$

Пусть $\left(I - \lambda\left(T^{-1} \oplus \Phi^*\left(\frac{1}{\lambda}\right)\right)\right)(f + g) = h,$ где $f \in R_T,\ g \in N_0$. Тогда

$$(I - \lambda T^{-1})f + \left(I - \lambda\Phi^*\left(\frac{1}{\lambda}\right)\right)g = h. \qquad (10)$$

Применим к обеим частям равенства (10) оператор $P_{N_0,\lambda}$:

$$g-\lambda P_{N_0,\lambda}\,\Phi^*\left(\frac{1}{\bar{\lambda}}\right)g=P_{N_0,\lambda}\,h,$$

$$g=\left(I-\lambda P_{N_0,\lambda}\,\Phi^*\left(\frac{1}{\bar{\lambda}}\right)\right)^{-1}P_{N_0,\lambda}\,h. \qquad (11)$$

С учётом (9) получаем отсюда

$$\tilde{P}_{N_0}\left(\tilde{T}-\lambda I\right)^{-1}|H=\frac{1}{\lambda}\left(\left(I-\lambda P_{N_0,\lambda}\,\Phi^*\left(\frac{1}{\bar{\lambda}}\right)\right)^{-1}P_{N_0,\lambda}-P_{N_0}\right),$$

то есть равенство (6) справедливо.

Теперь применим к обеим частям (10) оператор $P_N=\tilde{P}_N|H$ ортогонального проектирования в H на N:

$$P_N(f+g)-\lambda\Phi^*\left(\frac{1}{\bar{\lambda}}\right)g=P_N h.$$

Отсюда с учётом (9) и (11) получаем равенство (5).

Пусть $\tilde{T}$- унитарное расширение оператора T, которому отвечает функция $\Phi(\lambda)=0,\ |\lambda|>1$, и обобщённая резольвента имеет вид

$$R_\lambda=\left((T\oplus 0)-\lambda I\right)^{-1}=(T_0-\lambda I)^{-1},$$

где $T_0=T\oplus 0$ - нулевое расширение оператора T.

Это означает, что $\tilde{T}$ является дилатацией оператора T_0 . Из формул (3)-(6) получаем:

$$P_N R_\lambda=\begin{cases}0, & |\lambda|<1,\\ \frac{1}{\lambda}\,P_{N,\lambda}, & |\lambda|>1,\end{cases} \qquad (12)$$

$$P_{N_0}R_\lambda=\begin{cases}\frac{1}{\lambda}\left(P_{N_0,\lambda}-P_{N_0}\right), & |\lambda|<1,\\ -\frac{1}{\lambda}\,P_{N_0}, & |\lambda|>1.\end{cases} \qquad (13)$$

Обозначим через $P\left(e^{it}\right)$ граничное значение (предел в смысле сильной топологии по некасательным направлениям) голоморфной функции $\lambda\mapsto P_{N_0,\lambda}|N,\ |\lambda|<1,$ значениями которой являются сжатия [7]. Такой предел существует почти при всех $t\in[0;2\pi]$. В дальнейшем множество таких значений t будем обозначать через C_p.При любом $t\in C_p$ зададим неотрицательный ограниченный оператор $W(t)$ в пространстве $N\oplus N_0$:

$$W(t)=\begin{pmatrix}I_N & P^*\left(e^{it}\right)\\ P\left(e^{it}\right) & I_{N_0}\end{pmatrix}. \qquad (14)$$

Пусть $t\mapsto\tilde{E}(t),\ 0\le t\le 2\pi$,– спектральная функция оператора $\tilde{T}$, а $t\mapsto E(t),\ 0\le t\le 2\pi$,– обобщённая спектральная функция оператора T, $E(t)=P_{\mathrm{H}}\tilde{E}(t)|H.$

Введём следующие операторнозначные функции [11] переменной t:

$$E_{11}(t)=\tilde{P}_N\tilde{E}(t)|N=P_N E(t)|N,$$
$$E_{12}(t)=\tilde{P}_N\tilde{E}(t)|N_0=P_N E(t)|N_0,$$
$$E_{21}(t)=\tilde{P}_{N_0}\tilde{E}(t)\big|N=P_{N_0}E(t)\big|N,$$
$$E_{22}(t)=\tilde{P}_{N_0}\tilde{E}(t)\big|N_0=P_{N_0}E(t)\big|N_0.$$

Функции $t \mapsto E_{11}(t)$ и $t \mapsto E_{22}(t)$ являются обобщёнными разложениями единицы в пространствах N и N_0. При $t \in [0;2\pi]$ зададим в $N \oplus N_0$ ограниченный самосопряжённый оператор

$$E_{N\oplus N_0}(t) = \begin{pmatrix} E_{11}(t) & E_{12}(t) \\ E_{21}(t) & E_{22}(t) \end{pmatrix} \tag{15}$$

и функцию распределения $t \mapsto E_{N\oplus N_0}(t)$, которую будем называть $(N \oplus N_0)$ - обобщённой спектральной функцией оператора T и которая соответствует оператору $\tilde{T}$ и его спектральной функции $t \mapsto \tilde{E}(t)$.

Предложение 2. Для дилатационной обобщённой $(N \oplus N_0)-$ спектральной функции оператора T имеет место представление

$$E_{N\oplus N_0}(\lambda) = \int_0^{\lambda} W(t)d\mu(t), \tag{16}$$

где $\lambda \in [0;2\pi]$, $\mu(t) = \frac{1}{2\pi}t$.

Доказательство. С учётом (14) и (15) следует убедиться в том, что имеют место формулы

$$\begin{aligned} E_{11}(\lambda) &= \mu(\lambda)I_N, \\ E_{22}(\lambda) &= \mu(\lambda)I_{N_0}, \\ E_{12}(\lambda) &= \int_0^{\lambda} P^*\left(e^{it}\right)d\mu(t), \\ E_{21}(\lambda) &= \int_0^{\lambda} P\left(e^{it}\right)d\mu(t). \end{aligned} \tag{17}$$

Они следуют из (12) и (13) с помощью интегральной формулы Коши и соотношения $P_{N,\lambda}|N_0 = \left(P_{N_0\frac{1}{\bar{\lambda}}}|N\right)^*, \quad |\lambda| > 1.$

Напомним ещё некоторые факты из работы [11], сохраняя принятые в ней обозначения. Пусть

$$\tilde{H}_1 = \bigvee_{k\in\mathbb{Z}} \tilde{T}^k N \quad , \tilde{H}_2 = \bigvee_{k\in\mathbb{Z}} \tilde{T}^k N_0 .$$

Если оператор T вполне неунитарный и $\tilde{T}$ - его минимальное унитарное расширение, то $\tilde{H} = \tilde{H}_1 \vee \tilde{H}_2$.

Был введён унитарный оператор $\tilde{\Phi}\colon \tilde{H} \to \tilde{\mathfrak{H}}$ как продолжение по непрерывности изометрического оператора $\tilde{\Phi}$:

$$\tilde{\Phi}\left(\tilde{T}^m x + \tilde{T}^n y\right) = \begin{pmatrix} x^{[m]} \\ y^{[n]} \end{pmatrix},$$

$x \in N, y \in N_0, \ m,n \in \mathbb{Z}, \ x^{[m]}(t) = e^{imt}x, \ y^{[n]}(t) = e^{int}y, \ t \in [0;2\pi].$

Пространство $\tilde{\mathfrak{H}}$ является пополнением предгильбертова пространства, полученного факторизацией линейной оболочки функций вида

$$x^{[m]}(t) = e^{imt}x, \quad y^{[n]}(t) = e^{int}y, \ t \in [0;2\pi]$$

с полунормой

$$\left\| \begin{matrix} x^{[m]} \\ y^{[n]} \end{matrix} \right\|_{\tilde{\mathfrak{H}}}^2 = \int_0^{2\pi} \left(E_{N\oplus N_0}(dt) \begin{pmatrix} x^{[m]}(t) \\ y^{[n]}(t) \end{pmatrix}, \begin{pmatrix} x^{[m]}(t) \\ y^{[n]}(t) \end{pmatrix} \right)_{N\oplus N_0} .$$

В соответствии с этим скалярное произведение в $\tilde{\mathfrak{H}}$ задаётся с помощью плотного в нём многообразия $(N \oplus N_0)-$ значных функций по формуле

$$\left(\begin{pmatrix}\tilde{f}_1\\ \tilde{f}_2\end{pmatrix},\begin{pmatrix}\tilde{g}_1\\ \tilde{g}_2\end{pmatrix}\right)_{\tilde{\mathfrak{H}}}=\int_0^{2\pi}\left(E_{N\oplus N_0}(dt)\begin{pmatrix}\tilde{f}_1\\ \tilde{f}_2\end{pmatrix},\begin{pmatrix}\tilde{g}_1\\ \tilde{g}_2\end{pmatrix}\right)_{N\oplus N_0}.$$

Отсюда и из (16) следует, что скалярное произведение элементов $\begin{pmatrix}\tilde{f}_1\\ \tilde{f}_2\end{pmatrix}, \begin{pmatrix}\tilde{g}_1\\ \tilde{g}_2\end{pmatrix}$ пространства $\tilde{\mathfrak{H}}$ задаётся формулой

$$\left(\begin{pmatrix}\tilde{f}_1\\ \tilde{f}_2\end{pmatrix},\begin{pmatrix}\tilde{g}_1\\ \tilde{g}_2\end{pmatrix}\right)_{\tilde{\mathfrak{H}}}=\int_0^{2\pi}\left(W(t)\begin{pmatrix}\tilde{f}_1\\ \tilde{f}_2\end{pmatrix},\begin{pmatrix}\tilde{g}_1\\ \tilde{g}_2\end{pmatrix}\right)d\mu(t). \qquad (18)$$

При любом $t\in C_p$ положим $\tau_0(t)=(N\oplus N_0)\ominus KerW(t)$ и через $Y_0(t)$ обозначим оператор ортогонального проектирования в $N\oplus N_0$ на $\tau_0(t)$. В $\tau_0(t)$ $W(t)$ индуцирует самосопряжённый положительный оператор

$$W_0(t)=W(t)|\tau_0(t).$$

Введём предгильбертово пространство $\mathring{\tau}_-(t)$ совпадающее как линейное множество с $\tau_0(t)$, полагая для $x,y\epsilon(N\oplus N_0)$

$$(Y_0(t)x,Y_0(t)y)_{\mathring{\tau}_-}=(W(t)x,y)_{N\oplus N_0}. \qquad (19)$$

Через $\tau_-(t)$ обозначим пополнение пространства $\mathring{\tau}_-(t)$. Введём отображение $Y_-(t)$ пространства $N\oplus N_0$ в $\tau_-(t)$:

$$Y_-(t)=O_{-_0}(t)Y_0(t);$$

здесь $O_{-_0}(t)$ — оператор вложения $\tau_0(t)$ в $\tau_-(t)$. Отметим, что для любых $x,y\in N$

$$(Y_-(t)x,Y_-(t)y)_{\tau_-}=(x,y)_N \qquad (20)$$

и для любых $x,y\in N_0$

$$(Y_-(t)x,Y_-(t)y)_{\tau_-}=(x,y)_{N_0}. \qquad (21)$$

Обозначим через $P_1(t)$ и $P_2(t)$ действующие в $\tau_-(t)$ операторы ортогонального проектирования на $Y_-(t)N$ и $Y_-(t)N_0$ соответственно. Из (16) и (19) следует $P_2(t)Y_-(t)|N=Y_-(t)P\left(e^{it}\right)$ и

$$P_1(t)Y_-(t)|N_0=Y_-(t)P^*\left(e^{it}\right).$$

Это означает, что оператор

$$W_-^0(t)=Y_-(t)W_0(t)O_{-_0}^{-1},$$

в который преобразуется $W(t)$ в $\mathring{\tau}_-(t)$, обладает непрерывным продолжением

$$\overline{W_-^0(t)}=P_1(t)+P_2(t)$$

на всё пространство $\tau_-(t)$.

Введём гильбертово пространство $\tau_+(t)=W_0^{\frac{1}{2}}(t)\tau_0(t)$ со скалярным произведением $(x,y)_{\tau_+}=\left(W_0^{-\frac{1}{2}}(t)x,W_0^{-\frac{1}{2}}(t)y\right)_{\tau_0}$. Тогда оператор

$$\Pi_{+-}(t)=O_{0+}^{-1}(t)W_0(t)\,O_{-_0}^{-1}(t) \qquad (22)$$

является изометрическим, а его замыкание $\Pi(t)$ — унитарным оператором, действующим из $\tau_-(t)$ в $\tau_+(t)$, и

$$O_{-_+}(t)\Pi(t)=P_1(t)+P_2(t). \qquad (23)$$

Из (22) и (23) следует, что для любых элементов $x,y\in\tau_-(t)$ и при любом $t\in C_p$

$$(x,y)_{\tau_-} = \left(W_0^{-\frac{1}{2}}(t) O_{-0}^{-1}(t) \begin{pmatrix} P_1(t)x \\ P_2(t)x \end{pmatrix}, W_0^{-\frac{1}{2}}(t) O_{-0}^{-1}(t) \begin{pmatrix} P_1(t)y \\ P_2(t)y \end{pmatrix} \right)_{\tau_0}. \qquad (24)$$

Имеет место равенство ([11],(8))

$$P_{N_0} E(s)h = \int_0^s E_{22}(dt)\left(P_{\mathfrak{F}_2}\Phi h\right)(t), \qquad (25)$$

где $P_{\mathfrak{F}_2}$ – оператор ортогонального проектирования в $\mathfrak{F}$ на $\mathfrak{F}_2$.

Из формул (13), (17), (25) получаем при $h\epsilon N$ и $|\lambda| < 1,\ \lambda = re^{i\varphi}$:

$$P_{N_{0,\lambda}} h = \lambda P_{N_0} R_\lambda h + P_{N_0} h = \lambda P_{N_0} R_\lambda h - \frac{1}{\bar{\lambda}} P_{N_0} R\left(\frac{1}{\bar{\lambda}}\right) h =$$

$$= P_{N_0} \int_0^{2\pi} \left(\frac{\lambda}{e^{it} - \lambda} - \frac{1}{\bar{\lambda}(e^{it} - \frac{1}{\bar{\lambda}})} \right) E(dt)h = P_{N_0} \int_0^{2\pi} \left(\frac{\lambda}{e^{it} - \lambda} + \frac{1}{1 - \bar{\lambda} e^{it}} \right) E(dt)h =$$

$$= \int_0^{2\pi} \left(\frac{\lambda}{e^{it} - \lambda} + \frac{1}{1 - \bar{\lambda} e^{it}} \right) E_{22}(dt)\left(P_{\mathfrak{F}_2}\Phi h\right)(t) =$$

$$= \frac{1}{2\pi} \int_0^{2\pi} \frac{1 - r^2}{1 - 2r\cos(\varphi - t) + r^2} \left(P_{\mathfrak{F}_2}\Phi h\right)(t) dt.$$

Таким образом, значение $P_{N_{0,\lambda}} h$ является свёрткой функции $\left(P_{\mathfrak{F}_2}\Phi h\right)(t)$ с ядром Пуассона и из теореме Фату мы получаем, что почти при всех $t \in [0; 2\pi]$ функция $P_{N_{0,\lambda}}$, $|\lambda| < 1$, обладает угловыми граничными значениями $P_{N_0}\left(e^{it}\right)$ и при любом $h\epsilon H$ и почти при всех $t \in [0; 2\pi]$ имеет место равенство

$$P_{N_0}\left(e^{it}\right)h = \left(P_{\mathfrak{F}_2}\Phi h\right)(t). \qquad (26)$$

Аналогично рассматривая вместо оператора T оператор T^{-1}, получаем, что почти при всех $t \in [0; 2\pi]$ функция $P_{N,\lambda}$, $|\lambda| > 1$, обладает угловыми граничными значениями $P_N\left(e^{it}\right)$ и почти при всех $t \in [0; 2\pi]$ и любом $h\epsilon H$

$$P_N\left(e^{it}\right)h = \left(P_{\mathfrak{F}_1}\Phi h\right)(t). \qquad (27)$$

Формулы (18), (24), (26), (27), позволяют заключить, что для любых $x, y \in H$

$$(x,y) = \frac{1}{2\pi} \int_0^{2\pi} \left(W_0^{-\frac{1}{2}} Y_0(t) \begin{pmatrix} P_N\left(e^{it}\right)x \\ P_{N_0}\left(e^{it}\right)x \end{pmatrix}, W_0^{-\frac{1}{2}} Y_0(t) \begin{pmatrix} P_N\left(e^{it}\right)y \\ P_{N_0}\left(e^{it}\right)y \end{pmatrix} \right) dt$$

и имеет место

Теорема. Вполне неунитарный изометрический оператор T в гильбертовом пространстве H унитарно эквивалентен оператору $\boldsymbol{T}$ умножения на независимую переменную e^{it} в гильбертовом пространстве

$$\boldsymbol{H} = \left\{ \boldsymbol{x} \mid \boldsymbol{x}(t) = \begin{pmatrix} P_N\left(e^{it}\right)x \\ P_{N_0}\left(e^{it}\right)x \end{pmatrix}, x \in H \right\}$$

со скалярным произведением

$$(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2\pi}\int_0^{2\pi}\left(W_0^{-\frac{1}{2}}Y_0(t)\boldsymbol{x}(t), W_0^{-\frac{1}{2}}Y_0(t)\boldsymbol{y}(t)\right)dt\,.$$

Литература

1. Секефальви-Надь Б., Фояш Ч. Гармонический анализ операторов в гильбертовом пространстве.- М.: Мир, 1970.
2. Штраус А.В. К спектральной теории регулярных симметрических операторов. // Функциональный анализ. Вып.10.-Ульяновск, 1978, с. 145-153.
3. Штраус А.В. Функциональные модели и обобщённые спектральные функции симметрических операторов. Препринт.- Ульяновск: УлГПУ им. И.Н. Ульянова, 1997,- 70 с.
4. Штраус А.В. Функциональные модели и обобщённые спектральные функции симметрических операторов. // Алгебра и анализ. Том 10, вып. 5.- Санкт-Петербург, 1998, с.733-784.
5. Бродский М.С. Унитарные операторные узлы и их характеристические функции. // Успехи математических наук, 1978, т. 33, №4, с. 141-168.
6. Штраус Л.А. О представлении регулярного изометрического оператора. // Функциональный анализ. Спектральная теория. Вып. 18.-Ульяновск, 1982, с. 127-132.
7. Штраус Л.А. О связи характеристических функций изометрического оператора и унитарного узла. // Функциональный анализ. Гармонический анализ и теория меры. Вып. 19.-Ульяновск, 1982, с. 172-176.
8. Штраус Л.А. , Баринова И.В. К теории представления регулярного оператора. // Фундаментальные проблемы математики механики: Учёные записки Ульяновского государственного университета. Часть 1. Вып 1.- Ульяновск, 1996, с. 24-32.
9. Штраус Л.А. Модель регулярного операторного узла. // Функциональный анализ. Вып. 36.-Ульяновск, 1997, с. 75-79.
10. Штраус Л.А. Спектральное представление вполне неунитарного оператора. // Учёные записки Ульяновского государственного университета. Сер. Фундаментальные проблемы математики механики. Вып. 2(9).- Ульяновск, 2000, с. 84-88.
11. Штраус Л.А., Баринова И.В. Обобщённые спектральные функции изометрического оператора и отвечающие им спектральные преобразования (в печати).

БЕЗОПАСНОСТЬ ДАННЫХ В МОБИЛЬНЫХ ПЕРСОНАЛЬНЫХ УСТРОЙСТВАХ

А.А.Анисимов

Ульяновский государственный университет

Введение

В области современных технологий наблюдается тенденция к всесторонней взаимной интеграции. Беспроводные устройства быстро получают все более мощные процессоры, стремясь к возможностям персонального компьютера, а техника — смартфоны, автомобильные компьютеры и бытовые приборы нового поколения — обзаводится функциями беспроводной связи. Подобные тенденции таят в себе весомые опасности для пользователей, а для злоумышленников открываются новые возможности. Более того, выделяется новый вид криминальной деятельности, выражающийся в незаконном проникновении в сети мобильной связи в целях бесплатного получения услуг, а также завладения конфиденциальной информацией, циркулирующей в этих системах, и негативном воздействии на их элементы – индивидуальные устройства.

На данный момент выделяется отдельный класс портативных индивидуальных устройств, которые используются повседневно в обычной жизни, в различных организациях - мобильные персональные устройства (МПУ). Отличительными предъявляемыми к ним требованиями являются легкость использования, компактность и портативность. В целом, класс таких устройств весьма обширен. В него включаются смартфоны, планшеты, карманные персональные компьютеры, навигаторы и прочие портативные и миниатюрные вычислительные устройства. При этом тенденция увеличения объема циркулирующего информационного потока, применение современных средств связи (аудио и видео), средств криптозащиты и контроля функционирования ставит серьезную преграду для разработчиков МПУ с точки зрения обеспечения производительности.

Стоит также отметить, что, несмотря на различие в предназначении устройств, идет процесс к унификации управляющих операционных систем. Это позволяет разработчикам создавать программы и приложения, обеспечивающие безопасность, для всего класса, а не только под конкретный вид устройств.

Рассматривая МПУ можно отметить ряд присущих им особенностей, а именно:

1. Большой объем и разнородность обрабатываемой конфиденциальной информации.
2. Подключение к различным каналам связи, в том числе к общественным (недоверенным).
3. Наличие в МПУ дополнительного оборудования такого как камера, GPS/ГЛОНАСС, микрофон и других.

Суммируя, отметим, что МПУ отличаются повышенными требованиями к обеспечению безопасности при ограниченных возможностях её обеспечения. Наличие этого критического противоречия толкает разработчиков мобильных устройств искать новые подходы к обеспечению безопасности.

Целью статьи является обзор существующих видов информационного воздействия на МПУ и история их развития, а также механизмов их защиты.

История и развитие вирусов для мобильных устройств

История вирусов для мобильных устройств начинается в июне 2004 года, когда командой вирусописателей-профессионалов 29A был создан первый вирус для смартфонов. Самораспространяющийся вирус-червь Caribe, функционирует на базе операционной системы Symbian и распространяется при помощи технологии беспроводной передачи данных Bluetooth, за что получил название Worm.SymbOS.Cabir в классификации «Лаборатории Касперского».

Через месяц после Cabir антивирусные компании обнаружили очередную технологическую новинку. Virus.WinCE.Duts первый известный вирус для платформы Windows CE (Windows Mobile), а также — первый файловый вирус (file infector) для смартфонов. Duts заражает исполняемые файлы в корневой директории устройства, предварительно спросив разрешения у пользователя.

Вскоре, снова спустя всего 1 месяц после Duts, появился Backdoor.WinCE.Brador — первый бэкдор для мобильной платформы. Эта вредоносная программа открывает доступ к зараженному устройству — КПК или смартфону — по сети, ожидая подключения злоумышленника на определенном порту. Ее функционал позволяет передавать в обе стороны файлы и выводить на экран текстовые сообщения. Когда зараженное устройство подключается к интернету, бэкдор отсылает его IP-адрес по электронной почте своему хозяину.

На этом активность самых квалифицированных исследователей безопасности мобильных устройств, авторов концептуальных вирусов, представляющих радикально новые технологии в области вирусописательства, практически заканчивается. Последовавший за Brador Trojan.SymbOS.Mosquit представляет собой изначально безвредную игру для платформы Symbian (Mosquitos), в код которой неизвестный злоумышленник внес некоторые исправления. Модифицированная игра при запуске начинает отправлять SMS-сообщения на указанные в коде номера телефонов, подпадая под определение «троянской программы».

Со временем развитие устройств дало возможность использовать более мощные операционные системы. На данный момент лидирующее положение на этом рынке занимают iOS и Google Android. Начиная с 2009 года, для них создаются различные вирусы, и рост их объема настораживает. Во втором квартале 2012 г. по сравнению с показателями первого квартала практически втрое увеличилось количество Android-троянцев. За три месяца в коллекцию «Лаборатории Касперского» было добавлено более 14 900 вредоносных программ. Практически половина (49%) обработанных «Лабораторией Касперского» во втором квартале 2012 вредоносных файлов — это различные многофункциональные троянцы, которые крадут с телефона данные (имена контактов, почтовые адреса, телефоны и т.д.), а также могут загружать дополнительные модули с серверов злоумышленников. Четверть обнаруженных вредоносных программ для Android OS приходится на SMS-троянцев. Они отправляют без ведома хозяев мобильных устройств SMS на платные номера.

На данный момент к категории наиболее опасных вирусов можно отнести:

- для Android: Android.SmsSend, Android.Gongfu, Android.Plankton, Android.GoldDream, Android.Crusewind, Android.SpyEye, Android.DreamExploid, Android.Wukong;
- для iOS: iPhoneOS.HLLW.Ikee.

Как и все компьютерные вирусы, вирусы для МПУ можно классифицировать по поражаемым объектам, по поражаемым операционным системам и платформам, по технологиям, используемым вирусом, по языку, на котором написан вирус, по дополнительной вредоносной функциональности.

Но актуальнее выделить вирусы специфичные только для мобильных устройств. Их можно классифицировать по вредоносным возможностям (соответственно, и целям) следующим образом:

- отправление СМС и звонки на платные номера, а также рассылка СМС;
- включение микрофона без ведома владельца;
- получение данных от GPS-навигатора о местонахождении владельца;
- получение доступ к конфиденциальным данным, сохраненным в виде фотографий и звуковых записей, а также СМС-переписке и истории совершенных звонков;
- включение зараженных машин в управляемые ботнеты, координируемые из одного (или нескольких) командных центров.

Стоит отметить высокую скорость кражи информации с помощью последних вирусов. Так, Android.Gone.1 всего лишь за 60 секунд несанкционированно копирует всю хранящуюся на работающем под управлением Android мобильном телефоне информацию, включая идентификационную информацию о зараженном устройстве, контакты, сообщения, последние звонки, историю браузера и т. д. Украденные данные загружаются на специально созданный вирусописателями сайт.

Виды воздействий на МПУ

Среди видов информационно-технических воздействий на МПУ следует выделить воздействия на канал связи МПУ, что косвенно относится к защите самого МПУ, и, собственно, воздействия на сам МПУ как аппаратно-программного комплекса обработки информации. Безопасность любого компонента (ресурса) автоматизированной системы складывается из обеспечения трех его характеристик: конфиденциальности, целостности и доступности. Конфиденциальность компонента системы заключается в том, что он доступен только тем субъектам доступа (пользователям, программам, процессам), которым предоставлены на то соответствующие полномочия. Целостность компонента предполагает, что он может быть модифицирован только субъектом, имеющим для этого соответствующие права. Целостность является гарантией корректности (неизменности, работоспособности) компонента в любой момент времени. Доступность компонента означает, что имеющий соответствующие полномочия субъект может в любое время без особых проблем получить доступ к необходимому компоненту системы (ресурсу).

При воздействии на канал передачи данных нарушение конфиденциальности информации заключается в подмене трафика, целостности - в изменении и подмене трафика. Если же последний будет блокирован – нарушена доступность информации.

Воздействие на сам МПУ может проходить в этих же трех направлениях. При локальном или удаленном считывании информации нарушается конфиденциальность. Здесь может быть добыта всевозможная информация: адресная книга, SMS и MMS сообщения, GPS журнал, кэш браузера, кэш клавиатуры, сама память устройства. Нарушение целостности заключается в повреждении или подмене информации (подмена установленных приложений зловредным программным обеспечением (ПО), подмена сетевых адресов, подмена GPS данных). Такое нестандартное решение как перегрев процессора на МПУ используется для нарушения доступности информации на ряду с DoS, нарушением работы контроллера аккумулятора и др. в целях блокирования и нарушения работы устройства.

Анализ современных методов защиты МПУ

В настоящее время производители МПУ прилагают усилия для комплексной защиты производимых устройств, причем как в совершенствовании аппаратной части, так и, в большей степени, программной. При этом большая часть мер защиты обеспечивает сразу несколько направлений, взаимно дополняя друг друга на разных уровнях и повышая общую защищенность системы. Говоря о мерах защиты, следует отметить, что архитектура современных операционных систем (ОС) для МПУ изначально разрабатывалась со встроенными в нее механизмами обеспечения безопасности. Однако непрерывное развитие современных средств информационного воздействия постепенно снижают эффективность защиты, заставляя разрабатывать все более совершенные защитные механизмы. Наиболее значимыми из них являются:

- технологии изолирования («sandboxing»),
- шифрования памяти МПУ,
- технология рандомизации адресного пространства (ASLR).

Технология изолирования заключается в использовании запускаемыми программами ресурсов операционной системы через «прослойку» специальной системы управления доступом («песочница», «sandbox»). Для выполняемой программы система управления доступом создает ограничения по использованию ресурсов операционной системы исходя из понятий «типичного» и «нетипичного» поведения конкретной исполняемой программы.

Например, при нормальной работе графическому редактору не требуется доступ к Интернету или к хранилищу учетных записей пользователей и паролей к ним. Таким образом, исходя из реальных «типичных» потребностей программы, система управления доступом разрешает использование только необходимых ресурсов.

Такой подход позволяет в случае взлома исполняемой программы или наличия в ней недекларируемых возможностей не допустить компрометации всей системы и всех её ресурсов. В идеале, сам разработчик ПО должен позаботиться об ограничении допустимых действий своей программы. Однако, если такие ограничения отсутствуют, «песочница» нивелирует просчеты или умышленные закладки разработчика программного обеспечения, а также найденные впоследствии в нём уязвимости. Все современные операционные системы МПУ, такие как Android, iOS, Windows Phone 8 разработаны с внедренными в них подобными системами управления доступом. Например, при установке приложении на МПУ с операционной системой Android предупреждается о том, к чему именно будет иметь доступ работающее приложение.

Использование криптографических средств защиты позволяет обеспечить конфиденциальность информации как при попытке удаленного считывания информации с помощью внедренных в ОС МПУ специальных программных средств, так и при попытке локального считывания информации из внутренней памяти устройства. В силу того, что шифрование всей памяти МПУ при современных нормах от нескольких до десятков гигабайт требует значительных ресурсов, в миниатюрных устройствах применяют аппаратные модули шифрования.

Например, в смартфоне IPhone 4, работающего под управлением ОС iOS 4, применяется мощная система криптозащиты. Операционная система iOS 4 получила специальную систему хранения паролей (keybag) - каждый файл файловой системы шифруется индивидуальным ключем. Таким образом, при вскрытии ключа шифрования какого-либо отдельного файла компрометации всей информации не происходит. Сами ключи хранятся в keybag зашифрованные мастер-ключом, который в свою очередь генерируется на основе уникального идентификатора устройства и пароля, установленного пользователем. Привязка к уникальному идентификатору МПУ делает возможной генерацию мастер-ключа лишь на самом МПУ, и таким образом попытки подобрать мастер-ключ перебором, распараллеливая процесс между несколькими устройствами для достижения приемлемого времени нахождения ключа, становятся невозможными.

Android 3.0 и выше обеспечивает полное шифрование файловой системы, поэтому все пользовательские данные могут быть зашифрованы в ядре с помощью реализации библиотеки dmcrypt из AES128 с CBC и ESSIV: SHA256. Ключ шифрования защищён AES128 с помощью другого ключа, полученного из пароля пользователя. Он предотвращает несанкционированный доступ к хранимым данным без пароля пользователя устройства. Для обеспечения устойчивости к систематическим атакам подбора пароля (например, «радужных таблиц» или «грубая сила»), введенный пароль несколько раз сочетается со случайной солью (предварительно созданной случайной строкой определенной длины) и хэш функцией SHA1 с использованием стандартного алгоритма PBKDF2 до того, как будет использоваться ключ для расшифровки файловой системы. Для обеспечения устойчивости к словарю подбора пароля атак, Android обеспечивает сложность правил пароля, которые могут быть установлены администратором устройства и обеспечиваются операционной системой.

Технология защиты ASLR (Address space layout randomization) заключается в размещении частей программы случайным образом в адресном пространстве вычислительного устройства. Такими частями являются образ самого исполняемого файла, подгружаемые библиотеки, «куча» (heap) и стека – в зависимости от способа организации памяти. «Куча» - это динамическая память, из которой можно выделить фрагмент нужного размера и обращаться к ней из любой части программы, имея валидный указатель. Стек же, это структура данных по принципу "последний вошел - первый вышел". Используется в процессорах в первую очередь для организации вызовов функций, которые, как правило,

вложены друг в друга. Технология ASLR значительно усложняет успешную реализацию существующих уязвимостей. Например, даже если при помощи переполнения буфера или другим методом атакующий получит возможность передать управление по произвольному адресу, ему нужно будет угадать, где же именно расположен стек или куча или другие места памяти в которые он может поместить шелл-код (двоичный исполняемый код, который обычно передаёт управление командному процессору для пользования взломщиком). Важно отметить, что ASLR подразумевает под собой лишь общую идею, технологический подход к защите системы. Реализация технологии ASLR на различных операционных системах может сильно отличаться как по сложности и влиянию на производительность системы, так и по степени защиты от атак. Здесь под степенью защиты понимают вероятность угадывания атакующим расположения сегментов взламываемой программы.

При попытках применения ASLR в мобильных устройствах, как, впрочем, и при попытках применения других технологий, заимствованных из мира настольных компьютеров, перед разработчиками возникают проблемы сохранения приемлемого уровня быстродействия мобильных устройств. Рассматривая операционную систему Android, можно отметить, что в ней для решения проблемы быстродействия приложений был введен запускаемый при загрузке мобильного устройства процесс «zygote», который содержит в себе экземпляры общесистемных библиотек, окружений и виртуальной машины Dalvik. При запуске приложения, система делает копию уже загруженных в память проинициализированных ресурсов, таким образом заметно выигрывая в скорости запуска. Отсюда вытекает, что для всех установленных на мобильном устройстве приложений параметры расположения системных ресурсов будут одними и теми же. Впрочем, как утверждают разработчики Android 4, реализация ASLR в этой мобильной ОС была предназначена для усложнения потенциальных сетевых атак на устройство. Однако, несмотря на все вышеизложенные недостатки, ASLR остаётся достаточно эффективной технологией информационной защиты.

Помимо приведенных выше, существует еще ряд приемов, повышающих общую защищенность МПУ. К ним можно отнести подписывание устанавливаемых приложений и обновлений операционной системы электронной подписью (ЭП) в целях идентификации источника ПО и проверки целостности программного кода. В дальнейшем, исходя из результатов проверки ЭП и политики безопасности, устанавливаемой разработчиком ОС, применяется решение о допустимости или недопустимости установки ПО. Например, iPhone под управлением iOS позволяет устанавливать только ПО, подписанное разработчиком - компанией Apple - с помощью их электронной подписи.

В ОС Android 4.1 следует отметить включение по умолчанию для исполняемых файлов режима PIE (Position Independent Executable), защищающего исполняемые файлы от внедрения стороннего кода, благодаря случайному распределению адресного пространства. Дополнительно PIE-программы теперь собираются флагом BIND_NOW, позволяющим сократить число областей в памяти, в которых возможно внесение изменений для организации ответвления от хода выполнения программы, при применении основанных на попытках повреждения областей памяти эксплоитов.

Заключение

Мобильные устройства становятся наиболее используемыми устройствами в современном мире. Именно на них всё чаще хранится самая важная конфиденциальная информация, которая должна быть обеспечена безопасностью. Однако современные системы не обеспечивают комплексного подхода к этому вопросу, рассматривая лишь отдельные виды воздействий. Другой же проблемой является ограниченная производительность МПУ, что становится преградой для использования обычных методов, из-за чего внедрение систем защиты может заметить обычный пользователь в неудобстве при работе с устройством и медленном выполнении различных команд. Третья проблема, это внедрение все новых функций и каналов передачи данных, передающих важную конфиденциальную информацию. Например, функция Android Beam позволяет обмениваться данными с

помощью NFC (Near Field Communication) — технология беспроводной высокочастотной связи малого радиуса действия, которая дает возможность обмена данными между устройствами, находящимися на расстоянии около 10 сантиметров. Она может предназначаться не только для обмена файлами, приложениями, ссылками, картами или контактами, но и для оплаты вместо популярных сейчас пластиковых карт, в качестве карт доступа и т.п. в силу своего удобства.

Особо необходимо выделить важную тенденцию - рост числа вирусописателей, которые переключаются на разработку вредоносных программ для мобильных устройств. По данным «Лаборатории Касперского» во втором квартале 2012 г. по сравнению с показателями первого квартала практически втрое увеличилось количество троянских программ для ОС Android. За три месяца было обнаружено более 14 900 новых вредоносных приложений.

Злоумышленные программы эволюционируют и на качественном уровне: вирусописатели придумывают различные приемы, чтобы усложнить их анализ и детектирование. Развивается черный рынок услуг по распространению мобильных вредоносных программ, что в ближайшем будущем приведет к увеличению числа атак на пользователей мобильных устройств, при этом атаки станут более изощренными. Это позволяет предположить, что проблем с нарушением безопасности МПУ станет много больше, что несомненно повлияет на благосостояние пользователей и компаний.

Тем не менее, необходимо комплексно обеспечивать безопасность всех данных, циркулирующих в сфере мобильных устройств. На сегодняшний день не существуют четких позиций по использованию технологий защиты МПУ и практически отсутствуют научно обоснованные рекомендации. По мнению разработчиков антивирусного программного обеспечения необходимо использовать накопленный опыт применения средств и методов защиты в компьютерных сетях и экстраполировать его на обеспечение конфиденциальности, целостности и доступности информации в МПУ.

Литература

1. Захарчук И. И., Веселов Ю. Г., Еремеев М. А. Проблемы защиты мобильных персональных устройств от информационно-технического воздействия URL: http://techno-new.developer.stack.net/pdf/404286.html?__s=1 (дата публикации: май 2012).

2. Шевченко А. Появление и развитие вирусов для мобильных устройств URL: http://www.securelist.com/ru/analysis?pubid=170531631 (дата публикации: 27 сен 2005).

3. Информационный бюллетень компании «Доктор Веб». Мобильные устройства как угроза безопасности сетей компаний и домашних компьютеров пользователей URL: http://news.drweb.com/?i=2468&c=5&lng=ru&p=3 (дата составления: 15 мая 2012)

4. Наместников Ю. Развитие информационных угроз во втором квартале 2012 года URL: http://www.securelist.com/ru/analysis/208050763/Razvitie_informatsionnykh_ugroz_vo_vtorom_kvartale_2 (дата публикации: 8 августа 2012).
5. Android Security Overview. http://source.android.com/tech/security/index.html (дата доступа: 3 сентября 2012)
6. Jon Oberheide. Exploit Mitigations in Android Jelly Bean 4.1. URL: https://blog.duosecurity.com/2012/07/exploit-mitigations-in-android-jelly-bean-4-1/ (дата публикации: 16 июля 2012)
7. Максвелл С. Ядро Linux в комментариях. ДиаСофт, 2000.

УДК 004.75

РЕАЛИЗАЦИЯ ВЫЧИСЛИТЕЛЬНОГО КЛАСТЕРА В УЛГУ

Д.М.Ахметов, А.А.Чичев

Ульяновский государственный университет

Введение

В лаборатории «ОС и системного ПО» кафедры ИТ в рамках дипломной работы «Реализация вычислительного кластера на базе ОС Linux» (студент Ахметов Д. М., группа ПРИ-51) был реализован вычислительный кластер на базе набора библиотек и технологий OpenMPI. При реализации данного кластера компьютеров были применены технологии управления конфигурациями группы компьютеров, выполнения распределенных программ и методы удаленного администрирования компьютерных систем, находящихся в локальной сети университета.

Кластер полностью построен на широко распространенных аппаратных компонентах. На вычислительных узлах применен дистрибутив операционной системы Alt Linux Centaurus 6 (конфигурация server), а на консоли управления кластером Alt Linux 5 (конфигурация workstation). Система легко масштабируется и, при необходимости, может сравняться и превзойти по производительности существующие научные и промышленные распределенные вычислительные системы Ульяновской области. В лабораторных условиях кластер продемонстрировал хорошие результаты в прикладных задачах, таких как подбор паролей и биологическое моделирование молекул белка, что может быть интересно как будущим специалистам в области защиты данных, биологам, физикам, химикам заинтересованным в работе с пакетами моделирования биологических, физических, химических процессов, так и программистам, заинтересованным в разработке, отладке, тестировании и применении своих распределенных программ, которые могут выполняться на этом кластере.

1. Структура кластера

1.1. Узлы (ноды) кластера.

В качестве вычислительных узлов и консоли (компьютера, с которого осуществляется управление кластером при физическом доступе к кластеру) были использованны системные блоки обычных персональных компьютеров, сходных по характеристикам, используемым компонентам и объему установленной оперативной памяти. Все они реализованы на базе процессоров intel celeron с тактовой частотой 2.4Ghz и имеют 512-1024 мегабайта памяти.

Узлы не имеют устройств ввода-вывода, ориентированных на работу человека непосредственно за компьютером, таких как клавиатура и монитор, однако подключение этих устройств к узлам возможно, что делает возможным их одновременное использование в качестве персональных компьютеров в случае отключения узла от кластера. Консоль управления кластером имеет мышь, клавиатуру и монитор для управления задачами при непосредственном доступе. Управление кластером с компьютера-консоли может осуществляться через текстовый или графический интерфейс (использовался оконный менеджер KDE версии 3.5). В качестве графического адаптера для подключения монитора узлы используют видеокарту Nvidia GeForce MX440 с 64мб видеопамяти и возможностью аппаратного ускорения 3d, подключенную к шине AGP 8x.

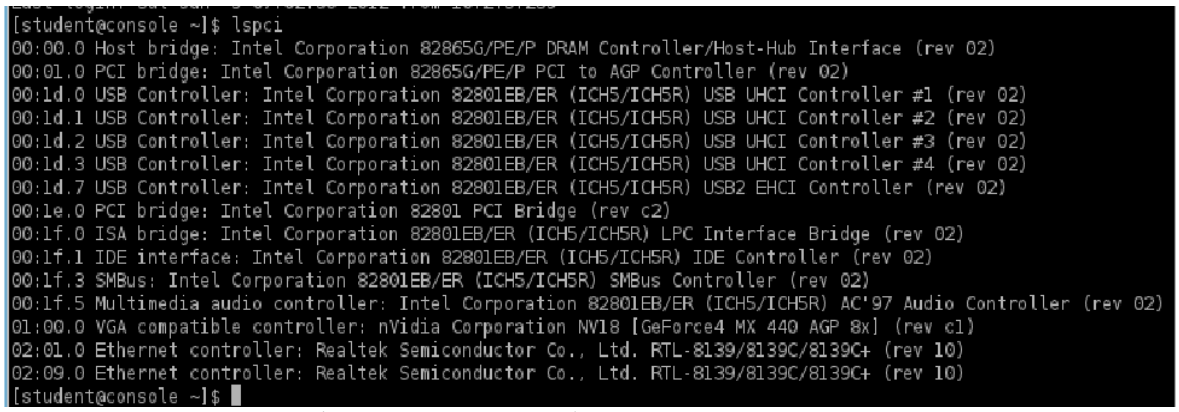


Рис. 1. Интегрированное оборудование системной платы.

Узлы собраны на системных платах intel 848P-A формата ATX (рис. 1), поддерживающих процессоры intel pentium (сокет 478). Платы имеют два слота для оперативной памяти, BIOS типа Phoenix, чипсет Intel 848P ICH5, северный мост Intel 82848P MCH, встроенные ide и sata-контроллеры, а так же встроенную аудио-карту, usb2.0-интерфейс и сетевой адаптер RTL-8139/8139C/8139C+, поддерживающий стандарты ethernet 10/100 Mbit. Установлен центральный процессор — intel celeron 2.4 Ghz, поддерживающий большинство современных низкоуровневых инструкций, которые могут использовать приложения, скомпиллированные в данной или аналогичной системе с процессором intel (рис. 2).

Рис. 2. Функциональность процессора.

Консоль кластера выступает шлюзом, поддерживая два сетевых интерфейса, локальный и внешний. Это дает возможность доступа кластера и узлов в интернет, а так же доступность всего кластера из локальной сети университета и даже из интернета, что делает возможным удаленное использование кластера.

Общий вид кластера приведен на рисунке 3.

Рис. 3. Общий вид кластера. Выполняются работы по настройке нод.

1.2. Организация сетевой подсистемы

Вычислительные узлы и консоль кластера объединены в локальную сеть. Узлы и консоль связывает один сетевой интерфейс, как это показано на рис. 4. Сеть построена на основе шестнадцатипортового коммутатора compex DS-2216 (рис. 5). В кластере используется сеть типа Ethernet 100Mbit, консоль кластера поддерживает два сетевых интерфейса (один интерфейс подключен к кластеру, второй — в сеть УлГУ), что позволяет осуществлять удаленное управление кластером. Учитывая специфику реализации кластера и количество узлов, такую конфигурацию сети можно считать оптимальной и экономически целесообразной, так как при использовании сети данного типа и примененном количестве узлов падение производительности не критично и ощутимо проявит себя только при росте количества узлов.

Избежать падения производительности из-за влияния сетевой подсистемы можно:

- применив сетевые адаптеры и коммутационное сетевое оборудование стандарта gigabit ethernet или 10 gigabit ethernet;
- использовав компьютерную сеть другого типа (Myrinet, Infiniband и другие, созданные специально для применения в распределенных вычислительных системах),
- использовав два сетевых интерфейса (для обмена данными и для управления, как это реализовано во многих известных mpi-кластерах).

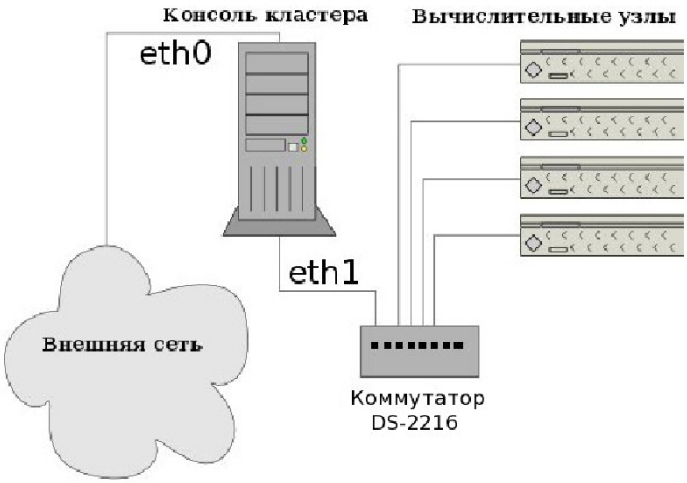


Рис. 4. Сеть кластера с одним сетевым интерфейсом.

Таким образом, в вычислительном кластере лаборатории «ОС и системного ПО» кафедры ИТ было применено то вычислительное и сетевое оборудование, которое можно было бы применить при недостаточном финансировании и отсутствии спонсорской поддержки проекта, а именно персональные компьютеры предыдущих поколений бюджетного ценового сегмента, которые все меньше и меньше применяются в компьютерных классах и на рабочих местах организаций, а так же достаточно древний шестнадцатипортовый коммутатор Compex DS-2216 (рис. 5), поддерживающий стандарты Ethernet 10\100Mbit.

Несмотря на то, что использованные в проекте компоненты в настоящее время часто признаются неликвидными и не имеющими большой материальной ценности, в составе вычислительного кластера они обрели новое качество. Производительность кластера из 9 узлов составила свыше 4000 Мегафлопс (рис. 6), что сопоставимо с одним современным четырехядерным компьютером при применениии SMP. Следовательно, в составе распределенный вычислительной системы устаревшие на несколько поколений компьютеры вполне могут обрести вторую жизнь и обеспечить должную производительность в прикладных задачах.

Рис. 5. Коммутатор Compex DS-2216

Рис. 6. Измерение производительности кластера.

1.3. Использованное системное ПО

На всех узлах и консоли управления кластером применена операционная система linux, а именно ее отечественная реализация в дистрибутиве Альт. Alt Linux это семейство дистрибутивов Linux, выпускаемых компанией «Альт Линукс» и её партнёрами, основывающихся на разработках русскоязычной команды разработчиков ALT Linux Team. Основа решений и дистрибутивов Alt Linux — репозиторий Сизиф, один из пяти крупнейших в мире банков пакетов свободных программ. Большим достоинством данного дистрибутива является доступность большого количества русской документации, форума поддержки, репозитория пакетов программ для любых нужд и целей, а так же государственная поддержка проекта - это делает его оптимальным решением для образовательных учреждений и бюджетных организаций. Также компания разработчиков Alt Linux выпускает и публикует в том числе и специализированные решения. Например в 2008 году была выпушена редакция Alt Linux 4.1 для кластеров, использовавшаяся в суперкомпьютерах СКИФ [6].

На вычислительных узлах была применена операционная система Alt Linux 6 Centaurus в варианте установки для сервера, на консоли кластера была использована система Alt Linux 5 в варианте установки Workstation. Все машины были дооснащены компиляторами и библиотеками языков Си и Фортран, а так же заголовочными файлами библиотек этих языков и иными компонентами, необходимыми для работы компиляторов и библиотек. Выбранная при установке Alt Linux опция «включить набор инструментов разработки и сборки приложений» по умолчанию добавляет в установку все необходимые компоненты для компиляции программ кроме компиляторов и библиотек языка fortran. Добавить поддержку fortran 77/90 необходимо отдельно. Это можно сделать, загрузив и установив несколько дополнительных пакетов приложений через пакетный менеджер или вручную из репозитория Alt Linux [6].

Пакет OpenMPI, использованный для реализации вычислительного кластера, доступен в репозитории Сизиф, но в рамках дипломной работы оказалось предпочтительнее собрать OpenMPI вручную, так как это дало возможность включить дополнительные опции, избежать возможные несовместимости, а также применить используемую компилятором по умолчанию оптимизацию программы под регистры процессора конкретной машины, что немаловажно как для «слабых» бюджетных компьютеров, так и для высокопроизводительных систем, где пренебрегать любой возможностью оптимизации программ не стоит. Программный интерфейс OpenMPI актуальных стабильных версий 1.5.5 (применявшегося до 14 мая 2012) и 1.6 (с 15 мая 2012) собирается на Alt Linux 5 и 6 версий без каких-либо дополнительных вмешательств, однако если собирать программу в системе, не дооснащенной компонентами языка fortran, OpenMPI будет собран без возможности создания и выполнения OpenMPI-совместимых программ на фортране, что делает

использование программ, созданных на этом языке невозможным в построенной распределенной системе.

Для упрощения администрирования и управления конфигурациями кластера компьютеров, связанных локальной сетью, был применен инструмент Puppet (рис. 7). Puppet - это средство автоматического управления настройками и конфигурациями любым парком linux-совместимых машин с клиент-серверной архитектурой, написанное на Ruby и решающий проблему настройки группы компьютеров, когда настраивать каждый из них индивидуально неприемлемо. На каждой вычислительной ноде кластера работает клиент, делающий обращения к серверу по http или https за обновленными файлами конфигураций, и, когда обновления становятся доступны, применяет их на вычислительных узлах согласно установленному сценарию. Система Puppet применима не только для вычислительных узлов компьютерного кластера, но и для гетерогенных сетей организаций с серверами, управляемыми linux устройствами, мобильными компьютерами и стационарными компьютерами пользователей. Подобная автоматизация управления конфигурациями компьютеров позволяет повысить эффективность работы информационных систем и сократить время простоя систем при плановом обслуживании и перенастройке (например, при масштабировании сети или распределенной системы).

Рис. 7. Система Puppet.

Для автоматизации процесса администрирования и оптимизации операционной системы linux был применен скрипт, ограничивающий список запускаемых по умолчанию сервисов в системе. В дистрибутиве Альт по умолчанию включены все устанавливаемые сервисы, что делает загрузку операционной системы очень долгой, увеличивает объем использованной оперативной памяти и снижает обшую производительность систем. Чтобы избежать подобных последствий, сразу после установки Alt Linux на очередной узел кластера можно выполнять автоматизированный скрипт в каталоге /etc/rc.d/init.d/, который даст неактуальным в контексте проектируемой системы скриптам пуска системных сервисов права, недопускающие выполнения скриптов инициализации, невостребованных сервисов при запуске системы.

Текст скрипта, отключающего лишние сервисы, выглядит так:

```
chmod 660 ahttpd, anacron, avahi-daemon, bacula-dir, bacula-fd, bacula-sd, bluetoothd,
clamd, crond, cups, dhcpd, dictd, freenx-server, functions, functions-bootsplash, functions-
compat, httpd2, lircd, mysqld, openvpn, outformatpostfix, postgresql, .provides.sh,
pulseaudio, random, slapd, smartd, smb, splash, squid, template, virtualbox, wine,
wine.outformat, wpa_supplicant
```

2. Примеры решения прикладных задач

Вычислительный MPI-совместимый кластер, построенный в лаборатории «ОС и Системного ПО» кафедры ИТ УлГУ в рамках дипломной работы, способен решать многие прикладные задачи, требующие высокой вычислительной мощности. С точки зрения пользователя и выполняемого приложения вычислительный кластер представляется единой вычислительной системой, что удобно для конечного пользователя и не требует от пользователя навыков работы в распределенной среде. К числу подобных задач обычно относят задачи перебора значений, такие как подбор пароля к имеющимся хешам, математические преобразования, задачи моделирования (химического, физического, биологического и т. д.) и некоторые другие.

Эти задачи решают как коммерческие, так и свободные программы и программные комплексы, такие как AMBER (моделирование молекулярной динамики), GROMACS (моделирование состояния молекул), FFTW (преобразования фурье для мультимедиа-библиотеки), Parallel Computing Toolbox (коммерческая среда математического моделирования для многопроцессорных систем и кластеров (рис. 8)), FLAME (агент-ориентированное моделирование), CASTEP (моделирование свойств материалов), Polcoms (моделирование водных экосистем), FRW (моделирование погодных условий на основе имеющихся данных), NWCHEM (химические задачи (рис. 9)).

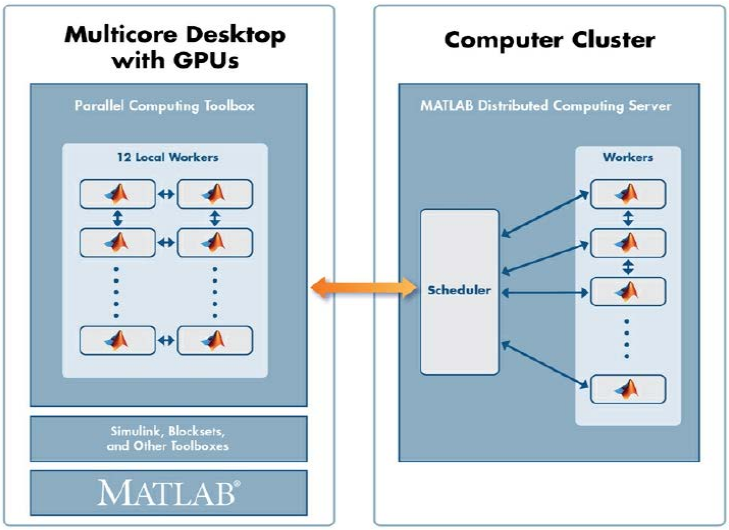


Рис. 8. Matlab - среда математического моделирования для распределенных систем.

В рамках дипломной работы на вычислительном кластере лаборатории «ОС и Системного ПО» кафедры ИТ УлГУ были скомпилированы и применены программные комплексы John и Gromacs. John это программа подбора хешей методом брутфорса или по словарю к md5-хешу, имеющая возможность компиляции и выполнения в распределенной системе OpenMPI. Для того, чтобы получить работоспособную программу, необходимо загрузить дистрибутив программы в виде исходного кода на консоль кластера и произвести компиляцию программы компилятором приложений распределенной среды из установленного ранее пакета OpenMPI. Программа John продемонстрировала отличные вычислительные возможности распределенной системы, показав близкую к линейной зависимость скорости подбора пароля от количества задействованных вычислительных нод.

Для того, чтобы компилляция программы john была выполнена с возможностью выполнения программы на вычислительном кластере, в сценарии сборки программы необходимо изменить компиллятор gcc на mpicc (являющийся частью установленной ранее программы OpenMPI). Когда john «соберется», исполняемый файл программы может быть выполнен как самостоятельно (запустить вручную исполняемый файл ./john в директории run проекта), так и с помощью mpirun, который необходим для выполнения программы в распределенной системе кластера.

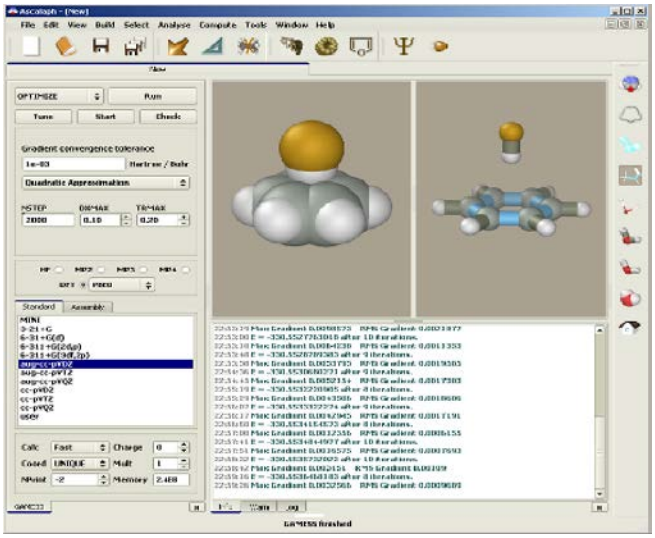


Рис. 9. NWCHEM - программный комплекс моделирования химических реакций.

Чтобы запустить подбор паролей на 9 узлах, в командной строке необходимо выполнить:

```
mpirun -hostfile l -np 9 ./john ~/root_hash
```

где
-hostfile указывает на текстовый файл со списком допустимых узлов,
-np 9 указывает на количество используемых узлов из списка допустимых,
./john – исполняемый файл программы, поддерживающей распределенное выполнение,
~./root_hash – текстовый файл, содержащий md5-хеш пароля пользователя root локальной системы, взятый из /etc/shadow (рис.10).

```
-bash-3.2$ mpirun -hostfile l -np 9 ./john ~/root_hash
Loaded 1 password hash (OpenBSD Blowfish [32/32 X2])
123QWE           (root)
guesses: 1  time: 0:00:09:16 DONE (Sat Jun  9 04:41:41 2012)  c/s: 48.50  trying: BLINK182 - 123QWE
Use the "--show" option to display all of the cracked passwords reliably
```

Рис. 10. Подбор пароля с помощью john на 9 узлах кластера.

Шестизначный несловарный пароль пользователя root был подобран за несколько минут, в то время как на одной машине, сравнимой по производительности с одним узлом кластера, подбор пароля занял бы значительно большее время.

Также, был настроен и протестирован программный комплекс Gromacs, позволяющий увидеть состояние молекул белков при заданных условиях в будущем.

Gromacs был также собран из исходного кода. Непосредственно перед началом компиляции gromacs были установлены необходимые зависимости: fftw3 и gsl. Для сборки fftw3 с поддержкой распределенных систем необходимо указывать следующие парамеры:

```
./configure --prefix=$SOFT --enable-long-double --enable-mpi —enable-openmp
make & make install
```

Параметры для сборки некластерной части gromacs будут:

```
./configure --with-gs --with-fft=fftw3
make & make install
```

Параметры для сборки части gromacs, отвечающей за работу в распределенных системах:

```
./configure --with-gsl --with -fft=fftw3 --program -suffix=_mpi_d \
--enable-mpi —enable-tmpi -profiling
make mdrun & make mdrun-install
```

Для удостоверения в правильности настройки, сборки и установки gromacs необходимо выполнить автоматизированный тест системы, который доступен на сайте gromacs в разделе self-test. Установка и настройка gromacs в распределенной компьютерной системе лаборатории университета прошла автоматизированные тесты:

```
[mpi@console ~]$ ./gmxtest.pl all
All 16 simple tests PASSED
All 13 complex tests PASSED
All 63 kernel tests PASSED
[mpi@console ~]$ ./gmxtest.pl -double all
All 16 simple tests PASSED
All 13 complex tests PASSED
All 63 kernel tests PASSED
[mpi@console ~]$ ./gmxtest.pl -np 9 all
Will test on 9 nodes
All 16 simple tests PASSED
```

3. Что ещё не сделано по теме

Ниже перечислены некоторые вопросы, которые могут быть выполнены в рамках курсовых и дипломных работ:

1. Пересобрать ядра ОС для консоли и, особенно важно, для узлов.
2. Добавить вторую сетевую подсистему.
3. Заменить сеть на gigabit ethernet.
4. Исследовать влияние компонентов кластера (узлов, сетевой подсистемы) на производительность — найти, адаптировать, разработать тесты, провести замеры и обработать результаты.
5. Подобрать и подготовить для использования прикладное ПО.
6. Организовать доступ из локальной сети УлГУ (с компьютеров кафедр и подразделений, набережная Свияги), в том числе подготовить документацию для пользователей.
7. Решить вопрос доступа к кластеру из удалённых подразделений УлГУ.
8. Подготовить технологию программирования на C/C++/Fortran.

Литература

1. http://ru.wikipedia.org/wiki/Ломоносов_(суперкомпьютер)
2. http://parallel.ru/cluster/
3. http://www.computerra.ru/vision/602217/
4. http://www.udman.ru/virtlab/paraclete
5. http://www.udman.ru/iam/ru/project/6
6. http://www.altlinux.ru/solutions/state/supercomputer/

ОРГАНИЗАЦИЯ УДАЛЕННОГО ДОСТУПА К КЛАСТЕРУ

Д.М.Ахметов, А.А.Чичев

Ульяновский государственный университет

В лаборатории «ОС и системного ПО» кафедры ИТ в рамках дипломной работы «Реализация вычислительного кластера на базе ОС Linux» (студент Ахметов Д. М., группа ПРИ-51) был реализован вычислительный кластер на базе набора библиотек и технологий OpenMPI. В данной статье рассматривается реализация удаленного доступа к кластеру.

Для управления кластером был реализован метод удаленного администрирования вычислительных систем, расположенных в локальной сети университета, путем тунелирования протоколов ssh и vnc через http-прокси-сервер для обеспечения доступа к кластеру из других подразделений университета. Удалённый доступ возможен как к текстовой командной строке каждого узла (включая консоль кластера), так и к графическому интерфейсу консоли кластера. Удаленная консоль реализована с помощью протокола SSH, удаленный рабочий стол — с помощью протокола VNC.

Если универсальность, единство и межплатформенность ssh не вызывает сомнений и часто делает его единственным правильным выбором в случае, когда необходима удаленная консоль для unix-подобной машины или компьютера с windows+cygwin, то протоколов удаленного доступа к рабочему столу компьютера имеется несколько. Как правило, этот протокол свой в каждой операционной системе, но есть и межплатформенные решения. Одно из таких — протокол VNC. Будучи распространяемым по лицензии GNU GPL он имеет открытый исходный код и реализацию как клиента, так и серверной части во многих операционных системах.

Если доступ к кластеру из локальной сети университета организовать достаточно просто, то реализовать стабильный доступ к кластеру из интернета оказалось сложной проблемой:

- доступ в интернет осуществляется через fw и прокси-сервер, ограничивающие абсолютно весь трафик из сети университета в интернет и из интернета к университетским компьютерам. Разрешены только протоколы http и https, а так же порты 80, 8080 и 443, которые стандартно их используют, соответственно сервисы ssh, ftp, vnc и другие запрещены и даже ping ресурсов интернета невозможен,
- в сети университета компьютеры имеют «серый» ip-адрес, то есть снаружи можно обратиться только к подсети, а не к компьютеру в ней,
- ограниченное время жизни http-https-соединения при отсутствии трафика через него.

Решение первого пункта реализуется с помощью пакета Corkscrew. Это инструмент тунелирования протокола ssh через различные прокси-серверы, существующий для многих операционных систем и способный пробросить ssh через почти любой широко используемый http-прокси. Corkscrew ставится на консоль кластера из репозитория AltLinux, после чего редактируется конфигурационный файл ~/.ssh/config, содержащий две строчки с информацией о хосте, до которого необходимо пробросить ssh, ip-адресе прокси в локальной сети и номере порта, на котором он работает. На удаленном хосте ssh должен принимать подключения по одному из доступных в защищенной прокси локальной сети портов, в нашем случае это будет 443, 80 или 8080. Листинг конфигурационного файла corkscrew приведен ниже:

```
Host <полное имя host'a в интернете>
ProxyCommand corkscrew  <ip-адрес прокси>  <порт прокси>  %h  %
```

Теперь с консоли кластера возможно подключение по ssh к одному удаленному хосту через тоннель при запрещенном ssh-трафике, однако это еще не делает компьютер лаборатории доступным из интернета.

Второй пункт удалось решить, пробросив локальный порт ssh-сервера консоли кластера на удаленную машину (удаленной машиной можно считать компьютер где-то в интернете). Сделать это оказалось возможным стандартными средствами программы ssh. Это позволило иметь ssh-соединение между компьютером в лаборатории и удаленным компьютером, несмотря на то, что иными средствами компьютер, имеющий «серый» ip-адрес, из глобальной сети быть доступен не может. Для проброса локального порта ssh-сервера на другой ssh-сервер, беспрепятственно доступный через интернет, на нем достаточно выполнить команду:

```
ssh -R [пробрасываемый_порт]:локальный_ip:[локальный_порт] –p
[удаленный_порт] удаленный_хост
```

В локальной сети университета компьютеры получают адреса и базовую информацию об используемой сети по протоколу dhcp, при этом адреса выделяются динамически. На удаленном компьютере ssh открыт на портах, доступных из сети университета для того, чтобы было возможным подключение через http-прокси.

Пробросив локальный порт ssh-сервера консоли кластера на удаленную машину, подключиться с удаленной машины к компьютеру в лаборатории можно будет по протоколу ssh, обратившись по ssh к localhost на порту, который был проброшен.

Третья проблема состоит в том, что адрес, используемый компьютером лаборатории может динамически меняться, что сделает невозможным переподключение при перезапуске dhcp-сервера, включения кластера в работу после перезагрузки консоли по причинам отключения энергии и другим причинам, вызывающим смену ip-адреса. Кроме того, время жизни соединения через http-прокси ограничено при отсутствии трафика. То есть, если между консолью кластера и удаленным сервером в интернете не передаются данные по ssh или scp, http-прокси оборвет связь без возможности автоматического восстановления ее стандартными средствами. Для решения проблемы был написан shell-скрипт командной оболочки операционной системы linux. Скрипт выполняется вместе с сервисами операционной системы на консоли кластера, инициируется при включении и загрузке оконного менеджера KDE, автоматически пробрасывает локальный порт ssh на удаленную машину, передавая динамически получаемый ip-адрес программе ssh, а также восстанавливает соединение, как только прокси-сервер обрывает его по причине отсутствия трафика между локальным и удаленным компьютерами. Так как скрипт фактически открывает командную строку на удаленной машине, поскольку это требуется для проброса локального порта ssh, включить его в системные службы невозможно, потому что открывшаяся удаленная командная строка останавливала бы инициализацию других служб «ожиданием» ввода команд от пользователя. Поэтому скрипт выполняется от имени локального пользователя user в окне виртуального терминала. Для того, чтобы автоматически позволить выполнять скрипты от пользователя в виртуальном терминале, был реализован автоматический вход в систему для пользователя user при включении или перезагрузке консоли кластера. Сделано это с помощью программы autologin, доступной в репозитории Alt Linux.

Для функционирования программы autologin, кроме установки ее, потребовалось создать кофигурационный файл **/etc/sysconfig/autologin** следующего содержания:

```
AUTOLOGIN=yes
USER=<логин пользователя>
```

где первая строчка указывает, что автоматический вход в систему включен, а вторая сообщает имя пользователя, вход в систему для которого будет происходить автоматически.

Скрипт, реализующий проброс порта ssh через http-прокси с передачей динамического ip-адреса при завершении соединения по тайм-ауту прокси выглядит так:

```
#! /bin/sh
i=0
# флажок, передающий состояние соединения, по умолчанию 0 (активно)
while test 1==1
# выполняем в бесконечном цикле все что далее
do
remote_ip=<полное имя host'a в интернете> # удаленный хост
remote_user=<логин пользователя> #пользователь на удаленном хосте
exist=`ps aux | grep $remote_user@$remote_ip | grep 9090`
# exist это переменная имеющая значение тогда, когда активно
# соединение
ip=`ifconfig  | grep 'inet addr:'| grep -v '127.0.0.1' | cut -d: -f2 | grep 10 | awk '{ print $1}'`
# ip это переменная, передающая динамический ip-адрес на внешнем  сетевом интерфейсе консоли кластера
echo "$ip"
# выводит актуальный адрес на экран
if test -n "$exist"
# если то, что соединение «живо» истина
then
if test $i -eq 0
# и при этом флажок активности соединения имеет значение 0
then
echo "Соединение активно с $(date)"
# будет выведено сообщение об активности соединения
# с данного момента времени fi
i=1
# в противном случае присваиваем флажку активности состояние,
# сигнализирующее неактивность
else
i=0
# либо (когда переменная exist говорит о неактивности)
echo "Остановлено... Подключение повторно"
# выведем сообщение о повторном подключении
ssh -R 9696:$ip:22 -p 8080 $remote_user@$remote_ip
# подключаемся по ssh с пробросом локального порта на
# удаленную машину, пробрасываемый порт 9696,
# локальный порт ssh консоли кластера 22,
# используемый для подключения порт 8080.
fi
sleep 1
# пауза программы в 1 секунду при любом из результатов
done
# конец алгоритма и программы.
```

Скрипт будет называться script и находиться в домашнем каталоге пользователя user. Для того чтобы он выполнялся как программа, а не был просто текстовым файлом, ему необходимо дать права на выполнение.

