МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК

Выпуск 19

ПРОГРАММИРОВАНИЕ, УПРАВЛЕНИЕ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

САНКТ-ПЕТЕРБУРГ
2005

Выпуск содержит материалы XXXIV научной и учебно-методической конференции СПбГУ ИТМО, посвященная 100-летию первого выпуска специалистов вуза
Конференция была проведена 2–4 февраля 2005 г. Санкт-Петербургским государственным университетом информационных технологий, механики и оптики в сотрудничестве с
ВНЦ ГОИ им. С.И. Вавилова,
Институтом аналитического приборостроения РАН,
Институтом проблем машиноведения РАН,
Комитетом по науке и высшей школе Администрации Санкт-Петербурга,
ВНИИМ им. Д.И. Менделеева,
ОАО «ЛОМО»,
ОАО «Техприбор»,
ОАО «Электроавтоматика»,
ЦНИИ «Электроприбор».

ISBN 5-7577-0276-1

1 НЕЛИНЕЙНЫЕ КОЛЕБАНИЯ И ПАРАМЕТРИЧЕСКАЯ ИДЕНТИФИКАЦИЯ МЕХАНИЧЕСКИХ СИСТЕМ

ИССЛЕДОВАНИЕ ДИНАМИКИ НЕЛИНЕЙНОЙ ПРИБОРНОЙ СИСТЕМЫ В УСЛОВИЯХ ПЕРИОДИЧЕСКИХ ВОЗМУЩЕНИЙ

С.Е. Иванов, Г.И. Мельников

Рассматривается математическая модель нелинейной приборной системы. Она имеет вид системы дифференциальных уравнений шестого порядка с периодическими параметрами и содержит нелинейные полиномиальные члены до четвертой степени. Предполагается отсутствие в системе внешних и внутренних резонансов. Применяется метод многочленных преобразований с целью упрощения вида динамических уравнений и выделения существенных констант, определяющих качество движения нелинейной системы.

Метод многочленных преобразований [1] является общим методом исследования нелинейных виброзащитных систем и других периодически нестационарных голономных систем с конечным числом степеней свободы. В результате применения метода многочленных преобразований получаем преобразованную автономную систему с точностью, принятой при выводе динамических уравнений. Полученную систему можно рассматривать как исходную систему, но записанную в новых фазовых переменных с допустимой точностью. Метод многочленных преобразований обеспечивает минимизацию количества постоянных параметров в системе дифференциальных уравнений. Преобразованная система содержит значительно меньшее количество ненулевых коэффициентов, чем исходная. Сокращение количества ненулевых коэффициентов существенно упрощает исследование сложной нелинейной системы. С помощью преобразованных систем упрощается задача исследования переходных и установившихся процессов исходных систем.

Рассмотрим математическую модель нелинейной виброзащитной системы с тремя степенями свободы, с правыми частями в виде многочленов до четвертой степени относительно фазовых переменных с постоянными и периодическими параметрами. Дифференциальные уравнения движения рассматриваемой виброзащитной системы имеют вид:

$$Aq+Bq+Cq=\sum_{|\mu|=1}^{4} g_{\mu}\cos(\omega t)^{\mu_1}\sin(\omega t)^{\mu_2}+\sum_{|\nu|=2}^{4} h_{\nu}\cos(\omega t)^{\nu_1}\sin(\omega t)^{\nu_2}q_1^{\nu_3}q_2^{\nu_4}q_3^{\nu_5}q_1^{\nu_6}q_2^{\nu_7}q_3^{\nu_8}, \quad (1)$$

где $q=[q_1,q_2,q_3]^T$ - вектор обобщенных координат системы, A,B,C - матрицы, третьего порядка, $\mu=(\mu_1\mu_2), \nu=(\nu_1\nu_2\nu_3\nu_4\nu_5\nu_6\nu_7\nu_8)$ - векторные индексы,

$|\mu|=\mu_1+\mu_2$, $|\nu|=\nu_1+\nu_2+...+\nu_8$, $g_{\mu}=[g^1{}_{\mu}, g^2{}_{\mu}, g^3{}_{\mu}]^T$ $h_{\nu}=[h^1{}_{\nu}, h^2{}_{\nu}, h^3{}_{\nu}]^T$ – столбцы.

Предполагается, что характеристическое уравнение $Det[A\lambda^2+B\lambda+C]=0$ имеет три пары комплексно сопряженных корней $\lambda_S, \overline{\lambda}_S$ с малыми отрицательными вещественными частями. Положим также, что компоненты вектора нелинейных частей $|g^S{}_{\mu}|<\varepsilon$, $|h^S{}_{\nu}|<\varepsilon$ малы.

Приведем алгоритм метода многочленных преобразований.

Введем комплексно-сопряженные переменные $q_0=\exp(i\omega t)$ и $\overline{q}_0=\exp(-i\omega t)$, $\lambda_1=i\omega$, тогда можно записать

$$\cos(\omega t)=\frac{1}{2}(q_0+\overline{q}_0) \text{ и } \sin(\omega t)=\frac{1}{2i}(q_0-\overline{q}_0). \quad (2)$$

Систему (1) можно представить в виде системы восьми дифференциальных уравнений первого порядка в нормальной форме Коши с фазовым вектором $X = [q_0 \overline{q}_0 q_1 q_2 q_3 \dot{q}_1 \dot{q}_2 \dot{q}_3]^T$

$$\dot{X} = PX + R \tag{3}$$

где постоянная квадратная блочная матрица восьмого порядка имеет вид

$$: P \equiv \begin{bmatrix} W & 0 & 0 \\ 0 & 0 & I \\ A^{-1}H & -A^{-1}C & -A^{-1}B \end{bmatrix}, R = \begin{bmatrix} 0 \\ 0 \\ A^{-1}G \end{bmatrix}, W = \begin{bmatrix} i\omega & 0 \\ 0 & -i\omega \end{bmatrix}$$

$$H = [0.5(g_{10} - ig_{01}) \quad 0.5(g_{10} + ig_{01})],$$

$$G = \sum_{|\mu|=2}^{4} (-i)^{\mu_2} 0.5^{\mu_1+\mu_2} g_\mu (q_0 + \overline{q}_0)^{\mu_1} (q_0 - \overline{q}_0)^{\mu_2} +$$

$$\sum_{|\nu|=2}^{4} (-i)^{\nu_2} 0.5^{\nu_1+\nu_2} h_\nu (q_0 + \overline{q}_0)^{\nu_1} (q_0 - \overline{q}_0)^{\nu_2} q_1^{\nu_3} q_2^{\nu_4} q_3^{\nu_5} \dot{q}_1^{\nu_6} \dot{q}_2^{\nu_7} \dot{q}_3^{\nu_8},$$

Линейным неособым преобразованием вида

$$Y = DX \tag{4}$$

линейная часть системы (3) приводится к диагональному виду:

$$\dot{Y} = \Lambda Y + R\Big|_{X \to D^{-1}Y}, \text{ где } \Lambda = diag[\lambda_1, \overline{\lambda_1}, ..., \lambda_4, \overline{\lambda_4}]. \tag{5}$$

Затем выполняется преобразование, содержащее многочлены четвертой степени включительно:

$$y_S = z_S + \sum_{|\nu|=2}^{4} a_\nu^S Z^\nu, (s = 3,...,8), Z^\nu \equiv z_1^{\nu_1} z_2^{\nu_2} ... z_8^{\nu_8}, \tag{6}$$

где a_ν^S – неизвестные коэффициенты преобразования.

Введенные комплексно-сопряженные переменные не преобразовываются $y_S = z_S \ (s = 1,2)$.

Результатом многочленного преобразования является автономная система:

$$\dot{z}_S = \lambda_S z_S + \sum_{|\nu|=2}^{4} q_\nu^S Z^\nu, \ (s = 3,...,8), \tag{7}$$

где q_ν^S – искомые коэффициенты преобразованной системы.

Особые значения векторного индекса при фиксированном S находятся как целочисленные неотрицательные решения двух уравнений [1,2]:

$$\sum_{k=1}^{8} \lambda_k \nu_k - \lambda_S \approx 0, \ \sum_{k=1}^{8} \nu_k = 2,3,4 \ . \tag{8}$$

Постоянные q_ν^s приравниваем к нулю при неособых значениях индексов; при таких значениях вычисляют постоянные a_ν^s. И, наоборот, при особых значениях индексов коэффициенты a_ν^s полагают равными нулю и вычисляют q_ν^s.

В нерезонансном случае, когда собственные частоты колебаний системы и частота вибрации не совпадают и не кратны, находим следующие особые индексы:

при q_ν^3: $\nu = (00100011), \nu = (00101100), \nu = (00210000), \nu = (11100000),$

при q_ν^5: $\nu = (00001011), \nu = (00002100), \nu = (00111000), \nu = (11001000),$

при q_ν^7: $\nu = (00000021), \nu = (00001110), \nu = (00110010), \nu = (11000010).$

В преобразованной системе (7) сделаем замену переменных:

$$z_S = \rho_S Exp\big(i(t \operatorname{Im} \lambda_S + \theta_S)\big), z_{S+1} = \rho_S Exp\big(i(t \operatorname{Im} \lambda_{S+1} - \theta_S)\big), s = 3,5,7; \ \ z_{1,2} \equiv Exp(\pm it\omega) \tag{9}$$

В результате получаем дифференциальное уравнение:

$$\dot{\rho}_S = \mathrm{Re}(\lambda_S)\rho_S + \mathrm{Re}(\psi_S),\ \ \dot{\theta}_S = \rho_S^{-1}\,\mathrm{Im}(\psi_S),\ \ s = 3,5,7$$

$$\psi_S = \sum_{|\nu|=2}^{4} q_\nu^S \rho_3^{\nu_3+\nu_4} \rho_5^{\nu_5+\nu_6} \rho_7^{\nu_7+\nu_8} Exp(i(\theta_3(\nu_3-\nu_4)+\theta_5(\nu_5-\nu_6)+\theta_7(\nu_7-\nu_8)-\theta_S)) \tag{10}$$

В нерезонансном случае экспонента не входит в систему (10), т.к. её степень равна нулю. Стационарные решения можно найти, приравнивая к нулю правые части системы (10).

Получив решение преобразованной системы (10) и подставив его в формулы замены переменных (9), найдем вектор Z. Вектор Y выражается через вектор Z по формулам многочленной замены (6). Решение системы (1), вектор X, выражается через вектор Y по формулам замены, обратной линейной: $X = D^{-1}Y$.

Получим алгоритмические формулы для расчета коэффициентов преобразования и преобразованной системы.

Запишем систему (5) в переменных Z многочленного преобразования (6)

$$\dot{y}_S = \lambda_S z_S + \lambda_S \sum_{|\nu|=2}^{4} a_\nu^S Z^\nu + R_S(Z) \tag{11}$$

Продифференцируем формулу многочленных преобразований (6), учитывая равенство (7) получим:

$$\dot{y}_S = \lambda_S z_S + \sum_{|\nu|=2}^{4} q_\nu^S Z^\nu + \sum_{|\nu|=2}^{4} (a_\nu^S Z^\nu \sum_{k=1}^{8} \lambda_k \nu_k) + \sum_{|\nu|=2}^{4} (a_\nu^S Z^\nu \sum_{k=3}^{8} \nu_k z_k^{-1} \sum_{|\mu|=2}^{4} q_\mu^k Z^\mu). \tag{12}$$

Из формул (11) и (12) получим равенство:

$$\sum_{|\nu|=2}^{4} q_\nu^S Z^\nu + \sum_{|\nu|=2}^{4} (a_\nu^S Z^\nu (\sum_{k=1}^{8} \lambda_k \nu_k - \lambda_S)) + \sum_{|\nu|=2}^{4} (a_\nu^S Z^\nu \sum_{k=3}^{8} \nu_k z_k^{-1} \sum_{|\mu|=2}^{4} q_\mu^k Z^\mu) = R_S(Z), \quad (s = 3,...,8) \tag{13}$$

Приравнивая в (13) коэффициенты при одинаковых степенях Z, получаем систему алгебраических уравнений для определения неизвестных коэффициентов преобразования и преобразованной системы.

Для записи суммы по векторному индексу в программе использовано представление суммы в следующем виде:

$$\sum_{|\nu|=2}^{4} p_{\nu_1,\nu_2,\nu_3,\nu_4,\nu_5,\nu_6,\nu_7,\nu_8} = \sum_{i_8=2}^{4} \sum_{i_7=0}^{i_8} \sum_{i_6=0}^{i_7} \sum_{i_5=0}^{i_6} \sum_{i_4=0}^{i_5} \sum_{i_3=0}^{i_4} \sum_{i_2=0}^{i_3} \sum_{i_1=0}^{i_2} p_{i_1,i_2-i_1,i_3-i_2,i_4-i_3,i_5-i_4,i_6-i_5,i_7-i_6,i_8-i_7} \tag{14}$$

На базе символьных преобразований многочленов была разработана программа для исследования методом многочленных преобразований нелинейных виброзащитных систем с тремя степенями свободы вида (1) и выполнены расчеты при заданных числовых значениях констант [3].

Рассмотрим виброзащитную систему с тремя степенями свободы с нелинейными правыми частями в виде многочлена третей степени от фазовых переменных с постоянными и периодическими параметрами [4]. Система виброзащиты состоит из объекта виброзащиты массой m_1 установленного на две платформы, находящиеся одна под другой, с массами m_2 и m_3, нижняя из которых закреплена на вибрирующем основании. Предполагается, что упругие элементы системы имеют вид полинома третей степени $kx + lx^2 + px^3$, демпфирующие элементы имеют нелинейную кубическую характеристику $c\dot{x} + d\dot{x}^3$. Внешнее гармоническое возмущение воздействует на основание[5]. Для получения уравнений движения виброзащитной системы используем уравнения Лагранжа.

Система уравнений движения виброзащитной системы имеет вид:

$$
\begin{aligned}
&m_1\ddot{x}_1 + c_1(\dot{x}_1 - \dot{x}_2) + d_1(\dot{x}_1 - \dot{x}_2)^3 + k_1(x_1 - x_2) + l_1(x_1 - x_2)^2 + p_1(x_1 - x_2)^3 = 0,\\
&m_2\ddot{x}_2 + c_1(\dot{x}_2 - \dot{x}_1) + d_1(\dot{x}_2 - \dot{x}_1)^3 + k_1(x_2 - x_1) - l_1(x_2 - x_1)^2 + p_1(x_2 - x_1)^3 +\\
&c_2(\dot{x}_2 - \dot{x}_3) + d_2(\dot{x}_2 - \dot{x}_3)^3 + k_2(x_2 - x_3) + l_2(x_2 - x_3)^2 + p_2(x_2 - x_3)^3 = 0, \qquad (15)\\
&m_2\ddot{x}_3 + c_2(\dot{x}_3 - \dot{x}_2) + d_2(\dot{x}_3 - \dot{x}_2)^3 + k_2(x_3 - x_2) - l_2(x_3 - x_2)^2 + p_2(x_3 - x_2)^3 +\\
&c_3(\dot{x}_3 - \dot{f}) + d_3(\dot{x}_3 - \dot{f})^3 + k_3(x_3 - f) + l_3(x_3 - f)^2 + p_3(x_3 - f)^3 = 0
\end{aligned}
$$

где x_1, x_2, x_3 – абсолютные перемещения по отношению к положению равновесия системы.

На основание действуют вертикальные колебания:

$$f(t) = a\sin(\omega t). \qquad (16)$$

Введем относительное перемещение:

$$\tilde{x}_1 = x_1 - f,\ \tilde{x}_2 = x_2 - f,\ \tilde{x}_3 = x_3 - f. \qquad (17)$$

Запишем уравнения движения (15) в новых переменных:

$$
\begin{aligned}
&m_1\ddot{\tilde{x}}_1 + c_1(\dot{\tilde{x}}_1 - \dot{\tilde{x}}_2) + d_1(\dot{\tilde{x}}_1 - \dot{\tilde{x}}_2)^3 + k_1(\tilde{x}_1 - \tilde{x}_2) + l_1(\tilde{x}_1 - \tilde{x}_2)^2 + p_1(\tilde{x}_1 - \tilde{x}_2)^3 = -m_1\ddot{f}_1,\\
&m_2\ddot{x}_2 + c_1(\dot{\tilde{x}}_2 - \dot{\tilde{x}}_1) + d_1(\dot{\tilde{x}}_2 - \dot{\tilde{x}}_1)^3 + k_1(\tilde{x}_2 - \tilde{x}_1) - l_1(\tilde{x}_2 - \tilde{x}_1)^2 + p_1(\tilde{x}_2 - \tilde{x}_1)^3 +\\
&c_2(\dot{\tilde{x}}_2 - \dot{\tilde{x}}_3) + d_2(\dot{\tilde{x}}_2 - \dot{\tilde{x}}_3)^3 + k_2(\tilde{x}_2 - \tilde{x}_3) + l_2(\tilde{x}_2 - \tilde{x}_3)^2 + p_2(\tilde{x}_2 - \tilde{x}_3)^3 = -m_2\ddot{f}, \qquad (18)\\
&m_2\ddot{x}_3 + c_2(\dot{\tilde{x}}_3 - \dot{\tilde{x}}_2) + d_2(\dot{\tilde{x}}_3 - \dot{\tilde{x}}_2)^3 + k_2(\tilde{x}_3 - \tilde{x}_2) - l_2(\tilde{x}_3 - \tilde{x}_2)^2 + p_2(\tilde{x}_3 - \tilde{x}_2)^3 +\\
&c_3\dot{\tilde{x}}_3 + d_3\dot{\tilde{x}}_3^{\,3} + k_3\tilde{x}_3 + l_3\tilde{x}_3^{\,2} + p_3\tilde{x}_3^{\,3} = -m_3\ddot{f}
\end{aligned}
$$

$$\ddot{f}(t) = -a\omega^2\sin(\omega t). \qquad (19)$$

Выполним многочленное преобразование системы (18) согласно схеме.

Запишем систему уравнений в матричной форме:

$$\dot{X} = RX + P, \text{ где } X = [\exp(i\omega t), \exp(-i\omega t), \tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \dot{\tilde{x}}_1, \dot{\tilde{x}}_2, \dot{\tilde{x}}_3]^T, \qquad (20)$$

Здесь P – нелинейный вектор системы. В результате линейной замены переменных

$$Y = AX \qquad (21)$$

получим систему с линейной диагональной матрицей вида

$$\dot{Y} = \Lambda Y + \tilde{P}. \qquad (22)$$

Выполним многочленную замену переменных:

$$y_S = z_S + \sum_{|\nu|=2}^{4} a_\nu^S Z^\nu, (s = 3,...,8),\ y_S = z_S\ (s = 1,2),\ Z^\nu \equiv z_1^{\nu_1} z_2^{\nu_2} z_3^{\nu_3} z_4^{\nu_4} z_5^{\nu_5} z_6^{\nu_6} z_7^{\nu_7} z_8^{\nu_8}. \qquad (23)$$

С точностью до членов четвертого порядка получаем автономную дифференциальную систему:

$$\dot{z}_3 = (\lambda_3 + q^3_{11100000})z_3,\ \dot{z}_5 = (\lambda_5 + q^5_{11001000})z_5\ ,\dot{z}_7 = (\lambda_7 + q^7_{11000010})z_5 \qquad (24)$$

Решение системы записывается в виде:

$$
\begin{aligned}
&z_{3,4} \equiv \rho_{01}\exp(t\,\mathrm{Re}(\lambda_3 + q^3_{11100000}) \pm i(\theta_{01} + t\,\mathrm{Im}(\lambda_3 + q^3_{11100000}))),\\
&z_{5,6} \equiv \rho_{02}\exp(t\,\mathrm{Re}(\lambda_5 + q^5_{11001000}) \pm i(\theta_{02} + t\,\mathrm{Im}(\lambda_5 + q^5_{11001000}))), \qquad (25)\\
&z_{7,8} \equiv \rho_{03}\exp(t\,\mathrm{Re}(\lambda_7 + q^7_{11000010}) \pm i(\theta_{03} + t\,\mathrm{Im}(\lambda_7 + q^7_{11000010}))).
\end{aligned}
$$

Рассмотрим виброзащитную систему (18) при следующих числовых значениях параметров:

$$
\begin{aligned}
&m_1 = 1.13, c_1 = 0.23, d_1 = 0.01, k_1 = 1.23, l_1 = 0.04, p_1 = 0.02,\\
&m_2 = 3.17, c_2 = 0.61, d_2 = 0.03, k_2 = 3.13, l_2 = 0.13, p_2 = 0.06,\\
&m_3 = 8.71, c_3 = 1.73, d_3 = 0.09, k_3 = 9.11, l_3 = 0.37, p_3 = 0.17
\end{aligned}
$$

На систему действует внешнее возмущение: $\omega = 2, a = 0.5$.

В этом случае коэффициенты преобразованной системы (24) имеют следующие значения:

$q^3_{11100000} = -0.086 + 0.011i, q^5_{11001000} = -0.154 + 0.012i, q^7_{11000010} = -0.044 + 0.001i$

Решение системы (24) принимает вид

$z_{3,4} \equiv \rho_{01} \exp(-0.297t \pm i(\theta_{01} + 1.480t))$,

$z_{5,6} \equiv \rho_{02} \exp(-0.273t \pm i(\theta_{02} + 1.130t))$,

$z_{7,8} \equiv \rho_{03} \exp(-0.083t \pm i(\theta_{03} + 0.635t))$.

Решение системы, определяющее установившееся движение имеет вид

$x_1 = -0.035\cos(2t) - 0.470\sin(2t)$,

$x_2 = -0.113\cos(2t) - 0.526\sin(2t)$,

$x_3 = -0.261\cos(2t) - 0.612\sin(2t)$.

Установившейся режим колебаний системы происходит с частотой внешней силы.

В результате применения метода многочленных преобразований получено в аналитическом виде решение для приборной механической системы, установленной на вибрирующей платформе посредством амортизаторов с нелинейными характеристиками полиномиального типа, в случае отсутствия внешних и внутренних резонансов [6, 7]. Разработан пакет программ для исследования методом многочленных преобразований нелинейных задач виброзащиты приборных систем. Получены алгоритмические формулы метода многочленных преобразований, удобные для составления программ с использованием символьных вычислений. Определены существенные динамические константы виброзащитной системы, характеризующие переходные процессы и установившиеся режимы колебаний. Полученные результаты проконтролированы численным решением методом Рунге-Кутта. Метод многочленных преобразований позволяет получить достаточно подробные качественные и количественные характеристики вынужденных колебаний систем, исследовать установившиеся режимы колебаний, а также изучать переходные процессы.

Литература

1. Мельников Г.И. Динамика нелинейных механических и электромеханических систем. Л: Маш., 1975. 198 с.
2. Мельников Г.И. К теории нелинейных колебаний. // Вестник ЛГУ.1964., №1. Вып.1. С.88–98
3. Иванов С.Е. О реализации численно-аналитического метода многочленных преобразований на компьютере. // Современные технологии: Труды молодых ученых ИТМО / Под ред. проф. С.А. Козлова. СПб: СПб ГИТМО(ТУ), 2001. С. 138–141.
4. Вибрации в технике. Справочник в 6-ти т., т.6, под ред. К.В. Фролова М.: Маш., 1995.. 456 с.
5. Андронов А.А., Витт А.А., Хайкин С.Э. Теория колебаний. - М: Наука, 1981. 568 с.
6. Фролов К.В. Нелинейные задачи динамики машин. М: Маш,1992.-376 с.
7. Фурман Ф. А., Фролов К. В. Прикладная теория виброзащитных систем. М: Маш.,1980. 317 с.

ПРИМЕНЕНИЕ КОМПЬЮТЕРНЫХ ПАКЕТОВ И АНИМАЦИЙ В ПРЕПОДАВАНИИ МЕХАНИКИ

В.Г. Мельников, С.Е. Иванов

В статье изложены основные научно-методические результаты, полученные на кафедре теоретической и прикладной механики в процессе разработки новых компьютерных методик в области преподавания механики по специальностям СПбГУ ИТМО. Изложены некоторые новые технологии, примененные в преподавании курсов механики и сопротивления материалов и особенности электронного учебно-методического комплекса, проиллюстрированные примерами.

Существенное место в новых технологиях преподавания фундаментальных дисциплин занимает визуализация учебного материала, приводящая к созданию общего зрительного впечатления о разделах курса. Некоторые зарубежные учебники по точным наукам содержат большое количество высококачественных иллюстраций, интенсивно подключающих зрительную память студента и формирующих целостное впечатление о дисциплине и более полный объем остаточных знаний. Современные системы компьютерного обеспечения открывают большие возможности для визуализации и интенсификации учебного процесса на аудиторных занятиях.

Кафедра теоретической и прикладной механики СПбГУ ИТМО в рамках разработки новых технологий в преподавании механики развивает два направления. Во-первых, мы критически пересматриваем содержание устоявшихся курсов теоретической механики и сопротивления материалов, полагая, что в условиях прогресса прикладной компьютерной математики дисциплины должны претерпеть изменения и воспринять нарастающие возможности [1]. Вторым направлением является разработка новых тестов, новых методических материалов для обеспечения лекций и практических занятий с систематическим применением видеоклипов и анимаций в аудиториях, оснащенных соответствующим компьютерным оборудованием. Имеется задел в подготовке виртуальных лабораторных работ по сопротивлению материалов и теоретической механике. Предполагается, что виртуальные работы должны дублировать реальные лабораторные работы либо выполняться без дублирования. На лекциях мы применяем систему компьютерных слайдов, включающих анимированные изображения объектов и основные расчетные формулы. В результате лектор частично освобождается от графической работы на доске и получает дополнительные возможности для объяснения рисунков и формул. При этом он может ограничиться пояснением порядка построения иллюстраций. На каждой последующей лекции преподаватель получает возможность быстрого обзора части пройденного материала и перехода к новой теме.

На кафедре теоретической и прикладной механики при участии доктора физико-математических наук, профессора Г.И. Мельникова создан электронный учебник по теоретической механике, имеющий в своем составе компьютерные анимации и решения типовых задач по новой компьютерной методике. Компьютерные анимации демонстрируют движение изучаемых механических объектов, при этом на объекты нанесены те или иные переменные векторные величины, характеризующие движение объекта. Демонстрируется изменение во времени векторных характеристик, перемещающихся вместе с подвижным объектом. В результате просмотра такой анимации у зрителя остается впечатление о движении объекта с позиции механики, что способствует усвоению теоретического материала. Другого рода анимации созданы для иллюстрации проявлений законов динамики, а также важных эффектов, играющих существенную роль в приборостроении, таких как резонанс, виброзащищенность, автоколебания, гироскопические явления, деформации. В целом каждая из анимаций, созданных при квалифицированной поддержке центра дистанционного обучения (ЦДО), несет большую смысловую нагрузку, демонстрирует проявление механических законов, структуры уравнений движения. Мы используем анимации и на лекциях, что положительно влияет на освое-

ние теоретического материала. Разработанные анимации и решения практических задач по новым компьютерным методикам были продемонстрированы в докладе на всероссийском семинаре-совещании заведующих кафедрами теоретической механики [2]. Для контроля знаний применяется тестирование в ЦДО, а также Palm-тестирование.

Объемы механических дисциплин за последние годы претерпели существенное сокращение. С другой стороны, применение компьютерных пакетов: MatLab, Maple, MatCad, Mathematica приводит к необходимости принципиальных изменений в методике проведения практических занятий и в содержании лекций. В нашем электронном учебнике найдены способы существенного сокращения промежуточных выкладок в процедуре решения задач теоретической механики и сопротивления материалов. Широко применяются матричные формы записи, свойственные математическим компьютерным пакетам. Разработана для механики небольшая система несложных матричных файл-функций, которые студент сам формирует в систем MatLab. Эти функции позволяют резко сократить и изменить процедуру решения сложных задач. В направлении внедрения компьютерных пакетов в практические занятия ведутся разработки в ряде вузов РФ. Нам удалось разработать методику применения компьютерных пакетов на более раннем этапе – непосредственно на стадии перехода от физической (механической) модели к математической модели. Математическая модель представляется в системе MatLab, после чего уравнения решаются и осуществляется физическая интерпретация результатов и расшифровка построенных графиков [3].

В качестве иллюстрации разработанной на кафедре методики приведем решение плоской и пространственной задачи статики. Студент создает две-три несложные m-функции, образованных из координат точек приложения сил. Процедура решения задачи состоит в изучении заданной физической модели (расчетной схемы) и записи данной задачи в переменных MatLab. Затем в виде одной строки непосредственно записывается матричное неоднородное алгебраическое уравнение равновесия, по которому система MatLab возвращает решение задачи в форме вектор-строки. Таким образом, из обычной процедуры решения задачи исключается трудоемкая операция составления системы алгебраических уравнений и операция перехода к матричной форме, система непосредственно записывается в матричной форме. Достигнутое сокращение приведено не в ущерб наглядности изучения физической модели. Заметим, что в новых изданных учебных пособиях, использующих компьютерные технологии, не исключена названная выше трудоемкая операция. Подобным подходом решаются задачи по определению динамических реакций, а также другие сложные задачи динамики. Покажем процесс решений, опуская рисунки.

Пример 1. Решение задачи о равновесии системы двух тел. Пусть система двух прямых шарнирно соединенных стержней закреплена на двух шарнирных опорах под углом Pi/6 к горизонтальной линии, проходящей через оба шарнира. Известны веса стержней G и Q и абсциссы точек приложения сил тяжести b и 3b. Определим реакции опор X1 Y1 X2 Y2 и реакцию соединяющего шарнира X3 Y3.

Прежде всего, дадим сквозную нумерацию всех приложенных сил, присвоив номера известным силам: G=G4, Q=Q5. Тем самым мы присвоили номера радиус-векторам точек приложения сил.

Обозначим в системе MatLab все необходимые символьные скалярные величины

```
>>syms X1 Y1 X2 Y2 X3 Y3 G4 Q5 b  real
```

Имеем вектор-строки точек приложения сил, выраженные через объявленные символы

```
>> r1=[0      0]
>> r2=[4*b   0]
>> r3=[2*b   2*b*sin(pi/6)]
>> r4=[b      b*sin(pi/6)]
```

>> r5=[3*b b*sin(pi/6)]

>> e4=[0 -1]

>> e5=[0 -1]

Составим вектор-строку неизвестных величин и вектор-строки сил

>>V=[X1 Y1 X2 Y2 X3 Y3]

>>F1=[X1 Y1]; F2=[X2 Y2]; F3=[X3 Y3]

Составим левую часть матричного уравнения равновесия тела как единого твердого тела.

В это выражение дополнительно включен равный нулю член F3*s230, с нулевой 2×3 матрицей s230.Такое включение обеспечивает формальное присутствие в выражении всех неизвестных. Каждое слагаемое есть бивектор той или другой силы:

>>F1*s23(r1)+F2*s23(r2)+F3*s230+G*e4*s23(r4) +Q*e5*s23(r5)

Затем отбросим второй стержень, сохранив его действие на первый стержень в виде неизвестной реакции F3, и составим левую часть матричного уравнения равновесия тела.

>>F1*s23(r1)+F2*s230+F3*s23(r3)+G*e4*s23(r4)

Выполним конкатенацию матриц в каждом уравнении, а именно, соединим неизвестные первые множители в строку, а известные вторые матричные множители в блочный столбец.

>>V*[s23(r1); s23(r2); s230]+ G*e4*s23(r4)+Q*e5*s23(r5)

>>V*[s23(r1); s230; s23(r3)]+G*e4*s23(r4)

Выполним горизонтальное объединение уравнений, что сводится к горизонтальной конкатенации вторых сомножителей, а также к горизонтальной конкатенации известных сил.

>>L1=[s23(r1); s23(r2); s230]

>>L2=[s23(r1); s230; s23(r3)]

>>W1=G*e4*s23(r4) + Q*e5*s23(r5)

>>W2=G*e4*s23(r4)

Получаем матрицу координат точек приложения неизвестных сил

>>L=[L1 L2]

и бивектор известных сил

>>W=[W1 W2]

Решение матричного уравнения равновесия системы двух стержней в этих обозначениях V*L+W=zeros(1,6) находится в результате правого деления матриц в системе MatLab.

>>V=-W/L

Ответ в виде вектор-строки искомых проекций неизвестных сил.

V =[1/2*Q+1/2*G, 1/4*Q+3/4*G, -1/2*Q-1/2*G, 3/4*Q+1/4*G, -1/2*Q-1/2*G, -1/4*Q+1/4*G]

Здесь использованы две m-функции: s23 и s230.

Пример 2. Решение задачи о равновесии пространственной системы сил.

Квадратная однородная полка весом G со стороной b находится в равновесии под действием реакции сферического шарнира F1, расположенного в начале координат, реакции цилиндрического шарнира F2, с координатой [b 0 0] и реакции троса T, заданной вектор-строкой T3=T*e3 с известым ортом-строкой e3 и неизвестным модулем силы T. К полке приложена пара сил с известным моментом M. Найти реакции опор конструкции.

Символьные скалярные величины

>>syms X1 Y1 Z1 X2 Z2 T3 G M a b real

Радиус-векторы точек приложения сил и строки неизвестных реакций

>> r1=[0 0 0]

```
>> r2=[0 b 0]
>> r3=[0 b -b/tan(a)]
>> r4=[0.5*b 0.5*b 0]
>> e3=[sin(a) 0 cos(a)]
>> e4=[0 0 -1]
>> e5=[-1 0 0]
>> F1=[X1 Y1 Z1]
>> F2=[X2 Z2]
>>V=[X1 Y1 Z1 X2 Z2 T3]
```

Составим матричное уравнение равновесия (левая часть)

```
>>F1*s36(r1)+F2*sy26(r2)+T3*e3*s36(r3)+G*e4*s36(r4)+M*[e5,zeros(1,3)]
```

Матрица координат неизвестных сил, сцепленная по вертикали из блоков.

```
>>L=[ s36(r1);sy26(r2);e3*s36(r3)]
```

Бивектор известных сил

```
>>W=G*e4*s36(r4)+M*[e5,zeros(1,3)]
```

Символьное решение как результат правого деления матриц в системе MatLab.

```
>>V=-W/L
V =[ 0, 0, 1/2*G-1/b*M, -1/2*tan(a)*G, 1/b*M, 1/2/cos(a)*G]
```

Таким образом, применены новые компьютерные технологии и матричные формы представления математических моделей в механике, что приводит к более компактному изложению основного материала в небольших курсах механики. Визуализация механического движения объектов достигается широким применением Flash-анимаций.

Литература

1. Мельников В.Г., Иванов С.Е., Мельнков Г.И. Теоретическая механика. Динамика. [Электронный ресурс] : для студентов приборостроительных. и др. специальностей / СПбГУ ИТМО- Режим доступа: http://de.ifmo.ru. Загл. с экрана.
2. Мельников В.Г. Метод идентификации тензоров инерции и центров масс твердых тел на программных движениях и устройство для его осуществления // III Всероссийское совещание-семинар заведующих кафедрами теоретической механики вузов РФ Пермь: ПГУ. 2004. с 23-24
3. Мельников Г.И., Мельников В.Г. Матричные символьные вычисления в среде MATLAB в электронном учебнике по теоретической механике // III Всероссийское совещание-семинар заведующих кафедрами теоретической механики вузов РФ Пермь: ПГУ. 2004. с. 37-39.

ВЫНУЖДЕННЫЕ КОЛЕБАНИЯ НЕЛИНЕЙНОЙ СИСТЕМЫ ПРИ РЕЗОНАНСАХ ВЫСШИХ ПОРЯДКОВ

А.Г. Кривошеев

В [1–3] развит метод многочленного преобразования динамических уравнений движения нелинейных механических систем с конечным числом степеней свободы. Этот метод применим к колебательным системам, для которых характерна малость вещественных частей собственных чисел матрицы линейной части уравнений. Многочленные преобразования могут выполняться численно-аналитически с применением ПК [4, 5].

В данной работе рассматривается применение метода многочленных преобразований к исследованию нелинейной колебательной системы с двумя степенями свободы в случаях внешних резонансов высших порядков. Разработан алгоритм преобразования исходных неавтономных дифференциальных уравнений к автономному виду, удобному для определения стационарных режимов колебаний и исследования их устойчивости.

Рассмотрим механическую систему с двумя степенями свободы и с голономными нестационарными периодическими связями. Допустим, что ее уравнения Лагранжа имеют матричный вид

$$A\ddot{q} + B\dot{q} + Cq = h_1 \cos\omega_0 t + h_2 \sin\omega_0 t + \\ + \sum_{|\nu|=2}^{3} h_\nu \cdot \cos^{\nu_0}\omega_0 t \cdot \sin^{\nu_0'}\omega_0 t \cdot q_1^{\nu_1} \cdot q_2^{\nu_2} \cdot \dot{q}_1^{\nu_1'} \cdot \dot{q}_2^{\nu_2'} \qquad (1)$$

Здесь $q=\left[q_1 q_2\right]^T$ – вектор обобщенных координат системы; A, B, C – постоянные матрицы, причем матрица A - обратимая; $h_1, h_2, h_\nu = \left[h_\nu^1, h_\nu^2\right]^T$ – постоянные векторы; суммирование ведется по векторному индексу $\nu=\left[\nu_0 \nu_1 \nu_2 \nu_0' \nu_1' \nu_2'\right]$ с нормой $|\nu|=\nu_0+\nu_1+\ldots+\nu_2'$. Предположим, что характеристическое уравнение $\det\left[A\lambda^2+B\lambda+C\right]=0$ имеет сопряженные комплексные корни $\lambda_s=\alpha_s+i\omega_s$, $\overline{\lambda}_s=\alpha_s-i\omega_s$, $\omega_s>0$, s = 1, 2 с малыми ненулевыми вещественными частями и различными мнимыми частями:

$$\max\left|\alpha_s\right|=\varepsilon \ll\omega_j, \quad j=0,1,2; \qquad \left|\omega_1-\omega_2\right|>>\varepsilon.$$

Считая коэффициенты h_ν^s достаточно малыми, приведем периодическую систему (1) методом многочленного преобразования к возможно более простому виду.

Введем дополнительную переменную $q_0=\cos\omega_0 t$, удовлетворяющую дифференциальному уравнению с фиксированными начальными данными:

$$\ddot{q}_0+\omega_0^2 q_0=0; \quad q_0(0)=1; \quad \dot{q}_0(0)=0. \qquad (2)$$

Тогда уравнения (1), (2) могут быть представлены как автономная система шестого порядка

$$\dot{x} = Px + \sum_{|\nu|=2}^{3} p_\nu x^\nu, \quad x=\left[q_0 q_1 q_2 \dot{q}_0 \dot{q}_1 \dot{q}_2\right]^T; \quad x^\nu = q_0^{\nu_0} \cdot \ldots \cdot \dot{q}_2^{\nu_2'}. \qquad (3)$$

Далее в системе (3) выполним линейную замену переменных $x=\tilde{A}y$, которая преобразует матрицу P к квазидиагональной форме

$$\widetilde{P}=\widetilde{A}^{-1}P\widetilde{A}=\begin{bmatrix}\Lambda & 0\\ 0 & \overline{\Lambda}\end{bmatrix};\quad \Lambda=\begin{bmatrix}\lambda_0 & 0 & 0\\ \mu & \lambda_1 & 0\\ 0 & 0 & \lambda_2\end{bmatrix}.$$

Поддиагональный элемент $\mu\neq 0$ в случае, когда собственные числа $\lambda_0=i\omega_0$, λ_1 близки к кратным, т.е. выполняется резонансное соотношение $|\omega_0-\omega_1|\approx\varepsilon$. В остальных случаях $\mu=0$, и матрица $\widetilde{P}$ является диагональной. После приведения подобных степенных одночленов получаем систему

$$\dot{y}=\widetilde{P}y+\sum_{|\nu|=2}^{3}f_\nu y^\nu,\quad y=\begin{bmatrix}y_0 y_1 y_2 \bar{y}_0 \bar{y}_1 \bar{y}_2\end{bmatrix}^T. \tag{4}$$

В системе (4) уравнения являются комплексно-сопряженными, причем переменные $y_0,\bar{y}_0$ удовлетворяют отделяющимся уравнениям

$$\dot{y}_0=\lambda_0 y_0;\quad \dot{\bar{y}}_0=\overline{\lambda}_0\bar{y}_0;\quad y_0(0)=\bar{y}_0(0)=1. \tag{5}$$

Затем в системе (4) выполним многочленную замену переменных

$$y=z+\sum_{|\nu|=2}^{3}a_\nu z^\nu. \tag{6}$$

Результатом такого преобразования с удержанием членов до третьего порядка включительно будет система

$$\dot{z}=\widetilde{P}z+\sum_{|\nu|=2}^{3}\widetilde{p}_\nu z^\nu,\quad z=\begin{bmatrix}z_0 z_1 z_2 \bar{z}_0 \bar{z}_1 \bar{z}_2\end{bmatrix};\quad \widetilde{p}_\nu=\begin{bmatrix}0\,\widetilde{p}_\nu^1\,\widetilde{p}_\nu^2\,0\,\widetilde{p}_\nu^3\,\widetilde{p}_\nu^4\end{bmatrix} \tag{7}$$

В этой системе два уравнения относительно $z_0,\bar{z}_0$ сохраняются в виде (5), а в остальных уравнениях формируется структура степенных одночленов с минимальным количеством ненулевых коэффициентов $\widetilde{p}_\nu^s$. Поскольку уравнения комплексно-сопряженные, можно рассматривать два уравнения при s = 1, 2. Коэффициенты системы $\widetilde{p}_\nu^s$ связаны с коэффициентами преобразования a_ν^s равенствами с рекуррентно вычисляемыми правыми частями [3]:

$$\widetilde{p}_\nu^s+d_\nu^s a_\nu^s=c_\nu^s,\quad |\nu|=2,3;\quad s=1,2. \tag{8}$$

Выбор $\widetilde{p}_\nu^s\neq 0$ подчинен условию малости по модулю делителей

$$\begin{aligned}&d_\nu^s=\nu_0\lambda_0+\nu_1\lambda_1+\ldots+\nu'_2\overline{\lambda}_2-\lambda_s\equiv \mathrm{Re}\,d_\nu^s+i\,\mathrm{Im}\,d_\nu^s,\\ &\mathrm{Re}\,d_\nu^s=\sum_{j=1}^{2}\left(\nu_j+\nu'_j\right)\alpha_j-\alpha_s;\quad \mathrm{Im}\,d_\nu^s=\sum_{j=1}^{2}\left(\nu_j-\nu'_j\right)\omega_j-\omega_s.\end{aligned} \tag{9}$$

с заранее назначенной малой константой $\widetilde{\varepsilon}>0$. При этом для каждого s = 1, 2 векторные индексы $\nu: |\nu|=2,3$ подразделяются на множество особых индексов $M_s=\left\{\nu: \left|d_\nu^s\right|\le\widetilde{\varepsilon}\right\}$, для которых $a_\nu^s=0$, $\widetilde{p}_\nu^s=c_\nu^s$ и множество неособых индексов, для которых $\widetilde{p}_\nu^s=0$, $a_\nu^s=c_\nu^s/d_\nu^s$, $\nu\notin M_s$. При сделанных предположениях о собственных числах λ_s малость делителей определяется малостью их мнимых частей, т.е. выявлением резонансных соотношений вида

$$\left|k_0\omega_0+k_1\omega_1+k_2\omega_2\right|\approx\varepsilon,\quad k_j=0,\pm1,\pm2,\pm3,\quad 2\le\sum_{j=0}^{2}\left|k_j\right|\le4. \tag{10}$$

Исходные уравнения (1) при условии отсутствия резонансных соотношений (10) рассмотрены в [6]; резонанс вида $\omega_1\approx\omega_0$ изучен в [7]. В данной работе рассмотрим случай резонанса $\omega_1\approx2\omega_0$, в котором особые индексы образуют следующие множества:

$M_1=\{[110100],\ [020010],\ [011001],[200000]\}$;

$M_2=\{[101100],[011010],[002001]\}$.

В нерезонансном случае каждое из множеств M_1 и M_2 состояло из трех индексов. В рассматриваемом резонансном случае к M_1 добавляется один (последний) особый индекс.

Первые три уравнения преобразованной системы (7) имеют вид

$$\begin{aligned}
&\dot z_0=i\omega_0 z_0,\quad z_0(0)=1;\\
&\dot z_1=\left(\lambda_1+\tilde p_0^1+\tilde p_1^1 z_1\bar z_1+\tilde p_2^1 z_2\bar z_2\right)z_1+\tilde p\, z_0^2;\\
&\dot z_2=\left(\lambda_2+\tilde p_0^2+\tilde p_1^2 z_1\bar z_1+\tilde p_2^2 z_2\bar z_2\right)z_2;
\end{aligned} \tag{11}$$

а остальные три уравнения являются попарно комплексно сопряженными с ними. В уравнениях (11) при коэффициентах $\tilde p_j^s=a_{sj}+ib_{sj}$ используется скалярный индекс j = 0,1,2, соответствующий порядку следования векторных индексов в множествах $M_s, s=1,2$, а также учтено равенство $z_0\bar z_0=1$. Для последнего индекса в множестве M_1 введен коэффициент $\tilde p=c\exp i\vartheta$.

Уравнения (11) содержат переменную $z_0=\exp i\omega_0 t$ и не являются автономными, но путем замены переменных $z_1=z_0^2u_1;\ z_2\equiv u_2$ приводятся к двум автономным уравнениям относительно переменных u_1,u_2:

$$\begin{aligned}
&\dot u_1=\left(\lambda_1-2\lambda_0+\tilde p_0^1+\tilde p_1^1u_1\bar u_1+\tilde p_2^1u_2\bar u_2\right)u_1;\\
&\dot u_2=\left(\lambda_2+\tilde p_0^2+\tilde p_1^2u_1\bar u_1+\tilde p_2^2u_2\bar u_2\right)u_2.
\end{aligned} \tag{12}$$

Представив в уравнения (12) переменные $u_s=\rho_s\exp i\varphi_s$ и разделив их вещественные и мнимые части, получаем автономную дифференциальную систему уравнений относительно переменных ρ_s,φ_s с вещественными коэффициентами:

$$\begin{cases}
\dot\rho_1=(a_1+a_{11}\rho_1^2+a_{12}\rho_2^2)\rho_1+c\cos(\varphi_1-\vartheta);\\
\rho_1\dot\varphi_1=(b_1+b_{11}\rho_1^2+b_{12}\rho_2^2)\rho_1-c\sin(\varphi_1-\vartheta);\\
\dot\rho_2=(a_2+a_{21}\rho_1^2+a_{22}\rho_2^2)\rho_2;
\end{cases} \tag{13}$$

$$\dot\varphi_2=(b_2+b_{21}\rho_1^2+b_{22}\rho_2^2),$$

где $a_1=\alpha_1+a_{10};\ b_1=\omega_1-2\omega_0+b_{10};\ a_2=\alpha_2+a_{20};\ b_2=\omega_2+b_{20}$.

В системе (13) замкнуты первые три уравнения относительно трех «медленных» переменных: двух амплитудных переменных ρ_1,ρ_2 и одной фазовой переменной φ_1. Структура правых частей этих уравнений более простая в сравнении с уравнениями в случае резонанса $\omega_1\approx\omega_0$, поскольку переменные ρ_1,ρ_2 и фаза φ_1 входят в них ад-

дитивно. При резонансе $\omega_1 \approx 3\omega_0$ аналогично получаем систему вида (13) путем соответствующего изменения при автономизации системы (11).

Таким образом, исходная неавтономная периодическая система (1) при внешних резонансах высших порядков приведена методом многочленных преобразований к автономной рекуррентной системе четвертого порядка, для которой задачи определения стационарных режимов колебаний и исследования их устойчивости решаются типовыми алгебраическими операциями. Решение этих задач выходит за рамки данной статьи.

Литература

1. Мельников Г. И. К теории нелинейных колебаний // Вестник ЛГУ. 1964. №1 С. 88–98.
2. Мельников Г. И. О характере затухания возмущенного движения в двух особых случаях // Вестник ЛГУ. 1965. №19 С. 99–111.
3. Мельников Г. И. Динамика нелинейных механических и электромеханических систем. Л.: Машиностроение, 1975. 200 с.
4. Кривошеев А.Г., Мельников Г.И. Вынужденные колебания механических систем с нелинейными характеристиками полиномиального вида // Прикладная механика. 1990. Т. 26. №1. С. 108–113.
5. Мельников Г.И., Кривошеев А.Г. О многочленном преобразовании нелинейных динамических уравнений // Проблемы алгебры и кибернетики. Материалы международной конференции. Гомель: 1995. С. 66.
6. Мельников Г.И., Кривошеев А.Г. О применении метода многочленных преобразований к теории нелинейных колебаний. / В сб. «Проблемы Пространства, Времени, Движения». Т. II. СПб,1997. С. 185–190.
7. Кривошеев А.Г. Вынужденные резонансные колебания нелинейной системы с двумя степенями свободы // Научно-технический вестник СПб ГИТМО(ТУ). Выпуск 3. Физические процессы, системы и технологии точной механики / Главный редактор В.Н. Васильев. СПб: СПб ГИТМО(ТУ), 2001. С. 5–8.

2 УПРАВЛЕНИЕ И ИНФОРМАТИКА В ТЕХНИЧЕСКИХ СИСТЕМАХ

АДАПТИВНАЯ КОМПЕНСАЦИЯ ПО ВЫХОДУ СМЕЩЕННОГО ГАРМОНИЧЕСКОГО ВОЗМУЩЕНИЯ ДЛЯ СТРОГО МИНИМАЛЬНО-ФАЗОВОГО ОБЪЕКТА

А.А., Кремлев А.С. Бобцов

Статья посвящена развитию методов компенсации гармонических возмущений с неизвестными параметрами по измерениям выходной переменной объекта. Предлагается подход к компенсации гармонического возмущения, который превосходит по ряду характеристик известные аналоги.

Введение

В данной работе предлагается новый алгоритм адаптивной компенсации периодического возмущения $w(t) = \sigma_0 + \sigma \sin(\omega t + \phi)$, действующего на линейный строго минимально-фазовый объект управления. Если частота $\omega > 0$ известна, то поставленная проблема тривиальна и может быть решена с использованием классических подходов, предусматривающих синтез наблюдателей возмущения $w(t) = \sigma_0 + \sigma \sin(\omega t + \phi)$. Если же частота неизвестна, то эта проблема представляет значительный интерес. Имеется ряд работ, посвященных управлению в условиях неизвестной частоты возмущающего воздействия $w(t) = \sigma_0 + \sigma \sin(\omega t + \phi)$. В данной статье мы будем развивать подходы, представленные в работах [1, 2]. В статье [1] предлагается компенсатор размерности (2*n*+6). Алгоритм синтеза наблюдателя сложен в реализации, и для его построения требуется много вычислений, а также знание нижней границы параметра ω. В развитие подхода [1], в работе [2] предлагается компенсационный регулятор размерности (*n*+4), обладающей простой структурой (в сравнении с [1]) и не предусматривающий при своем построении знания нижней границы параметра ω.

Предлагаемый в данной статье компенсатор возмущения проще по структуре и ниже по размерности по сравнению с регуляторами, предложенными в [1, 2]. Также следует отметить, что, в отличие от работ [1, 2], в данной статье предполагается, что объект управления может быть неустойчивым, а его параметры неизвестны (в [1, 2] рассматривались устойчивые объекты с известными параметрами).

Постановка задачи

Как и в работах [1, 2], рассмотрим линейный объект вида:

$$\dot{z} = \begin{bmatrix} -a_{n-1} & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -a_1 & 0 & \dots & 1 \\ -a_0 & 0 & \dots & 0 \end{bmatrix} z + \begin{bmatrix} b_{n-1} \\ b_{n-2} \\ \vdots \\ 0 \end{bmatrix} (u + w) = Fz + g(u + w), \quad (1)$$

$$y = [1 \quad 0 \quad \dots \quad 0] z = hz, \quad (2)$$

где вектор переменных состояния $z = z(t)$ не измеряется, u – сигнал управления, y – регулируемая переменная. Входное возмущение $w(t)$ представлено в виде функции:

$$w(t) = \sigma_0 + \sigma \sin(\omega t + \phi), \quad (3)$$

включающей в себя смещенную синусоиду с неизвестной амплитудой $\sigma > 0$, неизвестной частотой ω, неизвестной фазой ϕ и неизвестным сдвигом σ_0.