Сделать это можно так:

```
chmod +x /home/student/script
```

Для доступа к удаленному рабочему столу используется реализация протокола vnc, называемая x11vnc. Она доступна в репозитории Сизиф. После установки программы из

репозитория необходимо задать пароль доступа к vnc-серверу (удаленному рабочему столу консоли кластера). Сделать это можно так:

```
x11vnc  -storepasswd
```

SSH также позволяет пробросить локальный порт любого другого приложения на удаленную машину. Проблемы, которые возникают при пробросе удаленного порта vnc такие же, какие возникают при пробросе ssh через http-proxy, поэтому реализует проброс с передачей ip и переподключением при обрыве соединения через прокси по таймауту аналогичный скрипт. Текст скрипта:

```
#! /bin/sh
i=0
while test 1==1 do
   remote_ip=<полное имя host'a в интернете>
   remote_user=<логин пользователя>
   exist=`ps aux | grep $remote_user@$remote_ip | grep 9090`
   ip=`ifconfig  | grep 'inet addr:'| grep -v '127.0.0.1' | cut -d: -f2 |grep 10 | awk '{ print $1}'`
   echo "$ip"
   if test -n "$exist"
                                                then
      if test $i -eq 0
                                                  then
                                    echo "Соединение активно с $(date)"
      fi
      i=1
   else
      i=0
   echo "Остановлено... Подключение повторно"
                ssh -R 5555:$ip:5555 -p 8080 $remote_user@$remote_ip
                                                                                  # vnc будет использовать порт 5555.
fi
sleep 1
done
```

Скрипт был назван script2 и размещен в домашнем каталоге пользователя кластера. Скрипту script2 были даны права на выполнение.

Для автоматической инициализации скриптов в окнах виртуального терминала при запуске оконного менеджера после автоматического входа пользователя в систему необходимо разместить скрипты, вызывающие алгоритмы переподключения при остановке соединения прокси-сервером в виртуальном терминале. Скрипты автозапуска оконного менеджера kde находятся в ~/.kde/Autostart.

Вызов скрипта script в окне виртуального терминала при загрузке оконного менеджера осуществляется другим скриптом, который называется ssh и находится в ~/.kde/Autostart. Его содержание будет:

```
#!/bin/bash
konsole -e "~/script"
```

Первая строчка объявляет командную оболочку для выполнения shell-команд, вторая запускает виртуальный терминал с передачей ему имени скрипта script.

За инициализацию проброса vnc с поддержанием соединения при обрывах отвечает скрипт vnc из ~/.kde/Autostart. Содержание vnc будет:

```
#!/bin/bash
konsole -e "~/script2"
```

За старт vnc-сервера отвечает скрипт vncserver и находится он там же. Он содержит:

```
#!/bin/bash
x11vnc -rfbauth ~/.vnc/passwd -rfbport 5555
```

где
~/.vnc/passwd — путь к файлу шифрованых паролей,
-rfbport — используемый порт.

Ssh, vnc, vncserver должны иметь права на выполнение чтобы они могли быть выполнены автоматически при старте оконного менеджера kde.

РАЗРАБОТКА ТЕСТОВО-ОБУЧАЮЩЕЙ ПРОГРАММЫ НА ОСНОВЕ КОМБИНИРОВАННОЙ МОДЕЛИ ДЛЯ ЮРИДИЧЕСКИХ СПЕЦИАЛЬНОСТЕЙ

Ю.Е.Бочкарева, Н.А.Грачева

Ульяновский государственный университет

Стремительное развитие новых информационных технологий и применение их в различных сферах человеческой деятельности, в том числе и в правовой, обусловили необходимость приобретения будущими специалистами в области юридической деятельности знаний, включающих в себя понимание устройства и основных принципов работы персонального компьютера, изучение операционной системы и прикладных программ, знание современных принципов сбора, хранения и переработки информации.

Сегодня мы стоим на пороге создания качественно нового информационного общества. Жизнь и практическая деятельность в нем неразрывно связаны с грамотной организацией информационных процессов, освоением и использованием современных информационных технологий.

Эффективность работы будущего юриста во многом будет определяться тем, насколько умело и свободно он сможет использовать современные компьютерные информационные технологии в своей работе и насколько быстро он будет способен адаптироваться к их стремительному развитию. Одним из способов успешного усвоения новых знаний является использование программированного обучения в учебном процессе.

Программированное обучение — комплекс методов и средств обучения, принципом которого является получение знаний и навыков студентами за счет последовательного усвоения материла.

Существуют различные алгоритмы программированного обучения — линейная, разветвлённая, адаптивная, комбинированная и другие, реализованные с использованием компьютеров, электронных учебников, методических материалов. Основные принципы, которые должны быть реализованы в программированном обучении это последовательность, доступность, систематичность.

Разрабатываемая тестово-обучающая программа (ТОП) основана на комбинированном алгоритме. Так как он включает в себя элементы линейного, разветвленного и адаптивного программированного обучения. ТОП состоит из информационно-обучающего блока, блока тестирования, блока анализа результатов с рекомендациями о повторе той или иной темы и блока интегрированного тестирования по всей учебной дисциплине (рис.1).

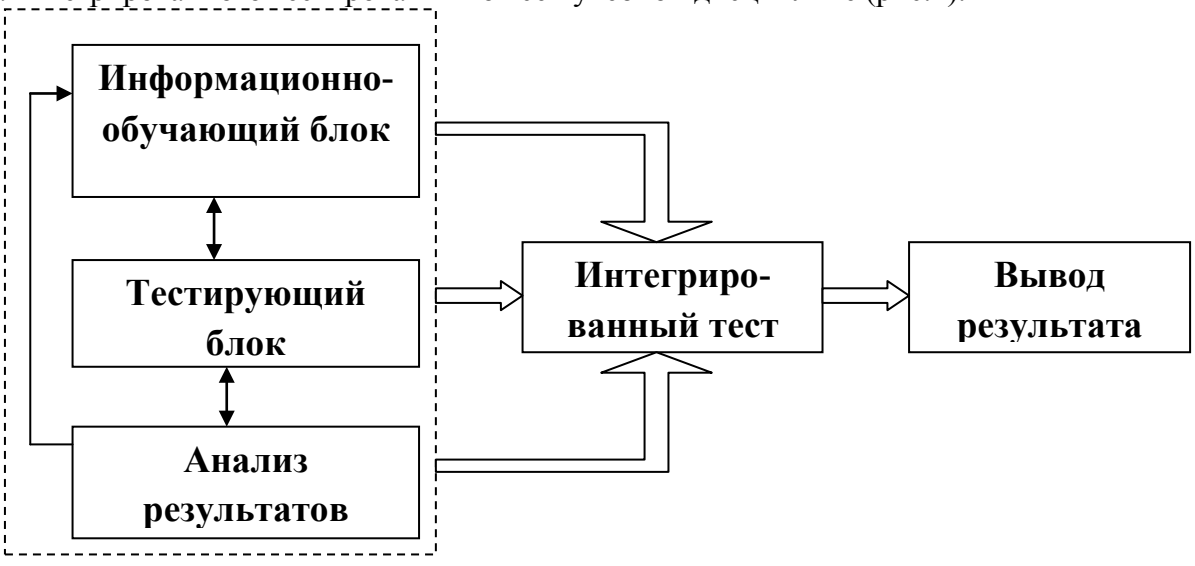


Рис.1 Тестово-обучающая программа

Информационно-обучающий блок включает в себя множество изучаемых тем представленных в виде текстового, графического, мультимедийного, табличного материала, с ссылками на электронные справочники, книги, словари и интернет ресурсы (рис. 2). Особенностью данного блока является возможность выбора обучаемым, в зависимости от уровня знаний, определенного раздела дисциплины.

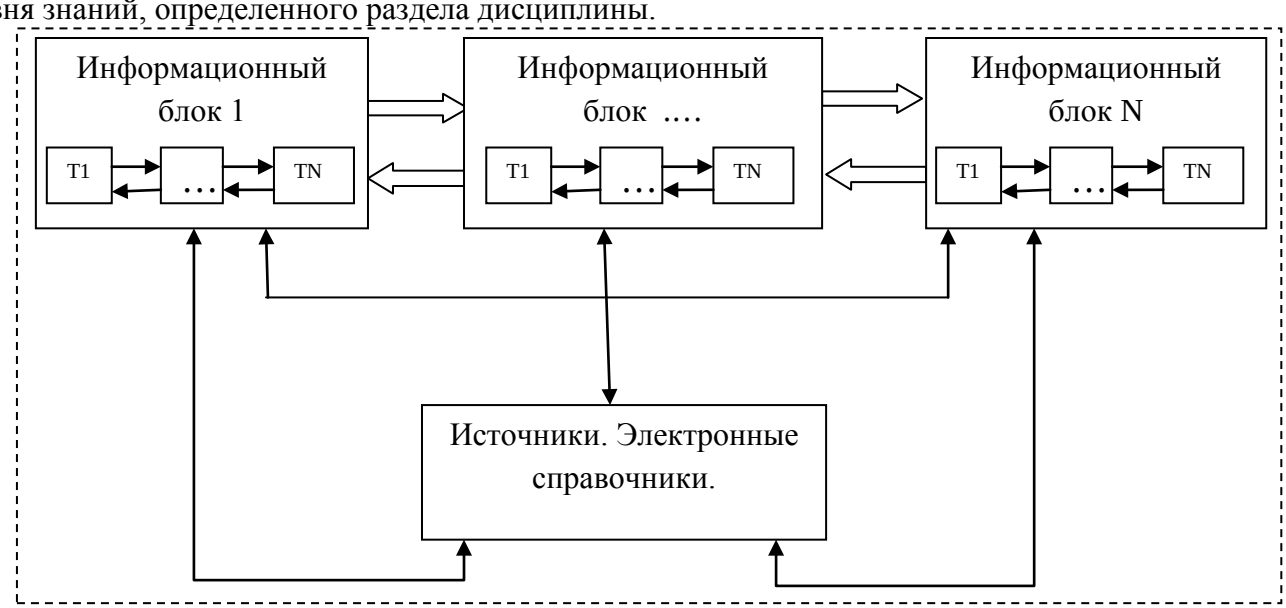


Рис.2 Информационно-обучающий блок (T1…TN – темы изучаемого раздела дисциплины)

Блок тестирования состоит из набора вопросов по каждой из изучаемых тем. К каждому из вопросов предлагаются на выбор несколько вариантов ответов (рис.3). Целью данного тестирования является обнаружение «пробелов» в усвоенных знаниях. Программа анализирует результат пройденного тестирования и в зависимости от него выдает рекомендации по дальнейшему использованию в ТОП информационно обучающего блока.

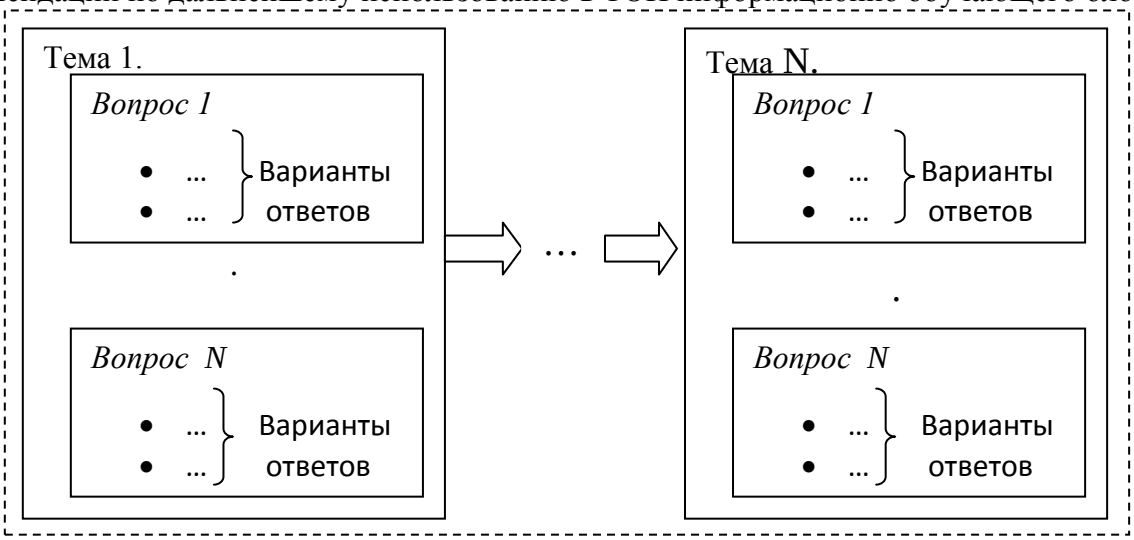


Рис.3 Блок тестирования

Интегрированный тест – аттестационная процедура, позволяющая максимально объективно оценить полученные знания за время изучения дисциплины. Сюда входят вопросы по всем разделам, пройденным в процессе обучения.

ТОП позволяет студентам эффективно усваивать большие объемы информации, так как она является уникальным информационным инструментом, позволяющая в доступной форме освоить сложный материал. Система предусматривает возможность обучения как студентов с низким уровнем знаний, так и студентов имеющих знания по данной проблеме.

В заключение, хотелось отметить ряд преимуществ в использовании ТОП это:

- оптимизация организации учебного процесса;
- высокая организация познавательной деятельности студента;
- увеличение количества получаемой информации студентом;
- реализация индивидуального подхода к обучению.

Таким образом, ТОП позволяет в высокой мере модернизировать образовательный процесс и повысить уровень усвоения новых знаний студентом.

Литература

1. Батешов Е.А. Основы технологизации компьютерного тестирования: Учебное пособие. – Астана: ТОО «Полиграф-мир», 2011. - 241 с.

2. Бочкарева Ю.Е., Грачева Н.А. Введение компьютерной тестово-обучающей системы для юридических специальностей/ Бочкарева Ю.Е., Грачева Н.А. //Актуальные проблемы современного образования: опыт и инновации: материалы Международной научно практической конференции (заочной): 25-26 ноября 2009 г. Часть 2. / отв. ред. А.Ю. Нагорнова; ГОУ УлГПУ им. И.Н. Ульянова. – Ульяновск: УлГУ, 2009.- С.50-51.

3. Запорожко, В.В. Готовность будущих учителей информатики к разработке и применению компьютерных средств обучения. / В.В. Запорожко // Педагогика, лингвистика и информационные технологии: Матер. МНПК. – Елец: ЕГУ , 2007. – Т. 2. – С. 238-244.

4. Красильникова В.А. Теория и технологии компьютерного обучения и тестирования. Монография/ В.А. Красильникова. – Москва: Дом педагогики, ИПК ГОУ ОГУ, 2009. – 339 с.

5. Рычкова, А.А. Развитие профессиональной самостоятельности будущих инженеров-программистов на основе дистанционных образовательных технологий / А.А. Рычкова //Проблема качества образования и профессиональной компетентности в условиях инновационного образовательного процесса: Материалы международной научно-практической конференции. – Сургут, 2008. – С. 75-88

ПРОБЛЕМЫ РАСПРЕДЕЛЕНИЯ ТРАФИКА И БЕЗОПАСНОСТИ В СЕТЯХ MPLS

Н.В.Будылдина, Н.П.Егорова

Уральский технический институт связи и информатики (филиал) ГОУ ВПО «Сибирский государственный университет телекоммуникаций и информатики»

Несмотря на то, что технология MPLS достигла значительного прогресса и обеспечивает широкий диапазон функциональных возможностей и приложений, ее реализация связана с большими сложностями. Производители, разрабатывающие технологию MPLS, а также организации, в настоящее время развертывающие MPLS сети, должны также учитывать состояние постоянного развития этой технологии и, ее влияние на производительность, и расширяемость сети [1].

Как было сказано выше, MPLS не является самостоятельной - она накладывается на технологии 2-го уровня, такие как Ethernet или ATM, и должна работать совместно с другими протоколами плоскости управления, такими как протоколы маршрутизации IP. Сложность развертывания MPLS возрастает из-за этого взаимодействия. В некоторых случаях в заданный сетевой сценарий может быть вовлечено четыре или более протоколов, требующих тщательной координации и подтверждения работоспособности сквозной системы. Интеграция традиционных услуг и развертывание новых, таких как VPN, требует туннелирования, которое, в свою очередь, расширяет требования настройки для данной сети [1]. Поэтому, возможность взаимодействия оборудования MPLS в разнородных сетях остается проблемой. Хотя достижения в технологии интегральных схем значительно улучшили характеристики современных маршрутизаторов, сложность MPLS в реальных сетевых приложениях вызывает проблемы с рабочими показателями и расширяемостью сети. Проблемы обычно возникают не в базовой сети MPLS, где данные просто переключаются с использованием меток, а на краю сети, где MPLS должна интегрироваться с не-MPLS сетями, и где инициируются услуги. Из-за объединения сетей загрузка трафика возрастает, и сети должны справляться с дополнительными задачами обработки трафика в реальном времени и трафика с приоритетом. Вот почему основной задачей при построении сетей на основе технологии MPLS остается оптимизация трафика.

Рассматривая более детально архитектуру протокола MPLS необходимо отметить, что в основном некоторые проблемы связаны с обеспечением безопасности при создании сети, а именно с несанкционированным доступом, несовершенной конфигурацией самой сети, атаками внутри сети, подмене меток и т.д.

Архитектура данного протокола обеспечивает защищенность на втором уровне с использованием обычных протоколов типа ATM или Frame Relay.

Базовый принцип защиты основывается на сокрытие структуры ядра MPLS сети. Из соображений безопасности операторы и заказчики обычно не хотят открывать сетевую топологию внешним сторонам. Это значительно снижает вероятность атак на сетевую инфраструктуру. Зная IP-адресацию, потенциальный злоумышленник в состоянии организовать атаку типа отказ в обслуживании направленную против сетей заказчика, или ядра MPLS сети. В общем, атаки на данный протокол (и сети, построенные на базе данного протокола) можно разделить на два типа:

- атаки типа отказ в обслуживании (denial-of-service, DoS),
- атаки, направленные на взлом самой сети (для получения информации).

Учитывая выше сказанное на практике должны быть предприняты многочисленные дополнительные меры безопасности, в основном по фильтрации пакетов.

Для защиты от атак второго вида, существует два базовых типа защиты: усиление защищенности самих протоколов, обеспечивать "невидимость" самой сети как таковой (использование межсетевых экранов и пакетных фильтров).

Для защиты от DoS атак наиболее устойчивый способ защиты - обеспечение

"невидимости" самой сети извне (использование межсетевых экранов или трансляции сетевых адресов (NetWork Address Translation,NAT) т.е. маршрутизаторы скрывают детали домашней сети).

MPLS распространяет наружу только необходимую информацию, это относится и к VPN клиентам. Адресация ядра MPLS сети может быть выполнена с использованием как частных (RFC 1918), так и публичных адресов. Так как в основном выходной интерфейс VPN сети - потенциально может быть Интернет – здесь работает протокол BGP, то наружу остальную внутреннюю информацию показывать не обязательно. Однако, если между пользовательским маршрутизатором (customer edge, CE) и граничным маршрутизатором провайдера (provider edge, PE) ядра MPLS сети используется протокол маршрутизации, то может еще и передаваться информация о маршрутах, тогда единственной требуемой информацией является адрес PE маршрутизатора. Если требуется этого избежать, то между PE и CE можно сконфигурировать статическую маршрутизацию. В этом случае ядро MPLS сети может быть полностью скрыто. В случае VPN услуги с одновременным разделяемым доступом в Интернет оператор, обычно, объявляет адресную информацию клиентов, желающих использовать Интернет вышестоящим или одноранговым провайдером. Сокрытие подобной информации может быть организовано с использованием NAT функциональности (трансляции адресов). Оператор в этом случае объявляет только адреса своего пограничного PE- маршрутизатора.

В случае использования "чистой" сети на базе MPLS-VPN сервиса, где нет подключения к Интернету, защищенность такая же как и у протоколов ATM/FR. При подключении к сети Интернет, провайдер обязан "открыть" хотя бы один адрес PE - маршрутизатора, что может повлечь за собой атаку.

Сама базовая сеть MPLS может атаковаться двумя способами:

- нападением на PE маршрутизатор;
- попытка навязывания ложных маршрутов.

Для атаки необходимо иметь адрес хотя бы одного маршрутизатора. При этом если сеть "спрятана", то атака не состоится - сеть будет "думать", что пакет атакующего - это лишь информация пользователя для передачи.

При использовании статических маршрутов между CE и PE задача для атакующего значительно усложняется. Однако при использовании протоколов маршрутизации типа RIP, OSPF, BGP для CE - маршрутизатора уже нужно знать как минимум идентификатор маршрутизатора (router ID, RID) PE в "базовой" сети MPLS, что означает упрощение задачи для атакующего. Для уменьшения риска атаки необходимо:

- использовать списки доступа (access control lists, ACL);
- возможно использовать аутентификацию с помощью Message Digest 5 в протоколах маршрутизации. Протоколы BGP, OSPF и RIP II поддерживают данную аутентификацию;
- проводить конфигурирование параметров протоколов маршрутизации с учетом безопасности.

При этом необходимо учесть, что использование статической маршрутизации не полностью защищает от атаки. Конечно, здесь можно не указывать адрес, а только указать интерфейс, но при этом атакующий все таки может попробовать угадать адрес.

Все указанное может работать, только если сама сеть принадлежит только одному провайдеру, т.к. от атак изнутри MPLS не защищена. При этом, если сеть MPLS сконфигурирована без учета вопросов защиты, то возможность атаки повышается. Для повышения защищенности можно использовать протокол IPSec. Данный протокол может быть настроен на CE-маршрутизаторе или может использоваться другое устройство.

Существует еще одна проблема - возможность подменить метку. Потенциальный злоумышленник может постараться получить доступ в сеть VPN с использованием "чужих" меток. Это может быть как вне сети (например, пользовательский CE-маршрутизатор) или внутри сети MPLS. Попытка послать к CE-пакет с "чужой" меткой в сеть MPLS, через

граничный маршрутизатор провайдера (PE) может увенчаться успехом. Так как CE-маршрутизатор не знает о существовании MPLS сети и считает, что он посылает IP- пакеты обычному маршрутизатору. Вся интеллектуальная работа делается PE устройством, где, в зависимости от конфигурации, выбирается и назначается нужная метка. Из соображений безопасности PE - маршрутизатор никогда не примет от пользовательского CE - маршрутизатора пакет с меткой. В маршрутизаторах Cisco подобные пакеты с меткой будут просто сброшены. Таким образом, невозможно вставить ложные метки в пакеты, так как никакие пакеты с метками не принимаются маршрутизатором [2].

Рассматривая далее проблемы протокола MPLS необходимо отметить, что протокол позволяет использовать в различных VPN одно и тоже адресное пространство, например частное адресное пространство (RFC 1918). Это достигается путем добавления 8 байтовой метки определителя маршрута (route distinguisher, RD) к каждому IPv4 маршруту. Использование RD позволяет пользователю оставлять без изменения адреса своих сетей. Однако, если используются протоколы маршрутизации между CE и PE маршрутизаторами (это не относится к статической маршрутизации) то проблема защиты усложняется. При использовании протокола маршрутизации существует необходимость прописать адрес маршрутизатора (статическая маршрутизация) в ядре MPLS для обмена информацией с маршрутизатором PE. Данный адрес должен быть уникален, и принадлежать внутреннему адресному пространству сети VPN MPLS. В случае, когда провайдер управляет CE – маршрутизатором, этой проблемы нет (пользователь не увидит данной конфигурации).

В принципе в сети достигается разделение информации о маршрутах. Каждый PE маршрутизатор поддерживает для каждого соединения VPN свою таблицу Виртуальной Маршрутизации и Коммутации (virtual routing and forwarding instance, VRF). Каждая таблица VRF работает с маршрутами одной VPN, статическими или динамическими, если между PE и CE работают механизмы динамической маршрутизации. Так как к каждому VPN привязан свой VRF, нет никакого взаимодействия между различными VPN на PE маршрутах. Таким образом, в сети MPLS разделение информации о маршрутах достигается тем, что для каждого VPN есть свой VRF.

В маршрутизаторах всей сети (из нескольких PE и нескольких точек подключения виртуальных сетей) такое разделение поддерживается посредством добавления уникальных идентификаторов VPN в мультипротокольном BGP (MP-BGP), подобным определителям маршрутов. Маршруты VPN через MPLS сеть передаются посредством MP-BGP и распространятся на PE - маршрутизаторы в конкретные VRF[2]. Таким образом, маршрутизация в MPLS сети разграничена для каждой отдельной VPN.

Исходя из вышесказанного, использование данного протокола позволяет решить следующие задачи безопасности сети:

- аутентификация сторон;
- проверка неизменности данных в пакете;
- защита от атак – replay.

При этом использование протокола защиты может сводиться к трем типам:

- точка-точка;
- точка-многоточка;
- полная связность.

Тем не менее, протокол MPLS не может обеспечить защиту:

1. Против ошибок в конфигурации ядра сети и атак внутри ядра сети. Все оговоренные механизмы безопасности подразумевают правильную конфигурацию всех используемых сетевых элементов MPLS сети (P и PE маршрутизаторов). Намеренные или случайные ошибки могут привести к нежелательным результатам, включая серьезные нарушения безопасности;
2. Шифрование и целостность данных. MPLS не обеспечивает услуг по шифрованию, целостности и аутентификации. Если существует такая потребность необходимо

использовать дополнительные средства шифрования;

3. Безопасность сети клиента. Можно обеспечить уровень защиты MPLS, сравнимый с уровнем альтернативных технологий предоставления VPN сервиса. Однако, безопасность ядра сети - это только один фактор полной безопасности сети заказчика. Угрозы в сегодняшних сетях существуют не только извне, но и изнутри. Для достижения хорошего уровня безопасности сети клиента в MPLS инфраструктуре, безопасность MPLS необходима, но не достаточна [2].

Литература

1. M. Clouqueur, W.D. Grover. Availability analysis and enhanced availability design in p-cycle-based networks, Photonic Networks Communications, vol. 10, no. 1, pp. 55-71, July 2005.

2. Eusebi Calle, Jose L. Marzo, Anna Urra. Protection Performance Components in MPLS Networks. Accepted in Computer Communications Journal, Elsevier 2004.

МЕТОД ПОСТРОЕНИЯ ОТНОСИТЕЛЬНО НАДЕЖНЫХ СТЕГАНОГРАФИЧЕСКИХ СИСТЕМ

М.Ф.Валишин

Ульяновский государственный университет

Введение

Стеганография – древнейшее искусство скрытой передачи данных по открытым каналам. Ранний период, который некоторые исследователи называют классическим, характеризовался индивидуальным подходом в каждом методе, отсутствием общей терминологической базы, принципов построения и верификации. Только в конце 90-х годов в связи с ростом компьютерных сетей и сетевого трафика стеганография приобрела черты научности. В 1996 году на первой международной конференции по сокрытию информации Биргит Фитсман предложил единую терминологию и обобщенную схему функционирования стеганографической системы. Позже Кристиан Кашен опубликовал работу «Digital Steganography», в которой дал классификацию стеганографических алгоритмов. Задав вектор развития, категории Кашена так и не были полностью заполнены. В данной работе дается обзор предложенной классификации, рассматриваются её недостатки, фундаментальные принципы построения стеганографических систем, а также предлагается новый подход к разделению алгоритмов сокрытия данных, основанный на относительной надежности. В конце приведен практический пример построения подобного алгоритма.

Абсолютная надежность

Для сравнения стеганографических систем используется множество характеристик и свойств. В качестве основной характеристики для методов сокрытия информации выбрана надежность, как способность стеганографической системы противодействовать методам выявления факта скрытой передачи данных. Кашен рассматривал только один вид атак: атаку бинарным классификатором, при которой исследуемый контейнер маркируется как стего или не-стего. Эффективность атаки определяется ошибками классификации, которые бывают двух типов: ложноположительного обнаружения и «промаха». Первый тип ошибок имеет место, когда пустой контейнер помечается как стего, второй – когда стего помечается как пустой контейнер. Считается, что оба типа ошибок равнозначны и выводится усредненная ошибка классификации. Тем не менее, в ряде прикладных задач «лучше отпустить виновного, чем поймать невиновного», т.е. стоит вопрос о минимизации ложных срабатываний классификатора. Стеганографическая система надежна относительно бинарного классификатора, если в результате эмпирической проверки ошибка классификации не превосходит заранее оговоренное пороговое значение. Другой подход определения надежности заключается в сравнении с вероятностным бинарным классификатором, который маркирует контейнер случайным образом, например, методом подбрасывания монеты.

Согласно классификации Кашена существуют совершенно надежные системы, статистически надежные системы и вычислительно надежные системы. Совершенно надежные системы – системы, в которых стего не отличим от пустого контейнера любым неограниченным классификатором. Статистически надежные системы – системы, для которых существует классификатор, возможно, неограниченный, различающий стего от пустого контейнера с точностью, пропорциональной размеру секретного сообщения. Вычислительно надежные системы – системы, для которых существует полиномиальный по времени вероятностный классификатор, различающий стего от пустого контейнера с точностью, пропорциональной размеру секретного сообщения.

Классификация Кашена задала вектор развития стеганографии: от простого алгоритма к более совершенному через методы стегоанализа. Например, одним из ярчайших примеров эволюции методов стеганографии является череда алгоритмов Jsteg, F5,Outguess, HUGO.

На основе анализа данных алгоритмов, можно выделить три ключевых, фундаментальных принципа построения стеганографических систем: принцип сокрытия за порогом чувствительности, принцип минимизации искажений и принцип сохранения статистической модели. Первый принцип означает, что для внедрения секретного сообщения в контейнер используется наименее заметная для визуальной атаки область. Например, известно, что среднестатистический человек менее чувствителен к градациям синего цвета и незначительным искажениям цветности, поэтому многие алгоритмы используют LSB метод для внедрения сообщения. Принцип минимизации искажений основан на уменьшении разницы между исходным контейнером и стего за счет применения специальных техник кодирования. В алгоритме F5 впервые была применена техника матричного кодирования. Развитие подхода к уменьшению искажения, вызываемого стеганографической системой, привело к созданию универсального метода минимизации влияния вложения с помощью решетчатых кодов. Наконец, принцип сохранения статистической модели является защитой от определенных методов стегоанализа, основанных на проверке некоторых статистических гипотез. Созданная против алгоритма Jsteg RS-атака основана на эвристическом предположении о постоянстве соотношения между регулярными и сингулярными группами. Более сложные классификаторы на основе метода машинного обучения базируются на некотором пространстве признаков, которое так же можно назвать статистической моделью.

Несмотря на успех по преодолению некоторых методов стегоанализа, в настоящее время все известные стеганографические алгоритмы успешно детектируются, в основном, с помощью методов машинного обучения. Согласно классификации по Кашену, все стеганографические алгоритмы вычислительно надежны. Более того, нет никаких доказательств существования совершенно надежных систем над эмпирическими данными (изображения, аудио-, видеоданные и т.д.). Также следует иметь в виду, что алгоритм классификации, построенный на основе метода машинного обучения, является полиномиальным по времени (фактически, работа алгоритма сводится к вычислению скалярного произведения на вектор-константу), но время его обучения (вычисления константного вектора) растет экспоненциально относительно размерности пространства признаков и объема обучающей выборки.

Классификация стеганографических систем, при которой абсолютно все системы разделяют одну категорию, не дает возможности сравнения и выбора алгоритма при решении прикладных задач, сдерживает распространение методов стеганографии.

Относительная надежность

Для большинства прикладных задач, которые могут быть разрешены с помощью стеганографии, требуется не абсолютная надежность и устойчивость к любому бинарному классификатору, а относительная надежность, как устойчивость к определенным классификаторам. Следует также отметить, что кроме атаки бинарным классификатором широко применяется атака искажением, при которой контейнер подвергается специальным фильтрам, нацеленным на разрушение стегоканала. Для фотографий могут применяться алгоритмы обработки изображений, накладываться различные эффекты. Например, резкость, размытие, деформация, шум и т.д.

Принципы Керкгоффса можно и нужно применять не только в стегоанализе, но и непосредственно при построении стеганографических систем. Опираясь на фиксированные границы применимости, которые трактуются как устойчивость к некоторым видам, заранее определенным атакам, мы можем строить относительно надежные стеганографические системы, имеющие прикладной характер и способные решать конкретные задачи. Таким образом, мы уходим от поиска «философского камня» стеганографии, абсолютно надежной стеганографической системы, к использованию методов сокрытия информации для решения прикладных задач.

Схема построения устойчивого к атакам алгоритма заключается в трех этапах: выявлении инварианта, внедрении сообщения в инвариант и восстановлении контейнера из инварианта.

Инвариант – это устойчивое к атакам соотношение внутри контейнера, которое можно использовать для сокрытия информации. В случае если атака является атакой искажением, то определение инварианта становится более очевидным: инвариант – это такое соотношение между элементами контейнера, которое не меняется после атаки искажением. Если речь идет об атаке бинарным классификатором, то инвариант – это та часть контейнера, которая не анализируется бинарным классификатором и не влияет на его работу.

Вторым этапом построения относительно надежной системы является внедрение сообщения в тело инварианта, т.е. получение стего-инварианта. На данном этапе метод внедрения может быть любым, например, LSB, ±1, матричное кодирование и т.д.

На третьем этапе необходимо из стего-инварианта получить стего-контейнер. Т.к. инвариант, по сути, является суръективным, но не инъективным отображением над множеством контейнеров, то для стего-инварианта существует несколько стего-контейнеров. Возникает задача выбора одного варианта из допустимых значений. Для ее решения существует две стратегии: можно выбрать любой контейнер, удовлетворяющий стего-инварианту, или наложить дополнительные условия. В качестве подобного условия можно взять один из фундаментальных принципов построения стеганографической системы. Например, принцип минимизации искажений, при котором вводится специальная метрика, численно характеризующая разницу между стего и контейнером, и ставится задача поиска подходящего стего с минимальной метрикой.

Рассмотрим предложенный способ построения относительно надежной системы на конкретном примере. Пусть выполняются условия «проблемы заключенных» Густава Симмонса: двое обвиняемых в преступлении заключены под стражей и помещены в соседние камеры. Они могут обмениваться сообщениями. Заключенные планируют побег, но для этого им необходимо построить скрытый канал передачи данных, так как все их сообщения тщательно рассматриваются Надзирателем, который в случае малейшего подозрения может сорвать их планы. Заключенные предварительно оговаривают кодовое слово и используют его для внедрения секретной информации о побеге в текст невинных с точки зрения Надзирателя сообщений. Если им удается скоординировать свои действия и остаться незамеченными, то проблема заключенных считается разрешимой. Пусть в качестве «сообщений», т.е. контейнеров для стеганографической системы, выступают цифровые изображения. Надзиратель для проверки может использовать бинарные классификаторы, однако, ввиду неопределенности в выборе заключенными алгоритма сокрытия информации, разумнее использовать специальные фильтры, призванные исказить или даже полностью разрушить стегоканал. Для внесения искажений в контейнер используются матричные фильтры: изображение разбивается на блоки фиксированной размерности, каждый блок независимо участвует в некоторой арифметической операции с матрицей атаки, полученный результат сохраняется обратно в контейнер.

Для данной задачи Надзиратель использует блоки (и, соответственно, матрицу атаки) размерности 3x3. В передаче используются только цифровые фотографии в градации серого, значение каждого пикселя варьируется в пределах [0; 255]. Из изображения извлекается последовательность пикселей слева направо, сверху вниз. Данная последовательность разбивается на отрезки по девять пикселей, из которых формируются блоки. Если длина последовательности не кратна девяти, тогда последние пиксели отбрасываются и не участвуют в искажении. Над блоками определены следующие атаки: искажение матричным умножением, искажение перестановкой столбцов, искажение добавочной матрицей. Если обозначить их литерами «m», «p», «a», то задача сводится к поиску mpa-надежной стеганографической системы.

Искажение матричным умножением – специальная атака над блоком 3x3, при которой исходный блок умножается справа на специальную случайную матрицу атаки. Для того чтобы уменьшить визуальные артефакты от искажения (цель Надзирателя разрушить только стегоканал, а не само изображение), на матрицу атаки вводятся ограничения. Во-первых, элементы матрицы неотрицательны и их сумма по строкам равна 1. Во-вторых, элементы

главной диагонали не превосходят пороговое значение, которое для данной задачи фиксировано и равно 0.7. Частным случаем матрицы атаки является единичная матрица, которая не вносит искажения в исходный блок.

Рис. 1 Исходное изображение (слева) и результат применения m-фильтра (справа)

Искажение матричным умножением проявляет себя в областях градиентной заливки (небо, фасад домов), и менее заметно на темных, хаотичных участках (кусты, крона деревьев) (Рис. 1).

Искажение перестановкой столбцов – разновидность атаки матричного умножения с той лишь разницей, что блок умножается справа на матрицу перестановки. Матрица перестаноовки (или подстановки) — квадратная бинарная матрица, в каждой строке и столбце которой находится лишь один единичный элемент. В результате умножения меняется порядок столбцов в исходном блоке. Подобное искажение проявляется в виде ряби на границах объектов (Рис. 2).

Рис. 2 Исходное изображение (слева) и результат применения p-фильтра (справа)

Наконец, искажение добавочной матрицей является наименее заметным и наиболее действующим для известных стеганографических систем фильтром. Исходный блок суммируется со случайной матрицей, элементами которой являются -1, 0, 1. Учитывая, что большинство известных алгоритмов сокрытия информации используют метод LSB или технику ±1, данный подход полностью перепишет секретное сообщение.

Рис. 3 Исходное изображение (слева) и результат применения a-фильтра (справа)

Наибольшее изменение вносится в близкие к нулю значения, т.е. в темные участки изображения, которые сложно оценить на глаз (Рис. 3).

Надзиратель использует все фильтры для внесения искажений, выбирая случайным образом один фильтр для каждого блока (Рис. 4).

Рис. 4 Исходное изображение (слева) и результат применения mpa-фильтра (справа)

Оценим внесенные искажения. Для этого найдем максимальное, минимальное и среднее отклонение значений пикселей полученного изображения от оригинала. Оценка была проведена на выборке из 50 цифровых изображений BossBase (таблица 1).

Таблица 1 Оценка искажений для предложенных фильтров

Фильтр	Минимум	Максимум	Среднее	σ
Искажение матричным умножением (M)	-77.00	125.00	-1.21	14.37
Искажение перестановкой столбцов (P)	-255.00	255.00	0.00	12.08
Искажение добавочной матрицей (A)	-1.00	1.00	-0.01	0.80
MPA-фильтр	-255.00	255.00	-0.41	10.87

Теперь проверим эффективность предложенного фильтра против известных стеганографических систем из семейства F-алгоритмов. В качестве сообщения взят отрывок из книги Тэрри Пратчетта «The Colour of Magic» (174 символа): «The gaming board was a carefully carved map of the Discworld, overprinted with squares. A number of beautifully modeled playing pieces were now occupying some of the squares.». Реализация алгоритмов взята из

библиотеки pysteg (Hans Georg Schaathun), которая была доработана для пакетной обработки цифровых изображений. Тестирование осуществлялось на выборке из 50 фотографий.

В результате эмпирической проверки ни один из алгоритмов Вестфелда не смог пройти mpa-фильтр, т.е. все алгоритмы F-семейства не являются mpa-надежными (Рис. 5).

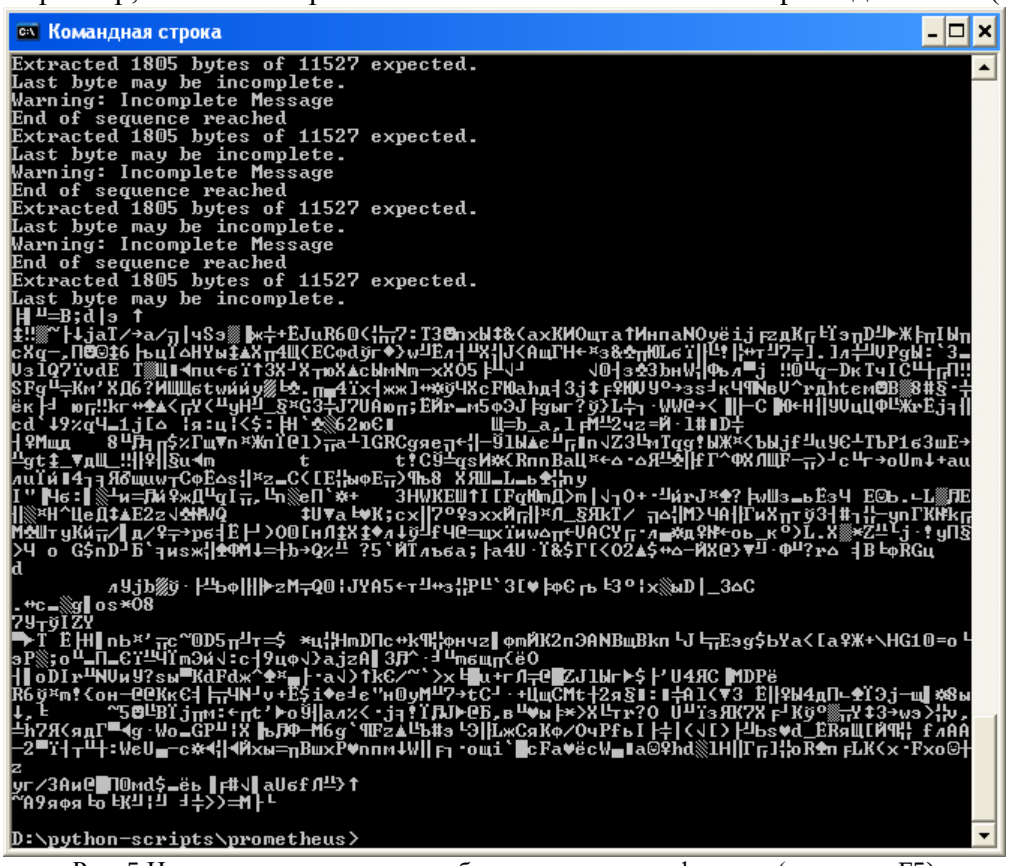


Рис. 5 Извлечение секретного сообщения после mpa-фильтра (алгоритм F5)

Таким образом, для решения поставленной задачи требуется разработать новую, mpa-надежную стеганографическую систему. Воспользуемся предложенной методологией и определим mpa-устойчивое соотношение. Так как искажение генерируется независимо для каждого блока 3x3, то искать инвариант следует в блоках. Обозначим как S сумму элементов строки исходного блока, а S^* - искаженного.

Утверждение 1. Искажение матричным умножением не меняет S. **Доказательство:**

Пусть A – исходный блок, B – матрица атаки, результат атаки: $C = A \times B$.

$$c_{ij} = \sum_{k=1}^{3} a_{ik} b_{kj}$$

$$S_1^* = \sum_{l=1}^{3} c_{1l} = \sum_{l=1}^{3}\sum_{k=1}^{3} a_{1k} b_{kl} = a_{11}\sum_{l=1}^{3} b_{1l} + a_{12}\sum_{l=1}^{3} b_{2l} + a_{13}\sum_{l=1}^{3} b_{3l}$$

$$= a_{11} + a_{12} + a_{13} = S_1$$

так как сумма элементов матрицы атаки по строкам равна 1.

Замечание. Так как значение пикселя целочисленное и находится в пределах [0; 255], возможны ошибки двух типов: округления и переполнения. При округлении в меньшую сторону S^* уменьшается не более чем на 2, т.е. $S + 2 \le S^* \le S$. При переполнении, когда значение становится больше 255, происходит еще большее уменьшение S за счет приведения

к допустимым границам. Следует отметить, что ошибка второго типа встречается только для $S > k$, где k – некоторое пороговое значение.

Утверждение 2. Искажение перестановкой строк не меняет S.

Утверждение 3. Искажение добавочной матрицей изменяет S на ±3.

Утверждение 4. Для mpa-фильтра и некоторого k выполняется соотношение:

$$S - 3 \leq S^* \leq S + 3.$$

Рассмотрим целочисленную функцию $f(a) = 8a + 4$ и обратную к ней $g(x) = x \text{ div } 8$. Выполняется следующее соотношение: $g(f(a) \pm 3) = a$. Это означает, что неполное частное при делении S^* с остатком на 8 является mpa-устойчивым значением, инвариантом для данного искажения. В качестве порогового значения k было выбрано 208.

Введем следующее правило для декодирования одного бита сообщения: 0 соответствует четное значение $g(S)$, 1 – нечетное. При внедрении необходимо из S вычислить неполное частное, изменить его в соответствии с битом сообщения и получить $S^* = f(a)$. Если в результате $S^* > k$, то желаемый бит внедряется в другую сумму S.

В результате, из исходного контейнера получается последовательность сумм трех значений. В последовательность $S_1, S_2, \ldots$ внедряется сообщение по описанной выше схеме и получается новая последовательность $S_1^*, S_2^*, \ldots$; чтобы сообщение было равномерно распределено по всему объему контейнера, последовательность предварительно перемешивается с помощью генератора псевдослучайных чисел.

Теперь необходимо из S^* получить значения элементов строки. Пусть в исходном блоке элементами являются a_1, a_2, a_3; $a_1+a_2+a_3 = S$. Требуется из S^* найти a_1^*, a_2^*, a_3^*. Так как множество целочисленных значений из диапазона [0; 255] удовлетворяет условию $S^* = a_1^* + a_2^* + a_3^*$, нужно ввести некоторое дополнительное правило. Исходя из предложенной методологии, будем минимизировать искажение, а именно сумму квадратов разности соответствующих коэффициентов: $(a_1^*-a_1)^2+(a_2^*-a_2)^2+(a_3^*-a_3)^2$.

Решая несложную систему, получим следующие формулы:

$$a_1^* = 1/3\,(S^* - a_2 - a_3 + 2a_1),$$
$$a_2^* = 1/3\,(S^* - a_1 - a_3 + 2a_2),$$
$$a_3^* = 1/3\,(S^* - a_1 - a_2 + 2a_3).$$

Представленная схема mpa-надежной стеганографической системы реализована на языке программирования *python 2.7* с использованием ряда математических и графических библиотек. Код включает в себя функции для кодирования сообщения в бинарное представление и обратного декодирования (Листинг 1), вычисления новых коэффициентов для строк блоков (Листинг 2), опции разбора переданных скрипту аргументов, алгоритмов извлечения и внедрения (Листинг 3, Листинг 4).

Листинг 1.

```
def coding(string):
                message = []
                for char in string:
                        code = ord(char)
                        bncode = "{0:0>8}".format(bin(code)[2:])
                        message += list(bncode)
                return message

def encoding(message):
                text = ""
        codes = array(message[:8*(len(message)/8)]).reshape(-1,8)
                for bncode in codes:
                        code = int("".join(bncode), 2)
                        char = chr(code)
                        text += char
                return text
```

Листинг 2.

```
def miniquad(nsma, triple, i):
        result=(nsma-triple[(i+1)%3]-triple[(i+2)%3]+2*triple[i%3])/3
```

```
                result = max(result, 0)
                result = min(result, nsma)
                return result

def mintriple(nsma, triple):
                ntriple = []
                for i in range(2):
                                ntriple.append(miniquad(nsma,triple,i))
                ntriple.append(nsma - sum(ntriple))
                return ntriple
```

Листинг 3.

```
random.seed(1331)
ktriples = list(enumerate(signal))
random.shuffle(ktriples)
for (key, triple) in ktriples:
        sma = sum(triple)
        (a, b) = divmod(sma, 8)
        if (a < 26):
        bit = (a % 2)
        message.append(str(bit))
mlen = int("".join(message[:20]),2)
message = message[20:20+mlen]
message = encoding(message)
```

Листинг 4.

```
random.seed(1331)
ktriples = list(enumerate(signal))
random.shuffle(ktriples)
for (key,triple) in ktriples:
        sma = sum(triple)
        (a, b) = divmod(sma, 8)
        if (a < 27):
                if bool(message):
                        bit = int(message[-1])
                        if (a + bit) % 2 != 0:
                            if (a != 0):
                                    if (b < 4): a -= 1
                                    else: a += 1
                            else: a = 1
                        nsma = a * 8 + 4
                        ntriple = mintriple(nsma, triple)
                        signal[key] = ntriple
                        if (a >= 26): continue
                        message.pop()
                else: break
```

Результат внедрения представлен на рисунке 6. Код работы скрипта:

```
>>>python script.py image/25.pgm –o resist
```

Рис. 6 Пустой контейнер (слево) и стего (справа)

Извлечение сообщения осуществляется флагом –x (Рис. 7):

>>>python script.py –x resist/25.pgm

```
D:\python-scripts\prometheus>python script.py image/25.pgm -o resist
Embed message in  image/25.pgm
Save in  resist/25.pgm

D:\python-scripts\prometheus>python script.py -x resist/25.pgm
Extract message from  resist/25.pgm
174 bytes
The gaming board was a carefully carved map of the Discworld, overprinted with s
quares. A number of beautifully modeled playing pieces were now occupying some o
f the squares.

D:\python-scripts\prometheus>_
```

Рис. 7 Извлечение секретного сообщения в командной строке

Проверим надежность к mpa-фильтру данную стеганографическую систему, применим mpa-фильтр к стего (Рис. 8, 9).

```
D:\python-scripts\prometheus>python distortion.py -mpa resist/25.pgm -o dirty
Distortion =>  resist/25.pgm
Save =>  dirty/25.pgm

D:\python-scripts\prometheus>
```

Рис. 8 Применение mpa-фильтра к стего

Рис. 9 Исходное стего изображение (слева) и обработанное mpa-фильтром (справа)

Теперь воспользуемся процедурой извлечения из искаженного контейнера (Рис. 10).

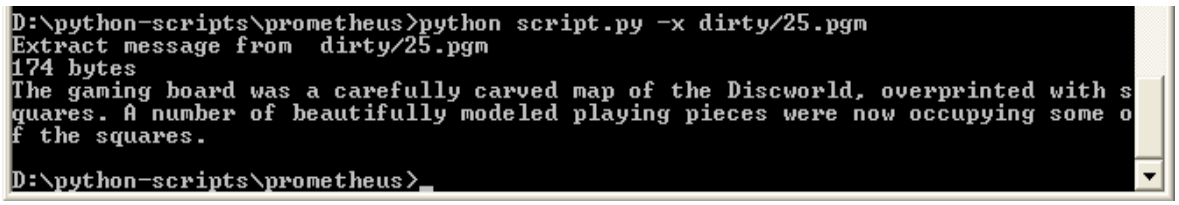


Рис. 10 Извлечение секретного сообщения после применения mpa-фильтра

Проведя аналогичные действия к выборке из 50 цифровых фотографий, было установлено, что предложенная стеганографическая система является полностью mpa-надежной.

mod8 бинарный классификатор

Проанализируем поведение описанного алгоритма. Очевидно, что в процессе внедрения сообщения число элементов с остатком от целочисленного деления на 8 равным 4 становится больше. Проведем гистограммный анализ от операции *g(x)* на выборках из пустых контейнеров и стего (Рис. 11). Для лучшего результата в гистограммном анализе участвуют только элементы из тех троек, в которых сумма значений меньше пороговой величины *k*.

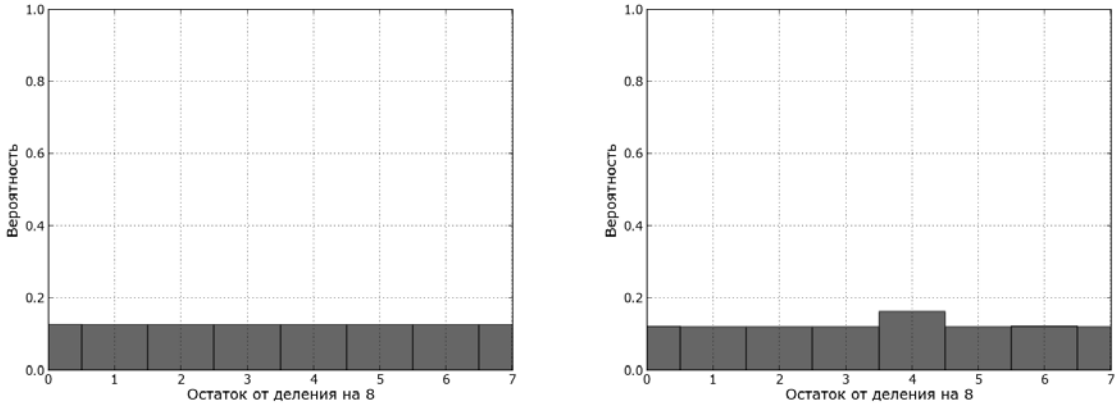


Рис. 11 Гистограммный анализ на выборке из исходных контейнеров (слева) и стего (справа)

Для некоторых фотографий, в которых при внедрении используется практически вся доступная емкость, разница более заметна (Рис. 12).

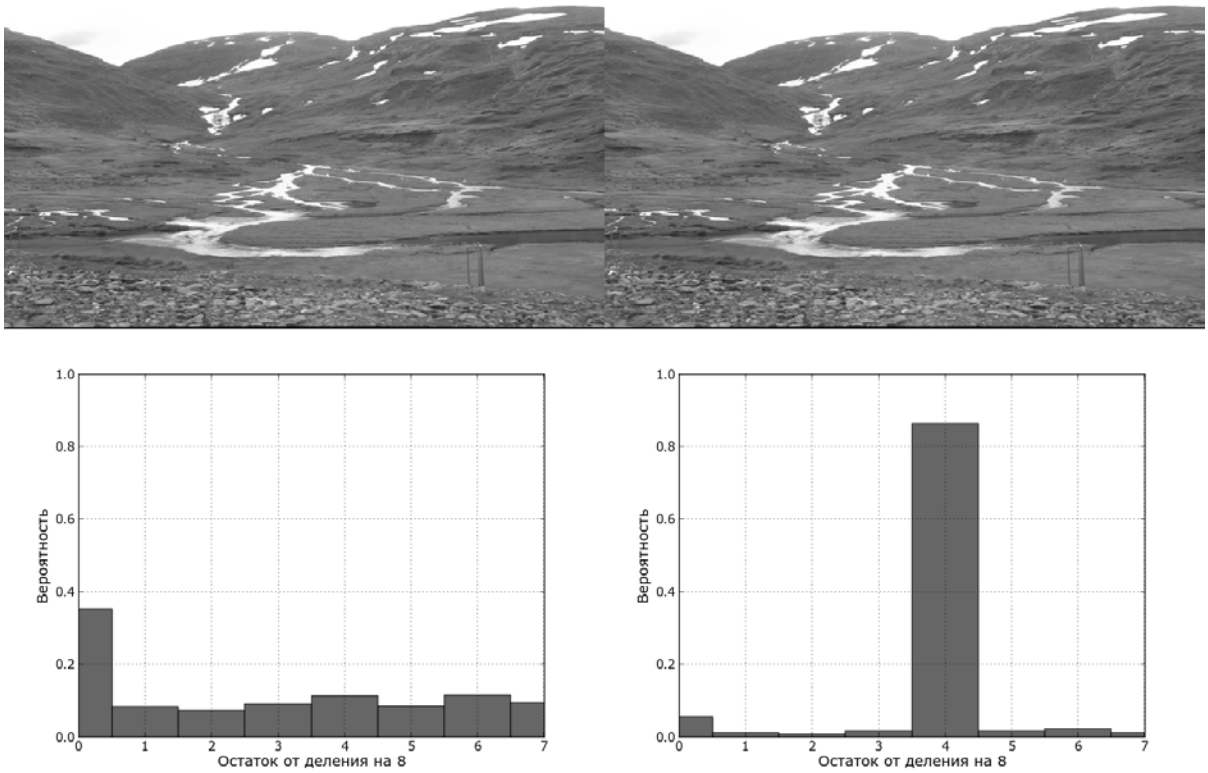


Рис. 12 Исходный контейнер (вверху слева), его гистограмма (внизу слева), стего контейнер (вверху справа), его гистограмма (внизу справа)

Таким образом, можно построить бинарный классификатор на основе анализа остатков от целочисленного деления на 8, который детектирует предложенный алгоритм. Это означает, что разработанная стеганографическая система является mpa-надежной, но не является mod8-надежной, а значит, не является абсолютно надежной по классификации Кашена. Хотелось бы добавить, что алгоритмы F-семейства являются mod8-надежными, но, как было показано, не проходят mpa-фильтр.

Последнее подтверждает основной тезис настоящей работы: стеганографическая система определяется границами применимости и наоборот. Классификация по границам более продуктивна и практична, так как позволяет решать прикладные задачи и находить наиболее приемлемые алгоритмы сокрытия информации.

Заключение

В рамках данной работы рассмотрена классификация надежности стеганографических систем Кашена, определена её роль в развитии стеганографии. Были установлены недостатки категориального деления.

В качестве альтернативы предложена классификация алгоритмов по границам их применимости, а также разработана методология построения подобных систем. На примере показан процесс поиска устойчивого к атакам соотношения в контейнере, внедрения сообщения и восстановления из соотношения стего.

Дальнейшее развитие связано с построением аналитического алгоритма для выявления устойчивых соотношений внутри контейнера, который позволит создать фабрику стеганографических систем для решения прикладных задач в области сокрытия информации.

СРАВНЕНИЕ МЕТОДОВ ФОРМИРОВАНИЯ ИНДЕКСОВ ДОСТОВЕРНОСТИ В СИСТЕМЕ С OFDM

А.А.Гладких, И.С.Линьков

Ульяновский государственный университет

В современных системах связи радиоканал является единственным средством передачи информации для высокоскоростных мобильных абонентов. Из-за случайных помех и многолучевости в таких каналах должны использоваться мощные методы защиты информации от ошибок [1,2]. По асимптотическим оценкам мягкие методы декодирования обеспечивают энергетический выигрыш до 3-х дБ. Поэтому такие методы активно внедряются в современные системы передачи данных. Наряду с этим, развиваются методы OFDM (ортогональное частотное уплотнение каналов), которые способны существенно повысить общую скорость передачи информации при относительно низкой скорости в каждом субканале. В каждом таком канале системы используются недвоичные методы модуляции с целью повышения эффективности частотного ресурса OFDM. Как правило, в таких системах каждый субканал использует КАМ-сигналы. Однако метод получения индексов достоверности для таких сигналов до сих пор не разработан. Целью работы является исследование метода получения индексов достоверности для сложных сигналов типа КАМ-4 и выработка предложений по использованию мягкого адаптивного декодера произведения кодов.

При исправлении индексов достоверности или стираний места ошибочных позиций в кодовой комбинации известны, поэтому мощность кода не тратится на поиск пораженных символов. Это обеспечивает повышение корректирующей способности кода примерно в два раза, при этом в кодовой комбинации целесообразно формировать предельно допустимое число стираний, зависящее от метрики Хемминга d.

Пусть при приеме символов в зависимости от результатов обработки информации из канала связи формируется в пределах кодовой комбинации раз за разом различное число стираний. Обозначим для таких условий приема через P_{os} вероятность ошибочного декодирования всей кодовой комбинации. Тогда

$$P_{os} = \sum_{i=0}^{s} P_i \cdot P_i' + \sum_{i=S+1}^{n} P_i ,$$

где P_i – вероятность стертых символов, а P_i' - вероятность появления ошибок в этой же кодовой комбинации при наличии ровно i стираний. Обычно в правильно спроектированной системе связи появление стираний приводит к уменьшению вероятности ошибок, следовательно,

$$P_i' > P_{i+1}' \quad (i = 0,1,..., d-1 = s).$$

Установим, что P_{0const} – вероятность ошибочного декодирования кодовой комбинации кода, когда каждый раз, используя принцип оценки надежности символов, в принятой комбинации формируется ровно s стираний. Покажем, что при реализации декодером условия $P_i' > P_{i+1}'$, для кодовых комбинаций длины n выполняется соотношение $P_{os} > P_{0const}$. Составим очевидное неравенство

$$P_s' \sum_{i=0}^{s} P_i + \sum_{i=s+1}^{n} P_i > P_s' \sum_{i=0}^{s} P_i + P_s' \sum_{i=s+1}^{n} P_i .$$

Поскольку $\sum_{i=0}^{s} P_i + \sum_{i=s+1}^{n} P_i = 1$, то получаем

$$P_s' \sum_{i=0}^{s} P_i + \sum_{i=s+1}^{n} P_i' > P_s' . \qquad (1)$$

Из условия $P_i' > P_{i+1}'$ вытекает, что $P_s' \sum_{i=s}^{s} P_i < \sum_{i=0}^{s} P_i' P_i$ и, усиливая неравенство (1), получаем $P_s' \sum_{i=s}^{s} P_i' P_i + \sum_{i=s+1}^{n} P_i > P_s'$, следовательно, $P_{os} > P_{0const}$.

Из последнего неравенства вытекает условие декодирования кодовой комбинации по стираниям: среди всех принятых символов выбираются d - 1 с минимальными оценками надежности, эти символы стираются, и по ним восстанавливается кодовый вектор. В случае жестких решений энергетический выигрыш оценивается выражением $D_h = 10lg(k/n(t+1))$дБ, а при реализации мягкого декодирования как $D_s = 10lg(k/n)d_{min}$дБ [2, 3], здесь: k – информационные символы в кодовом векторе длины n, t – число, исправляемых кодом ошибок. Отсюда следует, что асимптотический выигрыш в случае мягкого декодирования на 3 дБ выше, чем при реализации жесткой схемы принятия решения. Подобная оценка не является предельной, поскольку при декодировании не полностью используется введенная в код избыточность. Предположив, что код способен исправить ровно $n-k$ стираний, получим новую оценку в виде соотношения $D_{km} = 10lg(k(1-k/n+1/n))$дБ, которая показывает, что за пределами мягкого декодирования для двоичных кодов возможно получение дополнительного энергетического выигрыша. Следует заметить, что двоичные блоковые коды не являются максимально декодируемыми, поэтому для приближения к приведенной границе следует искать методы декодирования, не связанные с традиционными алгебраическими подходами к данной процедуре.

Использование индексов достоверности символов (ИДС) для сложных видов модуляций является нетривиальной задачей и требует специфического подхода. Классически эта проблема может быть решена с помощью метода концентрических окружностей, центры которых совпадают с номинальными точками регистрации сигнала созвездия. Пример такого разбиения показан на рисунке 1. Внутренней окружности присуждается наивысший индекс $\lambda_{\max}$, который убывает по мере удаления от центра. Внешняя окружность имеет минимальный индекс удаления $\lambda_{\min}$. Обычно используют семь градаций надежности индексов достоверности. Достоинством этого метода является простота реализации, поскольку на комплексной плоскости окружность описывается простым аналитическим соотношением, основным параметром которого является радиус окружности. Недостатком метода является размножение низких индексов достоверности, что отрицательно сказывается на второй ступени обработки данных непосредственно в мягком декодере.

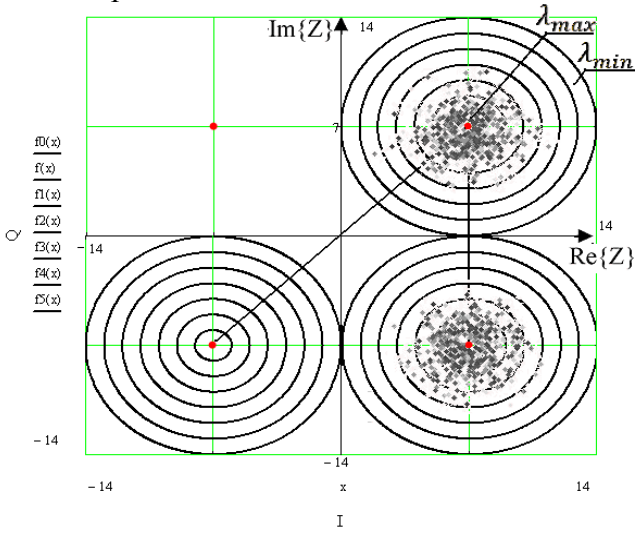


Рис. 1. Пример разбиения сигналов КАМ-4.

Новым решение данной задачи является применение распределения индексов достоверности в соответствии с гиперболическими границами их определения. Такой пример описанного

распределения показан на рисунке 2. Применено разделение области приема на 8 областей с ИДС меняющимися от 0 до 7. Гиперболы построены в соответствии с равномерным разделение участка между соседними узлами.

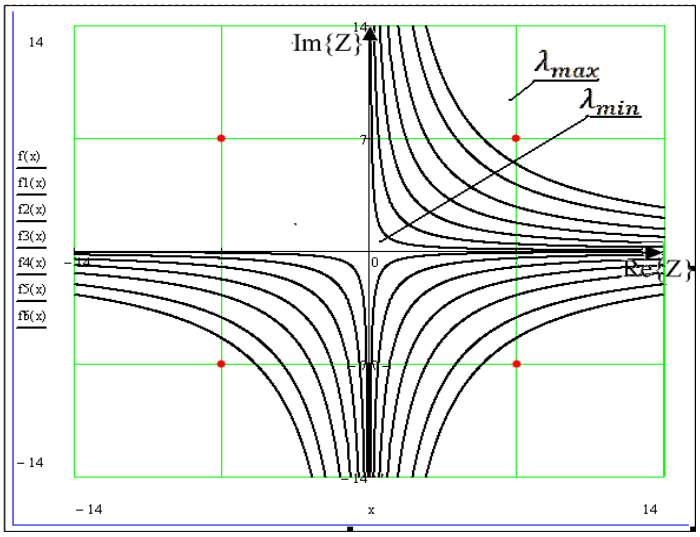


Рис. 2. Пример назначения границ для оценки индексов надежности по гиперболическому принципу

Очевидно, назначение подобных границ приводит к увеличению доли правильных оценок, совпадающих с $\lambda_{\max}$. На рисунке 2 показано распределение граничных гипербол, которые отвечают линейной функции, проходящей через центр созвездия к его диагоналям. Однако подобный подход не учитывает критического направления между номинальными точками созвездия, параллельным осям координат. На рисунке 3 показано распределение с использование гипербол с равномерным распределение зон для каждой оценки надежности, меньшей максимальной и больше минимальной.

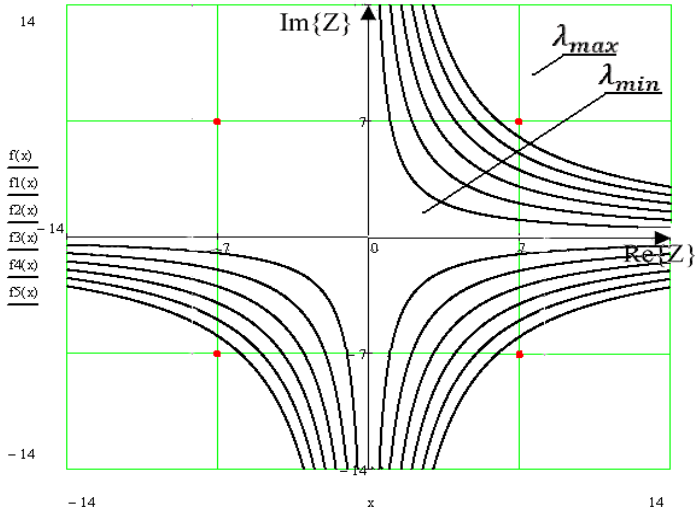


Рис. 3 Принцип распределения границ с равномерным распределение интервалов

Сравнение характеристик представленных методов формирования индексов достоверности показано на рисунке 4.

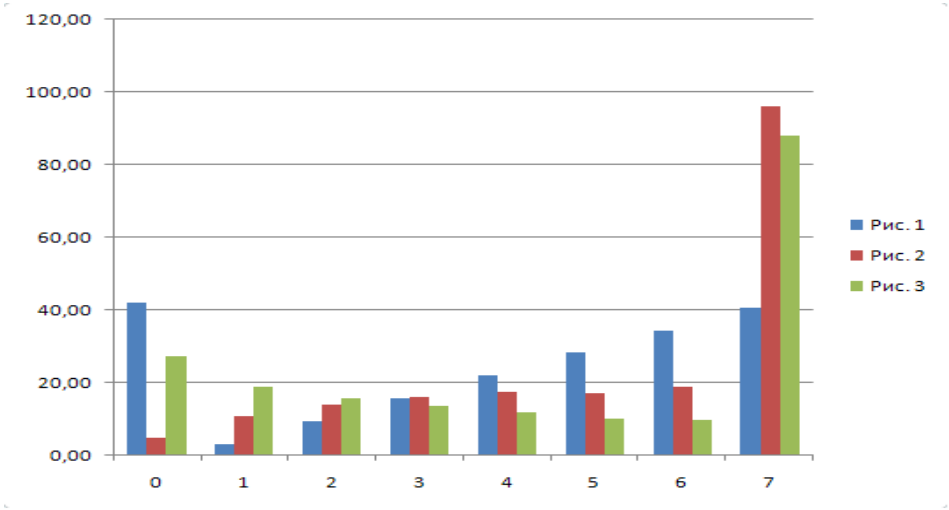


Рис.4 Сравнение различных видов распределения граничных оценок

При сравнительном анализе (рис 4.) распределения граничных оценок можно выявить несколько закономерностей.

1. Наибольшую площадь, на которой принятый сигнал получает оценку λ_{max} обеспечивает второй вид распределения (рис. 2). Преобладание наилучших оценок положительно сказывается на второй ступени обработки данных.

2. При третьем виде распределения площадь оценок последовательно уменьшается при их увеличении от 0 до 6. Это говорит о непригодности данного метода в наших условиях.

Естественно, что метод может быть применен только для КАМ-4, поскольку решающее правило существенно усложняется при использовании КАМ-16 и выше. Однако следует учитывать, что КАМ-16, КАМ-32, КАМ-64 используются как самостоятельные виды модуляции в относительно стабильных каналах с низкой вероятностью ошибки [4,5]. Применение КАМ-4 в OFDM позволяет компенсировать влияние помех на отдельные ортогональные каналы, а мягкое декодирование повысить энергетическую эффективность системы связи, что очень важно для повышения качества мобильной связи.

Современные методы обработки цифровых сигналов позволяют достаточно эффективно оценивать гиперболические границы в системе КАМ-сигналов и решить задачу мягкого декодирования дискретных сообщений.

Литература

1. Питерсон У., Уэлдон Э. Коды, исправляющие ошибки/ У. Питерсон, Э. Уэлдон; пер.сангл.; под ред. Р. Л. Добрушина и С. Н Самойленко. – М.: Мир, 1976. – 594 с.

2. Зяблов В. В. Высокоскоростная передача сообщений в реальных каналах/ В. В. Зяблов, Д.Л. Коробков, С.Л.Портной. – М.: Радио и связь, 1991. – 288 с.

3. Ибрагимов И. А., Хасьминский Р.З. Асимптотическая теория оценивания/ И. А. Ибрагимов, Р. З. Хасьминский. – М.: Наука, 1979. – 528 с.

4. Гладких А.А. Основы теории мягкого декодирования избыточных кодов в стирающем канале связи / А.А. Гладких – Ульяновск: УлГТУ, 2010. - 253 с.

5. Скляр Б. Цифровая связь. Теоретические основы и их практическое применение. – М.: Издательский дом «Вильямс», 2003.

АНАЛИЗ И СИНТЕЗ МЕТОДОВ РАСШИРЕНИЯ КОРРЕКТИРУЮЩИХ СПОСОБНОСТЕЙ БЛОКОВЫХ КОДОВ

А.А.Гладких, В.П.Смолеха, В.А.Лукьянов

Ульяновский государственный университет

Рассматриваются регулярные методы достижения асимптотических границ в процедуре декодирования блоковых кодов, основанные на быстром составлении списков наиболее вероятных кодовых комбинаций с использованием лексикографического метода, именуемого в работе методом кластерного анализа. Показана принципиальная возможность перехода от защитных зон в виде сфер к защитным зонам в виде прямоугольных параллепипидов, позволившего представить множество кодовых комбинаций в виде созвездий кластеров. Реализация метода основана на мягких решениях приемника.

В теории помехоустойчивого кодирования метрика Хэмминга играет основополагающую роль и ее суть общеизвестна. В тоже время для блоковых кодов, в случае перехода к укороченным кодам, хрестоматийным является факт обнаружения при использовании стандартной расстановки кода ошибок, кратность которых превосходит указанное расстояние [1]. Возникает вопрос: является ли подобный факт исключительным, а если нет, то, какие возможности открываются при попытке декодировать двоичные коды за пределами их конструктивных возможностей.

Целью является разработка и моделирование алгоритмов мягкого списочного декодирования блоковых кодов с применением целочисленных индексов достоверности символов (ИДС), обеспечивающих полное использование введенной в код избыточности.

Известно, что в канале с гауссовским шумом при $E/N_0 \to \infty$, где $E-$ энергия сигнала, приходящаяся на бит, а N_0- спектральная плотность гауссовского шума, в случае жестких решений энергетический выигрыш оценивается выражением $D_h = 10lg\left(k/n\left(t+1\right)\right)$дБ, а при реализации мягкого декодирования как $D_s = 10lg\left(k/n\right)d_{min}$дБ. В приведенных формулах: $k-$ число информационных символов в кодовом векторе длины n, $t-$ число, исправляемых кодом ошибок, а $d_{min}-$ метрика Хэмминга. Отсюда следует, что асимптотический выигрыш в случае мягкого декодирования на 3 дБ выше, чем при реализации жесткой схемы принятия решения [2,3]. Из-за метрики Хэмминга подобная оценка не является предельной, поскольку при декодировании не полностью используется введенная в код избыточность.

Предположив, что код способен исправить ровно $n-k$ стираний, получим новую оценку в виде соотношения $D_{km} = 10lg\left(k\left(1-k/n+1/n\right)\right)$дБ, которая показывает, что за пределами мягкого декодирования для двоичных кодов возможно получение дополнительного энергетического выигрыша. Следует заметить, что двоичные блоковые коды не являются максимально декодируемыми, поэтому для приближения к приведенной границе следует искать методы декодирования, не связанные с традиционными алгебраическими подходами к данной процедуре.

Одним из таких методов является метод декодирования с использованием упорядоченной статистики по убыванию ИДС символов в кодовых комбинациях блоковых кодов длины n и применения эквивалентных кодов. Декодирование начинается с упорядочивания компонент принятой последовательности по убыванию надежностей. Обозначим через $Y = \left(y_1, y_2, ..., y_n\right)$ упорядоченную последовательность ИДС, в которой $\left|y_1\right| \geq \left|y_2\right| \geq ... \geq \left|y_n\right|$. Назовем это переупорядочивание подстановкой λ_1 такой, что $y = \lambda_1\left(r\right)$, где $r = \left(r_1, r_2, ..., r_n\right)$ принятая последовательность символов. В ходе сортировки ИДС, отвечающих процедуре λ_1, создается перестановочная матрица R_{λ_1}. Следующий шаг алгоритма состоит в перестановке столбцов

порождающей матрицы $G=\left(I_{k\times k}:H^{T}_{(n-k)\times k}\right)$ в порядке, соответствующем последовательности Y. Выполняя $G\times R_{\lambda_1}$, получим $G'=\lambda_1[G(Y)]=\left(g_1'g_2'...g_n'\right)$, где $g_i'-$ i-й столбец матрицы G'. Естественно, матрица G' на данном шаге алгоритма не является систематической.

Продолжение алгоритма состоит в построении наиболее надежного базиса возможного эквивалентного кода. Начиная с первого столбца матрицы G', находятся первые k линейно независимых столбцов, которым в соответствии с Y соответствуют наибольшие ИДС. Остальные $(n-k)$ столбцов также упорядочиваются в порядке убывания их надежности. В результате получают отображение порождающей матрицы λ_2 такое, что $G''=\lambda_2[G']=\lambda_2[\lambda_1[G(Y)]]$. Применяя отображение λ_2 к последовательности Y, формируют новую переупорядоченную последовательность Z, где $Z=\lambda_2(Y)=(z_1,z_2,...z_k,z_{k+1},...,z_n)$. В этой последовательности $|z_1|\geq|z_2|\geq...\geq|z_k|\geq|z_{k+1}|\geq...\geq|z_n|$. Для проверки линейной независимости строк в матрице G' (только для двоичных кодов) декодер выделяет первые k столбцов и, формируя матрицу $S_{k\times k}$, вычисляя ее детерминант. При $det(S_{k\times k})\neq 0$, открывается возможность образования из матрицы G'' путем линейных преобразований ее строк и столбцов новой матрицы эквивалентного кода $G''_{сист}$ в систематической форме. При $det(S_{k\times k})=0$ матрице $S_{k\times k}$ наблюдается свойство линейной зависимости строк, что не позволяет сразу получить $G''_{сист}$. В случае линейной зависимости строк декодер переходит к итеративной процедуре преобразования $S_{k\times k}$ за счет смены мест столбцов с номерами k и $k+1$ в G' (первый шаг итерации). При отрицательном исходе этого шага итерации, осуществляется смена мест столбцов с номерами $k+1$ и $k+2$ (второй шаг итерации). Выполнение последующих шагов считается нецелесообразным из-за опасности манипуляции с ошибочными символами (столбцами). В этом случае комбинация отмечается как стирание для последующего его восстановления на уровне внешних декодеров в схеме последовательного турбокодирования.

Получив удовлетворительный результат по вычислению $det(S_{k\times k})$, декодер выполняет регулярную процедуру по вычислению матрицы $G''_{сист}$ через определение обратной матрицы, которая точно указывает на порядок сложения строк матрицы G'' для получения новой порождающей матрицы в систематической форме $G''_{сист}$.

В результате выполнения алгоритма декодеру становятся известными принятый вектор $V_{пр}$ с ошибками и упорядоченными по убыванию ИДС, матрица перестановок R_{λ_1} и результат ее умножения на вектор $V_{пр}$: $V'_{пр}=V_{пр}\times R_{\lambda_1}$, преобразованная в соответствии с Y порождающая матрица $G''_{сист}$ и новый вектор $V''_{пр}$ как результат умножения информационных разрядов с наиболее надежными ИДС из $V'_{пр}$ на $G''_{сист}$. Очевидно, что вычитание вектора $V''_{пр}$ из вектора $V'_{пр}$ определяет вектор ошибок V'_{er}. Этот вектор необходимо умножить на $R^{T}_{\lambda_1}$, чтобы получить истинный вектор ошибок V_{er}, действовавший в канале связи в ходе передачи информации. Сложение этого вектора с вектором $V_{пр}$ обеспечивает исправление ошибок. Анализ порождающих матриц нескольких блоковых кодов показал, что в общей массе подстановок в ходе упорядочивания ИДС отрицательный исход составляет от 25% до 30% от общего числа подстановок. Это позволяет повысить эффективность схем каскадного кодирования, а, приведенная выше оценка, получает вид $D_{km}=\mu\cdot 10lg\left(k\left(1-k/n+1/n\right)\right)$ дБ, где $0<\mu\leq 1$.

Применение метода к недвоичным кодам РС обеспечивает простоту исправления ошибок за счет получения безошибочного вектора путем перемножения группы наиболее надежных символов на матрицу эквивалентного кода и исключения процедуры поиска

синдромных уравнений и решения их системы в матричной форме. Восстановление истинного вектора осуществляется путем умножения этого вектора на транспонированную перестановочную матицу.

Другим способом достижения указанной границы декодирования двоичных кодов является метод разбиения пространства разрешенных кодовых комбинаций на кластеры и представления в каждом кластере комбинаций, относящихся к конкретному номеру кластера, в системе декартовых координат в двумерном евклидовом пространстве.

Суть рассматриваемого в работе способа обработки кодовых векторов заключается в том, что все множество разрешенных комбинаций блокового кода разбивается на подмножества (кластеры). Кластеры нумеруются по заранее оговоренному принципу путем выделения части разрядов из числа кодового вектора. При этом позиции разрядов для всего разрешенного множества комбинаций должны быть одинаковыми. В этом случае процесс выделения номера кластера соответствует принципам лексикографического метода. Оставшиеся символы кодовой комбинации разбиваются на две группы, каждая из которых образуют координаты по двум осям координатной плоскости. Такое разбиение приводит к размещению разрешенных комбинаций кода в трехмерном пространстве, при этом номера кластеров образуют плоскости, для которых известны координаты кодовых векторов, принадлежащие данному кластеру. Применение на практике данного метода основано на доказательстве ряда утверждений, использующих положения алгебраической теории групп, колец и полей [4, 5].

Пусть общее число комбинаций группового кода равно 2^k. Из любого циклического кода путем регулярных преобразований или линейных преобразований над строками порождающей матрицы G можно образовать систематический код с матрицей $G_s = [I_k \vdots P]$, порождающей тот же код. В единичной матрице I_k всегда можно выделить единичную матрицу меньшей размерности f, где $1 \le f \le k$.

Путем линейных преобразований над строками выделенной матрицы I_f, можно получить двоичное поле Галуа степени расширения f, при этом комбинации поля $GF(2^f)$ будут определять признак кластера или его номер. Поле $GF(2^f)$ содержится в поле $GF(2^k)$ ровно 2^{k-f} раз, следовательно, число кодовых комбинаций в одном кластере будет определяться этим же соотношением. Следовательно, если число двоичных символов, определяющих признак кластера равно f и $0 \le f \le k$, то число комбинаций такого кода, входящих в кластер одного признака, определяется соотношением 2^{k-f}. Очевидно, что при значении $f = 0$ все кодовые векторы входят в один кластер. При этом процедура обработки кодовой комбинации сводится к системе жесткого или мягкого декодирования по общеизвестным правилам.

Полное множество всех возможных кластеров блокового систематического кода может быть образовано путем линейной комбинации строк порождающей матрицы. Действительно, определив разряды кодовых комбинаций, указывающих на номер кластера (параметр f), выделим в порождающей матрице G первые f строк. В единичной матрице $I \in G$ под последней строкой из выбранных f строк образуется множество столбцов, содержащих только нули. Строки матрицы G с первыми f нулями являются кандидатами для формирования нулевого кластера. Поскольку комбинации всех строк матрицы G определяют кластер с номером $2^f - 1$, то все другие номера могут быть сформированы путем линейной комбинации соответствующих строк.

Для любого двоичного циклического кода с установленной базовой структурой бит, определяющей номер кластера, возможна однозначная идентификация номера кластера по любой другой группе двоичных символов адекватной базовой структуре. Рассмотренное свойство позволяет установить номер кластера зафиксированного приемником кодового вектора в случае его искажения при передаче по каналу связи. Для этого может быть использована другая группа разрядов той же кодовой комбинации, которые оказались принятыми с высокими значениями

ИДС. Подобный подход не исключает применения и итеративных преобразований разрядов кодового вектора в комплексе с известными проверочными соотношениями.

Каждый разряд любой координаты X или Y имеет вес кратный значению 2^i, где $i \in N \bigcup \{0\}$, $i \le (k-f)/2$. Принципиально это означает, что при восстановлении кодового вектора по признаку кластера значения координат X или Y мало изменяются при замене в младших разрядах единиц на нули и наоборот (принцип стеганографии).

Пример разбиения множества комбинаций кода Хэмминга (7,4,3) на кластеры представлен в таблице 1, а на рисунке 1 показана топология комбинаций в каждом кластере.

Таблица 1. Список кодовых комбинаций кода Хэмминга (7,4,3)

Номер комби-нации	Разряды					Признак кластера		X_{10}	Y_{10}	Номер комби-нации	Разряды					Признак кластера		X_{10}	Y_{10}
1	2		3			4		5	6	7	8			9		10		11	12
0	**0**	**0**	**0**	**0**	**0**	***0***	***0***	0	0	8	**0**	**0**	**0**	**1**	**0**	***1***	***1***	0	2
1	**0**	**0**	**1**	**0**	**1**	***1***	***0***	1	1	9	**0**	**0**	**1**	**1**	**1**	***0***	***1***	1	3
2	**0**	**1**	**0**	**1**	**1**	***0***	***0***	2	3	10	**0**	**1**	**0**	**0**	**1**	***1***	***1***	2	1
3	**0**	**1**	**1**	**1**	**0**	***1***	***0***	3	2	11	**0**	**1**	**1**	**0**	**0**	***0***	***1***	3	0
4	**1**	**0**	**0**	**1**	**1**	***1***	***0***	4	3	12	**1**	**0**	**0**	**0**	**1**	***0***	***1***	4	1
5	**1**	**0**	**1**	**1**	**0**	***0***	***0***	5	2	13	**1**	**0**	**1**	**0**	**0**	***1***	***1***	5	0
6	**1**	**1**	**0**	**0**	**0**	***1***	***0***	6	0	14	**1**	**1**	**0**	**1**	**0**	***0***	***1***	6	2
7	**1**	**1**	**1**	**0**	**1**	***0***	***0***	7	1	15	**1**	**1**	**1**	**1**	**1**	***1***	***1***	7	3

Очевидными особенностями такого разбиения комбинаций кода (7,4,3) являются:

- симметрия второго рода между четными и нечетными кластерами;
- размещение всех кодовых комбинаций на двумерной плоскости между нулевой и чисто единичной комбинацией;
- соотношение между симметричными вершинами кластеров для координаты X по модулю 2^3-1 и для координаты Y по модулю 2^2-1 (поскольку для X выделялось три разряда, а для Y выделялось два разряда). Принципиально под значения координат при других параметрах кода и иной нумерации кластеров могут выделяться одинаковое число разрядов. В этом случае значения кодовых комбинаций с симметричными координатами будут определяться по модулю 2^f.

Важно отметить, что при кластерном подходе появляется новая метрика в форме прямоугольной защитной зоны, определяемой прямыми линиями, проходящими через пары точек с координатами:

- для вертикальной границы $[(2^{\beta-1}-1);\ 0]$ и $[(2^{\beta-1}-1);\ 2^{\beta}]$;
- для горизонтальной границы $[0;\ (2^{\beta-1}-1)]$ и $[2^{\beta};(2^{\beta-1}-1)]$.

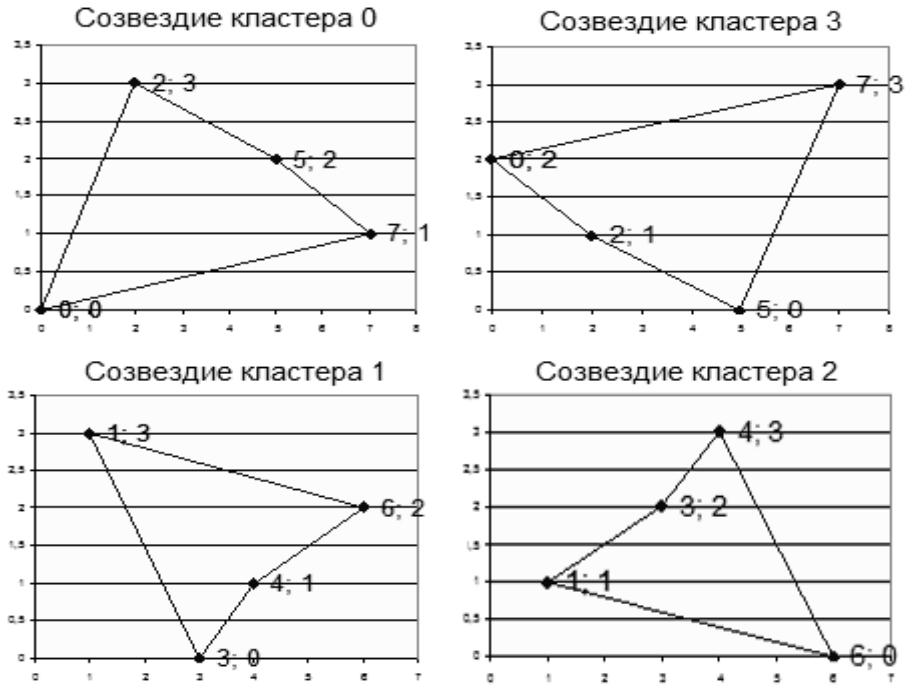


Рис. 1. Созвездия комбинаций кода (7,4,3), распределенных по кластерам

Применение кластерного подхода предполагает декодирование комбинаций по списку. В современных условиях это не является сложной задачей, поскольку прогресс в создании систем памяти значителен, а методы алгебраического декодирования кодов сохранили классическую форму. При алгебраическом декодировании декодер связан обязательной процедурой составления системы линейных уравнений и ее решения. Сложность этой процедуры зависит от конфигурации ошибок и не может быть решена итеративными методами. Предлагаемый подход позволяет повысить корректирующие возможности кода. Код (7,4,3), способен гарантированно исправить одну ошибку или два стирания. При кластерном декодировании для данного кода возможно исправление трех стираний. Все зависит от правильности определения кластера и старших разрядов координат.

Литература

1. Питерсон У., Уэлдон Э. Коды, исправляющие ошибки. Ред. Р. Л. Добрушина и С. Н Самойленко. – М.: Мир, 1976. – 594 с.

2. Зяблов В. В. Высокоскоростная передача сообщений в реальных каналах/ В. В. Зяблов, Д.Л. Коробков, С.Л.Портной. – М.: Радио и связь, 1991. – 288 с.

3. Ибрагимов И. А., Хасьминский Р.З. Асимптотическая теория оценивания – М.: Наука, 1979. – 528 с.

4. Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение.– М.: Техносфера, 2005. – 320 с.

5. Гладких А.А. Основы теории мягкого декодирования избыточных кодов в стирающем канале связи – Ульяновск: УлГТУ, 2010 – 379 с.

ЯЗЫК МОДЕЛИРОВАНИЯ RISK KIT В РЕШЕНИИ ЭКОНОМИЧЕСКИХ ЗАДАЧ

С.Н.Жиляков, К.В.Кумунжиев

Ульяновский государственный университет

Метод Монте-Карло играет возрастающую роль в области финансов, социальных наук и управлении рисками. Моделирование методом Монте-Карло является общим подходом для оценки характера распределений числовых характеристик в сложных условиях. Среди таких характеристик могут быть доходы, акционерная стоимость, потери портфеля активов и вообще любая характеристика, подверженная риску.

Несмотря на то, что метод появился в 40-х годах прошлого столетия, его использование до недавно времени сдерживалось большими затратами как на подготовку модели задачи, так и затратами компьютерных ресурсов для решения этих задач. Кроме того, использование метода требовало наличия у пользователя знаний в области разработки алгоритмов получения случайных чисел, создания самих моделей, реализации алгоритмов вычислений и оценки результатов моделирования.

Прогресс в области создания информационных технологий существенно сократил затраты. Появились пакеты на уровне языков моделирования, которые существенно снижают затраты как на подготовку и решение задачи, так и на подготовку самого пользователя. Одним из таких пакетов является Risk Kit.

Рассмотрим поэтапно процесс решения задачи методом Монте-Карло с использование пакета Risk Kit на примере задачи моделирования прибыли-убытков.

Построение модели

Пакет Risk Kit представляет надстройку для табличного процессора Microsoft Excel, позволяя, тем самым, использовать электронные таблицы в качестве платформы для создания моделей.

Рис. 13 - Панель инструментов Risk Kit

Рассмотрим базовую модель прибыли-убытков в качестве примера того как преобразовать детерминистическую модель в Монте-Карло. Для расчета прибыли и убытков возьмем значение оборота компании (B4) и вычтем из него все издержки и списания. Во-первых, вычтем значение материальных издержек (B5), персональных издержек (B6), а также непредвиденные расходы (B7). Получим промежуточное значение EBIT (доходы до выплаты процентов и налогов) в ячейке C8. Во-вторых, вычтем из получившегося значения расходы по выплатам процентов (B9) и внеплановые доходы (B10).

	A	B	C
1	***Расчет прибыли и убытков***		
2	в у.е.		
3		План	Итого (у.е)
4	Оборот	1 000,00	1 000,00
5	Материальные издержки	50%	500,00
6	Персональные издержки	325	325,00
7	Непредвиденные расходы	75,00	75,00
8	EBIT (Доходы до выплаты процентов и налогов)		100,00
9	Выплата процентов	-50,00	-50,00
10	Внеплановые доходы	0,00	0,00
11	EBT (Доходы до выплаты налогов)		50,00

Рис. 2- Детерминистическая модель прибыли-убытков

Интересными для нас значениями здесь являются доходы до выплаты процентов и налогов (EBIT) и доходы до выплаты налогов (EBT). На первом этапе построения модели Монте-Карло

определим источники неопределенности. Будем подразумевать, что оборот компании, персональные и материальные издержки, а также непредвиденные расходы подвержены риску. Целью является определить влияние этих параметров на значения EBIT и EBT.

Для описания случайного характера значения оборота, воспользуется треугольным распределением и установим значения A, B и C в 900, 1000 и 1050 соответственно. Для этого выделим ячейку B4, после чего воспользуемся кнопкой ‘Functions’ (Функции) на панели инструментов пакета Risk Kit и выберем ‘Triangular’ (Треугольное распределение) в списке доступных распределений. Установим значения A, B и C, оставив остальные поля пустыми.

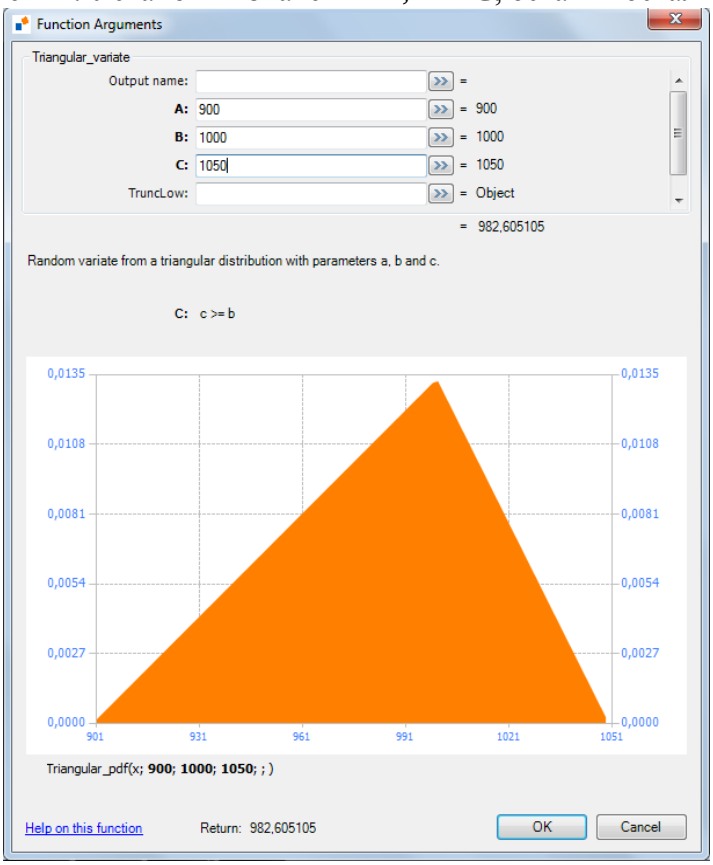


Рис. 3- Диалоговое окно определения параметров распределения

При нажатии на кнопку ‘ОК’ происходит вставка функции треугольного распределения в ранее выделенную ячейку в Excel. Заметим, что ячейка B4 теперь содержит случайное число. На каждом этапе итерации все случайные числа будут вновь сгенерированы.

Аналогичным образом заменим детерминированное значение материальных издержек функцией нормального распределения. В качестве математического ожидания и среднеквадратичного отклонения будем использовать значения 50 и 3 соответственно. Заметим, что материальные издержки в абсолютном выражении являются моделируемыми издержками в процентном соотношении к моделируемому обороту компании.

	A	B	C
1	***Расчет прибыли и убытков***		
2	в у.е.		
3		План	Итого (у.е)
4	Оборот	961,82	961,82
5	Материальные издержки	51%	=+C4*B5
6	Персональные издержки	325	325,00
7	Непредвиденные расходы	75,00	75,00
8	EBIT (Доходы до выплаты процентов и налогов)		68,09
9	Выплата процентов	-50,00	-50,00
10	Внеплановые доходы	0,00	0,00
11	EBT (Доходы до выплаты налогов)		18,09

Рис. 4 - Связь материальных издержек с оборотом компании

Персональные издержки моделируем по аналогии с материальными издержками с параметрами математического ожидания и среднеквадратичного отклонения 325 и 20 соответственно. Кроме того, установим параметр TruncLow в 300, означающий, что вновь сгенерированные случайные числа не могут принимать значения меньше 300.

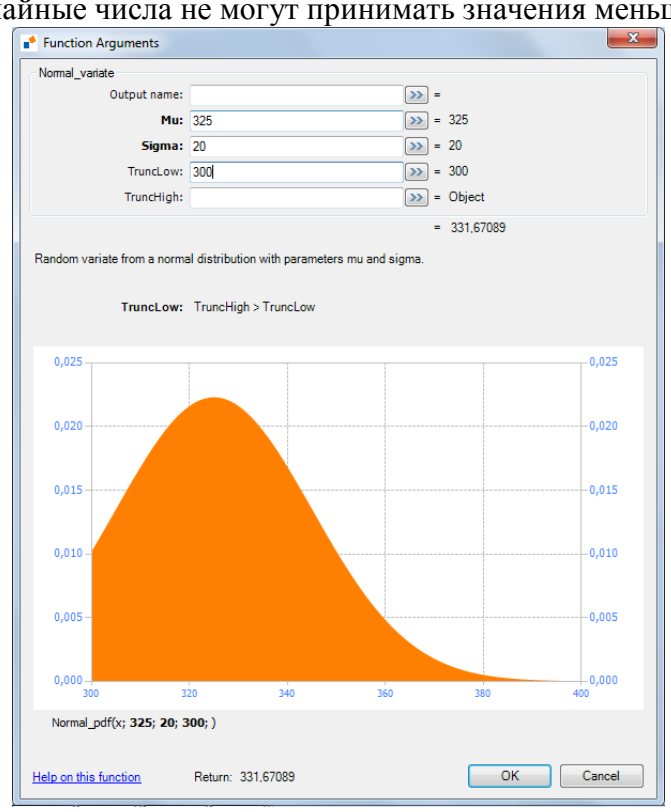


Рис. 5 - Функция нормального распределения с ограничением минимального значения

Наконец, внеплановые доходы компании подвержены кредитным рискам. При совершении клиентами неплатежей до оплаты всех счетов, компания теряет деньги. Будем подразумевать, что количество клиентских неплатежей в фискальном году представляет закон распределения Пуассона с ожидаемым значением неплатежей равным 5. В случае неплатежа, будем подразумевать, что величина каждой потери представляет PERT-распределение с минимальным и максимальным значениями 0 и 10 соответственно, и наиболее вероятным значением равным 3. Таким образом, мы имеем здесь два источника неопределенности. Во-первых, это число неплатежей, во-вторых, размер самих потерь имеет неопределенных характер и представляет собой сумму случайных чисел со случайным числом суммандов. Для моделирования такого поведения, воспользуемся функцией Compound пакета Risk Kit, которая принимает в качестве аргумента функцию распределения и размер суммируемой последовательности.

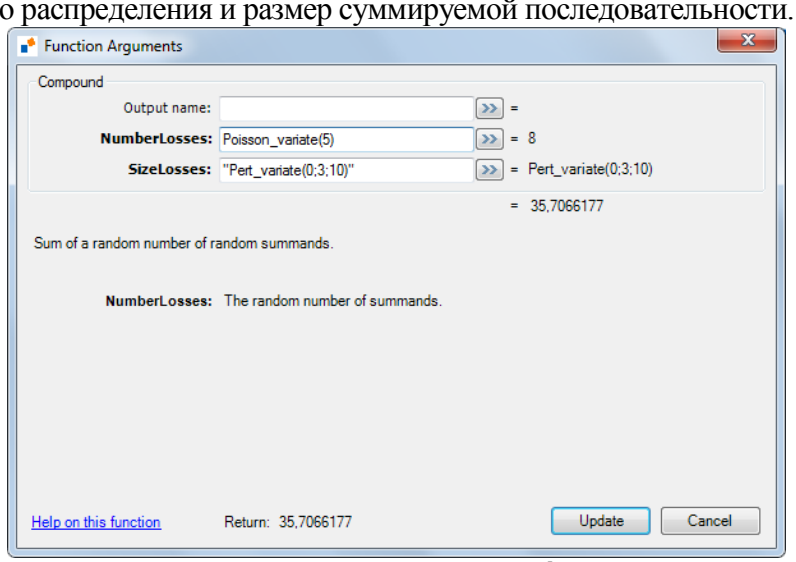


Рис. 6 - Функция Compound

Теперь мы получили интересующую нас модель.

	A	B	C
1	***Расчет прибыли и убытков***		
2	в у.е.		
3		План	Итого (у.е)
4	Оборот	1007,52	1 007,52
5	Материальные издержки	53%	534,51
6	Персональные издержки	330,13	330,13
7	Непредвиденные расходы	75,00	75,00
8	EBIT (Доходы до выплаты процентов и налогов)		67,89
9	Выплата процентов	-50,00	-50,00
10	Внеплановые доходы	9,73	0,00
11	EBT (Доходы до выплаты налогов)		17,89

Рис. 7 - Модель прибыли-убытков

Определение выходов модели

После того как модель описана, мы не можем запустить симуляцию, поскольку выходы модели все еще не определены. Как и в детерминистической модели, мы заинтересованы в поведении значений EBIT и EBT под влиянием рисков. Таким образом, мы определяем эти значения как выходы модели. Для этого достаточно выделить интересующие нас ячейки и воспользоваться кнопкой ‘Output’ (Выход) на панели инструментов Risk Kit.

	A	B	C
1	***Расчет прибыли и убытков***		
2	в у.е.		
3		План	Итого (у.е)
4	Оборот	1008,95	1 008,95
5	Материальные издержки	46%	468,88
6	Персональные издержки	347,31	347,31
7	Непредвиденные расходы	75,00	75,00
8	EBIT (Доходы до выплаты процентов и налогов)		117,76
9	Выплата процентов	-50,00	-50,00
10	Внеплановые доходы	24,75	0,00
11	EBT (Доходы до выплаты налогов)		67,76

Рис. 8 - Определение выходов модели

При этом, ячейки, отмеченные как выход модели, окрашиваются в оранжевый цвет.

Настройка параметров симуляции

Одним из главных параметров является число симуляций. Для изменения этого параметра необходимо воспользоваться кнопкой ‘Config’ (Параметры) на панели инструментов Risk Kit. В появившемся окне установите интересующее вас количество итераций. В данном примере мы оставим значений по умолчанию, равное 5000 итераций.

Помимо этого, установим флажок ‘Dump Standard Statistics’ (Получить Статистику), который позволяет получить статистику для выходов модели, включающую в себя среднее значение, среднеквадратичное отклонение, диапазон значений и т.д.

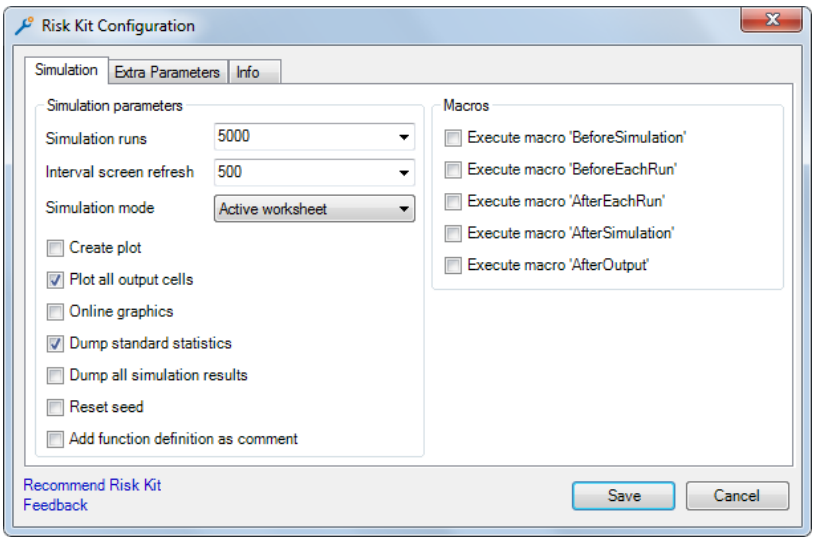


Рис. 9 - Окно настройки параметров симуляции

Установим также флажок ‘Plot all Output Cells’ (Создать Графики для всех выходов) для автоматического построения графиков для всех выходов модели по окончании симуляции.

Запуск симуляции

После того как настройка параметров завершена, мы можем приступить непосредственно к симуляции. Для этого воспользуемся кнопкой ‘Simulate’ (Симулировать) на панели инструментов Risk Kit.

В процессе симуляции, текущая итерация и общее количество итераций отображаются в строке состояния Excel.

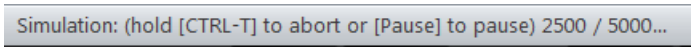


Рис. 10 - Процесс симуляции в строке состояния Excel

По окончании симуляции, Risk Kit выводит статистику для выходов модели на вновь созданный рабочий лист.

	A	B	C
1	**Number of Simulation Runs**	5000	
2	**Simulation Time**	08.90 sec.	
3			
4			
5			
6		**Out: EBIT**	**Out: EBT**
7	**Average Value**	88,21708004	38,21708004
8	**Standard Deviation**	37,65976144	37,65976144
9	**Variance**	1418,257632	1418,257632
10	**Skewness**	-0,044411531	-0,044411531
11	**Kurtosis**	2,958929195	2,958929195
12	**Coefficient of Variation**	0,426898753	0,985417028
13	**Range**	266,0765926	266,0765926
14	**Range5_95**	124,1719528	124,1719528
15	**Minimum Value**	-54,89263772	-104,8926377
16	**Expected Tail <= 0.1%**	-38,99071367	-88,99071367
17	**0.01% - Quantile**	-54,89263772	-104,8926377
18	**0.02% - Quantile**	-54,89263772	-104,8926377
19	**0.03% - Quantile**	-37,50445987	-87,50445987

Рис. 11 - Статистика моделируемых значений

Кроме того, Risk Kit автоматически создает графики для всех выходов модели, которые позволяют сделать вывод о характере моделируемых значений.

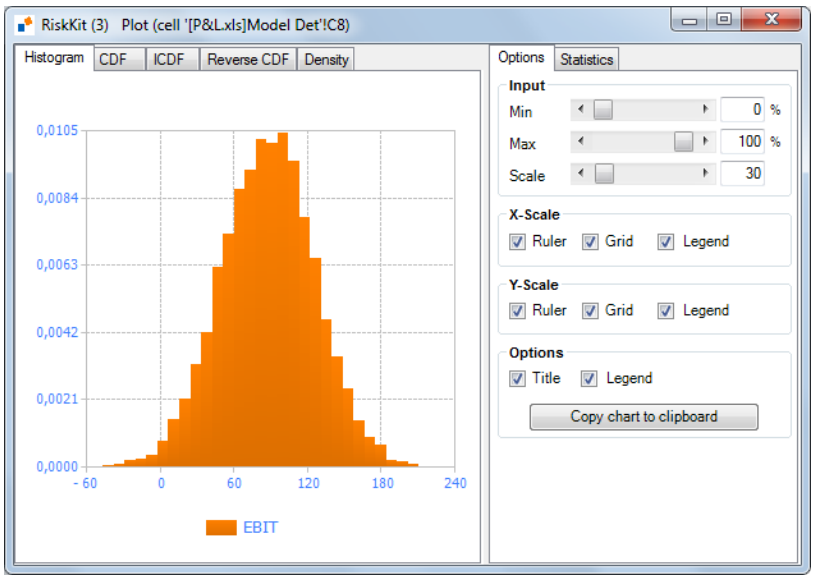


Рис. 12 - График вывода модели

Таким образом, применение пакета Risk Kit позволяет снизить затраты и требования к подготовке пользователя. Это делает доступным использование пакета широким кругом специалистов при минимальных требованиях к уровню их подготовки.

Литература

1. Ермаков С.М. Метод Монте-Карло *в* вычислительной математике. (Вводный курс). — Москва: Бином. Лаборатория знаний, 2011. – 192 с.

2. Джекел П. Применение методов Монте-Карло в финансах. –М.: Интернет-Трейдинг, 2004. – 256 с.

3. Белова И.М. Компьютерное моделирование. Учебно-методическое пособие для студентов направления «Прикладная математика и информатика» и специальности «Математическое обеспечение и администрирование информационных систем». — М.: МГИУ, 2007. — 81 с.

4. Wehrspohn, Uwe and Zhilyakov, Sergey, Rapid Prototyping of Monte-Carlo Simulations (May 2, 2011). Available at SSRN: http://ssrn.com/abstract=1831586 or http://dx.doi.org/10.2139/ssrn.1831586

УДК 004.416

ПОСТРОЕНИЕ СИСТЕМЫ ДОСТАВКИ ОБНОВЛЕНИЙ ПРОГРАММНЫХ ПРОДУКТОВ

В.Г.Захаров, А.Ю.Крайнов, С.В.Липатова, А.А.Смагин

Ульяновский государственный университет

Аннотация: статья посвящена рассмотрению вопроса автоматизации процесса сопровождения программной продукции, а именно, одному из основных его процессов – процессу обновления. Вниманию читателя представлен обзор существующих систем, обеспечивающих автоматизацию обновления ПП и предлагается вариант построения системы, учитывающий возможность расширения системы для автоматизации процесса анализа данных, получаемых по обратной связи с потребителем, и внедрения средств автоматизации документальной поддержки процесса сопровождения.

Ключевые слова: автоматизация сопровождения программной продукции, обновление программного обеспечения.

В быстро меняющимся мире информационных технологий качество сопровождения программного обеспечения (ПО) стало одним из важнейших критериев выбора программного продукта (ПП), гарантирующего, что программа не устареет, будет развиваться и адаптироваться к изменениям программно-аппаратных платформ [7].

Помимо поддержания ПП в актуальном состоянии, важной целью процессов сопровождения является ликвидация уязвимостей ПО. Способность анализировать угрозы безопасности корпоративного ПО и составлять план обновления с учётом этих данных является важным параметром при выборе поставщика программной продукции и системы обновления, особенно на западном рынке [1, 2].

При оценке качества сопровождения необходимо учитывать наличие подробной документации, эффективность службы поддержки, время пребывания фирмы на рынке и систему обновления ПП. С появлением сети Интернет процесс сопровождения существенно упростился: стало возможным не только удалённо оказывать техническую поддержку пользователям ПО, но и оперативно выпускать и применять обновления, чем многие разработчики ПО уже воспользовались [8].

Подходы к построению систем обновления

Однако процедура применения выпущенного обновления у разных разработчиков выглядит по-разному, и сейчас существует множество видов систем обновления (см. таблицу 1). Каждый вид разрабатывался для определенных систем и потребителей, имеющих разные потребности. Одни системы ориентируются, в первую очередь, на программную платформу, другие на конечного пользователя, третьи на разработчика или администратора.

Современные крупные компании, такие как Hewlett-Packard, при составлении моделей управления ИТ-инфраструктурой предприятия уже давно ориентируются на гетерогенные корпоративные системы, то есть системы, основанные на различных платформах и использующие ПО от различных поставщиков [4]. Для предприятия, занимающегося разработкой ПО под разные платформы, необходимо средство независимое от платформы и поддерживающие несколько ПП, т.е. системные и альтернативные службы обновлений, менеджеры обновлений операционных систем, системы обновлений отдельных ПП и открытые библиотеки автоматизации обновления не подходят и могут быть лишь составными элементами системы.

Также для предприятия, клиенты которого не физические лица, а различные фирмы и организации, необходима система обновления, ориентированная, прежде всего, на персонал по сопровождению, т.к. при развертывании систем и решении возникающих проблем взаимодействие происходит не с конечным пользователем, а с администраторами

информационных систем. В ряде организаций из-за специфики объектов сопровождения, любое изменение ПП согласовывается и производится администратором, конечный пользователь не имеет прав на изменение ПО. С этой точки зрения, наиболее подходящими являются системы обновления гетерогенного программного обеспечения корпоративных информационных систем.

Кроме выше перечисленного необходимо, чтобы современная система обновлений осуществляла:

- мониторинг сопровождаемых объектов;
- обратную связь пользователя (администратора системы) с разработчиком;
- консолидацию данных о сопровождаемом ПО, истории его изменений, текущего состояния сопровождаемых объектов на предприятии-разработчике и об установленных ПП, их обновлениях, текущем состоянии рабочих станции и процессов обновления на объектах.

На практике представлен ряд продуктов, ориентированных на решение перечисленных задач (таблица 2). Основные недостатки таких систем — это их высокая стоимость. Кроме этого в них не проработан вопрос безопасного и надежного распространения обновлений и системной информации при сопровождении ПП в гетерогенной среде, в которой присутствует открытые и закрытые сегменты. Внести изменения в готовый программный продукт по требованию распространителя обновлений является в случае коммерческих систем невозможной задачей, а в случае открытых — трудоемкой и не рентабельной, так как добавление модулей требует согласования на разных уровнях функционирования системы.

Таблица 1. Классы систем сопровождения ПО

№	Наименование класса ПО	Назначение	Примеры	Преимущества	Недостатки
1	*Системы обеспечения обновления гетерогенных корпоративных информационных систем*(*)	Обновление ПО от различных поставщиков, установленного в корпоративной среде.	➢ Windows Server Update Services (WSUS) ➢ ZENworks Patch Management ➢ IBM Tivoli Endpoint Manager	➢ Поддержка групп ПО и управление зависимостями(**) ➢ Управление обновлением ПО на рабочих станциях из одной точки. ➢ Развитый интерфейс пользователя и широкий выбор параметров. ➢ Ориентированы на пользовтелей-администраторов. ➢ Сбор статистики по состоянию обновления рабочих станций. ➢ Экспертный анализ собранных данных.	➢ Ориентированы на работу с ПО от крупных поставщиков и/или на конкретную платформу. ➢ Высокая стоимость. ➢ Функции интеграции с собственными ПП ограничены. ➢ Функции создания обновлений ограничены.
2	Системные службы обновления операционных систем	Обновление системного ПО, установленного на персональном компьютере или рабочей станции(ПК/PC).	➢ Windows Update ➢ Microsoft Update ➢ Apple Software Update	➢ Тесная интеграция с системными службами. ➢ Присутствуют функции обратной связи.	➢ Ориентированы на системное ПО. ➢ Функции интеграции с собственными ПП отсутствуют.
3	Альтернативные службы обновления операционных систем	Обновление системного ПО, установленного на ПК/PC (расширенные возможности по настройке).	➢ AutoPatcher ➢ Project Dakota ➢ Sparkle	➢ Расширенные возможности по настройке процесса обновления.	➢ Менее тесная интеграция с системными службами. ➢ Ориентированы на системное ПО. ➢ Функции интеграции с собственными ПП отсутствуют.
4	*Менеджеры пакетов операционных систем*	Обновление ПО (не обязательно системного), установленного на ПК/PC.	➢ RPM Package Manager ➢ Advanced Packaging Tool ➢ Windows Installer	➢ Поддержка групп ПО и управление зависимостями. ➢ Управление версиями. ➢ Ориентированы на работу с известными форматами пакетов обновлений. ➢ Не ограничены по работе с отдельным классом ПО. ➢ Часто являются проектами с	➢ Функции обновления ПО (в т. ч. настройки процесса) ограничены, часто предоставляются посредством сторонних модулей.

				открытым исходным кодом, который можно повторно использовать.	
5	Платформы цифровой дистрибьюции	Приобретение ПО и мультимедийного содержания и загрузка их на ПК.	➢ Steam ➢ MacAppStore ➢ Ubuntu Software Center	➢ Удобный и интуитивно понятный пользовательский интерфейс.	➢ Функции настройки обновлений сильно ограничены. ➢ Функции интеграции с собственными ПП ограничены.
6	Системы обновления линеек ПО	Обновление группы ПО, созданного некоторой крупной компанией или корпорацией.	➢ Google Update (Omaha) ➢ Mozilla Software Update ➢ Adobe Update	➢ Единообразный пользовательский интерфейс.	➢ Нет функций интеграции с собственными ПП.
7	Системы обновления отдельных ПП	Обновление отдельного программного продукта.	➢ Eclipse IDE Update ➢ XSpider Update ➢ Kaspersky Update Utility	➢ Полная интеграция с обновляемым программным продуктом.	➢ Нет функций интеграции с собственными ПП. ➢ Не рассчитаны на работу с группами ПО. ➢ Не рассчитаны на работу в фоновом режиме.
8	*Библиотеки и инструменты автоматизации обновления*	Добавление возможностей по обновлению в разрабатываемый ПП.	➢ JUpdater ➢ StableUpdate ➢ IndigoRoseTrueUpdate	➢ Полная интеграция с обновляемым программным продуктом. ➢ Ориентированы на работу с собственным ПО; широкие возможности по настройке процесса обновлений. ➢ Часто являются проектами с открытым исходным кодом, которые можно дополнять интересующей нас функциональностью.	➢ Ограниченная функциональность. ➢ Интегрируются отдельно в каждый ПП. ➢ Не рассчитаны на работу в фоновом режиме.

Потребности современных систем сопровождения ПП

Постоянная работа с разработкой и контролем обновлений подразумевает необходимость постоянного взаимодействия с пользователем ПП, т.е. наличия активной обратной связи, выработки новых управляющих решений, оперативного реагирования на ошибки и нарушения целостности данных в небезопасных каналах передачи сообщений.

Другими словами, требуется систематический контроль событий по обновлению ПО для удаленных пользователей для устранения неисправностей, контроль за состоянием процесса обновления у пользователей, выявление неисполнения или неправомочных действий с их стороны, и сбора статистики по нарушениям требований на стороне пользователей, оперативный учет и реагирование на просьбы о модификации потребителей.

Исходя из представленного выше анализа, предлагается создать систему, в которой особое внимание будет уделено схеме распространения обновления, обратной связи с объектом и пользователями, поэтапной обработке сообщений пользователей. Для этого система обновления должна представлять в общем виде систему с дуплексным каналом передачи данных, с соответствующей организацией и правилами поведения персонала сопровождения на стороне разработчика и персонала на объекте.

Такая схема взаимодействия позволит повысить качество сопровождения ПП и предоставит возможность для автоматизации в дальнейшем этого процесса (посредством внедрения алгоритмов классификации и ранжирования) и процесса документирования внесения изменений в систему (выпуска и проводки бюллетеней посредством алгоритмов генерации текстов).

Архитектура системы сопровождения ПП

Для реализации автоматизированной системы сопровождения программной продукции (АССПП) в распределенной среде предлагается выбрать следующую архитектуру (рис.1):

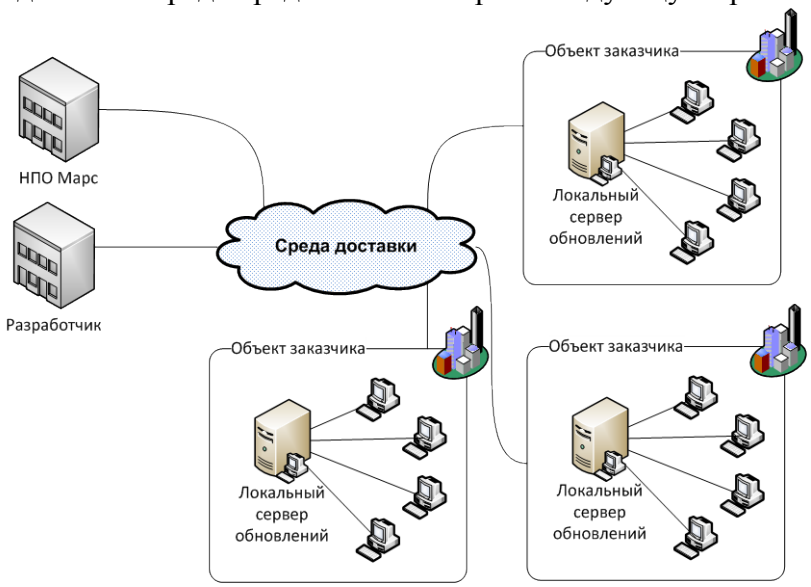


Рис.1. Общая архитектура АССПП

Таблица 2. Системы сопровождения программной продукции.

№	Название ПП	Класс	Описание	Релевантная функциональность
1	Windows Server Update Services (WSUS)	1	Система предназначена для доставки и установки обновлений на рабочие станции предприятия, работающие под управлением операционной системы Windows. Функционирует как промежуточный (прокси) сервер между службой обновления Microsoftи отдельным ПК. Система поддерживает возможность интеграции с ней собственного ПО (платформы Windows). Бесплатная для клиентов Microsoft.	➢ Пользователи системы – администраторы, поэтому конечные пользователи не должны отвечать за обновление ПО на своей рабочей станции. ➢ Управление обновлением из одной точки. ➢ Продуманный пользовательский интерфейс, позволяющий тонко настраивать процесс обновления.
2	IBM Tivoli Endpoint Manager	1	То же, но рабочие станции предприятия могут работать под разными платформами (Windows/Linux/Solarisи т. д.). Ориентирована на работу с разными поставщиками ПО (как правило, крупными компаниями:Microsoft, Adobe, IBM, Nortonи т. п.). Предоставляет расширенные функции по управлению версиями/зависимостями, мониторингу корпоративной сети и т. д.	➢ Кроссплатформенность. ➢ Управление зависимостями ПО, учёт системных требований. ➢ Мониторинг процесса обновления в реальном времени. ➢ Управление обновлениями на уровне мелких пакетов (агенту доставляются только нужные файлы обновлений).
3	ZENworksPatchManagement	1	То же, но возможно создание пакетов обновлений для собственных ПП с помощью дополнительного ПО. Система обеспечивает также дополнительные функции, такие как экспертный анализ уязвимостей рабочих станций.	➢ Поддержка принятия решений по обновлению на основе сбора статистики по рабочим станциям. ➢ Рассылка уведомлений.
4	WindowsInstaller	4	Система предназначена для установки ПОна рабочие станции, работающие под управлением ОС Windows. Функции обновления сильно сокращены, что затрудняет создание обновлений даже в стандартном режиме (когда обновления ПО производятся через отдельный установочный пакет).	➢ Ориентирована на конечного пользователя, что, с одной стороны, позволяет осуществлять строгий контроль за системой, но с другой стороны, оставляет место для нестрогого соблюдения правил безопасности. ➢ Интеграция с системными службами. ➢ Автоматизация процесса установки ПП. ➢ Возможность отложенного выполнения некоторых команд. ➢ Возможность отмены установки.
5	RPMPackageManager	4	То же, но ориентирована на платформу Linux.За счёт	➢ Открытый исходный код.

			сторонних дополнений функциональность системы может быть расширена.Функции обновления присутствуют, но не являются первоочередными.	➢ Контроль версий. ➢ Использование открытого и распространённого формата пакетов (*.rpm).
6	Advanced PackagingTool	4	Является надстройкой над RPM и расширяет её функции по установке и обновлению ПО.	➢ Предоставляет интерфейс интеграции для устанавливаемых (обновляемых) ПП. ➢ Расширенные функции обновления, такие как обновление связанных программных продуктов. ➢ Управление зависимостями.
7	JUpdater	8	Программная библиотека для проверки наличия и загрузки обновлений для Java-приложений с минимальным участием пользователя. Функционирует на уровне отдельного программного продукта.	➢ Пользователи системы – разработчики ПО. ➢ Система встраивается в каждый программный продукт в виде программной библиотеки. ➢ Открытый исходный код, что позволяет включить данную систему (расширив её функциональность) в собственную систему управления обновлениями.
8	StableUpdate	8	То же, но не ограничена платформой Java. Также доступны расширенные функции управления пакетами.	➢ Кроссплатформенность. ➢ Поддержка распределённых хранилищ обновлений. ➢ Строгий контроль версий и возможность возврата к предыдущей версии. ➢ Управление обновлениями на уровне мелких пакетов.
9	IndigoRoseTrueUpdate	8	Инструмент для автоматизации обновления разрабатываемых ПП. Представляет собой клиент-серверное приложение, клиентская часть которого поставляется вместе с ПП, а серверная настраивается с помощью скриптового языка. Это закрытый программный продукт, ориентированный на использование в операционной системе Windows.	➢ Простота интеграции с ПП.

Для обеспечения безопасности и надежности функционирования системы локальный сервер обновлений и рабочие станции пользователей находятся в закрытом сегменте сети (рис. 2).

Сообщения об ошибках и запросы на модификацию ПП поступают на сервер персонала сопровождения на объекте, где они группируются и систематизируются, выявляются наиболее распространенные сообщения. Затем агрегированные данные передаются на предприятие, где персонал по сопровождению проводит анализ и определяет приоритетность каждого сообщения. Выявляются критические ошибки ПО, сигнализирующие об угрозе нарушения целостности системы и правильности ее функционирования, и исправляются в первую очередь, что позволяет принимать превентивные меры, т.к. обнаружение ошибки на разных объектах может происходить в разное время, и обнаруженная на одном, она будет исправлена на всех объектах.

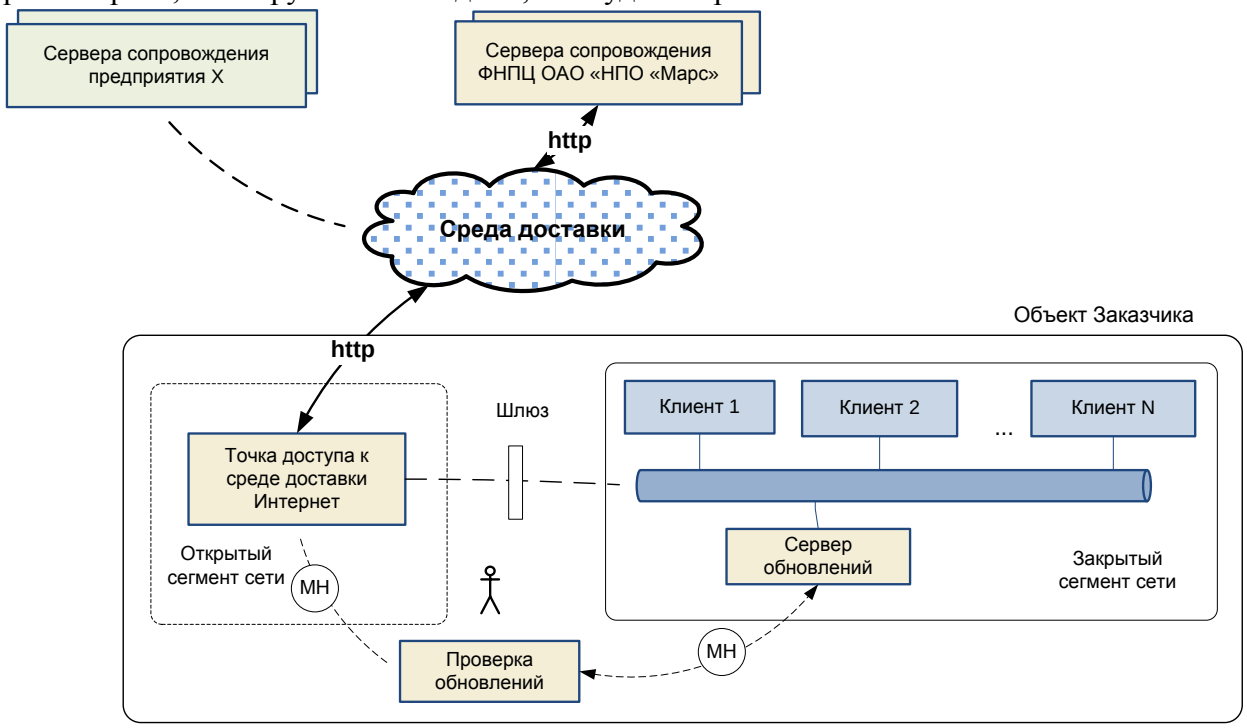


Рис. 2. Сегментация сети объекта

Между открытым и закрытым сегментом сети циркулируют данные о текущем состоянии объекта (от локального сервера обновлений к точке доступа) и обновления (от точки доступа к серверу обновлений). Обновления подвергаются проверке на вирусы, корректность, легитимность осуществляется службой эксплуатации объекта на отдельном автономном рабочем месте. Передача между сегментами сети может осуществляться в ручную или через шлюз.

- В процессе сопровождения участвуют следующие пользователи (рис. 3):
- *Разработчик* – персонал, находящийся на предприятии-поставщике и ответственный за обеспечение доступа потребителям к новым версиям ПП, регламентирование процесса обновления и оперативное реагирование при появлении ошибок в ПП.
- *Архивариус* – работник архива предприятия-поставщика ПП.
- *Администратор /персонал по сопровождению на предприятии* – персонал, ответственный за поддержание ПП на стороне потребителя в актуальном состоянии, находящийся на предприятии-поставщике ПП.
- *Администратор /персонал по сопровождению на объекте* – персонал, ответственный за поддержание ПП на стороне потребителя в актуальном состоянии, находящийся на объекте.
- *Пользователь* – персонал, непосредственно использующий ПП на рабочей станции.

В АССПП предусмотрены все эти типы пользователей и каждому из них доступен свой набор функций (рис.4).

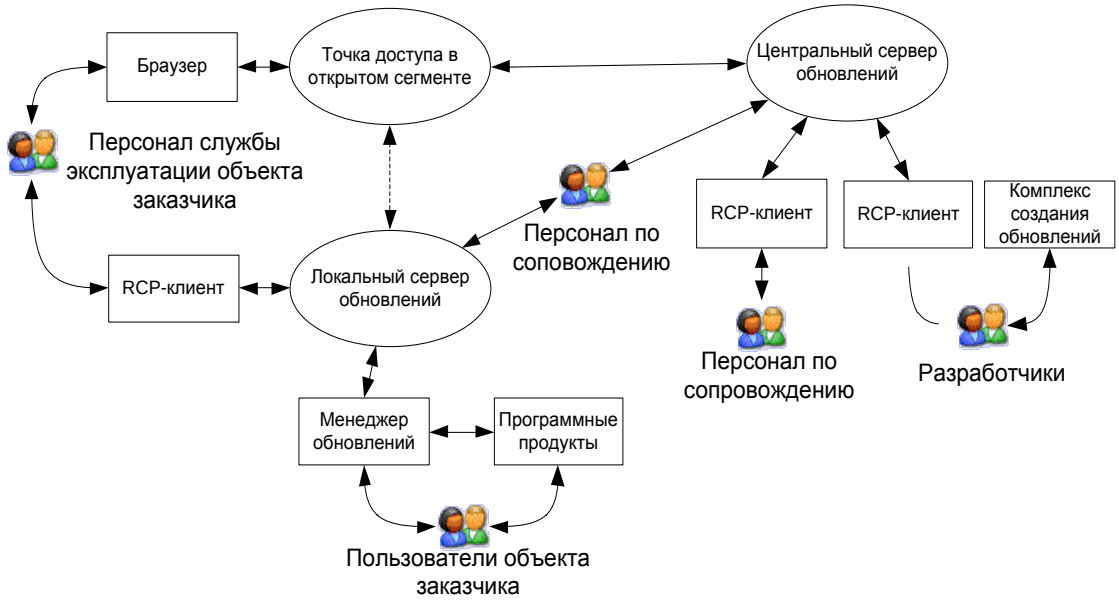


Рис. 3. Функции пользователей АССПП

Разработчик после создания ПП и / или обновления с помощью комплекса создания обновлений формирует архив обновления и помещает его на центральный сервер. Взаимодействие с сервером осуществляется через web-интерфейс. Персонал по сопровождению дополняет описание обновления правилами установки, правами доступа и т.д., так же используя браузер.

Обновление транспортируется на локальный сервер обновления или непосредственно персоналом сопровождения или сеть. При использовании сети персонал службы эксплуатации объекта (персонал сопровождения на объекте) через точку доступа в открытом сегменте, используя web-интерфейс, получает обновление и затем помещает его на локальный сервер обновлений.

Пользователь через менеджер обновлений может получить доступ ко всем доступным ему функциям и использовать установленное на его рабочей машине ПО.

Алгоритм обновления ПП

С учетом этого процесс обновления преобразуются в следующий вид (рис. 5).

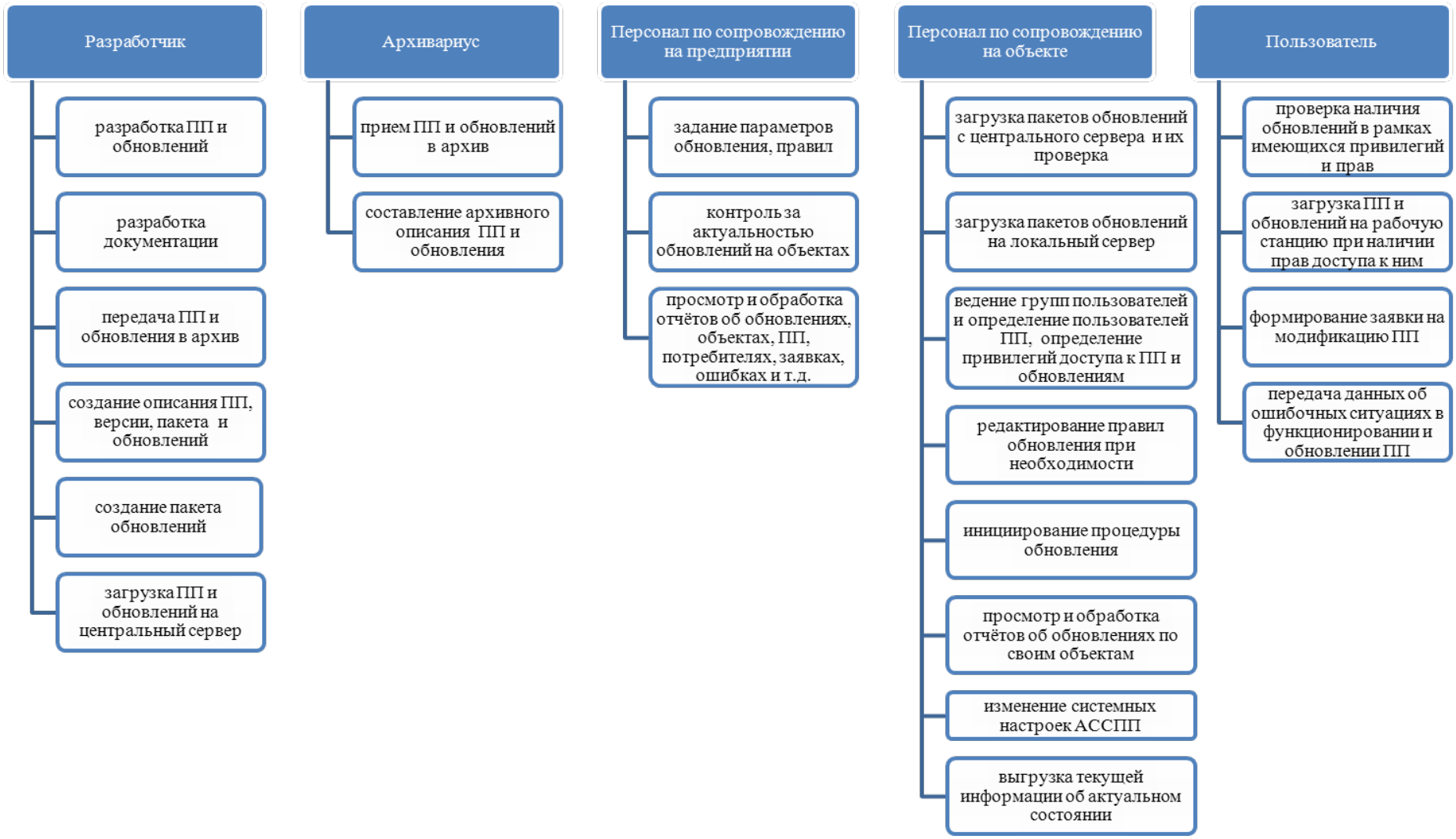


Рис. 4. Функции пользователей АССПП

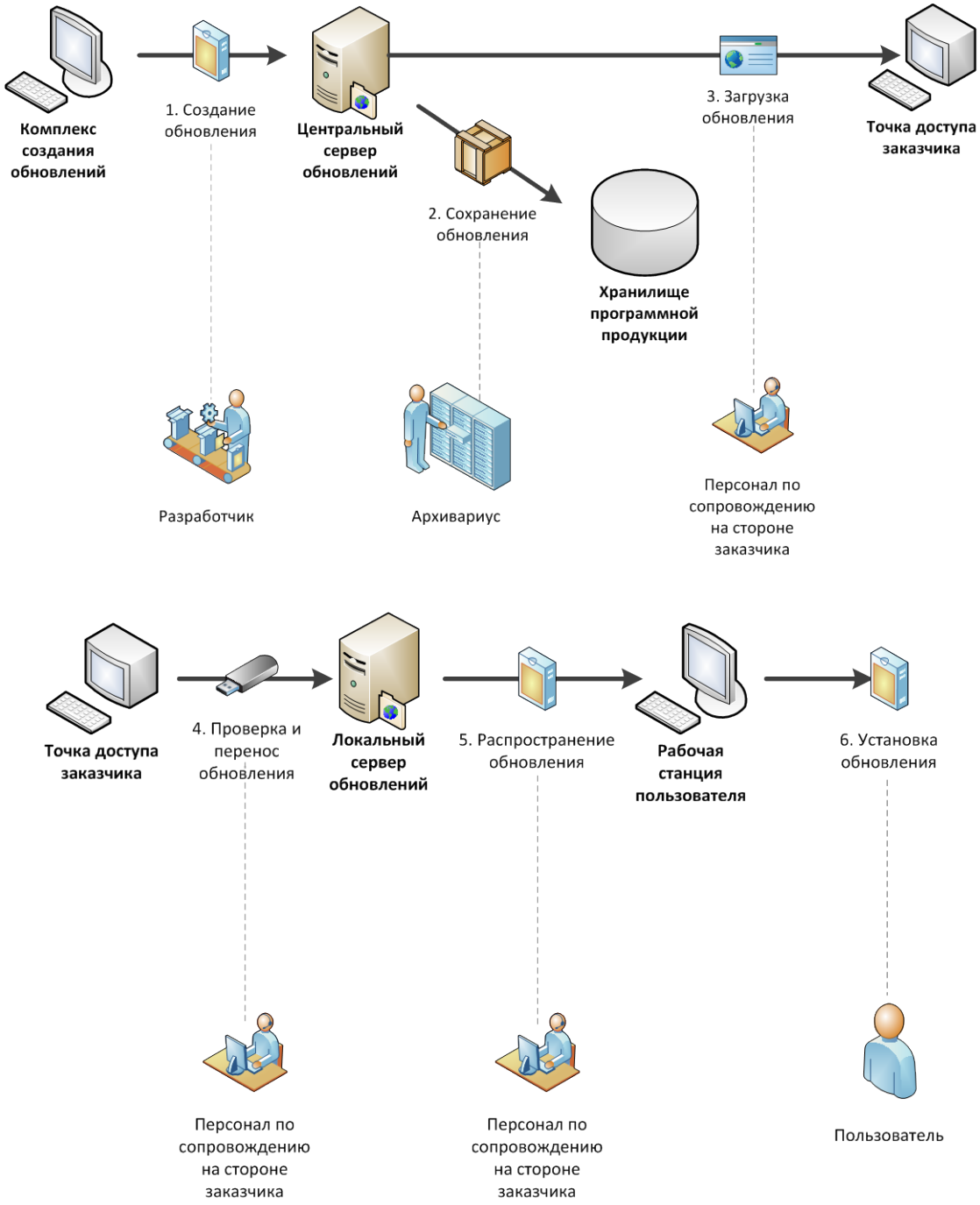


Рис. 5. Автоматизированный процесс обновления ПП

1. Разработчик создает пакет обновлений (пакет включает описание обновления, само обновление и правила его установки).

2. Созданное обновление помещается в архив предприятия и на центральный сервер обновления, позволяющий хранить информацию о ПП, его обновлениях, учитывать версионность и модульность ПП.

3. Производится загрузка пакета обновлений с центрального сервера с открытой точки доступа в сегменте сети объекта. В случаи необходимости повышения уровня безопасности передача обновления осуществляется по безопасным каналам связи.

4. Осуществляется всесторонняя проверка обновления на подлинность и отсутствие угроз безопасности. После прохождения проверки производится загрузка пакета обновления на локальный сервер обновлений в закрытый сегмент сети объекта.

5. С помощью интерфейсов АССПП осуществляется автоматизированное распространение пакета обновления по заданным рабочим станциям и контроль процесса обновления.

6. Осуществляется автоматизированная установка обновления на рабочей станции пользователя с оповещением персонала по сопровождению и пользователя о текущих событиях через интерфейсы АССПП.

Более подробно алгоритм установки обновления на рабочей станции может быть представлен следующими шагами:

1. Общая инициализация пакета обновления, в ходе которой проверяется целостность пакета и возможность произвести установку; также на этом шаге выбирается программный модуль-декодер формата пакета.

2. Получение разрешения от пользователя на установку пакета обновления; при этом пользователь может указать время проведения обновления (например, отложить до перезапуска системы).

3. Распаковка пакета и копирование файлов. Данная процедура состоит из трёх последовательных циклов обработки файла сценария установки:

3.1 Проверка целостности файлов, входящих в пакет обновления, а также файлов, находящихся на рабочей станции (проверка возможности их перезаписи).

3.2 Создание резервной копии локальных файлов.