Дадим следующие допущения относительно системы (1), (2):

- коэффициенты a_i, b_i, $0 \le i \le n-1$ неизвестны;
- полином $b(p) = p^{n-1} + b_{n-1}p^{n-2} + ... + b_1 p + b_0$ гурвицев, а коэффициент $b_0 > 0$.

Сформулируем цель управления как решение задачи синтеза алгоритма, обеспечивающего при любых начальных состояниях объекта выполнение условия

$$\lim_{t\to\infty} |y(t)| = 0. \qquad (4)$$

Синтез закона управления

Из работ [2, 3] известно, что при $k_1 = \alpha_0^3$, $k_2 = 3\alpha_0^2$, $k_3 = 3\alpha_0$ (где любое число $\alpha_0 > 0$) и любом $\theta > 0$ для моделирования сдвинутого гармонического возмущения можно воспользоваться системой вида

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = x_3, \\ \dot{x}_3 = -\theta x_2, \end{cases} \qquad (5)$$

$$w(t) = k_1 x_1 + k_2 x_2 + k_3 x_3. \qquad (6)$$

Представим алгоритм оценки в виде

$$\begin{cases} \dot{\hat{x}}_1 = \hat{x}_2, \\ \dot{\hat{x}}_2 = \hat{x}_3, \\ \dot{\hat{x}}_3 = -\hat{\theta}\hat{x}_2 + u_x, \end{cases} \qquad (7)$$

$$\hat{w}(t) = k_1\hat{x}_1 + k_2\hat{x}_2 + k_3\hat{x}_3, \qquad (8)$$

или в векторно-матричной форме

$$\dot{\hat{x}} = A_0\hat{x} - d\hat{\theta}\hat{x}_2 + du_x, \qquad (9)$$

$$\hat{w} = k^T\hat{x}, \qquad (10)$$

где матрица $A_0 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$, $d = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$, u_x – управление, обеспечивающее решение задачи компенсации возмущения, $k^T = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}$, $\hat{\theta}(t)$ – оценка неизвестного параметра θ.

Введем в рассмотрение новую переменную

$$\nu = (p+1)y = \frac{(p+1)b(p)}{a(p)}\tilde{w}, \qquad (11)$$

где

$$\tilde{w}(t) = w(t) - \hat{w}(t). \qquad (12)$$

Временно полагая, что функция $\dot{y}(t)$ измеряется, выберем алгоритм оценки неизвестного параметра θ в следующем виде:

$$\dot{\hat{\theta}} = -k_a\hat{x}_2\nu. \qquad (13)$$

Для расчета управления u_x введем в рассмотрение вектор невязки

$$\tilde{x} = x - \hat{x}, \qquad (14)$$

и параметрическую ошибку

$$\tilde{\theta} = \theta - \theta^{*}. \tag{15}$$

Дифференцируя уравнение (14), получаем:

$$\dot{\tilde{x}} = \dot{x} - \dot{x}^{*} = A_0 x - d\theta x_2 - A_0 x^{*} + d\theta^{*} x^{*}_2 - du_x =$$

$$= A_0 (x - x^{*}) + d[-(\tilde{\theta} + \theta^{*})(\tilde{x}_2 + x^{*}_2) + \theta^{*} x^{*}_2] - du_x =$$

$$= A_0 \tilde{x} - d\theta \tilde{x}_2 - d\tilde{\theta} x^{*}_2 - du_x, \tag{16}$$

$$\tilde{w} = k^T \tilde{x}. \tag{17}$$

Представим модель вход-состояние-выход (16), (17) в форме вход-выход:

$$\tilde{w} = \frac{\beta(p)}{\alpha(p)}(-u_x - \tilde{\theta} x_2) = W(p)(-u_x - \tilde{\theta} x_2), \tag{18}$$

где, в силу расчета вектора $k^T = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}$, передаточная функция $W(p) = \dfrac{\beta(p)}{\alpha(p)}$ строго минимально фазовая, т.е. полином $\beta(p)$ – Гурвицев, и относительная степень $W(p) = \dfrac{\beta(p)}{\alpha(p)}$ равна единице.

Подставляя уравнение (18) в (11), получаем

$$\nu = \frac{(p+1)b(p)\beta(p)}{a(p)\alpha(p)}(u_x - \tilde{\theta} x_2), \tag{19}$$

где передаточная функция $W_\nu(p) = \dfrac{(p+1)b(p)}{a(p)} \dfrac{\beta(p)}{\alpha(p)}$ строго минимально фазовая.

Пусть управление

$$u_x = \mu\nu, \tag{20}$$

тогда для системы (22) получаем

$$(a(p)\alpha(p) + \mu(p+1)b(p)\beta(p))\nu = (p+1)b(p)\beta(p)(-\tilde{\theta} x_2), \tag{21}$$

$$\nu = \frac{(p+1)b(p)\beta(p)}{a(p)\alpha(p) + \mu(p+1)b(p)\beta(p)}(-\tilde{\theta} x_2) = \Phi(p)(-\tilde{\theta} x_2). \tag{22}$$

Из теоремы Фрадкова о пассификации линейных систем (см., например, монографию [4]) известно, что существует число $\mu > 0$, для которого передаточная функция $\Phi(p)$ строго вещественно-положительная. Из строгой вещественной положительности $\Phi(p)$ следует, что алгоритм адаптации вида

$$\dot{\tilde{\theta}} = k_a x^{*}_2 \nu \tag{23}$$

обеспечивает асимптотическую устойчивость системы (22), последнее, в свою очередь, гарантирует выполнение цели управления (4) и возможность использования алгоритма (13).

Однако по условиям задачи сигнал $\dot{y}(t)$ не измеряется, а, следовательно, $\nu(t)$ не измеряется и управление $u_x = \mu\nu$ и алгоритм адаптации (13) нереализуемы. Построим реализуемую схему управления:

$$\dot{x}^{*} = A_0 x^{*} - d\theta^{*} \cdot x^{*}_2 + du_x = A_0 x^{*} - d\theta^{*} \cdot x^{*}_2 + d\mu y + d\mu\dot{y}. \tag{24}$$

Введем новую переменную

$$\xi = x^{*} - d\mu y, \tag{25}$$

тогда для настройки вектора $x^{*}(t)$ имеем

$$\begin{cases} \dot{\xi} = \dot{x}^{*} - d\mu\dot{y} = A_0 x^{*} - d\theta^{*} \cdot x^{*}_2 + d\mu y, \\ x^{*} = \xi + d\mu y. \end{cases} \tag{26}$$

Введем новую переменную, чтобы избавиться от неизвестной компоненты $\dot{y}$:

$$\eta = \hat{\theta} + \mu k_a \xi_2 y . \tag{27}$$

Дифференцируя уравнение (27), получаем

$$\dot{\eta} = \dot{\hat{\theta}} + \mu k_a \dot{\xi}_2 y + \mu k_a \xi_2 \dot{y} = -\mu k_a \xi_2 \dot{y} - \mu k_a \xi_2 y + \mu k_a \dot{\xi}_2 y + \mu k_a \xi_2 \dot{y} \tag{28}$$

где $\dot{\xi}_2 = \xi_3$ измеряется.

Из выражения (27) имеем реализуемую схему настройки параметра $\hat{\theta}$:

$$\begin{cases} \dot{\eta} = -\mu k_a \xi_2 y + k_a \xi_3 y, \\ \hat{\theta} = \eta - k_a \xi_2 y. \end{cases} \tag{29}$$

Таким образом, реализуемая схема управления представлена уравнениями (26), (29).

Заключение

В статье предложен новый алгоритм компенсации гармонического возмущения с неизвестными параметрами по измерениям выходной переменной объекта. Полученный результат превосходит решения, представленные в работах [1, 2], так как, в отличие от [1, 2]:

- неизвестен диапазон значений частоты ω (в работе [1] известна нижняя граница ω);
- структура данного регулятора является простой в сравнении с [1];
- размерность данного регулятора 4, что ниже, чем у аналогов [1, 2] (размерность регулятора в [1] $2n+6$, а в [2] $n+4$);
- в отличие от работ [1, 2], предполагается, что объект управления может быть неустойчивым и его параметры неизвестны (в работах [1, 2] полагается, что система асимптотически устойчива и параметры объекта известны).

Литература

1. Marino R., Santosuosso G.L., Tomei P. Robust adaptive compensation of biased sinusoidal disturbances with unknown frequency. // Automatica. 2003. V.39. P. 1755-1761.
2. Alexey Bobtsov, Artem Kremlev. Adaptive compensation of biased sinusoidal disturbances with unknown frequency. 16th IFAC World Congress, Prague, 2005.
3. Bobtsov A., Lyamin A., Romasheva D. Algorithm of parameter's identification of polyharmonic function // 15th IFAC World Congress on Automatic Control. Barcelona, Spain, 2002.
4. Мирошник И.В., Никифоров В.О., Фрадков А.Л. Нелинейное и адаптивное управление сложными динамическими системами. СПб.: Наука, 2000. 549 с.

СТАБИЛИЗАЦИЯ ХАОТИЧЕСКОЙ СИСТЕМЫ, ОПИСЫВАЕМОЙ УРАВНЕНИЕМ ВАН ДЕР ПОЛЯ

И.В. Амоскин, А.А. Блинников, А.А. Бобцов, Н.А. Николаев

В работе представлен подход к задаче стабилизации неопределенной хаотической системы, описываемой уравнением Ван дер Поля. Алгоритм управления использует измерения только выходной переменной системы, то есть без измерения ее производных или вектора состояния системы.

Введение

Проблема управления хаосом является областью интенсивных исследований последнего десятилетия. В настоящее время опубликовано множество работ, посвященных проблеме управления хаотическими системами, и выявлен целый ряд практических задач, где могут возникнуть хаотические режимы (см., например, обзоры [1? 2]). Теоретические и практические составляющие данной проблемы обусловлены тем, что колебательные и хаотические процессы часто встречаются в природе и технике. Формы их описания непрерывно развиваются и совершенствуются. Одним из классических примеров дифференциальных моделей, описывающих колебательные и хаотические процессы, является уравнение Ван дер Поля [3, 4].

В данной работе будет рассмотрена проблема управления хаосом на примере стабилизации хаотических процессов, возникающих в системе Ван дер Поля.

1. Постановка задачи

Рассмотрим нелинейный объект, описываемый уравнением Ван дер Поля

$$\ddot{y} - \tau_1(1 - y^2)\dot{y} + \tau_2 y = E\sin(\omega t) + u, \qquad (1)$$

где $\tau_1 > 0$, $\tau_2 > 0$.

Преобразуем модель (1) следующим образом

$$y = \frac{b(p)}{a(p)}u + \frac{d(p)}{a(p)}\varphi(y) + \frac{f(p)}{a(p)}w(t), \qquad (2)$$

где $p = d/dt$ – оператор дифференцирования; $b(p) = 1$; $a(p) = p^2 - \tau_1 p + \tau_2$ – неустойчивый полином ($\tau_1 > 0$ и $\tau_2 > 0$); $d(p) = d_1 p$, $d_1 = -1/3$; $f(p) = 1$; относительная степень передаточной функции $b(p)/a(p)$ $\rho = n - m = 2$; коэффициенты τ_1 и τ_2 предполагается неизвестным; $w(t) = E\sin(\omega t)$ – неизвестное ограниченное возмущение; функция $\varphi(y) = y^3$.

Цель управления – используя только измерения выходной переменной модели (2), найти закон управления, обеспечивающий сходимость выходной траектории нелинейной системы в некоторую область ε_0, границы которой могут быть уменьшены за счет соответствующего выбора коэффициентов регулятора.

2. Синтез алгоритма управления

Выберем закон управления вида

$$u = -\chi(p)(\mu + k)\xi, \qquad (3)$$

где коэффициент μ (принимает в общем случае достаточно большое значение) и полином $\chi(p)$ выбираются таким образом, чтобы полином $\gamma(p) = a(p) + \mu b(p)\chi(p)$ был гурвицевым; функция ξ формируется алгоритмом оценки вида

$$\dot{\xi} = \sigma(\nu - \xi), \qquad (4)$$

где функция $\nu = y + y^5$, а параметр k и функция σ выбираются в соответствии с требованиями, представленными ниже.

Подставляя (3) в уравнение (2), получаем

$$y = \frac{b(p)}{a(p)}[-\chi(p)(\mu + k)\nu + \chi(p)(\mu + k)\eta] + \frac{d(p)}{a(p)}\varphi(y) + \frac{f(p)}{a(p)}w(t), \quad (5)$$

где $\eta = \nu - \xi$ – функция отклонения (невязка).

Проводя несложные преобразования, для модели (5) имеем

$$a(p)y + \mu\chi(p)b(p)y = b(p)\chi(p)[(\mu + k)\eta - ky - (\mu + k)y^5] + d(p)\varphi(y) + f(p)w(t),$$

Принимая обозначения $\gamma(p) = a(p) + \mu\chi(p)b(p)$ и $\beta(p) = \chi(p)b(p)$, получаем

$$y = \frac{\beta(p)}{\gamma(p)}[-ky - (\mu + k)y^5 + (\mu + k)\eta + \overline{w}(t)] + \frac{d(p)}{\gamma(p)}\varphi(y), \quad (6)$$

где функция $\overline{w}(t) = \frac{f(p)}{\beta(p)}w(t)$ является гладкой и ограниченной, в силу вида функции $w(t)$.

Представим модель вход-выход (6) в виде модели вход-состояние-выход

$$\dot{x} = Ax + b(-ky - (\mu + k)y^5 + (\mu + k)\eta + \overline{w}(t)) + q\varphi(y), \; y = c^T x, \quad (7)$$

где $x \in R^2$ – вектор переменных состояния модели (7); A, b, q и c – соответствующие матрицы перехода от модели вход-выход (6) к модели вход-состояние-выход (7), причем в силу гурвицевости полинома $\gamma(p)$ и строгой минимальной фазовости модели (6), а также следствия 3 из статьи [5] можно указать симметрическую положительно определенную матрицу P, удовлетворяющую двум следующим матричным уравнениям:

$$A^T P + PA = -Q_1, \; Pb = c, \quad (8)$$

где $Q_1 = Q_1^T > 0$, значения матрицы Q_1 зависят от параметра μ и не зависят от параметра k.

Рассмотрим производную от функции отклонений η

$$\dot{\eta} = \dot{\nu} - \sigma(\nu - \xi) = \dot{y} + 5y^4\dot{y} - \sigma\eta = -\sigma\eta + \Omega\dot{y}, \quad (9)$$

где $\Omega = 1 + 5y^4$.

Сформулируем теорему, в которой будут указаны условия на расчет параметра k и функции σ, обеспечивающих выполнение цели управления.

Теорема. Существует параметр k и функция σ такие, что все траектории системы (7), (9) могут быть сведены в любую малую область за счет увеличения параметра k.

Доказательство. Рассмотрим функцию Ляпунова следующего вида:

$$V = V_1 + V_2, \quad (10)$$

где

$$V_1 = x^T Px, \quad (11)$$

$$V_2 = \eta^2. \quad (12)$$

Дифференцируя (11) по времени с учетом уравнений (7), получаем

$$\dot{V}_1 = x^T(A^T P + PA)x + 2(\mu + k)x^T Pb\eta - 2kx^T Pby - 2(\mu + k)x^T Pby^5 +$$

$$+ 2x^T Pb\overline{w} + 2x^T Pq\varphi(y), \qquad (13)$$

Подставляя в (13) уравнения (8), а также принимая во внимание соотношения

$$-2kx^T Pby = -2ky^2, \quad 2(\mu+k)x^T Pb\eta = 2(\mu+k)y\eta \le y^2 + (\mu+k)^2\eta^2,$$

$$2x^T Pb\overline{w} = 2y\overline{w} \le ky^2 + \frac{1}{k}\overline{w}^2, 2x^T Pq\varphi(y) \le \delta x^T Pqq^T Px + \delta^{-1}\left[\varphi(y)\right]^2,$$

для производной от функции Ляпунова (11) получаем

$$\dot{V}_1 \le x^T\left(-Q_1 + \delta Pqq^T P\right)x - (k-1)y^2 + (\mu+k)^2\eta^2 - 2(\mu+k)y^5 + \delta^{-1}\left[\varphi(y)\right]^2 + \frac{1}{k}\overline{w}^2, \qquad (14)$$

где малое число $\delta > 0$.

Дифференцируя (12) по времени с учетом уравнений (7), получаем

$$\dot{V}_2 = 2\eta\left(-\sigma\eta + \Omega\dot{y}\right) = -2\sigma\eta^2 + 2\Omega\eta c^T Ax - 2k\Omega\eta c^T by + 2(\mu+k)\Omega c^T b\eta^2 + 2\Omega\eta c^T q\varphi(y) -$$

$$- 2(\mu+k)\Omega\eta c^T by^5 + 2\eta\Omega c^T b\overline{w}, \qquad (15)$$

где вместо составляющей $\dot{y}$ в уравнении (15) было использовано слагаемое

$$\dot{y} = c^T\left(Ax - kby - (\mu+k)by^5 + (\mu+k)b\eta + b\overline{w} + q\varphi(y)\right).$$

Принимая во внимание соотношения

$$2\Omega\eta c^T Ax \le \delta^{-1}c^T AA^T c\Omega^2\eta^2 + \delta x^T x, \; -2k\Omega\eta c^T by \le k^2\left(c^T b\right)^2\Omega^2\eta^2 + y^2,$$

$$2\Omega\eta c^T q\varphi(y) \le k\left(c^T q\right)^2\Omega^2\eta^2 + k^{-1}\left[\varphi(y)\right]^2, \quad 2\eta\Omega c^T b\overline{w} \le k(c^T b)^2\Omega^2\eta^2 + k^{-1}\overline{w}^2,$$

$$-2(\mu+k)\Omega\eta c^T by^4 y \le (\mu+k)^2(c^T b)^2\Omega^2 y^8\eta^2 + y^2,$$

для производной от функции (12) имеем:

$$\dot{V}_2 \le -2\sigma\eta^2 + \delta^{-1}c^T AA^T c\Omega^2\eta^2 + \delta x^T x + k^2\left(c^T b\right)^2\Omega^2\eta^2 + y^2 + 2(\mu+k)\Omega c^T b\eta^2 +$$

$$+ k\left(c^T q\right)^2\Omega^2\eta^2 + k^{-1}\left[\varphi(y)\right]^2 + (\mu+k)^2(c^T b)^2\Omega^2 y^8\eta^2 + y^2 +$$

$$+ k(c^T b)^2\Omega^2\eta^2 + k^{-1}\overline{w}^2. \qquad (16)$$

Тогда для производной от функции Ляпунова (10) получаем

$$\dot{V} = \dot{V}_1 + \dot{V}_2 \le x^T\left(-Q_1 + \delta Pqq^T P + \delta I\right)x - (k-3)y^2 - 2(\mu+k)y^6 + \delta^{-1}\left[\varphi(y)\right]^2 + \frac{2}{k}\overline{w}^2 +$$

$$+ (-2\sigma + (\mu+k)^2 + \delta^{-1}c^T AA^T c\Omega^2 + k^2\left(c^T b\right)^2\Omega^2 + 2(\mu+k)\Omega c^T b +$$

$$+ k\left(c^T q\right)^2\Omega^2 + (\mu+k)^2(c^T b)^2\Omega^2 y^8 + k(c^T b)^2\Omega^2)\eta^2 + k^{-1}\left[\varphi(y)\right]^2. \qquad (17)$$

Выберем число $\delta > 0$ таким образом, чтобы было выполнено неравенство

$$\left(-Q_1 + \delta Pqq^T P + \delta I\right) \le -Q_2, \qquad (18)$$

где $Q_2 = Q_2^T$ – положительно определенная матрица.

Выберем функцию σ так, чтобы выполнялось соотношение

$$-2\sigma + (\mu+k)^2 + \delta^{-1}c^T AA^T c\Omega^2 + k^2\left(c^T b\right)^2\Omega^2 + 2(\mu+k)\Omega c^T b +$$

$$+ k\left(c^T q\right)^2\Omega^2 + (\mu+k)^2(c^T b)^2\Omega^2 y^8 + k\left(c^T b\right)^2\Omega^2 \le -\lambda, \qquad (19)$$

где число $\lambda > 0$.

Тогда, учитывая ограничения, налагаемые на нелинейность, для производной от функции Ляпунова (10) получаем

$$\dot{V} \le -x^T Q_2 x - \lambda\eta^2 - (k-3)y^2 - 2(\mu+k)y^6 + \left(\frac{1}{\delta} + \frac{1}{k}\right)\cdot y^6 + \frac{2}{k}\overline{w}^2. \qquad (20)$$

Выбирая число $k > 3$ следующим образом

$$k > \frac{1}{\delta} + \frac{1}{k}, \tag{21}$$

получаем

$$\dot{V} \le -x^T Q_2 x - \lambda\eta^2 + \frac{2}{k}\overline{w}^2, \tag{22}$$

Из неравенства (22), в силу ограниченности возмущения $|w(t)| \le w_0 < \infty$, следует, что существует такое число $k > 3$, что траектории системы (7), (9) могут быть сведены в любую заданную область ε_0, что и требовалось доказать.

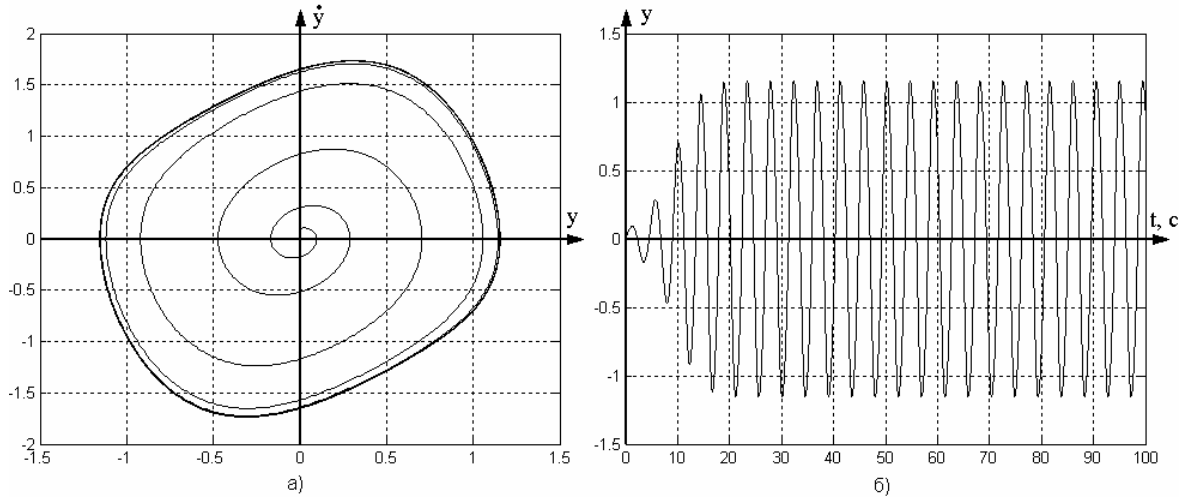


Рис. 1. Фазовый портрет (а) и переходный процесс (б) в системе (1) при $\dot{y}(0) = 0{,}1$, $w(t) = 0$

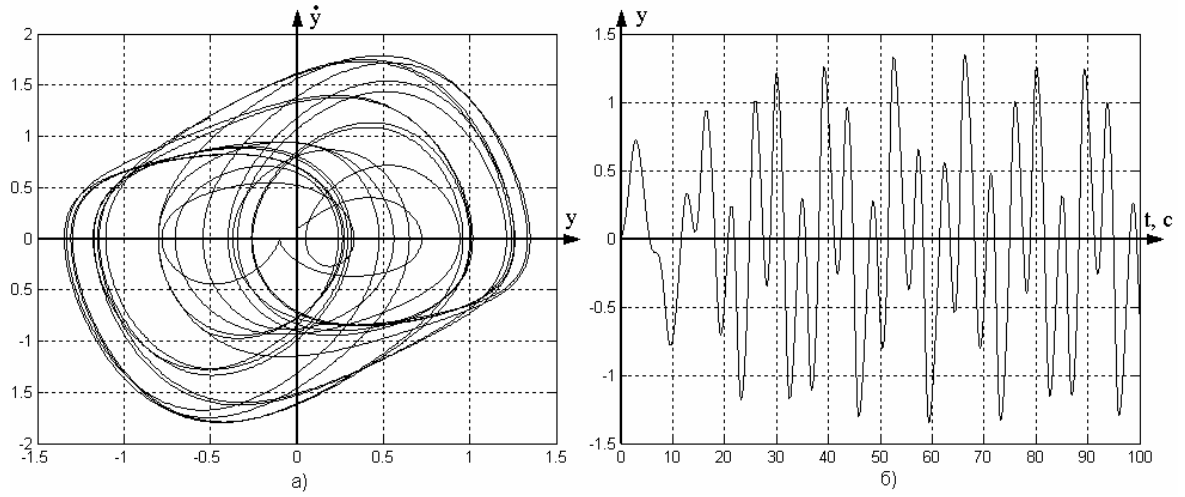


Рис. 2. Фазовый портрет (а) и переходный процесс (б) в системе (1) при $\dot{y}(0) = 0{,}1$, $w(t) = \sin(0{,}5t)$

Рассмотрим результаты компьютерного моделирования системы Ван дер Поля. Сначала будем полагать, что $w(t) = 0$ и в системе (1) возникают установившиеся колебания (система обладает устойчивым предельным циклом). Далее при гармоническом возмущении $w(t)$ обнаружим в системе (1) хаотические явления. И, наконец, на последнем этапе моделирования системы (1) с управлением (3), (4) обнаружим достижение заданной цели управления.

На рис. 1 и 2 приведены результаты компьютерного моделирования невозмущенной системы (1) при $\tau_1 = 0{,}5$, $\tau_2 = 2$, $w(t) = 0$ и возмущенной системы (1) при $\tau_1 = 0{,}5$, $\tau_2 = 2$,, $w(t) = \sin(0{,}5t)$ соответственно. Из результатов следует, что система (1) обладает устойчивым предельным циклом при отсутствии возмущающего воздействия ($w(t) = 0$), а при гармоническом возмущающем воздействии ($w(t) = \sin(0{,}5t)$) в ней появляются хаотические процессы. Для стабилизации системы (1) воспользуемся алгоритмом управления (3). Выберем полином $\chi(p) = p + 1$, тогда

$$u = -(p+1)(\mu + k)\xi = -(\mu + k)(\xi + \dot{\xi}), \qquad (23)$$

где функция ξ формируется алгоритмом оценки (4).

Функция σ выбирается, чтобы выполнялось соотношение (19), то есть

$$\sigma = (\mu + k)^2 + \left(1 + 2k + k^2 + (\mu + k)^2 y^{4\tau - 4}\right)\Omega^2 + 2(\mu + k)\Omega. \qquad (24)$$

Выберем параметр $\mu = 2$, и промоделируем систему управления для различных значений параметра k. Переходные процессы в замкнутой системе при ненулевых начальных условиях ($\dot{y}(0) = 0{,}1$) для значений параметров $k = 10$, и $k = 25$ представлены на рис. 3.

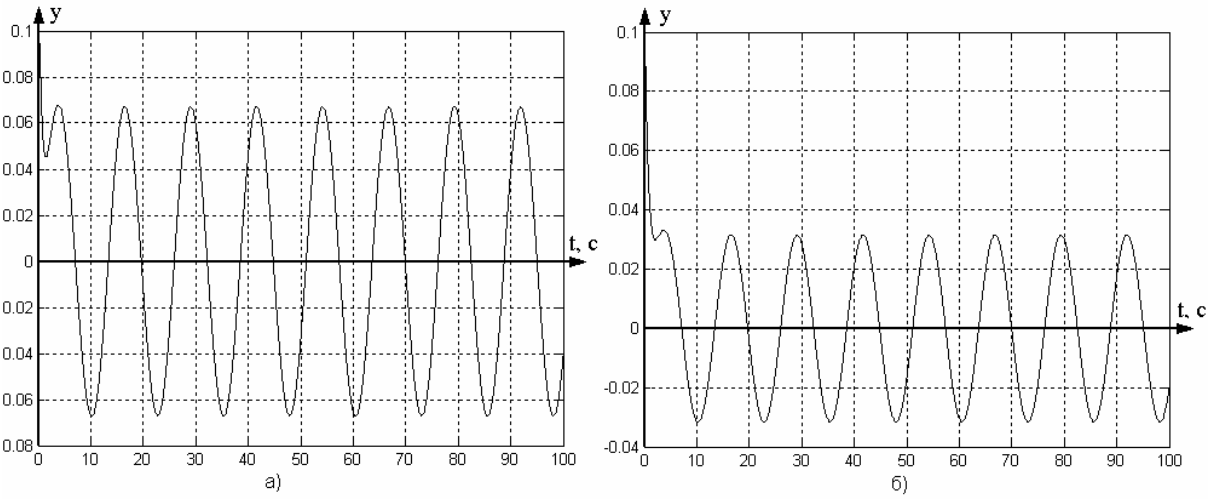


Рисунок 3 –Переходные процессы в системе (1), (23), (24) при $k = 10$ (а), $k = 25$ (б), $w(t) = \sin(0{,}5t)$

Временные диаграммы иллюстрируют работоспособность предложенного в работе алгоритма управления и достижение заданной цели управления. Из временных диаграмм видно, что выходная траектория системы $y(t)$ ограничена некоторой областью ε_0, величина которой, в свою очередь, уменьшается с увеличением коэффициента k.

Заключение

В работе предложен алгоритм управления по выходу нелинейной хаотической системой, описываемой уравнением Ван дер Поля. Предложенный алгоритм управления, обеспечивает сходимость выходной траектории нелинейной системы в некоторую область ε_0, причем эта область может быть уменьшена за счет увеличения параметра регулятора k.

Литература

1. Андриевский Б.Р., Фрадков А.Л. Управление хаосом: методы и приложения. Часть 1. Методы // АиТ. 2003. №5.
2. Андриевский Б.Р., Фрадков А.Л. Управление хаосом: методы и приложения. Часть 2. Приложения // АиТ. 2004. №4.
3. Андриевский Б.Р., Фрадков А.Л. Элементы математического моделирования в программных средах MATLAB5 и Scilab. СПб, 2001.
4. T. Gilbert and R.V. Gammon. Stable oscillations and devil's staircase in the Van der Pole oscillator // International journal of bifurcation and chaos. 2000. Vol.10. No. 1. P. 155-164.
5. Бобцов А.А., Николаев Н.А. Синтез управления нелинейными системами с функциональными и параметрическими неопределенностями на основе теоремы Фрадкова // АиТ. 2005. №1.

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ КОНФЛИКТОВ В ТЕХНИЧЕСКОМ ТВОРЧЕСТВЕ

А.Б. Бушуев

На пересечении теории катастроф и гомеостатики предложен метод синтеза моделей и количественные оценки стереотипов конфликтного поведения

Введение

Конфликтные ситуации используются в техническом творчестве для формирования в сознании изобретателя логической модели изобретательской задачи в виде технического противоречия (ТП) [1]. ТП – это противоречие между двумя конкурирующими частями, характеристиками, свойствами, показателями системы. В известном алгоритме решения изобретательских задач (АРИЗ) ТП состоит из двух составляющих:– ТП-1 и ТП-2. Например, ТП-1 – если автобус большой, то он комфортабельный, но не маневренный; с другой стороны, ТП-2 – если автобус маленький, то он маневренный, но не комфортабельный. Таким образом, свойства маневренности и комфортабельности при изменении конфликтной координаты, т.е. размера транспортного средства, конкурируют между собой. В измерительных устройствах конкурируют чувствительность и диапазон измерений, в процессах обработки – производительность и точность, и т.д.

Выявляя в прототипе ТП, развивая, усиливая, обостряя его, изобретатель подготавливает свое сознание к качественному скачку – появлению нового решения, которое, по законам диалектики, возникает после максимального обострения конфликта и разрешает противоречие. Скачкообразные, резкие изменения свойств системы при плавном изменении ее параметров изучаются в теории катастроф [2]. Поэтому в [3] предложена математическая модель ТП, использующая каноническую катастрофу типа «сборки».

Типы канонических катастроф (складка, сборка, ласточкин хвост, различные омбилики и т. п.) отличаются друг от друга видом потенциальной функции, которой они задаются. Катастрофа типа сборки задается потенциальной функцией $V(q)$, имеющей математическое выражение

$$V(q) = 0.25q^4 - 0.5\lambda \cdot q^2 - \mu \cdot q\,, \qquad (1)$$

где q – координата состояния катастрофы, λ и μ – управляющие параметры. Коранг катастрофы определяется количеством координат состояния, для сборки коранг равен 1.

Для физических систем потенциальная функция отождествляется с потенциальной энергией. При $\lambda < 0$ и $\mu = 0$ график потенциальной энергии $V=V(q)$ имеет один минимум (grad $V(x)=0$) , который определяет единственное и устойчивое состояние равновесия $q=0$, в которое и стремится система, называемая в этом случае градиентной. При переходе через критическое значение $\lambda = 0$ в градиентной системе происходит катастрофа: резко меняются ее свойства. При $\lambda > 0$ вместо одного состояния равновесия появляются три: два устойчивых $q = \pm\sqrt{\lambda}$ и одно неустойчивое $q=0$.

В изобретательской задаче в сознании изобретателя потенциальная функция отождествляется с нежелательным, отрицательным эффектом, которым обладает прототип изобретения. Считается, что прототип, например, понятие автобуса среднего размера, находится до катастрофы ($\lambda < 0$) в устойчивом равновесии $q=0$, т.е. в «яме» потенциальной функции. Появление ТП при $\lambda = 0$ «раскалывает» прототип: его состояние равновесия $q=0$ дает максимум потенциальной функции, т.е. прототип «взлетает» на вершину горы нежелательного эффекта и становится неустойчивым. Зато появляются два минимума нежелательного эффекта при $q = \pm\sqrt{\lambda}$, в одно из которых «сваливается» понятие маленького автобуса максимальной маневренности, а в другое – понятие

большого автобуса максимальной комфортабельности. ТП считается градиентной системой, поэтому дифференциальное уравнение развития конфликта получается из условия равенства антиградиента потенциальной функции и скорости изменения конфликтной координаты во времени

$$KT\frac{dq}{dt} = KT\dot{q} = -\frac{\partial V(q)}{\partial q} = -gradV(q) = -q^3 + \lambda \cdot q + \mu \ , \qquad (2)$$

где T – постоянная времени, учитывающая инерционность мышления изобретателя, K – масштабирующий множитель, λ – мощность конфликта [4], μ – объем внешнего ресурса, решающего задачу (в АРИЗе он называется Х-элементом). Мощность λ определяет свободное развитие конфликта ($\mu = 0$), так как задает расстояние между минимумами потенциальной функции в закритичной области ($\lambda > 0$). Х-элемент появляется ($\mu \neq 0$) после максимального обострения конфликта и определяет вынужденное его движение к разрешения ТП. Одно из возможных решений в примере – переменность длины большого автобуса («гармошка» или сочлененный автобус).

Недостатком модели (2) является ее одномерность, тогда как каждая из двух сторон ТП может иметь свою динамику. Например, одновременное повышение точности и устойчивости системы не всегда является антагонистичным. Следовательно, между двумя сторонами ТП могут складываться различные сценарии поведения, включающие и возможный компромисс. Поэтому в данной работе в рамках теории катастроф для количественной оценки конфликта предлагается двухкоординатная динамическая модель развития антагонистических свойств в виде компенсационного гомеостата [5], а также метод синтеза простых гомеостатов для моделирования стереотипов взаимодействия конфликтующих систем.

Динамическая модель собственного движения конфликта

Для двухмерной задачи используем каноническую катастрофу (назовем ее производящей) типа «гиперболическая омбилика» коранга 2, потенциальная функция которой задается выражением

$$V(x, y) = x^3 + y^3 - axy + bx + cy \, , \qquad (3)$$

где x и y – координаты состояния катастрофы, a,b,c – управляющие параметры.

Приравнивая антиградиент потенциальной функции вектору скоростей координат x и y, получаем систему дифференциальных уравнений

$$\dot{x} = -(3x^2 - ay + b), \quad \dot{y} = -(3y^2 - ax + c) \, . \qquad (4)$$

Рассмотрим собственное движение. При b=c=0 имеем

$$\dot{x} = -3x^2 + ay, \quad \dot{y} = -3y^2 + ax \, . \qquad (5)$$

Пусть x=-z, тогда, подставляя x=-z в (3) и (5), получаем уравнения для координат,

$$\dot{z} = 3z^2 - ay, \quad \dot{y} = -3y^2 - az \, , \qquad (6)$$

и для потенциальной функции,

$$V(y, z) = y^3 - z^3 + ayz \, . \qquad (7)$$

Приравнивая левые части уравнений (6) нулю, находим физически реализуемые и устойчивые состояния равновесия (при $a > 0$): $y_{уст} = a/3$, $z_{уст} = -a/3$.

Предположим, что конфликтующие части ТП при собственном движении развиваются во времени антисимметрично [6], каждая по своей логистической кривой. Логиста (или S-кривая) моделирует, в частности, закон размножения и гибели популяций и является решением уравнения Ферхюльста-Перла

$$\dot{u} = -mu^2 + nu, \quad \dot{w} = mw^2 + nw, \tag{8}$$

где u и w – относительные координаты, задающие эволюцию конфликтующих сторон, m и n – коэффициенты рождаемости и смертности. Первое уравнение системы (8) предназначено для координаты $u>0$, второе уравнение – для координаты $w<0$. Свойство антисимметричности задается уравнением

$$w = -u. \tag{9}$$

Относительная координата получается путем деления абсолютной координаты на модуль ее установившегося значения при собственном движении. Тогда, приравнивая левые части уравнений (8) нулю, получим устойчивые стационарные точки собственного движения: $u_{уст} = n/m = +1$, $w_{уст} = -n/m = -1$. Относительность координат позволяет сравнивать по величине конфликтующие части ТП, имеющие различную физическую природу.

Физически количество особей в популяции, точность системы, быстродействие не могут быть отрицательными. Поэтому знак «минус» приписывается координате условно, чтобы показать, что ее развитие противоположно развитию другой координаты.

При подходящем выборе коэффициентов уравнения (6) и (8) эквивалентны. Действительно, если каждое из уравнений (8) поделить на m и умножить на 3, тогда, с учетом того, что $w = -u$, получим

$$3\dot{u}/m = -3u^2 - aw, \quad 3\dot{w}/m = 3w^2 - au \ , \tag{10}$$

где $a = 3m/n = 3$. Левую часть (6) для выравнивания размерностей умножаем на KT:

$$KT\dot{z} = 3z^2 - ay, \qquad KT\dot{y} = -3y^2 - az. \tag{11}$$

При $KT = 3/m$, $u \equiv y$, $w \equiv z$, (10) и (11) полностью эквивалентны. Движение (10), (11) при эволюции по многообразию $z = -y$ является двумя раскалывающимися логистами (рис.1, точечные кривые 1 на этапе обострения конфликта). Будем считать, что $KT = 1$, т.е. системы (10) и (11) записаны в относительном времени. Кроме того, из условия $u \equiv y$, $w \equiv z$ следует, что u и z – также относительные координаты, тогда их установившееся значения равны соответственно 1 и -1.

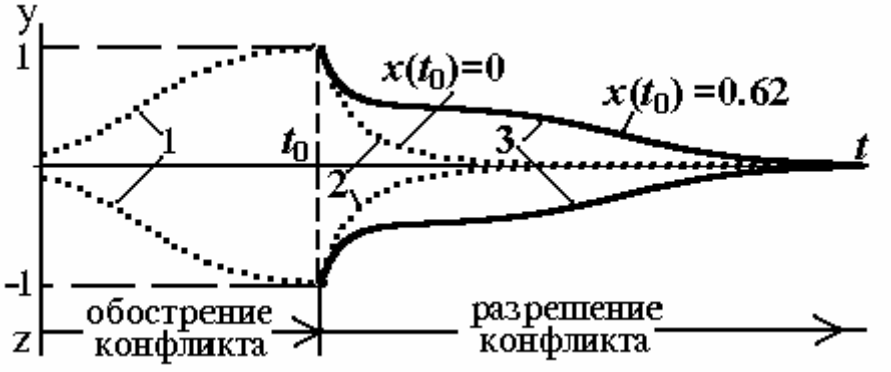


Рис. 1. Эволюция конфликта

Оценка конкурентной ситуации

Система уравнений (8) совместно с уравнением многообразия (9) обладает избыточностью. Таким образом, математика подсказывает, что эта система уравнений описывает явление гомеостаза, важнейшим свойством которого является избыточность. Гомеостаз – это способность поддерживать основные цели функционирования организации в условиях внутренних противоречий и внешних воздействий [5]. Гомеостаз характерен для живых организмов, например, у человека одна система все время повышает уровень сахара в крови, а другая система, работающая на другом физиологиче-

ском принципе, все время понижает уровень сахара в крови. В результате устанавливается динамическое равновесие уровня сахара. В технике гомеостаз используется, например, в двухвинтовом вертолете, лопасти винтов которого вращаются в противоположных направлениях.

Система, реализующая гомеостаз, называется гомеостатом. Компенсационный гомеостат представляет собой двухканальную систему, в которой конкурирующие каналы-антагонисты «склеиваются» перекрестными связями. Гомеостат потребляет больше энергии (информации), чем необходимо для работоспособности отдельных антагонистов. Через перекрестные связи антагонисты борются между собой, расходуя дополнительную энергию и информацию, хотя работают на одну общую главную функцию. Внутренняя борьба «закаляет», тренирует антагонистов, поэтому гомеостат обладает высокой потенциальной готовностью для противодействия внешним вредным возмущениям.

Степень гомеостаза (избыточность, острота борьбы антагонистов) в общем случае зависит от структуры перекрестных связей. Однако в системах (6) и (8) реализован так называемый инвариант компенсационного гомеостата, у которого степень гомеостаза не зависит от структуры перекрестных связей.

Используя условие $y=- z$, из (11) можно получить еще две модели инварианта

$$\dot{z} = -3zy - ay, \qquad \dot{y} = 3yz - az \ , \tag{12}$$

$$\dot{z} = 3y^2 - ay, \qquad \dot{y} = -3z^2 - az \ . \tag{13}$$

Легко показать, что системы (12) и (13) полностью эквивалентны системам (6) и (8), хотя структура перекрестных связей у всех инвариантов различна, а в системе (8) и вовсе отсутствует. Следовательно, перекрестные связи при собственном развитии конфликта взаимно компенсируются. Именно в этом смысле можно утверждать, что собственное движение всех гомеостатов является инвариантом по отношению к их структуре, к потерям энергии и информации в борьбе антагонистов. Количественной оценкой инварианта является величина потенциальной функции (7) производящей катастрофы в установившемся режиме собственного движения $V_{уст}(y, z) =-1$, которая задает начало отсчета.

При вынужденном движении гомеостатов под действием внешнего возмущения проявляется различие структуры, что показало моделирование систем уравнений (6), (8), (12), (13). Возмущающий входной сигнал в виде единичного ступенчатого воздействия подавался на канал y. Начальные условия $y(0)=0.1$, $z(0)=-0.1$, $a=3$, $b=0$, $c=-1$. Потенциальная функция $V(y,z)$ рассчитывалась по (3) с учетом $x=-z$. Результаты моделирования представлены в табл.1.

№ п/п	стереотип поведения	модель гомеостата	$V_{уст}$	$y_{уст}$	$z_{уст}$
1	компромисс	уравнения (12)	-1.963	1.333	-1.0
2	безразличие	уравнения (8)	-2.037	1.264	-1.0
3	конкуренция	уравнения (11)	-2.101	1.194	-1.093
4	конфронтация	уравнения (13)	$-\infty$	$-\infty$	∞

Таблица 1. Оценка стереотипов поведения

Из табл. 1 следует, что входной сигнал снижает потенциальные функции всех гомеостатов по сравнению с инвариантом. Поэтому антагонисты вынуждены за счет запаса начальных условий бороться за «выживание». Острота борьбы характеризуется снижением потенциальной функции относительно инварианта. Наиболее острая борьба характерна для 4-го стереотипа поведения, которая приводит гомеостат к неустойчиво-

сти и гибели, т.е. к разрешению противоречия. Другой крайний случай – 1-й стереотип, когда конфликтующие стороны идут на частичный компромисс, что позволяет добиться наименьшего снижения потенциальной функции по сравнению с другими стереотипами.

Очевидно, что по мере решения по АРИЗу изобретательская задача проходит последовательно все стадии обострения – от первой до четвертой, а в сознании изобретателя происходит перестройка структуры гомеостата противоречия. Потенциальная функция (или нежелательный эффект) снижается Разрешение противоречия, или рождение нового изобретения, означает изменение инварианта гомеостата – переход на согласованное собственное движение по многообразию $y=z$. При вынужденном согласованном движении стереотипы поведения становятся партнерскими и союзническими.

Синтез простых моделей гомеостатов

Используя канонические катастрофы, можно получать модели, отражающие основные стереотипы поведения взаимодействующих систем. Порядок синтеза моделей следующий:

1. Выбираем одну из канонических катастроф коранга 1 для моделирования одной системы или катастрофу коранга 2 для моделирования двух систем. Для большего числа систем можно взять несколько производящих катастроф коранга 1 и 2.
2. Считая системы градиентными, находим антиградиенты потенциальных функций и приравниваем их вектору скоростей координат. Получаем систему дифференциальных уравнений.
3. Находим параметры уравнений, обеспечивающие устойчивое собственное движение (или неустойчивое – для моделирования расходящихся процессов).
4. Переходя к зеркальному ($z=-y$) или прямому ($z=y$) отображениям, получаем инварианты стереотипов поведения соответственно конфликтных и согласованных взаимодействий.
5. Для численной оценки взаимодействий определяем начало отсчета потенциальной функции как установившееся значение потенциальной функции инварианта производящей катастрофы.
6. Для численной оценки вынужденного взаимодействия используем полное выражение потенциальной функции производящей катастрофы.

Рассмотрим пример синтеза гомеостата с двумя уровнями иерархии как модель малого коллектива (рис.2).

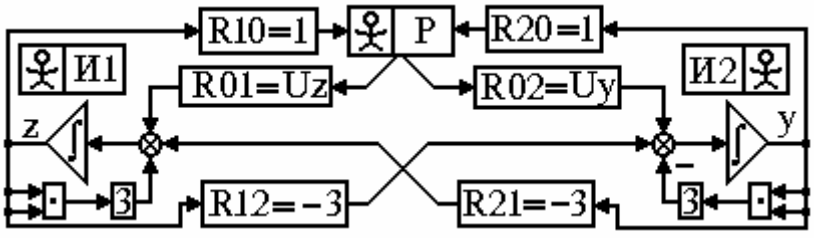


Рис. 2. Структура простейшего двухуровневого гомеостата

Нижний уровень представлен исполнителями И1 и И2 (терминология в соответствии с [5]), между которыми существует стереотип – конкуренция, задаваемая отношениями $R12$ и $R21$. Верхний уровень представлен руководителем Р, между которым и исполнителями И1, И2 существуют стереотипы взаимодействий, определяемые отношениями $R10$, $R01$ и $R20$, $R02$. В теории управления И1 и И2 являются каналами с перекрестными связями, а Р – регулятором, генерирующим управления Uz и Uy. В АРИЗе

исполнителями являются противоположные стороны ТП, а руководителем – Х-элемент, разрешающий противоречие.

Синтезируем гомеостат, в котором руководитель разрешает максимально обострившийся конфликт между исполнителями. Этап обострения конфликта (рис. 1) моделируем уже рассмотренными уравнениями (11) производящей гиперболической омбилики. В конце этапа y=+1, z=-1.На этом этапе руководитель не участвует.

На этапе разрешения конфликта в качестве производящей катастрофы используем «эллиптическую омбилику» с потенциальной функцией

$$V(x,y) = y^3 - 3x^2 y + a(x^2 + y^2) + by + cx .$$

После приравнивания антиградиента потенциальной функции и вектора скоростей координат x и y, получаем систему уравнений собственного (b=c=0) движения координат

$$\dot{x} = 6yx - 2ax, \quad \dot{y} = -3y^2 + 3x^2 - 2ay . \quad (14)$$

Назначая a=1.5, получаем устойчивое равновесие в точке y=0, x=0, поскольку задачей управления является перевод координаты y в 0. При y=-z из (14) получаем уравнения для зеркальной системы

$$\dot{x} = -6zx - 3x, \quad \dot{z} = 3z^2 - 3x^2 - 3z \quad (15)$$

Подставляем y=0.5(y-z) в первое уравнение системы (14), а z=0.5(z-y) в первое уравнение системы (15), находим инвариант первого уравнения (14) и (15)

$$\dot{x} = 3xy - 3xz - 3x . \quad (16)$$

Таким образом, система уравнений

$$\dot{y} = -3y^2 + 3x^2 - 3y, \quad \dot{z} = 3z^2 - 3x^2 - 3z, \quad \dot{x} = 3xy - 3xz - 3x \qquad (17)$$

задает движение гомеостата на этапе разрешения конфликта.

Синтезируем управление Uy и Uz, переводящее гомеостат (11) на движение по уравнениям (17) из условия тождественности уравнений (11) и (17) для y и z:

$$\dot{z} = 3z^2 - 3y + Uz = 3z^2 - 3x^2 - 3z, \qquad \dot{y} = -3y^2 - 3z + Uy = -3y^2 + 3x^2 - 3y,$$

откуда

$$Uz = -3x^2 + 3(y - z), \quad Uy = -Uz = 3x^2 - 3(y - z) . \qquad (18)$$

Первое слагаемое $\pm 3x^2$ можно назвать «мягкой» или динамической частью управления, так как его генерирует 3-е уравнение системы (17), имеющее собственную динамику. Если начальное условие $x(t_0)$ выбрать нулевым, где t_0 – начало разрешения конфликта, то «мягкого» управления не будет, и под действием «жесткого» управления ±3(y-z) конфликтные координаты движутся по быстро сходящимся логистам (точечные кривые 2 на рис. 1). При $|x(t_0)| \neq 0$ конфликт спадает медленнее (сплошные кривые 3 на рис.1). Чем больше $|x(t_0)|$, тем больше конфликт затягивается. При $|x(t_0)| \geq 0.627$ гомеостат неустойчив, кривые расходятся. Следовательно, изменяя $x(t_0)$, можно изменять качество переходных процессов, т.е. моделировать стиль руководства при разрешении конфликта.

Заключение

По результатам проведенных исследований можно сделать следующие выводы.

1. Степень гомеостаза можно количественно оценить изменением установившегося значения потенциальной функции производящей катастрофы.

2. Теория катастроф и стереотипы поведения диалектически тесно связаны. Катастрофы задают эволюцию состояний равновесия, на которые «навешивается» динамика

переходных процессов. Стереотипы поведения – притяжение и отталкивание для примитивных систем, конкуренция, сотрудничество, компромисс, подчинение и т.п. для более развитых систем – задают опорные точки, градации конфликтного и согласованного взаимодействий. Класс математического аппарата соответствует классу описываемого процесса, поэтому и модели получаются простыми – на уровне параметров порядка.

Литература

1. Альтшуллер Г.С. Найти идею. Новосибирск: Наука. Сиб. отд-ние, 1991.
2. Постон Т., Стюарт И. Теория катастроф и ее приложения. М.:Мир, 1980.
3. Бушуев А.Б., Мансурова О.К. Катастрофа типа «сборки» в изобретательской задаче // Научно-технический вестник СПбГИТМО (ТУ). Вып.11. СПб, 2003. С.137–140.
4. Bushuev A. Technical Contradiction Control on Invention Problem // The TRIZ Journal, December 2004. < http://www.triz-journal.com>
5. Горский Ю.М. Основы гомеостатики. Гармония и дисгармония в живых, природных, социальных и искусственных системах. Иркутск: Изд-во ИГЭА,1998.
6. Бушуев А.Б. Гомеостатика противоречий в ТРИЗ // Труды Международной конференции MA TRIZ Fest -2005 "Развитие ТРИЗ: достижения, проблемы, перспективы". СПб, 2005. С.103–109. Эл. версия Трудов на сайте: < http://www.metodolog.ru >.