3.3 Выполнение операций с файловой системой: копирование, удаление, модификация файлов; внесение изменений в системные реестры и каталоги.

4. Оповещение пользователя и персонала по сопровождению о результатах установки; в случае ошибки при установке — затребование у пользователя описания выполненных им действий для отчёта.

Благодаря возможности определения времени установки на шаге 2 появляется возможность установки обновлений, затрагивающих системные файлы, а трёхступенчатый процесс на шаге 3 обеспечивает безопасную процедуру установки пакета обновления и предоставляет возможность возврата к ранней версии.

Для поддержания актуального состояния ПП на объектах заказчика и устранения появляющихся ошибок функционирования ПО и его обновления, необходимо установление обратной связи объекта и поставщика ПП. Для этого на объекте производится сбор данных, а затем они передаются на центральный сервер обновления (рис. 6):

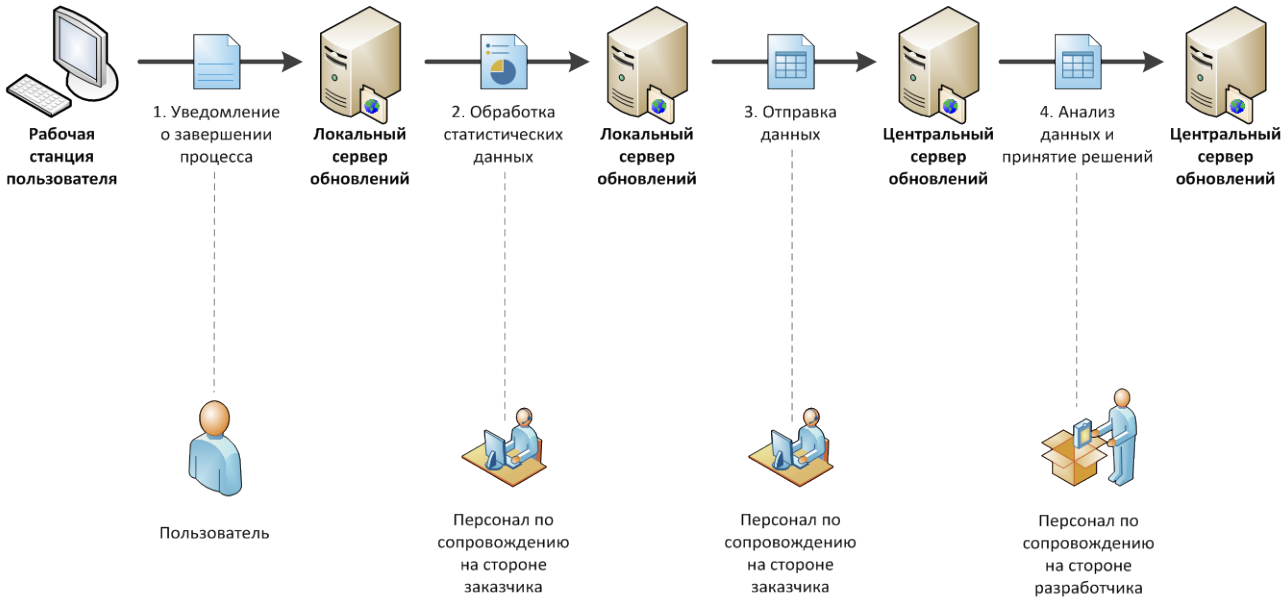


Рис. 6. Автоматизированный процесс получения данных о текущем состоянии объекта

1. После завершения процесса установки или обновления ПП пользователю на рабочей станции выдается сообщение. В зависимости от статуса завершения это может быть сообщение об успешной установке или об ошибке. Аналогичное сообщение, но в машинном формате передается на локальный сервер обновлений (любое действие по изменению состояния рабочей станции по зарегистрированным программам в АССПП заканчивается отсылкой сообщения локальному серверу).
2. Посредством интерфейса администратора персонал по сопровождению на объекте производит обработку накопленных данных об ошибках и о пожеланиях на модификацию, устраняя избыточность данных и подводя итог.
3. Накопленные данные отправляются на локальный сервер обновлений, где хранится информация со множества объектов.
4. Посредством интерфейса администратора персонал по сопровождению на предприятии производит обработку накопленных данных об ошибках и о пожеланиях на модификацию, выявляя существующие проблемы с ПП.

Результат анализа полученных данных с объектов является основанием для формирования заданий на доработку и устранения неполадок в функционировании ПП.

С помощью предлагаемого варианта организации системы сопровождения ПП может решаться ряд таких важных задач как:

- информирование поставщика о текущем состоянии объекта у потребителя (установленном ПО, аппаратно-программной платформе) для повышения эффективности сопровождения за счет организации подготовительной работы на стороне разработчика и дифференцированном подходе к задаче обновления на объектах (появляется возможность гибкого управления правилами обновления и установки для каждого из объектов);
- оперативная доставка обновления (уменьшаются требуемые для этого затраты ресурсов - человеческих, финансовых временных) и уменьшение задержки в процессе обновления ПО после появления новой версии;
- повышение эффективности модификаций ПП за счет использования автоматизированного хранилища ПП и его обновлений ПП, данных об отказах системы, ошибках обновления и принимаемых решениях по их отладке;
- организация автоматизированной обратной связи с объектом (оперативного получения информации об ошибках обновления, функционирования системы, пожелания о модификации

системы) и оперативное оценивание вносимых изменений в ПО на объектах пользователями, увеличение скорости обработки ошибок;

- организация развертывания и интеграции ПО на объекте удаленно, правила обновления переносятся на объект вместе с самим обновлением и система контролирует их применение (таким образом, решается проблема нехватки квалифицированных специалистов способных провести интеграцию при большом количестве объектов и ПО).

Литература

1. Lippmann R., Webster S., Stetson D. The effect of identifying vulnerabilities and patching software on the utility of network intrusion detection // Lecture Notes in Computer Science. — Springer-Verlag GmbH, 2002. — Т. 2516. — С. 307—326.
2. Savage M. Patch management software catching on // CRN Magazine. — US: Proquest ABI/INFORM, 2002. — № 1023. — С. 43.
3. Арустамов С. А., Генин М. Г. Минимизация рисков потери доступности программного обеспечения после установки обновлений или изменения функциональности // Научно-технический вестник информационных технологий, механики и оптики, 2011. — № 73. — С. 111—116.
4. Курц А. Л. и др. Принципы построения средств управления ИТ-инфраструктурой на примере модели ITSM компании HP / Курц А. Л., Фридман А. Л., Андерс Б. Н., Фандюшина Н. А., Чумаков Л. Я. // Системы и средства информатики. — М.: Институт проблем информатики РАН, 2008. — Т. 18. — № 2. — С. 69—85.
5. Петров Е. В. Обновление программного обеспечения в распределённых управляющих системах // Научно-технический вестник информационных технологий, механики и оптики, 2007. — № 45. — С. 65—70.
6. Смит Р. Публикация чужих обновлений в службах WSUS // Windows IT Pro/RE. — М.: Открытые системы, 2011. — № 3. — С. 42—46.
7. Тараненко А. Процесс управления обновлениями // Windows IT Pro/RE. — М.: Открытые системы, 2009. — № 7. — С. 34—37.
8. Шадрин В. В. Реализация системы синхронизированного обновления компонентов операционной системы, прикладного программного обеспечения и средств защиты // Горный информационно-аналитический бюллетень. — М.: Издательство Московского государственного горного университета, 2008. — № 2. — С. 303—309.

ПРОЦЕДУРЫ АНАЛИЗА ДОСТИЖИМОСТИ УСТОЙЧИВЫХ СОСТОЯНИЙ ЦИФРОВЫХ АВТОМАТОВ

В.В.Кожевников, А.А.Смагин

Ульяновский государственный университет

Введение

Процедуры разработаны на основе метода анализа достижимости устойчивых состояний логических схем цифровых автоматов [1]. В настоящей работе приводятся следующие процедуры:

- модельного представления цифровых автоматов,
- анализа достижимости устойчивых состояний цифровых автоматов,
- построения протоколов достижимости устойчивых состояний цифровых автоматов.

Моделирование проводится с целью анализа корректности логических схем цифровых автоматов и сводится к решению задач анализа достижимости и построения протоколов достижимости устойчивых состояний с последующим анализом безопасности схемы на базе полученных протоколов. В качестве инструмента моделирования используется математический аппарат сетей Петри (СП) [2].

Комплексная модель цифровых автоматов может быть представлена в виде уравнений состояний СП из класса уравнений Мурата [3]. Задачи анализа достижимости и построения протоколов достижимости устойчивых состояний цифровых автоматов сводятся решению уравнений состояний СП при заданном критерии достижимости.

1. Процедура модельного представления логических схем цифровых автоматов

Исходной информацией для построения сетевой модели логических схем служит описание структурной схемы. Степень декомпозиции компонентов схемы должна обеспечивать возможность представления этих компонентов в виде таблицы истинности. При этом точность моделирования зависит от степени декомпозиции компонентов схемы.

Процедура модельного представления логических схем включает в себя следующие этапы:

1.1 Представление исходной структурной схемы в виде маркированного графа

Структурная схема преобразуется в маркированный граф путем интерпретации входов и выходов логических схем и структурных компонентов позициями маркированного графа, а самих компонентов и линий соединения составными и простыми переходами соответственно.

В качестве примера на рисунке 1 приведена структурная схема асинхронного RS-триггера представленная в виде маркированный графа.

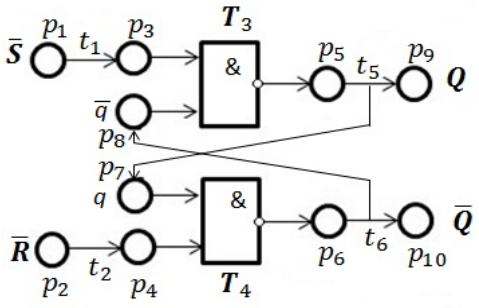


Рис.1. Маркированный граф схемы RS-триггера

1.2. Преобразование маркированного графа в СП со свободным выбором

Таблицы истинности компонентов преобразуются в сетевые модели путем интерпретации наборов из таблицы истинности переходами, а соответствующей логики входными и выходными дугами переходов. На рисунке 2 приведен маркированный граф

схемы RS-триггера , который в результате подстановки сетевых моделей компонентов преобразуется в СП со свободным выбором.

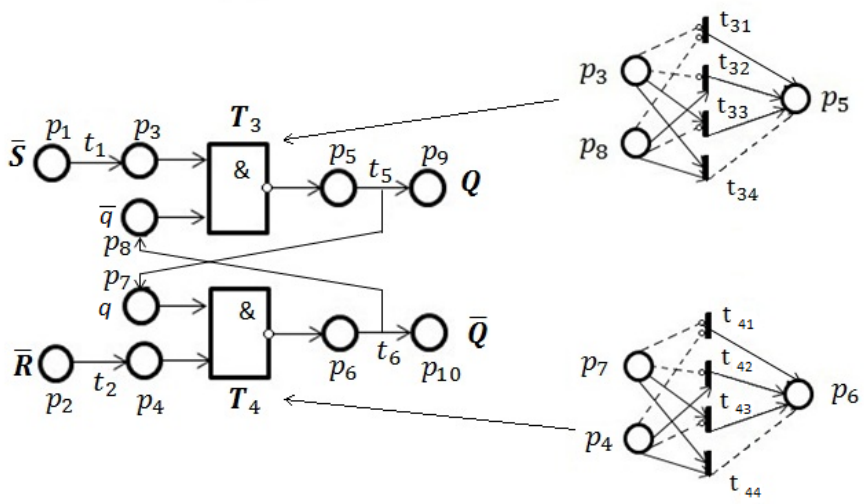


Рис.2. Сетевая модель схемы RS-триггера.

1.3. Представление сетевой модели в виде матрицы инцидентности

Графическая форма представления логических схем позволяет перейти от описания логической схемы к их математическому представлению в виде матрицы инцидентности **A** = $\mathbf{A}^{+1}$ - $\mathbf{A}^{-1}$, где $\mathbf{A}^{-1}$ - матрица, задающая множество отношений между входными позициями и переходами , $\mathbf{A}^{+1}$ - матрица, задающая множество отношений между переходами и выходными позициями переходов. Таблицы истинности компонентов схемы, где единичные значения входных переменных берутся со знаком минус, а выходные со знаком плюс, представляют собой матрицы инцидентности компонентов.

Матрица инцидентности сетевой модели RS-триггера **A** может быть представлена следующим образом:

A =

		t_1	t_2	t_{31}	t_{32}	t_{33}	t_{34}	t_{41}	t_{42}	t_{43}	t_{44}	t_5	t_6
$\bar{S}$	p_1	−1	0	0	0	0	0	0	0	0	0	0	0
$\bar{R}$	p_2	0	−1	0	0	0	0	0	0	0	0	0	0
	p_3	1	0	0	0	−1	−1	0	0	0	0	0	0
	p_4	0	1	0	0	0	0	0	−1	0	−1	0	0
	p_5	0	0	1	1	1	0	0	0	0	0	−1	0
	p_6	0	0	0	0	0	0	1	1	1	0	0	−1
q	p_7	0	0	0	0	0	0	0	0	−1	−1	1	0
$\bar{q}$	p_8	0	0	0	−1	0	−1	0	0	0	0	0	1
Q	p_9	0	0	0	0	0	0	0	0	0	0	1	0
$\bar{Q}$	p_{10}	0	0	0	0	0	0	0	0	0	0	0	1

Матрицы $\mathbf{A}^{+1}$ и $\mathbf{A}^{-1}$ могут быть представлены соответственно:

$$A^- = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \\ p_{10} \end{array} \begin{array}{c} \begin{array}{cccccccccccc} t_1 & t_2 & t_{31} & t_{32} & t_{33} & t_{34} & t_{41} & t_{42} & t_{43} & t_{44} & t_5 & t_6 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$A^+ = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \\ p_9 \\ p_{10} \end{array} \begin{array}{c} \begin{array}{cccccccccccc} t_1 & t_2 & t_{31} & t_{32} & t_{33} & t_{34} & t_{41} & t_{42} & t_{43} & t_{44} & t_5 & t_6 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

2. Процедура анализа достижимости устойчивых состояний цифрового автомата

Задача анализа достижимости устойчивых состояний сети в свою очередь может быть сведена к решению уравнения состояния:

$$\Delta\boldsymbol{\mu} = A\,\boldsymbol{\tau}\ , \qquad (1)$$

где $\Delta\boldsymbol{\mu} = \boldsymbol{\mu}_f - \boldsymbol{\mu}_0$, $\boldsymbol{\mu}_0$ - вектор начальной разметки сети, $\boldsymbol{\mu}_f$ - вектор конечной разметки сети, $\boldsymbol{\tau}$ - вектор покрытия множества переходов сети. Множество достижимых состояний сети определяется множеством пар векторов $\{\Delta\boldsymbol{\mu}, \boldsymbol{\tau}\}$, которые определяют диаграммы переходов и состояний цифрового автомата (диаграммы Мура).

Процедура анализа достижимости устойчивых состояний цифровых автоматов включает в себя следующие этапы:

2.1. Построение уравнения состояний цифровых автоматов

Выполняется путем подстановки матрицы инциндентности в уравнение состояний (1). Для RS-триггера уравнение состояний (1) примет вид:

$$\begin{bmatrix} y \\ y \\ y \\ y \\ y \\ y \\ y \\ y \\ y \\ y \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{bmatrix}$$

2.2. Формирование критерия достижимисти

Выполняется путем определенич вектора $\Delta\boldsymbol{\mu}$ на множестве внутренних позиций сети как вектора $\Delta\boldsymbol{\mu}(P^0) = \mathbf{0}$, где P^0 - множество внутренних позиций сети**.** В результате однородное уравнение состояний для RS-триггера примет вид:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

2.3. Решение уравнения состояний цифровых автоматов

Множество решений уравнения состояний (1) может быть получено любым из существующих методов решения систем линейных алгебраических уравнений при частичном определении вектора $\boldsymbol{\tau}$. Частичное определение вектора $\boldsymbol{\tau}$ на множестве переходов компонентов схемы путем перебора их возможных комбинаций:

	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_8	τ_9	τ_{10}	τ_{11}	τ_{12}	τ_{13}	τ_{14}	τ_{15}	τ_{16}
t_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1	x_1
t_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2	x_2
t_{31}	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
t_{32}	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
t_{33}	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
t_{34}	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
t_{41}	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
t_{42}	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
t_{43}	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
t_{44}	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
t_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5	x_5
t_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6	x_6

Каждая из комбинаций переходов в последовательно подставляется в уравнение уравнение (1). Для $\boldsymbol{\tau}_2$ уравнение (1) на множестве внутренних позиций сети примет вид:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

В результате решения уравнения получаем следующие значения переходов вектора $\boldsymbol{\tau}_2$:

$\boldsymbol{\tau}_2 =$

Доопределение вектора $\boldsymbol{\Delta\mu}$ выполняется подстановки полученного вектор $\boldsymbol{\tau}_2$ уравнение состояний (1) на множестве всех позиций сети :

$$\begin{bmatrix} y_1 \\ y_2 \\ 0 \\ 0 \\ 0 \\ 0 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

В результате решения уравнения имеем вектор $\boldsymbol{\Delta\mu_2}$ **:**

$$\Delta\boldsymbol{\mu}_2 = [0000001011]$$

2.4. Вычисление множества достижимых состояний автомата

Выполнается путем разложения вектор $\boldsymbol{\Delta\mu}$ на вектор начальной и вектор конечной разметки на множество внутренних позиций сети, входящих в состав обратных связей и определяющих состояние автомата P^q. Так на примере вектора $\boldsymbol{\Delta\mu_2}$ вектор начальной и вектор конечной разметки могут быть получены следующим образом:

$$\mu_0(\boldsymbol{P^q}) = \boldsymbol{A^-}(\boldsymbol{P^q}, \mathbf{T})\boldsymbol{\tau}_2$$

$$\mu_0(\boldsymbol{P^q}) = \left[\begin{array}{cccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hdashline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$\mu_0 = [0000000100]$$

$$\mu_f(P^q) = A^+(P^q, \mathbf{T})\boldsymbol{\tau}_2$$

$$\mu_f(P^q) = \left[\begin{array}{cccccccccccc} 0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0\\ 1&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&1&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&1&1&0&0&0\\ \hline 0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&1\\ \hline 0&0&0&0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&0&0&0&1 \end{array}\right] \cdot \left[\begin{array}{c} 0\\0\\0\\1\\0\\0\\1\\0\\0\\0\\1\\1 \end{array}\right]$$

$$\mu_f = [0000001111]$$

Все множество решений для RS-триггера приведено ниже**:**

$\tau_1 = [001000100011]$ $\Delta\mu_1$=[0 000001111] = [0000001111] – [0000000000]

$\tau_2 = [000100100011]$ $\Delta\mu_2$=[0000001011] = [0000001111] – [0000000100]

$\tau_3 = [100010100011]$ $\Delta\mu_3$=[-1000001111] = [0000001111] – [1000000000]

$\tau_4 = [100001100001]$ $\Delta\mu_4$=[-1000000001] = [0000000101] – [1000000100]

$\tau_5 = [011000010011]$ $\Delta\mu_5$=[0-100001111] = [0000001111] – [0100000000]

$\tau_6 = [010100010011]$ $\Delta\mu_6$=[0-100001011] = [0000001111] – [0100000100]

$\tau_7 = [110010010011]$ $\Delta\mu_7$=[-1-100001111] = [0000001111] – [1100000000]

$\tau_8 = [110001010001]$ $\Delta\mu_8$=[-1-100000001] = [0000000101] – [1100000100]

$\tau_9 = [001000001011]$ $\Delta\mu_9$=[0000000111] = [0000001111] – [0000001000]

$\tau_{10} = [000100001011]$ $\Delta\mu_{10}$=[0000000011] = [0000001111] – [0000001100]

$\tau_{11} = [100010001011]$ $\Delta\mu_{11}$=[-1000000111] = [0000001111] – [1000001000]

$\tau_{12} = [100001001001]$ $\Delta\mu_{12}$=[-100000-1001] = [0000000101] – [1000001100]

$\tau_{13} = [011000000110]$ $\Delta\mu_{13}$=[0-100000010] = [0000001010] – [0100001000]

$\tau_{14} = [010100000110]$ $\Delta\mu_{14}$=[0-100000-110] = [0000001010] – [0100001100]

$\tau_{15} = [110010000110]$ $\Delta\mu_{15}$=[-1-100000010] = [0000001010] – [1100001000]

$\tau_{16} = [110001000100]$ $\Delta\mu_{16}$=[-1-10000-1-100]=[0000000000] – [1100001100]

Множество переходов между достижимыми устойчивыми состояниями сети определяется множеством пар векторов $\{\Delta\boldsymbol{\mu}, \boldsymbol{\tau}\}$, которые могут быть представлены в виде диаграммы переходов и состояний автоматов (диаграммы Мура). Соответствующая диаграмма переходов и состояний RS-триггера приведена на рисунке 3.

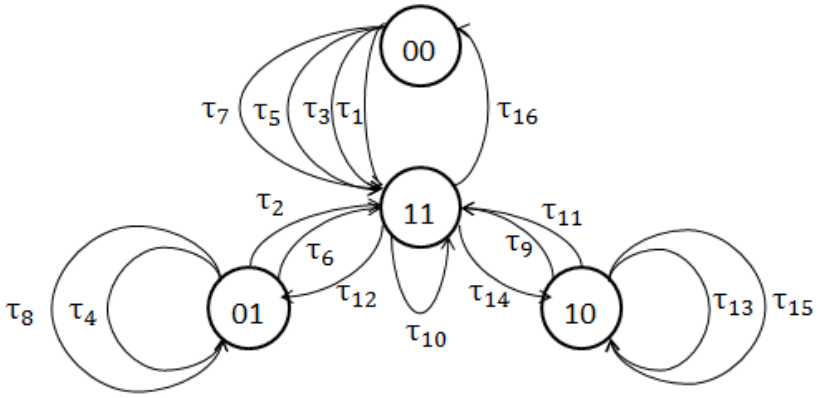


Рис. 3. Диаграмма Мура RS-триггера

Состояние $\mu(P^q) = \{00\}$ – является неустойчивым и не может бить сохранено ни при одной входной комбинации сигналов S и R на входах триггера. Состояния $\mu(P^q) = \{01\}$ и $\mu(P^q) = \{10\}$ являются устойчивыми и могут сохраняться бесконечно долго при отсутствии сигналов S и R на входах триггера. Состояние $\mu(P^q) = \{11\}$ является переходным и может быть сохранено только при подаче аналогичных сигналов S и R на входы триггера. Из приведенной на рис.3. диаграммы видно, что для перехода из одного устойчивого состояния в другое необходимо выполнить два цикла срабатывания RS-триггера.

3. Процедура построения протоколов достижимости устойчивых состояний цифрового автомата

Задача построения протоколов достижимости устойчивых состояний сводится к вычислению последовательности векторов запуска переходов и текущей разметки сети для каждого вектора τ начиная с вектора μ_0 и до тех пор, пока не будет достигнута разметка μ_f.

Последовательность векторов запуска переходов и векторов текущей разметки может быть получена путём итеративного решения системы линейных алгебраических уравнений:

$$\mu_к = \mu_{к-1} + A \cdot \overline{u_k}, \quad (2)$$

где $\mu_к$ – вектор текущей разметки сети, $\overline{u_k}$ – вектор запуска переходов в сети, для которого на каждом шаге итерации k = 1,n выполняется условие:

$$\mu_{k-1} + A \cdot u_k \geq 0 \quad (3)$$

Уравнение (3) определяет правило смены разметки сетей Петри, условие (4) - правило запуска переходов сетей Петри соответственно.

Процедура построения протоколов достижимости устойчивых состояний RS- триггера включает в себя следующие этапы:

3.1. Вектору текущей разметки присваивается значение вектора начальной разметки RS- триггера для перехода τ2

$$\mu_{к-1} = \mu_{2\,0}$$

$$\mu_{к-1} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

3.2. Вычисление текущего вектора $u_{k\cdot}$

Каждый переход входящий в состав вектора **τ2** проверяем на выполнение условия (3). Все переходы удовлетворяющие условию (3) составляют вектор $u_{k\cdot}$.

Вектор **τ**2 может быть разложен на четыре составляющих вектора каждый из которых проверяется на выполнение условия (3) :

a) **[000100000000]**
b) **[000000100000]**
c) **[000000000010]**
d) **[000000000001]**

Для вектора **a** :

$$\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} =$$

$$= \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\-1\\0\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} \qquad \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} \geq 0$$

условие (3) выполняется.

Для вектора **b :**

$$\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} =$$

$$= \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\0\\0\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\1\\0\\0\end{bmatrix} \qquad \begin{bmatrix}0\\0\\0\\0\\0\\1\\0\\1\\0\\0\end{bmatrix} \geq 0$$

условие (3) выполняется.

Для вектора **c** :

$$\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\0\end{bmatrix} =$$

$$= \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\-1\\0\\1\\0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\-1\\0\\1\\1\\1\\0\end{bmatrix} \qquad \begin{bmatrix}0\\0\\0\\0\\-1\\0\\1\\1\\1\\0\end{bmatrix} < 0$$

условие (3) не выполняется.

Для вектора **d** :

$$\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\end{bmatrix} =$$

$$= \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\0\\-1\\0\\1\\0\\1\end{bmatrix} \quad <0$$

условие (3) не выполняется.

В результате u_{21} примет вид:

$$u_{21}=[000100100000]$$

3.3. Вычисление текущего вектора μ_k.

Подставляем u_{21} в уравнение (3.3). Определяем следующий вектор текущей разметки μ_k:

$$\mu_{21} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\1\\0\\0\\1\\0\\0\\0\\0\\0\end{bmatrix}$$

$$\mu_{21} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\-1\\0\\0\end{bmatrix} \qquad \mu_{21} = \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix}$$

3.4. Если вектор = μ_{f2}, то конец. Иначе τ2 =τ2 - u_k:

$$\begin{bmatrix}0\\0\\0\\1\\0\\0\\1\\0\\0\\0\\1\\1\end{bmatrix} - \begin{bmatrix}0\\0\\0\\1\\0\\0\\1\\0\\0\\0\\1\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\end{bmatrix}$$

3.5. Вычисление текущего вектора u_k

Оставшиеся переходы вектора $\tau 2$ проверяем на выполнение условия (3). Все переходы удовлетворяющие условие (3) составляют вектор u_k

a) **[000000000010]**
b) **[000000000001]**

Для векторв **a** :

$$\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix} \cdot \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\0\end{bmatrix} =$$

$$= \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\-1\\0\\1\\0\\1\\0\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\1\\0\\0\end{bmatrix} \quad \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\1\\0\\0\end{bmatrix} \geq 0$$

условие (3) выполняется.

Для вектора **b** :

$$\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix}+\begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\end{bmatrix}=$$

$$=\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix}+\begin{bmatrix}0\\0\\0\\0\\0\\-1\\0\\1\\0\\1\end{bmatrix}=\begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\1\\0\\1\end{bmatrix}\qquad\begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\1\\0\\1\end{bmatrix}\geq 0$$

условие (3) выполняется.

В результате
$u_{22} = [00000000011]$

3.6. Вычисление текущего вектора μ_k.

Подставляем u_k в уравнение (2). Определяем следующий вектор разметки μ_k:

$$\mu_{22}=\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix}+\begin{bmatrix}-1&0&0&0&0&0&0&0&0&0&0&0\\0&-1&0&0&0&0&0&0&0&0&0&0\\1&0&0&0&-1&-1&0&0&0&0&0&0\\0&1&0&0&0&0&0&-1&0&-1&0&0\\0&0&1&1&1&0&0&0&0&0&-1&0\\0&0&0&0&0&0&1&1&1&0&0&-1\\0&0&0&0&0&0&0&0&-1&-1&1&0\\0&0&0&-1&0&-1&0&0&0&0&0&1\\0&0&0&0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\end{bmatrix}$$

$$\mu_{22} = \mu_{f2}$$

$$\mu_{22} = \begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\\-1\\-1\\1\\1\\1\\1\end{bmatrix} \qquad \mu_{22} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix} \qquad \mu_{f2} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\1\\1\\1\end{bmatrix}$$

3.7. Если вектор $\mu_k = \mu_{f2}$, то конец.

Полученный протокол может быть записан следующим образом

τ_2 = [000100100011] μ_{20} = [0000000100]

u_{21} = [000100100000] μ_{21} = [0000110000]

u_{22} = [000000000011] μ_{22} = [0000001111]

μ_{f2} = [0000001111]

Аналогичным образом получены остальные протоколы.
Протоколы достижимости состояния RS-триггера:

τ_1 = [001000100011] μ_{10} = [0000000000] μ_1 = [0000001111]
u_{11} = [001000100000] μ_{11} = [0000110000]
u_{12} = [000000000011] μ_{12} = [0000001111]

τ_2 = [000100100011] μ_{20} = [0000000100] μ_2 = [0000001111]
u_{21} = [000100100000] μ_{21} = [0000110000]
u_{22} = [000000000011] μ_{22} = [0000001111]

τ_3 = [100010100011] μ_{30} = [1000000000] μ_3 = [0000001111]
u_{31} = [100000100000] μ_{31} = [0010010000]
u_{32} = [000010000001] μ_{32} = [0000100101]
u_{33} = [000000000010] μ_{33} = [0000001111]

τ_4 = [100001100001] μ_{40} = [1000000100] μ_4 = [0000000101]
u_{41} = [100000100000] μ_{41} = [0010010100]
u_{42} = [000001000001] μ_{42} = [0000000101]

τ_5 = [011000010011] μ_{50} = [0100000000] μ_5 = [0000001111]
u_{51} = [011000000000] μ_{51} = [0001100000]
u_{52} = [000000010010] μ_{52} = [0000011010]
u_{53} = [000000000001] μ_{53} = [0000001111]

τ_6 = [010100010011] μ_{60} = [0100000100] μ_6 = [0000001111]
u_{61} = [010100000000] μ_{61} = [0001100000]
u_{62} = [000000010010] μ_{62} = [0000011010]
u_{63} = [000000000001] μ_{63} = [0000001111]

τ_7 = [110010010011] μ_{70} = [1100000000] μ_7 = [0000001111]
u_{71} = [110000000000] μ_{71} = [0011000000]
u_{72} = [000010010000] μ_{72} = [0000110000]
u_{73} = [000000000011] μ_{73} = [0000001111]

τ_8 = [110001010001] μ_{80} = [1100000100] μ_8 = [0000000101]
u_{81} = [110000000000] μ_{81} = [0011000100]
u_{82} = [000001010000] μ_{82} = [0000010000]
u_{83} = [000000000001] μ_{83} = [0000000101]

τ_9 = [001000001011] μ_{90} = [0000001000] μ_9 = [0000001111]
u_{91} = [001000001000] μ_{91} = [0000110000]
u_{92} = [000000000011] μ_{92} = [0000001111]

τ_{10} = [000100001011] μ_{100} = [0000001100] μ_{10} = [0000001111]
u_{101} = [000100001000] μ_{101} = [0000110000]
u_{102} = [000000000011] μ_{102} = [0000001111]

τ_{11} = [100010001011] μ_{110} = [1000001000] μ_{11} = [0000001111]
u_{111} = [100000001000] μ_{111} = [0010010000]
u_{112} = [000010000001] μ_{112} = [0000100101]
u_{113} = [000000000010] μ_{113} = [0000001111]

τ_{12} = [100001001001] μ_{120} = [1000001100] μ_{12} = [0000000101]
u_{121} = [100000001000] μ_{121} = [0010010100]
u_{122} = [000001000001] μ_{122} = [0000000101]

τ_{13} = [011000000110] μ_{130} = [0100001000] μ_{13} = [0000001010]
u_{131} = [011000000000] μ_{131} = [0001101000]
u_{132} = [000000000110] μ_{132} = [0000001010]

τ_{14} = [010100000110] μ_{140} = [0100001100] μ_{14} = [0000001010]
u_{141} = [010100000000] μ_{141} = [0001101000]
u_{142} = [000000000110] μ_{142} = [0000001010]

τ_{15} = [110010000110] μ_{150} = [1100001000] μ_{15} = [0000001010]
u_{151} = [110000000000] μ_{151} = [0011001000]
u_{152} = [000010000100] μ_{152} = [0000100000]
u_{153} = [000000000010] μ_{153} = [0000001010]

τ_{16} = [110001000100] μ_{160} = [1100001100] μ_{16} = [0000000000]
u_{161} = [110000000000] μ_{161} = [0011001100]
u_{162} = [000001000100] μ_{162} = [0000000000]

Следует отметить, что полученные протоколы не отражают реальной длительности сигналов и задержек компонентов схемы. Время срабатывания переходов принимается

равным одному дискретному интервалу времени. Соответственно, время достижимости интерпретируется как количество итераций.

Заключение

На примере исследования асинхронного RS-триггера показано, что проблема анализа достижимости цифровых автоматов на базе их представления в виде уравнения состояния сетей Петри разрешима. Для анализа достижимости могут быть использованы стандартные процедуры и средства решения систем линейно-алгебраических уравнений.

Литература

1. Кожевников В.В. Методы математического моделирования логических схем. Ученые записки УлГУ, 2011.
2. Мурата.Т. Сети Петри. Свойства, анализ, приложения. ТИИЭР, 1989, №44.
3. Питерсон Д. Теория сетей Петри и моделирование систем. М.: Мир, 1984.

УДК 004.832.2

О ПРИМЕНИМОСТИ И ОСОБЕННОСТЯХ РЕАЛИЗАЦИИ ЭФФЕКТИВНОГО АЛГОРИТМА ОДНОВРЕМЕННОГО ПОИСКА МАКСИМАЛЬНОГО И МИНИМАЛЬНОГО ЭЛЕМЕНТОВ

А.Е.Кондратьев, О.А.Фатьянова

Ульяновский государственный университет

Потребность в одновременном поиске минимума и максимума в массиве возникает при решении целого ряда типичных задач программирования, в частности:

- для отображения множества точек в таком масштабе, чтобы они полностью занимали некоторую прямоугольную область;
- для двунаправленного варианта сортировки методом выбора [5];
- для обработки экспертных данных, когда требуется отбрасывать маргинальные оценки, например, робастная оценка [6].

Алгоритмы поиска одновременного поиска минимума и максимума давно известны и освещены в целом ряде широко известных публикаций. Эти публикации носят скорее теоретический характер, поскольку в них эффективность алгоритма трактуется как минимальное количество сравнений. Однако с точки зрения практической реализации зачастую куда более важной оказывается другая характеристика — общее время работы алгоритма, которая, в свою очередь, зависит от ряда особенностей технической реализации, в частности:

- использования различных типов данных для представления массивов в конкретном языке программирования,
- базового типа массива, то есть типа элементов массива,
- объема дополнительной памяти и так далее.

В данной работе проводится сравнительный анализ различных вариантов реализации эффективного алгоритма на языке высокого уровня C++. На основе результатов вычислительных экспериментов выдвигаются практические рекомендации для выбора того или иного варианта реализации.

1. Эффективный алгоритм одновременного поиска максимального и минимального элементов

Рассмотрим ситуацию, когда требуется найти только минимум или только максимум. В этом случае используется следующий алгоритм**:**

A1. Алгоритм, основанный на сравнении пар элементов:

```
max=x[0];
for(int i=1;i<n;i++)
  if (x[i]>max) max=x[i];
```

В [1] сформулирована и доказана следующая лемма, дающая нижнюю границу количества сравнений для алгоритма A1:

В любом алгоритме нахождения максимума среди n элементов, основанном на сравнении пар элементов, необходимо выполнить, по крайней мере, n-1 сравнений.

Если одновременно требуется найти и минимум, и максимум, то алгоритм A1 сделает это за k1=2*(n-1) сравнений. Однако, в [2] приведен алгоритм, использующий меньшее число сравнений:

k2=3n/2-2 (если n чётно) и 3(n+1)/2 (если n нечётно)

A2. Алгоритм, основанный на предварительном разбиении исходного массива на два множества (множество кандидатов в максимум и множество кандидатов в минимум):

```
пусть есть множество X={x[0],…x[n-1]}
for(int i=0,j=n-1;i<=j,i++,j--)
```

```
if (x[i]<x[j])
        {
                                                                положить x[i] в множество A;
                                                                положить x[j] в множество B;
        }
        else
        {
                                                                положить x[i] в множество B;
                                                                положить x[j] в множество A;
        }
        найти минимальный элемент в множестве A;
        найти максимальный элемент в множестве B;
```

В [2] доказано, что алгоритм А2 (который также можно найти в [3] и [4]) является оптимальным с точки зрения количества сравнений.

Запишем следующее соотношение, позволяющее оценить в процентах преимущество Алгоритма А2 перед Алгоритмом А1:

$$k\_отн=(k1-k2)/max(k1,k2)=(2n-2-3n/2-2)/max(3n/2-2,2n-2)*100\%=25\% \quad (1)$$

2.Способы реализации

2.1. Разбиение на два множества через два дополнительных массива

Реализация «в лоб» алгоритма из [2] приводит к необходимости использования дополнительной памяти для хранения двух множеств элементов (итого потребуется хранить дополнительно n элементов), что резко снижает его эффективность как с точки зрения потребления памяти, так и с точки зрения дополнительного времени, которое будет необходимо потратить на формирование данных массивов.

2.2. Разбиение на два множества в рамках одного массива

Если разбиение на два множества осуществлять в исходном массиве, то упомянутых выше затрат памяти и времени можно избежать. Тогда часть алгоритма А2, отвечающая за разбиение на два множества, примет следующий вид.

```
пусть есть множество X={x[0],…x[n-1]}
for(int i=0,j=n-1;i<=j,i++,j--)
if (x[i]>x[j]) swap(x[i],x[j]);
```

К недостаткам этого способа реализации следует отнести:

- дополнительное время, которое тратится на переформирование массива
- сам факт изменения исходного массива, так как это может быть нежелательно.

2.3. Разбиение на два множества без изменения массива

Основываясь на идеях, описанных в [3], запишем алгоритм, который сможет решать поставленную задачу без изменения входного массива:

```
if (n%2==0)
        {
        if(x[0]>x[1])
                                {
                                                                min=x[1];
                                                                max=x[0];
        }
                                else
                                 {
                                                                min=x[0];
                                                                max=x[1];
        }
                                st=2;
}
else
{
        min=x[0];
                                max=x[0];
                                st=1;
}
```

```
for(int i=st;i<n;i+=2)
{
        if (x[i]>x[i+1])
                {
                        if (x[i]>max) max=x[i];
                        if (x[i+1]<min) min=x[i+1];
                }
        else
                {
                        if (x[i]<min) min=x[i];
                        if (x[i+1]>max) max=x[i+1];
                }
}
```

3. Численное моделирование

Обозначим общее время работы Алгоритма A1 как t1, время работы алгоритма A2 как t2. Проверим, выполняется ли соотношение (1) не только для k1 и k2, но и для t1 и t2, то есть верно ли, что

$$t\_отн=(t1-t2)/max(t2,t1)=25\% \quad (2)$$

Проведем моделирование для трёх различных типов данных для представления массивов (array, valarray, vector), а также для различных базовых типов.

Таблица 1. Базовый тип int.

количество итераций	время работы, с							t_отн			
	Алгоритм A1				Алгоритм A2						
	vector	valarray	valarray STD алгоритм	array	vector	valarray	array	vector	valarray	valarray STD алгоритм	array
1000000	0.003	0.002	0.003	0.001	0.008	0.006	0.006	63%	67%	50%	83%
2000000	0.006	0.003	0.005	0.003	0.015	0.009	0.011	60%	67%	44%	73%
3000000	0.009	0.004	0.007	0.005	0.021	0.014	0.015	57%	71%	50%	67%
4000000	0.011	0.006	0.010	0.006	0.028	0.018	0.021	61%	67%	44%	71%
5000000	0.015	0.010	0.016	0.012	0.040	0.024	0.026	63%	58%	33%	54%
6000000	0.016	0.010	0.015	0.009	0.041	0.026	0.029	61%	62%	42%	69%
7000000	0.024	0.014	0.025	0.012	0.057	0.033	0.037	58%	58%	24%	68%
8000000	0.074	0.018	0.024	0.018	0.074	0.042	0.045	0%	57%	43%	60%
9000000	0.030	0.015	0.022	0.017	0.065	0.042	0.047	54%	64%	48%	64%
10000000	0.030	0.019	0.030	0.018	0.082	0.054	0.058	63%	65%	44%	69%

Результаты вычислительных экспериментов показывают, что для базового типа int соотношение (2) не выполняется: Алгоритм 1, вопреки теоретическим выкладкам, дает лучший результат, чем A2. Очевидно, это связано с тем, что алгоритм A1 допускает выполнение одной операции сразу для нескольких элементов и, таким образом, получает преимущество на современных процессорах с векторной архитектурой.

Для Алгоритма A2 подобная оптимизация вычислений невозможна. Более того, алгоритм A2 создает проблемы и при конвейеризации: только после выполнения первой операции сравнения становится понятно направление ветвления, таким образом блок предсказания переходов часто допускает ошибки.

Но это не означает, что Алгоритм A2 представляет только теоретический интерес. Для типов составных типов данных, в частности, для типа данных string были проведены дополнительные эксперименты. Длину строки в символах обозначим как m.

Таблица 2. m=2

1000000	0.065	0.066	0.055	0.063	0.054	0.055	0.048	-17%	-17%	0%	-24%
2000000	0.129	0.116	0.112	0.115	0.116	0.102	0.094	-10%	-12%	-9%	-18%
3000000	0.190	0.171	0.177	0.198	0.188	0.171	0.162	-1%	0%	-3%	-18%
4000000	0.246	0.233	0.232	0.228	0.223	0.191	0.197	-9%	-18%	-18%	-14%
5000000	0.308	0.274	0.280	0.289	0.283	0.238	0.244	-8%	-13%	-15%	-16%
6000000	0.390	0.346	0.334	0.368	0.341	0.304	0.318	-13%	-12%	-9%	-14%
7000000	0.437	0.386	0.367	0.371	0.385	0.348	0.366	-12%	-10%	-5%	-1%

8000000	0.482	0.450	0.419	0.428	0.426	0.398	0.378	-12%	-12%	-5%	-12%
9000000	0.555	0.502	0.505	0.524	0.517	0.458	0.499	-7%	-9%	-9%	-5%
10000000	0.693	0.581	0.560	0.593	0.556	0.510	0.500	-20%	-12%	-9%	-16%

Таблица 3. m=5

1000000	0.077	0.067	0.065	0.067	0.064	0.054	0.062	-17%	-19%	-17%	-7%
2000000	0.150	0.128	0.123	0.129	0.121	0.105	0.109	-19%	-18%	-15%	-16%
3000000	0.203	0.196	0.187	0.196	0.174	0.158	0.161	-14%	-19%	-16%	-18%
4000000	0.290	0.266	0.280	0.270	0.233	0.217	0.214	-20%	-18%	-23%	-21%
5000000	0.381	0.375	0.347	0.321	0.307	0.290	0.281	-19%	-23%	-16%	-12%
6000000	0.417	0.389	0.371	0.379	0.353	0.324	0.318	-15%	-17%	-13%	-16%
7000000	0.501	0.471	0.448	0.459	0.426	0.374	0.374	-15%	-21%	-17%	-19%
8000000	0.569	0.520	0.520	0.524	0.500	0.443	0.446	-12%	-15%	-15%	-15%
9000000	0.682	0.659	0.588	0.610	0.552	0.485	0.494	-19%	-26%	-18%	-19%
10000000	0.789	0.658	0.637	0.656	0.606	0.549	0.541	-23%	-17%	-14%	-18%

Результаты экспериментов для составных типов данных показывают, что для строк длины m>=2 Алгоритм A2 работает быстрее, чем Алгоритм A1. С ростом m величина t_отн быстро приближается к теоретической оценке из соотношения (1). Достичь этой оценке не удаётся в силу того, что в Алгоритме A2 не оптимизируются операции индексации, на которые также тратится время.

4. Выводы

Показано, что Алгоритм A2 может быть реализован таким образом, чтобы исключить изменение исходного массива и использование дополнительной памяти.

В случае, если базовым типом массива является простой тип (int, double и тому подобные), Алгоритм A1 обеспечивает меньшее время работы. Для составных типов предпочтительнее использовать Алгоритм A2.

В зависимости от общего времени работы типы данных для представления массива ранжируются следующим образом (при фиксированном алгоритме, от лучшего времени к худшему): array, valarray,vector. Однако эти различия не являются существенными.

Литература

1. Кнут Д. *Искусство программирования*. В 3т. Т 3. Сортировка и поиск: пер. с англ. 2-е изд. М.: Вильямс, 2007. 824 с. [The Art of Computer Programming, vol.3. Sorting and Searching].
2. Ira Pohl. *A sorting problem and its complexity*// Communications of the ACM, Volume 15 Issue 6, ACM New York, NY, USA, June 1972, — PP. 462–464. Режим доступа http://www.cs.toronto.edu/~sam/teaching/263/handouts/min-max.pdf (дата обращения 01.05.2012).
3. *Алгоритмы: построение и анализ* / Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн; под ред. И. В. Красикова. 2-е изд. М.: Вильямс, 2005. 1296 с., стр. 242.
4. Goodrich M.T., Tamassia R. Algorithm Design: Foundations, Analysis, and Internet Examples. John Wiley & Sons, 2002.
5. Роберт Седжвик Часть III. Глава 6. Элементарные методы сортировки: 6.2 Сортировка выбором // Алгоритмы на C++ = Algorithms in C++. — М.: «Вильямс», 2011. — С. 246-247. — ISBN 978-5-8459-1650-1.
6. Хампель Ф., Рончетти Э., Рауссеу П., Штаэль В. *Робастность в статистике. Подход на основе функций влияния* = Robust statistics: the approach based on influence functions. — М.: Мир, 1989.

УДК 004.416

ПРИМЕНЕНИЕ СИСТЕМ ПРИНЯТИЯ РЕШЕНИЙ В СИСТЕМАХ СОПРОВОЖДЕНИЯ ПРОГРАММНОЙ ПРОДУКЦИИ

А.Ю.Крайнов

Ульяновский государственный университет

1. Введение

Сопровождение программного обеспечения (ПО) является одним из важнейших и, в подавляющем большинстве случаев, наиболее продолжительных этапов жизненного цикла ПО. Сопровождение ПО базируется на постоянно поддерживаемой обратной связи, в ходе которой пользователь ПО сообщает разработчику о возникающих сложностях при работе с программой, необходимых изменениях функциональности и т. п. Неотъемлемой составляющей этого комплексного процесса является сбор информации о найденных ошибках. Особенно это важно на начальном этапе сопровождения, так как в условиях современного рынка ПО не всегда удаётся провести тщательные испытания продуктов до их внедрения.

Для сбора данных об ошибках могут применяться несколько подходов:

1. Применение прикладных систем отслеживания ошибок (англ. bug tracking system), таких как Atlassian JIRA, Trac или Bugzilla [1].

В этом случае разработчик некоторого продукта создаёт некоторый доступный пользователю ресурс (обычно — веб-сайт), на котором пользователи могут разместить информацию о возникающих в работе этого продукта проблемах. Главным недостатком такого варианта можно назвать то, что пользователь должен: а) знать о существовании ресурса, куда можно обратиться; б) обладать временем для регистрации заявки на этом ресурсе; в) обладать опытом работы с ПО, чтобы тщательно описать возникшую проблему.

2. Применение встроенных в программные продукты средства поддержания обратной связи [2].

В этом случае ПО содержит модуль, ответственный за контроль хода выполнения программы и отслеживание ошибочных ситуаций. При возникновении оных эти средства отправляют отчёт о возникшей проблеме на сервер разработчика, а также запрашивают у пользователя дополнительную информацию, например, порядок действий для воспроизведения проблемы. Этот подход, очевидно, не требует инициативы пользователя и более удобен для него в плане эргономичности.

Использование подобных модулей, тем не менее, не устраняет всех недостатков систем сопровождения программной продукции. Данная статья посвящена попытке описания более удобной и отвечающей современным потребностям разработчиков ПО архитектуры системы, основанной на интеграции системы сопровождения с системой управления бизнес-процессами (и, в частности, системой принятия решений).

Для определения требуемой функциональности рассмотрим некоторые ключевые недостатки существующих систем сопровождения.

2. Недостатки традиционных способов сбора описаний ошибок

1. Пользователь может дать неправильное или неполное описание ошибки.

Эта проблема имеет под собой два аспекта: а) человеческий фактор; б) недостатки пользовательского интерфейса, предоставляющего слишком большую свободу. И если человеческий фактор невозможно устранить без сложных организационных мер, то пользовательский интерфейс поддаётся исправлению. Так, более полное описание возникшей ошибки можно получить, если вместо простого текстового поля дать

пользователю возможность заполнения некоторого формуляра с фиксированным набором полей и значений.

2. Отчёты об ошибках, выраженные в текстовом виде, сложно классифицировать.

Особенность систем сопровождения ПП состоит в том, что одна и та же проблема может возникнуть на множестве рабочих станций. Соответственно, отчёты об этой ошибке от разных пользователей будут в той или иной мере повторять друг друга. Если количество пользователей продукта достаточно велико, то эта ситуация может очень сильно усложнить работу персонала, ответственного за принятие отчётов.

Методы классификации текстов позволят выявить отчёты, касающиеся одной темы или имеющие схожее содержимое. Однако обработка и классификация текстовых документов сама по себе является сложной задачей. В случае с описаниями ошибок эта задача усложняется из-за следующих особенностей таких текстов: а) небольшой объём; б) ограниченность словарного запаса; в) высокая вероятность наличия языковых ошибок. Эти проблемы частично решаются, если в системе обратной связи мы применяем формуляр с фиксированными полями: тогда наборы характеристик отчётов могут быть в некоторой мере формализованными, и к ним могут быть применены более развитые методы классификации.

Для тщательной классификации чисто текстовых запросов, тем не менее, необходима разработка специальных программных средств, основанных на технологиях обработки естественного языка и учитывающих особенности отчётов об ошибках.

3. Модуль обратной связи проектируется для каждого программного продукта отдельно.

На этапе разработки ПО на создание (а также тестирование) собственного модуля и соответствующей серверной подсистемы должны быть потрачены некоторые ресурсы, что негативно сказывается на сроках разработки и стоимости полученного продукта. С другой стороны, это приводит к тому, что на отдельной рабочей станции может быть установлено несколько программных продуктов с разными реализациями и пользовательскими интерфейсами. Такой подход сильно снижает эргономические показатели системы в целом, а в некоторых случаях и увеличивает время обучения персонала, работающего с этой системой.

Решение этой проблемы заключается в создании унифицированного модуля обратной связи, который может работать с любым программным продуктом и быть связан с единой системой сбора отчётов на сервере. Подобный модуль может быть как встроен в программный продукт в качестве библиотеки, так и работать в качестве сервиса операционной системы, обслуживающего запросы от нескольких продуктов.

4. Алгоритмы выявления ошибок и реакция на возникновение ошибки не являются перенастраиваемыми.

С течением времени бизнес-процессы сопровождения, сбора отчётов и обработки ошибок могут меняться. Если модуль обратной связи опирается на жёстко заложенные в него алгоритмы, то реализация новых процессов потребует обновления модуля. Однако такое обновление может быть нежелательно в следующих случаях: а) процессы сопровождения изменяются слишком быстро; б) мы имеем дело с сертифицированной системой, обновление которой осложнено организационными ограничениями. Решение этой проблемы заключается в разработке такой архитектуры системы сопровождения, при которой модуль обратной связи может быть легко связан с сервером бизнес-процессов на стороне разработчика.

3. Сбор данных об ошибках с помощью программного агента

Для выбора архитектурной основы системы сопровождения вновь обратимся к сравнению систем, описанных в п. 1 и рассмотрим следующие схемы.

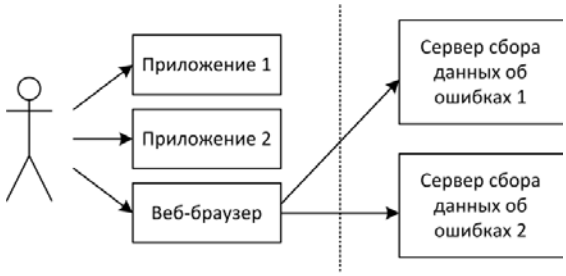


Рис. 1. Использование системы отслеживания ошибок

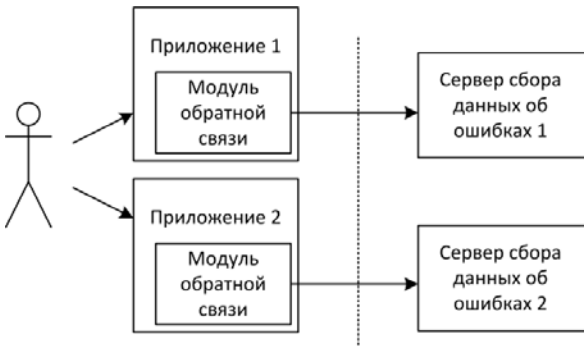


Рис. 2. Использование системы со встроенным модулем обратной связи

На рисунке 1 изображена схема работы пользователя с системами отслеживания ошибок (п. 1.1 настоящей статьи). Особенностью данного процесса взаимодействия является использование единой точки доступа (веб-браузера) к интерфейсу множества систем. Интересным преимуществом подобной структуры является также то, что на рабочей станции пользователя не устанавливается каких-либо дополнительных модулей для поддержки программных продуктов.

На рисунке 2 приведена схема работы пользователя со встроенной системой обратной связи, о которой говорилось в п. 1.2. Как мы видим, программный продукт теперь сам может контролировать создание отчётов об ошибках. Важной особенностью данного подхода является также тесная интеграция модулей обратной связи и самих продуктов.

Сочетания некоторых преимуществ первого и второго вариантов структуры системы сопровождения можно добиться, если применить подход на основе программного агента, о котором шла речь в п. 2.3 данной статьи (см. рис. 3). В этом случае специальный сервис, постоянно находящийся в памяти операционной системы, отслеживает появление ошибок в программных продуктах, отвечает за создание отчётов и отправку их на сервер. Кроме прочего, такой программный модуль может быть легко интегрирован с централизованной системой сбора ошибок.

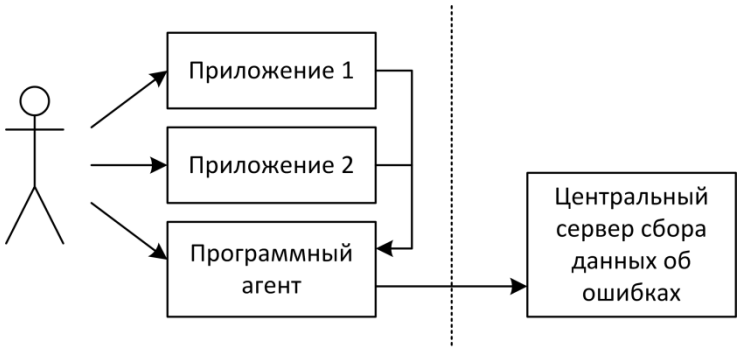


Рис. 3. Использование программного агента для сбора данных о работе ПП

Сравнение архитектурных преимуществ и недостатков трёх рассмотренных вариантов приведено в таблице 1.

Таблица 1. Сравнение архитектурных подходов к системам сопровождения ПП

	Преимущества	Недостатки
Системы отслеживания ошибок	Веб-интерфейс не потребляет дополнительных ресурсов РС Единая точка создания отчётов для всех приложений Лёгкая расширяемость системы	Требуется «активный» и «грамотный» в компьютерном плане пользователь Нет интеграции с программным продуктом
Системы со встроенными модулями обратной связи	Удобство доступа к системе сбора отчётов Интеграция с программным продуктом	При запущенном продукте модуль потребляет ресурсы РС Нет унифицированного интерфейса
Системы на основе программного агента	Единая точка создания отчётов для всех приложений Удобство доступа к системе сбора отчётов	Сервис потребляет память, даже если не запущено ни одного продукта

4. Ведение базы прецедентов ошибок

Переход к схеме с использованием программного агента, помимо архитектурных и эргономических преимуществ, может решить также проблему, указанную в п. 2.4: жесткость алгоритмов реагирования на ошибки. Рассмотрим пример процесса, который может происходить при использовании встроенного модуля обратной связи (рис. 4).

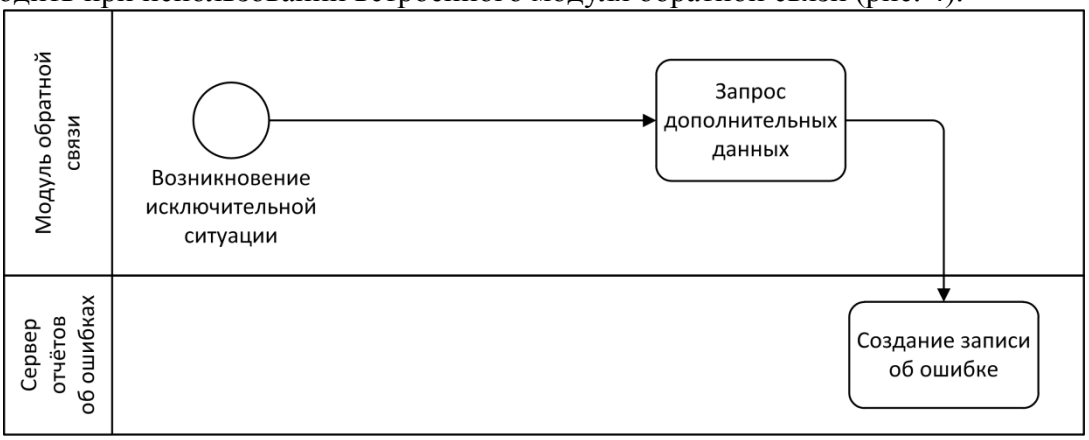


Рис. 4. Типичный бизнес-процесс сбора отчётов об ошибках

При возникновении ошибочной ситуации модуль обратной связи запрашивает дополнительное описание проблемы (например, перечень действий, которые привели к ошибке) от пользователя, после чего формирует отчёт и передаёт его на сервер. Этот алгоритм повторяется каждый раз при каждой ошибке. При этом не учитываются следующие аспекты:

1. Подобная ошибка могла уже возникнуть у других пользователей, которые уже предоставили всю необходимую информацию. Нет необходимости запрашивать дополнительные сведения у пользователя.
2. Возникшая проблема может быть уже решена. Нет необходимости запрашивать дополнительные сведения у пользователя; более того, его необходимо уведомить о способе решения этой проблемы.

Для решения этой проблемы, как уже было отмечено в п. 2.4, можно интегрировать модуль обратной связи с неким центральным сервером, на котором хранятся правила реагирования на те или иные ошибки (рис. 5). Такую схему взаимодействия можно изобразить следующим бизнес-процессом (рис. 6).

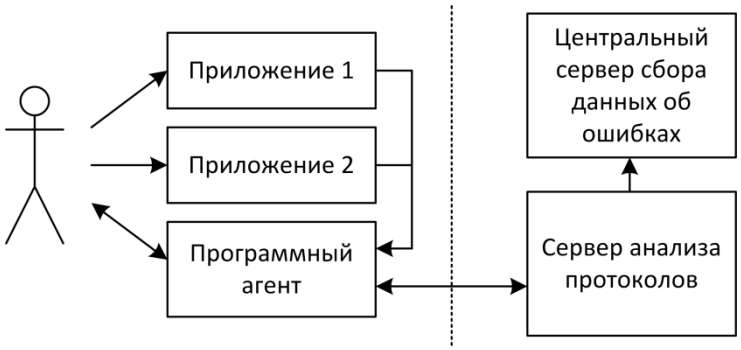


Рис. 5. Применение системы предварительного анализа протоколов

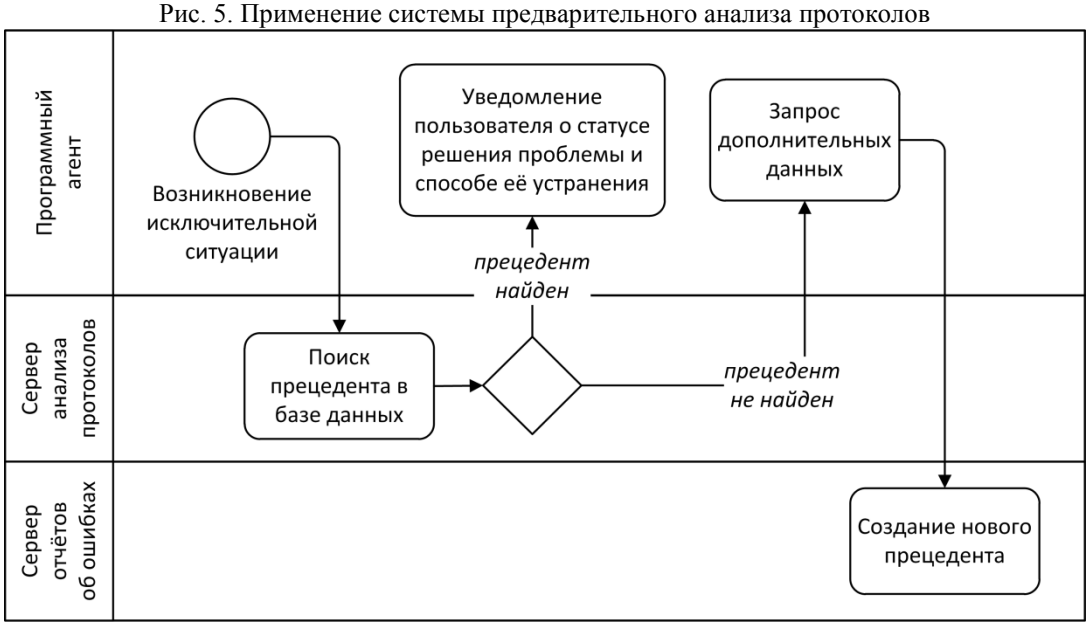


Рис. 6. Бизнес-процесс сбора отчётов об ошибках при использовании системы предварительного анализа протоколов

В этом случае при возникновении ошибочной ситуации программный агент сначала отправляет автоматический отчёт об ошибке с программным протоколом на сервер анализа протоколов. Если подобная ошибка («прецедент») уже была зарегистрирована в базе правил, то программный агент информирует пользователя о возможных способах решения проблемы. В противном случае у пользователя запрашиваются дополнительные сведения об ошибке (как это было в предыдущем процессе), после чего на их основании создаётся новый прецедент.

Ключевым элементом архитектуры системы сопровождения здесь становится сервер анализа протоколов (который представляет собой сервер бизнес-правил или систему принятия решений). В его задачи входят:

1. Ведение базы прецедентов, в которой хранятся зарегистрированные ошибки, их характерные признаки и их соответствия с ошибками, зарегистрированными на сервере отчётов об ошибках.
2. Анализ принятого от агента протокола работы программы. Протокол работы программы представляет собой текстовые данные. Текст протокола, как правило, имеет некий установленный формат, а также более формализован, чем текстовое сообщение от пользователя. Тем не менее при принятии протоколов необходимо предусмотреть применение как средств обработки естественного языка, так и интеллектуального анализа текстов (англ. *text mining*).
3. Поиск в базе прецедентов описания ошибки из принятого протокола. При поиске, помимо прочего, должна учитываться полнота текущего описания (необходимость в дополнительных данных от пользователя).
4. Применение гибких алгоритмов сбора ошибок.

5. Создание прецедента на основе принятого отчёта.

Функциональность подобного сервера частично реализована в существующих на рынке решениях, относящихся к группе EDM (англ. *enterprise decision management* [3]), таких как JBoss jBPM / Drools, CLIPS, OpenL Tables и др.

5. Отслеживание процесса исполнения программного обеспечения

Подход на основе встроенных модулей обратной связи обладает очень важным преимуществом: такие модули могут быть тесно интегрированы с программным продуктом, благодаря чему появляется возможность мониторинга процесса выполнения этого продукта. В частности, по сравнению с системами отслеживания ошибок встроенный модуль будет обладать подробной информацией о системном окружении РС (операционной системе, типе процессора и т. п.), потребляемых ресурсах (оперативной памяти, загрузке процессора), а также содержать текущий внутренний протокол ПП. Вся эта полезная информация может быть отправлена в момент формирования отчёта об ошибке на сервер сбора отчётов.

В то время как программный агент может производить мониторинг системного окружения РС, сам по себе он не может получить большей информации. В то же время для процесса, определённого в п. 4, требуется более полная информация об ошибках.

Предметная область, занимающаяся исследованием возможностей и средств наблюдения за выполнением приложений, называется «управление производительностью приложений» (англ. *application performance management*) [4]. Для сбора дополнительных данных о производительности программ в программной инженерии разработаны следующие методы:

1. Профилирование (англ. *profiling*). Это процесс сбора низкоуровневых характеристик выполняемой программы. Обычно применяется на стадии разработки ПО для выявления проблем в производительности. Существует два основных вида профилирования:
   - событийное: основано на механизмах перехвата событий, встроенных в язык программирования;
   - статистическое: основано на сканировании контекста выполнения программы через определённые интервалы времени.
2. Инструментирование (англ. *instrumentation*). Это процесс анализа производительности программы, происходящий благодаря внедрению в программный код специальных блоков, собирающих данные о процессе её выполнения.
3. Анализ протоколов (англ. *log analysis*). В узком смысле это процесс исследования записей, сформированных самим программным продуктом, в целях выявления образцов, соответствующих ошибкам в работе продукта. Может быть двух видов:
   - активный: средства анализа интегрированы с подсистемой протоколирования и, как следствие, могут работать в режиме реального времени;
   - пассивный: средства анализа производят сканирование уже записанных протоколов через определённые интервалы времени.
4. Анализ процесса выполнения (англ. *runtime intelligence*). Это комплексный процесс исследования статистики использования пользователями одного или нескольких программных продуктов. Связан с областью бизнес-аналитики и, как правило, применяется в корпоративных системах.

Не все перечисленные методы могут быть применены при разработке агента системы сопровождения. При выборе подходящих средств должны быть учтены следующие моменты:

1. Оригинальный программный продукт не должен быть изменён указанными средствами ни на этапе разработки, ни на этапе выполнения.
2. Применение указанных средств не должно приводить к нестабильной работе продукта.
3. Применение указанных средств не должно приводить к существенному падению производительности РС.

Для процесса принятия решений, описанного в п. 4., при наблюдении за сопровождаемым программным продуктом наибольшую пользу могут принести методы активного и пассивного анализа протоколов. Обе эти категории методов удовлетворяют указанным требованиям. Активный анализ протоколов, как правило, может быть произведён для тех приложений, которые были разработаны с учётом модульных принципов и используют для протоколирования стандартные средства и библиотеки. Так, например, для приложений на языке Java, использующих библиотеки SLF4J или LogBack [5] представляется возможным создание адаптера к этим подсистемам. Пассивный анализ протоколов может быть применён к практически любым приложениям. Он может быть произведён как на основе собственных средств, так и с использованием существующих программ обработки файлов протоколов (англ. *log file viewer*), таких как Chainsaw, Lilith, OtrosLogViewer или Legit.

6. Заключение

В статье была рассмотрена архитектура системы сопровождения программной продукции, интегрированная с системой предварительного анализа протоколов и принятия решений. Ожидается, что она позволит усовершенствовать процесс сбора отчётов об ошибках и существенно снизить нагрузку как на конечных пользователей, так и на администраторов системы сопровождения. Конкретные средства реализации этой архитектуры зависят от организационных требований того или иного предприятия, однако ключевые компоненты могут быть созданы на основе существующих открытых (open-source) решений.

Литература

1. Колин А. Обзор систем отслеживания ошибок // Центр компетенций Atlassian. — 2010. — URL: http://www.teamlead.ru/x/ZwDx
2. Ballmer S. Connecting with Customers // Microsoft Executive E-Mail. — 2002. — URL: http://www.microsoft.com/mscorp/execmail/2002/10-02customers.mspx
3. Dubray J.-J. Book Excerpt and Review: Smart (Enough) Systems // InfoQ. — 2007. — URL: http://www.infoq.com/articles/taylor-smart-enough-systems
4. Уайтхед Н. Мониторинг работы Java-приложений // IBM developerWorks. — 2009. — URL: http://www.ibm.com/developerworks/ru/library/j-rtm1/index.html
5. Java Logging // Habrahabr.ru. — 2011. — URL: http://habrahabr.ru/post/113145/

ОБ ОДНОМ ИЗ СПОСОБОВ ОПИСАНИЯ ФУНКЦОНИРОВАНИЯ ПРОГРАММНЫХ И ТЕХНИЧЕСКИХ КОМПЛЕКСОВ И ЕГО ИСПОЛЬЗОВАНИЕ ПРИ ОРГАНИЗАЦИИ АВТОМАТИЗИРОВАННЫХ ТРЕНИРОВОК

О.В.Краснов

Ульяновский государственный университет

Аннотация

С ростом сложности современных автоматизированных систем управления возрастают требования и к средствам обучения работе с ними. Вместе с тем, важно не только научить будущих операторов индивидуальной работе с программным комплексом, но и выработать навыки эффективного взаимодействия в команде, направленного на достижение общей цели. Большинство современных средств организации обучения и тренировок либо не учитывают этот аспект, либо узко специализированы. Это ведет к большим затрам временных и финансовых ресурсов, а также требует наличия высококвалифицированных кадров. В настоящей работе представлена модель описания функционирования программных комплексов, использование которой при построении обучающих систем, во-первых, позволит учесть необходимость отработки действий в команде, во-вторых, будет обладать достаточной универсальностью и применимостью для большого числа комплексов, в-третьих, упростит эксплуатацию систем обучения.

Ключевые слова: автоматизированное обучение, тренажеры, групповое обучение, модель.

В последнее время, в списке требований к разрабатываемым автоматизированным системам управления (АСУ) всё чаще можно увидеть требование обеспечения подготовки персонала к работе с разрабатываемой системой. Данное требование вполне объяснимо, ведь разработка АСУ, какие бы сложные задачи она не решала - не самоцель для заказчика. Ему необходимо решить поставленные задачи в некоторой предметной области при помощи предоставленного разработчиком инструмента, используя этот инструмент эффективно. То насколько эффективно используется и насколько эффективно решает поставленные задачи данный инструмент важно не только для заказчика, но и для его разработчика. Это влияет на имидж разработчика и разработанного им инструмента. Поэтому не стоит забывать про персонал, который будет использовать данный инструмент в своей работе.

1. Основные направления решения задачи подготовки пользователей АСУ

В настоящее время задача подготовки персонала к использованию АСУ решается по нескольким основным направлениям.

Во-первых, это так называемые [1] презентационные автоматизированные обучающие системы (АОС), суть которых заключается в представлении обучающей информации пользователю в форме всевозможных руководств, поставляемых с программным продуктом. Они могут иметь удобные системы поиска информации и навигации, их можно использовать как справочный материал. К недостаткам данных систем часто относят тот факт, что пользователям, у которых отсутствует опыт работы с соответствующим программным обеспечением, достаточно сложно делать первые шаги в освоении системы без наставника. Другой недостаток – отсутствие обратной связи (обучаемый находится в роли «пассивного наблюдателя, от которого не требуется никаких откликов по взаимодействию с АОС» [1]). В последнее время, разработчики начали предлагать пользователям, так называемые, интерактивные обучающие курсы, которые дают краткую вводную в программный продукт, цель контроля усвоения знаний не ставится. Другой разновидностью презентационных АОС являются различные информационные ресурсы, открытые для доступа пользователей через сеть Internet. Несмотря на то, что появляется возможность организации связи между пользователями и разработчиками АСУ и легче поддерживать в актуальном состоянии, все вышеописанные недостатки в той или иной степени сохраняются.

Во-вторых, это организация учебных теоретических курсов, совмещенных с практическими занятиями. Данный подход лишен ранее указанных недостатков, но появляются новые – затраты со стороны заказчика АСУ (как финансовые, так и временные), связанные с командированием работников в специализированные учебные центры, что в некоторых случаях не всегда удобно. С другой стороны, этот подход требует от разработчиков затрат определенных ресурсов, что также не всегда возможно.

В-третьих, это организация дистанционного обучения. Дистанционная система обучения, лишённая ранее перечисленных недостатков, сокращает затраты на обучение и может совмещать в себе все преимущества ранее перечисленных за счёт включения в себя данных подходов и расширения их дополнительными возможностями. Однако, не все производители ПО в настоящее имеют достаточные материальные, технологические и кадровые ресурсы, для развёртывания у себя подобных систем обучения.

В четвертых, это применение всевозможных тренажерных комплексов (симуляторов), которые могут поставляться заказчику АСУ. Их использование позволяет закрепить теоретические знания и отработать навыки практического использования систем управления различными процессами, аппаратами или транспортными средствами в имитируемых тренажером условиях, что дает очень высокие результаты. Тренажеры широко применяются в различных отраслях деятельности человека: медицина, авиация, космонавтика, энергетика, военное дело и т.д. Так как при разработке тренажеров требуются глубокие знания в соответствующих предметных областях, использование тренажерного комплекса, рассчитанного на какую-либо конкретную предметную область затруднительно при решении задач обучения в другой предметной области. Чаще всего тренажерный комплекс разрабатывается «с нуля» для использования в новой предметной области или при переходе на новую техническую или программную базу. Это одна из причин, по которой, разработка симуляторов требует существенных затрат временых, людских и материальных ресурсов. Причем, если подобный комплекс предназначается для отработки не только индивидуальных навыков работы но и навыков взаимодействия в команде, задача намного усложняется.

2. Описание функционирования системы с использованием графа состояний

Если рассмотреть процесс обучения с общих позиций, его можно представить как взаимодействие преподавателя и обучаемого, направленное на формирование у обучаемого багажа знаний (теоретическое обучение), навыков и умений (практическое обучение), применимых в некоторой предметной области.

Если говорить о практическом обучении, то во время изучения какой-либо автоматизированной системы немаловажную роль играет отработка навыков взаимодействия операторов с этой системой и с другими операторами, использующими ее (коллективное взаимодействие). Как правило, операторы имеют разные функциональные роли в команде (индивидуальная тренировка, в которой один обучаемый, является частным случаем). Будем называть систему, работе на которой необходимо научить оператора, целевой системой (ЦС).

Любую ЦС и её компоненты можно представить в виде графа. Вершинами (в дальнейшем, будем называть их контрольными точками) этого графа являются состояния системы или её отдельного компонента. Если две вершины графа соединены между собой направленным ребром, где первая вершина соответствует началу ребра, а вторая – концу, и система имеет состояние, описываемое первой вершиной, то возможен переход в состояние, описываемое второй вершиной, при условии что оператор предпринял соответствующее воздействие на систему. При таком подходе, успешным решением задачи тренировки является перевод системы из начального состояния в конечное по одному из нескольких путей, соединяющих начальную и конечную вершины графа состояний.

В качестве достаточно упрощенного примера, попробуем описать процесс функционирования цифрового фотоаппарата с помощью такого графа состояний (рисунок 1).

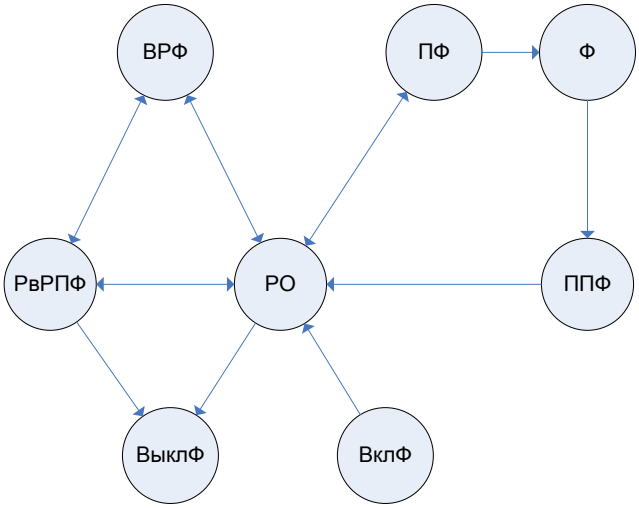


Рис. 1. Описание процесса функционирования цифрового фотоаппарата с помощью графа состояний

Граф состоит из следующих вершин:

«ВклФ» - включение фотоаппарата;

«ВыклФ» - выключение фотоаппарата;

«РО» - режим ожидания (ожидает дальнейших действий, но не находится в режиме просмотра фотографий);

«РвРПФ» - работа в режиме просмотра фотографий;

«ВРФ» - выбор режима фотосъёмки (изменение положения поворотного переключателя, задающего режим съёмки);

«ПФ» - подготовка к фотографированию (определение качества освещения, автоматическая фокусировка и т.п.);

«Ф» - фотосъёмка;

«ППФ» - предварительный просмотр сделанной фотографии.

Эти вершины соединены друг с другом с помощью направленных рёбер двух типов: однонаправленные и двунаправленные. Однонаправленные рёбра разрешают переход из вершины, соответствующей началу ребра, но запрещают прямой переход обратно. Двунаправленные рёбра, в отличие от однонаправленных, разрешают такие переходы.

Опишем изображённые переходы с помощью таблицы 1.

Таблица 1. Описание переходов графа состояний, изображённого на рисунке 1

Переход	Описание
«ВклФ» - «РО»	После включения фотоаппарат автоматически переходит в режим ожидания.
«РО» - «РвРПФ»	Из режима ожидания пользователь может перевести фотоаппарат в режим просмотра фотографий.
«РО» - «ВРФ»	Находясь в режиме ожидания, пользователь может изменить режим фотосъёмки
«РО» - «ПФ»	Непосредственно перед тем как сделать снимок, фотоаппарат выполняет некоторые подготовительные операции, такие как определение качества освещения, автоматическая фокусировка и т.п. Этим подготовительным операциям предшествует режим ожидания.
«РО» - «ВыклФ»	Находясь в режиме ожидания, фотоаппарат может быть выключен пользователем, или выключиться самостоятельно по истечении некоторого количества времени.

«РвРПФ» - «ВРФ»	Находясь в режиме просмотра фотографий пользователь может изменить режим фотосъёмки.
«РвРПФ» - «РО»	Находясь в режиме просмотра фотографий, пользователь может переключить его в основной режим ожидания.
«РвРПФ» - «ВыклФ»	Находясь в режиме просмотра фотографий, фотоаппарат может быть выключен пользователем, или выключиться самостоятельно по истечении некоторого количества времени.
«ВРФ» - «РвРПФ»	После установки режима фотосъёмки фотоаппарат возвращается в режим просмотра фотографий, если до выполнения этой операции находился в нём.
«ВРФ» - «РО»	После установки режима фотосъёмки фотоаппарат возвращается в основной режим ожидания, если до выполнения этой операции находился в нём.
«ПФ» - «РО»	После выполнения первоначальной подготовки к фотосъёмке пользователь может отказаться от выполнения съёмки, тогда фотоаппарат возвращается в основной режим ожидания.
«ПФ» - «Ф»	После выполнения первоначальной подготовки к фотосъёмке пользователь может выполнить фотосъёмку
«Ф» - «ППФ»	После выполнения фотосъёмки на экран фотоаппарата отображается сделанная фотография. Пользователь, может либо сохранить эту фотографию в память фотоаппарата, либо отказаться от её сохранения.
«ППФ» - «РО»	После завершения предварительного просмотра сделанной фотографии фотоаппарат переходит в основной режим ожидания.

Таким образом, при функционировании фотоаппарат как бы «перемещается» по графу состояний, представленному на рисунке 1. В каждый момент времени фотоаппарат «находится» в одной из вершин этого графа.

Некоторые вершины графа могут характеризоваться дополнительными атрибутами. Например, покинув вершину «ППФ» (предварительный просмотр фотографии), мы можем однозначно определить, сохранил ли пользователь сделанную фотографию. Покинув вершину «ППФ» и сохранив фотографию, мы можем прикрепить к атрибутам, характеризующим этот факт, файл, содержащий сохранённую фотографию.

Разложив по времени последовательность пройденных контрольных точек и соответствующие им конкретные значения атрибутов, зарегистрированных в них, мы можем получить так называемый след.

След – последовательность пройденных контрольных точек на графе состояний целевой системы, которым соответствуют конкретные наборы значений соответствующих им атрибутов (в том числе и файловых). Порядок следования этих точек в последовательности соответствует очерёдности их прохождения во времени.

Данные следы можно использовать в различных целях, например, для представления последовательности действий, приведших к возникновению ошибочной ситуации или поломке фотоаппарата. В нашем случае, их можно использовать для регистрации действий

пользователей при выполнении учебной задачи и значений параметров, детализирующих конкретные состояния системы.

3. Концепция построения системы автоматизации тренировочного процесса

Будем рассмотривать АСУ как совокупность автоматизированных рабочих мест (АРМ), на которых выполняются программы из состава АСУ и которые объединены в вычислительную сеть. Пусть, также, имеется АРМ руководителя обучения, которое имеет возможность подключения к указанной вычислительной сети. Применяя вышеописанный подход к описанию функционирования АСУ с использованием графа ее состояний был построен опытный образец системы проведения тренировок, удовлетворяющий следующим основным требованиям:

- кроме поддержки возможности проведения тренировок в специализированных учебных классах, должна быть предусмотрена возможность проведения тренировок непосредственно на рабочих местах операторов АСУ;
- подготовка учебной задачи с вводом необходимых исходных данных и настройка сценариев тренировок должны быть достаточно просты для неподготовленного преподавателя;
- должна быть предусмотрена возможность настройки и выдачи автоматизированных подсказок для выдачи в адрес обучаемых во время проведения тренировки;
- должна быть предусмотрена возможноть обеспечения оперативного перестроения учебной организационной структуры по количеству участников тренировки и направлениям деятельности в соответствии с решаемыми функциональными задачами каждого тренируемого лица с поддержкой автоматического выполнения действий за отсутствующих лиц;
- обеспечение мониторинга, контроля и коррекции действий обучаемых, а также документирование хода тренировки на АРМ руководителя обучения;
- наличие функций оперативного управления ходом тренировки, в том числе, функция ее приостановки (для анализа выполнения сценария тренировки, его корректировки и выдачи пояснений тренируемым лицам) при необходимости, с сохранением возможности возобновления тренировки в состоянии, соответствующем моменту приостановки тренировки.

Для организации процесса тренировок использовалась схема взаимодействия программного обеспечения АСУ и системы организации тренировок, изображенная на рисунке 2.

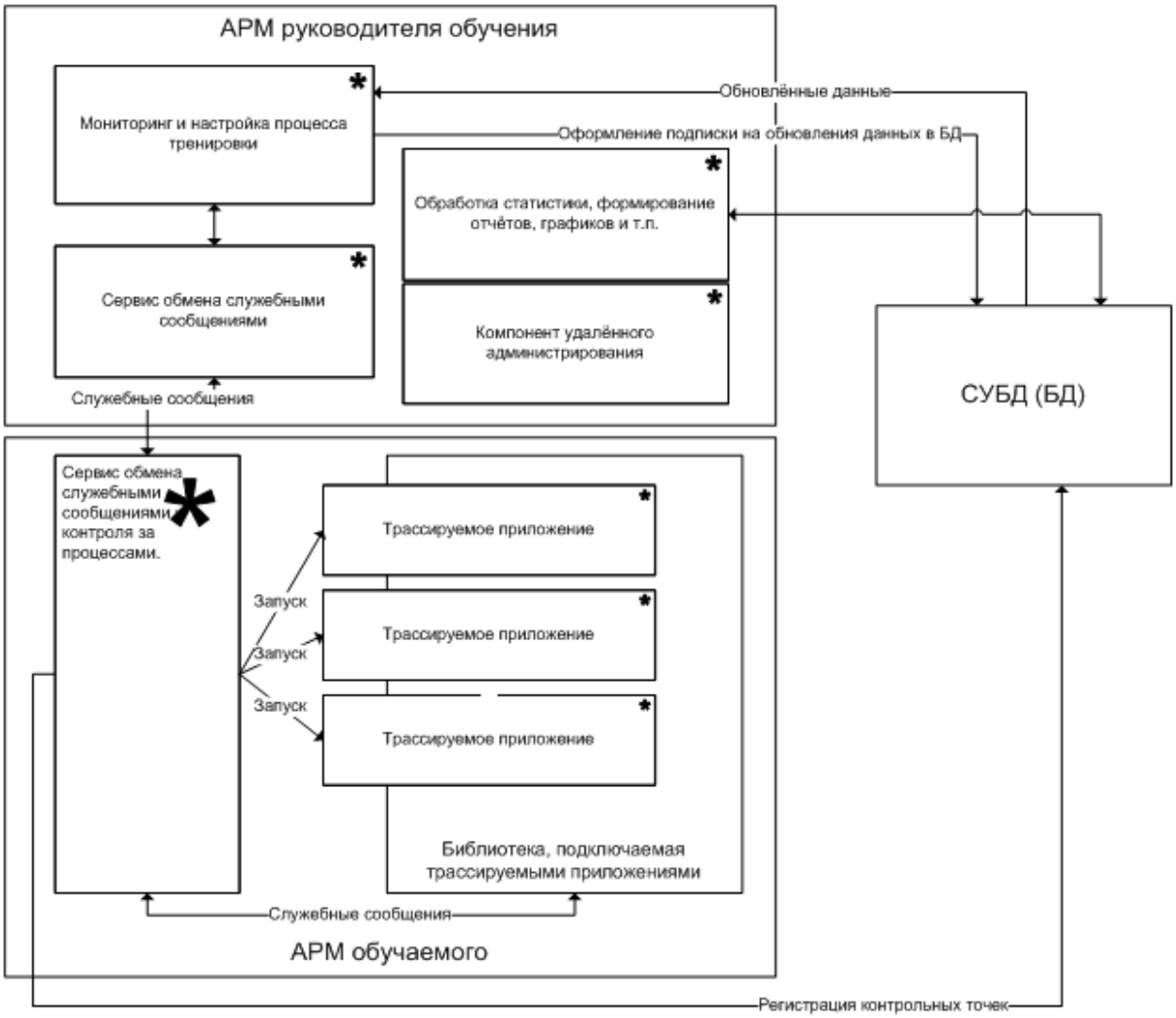


Рис. 2. Схема организации взаимодействия программного обеспечения АСУ и системы автоматизации тренировок

На данной схеме изображены приложения, функционирующие на АРМ-ах руководителя обучения и обучаемых. Под трассируемым приложением понимается приложение, входящее в состав АСУ, которое используется для решения функциональных задач на соответствующем АРМ пользователя АСУ. Данное программное обеспечение было доработано для подключения библиотеки трассировки, которая выполняет следующие функции:

- регистрация контрольных точек при решении задач выполняемых на АРМ обучаемых. Контрольные точки идентифицируются уникальным наименованием, которое отображается на АРМ руководителя тренировки при регистрации ее прохождения. Кроме наименования, контрольная точка может иметь набор атрибутов вида «наименование_атрибута» = «значение_атрибута». Например, если вернуться к примеру с фотоаппаратом, контрольная точка с идентификатором «Выбор режима фотосъемки» может иметь следующие дополнительные атрибуты: «Режим фотосъемки» = «Автоматический»; «Вспышка» = «Выключено». Также контрольная точка может содержать информацию о файлах, подготовленных при ее регистрации, например, имя файла, содержащего отснятую фотографию;

- управление приложением, подключившим данную библиотеку с целью принудительного перевода в указанную контрольную точку. Для реализации этой функции, библиотека использует механизм функций обратного вызова. При получении сигнала о необходимости перехода в необходимую контрольную точку, библиотека вызывает соответствующую функцию, указатель на которую был передан приложением при

инициализации библиотеки, с указанием атрибутов контрольной точки в которую необходимо перейти. Возможны два результата выполнения функции. Если из текущего состояния приложения возможен переход в указанную контрольную точку, функция применяет соответствующие атрибуты, переводит приложение в указанное состояние и возвращает код успешного завершения выполнения функции. Библиотека отправляет служебное сообщение в адрес АРМ руководителя об успешном переходе в указанную контрольную точку. В случае, если переход в указанную контрольную точку невозможен, функция возвращает код неудачного завершения работы функции и библиотека отправляет служебное сообщение о невозможности перехода приложения в состояние, характеризующееся указанной контрольной точкой.

Сервис обмена служебными сообщениями выполняет функции обмена короткими сообщениями руководителя тренировки и обучаемого (чат), отправку заранее настроенных автоматизированных подсказок (привязываются к конкретным этапам тренировки), отправку трассируемым приложениям указаний о приостановке/возобновлении тренировки или о необходимости принудительного перехода приложений в указанные контрольные точки при поступлении соответствующего автоматического запроса с АРМ руководителя обучения.

Собранная в базе данных (БД) информация о трассировке приложений может быть использована для построения отчётов об успеваемости обучаемых компонентом сбора и обработки статистической информации.

Компонент мониторинга и настройки процесса тренировки предназначен для включения режимов функционирования АСУ (тренировочный, штатный и технологический), создания и настройки сценариев тренировок, управления их ходом (запуск, приостановка, возобновление и полная остановка), а также отображения руководителю тренировки информации о состоянии АСУ (используются диаграммы Ганта или граф состояний), информации о пройденных контрольных точках, участниках тренировки и т.п. Следует отметить, что преподавателю нет необходимости самостоятельно настраивать характеристики графа состояний системы. Вся необходимая информация может быть собрана в технологическом режиме, когда АСУ функционирует в штатном режиме, но библиотека трассировки включена и регистрирует проходимые АСУ контрольные точки с целью выявления контрольных точек, заложенных разработчиками конкретной АСУ и переходов между ними.

Разработчикам АСУ, которые хотят включить поддержку подобного подхода требуется определиться с составом и характеристиками контрольных точек, подключить библиотеку трассировки в своих приложениях и реализовать следующие функции:

- регистрация прохождения КТ и их атрибутов (в том числе файлов, формируемых при прохождении данных КТ) за счет вызова функций, экспортируемых библиотекой трассировки;
- принудительный переход из текущей КТ в указанную (если это возможно) с применением параметров, соответствующих указанной КТ (в том числе значений файловых атрибутов), по запросу библиотеки трассировки.

Все возникающие затруднения неопытных пользователей АСУ, возникающие во время тренировки, преподаватель может решить следующими способами:

- приостановить тренировку, дать соответствующие пояснения и возобновить ее;
- воспользоваться функцией принудительного перехода в соответствующие контрольные точки (эта функция используется автоматически при отработке действий за отсутствующих лиц);
- используя компонент удаленного администрирования (используются стандартные средства подключения к удаленному рабочему столу) выполнить действия за соответствующего пользователя самостоятельно, при этом пользователь будет иметь возможность наблюдать за действиями преподавателя на рабочем столе своего АРМ.

Таким образом, алгоритм действий руководителя обучения следующий:

- формирование сценария тренировки;
- распределение ролей между обучаемыми;
- определение сценариев автоматического выполнения действий за отсутствующих должностных лиц;
- запуск тренировки;
- анализ хода тренировки;
- отправка сообщений или корректирующих воздействий на АРМ-ы обучаемых с использованием сервиса обмена служебными сообщениями (с возможностью приостановки тренировки для разъяснения каких-либо тонкостей работы в системе, после которых работу в системе можно возобновить);
- оценка успешности выполнения действий операторов АРМ.

Задача обучаемого сводится к выполнению своей роли в соответствии с заданием тренировки, используя программное обеспечение из состава АСУ. При необходимости он может связаться с руководителем тренировки для получения дополнительных инструкций, либо просмотреть автоматические подсказки, полученные на соответствующем этапе тренировки.

4. Заключение

Несмотря на свою простоту, приведенный выше способ описания процессов функционирования АСУ, положеный в основу разработанной системы, показал свою эффективность. Макет системы организации тренировок оказался достаточно универсальным и позволяет отрабатывать навыки не только индивидуального использования средств целевой АСУ, но и навыки взаимодействия в группе, каждый участник которой должен эффективно выполнять свою роль для достижения общей цели. Также система не требовательна к уровню подготовки преподавателя, что позволяет, при соблюдении вышеописанных правил разработки приложений, входящих в АСУ, использовать ее заказчиком без какого-либо участия со стороны разработчика, т.к. граф состояний системы строится не преподавателем, а в процессе эксплуатации ее в, так называемом, технологическом режиме за счет регистрации переходов из одной контрольной точки в другую. Это очень важный момент, т.к. состав контрольных точек определяется разработчиком и может изменяться при переходе от одной версии продукта к другой.

Отдельные компоненты системы могут также применяться и внекоторых побочных областях, не связанных с решением задач организации тренировочного процесса. Например, при сопровождении программной продукции данные трассировки работы приложений из состава АСУ могут использоваться для анализа причин сбоев в их работе.

Вместе с тем, модель описания процесса тренировки нуждается в дальнейшей доработке с целью преодоления основного недостатка системы – необходимость доработки программного обеспечения АСУ для подключения библиотеки трассировки. Другими словами, проработка вопроса отказа от использования библиотеки трассировки в пользу некоторой виртуальной среды, в рамках которой выполнялись бы трассируемые приложения. Такая виртуальная среда могла бы самостоятельно (при возникновении некоторых событий) идентифицировать и регистрировать изменения состояний системы, а также переводить ее в определенные состояния по требованию руководителя тренировки. Применение ее позволило бы исключить необходимость доработки программного обеспечения АСУ с целью включения в него функций регистрации и принудительного перехода в контрольные точки. В этом случае, от разработчиков АСУ не требовалось бы вообще никаких доработок разрабатываемых ими продуктов.

Литература

1. Мельников А.В., Цытович П.Л. Принципы построения обучающих систем и их классификация. http://scholar.urc.ac.ru:8002/ped_journal/numero4/pedag/tsit3.html.ru.

2. Шабаев А.И. Тяжело в учении – легко в бою. Информатизация и Системы Управления в Промышленности, 2005, №4.

3. Дозорцев В.М. Компьютерные тренажеры реального времени для обучения и переподготовки операторов и технологического персонала потенциально опасных производств. М: Приборы и системы управления, 1996, №8. С.30-31.

4. Дозорцев В.М., Шестаков Н.В. Компьютерные тренажеры для нефтехимии и нефтепереработки: опыт внедрения на российском рынке. М.: Приборы и системы управления, 1998, №1. С.27-32.

5. Дозорцев В.М. Обучение операторов технологических процессов на базе компьютерных тренажеров. М.: Приборы и системы управления, 1999, №8. С.61-70.

6. Войт Н.Н. Разработка методов и средств адаптивного управления процессом обучения в автоматизированном проектировании. Диссертация на соискание ученой степени кандидата технических наук. Ульяновск: ГОУ ВПО «Ульяновский государственный технический университет», 2009.

7. Кольцов А.С. Автоматизированные системы управления учебным процессом: учеб. Пособие / А.С. Кольцов, Е.Д. Федорков. Воронеж: ГОУ ВПО «Воронежский государственный технический университет», 2007. 179 с.

ИССЛЕДОВАНИЕ ПАКЕТА СТАТИСТИЧЕСКИХ ТЕСТОВ NIST И ИХ ПРИМЕНИМОСТИ НА ПРАКТИКЕ

М.Ю.Леонтьев

Ульяновский государственный университет

В связи с нарастающей информатизацией общества, защита информации будет продолжать играть ключевую роль в этом процессе. Криптография как часть системы защиты информации развивается и появляются новые методы шифрования данных, но неизменным остаётся то, что она использовала и продолжает использовать генераторы псевдослучайных последовательностей (далее ПСП) для своих целей. Псевдослучайные последовательности, порождаемые любым генератором для криптографических целей, подлежат обязательному тестированию. В данной работе представлено тестирование ПСП с использованием пакета статистических тестов NIST и сравнение результатов с известными алгоритмами шифрования DESи BBS.

Введение

Основным инструментом для взлома большинства алгоритмов шифрования является частотный анализ. Данный метод основывается на предположении о существовании нетривиального статистического распределения символов, а также их последовательностей одновременно и в открытом тексте, и в шифротексте. Причём данное распределение будет сохраняться с точностью до замены символов, как в процессе шифрования, так и в процессе дешифрования. В связи с этим, одной из важнейших характеристик стойкого алгоритма шифрования является его статистическая безопасность, достигаемая путем введения в криптосистему статистически безопасного генератора ПСП.

Генераторы псевдослучайных последовательностей

Генераторы ПСП являютсянеотъемлемыми элементами любой системы защиты, они используются в существующих криптосистемах для генерации ключевой информации и задания ряда параметров криптосистем.

Основной проблемой классической криптографии долгое время являлась трудность генерации секретного ключа. Физическое моделирование случайности с помощью такихфизических явлений как, например, радиоактивное излучение или дробовой шум в электронной лампе является довольно сложным и дорогостоящим, а использование нажатия клавиш и движение мыши требует усилий пользователя и к тому же не дают полностью настоящих случайных процессов. Поэтому вместо физического моделирования используют методы математического моделирования случайности и генерации случайных последовательностей в виде программ для ЭВМ или специализированных устройств.

Эти программы и устройства хотя и называются генераторами случайных чисел, на самом деле генерируют детерминированные последовательности, которые только кажутся случайными по своим свойствам и поэтому называются псевдослучайными последовательностями.От них требуется, чтобы, даже зная закон формирования, но, не зная ключа в виде заданных начальных условий, никто не смог бы отличить генерируемую последовательность от случайной, как будто она получена путем бросания идеальных игровых костей.

Генератор псевдослучайных чисел (ГПСЧ, англ. Pseudorandom number generator, PRNG) — алгоритм, генерирующий последовательность чисел, элементы которой почти независимы друг от друга и подчиняются заданному распределению (обычно равномерному).

Можно сформировать три основных требования, которым должны удовлетворять криптографическистойкие генераторы псевдослучайных последовательностей или гаммы.

1. Период гаммы должен быть достаточно большим для шифрования сообщений различной длины.

2. Гамма должна быть трудно предсказуемой. Это значит, что если известны тип генератора и кусок гаммы, то невозможно предсказать следующий за этим куском бит гаммы или предшествующий этому куску бит гаммы.

3. Генерирование гаммы не должно быть связано с большими техническими и организационными трудностями.

Отсюда следует, что одним из подходов к построению качественного генератора ПСП является преобразование задачи построения криптографически сильного генератора к задаче построения статистически безопасного генератора.

Статистические тесты NIST

Тесты NIST (англ. *Information Technology Laboratory*)— это пакет статистических тестов, разработанный Лабораторией информационных технологий, являющейся главной исследовательской организацией Национального института стандартов и технологий (NIST). В его состав входят 15 статистических тестов, целью которых является определение меры случайности двоичных последовательностей, порождённых либо аппаратными, либо программными генераторами случайных чисел. Эти тесты основаны на различных статистических свойствах, присущих только случайным последовательностям.

Все 15 тестов объединяет вычисление величины p равной вероятности того, что сгенерированнаяпсевдо последовательность менее случайна, чем реальная.

Если вычисленное в ходе теста значение вероятности $p < 0{,}01$, то данная двоичная последовательность не является истинно случайной. В противном случае, последовательность носит случайный характер. Стоит отметить, что все последующие тесты проводятся при условии, что пройдены предыдущие тесты в том порядке в котором они изложены.

В таблице 1. даны общие характеристики каждого статистического теста NIST.

Таблица 1. Статистические тесты NIST

№	Наименование теста	Определяемый дефект
1.	Частотный побитовый тест	Слишком много нулей или единиц
2.	Частотный блочный тест	Слишком много нулей или единиц в блоке m-бит
3.	Тест на последовательность одинаковых битов	Большое (малое) число последовательностей нулей и единиц свидетельствует, что колебание потока бит слишком быстрое (медленное)
4.	Тест на самую длинную последовательность единиц в блоке	Отклонение в распределении последовательности единиц
5.	Тест рангов бинарных матриц	Отклонение рангов матриц от соответствующего распределения для истинно случайной последовательности, связанное с периодичностью последовательностей
6.	Спектральный тест	Периодические свойства последовательности
7.	Тест на совпадение неперекрывающихся шаблонов	Непериодические шаблоны встречаются слишком часто
8.	Тест на совпадение перекрывающихся шаблонов	Слишком часто встречаются m-битные последовательности единиц
9.	Универсальный статистический тест Маурера	Сжимаемость (регулярность) последовательности
10	Тест на линейную сложность	Отклонение от распределения линейной сложности для конечной длины подстроки
11	Тест на периодичность	Неравномерность распределения m-битных слов
12	Тест приблизительной	Неравномерность распределения m-битных слов. Малые

	энтропии	значения означают высокую повторяемость
13	Тест кумулятивных сумм	Слишком много нулей или единиц в начале(в конце) подпоследовательности
14	Тест на произвольные отклонения	Отклонение от распределения числа появления последовательностей определенного периода
15	Другой тест на произвольные отклонения	Отклонение от распределения числа появления последовательностей определенного (большего чем в предыдущем тесте) периода

Экспериментальные исследования

За основу исследований был взят аппаратный ГПСЧ с «шумящим» элементом, так как он обеспечивает для криптографических систем очень надежный источник энтропии. В подобныхгенераторах используют оцифровку сигнала с генератора шума, основанного на тепловом, дробовом, или даже квантовых эффектах. Шумящим элементом обычно служит специальный диод или стабилитрон, сигнал с которого усиливают и подают на компаратор, формирующий двоичный битовый поток. Для того чтобы порог срабатывания компаратора не влиял на статистические свойства полученного сигнала, применяют два генератора шума, работающие на один компаратор.

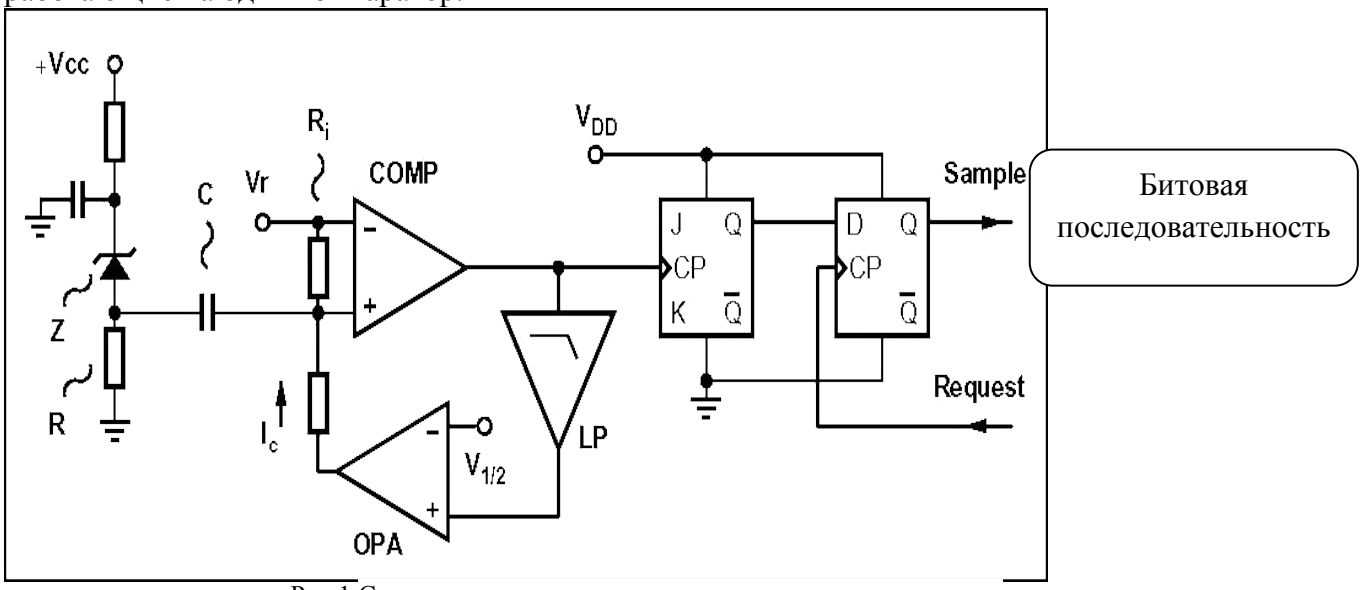


Рис.1 Схема аппаратного генератора с «шумящим» элементом

Представленная на рис. 1 схема состоит из пяти блоков:

1. Источник электрических шумов;
2. DC разделительный конденсатор C;
3. Схема для оцифровки шумового напряжения, состоящая из компаратора
4. Счетная схема (JK триггер);
5. Схема выборки, которая обеспечивает вывод последовательности по внешнему запросу.

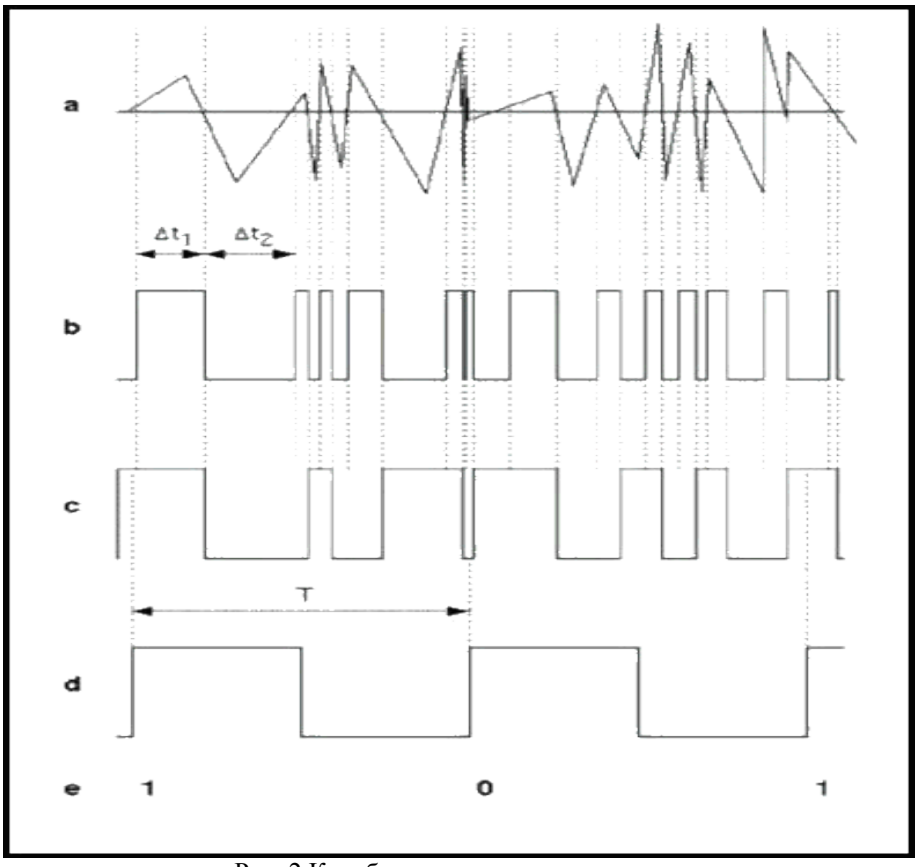


Рис. 2 Колебания напряжения на диоде

Сигналы в разных точках аппаратного генератора:

а. аналоговые шумы диода
б. выход компаратора COMP
в. выход Q в JK триггера
г. окончательный вывод генератора (триггер)

Чтобы оценить качество и статистические свойства данного генератора, проведем тестирование сгенерированных случайных последовательностей с помощью тестов NIST. Данные тесты использовались при проведении AES.

Advanced Encryption Standard, AES — конкурс, организованный NIST в 1997 году для выбора нового криптографического стандарта, который должен был стать преемником DES.

Различные статистические тесты могут применяться к ПСП для того чтобы сравнить ее с истинно случайной последовательностью. Случайность - вероятностное свойство: это означает, что свойства случайной последовательности могут быть охарактеризованы и описаны в терминах теории вероятностей. Вероятный результат статистических тестов, применяемых к истинно случайной последовательности, известен априорно и может быть описан в вероятностных терминах. Существует бесконечное число возможных статистических тестов, оценивающих присутствие или отсутствие «образца», который при обнаружении указал бы, что последовательность не случайна. Поскольку существует так много тестов, оценивающих, является ли последовательность случайной или нет, никакой определенный конечный набор тестов не считают «законченным».

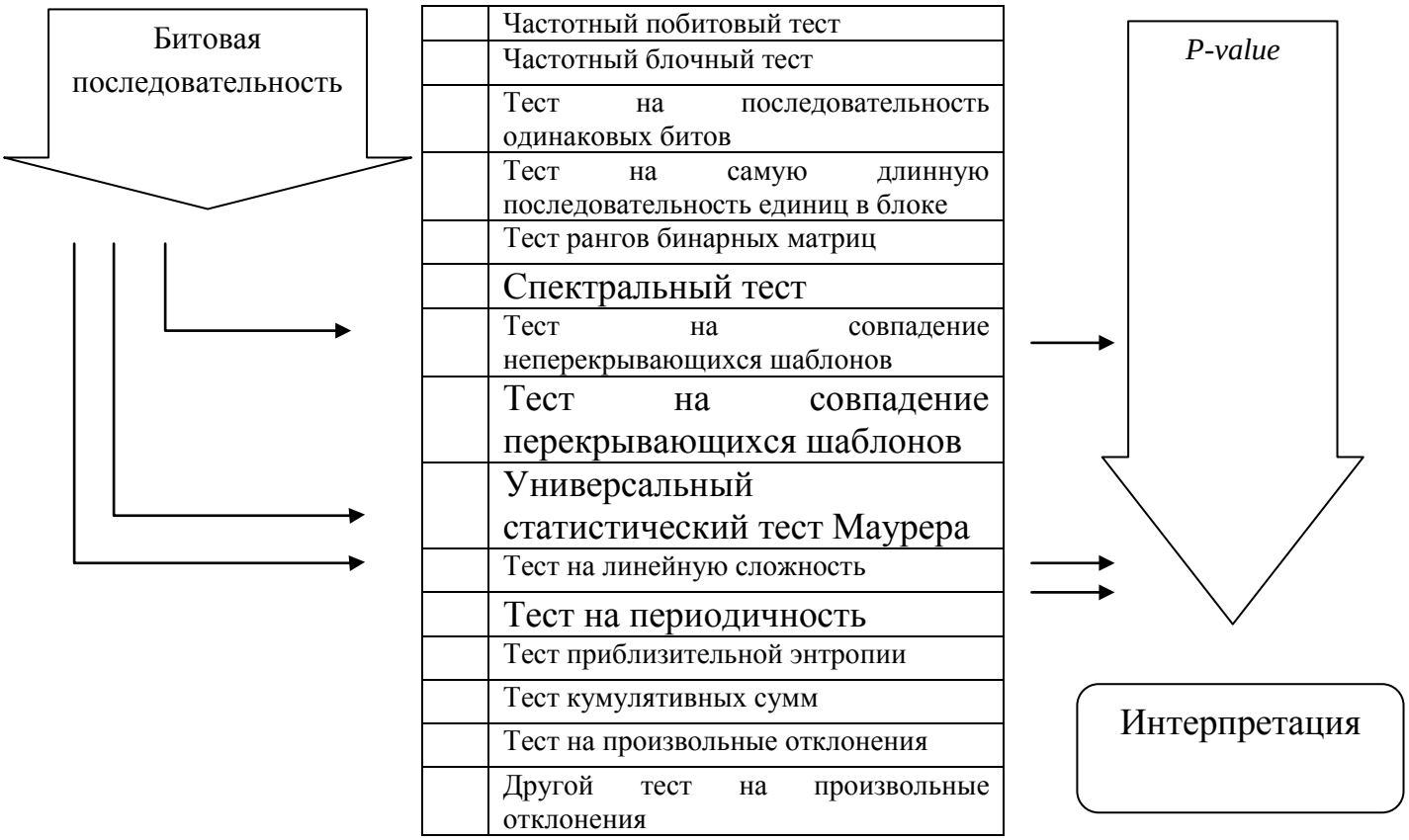


Рис.3 Схема прохождения NIST тестов

Тестовая статистика использует вычисление значения *P-value* - вероятность того, что идеальная модель генератора случайных чисел произвела бы последовательность менее случайную, чем исследуемая, для каждого конкретного теста NIST.

Если *P-value*для теста равно 1, то последовательность абсолютно случайна. *P-value,* равное 0, указывает, что последовательность абсолютно неслучайна.

При *P-value* $\geq$ 0.001 последовательность рассматривается как случайная с доверительностью 99.9%. При *P-value* < 0.001 последовательность рассматривается как неслучайная с доверительностью 99.9%.

По отношению к исследуемым последовательностям можно сделать следующие предположения.

1. Равномерность. В любой точке при генерации последовательности случайных или псевдослучайных битов 0 и 1 равновероятны и вероятности их появления равны 1/2. Ожидаемое число нулей (или единиц) равно *n/2*, где *n* – длина последовательности.
2. Масштабируемость. Любой тест, применимый к последовательности, может также применяться к произвольной подпоследовательности. Если последовательность случайна, то любая ее подпоследовательность должна также быть случайна. Следовательно, любая подпоследовательность должна пройти все тесты на случайность.
3. Полнота. Поведение генератора ПСП связано с начальным заполнением, поэтому неверно делать заключение о качестве генератора, основываясь на результатах анализа последовательности при каком-то одном начальном заполнении. Аналогично неверно делать заключение о генераторе случайных чисел, основываясь только на результатах анализа одного произведенного им фрагмента последовательности.

Итак, оценка статистических испытаний основана на проверке гипотезы о случайности исследуемой последовательности нулей и единиц. Таблица 2 показывает пошаговый процесс, позволяющий оценить конкретную двоичную последовательность.

Таблица 2. Процедура оценки

№ шага	Пошаговый процесс	Комментарии
1.	Постановка гипотезы	Предполагаем, что последовательность является случайной
2.	Вычисление тестовой статистики последовательности	Проводим тестирование на битовом уровне
3.	Вычисление P-value	P-value[0,1]
4.	Сравнение P-value с α	Задаем α, где α [0.001;0.01], если P-value >α – тесты пройдены

Для тестирования использовалось программное обеспечение Национального института стандартов и технологийniststs 2.1.1 свободное для скачивания с сайта [7]. С его помощью были получены результаты для аппаратного ГПСЧ на «шумящем» элементе, которые приведены в таблице 3.

Оценка статистических свойств последовательностей требует значительного объема аппаратных ресурсов (в первую очередь, процессорного времени), поэтому тестирование можно производить по итеративной схеме в две фазы: предварительную и основную, различающиеся наборами тестов и длиной исследуемых последовательностей.

На предварительной фазе проводился сокращенный набор тестов для выходных последовательностей небольшой длины, что позволяет на ранней стадии исключить генераторы, выходные последовательности которых обладают существенными статистическими недостатками.

Сокращенный набор тестов NIST состоит из 5 тестов.

1. Частотный побитовый тест
2. Частотный блочный тест
3. Тест на последовательность одинаковых битов
4. Тест на самую длинную последовательность единиц в блоке
5. Тест кумулятивных сумм

При успешном прохождении сокращенного набора тестов можно переходить на основную фазу. На основной фазе набор тестов выполнялся в полном объеме. Длина последовательностей и параметры тестов были выбраны в соответствии с рекомендациями NIST[7].

Для осуществления тестирования были выбраны следующие параметры:

1. Длина тестируемой последовательности $n = 10^6$ бит.
2. Количество тестируемых последовательностей $m = 100$. Таким образом, объем тестируемой выборки составил $N = 10^6 \times 100 = 10^8$ бит.
3. Количество тестов q=15.

Результаты тестирования псевдослучайных последовательностей, полученных с помощью известного алгоритма шифрования DES и генератора BBS(Blum-Blum-Shub)так же приведены для сравнения в таблице 3.

DES (Data Encryption Standard) — симметричный алгоритм шифрования, разработанный фирмой IBM и утвержденный правительством США в 1977 году как официальный стандарт, предназначенный для использования в государственных и правительственных учреждениях США.

BBS (Algorithm Blum — Blum — Shub) — это генератор псевдослучайных чисел, предложенный в 1986 году Ленор Блюм, Мануэлем Блюмом и Майклом Шубом, который имеет высокую стойкость, которая обеспечивается качеством генератора исходя из вычислительной сложности задачи факторизации чисел.

Таблица 3. Результаты тестирования

№	Статистический тест	DES	BBS	Аппаратн. генератор
1.	Частотный тест	0.2517	0.7399	0.9240
2.	Проверка кумулятивных сумм	0.0670	0.5341	0.8343
3.	Проверка «дырок» в подпоследовательностях	0.3226	0.0351	0.0456
4.	Проверка «дырок»	0.6374	0.9114	0.3041
5.	Проверка рангов матриц	0.3748	0.2133	0.7197
6.	Спектральный тест	0.1209	0.1223	0.3504
7.	Проверка непересекающихся шаблонов	0.4106	0.4431	0.0909
8.	Проверка пересекающихся шаблонов	0.8196	0.5341	0.7791
9.	Универсальный статистический тест	0.0744	0.2103	0.2022
10.	Проверка случайных отклонений	0.3212	0.6660	0.8623
11.	Разновидность проверки случайных отклонений	0.3334	0.8323	0.7727
12.	Проверка аппроксимированной энтропии	0.4528	0.4002	0.9240
13.	Проверка серий	0.8037	0.1442	0.8831
14.	Сжатие при помощи алгоритма Лемпела-Зива	0.6225	0.1223	0.2022
15	Линейная сложность	0.8659	0.4507	0.8165

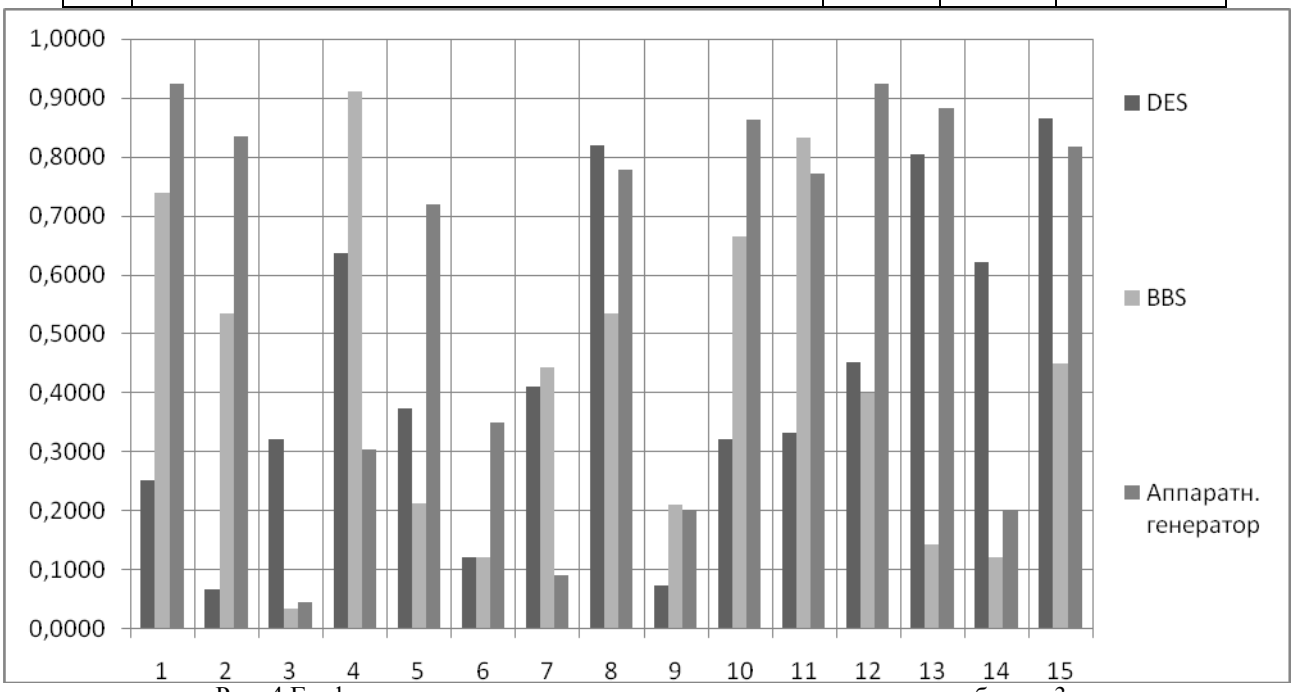


Рис. 4 Графическое представление результатов тестирования из таблицы 3.

Из графического представления таблицы 3 на рисунке 4,видно, что по большинству таких значимых параметров, как проверка серий, проверка аппроксимированной энтропии, проверка случайных отклонений и других,данный подход к построению генератора ПСП на «шумящем» элементедает лучшие результаты. Все 15 тестов имеют значение *P-value* ≥ 0.001, а значит, считаются пройденными, с доверительностью 99.9%.

Заключение

В связи с тем, что одним из основных подходов к построению качественного генератора ПСП является преобразование задачи построения криптографически сильного генератора к задаче построения статистически безопасного генератора, статистические тесты NIST можно и нужно брать на вооружение всем разработчикам генераторов случайных последовательностей для того что бы ещё на ранней стадии проектирования выявить и устранить их статистические недостатки, что позволит существенно повысить качество производства и создавать статистически безопасные генераторы ПСП.

Литература

1. Иванов М.А. Теория, применение и оценка качества генераторов псевдослучайных последовательностей / М.А. Иванов, И.В. Чугунков. – М.: КУДИЦ-ОБРАЗ, 2003. – 240 с.

2. Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях / М.А. Иванов. – М.: КУДИЦ-ОБРАЗ, 2001. – 368 с.

3. Молдовян А.А. Криптография: скоростные шифры / А.А. Молдовян, Н.А. Молдовян, Н.Д. Гуц, Б.В. Изотов. – СПб.: БХВ – Петербург, 2002. – 496 с.

4. Молдовян Н.А. Криптография: от примитивов к синтезу алгоритмов / Н.А.Молдовян, А.А. Молдовян, М.А. Еремеев. – СПб.: БХВ. – Петербург, 2004. – 448 с.

5. Алферов А.П. Основы криптографии / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин. – М.: Гелиос АРВ, 2002. – 480 с.

6. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М.: Издательство ТРИУМФ, 2003. – 816 с.

7. A Statistical Test Suite for the Validation of Random and Pseudorandom Number Generations. NISTSpecialPublication 800-22. http://csrc.nist.gov

8. Национальный институт стандартов и технологий (США) http://www.nist.gov/

ЛАБОРАТОРНЫЙ КОМПЛЕКС “БЕСЕДА”

В.А.Лукьянов, В.П.Смолеха

Ульяновский государственный университет

В настоящей статье описывается лабораторный комплекс, который планируется развернуть на кафедре ТТС для обеспечения выполнения студентами лабораторных работ согласно рабочих программ по дисциплинам ”Сети связи и системы коммутации”, “Глобальные сети”, “Администрирование сетей и сервисов” и ряда других.

Структура комплекса представлена на рисунке 1.

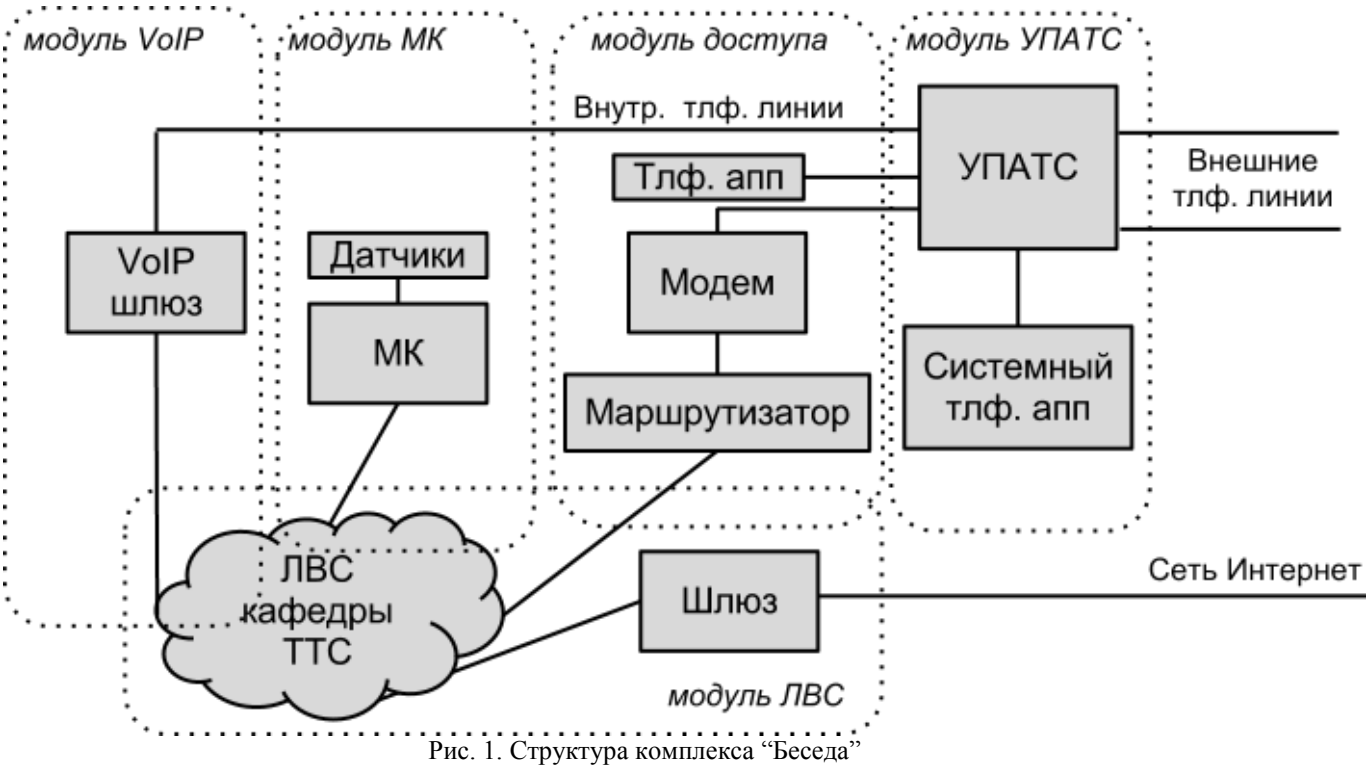


Рис. 1. Структура комплекса “Беседа”

Комплекс представляет собой комплект оборудования, подсоединённого к локальной вычислительной сети кафедры ТТС. Предлагаемое подключение оборудования позволяет изучать и практически исследовать следующие вопросы:

- программирование УПАТС;
- организация удаленного доступа к глобальной сети посредством шлюза;
- организация удаленного доступа к сети посредством модема телефонных линий;
- настройка маршрутизации в корпоративной сети;
- управление VoIP-шлюзом;
- администрирование VoIP-АТС;
- программирование микроконтроллерной системы;
- обеспечение работы элементов системы “умный дом” в составе корпоративной сети с возможностью удаленного управления.

Программирование УПАТС

Программирование УПАТС возможно осуществлять локально с системного телефона, а также по локальной сети с помощью рабочей станции ЛВС и удаленно с использованием удаленного подключения с помощью модема, используя ресурсы телефонной сети общего пользования.

Организация удаленного доступа к глобальной сети посредством шлюза

Изучение вопросов удаленного доступа посредством шлюза осуществляется на примере настройки одной из рабочих станций, работающей под управлением сетевой операционной системы.

Организация удаленного доступа к сети посредством модема телефонных линий

С использованием комплекта модемов телефонных линий становится возможным изучать проблемы, возникающие в результате осуществления передачи дискретных сообщений по каналам связи, в которых осуществляется многократное преобразование аналоговых и цифровых сигналов.

Настройка маршрутизации в корпоративной сети

Маршрутизация изучается на примере программирования операционной системы аппаратного маршрутизатора CISCO, который используется в качестве маршрутизатора удаленного доступа.

Управление VoIP-шлюзом

Использование в составе комплекса аппаратного VoIP шлюза позволяет изучать вопросы непосредственного управления шлюзом, а также обеспечить функционирование программной Vo-IP АТС.

Администрирование VoIP-АТС

Изучение вопросов администрирования VoIP-АТС осуществляется путем программирования программной VoIP-АТС, реализованной на Свободном ПО и развернутой на базе одной из рабочий станций ЛВС кафедры ТТС.

Программирование микроконтроллерной системы

Программирование микроконтроллерной системы осуществляется с использованием готовых аппаратных решений на базе микроконтроллеров AVR.

Обеспечение работы элементов системы “умный дом” в составе корпоративной сети с возможностью удаленного управления

Использованием интерфейсов взаимосвязи программаторов и ЛВС становится возможным обеспечить функционирование запрограммированных образцов микроконтроллеров в составе плат разработчика со всеми остальными элементами ЛВС. Более того, обеспечив удаленный доступ к ЛВС можно осуществить возможность удаленного управления элементами микроконтроллерной системы, тем самым реализовав на базе предлагаемого комплекса возможность изучения элементов управления системы “умный дом”.

Таким образом, у студентов появляется возможность отработать большинство вопросов администрирования корпоративных компьютерных и телекоммуникационных сетей на базе существующей лаборатории. Особенностью предлагаемого комплекса является возможность отрабатывать различные варианты администрирования в инфокоммуникационных сетях и обеспечение взаимосвязи изучаемых дисциплин, что повышает качество инженерной подготовки выпускников кафедры ТТС.

УДК 004:364-056.24

МОДЕЛИРОВАНИЕ ЦЕНТРАЛИЗОВАННОЙ СИСТЕМЫ ЕДИНОГО РЕЕСТРА ИНФОКОММУНИКАЦИОННЫХ УСЛУГ

Е.В.Лучникова, Е.Г.Чекал

Ульяновский государственный университет

На сегодняшний день существует проблема сбора и обмена между различными ведомствами информацией об исполнении индивидуальной программы реабилитации ребенка-инвалида. В процессе восстановления здоровья ребёнка с патологией участвуют реабилитационные центры, больницы и поликлиники, коррекционные школы и детские сады компенсирующего вида, общественные организации, территориальные органы социальной защиты населения и другие ведомства – всего около 200 учреждений Ульяновской области. Семье оказывают различные услуги: психолого-педагогические, медико-социальные, услуги по назначению и применению технических средств реабилитации, разрабатываются программы социальной адаптации и профессиональной ориентации, прогнозируется трудовая занятость человека с ограниченными возможностями. Необходимы управление и координация процессом оказания помощи ребёнку. Таким координирующим центром в Ульяновской области выступает областной реабилитационный центр для детей и подростков с ограниченными возможностями «Подсолнух». В настоящее время сбор сведений о состоянии ребёнка-инвалида осуществляется со слов родителей, либо по запросам в различные специализированные учреждения. Как следствие, существуют многочисленные случаи, когда ребёнку не оказывался ни один вид услуг, либо это делалось бессистемно, а как следствие не эффективно.

Кроме того, отсутствие сводного реестра всех детей с инвалидностью и оказанных им услуг не позволяет рационально распределять финансовые средства, выделяемые на эти цели, не обеспечивает преемственность и непрерывность реабилитационного процесса в системе учреждений разных ведомств. Создание автоматизированной информационной системы в рамках нашего региона позволит устранить данные недостатки. Аналогов подобных систем в Ульяновской области не существует. В Российской Федерации не проводилось исследований в области длительного накопления информации о пациенте, как о результатах изменения ограничений жизнедеятельности инвалида (в соответствии с МКФ [5]) в процессе реабилитации с момента рождения и на протяжении всей жизни и о его жизнеустройстве. Ведь идеальным результатом любой системной реабилитации предполагается независимая жизнь инвалида, самостоятельное решение им своих проблем. Поэтому проектирование данной системы актуально как с точки зрения технической (автоматизация процесса накопления и обмена информацией), так и с точки зрения социальной (создание дополнительных условий для повышения качества жизни ребёнка-инвалида). Однако, актуальность проблем предопределяет ряд сложностей. Качественного выполнения требований к автоматизированной системе учёта процесса реабилитации детей с инвалидностью можно добиться только созданием централизованной системы единого реестра инфокоммуникационных услуг (ЦС ЕРИУ), оказываемых детям с инвалидностью .

Для разработки системы необходимо произвести построение и исследование её моделей. Построение моделей ЦС ЕРИУ осуществлялось на унифицированном языке моделирования (UML) и по методологиям IDEF0, IDEF3, DFD. Разработаны следующие модели:

UML

- диаграммы классов;
- диаграммы объектов;
- диаграммы прецедентов;
- диаграммы последовательностей;
- диаграммы кооперации;
- диаграммы состояний;
- диаграммы действий;
- диаграммы компонентов.

IDEF, IDEF3, DFD

- функциональная модель (IDEF0);
- модель взаимодействия процессов (IDEF3);
- модель информационных потоков (DFD).

Построение ЦС ЕРИУ предусматривает создание базы данных, содержащей информацию о детях, об особенностях их заболеваний, о социо-культурном развитии ребёнка, о социальном паспорте семьи, об оказываемых реабилитационных программах и услугах, предоставляемых социо-защитными учреждениями, территориальными органами социальной защиты, учреждениями образования, здравоохранения и культуры Ульяновской области. В связи с отсутствием подобных систем и отлаженного межведомственного взаимодействия, одной из сложностей при проектировании стало выявление содержания будущей базы данных (см. рис.1) и определения функциональных возможностей и процессов ЦС ЕРИУ (см. рис.2).

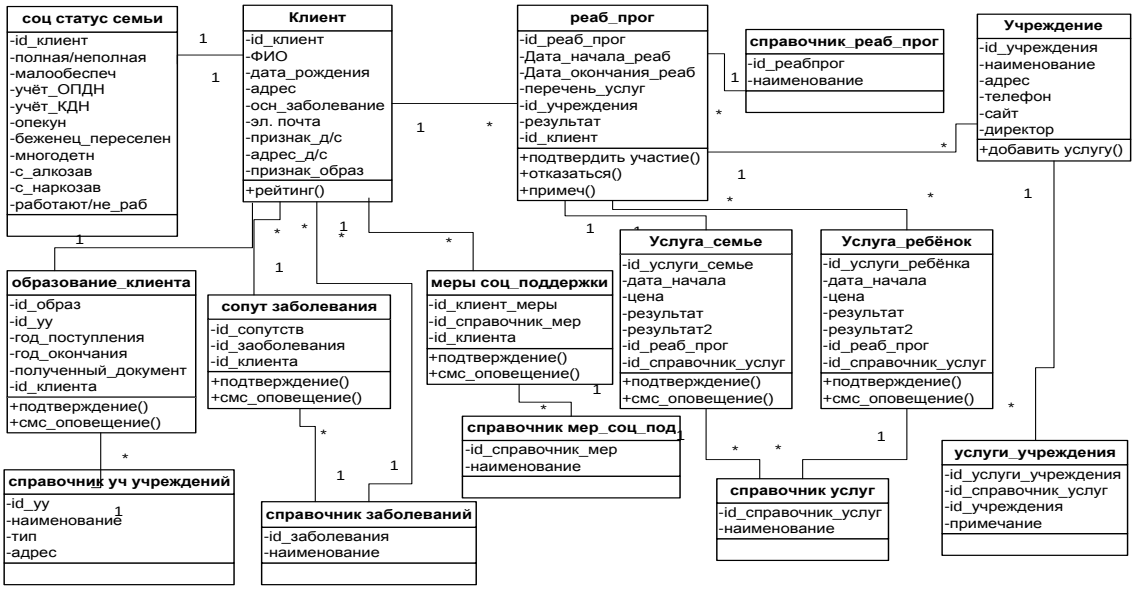


Рис. 1. Диаграмма классов.

В данной системе будут содержаться персональные данные и сведения о здоровье пациента, которые носят самый высокий уровень защиты – К1.

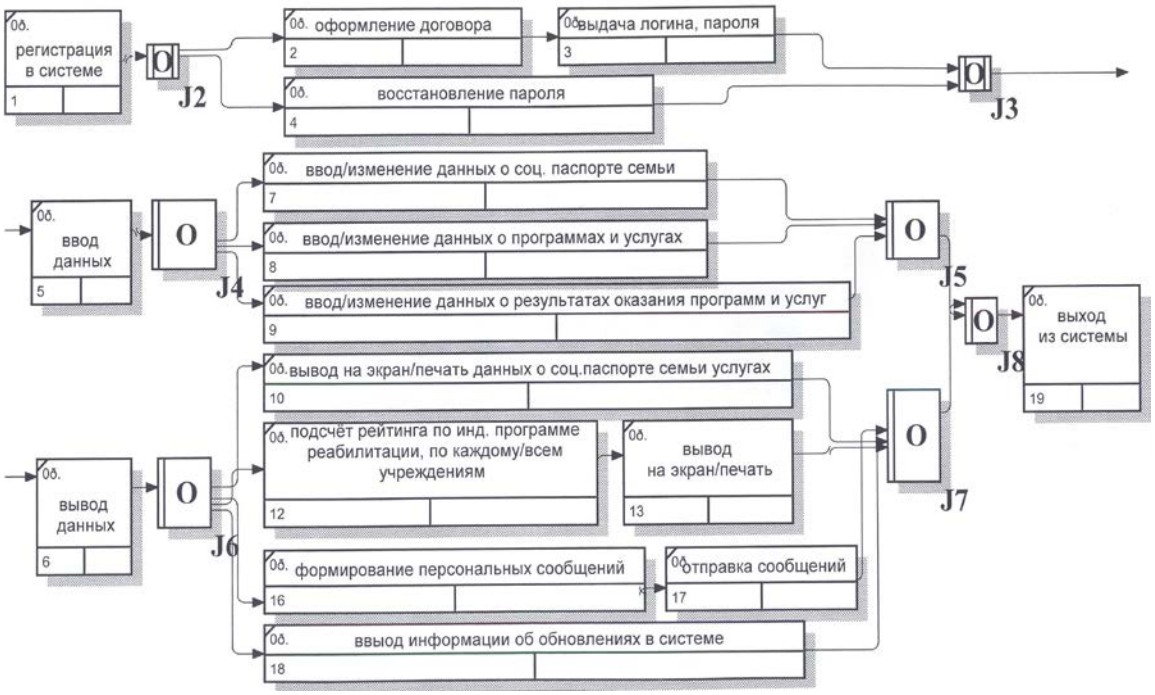


Рис. 2. Модель взаимодействия процессов.

Проведенный анализ способов защиты информации [6] показал, что наиболее целесообразным будет принятие следующей ролевой политики:

- *администратор*: имеет доступ ко всей ЦС ЕРИУ, в том числе право изменения, конфигурирования системы;
- учреждения, включенные в процесс реабилитации детей с инвалидностью:
  - *администрация учреждения* - обладают возможностью видеть отражение всех результатов всех категорий граждан, но имеют право вносить изменения только в рамках своего учреждения;
  - *специалист учреждения* — видит все данные внутри данного учреждения и статистическую информацию других учреждений. Имеет право вносить изменения только своего профиля;
  - *регистратор учреждения* — только получать статистические данные своего учреждения;
- *пациент* — видит только свой профиль, созданный на основе договора, в котором определяется соответствие идентификационного номера и ФИО клиента, а также пароль доступа к данным через веб-портал. Возможность вносить изменения в интерактивном модуле: режим «диалогового окна» с ЦС ЕРИУ, «вопрос-ответ» со специалистами учреждения, а также информирование служб о своей готовности или не готовности получать услуги в виде проставления флага в определенной графе.

Фрагмент ролевой модели представлен на диаграмме прецедентов (см. рис.3).