САМООРГАНИЗУЮЩЕЕСЯ УСТРОЙСТВО ДИСКРЕТНОЙ АВТОМАТИКИ В ЗАДАЧАХ ДИНАМИЧЕСКОЙ ИДЕНТИФИКАЦИИ ПРОЦЕССОВ БЕЗ ПАМЯТИ НАД ПОЛЕМ $GF(2)$

А.А. Мельников

Рассматривается задача идентификации процессов без памяти над конечным полем Галуа $GF(2)$. Предложена аналитическая база для конструирования самоорганизующегося устройства дискретной автоматики, позволяющего осуществлять идентификацию процессов без памяти над $GF(2)$.

Введение

Ставится и решается задача идентификации [4–6] процессов, параметризованных дискретным временем k, без памяти над конечным полем Галуа $GF(2)$, протекающих в некоторой технической среде. При этом целевой частью задачи является формирование аналитической базы для конструирования устройства дискретной автоматики (УДА), позволяющего решать указанную задачу в режиме реального времени. Решение поставленной задачи идентификации осуществляется параллельно в двух ее версиях – структурной и параметрической. Первая своей целью ставит формирование в среде двоичных динамических систем (ДДС) необходимой модели для осуществления процедуры идентификации, вторая – идентификацию параметров модели технической среды. Совокупно задача решается в среде УДА средствами самоорганизующегося двоичного динамического наблюдающего устройства, коими обеспечивается характеристика самоорганизации устройства.

Основной результат

Предварим полученные результаты решения задачи определением.

Определение. Под *дискретным процессом без памяти над полем* $GF(2)$ будем понимать процесс φ, происходящий в некоторой технической среде и характеризующийся тем, что для каждого j-го воздействия u, $\dim u = r$ на техническую среду, представленного набором переменных u: $\left\{u_i,\ i=\overline{1,r}\right\}_{j=\overline{1,2^r}}$, $u_i \in \{0,1\}$, он всегда формирует однозначный выход (отклик) $\nu_j \in \{0,1\}$ так, что

$$\varphi:\ \nu(k) = f(u(k)).\ \square \qquad (1)$$

По существу, описание (1) задает отношение «*вход – выход*» некоторой технической среды, которая формирует сигнал выхода как булеву функцию (БФ), зависящую исключительно от входного воздействия u. Введенное определение позволяет уточнить формулировку поставленной задачи идентификации следующим образом. Требуется сформировать аналитическую базу конструирования УДА, которое по представлению на его вход всех 2^r пар $\{u(k), \nu(k)\}$ «самосконфигурируется» и обеспечит решение поставленной задачи с формированием описания модели технической среды в форме (1). Структурно формулировку задачи можно представить так, как показано на рис. 1.

Задача решается средствами УДА, построенного в форме двоичного динамического наблюдающего устройства (ДНУ) при использовании возможностей нейросетевых технологий [1,3,7,8]. В такой постановке аналитическое описание процедуры самоорганизации ДНУ по аналогии с оной в среде нейронных сетей, где обеспечивается на-

стройка весов синаптических связей [1,7] нейронов и формирование целевого выхода, над полем $GF(2)$ примет вид

$$z(k+1)=\mathbf{A}z(k)+\mathbf{B}u(k)+\mathbf{L}e(k); \quad (2)$$

$$w(k)=\mathbf{R}z(k), \quad (3)$$

$$y(k)=\mathbf{C}w(k), \quad (4)$$

где z – вектор состояния, $e=y+\nu$ – вектор невязки наблюдения (ВНН), w – вектор синаптических весов, y – вектор выхода, $\mathbf{A}$ – единичная матрица состояния, $\mathbf{B}$ – матрица входа, $\mathbf{L}$ – матрица входа по ВНН, $\mathbf{R}=\mathbf{B}^T$ – матрица выхода-выборки весов, $\mathbf{C}=\mathbf{B}^T$ – матрица выхода, при этом матричные компоненты по размерности согласованы с соответствующими векторами z,u,e,w, а операция «+» с учетом модулярной арифметики в $GF(2)$ осуществляется по модулю два ($\mathrm{mod}\,2$).

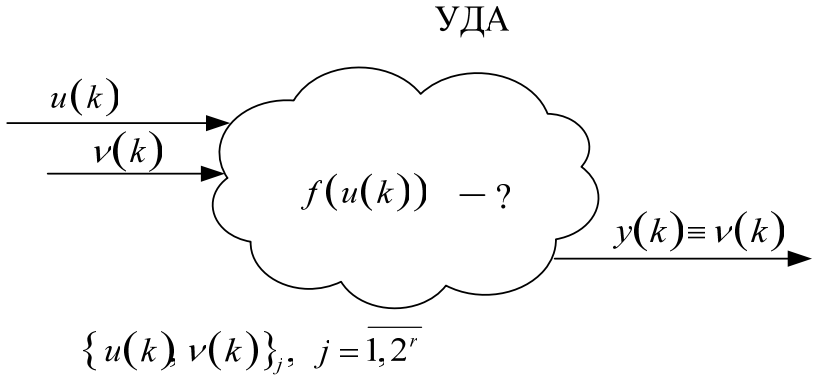


Рис. 1. Пояснение формулировки задачи

Заметим, что реализация БФ в рамках алгебры Буля не использует линейную операцию умножения и сложения по $\mathrm{mod}\,2$, представляющие операции линейной алгебры. Указанное обстоятельство приводит к проблеме конструирования линейного ДНУ (2)–(4), настраиваемого на произвольную БФ (1): система (2)–(4) не может обеспечить функциональную полноту для реализации произвольной БФ. Проблема решается представлением произвольной БФ в базисе Жегалкина [2] с линейными операциями умножения и сложения по $\mathrm{mod}\,2$ алгебры логики, в которую входит и алгебра Буля. Погружение БФ в базис Жегалкина приводит к полиномам Жегалкина, которые в развернутой форме имеют вид

$$f(x)=f(0)\oplus\sum_{i=1}^{n}\alpha_{\eta,i}x_i\cdot\oplus\sum_{\substack{i,j=1\\ i\neq j}}^{n}\alpha_{\eta-1,i}\,x_ix_j\cdot\oplus\ldots\oplus\sum_{i_1,i_2,\ldots i_m=1}^{n}\alpha_{\eta-m,i}x_{i1}x_{i2}\ldots x_{im}\cdot\oplus\ldots$$

$$\ldots\oplus\alpha_{1,i}x_1x_2\ldots x_n,\quad \eta=2^n-n-1, \quad (5)$$

где $f(0)$ – значение БФ на наборе переменных $x_1x_2\ \ldots\ x_n$, имеющем нулевое значение, $i_1,i_2,\ldots i_n\in(\overline{1,n})$ и попарно не равны друг другу, $\alpha_{j,i}\in\{0,1\}$ – коэффициенты полинома.

Расширим линейное пространство, образованное булевыми переменными x_i, $i=\overline{1,n}$, элементами $\tilde{x}_\rho$, представляющими собой конъюнкции переменных в полиноме (5) так, что

$$\{\widetilde{x}\}:\ \begin{cases} \widetilde{x}_\ell = \{x_i x_j\}_\ell; \quad i,j=\overline{1,n},\ i\neq j,\ \ell=\overline{1,C_n^2}; \\ \widetilde{x}_{C_n^2+\ell} = \{x_i x_j x_k\}_\ell; \quad i,j,k=\overline{1,n},\ i\neq j\neq k,\ \ell=\overline{1,C_n^3}; \\ \vdots \\ \widetilde{x}_\eta = x_1 x_2 \ldots x_n, \end{cases} \tag{6}$$

где $C_\bullet^\circ$ — число сочетаний из $(\bullet)$ элементов по $(\circ)$ выбранным элементам, представляющих собой переменные x_i, $i=\overline{1,n}$, и вычисляемое в силу правила $C_n^m = \dfrac{n!}{m!(n-m)!}$.

При этом расширим соответствующим образом и вектор w весов синаптических связей так, что $\dim\widetilde{w}=\eta$. Тогда модель (2)–(4) процессов в ДНУ примет вид

$$\widetilde{\widetilde{z}}(k+1) = \widetilde{\mathbf{A}}\widetilde{\widetilde{z}}(k) + \widetilde{\mathbf{B}}u(k) + \widetilde{\mathbf{L}}\left[\widetilde{\mathbf{C}}\widetilde{\mathbf{R}}\widetilde{\widetilde{z}}(k) + \nu(k)\right]; \tag{7}$$

$$y(k) = \widetilde{\mathbf{C}}\widetilde{\mathbf{R}}\widetilde{\widetilde{z}}(k), \tag{8}$$

где, с учетом отождествления переменных в форме $z_i \equiv x_i, \widetilde{z}_j \equiv \widetilde{x}_j$ агрегированный вектор $\widetilde{\widetilde{z}}$ состояния ДНУ имеет вид $\widetilde{\widetilde{z}} = \left[z^T \ \vdots\ \widetilde{z}^T\right]^T$.

Методологически процесс самоконфигурации ДНУ осуществляется в силу следующих соображений. Строится линейная двоичная динамическая система, которая для каждой пары $\{k=0,k\}$ дискретных моментов времени фиксирует булеву разность $\partial u(k)$ значений вектора u, $\dim u = r$ в форме

$$\partial u(k) = u(0) \oplus u(k). \tag{9}$$

Далее аналогичным образом строится еще одна линейная ДДС, которая для каждой пары $\{k=0,k\}$ дискретных моментов времени фиксирует булеву разность $\partial e(k)$ значений вектора ν так, что

$$\partial e(k) = \nu(0) \oplus \nu(k). \tag{10}$$

С использованием значений булевых разностей (9), (10) осуществляется процедура «самоконфигурации» ДНУ. Суть процедуры самоконфигурации ДНУ сформулируем в форме следующего утверждения.

Утверждение. Для формирования аналитического представления скалярной БФ $f(u(k))$, $\dim u = r$ в форме полинома Жегалкина достаточно вычислить в силу соотношений (9), (10) булевы разности $\partial u(k)$ и $\partial e(k)$ на множестве полной мощности 2^r пар $\{u(k), \nu(k)\}$ (см. рис. 1) при зафиксированной паре $\{u(0), \nu(0)\}$, при этом для каждой пары $\{k, k+1\}$ дискретного времени заполнить *карту модулярных сумм* (КМС) таблицы 1 булевых разностей (ТБР) в силу следующего правила. Графа $\nu(0)$ КМС для всех итераций заполняется значением $\nu(0)\big|_{u(0)}$. Далее заполнение КМС производится так, что напротив булевого терма, соответствующего значению $\partial u(k)=1$ на текущей паре $\{u(k), \nu(k)\}$, ставится единица, если сумма по $\mathrm{mod}\,2$ уже имеющихся единиц в КМС на соответствующих конъюнкциях булевых переменных не дает верное значение $\partial e(k)$ для этой пары. Итерации заканчиваются, когда вариации $\partial e(k)$ на очередных всех 2^r парах $\{u(k), \nu(k)\}$ не приводят к изменению КМС. При этом число ℓ итераций по формированию КМС не превысит r так, что

$$\ell \le r. \tag{11}$$

Заполнение графы «*Совокупная сумма по каждому терму*» КМС осуществляется по завершению итераций постолбцовым суммированием по $\mathrm{mod}\,2$ единиц, содержащихся в строках при соответствующих булевых термах.

Таблица 1

Булевы термы полинома Жегалкина	Итерации				Совокупная сумма по каждому терму
	1	2	…	r	
$\nu(0)$	$\nu(0)$				$\nu(0)$
u_1	$(\bullet)_{11}$	$(\bullet)_{12}$	…	$(\bullet)_{1r}$	$\sum_{i=1}^{r}(\bullet)_{1i}$
u_2	$(\bullet)_{21}$	$(\bullet)_{22}$	…	$(\bullet)_{2r}$	$\sum_{i=1}^{r}(\bullet)_{2i}$
⋮	⋮	⋮	⋮	⋮	⋮
u_r	$(\bullet)_{r1}$	$(\bullet)_{r2}$	…	$(\bullet)_{rr}$	⋮
u_1u_2	$(\bullet)_{r+1\,1}$	$(\bullet)_{r+1\,2}$	…	$(\bullet)_{r+1\,r}$	⋮
u_1u_3	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮
u_1u_r	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮
u_2u_3	⋮	⋮	⋮	⋮	⋮
u_2u_4	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮
u_2u_r	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮
$\&_{j=1}^{r} u_j$	$(\bullet)_{\chi 1}$	$(\bullet)_{\chi 2}$	…	$(\bullet)_{\chi r}$	$\sum_{i=1}^{r}(\bullet)_{\chi i}$

Карта модулярных сумм булевых разностей

Восстановление искомой БФ $f(u(k))$ вида (1) в форме полинома (5) по построенной КМС производится суммированием по $\mathrm{mod}\,2$ тех термов, напротив которых графа «*Совокупная сумма по каждому терму*» КМС содержит единицу, при этом переменные булевых термов, которые соответствуют прямому (не инверсному) значению их в векторе $u(0)$, следует проинвертировать. □

Доказательство утверждения строится на процедуре (Теорема Т1 В.А. Горбатова) [2] разложения произвольной БФ в полином Жегалкина. ■

Проиллюстрируем использование положений утверждения на примере.

Пример

Требуется сформировать БФ $\nu = f(u_3, u_2, u_1)$, принимающую значение логической единицы на наборах $\{\bar{u}_3, u_2, u_1\}$, $\{u_3, \bar{u}_2, \bar{u}_1\}$, $\{u_3, u_2, \bar{u}_1\}$ при условии $u(0) = \{0\ 1\ 1\}$. □

В соответствии с положениями утверждения будем иметь максимальное число итераций, равное трем ($\ell = 3$), и ТБР в форме табл. 2. В таблице обозначение $1_{(\bullet)}$ имеет смысл того, что соответствующая ячейка была заполнена единицей по счету $(\bullet)$ формирования единиц. Совокупная сумма единиц КМС по каждому терму дает единицу, за

исключением терма $\bar{u}_1\bar{u}_3$ для которого она дает $1_6 \oplus 1_9 = 0$. Конструирование по столбцу «*Совокупная сумма по каждому терму*» ТБР искомой БФ в форме полинома Жегалкина дает:

$$f(u_3, u_2, u_1) = 1 \oplus \bar{u}_1 \oplus \bar{u}_2 \oplus u_3 \oplus \bar{u}_1\bar{u}_2 \oplus \bar{u}_2 u_3 \oplus \bar{u}_1\bar{u}_2 u_3 .$$

Из таблицы видно, что число итераций, потребовавшихся для заполнения КМС, оказалось равным двум, что не противоречит постановочной части утверждения. ■

Таблица 2

Булевы термы полинома Жегалкина	Итерации			Совокупная сумма по каждому терму
	1	2	3	
$\nu(0)$	1_1			1
$\bar{u}_1$	1_4	—	—	1
$\bar{u}_2$	1_3	—	—	1
u_3	1_7	—	—	1
$\bar{u}_1\bar{u}_2$	1_2	—	—	1
$\bar{u}_1 u_3$	1_6	1_9	—	0
$\bar{u}_2 u_3$	—	1_8	—	1
$\bar{u}_1\bar{u}_2 u_3$	1_5	—	—	1
			Карта модулярных сумм	

По завершению процедуры самоорганизации ДНУ (2)–(4) последнее коммутируется с режима обучения на режим формирования сигналов целевого выхода так, что с учетом структуры агрегированного вектора $\tilde{\tilde{z}}$ состояния ДНУ описание этого режима имеет вид

$$\tilde{\tilde{z}}(k+1) = \tilde{\mathbf{A}}\tilde{\tilde{z}}(k); \qquad (12)$$

$$y(k) = \tilde{\mathbf{C}}\tilde{\mathbf{R}}\tilde{\tilde{z}}(k). \qquad (13)$$

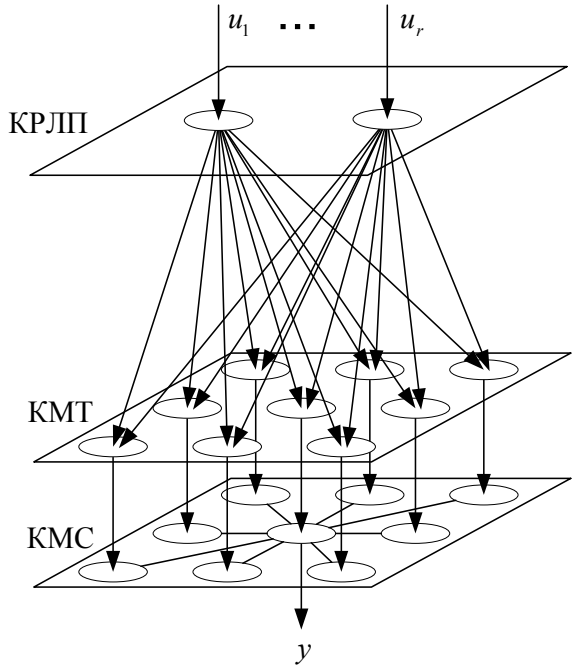


Рис. 2. Результирующая структура в форме нейронной сети

Результаты исследований, выполненных в параграфе, позволяют сформировать структуру (рис. 2) ДНУ в форме нейронной сети, схожей с сетью Кохонена [3, 7, 8]. Сформированная сеть включает: карту расширения линейного пространства – КРЛП, карту модулярных термов – КМТ и карту модулярных сумм – КМС. Карта КРЛП вычислений не выполняет, ее назначение – разветвление входных сигналов u_i, $i=\overline{1,r}$ с весом $w_{ji}, i=\overline{1,r}$; $j=\overline{1,2^r}$ на все с учетом процедуры (6) расширения линейного пространства $j=2^r$ нейронов карты КМТ. Последняя карта – КМС – осуществляет суммирование по $\mathrm{mod}\,2$ сигналов выхода всех нейронов КМТ, чем обеспечивает формирование выходного целевого сигнала y сети.

Результат работы сети в форме настроенных весов синаптических связей позволяет напрямую получить из них коэффициенты $\alpha_{j,i}$ полинома (5), представляющего собой искомую БФ вида (1) описания процесса без памяти, протекающего в среде исследуемой технической среды.

Заключение

Содержательная часть данной работы представляет достаточную аналитическую базу, позволяющую строить самоорганизующееся УДА, по структуре подобное нейронной сети, решающее поставленную задачу в режиме реального времени. Следует ожидать, что распространение полученных результатов на решение задач идентификации процессов с памятью над $GF(2)$, модельно представимые в форме конечного автомата Мура или автомата Мили, даст позитивный результат, а структура УДА при этом будет иметь иерархический вид.

Литература

1. Головко В.А. Нейронные сети: обучение, организация и применение. Кн. 4: Учеб. пособие для вузов / Общая ред. А. И. Галушкина. М.: ИПРЖР, 2001.
2. Горбатов В.А. Фундаментальные основы дискретной автоматики. Информационная математика. М.: Наука. Физматлит, 1999.
3. Круглов В.В., Борисов В.В. Искусственные нейронные сети. Теория и практика. М.: Горячая линия - Телеком, 2001.
4. Кузовков Н.Т., Карабанов В.А., Салычев О.С. Непрерывные и дискретные системы управления и методы идентификации. М.: Машиностроение, 1978.
5. Пупков К.А., Егупов Н.Д., Трофимов А.И. Статистические методы анализа, синтеза и идентификации систем автоматического управления. М. МГТУ им. Н.Э. Баумана, 1998.
6. Современные методы идентификации систем / Под ред. П. Эйхскоффа. М.: Мир, 1983.
7. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика. М.: Мир, 1992.
8. Kohonen T. Self-organization and associative memory. Series in Information Sciences, vol. 8. Berlin: Springer Verlag, 1984.

ИСПОЛЬЗОВАНИЕ СИСТЕМНЫХ ГРАМИАНОВ В ЗАДАЧАХ ПАРАМЕТРИЧЕСКОЙ ИНВАРИАНТНОСТИ НЕПРЕРЫВНЫХ СИСТЕМ

Т.А. Акунов, О.В. Слита, А.В. Ушаков

Рассматриваются возможности использования системных грамианов в задачах параметрической инвариантности непрерывных систем управления. Формулируется алгоритм оценки достигаемой ε – инвариантности.

Введение

Проблема обеспечения параметрической инвариантности выхода проектируемой системы относительно модельной неопределенности объекта всегда была и остается важной задачей современной теории управления, так как решение данной проблемы гарантирует стабильность функционирования системы в составе технической среды обслуживаемого технологического процесса. Системные грамианы [1, 2] как один из способов математического описания системы могут быть использованы при решении задачи количественной оценки достигаемой параметрической инвариантности.

Первоначально грамианы использовались для анализа структурных свойств – управляемости и наблюдаемости объектов управления [2]. Однако грамианы можно использовать для того, чтобы проверить, является ли система параметрически инвариантной, для контроля ε–инвариантности [3] системы, а также для ранжирования неопределенностей по степени их влияния на выход системы.

Данная работа посвящена использованию возможностей системных грамианов в задаче оценки эффекта введения в систему закона управления, доставляющего ей параметрическую инвариантность, и для контроля ε– инвариантности системы.

1. Постановка задачи параметрической инвариантности

Рассматривается линейный непрерывный объект, неопределенность знания параметров структурных компонентов которого путем выбора соответствующего базиса [4,5] представима неопределенностью задания его матрицы состояния так, что его уравнение динамики принимает вид:

$$\dot{x}(t) = (A + \Delta A)x(t) + Bu(t); \quad x(0); \quad y(t) = Cx(t). \qquad (1)$$

В выражении (1) x, u, y – соответственно векторы состояния, управления и выхода; $x \in R^n$, $u \in R^r$, $y \in R^m$, A, B, C – соответственно номинальная компонента матрицы состояния объекта управления (ОУ), его матрицы управления и выхода, согласованные по размерности с векторными переменными: $A \in R^{n\times n}$, $B \in R^{n\times r}$, $C \in R^{m\times n}$, ΔA – матричная вариация матрицы состояния.

Закон управления (ЗУ) синтезируется в виде прямой связи по внешнему задающему воздействию $g(t)$ с матрицей K_g и обратной связи по состоянию объекта $x(t)$ с матрицей K, образующих аддитивную композицию, в предположении полной их измеримости

$$u(t) = K_g g(t) - Kx(t). \qquad (2)$$

Структурное объединение ОУ (1) и ЗУ (2) образует систему с векторно-матричным представлением

$$\dot{x}(t) = Fx(t) + Gg(t) + \Delta Fx(t); \quad x(0); \qquad (3)$$

$$y(t) = Cx(t); \quad e(t) = g(t) - y(t). \qquad (4)$$

В (3), (4) $e(t)$ – ошибка воспроизведения системой задающего воздействия, матрицы F и G имеют представление

$$F = A - BK,\ G = BK_g, \tag{5}$$

ΔF – матричная вариация матрицы состояния системы, удовлетворяющая равенству

$$\Delta F = \Delta A. \tag{6}$$

Инвариантность выхода системы (а, следовательно, и ошибки) можно записать в форме

$$y(t, g(t), F, \Delta F = \Delta A \neq 0) = y(t, g(t), F, \Delta F = \Delta A \equiv 0). \tag{7}$$

Представим матричную вариацию $\Delta F = \Delta A$ в аддитивной форме так, чтобы выполнялось условие

$$p = \arg\min\left\{\Delta A = \sum_{j=1}^{p} \Delta A_j \ \&\ rank\Delta A = 1\right\}. \tag{8}$$

При этом мультипликативный компонент $\Delta F\, x(t) = \Delta A x(t)$ в (3) представим в форме:

$$\begin{aligned}\Delta F\, x(t) = \Delta A x(t) = &\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}\begin{bmatrix}\Delta A_{11} & \Delta A_{12} & \dots & \Delta A_{1n}\end{bmatrix}\begin{bmatrix}x_1(t)\\x_2(t)\\\vdots\\x_n(t)\end{bmatrix} + \\ &+\begin{bmatrix}0\\1\\\vdots\\0\end{bmatrix}\begin{bmatrix}\Delta A_{21} & \Delta A_{22} & \dots & \Delta A_{2n}\end{bmatrix}\begin{bmatrix}x_1(t)\\x_2(t)\\\vdots\\x_n(t)\end{bmatrix} + \\ &+\dots+\begin{bmatrix}0\\0\\\vdots\\1\end{bmatrix}\begin{bmatrix}\Delta A_{n1} & \Delta A_{n2} & \dots & \Delta A_{nn}\end{bmatrix}\begin{bmatrix}x_1(t)\\x_2(t)\\\vdots\\x_n(t)\end{bmatrix}\end{aligned} \tag{9}$$

Первые сомножители (матрицы-столбцы D_j размерности $(n \times 1)$) слагаемых соотношения (9) позволяют сформировать матрицу D в виде

$$D = row\{D_j, j = 1, p\}. \tag{10}$$

Сформируем вектор внешнего «параметрического» воздействия

$$\zeta(t) = col\{\zeta_j(t),\ j = 1, p\},\ \ p \le n\ , \tag{11}$$

где j–е компоненты ζ_j строятся на мультипликативных векторных структурах

$$\begin{bmatrix}\Delta A_{j1} & \Delta A_{j2} & \dots & \Delta A_{jn}\end{bmatrix}\begin{bmatrix}x_1(t)\\x_2(t)\\\vdots\\x_n(t)\end{bmatrix}. \tag{12}$$

Объединение (11) и (12) позволяет представить векторно-матричный компонент $\Delta F\, x(t) = \Delta A x(t)$ в форме:

$$\Delta Ax(t) = \sum_{j=1}^{p} D_j \zeta(t)_j = D\zeta(t)\,. \qquad (13)$$

Подстановка (10) с учетом (9) в (3) позволяет записать:

$$\dot{x}(t) = Fx(t) + Gg(t) + D\zeta(t);\ y(t) = Cx(t)\,. \qquad (14)$$

Поставленная задача обеспечения параметрической инвариантности в форме (7) при использовании модели (14) принимает вид

$$y(t, F, g(t), \zeta(t) \neq 0) = y(t, F, g(t), \zeta(t) \equiv 0)\,. \qquad (15)$$

Соотношение (15), по существу, содержит доказательство следующего утверждения [6].

Утверждение 1. Чтобы система (9) обладала параметрической инвариантностью в смысле условия (15), т.е. чтобы передаточная функция (матрица) «параметрический вход ζ – выход системы y» $\Phi_{y\zeta}(s)$ была бы нулевой, а именно выполнялось равенство

$$\Phi_{y\zeta}(s) = C(sI - F)^{-1} D = 0\,. \qquad (16)$$

достаточно, чтобы столбцы D_j матрицы D были бы собственными векторами матрицы F, при этом они принадлежали ядру матрицы C. □

Доказательство. Пусть D_j является собственным вектором матрицы F, соответствующим ее собственному значению λ_j, что позволяет записать

$$FD_j = \lambda_j D_j\,. \qquad (17)$$

Воспользуемся свойством матричной функции $f((*))$ от матрицы $(*)$ сохранять геометрический спектр собственных значений исходной матрицы и иметь в качестве элементов алгебраического спектра собственных значений компоненты $f(\lambda_j)$ [3], которое позволяет записать

$$f(F)D_j = f(\lambda_j)D_j\,. \qquad (18)$$

Если (18) подставить в (16), где матричная функция от $f(F)$ имеет вид $f(F) = (sI - F)^{-1}$, то получим

$$\Phi_{y\zeta_j}(s) = C(sI - F)^{-1} D_j = C(s - \lambda_j)^{-1} D_j = (s - \lambda_j)^{-1} CD_j\,. \qquad \blacksquare \qquad (19)$$

Заметим, что в случае невыполнения условия (16), гарантирующего полную параметрическую инвариантность выхода, оно может быть заменено контролем близости передаточной функции (16) к нулю путем оценки ее нормы. Необходимо сказать, что контроль близости к нулю передаточной функции путем оценки ее нормы является непростой математической проблемой. В этой связи возникшая трудность может быть преодолена, если воспользоваться аппаратом системных грамианов.

2. Системные грамианы в задаче формирования оценки параметрической ε–инвариантности. Основной результат

Невыполнение условия (16), по существу, переводит задачу обеспечения параметрической инвариантности выхода непрерывной системы к неопределенностям матричных компонентов исходного объекта из разряда полной (абсолютной) параметрической инвариантности в разряд ε–инвариантности. Для оценки величины ε достигаемой ε–инвариантности воспользуемся возможностями системных грамианов.

В проблемно-ориентированном виде интерес представляет грамиан управляемости системы (14) применительно к динамическому каналу «параметрический вход $\zeta(t)$ – выход системы $y(t)$».

Грамиан управляемости канала «параметрический вход – выход системы» имеет вид

$$W_{y\zeta} = \int Ce^{Ft} DD^T e^{Ft} C^T dt = CW_{x\zeta} C^T , \qquad (20)$$

где $W_{x\zeta}$ – грамиан управляемости по состоянию, который для устойчивых систем имеет стационарную реализацию, вычисляемую в силу решения уравнения типа Ляпунова [1, 4]

$$FW_{x\zeta} + W_{x\zeta} F^T = -DD^T . \qquad (21)$$

Для оценки величины ε, до которой в случае невыполнения условия (16) обеспечивается параметрическая инвариантность выхода относительно параметрического внешнего воздействия $\zeta(t)$, вычислим спектр сингулярных чисел [6] грамиана $W_{y\zeta}$

$$\sigma_\alpha = \{W_{y\zeta}\} = \{\alpha_{jy\zeta},\ j = 1, m\}. \qquad (22)$$

Максимальное сингулярное число $\alpha_{y\zeta M}$ определяет максимальное значение ε, а минимальное сингулярное число $\alpha_{y\zeta m}$ определяет минимальное значение, так что в общем случае, когда непрерывная система является системой с многомерным выходом, величина ε задается оценочными неравенствами

$$\alpha^{1/2}{}_{y\zeta M} \le \varepsilon \le \alpha^{1/2}{}_{y\zeta m} . \qquad (23)$$

Нетрудно видеть, что, если непрерывная система имеет скалярный выход, то соотношение (23) содержит в себе доказательство следующего утверждения.

Утверждение 2. Если алгебраический спектр сингулярных чисел системного грамиана $W_{y\zeta}$ содержит только нулевые элементы, то система (14) характеризуется параметрической инвариантностью выхода относительно параметрического внешнего воздействия $\zeta(t)$, или, что то же самое, относительно параметрической неопределенности матричных компонентов исходного ОУ (1). □

Полученные оценки величины ε могут быть использованы в качестве показателя достигнутой инвариантности до ε в смысле оценки нормы передаточной матрицы динамического канала ««параметрическое воздействие -–выход системы».

В случае, если при синтезе регуляторов встает задача сравнения конкурирующих вариантов по степени достижения величины ε в задаче обеспечения параметрической ε–инвариантности, то математически корректным является использование обобщенного характеристического уравнения, построенного на паре грамианов $W_{y\zeta}$, $\widetilde{W}_{y\zeta}$.

$$W_{y\zeta} = CW_{x\zeta} C , \qquad (24)$$

$$\widetilde{W}_{y\zeta} = C\widetilde{W}_{x\zeta} C , \qquad (25)$$

где $W_{x\zeta} = \arg\{FW_{x\zeta} + W_{x\zeta} F^T = -DD^T\}$, $\widetilde{W}_{x\zeta} = \arg\{F\widetilde{W}_{x\zeta} + \widetilde{W}_{x\zeta} F^T = -DD^T\}$.

Это обобщенное характеристическое уравнение имеет вид:

$$\det(\mu\widetilde{W}_{y\zeta} - W_{y\zeta}) = 0 . \qquad (26)$$

Если корни этого уравнения будут меньше единицы, то вариант, характеризующийся системным грамианом $W_{y\zeta}$ в смысле достигаемой ε–параметрической инвариантности, будет предпочтительнее варианта, характеризующегося системным грамианом $\widetilde{W}_{y\zeta}$.

Таким образом, оценка достигаемой ε–инвариантности может быть осуществлена с помощью следующего алгоритма.

1) Построить векторно-матричное представление ОУ в форме (1) так, чтобы все неопределенные компоненты были бы сосредоточены только в матрице состояния.
2) Синтезировать ЗУ вида (2), который доставлял бы номинальной версии (G, F, C) системы $(G, F + \Delta F, C)$ требуемые динамические свойства при произвольной структуре собственных векторов.
3) Представить мультипликативный компонент $\Delta Fx(t) = \Delta Ax(t)$ в форме (9) и составить матрицу D.
4) Решить уравнение (21) относительно грамиана управляемости по состоянию $W_{x\zeta}$ и вычислить грамиан управляемости по выходу $W_{y\zeta}$.
5) Вычислить спектр сингулярных чисел грамиана $W_{y\zeta}$.
6) Оценить значение ε с помощью неравенств (23).

Заключение

Аппарат системных грамианов в задачах параметрической инвариантности в случае, когда исследуемая система представляет собой системы типа MIMO (многомерный вход – многомерный выход) позволяет сконструировать скалярную оценку интегрального эффекта достижения параметрической инвариантности. Это особенно важно в тех случаях, когда условие $CD = 0$ выполняется не по всем m выходам многомерной системы.

Литература

1. Мироновский Л.А. Функциональное диагностирование динамических систем: Научное издание / СПб, 1998.
2. Акунов Т.А., Алишеров С., Оморов Р.О., Ушаков А.В. Матричные уравнения в задачах управления и наблюдения непрерывными объектами. Бишкек: Илим, 1991.
3. Уланов Г.М. Инвариантность до ε в комбинированных системах автоматического регулирования. – Сборник «Теория инвариантности и ее применение в автоматических устройствах». Труды совещания, состоявшегося в Киеве 16-го октября 1958 г. М., 1959.
4. Андреев Ю.Н. Управление конечномерными линейными объектами. М.: Наука, 1976.
5. Справочник по теории автоматического управления / Под ред. А.А. Красовского. М.: Наука, 1987.
6. Никифоров В.О., Ушаков А.В. Управление в условиях неопределенности: чувствительность, адаптация, робастность. СПб: СПб ГИТМО (ТУ), 2002.
7. Голуб Дж., Ван Лоун Ч. Матричные вычисления / Пер. с англ. М.: Мир, 1999.

ВЫРОЖДЕНИЕ СЛОЖНЫХ ДИНАМИЧЕСКИХ СИСТЕМ С РАВНОТЕМПОВЫМИ СТРУКТУРНЫМИ КОМПОНЕНТАМИ

Н. А. Дударенко, А. В. Ушаков

Рассматривается проблема вырождения сложных динамических систем с равнотемповыми структурными компонентами. Поставленная задача решается с помощью аппарата вещественнозначных передаточных матриц вход-выход, сконструированных на основе решения матричного уравнения Сильвестра. Для количественной оценки вырождения используются функционалы вырождения.

Введение

Проблемы вырождения сложных динамических систем типа многомерный вход-многомерный выход, т.е. систем MIMO-типа, распадаются на две группы: первая группа состоит в содержательной постановке задачи вырождения, вторая – в конструировании инструментария контроля вырождения. Это важное для функционирования сложных динамических систем свойство остается пока малоизученным [1, 2].

Содержательно вырождение сложной системы состоит в такой ее функциональной деформации, при которой размерность ее функциональных возможностей сокращается. Математически вырождение означает сокращение размерности образа линейного оператора, реализуемого системой, который отображает «пространство намерений» в «пространство осуществляемых реализаций», т.е. изменение ранга этого оператора. Ранг оператора является целочисленной характеристикой. В этой связи нужен такой инструментарий, который позволил бы непрерывно оценивать появляющуюся в системе тенденцию к возможному ее вырождению. Этот инструментарий строится на использовании сингулярного разложения [3, 4] критериальных матриц, позволяющего контролировать их обусловленность, а, следовательно, склонность системы к вырождению.

Очевидно, источников вырождения сложной динамической системы достаточно много. В данной работе рассматриваются проблемы, связанные с вырождением сложных динамических систем с равнотемповыми структурными компонентами, которые «генетически» являются идеально обусловленными. Однако по причинам параметрического характера, проявляющегося в появлении перекрестных связей, нарушении равнотемповости, неравномерном распределении заявок по входам, это свойство может быть утрачено, и система может приобрести склонность к вырождению.

1. Постановка задачи. Инструментарий контроля

Конструирование инструментария контроля вырождения матриц операторов “вход-выход” (МОВВ) сложных систем будем осуществлять, опираясь на возможности сингулярного разложения матриц.

Пусть задана сложная многомерная динамическая система, реализующая некий линейный оператор с матрицей N.

Утверждение 1. *Произвольная матрица N порождает линейную алгебраическую задачу (ЛАЗ) вида*

$$\eta(w) = N(w,\theta)\chi(w) \qquad (1)$$

где $N(w,\theta)$ – $p \times p$ – матрица для любых w, θ; $\eta(w)$, $\chi(w)$ – p-мерные векторы, θ – ρ-мерный параметр, изменяющий алгебраические свойства матрицы N. □

Доказательство утверждения строится на сингулярном (SVD) разложении матрицы N [5]. ■

Геометрическая интерпретация исходной ЛАЗ (1) состоит в том, что единичная сфера в пространстве, натянутом на векторы χ, отображается в эллипсоид, положение

полуосей которого определяется элементами левого сингулярного базиса, а размер полуосей совпадает с сингулярными числами матрицы N.

Будем рассматривать ЛАЗ как инструментальную модель контроля вырождения на спектре сингулярных чисел, порождаемых ими сепаратных чисел обусловленности или функционалов сепаратного вырождения.

Степень близости матрицы N к ее вырождению может быть оценена с помощью глобального числа обусловленности $C\{N\}$ этой матрицы, при этом решение задачи заметно обогатится, если помимо глобального числа обусловленности $C\{N\}$ контролируются ее сепаратные числа обусловленности $C_j\{N\}$, определяемые на спектре сингулярных чисел матрицы N с помощью соотношения

$$C\{N\} = \alpha_M\{N\}\alpha_m^{-1}\{N\},\ \ C_v\{N\} = \alpha_M\{N\}\alpha_{v+1}^{-1}\{N\}; \tag{2}$$

где $\alpha_M\{N\} = \alpha_1\{N\}$, $\alpha_m\{N\} = \alpha_p\{N\}$, $\alpha_{v+1}\{N\}, v = \overline{0, p-1}$, – соответственно максимальное, минимальное и v-ое сингулярные числа матрицы N, вычисляемые в силу соотношения

$$\alpha_j = \left|\mu_j^{1/2}\right|; j = \overline{1, p},\quad \mu_j : \det(\mu I - N^T N) = 0. \tag{3}$$

Числа обусловленности для оценки вырождения позволяют сконструировать функционалы вырождения J_{Dv}, задаваемые соотношением

$$J_{Dv} = C_v^{-1}\{N\}. \tag{4}$$

По свойству чисел обусловленности [3, 4] функционал вырождения удовлетворяет неравенствам:

$$0 \le J_{Dv} = C_v^{-1}\{N\} \le 1. \tag{5}$$

Таким образом, процесс вырождения ЛАЗ можно отслеживать по последовательному обнулению функционалов вырождения J_{Dv}, контроль граничных значений которых в пределах 0 и 1 заметно проще контроля граничных значений в пределах 1 и ∞ чисел обусловленности.

Поставим задачу конструирования критериальных матриц N операторов вырождения вход-выход сложных динамических систем с последующим применением к ним разработанной схемы контроля вырождения.

2. Банк критериальных матриц в задаче контроля вырождения сложных динамических систем

Задачу построения критериальных матриц будем решать на примере непрерывной многомерной системы вида

$$\dot{x}(t) = Fx(t) + Gg(t);\ x(0);\ y(t) = Cx(t). \tag{6}$$

где x, g, y – векторы состояния, задающего воздействия и выходы соответственно, $x \in R^n$; $g, y \in R^m$; F, G, C – матрицы состояния системы, входа и выхода объекта управления, согласованные по размерности с размерностью векторов x, g и y так, что $F \in R^{n\times n}$; $G, C^T \in R^{n\times m}$.

2.1. Критериальные матрицы во временной области

Рассмотрим поведение системы (6) при конечномерном внешнем воздействии. Источник внешнего конечномерного воздействия зададим в форме автономной непрерывной системы, имеющей представление

$$\dot{z}(t) = Ez(t);\ z(0);\ g(t) = Pz(t), \tag{7}$$

где $z \in R^l$, $E \in R^{l \times l}$; $P \in R^{m \times l}$; z – вектор состояния модели задающего воздействия (МЗВ), E, P – матрицы состояния и выхода МЗВ соответственно, причем матрица P удовлетворяет условию: $P \cdot P^T = I$, где I – единичная матрица размерности $m \times m$.

Утверждение 2. *Решение системы (6) для переменных состояния* $x(t)$ *и выхода* $y(t)$ *для случая задающего воздействия* $g(t)$ *вида (7) может быть записано в форме*

$$x(t) = e^{Ft}x(0) + [Te^{Et} - e^{Ft}T]z(0), \quad (8)$$

$$y(t) = Ce^{Ft}x(0) + C[Te^{Et} - e^{Ft}T]z(0), \quad (9)$$

где матрица T *ищется из решения матричного уравнения Сильвестра [1,2]*

$$TE - FT = GP. \quad (10)$$

□

Доказательство утверждения можно найти в работе [6]. ■

Рассмотрим соотношения (8) и (9) для ступенчатого входного воздействия, для которого оказываются справедливыми соотношения

$$E = |0|, \; e^{Et} = I, \quad (11)$$

матрица T решения уравнения Сильвестра (10) принимает вид

$$T = -F^{-1}GP. \quad (12)$$

В силу (11) и (12) соотношения (8) и (9) записываются в форме

$$x(t) = e^{Ft}x(0) + (e^{Ft} - I)F^{-1}Gg(0), \quad (13)$$

$$y(t) = Ce^{Ft}x(0) + C(e^{Ft} - I)F^{-1}Gg(0). \quad (14)$$

Нетрудно видеть, что (13), (14) сводят задачу исследования процессов в сложной MIMO-системе (6) во временной области при ступенчатом воздействии к линейной алгебраической задаче (1), где критериальная матрица $N(w,\theta)$ принимает смысл матриц $H_x = (e^{Ft} - I)F^{-1}G$ и $H_y = C(e^{Ft} - I)F^{-1}G$ при описании вынужденных составляющих движений системы (6), и матриц $H_{свx}(t) = e^{F(t)}$ и $H_{св_y}(t) = Ce^{F(t)}$ при описании свободных составляющих этих движений. Сравнение (1) с (13) и (14) позволяет заметить, что вектор $\eta(w)$ реализуется в форме векторов $x(t)$ и $y(t)$, а вектор $\chi(w)$ соответствует векторам $z(0)$ и $x(t_n)$ для вынужденной и свободной составляющих соответственно, переменная *w* является непрерывным временем *t*, а параметр θ, осуществляющий модификацию матрицы *N*, задается параметрами исходной матрицы *F*.

Таким образом, вынужденная составляющая движения системы MIMO-типа (6) описывает поведение системы на этапе «принятия решения на выполнение заявки», а задание свободной составляющей движения описывает поведение системы на этапе «выполнения заявки» на основе накопленного сложной системой ресурса в форме $x(t_n) = (e^{Ft_n} - I)F^{-1}Gg(0) \cong -F^{-1}Gg(0)$ на фиксированный момент времени t_n.

2.2. Критериальные матрицы при внешнем гармоническом воздействии

Теперь рассмотрим случай поведения системы (6) при внешнем гармоническом воздействии, формируемом системой вида (7), где $z \in R^l$, $E \in R^{l \times l}$; $P \in R^{m \times l}$; z – вектор состояния МЗВ, E, P – матрицы состояния и выхода МЗВ соответственно. МЗВ выбирается минимальной размерности, но такой, чтобы ее выход

$$g(t) = Pz(t), \text{ где } z(t) = e^{Et}z(0) \quad (15)$$

на множестве начальных состояний $z(0)$ адекватно представлял весь класс конечномерных задающих воздействий системы (6), представляющих собой линейную комбинацию гармоник. Задача построения критериальных матриц ЛАЗ (1) для случая гармонического воздействия осуществляется в силу утверждения.

Утверждение 3. *Для случая многочастотного гармонического задающего воздействия $g(t)$ задача управления системой (6) по переменным $x(t)$ и $y(t)$ сводится к линейной алгебраической задаче вида (1), записываемой в форме*

$$x(t) = N_x(t,\Omega)z(0), \quad y(t) = N_y(t,\Omega)z(0), \tag{16}$$

где

$$N_x(t,\Omega) = Tdiag\{e^{E_{jj}t} = \begin{vmatrix} \cos\omega_j t & \sin\omega_j t \\ -\sin\omega_j t & \cos\omega_j t \end{vmatrix}; j = \overline{1,m}\}, \quad N_y(t,\Omega) = CN_x(t,\Omega) \tag{17}$$

$$\Omega = col(\omega_j, j = \overline{1,m}), \tag{18}$$

матрица T удовлетворяет уравнению Сильвестра (10). □

Доказательство утверждения можно найти в работе [7]. ■

Для случая многомерного одночастотного воздействия частоты ω матрица T имеет вид $T(\omega) = row\{-[F^2 + \omega^2 I]^{-1}[FG_j \;\; \omega G_j] : j = \overline{1,m}\}$, а для случая многомерного многочастотного воздействия $T(\Omega) = row\{-[F^2 + \omega_j^2 I]^{-1}[FG_j \;\; \omega_j G_j] : j = \overline{1,m}\}$, где $\Omega = col\{\omega_j = \omega\gamma_j, j = \overline{1,m}\}$, γ_j – коэффициент распределения частот многочастотного гармонического задающего воздействия по входам системы, $0 \le \gamma_j \le 1$.

2.3. Критериальные матрицы при стохастических экзогенных воздействиях

Рассмотрим поведение системы (6) при стохастических воздействиях стационарных в широком смысле типа «белый» $w(t)$ и «окрашенный» $\xi(t)$ шумы.

Утверждение 4. *Пусть система (6) возбуждается стохастическим внешним воздействием стационарным в широком смысле типа «белый шум» $w(t)$ с матрицей интенсивности типа $Q = diagQ_{jj}, j = \overline{1,m}$, так что в (1) $g(t)$ принимает смысл $w(t)$, тогда матрица спектральной плотности $S_y(\omega)$ по выходу задается выражением*

$$S_y(\omega) = CS_x(\omega)C^T = -2CF(F^2 + \omega^2 I)^{-1} D_x C^T, \tag{19}$$

при этом матрица $D_x \overset{\Delta}{=} M[x(t) \cdot x^T(t)]$, являясь матрицей дисперсии вектора состояния, где $M[(\cdot)]$ есть оператор вычисления математического ожидания стохастической переменной $(\cdot)$, вычисляется с помощью уравнения типа Ляпунова [4]

$$FD_x + D_x F^T = -GQG^T. \tag{20}$$

□

Утверждение 5. *Пусть система (6) возбуждается стохастическим воздействием $\xi(t)$ стационарным в широком смысле типа "окрашенный шум", моделируемым выходом формирующего фильтра, возбуждаемого по входу "белым шумом" $w(t)$ с матрицей интенсивности Q, вида*

$$\dot{z}_\phi(t) = \Gamma_\phi z_\phi(t) + G_\phi \xi(t); \; \xi(t) = P_\phi z_\phi(t), \tag{21}$$

где $z_\phi \in R^l$, $\Gamma_\phi \in R^{l\times l}$, $P_\phi \in R^{m\times l}$, z_ϕ – вектор состояния модели формирующего фильтра (МФФ), Γ_ϕ, G_ϕ, P_ϕ – матрицы состояния, входа и выхода модели МФФ соответственно.

Тогда матрица спектральной плотности выхода $S_y(\omega)$ системы (6) задается выражением (19), где матрица дисперсии D_x определяется с помощью выражения

$$D_x = \widetilde{C}_x \widetilde{D}_x \widetilde{C}_x^T, \tag{22}$$

$\widetilde{C}_x = \left| I_{n\times n} \quad 0_{n\times l} \right|$, *при этом* $\widetilde{D}_x \overset{\Delta}{=} M[\widetilde{x}(t)\cdot\widetilde{x}^T(t)]$ *– матрица дисперсии составного вектора* $\widetilde{x} = \left| x^T \quad z_\phi^T \right|^T$ *– определяется в силу уравнения типа уравнения Ляпунова, записываемого в форме*

$$\widetilde{F}\widetilde{D}_x + \widetilde{D}_x\widetilde{F}^T = -\widetilde{G}Q\widetilde{G}^T, \tag{23}$$

в котором матрицы $\widetilde{F}$ *и* $\widetilde{G}$ *составной системы имеют представление*

$$\widetilde{F} = \begin{vmatrix} F & GP_\phi \\ 0 & \Gamma_\phi \end{vmatrix}, \quad \widetilde{G} = \begin{vmatrix} 0 \\ G \end{vmatrix}. \tag{24}$$

□

Таким образом, при стохастических экзогенных воздействиях критериальные матрицы представимы матрицей спектральной плотности $S_y(\omega)$.

3. Факторы, порождающие вырождение сложных динамических систем с равнотемповыми структурными компонентами. Основной результат

Построенный банк критериальных матриц N_y позволяет осуществить контроль вырождения сложной динамической системы в соответствии со следующим алгоритмом.

1. Задать векторно-матричное описание сложной динамической системы в форме (6) и зафиксировать ее параметры.
2. Задать допустимый уровень $C_{Fj}\{N_y\}$ сепаратных частотных чисел обусловленности матрицы N_y отношения вход-выход исследуемой системы и сконструировать на их основе функционалы вырождения.
3. Для случая гармонического воздействия задать реализацию вектора $\gamma = col(\gamma_j, j = \overline{1,m})$ распределения частот по входам системы; для стохастического воздействия типа «белый шум» установить значения интенсивностей Q_{jj} для каждого входа системы; для стохастического воздействия типа «окрашенный шум» задать эффективные полосы пропускания формирующих фильтров, а, следовательно, вектор распределения γ эффективных полос формирующих фильтров по входам.
4. Задать значение частоты ω, порождающей вектор $\Omega = \gamma\omega = col\{\omega_j = \gamma_j\omega,\ j = \overline{1,m}\}$ частот внешнего воздействия по входам системы.
5. По данным п.4 сформировать матрицу состояния МЗВ.
6. Решить уравнение Сильвестра вида (10), если имеет место гармоническое внешнее воздействие. Для случаев стохастического внешнего воздействия типа «белый» или «окрашенный» шумы необходимо решить уравнение типа уравнения Ляпунова в формах (20) или (23) соответственно.
7. Вычислить значения сепаратных чисел обусловленности $C_{Fj} = \{N_y\}$ и функционалов вырождения $J_{D_{Fv}}$.
8. Сравнить результат п. 7 с заданием п. 2 на предмет выполнения неравенства $C_F\{N_y\} \geq C_{Fj}\{N_y\}$ или $J_{D_F}\{N_y\} \leq J_{D_{Fv}}\{N_y\}$. В случае его выполнения – переход к п. 9, в противном случае – к п. 4.
9. Зафиксировать результаты в виде вектора γ распределения частот по входам и значения частот ω, при которых наступает вырождение данного индекса.

Специфика контроля вырождения систем MIMO-типа с равнотемповыми структурными компонентами состоит в том, что все матрицы системы (6) (G, F, C) являются диагональными с равными диагональными элементами. При всех модельных представлениях входных заявок, равномерно распределенных по входам, числа обусловленности и функционалы вырождения всех критериальных матриц оказываются единичными. Однако эту идеальную обусловленность (невырожденность) могут нарушить следующие факторы возможного вырождения систем:

1. во временной области - нарушение равнотемповости структурных компонентов;
2. в частотной области – неравномерное распределение частот по каналам;
3. при стохастическом экзогенном воздействии
   типа «белый шум» – различные компоненты матрицы интенсивности;
   типа «окрашенный шум» – различные эффективные полосы пропускания формирующих фильтров;
4. при параметрических модификациях - появление перекрестных связей.

Таким образом, можно сформулировать следующее утверждение.

Утверждение 6. *При любом методе моделирования потока входных намерений система с равнотемповыми компонентами без связей между ними и равномерном распределении гармонических и (или) стохастических воздействий не вырождается.*

□

4. Пример

Рассмотрим непрерывную систему вида

$$\dot{x}(t) = Fx(t) + Gg(t);\ x(0) = 0;\ y(t) = Cx(t),$$

где

$$F = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -216 & -72 & -12 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -216 & -72 & -12 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -216 & -72 & -12 \end{bmatrix}, \quad G = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 216 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 216 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 216 \end{bmatrix}, \quad C^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Из приведенных матриц видно, что система имеет три входа и три выхода, каждый сепаратный канал имеет третий порядок, связей между соседними каналами нет, матрицы состояния каждого из сепаратных каналов характеризуются распределением мод Баттерворда с характеристической частотой $w_{01,02,03} = 6\,\text{с}^{-1}$.

Проиллюстрируем, основываясь на предложенной технологии, поведение равнотемповой системы под влиянием факторов, которые могут нарушить ее идеальную обусловленность.

1. Внешнее ступенчатое воздействие для случаев реализации системы а) с исходными параметрами; б) при появлении перекрестных связей $k_{12,23} = 0.1$ и $k_{21,32} = -0.1$; в) при нарушении равнотемповости. Используем представление матриц состояния первого и третьего сепаратных каналов распределением мод Баттерворда с характеристическими частотами $w_{01} = 2\,\text{с}^{-1}$ и $w_{03} = 18\,\text{с}^{-1}$ соответственно. Результаты моделирования зависимостей функционалов вырождения J_{Dv} ($v = 1$) от времени t приведены на рис. 1 (а, б, в).

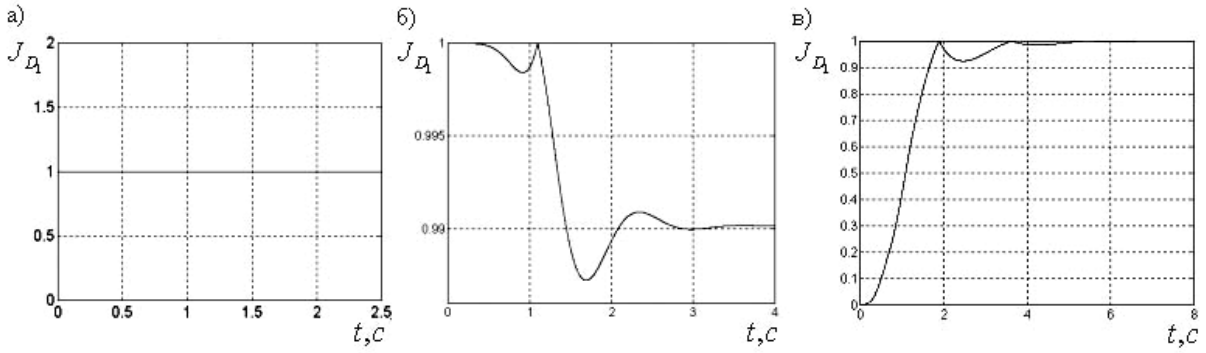


Рис. 1. Зависимости функционала вырождения от времени для непрерывной системы при внешнем ступенчатом воздействии

2. Векторное многочастотное гармоническое воздействие $g(t)$ с вектором распределения частот вида $\Omega = \gamma\omega = col\{\omega_j = \gamma_j\omega_{0j},\ j = \overline{1,3}\}$ для случаев реализации а) $\gamma^T = [1 \quad 1 \quad 1]$; б) $\gamma^T = [1/9 \quad 1 \quad 1/3]$. Результаты моделирования зависимостей функционалов вырождения J_{Dv} $(v = 1)$ от частоты ω приведены на рис. 2 (а, б).

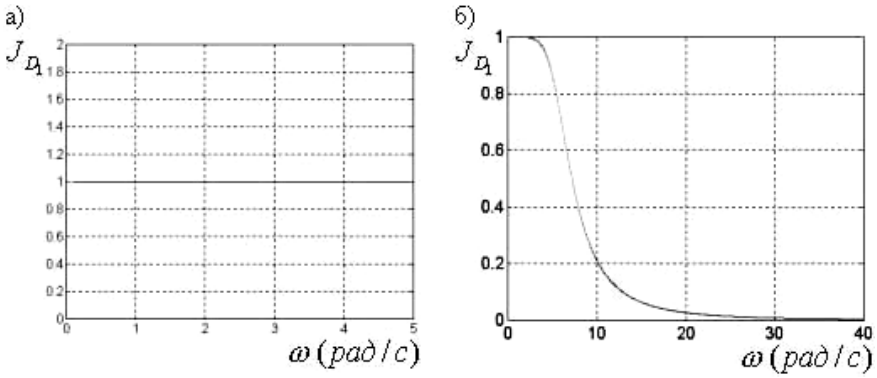


Рис. 2. Зависимости функционала вырождения от частоты для непрерывной системы при векторном многочастотном гармоническом воздействии

3. Внешнее стохастическое воздействие $g(t) = w(t)$ типа «белый шум» с канальными интенсивностями $Q_j = \gamma_j Q_0, j = \overline{1,3}$, где $Q_0 = 1$ $(*^2 \cdot c)$ для случаев реализации а) $\gamma^T = [1 \quad 1 \quad 1]$; б) $\gamma^T = [1/3 \quad 1 \quad 1/9]$. Результаты моделирования зависимостей функционалов вырождения J_{Dv} $(v = 1)$ от частоты ω приведены на рис. 3 (а, б).

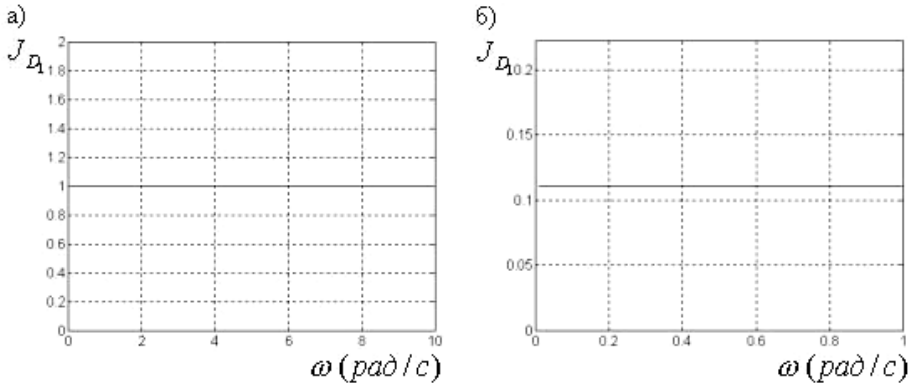


Рис. 3. Зависимости функционала вырождения от частоты для непрерывной системы при стохастическом воздействии типа «белый шум»

Заключение

Вырождение сложных систем представляет собой модельную концепцию, которая содержательно в ее предельной реализации представляет процесс деформации сложной конструкции. Предложенная технология контроля вырождения сложных динамических систем позволяет ни только оценивать исходную систему, но и выявлять возможные причины, которые могут привести к вырождению изначально идеально обусловленную систему.

Литература

1. Заде Л., Дезоер Ч. Теория линейных систем./ Пер. с англ. М.: Наука, 1970.
2. Месарович М., Такахара Я. Общая теория систем: математические основы // Под ред. С. В. Емельянова. М.: Мир. 1978.
3. Воеводин В.В., Кузнецов Ю.А. . Матрицы и вычисления. М.: Наука, 1984.
4. Голуб Дж., Ван Лоун Ч. Матричные вычисления./Пер. с англ. М.: Мир, 1999.
5. Дударенко Н.А., Ушаков А.В. Технология контроля вырождения сложных динамических систем с помощью частотных сепаратных чисел обусловленности. // Современные технологии: Сборник статей / Под ред. С.А. Козлова. СПб: СПбГУ ИТМО, 2003. 298 с.
6. Икрамов Х.Д. Численное решение матричных уравнений. / Под ред. Д.К. Фаддеева. М.: Наука. Главная редакция физико-математической литературы, 1984.
7. Dudarenko N. Degeneration control of complex dynamic systems. Preprints of 10th International Student Olympiad on Automatic Control (Baltic Olympiad), St.-Petersburg, Russia, 2004, SPb: SPSUITMO, 2004.

ФАКТОР РАЗМЕРНОСТИ ПРИ СИНТЕЗЕ ЦИФРОВОГО ДИСТАНЦИОННОГО УПРАВЛЕНИЯ С УЧЕТОМ ЗАПАЗДЫВАНИЯ В ДВОИЧНОМ КАНАЛЕ СВЯЗИ

О.С. Осипцева, А.В. Ушаков

Рассматриваются проблемы цифрового дистанционного управления с учетом фактора канальной среды. Предлагается алгоритм синтеза цифрового дистанционного управления в классе модальных управлений агрегированным дискретным объектом, характеризующийся регулятором на базе концепции агрегированного интервала дискретности «прямой канал – объект управления – обратный канал». Положение статьи иллюстрируется примером.

Введение

Принципиальным отличием цифрового дистанционного управления непрерывным объектом от такого же управления в условиях компактного размещения объекта и регулятора является фактор канальной среды. Фактор канальной среды расширяет размерность модели объекта в его дискретном представлении, причем это расширение происходит в силу четырехфазного кодового преобразования: параллельного кода, выработанного управления в последовательный, согласованный с предоставленным каналом связи; последовательного в параллельный в прямом канале связи и таких же преобразований в обратном канале связи. Размерность параллельного кода определяется используемыми АЦП и ЦАП [1]. Так, если используются восьмиразрядные АЦП и ЦАП, то размерность исходной задачи управления в его дискретном представлении с учетом фактора канальной среды увеличивается на тридцать два порядка. Если встанет дополнительная задача обеспечения помехоустойчивости передачи, то эта размерность дополнительно увеличивается на $2m$ разрядов, где m – число проверочных разрядов помехозащищенного кода. К этому надо добавить, что размещение объекта управления на контролируемом пункте и цифрового регулятора на пункте управления приводит к неизбежности построения регулятора в динамической версии, содержащей в своем составе динамическое наблюдающее устройство, размерность которого не менее размерности дискретной модели объекта управления с учетом фактора канальной среды. Таким образом, если исходный объект управления имеет второй порядок, то с учетом фактора канальной среды даже в условиях отсутствия помех проектируемая система за счет расширения размерности дискретной модели и построения полного наблюдателя получит размерность, равную шестидесяти четырем. В такой ситуации возникают определенные проблемы синтеза цифрового дистанционного управления. Первой из них является проблема отсутствия на настоящий момент «банка» модальных моделей сверхвысокой размерности. Второй проблемой является отсутствие гарантии вычислительной устойчивости всех матричных процедур с матричными компонентами высокой размерности при синтезе модального управления, опирающегося на решение уравнения Сильвестра [2, 3].

В этой связи наиболее конструктивным разрешением возникших проблем является введение в рассмотрение агрегированного интервала дискретности, равного суммарной длительности преобразования параллельного кода в последовательный, последовательного в параллельный как в прямом, так и обратном каналах связи, так что размерность агрегированного дискретного объекта увеличивается всего на два порядка.

В случае, если шумовая среда в канале связи такова, что неизбежным становится введение избыточных разрядов кода для обеспечения помехоустойчивости, то этот фактор может быть учтен путем дискретного модельного представления составной системы «канальная среда – объект управления» с интервальным значением интервала дискретности агрегированного дискретного объекта.

Предлагаемый подход ограничивает возможности достижения временных показателей процессов системы в переходном режиме. В свою очередь, в силу теоремы К.Шеннона – В. Котельникова агрегированный интервал дискретности при синтезе системы накладывает заметные ограничения на выбор характеристической частоты ω_0 параметризованной модальной модели [1, 2].

1. Синтез цифрового дистанционного управления на базе концепции агрегированного интервала дискретности дискретного модельного представления системы «канальная среда – объект управления»

В соответствии с модельной концепцией, высказанной во введении, ставится задача синтеза закона управления, обеспечивающего заданные показатели качества процессов в переходном и установившемся режимах для управляемого объекта, который в оболочке синтеза представлен агрегированной дискретной версией объекта с агрегированным интервалом дискретности. Агрегирование построено на последовательном соединении трех дискретных систем: первая система, модельная, представляет прямой канал связи; вторая представляет дискретное модельное описание исходного управляемого объекта управления; третья – обратный канал связи в его дискретном описании (рис. 1). Цифровое дистанционное управление строится как цифровое модальное динамическое управление.

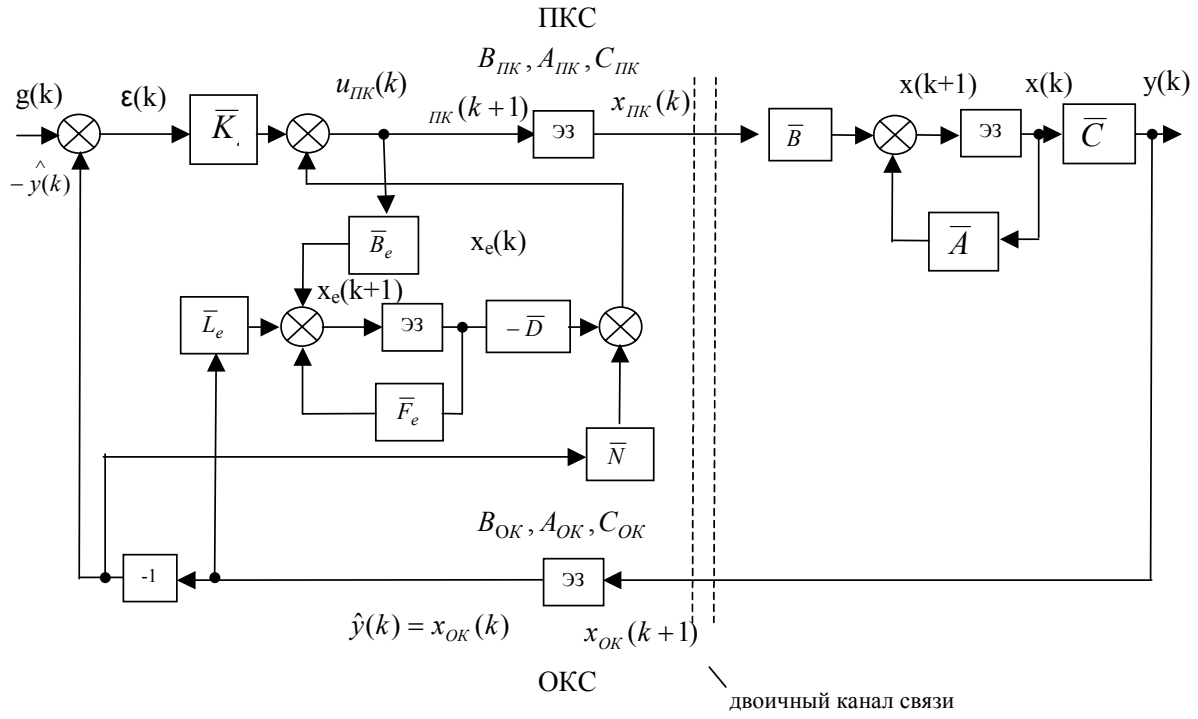


Рис. 1. Схема цифровой системы дистанционного управления с агрегированным интервалом дискретности

Ниже приводится алгоритм синтеза цифрового модального управления с агрегированным интервалом дискретности.

1. Сформировать требования к показателям качества системы цифрового дистанционного управления в переходном и установившемся режимах функционирования.

2. Априорно оценить разрядности $n_{\text{р}}$ устройств кодового преобразования и диапазон скоростей передачи предоставляемого телемеханического протокола (ТМП): $c = (\Delta t)^{-1}$

3. Сформировать агрегированный интервал дискретности $\Delta t_{\text{а}}$ для случая двоичного канала связи (ДКС) без помех в силу соотношения: $t_{\text{а}}=2(\Delta t)\, n_{\text{р}}$.

4. Сформировать векторно-матричное модельное представление (ВММП) непрерывного объекта управления (НОУ):

$$\dot{x}(t) = Ax(t) + Bu_{OY}(t);\quad x(0);\quad y(t) = Cx(t), \tag{1}$$

где $x, u_{\text{ОУ}}, y$ – соответственно векторы состояния, управления и выхода объектов; $x \in R^n, u_{OY} \in R^r, y \in R^m$; A, B, C – соответственно матрицы состояния, управления и выхода, согласованные по размерности с векторами x, u, y.

5. Сформировать векторно-матричное описание дискретного представления объекта

$$x(k+1) = \overline{A}x(k) + \overline{B}u_{OY}(k);\ x(0);\ \ y(k) = \overline{C}x(k) \tag{2}$$

где k – дискретное время, выраженное в числе интервалов дискретности, длительностью Δt так, что непрерывное время t и дискретное k связаны отношением $t = (\Delta t_a) \cdot k$; матрицы $\overline{A}, \overline{B}, \overline{C}$ вычисляются в силу соотношений:

$$\overline{A} = \exp(A \cdot \Delta t_a);\ \overline{B} = A^{-1} \cdot (\overline{A} - I) \cdot B;\ \overline{C} = C \tag{3}$$

6. Построить модельное представление прямого и обратного каналов связи, осуществляющих задержку дискретного сигнала на один такт длительностью Δt_a в форме

$$x_{ПК}(k+1) = A_{ПК}x_{ПК}(k) + B_{ПК}u_{ПК}(k);\ x(0);\ \ y_{ПК}(k) = C_{ПК}x_{ПК}(k) \tag{4}$$

$$u_{OY}(k) = y_{ПК}(k) \tag{5}$$

$$x_{OK}(k+1) = A_{OK}x_{OK}(k) + B_{OK}u_{OK}(k);\ \ x(0);\ \ y_{OK}(k) = C_{OK}x_{OK}(k) \tag{6}$$

где $x_{ПК}, u_{ПК}, y_{ПК}, x_{OK}, u_{OK}, y_{OK}$ – соответственно векторы состояния, управления и выхода в прямом и обратном каналах единичной размерности; $x_{ПК} \in G^n, u_{ПК} \in G^r, y_{ПК} \in G^m$, $x_{OK} \in G^n, u_{OK} \in G^r, y_{OK} \in G^m$, $A_{ПК} = A_{OK} = 0$; $B_{ПК} = B_{OK} = [1]$; $C_{ПК} = C_{OK} = [1]$.

7. Сформировать агрегированный дискретный объект управления (АДОУ), составленный из последовательного соединения прямого канала связи, дискретного объекта управления (ДОУ) и обратного канала связи с вектором состояния $x_a = \left[x_{ПК}{}^T; x_{ДОУ}{}^T; x_{OK}{}^T\right]^T$ размерности $n_{\text{а}}=n+2$, вектором регулированного выхода y, вектором измеряемого выхода $y_{\text{и}}$, представляющим собой выход ОКС, и матрицами ($A_{\text{а}}$, $B_{\text{а}}$, $C_{\text{а}}$, $C_{\text{аи}}$).

$$x_A(k+1) = \left[x_{ПК}(k+1); x(k+1); x_{OK}(k+1)\right]^{-1} \tag{7}$$

$$x_A(k+1) = A_A x_A(k) + B_A u_{ПК}(k);\ \ x(0);\ \ y(k) = C_A x_A(k);\ \hat{y}(k) = \hat{C}_A\, x(k), \tag{8}$$

где

$$A_A = \left.\begin{bmatrix} A_{ПК} & 0 & 0 \\ C_{ПК} & \overline{A} & 0 \\ 0 & \overline{C} & A_{OK} \end{bmatrix}\right|_{A_{ПК}=0; A_{OK}=0; C_{ПК}=I} = \begin{bmatrix} 0 & 0 & 0 \\ \overline{B} & \overline{A} & 0 \\ 0 & \overline{C} & 0 \end{bmatrix}; \tag{9}$$

$$B_A = \left[\begin{matrix} B_{ПК} \\ 0 \\ 0 \end{matrix}\right]\Bigg\|_{B_{ПК}=I} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix};\ C_A = [0 \quad C \quad 0];\ \hat{C}_A = [0 \quad 0 \quad 1]; \tag{10}$$

$$A_{ПК} = A_{ОК} = 0; C_{ПК} = B_{ПК} = I. \tag{11}$$

8. Сформировать априорную оценку $t_{ап}$ длительности переходного процесса $t_п$ для случая системы дистанционного цифрового управления с регулятором без наблюдателя в форме $t_{ап}=(\Delta t_а)\ n_а$ и для случая регулятора с наблюдателем в форме $t_{ап}=2(\Delta t_а)\ n_а$.

9. Проверить выполнение условия $t_{пт} \le t_{ап}$, где $t_{пт}$ – требуемая по техническому заданию длительность переходного процесса, при этом в случае выполнения неравенства – переход к п.10 алгоритма, в случае невыполнения – осуществление действий:

9.1. переход к п.1 с целью согласования технического задания на предмет снижения требований к величине $t_{пт}$ с последующим переходом к п. 9;

9.2. переход к п. 2 с целью смены ТМ-протокола на ТМП с большей скоростью передачи, с последующим переходом к п.10;

9.3. если условие $t_{пт} \le t_{ап}$ выполняется для случая системы дистанционного цифрового управления с регулятором без наблюдателя, то совершить переход к реализации дискретного наблюдателя с интервалом дискретности $\Delta t_н$, таким, чтобы процесс наблюдения совершался бы за один такт «канального времени».

10. Сформировать закон управления в форме комбинации обратной связи по состоянию x_A с матрицей $\overline{K}$ и прямой связи по задающему воздействию $g(k)$ матрицей $\overline{K}_g$

$$u_{ПК}(k) = \overline{K}_g g(k) - \overline{K}x(k) \tag{12}$$

с использованием метода модального управления [5].

11. Выбрать непрерывную динамическую модальную модель (ММ) в форме пары матриц (Γ_A, H_A) желаемого поведения «вход-выход» проектируемой системы. Γ_A – матрица состояния ММ – является носителем желаемой структуры собственных значений размерностью $n_a \times n_a$ $\dim \mathrm{H}_A = \dim B_A^{\ T}$; (Γ_A, H_A) – наблюдаемая пара матриц.

12. Сконструировать дискретную версию модальной модели с парой матриц $(\overline{\Gamma}_A, \overline{\mathrm{H}}_A)$, где матрица $\overline{\Gamma}_A$ вычисляется с помощью соотношения $\overline{\Gamma}_A = \exp(\Gamma_A \cdot \Delta t_a)$, а матрица $\overline{\mathrm{H}}_A$ вычисляется на основе пары матричных уравнений Сильвестра для непрерывного и дискретного случаев:

$$\mathrm{M}_A\Gamma_A - \mathrm{A}_A\mathrm{M}_A = -\mathrm{B}_A\mathrm{H}_A \tag{13}$$

относительно матрицы M_A.

$$\overline{\mathrm{M}}_A\overline{\Gamma}_A - \overline{\mathrm{A}}_A\overline{\mathrm{M}}_A = -\overline{\mathrm{B}}_A\overline{\mathrm{H}}_A \tag{14}$$

при известной матрице $\overline{\mathrm{H}}_A$ относительно матрицы $\overline{\mathrm{M}}_A$.

13. Сформировать матрицу $\overline{K}_g$ с помощью соотношений прямых связей по задающему воздействию и из условия равенства регулируемого выхода и задающего воздействия в неподвижном состоянии

$$\overline{K}_g = \arg\{\overline{C}_A(zI - \overline{F}_A)^{-1} \cdot \overline{B}_A\overline{K}_g\big|_{z=1} = I\} = [\overline{C}_A(I - \overline{F}_A)^{-1} \cdot \overline{B}_A]^{-1} \tag{15}$$

$$\overline{M}_A\overline{\Gamma}_A = \overline{F}_A\overline{M}_A \tag{16}$$

$$\overline{K}_g = [\overline{C}_A(I - \overline{M}_A\overline{\Gamma}_A\overline{M}_A^{-1})^{-1} \cdot \overline{B}_A]^{-1} \tag{17}$$

14. Построить цифровой закон управления, использующий сигнал ошибки $\varepsilon(k) = g(k) - y(k)$ по выходу:

$$u_{ПК}(k)=\overline{K}_g g(k)-\overline{K}x(k)=\overline{K}_g g(k)-\overline{K}_y y(k)-\overline{K}_x x(k)\Big|_{\overline{K}_g=\overline{K}_y=\overline{K}_\varepsilon}=\overline{K}_\varepsilon \varepsilon(k)-\overline{K}_x x(k). \quad (18)$$

15. Сформировать динамическое наблюдающее устройство вектора состояния $x_A(k)$ объекта (2) в форме

$$x_e(k+1)=\overline{F}_e x_e(k)+\overline{L}_e \hat{y}(k)+\overline{B}_e u(k), \quad (19)$$

где матрицы динамического наблюдающего устройства выбираются из условия

$$\overline{F}_e=\arg\{\sigma\{\overline{F}_e\}\prec\sigma\{\overline{F}_A\}\ \&\ \sigma\{\overline{A}_A\}\bigcap\sigma\{\overline{F}_e\}=0\}, \quad (20)$$

$$\overline{L}_e=\arg\{contr(\overline{F}_e,\overline{L}_e)\}, \quad (21)$$

$$\overline{B}_e=\overline{T}_e\overline{B}_A \quad (22)$$

Здесь $\prec$ – знак мажоризации, означающий в данной задаче, что моды матрицы состояния наблюдателя локализованы на комплексной плоскости левее вдоль вещественной оси мод матрицы состояния системы.

16. Вычислить матрицу $\overline{T}_e$ подобия вектора наблюдения $x_e(k)$ вектору состояния $x_A(k)$, задаваемого в форме

$$x_e(k)=\overline{T}_e x_A(k)-\overline{\Theta}_e(k), \quad (23)$$

в силу решения матричного уравнения Сильвестра

$$\overline{T}_e\overline{A}_A-\overline{F}_e\overline{T}_e=\overline{L}_e\overline{C}_{e,} \quad (24)$$

которое совместно с (22) обеспечивает асимптотическую сходимость к нулю вектора невязки наблюдения $\overline{\Theta}_e(k)$,

$$\overline{\Theta}_e(k+1)=\overline{F}_e\overline{\Theta}_e(k);\ \overline{\Theta}_e(0)=\overline{T}_e x(0)-x_e(0), \quad (25)$$

$$\overline{\Theta}_e(k)=(\overline{F}_e)^k\overline{\Theta}_e(0) \quad (26)$$

17. Сформировать динамическую версию закона управления (18)

$$u_{ПК}(k)=\overline{K}_\varepsilon\varepsilon(k)-\overline{N}\hat{y}(k)-\overline{D}x_e(k). \quad (27)$$

18. Проверить работоспособность просинтезированного цифрового дистанционного устройства управления и оценить его динамические свойства в модельной среде MatLab [6,7].

2. Пример алгоритм синтеза цифрового модального управления с агрегированным интервалом дискретности

В соответствии с алгоритмом синтеза, приведенным в пункте 1, осуществим следующие действия.

1. Формирование требования к системе: дана система дистанционного цифрового управления электроприводом второго порядка со показателями качества – перерегулирование системы $\sigma\leq 11\%$, время переходного процесса $t_{пп}=0.7$с.

2. Выбор канальной среды: принята среда, формируемая средствами системы телемеханики, в которой организована передача данных со скоростью 400 бит/с, длительность одного бита (элементарного сигнала) кода равна $\Delta t=2.5$ мс, $n_р=8$. Используются восьмиразрядные АЦП и ЦАП, $n_р=8$.

3. Формирование агрегированного интервала дискретности $\Delta t_а$ для модальной модели $\Delta t_а=0{,}04$с.

4. Формирование векторно-матричного модельного представления (ВММП) непрерывного объекта управления (НОУ):

$$\dot{x}(t)=Ax(t)+Bu_{ОУ}(t);\quad x(0);\quad y(t)=Cx(t),$$

где $A=\begin{bmatrix}0&1\\0&0\end{bmatrix};B=\begin{bmatrix}0\\1\end{bmatrix};C=\begin{bmatrix}1&0\end{bmatrix}.$

5. Формирование векторно-матричного описания дискретного представления объекта

$$x(k+1)=\overline{A}x(k)+\overline{B}u_{OY}(k);\ x(0);\ \ y(k)=\overline{C}x(k),$$

где $\overline{A}=\begin{bmatrix}1&0{,}04\\0&1\end{bmatrix};\overline{B}=\begin{bmatrix}0\\0{,}04\end{bmatrix};\overline{C}=\begin{bmatrix}1&0\end{bmatrix}.$

6. Выбор матриц модельного представления прямого и обратного каналов связи, осуществляющих задержку дискретного сигнала на один такт длительностью Δt_a:

$$A_{ПК}=A_{OK}=0;\ B_{ПК}=B_{OK}=[1];\ C_{ПК}=C_{OK}=[1].$$

7. Формирование агрегированного дискретного объекта управления

$$x_A(k+1)=A_Ax_A(k)+B_Au_{ПК}(k);\ \ x(0);\ \ y(k)=C_Ax_A(k);\ \hat{y}(k)=\hat{C}_A\,x(k),$$

где

$$A_A=\begin{bmatrix}0&0&0\\\overline{B}&\overline{A}&0\\0&\overline{C}&0\end{bmatrix}=\begin{bmatrix}0&0&0&0\\0&1&0{,}04&0\\0{,}04&0&1&0\\0&1&0&0\end{bmatrix};\ B_A=\begin{bmatrix}1\\0\\0\\0\end{bmatrix};C_A=\begin{bmatrix}0&1&0&0\end{bmatrix};\hat{C}_A=\begin{bmatrix}0&0&0&1\end{bmatrix}.$$

8. Формирование априорной оценки $t_{ап}$: для случая системы дистанционного цифрового управления с регулятором без наблюдателя $t_{ап}$= 0,16; для случая регулятора с наблюдателем $t_{ап}$= 0,32.

9. Проверка выполнения условия $t_{пт}\le t_{ап}$: $t_{пт}$= 0,7 с.

10. Формирование закона управления в форме комбинации обратной связи по состоянию x_A с матрицей $\overline{K}$ и прямой связи по задающему воздействию $g(k)$ матрицей $\overline{K}_g$:

$$u_{ПК}(k)=\overline{K}_g\,g(k)-\overline{K}x(k).$$

11. Выбор непрерывной динамической модальной модели (ММ) в форме пары матриц (Γ_A,H_A) желаемого поведения «вход-выход» проектируемой системы: динамическая модальная модель с распределением мод Баттерворта четвертого порядка $a(s)=s^4+2.6\omega_0s^3+3.4\omega_0^2s^2+2.6\omega_0^3s+\omega_0^4$, доставляющей системе перерегулирование σ=11% и время переходного процесса $t_{пп}$=0.7с.

Характеристическая частота модальной модели в силу теоремы К. Шеннона – В. Котельникова не должна превышать данного значения $\omega_0=\omega_н=\pi/\Delta t_a\approx 78.54$.

$$\Gamma=\begin{bmatrix}-72{,}56&30{,}056&0&0\\-30{,}056&-72{,}56&0&0\\0&0&-55{,}54&55{,}54\\0&0&-55{,}54&-55{,}54\end{bmatrix};\ \ \mathrm{H}=[1111].$$

12 Конструирование дискретной версии модальной модели с парой матриц $(\overline{\Gamma}_A,\overline{\mathrm{H}}_A)$:

$$\overline{\Gamma}=\begin{bmatrix}0{,}0198&0{,}0512&0&0\\-0{,}0512&0{,}0198&0&0\\0&0&-0{,}0656&0{,}086\\0&0&-0{,}086&-0{,}0656\end{bmatrix};\ \ \overline{\mathrm{H}}=[1111].$$

13. Формирование матрицы $\overline{K}_g$:

$\overline{K}_g = 688{,}36$.

14. Построение цифрового закона управления:

$$u_{ПК}(k) = \overline{K}_g g(k) - \overline{K}x(k) = \overline{K}_g g(k) - \overline{K}_y y(k) - \overline{K}_x x(k)\Big|_{\overline{K}_g = \overline{K}_y = \overline{K}_\varepsilon} = \overline{K}_\varepsilon \varepsilon(k) - \overline{K}_x x(k).$$

15. Формирование динамического наблюдающего устройства вектора состояния $x_A(k)$ объекта (2) в форме $x_e(k+1) = \overline{F}_e x_e(k) + \overline{L}_e \hat{y}(k) + \overline{B}_e u(k)$.

16. Вычисление матрицы $\overline{T}_e$ подобия вектора наблюдения $x_e(k)$ вектору состояния $x_A(k)$:

$$\overline{T}_e = \begin{bmatrix} 10{,}431 & -9{,}3608 & 0{,}3288 & 0{,}7783 \\ -23{,}559 & 24{,}523 & -1{,}018 & -0{,}044 \\ 12{,}921 & -11{,}914 & 0{,}4364 & 0{,}0584 \\ -1{,}751 & 2{,}6088 & -0{,}1333 & 0{,}1578 \end{bmatrix}.$$

17. Формирование динамической версии закона управления

$$u_{ПК}(k) = \overline{K}_\varepsilon \varepsilon(k) - \overline{N}\,\hat{y}(k) - \overline{D}x_e(k).$$

18. Проверка работоспособности просинтезированного цифрового дистанционного устройства управления и оценка его динамических свойств в модельной среде MatLab:

Для иллюстрации процессов системы с цифровым динамическим регулятором проведено моделирование переходных процессов для значений характеристических частот модальной модели системы и наблюдателя. При помощи пакета MatLab было проведено моделирование данной системы, результат которого представлен на рис. 2.

Переходный процесс при ступенчатом воздействии длится четыре такта, так как динамика наблюдателя не проявляется, наблюдатель находится в нулевом состоянии. При ненулевом начальном условии состояние объекта длительности процессов составляет восемь тактов, что вызвано несогласованностью состояний объекта и наблюдающего устройства.

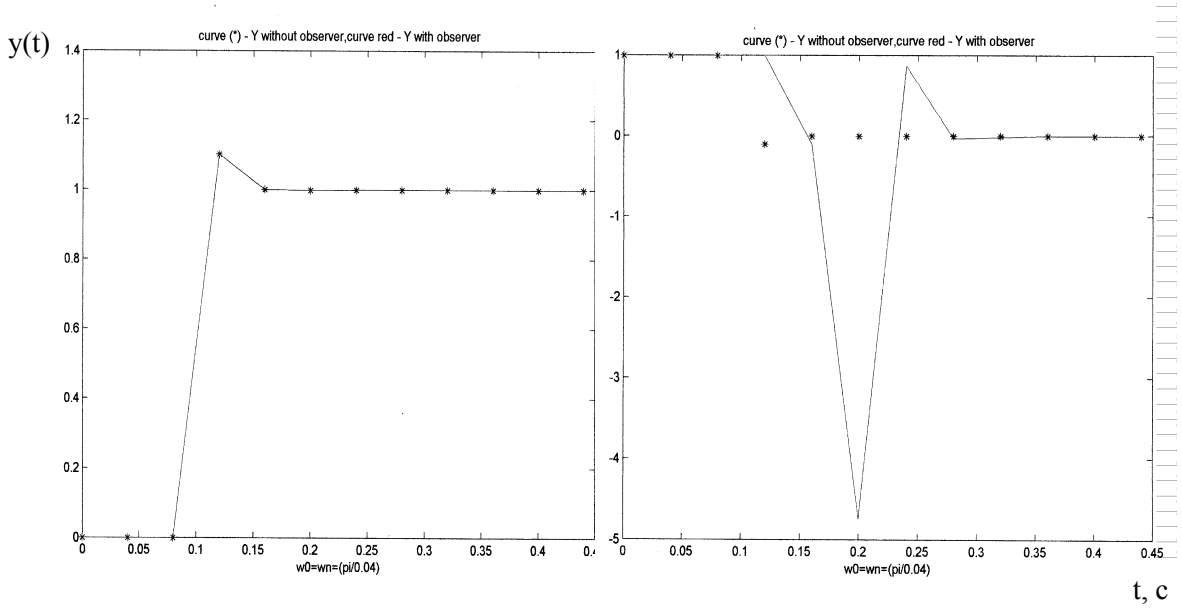


Рис. 2. Процессы в цифровой системе дистанционного управления с агрегированным интервалом дискретности с наблюдателем при единичном скачке и единичном начальном значении выхода

Заключение

Использованный авторами прием синтеза цифрового дистанционного управления, основанный на введении агрегированного интервала дискретности, позволил удовлетворить техническим требованиям и заметно сократить аппаратное пространство при конструировании технической среды наблюдения.

Заметим, что если бы задача решалась «в лоб», то ее полная размерность при данных параметрах ЦАП и АЦП с учетом размерности ее среды наблюдения составила бы 66. При интервале дискретности, обеспеченном выбранным телемеханическим протоколом $\Delta t = 2{,}5 \cdot 10^{-3}$с, достижимое время переходного процесса составило бы величину 0,165 с. Таким образом, констатируется классический кибернетический результат, состоящий в возможности обмена времени на объект аппаратного пространства схемотехнической реализации.

Дополнительно положительным свойством модели представления с использованием агрегированного интервала дискретности является низкая размерность модальной модели, которое гарантирует в общем случае небольшое перерегулирование, в отличие от ситуации полной размерности, где перерегулирование при таких размерностях в несколько десятков может оказаться довольно высоким.

Литература

1. Ту Ю., Современная теория управления / Пер. с англ. М.: Машиностроение,1971.
2. Синтез дискретных регуляторов при помощи ЭВМ / В.В. Григорьев, В.Н. Дроздов, В.В. Лаврентьев, А.В. Ушаков. Л.: Машиностроение, Ленингр. отд-ние,1983.
3. Изерман Р. Цифровые системы управления / Пер. с англ. М.: Мир, 1984.
4. Никифоров В.О., Ушаков А.В. Управление ив условиях неопределенности: чувствительность, адаптация, робастность. СПб: СПбГИТМО (ТУ), 2002. 232 с., ил.29.
5. Ушаков А.В. Обобщенное модальное управление // Изв. вузов. Приборостроение. 2000. №3. С.8–16.
6. Olga.S. Osiptseva, The Influence of delay in binary channel on the quality of remote digital control in PP-protocol / Preprints of 10th International Student Olympiad on Automatic Control (Baltic Olympiad), St.-Petersburg,Russia, 2004, SPb:SPSUITMO, 2004.
7. Осипцева О.С., Ушаков А.В. Влияние запаздывания в двоичном канале без помех на качество дистанционного цифрового управления. Научно-технический вестник СПбГУ ИТМО. Выпуск 14. Информационные технологии, вычислительные и управляющие системы / Главный редактор д.т.н., проф. В.Н. Васильев СПб: СПбГУ ИТМО, 2004.

ОБЕСПЕЧЕНИЕ СТАБИЛЬНОСТИ ПОКАЗАТЕЛЕЙ КАЧЕСТВА В ЗАДАЧЕ УПРАВЛЕНИЯ ДИНАМИЧЕСКИМ ОБЪЕКТОМ С ИНТЕРВАЛЬНЫМИ ПАРАМЕТРАМИ ПРИ КОНЕЧНОМЕРНОМ ЗАДАЮЩЕМ ВОЗДЕЙСТВИИ

Т.А Акунов, С.А. Сударчиков, А.В. Ушаков

Решается задача обеспечения стабильности показателей качества при управлении многомерным непрерывным динамическим объектом с интервальными параметрами при конечномерном задающем входном воздействии.

Введение. Постановка задачи

Рассматривается многомерный непрерывный динамический объект управления, матричные компоненты которого характеризуются параметрической неопределенностью, задаваемой в интервальной форме. Предполагается, что модельная параметрическая неопределенность, в силу выбора базиса представления объекта или включения на входе некоторой буферной системы минимальной размерности, может быть только в матрице состояния исходного объекта управления. Введение буферной системы является конструктивным решением задачи достижения требуемой интервальности матричных компонентов проектируемой системы в случае, если в исходном объекте управления интервальными оказались матрицы $[A]$ состояния и $[B]$ управления [1]. На основе анализа конечномерного входного задающего воздействия в переходном и установившемся режимах [2] ставится задача синтеза обобщенного изодромного управления, обеспечивающего системе стабильные показатели качества.

Интервальное модельное представление исходного объекта управления

Объект управления с интервальной матрицей состояния и интервальной матрицей управления задается векторно-матричной моделью

$$\dot{x}(t) = [A]x(t) + [B]u(t); x(0); y(t) = Cx(t), \qquad (1)$$

где $x \in R^n, u \in R^r, y \in R^m$ – соответственно векторы состояния, управления и выхода ОУ; $[A], [B], C$ – интервальная матрица состояния, интервальная матрица управления и матрица выхода, согласованные по размерности с переменными модели (1). Интервальная матрица состояния $[A]$ представляется в виде аддитивной композиции медианной и интервальной составляющих

$$[A] = A_0 + [\Delta A] = A_0 + \left[\underline{\Delta A}, \overline{\Delta A}\right], \qquad (2)$$

где

$$A_0 = row\left\{col\left(\underline{A}_{0ij}; i = \overline{1,n}\right); j = \overline{1,n}\right\},$$

$$\underline{\Delta A} = row\left\{col\left(\underline{\Delta A}_{ij}; i = \overline{1,n}\right); j = \overline{1,n}\right\};$$

$$\overline{\Delta A} = row\left\{col\left(\overline{\Delta A}_{ij}; i = \overline{1,n}\right); j = \overline{1,n}\right\}$$

$$A_{0ij} = 0.5(\underline{A_{ij}} + \overline{A}_{ij}); \underline{\Delta A_{ij}} = \underline{A}_{ij} - A_{0ij}; \overline{\Delta A_{ij}} = \overline{A}_{ij} - A_{0ij}.$$

A_0 и $[\Delta A]$ – соответственно медианная и интервальная составляющая интервальной матрицы $[A]$.

Дополним исходный ОУ (1) буферной системой

$$\dot{x}_B(t) = A_B x_B(t) + B_B u_B(t); x(0); y_B(t) = C_B x_B(t), \qquad (3)$$

где x_B – ℓ –мерный вектор состояния буферной системы (БС), u_B – r -мерный вектор входа, y_B – r -мерный вектор выхода, A_B – $(n_B \times n_B)$ – матрица состояния БС, B_B – $(n_B \times r)$– матрица входа, C_B – $(r \times \ell)$ – матрица выхода.

Агрегирование ОУ (1) и БС (2) осуществляется путем наложения условия

$$u(t) = y_B(t). \tag{4}$$

Введем в рассмотрение вектор состояния размерности $\tilde{n} = n + n_B$, $\tilde{x} = col\{x, x_B\}$ агрегированного объекта управления. Тогда векторно-матричное описание агрегированного объекта примет вид

$$\dot{\tilde{x}}(t) = [\tilde{A}]\tilde{x}(t) + \tilde{B}\tilde{u}(t), \tag{5}$$

$$x(t) = \tilde{C}_x\tilde{x}(t);\ y_B = \tilde{C}_B\tilde{x}(t);\ y(t) = \tilde{C}_y\tilde{x}(t), \tag{6}$$

где в силу (1) и (2) с учетом (3) матричные компоненты принимают вид

$$[\tilde{A}] = \begin{bmatrix} [A] & [B]C_B \\ 0 & A_B \end{bmatrix};\quad \tilde{B} = \begin{bmatrix} 0 \\ B_B \end{bmatrix}. \tag{7}$$

$$\tilde{C}_x = \begin{bmatrix} I_{n\times n} & 0_{n_B\times n_B} \end{bmatrix},\quad \tilde{C}_B = \begin{bmatrix} 0_{n\times n} & I_{n_B\times n_B} \end{bmatrix},\quad \tilde{C}_y = \begin{bmatrix} C & 0_{n_B\times n_B} \end{bmatrix}. \tag{8}$$

Нетрудно видеть из (7), что условие интервальности сохранилось только в матрице состояния, для которой можно записать

$$[\tilde{A}] = \tilde{A}_0 + [\Delta\tilde{A}], \tag{9}$$

где $\tilde{A}_0 = \begin{bmatrix} A_0 & B_0C_B \\ 0 & A_B \end{bmatrix}$, $\Delta\tilde{A} = \begin{bmatrix} \Delta A & [\Delta B]C_B \\ 0 & 0 \end{bmatrix}$.

Основной результат

Основной результат может быть представлен в виде утверждения.

Утверждение 1. Пусть медианная версия агрегированного объекта (5), (6)

$$\dot{\tilde{x}}(t) = \tilde{A}_0\tilde{x}(t) + \tilde{B}_0\tilde{u}(t),\ y(t) = \tilde{C}_y\tilde{x}(t) \tag{10}$$

должна воспроизводить внешнее задающее конечномерное воздействие $g(t)$ с нулевой установившейся ошибкой

$$\varepsilon(t) = g(t) - y(t);\quad \lim_{t\to\infty}\varepsilon(t) = 0. \tag{11}$$

Пусть модель конечномерного воздействия представима автономной системой минимальной размерности ℓ в форме

$$\dot{z}(t) = \Gamma z(t);\ z(0);\ g(t) = Pz(t), \tag{12}$$

где $z \in R^{\ell}$; $g \in R^m$; $\Gamma \in R^{\ell\times\ell}$; $P \in R^{m\times\ell}$. Пусть также при построении агрегированного ОУ (5) выполнены матричные условия

$$A_B = \Gamma;\quad C_B = P \tag{13}$$

так, что матрица $\tilde{A}_0$ принимает вид

$$\tilde{A}_0 = \begin{bmatrix} A_0 & B_0P \\ 0 & \Gamma \end{bmatrix}, \tag{14}$$

тогда поставленная задача получает решение с помощью управления

$$\tilde{u}(t) = \tilde{K}\tilde{\varsigma}(t), \tag{15}$$

где вектор ошибки слежения по состоянию задается в форме

$$\tilde{\varsigma}(t) = \tilde{T}z(t) - \tilde{x}(t), \tag{16}$$

если матрица подобия удовлетворяет матричным соотношениям

$$\tilde{T}\tilde{\Gamma} - \tilde{A}_0\tilde{T} = 0 \quad \tilde{P} - \tilde{C}\tilde{T} = 0. \qquad \square \tag{17}$$

Доказательство. Для доказательства положения утверждения строится модель ошибки слежения по состоянию. Для чего продифференцируем соотношение (16) по времени, в результате чего получим

$$\dot{\tilde{\varsigma}}(t) = \tilde{T}\dot{z}(t) - \dot{\tilde{x}}(t). \tag{18}$$

Если в (18) подставить (10) и (12), учесть (16), то для модели ошибки становится справедливым следующее представление

$$\dot{\tilde{\varsigma}}(t) = \tilde{A}_0\tilde{\varsigma}(t) + \left(\tilde{T}\Gamma - \tilde{A}_0\tilde{T}\right)z(t) - \tilde{B}\tilde{u}(t); \; \tilde{\varsigma}(0) = \tilde{T}z(0) - \tilde{x}(0). \tag{19}$$

Учтем в (18) первое из матричных соотношений (17) и подставим в него (15), тогда получим окончательную модель для ошибки слежения по вектору состояния

$$\dot{\tilde{\varsigma}}(t) = \tilde{F}_0\tilde{\varsigma}(t); \quad \tilde{\varsigma}(0), \tag{20}$$

где $\tilde{F}_0 = \tilde{A}_0 - \tilde{B}\tilde{K}$.

Явное решение для системы (20) имеет вид

$$\tilde{\varsigma}(t) = \exp\left(\tilde{F}_0 t\right)\tilde{\varsigma}(0). \tag{21}$$

Это решение обладает свойством

$$\lim_{t\to\infty} \tilde{\varsigma}(t) = 0, \tag{22}$$

если матрица состояния $\tilde{F}_0$ гурвицева, причем требуемый темп сходимости в (22) определяется структурой собственных значений (мод) матрицы $\tilde{F}_0$, тем самым задача обеспечения нулевой ошибки слежения за конечномерным задающим воздействием сводится к задаче модального управления при выполнении соотношения (17).

Теперь покажем, что выполнение условия (22) с одновременным учетом второго матричного соотношения (17) гарантирует выполнение условия (11).

Для ошибки слежения по выходу за конечномерным задающим воздействием в силу (12) и (17) можно записать

$$\varepsilon(t) = \tilde{C}\tilde{\varsigma}(t) - \left(\tilde{P} - \tilde{C}\tilde{T}\right)z(t). \tag{23}$$

Подстановка в (22) второго соотношения (16) дает

$$\varepsilon(t) = \tilde{C}\tilde{\varsigma}(t); \quad \lim_{t\to\infty}\varepsilon(t) = \tilde{C}\lim_{t\to\infty}\tilde{\varsigma}(t) = 0. \tag{24}$$

Ключевым моментом в получении результатов (21), (22) и (24) является нетривиальное решение (17). Условием этого решения является пересечение алгебраических спектров собственных значений матриц Γ и $\tilde{A}_0$ [3, 4]. Это условие в данном случае выполняется, так как матрица $\tilde{A}$ имеет треугольный вид (14), для которого можно записать

$$\sigma\left\{\tilde{A}_0\right\} = \sigma\{A_0\} \cup \sigma\{\Gamma\}. \qquad \blacksquare \tag{25}$$

Фактор интервальности матрицы состояния агрегированного ОУ (14) учтем с помощью следующего утверждения.

Утверждение 2.

Закон управления (15) не меняет оценки абсолютной интервальности матрицы состояния, так что выполняется равенство

$$\Delta_I\tilde{F} = \Delta_I\tilde{A}, \tag{26}$$

но при этом изменяет значение оценки $\delta_I \widetilde{F}$ относительной интервальности интервальной матрицы $\left[\widetilde{F}\right]$ состояния системы

$$\delta_I \widetilde{F} \overset{\Delta}{=} \frac{\left\|[\Delta \widetilde{F}]\right\|}{\left\|\widetilde{F}_0\right\|}.$$

Доказательство утверждения использует соотношения (26), позволяющие записать

$$\delta_I F \overset{\Delta}{=} \frac{\left\|[\Delta \widetilde{F}]\right\|}{\left\|\widetilde{F}_0\right\|} = \frac{\left\|[\Delta \widetilde{A}]\right\|}{\left\|\widetilde{A}_0 - \widetilde{B}\widetilde{K}\right\|}. \qquad \blacksquare \ (27)$$

Положения утверждений 1 и 2 позволяют сформировать алгоритм синтеза обобщенного изодромного управления методами модального управления.

Поставленную задачу синтеза обобщенного изодромного управления в форме (15) будем решать в два этапа. На первом этапе синтезируется матрица $\widetilde{K}$ в предположении непосредственной измеримости вектора ошибки $\widetilde{\zeta}(t)$ с одновременным контролем достижимого значения оценки матрицы состояния системы. На втором этапе синтезируется устройство, которое формирует его асимптотическую оценку. В реализации такого подхода алгоритм принимает вид, представленный ниже.