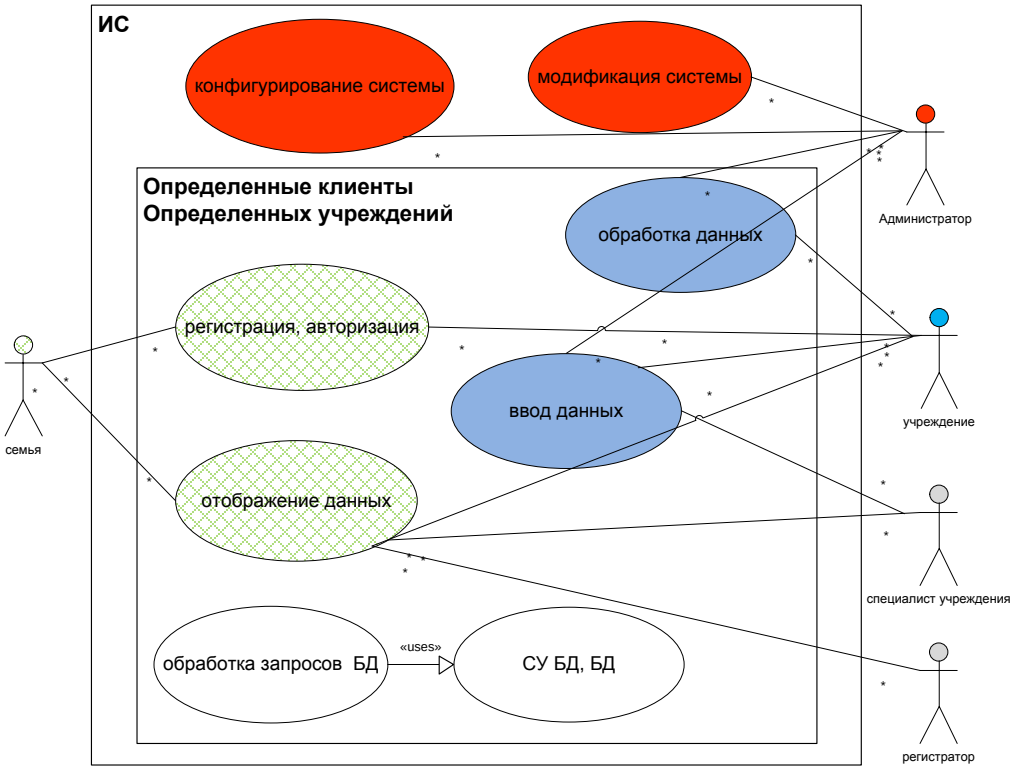


Рис. 3. Диаграмма прецедентов.

ЦС ЕРИУ будет располагаться в государственных учреждениях специального типа (реабилитационные центры). Предусматривается оснащение серверов системы следующими средствами защиты:

a) антивирусное ПО;
b) средство криптографической защиты «КриптоПро»;
c) электронный замок «Соболь».

Доступ к данным организуется через веб-портал, который позволит обеспечить сбор и обмен информацией между различными ведомствами об исполнении программы реабилитации ребенка-инвалида.

Анализ трафика и нагрузок на ЦС ЕРИУ требуют отдельного исследованияи в данной статье не рассматривается.

С внедрением единой централизованной системы родитель ребёнка сможет оперативно и полно получать информацию о доступных услугах и высказывать мнение о результатах лечения, а учреждения при взаимодействии между собой – планировать приём пациента на реабилитацию или предложить семье свои услуги.

Проект будет разработан на основе свободного программного обеспечения (AnyLogic, СУБД MySQL, CMS Joomla, Apache, PHP и др.) (см. рис.4).

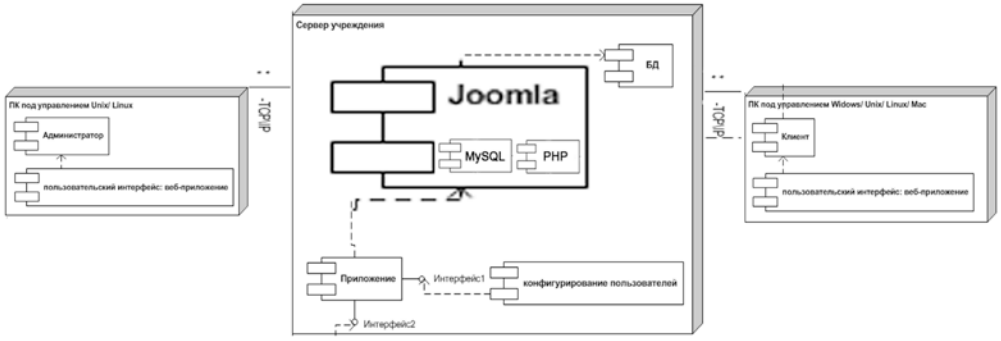


Рис. 4. Диаграмма развертывания.

Дополнительным преимуществом разрабатываемой ЦС ЕРИУ будет являться возможность оценивать эффективность реабилитационных мероприятий по единым параметрам всеми участниками реабилитационного процесса (в соответствие с ФЗ №181 от 11 июля 2011 г.).

Реализация данной ЦС ЕРИУ позволит решить ряд проблем, связанных с реабилитацией детей-инвалидов в Ульяновской области. Прототип системы можно будет использовать в других регионах России.

Литература

1. Документация ОГКУСО РЦ «Подсолнух».
2. М.Фаулер «Основы UML» Имбо.2002 г.
3. Федеральный закон РФ от 11.07.2011 № 181-ФЗ
4. http://www.cryptopro.ru/
5. Международная классификация функционирования, ограничений жизнедеятельности и здоровья (МКФ), Всемирная организация здравоохранения, 2001 г.
6. Лучникова Е.В., Коновалов С.В., Чекал Е.Г. Концепции защиты данных в системе единого реестра инфокоммуникационных услуг // сборник материалов VIII международной научно-практической конференции «Перспективы развития информационных технологий», г.Новосибирск, 2012 г. – 164 с.

МОДЕЛЬ СРЕДНЕГО И МАЛОГО ПРЕДПРИЯТИЙ, ИСПОЛЬЗУЮЩИХ КРЕДИТЫ КАК ИНВЕСТИЦИИ

А.С.Орлов

Ульяновский государственный университет

Любое управление в экономике связано с выработкой и принятием управленческих решений, воплощающихся в управляющие воздействия. В ходе поиска и анализа возможных решений, выбора предпочтительного из них, формирования управляющих воздействий субъекты управления стремятся установить, насколько им удалось отобрать лучший вариант, как в действительности "сработает" принятое решение и каковы будут его последствия. Эксперимент в экономике осуществить очень трудно, так как любая экономическая деятельность связана с людьми, а пробовать на людях разные варианты управления, проверять их последствия опасно, к тому же это дорого и отнимает много времени (в большинстве случаев субъект управления не имеет возможности затягивать принятие решений, ожидая проведения эксперимента). Поэтому для выработки управленческих решений приходится привлекать на помощь математические модели, дополняющие мысленные представления, иллюстрирующие ожидаемую картину управляемого процесса в виде цифр, кривых, графиков, таблиц.

Рассмотрим ситуацию единовременного кредитования малого предприятия, осуществляющего равномерное погашение долга с учетом начисления процентов, что сказывается на его показателях прибыли (возмещение основного долга) и себестоимости (затраты, связанные с выплатой процента).

Считаем, что предоставление единовременного кредита в момент времени $t = 0$ в размере C_0 отражается в модели путем увеличения стоимости начальных основных производственных фондов S_0 на сумму кредита C_0. По кредиту начисляются сложные проценты, непрерывным аналогом которых является функция e^{rt}. Таким образом, размер долгового обязательства $D(T)$, погашаемого к моменту t, составляет величину

$$D(t) = C_0 e^{rt}; \forall t = \overline{0,T} \qquad (1)$$

При условии равномерного погашения долга, выданного на период T, величина выплачиваемой в каждый момент t суммы долговых обязательств $L(t)$ является постоянной и рассчитывается следующим образом

$$L(t) = \frac{C_0 e^{rT}}{T} = const \ . \qquad (2)$$

Величина $L(t)$ представлена в виде суммы двух слагаемых: $\hat{H}$ - части основного долга в момент t; $\hat{h}$ - процентов, выплачиваемых в этом же периоде

$$L(t) = \frac{C_0 e^{rT}}{T} = \frac{C_0 \left(e^{rT} - 1\right) + C_0}{T} = \hat{H} + \hat{h} \ , \qquad (3)$$

где $\hat{H} = \frac{C_0}{T}$, $\hat{h} = \frac{C_0 \left(e^{rT} - 1\right)}{T}$.

Константа $\hat{H}$ уменьшает прибыль малого предприятия $P(t)$ для каждого t, а константа $\hat{h}$ - обусловливает рост себестоимости следующим образом

$$v' = v + \frac{\hat{h}}{Y(t)} \ , \qquad (4)$$

где c' - новая себестоимость.

Следовательно, величина общей прибыли $P^{об}$ изменяется таким образом, что

$$P^{об}(t) = \left[1 - v - \frac{\hat{v}}{Y(t)}\right] P(t) = (1-v)Y(t) - \hat{v}. \quad (5)$$

С учетом сделанных предположений математическая модель среднего и малого предприятия может быть представлена следующим образом [2].

$$\begin{cases} \dfrac{dF(t)}{dt} = u\left[P(t) - \hat{H}\right] \\ Y(t) = \alpha F(t), \\ P(t) = P^{об}(t) - N(t) \\ F' = F_0 + C_0 \\ P^{об}(t) = (1-v)Y(t) - \hat{h} \\ N(t) = \beta_1 Y(t) + \beta_2 \hat{\kappa}(1-u)P(t) \end{cases} \quad (6)$$

где u - доля чистой прибыли, отчисляемой на реинвестирование;

$P^{об}(t)$ – общая прибыль среднего и малого предприятия;

$P(t)$ – чистая прибыль среднего и малого предприятия;

$P(t)$ – сумма налоговых отчислений;

$Y(t)$ – выпуск продукции в момент времени t в стоимостном выражении;

β_1, β_2 – ставки налогообложения на объем выпуска и прибыль соответственно;

$\hat{\kappa}$ - доля реинвестируемых средств прибыли;

α – показатель фондоотдачи;

v – себестоимость продукции;

$F(t)$ – стоимость основных производственных фондов.

Определим динамику основных производственных фондов. Для этого решим нелинейное дифференциальное уравнение:

$$\frac{dF}{dt} = \tilde{a}[F(t)]^{\alpha} + C(t) \text{ , при } C(t) = 0.$$

Данное уравнение является однородным уравнением Бернулли. Уравнение Бернулли может быть сведено к линейному уравнению, путем следующих преобразований.

Обозначим $\frac{dF}{dt} = F'(t)$, тогда можно записать

$$F'(t) = \tilde{a}[F(t)]^{\alpha} . \quad (7)$$

Разделим обе части уравнения на $[F(t)]^{\alpha}$, тогда $[F(t)]^{-\alpha} \cdot F'(t) = \tilde{a}$.

Так как $\left([F(t)]^{1-\alpha}\right)' = (1-\alpha)[F(t)]^{-\alpha} \cdot F'(t)$, то $(1-\alpha)[F(t)]^{-\alpha} \cdot F'(t) = (1-\alpha)\tilde{a}$

$$\left([F(t)]^{1-\alpha}\right)' = (1-\alpha)\tilde{a} .$$

Таким образом, получено линейное дифференциальное уравнение с неизвестной функцией $\left([F(t)]^{1-\alpha}\right)$. Такое уравнение имеет следующее решение

$$[F(t)]^{1-\alpha} = \frac{1}{\eta(t)}\left(\int \eta(t)(1-\alpha)\tilde{a}\,dt + L\right), \quad (8)$$

где $\eta(t) = \exp\left\{\int 0 \cdot (1-\alpha)dt\right\} = 1$.

Следовательно $[F(t)]^{1-\alpha} = \int (1-\alpha)\tilde{a}\,dt + L$, $[F(t)]^{1-\alpha} = (1-\alpha)\tilde{a}t + L$,

$$F(t) = \left((1-\alpha)\tilde{a}t + L\right)^{\frac{1}{1-\alpha}} . \quad (9)$$

Постоянную L - найдем из условия $F(0)=F_0$ при $t=0$.

$$L=[K_0]^{1-\alpha} \ . \qquad (10)$$

Анализ соотношения (9) свидетельствует, что темп роста системы в значительной степени определяется показателем $\tilde{a}$, зависящим главным образом от внутреннего экономического механизма малого предприятия; тем не менее, соотношение констант, определяющих условия кредитования и формирующих сомножитель экспоненты, может существенно повлиять на динамику его основных производственных фондов.

Аналогично исследуется схема равномерного погашения кредитной задолженности с наличием процентов в дискретном времени. Тогда процентные платежи рассчитываются следующим образом

$$M=C_0 r+\left(C_0-\frac{C_0}{T}\right)r+\left(C_0-\frac{2C_0}{T}\right)r+....+\left(C_0-\frac{(T-1)C_0}{T}\right)r=C_0 r\frac{T+1}{2T} \ .$$

Платеж в дискретный момент t, как и ранее, состоит из погашения основного долга и процентов

$$Y(t)=\frac{C_0}{T}+\frac{C_0 r(T+1)}{2T}=\hat{H}+\hat{h}, \qquad (11)$$

где $\hat{H}=\frac{C_0}{T}$, $\hat{h}=\frac{C_0 r(T+1)}{2T}$. При этом $\int_0^T \hat{H}dt=\frac{C_0}{T}\int_0^T dt=C_0$ - основной долг, $\int_0^T \hat{h}dt=\int_0^T \frac{C_0 r(T+1)}{2T}dt=\frac{C_0 r(T+1)}{2}$ - начисленный процент.

Литература

1. Бассовский Л.Е. Прогнозирование и планирование в условиях рынка. Учеб. пособие. – М.: ИНФРА-М, 2007.
2. Егорова Н.Е., Маренный М.А. Малые предприятия: предпринимательские стратегии и кооперация. - М.: Спутник +, 2004.
3. Морозов В. К., Рогачев Г. Н. Моделирование информационных и динамических систем. – М.: Академия. – 2011.
4. Снетков Н. Н. Имитационное моделирование экономических процессов. - М.: Изд. центр ЕАОИ. - 2008.

КОМПОЗИЦИОННЫЕ ШИФРЫ

П.В.Потапов

Ульяновский государственный университет

1. Введение. Основные понятия. Классификация шифров.

Криптография - это область знаний, относящаяся к средствам и методам преобразования сообщений в непонятную для посторонних форму, а также к средствам и методам проверки подлинности сообщений. Эта область является классическим примером соревнования «брони и снаряда»- средств защиты и нападения. Долгое время криптография (наука о шифрах) была засекречена, так как применялась, в основном, для защиты государственных и военных секретов. В настоящее время методы и средства криптографии используются для обеспечения информационной безопасности не только государства, но и частных лиц.

Как передать нужную информацию нужному адресату в тайне от других? Каждый из вас в разное время и с разными целями наверняка пытался решить для себя эту задачу. Есть три возможных пути решения данной проблемы:

1. Создать абсолютно надежный, недоступный для других канал связи между абонентами.
2. Использовать общедоступный канал связи, но скрыть сам факт передачи информации.
3. Использовать общедоступный канал связи, но передавать по нему нужную информацию в преобразованном виде, так чтобы восстановить ее мог только адресат.

Проанализируем эти три пути решения:

1. При современном уровне развития науки и техники сделать такой канал связи между удаленными абонентами для неоднократной передачи больших объемов информации практически нереально.
2. Разработкой средств и методов скрытия факта передачи сообщения занимается стеганография.
3. Разработкой методов преобразования (шифрования) информации с целью ее защиты от незаконных пользователей занимается криптография.

Такие методы и способы преобразования информации называются шифрами.

Определение. Шифрование – процесс применения шифра к защищаемой информации, т.е. преобразование открытого текста в шифрованное сообщение (шифртекст, криптограмму) с помощью определенных правил (ключей), содержащихся в шифре.

Чтобы дать определение шифрам необходимо вести в рассмотрение ряд понятий. Пусть X - множество возможных открытых текстов; S – множество шифрованных текстов (криптограмм); K- множество ключей.

Определение. Шифр – это совокупность инъективных отображений множества открытых текстов во множество шифрованных текстов, проиндексированная элементами из множества ключей:

$$\{F_k : X \to S, k \in K\}.$$

Определение. Расшифрование – это процесс восстановления открытого текста по шифрованному при известном ключе.

Определение. Дешифрирование – это процесс восстановления открытого текста по шифрованному при неизвестном ключе.

Для того чтобы зашифровать сообщение $X \in X$, нужно выбрать ключ $k \in K$ и применить к X отображение F_k. Получившийся результат $F_k(X) = S$ является результатом шифрования текста X на ключе *K*.

Все шифры различают, прежде всего, на шифры замены, шифры перестановки, композиционные шифры. На рисунке 1 представлена классификация шифров.

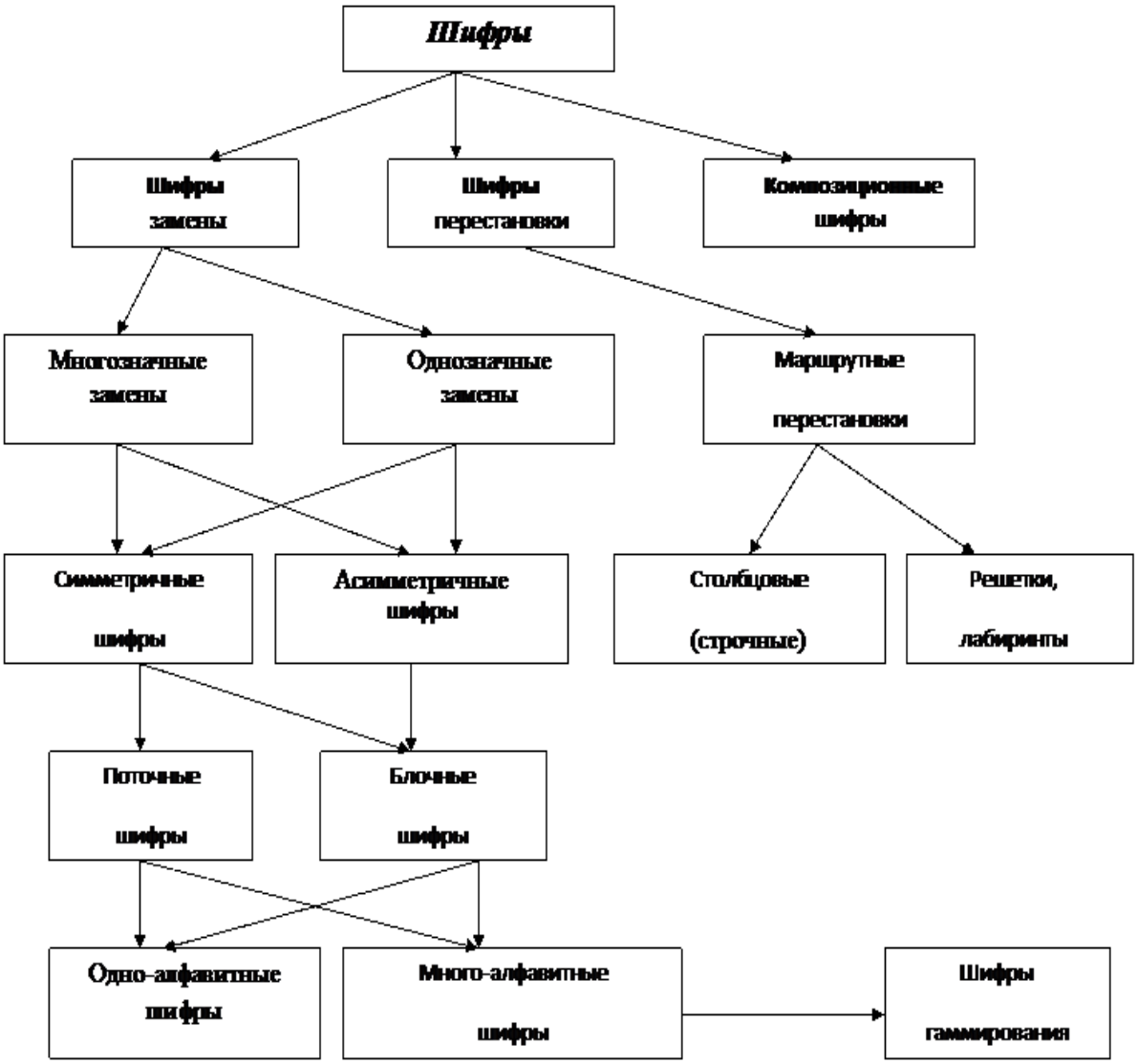


Рисунок 1 – Классификация шифров

Определение. Шифром замены – называется шифр, в котором фрагменты открытого текста заменяются некоторыми их эквивалентами в шифротексте. Если ключ шифрования совпадает с ключом расшифрования, то такие шифры называют *симметричными*, если же ключи различны, то такие шифры называют *асимметричными*. Рассмотрим некоторые примеры шифров замены. Пусть шифруемые сообщения состоят из символов алфавита А, включающего буквы русского языка (без буквы ё) и знака пробела. Имеется таблица, задающая соответствие между символами этого алфавита и произвольным порядком букв:

а	б	в	г	д	е	ж	з	и	к	л	м	н	о	п	р
г	л	ь	п	д	р	а	м	ц	в	э	ъ	х	о	б	н

с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
с	ж	я	и	ю	к	щ	ф	е	у	ы	ч	ш	т	з

Такой шифр называется шифром простой однобуквенной замены. По ключу удобно проводить зашифрование и расшифрование: при зашифровании каждая буква открытого текста заменяется на соответствующую букву из второй строки (а на г и т.д.) При расшифровании, наоборот, г заменяется на а и т.д. При шифровании и расшифровании надо помнить вторую строчку, то есть ключ. Запомнить произвольный порядок букв алфавита достаточно сложно. Поэтому всегда пытались придумать какое-либо правило, по которому можно просто восстановить вторую строчку. Одним из первых шифров, известных из истории, был так называемый *шифр Цезаря*, для которого вторая строка является последовательностью, записанной в алфавитном порядке, но начинающейся не с буквы а:

а б в г …… ю я
г д е ж…….б в

Другим примером шифра замены может служить лозунговый шифр. Здесь запоминание ключевой последовательности основано на лозунге - легко запоминаемом слове. Например, выберем слово-лозунг «белка» и заполним вторую строку таблицы по следующему правилу: сначала выписываем слово-лозунг, а затем выписываем в алфавитном порядке буквы алфавита, не вошедшие в слово-лозунг.

Рассмотрим еще один пример шифра однозначной замены- шифр гаммирования, получивший наибольшее практическое применение. Пусть имеется сообщение $t_1...t_n$, представляющее собой последовательность слов из алфавитаA, состоящего из 32 букв русского алфавита (буквы е и ё не различаются). КлючK представляет собой последовательность чисел $k_{11}...k_n$, из множества чисел M={0…32}. Зашифрованный текст $s_1...s_n$, вычисляется по следующей формуле:

$$S_i = (C(t_i) + k_i) \bmod 33, i = 1...n,$$

где C: $A \to M$ функция, отображающая символ в его порядковый номер. Запись $d=(a+b) \bmod m$ означает, что dсовпадает с остатком от деления на mсуммы чисел aиb. (например 1=(4+7) mod 10).

Расшифровать сообщение можно с помощью формулы:

$$t_i = C^{-1}((S_i + (33 - k_i)) \bmod 33).$$

В данном случае через C^{-1} обозначается функция, отображающая число из множества от 0 до 32 в символ (букву или пробел) с соответствующим номером.

Будем предполагать, что ключ выбирается случайно равновероятно из M^n – множества векторов длины n, координаты которых выбираются из M (или что то же самое, каждый из элементов ключа k_i, $i=1...n$, выбирается случайно равновероятно и независимо из M). Определенный таким образом шифр называется шифром *гаммирования* со случайной равновероятной гаммой (гаммой принято называть последовательность чисел $k_1...k_n$, складываемую по модулю с шифруемым сообщением).

При таком методе шифрования количество символов в криптограмме совпадает с количеством символов сообщения, то есть совпадают их длины. Чтобы скрыть от противника длину сообщения, его можно дополнить пробелами до некоторой фиксированной длины, которую не превосходит стандартное сообщение. В приведенном ниже примере мы будем дополнять сообщения до 10 символов.

Пример1: зашифруем с помощью описанного выше шифра гаммирования сообщение «НАСТУПАТЬ» на ключе «05 12 03 29 24 12 30 14 11 31». Для этого заменим буквы нашего сообщения их порядковыми номерами в алфавите и сложим эти номера с элементами ключевой последовательности по модулю 33:
13 00 17 18 19 15 00 18 28 32 – шифруемое сообщение в цифровом виде
05 12 03 29 24 12 30 14 11 31 – ключ
18 12 20 14 10 27 30 32 06 30 – результат шифрования.

Криптоаналитик противника, зная алгоритм шифрования и имея зашифрованный текст, но, не имея ключа, не сможет извлечь для себя никакой другой информации, кроме самого факта передачи сообщения (факт передачи сообщения можно замаскировать передавая с определенной частотой «пустые» сообщения). Делая разные предположения с ключе, криптоаналитик будет получать разные открытые тексты.

Рассмотренный метод шифрования можно было бы считать идеальным, если бы ни один существенный недостаток – длина ключа при гаммировании случайной равновероятной гаммой совпадает с длинной шифруемого текста. Конечно, современные технические средства позволяют обеспечить хранение большого объема ключевых данных, используя для этого, например, лазерные диски. Однако во многих случаях реализация данного метода будет дорогой и неудобной. Для того чтобы упростить данный алгоритм шифрования, перейдем к алфавиту из 32 символов, отбросив пробел из нашего исходного алфавита *M*.

Вместо пробела в сообщениях будем использовать одну из редко встречающихся букв, например «Ф». Будем предполагать, что ключ $K=(Y_1\ Y_2\ Y_3)$ состоит из трех чисел, выбранных случайно равновероятно и независимо из множества {0…31}. С помощью рекуррентного соотношения $Y_t=(Y_{t-1}+Y_{t-3})\bmod 32$ порождаются элементы рекуррентной последовательности $Y_1 \ldots Y_{n+1}$ для t>3. Затем по формуле:

$$Z_i=(Y_t+Y_{t+1})\bmod 32, t=1\ldots n \qquad (*)$$

вычисляется псевдослучайная последовательность $z_1 \ldots z_n$, используемая вместо случайной гаммы. Как и в предыдущем примере, шифрование заключается в сложении по модулю 32 элементов гаммы с порядковыми номерами символов шифруемого текста.

Пример 2: зашифруем на ключе *K*=(04,31,15) сообщение «ПРИКАЗЫВАЮ НАСТУПАТЬ»

Рекуррентная последовательность $Y_1 \ldots Y_n+1$ в данном случае будет иметь вид «04 31 15 19 18 01 20 06 07 27 01 08 03 04 12 15 19 31 14 01 00». Сложим по модулю 32 порядковые номера символов нашего сообщения с элементами псевдослучайной последовательности (гаммы), полученной по формуле (*):

15 16 08 10 00 07 27 02 00 30 20 13 00 17 18 19 15 00 18 28 – сообщение
03 14 02 05 19 21 26 13 02 28 09 11 07 16 27 02 18 13 15 01 – гамма
18 30 10 15 19 28 21 15 02 26 29 24 07 01 13 21 01 13 01 29 - зашифрованное сообщение.

Определение. Шифром перестановки – называется шифр, преобразования из которого изменяют только порядок следования символов исходного текста, но не изменяют их самих (буквы открытого текста при шифровании лишь меняются местами друг с другом). Ключом шифра является перестановка номеров букв открытого текста. Рассмотрим преобразование предназначенное для зашифрования сообщения длиной *n* символов. Его можно представить с помощью таблицы:

$$\begin{matrix} 1 & 2 & \cdots & n \\ i_1 & i_2 & & i_n \end{matrix},$$

где i_1 - номер места шифртекста, на которое попадает первая буква исходного сообщения при выбранном преобразовании и т.д. В верхней строке таблицы выписаны по порядку числа от 1 до *n,* а в нижней те же числа, но в произвольном порядке. Такая таблица называется подстановкой степени *n.*

Пример3: если для преобразования используется подстановка:

1. 2 3 4 5 6
4. 2 3 5 1 6

и в соответствии с ней зашифровывается слово МОСКВА, то получится КОСВМА или УЧЕНИК получится НЧЕИУК.

Число различных преобразований шифра перестановки, предназначенного для зашифрования текста длиной *n*, меньше либо равно n!.

При шифровании очень длинных сообщений такой шифр не рационален, так как приходиться работать с длинными таблицами. Широкое распространение получили шифры перестановки, использующие некоторую геометрическую фигуру. Преобразования из этого шифра состоят в том, что в фигуру исходный текст вписывается по ходу одного «маршрута», а затем по ходу другого выписывается из нее. Такой шифр называется маршрутной перестановкой.

Пример 4: зашифруем фразу «основыкриптографии», используя прямоугольную таблицу размером 3 на 6:

о	с	н	о	в	ы
о	т	п	и	р	к
г	р	а	ф	и	и

Выписываем сообщение по маршруту: по вертикали, начиная с верхнего правого угла и двигаясь поочередно сверху вниз и снизу вверх: «ыкиирвоифапнстргоо».

Широко распространена разновидность шифра маршрутной перестановки, называемая *шифром вертикальной перестановки*. В нем используется прямоугольник, в который сообщение вписывается обычным способом (по строкам слева направо). Выписывается шифрограмма по вертикали а столбцы при этом берутся в порядке, определяемом ключом.

Пример 5: на ключе К (5 1 4 2 6 3) зашифровать сообщение: «маршрутныеперестановки» при помощи прямоугольника 4 на 6.

5	**1**	**4**	**2**	**6**	**3**
м	а	р	ш	р	у
т	н	ы	е	п	е
р	е	с	т	а	н
о	в	к	и		

Таким образом получили шифрограмму - «аневшетиуенрыскмтрорпа». При расшифровании, в первую очередь надо определить число длинных столбцов, то есть число букв в последней строке прямоугольника. Для этого нужно разделить число букв в сообщении на длину числового ключа. Тогда остаток от деления и будет искомым числом. В нашем примере: 22=3*6+4, поэтому в заполненной таблице имеется 4 длинных и 2 коротких столбца.

Одной из разновидностей шифров простой замены являются *блочные шифры*. Простейший блочный шифр оперирует с биграммнымишифровеличинами. Одним из первых таких шифров был биграммный шифр Порта и Плейфера. Рассмотрим описание шифра Плейфера, широко распространенного в начале нашего века. Основой шифра Плейфера является прямоугольная таблица, размером 5x6 (5 строк, 6 столбцов), в которую записан систематически перемешанный алфавит (без букв ё, Й, ь). В первую строку вписывается ключевое слово, затем перечисляются все буквы алфавита по порядку за исключением тех, что встречаются в ключевом слове. Правило зашифрования состоит в следующем:

1. Буквы биграммы (i,j), $i \neq j$ находятся в данной таблице. При шифровании биграмма (i,j) заменяется биграммой (k,l), где k и l определяются в соответствии с правилами 2-4.

2. Если i и j не лежат в одной строке или одном столбце, то их позиции образуют противоположные вершины прямоугольника. Тогда k и i – другая пара вершин, причем k вершина, лежащая в том же столбце, что и i.

3. Если i и j лежат в одной строке, то k и i – буквы той же строки, расположенные непосредственно справа от i и j соответственно. При этом, если одна из букв – последняя в строчке, то считается, что ее «правым соседом» является первая буква той же строки.

4. Если i и j лежат в одном столбце, то k и i– буквы того же столбца, расположенные непосредственно снизу от i и j соответственно. При этом, если одна из букв – последняя в столбце, то она заменяется на первую букву того же столбца.

При шифровании открытый текст представляется в виде последовательности биграмм. Если текст имеет нечетную длину или содержит биграмму, состоящую из одинаковых букв, то в него добавляются «пустышки» таким образом. «Пустышкой» является некоторая редкая для данного типа текста буква, которая вставляется между одинаковыми буквами биграммы или добавляется в текст для того, чтобы его длинна стала четной. Такие изменения открытого текста не мешают при расшифровании.

Композиционным шифром – называются всевозможные композиции различных шифров. То есть, с целью повышения надежности шифрования шифрованный текст, полученный применением некоторого шифра, может быть еще раз зашифрован с помощью другого шифра.

2. Композиционные шифры. Объединение блочных шифров. Варианты объединения

Известно множество путей объединения блочных алгоритмов для получения новых алгоритмов. Создание подобных схем стимулируется желанием повысить безопасность, избежав трудности проектирования нового алгоритма. Так, алгоритм DES относится к надежным алгоритмам, он подвергался криптоанализу добрых 20 лет и, тем не менее, наилучшим способом взлома остается лобовое вскрытие. Однако ключ DES слишком короток. Разве не плохо было бы использовать DES в качестве компонента другого алгоритма с более длинным ключом? Это позволило бы воспользоваться преимуществами обоих систем: устойчивостью, гарантированной двумя десятилетиям криптоанализа, и длинным ключом.

Один из методов объединения - многократное шифрование. В этом случае для шифрования одного и того же блока открытого текста алгоритм шифрования используется несколько раз с несколькими ключами. Каскадное шифрование подобно многократному шифрованию, но использует различные алгоритмы. Известны и другие методы.

Повторное шифрование блока открытого текста одним и тем же ключом с помощью того же или другого алгоритма неэффективно. Повторное использование того же алгоритма не повышает сложность лобового вскрытия. (Мы предполагаем, что криптоаналитику известны алгоритм и число операций шифрования). При использовании различных алгоритмов сложность лобового вскрытия может, как возрастать, так и оставаться неизменной. При этом нужно убедиться в том, что ключи для последовательныхшифрований различны и независимы.

2.1. Двойное шифрование

К наивным способам повышения надежности алгоритма относится шифрование блока дважды с двумя различными ключами. Сначала блок зашифровывается первым ключом, а получившийся шифртекст - вторым ключом. Расшифрование выполняется в обратном порядке.

$$C = E_{K1}(E_{k2}(P))$$
$$P = D_{K1}(D_{K1}(C))$$

Если блочный алгоритм образует группу, всегда существует такой *K3,* для которого:

$$C = E_{K2}(E_{K1}(P)) = E_{K3}(P)$$

Если алгоритм не образует группу, взломать итоговый дважды зашифрованный блок шифртекста с помощью полного перебора намного сложнее. Вместо 2^n (где n – длинаключа в

битах), потребуется 2^{2n} попыток. Если алгоритм использует 64-битовый ключ, для обнаружения ключей, которыми дважды зашифрован шифртекст, понадобится 2^{128} попыток.

Однако при атаке с известным открытым текстом это не так. Меркл иХеллман предложили способ согласования памяти и времени, который позволяет вскрыть такую схему двойного шифрования за 2^{n+1}шифрований, а не за 2^{2n}. (Они использовали эту схему против DES, но результаты можно обобщить на все блочные алгоритмы). Такая атака называется «встреча посередине»: с одной стороны выполняется зашифрование, а с другой - расшифрование, а полученные посередине результаты сравниваются.

В этой атаке криптоаналитику известны значения P_1, C_1, P_2и C_2, такие что:

$$C_1=E_{K2}(E_{K1}(P_1))$$
$$C_2=E_{K2}(E_{K1}(P_2))$$

Для каждого возможного K криптоаналитик рассчитывает $E_K(P_1)$ и сохраняет результат в памяти. Собрав все результаты, он для каждогоKвычисляет $D_K(C_1)$ и ищет в памяти такой же результат. Если такой результат обнаружен, то, возможно, что текущий ключ – K_2, а ключ для результата в памяти – K_1. Затем криптоаналитик зашифровывает P_2ключами K_1и K_2. Если он получает C_2, он может почти быть убежденным (с вероятностью 1 к 2^{2m-2n}, где m - размер блока), что он восстановил и K_1и K_2. В противном случае он продолжает поиск. Максимальное количество попыток шифрования, которое ему придется предпринять, составляет $2*2^n$, т.е. 2^{n+1}. Если вероятность ошибки слишком велика, криптоаналитик может использовать третий блокшифртекста, обеспечивая вероятность успеха 1 к 2^{3m-2n}. Существуют и другие способы оптимизации.

Для такого вскрытия нужен большой объем памяти: 2^n блоков. Для 56-битового ключа нужно хранить 2^{56} 64-битовых блоков, или 10^{17} байт. Такой объем памяти пока еще трудно себе представить, но этого хватает, чтобы убедить самых осторожных криптографов в ненадежности двойного шифрования.

При 128-битовом ключе для хранения промежуточных результатов потребуется огромная память в 10^{39} байт. Если предположить, что каждый бит информации хранится на единственном атоме алюминия, запоминающее устройство, нужное для такого вскрытия, будет представлять собой алюминиевый куб с ребром 1 км. Кроме того, понадобится куда-то его поставить. Так что атака «встреча посередине» при ключах такого размера представляется невозможной.

Другой способ двойного шифрования, который иногда называют методом Дэвиса-Прайса (Davies-Price), представляет собой вариант режима шифрования CBC.

$$C_i=E_{k1}(P_i\oplus E_{k2}(C_{i-1}))$$
$$P_i=D_{k1}(C_i)\oplus E_{k2}(C_{i-1})$$

Утверждается, что «у этого режима нет никаких особых достоинств», к тому же он, по-видимому, столь же уязвим к атаке «встреча посередине», как и другие режимы двойного шифрования.

2.2. Тройное шифрование

2.2.1. *Тройное шифрование с двумя ключами*

В более удачном методе, предложенном Тачменом, блок обрабатывается три раза с использованием двух ключей: первым ключом, вторым ключом и снова первым ключом. Тачмен предлагает, чтобы отправитель сначала зашифровал сообщение первым ключом, затем расшифровал вторым, и окончательно зашифровал первым ключом. Получатель расшифровывает сообщение первым ключом, затем зашифровывает вторым и, наконец, расшифровывает первым.

$$C = E_{K1}(D_{K2}(E_{K1}(P)))$$
$$P = D_{K1}(E_{K1}(D_{K1}(C)))$$

Иногда такой режим называют режимом зашифрование-расшифрование-зашифрование (Encrypt-Decrypt-Encrypt - EDE). Если блочный алгоритм использует n-битовый ключ, длина ключа описанной схемы составляет $2n$ бит. Эта остроумная связка ключей (зашифрования-

расшифрования-зашифрования) разработана в корпорации IBM для совместимости с существующими реализациями алгоритма: задание двух одинаковых ключей эквивалентно одинарному шифрованию. Такая схема EDE сама по себе не обеспечивает заведомую безопасность, однако этот режим использовался для улучшения алгоритма DES в стандартах X9.17 и ISO 8732.

Для предотвращения описанной выше атаки «встреча посередине», использование ключей *K1* и *K2* чередуется. Если $C=E_{K2}(E_{K1}(E_{K1}(P)))$, то криптоаналитик может заранее вычислить $E_{K1}(E_{K1}(P)))$ для любого возможного *K1*, а затем выполнить вскрытие. Для этого потребуется только 2^{n+2} шифрований.

Тройное шифрование с двумя ключами устойчиво к такой атаке. Но Меркл и Хеллман разработали другой способ согласования памяти и времени, который позволяет взломать этот алгоритм шифрования за 2^{n-1} действий, используя 2^n блоков памяти.

Для каждого возможного *K2* расшифровывают 0 и сохраняют результат в памяти. Затем расшифровывают 0 для каждого возможного *K1*, чтобы получить *P*. Выполняют тройное зашифрование *P*, чтобы получить *C*, и затем расшифровывают *C* ключом *K1*. Если полученное значение совпадает со значением (хранящимся в памяти), полученным при расшифровании 0 ключом *K2*, то, возможно, пара *K1K2* и будет искомым результатом. Проверяют, так ли это. Если нет, продолжают поиск.

Для выполнения этого вскрытия с подобранным открытым текстом нужна память огромного объема. Понадобится 2^n времени и памяти, а также 2^m подобранных открытых текстов. Атака не слишком практична, но все же указывает на некоторую слабость этого метода.

Пауль ван Оорсчот (Paul van Oorschot) и Майкл Винер (Michael Wiener) преобразовали эту атаку к атаке на основе открытых текстов, для которой их нужно *p* штук. В примере предполагается использование режима EDE.

1) Предположить первое промежуточное значение *a*.

2) Используя известный открытый текст, свести в таблицу для каждого возможного *K1* второе промежуточное значение *b* при первом промежуточном значении, равном *a:*

$$b=D_{K1}(C)$$

где *C* - шифртекст, полученный по известному открытому тексту.

3) Для каждого возможного *K2* найти в таблице элементы с совпадающим вторым промежуточным значением *b:*

$$b = E_{K2}(a)$$

4) Вероятность успеха равна p/m, где *p* - число известных открытых текстов, а *m* -размер блока. Если совпадения не обнаружены, нужно выбрать другое значение *a* и начать сначала.

Атака требует $2^{n+m}/p$ времени и *p* - памяти. Для алгоритма DES это составляет 2^{120}/p. При *p,* больших 256, эта атака выполняется быстрее, чем полный перебор.

2.2.2. *Тройное шифрование с тремя ключами*

Если используют тройное шифрование, рекомендуется использовать три разных ключа. Общая длина ключа станет больше, но хранение ключа обычно не вызывает затруднений. Биты дешевы.

$$C = E_{K3}(D_{K2}(E_{K1}(P)))$$
$$P = D_{K1}(E_{K2}(D_{K3}(C)))$$

Для наилучшего вскрытия с согласованием памяти и времени, примером которого служит «встреча посередине», понадобятся 2^{2n} операций и 2^n блоков памяти. Тройное шифрование с тремя независимыми ключами настолько надежно, насколько на первый взгляд кажется надежным двойное шифрование.

2.2.3. *Тройное шифрование с минимальным ключом*

Известен более надежный метод использования тройного шифрования с двумя ключами, препятствующий описанной атаке и называемый тройным шифрованием с

минимальным ключом (TripleEncryptionwithMinimumKey - TEMK). Фокус в том, чтобы получить три ключа из двух: *X1* и *X2*.

$$K_1=E_{X1}(D_{X2}(E_{X1}(T_1)))$$
$$K_2 = E_{X1}(D_{X2}(E_{X1}(T_2)))$$
$$K_3=E_{X1}(D_{X1}(E_{X1}(T_3)))$$

Здесь T_1, T_2 и T_3 - константы, которые необязательно хранить в секрете. Эта схема гарантирует, что для любой конкретной пары ключей наилучшим методом взлома будет вскрытие с известным открытым текстом.

2.2.4. *Режимы тройного шифрования*

Недостаточно просто определить тройное шифрование, его можно выполнить несколькими методами. Решение зависит от требуемых безопасности и эффективности.

Вот два возможных режима тройного шифрования:

Внутренний CBC: Файл зашифровывается в режиме CBC три раза (Рис. 1а). Для этого нужны три различных вектора инициализации (ВИ).

$$C_i=E_{K3}(S_i \oplus C_{i-1});\ S_i=D_{K2}(T \oplus S_{i-1});\ T_i=E_{K1}(P_i \oplus T_{i-1})$$
$$P_i=T_{i-1} \oplus D_{K1}(T_i);\ T_i=S_{i-1} \oplus E_{K2}(S_i);\ S_i=C_{i-1} \oplus D_{K3}(C_i)$$

Где C_0, S_0 и T_0 - векторы инициализации.

Внешний CBC: Файл шифруется с помощью тройного шифрования (один раз) в режиме CBC (Рис. 2). Для этого нужен один вектор ВИ.

$$C_i=E_{K3}(D_{K2}(E_{K1}(P_i \oplus C_{i-1})))$$
$$P_i=C_{i-1} \oplus D_{K1}(E_{K2}(D_{K3}(C_i)))$$

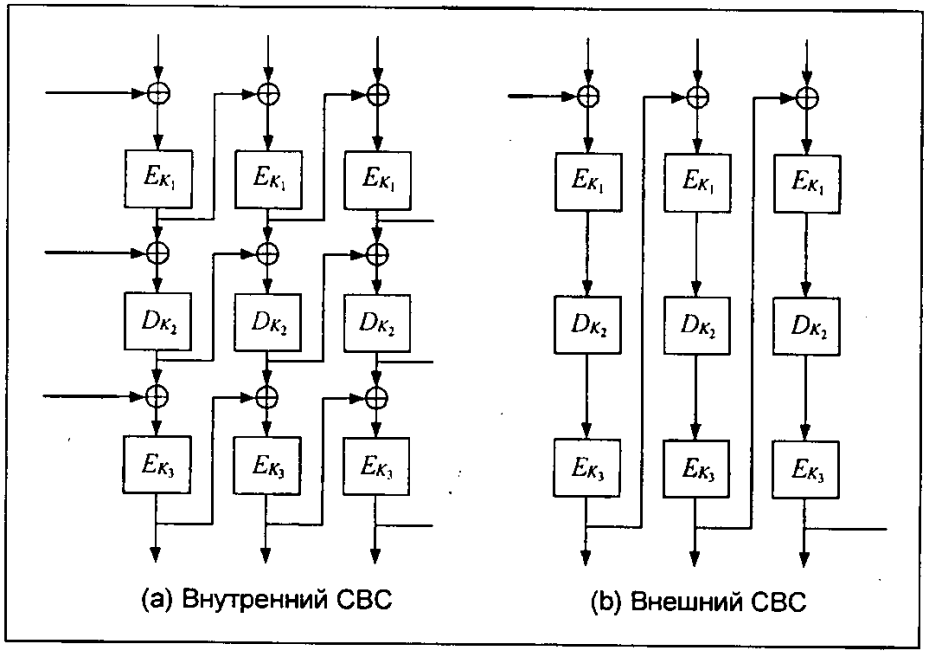


Рис. 2. Тройное шифрование в режиме CBC

Оба режима требуют больше ресурсов, чем однократное шифрование: больше аппаратуры или больше времени. Однако при установке трех шифровальных микросхем производительность внутреннего CBC не меньше, чем при однократном шифровании. Так как три шифрования CBC независимы, три микросхемы могут быть загружены постоянно, подавая свой выход себе на вход.

Напротив, во внешнем CBC обратная связь лежит вне трех процессов шифрования. Это означает, что даже при использовании трех микросхем производительность составит только треть производительности однократного шифрования. Чтобы получить ту же производительность для внешнего CBC, потребуется чередование векторов ВИ.

$$C_i= E_{K3}(D_{K2}(E_{K1}(P_i \oplus C_{i-3})))$$

где C_0, C_{-1} и C_{-2}- векторы инициализации. Это не поможет при программной реализации, разве только при использовании параллельного компьютера.

К сожалению, менее сложный режим также и менее безопасен. Бихам проанализировал устойчивость различных режимов к дифференциальному криптоанализу с подобранными шифртекстами и обнаружил, что внутренний CBC только незначительно надежнее однократного шифрования. Если рассматривать тройное шифрование как большой единый алгоритм, его внутренние обратные связи позволяют вводить внешнюю и известную информацию во внутреннюю структуру алгоритма, что облегчает криптоанализ. Для дифференциальных атак нужно огромное количество подобранных шифртекстов, что делает эти вскрытия не слишком практичными, но этих результатов должно хватить, чтобы насторожить скептически настроенных пользователей. Анализ устойчивости алгоритмов к вскрытиям «в лоб» и «встречей посередине» показал, что и этом отношении оба варианта одинаково надежны.

Кроме перечисленных, известны и другие режимы. Можно зашифровать файл один раз и режиме ECB, затем дважды в CBC, или один раз в CBC, один в ECB и еще раз в CBC, или дважды в CBC и один раз в ECB. Бихам показал, что эти варианты отнюдь не устойчивее однократного DES при вскрытии методом дифференциального криптоанализа с подобранным открытым текстом. Он не оставил больших надежд и для других вариантов. Если вы собираетесь применять тройное шифрование, используйте режимы с внешней обратной связью.

2.2.5. *Варианты тройного шифрования*

Прежде чем было доказано, что DES не образует группу, предлагались различные схемы многократного шифрования. Одним из способов гарантировать, что тройное шифрование не выродится в однократное, было изменение эффективной длины блока. Простой метод предполагает дополнять блок битами. С этой целью между первым и вторым, а также между вторым и третьим шифрованиями текст дополняется строкой случайных битов длиной в полблока (Рис. 6). Если p - это функция дополнения, то:

$$C = E_{K3}(p(E_{K2}(p(E_{K1}(P)))))$$

Дополнение не только маскирует структуру текста, но и обеспечивает перекрытие блоков шифрования, примерно как кирпичи в стене. Длина сообщения увеличивается только на один блок.

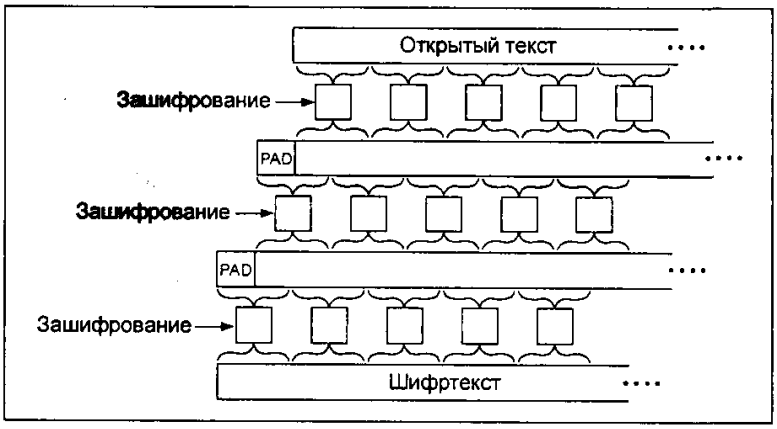


Рис. 3. Тройное шифрование с дополнением

В другом методе, предложенном Карлом Эллисоном (CarlEllison), между тремя шифрованиями используется некоторая бесключевая функция перестановки. Перестановка должна работать с большими блоками - 8 Кбайт или около этого, что делает эффективный размер блока для этого варианта равным 8 Кбайт. Если перестановка выполняется быстро, этот вариант ненамного медленнее, чем базовое тройное шифрование.

$$C = E_{K1}(T(E_{K2}(T(E_{K1}(P)))))$$

T собирает входной (длинойдо 8 Кбайт) и использует генератор псевдослучайных чисел для его перемешивания. Изменение одного входного бита приводит к изменению восьми байтов результата первого шифрования, до 64 байтов - результата второго шифрования и до 512 байтов - результата третьего шифрования. Если каждый блочный алгоритм работает в режиме CBC, как предполагалось первоначально, изменение единичного входного бита, скорее всего, приведет к изменению всего 8-килобайтового блока, даже если это не первый блок.

Новейший вариант этой схемы противодействует атаке на внутренний CBC, предложенной Бихамом, добавлением процедуры отбеливания, позволяющей замаскировать структуру открытых текстов. Эта процедура представляет собой потоковую операцию XOR с криптографически надежным генератором псевдослучайных чисел и обозначена ниже как *R*. *T* мешает криптоаналитику определить априорно ключ, использованный для шифрования любого заданного входного байта последнего шифрования. Второе шифрование обозначено *nE*(шифрование с циклическим использованием *n*различных ключей):

$$C = E_{K3}(R(T(nE_{K2}(T(E_{K1}(R))))))$$

Все шифрования выполняются в режиме ECB, используется не меньше *n+2* ключей шифрования и криптографически стойкий генератор псевдослучайных чисел.

В этой схеме предлагалось использование алгоритма DES, однако она работает с любым блочным алгоритмом. Мне неизвестны факты анализа надежности этой схемы.

2.3. Удвоение длины блока

В академических кругах давно спорят на тему, достаточна ли 64-битовая длина блока. С одной стороны, 64-битовый блок обеспечивает рассеивание открытого текста только на 8 байтов шифртекста. С другой стороны, более длинный блок затрудняет надежную маскировку структуры, а, кроме того, увеличивает вероятность ошибок.

Выдвигались предложения удваивать длину блока алгоритма с помощью многократного шифрования. Прежде, чем реализовывать одно из них, можно оценить возможность вскрытия «встреча посередине». Схема Ричарда Аутбриджа (RichardOuterbridge), показанная на рис. 4, ничуть не безопаснее тройного шифрования с одинарным блоком и двумя ключами.

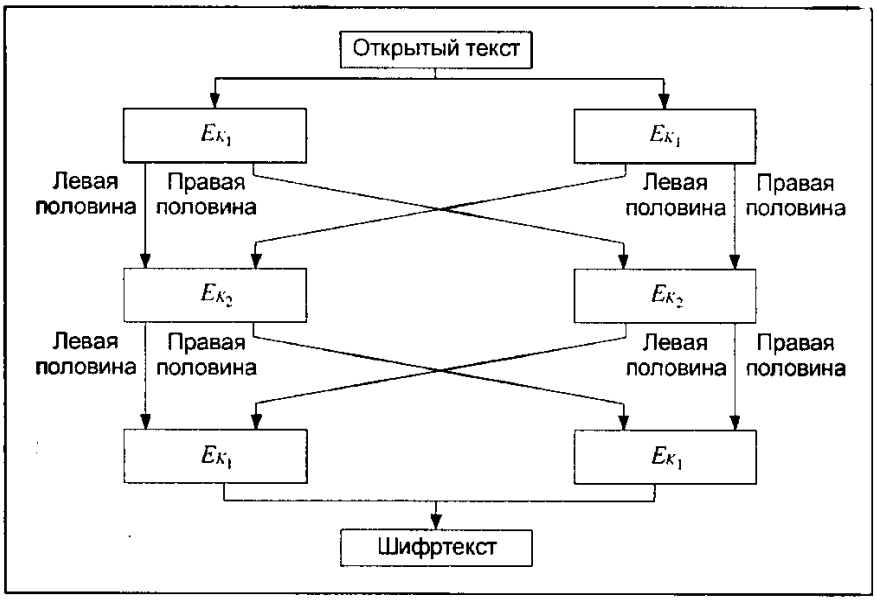


Рис. 4. Удвоение длины блока

Однако подобный прием не быстрее обычного тройного шифрования: для шифрования двух блоков данных все так же нужно шесть шифрований. Характеристики обычного тройного шифрования известны, а за новыми конструкциями часто скрываются новые проблемы.

2.4. Другие схемы многократного шифрования

Недостаток тройного шифрования с двумя ключами заключается в том, что при увеличении вдвое пространства ключей нужно выполнять три шифрования каждого блока открытого текста. Поэтому существуют хитрые способы объединения двух шифрований, которые удвоили бы пространство ключей.

2.4.1 *Двойной режим OFB/счетчика*

Этот метод использует блочный алгоритм для генерации двух гамм, которые используются для шифрования открытого текста.

$$S_i = E_{K1}(S_{i-1} \oplus I_1);\ I_1 = I_1 + 1$$
$$T_i = E_{K2}(T_{i-1} \oplus I_2);\ I_2 = I_2 + 1$$
$$C_i = P_i \oplus S_i \oplus T$$

где S_i и T_i - внутренние переменные, а I_1, и I_2 - счетчики. Две копии блочного алгоритма работают в некотором гибридном режиме OFB/счетчика, а открытый текст, S_i, и T_iобъединяются операцией XOR. При этом ключи *K1*и *K2* независимы.

2.4.2. *Режим ECB + OFB*

Этот метод разработан для шифрования нескольких сообщений фиксированной длины, например, блоков диска. Используются два ключа: *K1*и *K2*. Сначала для генерации маски блока нужной длины используется выбранный алгоритм и ключ *K1*. Эта маска впоследствии используется повторно для шифрования сообщений теми же ключами. Затем выполняется операция XOR над открытым текстом сообщения и маской. Наконец результат этой операции шифруется с помощью выбранного алгоритма и ключа *K2*в режиме ECB.

Этот метод анализировался только в той работе, в которой он и был опубликован. Понятно, что он не слабее одинарного шифрования ECB, и, возможно, столь же устойчив, как и двойное применение алгоритма. Вероятно, криптоаналитик может выполнять независимый поиск двух ключей, если получит несколько файлов открытого текста, зашифрованных одним ключом.

Чтобы затруднить анализ идентичных блоков в одних и тех же местах различных сообщений, можно использовать вектор инициализации (ВИ). В отличие от использования векторов ВИ в других режимах, в данном случае перед шифрованием ECB выполняется операция XOR над каждым блоком сообщения и вектором ВИ.

Мэтт Блейз (MattBlaze) разработал этот режим для своей криптографической файловой системы (CryptographicFileSystem - CFS) UNIX. Это удачный режим, поскольку задержку вызывает только одно шифрование в режиме ECB - маску можно генерировать только один раз и сохранить. В CFS в качестве блочного алгоритма используется DES.

2.4.3. *Схема xDES^i*

DES может использоваться, как компонент ряда блочных алгоритмов с увеличенными размерами ключей и блоков. Эти схемы никак не зависят от DES, и в них может использоваться любой блочный алгоритм.

Первый, xDES[1], представляет собой схему Любы-Ракоффа с блочным шифром в качестве базовой функции. Размер блока вдвое больше размера блока используемого блочного шифра, а размер ключа втрое больше, чем у используемого блочного шифра. В каждом из трех раундов правая половина шифруется блочным алгоритмом и одним из ключей, затем выполняется операция XOR результата с левой половиной, и половины переставляются.

Это быстрее обычного тройного шифрования, так как тремя шифрованиями шифруется блок, длина которого вдвое больше длины блока используемого блочного алгоритма. Но при этом возможна простая атака «встреча посередине», которая позволяет найти ключ с помощью таблицы размером 2^k, где k - размер ключа блочного алгоритма. Правая половина блока открытого текста шифруется с помощью всех возможных значений *K1*,ивыполняется операция XOR с левой половиной открытого текста, и полученные значения сохраняются в таблице. Затем правая половина шифртекста шифруется с помощью всех возможных

значений *K3*, и выполняется поиск совпадений в таблице. При совпадении пара ключей *K1*и *K3* - возможный вариант правого ключа. После нескольких попыток вскрытия останется только один кандидат. Таким образом, xDES[1] нельзя назвать идеальным решением. Более того, известно вскрытие с подобранным открытым текстом, доказывающее, что xDES[1]ненамного прочнее используемого в нем блочного алгоритма.

В xDES[2] эта идея расширяется до 5-раундового алгоритма, размер блока которого в 4, а размер ключа - в 10 раз превышают размеры блока и ключа используемого блочного шифра. На Рис. 5. показан один этап xDES[2], каждый из четырех подблоков по размеру равен блоку используемого блочного шифра, а все 10 ключей независимы.

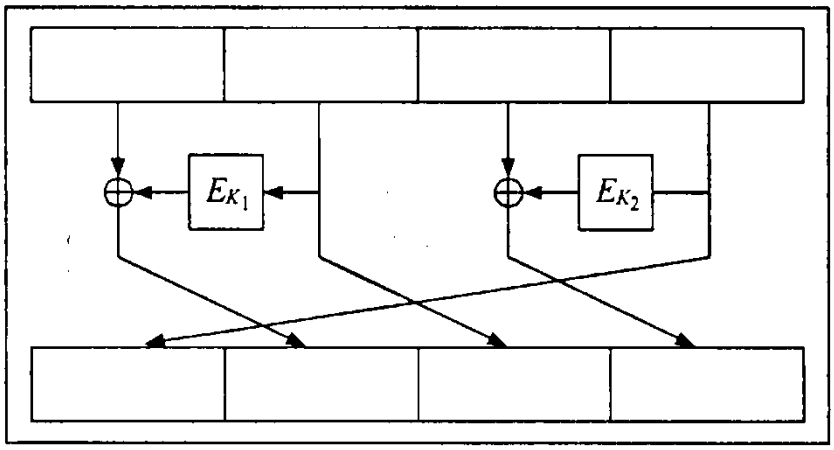


Рис. 5. Один этап xDES[2]

Эта схема также быстрее тройного шифрования: для шифрования блока, который в четыре раза больше блока используемого блочного шифра, нужно 10 шифрований. Однако этот метод уязвим к дифференциальному криптоанализу, и использовать его не стоит. Такая схема остается чувствительной к дифференциальномукриптоанализу, даже если используется DES с независимыми ключами раундов.

При $i \geq 3$ xDES[i] вероятно слишком громоздок, чтобы использовать его в качестве блочного алгоритма. Например, размер блока xDES[3] в 6 раз больше, чем у лежащего и основе блочного шифра, ключ в 21 раз длиннее, а для шифрования блока, который в 6 раз длиннее блока, лежащего в основе блочного шифра, нужно 21 шифрование. Тройное шифрование выполняется быстрее.

2.4.4. *Пятикратное шифрование*

Если тройное шифрование недостаточно надежно – к примеру, хотят зашифровать ключи тройного шифрования еще более сильным алгоритмом - кратность шифрования можно увеличить. Очень устойчиво к вскрытию «встреча посередине» пятикратное шифрование. (Аргументы, аналогичные рассмотренным для двойного шифрования, показывают, что четырехкратное шифрование по сравнению с тройным лишь незначительно повышает надежность).

$$C = E_{K1}(D_{K2}(E_{K3}(D_{K2}(E_{K1}(P)))))$$
$$P = D_{K1}(E_{K2}(D_{K3}(E_{K2}(D_{K1}(C)))))$$

Эта конструкция обратно совместима с тройным шифрованием, если *K2=K3,* и с однократным шифрованием, если *K1=K2=K3*. Конечно, она будет еще надежней, если использовать пять независимых ключей.

2.5. Уменьшение длины ключа в CDMF

Этот метод разработан в IBM для продукта CDMF (CommercialDataMaskingFacility - аппаратура закрытия коммерческих данных). Он предназначен для преобразования 56-битового ключа DES в 40-битовый ключ, разрешенный для экспорта. Предполагается, что в первоначальный ключ DES включены биты четности.

1. Обнуляются биты четности: биты 8, 16, 24, 32, 40, 48, 56, 64.

2. Результат этапа 1 шифруется с помощью DES ключом 0xc408b0540ba1e0ae, результат шифрования объединяется операцией XOR с результатом этапа 1.
3. В результате этапа 2 обнуляются следующие биты: 1, 2, 3, 4, 8, 16, 17, 18, 19, 20, 24, 32, 33, 34, 35, 36, 40, 48, 49, 50, 51, 52, 56, 64.
4. Результат этапа 3 шифруется с помощью DES ключом 0xef2c041ce6382fe6.
Полученный ключ используется для шифрования сообщения.

Но не следует забывать, что этот метод укорачивает ключ и, следовательно, ослабляет алгоритм.

2.6. Отбеливание

Отбеливанием *(whitening)* называют такое преобразование, при котором выполняется операция XOR над входом блочного алгоритма и частью ключа и XOR над выходом блочного алгоритма и другой частью ключа. Впервые этот метод применен для варианта DESX, разработанного в RSADataSecurity, Inc., а затем (по-видимому, независимо) в Khufu и Khafre. (Необычное имя методу дано Ривестом).

Смысл этих действий в том, чтобы помешать криптоаналитику восстановить пары «открытый текст/шифртекст» для исследуемого блочного алгоритма. Метод заставляет криптоаналитика угадывать не только ключ алгоритма, но и одно из значений отбеливания. Так как операция XOR выполняется и до, и после исполнения блочного алгоритма, этот метод считается устойчивым к атаке «встреча посередине».

$$C = K3 \oplus E_{K1}(P \oplus K1)$$
$$P = K1 \oplus D_{K2}(C \oplus K3)$$

Если $K1=K3$, то для вскрытия «в лоб» потребуется $2^{n+m/p}$ операций, где n - размер ключа, m - размер блока, а p -число известных открытых текстов. Если $K1$ и $K3$. различны, то для вскрытия «в лоб» с тремя известными открытыми текстамипотребуется 2^{n+m+1}операций. Против дифференциального и линейного криптоанализа такие меры обеспечивают защиту на уровне всего нескольких битов ключа. Но с вычислительной точки зрения это очень дешевый способ повышения надежности блочного алгоритма.

2.7. Каскадное применение блочных алгоритмов

Есть вариант шифрования сначала алгоритмом A и ключом K_A, а затем еще раз алгоритмом B и ключом K_B? Может быть, у Алисы и Боба различные мнения о том, какой алгоритм надежнее: Алиса хочет пользоваться алгоритмом A, а Боб - алгоритмом B. Этот прием, иногда называемый каскадным применением, можно распространить и на большее количество алгоритмов и ключей.

Пессимисты утверждают, что совместное использование двух алгоритмов не гарантирует повышения безопасности. Алгоритмы могут взаимодействовать каким-то хитрым способом, что на самом деле их надежность даже *уменьшится.* Даже тройное шифрование тремя различными алгоритмами может оказаться не столь безопасным, как вы рассчитываете. Криптография - довольно тонкое искусство, поэтому если не совсем понимать, что и как делать, то можете легко попасть впросак.

Действительность намного светлее. Упомянутые предостережения верны, только если различные ключи зависят друг от друга. Если все используемые ключи независимы, сложность взлома последовательности алгоритмов, по крайней мере, не меньше сложности взлома первого из применяемых алгоритмов. Если второй алгоритм уязвим к атаке с подобранным открытым текстом, то первый алгоритм может облегчить эту атаку и при каскадном применении может сделать второй алгоритм уязвимым к атаке с известным открытым текстом. Такое возможное облегчение вскрытия не ограничивается только алгоритмами шифрования: если вы позволите кому-то другому определить любой из алгоритмов, делающих что-то с вашим сообщением до шифрования, стоит удостовериться, что ваше шифрование устойчиво по отношению к атаке с подобранным открытым текстом. Можно заметить, что наиболее часто используемым алгоритмом для сжатия и оцифровки

речи до модемных скоростей, применяемым перед любым алгоритмом шифрования, служит CELP, разработанный в АНБ.

Это можно сформулировать и иначе: при вскрытии с подобранным открытым текстом каскад шифров взломать не легче, чем любой из шифров каскада. Предыдущий результат показал, что взломать каскад, по крайней мере, не легче, чем самый прочный из шифров каскада. Однако в основе этих результатов лежат некоторые не сформулированные предположения. Только если алгоритмы коммутативны, как в случае каскадных потоковых шифров (или блочных шифров в режиме OFB), надежность их каскада не меньше, чем у сильнейшего из используемых алгоритмов.

Если Алиса и Боб не доверяют алгоритмам друг друга, они могут использовать их каскадом. Для потоковых алгоритмов порядок шифрования значения не имеет. При использовании блочных алгоритмов Алиса может сначала использовать алгоритм*A,* а затем алгоритм *B*. Боб, который больше доверяет алгоритму *B,* может использовать алгоритм *B* перед алгоритмом *A*. Между алгоритмами они могут вставить хороший потоковый шифр. Это не причинит вреда и может значительно повысить безопасность.

Ключи для каждого алгоритма в каскаде должны быть независимыми. Если алгоритм*A*использует 64-битовый ключ, а алгоритм *B* - 128-битовый ключ, то получившийся каскад должен использовать 192-битовый ключ. При использовании зависимых ключей у пессимистов гораздо больше шансов оказаться правыми.

2.8. Объединение нескольких блочных алгоритмов

Есть еще один способ объединения нескольких блочных алгоритмов, надежность которого с гарантией не хуже надежности обоих алгоритмов. Для двух алгоритмов (и двух независимых ключей):

1. Генерируется строка случайных битов *R*того же размера, что и сообщение *M*.
2.Зашифровывается *R*первым алгоритмом.
3. Зашифровывается $M \oplus R$вторым алгоритмом.
4. Шифртекст сообщения представляет собой объединение результатов этапов 2 и 3.

При условии, что строка случайных битов действительно случайна, этот метод шифрует *M* с помощью одноразового блокнота, а затем содержимое блокнота и получившееся сообщение шифруются каждым из двух алгоритмов. Так как для восстановления *M* необходимо и то, и другое, криптоаналитику придется взламывать оба алгоритма. К недостаткам относится удвоение размера шифртекста по сравнению с открытым текстом.

Этот метод можно расширить для нескольких алгоритмов, но добавление каждого алгоритма увеличивает размер шифртекста. Сама по себе идея неплоха, но не слишком практична.

Литература

1. Жданов О.Н., Золотарев В.В. Методы и средства криптографической защиты информации: Учебное пособие / О.Н.Жданов, В.В.Золотарев; СибГАУ. – Красноярск, 2007. – 217 с.

2. Алферов А.П. Основы криптографии : учеб.пособие.

3. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. – М: Гелиос АРВ, 2001. – 480 с., ил.

4. Андреев Н.Н. Основоположник отечественной засекреченной телефонной связи / Н.Н.Андреев, А.П.Петерсон

5. Прянишников К.В., Старовойтов А.В.// Радиотехника, 1998. – № 8.– С. 8–12.

6. Анин Б.Ю. Защита компьютерной информации. / Б. Ю. Анин. –СПб.: БХВ-Петербург, 2000. – 384 с.

7. Бабаш А.В. Криптография-М / А.В.Бабаш, Г.П.Шанкин. –СОЛОН-Р, 2002. – 512 с.

8. Введение в криптографию / под общ. ред. В.В.Ященко. – 3-е изд.,доп. – М. : Изд-во МЦНМО : ЧеРо, 2000. – 288 с.

УДК 681.3.06

МАТЕМАТИЧЕСКОЕ ОБОСНОВАНИЕ АЛГОРИТМА ТЕСТИРОВАНИЯ ПРОГРАММ

А.А.Смагин, А.А.Булаев

Ульяновский государственный университет

Исходные условия.

1. Пусть заданы два множества: конечное множество С (константы и неизменяемые переменные) и конечное множество А (имена констант и неизменяемых переменных), причем мощности Р множеств С и А равна, т.е. Р (С)= Р (А), где С= {c_1, c_2,..., c_n}, А= {a_1, a_2,..., a_n} и пусть между элементами множеств С и А установлено взаимнооднозначное отображение (биекция) γ. Представим взаимосвязь множеств С и А моделью 1

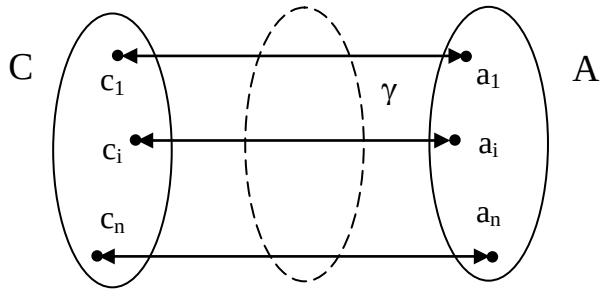


Рис. 1. Модель I.

2. Пусть заданы два других конечных множества C_p (значения констант и неизменяемых переменных) и A_p (реальные значения адресов- физические адреса элементов памяти), мощности которых равны, т.е. $P(C_P)$= Р (A_p), C_p={c_p, c_{P2},... c_{pn}}, A_p={a_{P1}, a_{P2},...a_{pn}} и пусть между элементами множеств C_p и A_p установлено отношение биекции β или

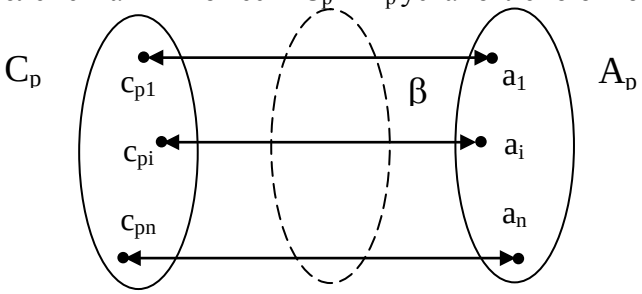


Рис. 2. Модель II.