1. Составить $([A], [B], C)$ представления исходного ОУ (1).
2. На основе анализа входного задающего воздействия построить его конечномерную модель с матрицами (Γ, P) (12).
3. Сформировать агрегированный объект управления (1) и БС (3), матричные компоненты которой совпадают с матричными компонентами конечномерного входного воздействия.
4. Сформировать требования к динамическим свойствам системы в переходном и установившемся режимах, задав их в форме желаемой структуры мод и условия обеспечения нулевой установившейся ошибки слежения, а также в виде требований к значению оценки $\delta_{IR} \widetilde{F}$ относительной интервальности матрицы состояния агрегированной системы.
5. На основе сформированной структуры мод сконструировать модальную модель в виде наблюдаемой пары матриц $\left(\widetilde{\Lambda}, \widetilde{\mathrm{H}}\right)$ с нормой, удовлетворяющей требованиям к значению интервальности

$$\widetilde{\Lambda} = \arg\left\{ \widetilde{\Lambda} = diag\left\{\widetilde{\lambda}_i \in \sigma\left\{\widetilde{F}\right\}\right\} \& \ \delta_I \widetilde{F} = \frac{\left\|\left[\Delta \widetilde{\mathrm{A}}\right]\right\|}{\left\|\widetilde{M}\widetilde{\Lambda}\widetilde{M}^{-1}\right\|} \le \delta_{IR} \widetilde{F} \right\}.$$

6. Решить уравнения Сильвестра [5, 6] $\widetilde{M}\widetilde{\Lambda} - \widetilde{A}_0 \widetilde{M} = -\widetilde{B}\widetilde{\mathrm{H}}$ относительно матрицы подобия $\widetilde{M}$ для медианной версии агрегированного ОУ.
7. Сконструировать матрицу $\widetilde{M}\widetilde{\Lambda}\widetilde{M}^{-1}$, вычислить ее норму $\left\|\widetilde{M}\widetilde{\Lambda}\widetilde{M}^{-1}\right\|$ и осуществить проверку выполнения требования к интервальности интервальной матрицы состояния проектируемой системы, в случае его невыполнения осуществить возвращение к п.5, в противном случае – к п.8.
8. Вычислить матрицу обратной связи $\widetilde{K}$ в форме $\widetilde{K} = \widetilde{H}\widetilde{M}^{-1}$ обобщенного изодромного управления $\widetilde{u}(t) = \widetilde{K}\widetilde{\zeta}(t)$.
9. Сформировать реализационную версию закона обобщенного изодромного управления (ЗОИУ)

$$\tilde{u}(t) = \tilde{K}_{\varepsilon}\varepsilon(t) + \tilde{N}\tilde{\zeta}_H \tag{28}$$

на основе измерения ошибки $\varepsilon(t)$ по выходу системы и вектора состояния $\tilde{\zeta}_H$ динамического наблюдателя вектора $\tilde{\zeta}$ ошибки слежения по состоянию

$$\dot{\zeta}_H = \tilde{F}_H\zeta_H + \tilde{L}_H\varepsilon(t), \tag{29}$$

опирающегося на модельное представление $\dot{\zeta}(t) = \tilde{F}\tilde{\zeta}(t);\quad \varepsilon(t) = \tilde{C}_y\tilde{\zeta}(t)$, чтобы матричные компоненты (25) вычислить в силу соотношения

$$\left(\tilde{K}_{\varepsilon}, N\right) = \arg\left\{\left[\tilde{K}_{\varepsilon}\quad N\right]\begin{bmatrix}\tilde{C}\\ \tilde{\Pi}\end{bmatrix} = \tilde{K}\right\}, \tag{30}$$

для которого матрицу $\tilde{\Pi}$ вычислить из решения уравнения Сильвестра

$$\tilde{\Pi}\tilde{F} - \tilde{F}_H\tilde{\Pi} = -\tilde{L}_H\tilde{C}. \tag{31}$$

10. Провести компьютерный эксперимент в среде программой оболочки MATLAB с целью проверки корректности назначения собственных значений матрицы состояния наблюдателя (29) вектора ошибки слежения по состоянию на основе медианной версии интервального модельного представления агрегированной системы

$$\dot{\nu}(t) = [F_{\nu}]\nu(t);\ \nu(0)\ \varepsilon(t) = C_{\nu}\nu(t), \tag{32}$$

где

$$\nu = col\left\{\tilde{\zeta},\ \Theta_{\zeta_H} = \Pi\tilde{\zeta} - \tilde{\zeta}_H\right\};\ [F_{\nu}] = \begin{bmatrix}\left[\tilde{F}\right] & \tilde{B}N\\ 0 & \tilde{F}_H\end{bmatrix};\ C_{\nu} = \begin{bmatrix}\tilde{C} & 0\end{bmatrix}. \quad \blacksquare\ (30)$$

Следует заметить, что введение в состав системы наблюдателя не увеличивает оценки относительной интервальности матрицы состояния спроектированной системы, поэтому необходимость контроля его влияния на эту оценку исчезает [1]. Более того, оценка относительной интервальности матрицы состояния спроектированной системы в предположении, что вектор ошибки слежения по состоянию полностью измерим, формируется в п. 5 процедуры синтеза тела алгоритма, поэтому по завершении выполнения алгоритма $\delta_I\tilde{F}$ оказывается вычисленной.

Пример

Для иллюстрации полученных результатов рассматривается изодромное управление объектом с интервальными матрицами состояния и управления, которые имеют вид $[A] = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ [-1;0] & [-10;10] & [-7;7]\end{bmatrix}$, $[B] = \begin{bmatrix}0 & 0 & [1.5;0.5]\end{bmatrix}^T$. Матрица выхода объекта имеет фиксированные параметры и записывается в форме $C = \begin{bmatrix}1 & 0 & 0\end{bmatrix}$. Матричные компоненты источника конечномерного входного воздействия $\dot{z}(t) = \Gamma z(t);\ z(0);\ g(t) = Pz(t)$ в случае гармонического входного воздействия при $\omega = 5c^{-1}$ получают реализации $\Gamma = \begin{bmatrix}0 & 5\\ -5 & 0\end{bmatrix}$, $P = \begin{bmatrix}1 & 0\end{bmatrix}$. Модифицируем ОУ путем введения буферной системы согласно п. 3 алгоритма с тем, чтобы интервальной оказалась только расширенная матри-

ца состояния $[\widetilde{A}]$, так что $[\widetilde{A}]=\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ [-1;0] & [-10;10] & [-7;7] & [1.5;0.5] & 0 \\ 0 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & -5 & 0 \end{bmatrix}$. Матрицы управления $\widetilde{B}$ и выхода $\widetilde{C}$ расширенной системы соответственно принимают вид $\widetilde{B}=[0 \quad 0 \quad 0 \quad 0 \quad 1]^T$, $\widetilde{C}=[1 \quad 0 \quad 0 \quad 0 \quad 0]$, причем пара матриц $(\widetilde{A}_0, \widetilde{B})$ – управляемая. Реализуем алгоритм синтеза обобщенного изодромного управления, опирающегося на обобщенное модальное управление. Полученные результаты приводятся в виде кривых $y(t), z(t), \varepsilon(t)$ (рис. 1).

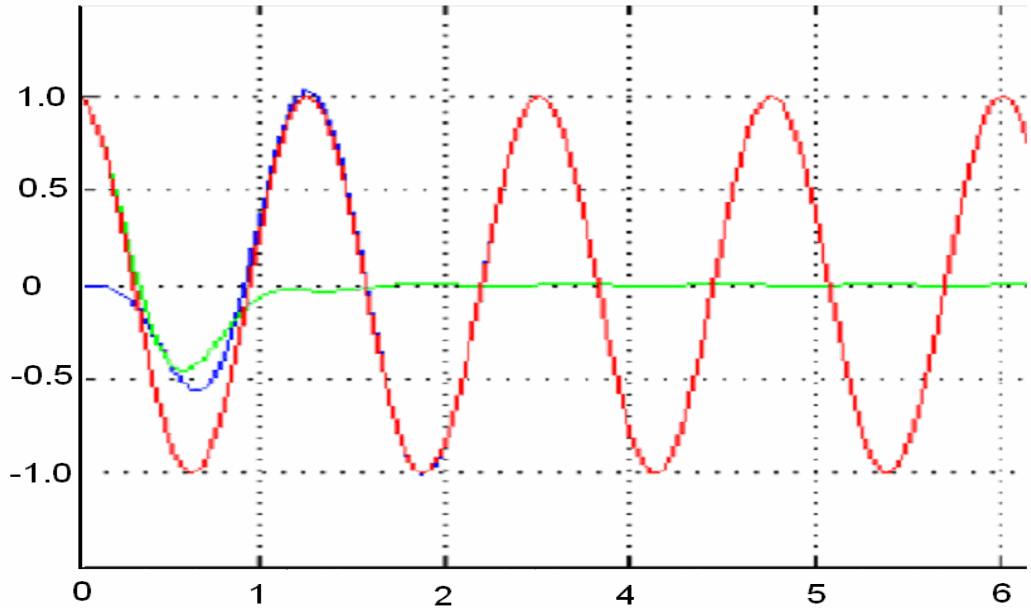


Рис. 1. Траектории $y(t), z(t), \varepsilon(t)$

Заключение

Агрегирование исходного объекта управления и некоторой буферной системы минимальной размерности позволяет сохранить интервальность только в матрице состояния агрегированного объекта. Руководствуясь предложенным алгоритмом, становится возможным управлять интервальностью путем влияния на медианную составляющую матрицы состояния агрегированной системы или через назначения соответствующих значений матрицы обратных связей, а также осуществлять контроль относительной интервальности матричных компонентов агрегированной системы с использованием аппарата теории чувствительности. В силу специфики задачи управления динамическим объектом с интервальными параметрами при конечномерном задающем воздействии наибольшей конструктивностью обладают функции траекторной чувствительности.

Литература

1. Никифоров В.О., Ушаков А.В. Управление в условиях неопределенности: чувствительность, адаптация, робастность. СПб: СПбГИТМО(ТУ), 2002.

2. Уонэм М. Линейные многомерные системы управления: Геометрический подход. / Пер. с англ. М.: Наука, 1980. 376 с.
3. Ланкастер П. Теория матриц. / Пер. с англ. М.: Наука. Главная редакция физико-математической литературы, 1984.
4. Акунов Т.А., Алишеров С., Оморов Р.О., Ушаков А.В. Матричные уравнения в задачах управления и наблюдения непрерывными объектами. Бишкек: Илим, 1991.
5. Синтез дискретных регуляторов при помощи ЭВМ / В.В. Григорьев, В.Н. Дроздов, В.В. Лаврентьев, А.В. Ушаков. Л: Машиностроение, Ленингр. отд-ние, 1983.
6. Ушаков А.В. Обобщенное модальное управление. // Изв. вузов. Приборостроение. 2000. Т.43. №3.

АЛГОРИТМ УПРАВЛЕНИЯ ДВИЖЕНИЕМ ШАГАЮЩЕГО РОБОТА

Р.А. Алексеев, И.В. Мирошник

Рассматривается задача синтеза алгоритма управления движением шагающего робота с использованием методов траекторного планирования и согласованного управления. Формулируются локальные задачи управления, которые сводятся к простым задачам стабилизации задачно-ориентированных координат. Приведены результаты моделирования

1. Введение

Антропоморфные шагающие механизмы являются в настоящее время одним из ведущих направлений научно-технических разработок. Сейчас известно не менее 160 реализаций шагающих аппаратов[7–10, 12–13]. Любое устройство такого типа состоит из цельной или составной верхней части – корпуса, платформы с оборудованием, торса с руками (манипуляторами), кабины с оператором и некоторого количества механических ног (педипуляторов). Существуют механизмы на шести, четырех, двух и даже на трех ногах, а также конструкции смешанного типа, например, имеющие две ноги и два колеса. Двуногие роботы статически наименее устойчивы, однако в определенных направлениях деятельности именно их использование может быть оптимальным.

Двуногая ходьба предпочтительна там, где обстановка не позволяет маневрировать громоздкой четырех- или шестиногой платформе, а сложный характер поверхности (щербины, неровности, ступени, трещины и проч.) не дает возможности применять колесный способ передвижения. Двуногая ходьба приоритетна у роботов сферы обслуживания, домашнего хозяйства, при работе в различных человеческих пространствах, производственных помещениях, шахтах, тоннелях. Также это могут быть опасные и вредные для людей зоны.

2. Плоскость основного движения

Движения робота можно разделить на движения в продольной (сагиттальной) плоскости и движения в поперечной (латеральной) плоскости. Движения в поперечной плоскости служат лишь для поддержания устойчивости. Основные же движения производятся в продольной плоскости, поэтому многие разработчики на первых порах рассматривают лишь продольное движение, на макете блокируя боковое движение неким устройством поддержки, которое жестко удерживает робота от падений в стороны, но при этом совершенно не мешает ему падать по курсу (вперед или назад) [8]. В связи с этим мы ограничиваемся рассмотрением плоского продольного движения.

3. Методика траекторного планирования и согласованного управления

Применение метода траекторного планирования предполагает вначале выбор стратегии движения и задание эталонных точек циклов (точек сопряжения). Далее следует построение кривых требуемых траекторий, соединяющих эти точки, а затем получение условий согласования, определяющих некие соотношения в движениях отдельных звеньев [2–4, 6, 7].

Пусть $\boldsymbol{x}$ – вектор переменных состояния объекта. Каналы управления независимы и нормированы:

$$\dot{x} = u \qquad (1)$$

Для постановки задачи управления вводим вектор условий (задачно-ориентированные координаты), который может содержать отклонения ***e*** и скорости ***s***, определяемые через ***x***:

$$\begin{bmatrix} e \\ s \end{bmatrix} = \begin{bmatrix} \varphi(x) \\ \psi(x) \end{bmatrix} = \Phi(x) \qquad (2).$$

Замечание: Условия вида $\varphi(x) = 0$ называются *условиями согласования*.

Дифференцируя (2), имеем эквивалентную (задачно-ориентированную) модель

$$\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix} = \frac{\partial \Phi}{\partial x}\dot{x} \qquad (3).$$

Обозначим Якобиан отображения

$$G(x) = \frac{\partial \Phi}{\partial x} = \begin{bmatrix} \partial\varphi/\partial x \\ \partial\psi/\partial x \end{bmatrix}, \qquad (4)$$

тогда

$$\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix} = G(x)u\,. \qquad (5)$$

Из (5), учитывая (1), можно получить выражение для вектора управления:

$$u = G^{-1}\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix}. \qquad (6)$$

Модель (5) приобретает форму

$$\dot{e} = u_e \qquad (7а)$$

$$\dot{s} = u_s \qquad (7б)$$

где $\boldsymbol{u_e}$ – вектор поперечного управления, $\boldsymbol{u_s}$ – вектор продольного управления. Наконец, выбирая закон управления

$$u_e = -Ke, \qquad (8а)$$

$$u_s = V^*, \qquad (8б)$$

получаем требуемые свойства замкнутой системы – минимизацию вектора отклонений ***e*** (задача (а)) и стабилизацию вектора проходимого за единицу времени пути ***s***, т.е. удержания вектора требуемых скоростей $\boldsymbol{V^*}$ (задача (б)).

На рис. 1 показана структура системы с траекторным управлением.

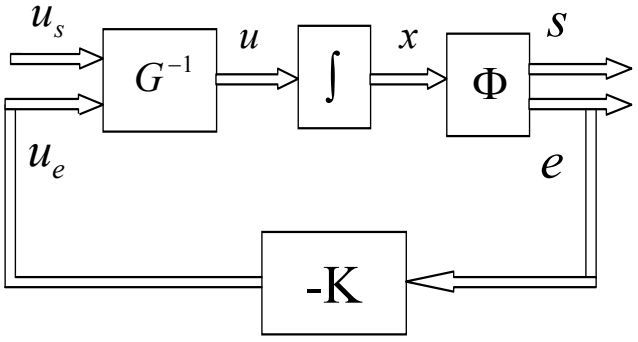


Рис. 1. Общая структура системы траекторного управления

4. Модель шагающего механизма

Кинематическую модель двуногого робота представляем семизвенной цепью (см. рис.2). В качестве начальной точки берем носок опорной, т.е. передней в данной фазе, ноги. При этом носок другой (задней) ноги считается свободным (это конец последнего звена всей цепи). Тогда система координат такой модели жестко связана с поверхно-

стью опоры и называется неподвижной. Каждое последующее звено развернуто относительно предыдущего на угол q_i и имеет протяженность l_i [5].

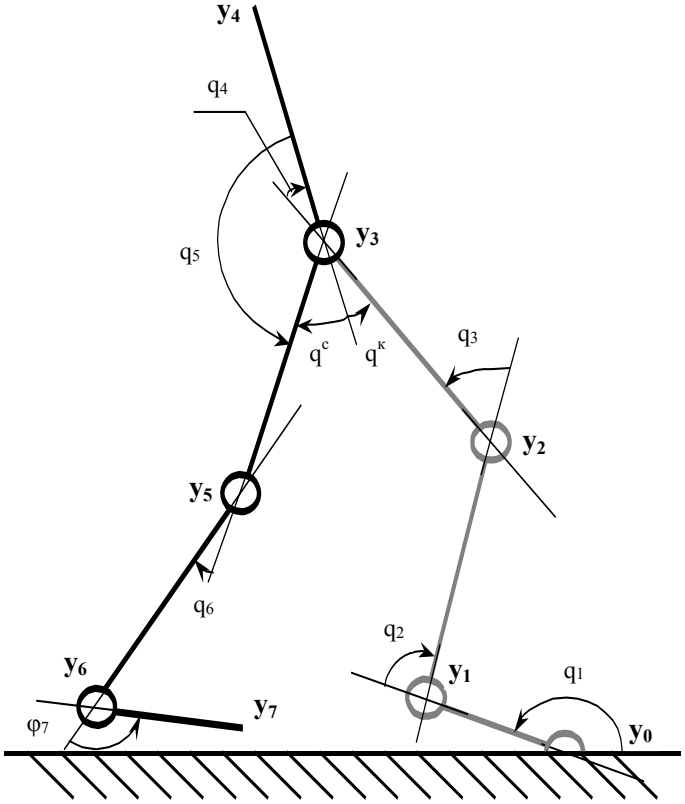


Рис. 2. Кинематическая схема робота

Замечание. На рис. 2 свободная ступня умышленно оторвана от поверхности, чтобы показать возможность движения. Точка y_7 в модели не привязана к опоре.

5. Объект управления

Для кинематической цепочки из n звеньев механику любого i-го звена ($i = \overline{1,n}$) опишем рекуррентной системой уравнений:

$$\begin{cases} \dot{q}_i = B_i u_i & (9) \\ \alpha_i = \sum_{j=1}^{i} q_j \quad , & (10) \\ y_i^T = y_{i-1}^T + z_i^T \cdot T(\alpha_i) & (11) \end{cases}$$

где q_i – угол поворота i-го привода (обобщенная координата), u_i. – управляющее воздействие на привод i-го сочленения, $y_i^T = \begin{bmatrix} y_{i1} & y_{i2} \end{bmatrix}$ – вектор декартовых координат конечной точки i-го звена, α_i – углы (абсолютной) ориентации звеньев, $T(\alpha_i) = \begin{bmatrix} \cos(\alpha_i) & \sin(\alpha_i) \\ -\sin(\alpha_i) & \cos(\alpha_i) \end{bmatrix}$ – матрица вращения, $z_i^T = \begin{bmatrix} z_{i1} & z_{i2} \end{bmatrix}$ – вектор координат конечной точки i-го звена (в системе координат звена). Поскольку в системе нет изломанных звеньев, то поперечная протяженность звеньев отсутствует, т. е. $z_i^T = \begin{bmatrix} l_i & 0 \end{bmatrix}$, где l_i – длина звена.

Уравнение (9) представляет собой кинематическую модель механизма, а уравнения (10) и (11) описывают прямую кинематику робота по угловым и линейным (декартовым) координатам звеньев.

При $i = \overline{1,n}$ обозначим: $\boldsymbol{q}=\{q_i\}$ – вектор обобщенных координат, $\boldsymbol{u}=\{u_i\}$ – вектор управлений, $\boldsymbol{\alpha}=\{\alpha_i\}$ – вектор абсолютных углов поворота звеньев, $\boldsymbol{y}=\{y_i^T\}$ – вектор декартовых координат конечных точек звеньев размерности [n×2]. Теперь перейдем к компактной форме описания объекта управления (см. рис. 3):

$$\dot{q} = Bu\,, \tag{12}$$

$$\alpha = R(q)\,, \tag{13}$$

$$y = h(\alpha)\,. \tag{14}$$

Здесь $\boldsymbol{B}=diag\{b_i\}$ **–** диагональная матрица коэффициентов передачи, $\boldsymbol{R}$ – матрица перехода от относительных угловых координат к абсолютным угловым координатам, которая для случая одной кинематической цепочки является нижней диагональной матрицей и рассчитывается по формуле

$\boldsymbol{R}= \{r^T_{ij}\}$,

где $r_i^T = \sum_{j=1}^{i} q_j$, $\boldsymbol{h}$ – вектор-функция, рассчитываемая по формуле

$$h = \begin{bmatrix} y_0^T \\ y_1^T \\ \vdots \\ y_{n-1}^T \end{bmatrix} + \begin{bmatrix} z_1^T T(\alpha_i) \\ z_2^T T(\alpha_2) \\ \vdots \\ z_n^T T(\alpha_n) \end{bmatrix}. \tag{15}$$

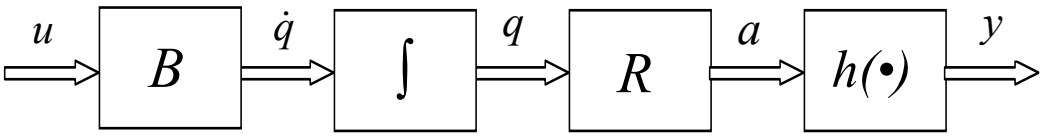


Рис. 3. Структурная схема механизма как объекта управления

6. Концепция ходьбы

Изучив различные алгоритмы ходьбы [1, 7-12], можно сделать ряд выводов.

1. Перемещение приставным шаганием (шаг одной ногой, затем приставление другой ноги) снижает скорость общего перемещения механизма.
2. Использование безопорной фазы («фазы полета») в цикле шага связано со сложностью определения координат всего робота в этой фазе.
3. Двухфазный алгоритм проще трехфазного по реализации, и в нем меньше сопряжений (переключений режимов).
4. Фазы с отсутствием горизонтального движения торса относительно земли существенно снижают общую скорость робота.
5. Остановки движения торса в какой-либо фазе ходьбы приводят к рывкам при движении массивной верхней части, что существенно снижает устойчивость походки.
6. Полное распрямление маховой ноги перед отрывом и при касании поверхности существенно усложняет процесс управления ходьбой из-за возможного заклинивания приводов в распрямленном положении;
7. Циклическое вертикальное перемещение торса в цикле ходьбы тоже ослабляет устойчивость походки и расходует лишнюю энергию.

Проанализировав известные варианты походки, изложим основные положения желаемого алгоритма ходьбы.

1. Для сохранения роботом статического равновесия горизонтальная проекция его центра масс должна проходить через поверхность опоры.
2. Цикл одиночного шага должен состоять из двух фаз – переноса тяжести с задней ноги на переднюю (двухопорная стадия) и перемещения маховой ноги вперед на шаг (одноопорная стадия).
3. Для избежания рывков в обеих фазах массивный торс должен сохранять свою горизонтальную скорость постоянной.
4. Не допускаются распрямления маховой ноги перед отрывом и при касании поверхности, во избежание неоднозначности отработки траекторий.
5. Торс не должен иметь вертикального движения, так как оно ослабляет устойчивость походки и расходует лишнюю энергию.

Синергия движений в двух фазах разработанной походки приведена на рис. 4.

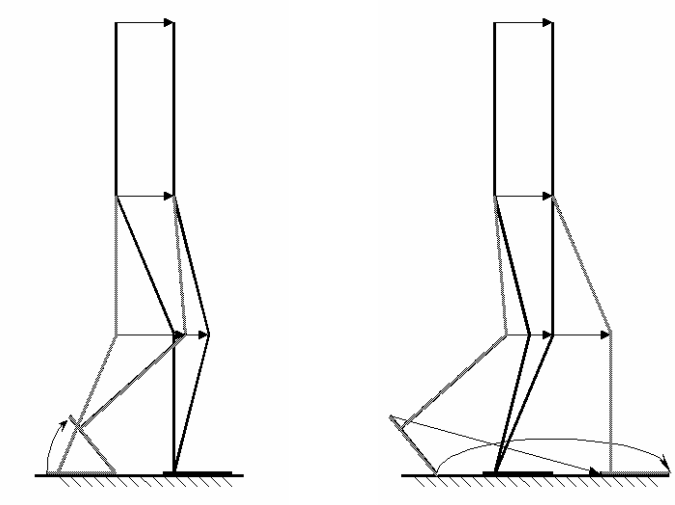


Рис. 4. Две фазы движения (перенос опорной нагрузки и переставление ноги)

7. Планирование траекторий

Процедура синтеза предусматривает получение модели движения в декартовых координатах, преобразование к задачно-ориентированным координатам, а затем синтез управлений, обеспечивающих решение поставленных задач (а) и (б).

Алгоритм движения робота содержит три группы условий.

Опорная нога и корпус в обеих фазах движения:

1. Стопа опорной ноги горизонтальна $y_{11} = c_1$,
   $c_1 = -15$ (горизонтальная координата голеностопного шарнира);
2. Высота таза над поверхностью постоянна $y_{32} = c_2$,
   $c_2 = 30(2 + \sqrt{3})$ (вертикальная координата тазобедренных шарниров);
3. Корпус движется горизонтально $\dot{y}_{41} = -V_к$,
   $V_к = 1$ (скорость продольного перемещения корпуса);
4. Корпус держится вертикально $\alpha_4 = Q_1$,
   $Q_1 = \pi / 2$ (угол корпуса относительно горизонтали).

Маховая нога в фазе переноса:

1. Носок не перемещается по горизонтали $y_{71} = c_3$,
   $c_3 = -30$ (горизонтальная координата носка);

2. Носок опирается на поверхность $y_{72} = c_4$,

   $c_4 = 0$ (вертикальная координата носка);

3. Ноги движутся в противоходе $\alpha_3 + \alpha_4 = Q_2$,

   $Q_2 = \pi / 6$ (суммарный угол положений обоих бедер).

   Маховая нога в фазе шага:

1. Пятка движется по наклонной $y_{62} = -ky_{61} + b$,

   $k = 0{,}25$ и $b = 3{,}75$ (угол наклона и высота нисходящей траектории);

2. Носок перемещается на шаг $\dot{y}_{71} = 4V_к$;

3. Носок описывает дугу косинуса $y_{72} = -A \cdot \cos(\omega y_{71})$,

   где $A = 6$ (высота подъема носка), $\omega = \dfrac{2\pi V_к}{L_ш} = \dfrac{2\pi}{60}$ (частота одиночного шага), $L_ш$=60 (длина шага).

8. Постановка задачи

Введем задачно-ориентированные координаты – отклонения ***e***=*{e_i}*, где $i = \overline{1,k}$ и пути ***s***=*{s_i}*, где $i = \overline{1,l}$. Общее число условий для каждой фазы движения должно быть равно числу звеньев механизма $k + l = n$, в данном случае n=7.

Фаза переноса:

1. $e_1 = y_{11} - c_1$
2. $e_2 = y_{32} - c_2$
3. $s_1 = \dot{y}_{31} = V_к$
4. $e_3 = \alpha_4 - Q_1$
5. $e_3 = y_{71} - c_3$
6. $e_4 = y_{72} - c_4$
7. $e_5 = \alpha_3 + \alpha_4 - Q_2$

Фаза шага:

1. $e_1 = y_{11} - c_1$
2. $e_2 = y_{32} - c_2$
3. $s_1 = \dot{y}_{31} = V_к$
4. $e_3 = \alpha_4 - Q_1$
5. $e_6 = ky_{61} + y_{62} - b$
6. $s_2 = \dot{y}_{71} = 4V_к$
7. $e_7 = y_{72} - A\cos(\omega y_{71})$

Значения параметров: $c_1 = -15$; $c_2 = 30(2 + \sqrt{3})$; $c_3 = -30$; $c_4 = 0$; $V_к = 1$; $Q_1 = \pi / 2$; $k = 0{,}25$; $b = 3{,}75$; $A = 6$; $\omega = \pi/30$.

Совокупность этих уравнений может быть записана в компактной форме

$$\begin{bmatrix} e \\ s \end{bmatrix} = \Phi(y, \alpha, q), \qquad (16)$$

где **e**=*{e_i}*, **s**=*{s_j}*, *ı=1…k, j=1…l, k+l=n,* где n – число звеньев механизма (степеней свободы). Задача управления состоит в минимизации вектора отклонений ***e*** (задача (а)) и стабилизации вектора проходимого пути ***s*** за единицу времени, т. е. удержании вектора требуемых скоростей $\boldsymbol{V}^* = \dot{s}$ (задача (б)).

9. Синтез управления

Дифференцируя уравнения прямой кинематики (10)–(11) и подставляя (9), получаем:

$$\dot{y}_i^T = \dot{y}_{i-1}^T + \dot{\alpha}_i z_i^T T(\alpha_i) E = \dot{y}_{i-1}^T + r_i^T \dot{q} \cdot z_i^T T(\alpha_i) E = \dot{y}_{i-1}^T + r_i^T Bu \cdot z_i^T T(\alpha_i) E$$

т. е.

$$\dot{y}_i^T = \dot{y}_{i-1}^T + z_i^T T(\alpha_i) E r_i^T B u \qquad (17)$$

где $E = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$,. или в компактной форме

$$\dot{y} = G_y(\alpha) B u, \qquad (18)$$

где $G_y(\alpha) = \frac{\partial y}{\partial \alpha} \cdot \frac{\partial \alpha}{\partial q}$. Дифференцируя уравнение (16), получаем задачно-ориентированную модель робота

$$\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix} = \left(\frac{\partial \Phi}{\partial y} G_y(\alpha) R + \frac{\partial \Phi}{\partial q} \right) B u \qquad (19)$$

или

$$\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix} = G(q, \alpha, y) \tilde{u}, \qquad (20)$$

где введены обозначения

$$G = \frac{\partial \Phi}{\partial y} G_y(\alpha) R + \frac{\partial \Phi}{\partial q}, \quad \tilde{u} = B u.$$

Следуя стандартной методике согласованного управления [2–5, 7], осуществим преобразование управления по формуле

$$G(q, \alpha, y) \tilde{u} = \begin{bmatrix} u_e \\ u_s \end{bmatrix} \qquad (21)$$

и перепишем модель (20) в виде

$$\begin{bmatrix} \dot{e} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} u_e \\ u_s \end{bmatrix}, \qquad (22)$$

где $\boldsymbol{u_e}$ – вектор управлений по отклонениям $\boldsymbol{e}$, $\boldsymbol{u_s}$ – вектор управлений по перемещениям $\boldsymbol{s}$. Выбирая

$$u_e = -Ke, \qquad (23а)$$

$$u_s = V^*, \qquad (23б)$$

где $\boldsymbol{K}=diag\{k_i\}$, $k_i>0$ – коэффициенты обратной связи, получаем

$$\dot{e} = -Ke, \qquad (24а)$$

$$\dot{s} = V^*, \qquad (24б)$$

что обеспечивает минимизацию отклонений $\boldsymbol{e}$ и поддержание требуемых скоростей продольного движения $\boldsymbol{V}^*$, т.е. решение задач (а) и (б).

Для нахождения вектора управляющих воздействий системы $\boldsymbol{u}$ воспользуемся выражениями (21) и (22) и получим:

$$\tilde{u} = G^{-1}(q, \alpha, y) \begin{bmatrix} u_e \\ u_s \end{bmatrix}, \qquad (25)$$

$$u = B^{-1} \tilde{u}. \qquad (26)$$

Структурная схема системы управления (см. рис. 5) состоит из объекта управления (уравнения (12)–(14)), блока получения (расчета или измерения) отклонений e_i (уравнение (16)), обратного преобразования управлений (выражения (25)–(26)), задатчика продольной скорости (24б) и регулятора отклонений (24а), образующего основные обратные связи системы.

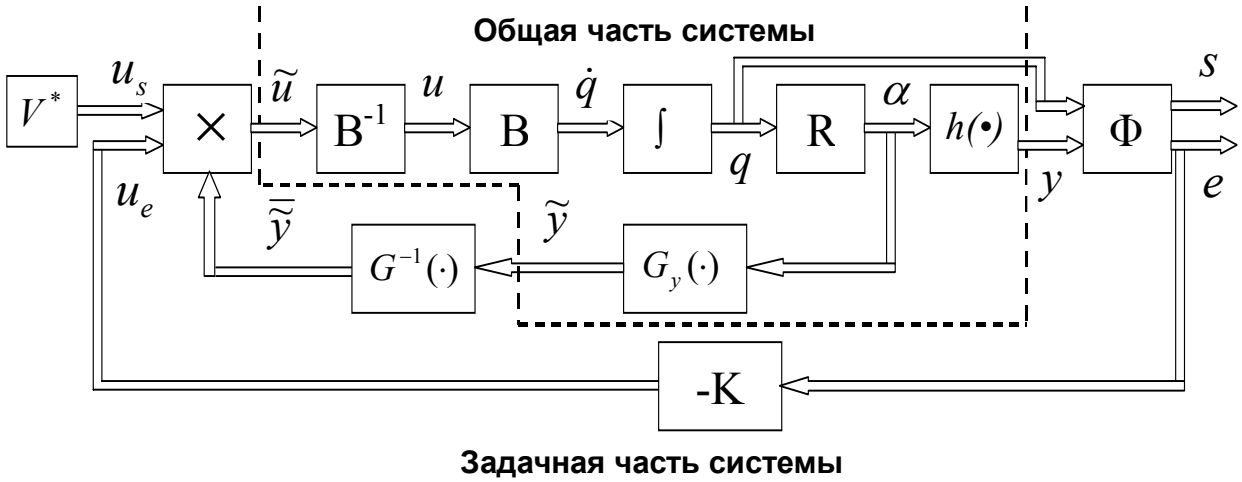


Рис. 5. Структурная схема системы управления двуногим роботом

10. Результаты моделирования

Для проверки разработанного алгоритма проведено моделирование на программном продукте MatLab 4.0 Simulink 1.2. Поведение звеньев механизма, полученное в результате моделирования сдвоенного шага (полный цикл ходьбы), представлено в виде циклограммы (см. рис. 6).

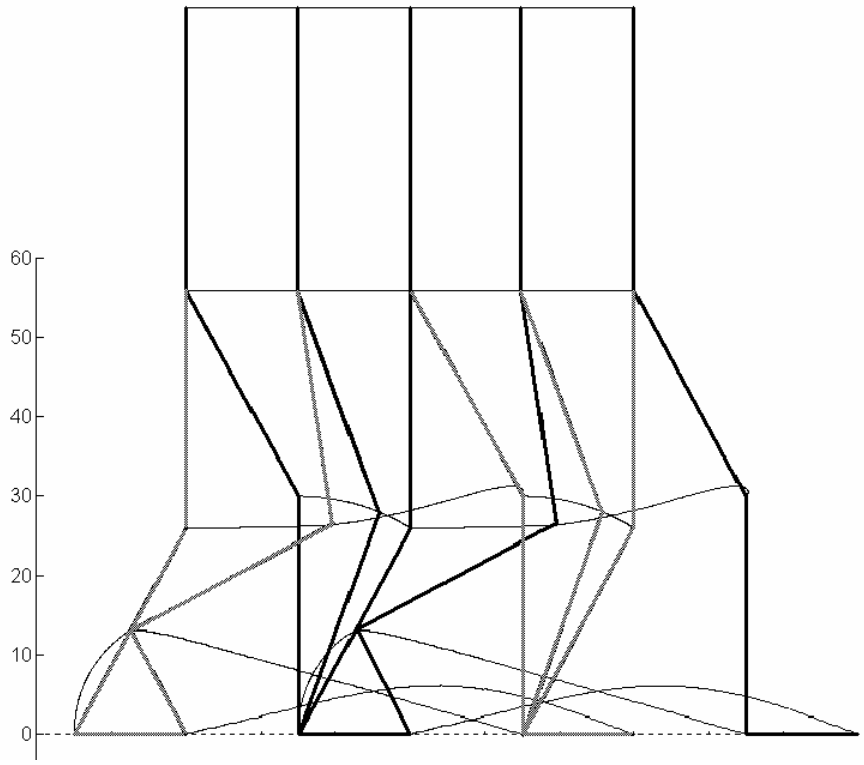


Рис. 6. Циклограмма движения робота (моделирование)

Полученная циклограмма полностью соответствует заданному алгоритму ходьбы. Шаги правой и левой ногами полностью идентичны. Соблюдается ширина шага. Постоянна высота корпуса над поверхностью ходьбы, т.е. отсутствует вертикальное движение корпуса. Скорость горизонтального перемещения торса относительно земли постоянна и не имеет рывков и замедлений.

Литература

1. Вукобратович М.. Шагающие роботы и антропоморфные механизмы. М.: Мир, 1976.
2. Мирошник И.В. Согласованное управление многоканальными системами. Л.: Энергоатомиздат, 1990.
3. Мирошник И.В., Никифоров В.О. Методы координации в задачах планирования и управления пространственным движением манипуляционных роботов. СПб: Наука, 1998.
4. Мирошник И.В., Никифоров В.О., Фрадков А.Л. Нелинейное и адаптивное управление сложными динамическими системами. СПб: Наука, 2000.
5. Мирошник И.В. Нелинейные системы. Анализ и управление. СПб: СПб ИТМО (ТУ), 2002.
6. Механика промышленных роботов. Кн. 1. Е. И. Воробьев, О. Д. Егоров, С. А. Попов. Расчет и проектирование механизмов. М.: Высшая школа, 1988.
7. A. Albert. Intelligente Bahnplanung und Regelung für einen autonomen, zweibeinigen Roboter. // VDI-Fortschrittberichte, Reihe 8, Nr. 927, VDI-Verlag, Düsseldorf, 2002
8. T. Asfour, K. Berns, J. Schelling and R. Dillmann. Programming of manipulation tasks of the humanoid robot ARMAR. IEEE Int. Conf. Advanced Robotics, Tokyo, Oct. 1999.
9. C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas-de-Wit and J. W. Grizzle. RABBIT: A Testbed for Advanced Control Theory. // IEEE Control Systems Mag., v. 23, №5, Oct. 2003, pp. 57-79
10. Y. Fujimoto, A. Kawamura. Simulation of an Autonomous Biped Walking Robot Including Environmental Force Interaction. // IEEE Robotics & Automation Mag., v. 5, №2, June 1998, pp 33-41.
11. A. Ijspeert, J. Nakanishi and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. // IEEE ICRA, Washington, May 2002, pp. 1398-1403.
12. Kajita, S. and Tani, K., An analysis of experimentation of a biped robot Meltran II. Proc. of 3rd International Workshop on Advanced Motion Control (UC Berkeley), pp.417-420, 1993.
13. S. Talebi, M. Buehler and E. Papadopoulos. Towards Dynamic Step Climbing For A Quadruped Robot with Compliant Legs.

МЕТОД ВНУТРЕННЕЙ МОДЕЛИ В ЗАДАЧЕ АКТИВНОЙ ВИБРОЗАЩИТЫ

Г.В. Лукьянова, В.О. Никифоров, И.В. Сергачев

Представлен синтез регулятора компенсации внешних возмущений, основанный на использовании метода внутренней модели. Показано практическое применение предлагаемого регулятора для системы активной виброзащиты (САВ), приведены результаты экспериментов.

Введение

Задача компенсации внешних возмущений, в частности – вибраций, является одной из актуальных проблем современного машиностроения. Один из подходов к ее решению состоит в использовании принципа внутренней модели [1–4]. В соответствии с данным принципом внешнее детерминированное возмущение рассматривается в качестве выхода автономной системы (так называемого *генератора возмущений*), возбуждаемого ненулевыми начальными условиями. Для полной компенсации внешнего возмущения модель его генератора должна быть соответствующим образом учтена (воспроизведена) в структуре регулятора.

В настоящее время принцип внутренней модели хорошо разработан для широких классов линейных [1–6] и нелинейных систем [7–9], систем с неизвестными параметрами [10–12]. Также он нашел свое применение в задачах управления реальными техническими объектами [13–16]. Одним из примеров такого объекта может служить система активной виброзащиты.

Известно, что в низкочастотной области спектра возможности снижения уровня вибрации механизмов до требуемых норм традиционными пассивными средствами типа вибропоглощения и виброизоляции весьма ограничены. В транспортных средствах типа судов такие ограничения еще более существенны, поскольку практически отсутствуют резервы увеличения массы и объема для размещения пассивных средств защиты, тем больших, чем ниже граничная частота рабочего диапазона. Недостаточность реальных возможностей снижения уровня вибрации пассивными средствами привлекла пристальное внимание к системам активной защиты, не требующим больших масс и габаритов. В настоящее время можно считать общепризнанным, что активные системы защиты при снижении уровня вибрации являются органичным дополнением пассивным средствам в низкочастотной области спектра, расширяющими частотный диапазон в сторону более низких частот.

1. Постановка задачи

Конструкция макета системы активной виброзащиты представлена на рис. 1. Источник вибрации 1, расположенный на подвижной платформе 2, создает вибрации, условно представленные на рисунке в виде внешнего воздействия δ. Для защиты виброизолируемого основания 7 от вибрации δ в макете используются системы пассивного и активного подавления вибраций. Пассивной системой являются резиновые амортизаторы 3. Активным элементом САВ служит электромагнит (ЭМ) 8, расположенный на подвижной и неподвижной платформах 2 и 5, соответственно. Так как электромагнит содержит одну обмотку, то он может только притягивать к себе подвижную часть макета общей массой M.

Управление электромагнитом осуществляется следующим образом. Величины, снимаемые с датчиков силы 6, усредняются на сумматоре 12 и после преобразования в АЦП 13 подаются на один из входов регулятора 9. На второй вход регулятора подается сигнал с аналогово-цифрового преобразователя (АЦП) 11. На АЦП поступает сигнал с

датчика Холла 4, который преобразует величину напряженности магнитного поля в соответствующее ему напряжение. Регулятор на основе полученной информации вырабатывает управляющее воздействие, которое после преобразования цифро-аналоговым преобразователем (ЦАП) и усиления в ШИМ 10 подается в виде питающего напряжения на электромагнит 8.

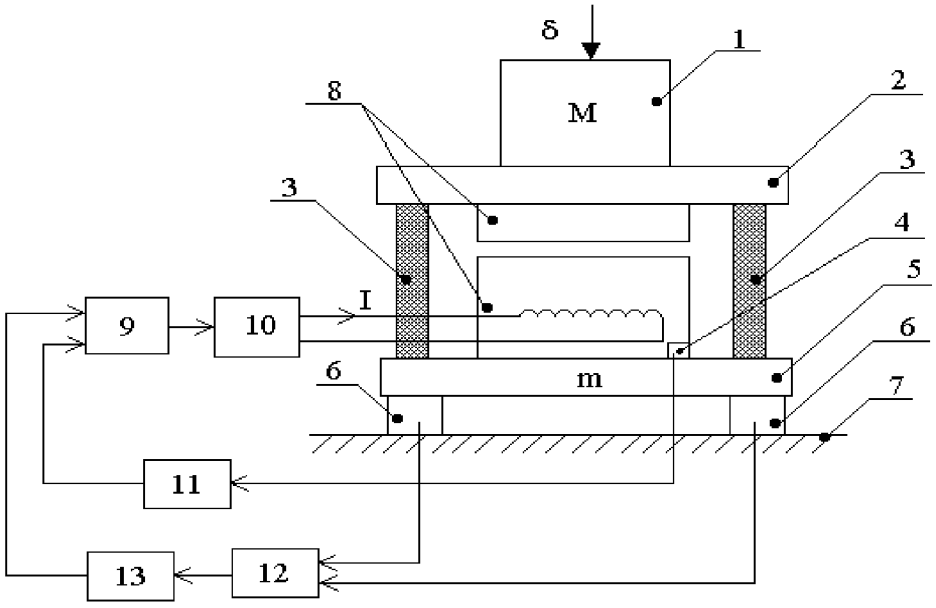


Рис. 1. Структурная схема САВ

Задача САВ состоит в компенсации внешних возмущений в виде вибраций в широкополосном диапазоне частот, т.е. необходимо синтезировать управление, которое:

- обеспечивает заданные динамические показатели качества (например, перерегулирование и время переходного процесса);
- обеспечивает заданную точность в установившемся или динамическом режиме, что в свою очередь требует компенсации возмущения.

2. Подключение обратной связи

Будем называть макет САВ с действующими на него возмущениями объектом управления (ОУ). Амплитудно-частотная характеристика (АЧХ) ОУ имеет вид, представленный на рис. 2 (кривая 1). АЧХ имеет ярко выраженный резонанс на частоте ω_p. Этот резонанс может быть вызван включением упругих амортизаторов в кинематическую цепь виброизолирующего устройства.

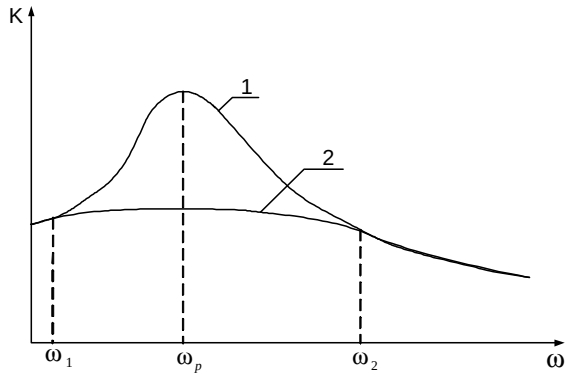


Рис. 2. Амплитудно-частотная характеристика ОУ

Стандартное решение задачи – синтез регулятора стабилизации на основе заданных динамических показателей качества. Структура регулятора стабилизации приведена на рис. 3, где $W_c(p)$ – передаточная функция регулятора; K_c – приведенный коэффициент обратной связи. Амплитудно-частотная характеристика стабилизированной системы приведена на рис. 2 (кривая 2).

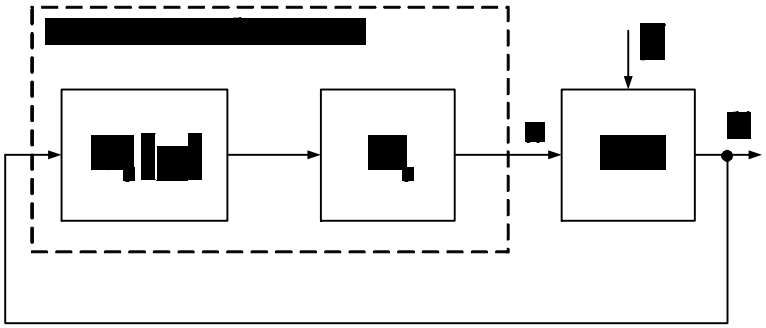


Рис. 3. Структурная схема стабилизированного ОУ

Для определения эффективности подавления возмущений рассматривают два показателя: разницу логарифмических амплитудных характеристик разомкнутого и замкнутого контуров на резонансной частоте $\Delta = L_p - L_з$ в дБ; диапазон подавления возмущений от ω_1 до ω_2 (см. рис. 2).

Эффективность подавления может быть улучшена за счет увеличения приведенного коэффициента K_c (см. рис. 4), но на практике невозможно увеличивать K_c до бесконечно больших значений. Таким образом, остается нерешенной задача повышения эффективности компенсации внешних возмущений, не усиливая коэффициент K_c.

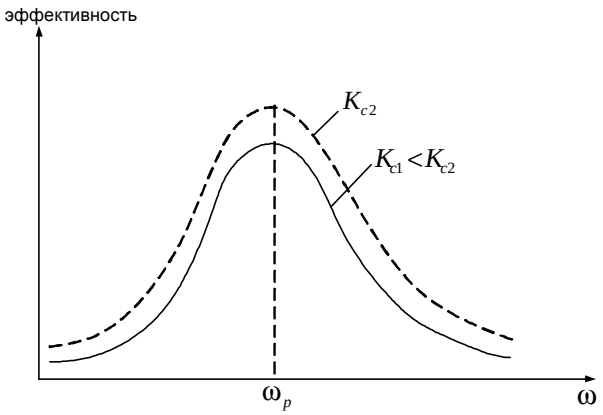


Рис. 4. Типичная характеристика эффективности подавления возмущений

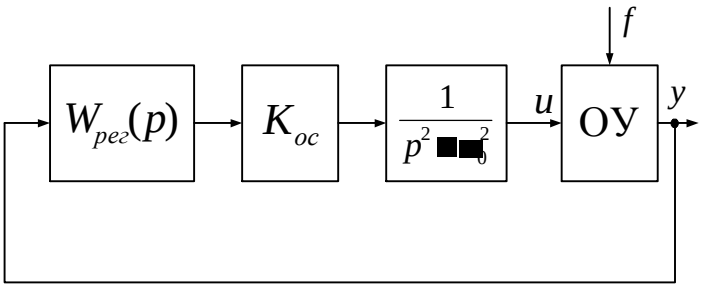


Рис. 5. Структурная схема замкнутой системы

Один из способов повышения эффективности состоит в использовании принципа внутренней модели, включенной в главную обратную связь. Рассмотрим пример подавления одной гармоники. Структурная схема замкнутой системы приведена на рис. 5, где $W_{рег}(p)$ – передаточная функция регулятора стабилизации; $K_{ос}$ – коэффициент обратной связи; $1/\left(p^2+\omega_0^2\right)$ – резонансный контур, обеспечивающий подавление возмущений на частоте ω_0 и повышающий эффективность подавления; f – внешнее возмущение, действующее на объект управления; u – сигнал управления. Данная модель позволяет обеспечить типичную характеристику, представленную на рис. 6. Из рисунка видно, что на частоте ω_0 происходит «вырезание» внешнего возмущения (кривая 1). Кривая 2 показывает эффективность подавления внешних возмущений.

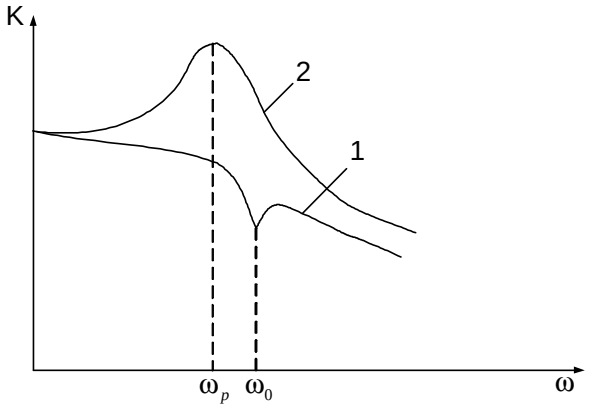


Рис. 6. Типичная характеристика

Перечислим недостатки такой схемы:

- при изменении ω_0 необходимо пересчитывать всю систему;
- стабилизация колебательной системы высокого порядка, что приводит к получению бесконечно больших коэффициентов обратной связи;
- сложность расчета регулятора стабилизации $W_{рег}(p)$, что делает невозможным экспериментальную подстройку.

3. Раздельный синтез

Раздельный синтез позволяет сначала решить задачу стабилизации объекта, а потом – задачу компенсации внешних возмущений без ухудшения динамических показателей качества системы.

Рассматривается линейный динамический объект

$$A(p)y(t)=kB(p)\big(u(t)+f(t)\big), \qquad (1)$$

где $y(t)$ – регулируемая переменная; $p=d/dt$ – оператор дифференцирования; $A(p)=p^n+a_{n-1}p^{n-1}+\text{h}+a_0$ – известный полином; $B(p)=p^m+b_{m-1}p^{m-1}+\text{h}+b_0$ – известный нормированный полином; k – постоянный коэффициент; $u(t)$ – сигнал управления; $f(t)$ – внешнее детерминированное возмущение. Полагаем, что на данный объект действует внешнее возмущение синусоидальной формы $f(t)=A\sin\omega_0$, где A – амплитуда, ω_0 – частота возмущения. Необходимо синтезировать управление, обеспечивающее стабилизацию объекта управления путем задания неявной эталонной модели. Для этого задается желаемый полином с требуемыми динамическими показателями качества замкнутой системы:

$$\Phi(p)=\frac{k_\Sigma}{p^{n-m}+\alpha_{n-m-1}p^{n-m-1}+\ldots+\alpha_0}=\frac{k_\Sigma}{\alpha(p)}, \quad (2)$$

где k_Σ – коэффициент усиления; $\alpha(p)$ – желаемый типовой характеристический полином. При решении сформулированной задачи полагаем, что $B(p)$ гурвицев, а полином $f(t)$ имеет преобразование Лапласа: $F(s)=L\{f(t)\}=\frac{\psi(s)}{\varphi(s)}$, где $\varphi(s)=s^\mu+\varphi_{\mu-1}p^{\mu-1}+\ldots+\varphi_0$.

4. Структура регулятора стабилизации

Сформируем управление в виде

$$u(t)=W_1(p)\big(\upsilon(t)-W_2(p)y(t)\big), \quad (3)$$

где $W_1(p)=\frac{k_\Sigma}{k}\frac{\gamma}{\overline{D}(p)B(p)}$, $W_2(p)=\frac{\overline{R}(p)}{\gamma(p)}$ – передаточные функции регулятора, $\upsilon(t)$ – сигнал управления, компенсирующий внешнее возмущение, который мы определим позже.

Полиномы $\overline{D}(p)=p^{n-m-1}+\overline{d}_{n-m-2}p^{n-m-2}+\ldots+\overline{d}_0$ и $\overline{R}(p)=\overline{r}_{n-1}p^{n-1}+\overline{r}_{n-2}p^{n-2}+\ldots+\overline{r}_0$ являются решением уравнения

$$\gamma(p)\alpha(p)=A(p)\overline{D}(p)+k_\Sigma(p)\overline{R}(p), \quad (4)$$

где $\gamma(p)=p^{n-1}+\gamma_{n-2}p^{n-2}+\ldots+\gamma_0$ – произвольный гурвицевый полином, определяющий скорость сходимости переходного процесса. Следовательно, требуемое управление определяется выражением:

$$u(t)=\frac{k_\Sigma}{k}\frac{\gamma}{\overline{D}(p)B(p)}\left(\upsilon(t)-\frac{\overline{R}(p)}{\gamma(p)}y(t)\right).. \quad (5)$$

Разрешая уравнение (4) относительно $\overline{D}(p)$ и $\overline{R}(p)$, получим алгоритм нахождения полиномов $\overline{D}(p)$ и $\overline{R}(p)$:

$$\begin{cases}\overline{d}_{n-m-2}=\alpha_{n-m-1}+\gamma_{n-2}-a_{n-1}\\ \overline{d}_{n-m-3}=\alpha_{n-m-2}+\gamma_{n-2}\alpha_{n-m-1}+\gamma_{n-3}-a_{n-1}\overline{d}_{n-m-2}-a_{n-2}\\ \ldots\\ \overline{d}_0=\gamma_m+\gamma_{m+1}\alpha_{n-m-1}+\gamma_{m+2}\alpha_{n-m-2}+\ldots-a_{m+1}-a_{m+2}\overline{d}_{n-m-2}-\ldots-a_{n-1}\overline{d}_1\\ \overline{r}_{n-1}=\gamma_{m-1}+\gamma_{m+1}\alpha_{n-m-1}+\ldots-a_m-a_{m+1}\overline{d}_{n-m-2}-\ldots-a_{n-1}\overline{d}_0\\ \ldots\\ \overline{r}_0=\gamma_0\alpha_0-a_0\overline{d}_0.\end{cases} \quad (6)$$

Структурная схема замкнутой системы приведена на рис. 7.

5. Структура регулятора компенсации возмущений

Рассмотрим линейный динамический объект

$$\alpha(p)y(t)=k_\Sigma\big(\upsilon(t)+\overline{f}(t)\big), \quad (7)$$

где $y(t)$ – регулируемая переменная, $\alpha(p)=p^n+\alpha_{n-1}p^{n-1}+\ldots+\alpha_0$ – известный нормированный полином; $p=d/dt$ – оператор дифференцирования; k_Σ – коэффициент уси-

ления; $\upsilon(t)$ – сигнал управления; $\bar{f}(t) = \dfrac{k}{k_\Sigma} \dfrac{B(p)\overline{D}(p)}{\gamma(p)} f(t)$ – внешнее возмущение, действующее на стабилизированный объект.

Рассматриваемая задача состоит в синтезе управления (построении контура компенсации возмущений), обеспечивающего компенсацию возмущений на заданной частоте ω_0. Контур подавления возмущений состоит из регулятора $Q(p)$ и коэффициента усиления ρ (см. рис. 8).

К контуру компенсации возмущений предъявляются следующие требования:

1) для любых $\overline{\rho}_{\max} > \rho > 0$ не нарушает устойчивость всей системы;

2) не изменяет коэффициента передачи по возмущению замкнутой системы;

3) минимально искажает частотную характеристику.

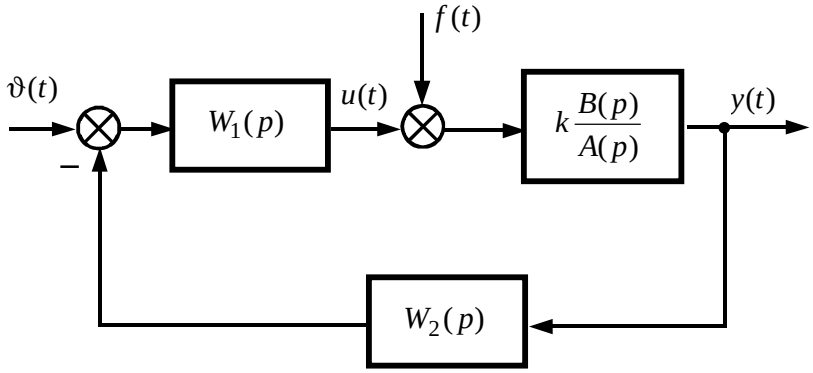


Рис. 7. Структурная схема замкнутой системы с внешним возмущением

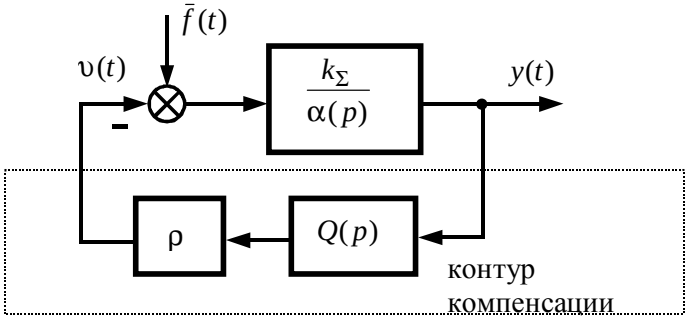


Рис. 8. Структурная схема замкнутой системы при нулевом задающем воздействии

Сформируем управление $\upsilon(t)$ в виде:

$$\upsilon(t) = -\rho Q(p) y(t) = -\rho \frac{R(p)}{\varphi(p) D(p)} y(t), \tag{8}$$

где полиномы $D(p) = p^{n-1} + d_{n-2} p^{n-2} + \ldots + d_0$ и $R(p) = r_{n+\mu-1} p^{n+\mu-1} + r_{n+\mu-2} p^{n+\mu-2} + \ldots + r_0$ являются решением уравнения

$$\delta(p) = \alpha(p)\varphi(p)D(p) + k_\Sigma R(p), \tag{9}$$

а $\delta(p) = p^{2n+\mu-1} + \delta_{2n+\mu-2} p^{2n+\mu-2} + \ldots + \delta_0$ – нормированный полином;

Исходя из требований к контуру компенсации возмущений, было определено, что

$$\delta(p) = \overline{\alpha}(p)\alpha(p),$$

где $\overline{\alpha}(p) = p^{n+1} + \overline{\alpha}_n p^n + \ldots + \overline{\alpha}_0$.

$$R(p) = r_1 p \alpha(p),$$

тогда управление имеет вид:

$$\upsilon(t) = -k_\Sigma r_1 \frac{p\alpha(p)}{\varphi(p)D(p)}.$$

Приведем коэффициенты полиномов $D(p)$ и $R(p)$ для объекта управления 1,2,3,4 и 5-го порядков:

$n = 1$:

$D(p) = 1$; r_1 – любой.

$n = 2$:

$$d_0 = 3\sqrt{3}\omega_0, \quad r_1 = \frac{\bar{\alpha}_1 - \omega_0^2}{k_\Sigma} = \frac{1}{k_\Sigma} 8\omega_0^2.$$

$n = 3$:

$$d_1 = \left(4 + 4\sqrt{2}\right)\omega_0, \qquad d_0 = \left(17 + 12\sqrt{2}\right)\omega_0^2, \quad r_1 = \frac{1}{k_\Sigma}\left(24 + 16\sqrt{2}\right)\omega_0^3.$$

$n = 4$:

$$d_2 = 5\left(5 + 2\sqrt{5}\right)^{1/2}\omega_0, \quad d_1 = \left(49 + 20\sqrt{5}\right)\omega_0^2,$$

$$d_0 = \left(45 + 20\sqrt{5}\right)\left(5 + 2\sqrt{5}\right)^{1/2}\omega_0^3, \qquad r_1 = \frac{1}{k_\Sigma}\left(176 + 80\sqrt{5}\right)\omega_0^4.$$

$n = 5$:

$$d_3 = \left(12 + 6\sqrt{3}\right)\omega_0, \qquad d_2 = \left(104 + 60\sqrt{3}\right)\omega_0^2.$$

$$d_1 = \left(508 + 294\sqrt{3}\right)\omega_0^3, \qquad d_0 = \left(1351 + 780\sqrt{3}\right)\omega_0^4,$$

$$r_1 = \frac{1}{k_\Sigma}\left(1664 + 960\sqrt{3}\right)\omega_0^5.$$

6. Результаты экспериментов

Эксперимент проводился на экспериментальном образце системы активной виброзащиты, собранном в ЦНИИ «Электроприбор».

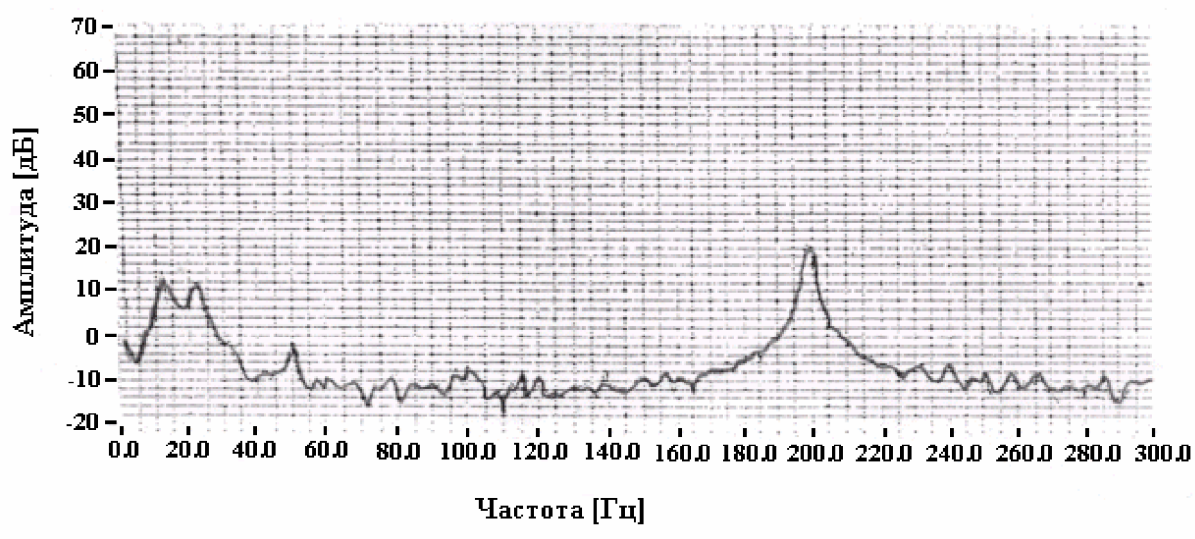


Рис. 9. График АЧХ при выключенной системе

При возбуждении стенда синусоидальным сигналом на частоте 200 Гц были сняты амплитудно-частотные характеристики сигнала с датчиков силы при выключенной и

включенной системе. Уровень вибрационных сил при выключенной системе на частоте 200 Гц имел значение 18,823016 дБ. Соответствующее значение для включенной системы на частоте 200 Гц составило -2,63887 дБ. Уровень подавления вибрационных сил К на частоте 200 Гц составил 21.5 дБ.

Графики АЧХ, полученные при эксперименте, приведены на рис. 9, 10.

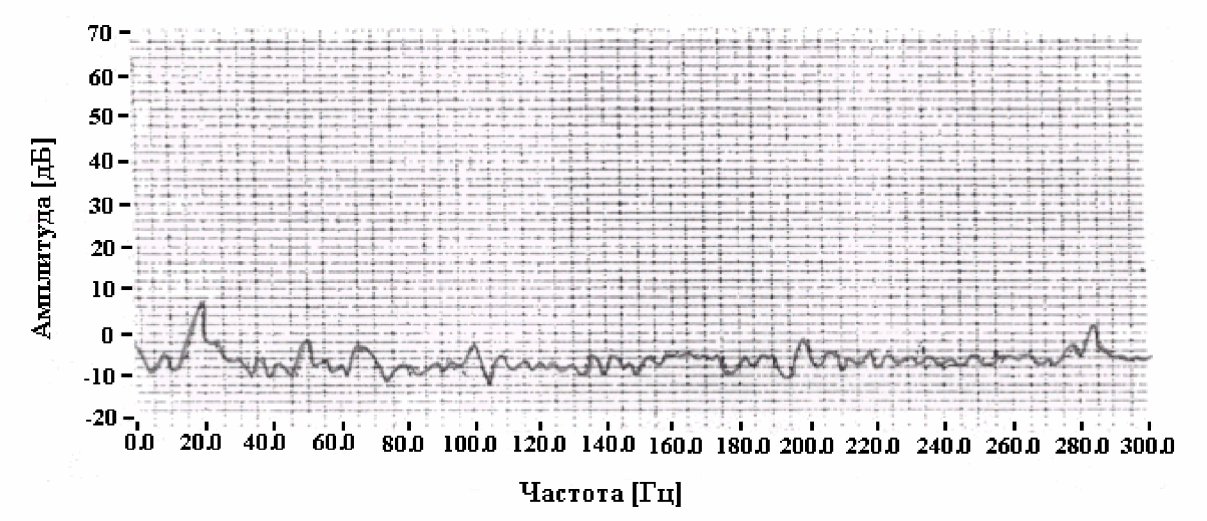


Рис. 10. График АЧХ при включенной системе

Работа выполнена при финансовой поддержке в рамках тематического плана НИР СПбГУИТМО

Литература

1. Johnson C.D. Accommodation of external disturbances in linear regulator and servomechanism problems // IEEE Transactions on Automatic Control, 1971, vol. 16, No. 6.
2. Francis B.A., Wonham W.M. The internal model principle for linear multivariable regulators // Applied Mathematics and Optimization, 1975, No. 2.
3. Davison E.J. The robust control of a servomechanism problem for linear time-invariant multivariable systems \\ IEEE Transactions on Automatic Control, 1976, vol. 21, No. 1.
4. Уонем М. Линейные многомерные системы управления: Геометрический подход. М.: Наука, 1980.
5. Синтез дискретных регуляторов при помощи ЭВМ / В.В. Григорьев, В.Н. Дроздов, В.В. Лаврентьев, А.В. Ушаков – Л.: Машиностроение, 1983.
6. Акунов Т.А., Алишеров С., Оморов Р.О., Ушаков А.В. Матричные уравнения в задачах управления и наблюдения непрерывных объектов. – Бишкек: Илим, 1991.
7. Di Benedetto M.D. Synthesis of an internal model for nonlinear output regulation // International Journal of Control, 1987, vol. 45.
8. Khalil H.K. Robust servomechanism output feedback controller for feedback linearizable systems // Automatica, 1994, vol. 30, No. 10.
9. Никифоров В.О. Нелинейная система управления с компенсацией внешних детерминированных возмущений // Известия РАН. Теория и системы управления, 1997, № 4.
10. Никифоров В.О. Адаптивная стабилизация линейного объекта, подверженного внешним детерминированным возмущениям // Известия РАН. Теория и системы управления, 1997, № 2.

11. Nikiforov V.O. Adaptive non-linear tracking with complete compensation of unknown disturbances // European Journal of Control, vol. 4, No. 2, 1998.
12. Nikiforov V.O. Nonlinear servocompensation of unknown external disturbances // Automatica, 2001, vol. 37.
13. Buchner H.J., Hemami H. Servocompensation of disturbance in robotic systems // International Journal of Control, 1988, vol. 48.
14. Никифоров В.О., Дроздов В.Н. Адаптивное управление мехатронным поворотным столом // Мехатроника, автоматизация и управление, 2002. Часть I – № 4. Часть II – № 5.
15. Дроздов В.Н., Мирошник И.В., Скорубский И.В. Системы автоматического управления с микроЭВМ. – Ленинград: Машиностроение, 1989.
16. Андриевский Б.Р., Фрадков А.Л. Избранные главы теории автоматического управления. – Санкт-Петербург: Наука, 1999.

3 БАЗЫ ДАННЫХ И ИНФОРМАЦИОННЫЕ СИСТЕМЫ

МНОГОКОМПОНЕНТНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА СПБГУ ИТМО

Г.Ю. Громов, Д.А. Дорохин, В.В. Кириллов, А.В. Скатин, В.С. Черемухин

Цель данной статьи – ознакомление с многокомпонентной информационной системой СПБГУ ИТМО, основной ее частью – корпоративной информационной системой университета (ИСУ) и «окном» в ИСУ – корпоративным порталом, который позволяет получить доступ к всевозможным университетским информационным ресурсам.

Введение

В статье рассматривается назначение информационной системы СПбГУ ИТМО, краткая история ее создания и развития. В результате сегодняшняя информационная система построена по принципу многокомпонентности, являющемуся базовым при создании информационных систем нового поколения [1]. Использование этого принципа делает возможным производить поэтапное внедрение компонентов системы. На первом этапе внедрения устанавливаются (или заменяются уже устаревшие) компоненты системы на те рабочие места, которые нуждаются в обновлении программного обеспечения. На втором этапе происходит развитие системы с подсоединением новых компонентов и отработкой межкомпонентных связей. Возможность применения такой методики внедрения обеспечивает ее достаточно простое тиражирование и адаптацию к местным условиям. Таким образом, автоматизированная информационная система нового поколения - это многокомпонентная система с распределенной базой данных.

Назначение информационной системы

Назначение информационной систем – получение эффективного средства информационной поддержки формирования, контроля и реализации государственной политики в сфере образования, что достигается за счет:

- создания единого информационного пространства;
- кардинального сокращения времени, необходимого на прохождение информации, требующейся для принятия управленческих решений;
- введение единого стандарта работы с электронными документами;
- автоматизации и повышения эффективности работы сотрудников и подразделений путем внедрения специализированных приложений и средств поддержки групповой работы;
- создания для руководства системы поддержки принятия решений (СППР).

Краткая история создания системы

Первые работы по автоматизации отдельных задач, выполняемых управленческим персоналом вуза, стали проводиться с начала 70-х годов в рамках программы «Автоматизированная информационная система высшей школы» (АИС ВШ). Создавались файловые программные комплексы, работающие на ЭВМ «Минск», «ЕС ЭВМ» и других универсальных ЭВМ того времени. В ЛИТМО на ЕС ЭВМ стали функционировать подсистемы: «Расчет заработной платы» и «Абитуриент».

В 80-е годы в стране стали появляться первые информационные системы, использующие системы управления базами данных (СУБД): ОКА (IMS), ДИСОД (ADABAS), Oracle 5 и 6. Продолжали развиваться и файловые программные комплексы. В ЛИТМО началась инициативная работа по автоматизации составления рабочих планов и расчету преподавательской нагрузки. Такая подсистема с 1978 года функционировала на Мини-ЭВМ СМ4, а в 1984 году была перенесена на персональную ЭВМ «Искра 226».

В конце 80-х годов в России стали появляться персональные ЭВМ IBM PC и множество СУБД для них (dBase, R:Base, Paradox, FoxPro …). С этого момента начались работы по созданию в вузах различных локальных информационных систем, автоматизирующих деятельность отдела кадров, расчетного отдела, материального отдела, кассы, общей бухгалтерии, деканатов, приемной комиссии, научно-исследовательской части и других подразделений.

В ЛИТМО также стали создаваться подобные системы с использованием СУБД R:Base for DOS, обладающей развитым генератором приложений. В период с 1989 по 1999 год в ЛИТМО на основе этой СУБД были созданы и введены в эксплуатацию локальные информационные системы («Штатные расписания», «Основные средства», «Операции на расчетных счетах», «Главная книга», «Договора НИЧ», «Учебный процесс», «Расчет заработной платы», «Абитуриент»). За несколько лет эксплуатации этих локальных систем в них появилось достаточно много противоречивых данных, например:

- один и тот же сотрудник существует какое-то время в разных системах под разными учетными (табельными) номерами, имеет разные фамилии (например, Зальколина и Золколина) и разные оклады (1395 и 1305);
- не совпадают сведения о численности студентов и сотрудников, поступающие из локальных систем отдела кадров, учебной части, деканатов, кафедр;
- встречаются случаи назначения на стипендию неуспевающих студентов.

Единственным путем для исключения подобных и множества других ошибок был путь объединения локальных систем в интегрированную информационную систему с единой базой данных, в которой будет содержаться непересекающаяся информация из всех локальных систем [2]. Пользовательские приложения бывших локальных систем будут общаться с этой единой базой данных.

Использование всеми приложениями единой базы данных и организация ввода в нее данных, когда новые данные вводятся только один раз через специально предназначенное для этой цели приложение и помещаются в месте, доступном для других приложений, естественно, не исключает появления ошибок. Однако в этом случае правильность данных оценивается многими пользователями всех приложений, и исправлять такую ошибку надо в единственном месте. Другим шагом на пути исключения противоречий является формирование приказов с помощью специального приложения информационной системы, позволяющего подготовить приказ, распечатать его единственный экземпляр, внимательно проверить, утвердить и только после этого дать доступ к данным этого приказа. Такая технология полностью исключает появление ошибок.

Понимая неизбежность перехода от локальных систем к единой интегрированной информационной системе, в 1993 году на кафедре вычислительной техники начались работы по освоению профессиональных СУБД (Ingres, а затем Oracle), их внедрению в учебный процесс и отдел АСУ. Внедрение интегрированной информационной системы с использованием профессиональной СУБД было невозможно из-за недостаточной мощности вычислительных ресурсов на рабочих местах пользователей.

Первые шаги по созданию системы

В 1999 году в Центрально-Европейском университете Будапешта состоялся семинар «Информационные системы в управлении вузом». Участники семинара (среди ко-

торых было много представителей университетов России) были единодушны в том, что разработка интегрированной информационно-аналитической системы управления вузом является одним из приоритетных направлений создания общероссийской университетской корпоративной сети, необходимой для совершенствования системы российского высшего образования.

На семинаре «Информационные интегрированные системы управления вузом» (Санкт-Петербург, 16-18 ноября 1999 г.) был создан консорциум из одиннадцати университетов (руководитель ректор СПбГИТМО(ТУ) В.Н. Васильев), который взялся за разработку и внедрение «Интегрированной информационно-аналитической системы управления вузом» (ИИАС). В объемном документе по концепции создания ИИАС приводились основные принципы построения такой системы и примерное распределение работ на первый этап (январь–июль 2000 года), который финансировался институтом «Открытое общество» (Фонд Сороса).

Сначала решался вопрос о возможности использовании для этой цели одной из промышленных систем управления предприятием. В 1999 году на рынке программных продуктов отсутствовала система управления предприятием, учитывающая специфику работы высшей школы. Участники работы вполне обоснованно решили, что она и не может быть создана без участия вузов. Поэтому стали проводиться исследования по возможностям создания новой системы на основе какой-либо промышленной (БОСС-КОРПОРАЦИЯ, ПАРУС, 1С БУХГАЛТЕРИЯ и т.п.). Однако такие системы являются закрытыми, и их сложно интегрировать с задачами управления учебным процессом, организацией приема в вуз и т.п. Поэтому было принято решение приступить к разработке оригинальной системы.

В результате совместной разработки была создана запланированная часть ИИАС (управление учебным процессом), сконцентрирован опыт одиннадцати вузов по исследованию предметной области, обобщен опыт разных групп разработчиков по подходам к проектированию и использованию инструментария для автоматизации проектирования, опробованы несколько вариантов модели совместного создания большой информационной системы территориально-разобщенными коллективами разработчиков.

Однако были выявлены некоторые сложности совместной разработки:

- различия в организационных структурах университетов и терминологии, что затрудняет процесс создания интегрированной модели и приложений;
- территориальная удаленность (сложности оперативного общения большого коллектива приводят к существенному замедлению продвижения проекта);
- неприспособленность средств разработки и технических характеристик вычислительных сетей для удаленного использования репозитория (централизованного хранилища сведений о составе и деталях совместно создаваемой информационной системы).

К сожалению, руководство Фонда Сороса приостановило финансирование, что привело к замораживанию совместных работ по созданию ИИАС. Однако наш университет (и ряд других вузов) стал независимо развивать свои информационные системы, используя опыт по созданию ИИАС.

С января 2001 года по контракту с Национальным фондом (НФПК) подготовки кадров в ИТМО стала создаваться «Информационная система университета» (ИСУ). В январе 2002 года первая ее часть была введена в эксплуатацию. В течение 2002 года производилась доработка ИСУ, и к декабрю 2002 года были введены в промышленную и (или) опытную эксплуатацию все предусмотренные контрактом подсистемы.

В настоящее время производится модернизация ранее созданных и внедрение новых приложений. Сегодняшний состав основных приложений ИСУ показан в верхней части рис.1 (светло-серый цвет).

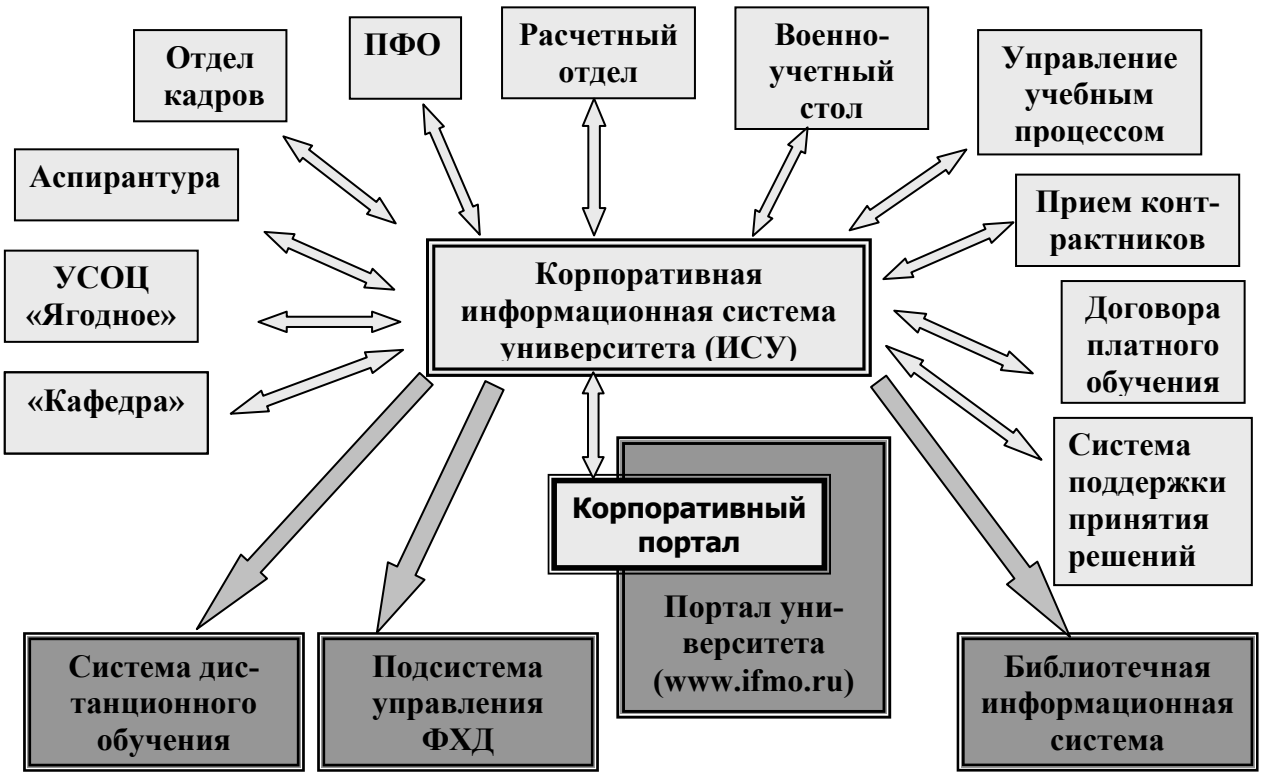


Рис. 1. Многокомпонентная информационная система СПБГУ ИТМО

Переход к многокомпонентной организации информационной системы

Уже в самом начале внедрения ИСУ появилась необходимость в расширении состава решаемых ею крупных задач, не входящих в круг задач по управлению университетом. Первой из них была задача по созданию системы дистанционного обучения, которая должна использовать сведения о структуре университета, студентах и преподавателях, учебных планах и других объектах, содержащихся в ИСУ. Эту систему можно было бы реализовать как приложение ИСУ (использовав единую базу данных), либо создав отдельную систему и организовав непротиворечивый обмен данными между нею и ИСУ.

В связи с тем, что задача, решаемая системой дистанционного обучения, чрезвычайно ресурсоемка, имеет большое число пользователей и должна сопровождаться специальным подразделением, было признано целесообразным создание отдельной системы, которая также будет создана с использованием СУБД Oracle. Так появилось два зависимых компонента распределенной информационной системы СПбГУ ИТМО. Между ними был организован информационный обмен данными с помощью штатных средств СУБД (репликация), которые обеспечивают достоверность этих данных.

В 2003 году университет приобрел «Автоматизированную библиотечную информационную систему» (АБИС), разработанную с использованием СУБД Oracle. Ей также требовались данные о подразделениях, студентах и сотрудниках, т.е. появился третий компонент информационной системы СПбГУ ИТМО (см. рисунок).

Примерно в то же время у студентов, преподавателей и, тем более, руководителей учебных подразделений стала возникать насущная необходимость иметь оперативный доступ к информации из базы данных системы. До появления четвертого компонента – корпоративного Интернет-портала – они должны были «выпрашивать» такую информацию у сотрудников деканатов и административных подразделений, на компьютеры которых установлены те или иные приложения ИСУ.

К сожалению, сотрудники центра информационных систем (ЦИС) уже сегодня с трудом администрируют около 50-ти приложений, установленных на компьютерах различных отделов, расположенных в трех корпусах университета. Поэтому единственным приемлемым разрешением создавшейся ситуации могло стать широкое использование корпоративного портала.

Основная задача корпоративного портала – создание и поддержка единой интегрированной среды для ежедневной работы студентов, аспирантов и сотрудников ГУИТМО со всевозможными университетскими информационными ресурсами (хранилищами данных, информационными системами, разнообразными приложениями и т.п.). Портал предоставляет каждому зарегистрированному сотруднику возможность опубликовать информацию о себе (контактная информация, фото, направления научных исследований, ссылку на свою www-страницу или любую другую информацию), а также обеспечивает возможность персонализации своего рабочего места.

Руководители подразделений могут также опубликовать информацию о своих подразделениях и выложить для определенных пользователей любую документацию.

Заключение

Создание единого информационного пространства университета с помощью построения единой интегрированной информационной системы потребовало больших и неоправданных усилий. Было решено перейти от единой к многокомпонентной информационной системе, в которой полностью обеспечивался основополагающий принцип построения автоматизированных информационных систем - отсутствие дублирования ввода исходных данных (информация, введенная в один из компонентов системы, может быть использована любым другим ее компонентом). Такая структура позволила органически включить в информационную систему компонент для создания хранилища данных, разделяя системы оперативного действия и системы поддержки принятия решения.

Модульность построения системы и принцип одноразового ввода дают возможность гибко варьировать конфигурацией такой системы и поэтапно внедрять разрабатываемые и (или) приобретаемые компоненты.

Литература

1. Власов А.И., Лыткин С.Л., Яковлев В.Л. Краткое практическое руководство разработчика информационных систем на базе СУБД Oracle: Библиотечка журнала «Информационные технологии» М.: изд-во Машиностроение, 2000. 120 с. ил.
2. Дейт К. Введение в системы баз данных. / 7-изд. К.: Издательский дом «Вильямс», 2002. 1072 с.

ИСПОЛЬЗОВАНИЕ АНАЛИТИЧЕСКИХ ФУНКЦИЙ В СУБД ORACLE

Г.Ю. Громов, Д.А. Дорохин

В статье рассматривается использование аналитических функций – новой возможности в СУБД Oracle, позволяющей упростить написание запросов для целого ряда задач и повысить их производительность.

Введение

Аналитические функции, появившиеся в версии Oracle 8.1.6, предназначены для ряда задач, решение которых с использованием стандартных возможностей языка SQL затруднительно, а производительность таких решений крайне низка. Типичными задачами такого класса являются[1]:

- подсчеты промежуточных сумм;
- подсчет процентов в группе;
- подсчет скользящего среднего;
- ранжирующие функции;
- запросы, возвращающие N первых строк.

Использование аналитических функций при выполнении запросов указанных классов позволяет добиться больших преимуществ по сравнению с традиционными решениями. Запросы пишутся в более простой и интуитивно понятной форме, что упрощает восприятие кода и его сопровождения. Вместо выполнения нескольких промежуточных запросов с обработкой их результатов на клиенте или выполнения запроса с вложенными подзапросами и подставляемыми представлениями на сервер отсылается и там выполняется только один запрос, что позволяет уменьшить нагрузку на сеть и обработку результатов на клиенте. Немаловажным является увеличение производительности запросов по сравнению со стандартными решениями, что достигается использованием специальных планов выполнения.

Использование аналитических функций

Синтаксис вызова аналитической функции:
функция (аргумент, аргумент, …)
OVER
(<конструкция фрагментации> <конструкция сортировки> <конструкция окна>)

Рассмотрим более подробно элементы вызова аналитической функции.
Функция – имя аналитической функции (sum, max, min и т.д.).
OVER – ключевое слово, указывающее на вызов аналитической функции.
<конструкция фрагментации> задает условия разбиения результирующего множества на группы (если конструкция фрагментации отсутствует, обрабатывается все результирующее множество). Каждая группа обрабатывается независимо. Стоит отметить, что у каждого вызова в аналитической функции в пределах одного запроса могут быть разные конструкции фрагментации.
<конструкция сортировки> определяет, как упорядочиваются данные в группе при вызове аналитической функции. Необходимость применения данной конструкции зависит от вызываемой аналитической функции.
<конструкция окна> позволяет задать постоянный или переменный набор данных в пределах группы, с которыми будет работать аналитическая функция.

Рассмотрим примеры использования аналитических функций на основе запросов к таблицам стандартной схемы scott:

```
SELECT deptno, ename, sal,
  sum(sal) OVER (PARTITION BY deptno
   ORDER BY ename
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) dept_sal,
  sum(sal) OVER (ORDER BY deptno, ename
   ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) total_sal
FROM emp
ORDER BY deptno, ename;
```

Результаты выполнения запроса:

DEPTNO	ENAME	SAL	DEPT_SAL	TOTAL_SAL
10	CLARK	2450	2450	2450
10	KING	5000	7450	7450
10	MILLER	1300	8750	8750
20	ADAMS	1100	1100	9850
20	FORD	3000	4100	12850
20	JONES	2975	7075	15825
20	SCOTT	3000	10075	18825
20	SMITH	800	10875	19625

Как видно из данного примера, мы получили две нарастающие суммы зарплат. первая вычисляет нарастающую сумму зарплата сотрудников конкретного отдела, вторая – общую зарплату всех сотрудников.

Рассмотрим более подробно элементы приведенного запроса:

PARTITION BY deptno – задает группу для обработки аналитической функцией. В данном случае будут обрабатываться строки, относящиеся к одноименному отделу. Для общей суммы зарплат не задана конструкция фрагментации, поэтому будут обрабатываться все строки.

ORDER BY ename – указывает порядок применения аналитической функции к строкам группы. В примере аналитическая функция будет применяться к строкам отдела, отсортированным по именам сотрудников. Стоит особо отметить, что сортировка вызова аналитической функции влияет только на массив строк группы, который ей обрабатывается, но не задает параметры результатов запроса, которые, в свою очередь, не влияют на формирование массива строк обрабатываемых аналитической функцией.

ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW – задание границ окна. Данное выражение является выражением по умолчанию и указывает на обработку всех предшествующих строк и текущей

Как видно из примера, мы получили простой и наглядный запрос, который без использования аналитических функций можно написать только с помощью сложного запроса, использующего подзапросы или подставляемые представления.

Наиболее сложным для понимания в использовании аналитических функций является задание конструкции окна. Поэтому рассмотрим более подробно различные способы ее определения [2].

UNBOUNDED PRECEDING – окно включает в себя строки группы с первой до текущей включительно.

CURRENT ROW – окно включает только текущую строку.

n PRECEDING – в окно входят n строк до текущей.

n FOLLOWING – в окно входят n строк после текущей.

Рассмотрим применение данных конструкций задания окна на примере:

```
SELECT deptno, ename, sal,
  sum(sal) OVER (ORDER BY deptno, ename
    ROWS UNBOUNDED PRECEDING) sum_1,
```

```
  sum(sal) OVER (ORDER BY deptno, ename
    ROWS CURRENT ROW) sum_2,
      sum(sal) OVER (ORDER BY deptno, ename
    ROWS 2 PRECEDING) sum_3,
  sum(sal) OVER (ORDER BY deptno, ename
    ROWS BETWEEN CURRENT ROW AND 2 FOLLOWING) sum_4
FROM emp
ORDER BY deptno, ename;
```

Результаты выполнения запроса:

DEPTNO	ENAME	SAL	SUM_1	SUM_2	SUM_3	SUM_4
10	CLARK	2450	2450	2450	2450	8750
10	KING	5000	7450	5000	7450	7400
10	MILLER	1300	8750	1300	8750	5400
20	ADAMS	1100	9850	1100	7400	7075
20	FORD	3000	12850	3000	5400	8975
20	JONES	2975	15825	2975	7075	6775
20	SCOTT	3000	18825	3000	8975	5400
20	SMITH	800	19625	800	6775	5250

ROWS UNBOUNDED PRECEDING – позволяет вычислить нарастающую общую сумму зарплат все сотрудников. Обрабатывает строки в группе с первой по текущую.
ROWS CURRENT ROW – обрабатывается только зарплата текущего сотрудника.
ROWS 2 PRECEDING – в окно входит текущая строка и две предшествующих.
ROWS BETWEEN CURRENT ROW AND 2 FOLLOWING – сумма зарплат текущего сотрудника и двух последующих (в таблице приведен сокращенный результат запроса, значения последних строк поля sum_4 основываются на не представленных строках).

Заключение

В статье были рассмотрены различные способы вызова аналитических функций. Наибольшую сложность представляет задание сложных конструкций окна, поэтому данная конструкция была подробно рассмотрена на последнем примере. Еще раз хочется отметить, что использование аналитических функций позволяет существенно упростить восприятие кода, улучшить производительность выполнения запроса, снизить сетевой трафик и перенести всю обработку запроса на сервер.

Литература

1. Кайт Т. Oracle для профессионалов. М.: DiaSoft, 2003.
2. Diana Lorentz, “Oracle9i SQL Reference, Release 2 (9.2)”, Oracle Corporation, 2002

WORKFLOW ДЛЯ ПОРТАЛОВ

А.В. Скатин, В.В. Кириллов

Цель данной статьи – описать четвертое поколение порталов и показать ошибочность утверждения, что для построения порталов четвертого поколения надо пройти все предыдущие.

Введение

Развитие порталов происходило постепенно, по мере осознания их роли в организации, для которой они разрабатывались.

Первое поколение порталов – это контентные порталы, которые являлись усовершенствованием обычных web-страниц путем обеспечения доступа к различным системам. Второе поколение – это порталы доступа к приложениям, что качественно отличалось от первого поколения, так как данные, отображаемые на портале, могли быть результатом действия многих разрозненных приложений, которые при общем отображении давали порой более правильные выводы, чем при анализе последовательном анализе данных из таких приложений. Третье поколение (большинство существующих) порталов – это порталы совместной работы. Это поколение является достаточным для любого предприятия, но снова и снова будет возникать вопрос интеграции с новыми приложениями и системами, который уже стал камнем преткновения для многих сложных проектов в их дальнейшем развитии.

Порталы четвертого поколения – это процессные порталы, для которых основной целью является обеспечение работы с системами workflow (последовательность выполняемых действий, технологический процесс, технология).

Задача данной статьи – показать простоту реализации порталов четвертого поколения и опровергнуть распространенное мнение, что для разработки портала четвертого поколения необходимо пройти через все предыдущие этапы. Это неверно, так как последнее состояние порталов не является развитием первых, а представляет собой качественный скачок.

Чтобы порталы достигли данного состояния, разработчикам (порталов, ERP систем, инструментов для управления контентом и т.п.) следует решить, может быть, более серьезную проблему, которая определит дальнейшее развитие порталов и информационных систем в целом – это консолидация различных портальных инфраструктур (portal frameworks).

Консолидация портальных инфраструктур

Корень объединения портальных инфраструктур кроется в сервисно-ориентированной архитектуре (Service Oriented Architecture, SOA). На практике SOA представляет собой наличие разделяемого сервиса, который может вызываться различными системами для его инкапсуляции в процессы данных систем.

Можно предположить, что такого рода архитектура должна быть универсальной, без привязки к конкретному способу доступа и обработки данных, что в настоящее время находит применение в web-сервисах. Данная технология уже имеет некоторые реализации, такие как управления контентом с помощью UDDI (Universal Description, Discovery and Integration), развитие которого получило название «шина корпоративных сервисов» (Enterprise Services bus, EBS). Схематично взаимодействие систем по средствам EBS представлено на рис. 1.

Модель EBS рассматривается не как взаимодействие по средствам сети, а как собственно сеть взаимодействующих сервисов, что принципиально отлично от типичного понимания распределенных компонентов. Одно из революционных решений этой мо-

дели – то, что она позволяет ввести лишний, дополнительный, заранее не известный шаг обработки данных, при этом не трогая приложения. Его реализация заключается в разработке сервиса и простой переборке шины для направления потока данных через новый сервис. Схематически роль EBS в SOA представлена на рис. 2.

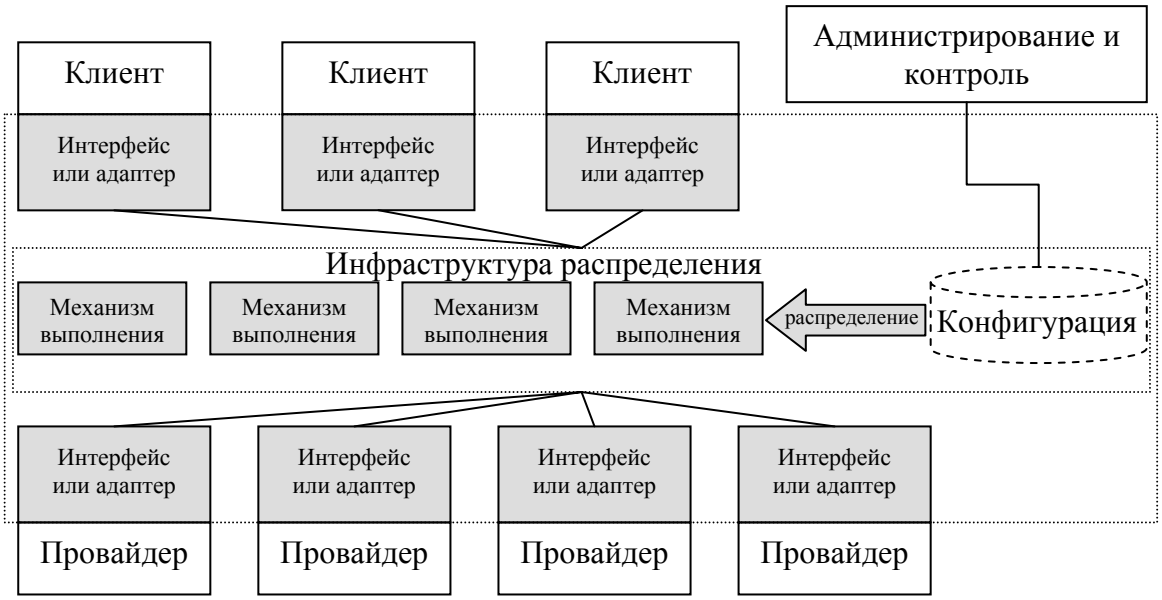


Рис. 1. Центральное управление через EBS

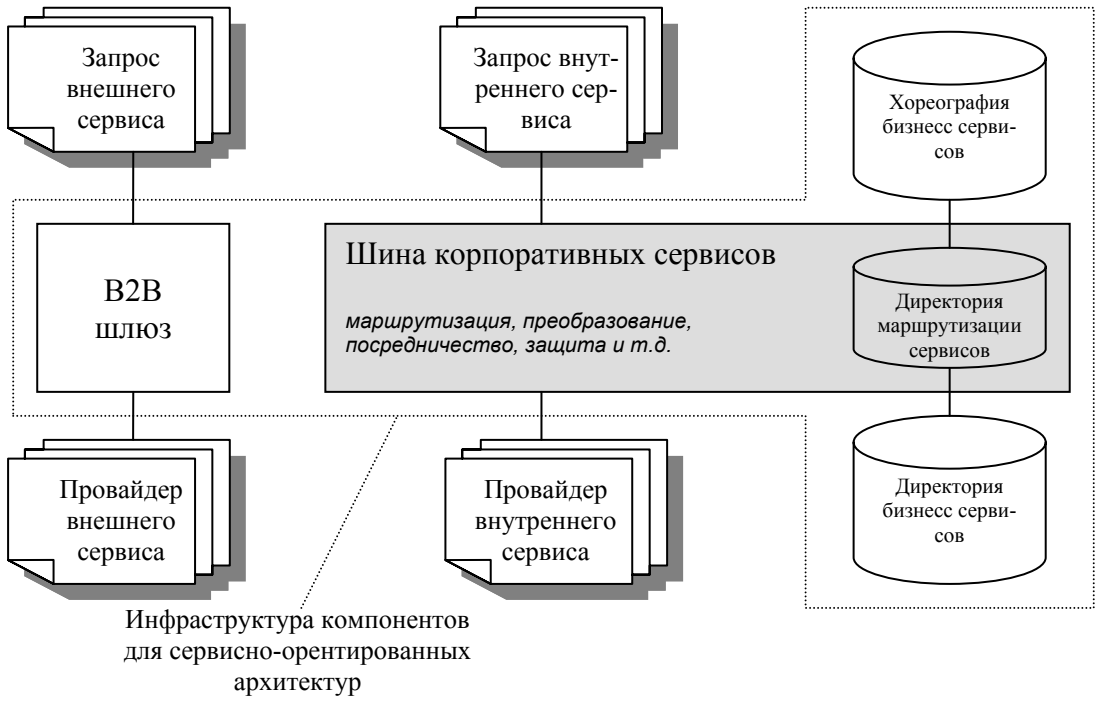


Рис. 2. Роль EBS в SOA

В модели web-сервисов кроется опасность, которая подкупает своей простотой. Около двух лет назад, осознав, что SOA системы являются следующим этапом развития информационных систем в целом, фирмы-разработчики систем разного рода реализовали большое количество специфических механизмов взаимодействия сервисов, что приводило к замедлению развития систем на реальных предприятиях (или, что более

часто, к останову развития предприятия). С принятиями двух стандартов JSR 227 (Java Specification Requests 227) и WSRP (Web Services for Remote Portlets) ситуация несколько изменилась, и стала ясна общая цель разработчиков информационных систем. Данные стандарты не нацелены на что-либо конкретное и описывают взаимодействие с абстрактным ресурсом.

Workflow для порталов

Workflow – это информационная среда, которая позволяет перейти к процессно-ориентированной структуре управления предприятием с помощью создания инфраструктуры, обеспечивающей исполнение формализованного бизнес-процесса. Такая процессно-ориентированная структура позволяет получить общую картину в организации. Преимущества процессно-ориентированной структуры:

- сокращение издержек при передаче задания от одного структурного подразделения к другому;
- заинтересованность каждого сотрудника в конечном, с точки зрения всего предприятия, результате;
- быстрая реакция предприятия на изменения рыночной ситуации;
- оценка деятельности структурной единицы в контексте бизнес-процесса.

Наиболее эффективно применение workflow в среде порталов в следующих случаях.

- появление новых логических связей или усиление существующих, которые являются необходимыми для предприятия, что достигается за счет интеграции различных приложений в одно место;
- единая регистрация и, как следствие, доступ к различной информации из разного рода приложений, что ранее требовало несопоставимо большего времени.

Сервис и процесс

Развитие информационных систем определено по направлению SOA. При этом деятельность человека представляет собой участие в различных процессах. Таким образом, дальнейшее развитие информационных систем должно привести к тому, что сервис или группу сервисов возможно будет приравнивать к процессу. Следовательно, для того, чтобы формализовать работу какого либо предприятия, организации следует определить процессы и соответствующие сервисы, так в общем случае и поступают при постановке задачи. Совокупность SOA а EBS, а также WSRP и JSR 227 позволяет, не отступая от постановки задачи, реализовать ее именно в том виде, в котором определили ранее. После этого следует провести исследование (поиск по UDDI), в результате которого выяснить те подзадачи (сервисы), которые уже реализованы кем-то (например, курсы валют, которые в большинстве случаев берутся с МВБ), и описать в своем реестре. Связь между постановкой задачи и ее реализацией представлена в таблице 1.

	Постановка задачи	Реализация задачи
1	Формирование общей задачи	Технологии при реализации задачи SOA, EBS, WSRP, JSR и т.п.
2	Разбиение общей задачи на подзадачи	Сервисы
3	Определение связей подзадач, для выполнения общей задачи	EBS

Таблица 1. Сравнение постановки и реализации задачи

При постановке задачи 2 и 3 пункты могут дополняться и изменяться, но более опасно на настоящий момент для большинства систем не просто добавление новой подзадачи, а ее изменение. А технология EBS, как было отмечено ранее, изначально подразумевает такой ход событий. Таким образом, даже на этапе реализации задачи возможно изменение не только задач, но и связей между них.

Гибкое изменение системы заложено в самой модели SOA, и, следовательно, предприятия, обладающие такой информационной системой, могут безболезненно и с предвиденным результатом менять подзадачи и связи между ними. Более того, гибкость системы будет предполагать развитие систем предприятия.

Данный принцип существования системы ранее не был заложен ни в одну из технологий построения информационных систем. Кроме того, мнение о том, что система, построенная по модульному принципу, расширяема, дополняема и изменяема, не является для модели SOA опровержением ее шага вперед в этом направлении, так как SOA существует как изменяемая система.

Качественное отличие построения порталов четвертого поколения от первых трех состоит в том, что оно основывается на процессах, существует вместе с процессами и без них немыслимо.

Вывод

Практический вывод данной статьи может быть следующим. Не следует проходить через все этапы построения порталов для построения порталов четвертого поколения по следующим причинам:

- построение портала четвертого поколения отображает саму задачу в силу наличия большой описательной части на этапе реализации;
- логика построения порталов как SOA системы отлична от предыдущих, что может только затруднить переход к SOA порталу при развитии из первых поколений;
- построение SOA портала понятнее в силу своей простоты или привычности решения проблем организации процесса

Литература

1. Архитектура WWW - http://www.w3.org/2001/tag/webarch/
2. Раздел WSDL W3C консорциума – http://www.w3.org/TR/wsdl
3. Раздел архитектуры web-сервисов W3C консорциума – http://www.w3.org/TR/ws-arch
4. Раздел SOAP W3C консорциума – http://www.w3.org/TR/soap
5. Сайт Web-сервисы и сервисно-ориентированная архитектура - http://www.service-architecture.com/

ОЦЕНКА ОБЪЕМОВ МНОГОМЕРНЫХ КУБОВ В OLAP СИСТЕМАХ

А.К. Дорожкин

Статья посвящена рассмотрению вопросов, связанных с оценкой объемов многомерных кубов, являющихся основой OLAP систем, в зависимости от их структуры и характера данных, лежащих в их основе. Помимо этого, в статье представлен способ расчета объема многомерного куба независимо от реализации OLAP сервера.

Введение

Специфика практически всех систем поддержки принятия решений заключается в хранении и обработке огромного объема данных, на основе которого аналитик принимает то или иное решение. Размеры хранилищ данных достигают размеров в несколько терабайт, и, если лидеры на рынке реляционных баз данных еще в состоянии поддерживать такие объемы информации, то для OLAP серверов подобные объемы данных на сегодняшний день являются недостижимыми. Поэтому встает необходимость в прогнозировании объема многомерной базы данных.

Еще из школьного курса геометрии известно, что объем многомерного куба определяется как произведение длин его ребер. В многомерных базах данных аналогом длины ребра является количество членов измерения. Однако в OLAP системах не все так просто, как в геометрии. Из теории многомерных баз данных известно, что объем многомерного куба, помимо числа членов измерений, еще зависит от двух составляющих. Это плотные и разреженные измерения, а также хранение агрегированных показателей. Чем выше степень разреженности куба, тем сильнее оказывают влияние на его объем характер исходных данных, лежащих в основе куба.