3. Пусть мощности множеств А и A_p, которые определены в п.1 и п.2, равны: Р (А)= Р (A_p) и между элементами А и A_p существует взаимнооднозначное отображение α

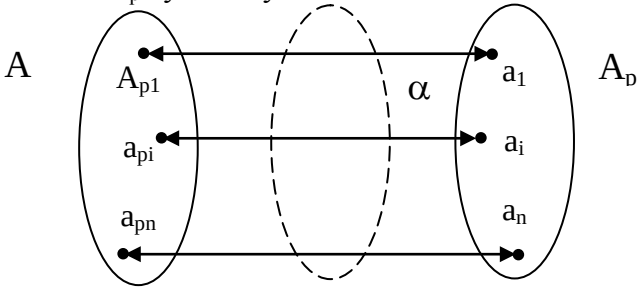


Рис. 3. Модель III.

4. Пусть мощности множеств С и C_p, которые определены в п.1 и п.2, равны Р(С)= $P(C_P)$ и между элементами С и C_p установлено отношение взаимнооднозначного отображения z или биекции

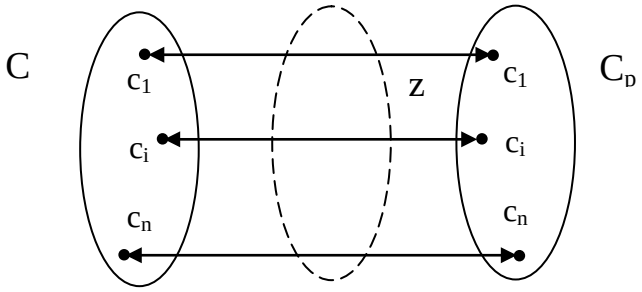


Рис. 4. Модель IV.

Свяжем заданные множества C, A, C_p, A_p отношениями вида коммутативной диаграммы, которая отражает указанные выше свойства пар множеств

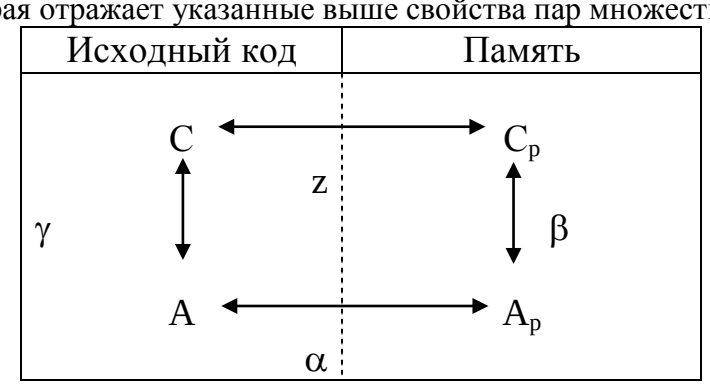


Рис. 5. Модель V.

Модель V (рис.5) определяет условия, при которых происходит процесс построения карты памяти по исходному коду. Модель V является логической моделью и позволяет обосновать построение способа тестирования функционирования программы создания карты памяти.

С этой целью преобразуем модель V к другому, более удобному представлению. Введем два новых множества G и R, состоящие из элементов, каждый из которых в свою очередь, включает в себя пару других элементов из множеств C, A, C_p, A_p. Пусть $G=\{g_j, i= I, n\}$, а $R=\{p_j, i= 1, n\}$,причем $g_i=\{c_i, a_j\}$, $p_i=\{c_{pi}, a_{pi}\}$, G и R равномощны и между элементами gj и pi имеется взаимнооднозначное отображение 0 (т.е. элементы g_j и p_i сами двухэлементные подмножества)

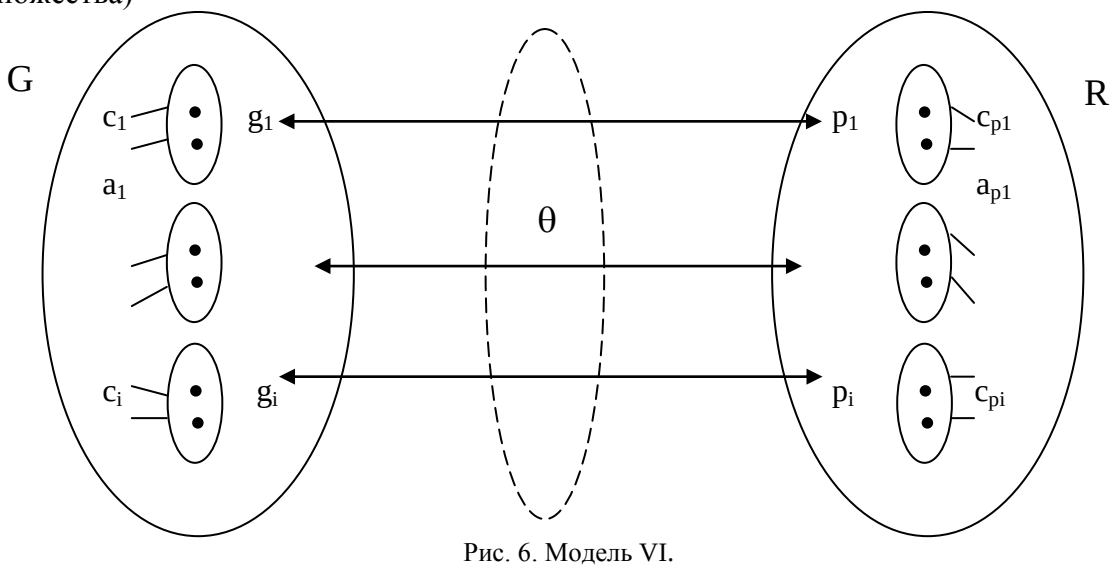


Рис. 6. Модель VI.

Из рассмотрения модели VI (рис.6) вытекает следующая формулировка задачи тестирования.

При заданных G и R необходимо обеспечить, чтобы отображение 0 являлось биекцией.

Другими словами, для любой пары $g=(c_j, a_j)$ из G - константа (неизменная переменная) + имя константы (неизменной переменной) - всегда найдется взаимнооднозначное

соответствие парой (c_{Pj}, a_{Pj}) - реальное значение константы, записанное в ячейку памяти с адресом a_{pi} и для любой пары p_j= (c_{pi} a_{pj}) из R можно найти соответствующую ей пару (c_j, a_j) из G. Иначе говоря, должно выполняться строгое элементное (попарное) соответствие между элементами G и R. Введем в модель V динамику путем воздействия на множества C_p и A_p двух случайных внешних факторов Ф, и Ф2.

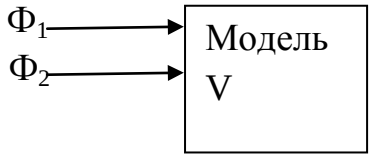


Рис. 7. Динамическая модель. Воздействие факторов $Ф_1$ и $Ф_2$.

Фактор Ф1 случайным образом воздействует на множество A_p и может изменять мощность P (A_p), т.е. количество элементов A_p. Фактор Ф2 случайным образом может воздействовать на элементы множества Ср, заменяя их элементами из этого или же другого числового множества.

Используя динамическую модель (рис.7) сформулируем следующие утверждения.

Утверждение 1. Изменение (уменьшение) мощности множества A_p под воздействием фактора Ф влечет за собой не выполнение биекций β (Модель II, рис.2) и α (Модель III, рис.3), что указывает на ошибку, выражающуюся в потере одного или нескольких реальных адресов ячеек памяти для заданных констант (неизменяемых переменных) и следовательно, и потере в записи в память констант (неизменных переменных).

Утверждение 2. Воздействие фактора Ф2 на множество C_p влечет за собой не выполнение биекций (Модель IV, рис.4) z и β (Модель II, рис.2), что указывает на неправильную запись значений констант (неизменяемых переменных) и на несоответствие имени константы (неизменяемой переменной) и ее значения, указанного в исходном коде.

Утверждение 3. Выполнение хотя бы одного из утверждений 1 и 2 или обоих одновременно приводит к невыполнению отображения z (модель VI, рис.6).

Последнее означает, что диаграмма (модель V, рис.5) являющаяся моделью процесса записи данных в реальную память по исходному коду теряет свое свойство коммутативности, что нарушает исходные условия, а следовательно делает невозможным корректную работу программы формирования карты памяти. Из вышесказанного следует, что в основу тестирования следует положить выполнение двух основных требований:

1) контроль соответствия числа констант (неизменяемых переменных) числу созданных компилятором адресов (исключение факториала Ф1;

2) контроль правильности (соответствия) записываемых в память констант данным в исходном коде (исключение фактора Ф2).

МЕТОДЫ ОЦЕНКИ КОМПЕТЕНЦИЙ ВЫПУСКНИКА ВУЗА

А.А.Смагин, С.В.Липатова, О.Л.Курилова

Ульяновский государственный университет

Компетентностный подход в обучении начал формироваться в 70-х годах XX в. в США и ряде стран Западной Европы в связи с несоответствием запросов рынка труда и уровня профессиональной подготовки выпускников.

На сегодняшний день термины «компетентность», «компетенции» однозначно не определены, разные исследователи приписывают компетенциям разный набор характеристик (см. таблицу 1).

Таблица 1. Определение свойств компетентности российскими учеными.

№	Характеристики	Б.Д. Эльконин	Г.Б. Голуб, О.В. Чуракова	В.А. Прудникова	Е.А. Коган	Л.М. Долгова	С.Е. Шишов, В.А. Калней	А.К. Маркова
1.	Включает развивающее обучение	+			+		+	
2.	Проявляется и формируется в процессе деятельности		+	+		+		+
3.	Включает способность к изменениям в ответ на сложившуюся ситуацию, установить связь между знанием и ситуацией		+				+	
4.	Значимы умения социального типа, а не качества выпускника.				+			
5.	Включает способность действовать на основе полученных знаний.	+				+	+	
7.	Значимы обстоятельствам применения компетенции, а не сумме знаний и умений						+	
8.	Зависимость понятий «компетентность» и «квалификация»							+

Обобщая эти взгляды на понятие, можно определить компетенцию следующим образом – это характеристика специалиста (выпускника), оценивающая его способность адаптировать к изменяющимся условиям (ситуации) свои профессиональные и социальные знания и умения как имеющиеся, так и полученные в процессе профессиональной деятельности и развивающего обучения. Это характеристика специалиста (выпускника), которая определяет его квалификацию.

С точки зрения организации (заказчика и потребителя образовательного продукта – выпускника), компетенция – это набор определенных качеств и поведенческих характеристик, которым должен обладать сотрудник; при этом вариативность компетенций связана не только с отраслью деятельности компании (например, добывающая отрасль, сфера услуг, банковское дело), а с занимаемой позицией (линейный сотрудник, менеджер проекта, бухгалтер, экономист, технический специалист и т.п.).

Цели и результаты образования, формируются через достижение определенных компетенций.

На рисунке 1 схематично представлены элементы компетенций (знания, умения, навыки, поведение) и их методы оценивания в учебном процессе.

Рис.1 Структура компетенции

Существует много методов, позволяющих произвести оценку таких составляющих компетенций как знания, умения, навыки, сформированность компетенций студентов. Они отличаются не только способом взаимодействия с оцениваемым, но и трудозатратностью, имеющейся информационной поддержкой и т.д. (см. таблицу 2).

Таблица 2. Классификация методов оценки компетенций[12], [13].

Метод	Групповой / индивидуальный	Требуется участие тестируемого	Реализация в виде ПО	Временные ресурсы	экономические затраты	Требует участия эксперта
Оценочно-рейтинговая система	индивидуальный	_	+	долгосрочные	+	+
Биографическое анкетирование	индивидуальный	+	_	краткосрочные	_	_
Интервью	индивидуальный	+	_	краткосрочные	+	+
Тесты	индивидуальный	+	+	краткосрочные	+	_
Кейс-метод	индивидуальный	+	_	краткосрочные	+	+
Деловые игры	групповой	+	_	краткосрочные	+	+
Моделирующие игры	групповой	+	+	краткосрочные	+	+
Ролевые игры	групповой	+	_	краткосрочные	+	+
Групповая дискуссия	групповой	+	_	краткосрочные	+	+
Портфолио	индивидуальный	_	+	долгосрочные	-	_
Метод развивающейся кооперации	групповой	+	_	краткосрочные	+	+
Проектный метод	индивидуальный	+	+	краткосрочные	+	_
Мозговой штурм	групповой	+	_	краткосрочные	+	+

Перечисленные методы оценивают одну / несколько составляющих компетенции, но ни один не позволяет всесторонне, полно и в целом оценить различные компетенции.

Предлагается для получения общей оценки компетенций выполнить следующие действия:

проранжировать компетенции по степени их значимости для работодателей – социальных партнеров, которые обязательно должны быть привлечены к ассессменту (оценке);

выбрать методы оценивания компетенций;

сформировать оценочные шкалы, например, от 1 до 5, где 5 – это не менее 80% позитивных и отсутствие негативных проявлений студента по каждой компетенции, а единица – наоборот, 80% негативных проявлений и отсутствие позитивных;

собрать данные по конкретному выпускнику и провести оценку согласно выбранным методам;

используя сформированные шкалы получить общую оценку. Приемлемый уровень развития компетенции отражает средний балл («тройку») — это 60% позитивных образцов поведения и присутствие некоторых негативных образцов.

работа по «настройке» инструментов оценки является, пожалуй, более важной, чем сама оценка, и требует участия не только педагогов, но и сторонних экспертов (в идеале, менеджера по персоналу компании-работодателя): только тогда полученные результаты могут быть действительно объективными.

При оценке компетенций требуется обратная связь, т.е. предоставление студенту развернутого отзыва о выполненной им работе с указанием сильных и слабых сторон, а также конкретных рекомендаций. Правильно организованная обратная связь может стать дополнительным мотивационным фактором для дальнейшего обучения и развития студента в рамках выбранной им специальности. Например, выпускник по направлению подготовки «Информационные системы и технологии» с квалификацией «бакалавр», согласно Федеральному государственному образовательному стандарту [15] должен обладать общекультурными и профессиональными компетенциями, которые представлены на рис.2.

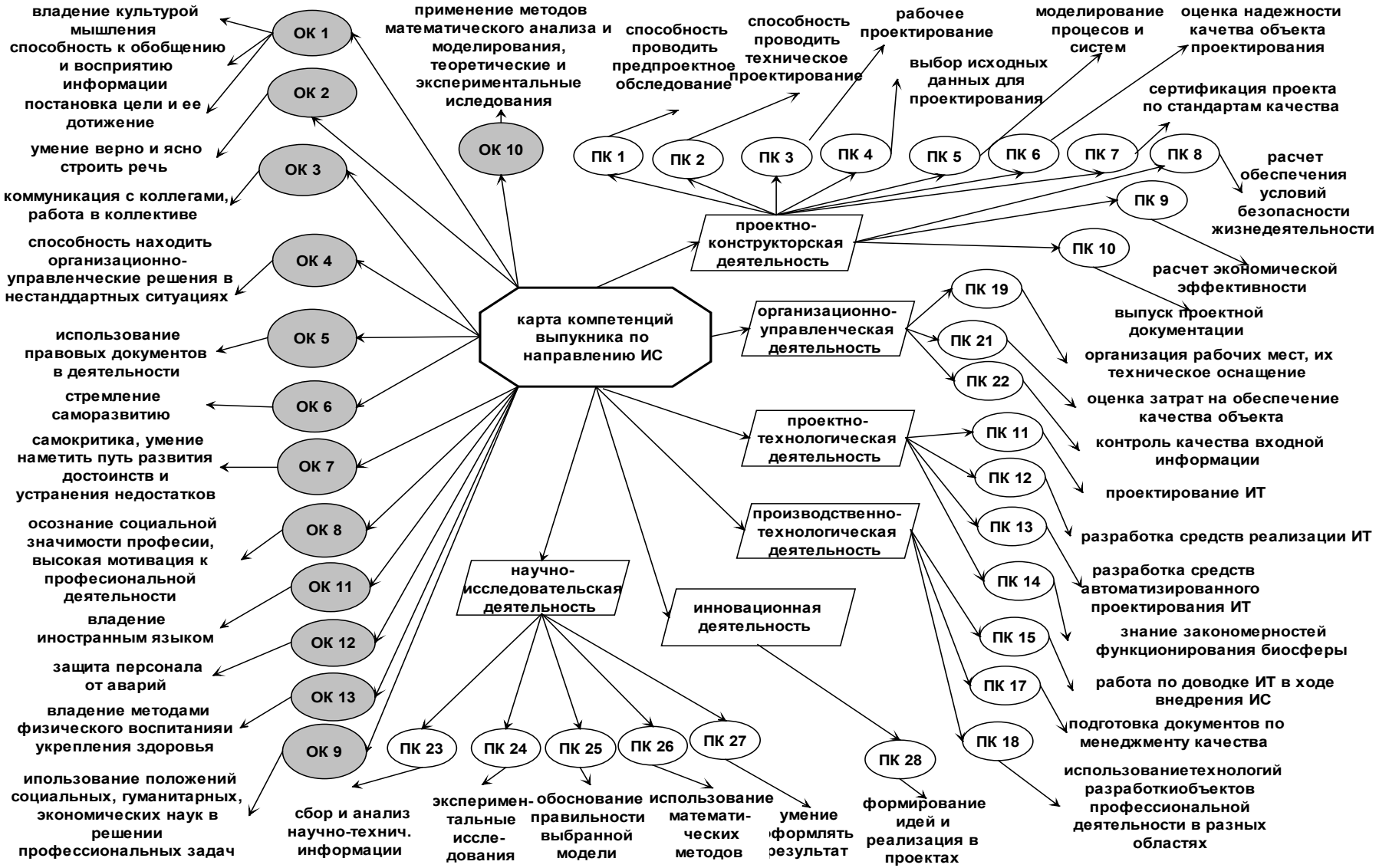


Рис. 2 Карта компетенций выпускника по направлению ИС.

Карта компетенций является моделью компетенций, которыми обладает выпускник, включающая категории профессиональных компетенций и их составляющие. Модель является формализацией представлений о компетенциях бакалавра в области информационных технологий. Аналогичную модель можно построить для любой специальности и любого образовательного этапа (класса школы, курса университета или колледж). Для определения уровня компетенций конкретного выпускника необходимо подобрать наиболее оптимальный способ оценки каждой из компетенций, представленных на карте (см. таблицу 3).

Проанализировав информационные источники [1] - [7], для оценки общекультурных компетенций можно использовать следующие психодиагностические методики и тесты (см. таблицу 3), которые автоматизированы и представлены множеством программных средств, например Экспериментально-диагностический комплекс (ЭДК) [16], Psychometric Expert [17], Maintest 4 [18].

Таблица 3. Компетенции, методы и шкалы оценки.

Компетенции	**Описание**	**Метод оценки**	**шкалы**	**Мин.\макс.**	**шаг**
общекультурные					
ОК–1	владение культурой мышления	Тест структуры интеллекта Амтхауэра [1]	интервальная	мин=20 макс=80	
	способность к обобщению, анализу, восприятию информации	Методика Кэттелла [2], [3]	интервальная	мин=1 макс=10	
	способность к постановке цели и выбору путей ее достижения	Тест смыложизненных ориентаций Леонтьева [4]	интервальная	мин=1 макс=7	
ОК-2	умение логически верно и аргументировано строить устную и письменную речь	Методика Кэттела [2], [3]	интервальная	мин=1 макс=10	
ОК-3 …	готовность к кооперации с коллегами, работе в коллективе	Методика Е.П. Ильина и П.А. Ковалева «Личностная агрессивность, и конфликтность» [6], [7]	интервальная	мин=0 макс=10	
		Уровень мотивации достижения (Мехрабиан) [5]	интервальная	мин=30 макс=210	
профессиональные					
ПК–1	способность проводить предпроектное обследование (инжиниринг) объекта проектирования, системный анализ предметной области, их взаимосвязей	Суммарная оценка по предметам	1) порядковая "2, 3, 4, 5"	мин=2 макс=5	1
			2) номинальная "зачет/ незачет"		
ПК–2	способность проводить	Суммарная оценка по предметам	1) порядковая "2, 3, 4, 5"	мин=2 макс=5	1

	техническое проектирование (реинжиниринг)		2) номинальная "зачет/ незачет"		
ПК–3 …	способность проводить рабочее проектирование	Суммарная оценка по предметам	1) порядковая "2, 3, 4, 5"	мин=2 макс=5	1
			2) номинальная "зачет/ незачет"		

Результатом применения перечисленных методов оценки является набор параметров, характеризующих отдельные свойства (знания, навыки) выпускника. Например, для общекультурной компетенции 3 (ОК-3) можно применить методики, описанные в таблице 3 с выходными параметрами (рис.3):

Рис. 3. Методики и выходные параметры ОК-3

Оценкой профессиональных компетенций занимаются в учебных заведениях в процессе обучения. Поэтому можно использовать сведения об успеваемости студента (выпускника) по конкретным дисциплинам для оценки составляющих компетенций. Для выявления связи между дисциплинами и компетенциями был разработан граф на основе учебного плана кафедры «Телекоммуникационные технологии и сети» УлГУ,

показывающий формирование компетенций в ходе изучения дисциплин. Фрагмент графа представлен на рис. 4.

Для получения общей оценки по профессиональным компетенциям необходимо учитывать элементы компетенций, которые изучаются и измеряются в разных дисциплинах. Для формализации связей между элементами компетенций и их оценками предлагается использовать табличное представление (см. таблица 4).

В таблице 4 приводится фрагмент матрицы компетенций, которая может быть положена в основу формирования оценочных средств контроля качества компетенций, которыми обладает выпускник. Такая матрица соответствует ФГОС ВПО [15] по направлению ИС.

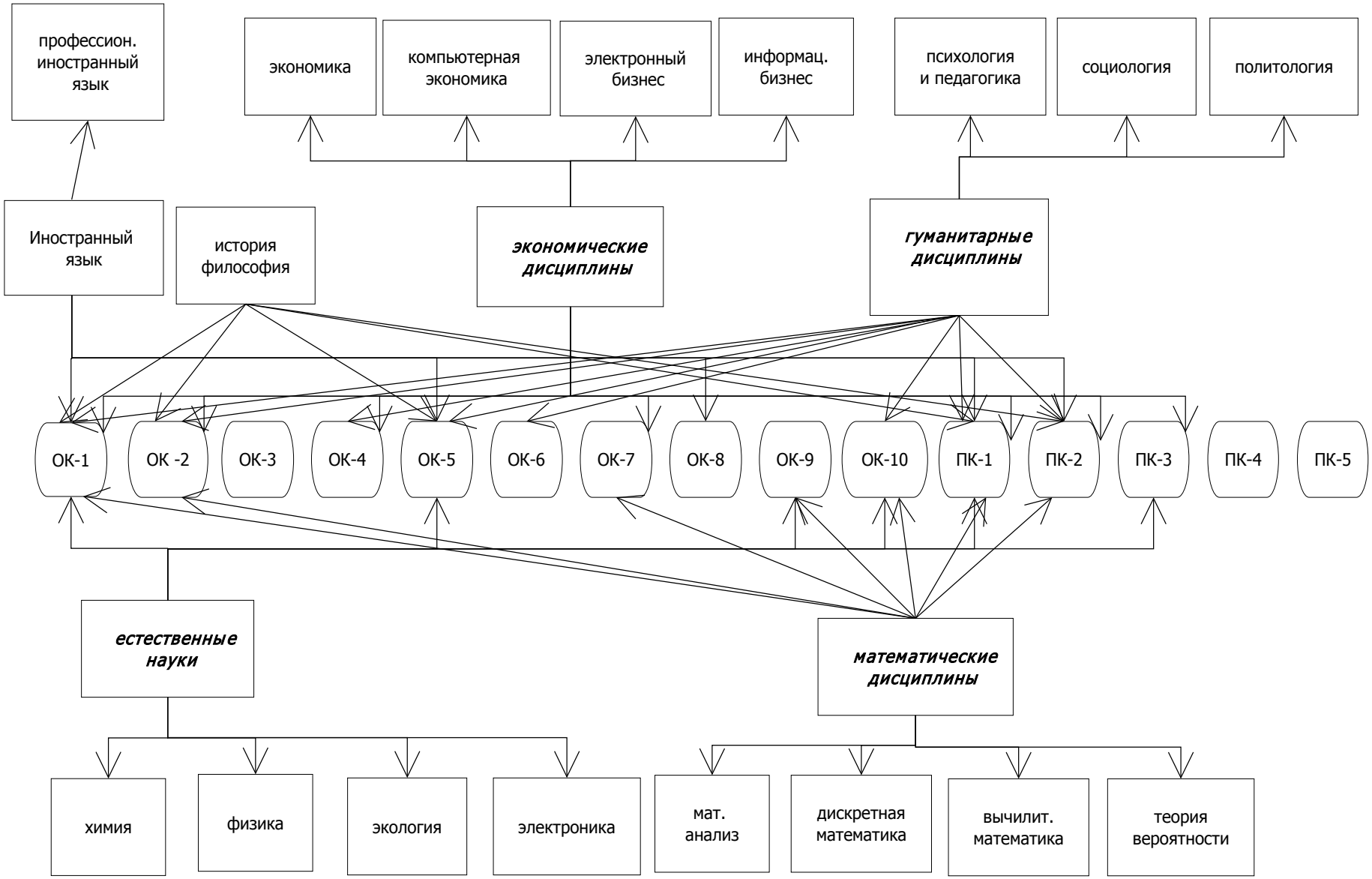


Рис. 4. Фрагмент графа компетенций и дисциплин.

Таблица 4. Фрагмент матрицы компетенций для специальности ИС.

дисциплина	**ПК-1**	**ПК-2**	**ПК-3**	**ПК-4**	**ПК-5**	**ПК-6**	**ПК-7**	**ПК-8**	**ПК-9**	**ПК-10**	**ПК-11**	**ПК-12**	**…**
интеллектуальные системы и технологии	О	О											О
инструментальные средства ИС	О	О											О
инфокоммуникационные системы и сети	О	О				О					О		
методы и средства проектирования ИС	О	О											
метрология, стандартизация и сертификация ИТ	О	О	О	О									
безопасность жизнедеятельности	О	О	О		О								
администрирование в ИС и сетях	О	О				О					О		
компьютерная геометрия и графика	О	О											О
надежность ИС	О	О	О						О	О			
аппаратные средства ЭВМ	О	О											
прикладное ПО ЭВМ и сетей	О	О											О
…													
Учебная практика	О	О	О		О	О						О	О
Научно-исследовательская работа	О	О				О		О	О	О	О	О	О
Производственная практика	О	О	О	О	О			О		О	О		О
Итоговая государственная аттестация	О	О	О	О	О	О	О	О	О	О	О	О	О
Итоговая оценка компетенции	$O_{ПК_1}$	$O_{ПК_2}$	$O_{ПК_3}$	$O_{ПК_4}$	$O_{ПК5}$	$O_{ПК_6}$	$O_{ПК_7}$	$O_{ПК_8}$	$O_{ПК_9}$	$O_{ПК_{10}}$	$O_{ПК_{11}}$	$O_{ПК_{12}}$	…

Вместо символа «О» для оценки конкретного выпускника из ведомостей или зачетной книжки подставляются оценки по дисциплинам. Итоговую оценку для профессиональной компетенции предлагается считать по формуле:

$$O_{\text{ПК}_i} = \frac{1}{2}\sum_{j=1}^{N}\left(\frac{O_{ji} * \omega_i}{n_i} + t_j\right),$$

$$\omega_i = \text{Ч}_i * \left(\sum_{j=1}^{N} \text{Ч}_j\right)^{-1},$$

где N – количество дисциплин за все время обучения,

n_i - количество оценок в i-м столбце,

i – номер компетенции,

j – номер дисциплины,

ω_i - вес элемента компетенции, который зависит от количества часов дисциплины в процессе учебы,

t_j - результаты проверки остаточных знаний по дисциплинам в конце учебы студента,

$Ч_j$ – количество часов, отведенное на изучение дисциплины.

В итоге, оценив профессиональные компетенции выпускника можно построить профили профессиональных компетенций рис.5.

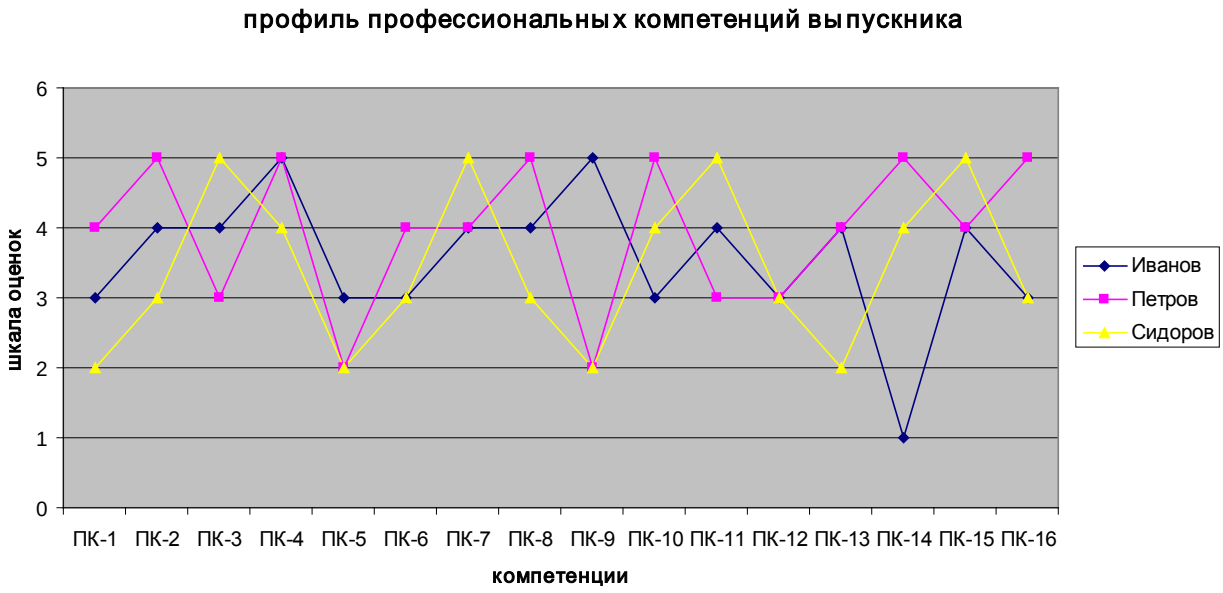


Рис. 5. Профили профессиональных компетенций выпускника.

Для подбора сотрудников согласно компетентностному подходу кадровыми службами необходимы профили должностей, в которых определяется требуемый уровень владения каждой из компетенций для конкретной должности. Примеры профилей должностей представлены в таблице 5.

Таблица 5. Профили должностей.

должность / компетенция	Архитектор информационных систем		Руководитель		Секретарь		Системный аналитик		Эксперт		Ведущий менеджер IT		Менеджер по работе с клиентами		Начальник отдела продаж	
	min	max	min	max	min	max	min	max	min	max	min	max	min	max	min	max
ПК-1	3	4	4	5	3	4	4	5	4	5	3	4	3	4	3	4
ПК-2	4	5	4	5	2	3	3	4	4	5	3	4	4	5	2	3
ПК-3	4	5	3	4	3	4	2	3	3	4	4	5	3	4	4	5
ПК-4	3	4	4	5	3	4	4	5	4	5	3	4	3	4	3	4
ПК-5	4	5	4	5	2	3	3	4	4	5	3	4	4	5	2	3
ПК-6	4	5	3	4	3	4	2	3	3	4	4	5	3	4	4	5
ПК-7	4	5	4	5	2	3	3	4	4	5	3	4	4	5	2	3
…																

При наличии профилей выпускника и должностей можно оценить профессиональную пригодность специалиста для конкретной должности. Для этого предлагается соединить графические представления компетенций выпускника и профилей должностей (рис. 6).

На графике «слой должности» демонстрирует пределы допустимых значений компетенций специалиста. Причем, значения находящиеся ниже слоя не допустимы, а выше возможны. Но при наличии большого количества значений оценок компетенций выше слоя должности, рекомендуется специалисту претендовать на другую должность (выше по иерархии или уровню профессиональной ответственности).

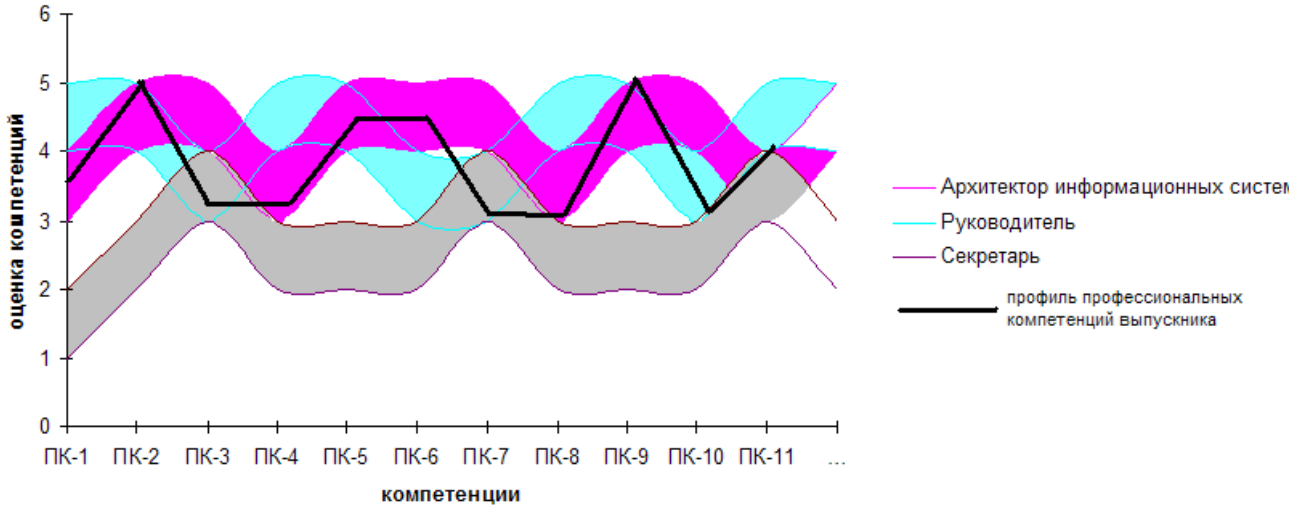


Рис. 6. Соответствие профиля компетенций выпускника определенным должностям.

Описанный метод позволяет эффективно оценивать компетентность учащихся, выпускников, специалистов как профессиональную, так и общекультурную. Получаемая оценка базируется на имеющихся образовательных стандартах и имеет практическую значимость, так как ее может использовать работник кадровых служб, настраивая профили должностей согласно требованиям предприятия. Кроме того, предлагаемый подход к оценке компетенций можно использовать при решении следующих задач:

- при разработке национальных стандартов в сфере профессиональной деятельности для создания унифицированной системы сертификации в различных регионах;
- при разработке учебных программ для повышения квалификации и переквалификации, развития карьеры;
- при создании паспортов профессий и должностных инструкций, которые используются как на предприятиях, так и рекрутинговыми компаниями и биржами труда;
- при разработке учебных курсов для повышения квалификационного уровня без отрыва от работы;
- при разработке систем оценивания и тестирования на соответствие профессиональным сертификатам, паспорту должности, индивидуального уровня профессионализма.

Литература

1. Елисеев О.П., Амтхауэра Р. Тест структуры интеллекта (TSI). Практикум по психологии личности - СПб.: 2003 — С. 342-370.

2. Капустина А. Н. Многофакторная личностная методика Р. Кэттеллла. - СПб.: Речь, 2001. — 112 с.

3. Практическая психодиагностика. Методики и тесты. Учебное пособие. - Самара: Издательский Дом «Бахрах», 1998.

4. Леонтьев Д.А. Тест смысложизненных ориентации (СЖО). 2-е изд. — М.: Смысл, 2000. — 18 с.

5. Фетискин Н.П., Козлов В.В., Мануйлов Г.М. Диагностика мотивации достижения (А.Мехрабиан) / Социально-психологическая диагностика развития личности и малых групп. – М.: 2002. — С.98-102.

6. Ильин Е.П. Мотивация и мотивы. — СПб.: Издательство "Питер", 2000. — С.401-405.

7. Дерманова И.Б. Методика «Личностная агрессивность и конфликтность» / Диагностика эмоционально-нравственного развития. — СПб.: 2002. — С.142-146.

8. Данько Т.П. Методологические вопросы оценки компетенций по стандартам третьего поколения// III Всероссийская научно-практическая конференция «Оценка компетенций и результатов обучения студентов в соответствии с требованиями ФГОС» на базе Российского экономического университета имени Г.В. Плеханова в г. Москве.

9. Морозова Г.Б. Концепция компетентности в практике профессионального отбора. //Сб. науч. тр. — М.: Всероссийский научно-практический центр профориентации и психологической поддержки населения Минтруда России, 2000— Вып. 3.

10. Еникеев М.И.. Психологическая диагностика.// Стандартизированные тесты. М.: 2003.

11. Шишов С.Е., Кальней В.А. Мониторинг качества образования в школе. — М.: Педагогическое общество России, 1999.

12. Лайл М. Спенсер, Сайн М. Спенсер. Компетенции at work. Модели максимальной эффективности работы. — М.: HIPPO, 2005. — С. 372.

13. Уиддет С., Холфорд С., Руководство по компетенциям. — М.: 2008. — С. 228.

14. Боюр Р.В. Практическое применение матрицы компетенций для мониторинга соответствия компетенций обучающихся в вузе внешним требованиям. [Электронный ресурс]. URL: http://edu.tltsu.ru/sites/sites_content/site117/html/media2286/practical_matrix.doc (дата обращения: 12.09.2012).

15. Макет федерального Государственного образовательного стандарта высшего профессионального образования 3 поколения. [Электронный ресурс]. URL: http://www.edu.ru/db/portal/spe/3v.htm (дата обращения: 12.09.2012).

16. Экспериментально-диагностический комплекс (ЭДК) [Электронный ресурс]. URL: http://testpsy.net/ru(дата обращения: 12.09.2012).

17. Psychometric Expert. [Электронный ресурс]. URL: http://www.psychometrica.ru (дата обращения: 12.09.2012).

18. Система тестирования Maintest 4, HR-Лаборатория "Human Technologies". [Электронный ресурс]. URL: http://www.ht.ru/tests/maintest4/#tability (дата обращения: 12.09.2012).

ГРАФО-МАТРИЧНЫЙ ПОДХОД К КОДИРОВАНИЮ ЦЕЛЫХ ЧИСЕЛ

А.А.Смагин, П.И.Смикун

Ульяновский государственный университет

В основу подхода положен ряд следующих принципов

1. Принцип матрично-алгоритмического кодирования, который опирается на использование матрицы чисел и алгоритма построения в матрице чисел кода исходного числа.
2. Принцип наложения ориентированного графа на матрицу.
3. Принцип маршрутизации – использование выделяемого в графе пути как кода исходного числа.
4. Принцип структурного преобразования матрицы чисел, обеспечивающий конечность пути в графе и отсечение лишних вариантов в маршрутизации.
5. Принцип взаимнооднозначности кодирования.

Таким образом, в подходе задействовано две информационно-математические структуры данных – матрицы и граф. С точки зрения иерархии подхода на нижнем уровне находится матрица целых чисел, с помощью которых происходит кодирование исходного числа, а на втором – верхнем уровне «располагается» граф, который определяет способ кодирования.

Совмещение информационных структур – матрицы и графа, представляет собой главную идею матрично-алгоритмического кодирования. Совмещение осуществляется наложением полного графа вершинами на позиции чисел матрицы так, что ребра графа связывают между собой матричные числа по некоторому закону или правилу. С помощью графа можно осуществлять маршрутизацию по матричным числам и формировать последовательности (цепочки) из этих чисел. Полученные последовательности чисел могут образовывать кодовые комбинации .

Для кодирования предлагается использовать не сами матричные числа, а переходы к ним от других матричных чисел, которые участвуют в маршруте, что отображается дугами ориентированного графа и назначением им символов двоичного алфавита.

Рассмотрим квадратную числовую матрицу размерностью 4x4. Все позиции чисел матрицы имеют координаты *i,j,* где *i,j* = {1,2,3,4}. Отождествим позиции матрицы с вершинами (узлами) неориентированного графа и каждой вершине (узлу) графа припишем числовое значение матрицы, соответствующее данной позиции. Далее, свяжем все соседние позиции матрицы ребрами между собой всеми возможными способами. Получим полный объектный координатный граф (ОКГ), который представлен на рисунке 1.

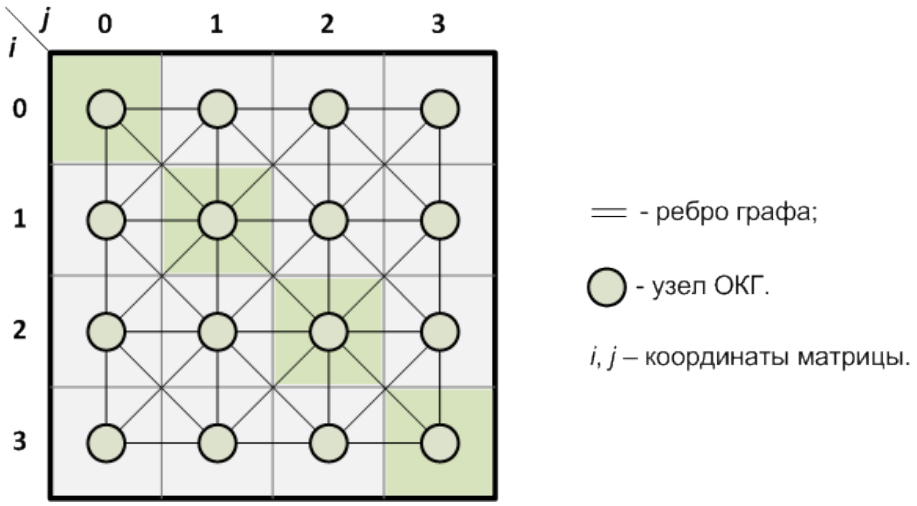


Рис. 1. Объектный координатный граф

Количество узлов ОКГ равно числу позиций матрицы. Каждый узел ОКГ имеет два атрибута:

1. координаты (i,j);
2. числовое значение $n_{i,j}$.

Пример: для матрицы (2x2) объектный граф представлен на рисунке 2

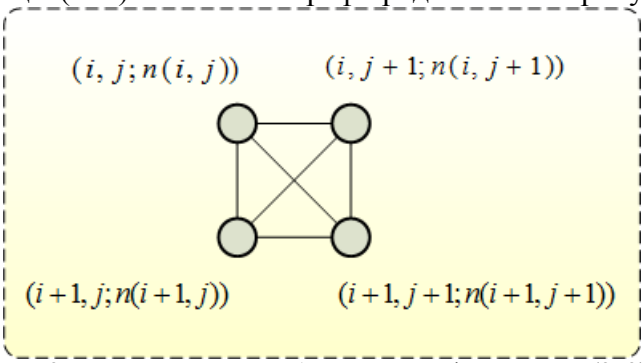


Рис. 2 . Пример полного объектного графа матрицы (2x2)

Пусть кодирование отождествляется с некоторым путем (ориентированным маршрутом без циклов) в полном графе, включающем в себя множество последовательно связанных узлов, у которого имеется начало и конец. С этой целью необходимо полный граф (nxn) преобразовать в ориентированный и для того, чтобы обеспечить отсутствие циклов, требуется удалить из вершин некоторые входы и выходы по следующем) правилу:

1. убрать все строчные дуги, соединяющие соседние вершины с координатами (i,j) с $(i,j+1)$ и $(i,j-1)$ с (i,j).
2. убрать диагональные дуги с координатами $(i+1,j)$ и $(i,j+1)$.

Тогда ОКГ, представленный на рис. 2, преобразуется к следующему виду (рис. 3). Такое построение ориентированного ОКГ позволяет заключить в нем все возможные пути с начальной точки (точки старта). Но при этом возникает необходимость построения правила выделения из всех возможных единственного пути.

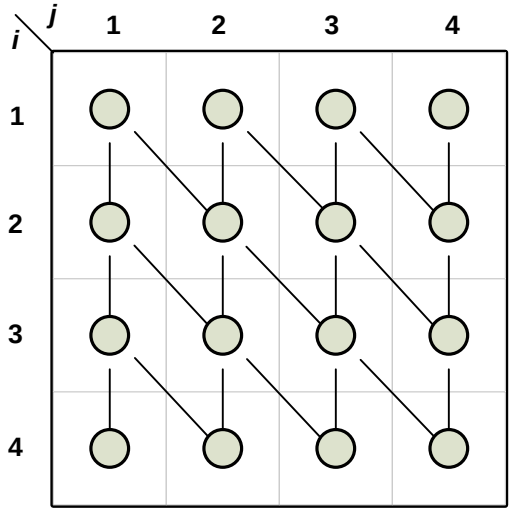


Рис. 3 Неполный направленный объектный граф

Необходимо выполнить такое движение по полному ОКГ, чтобы оно приводило к логическому завершению пути.

Для того, чтобы закодировать заданное целое число, например, бинарным кодом с помощью других целых чисел наиболее часто используется его представление с помощью разложения на множество других чисел, меньших по значению т.е. в виде суммы.. Завершающим шагом - окончанием пути - может служить условие о том, что «дальнейших» чисел, участвующих в разложении в матрице нет. Это ограничение реализуется конечной длиной пути которое может быть получено, например, структурным преобразованием матрицы кодирования.

Пусть в верхней части матрицы будут записаны наименьшие по значению числа, а в нижней части – наибольшие. Тогда в самой верхней строке целесообразно разместить минимальные числа –нули или близкие к нему, а на последующих нижних строках – большие по значению числа.

Объектно-координатный граф, все дуги которого направлены вверх и вверх-влево, наложенный на матрицу чисел, однозначно ориентирует все возможные пути в графе к нулевым числам. Началом пути может быть любая вершина, а завершаться путь будет в вершинах верхней строки.

Вследствие этого возникает избыточность, которую можно устранить за счет структурного преобразования матрицы. Для этого предлагается из квадратной матрицы получить нижнетреуголъную левую матрицу, в верхнем левом углу которой сгруппировать малые числа, а нуль поместить в позицию матрицы с координатами (1, 1). Преобразованная матрица и граф принимает окончательный вид, представленный на рисунке 4.

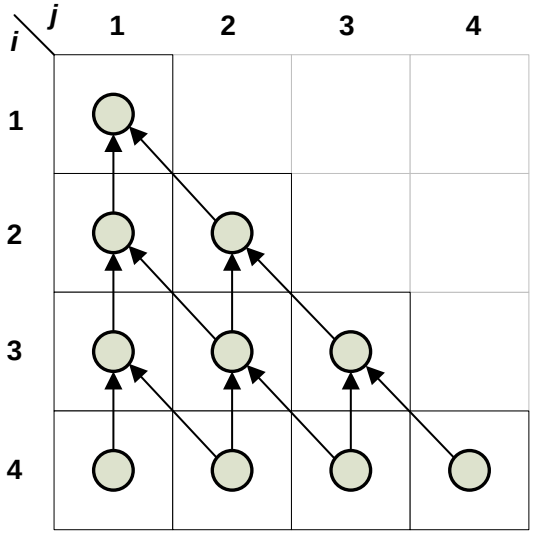


Рис. 4. Треугольная координатная матрица

Полученный треугольный ориентированный граф имеет свойство направленности всех путей в позицию матрицы (1, 1) и представляет собой множество связанных бинарных деревьев и множество корней, из которых формируются пути как последовательности вершин с дугами без повторений и циклов. Конечная вершина (1,1) у всех одна и та же. Длина пути равна номеру строки матрицы, на которой находится стартовая вершина (начало). Число возможных путей из некоторой позиции матрицы (i, j) равно, причем из всех путей формируется только один – кодовый (условие однозначности).

Числа для заполнения треугольной матрицы могут выбираться по-разному, однако их значения оказывают влияние как на само кодирование, так и на обратную процедуру - декодирование.

Так, при кодировании двоичным безизбыточным кодом целых неотрицательных чисел, с помощью которых можно представлять буквы, символы, знаки текста, пикселы изображений, целесообразно рассмотреть натуральный ряд неотрицательных чисел, начиная с единицы. Среди чисел этого ряда нужно выбрать такие, которые можно использовать для построения пути (кода) по числовому полю матрицы. Первым требованием является простой способ их получения (для компьютерного представления), второй – набор выбираемых чисел с целью покрытия возможных вариантов путей кодирования, в третьих - порядок расположения в таблице с учетом роста их значений сверху вниз.

Критерием организации матриц, процессов кодирования /декодирования с их помощью является сложность, которая определяется через параметры размера матрицы и временных затрат на прямое и обратное преобразования. Кроме этого, в некоторых случаях необходимо приводить исходные числа к диапазону чисел, с которыми матрица «работает», а после декодирования последующий учет этой предварительной процедуры.

Дополнительно отметим, что в некоторых случаях в матрице могут встречаться одинаковые числа, однако при определенных условиях эти числа не нарушают принцип взаимно-однозначности прямых и обратных матричных преобразований. Таким образом, на получение матрицы кодирования влияют многие факторы, без учета которых кодирование может оказаться малоэффективным.

В заключении отметим, что графо-матричный подход представляет собой двухуровневый процесс кодирования данных, включающий попеременное обращение к матричным числам и дугам наложенного графа, разнесенные по времени (маршруту) и плоскости (позициям матрицы).

Литература

1. Берлекэмп Э. Алгебраическая теория кодирования. М.: Мир, 1971.477 с.
2. Биркгоф Г., Барти Т. Современная прикладная алгебра. М.: Мир, 1976. 243 с.
3. Смагин А.А., Смикун П.И., Терентьева Ю.Ю. Об одном способе построения блоковых кодов. // Известия Самарского научного центра РАН. Специальный выпуск. Четверть века изысканий и экспериментов по созданию уникальных технологий и материалов для авиаракетостроения. УНТЦ-ФГУП ВИАМ –Самара. Изд-во Самарского научного центра РАН, 2008 –Т.4,№3,2008. С.99-102.

ОСОБЕННОСТИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ СУЩЕСТВУЮЩИХ НАРАБОТОК

В.В.Трясцин

Ульяновский государственный университет

1. Общие сведения по разработке программного обеспечения.

Разработка программного обеспечения - это процесс создания особого вида интеллектуальной собственности, в виде совокупности программ, данных и документации на основании требований заказчика, в общем виде реализующая ответ на поставленный заказчиком вопрос. Если рассматривать процесс разработки программного обеспечения как некий бизнес-процесс (см. рис. 1),

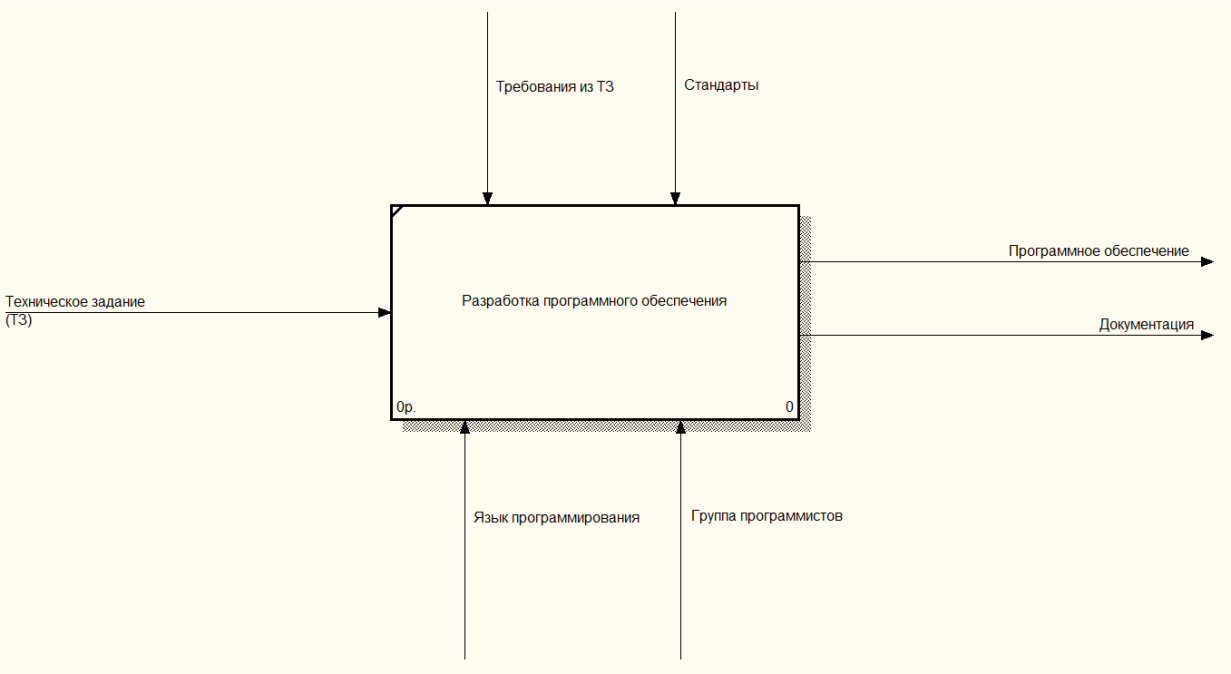


Рис. 1. Модель Входов-Выходов для процесса разработки программного обеспечения

то входом (основанием) разработки является Техническое Задание (ТЗ), а выходом оформленное в виде комплекса программ, решение поставленной в ТЗ задачи. Но не всегда комплекс программ (или иначе - готовое программное обеспечение), поставленный заказчику на том или ином виде носителей информации, является окончанием всех работ. Обычно после стартуют следующие этапы жизненного цикла программного обеспечения[1] (см.рис. 2).

Рис. 2. Этапы жизненного цикла ПО

Так после сдачи программного обеспечения и рабочей документации, заказчик начинает эксплуатацию полученного изделия. Разработчик со своей стороны участвует во внедрении программного комплекса на ЭВМ заказчика, исправляет ошибки, выявленные во

[1] Согласно ГОСТ 34.601-90

время эксплуатации, пишет необходимые обновления. Этап утилизации изделия, свойственный для материальных изделий для программного обеспечения, как правило, отсутствует. Вместо этого, каждый программный продукт, разработанный однажды, впоследствии неоднократно дорабатывается и дополняется согласно возникающим потребностям заказчика до тех пор пока доработка возможна и допустима, а также целесообразна.

Таким образом, программное обеспечение - это динамичный объект, расширяемый, дополняемый и корректируемый с течением времени, а разработка программного обеспечения - это цикличный процесс, при помощи которого программное обеспечение формируется и изменяется. Наиболее близкой и похожей по сути является модель PDCA цикла Деминга-Шухарта[1] (см. рис. 3).

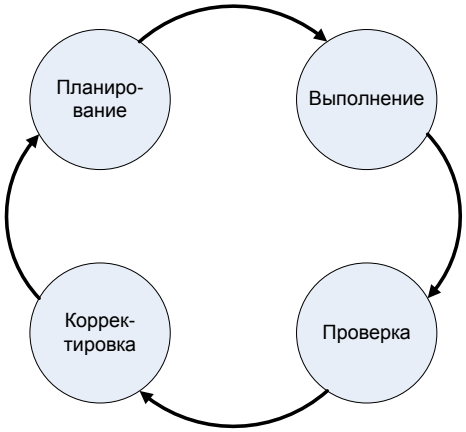


Рис. 3. Модель PDCA.

2. Утилизация программного обеспечения.

Утилизация – это процесс переработки чего-либо вышедшего из употребления или утратившего необходимую функциональность с целью либо повторного использования как всего объекта в целом, так и его составляющих, либо безопасной ликвидации объекта. Утилизация программного обеспечения вместе с тем носит свои характерные черты. Часть авторов в своих статьях[2, 3] рассматривает процесс создания программного обеспечения через призму взаимодействия с окружающей средой и, в частности, с природой. Так как это происходит с материальными продуктами. В развитии этих идет в ведущих компаниях в сфере IT-технологий сформировалась концепция Green Software Strategy («Стратегия экологичного программного обеспечения»)[4]. Суть данной концепции в бережливом использовании ресурсов (как вещественных, так и производственных) и в минимизации воздействия на окружающую среду. Данный эффект достигается при рациональном использовании компьютерной техники, создания условий энергосбережения, решение некоторых экологических проблем при помощи информационных технологий (электронный документооборот вместо бумажных документов).

Проводя аналогию программного продукта и материального продукта, утилизация рассматривается как процесс подготовки продукта и его составляющих к повторному использованию. Само повторное использование продукта рассматривается в трех аспектах: восстановление, повторное использование, переработка.

Восстановление продукта – иногда называется так же ремонтом, а, применимо к программному продукту, доработкой (выпуск патчей, обновлений, заплаток и т.п.) Согласно Рис. **2** для программного обеспечения восстановление продукта и есть этап сопровождения ПО.

Повторное использование – это возможность использования продукта после завершения периода его эксплуатации. К примеру, часть высокоточных механизмов в результате износа уже не могут выполнять операции с запланированной степенью точности. Но их можно

использовать в производстве, где такой квалитет точности не является необходимым условием. Применимо к программному обеспечению речь, как правило, идет о повторном использовании приложений, модулей, функций и библиотек, обладающих некоей универсальностью.

Переработка производится тогда, когда продукт уже нельзя восстановить или повторно использовать так, как он существует. При переработке продукт расчленяется на его составляющие, они в свою очередь на разделяются на компоненты и так далее, пока эти компоненты после определенной доработки нельзя будет использовать в разработке нового продукта. Применимо к программному обеспечению, процесс переработки представляет собой к переосмыслению программному продукта, вычленения логики функционирования, определения основных его взаимосвязей с целью разработке принципиально нового программного обеспечения. Обычно переработка происходит при смене платформы, операционной системы, языка программирования программного продукта в свете новых требований заказчика.

3. Разработка программного обеспечения на государственных предприятиях.

Разработка программного обеспечения на государственных предприятиях имеет свои характерные черты. Разработка программного обеспечения на том или ином предприятии начинается, когда заказчик формирует опытно-конструкторские разработки (далее ОКР) и определяет типовые требования под те или иные функции по автоматизации процессов в органах госструктур. Структура органов довольна статична и от ОКРа к ОКРу практически не меняется. Функции органов госструктур также практически не меняются. Ключевые изменения в основном затрагивают платформу, на которой выполняется требуемое программное обеспечение, язык программирования, так как они также с течением времени совершенствуются и видоизменяются открывая все новые и новые возможности, также требуется оптимизация времени и других ресурсов, более жесткая система разграничения прав доступа и т.п. В общем виде, ОКР можно представить как некую задачу по решению конкретной актуальной проблемы. Для решения задачи разработчику программного обеспечения ставится уникальная цель, на которую накладываются жесткие ограничения по времени готовности, доступным ресурсам, также формируется перечень требований, что обязательно должны быть выполнены.

Разработка программного обеспечения - это циклический процесс. На основании чего можно сделать вывод: Разработка программного обеспечения на государственных предприятиях по своей сути есть нечто среднее и промежуточное между проектами и процессами, а согласно ГОСТ 34.601-90 фазы жизненного цикла программного обеспечения от формирования требований к ПО и до ввода в действия по сути есть проект, а сопровождение ПО есть процесс. При этом важно понимать, что когда в рамках сопровождения ПО накопится критичный уровень ошибок, необходимых исправлений и недочетов - это приведет к началу нового ОКРа и нового проекта.

4. Опыт работы отечественных государственных предприятий в разработке программного обеспечения.

На основании того, что история автоматизации процессов и функций органов госструктур насчитывает уже более 50 лет, можно заметить, что у разработчиков скопился огромный багаж программного обеспечения, который фактически выполняет схожие функции с небольшими изменениями. И, как показывает практика, этот багаж практически никак ни используется повторно, а каждый новый ОКР приводит к разработке программного обеспечения с нуля. Логично предположить, что если использовать уже существующие разработки либо в том виде как они есть, либо с незначительными изменениями, то можно сократить время разработки программного обеспечения, при этом стоит учитывать, что многие функции органов выполняются схожим образом, но с разными входными данными.

Таким образом, возникает проблема о необходимости разработки алгоритма по оценке возможности повторного использования программного обеспечения с целью адекватного

использования готовых наработок при разработке нового программного обеспечения. Но не все уже разработанное программное обеспечение можно использовать повторно как есть. В отдельных случаях требуется его доработка под новые требования. А, порой, готовое программное обеспечение и вовсе нельзя использовать повторно.

5. Описание подходов к разработке нового программного обеспечения с учетом существующих наработок.

Если рассматривать каждую опытно-конструкторскую разработку как решение конкретной задачи по автоматизации процессов управления органов военного управления, на выходе которого получается готовое программное обеспечение, то разумно разбить решение задачи на решения ряда подзадач из которых искомая задача состоит, подзадачи декомпозировать далее и так до тех пор, пока каждая подзадача не будет реализовывать одну какую-то конкретную функцию так, как это принято в построении моделей бизнес-процессов, то можно будет легко оценить чем один ОКР отличается от другого. Полученный бизнес-процесс решения конкретной задачи также обеспечивает прозрачность процессов ведущихся разработок, позволяет однозначно определить необходимый состав программного изделия, распределить работы по группам программистов и далее вплоть до конкретных лиц. Данный бизнес-процесс решения конкретной задачи можно оптимизировать, оценив какие функции являются повторяющимися, какие функции являются универсальными, какие - уникальными, а какие выполняются - разово. Данные, полученные в результате анализа конкретной задачи и оформленные как бизнес-процесс, можно использовать как основные организационные и функциональные требования к программному обеспечению, а также позволяет распределить конкретные работы по группам разработчиков-программистов.

Бизнес-процесс решения конкретной задачи, как правило, оформляется с использованием нотаций графического моделирования бизнес процессов: IDEF0, IDEF3, DFD, Workflow chart, BPMN и другие. Полученная модель также сохраняется в базу информационных ресурсов как объект, который можно использовать повторно с доработкой. То есть при начале новых работ по программному обеспечению можно сразу ввести в поиск ключевые слова для поиска наиболее подходящего алгоритма описанного в модели так, чтобы была возможность его дальнейшей доработки.

Модель бизнес-процесса, оформленная в той или иной нотации графического моделирования и сохраненная в базе данных, имеет смысл не только в разработке нового ПО, но и при разработке ПО с учетом существующих разработок. Для этого необходимо все готовые программные продукты и средства занести в базу данных. При этом каждый занесенный объект необходимо наделить его уникальными характеристиками, описанием его функционала так, ссылками на разработчика данного объекта и документацию, чтобы была возможность оценить степень его пригодности к повторному использованию. Фрагмент примерной структуры базы данных представлен на рисунке 4.

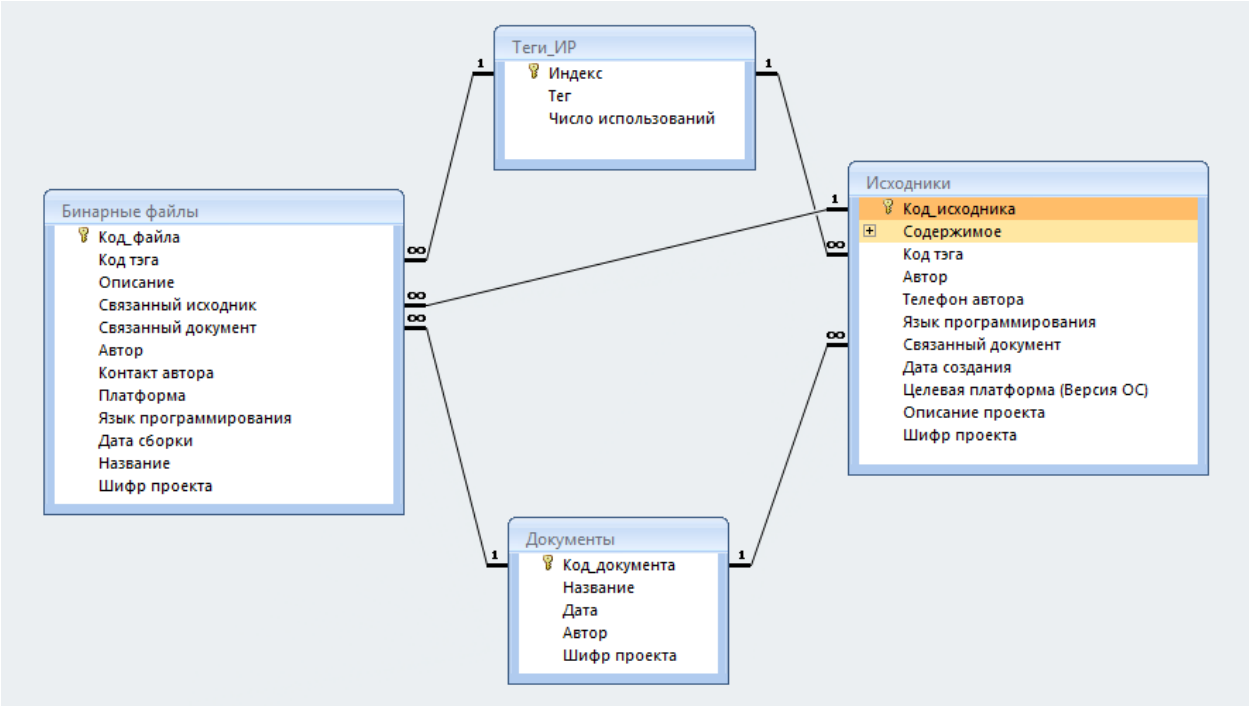


Рис. 4. Фрагмент структуры базы данных информационных ресурсов.

Стоит заметить, что полученные информационные ресурсы, хранящиеся в базе данных, по своей сути будут крайне неоднородными - это исходные коды программных продуктов с возможным подробным описанием каждого исходного файла и определением функционала каждой процедуры и функции в нем, бинарные исполняемые файлы, собранные под ту или иную операционную систему, исполняемые архивы (к примеру - jar-файлы), библиотеки данных, библиотеки функций, плагины, файлы настроек и т.д. При этом в силу неоднородности информационных ресурсов описание их функционала будет сильно различаться по объему. Тем важнее тщательная разработка алгоритма оценки о возможности повторного использования информационного ресурса.

Предположим теперь, что у нас есть готовая база подобных информационных ресурсов, создан алгоритм поиска по заданным параметрам с заданной степенью соответствия, тогда необходимо на этапе эскизного проекта (см.рис. 2) разработать описанную выше модель работ в виде бизнес-процесса. После чего на основании каждой полученной функции бизнес-процесса необходимо осуществить поиск подходящих информационных ресурсов. На выходе подзадачи и отдельные функции бизнес-процесса будут частично "закрыты" уже готовыми решениями (см. рис. 5). Что в конечном счете приведет к снижению стоимости итогового продукта, снижению времени разработки, эффективному использованию наработок, прозрачности ведущихся разработок.

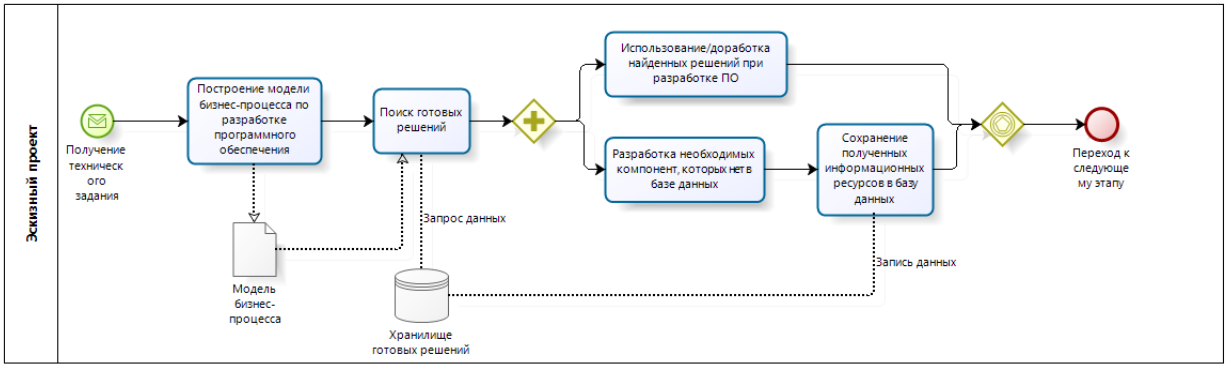


Рис. 5. Схема работы с готовыми наработками при разработке нового ПО.

6. Формализованное описание проблематики проектирования программных изделий с учетом существующих наработок.

Становится целесообразной задача разработки и внедрения не просто автоматизированных систем обработки информации для обеспечения проектирования и разработки программных изделий, но и интегрированных с ними автоматизированных систем, направленных на решение следующих проблем:

1. Обеспечение более высокой степени эффективности разработки программных изделий и прозрачности самого процесса принятия решения по использованию готовых наработок за счет формирования системы поддержки принятия решений (СППР).
2. Разработка механизмов оценки качества принятых решений.

Выбор в реальных ситуациях требует выполнения ряда операций, одни из которых более эффективно выполняет человек, а другие может эффективно выполнять программное обеспечение. Эффективное объединение их достоинств при одновременной компенсации недостатков и воплощается в автоматизированных системах поддержки принятие решений.

Роль программного обеспечения в поддержке принятия решений заключается не в получении окончательного решения как такового, а в осуществлении предварительной подготовки информации об объекте управления и неконтролируемых факторах (среде), с целью помочь просмотреть последствия принятия тех или иных решений, а также в представлении всей этой информации в наглядном и удобном для принятия решений виде.

Для анализа и выработки предложений в СППР используются разные методы. Это могут быть: информационный поиск, интеллектуальный анализ данных, поиск знаний в базах данных, рассуждение на основе прецедентов, имитационное моделирование, генетические алгоритмы, нейронные сети и др. Некоторые из этих методов были разработаны в рамках искусственного интеллекта[5].

Далее для успешного решения поставленных задач необходимо и достаточно:

1. Построить качественную, адекватную действительности, но не перегруженную математическую модель, описывающую область принятия решения при проектировании и разработке программных изделий.
2. Подобрать наиболее подходящий математический каркас для СППР (используя методы искусственного интеллекта, теории принятия решений).
3. Разработать автоматизированную информационную систему, решающую основные задачи проектирования и разработки программных изделий, с интегрированной системой поддержки принятия решения.