Целью данной работы является создание аналитической модели, позволяющей прогнозировать объем многомерного куба. Ввиду того, что модель не имеет привязки к конкретной реализации сервера многомерной базы данных, для определения объема многомерного куба мы не будем учитывать объемы, занимаемые метаданными и зарезервированным пространством, а также различные механизмы оптимизации хранимых данных, такие как, например, сжатие хранимых данных, используемые в Cognos PowerPlay и Microsoft Analysis Services. Поэтому размер многомерного куба мы будем определять по числу заполненных ячеек.

Для оценки объема многомерного куба мы будем использовать понятие размера многомерного куба. Чтобы различать эти понятия, определим объем как пространство, физически занимаемое на жестком диске или любом другом носителе тем или иным объектом. А под размером будем понимать значение, вычисляемое на основе логических параметров объекта, таких как количество записей таблице, количество измерений куба, размер поля данных и т.д.

Агрегация

С понятием агрегация в многомерных базах данных очень тесно связано понятие «взрыв» базы данных. Данный процесс заключается в том, что количество ячеек агрегированных данных больше, чем детальных, к тому же агрегированные данные оказываются в несколько раз плотнее, чем детальные. Поэтому зачастую при загрузке 10–50 Мб детальных данных результирующий объем многомерной базы данных получается около 1 Гб. В своем стремлении заранее рассчитать все возможные значения разработчики многомерных баз данных иногда забывают, что в очень большой многомерной базе данных потери времени на операциях ввода/вывода могут превосходить затраты времени на расчет многих значений «на лету». Поэтому рекомендуется [3] заранее рас-

считывать одну треть необходимой информации, оставляя остальные показатели для расчета «на лету». В данной работе нас больше интересуют вопросы, связанные не с проектированием многомерных баз данных, а с количественной оценкой «взрыва» базы данных, т.е. определение количества новых ячеек многомерного куба, которые появятся в результате выполнения агрегации.

Прежде чем приступить к рассмотрению вопросов, связанных с агрегацией, введем понятие плотности детальных данных, которое определим как отношение числа непустых ячеек на последних уровнях иерархий к произведению числа членов на этих уровнях:

$$\rho_{\det} = \frac{N_{\det}}{\prod_{i=1}^{n} d_{i,ll}}, \qquad (1)$$

где $d_{i,\mathrm{ll}}$ – число членов на последнем уровне иерархии i-ого измерения, N_{det} – число заполненных ячеек детальных данных (количество загруженных фактов).

В случае, когда плотность детальных данных равна 1, количество ячеек агрегированных данных определяется как разность между количеством ячеек всего куба и количеством ячеек детальных данных:

$$N_{agg} = N_{cube} - N_{\det} = \prod_{i=1}^{n} d_i - \prod_{i=1}^{n} d_{i,ll}. \qquad (2)$$

В [4] описаны и протестированы различные алгоритмы оценки объема многомерных кубов, однако они связаны либо с подсчетом вероятностей, либо с оценкой объема многомерного куба на основе выборки, что не совсем подходит для данной работы, так как здесь необходима аналитическая зависимость. Поэтому требуется найти такое подмножество данных, к которому можно эффективно приложить понятие плотности. Минимальной единицей агрегации является ячейка агрегации, под которой будем понимать набор ячеек, принадлежащих j-ому уровню иерархии и лежащих на пересечении конкретных членов измерений, по крайней мере, одно из которых принадлежит j-1 (более высокому) уровню иерархии.

Для наглядности на рис. 1 представлен двумерный вариант процесса агрегации.

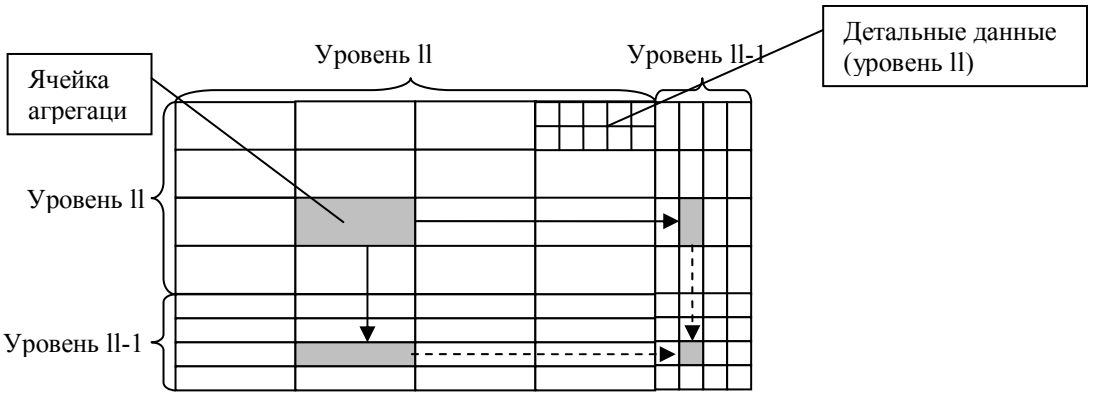


Рис. 1. Агрегация

На рис. 1 пунктирной линией изображены альтернативные варианты агрегации. Процесс агрегирования может происходить по одному из этих двух направлений. Как видно из рис. 1, одна ячейка агрегации порождает n!+1 ячеек агрегации на предыдущих уровнях иерархий (n – число измерений), т.е. для двумерного варианта – 3, для трехмерного – 7 и т.д. Легко понять, что максимальное число агрегатов будет достигнуто, если порождаемые ячейки будут также заполнены полностью. В этом случае ячейка аг-

регации считается полностью заполненной, т.е. любое увеличение ячеек в рамках данной ячейки агрегации не приведет к увеличению числа агрегатов. Количество ячеек на последующем уровне иерархии, которое формирует одна ячейка агрегации, можно выразить следующим выражением:

$$N_l^{ext} = \sum_{i=1}^{n}\left(\prod_{j=1}^{n}\left(d_{j,l+1} + \mathrm{d}_{j,l}\right) - \prod_{j=1}^{n} d_{j,l+1}\right). \quad (3)$$

Однако определение общего количества ячеек агрегированных данных на основании понятия ячейки агрегации является достаточно трудоемким и ненаглядным, особенно в условиях разреженных данных. Для определения числа агрегатов в случае неполного заполнения куба, т.е. зависимости числа агрегатов от плотности детальных данных, будем использовать понятие кубоида, определенное в [5] как подмножество членов измерений куба, содержащее агрегированные значения. В данной работе под кубоидом мы будем понимать набор ячеек, принадлежащих определенному набору $\{lc1, lc2, ..., lcn\}$ уровней всех измерений. Легко понять, что количество кубоидов многомерного куба равно

$$N_{cuboid} = \prod_{i=1}^{n} l_i \quad . \quad (4)$$

В общем случае количество агрегированных данных можно рассчитать как произведение плотности данных в каждом кубоиде на его «геометрический объем» (максимально возможное количество ячеек):

$$N_{agg} = \sum_{i=1}^{N_{cuboid}-1} \rho_i * S_i \quad , \quad (5)$$

где значение -1 отвечает за исключение детальных данных из списка кубоидов, так как они не являются агрегированными данными, ρ_i – плотность данных в *i*-ом кубоиде, *Si* – максимально возможное количество ячеек в *i*-ом кубоиде (или его «геометрический объем»), которое определяется как

$$S_{j\in\{lc1,lc2,...,lcn\}} = \prod_{i=1}^{n} d_{i,lci} \quad . \quad (6)$$

Выражение (6) определяет количество ячеек, принадлежащих кубоиду, лежащему на пересечении *lc*1, *lc*2, ..., *lcn* уровней всех иерархий.

Для окончательного вывода зависимости между плотностью детальных данных и количеством агрегатов необходимо вывести закономерность между ρ_i и ρ. Как известно из [3], агрегированные данные значительно плотнее, чем детальные, поэтому в качестве оператора перехода от ρ_i к ρ можно использовать отношение количества ячеек кубоидов:

$$\frac{\rho}{\rho_{lc1,lc2,...,lcn}} \sim \frac{S_{lc1,lc2,...,lcn}}{S_{\det}} \quad , \quad (7)$$

где $\rho_{lc1,lc2,...,lcn}$ – текущие уровни соответствующих измерений, а $S_{lc1,\,lc2,\,...,\,lcn}$ – количество ячеек данного кубоида. Однако эта зависимость носит не линейный, а обратно экспоненциальный характер, так как плотность кубоида не может быть больше 1 (количество заполненных ячеек не может превысить количество ячеек вообще). Как показывают эксперименты (см. следующий раздел), зависимость плотности кубоида от плотности детальных данных носит следующих характер:

$$\rho_{lc1,lc2,...,lcn} \approx 1 - e^{-\rho\frac{S_{\det}}{S_{lc1,lc2,...,lcn}}} \quad . \quad (8)$$

Таким образом, объединив выражения (5) и (8), получим число агрегированных ячеек в зависимости от плотности детальных данных:

$$N_{agg} \approx \sum_{i=1}^{N_{cuboid}-1} S_i * \left(1 - e^{-\rho \frac{S_{\det}}{S_{lc1,lc2,\ldots,lcn}}}\right). \quad (9)$$

Проверка модели

Для подтверждения адекватности разработанной модели был проведен ряд экспериментов и на основе полученных результатов выполнен расчет основных показателей. Для более объективного рассмотрения изложенного выше теоретического материала в качестве серверов многомерных баз данных использовались три популярных продукта от лидеров на рынке программного обеспечения. Это Analysis Services от компании Microsoft, PowerPlay от компании Cognos и Express Server от компании Oracle. Распределение членов по уровням измерений представлено в табл. 1.

Уровень\Измерение	Dim1	Dim2	Dim3	Dim4
1	1	1	1	1
2	10	10	10	10
3	20	100		
4	100			
5	1000			

Таблица 1. Распределение членов по измерениям

Таким образом, общее число ячеек детальных данных составило 10 млн., всего ячеек в кубе – 15,2 млн. В качестве источника данных для многомерных кубов использовалась база данных Oracle со схемой «звезда», в которую загружались данные, индексируемые по измерениям случайным образом. Были выполнены измерения объемов многомерных баз данных, получаемые после выполнения агрегации, а так же с помощью программы на языке Oracle Express подсчитаны количество заполненных ячеек во всех кубоидах многомерного куба, в том числе и детальных данных, а также плотности кубоидов, на основании числа заполненных ячеек и распределения членов измерений по уровням иерархий.

Следует отметить два момента. В-первых, так как для всех трех многомерных баз данных использовался один источник данных, то распределение плотностей кубоидов в базах данных Microsoft Analysis Services и Cognos PowerPlay было точно таким же, как и в Oracle Express, поэтому повторного подсчета заполненных ячеек не требовалось. Второй момент касается заполнения многомерного куба случайным индексированием по измерениям. Так, например, в таблицу фактов хранилища данных загружается 50 000 записей с индексацией по измерениям случайным образом. Однако это совсем не значит, что многомерный куб будет содержат все 50 000 записей, так как некоторые комбинации внешних ключей таблиц-размерностей будут повторяться и, соответственно, несколько записей таблицы фактов попадут в одну ячейку.

Согласно выражению (4), число кубоидов данного многомерного куба равняется 2*2*3*5=60. Мы не будем приводить здесь всю таблицу плотностей кубоидов полученного многомерного куба, ввиду ее большого размера, а приведем только основные характеристики эксперимента, т.е. зависимости общего количества ячеек и объемов многомерных баз данных (см. табл. 2), а также плотности некоторых кубоидов от плотности детальных данных (см. табл. 3).

Для наглядности результаты представлены также в графическом виде (рис. 2).

Число		Плотность детальных данных	Всего ячеек многомерного куба	Объем многомерной базы данных (МБ)		
Записей таблицы фактов	Ячеек детальных данных			Oracle Expres	MS SQL	Cognos
1000	1000	0.0001	38786	7.58	0.24	0.34
10000	9996	0.0009996	249367	28.58	0.55	0.59
50000	49878	0.0049878	789525	64.66	0.84	1.22
100000	99503	0.0099503	1254353	87.58	1.05	1.97
200000	198046	0.0198046	1931621	110.08	1.46	3.53
500000	487725	0.0487725	3229843	122.66	2.67	8.25
1000000	951527	0.0951527	4669180	123.58	4.33	4.34
5000000	3935587	0.3935587	9098883	123.23	19.19	14.75
10000000	6321590	0.632159	11511878	123.58	37.66	22.84

Таблица 2 Результаты экспериментов по агрегации

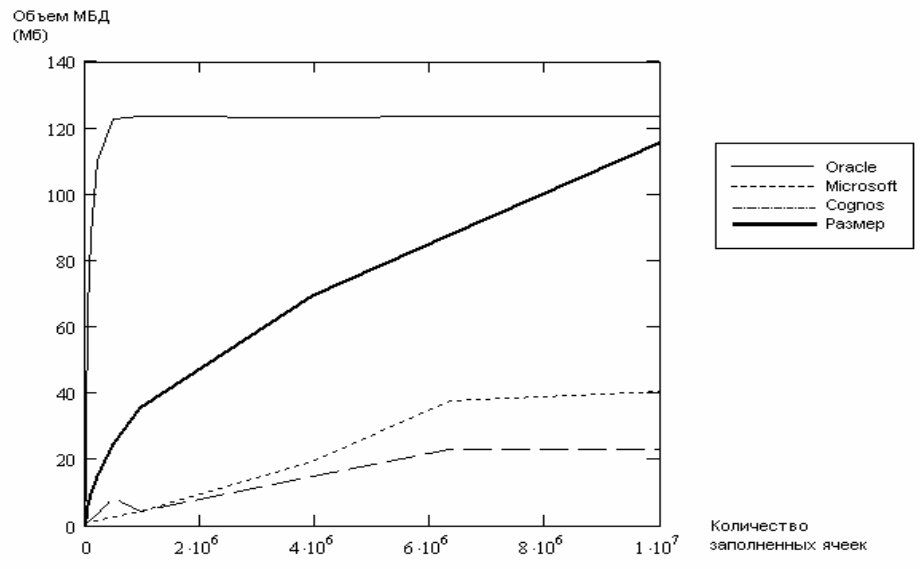


Рис. 2. Объем данных многомерной базы данных от числа детальных данных

Как видно из табл. 2, объем данных Oracle Express примерно в три раза превышает объем МБД от Microsoft и Cognos. В первую очередь это зависит от механизмов оптимизации хранения данных, так как многомерные базы данных были заполнены одним и тем же набором данных, и размер типа показателя во всех базах данных составлял 8 байт. Многомерные кубы в рамках данного эксперимента заполнялись единицами, и продукты от Microsoft и Cognos, применив механизмы оптимизации хранения данных, добились более компактного представления данных. Однако, как хорошо видно из рис. 2, объем многомерной базы Oracle стабилизировался при значении числа записей в таблице фактов 500 тысяч, в то время как объемы данных для продуктов Microsoft и Cognos продолжали линейно расти. Это связано с механизмом управления плотными измерениями, которые в схеме многомерной базы данных для Oracle соответствовали Dim3 и Dim4.

Значение Full на рис. 2 соответствует случаю, когда плотность детальных данных равна 100%, так как из табл. 2 видно, что даже при случае 10 млн. записей в таблице фактов, что соответствует максимально возможному числу ячеек детальных данных, плотность детальных данных составляет всего лишь 63% из-за повторений записей в

случайном наборе данных. График, обозначенный толстой черной линией, соответствует размеру данных, который определяется как произведение числа заполненных ячеек на размер типа данных показателя (8 байт для всех баз данных).

Отношение размера данных и фактического объема многомерной базы данных представлено на рис. 3.

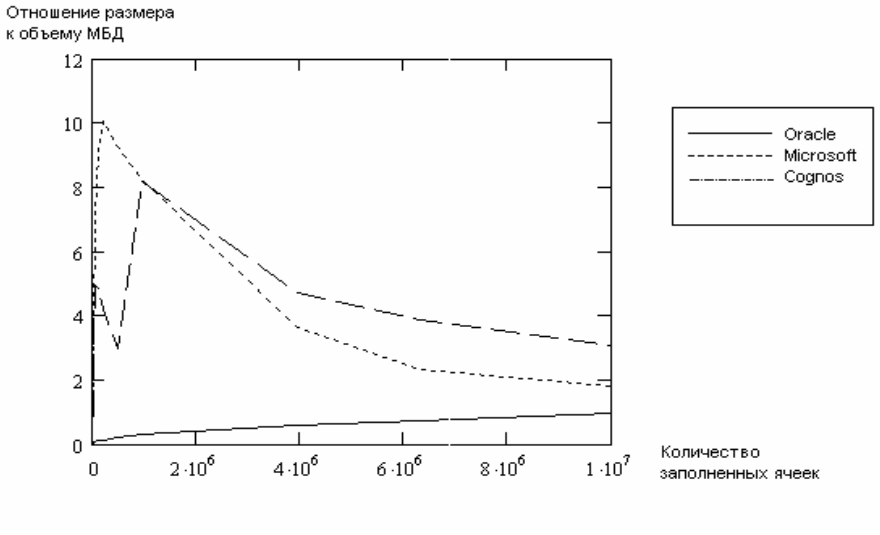


Рис. 3. Отношение размера МБД и ее реального объема

Как видно из рис. 3, отношения объема многомерной базы данных и ее размера трех серверов многомерных баз данных сходятся с ростом числа загруженных фактов.

Рассмотрим плотности кубоидов в зависимости от плотности детальных данных. В таблице 3 индексы кубоидов соответствуют номеру измерения. Как уже было отмечено выше, в таблице 3 представлены далеко не все из 60 имеющихся кубоидов, так как только кубоиды с номерами 23* и далее имеют плотность меньше 100% практически для всех значений плотности детальных данных. Для наглядности результаты представлены в графическом виде (рис. 4).

Плотность детальных данных	2321	3322	3312	4222	5321
0.0001	0.0944	0.00499	0.04875	0.00994	0.001
0.0009996	0.6346	0.048785	0.3941	0.09494	0.009954
0.0049878	0.9927	0.221335	0.9195	0.39385	0.048821
0.0099503	1	0.39353	0.9936	0.6313	0.095138
0.0198046	1	0.63162	0.99995	0.86407	0.181343
0.0487725	1	0.918245	1	0.993	0.393158
0.0951527	1	0.99335	1	0.99995	0.631938
0.3935587	1	1	1	1	0.993109
0.632159	1	1	1	1	0.999947

Таблица 3. Плотности кубоидов

Как видно из рис. 4, характер роста плотности кубоидов близок к $1-\frac{1}{e^x}$ и определяется согласно выражению (8). В табл. 4 приведены результаты расчета плотности выбранных ранее кубоидов, а в табл. 5 – выраженные в процентах погрешности расчетов по сравнению с реальными данными.

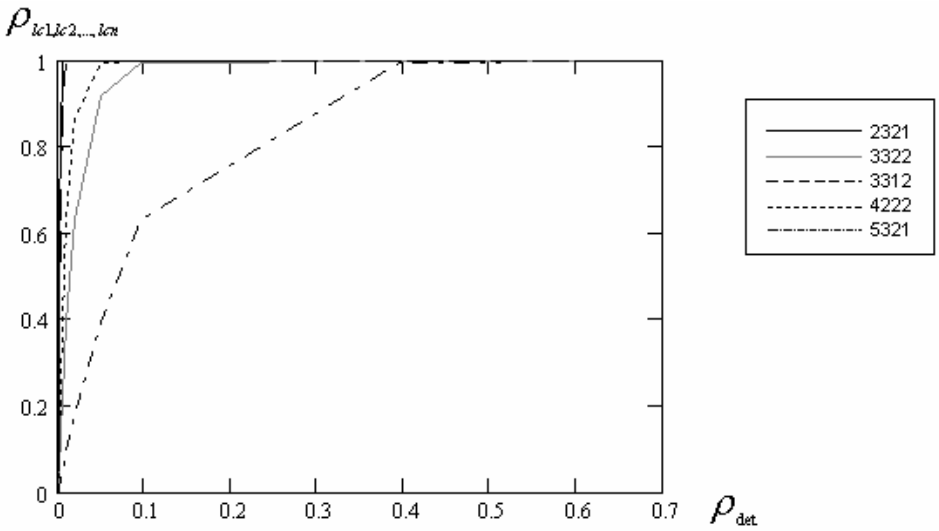


Рис. 4. Плотности кубоидов в зависимости от плотности детальных данных

Плотность детальных данных	**2321**	**3322**	**3312**	**4222**	**5321**
0.0001	**0.0952**	**0.00499**	**0.0488**	**0.00995**	**0.00099**
0.0009996	**0.632**	**0.04875**	**0.3933**	**0.09513**	**0.00995**
0.0049878	**0.9937**	**0.221**	**0.9174**	**0.39273**	**0.0486**
0.0099503	**0.99995**	**0.392**	**0.9931**	**0.63029**	**0.0947**
0.0198046	**0.99999**	**0.628**	**0.99995**	**0.86199**	**0.1797**
0.0487725	**1**	**0.913**	**1**	**0.99238**	**0.3859**
0.0951527	**1**	**0.991**	**1**	**0.99993**	**0.6138**
0.3935587	**1**	**0.9999**	**1**	**1**	**0.9805**
0.632159	**1**	**1**	**1**	**1**	**0.99826**

Таблица 4. Рассчитанные плотности кубоидов

Плотность детальных данных	**2321**	**3322**	**3312**	**4222**	**5321**
0.0001	**0.81**	**0.049**	**0.042**	**0.102**	**0.0499**
0.0009996	**0.41**	**0.068**	**0.191**	**0.196**	**0.0783**
0.0049878	**0.048**	**0.276**	**0.227**	**0.285**	**0.341**
0.0099503	**0.004**	**0.398**	**0.051**	**0.1604**	**0.447**
0.0198046	**0**	**0.493**	**0**	**0.2402**	**0.924**
0.0487725	**0**	**0.602**	**0**	**0.062**	**1.826**
0.0951527	**0**	**0.195**	**0**	**0.002**	**2.862**
0.3935587	**0**	**0**	**0**	**0**	**1.273**
0.632159	**0**	**0**	**0**	**0**	**0.174**

Таблица 5. Погрешность расчетов плотности кубоидов

Как видно из табл. 5, погрешность рассчитанных значений незначительна и не превышает 3%, что говорит об адекватности полученных аналитических зависимостей.

Однако в табл. 5 заметно некоторое увеличение значений погрешности для крайнего правого столбца. Это связано с тем, что данный столбец отвечает за кубоиды с относительно большим «геометрическим» объемом. Следовательно, чем больше «геометрический» объем кубоида, тем больше погрешность расчета его плотности.

Исходя из рассчитанных значений плотностей кубоидов и их «геометрических» размеров, можно рассчитать значение размера многомерного куба по формуле (9), не прибегая к подсчету всего числа заполненных ячеек, как это было при формировании табл. 2. Таким образом, можно спрогнозировать приблизительный объем многомерного куба. Результаты данного расчета представлены в табл. 6.

Как видно из таблицы 6, ошибка не превышает 2%, что говорит о хорошей точности метода расчета.

Плотность детальных данных	Расчетный размер (Мб)	Реальный размер (МБ)	Ошибка (%)
0.0001	0.297	0.296	0.31
0.0009996	1.901	1.903	0.08
0.0049878	6.009	6.024	0.25
0.0099503	9.544	9.57	0.27
0.0198046	14.67	14.74	0.47
0.0487725	24.40	24.64	0.97
0.0951527	35.062	35.62	1,57
0.3935587	69.03	69.42	0.56
0.632159	87.77	87.83	0.061

Таблица 6. Расчетное значение размера

Заключение

В статье были рассмотрены основные вопросы, связанные с оценкой объемов многомерных кубов в зависимости от их структуры и характера данных, лежащих в их основе, а также предложен метод расчета количества заполненных ячеек агрегированных данных на основании плотности детальных данных. Предложенный метод, в отличие от ранее известных, требует меньше вычислительных затрат, при достаточной точности результатов, что подтверждено проведенными экспериментами.

Литература

1. Oracle Express: Database design and performance guide [Электронный ресурс]. Oracle Corp. Режим доступа http://otn.oracle.com – 2001.
2. D. Stark, S. Humphrey Data Blocks – The Key to Optimizing Hyperion [Электронный ресурс]. Mountain View, CA: Analysis Team, Inc. Режим доступа http://www.analysisteam.com – 2003.
3. N. Pendse, Database explosion [Электронный ресурс]. London: Optima Publishing, Режим доступа http://www.olapreport.com – 2005
4. A. Shukla, P. M. Deshpande, J. F. Naughton, and K. Ramasamy, Storage estimation for multidimensional aggregates in the presence of hierarchies // Proceedings of 22nd Very Large Dabases. Bombay, India: Mumbai, 1996. P. 522–531.
5. Y. Feng, D. Agrawal, Range CUBE: Efficient Cube Computation by Exploiting Data Correlation // 20th International Conference on Data Engineering . Boston: IEEE Computer Society, 2003. P. 658–671.

МОДЕЛЬ РАСПРЕДЕЛЕННОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ НА СЕТИ ПЕТРИ

С.К. Дорожкин

В статье рассматриваются вопросы построения моделей распределенных вычислительных систем с помощью раскрашенных или цветных сетей Петри. В работе приводится пример построения модели распределенной вычислительной системы.

В настоящее время наблюдается рост числа проектов по разработке распределенных и параллельных систем для многих прикладных областей – от больших масштабируемых систем в области телекоммуникации до систем среднего и начального уровня в области web-приложений. Разработка вычислительных систем этого класса является сложным процессом. И наиболее эффективным подходом к проектированию такого рода систем является моделирование.

Цветные или *раскрашенные* сети Петри (Colored Petri Nets) обеспечивают среду для создания, тестирования и анализа моделей распределенных вычислительных систем. Модель системы, построенная на цветной сети Петри, описывает состояния, в которых может находиться моделируемая система, и переходы между этими состояниями. Эти состояния системы в терминах сетей Петри принято называть *местами*. Сети Петри применяются для моделирования различных вычислительных систем и процессов, начиная от телекоммуникационных протоколов и системного программного обеспечения и заканчивая моделированием аппаратуры больших вычислительных систем и встроенных управляющих систем специального назначения.

Разработка сетей Петри была инициирована потребностью в создании промышленного языка моделирования, имеющего сильную теоретическую базу и являющегося достаточно гибким для использования на практике при разработке промышленных вычислительных систем. Для достижения этой цели и были созданы *цветные* или *раскрашенные* сети Петри. Они сочетают в себе свойства сетей Петри по моделированию параллельных и синхронных процессов, а примитивы языка программирования позволяют использовать данные разных типов, которые называются, в терминах сети Петри, *цветными наборами* или *мультимножествами*.

Модели, построенные на цветных сетях Петри, могут быть построены из составных модулей, которые могут представлять собой также сети Петри. Это особенно важно, когда моделируются большие промышленные системы, к которым можно отнести распределенные вычислительные системы. Модульная концепция цветных сетей Петри основывается на иерархическом механизме, который поддерживает разработку модели вычислительной системы как снизу вверх, так и сверху вниз. Новые модели могут быть созданы из уже готовых модулей, что позволяет значительно упростить процесс построения модели вычислительной модели, а также одни и те же модули могут использоваться для моделирования различных узлов вычислительной системы, выполняющих одинаковые функции.

Модели вычислительных систем, построенные на сетях Петри – исполняемые, т.е. они позволяют исследовать поведение вычислительной системы, симулируя процессы, которые протекают в реальной вычислительной системе. Симуляция работы вычислительной системы на модели используется для исследования поведения системы в той или иной ситуации, для проверки корректности функционирования вычислительной системы или для исследования ее производительности.

Время выполнения задачи или время ответа играет серьезную роль в работе распределенной вычислительной системы. Корректное функционирование большинства систем зависит от времени, затрачиваемого на выполнения определенных действий. Различные варианты реализации вычислительной системы могут серьезно влиять на время выполнения одних и тех же операций. Моделирование призвано помочь решить подобную задачу. В цветных сетях Петри имеется возможность моделировать время,

затраченное на выполнение операций в системе. Такие модели позволяют анализировать производительность вычислительных систем.

В качестве примера рассмотрим модель распределенной вычислительной системы, обеспечивающей работу с распределенной базой данных. Моделируемая вычислительная система включает в себя несколько серверов. Каждый сервер хранит свою копию базы данных, которая управляется локальным менеджером базы данных. Модель, построенная с помощью цветной сети Петри, показывает процесс синхронизации контента базы данных в случае изменения данных на одном из вычислительных узлов. Таким образом, имеется набор менеджеров:

$DBM=\{d1, d2, ..., dn\}$.

Каждый менеджер может вносить изменения в свою локальную копию базы, а затем должен послать сообщение с требованием провести соответствующие обновления всем остальным менеджерам. В данном примере мы не будем рассматривать содержание сообщения с требованием изменить содержание базы, нас будет интересовать только его заголовок, который описывает *отправителя* (Sender) и *получателя* (Receiver). Таким образом, у нас есть набор сообщений:

$$MS = \{(s,r) \mid s,r \in DBM \wedge s \neq r\},$$

где отправитель s и получатель r должны находится на различных узлах вычислительной системы. Когда менеджер s делает обновление в своей локальной базе, он обязан послать всем остальным менеджерам следующее сообщение:

$$M(s) = \sum_{r \in DBM - \{s\}} 1'(s,r),$$

где сумма обозначает формирование набора из n - 1 элементов сложением мультимножеств $1'\{s,r\} \mid r \in DBM - \{s\}$, каждое из которых содержит один элемент. Результирующий набор содержит по одному сообщению для каждого получателя r, имеющего в заголовке s в качестве отправителя. Все вместе представленные выше определения формируют следующую систему:

constants: n : integer (* $n \geq 3$ *);

colour sets: $DBM = \{d1, d2, ..., dn\}$;

$MS = \{(s,r) \mid s,r \in DBM \wedge s \neq r\}$

E = {e};

functions: $M(s) = \sum_{r \in DBM - \{s\}} 1'(s,r)$

variables: $s, r : DBM$;

Графическое изображение модели вычислительной системы представлено на рис. 1.

Каждый менеджер базы данных может находиться в трех различных состояниях: *Inactive (Неактивен), Waiting (Ожидает)* и *Performing (Выполняет* обновление*)*. В состоянии *Waiting* менеджер ожидает запрос с подтверждением о том, что его запрос на обновлением удаленной копии базы выполнен другим менеджером. В состоянии *Performing* менеджер производит обновление базы по запросу удаленного менеджера.

Каждое сообщение в модели может иметь следующие состояния: *Unused, Send, Received, Acknowledged,* а сама система может находится в состоянии *Active (Активна)* и *Passive (Пассивна).*

Изначально все менеджеры находятся в состоянии *Inactive*, а все сообщения в *Unused*. Такое состояние определяется инициализационным выражением. *DBM.all*() формирует набор, который содержит только один маркер каждого типа (цвета) в наборе *DBM*. Аналогично *MES.all*() формирует набор, который содержит только один маркер каждого типа (цвета) в наборе *MS.*

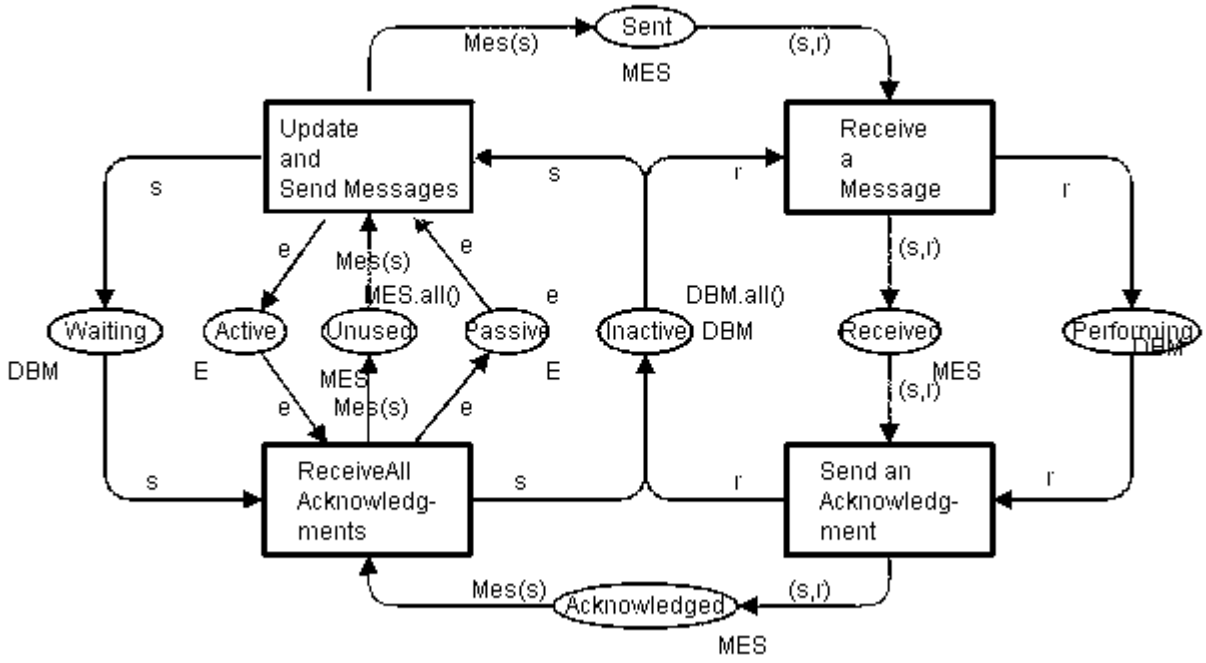


Рис. 1. Графическое изображение модели распределенной вычислительной системы

Когда менеджер s проводит обновление (состояние *Update*) базы, он должен отправить об этом сообщение всем остальным менеджерам. После обновления локальной копии данных состояние менеджера меняется с *Inactive* на *Waiting*, а состояние сообщения меняется с *Unused* на *Sent*. После этого менеджер ожидает, пока все остальные менеджеры пришлют ему оповещение о том, что они произвели обновление своих локальных копий базы данных.

Когда один из этих удаленных менеджеров *r* получает сообщение, он меняет свое состояние с *Inactive* на *Performing*, т.е. производит обновление, а состояние соответствующего сообщения от менеджера *s* меняется с *Sent* на *Received*. После этого менеджер *r* должен отправить уведомление *Acknowledgment*, подтверждая таким образом, что он закончил обновление своей копии данных, а его состояние при этом меняется с *Performing* на *Inactive*. Состояние сообщения меняется с *Received* на *Acknowledged*. Когда все сообщения *M*(*s*), которые были посланы менеджером *s*, получат статус *Acknowledged*, менеджер *s* меняет свое состояние с *Waiting* на *Inactive*, так как это означает, что все удаленные менеджеры произвели обновление базы. Состояние сообщения *M*(*s*) меняется с *Acknowledged* обратно на *Unused*.

Чтобы обеспечить актуальность различных копий на узлах распределенной вычислительной системы, такая простая схема синхронизации позволяет произвести только одно обновление в один момент времени. Другими словами, когда менеджер инициировал обновление, оно должно быть выполнено на всех локальных копиях базы данных, и только после этого новое обновление базы может быть инициировано менеджером. Для этого используется состояние *Passive*. В нем может находиться только один маркер, что соответствует тому, что только один менеджер может послать запрос на обновление базы данных остальным.

Эта модель демонстрирует основные свойства сети Петри: параллельность, конфликтность, зависимость по условию.

В первоначальном состоянии модели распределенной вычислительной системы переход *Update and Send Message* открыт для всех менеджеров, но сработать он может только для одного менеджера базы данных (из-за единственного маркера в месте *Passive*). Такая ситуация называется конфликтом, из-за того что связанные элементы

доступны поодиночке, но одновременно не разрешены. Можно сказать, что переход *Update and Send Message* конфликтен по отношению к себе.

Когда переход *Update and Send Message* выполняется для некоторого менеджера *s*, то переход *Receive a Message* в этот момент параллельно доступен для всех менеджеров, отличных от *s*. Такая ситуация называется параллельностью, и можно сказать что переход *Receive a Message* параллелен сам себе.

Переход *Receive all Acknowledgments* разрешен только, тогда когда переход *Update and Send Message* выполняется для какого-то определенного менеджера *s*, а переходы *Receive a Message* и *Send an Acknowledgment* выполнились для всех менеджеров, отличных от *s*. Такая ситуация называется зависимостью по условию (потому что есть элемент, который может сработать только после срабатывания другого элемента).

Симуляция работы вычислительной системы используется с той же целью, что и тестирование программного обеспечения. К сожалению, получить данные о порядке функционирования вычислительной системы, достаточные для ее анализа, с помощью симуляции можно только для достаточно простой вычислительной системы. Для сложных систем симуляция не даст необходимой информации о порядке функционирования вычислительной системы или отдельных ее узлов, поэтому для получения более детальной информации о вычислительной системе используется метод анализа состояний системы.

При анализе состояний моделируемой системы используется так называемый граф состояний. На нем отражаются все возможные состояния моделируемой системы и переходы между ними. Таким образом, появляется возможность проверить корректность функционирования моделируемой системы. Пример графа состояний для рассматриваемой модели приведен на рис. 2.

Узлами на данном графе являются состояния, принимаемые системой в процессе ее функционирования, т.е. все возможные разметки сети Петри, а дуги этого графа отмечают переходы из одного состояния в другое с помощью связующих элементов.

Недостатком такого метода исследования поведения системы является то, что граф состояний будет сильно усложняться с ростом количества узлов, на которых будут работать локальные менеджеры базы данных. Сравнительные данные приведены в таблице.

Приведенные в таблице данные показывают, что в общем случае граф состояний цветной сети Петри растет очень быстро с увеличением количества цветов (различных типов данных) сети Петри. Однако на практике достаточно рассмотреть модель с небольшим объемом различных цветов, чтобы установить корректность работы моделируемой системы. В данном случае достаточно проверить правильность функционирования распределенной базы данных на наборе из 3 или 4 менеджеров (которые кодируются разными цветами). В случае корректной работы с таким количеством менеджеров можно быть уверенными в том, что система будет работать корректно и с большим числом узлов.

К сожалению, такой подход неприменим при исследовании производительности моделируемой системы. В этом случае необходимо вводить параметры, отражающие время обработки запросов в узлах системы. Время в сетях Петри вводится посредством так называемых глобальных часов. Время может отсчитываться дискретно или постоянно. Чтобы ввести время в модель, необходимо к каждому маркеру добавить переменную времени или, как она еще называется, временную метку. Временная метка обозначает момент модельного времени, после наступления которого, маркер может быть использован, то есть, перенесен из одного места в другое.

Выполнение модели, содержащей временные элементы, похоже на выполнение моделей в других языках с дискретным временем, содержащих очереди событий. Система находится в одном состоянии до тех пор, пока есть маркеры, разрешающие переход в данный момент времени. Когда маркеры, содержащие переменную времени со временем исполнения, равным текущему, кончаются, система увеличивает системное

время на величину временной переменной с минимальным значением. Каждая разметка сети существует в ограниченном интервале модельного времени.

Введем для перехода *Update and Send Message* переменную @+5. Это значит, что выполнение операции *Update and Send Message* занимает 5 единиц модельного времени, и, следовательно, переменные времени в маркерах места *Passive* получают значение +5 единиц времени после срабатывания перехода *Update and Send Message*.

Менеджеры базы данных DBM	Узлы	Дуги
2	7	8
3	28	42
4	109	224
5	406	1,090
6	4,459	4,872
7	5,104	20,426
8	17,497	81,664
9	59,050	314,946
10	196,831	1,181,000
15	71,744,536	669,615,690
20	23,245,229,341	249,439,571,680

Табл. Рост числа элементов графа состояний, в зависимости от числа цветов сети

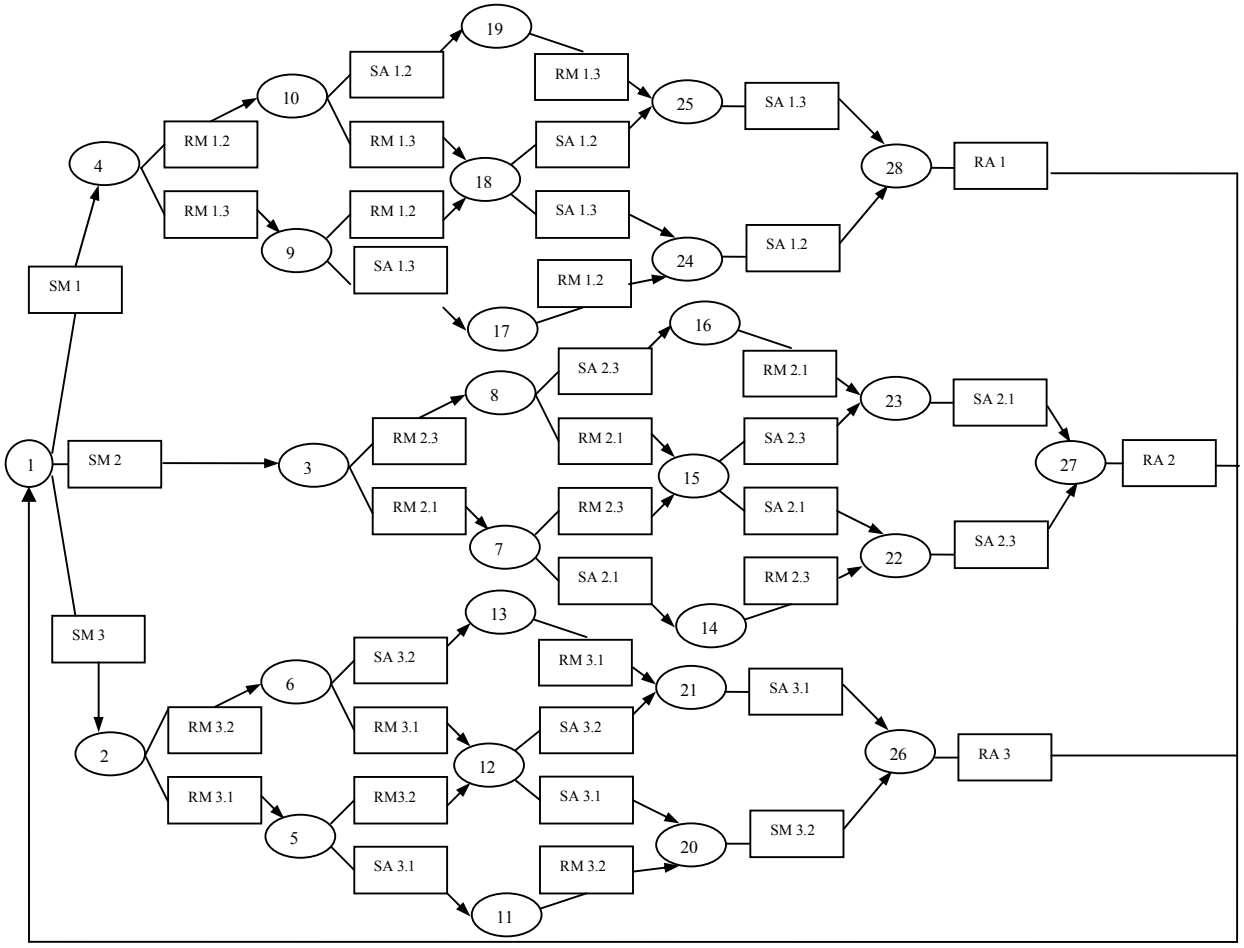


Рис. 2. Граф состояний моделируемой системы

Использование переменных времени в моделях позволяет оценить загрузку от-

дельных узлов вычислительной системы и производительной вычислительной системы в целом.

Модель распределенной вычислительной системы, построенная с помощью сети Петри, является эффективным инструментом, используемым при проектировании вычислительной системы. Аппарат сетей Петри позволяет получить абстрактное описание процессов функционирования вычислительной системы. Результаты моделирования позволяют получить информацию о порядке функционирования как всей вычислительной системы, так и отдельных ее узлов и оценить ее производительность.

В данной работе рассмотрены основные вопросы построения моделей распределенных вычислительных систем, с помощью цветных сетей Петри. В работе приведен пример построения модели распределенной вычислительной системы.

Литература

1. Котов В.Е. Сети Петри. М.: Наука, 1984.
2. Harper Robert Programming in Standard ML. Carnegie Mellon University Spring Semester 2001. Working draft. November, 2004.
3. Kurt Jensen Introduction to the practical use of coloured Petri nets. Springer-Verlag. 1998.
4. Kurt Jensen, Lars M. Kristensen. The practitioner’s guide to coloured Petri nets. Springer-Verlag, 1998.

ОСОБЕННОСТИ ПРЕПОДАВАНИЯ RATIONAL UNIFIED PROCESS В СОСТАВЕ ДИСЦИПЛИНЫ «ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ»

А.Е. Харитонова

Рассматриваются проблемы, связанные с изучением технологии RUP в рамках дисциплины «Проектирование информационных систем», и предлагаются различные способы их решения.

Дисциплина «Проектирование информационных систем» (ПИС) играет важную роль в подготовке специалистов по направлению «Информатика и вычислительная техника». В рамках данной дисциплины изучаются различные методы проектирования ПО, такие как каскадный метод, метод потоков данных, XP (eXtreme Programming, экстремальное программирование) и т.п.[1]. Отведенное на курс время ранее не позволяло давать важный материал в достаточном объеме. Фактически времени хватало только на общий обзор существующих подходов. Из всех упоминаемых подходов лишь каскадный метод рассматривался более подробно, чем остальные. В связи с тем, что продолжительность курса увеличилась вдвое, появилась возможность уделить какому-либо методу больше внимания. Таким методом был выбран более современный Rational Unified Proceess (RUP, унифицированный процесс Rational) [2, 3], который ранее не входил в состав указанной дисциплины.

Выбор RUP обусловлен наличием у него ряда свойств:

- это более современный подход к проектированию информационных систем, в отличие от изучавшихся ранее;
- он объединяет в себе достаточно большой набор традиционных методов;
- он включает в себя принципы управления процессом разработки;
- он обладает понятной структурой;
- он хорошо формализован.

В RUP следует выделить три особенности:

- это итеративный, архитектурно-центричный, основанный на прецедентах использования подход к разработке программного обеспечения;
- это четко определенная и структурированная технология программной инженерии;
- это использование технологических продуктов для программной инженерии.

В дисциплине «ПИС» в той или иной мере изучаются все эти особенности. Первый – подход к проектированию – подробно освящается на лекциях. Хотя RUP адаптирован под разработку информационных систем, предлагаемый им подход к проектированию и управлению процессом разработки может успешно применяться и в других областях. RUP направлен на минимизацию рисков, при этом под рисками понимаются любые угрозы, которые могут помешать успешному завершению проекта.

Большую роль в понимании итеративности, одного из основных свойств RUP, играет сравнение RUP с каскадным подходом. Каскадный подход дает хорошие результаты, когда имеется исчерпывающее представление о решаемой проблеме, причем это представление в процессе разработки не меняется. Как показано на рис.1, в жизненном цикле программного продукта при использовании каскадного подхода неэффективно снижаются риски на ранних этапах разработки. Это связано с тем, что тестирование начинается на последних этапах проекта, где цена устранения дефектов гораздо выше, чем при их раннем обнаружении и устранении. Кроме того, при использовании модели жизненного цикла по каскадному принципу велика вероятность срыва сроков сдачи продукта.

При итеративной разработке проект разбивается на итерации (рис. 2), каждая из которых аналогична полному мини-каскадному циклу. Хотя каждая отдельная итерация может казаться последовательной, как истинный мини-каскад, реальная последовательность может быть нелинейной. Например, можно работать с требованиями, затем заняться анализом и проектированием, затем снова работать с требованиями, затем тес-

тировать, затем снова анализировать и разрабатывать в той последовательности, которая требуется для выполнения работы.

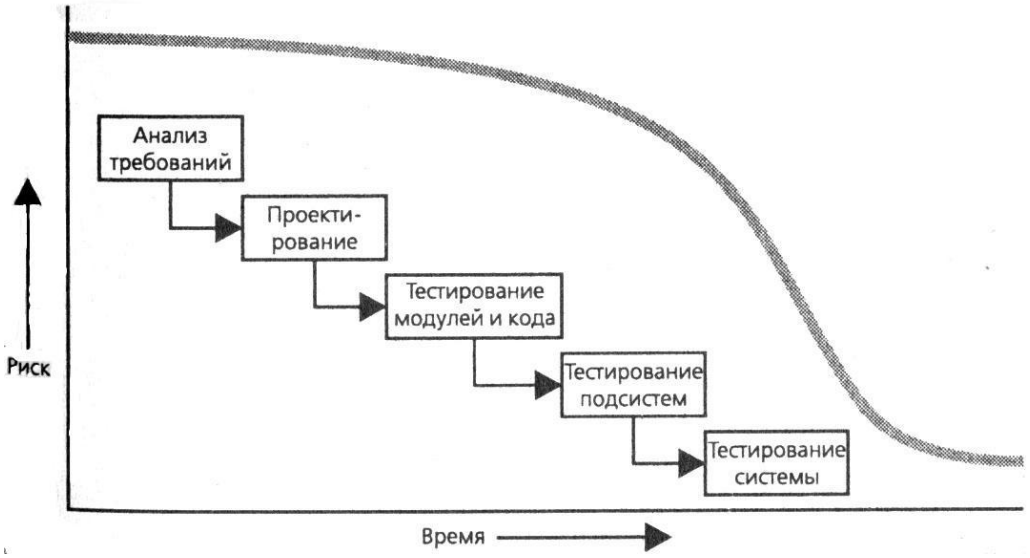


Рис. 1. Жизненный цикл продукта при использовании каскадного подхода

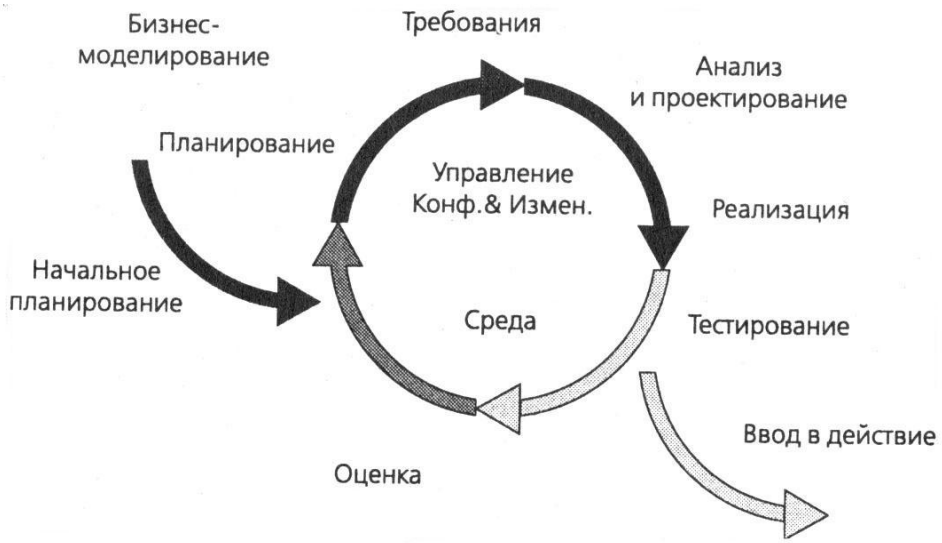


Рис. 2. Итерации RUP

Каждая итерация имеет свою цель и ограничена во времени. Если работа над какой-то функциональной возможностью не выполняется в срок, то она должна быть перенесена в одну из следующих итераций. Нельзя увеличивать время, выделенное на итерацию. Свойство ограничения времени в итеративной разработке и регулярная корректировка плана проекта на каждой итерации позволяют более эффективно реагировать на изменения, чем при использовании каскадного подхода (рис. 3).

Одна из причин, по которой итеративный подход к разработке информационных систем заменил каскадный, заключается в том, что он снижает риски на более ранних этапах разработки. Это повышает вероятность того, что проект завершится удачно. Использование эволюционирующего прототипа или, иначе, исполняемой архитектуры проектируемой системы существенным образом снижает риски. Кроме того, это позволяет практически в любой момент времени после создания такого прототипа иметь работающую систему с ограниченной функциональностью.