Для создания качественной математической модели воспользуемся индуктивным методом построения, позволяющим нам перейти от частных бизнес-процессов к обобщенным.

Определим вход системы как вектор X {n}, где n – количество параметров (критериев), учитываемых в системе. Например, для проектирования и разработки программного обеспечения такими критериями могут быть алгоритмы решения подобных задач, язык программирования, аппаратная платформа, операционная система, какой тип выходных данных системы, универсальное это программное средство или выполняющее узкий круг программных действий, какие должны быть входные и выходные параметры и др.

Обозначим выход системы как вектор Y {m}, m – количество выходных параметров, необходимых для полного описания отклика системы в целом на подачу входных критериев X.

Теперь представим общую систему принятия решения как совокупность трех взаимодействующих подсистем:

1. Идеальная поисковая подсистема.
2. Подсистема, основанная на базе прецедентов.

Идеальная поисковая подсистема - часть модели, с обозначенными выше входом и выходом, основной частью которой является функция поиска N(X), которая обрабатывает и

классифицирует взвешенные входные критерии и выдает соответствующий им результат (рис. 6).

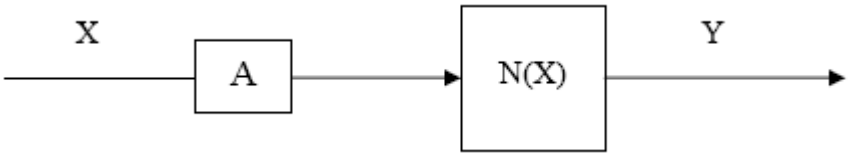


Рис. 6. Идеальная поисковая система

Данная подсистема названа идеальной потому, что результат функции N(X) целиком и полностью зависит только от текущего вектора X и состояния тех требований и критериев оценки поиска того или иного программного изделия. Человек в данном процессе не участвует. A – вектор весовых коэффициентов – определяет участие критерия при формировании конечного результата. Вопрос нахождения значений вектора A представляется сложным, так как детерминированный алгоритм подобрать сложно, возможно использование обучаемой нейронной сети с коррекцией коэффициентов. Так как вектор A должен быть близок к идеальному, то необходимо использование большого обобщенного массива данных для эмпирического подбора его значений. Для ранних исследований возможно пренебречь вектором A.

Одним из основных вопросов данной подсистемы является выбор математической методики для построения функции N(X). Очевидно, что использование детерминированной линейной функции хоть и возможно, но не целесообразно ввиду сложности поиска универсального функционала. Адекватными здесь могут быть экспертные системы, интеллектуальный анализ данных (Data Mining), рассуждения на основе прецедентов, нейронные сети.

Подсистема, основанная на базе прецедентов, работает аналогично идеальной нормативной подсистеме. Отличия заключаются в характере функции P(X), преобразующей вектор входных критериев X в вектор выходных параметров Y.

По сути P(X) – это алгоритм, который, оперируя базой прецедентов, отыскивает аналогичные входному воздействию прецеденты и формирует усредненный выход. Одним из наиболее подходящих методов решения подобных задач является метод рассуждения по прецедентам (casebased reasoning, CBR) - метод формирования умозаключений, опирающийся не на логический вывод от исходных посылок (логические рассуждения), а на поиск и анализ случаев формирования подобных умозаключений в прошлом.

Разумеется, такие умозаключения не являются достоверными и требуют верификации. Проверка корректности умозаключения может являться частью CBR-процесса.

С точки зрения решения задач, рассуждения по прецедентам — это метод получения решения путем поиска подобных проблемных ситуаций в памяти, хранящей прошлый опыт решения задач, и адаптации найденных решений к новым условиям. Применение CBR для решения задач оправдано в случае выполнения следующих условий, касающихся природы прикладной области:

1. Подобные задачи должны иметь подобные решения (принцип регулярности). В этом случае накопленный опыт решения задач может служить отправной точкой процесса поиска решения для новых подобных задач.
2. Виды задач, с которыми сталкивается решатель, должны иметь тенденцию к повторению. Это условие гарантирует, что для многих проблем в будущем будет существовать аналог в прошлом опыте.

Фактически, прецедент — это пара <постановка задачи, метод решения>. Прецеденты хранятся в специальном хранилище, называемом библиотекой прецедентов. Методология рассуждений на основе прецедентов (в общем случае) реализуется в циклической процедуре (4Re-процессы), состоящей из четырех процессов:

1. Поиска подходящего прецедента (retrieve);
2. Применение метода решения (из прецедента) к новой постановке задачи (reuse);
3. Проверка полученного решения и, если необходимо, корректировка (адаптация) решения (revise);

4. Сохранение полученного решения (как прецедента) для последующего использования (retain).

При составлении общей схемы модели принятия решений при проектировании и разработке программных изделий (Рис. 7) необходимо учесть, что идеальная подсистема и подсистема, основанная на базе прецедентов, должны работать параллельно для того, чтобы достичь определенной степени независимости и объективности выходов для формирования механизмов коррекции и/или оценки принятых решений.

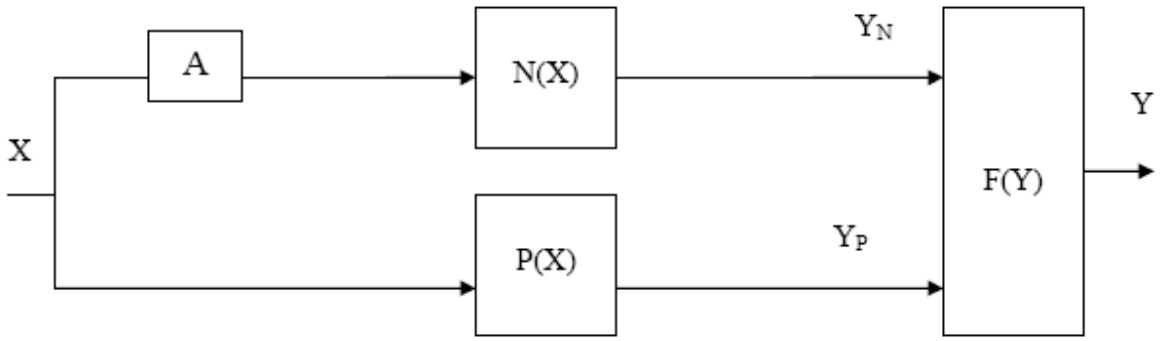


Рис. 7. Общая схема модели принятия решения при проектировании и разработке программных изделий

F(Y) – линейная функция, определяющая зависимость между выходом Y_N идеальной подсистемы и Y_P – подсистемы, основанной на базе прецедентов. Простейший вид: $F(Y) = (Y_N+Y_P)*0.5$ - усредняет значение выхода и выдает общий ответ СППР.

При возникновении задачи оценки качества принимаемых решений функция может принять вид:

$$F(Y) = F(Y_N-Y_P)$$

Далее при обработке пар (X, F(Y)), (X, Y_N) и (X, Y_P) статистическими методами можно получить непосредственные критерии качества работоспособности функции поиска и подбора информационных ресурсов.

Обобщая некоторые результаты, полученные при построении модели, стоит отметить, что особый интерес вызывает выбор конкретных математических методов, используемых в блоках N(X) и особенно P(X). Для работы с базой прецедентов максимально подходящим представляется указанный выше метод рассуждения по прецедентам, активно применяемый в настоящее время в различных областях (медицинской диагностике, диагностике спутникового оборудования, системах машинного обучения и т.д.). Апробированный во многих сферах и вполне доказавший свою состоятельность математический аппарат рассуждения по прецедентам должен дать (конечно, с определенными адаптационными доработками) неплохие результаты в системе поддержки принятия решения при разработке и проектировании программных изделий на предложенной математической модели.

Литература

1. Репин В.В., Елиферов В.Г. Процессный подход к управлению. Моделирование бизнес-процессов. – М.:РИА «Стандарты и качество», 2008. – 408 с. – ISBN 978-5-94938-063-5

2. Сидоров М.О. Экология программного обеспечения. – Материалы Всеукраинской конференции аспирантов и студентов «Инженерия программного обеспечения 2006» – К.: НАУ, 2006.

3. Сидоров Н.А. Повторное использование программного обеспечения// Кибернетика. – 1989. - №3 – с.46-51

4. IBM Software: A green strategy for your entire organization. IBM Software for a greener world June. 2008. NY 10589. U.S.A. Produced in the United States of America. May 2008.

5. Бальзамов А.А. Построение модели и выбор математических методов системы поддержки принятия решения в судебном производстве. // Мордовский государственный университет им. Н.П. Огарева.

УДК 621.391

ОБОСНОВАНИЕ ПАРАМЕТРОВ НЕПАРАМЕТРИЧЕСКОЙ ПРОЦЕДУРЫ ВОССТАНОВЛЕНИЯ АПРИОРНО НЕОПРЕДЕЛЕННОЙ ПЛОТНОСТИ РАСПРЕДЕЛЕНИЯ ВЕРОЯТНОСТЕЙ

Ю.Д.Украинцев[1], К.Ю.Украинцев[2], А.С.Нагорнов[1]

[1]Ульяновский государственный университет

[2]Ульяновский государственный технический университет

Представлен выбор параметров парзеновской процедуры восстановления неизвестной плотности распределения вероятности наблюдаемого на входе приемника сигнала (как наиболее полной его характеристики). В дальнейшем эта процедура будет использована для модификации байесовского подхода принятия решений при наличии непараметрической статистики относительно сигналов и помех на линиях связи. Это позволит повысить качество работы современных линий беспроводной связи.

Ключевые слова: плотность распределения, выборка мгновенных значений огибающей принимаемого сигнала, аппроксимирующее ядро, критерий оценки.

Введение

Современный этап развития мировой цивилизации характеризуется переходом от индустриального общества к информационному обществу. Данная фаза развития человечества предполагает новые формы социальной и экономической деятельности, обеспечивающих получение новых знаний.

Материальной основой (инфраструктурой) построения информационного общества является современная информационно-телекоммуникационная сеть, базирующаяся на транспортных потоках, образованных системами спутниковой, радиорелейной и оптико-волоконных линий связи, т.е. на составных каналах связи.

На современных линиях связи транспортным потоком информации на физическом уровне модели OSI является цифровой бинарный канал, основным показателем качества которого является вероятность ошибочного приема. Эта вероятность ошибочного приема на линиях беспроводной связи в основе своей зависит от условий распространения радиоволн. Именно среда распространения радиоволн влияет на статистические характеристики сигналов и помех на входе радиоприемного устройства.

В настоящее время все пороговые схемы принятия решений строятся на основе хорошо изученного байесовского подхода, полагая, что статистические характеристики сигналов и помех описываются гауссовской плотностью распределения вероятностей. Экспериментальные исследования статистических характеристик сигналов и помех, проводимые на линиях беспроводной связи [1,2,3] показывают, что их плотности распределения вероятностей подчиняются не нормальному, а релеевскому, райсовскому или логарифмически-нормальному законам распределения. Более того, весьма проблематично определение типа представленных законов распределения в течение реального сеанса связи. Учитывая, что наиболее «тяжелым» является релеевский закон, имеющий вид:

$$Z^2 = \sqrt{(X^2 + Y^2)} \quad , \qquad (1)$$

где: X и Y – плотности распределения вероятностей (ПРВ), подчиняющихся нормальному распределению.

При конструировании устройств принятия решений на основе байесовского подхода все расчеты ведутся на наихудший случай, т.е. (релеевское распределение).

В свою очередь, расчеты на наихудший случай не всегда экономически являются оптимальными.

Все это свидетельствует о целесообразности текущей оценки ПРВ мгновенных значений огибающей сигнала на входе радиоприемного устройства и уже на ее основании определение классическим байесовским методом порога решающей схемы. Подобная оценка позволит построить системы принятия решений инвариантные не только к атмосферным, но и индустриальным помехам.

Одним из классов оценки неизвестной априорно неизвестных статистических характеристик, каковыми и являются мгновенные значения огибающей сигнала на входе радиоприемного устройства, восстанавливающих сглаженную плотность распределения вероятностей является метод Парзена – Розенблатта. При этом наблюдаются ярко выраженные значения наиболее вероятного значения оцениваемого параметра (моды) и «хвостов» ПРВ, что позволит в дальнейшем использовать полученное значение ПРВ для определения порога решающей схемы на основе байесовского подхода. Наиболее сложным моментом при решении поставленной задачи является выбор параметров парзеновской оценки ПРВ.

В настоящей работе на основе имитационного моделирования проведены исследования по выбору параметров. Параметрами парзеновской процедуры являются:

- вид аппроксимирующего ядра (в работе использованы 4-е ядра;
- ширина окна аппроксимирующего ядра со своими параметрами;
- объем необходимой выборки.

Постановка задачи

Дана выборка

$$X = \chi_1, \chi_2, \chi_3 \cdots \chi_n \tag{2}$$

из генеральной совокупности {x} независимых, одинаково распределенных случайных величин, подчиняющихся неизвестной плотности распределения вероятностей мгновенных значений огибающей принимаемого сигнала W(x).

На основе обработки полученной выборки рекуррентным парзеновским методом при различных его параметрах необходимо получить оценку W(x) [8,9,10].

$$W_N(X) = W_{N-1}(X) + 1/N(W_{N-1}(X) + 1/h_N \cdot K(y)) \tag{3}$$

где: $W_N(X)$ – оценочное значение W(x) при ограниченном объеме выборки;

$K(y)$ – аппроксимирующая функция (вид исследуемых функций представлен ниже);

$h_N = C \cdot N^{-\alpha}$ –ширина аппроксимирующего ядра $K(y)$; при этом: $0 < \alpha < 0.5$;

C – постоянная, влияющая на ширину аппроксимирующего ядра;

N - объем выборки.

Путем сравнения получаемых оценок W(x) с плотностями распределения вероятностей, подчиняющихся известным законам, обосновать выбор оптимальных значений параметров при ограниченном объеме выборки.

При проведении анализа использовались четыре аппроксимирующие функции, заимствованные из [1] представленные ниже:

$$K(y) = (1/\sqrt{2\pi}) \cdot \exp(-(X - \chi_i)^2 / h_N{}^2) \tag{4}$$

$$K(y) = 1/2 \cdot \exp(-abs(X - \chi_i)/h_N) \tag{5}$$

$$K(y) = 1/2\pi \cdot ((\sin(X - \chi_i)/2h_N)/(X - \chi_i)/2h_N)^2 \tag{6}$$

$$K(y) = 1/\pi \cdot (1/(1 + (X - \chi_i)/h_N)^2) \tag{7}$$

где: χ_i - текущие мгновенные значения огибающей сигнала на входе приемника;

X – значение из интервала восстановления (оценки) плотности распределения вероятностей сигнала.

Цель работы: определить оптимальные в соответствии с критерием Колмогорова-Смирнова параметры парзеновской процедуры восстановления априорно неопределенной ПРВ.

Решение задачи.

Исследование и выбор параметров парзеновской процедуры оценки проведе метом имитационного моделирования. При этом в качестве исследуемых законов использовались: нормальный, релеевский и логарифмически-нормальный законы распределений, как наиболее часто наблюдаемые на современных телекоммуникационных линиях[10]. Случайные числа, подчиняющиеся этим законам, воспроизводились с помощью известных датчиков случайных чисел.

В качестве эталонного закона распределения использовался закон, выведенный по известным математическим выражениям. Для получения оценочных значений качества восстановления известной плотности распределения вероятности в работе использовался непараметрический критерий Колмогорова-Смирнова, позволяющий определить соответствие оцениваемой плотности распределения вероятности теоретической. В работе [4] показано, что данный критерий наиболее эффективен в тех случаях, когда объем выборки N достаточно велик (N>100).

$$\delta = W_N(X) - W(X), \qquad (8)$$

где: $W_N(X)$ и $W(X)$ – соответственно истинная и теоретическая плотности распределения вероятностей принимаемого сигнала.

Результаты имитационного моделирования, представлены на графиках (рис. 1 – 4). Каждая пара графиков соответствует оценке плотности распределения вероятностей сигнала на основе парзеновской процедуры при конкретном выборе аппроксимирующей функции. Зеленый цветов выделен график, соответствующий теоретической плотности распределения вероятности, синим цветом показан график, соответствующий оцениваемой плотности распределения вероятности, красным цветом показан график, полученный методом Колмогорова-Смирнова, показывающий разность между оцениваемой и теоретической плотностью распределения вероятности. При анализе используется метод Монте – Карло. Варьируемыми параметрами процедуры оценивания являются: $K(y)$ - парзеновские ядра, представленные в таблице 1; С - параметр масштаба, изменяемый дискретно в пределах от 1 до 10; α - параметр, влияющий на ширину аппроксимирующего ядра, изменяемый в пределах от 0 до 0,5.

Результаты решения задачи

1.Оценка точности восстановления случайных значений, подчиняющихся нормальной плотности распределения вероятностей:

1.1 При аппроксимирующей функции: $K(y) = (1/\sqrt{2\pi}) \cdot \exp(-(X - \chi_i)^2 / h_N{}^2)$

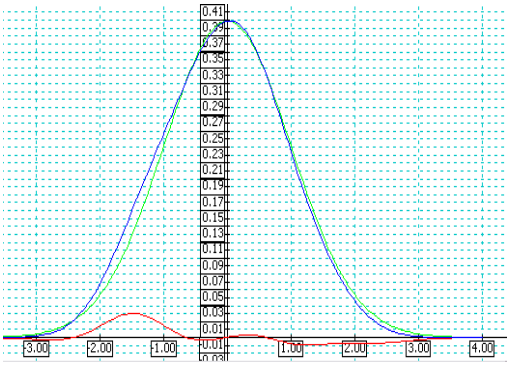


Рис.1 График восстановление ПРВ при N=100; C=1,2; θ= 0.18.

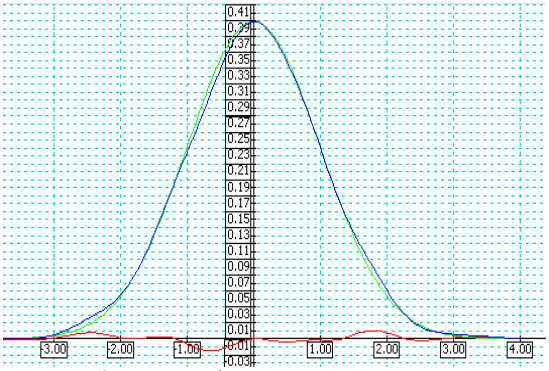


Рис. 2 График восстановление ПРВ при N=1000; C=1,2; θ= 0.18.

Из представленных графиков даже визуально видна высокая точность восстановления нормального распределения не только максимума (моды), но и «хвостов», что свидетельствует о возможности использовании восстановленной ПРВ, при определении в дальнейшем порога решающей схемы приемного устройства на основе известного байесовского подхода.

1.2 При аппроксимирующей функции: $K(y) = 1/2 \cdot \exp(-abs(X - \chi_i)/h_N)$

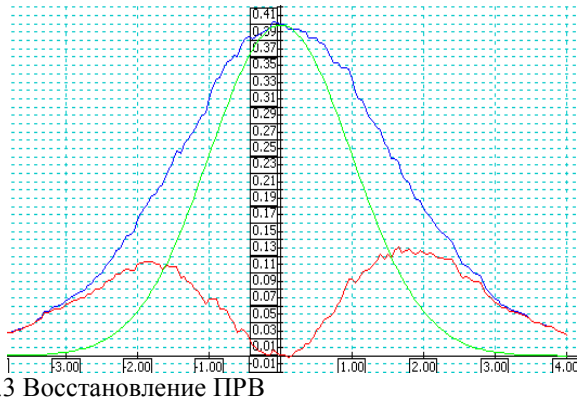


Рис.3 Восстановление ПРВ
при N=100; C=1,2; θ= 0,18.

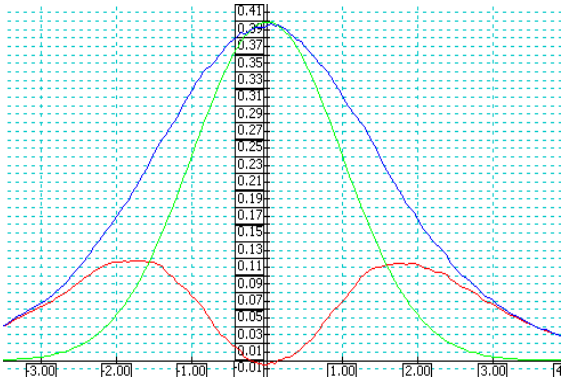


Рис.4 Восстановление ПРВ
при N=1000; C=1,2; θ= 0,18.

1.3 При аппроксимирующей функции $K(y) = 1/2\pi \cdot ((\sin(X - \chi_i)/2h_N)/(X - \chi_i)/2h_N)^2$

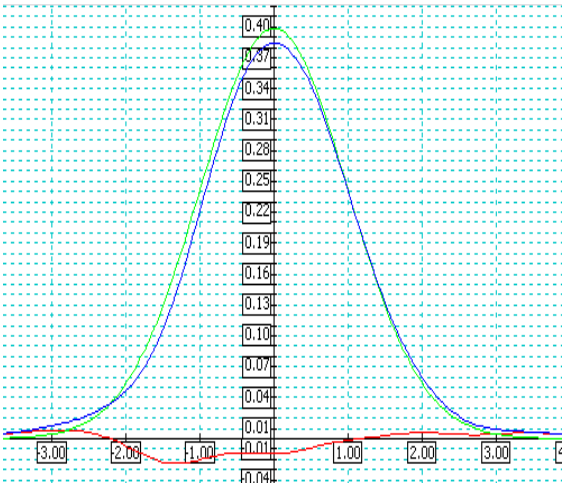


Рис.5 Восстановление ПРВ
при N=100 , C=1,2; θ = 0,18.

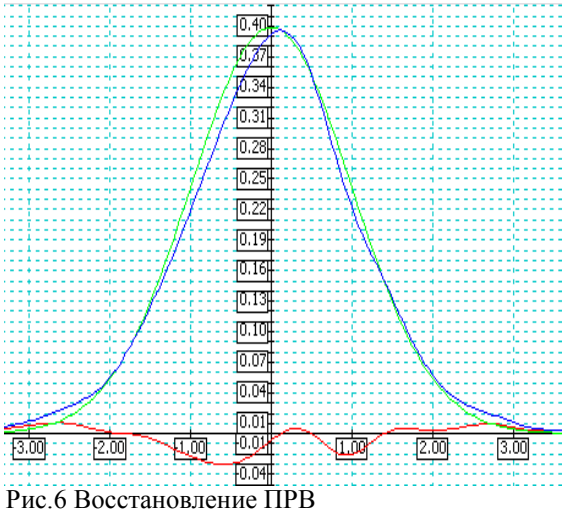


Рис.6 Восстановление ПРВ
при N=1000 , C=1,2; θ = 0,18

1.4 При аппроксимирующей функции: $K(y) = 1/\pi \cdot (1/(1+(X - \chi_i)/h_N)^2)$

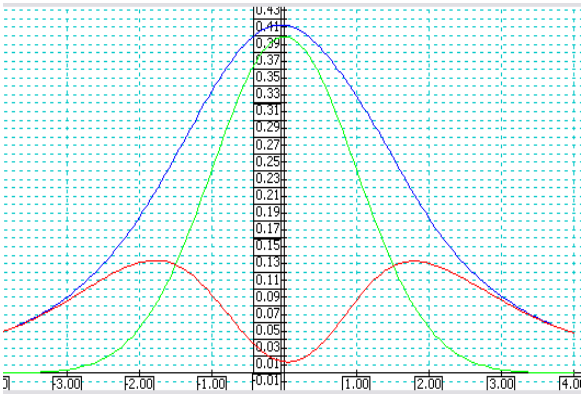


Рис.7 Восстановление ПРВ
при N=100; C=1.2; θ = 0,18.

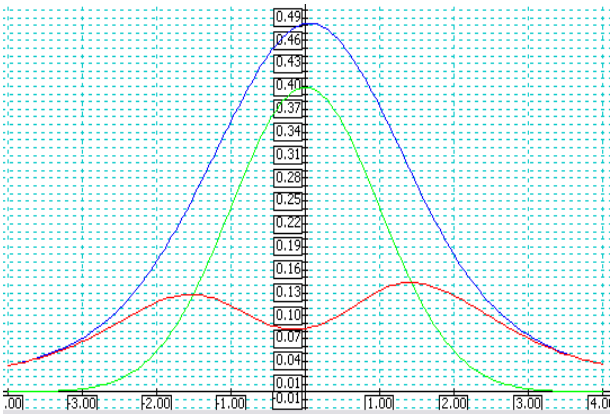


Рис.8 Восстановление ПРВ
при N=1000; C=1,2; θ= 0,18.

Из графиков, приведенных на рис 3-8 видно, что ошибка точности восстановления ПРВ значительно выше, чем на рис 1-2. Это говорит о целесообразности использования аппроксимирующей функции вида (4) для восстановления нормального закона распределения, ошибка восстановления даже «хвостов которого не превышает 1% при объеме выборки равном 1000 отсчетов.

2. Оценка точности восстановления случайных значений, подчиняющихся релеевской плотности распределения вероятностей:

2.1 При аппроксимирующей функции: $K(y) = (1/\sqrt{2\pi}) \cdot \exp(-(X - \chi_i)^2 / h_N{}^2)$

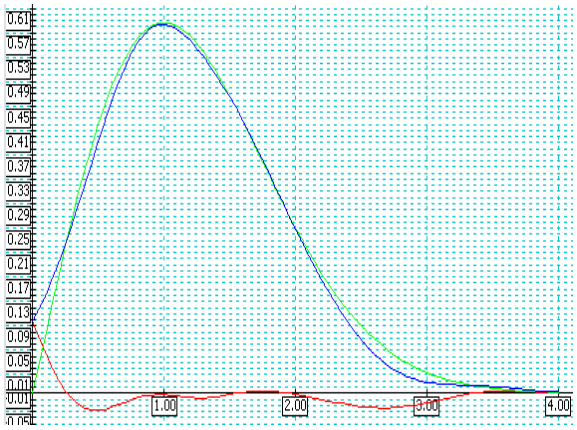


Рис. 9 Восстановление ПРВ
при N=100; C=1,2; θ=0,18;

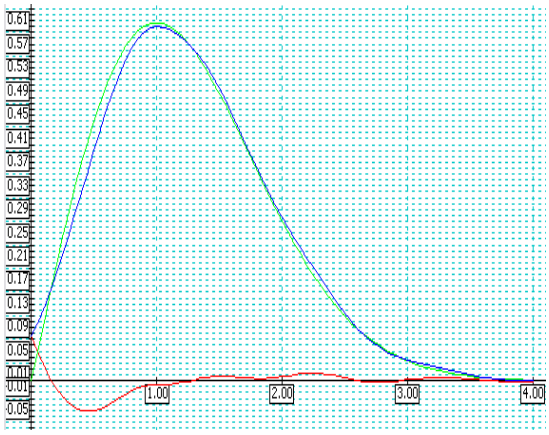


Рис.10 Восстановление ПРВ
при N=1000, C=1,2; θ=0,18.

2.2 При аппроксимирующей функции: $K(y) = 1/2 \cdot \exp(-abs(X - \chi_i) / h_N)$

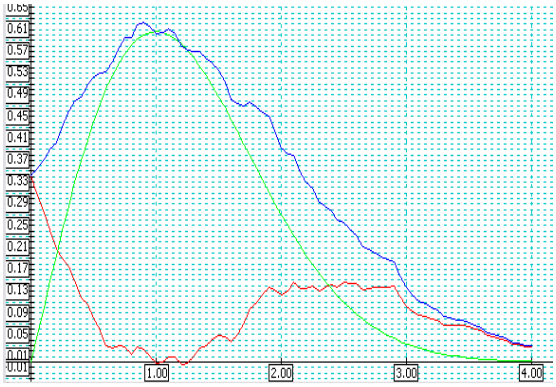


Рис.11 Восстановление ПРВ
при N=100, C=1,2; θ = 0,18.

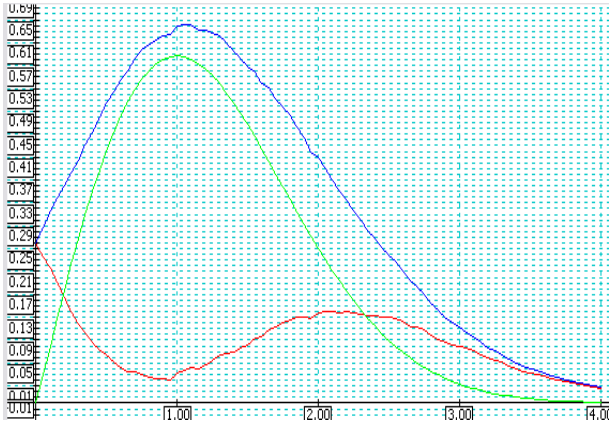


Рис.12 Восстановление ПРВ
при N=1000, C=1,2; θ= 0,18.

2.3 При аппроксимирующей функции $K(y) = 1/2\pi \cdot ((\sin(X - \chi_i) / 2h_N) / (X - \chi_i) / 2h_N)^2$

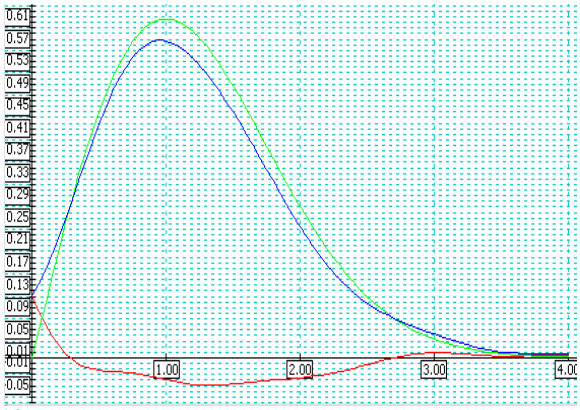


Рис.13 Восстановление ПРВ
при N=100, C=1,2; θ= 0,18.

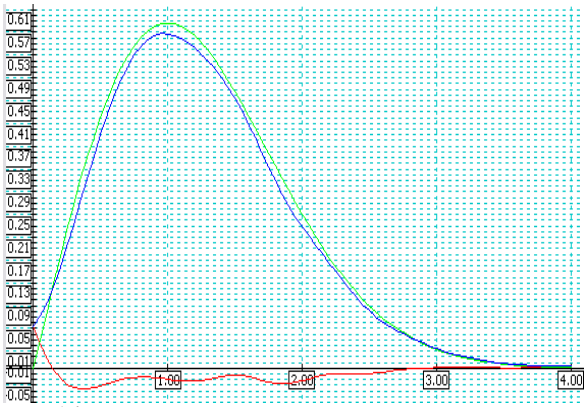


Рис.14 Восстановление ПРВ
при N=1000, C=1,2; θ= 0,18.

2.4 При аппроксимирующей функции: $K(y) = 1/\pi \cdot (1/(1 + (X - \chi_i) / h_N)^2)$

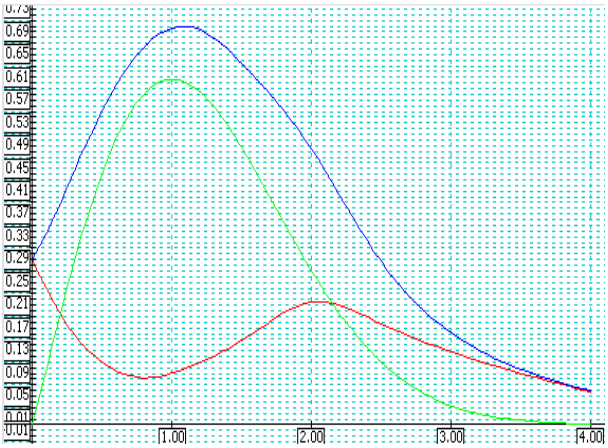


Рис.15 Восстановление ПРВ
при N=100, C=1.2; θ = 0,18.

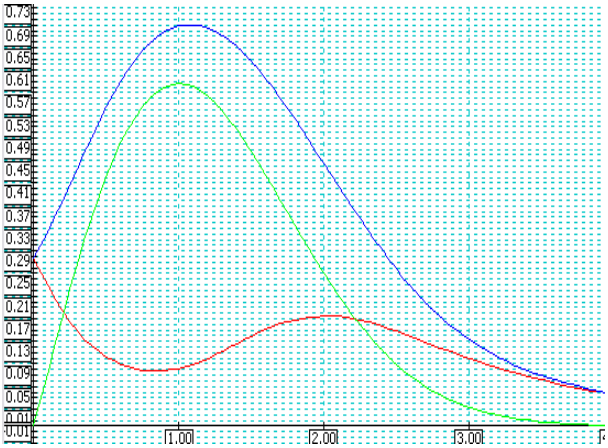


Рис.16 Восстановление ПРВ
при N=1000 , C=1,2; θ = 0,18.

Анализ графиков рис. 9-16 свидетельствут о том, что, как и выше наибольшая точность восстановления релеевской ПРВ достигается при аппрокимирующей функции вида (4).

3. Оценка точности восстановления случайных значений, подчиняющихся логарифмически-нормальной плотности распределения вероятностей:

3.1 <u>При аппроксимирующей функции</u>: $K(y) = (1/\sqrt{2\pi}) \cdot \exp(-(X - \chi_i)^2 / h_N{}^2)$

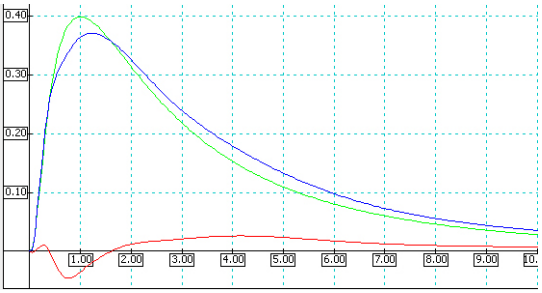


Рис.17 Восстановление ПРВ
при N=100; C=1,2, θ= 0.18.

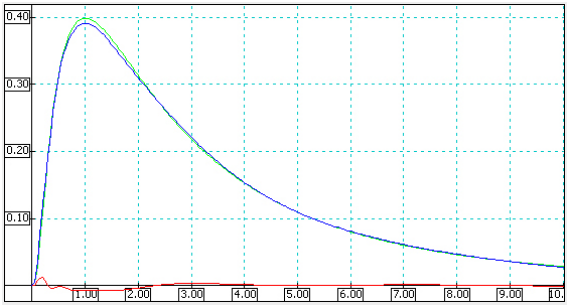


Рис.18. Восстановление ПРВ при
N=1000; C=1,2; θ= 0,18.

3.2 <u>При аппроксимирующей функции</u>: $K(y) = 1/2 \cdot \exp(-abs(X - \chi_i) / h_N)$

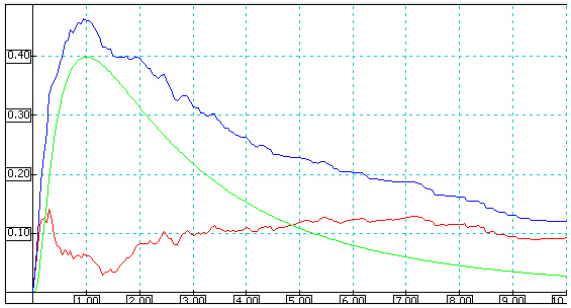


Рис. 19 Восстановление ПРВ при
N=100; C=1,2; θ= 0.18.

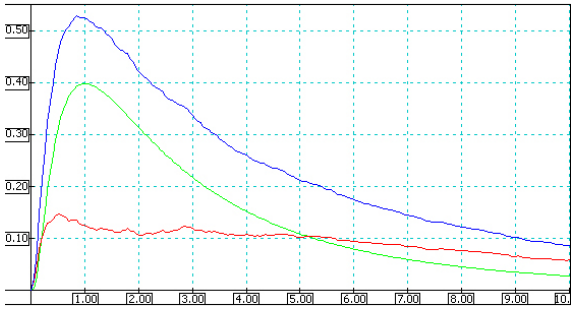


Рис. 20 Восстановление ПРВ при
N=1000; C=1,2; θ= 0.18.

3.3 <u>При аппроксимирующей функции</u> $K(y) = 1/2\pi \cdot ((\sin(X - \chi_i) / 2h_N) / (X - \chi_i) / 2h_N)^2$

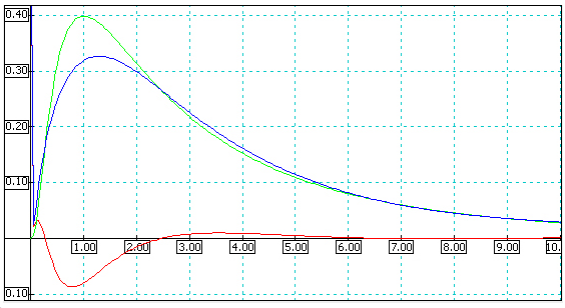


Рис.21 Восстановление ПРВ
при N=100; C=1,2; θ= 0,18.

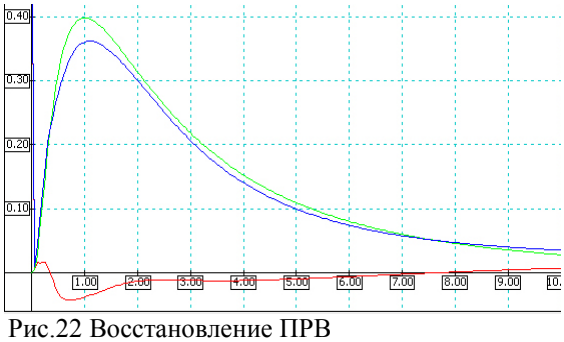


Рис.22 Восстановление ПРВ
при N=1000; C=1,2; θ= 0,18.

3.4 При аппроксимирующей функции: $K(y) = 1/\pi \cdot (1/(1+(X-\chi_i)/h_N)^2)$

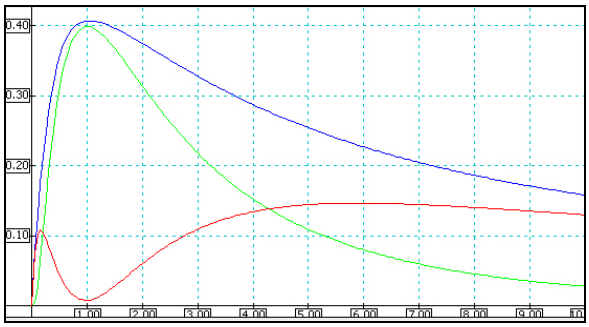


Рис.23 Восстановление ПРВ при N=100; C=1,2; θ= 0,18.

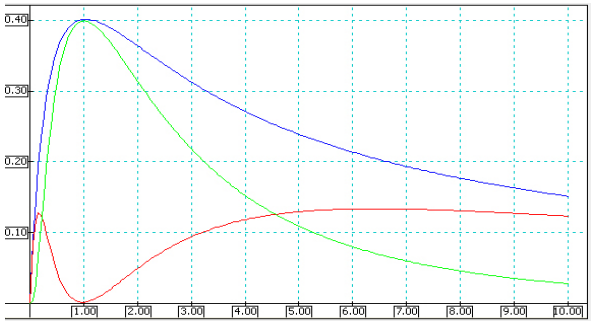


Рис.24 Восстановление ПРВ при N=1000; C=1,2; θ= 0,18.

Анализ графиков рис. 17-24 подтверждает о том, что наилучшее качество восстановления ПРВ, подчиняющейся логарифмически-нормальному закону, также при использовании аппроксимирующей функции вида (4).

В ходе имитационного моделирования дополнительно проведена оценка времени, необходимого для восстановления исследуемых плотностей распределения вероятностей. В результате получено, что для выборки объемом 100 отсчетов это время не превышает 10 мс, а для выборки, объемом 1000 отсчетов - 20 мс. независимо от восстанавливаемого закона, что вполне приемлемо для современных телекоммуникационных сетей.

Вывод

В работе обоснован выбор параметров Парзеновской процедуры восстановления неизвестной плотности распределения вероятностей. Достоинства парзеновской процедуры в том, что она позволяет обеспечить получение не дискретной, как при использовании метода гистограмм, а сглаженной кривой оцениваемой ПРВ в автоматизированном режиме.

В ходе имитационного моделирования установлено:

- при оптимально подобранных параметрах аппроксимирующего ядра вида (4) и объеме выборки равной 1000 с надежностью 0,997 восстанавливается реальная плотность распределения вероятностей.
- с высокой достоверностью оценивается ПРВ нормального закона распределения (рис.2), (не только мода, но и ее «хвосты»), что дает возможность изучить в дальнейшем алгоритмы классификации сигналов и помех классическим методом максимального правдоподобия при использовании данных не многолетних данных статистических испытаний, а текущей статистики, обработанной методом Парзена – Розенблатта;
- независимо от закона распределения при объеме выборки, равном 1000, с высокой точностью оценивается наиболее вероятное значение сигнала (мода), что позволяет в будущем классифицировать наличие помех на основе сдвига не выборочного среднего, а моды, как наиболее вероятного значения;
- анализ скорости проводимых вычислений показывает, что это время, равное 20 мс, что меньше времени, необходимого для работы преобразователя речи.

Таким образом, предложенный подход может быть эффективно применен как при проведении статистических испытаний в период опытной эксплуатации, так и при построении радиоприемных устройств, работающих в условиях непараметрической неопределенности относительно статистических характеристик сигналов и помех.

Литература

1. Басс Ф. Г., Брауде С. Я. И др. Флуктуации электромагнитных волн в тропосфере при наличии поверхности раздела, УФН, т. XXIII, вып.1, 1967, с.89-114.

2. Гусятинский И. А., Немировский А. С. И др. Дальняя тропосферная радиосвязь. – М.: Связь,1968, 246 с.

3. Немировский М.С., Шорин О.А., Бабин А.И., Сартаков А.Л. -Беспроводные технологии от последней мили до последнего дюйма: Учебное пособие.- М.: Эко - Тренз, 2010.- 400с.

4. Орлов А. И., Прикладная статистика. – М.: Экзамен, 2004. -250 с.

5. Репин В.Г. Обнаружение сигнала с неизвестными моментами появления и исчезновения // Пробл.передачи информ. 1991. Т. 27. № 1. - с. 61-72.

6. Смирнов Н.В. Асимптотическая мощноть некторых непараметрических критериев // труды Всесоюзного совещания по математической статистике. Ереван, 1960.

7. Тропченко А Ю., Тропченко А.А. Цифровая обработка сигналов. Методы предварительной обработки. Учебное пособие по дисциплине "Теоретическая информатика". – СПб: СПбГУ ИТМО, 2009.

8. Украинцев Ю. Д., Украинцев К. Ю., Сравнительный анализ парзеновских (непараметрических) процедур восстановления ПРВ. В Сб. «Современные проблемы создания и эксплуатации радиотехнических систем».труды шестой научно – практической конференции (с участием СНГ). Ульяновск, 2009, стр. 233-236.

9. Parzen, On estimation of a probability density function and mode, Ann. Math. Statist. 33, 3 (1962), с. 1065-1-76.

10. M. Rosenblatt, Remarks on some nonparametric estinates of a density function, Ann. Math/ Statist, 27,3 (1956), с 832-837.

УДК 378.4:004-051

ПОДГОТОВКА ИТ-СПЕЦИАЛИСТОВ В УНИВЕРСИТЕТЕ

Е.Г.Чекал, А.А.Чичев

Ульяновский государственный университет

Аннотация

Отечественная промышленность упрекает ВУЗы за слабую подготовку ИТ-специалистов. В статье определяется, почему это происходит и формулируется ответ: как сделать так, чтобы выпускники ИТ-специальностей реально обладали нужными умениями и навыками — перейти на лабораторный метод обучения ИТ-специалистов.

Предисловие

Что нас подвигло на написание статьи. То, что очень много говорят о проблемах высшего образования, что готовят неудовлетворительно, не соответствующим образом, что ВУЗы не дают практических навыков и т. д., в т. ч. говорят весьма серьёзные люди [9], проводятся многочисленные конференции о проблемах высшего образования (например, под эгидой АПКИТ [8]), но никто не говорит о том, КАК организовать процесс обучения в высшей школе так, чтобы независимо от субъекта (прежде всего, от преподавателей, студентов пока оставим в покое) из стен ВУЗа выходил достаточно квалифицированный специалист, про которого бизнес уже не мог бы сказать, что он не соответствует. Как сделать так, чтобы процесс обучения гарантированно давал нужный результат?

1. Постановка вопроса

1.1. Гетерогенная сетевая среда

В настоящее время в организациях/фирмах все более широко используются различные операционные среды. Эта тенденция стала особенно заметной в результате:

a) реализации пилотного проекта внедрения СПО (свободного программного обеспечения) в школах, в частности, в том числе и в Ульяновской области ряд школ теперь использует СПО, тем самым обеспечивая подготовку пользователей в другой операционной среде. Это стало заметно по знаниям и умениям абитуриентов: если 4-6 лет назад в группе (20 человек) было 1-3 студента, работающих в Linux, то в настоящее время количество таковых увеличилось до 3-5 человек на группу. Конечно, этого всё равно мало — ведь, речь идёт о подготовке специалистов по ИТ (информационным технологиям), но сдвиг есть;

b) успехов в борьбе с пиратством — уровень пиратства в РФ в последние годы неуклонно снижается и в экономике становится не только опасно пользоваться нелицензионным ПО, но и просто неприличным;

c) осознания бизнесом факта, что, оказывается можно не платить за то, за что платить не надо, что оказывается есть альтернативные средства поддержки информационных технологий в бизнесе, базирующихся на открытом и бесплатном ПО и что цена вопроса — примерно миллиард долларов (только общесистемного ПО РФ закупает примерно на $1.5 млрд в год, из них $1 млрд - вполне мог бы быть заменён на СПО, а, ведь, есть ещё и прикладное, которого закупается ещё на примерно $2 млрд в год [1,2,3]) — это деньги, которые точно уходят из страны и никогда назад не вернутся.

Эта тенденция получила формальное обоснование после выхода распоряжения Правительства РФ №2299-р от 17.12.10 года, которое фактически определило гетерогенность (многоплатформность) корпоративной операционной среды.

А это означает, что ИТ-специалист должен уметь создавать и поддерживать такую многоплатформную корпоративную операционную среду.

Кроме того, в настоящее время ИТ-специалисты должны владеть достаточно хорошими навыками в области сетевых технологий, ибо автономных компьютеров (не подключенных к

сети) в корпоративной среде практически не стало. Но исторически сложилось так, что основой корпоративной сетевой инфраструктуры в настоящее время является стек протоколов TCP/IP — штатный unix'овый стек протоколов. Именно на нём (и соответственно, на ОС unix) базируется вся сетевая инфраструктура современной экономики. И даже там, где в компаниях/организациях используется ОС Windows (в том числе AD), на самом деле эта ОС является всего лишь frontend'ом (графической оболочкой) для конфигурирования unix'ового стека протоколов.

Следовательно, подготовка специалистов по ИТ должна осуществляться в гетерогенной сетевой среде.

1.2. Подготовка ИТ- и не ИТ-специалистов

Требования образовательных стандартов определяют подготовку ИТ-специалистов по трём уровням компетентности:

- знания — теоретическое знание вопроса,
- умения — студент должен теоретически знать «как сделать и почему так» и по крайней мере один раз «сделать так» на практических занятиях,
- навыки — студент должен понимать, почему надо «делать так» и несколько раз «сделать так» на практических занятиях.

Именно второй и особенно третий уровни компетентности обычно требует работодатель при найме сотрудника, когда в объявлении о вакансии говорит о необходимости наличия у кандидата 2-3-летнего опыта работы в определённой предметной области.

Формально направления подготовки специалистов с высшим профессиональным образованием определяются соответствующими образовательными стандартами.

Понятно, что программы обучения специалистов для разных предметных областей (для разных специальностей) различны. Но образовательные стандарты определяют подготовку специалистов не только по профильным дисциплинам (примерно треть объёма подготовки в часах), но также по достаточно широкому перечню других безусловно важных предметов (две трети объёма подготовки в часах). Причём, даже профильные дисциплины охватывают не только узкую предметную область специальности, но и соседние (смежные) предметные области. Также понятно, что подготовка по профильным дисциплинам должна быть более глубокой, а подготовка по соседним (смежным) предметным областям менее глубокой, скажем так, ознакомительной.

Так ИТ-дисциплины в настоящее время читаются всем специальностям, но очевидно, что подготовка ИТ-специалистов и не ИТ-специалистов в части информационных технологий должна существенно различаться:

1) Для не ИТ-специалистов вычислительная техника, сети и системное ПО — всего лишь орудие труда, как авторучка, главное, чтоб писала; то есть, назначение ПЭВМ — выполнять прикладную программу, автоматизирующую деятельность в некоторой предметной области, а характеристики ЭВМ, ОС, сетевая инфраструктура и системное ПО — за пределами профессиональных интересов.

2) Наоборот, для ИТ-специалиста аппаратная часть ЭВМ, ОС, сетевая инфраструктура, системное ПО — предмет труда и полностью находится в сфере профессиональных интересов.

Следовательно, подготовка ИТ-специалистов должна осуществляться в условиях полной доступности не только для преподавателей, но и для студентов учебной операционной среды для изменения, конфигурирования.

Вывод: подготовка специалистов по ИТ должна осуществляться в гетерогенной сетевой среде в условиях полной доступности операционной среды для изменения, конфигурирования.

3. Текущее состояние: «как есть»

Однако, в настоящее время, технология подготовки ИТ- и не ИТ-специалистов в части информационных технологий в большинстве ВУЗов России практически не различается. И тех, и других информационным технологиям обучают в обычных компьютерных классах в условиях жёстко контролируемой конфигурации аппаратных и программных средств. Жёсткость операционной среды обусловлена статусом компьютерного класса — это разделяемый ресурс и в соответствии с утверждённым расписанием он после завершения одного занятия по некоторой дисциплине должен быть почти немедленно готов к проведению другого занятия по другой дисциплине. Следовательно, в процессе проведения занятия операционную среду класса нельзя изменять и тем более разрушать. Изменять операционную среду может только администратор во время регламентных работ, а если преподавателю необходимо для занятия нечто специфическое, то он должен оговаривать это заранее, получать разрешение и ждать воплощения задуманного по меньшей мере до следующего регламентного периода.

Отсюда следует вывод о невозможности полноценной подготовки ИТ-специалистов в условиях, когда учебный процесс ориентирован на использование обычных компьютерных классов, поскольку таким образом организованный учебный процесс не обеспечивает получения студентами ИТ-специальностей не только навыков, но даже умений в их предметной области.

Что мы и наблюдаем в реальности и отсюда претензии работодателей к слабой подготовке выпускников ВУЗов по ИТ-специальностям [4, 7, 8].

4. Причины: почему «так есть»

Нынешняя ситуация с подготовкой всех студентов без различия специальностей в части информационных технологий в обычных компьютерных классах сложилась в 90-е годы из-за слабого финансирования ВУЗов и, соответственно, из-за невозможности приобретать на регулярной основе современное оборудование и ПО, которое к тому же очень быстро устаревает. Исключением являются физики, химики и другие «древние» практические специальности, появившиеся давно. За сотни лет их существования утвердилось понимание того, что подобных специалистов нужно готовить в лабораторных условиях и даже проблемы с финансированием в 90-е годы не разрушило этого понимания. А ИТ-специальности появились в ВУЗах как раз в 90-е годы. И, несмотря на то, что эти новые специальности тоже являются весьма практическими (кому нужен ИТ-специалист — теоретик?) из-за недофинансирования ВУЗы вынужденно реализуют подготовку студентов в одинаковых условиях без учёта специфических требований к специальностям. За прошедшие 20 лет это состояние (недофинансирование) стало привычным и, соответственно, привычным стал и технологический процесс обучения — уже никто не задумывается (и не вспоминает) о том, что вообще-то, процесс обучения ИТ- и не ИТ-специалистов должен быть различным. Сформировалась традиция.

Причём, эта «привычность» и, соответственно, непонимание причин, почему «так есть» - тотальное:

- как со стороны образовательных структур (структур поставляющих: преподавателей, административных структур ВУЗов, Министерства образования);
- так и со стороны бизнес-сообщества (структур потребляющих), которое вроде как говорит, что готовят плохо и даже иногда указывает, где (в каком месте образовательного процесса готовят плохо [4]), но не говорит почему.

И если непонимание со стороны образовательных структур ещё можно как-то понять (нет же денег на образовательный процесс и зарплаты преподователей лучше не говорить какие), то непонимание со стороны бизнес-сообщества выглядит, как бы это помягче сказать, несколько цинично: может денег жалко? Ведь, создание и оснащение лабораторий — затратное дело.

5. Как должно быть

Подготовка ИТ-специалистов должна быть такой же, как подготовка математиков, физиков:

- студент станет математиком/физиком тогда и только тогда, когда он самостоятельно своими пальчиками и своим разумом научится что-то выводить, доказывать, обосновывать, ставить эксперименты и анализировать результаты;
- аналогично, ИТ-специалист станет таковым тогда и только тогда, когда он своими пальчиками и своим разумом научится создавать, настраивать, устанавливать, конфигурить, сопровождать корпоративную операционную среду.

Но подобное возможно только в условиях лабораторного обучения, в условиях, когда даны компьютеры, другое оборудование, ПО и некоторая базовая сетевая инфраструктура, а студент должен на этой основе установить, настроить, сконфигурить, создать и исследовать что-то своё.

При этом учебный процесс организуется так, чтобы максимально приблизить его к реальным условиям, в которых окажется выпускник университета на своей будущей работе. То есть, в лабораториях моделируется, как можно ближе к реальности, будущая профессиональная деятельность ИТ-специалиста. Для этого при выполнения практических занятий студенту предоставляется полный доступ к оборудованию и программному обеспечению в лабораториях (права root'а/администратора) и он должен с максимальной самостоятельностью выполнять задания.

То есть, для формирования навыков (приобретения опыта работы) студент должен активно работать в лабораториях университета. Это полезно также и потому, что в фирмах/организациях (на месте будущей работы) лишнего оборудования, как правило и почти всегда, нет, и, следовательно, изучить/освоить что-то ещё, кроме своих штатных обязанностей в соответствии со своей должностной инструкцией практически невозможно. А в лабораториях университета доступен (должно быть так) достаточно широкий спектр оборудования и ПО.

Однако, возникает вопрос: где деньги взять на оснащение лабораторий?

6. Решение

Поэтому у решения есть две составляющих: финансовая и технологическая.

Финансовая составляющая определяет, где деньги брать на оснащение лабораторий.

Технологическая составляющая определяет, как должен быть организован учебный процесс.

6.1. Решение западное

Оно очень простое. В ВУЗах развитых стран проблемы финансирования в значительной мере решаются посредством организации спонсорской помощи. Бюджет самых известных и престижных ВУЗов, например, США формируется в значительной степени за счёт спонсорской помощи бывших выпускников (до 45 процентов за счёт поступлений из endowment-фондов (целевых фондов) плюс прямая спонсорская помощь) [5,6]. Для выпускника МИТа, Гарварда, Беркли является нормальным действием помочь alma mater. Более того, подобные действия в обществе воспринимаются положительно — объём оказываемой спонсорской помощи родному ВУЗу является косвенным критерием успешности выпускника.

Этому вопросу уделяется большое внимание и в ВУЗах в процессе обучения обеспечивается формирование у студентов необходимого мировоззрения и убеждений, в частности убеждения в том, что именно здесь, в стенах ВУЗа, их «делают», закладывают основы их карьеры, успеха в бизнесе, их будущего процветания.

Аналогично, отработана и технологическая часть решения: не секрет, что значительная часть проектов в СПО и OpenSource (в том числе, почти все крупные проекты) разрабатываются в лабораториях западных университетов. Эти проекты являются долговременной (переходящей) базой для подготовки ИТ-специалистов.

Кроме того, лаборатории являются базой для выполнения коммерческих проектов, которые также дают значительный вклад в бюджет ВУЗов.

6.2. Решение российское

К сожалению, «западный» метод решения финансовых проблем у нас в стране законодательно только начинает регламентироваться (закон о целевых фондах принят только в 2004 году) и в ВУЗах почти не получил распространения за очень редким исключением (крупнейший эндаумент-фонд СПбГУ по состоянию на конец 2011 года равен примерно 600 млн рублей [6], на втором месте фонд МГИМО — примерно 20 млн рублей, Финансового университета — примерно 10 млн рублей и т. д.; целевой фонд УлГУ, как и сотен других ВУЗов — 0 рублей; и это никак несравнимо с эндаумент-фондами ВУЗов США в миллиарды и десятки миллиардов $).

Прямая спонсорская помощь также практически отсутствует.

В силу отсутствия лабораторной базы ВУЗы не способны выполнять и коммерческие проекты — нет оборудования и, соответственно, нет и специалистов, способных его использовать, следовательно и заработать на выполнении коммерческих проектов ВУЗ не может. Поэтому разговоры о переносе исследований и разработок в ВУЗы в условиях отсутствия лабораторной базы — это лишь пока благие намерения.

То есть, своего российского решения финансовой составляющей нет. И причин этому несколько.

1. Уровень жизни. В результате развала 90-х народонаселение РФ обеднело на порядок: если уровень жизни советского среднеульяновца середины 80-х был только в 2-3 раза ниже уровня жизни среднеамериканской глубинки, то современный среднеульяновец имеет уровень жизни примерно в 20 (двадцать) раз ниже уровня жизни современной среднеамериканской глубинки. Какая уж тут спонсорская помощь.
2. Менталитет выпускника. Да в мыслях нет у успешного выпускника российского ВУЗа (кем бы он ни стал, как бы он не был обеспечен) оказать спонсорскую помощь alma mater. И не только потому, что в ВУЗе ему не «привили» мысль о том, что помочь родному ВУЗу — это хорошо, а ещё и потому, что выпускник российского ВУЗа, как правило, «воображает», что это он САМ, самостоятельно достиг своего успеха, своего богатства, а ВУЗ. . ., ну что, ВУЗ. . ., ВУЗ — это было давно, да и учили там совсем не тому. . .
3. Менталитет российского бизнеса. В западной экономике вновь появившиеся деньги, как правило, реинвестируются, руководствуясь принципом: «деньги должны делать деньги». Но в российском бизнесе вариант реинвестиции в производство применяется не часто. Об этом свидетельствует статистика МинФина о количестве вывозимых миллиардов $.
4. У студентов ВУЗов не формируется убеждение о том, что именно здесь, в стенах этого ВУЗа, из них готовят элиту общества, самую образованную, самую продвинутую часть общества, что именно здесь, в ВУЗе, им дают ВЫСШЕЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАНИЕ, которое станет основой, базой их будущего успеха как технических специалистов, или поможет занять подобающее место в административных структурах, или послужит основой успешного бизнеса. Вся их будущая жизнь, весь их будущий успех как технических специалистов, как руководителей, как бизнесменов закладывается в эти 4-6 лет учёбы.

А сами они до этой мысли редко доходят и, соответственно, не появляется мысль о том, чтобы сказать родному ВУЗу спасибо.

Таким образом, ВУЗам не приходится рассчитывать ни на спонсорскую помощь, ни на возможность самостоятельно заработать деньги на оснащение лабораторий.

Поэтому пока приходится только «уповать»

- либо на целевое финансировании государства на лабораторное обучение (Вы знаете о таком? Мы — нет. Например, в новых образовательных стандартах даже слов таких нет о необходимости наличия в ВУЗах лабораторий при подготовке ИТ-специалистов);

- либо на сознательность руководства ВУЗа — это более реально, но. . .

6.3. Решение методом виртуализации и его минусы

Технологическая часть решения иногда реализуется с помощью технологии виртуализации. Однако, по мнению авторов, это не позволяет достаточно близко к реальности смоделировать будущую профессиональную деятельность в силу явной искусственности порождаемой операционной среды: студент, конечно, получает знания и даже навыки по созданию и управлению виртуальными средами, но в реальной экономике в среднем и малом бизнесе, где чаще всего и оказывается выпускник, технология виртуализации используется не часто. То есть, при таком подходе студент опять же не выполняет полностью весь спектр работ, с которыми он столкнётся в будущем.

К тому же при таком подходе также возникают существенные ограничения по доступу к оборудованию и базовой операционной среде. Да и не любые задачи по созданию корпоративной среды можно решать с помощью виртуализации.

Кроме того, организация учебного процесса с использованием технологии виртуализации существенно более требовательна к аппаратным средствам: требует достаточно мощного и, соответственно, более дорогого оборудования.

6.4. Полное лабораторное решение и его минусы

Предлагаемый подход к формированию практических навыков будущих ИТ-специалистов реализуется посредством организации учебного процесса в научно-практических лабораториях университета: «Лаборатории аппаратных средств Информационных Систем» кафедры ТТС ФМиИТ и «Лаборатории операционных систем и системного программирования» кафедры ИТ ФМиИТ.

При этом одной из целей обучения является формирование у студентов умений и навыков установки, настройки, сопровождения и использования:

1) нескольких операционных сред на компьютеры лабораторий — важно, потому, что современная корпоративная среда — гетерогенная;
2) корпоративной среды и в ней сетевых сервисов — важно, потому, что современная корпоративная среда — мультисервисная;
3) приложений и мультиплатформных сред разработки — важно, потому что ИТ-специалист должен обеспечивать поддержку прикладного ПО и уметь разрабатывать его для разных операционных сред;
4) разработки распределённых приложений в архитектуре клиент-сервер, в частности, SOA — самой распространённой сейчас клиент-серверной архитектуре.

Даже самый простой в смысле требований к организации учебного процесса пункт 4 уже с трудом и не полностью может быть реализован в обычном компьютерном классе, а пункты 1-3 вообще невозможно выполнить.

Сложность заключается ещё и в том, что состав лабораторных и курсовых работ значительно меняется каждый год и все работы используют СПО и, если не оговорено специально, то ОС linux.

Минусы данного метода решения:

- оно является более дорогостоящим даже в самых простых реализациях, поскольку требует более многочисленного, разнообразного оборудования (правда, не обязательно нового), нежели используемое в обычном копьютерном классе, где оборудование должно быть почти всегда новым, но простым, единообразным, унифицированным (однако в настоящее время наши лаборатории комплектуются по «остаточно-списательному принципу»: то есть, компьютеры и оборудование, что уже не могут использоваться в компьютерных классах с ОС Windows, частично передаются в лаборатории (например, сейчас в лаборатории «ОС и системного программирования» кафедры ИТ находятся 20 компьютеров 2004 года выпуска с CRT-мониторами); то есть, за три года существования лабораторий они обошлись ВУЗу в весьма незначительную сумму);

- оно требует соответствующей подготовки преподавателей [4]: они должны быть реальными практическими специалистами с опытом работы; а реального специалиста с небольшой, но хорошей зарплатой в несколько тысяч $ не прельстишь зарплатой преподавателя.

Замечания к решению.

Замечание 1. Иногда возникает вопрос об эффективности использования аудитории/класса/оборудования. Как правило, он формулируется в таком виде: «Аудиторий/компьютерных_классов не хватает, а у вас «недогруз» по часам».

Контрвопрос. А что важнее для университета: качество подготовки студентов или «загрузка» по часам?

Ответ на этот контрвопрос фактически является ответом на вопросы о рейтинге, имидже и престижности ВУЗа, о том, кем станут его выпускники, кого и для кого ВУЗ готовит и, вообще, дальше-то ВУЗ будет существовать?

Замечание 2. Выбор программного обеспечения. Оно безусловно должно быть открытым, более того, крайне желательно, чтобы оно было класса Open Source. Это обусловлено следующим:

- главный и основной принцип обучения - «Делай как я!», OpenSource этому удовлетворяет; то есть, обеспечивается доступность исходников очень высокого качества, которые можно использовать в качестве примера и образца в дисциплинах программирования, всегда присутствующих в программах подготовки ИТ-специалистов; учитывая чрезвычайно высокую сложность ПО современных систем — данный пункт очень важен. То есть, студенты реально знакомятся с лучшими примерами разработки сложных систем. Также Open Source позволяет идти от достигнутого, развивая проекты дальше и выше;
- на проприетарное ПО денег точно нет: «кто не верит — пусть проверит» - подсчитайте стоимость проприетарного ПО (обязательно профессионального — мы же специалистов готовим), необходимого для выполнения лабораторных и курсовых работ, которые к тому же нужно менять ежегодно.

Замечание 3. Кроме того, общесистемное ПО должно быть достаточно хорошо локализованным. Это важное требование — следствие слабого знания студентами технического английского; за последние два десятилетия на кафедрах английского языка сложилось весьма превратное представление о том, как надо учить английскому студентов технических специальностей: вместо перевода и прежде всего перевода и в третьих, письменной речи — обратного перевода (это обязательно потребуется техническому специалисту и это требуется уже на этапе обучения), студентов учат разговорному английскому, который техническому специалисту понадобится разве что для того, чтобы слетать в отпуск в Грецию (или для того, чтобы эмигрировать?).

Данные замечания не могут определять выбор общесистемного ПО для подготовки не ИТ-специалистов — для них наличие исходников не является определяющим фактором, а прикладное ПО, как правило, локализовано хорошо.

6.5. Производственная и преддипломная практика и стажировка

Кто-то скажет: «Есть производственная практика, в рамках которой . . .». А кто-то другой скажет: «А у нас применяется метод стажировки студентов в заинтересованных фирмах по месту будущей работы, начиная с 4-5 курсов . . .».

Рассмотрим эти «недоальтернативы».

6.5.1. Производственная практика — как она проходит

Студенты появляются с направлениями в фирмах/организациях. Им рассказывают о внутреннем распорядке, о целях и направлениях деятельности организации, об истории фирмы (в музей сводят, если он есть), возможно даже «прикрепят» к штатным сотрудникам. Но эти «закреплённые» сотрудники заняты делом, а что поручишь неграмотным студентам? Учить их — времени нет, да и не все умеют и хотят передавать знания, а план надо делать,

денежки зарабатывать, за этих «стажёров» редко платят, а если и платят, то мало. Итог: студент-практикант - обуза. Придёт он раз-два-три. Никто им не интересуется. И всё. Через два месяца он приходит за справкой о прохождении производственной практики, которую ему с радостью выдают. Мы не утрируем и не упрощаем — это действительно так происходит.

Где здесь получение умений и навыков?

То есть, производственная практика — это пережиток советской системы образования, в которой она действительно работала в условиях плановой экономики. А сейчас другие условия и в них она уже ничего не даёт — превратилась в фикцию.

6.5.2. Преддипломная практика более эффективна, особенно если студент проходит её по месту будущей работы. Она позволяет сэкономить несколько месяцев, которые обычно уходят на адаптацию нового сотрудника. Но что можно изучить и освоить за два месяца?

6.5.3. Стажировка в фирмах/организациях по месту будущей работы. В этом случае студенты, как правило, реально работают, изучают технологии фирмы, заняты в проектах. То есть, налицо реальное получение умений и навыков.

Но здесь возникает другая проблема: студенты настолько «погружаются» в фирму — почти штатный сотрудник и хочется себя проявить, что они очень часто «кладут» на учёбу (особенно на непрофильные предметы): начинаются пропуски занятий, иногда 100-процентные, появляются только в сессию: «А я работаю . . .». Естественно, учебный план они не выполняет, знания по большинству предметов — едва на тройку, кое-как дотягивают до диплома и получается что-то узкоспециализированное под фирму, «подобное флюсу» ((С)К. Прутков).

Это хорошо? Не очень. В конкретной фирме ему работать не вечно. Знания других (пропущенных) дисциплин однажды потребуются — недаром они в программе. То есть, студенты очень часто фактически себя «зарывают», урезая свой будущий потенциал специалиста с ВПО.

7. Что получилось

Наши дипломники хорошо устраиваются и работодатель ими доволен. Три года подряд наши студенты (по нашим курсовым) выигрывают студенческие гранты нашего университета. Совместно со студентами за три года опубликовано 10 научных статей. Четырежды студенты докладывали о своих разработках на международных и всероссийских конференциях. ГЭК по специальности «Информационные системы» (ИС) отмечает улучшение знаний по общему ПО и сетям (студенты этой специальности у нас в лаборатории работают несколько семестров, поэтому на них сказывается влияние лабораторного обучения больше всего).

Эти результаты появились вдруг, когда мы перешли на лабораторное обучение по своим дисциплинам.

8. Заключение

В результате многолетних размышлений о причинах слабой подготовки в ВУЗах специалистов по ИТ авторы пришли к выводам:

1. По оценкам авторов на каждые 2-3 читаемых на кафедре дисциплины должна быть как минимум одна лаборатория.
2. Ответственными за лаборатории (не обязательно материально ответственными) должны быть преподаватели, которые ведут занятия в данной лаборатории.
3. Лаборатории должны являться базой/основой для разработок и исследований по соответствующим дисциплинам специализации лабораторий, в том числе, базой для выполнения сторонних заказов/проектов (если преподаватель может и хочет).
4. В оснащении лабораторий должен участвовать бизнес как заинтересованная сторона.

5. Для формального закрепления лабораторного обучения ИТ-специалистов необходимо изменить Государственные образовательные стандарты по специальностям ИТ в части определения требований к учебной базе.

6. В лабораториях не требуются лаборанты: все работы должны выполняться преподавателем (это его практика) и студентами (это их умения и навыки).

Таким образом, наличие лабораторной базы, подбор лабораторных и курсовых работ в соответствии с потребностями будущей работы выпускников, привлечение студентов к выполнению проектов поможет ВУЗам реально повысить уровень компетентности выпускников.

Литература

1. http://www.cnews.ru/news/top/index.shtml?2011/06/30/445903
2. http://www.erumpo.ru/about/tenders/review_of_software_market
3. http://www.expert.ru/graphs/expert/2008/06/document371107
4. http://www.infosecurity.ru/cgi-bin/cart/arts.pl?a=060825&id=156754
5. http://www.fondperspectiva.ru/?id=622
6. http://jourrnal.spbu.ru/?p=5387
7. http://www.apkit.ru/commitees/education/meetings/standarts.php
8. http://2012.ит-образование.рф/upload/IT-EDUCATION-2012-book.pdf
9. Садовничий В. Уровень подготовки ИТ-специалистов должен удовлетворять потребности рынка / Виктор Садовничий // Национальные проекты. - 2010. - N 7/8. - С. 72-74.*