Методология RUP базируется на двух составляющих – динамической и статической (рис. 4). Динамическая составляющая представляет собой развитие во времени

процесса проектирования, который состоит из 4 фаз. Она показывает, как процесс, выраженный в форме циклов, фаз, итераций и вех, развертывается в ходе жизненного цикла проекта. Статическая структура показывает, как элементы процесса проектирования – роли, задачи, артефакты, дисциплины – логически группируются в главные дисциплины процесса.

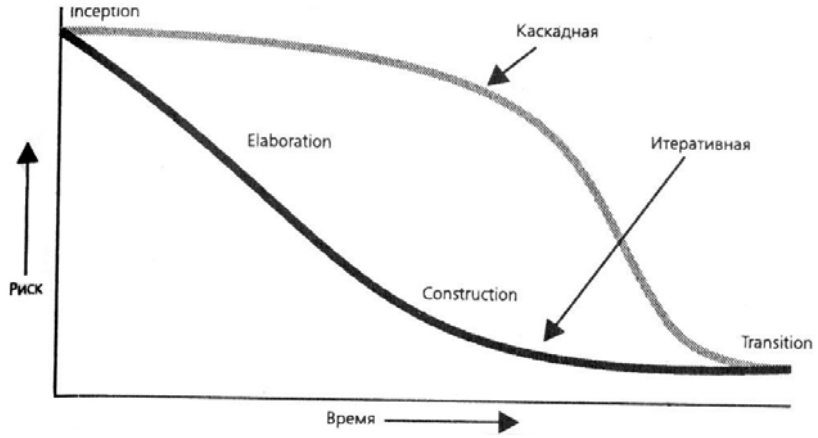


Рис. 3. Графики жизненного цикла продукта для каскадного и итеративного подходов

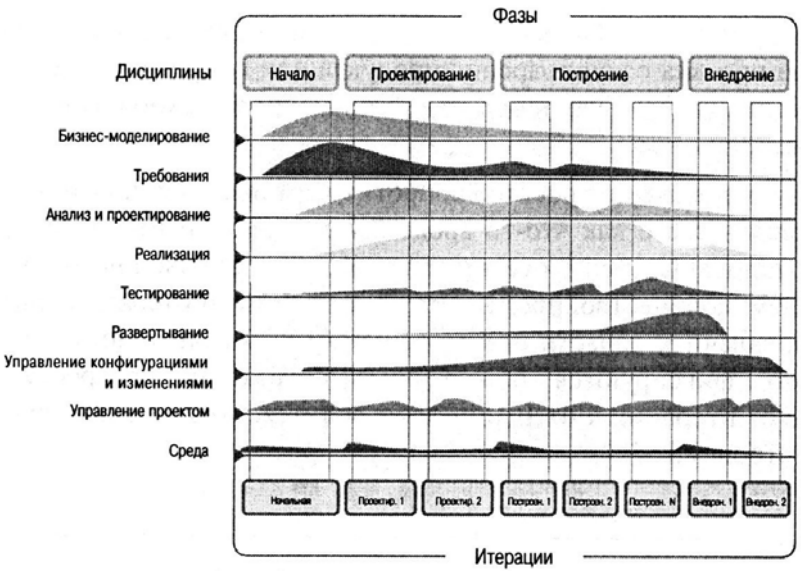


Рис. 4. Общая архитектура RUP

Процесс проектирования указывает, к*то* и *что* делает, *как* и *когда.* Модельные элементы процесса RUP называются, соответственно, ролями, задачами, артефактами и рабочими процессами.

Роль показывает, как человек или группа должны работать, указывает области ответственности и компетенции, которыми должен обладать носитель данной роли.

Задача – это единица работы, выполнение которой можно потребовать от конкретной роли, а артефакт фиксирует её результаты.

Рабочий процесс является способом описания последовательностей задач, создающих какой-либо полезный артефакт и описывающий взаимодействия между ролями. Кроме того, в RUP входят не только методики проектирования, но и принципы управления процессом разработки. Они являются неотъемлемой и уникальной частью каждого проекта.

Упомянутые выше модельные элементы и структура RUP (рис. 4) рассматриваются преимущественно на лекциях. В общем случае опытный менеджер проекта на основе этих данных уже может строить процесс разработки ПО, но для студентов требуется дополнительная информация и практика.

Второй особенность RUP, а именно технологическая, преимущественно изучается на практических занятиях. Одной из основных составляющих RUP является правило, что не следует выполнять те действия в разработке, которые не нужны для снижения рисков. При этом возникает противоречие с общими принципами преподавания, когда нужно дать материал наиболее полно, т.е. надо сделать максимум работ, входящих в процесс проектирования, в том числе и избыточных. Кроме того, описание деталей и особенностей RUP выглядит довольно однообразно и занимает очень большой объем, что затрудняет подачу материала. В связи с вышесказанным было принято решение использовать предлагаемые в составе RUP шаблоны документов, создаваемых при проектировании.

На практических занятиях студенты создают информационную систему. Это происходит в виде своеобразной деловой игры. Главной особенностью такой информационной системы является то, что, как правило, используется фантастическая предметная область. Для этого существует ряд причин:

- границы создаваемой системы полностью определяются студентом и преподавателем, что позволяет надеяться на полную реализацию данного проекта;
- в обязательно порядке происходит бизнес-моделирование. Это обеспечивает четкое понимание того, что должно быть сделано. Как показывает практика, при реализации системы с реальной предметной областью этот момент часто игнорируется, и созданная система оказывается непригодной к использованию, так как непонятно, для чего эта система создавалась и какие задачи решает;
- при работе с реальной предметной областью потребуется консультация эксперта, тогда как при фантастической предметной области учащийся сам выступает в роли такого эксперта;
- процесс создания фантастической системы получается живым и увлекательным.

Контроль над учебным процессов в ходе практических занятий осуществляется путем проверки так называемых артефактов. В терминологии RUP артефакт представляет собой некоторый носитель информации, создаваемой, изменяемой или используемой процессом. Артефакты – это материальные элементы проекта, то, что процесс проектирования создает и использует при движении к своему завершению. Они могут иметь разные формы:

- модель;
- элемент модели;
- документ;
- исходный код;
- исполняемые файлы.

RUP предлагает ряд артефактов, которые в каком-либо виде должны быть представлены при разработке проекта информационной системы. Это артефакты: Glossary (глоссарий), Vision (Представление), Risk List (список рисков), Software Development Plan (план разработки программного обеспечения), Development Case (прецедент разработки), Use-Case Model (модель прецедентов использования), Prototypes (прототипы), Software Architecture Document (документ архитектуры ПО), Design Model (модель проекта) и т.д. [4].

Указанные артефакты являются обязательными этапами сдачи на практических занятиях. Кроме того, в зависимости от содержания последних, преподавателем назначаются дополнительные артефакты, даже если они и не требуются по ходу проекта. Дополнительными являются Target Organization Assessment (оценка целевой организации), Business Rules (бизнес-правила), Business Use-Case Model (бизнес-модель прецедентов использования), Measurement plan (план измерений), Product Acceptance plan

(план приемки продукта) и т.д. [4] Большинство артефактов представляют собой текстовые документы и имеют шаблоны, заполняя которые, можно получить представление о содержимом данного артефакта и основных принципах составления такого рода документов, что помогает студентам определить, какие пункты им следует заполнять применительно к разработке конкретной системы.

Основной задачей, которая ставилась перед студентами в первом семестре, являлось понять наиболее важные компоненты рассматриваемой ими предметной области, проще говоря, ответить на вопрос: «Какую проблему в данной предметной области будет решать проектируемая система?». В рамках практических занятий с некоторыми проектами возникали сложности: рассматриваемая фантастическая предметная область без проектируемой системы существовать просто не может. Для такой предметной области было сложно смоделировать бизнес-процессы без участия проектируемой системы.

Примерно на одном уровне по степени важности стояли задачи определения конкретных ключевых требований к системе и разработки базовой архитектуры.

Главной целью второго семестра и всего курса практических занятий стоит разработка и реализация готового программного продукта. Требования к конечному продукту максимально приближены к требованиям, предъявляемым к реальному коммерческому продукту. Поэтому целесообразно будет пройти весь цикл производства ПО, соответствующий коммерческому изделию. Таким образом, в соответствии со сроками выпусков и итераций, указанных в Software Development Plan (план разработки программного обеспечения), происходит поэтапная сдача проекта.

Первый этап сдачи проекта – демонстрация исполняемой архитектуры. В терминологии RUP исполняемая архитектура обозначает частичную реализацию системы, создаваемую для того, чтобы продемонстрировать, что архитектура может обеспечить ключевые функции. Кроме того, исполняемая архитектура должна демонстрировать требуемые нефункциональные показатели, такие как производительность, надежность, масштабируемость и прочие. Исполняемая архитектура уже является материалом для тестирования, что позволяет строить полнофункциональную систему на стабильной основе, не опасаясь большого количества переделок. Требование, предъявляемое к исполняемой архитектуре студенческого проекта, заключается в том, что с функциональной точки зрения получившийся прототип должен правильно выполнять как минимум одно логическое действие из определенных в требованиях к системе.

Далее происходит сдача промежуточных сборок продукта, обладающих всё большей функциональностью. В зависимости от запланированной сложности проектируемой системы в фазе «построение» итераций, как правило, не менее двух, а в ряде случаев и до 4-х. Очередной важной вехой является демонстрация бета-версии системы. В данном случае бета-версия является завершающей для фазы «построение». В составе фазы «Внедрение» планируется доводка бета-версии до конечного продукта и подготовка различной документации к проекту. Конечная сдача проектируемой системы будет проходить как презентация коммерческого продукта с анализом процесса разработки.

Выше было описано изучение двух особенностей RUP. Изучение третьей особенности – RUP как технологический продукт для разработки ПО – оставлено на усмотрение студентов. Предлагаемый программный комплекс достаточно сложен и для такого класса проектов обладает большой избыточностью. Поэтому он используется лишь частично или не используется вообще, так как разработчики RUP не привязывают жестко первые две особенности к третьей. Соотнесение действий в RUP с каким-либо приложением из набора носят рекомендательный характер. Поэтому в данном курсе студентам рекомендуются некоторые приложения для проектирования, но их использование не является обязательным.

Таким образом, в результате курса «Проектирование информационных систем» студенты:

- ознакомятся с современными методами проектирования информационных систем;
- получат базовое представление о самом процессе разработки ПО;
- попробуют себя почти во всех ролях команды разработчиков, таких как, например, менеджер проекта, архитектор, аналитик, разработчик и тестировщик, и ознакомятся с их основными функциями;
- создадут работающую информационную систему;
- научатся пользоваться современным ПО для разработки информационных систем.

В настоящее время происходит подготовка материалов для создания методических указаний для практических занятий по дисциплине «ПИС» с использованием RUP.

Литература

1. Буч Г. Объектно-ориентированный анализ и проектирование с примерами на C++. 2-е издание. С-Пб: Невский диалект, 1999.
2. Кролл П., Кратчен Ф. Rational Unified Process – это легко. Руководство по RUP для практиков. М.: Кудиц-образ, 2004.
3. Поллис Г., Огастин Л., Лоу К., Мадхар Д. Разработка программных проектов на основе RUP. М.: Бином, 2005.
4. Документация в составе продукта RUP.

4 СЕТИ ЭВМ И ТЕЛЕКОММУНИКАЦИОННЫЕ СИСТЕМЫ

СЖАТИЕ ИНТЕРФЕРОМЕТРИЧЕСКИХ ИЗОБРАЖЕНИЙ НА ОСНОВЕ СЕГМЕНТАЦИИ И ОПИСАНИЯ КОНТУРОВ ПОЛОС

А.Ю. Тропченко, А.А. Тропченко, М.М.Федосов

Введение

В настоящее время известны различные методы сжатия изображений. Они являются универсальными и ориентированы на достаточно широкий класс изображений [1]. Самым известным из них является метод JPEG, однако одним из основных недостатков такого метода является жесткое разбиение исходного изображения на фрагменты без учета структуры самого изображения [2].

Рассмотрим задачу сжатия достаточно специфических изображений – интерферометрических изображений. Алгоритм сжатия подобных изображений требует учета специфических особенностей этих изображений. Интерферометрические изображения являются полутоновыми и имеют повторяющийся характер полос (см. рис. 1,а и рис. 2,а). Отдельные полосы на изображении можно рассматривать как отдельные области изображения, пикселы которых обладают в силу статистической зависимости близкой яркостью. При этом полосы имеют строго определенную форму и достаточно большую площадь [3]. Наиболее известным методом, учитывающим структуру изображений, является фрактальный метод [4, 5]. Однако такой метод имеет немало недостатков, основной из которых – большая вычислительная сложность [2]. В предлагаемом алгоритме сжатия интерферометрических изображений за типовую область (своего рода фрактал) выбирается отдельная полоса интерферограммы, а все изображение рассматривается как совокупность таких полос. Поэтому, если учитывать повторяющийся характер полос, то можно добиться более высоких коэффициентов сжатия, чем позволяет JPEG.

Алгоритм компрессии интерферометрических изображений

При сжатии прежде всего требуется выполнить выделение локальных областей различной яркости на интерферограмме, каждой из которых соответствует отдельная полоса, образуемая пикселами с близкой яркостью. Для выделения областей с близкой яркостью пикселов на изображении можно использовать различные методы сегментации изображения. Классические методы сегментации основаны на использовании порога интенсивности [6]. Достаточно эффективным методом сегментации является сегментация на основе адаптивно выбираемого порога [7, 8].

Определение порогов в этом случае связано с анализом гистограммы – отображения из множества $\{\alpha, \ldots, \beta\}$ значений яркости в множество натуральных чисел, каждому $b \in \{\alpha, \ldots, \beta\}$ сопоставляется число точек (m, n), для которых $B(m, n) = b$.

Глобальный максимум гистограммы соответствует наиболее часто встречающемуся значению яркости $b^0_{\max}$. В большинстве задач доминирует фон, так что значение $b^0_{\max}$ отвечает фону. Следует ожидать, что и близкие к $b^0_{\max}$ значения яркости также соответствуют фону. Для определения порога, отделяющего яркость объектов от яркости фона, достаточно располагать дополнительной информацией.

Допустим, что известно соотношение, связывающее яркость любой точки фона и объектов, например,

$$B(m, n) - B(u, v) > T, \qquad (1)$$

для любых точек (*m, n*), принадлежащих фону, и (*u, v*), принадлежащих объектам сцены. На основании (1) можно заключить, что для любой точки (*u, v*) любого объекта выполнено условие

$$B(u, v) < b^0_{\max} - T. \quad (2)$$

Найдем теперь глобальный максимум части гистограммы – в области $[\alpha, b^0_{max} - T]$. Пусть он достигается в точке $b^1_{\max}$. Следует ожидать, что $b^1_{\max}$ – яркость наиболее встречающаяся в точках объектов. Таким образом, можно заключить, что для любой точки (*m, n*) области фона выполнено условие

$$B(m, n) > b^1_{\max} + T. \quad (3)$$

Соотношения (2) и (3) и определяют пороги яркости для точек объектов и фона.

После сегментации можно получить бинарное изображение – маску, единичные биты которой соответствуют выделенным локальным областям (светлым полосам интерферограммы), а нулевые биты – черным полосам или фону. Тогда можно считать, что единичные области маски – это сжимаемые объекты, а нулевые – фон. Тем самым объем бинарной маски можно сжать уже минимум в 2 раза.

Объем изображения можно существенно сократить, если описать контур каждой выделенной области с помощью цепного кода Фримена, что достаточно просто производится на практике. Такой подход позволяет от двумерных объектов перейти к их одномерному (векторному) описанию, т.е. построение цепного кода может рассматриваться как процедура векторизации изображения. Как известно, цепной код для каждой точки контура строится как трехбитовый код, определяющий направление перехода к следующей точке согласно матрице связности [9]. В результате векторизации получаем набор начальных точек и вектора, подробно описывающих каждую область. Это основной этап компрессии, так как именно на этом этапе происходит основное уменьшение объема исходного изображения.

Так как любое из направлений кодируется трехбитным кодом, то для их представления можно использовать формат данных типа упакованных байтов. Иначе говоря, в одно 16-ти разрядное машинное полуслово записывается пять векторов направлений для пяти соседних точек контура. Очевидно, что это позволяет сократить объем векторизованного изображения не менее чем в 3–4 раза.

Для дальнейшего сжатия данных используются процедуры выделения RLE-цепочек, описывающих повторяющиеся коды направлений, и последующего оптимального кодирования, например, с использованием кода Хаффмана или метода арифметического кодирования [10].

Таким образом, предлагаемый *алгоритм компрессии интерферометрического изображения* состоит из следующих этапов:

- сегментации исходного изображения, например, на основе адаптивно выбираемого порогового ограничения;
- построение бинарной маски, у которой светлым полосам интерферограммы соответствуют едичные области;
- векторизация изображения с помощью цепного кода Фримена;
- упаковка трехбитных кодов направления в полуслова;
- выделение RLE-цепочек и кодирование данных с использованием оптимального кода.

Нетрудно видеть, что первые два этапа приводят к потерям при восстановлении изображения, в связи с чем данный метод относится к алгоритмам сжатия с потерями.

Алгоритм декомпрессии интерферометрических изображений

Для декомпрессии необходимо выполнить указанные выше этапы в обратном порядке. Заметим, что декодирование оптимального кода и получение цепочек повто-

ряющихся кодов направлений выполняется по традиционной схеме, например, также как и для метода JPEG. Затем необходимо распаковать данные, т.е. из каждого полуслова выделить пять соседних кодов направлений. Далее выполняется растеризация изображения, т.е. из векторного представления оно преобразуется в растровое. Для этого, начиная с начальной точки, с помощью вектора направления строится контур области. Затем необходимо закрасить область, ограниченную контуром. Для заливки может использоваться растровая развертка или затравочное заполнение.

Так как области имеют достаточно большой размер, то быстрее будет выполняться метод растровой развертки. Он состоит в следующем. Для каждой сканирующей строки, пересекающей ребра многоугольника, если пересечение находится слева от перегородки, то перекрасить все пикселы, центры которых лежат справа от пересечения сканирующей строки с ребром и слева от перегородки. Если пересечение находится справа от перегородки, то перекрасить все пикселы, центры которых находятся слева от пересечения строки с ребром и справа от перегородки. Перегородку проводят через одну из точек области, например, начальную.

После того, как выполнена заливка, изображение имеет резкие границы локальных областей. В целях сглаживания границ целесообразно выполнить усредняющую (медианную) фильтрацию всех граничных фрагментов области [6]. Заметим, что медианная фильтрация эффективна только в тех случаях, когда изображение имеет достаточно резкие границы.

При медианной фильтрации выбирается окно размером $(2k + 1) \times (2k + 1)$, и в каждой точке (i, j) растра яркость пересчитывается по следующему правилу. Расположим окно так, чтобы центр совпадал с точкой (i, j), и пронумеруем яркости $(2k + 1)^2$ элементов изображения, попавших в окно, в порядке возрастания: $b_1 \le b_2 \le \ldots \le b_l$. Набор $b_1, \ldots, b_l$ может содержать пронумерованные различными индексами равные значения. Медианой неубывающего набора $b_1, \ldots, b_l$ называется ее средний элемент b_m, $m = (l + 1)/2$. Медианная фильтрация заключается в замене значения яркости центрального элемента окна значением медианы неубывающего набора яркостей элементов, попавших в окно.

Заметим, что если объект состоит из $l \le ½ (2k + 1)^2$ точек (т.е. по площади не превосходит половины площади окна), он заведомо уничтожается, что вносит некоторые искажения.

Анализ эффективности предложенного метода сжатия

Предложенный метод достаточно прост для реализации. Но вместе с тем он должен давать высокие результаты (коэффициент сжатия), так как учитывает специфические особенности интерферометрических изображений. Для определения эффективности предложенного алгоритма сжатия-восстановления выбраны следующие основные величины, характеризующие метод сжатия:

- коэффициент сжатия ($K_{сж}$), определяющий, во сколько раз файл, хранящий сжатое изображение, меньше файла, хранящего исходное изображение;
- относительное среднеквадратичное отклонение (СКО) между исходным и полученным после сжатия изображением, которое выражается в процентах от динамического диапазона допустимых значений пикселов.

Рассмотрим эффективность разработанного алгоритма на примере двух изображений FourOpt.bmp и FromPhase.bmp (рис. 1,а и рис. 2,а). Для большей наглядности эффективности сравним результаты разработанного алгоритма и алгоритма JPEG.. Необходимо отметить, что использовался коэффициент сжатия JPEG, обеспечивающий сравнимую погрешность при восстановлении. Результаты для двух изображений приведены в таблице, а восстановленное изображение для первого примера представлено

на рис. 1,б), а для второго – на рис. 2,б. Исходя из рассчитанной относительной среднеквадратичной погрешности восстановленных изображений и полученных значений коэффициента сжатия, видно, что разработанный алгоритм является очень эффективным для сжатия интерферометрических изображений.

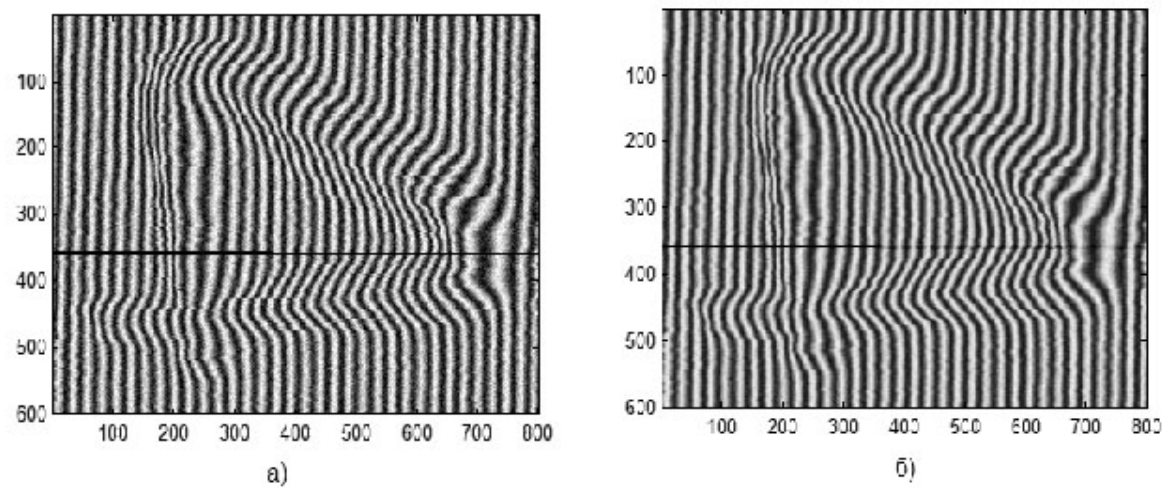


Рис. 1. Изображение FourOpt.bmp:
а) исходное изображение, б) восстановленное изображение

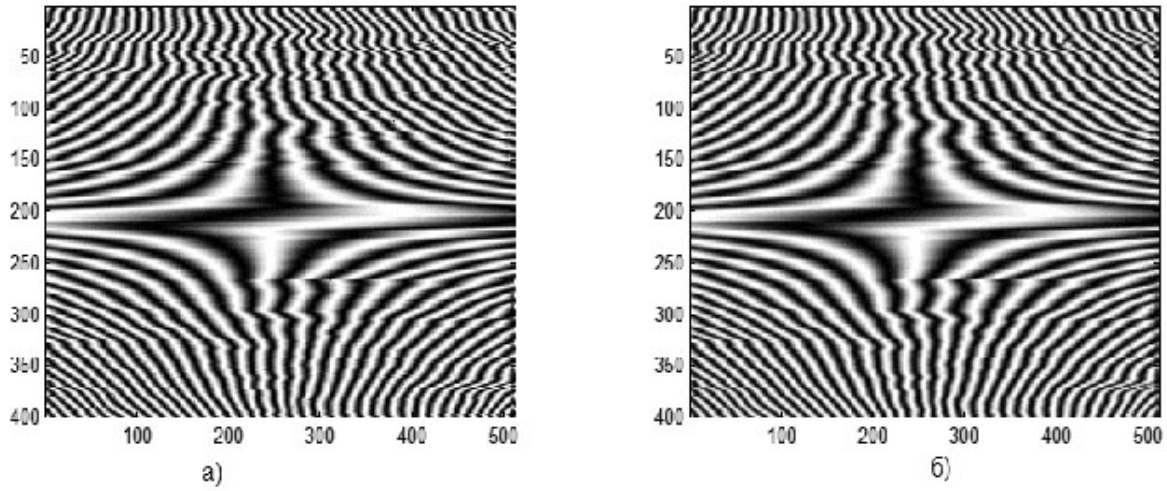


Рис. 2. Сравнительное изображение FromPhase.bmp:
а) исходное изображение, б) восстановленное изображение

Характеристики	Исходный файл	JPEG	Разработанный алгоритм
FourOpt.bmp			
Размер файла	481 078	81 946	24 153
Коффициент сжатия		5,87	19,92
СКО		5,5%	5,34%
FromPhase.bmp			
Размер файла	205 878	33 045	13 033
Коффициент сжатия		14,56	36,91
СКО		8%	7,68%

Таблица. Сравнительные характеристики сжатия

Заключение

Таким образом, результаты, полученные при сжатии интерферограмм и их последующей декомпрессии, свидетельствуют об эффективности предложенного метода сжатия. К достоинствам данного метода можно отнести высокую степень сжатия и малую степень искажения. Недостаткам является невозможность регулирования степени сжатия изображения.

Характеристики предложенного метода сжатия можно улучшить, если при компрессии исходного изображения после упаковки кодов вектора направления дополнительно использовать более эффективные методы квазиоптимального кодирования [10].

Литература

1. Ватолин Д., Ратушняк А., Смирнов М., Юкин В. Методы сжатия данных.М.: Диалог-МИФИ, 2002.
2. Вотолин Д.С. Алгоритмы сжатия изображений. / Учебное пособие. М.: МГУ, 1998. 52 с.
3. Васильев В.Н., Гуров И.П. Компьютерная обработка сигналов в приложении к интерферометрическим системам. СПб: bhv, 1998.
4. Fisher Y. Fractal image compression. // Sig-Graf 92. 1992. № 4.
5. Jacquin A. Fractal image coding based on a theory of iterated contractive image transformations. // Open systems. 1995. № 5.
6. Прэтт У. Цифровая обработка изображений. Т.1, 2. М.: Мир, 1982.
7. Глущик Р.В. Процедуры распознавания и локализации объектов на изображении. / В сб. «Современные технологии. Труды молодых ученых ИТМО». СПб, СПбГИТМО, 2001. С.106–109.
8. Ожиганов А.А., Тропченко А.А., Тропченко А.Ю. Модифицированный фрактальный метод сжатия многоуровневых изображений. // Информационные технологии. 2003. № 4.
9. Бутаков Е.А., Островский В.И., Фадеев И.Л. Обработка изображений на ЭВМ. // М.: Радио и связь, 1987.
10. Семенюк В.В. Экономное кодирование дискретной информации. СПб: СПБ ГИТМО (ТУ), 2001.

ИСПОЛЬЗОВАНИЕ ДИСКРЕТНЫХ ВЕЙВЛЕТ-ПРЕОБРАЗОВАНИЙ ДЛЯ ЦИФРОВОГО МАРКИРОВАНИЯ ИЗОБРАЖЕНИЙ

М.В. Гришин, А.А. Ожиганов, А.Ю. Тропченко

Рассматривается модификация метода маркирования изображений на основе дискретных ортогональных вейвлет преобразований.

Введение

Проблема защиты авторского права на мультимедиа информацию существовала всегда, но стала особенно актуальной с развитием средств вычислительной техники и средств телекоммуникации.

Для защиты данных можно использовать хорошо зарекомендовавшие себя алгоритмы шифрования, но для мультимедиа данных это не лучший выбор. Зашифрованную мультимедиа информацию невозможно ни увидеть, ни услышать без расшифровки. Для данной категории информации хотелось бы иметь такие средства защиты, которые позволяли бы «подписывать» данные без потери информативности, т.е. заметного ухудшения качества изображения или звука. В этом случае «подписанный» объект можно слушать или просматривать, но одновременно можно в любой момент доказать, кому он принадлежит. При этом особенно важным является устойчивость цифровой подписи. Сжатие мультимедийных данных, стандартные методы цифровой обработки сигналов: фильтрации, масштабирование, повороты - все это оказывает влияние на цифровую подпись.

Этим искажениям в той или иной степени противостоят методы цифрового маркирования мультимедиа информации.

Подпись, скрываемую в данных, называют «цифровым водяным знаком» – ***ЦВЗ***. ЦВЗ может иметь различную природу: логотип компании, случайная последовательность чисел или начальные параметры для ее генерации.

Цель маркирования – определение:

- владельца объекта маркирования;
- изменений, произведенных над объектом маркирования;
- легальности объекта маркирования (права его использования).

Вне зависимости от типа ЦВЗ и области его применения можно привести некоторые обобщенные схемы создания и извлечения ЦВЗ. В общем виде проблема маркирования изображений рассматривается как проблема передачи слабого сигнала малой мощности (ЦВЗ) в широкополосном сигнале (изображении) таким образом, чтобы быть визуально не воспринимаемым и устойчивым к искажениям, которые могут появиться в процессе передачи информации.

На рис. 1 представлен процесс получения маркированного изображения.

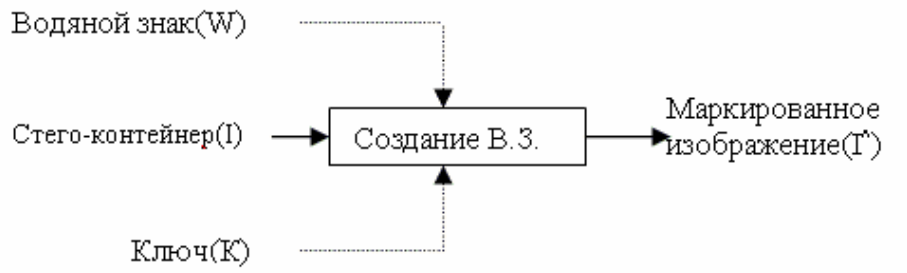


Рис. 1. Процесс маркирования изображения: I – оригинал изображения; W – водяной знак; K – ключ (начальное значение для источника гененирования случайной величины)

Процесс добавления водяного знака может быть записан так: I+K+W→I^.
Процесс идентификации водяного знака представлен на рис. 2.

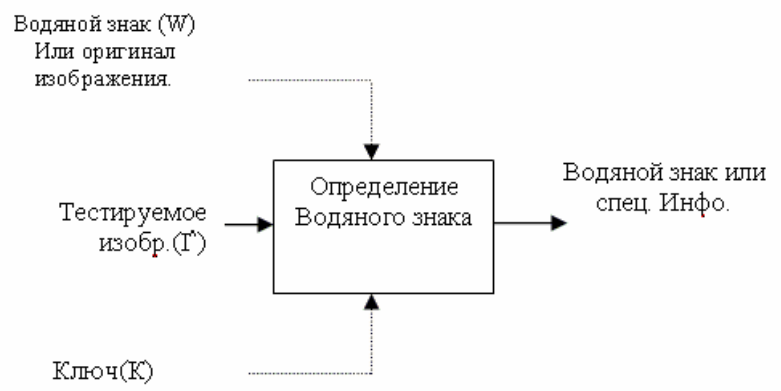


Рис. 2. Процесс идентификации водяного знака

Этот процесс восстанавливает ЦВЗ или дает информацию о том, существует ли подобный знак в тестируемом изображении.

Предложенный в статье метод маркирования является модификацией методов Corvi и миксинг-системы [1, 2].

Пространственное преобразование плоской области

Вначале рассмотрим метод преобразования черно-белого логотипа, выступающего в качестве ЦВЗ, в шум или в псевдослучайную последовательность. Обычно логотип представляет собой неоднородную картинку, в которой имеются большие области черного и белого, и при добавлении в изображение он может исказить последнее. Поэтому необходимо распределить ЦВЗ по всей поверхности изображения, превращая его в «шум». Для этого можно использовать пространственное преобразование плоской области U, определяемое формулой:

$$r' = Ar (\mathrm{mod}\, M), \begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} (\mathrm{mod}\, M), \quad (1)$$

где $a_{ij} \in Z$, $\det A = 1$ и $\lambda_{1,2} \notin \{-1,0,1\}$ – собственные значения матрицы A, x, y – координаты точки в логотипе, r – положение точки в логотипе до преобразования, r' – положение точки в логотипе после преобразования, M – размер логотипа.

Итерационное воздействие матрицей A на точку $r_0 \in U$ образует динамическую систему. Эту систему можно представить следующим образом:

$$r_{n+1} = Ar_n (\mathrm{mod}\, M), \quad (2)$$

где $n = 0, 1, 2\ldots$.

Рассмотрим одно из таких преобразований с матрицей A вида ($a_{11}=a_{12}=a_{21}=1$, $a_{22}=2$). В качестве логотипа используется изображение «кот». На рис. 3 изображены различные виды одного логотипа после итерационного воздействия на него матрицей A. Из первоначального состояния (рис. 3(а)) после первой итерации логотип преобразуется в рис. 3(б). После десятой итерации логотип преобразуется в псевдослучайную последовательность (рис. 3(в)). Если преобразование итерационно повторить 24 раза, то логотип вернется в первоначальное состояние (рис. 3(д)).

Оригинал n=1 n=2 n=10 n=24

а) б) в) г) д)

Рис. 3. Динамическое изменение логотипа под воздействием матрицы *A*

Обычно в преобразовании подобного типа используют одну из стандартных матриц A, записываемую в виде:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \end{pmatrix} (\mathrm{mod}\, M), \quad (3)$$

где $k \in [1, M) \subset Z$.

Подобные преобразования являются периодическими, т.е. существует номер итерационного цикла T, такой, что $r_0 = r_T$ (период подбирается опытным путем для каждого логотипа).

Маркирование изображений в вейвлет-пространстве

В процессе маркирования маркируемое изображение подвергается 3-х уровневому вейвлет-преобразованию Хаара [3, 4]. Маркируется только низкочастотная, аппроксимирующая составляющая (*LL* подуровень).

Цифровой знак представляет собой черно-белый логотип. Размер логотипа не должен превышать размеры аппроксимирующей составляющей вейвлет преобразования. Если размеры логотипа меньше размеров *LL* подуровня, то размер холста логотипа увеличивается до размеров *LL* подуровня, а сам логотип остается в центре этого холста.

К каждой точке полученного ЦВЗ применяется алгоритм перемешивания, описанный выше. Преобразование 1 применяется к ЦВЗ n раз. Число n подбирается экспериментально (преобразование повторяется до тех пор, пока спектральная плотность логотипа не станет постоянной).

Процесс маркирования можно представить следующим образом:

$$f'(x, y) = f_{mean} + \left[(f(x, y) - f_{mean}) + ((1 + w(x, y)) \cdot \alpha \right], \quad (4)$$

где f_{mean} – среднее значение коэффициентов LL подуровня, $f'(x, y)$ – маркированный коэффициент LL подуровня в позиции (x,y), $f(x, y)$ – исходный коэффициент *LL* подуровня в позиции (x, y), $w(x, y)$ – отчет цифрового знака в позиции (x,y), α – коэффициент, определяющий силу добавления ЦВЗ.

Затем, для получения маркированного изображения, к измененным вейвлет-коэффициентам применяется обратное вейвлет-преобразование.

При извлечении ЦВЗ требует оригинал *LL* подуровня. Процедура извлечения ЦВЗ обратна процедуре добавления.

После извлечения ЦВЗ $w(x, y)$ к нему T-n раз применяется преобразование (1). Здесь n – число «перемешиваний», применяемое к логотипу перед маркированием, а T – количество итерационных циклов (постоянная величина для данного логотипа).

На рис. 4 и 5 представлены извлеченные ЦВЗ после JPEG сжатия и зашумления изображения. Из рис. 4 видно, что при сжатии JPEG-ом до значения QF=30 логотип уверенно извлекается и визуально различим. Сжатие изображений с QF < 30 приводит к достаточно сильному искажению изображения и широко не применяется в коммерче-

ских целях. Поэтому можно заключить, что модифицированный метод Corvi устойчив к JPEG сжатию на достаточно широком интервале *QF*.

Из рис. 5 видно, что до уровня шума, равного 5, ЦВЗ, хоть и извлекается достаточно зашумленным, но визуально различим. Нужно отметить, что при зашумлении уровнем шума от 5 и более изображение уже теряет свою коммерческую ценность. Поэтому можно заключить, что модифицированный метод Corvi устойчив к зашумлению маркированного изображения при уровнях шума до 5.

Таким образом, можно заключить, что предложенный метод маркирования показывает хорошую устойчивость к атакам различного типа. Хотя для восстановления ЦВЗ требуется оригинал изображения, хранить этот оригинал целиком нет необходимости. Поскольку маркируется только аппроксимирующая составляющая вейвлет-преобразования, то и для восстановления ЦВЗ потребуется оригинал только аппроксимирующей составляющей. Этот факт позволяет значительно сэкономить на хранении оригиналов изображений, поскольку даже при 3-х уровневом преобразовании размер аппроксимирующей составляющей меньше полного размера изображения в 64 раза.

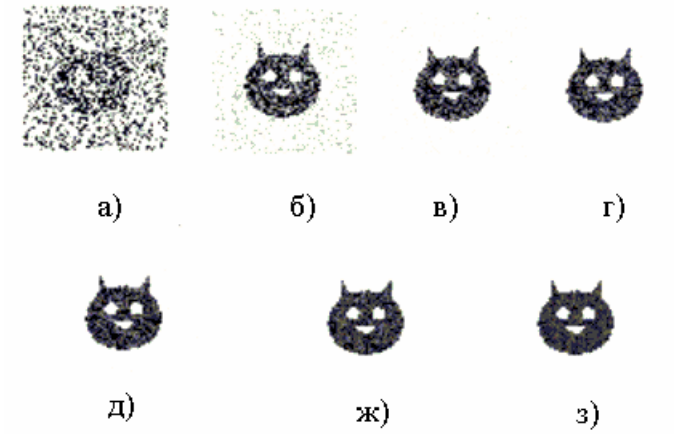
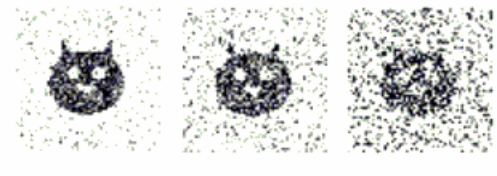


Рис. 4. Сравнение качества извлечения ЦВЗ из маркированного изображения, сжатого JPEG алгоритмом различным качеством:
а) QF=10, б) QF=30, в) QF=40, г) QF=50, д) QF=60, ж) QF=70, з) QF=80



а) б) в)

Рис. 5. Сравнение качества извлечения ЦВЗ из маркированного изображения, зашумленного гауссовым шумом:
а) с уровнем шума 1, б) с уровнем шума 2, а) с уровнем шума 5.

Поскольку данный метод маркирования скрывает цифровую подпись в аппроксимирующей составляющей преобразованного изображения, то он может быть адаптирован для применения совместно с алгоритмами сжатия изображений на основе вейвлет-преобразований, поскольку подобные алгоритмы стараются полностью сохранить аппроксимирующую составляющую вейвлет преобразования.

Заключение

В работе рассмотрен новый метод маркирования изображений с использованием дискретного вейвлет-преобразования. Представлены результаты экспериментов по из-

влечению ЦВЗ после внесения в маркированное изображение различных искажений. Даны рекомендации по применению данного метода совместно с алгоритмами сжатия изображений.

Литература

1. Peter Meerwald. Digital image watermarking in the wavelet transform domain. // Salzburg, am 11. Janner 2001,

2. George Voyatzis and Ioannis Pitas. Digital image watermarking using mixing systems. // Computer & Graphics, 22(4):405-416, August 1998.

3. Дьяконов В.П. От теории к практике. Вейвлеты. М.: Солон-Р, 2002.

4. Воробьев В.И., Грибунин В.Г. Теория и практика вейвлет-преобразований. Спб: Военный университет связи; 2000.

ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ КЭШИРОВАННОГО ДИСКОВОГО ВВОДА/ВЫВОДА

В.С. Зотеев, В.В. Шефов, Б.Д. Тимченко

Представляются организация и результаты экспериментального исследования производительности дисковой памяти Windows-систем. Определяется длительность выполнения кэшированных операций дискового ввода/вывода для последующего использования в качестве параметров обслуживания в аналитических и имитационных моделях производительности Windows-систем.

Введение

В современных системах обработки данных дисковая память, наряду с процессором и основной памятью, является ключевым ресурсом, определяющим качество обслуживания. Технические параметры накопителей, такие как время позиционирования, скорость вращения, плотность записи, постоянно улучшаются, что ведет к соответствующему увеличению их технической (номинальной) производительности. Одной из обычно используемых характеристик здесь является паспортная величина среднего времени доступа, либо его составляющие (см., например, [10]).

Одновременно с увеличением номинальной производительности чисто технологическими средствами повсеместно используются архитектурно-программные решения, направленные на увеличение производительности при обслуживании запросов программ на дисковый ввод/вывод. Здесь характеристикой служит длительность обслуживания, обеспечиваемая в определенной системной среде (что можно назвать системной производительностью), зависящая как от номинальной производительности накопителя, так и целого ряда системных факторов. Особо подчеркнем, что здесь имеется в виду именно время обслуживания, а не время выполнения запроса на ввод/вывод, в общем случае включающее время ожидания.

Важнейшим системным фактором в определении длительности обслуживания является дисковое кэширование, обеспечиваемое во всех современных программно-аппаратных средах. Говоря в дальнейшем о кэшировании, мы будем иметь в виду системы под управлением Windows 2000 и выше, а в более узком смысле серверные системы, имеющие в своем составе изолированные накопители. Тем самым специфика систем хранения с RAID-массивами разного уровня в этой работе не рассматривается.

Работа дисковой памяти в системах обработки данных исследовалась как модельно, так и экспериментально для различных аппаратно-программных платформ [1, 2, 6, 7]. Интерес к таким исследованиям сохраняется и в настоящее время в связи с тем, что узкие места при работе систем образуются в дисках гораздо чаще, чем это можно ожидать. Одной из причин этого является использование в системах накопителей большой единичной емкости, а также в общем не всегда эффективное управление размещением данных и логическими томами, особенно в серверах начального и среднего уровня.

Говоря об исследованиях производительности систем обработки данных в целом и дисковых подсистем, в частности, необходимо выделить ряд подходов, различающихся методами, инструментом и организацией.

1. Экспериментальный подход, включающий пассивный и активный эксперимент. В первом случае говорят о системометрии (мониторинге) – сборе данных на работающей системе для определения, в частности, показателей производительности [11]. Под активным экспериментом понимают фактически испытание системы – мониторинг ее работы на некоторой искусственной нагрузке, например, при проведении нагрузочного тестирования.
2. Аналитическое моделирование, при котором используется модель производительности, реализуемая математически (расчетным путем) [6]. Такие модели чаще всего строятся в терминах абстрактных обслуживающих приборов – систем массового об-

служивания [8, 11]. Важнейшим в определении обслуживающего прибора является закон обслуживания, определяющий длительность обслуживания, в простейших моделях задаваемую его средней величиной.

3. Имитационные модели, принимающие форму программы, событийно моделирующей работу системы в дискретном времени. Имитационная модель, реализуемая как выполнение (прогон) программы, в основе своей имеет ту же абстракцию обслуживающего прибора, что и аналитические модели.

Качественное различие экспериментальных и модельных подходов состоит в возможности использования последних для проведения исследований в некоторой области варьирования параметров, влияющих на производительность. Хотя аналитические модели для этого гораздо лучше приспособлены, имеются аргументы и в пользу имитационных моделей. Важно, что любая модель для производства модельного эксперимента должна быть параметрически определена (идентифицирована). Таким образом, далее будет говориться об экспериментальном определении длительности обслуживания в дисковых накопителях Windows-систем для использования этих величин в качестве модельных параметров.

Для проведения экспериментального исследования необходимы:

1. определенные концептуальные представления о структуре и организации исследуемой системы;
2. обоснованный выбор инструмента – мониторного средства;
3. способ (организация) измерений и обработки данных.

Для изложения подхода будем исходить из организации выполнения дисковой операции и ее мониторинга в Windows, представленной на рис. 1.

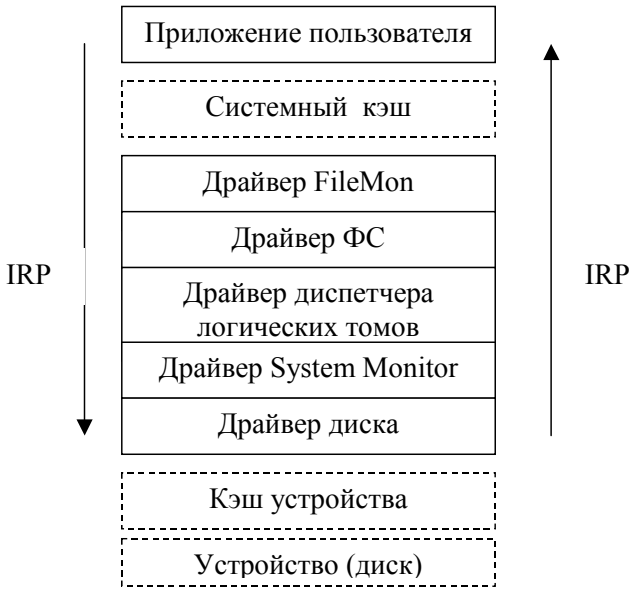


Рис. 1. Выполнение запроса на ввод/вывод и его мониторинг

Здесь кэш устройства – внутренняя память контроллера диска (2–8 Мб и более для современных дисков), используемая обычно для упреждающего чтения (в нее считываются как запрашиваемые данные, так и следующие за ними данные до конца дорожки диска) и отложенной записи (операция записи считается выполненной после попадания данных в кэш) [6]. Чаще всего возможно отключение кэширования записи на уровне устройства (физического тома).

Системный кэш – это часть резидентного набора операционной системы, под которую выделяется некоторый объем физической памяти, динамически изменяющийся в процессе работы. Кэш Windows служит для хранения частей файлов, обращения к которым были в последнее время, использует страничный механизм и работает в тесном

взаимодействии с диспетчером виртуальной памяти. Прозрачные для приложений части файла проецируются на область адресов системного кэша, и операция ввода/вывода сводится к копированию данных из буфера пользовательского приложения в область кэша (нужную проекцию файла) при записи и в обратном направлении – при чтении [4]. Системное кэширование можно отключать на уровне отдельных файлов (путем указания особых флагов при создании/открытии файла [2]). Однако, как показали исследования [7], большинство операций ввода/вывода кэшируются со стороны Windows.

При вызове пользовательским приложением функции чтения/записи обычно (но не всегда) формируется пакет запроса ввода/вывода – IRP (I/O Request Packet), структура данных, описывающая запрос [3, 9]. В состав IRP входят поле типа операции, указатели на объект «файл», размер чтения/записи и другая информация. В случае некэшированного ввода/вывода IRP последовательно проходит через стек протоколов Windows сверху вниз – через драйверы фильтров файловой системы (в случае их наличия), драйвер файловой системы и так далее до драйвера диска. После выполнения устройством операции ввода/вывода IRP проходит стек снизу вверх, после чего результаты операции доступны пользовательскому приложению. Ряд мониторных средств использует данную организацию ввода/вывода и фиксирует временные метки прохождения IRP через различные драйверы (стандартный монитор Windows – System Monitor – применяет свой драйвер diskperf.sys [5], а FileMon – драйвер-фильтр файловой системы).

Ситуация меняется в случае кэшированного ввода/вывода. В этом случае IRP может вообще не создаваться (случай так называемого быстрого ввода/вывода – операции, при которой диспетчер ввода/вывода обращается напрямую к диспетчеру кэша, минуя драйвер файловой системы) или не опускаться по стеку протоколов ниже уровня драйвера файловой системы.

Выбор мониторного средства является принципиальным при проведении любых экспериментальных исследований. Поскольку в экспериментальном подходе большое значение имеет воспроизводимость экспериментов, предпочтительным является использование встроенных измерительных средств, которым в Windows является System Monitor [3]. Однако этот монитор обладает принципиальным недостатком: его разрешение ввода/вывода не превышает 1 мс. Так как в случае попадания в кэш при чтении и отложенной записи IRP не доходит до драйвера diskperf (или вообще не создается), System Monitor фактически не способен определять времена выполнения кэшированных операций чтения/записи. Поэтому нами было использовано другое средство – измеритель FileMon, специально предназначенный для мониторинга ввода/вывода на уровне IRP (сайт разработчиков – www.sysinternals.com). FileMon способен определять длительности операций ввода/вывода (в том числе кэшированного – путем перехвата вызовов функций диспетчера кэша) с разрешением в доли мкс.

Эксперименты проводились на ПК со следующей конфигурацией: ОС Windows XP HE SP1, RAM 512Мб, 2 HDD Seagate. Файлы, с которыми работали генераторы нагрузки, создавались на жестком диске, где отсутствовали файлы ОС, в разделе с ФС FAT32 (размер кластера 8 Кб).

В качестве генераторов нагрузки использовались две общедоступных утилиты – Intel IOMeter и SQLIO2 ([1]). SQLIO2 является приложением, используемым Microsoft NT Performance Group в исследованиях производительности Windows, а IOMeter (www.intel.com) – бенчмарк ввода/вывода фирмы Intel, имеющий функции генератора нагрузки и монитора.

Эксперименты проводились для различных параметров создаваемой нагрузки – последовательного/случайного кэшируемого/некэшируемого чтения/записи – в однозадачном режиме. Запись и чтение велись порциями, равными одному кластеру. Экспериментальные результаты в форме средних значений представлены на рис. 2, 3.

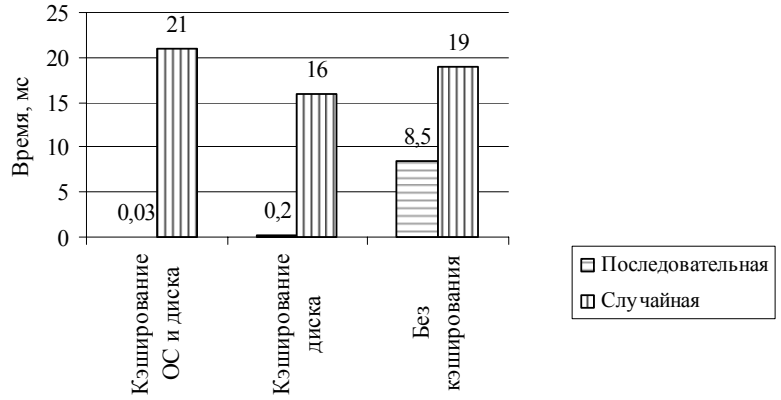


Рис. 2. Значения среднего времени операции записи

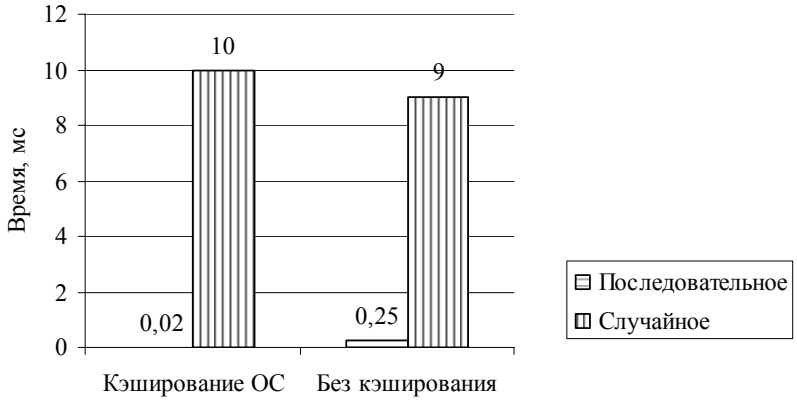


Рис. 3. Значения среднего времени операции чтения

Главный вывод состоит в том, что кэширование (как на уровне жесткого диска, так и на уровне операционной системы) на несколько порядков уменьшает время операций последовательного чтения/записи. Физическая же операция записи (при последовательной записи) происходит только при отключенном на двух уровнях кэшировании. Ее средняя длительность при синхронной последовательной записи примерно равна времени полного оборота жесткого диска (1 / 7200 об/мин = 8,3 мс).

При случайном чтении из файла большого размера времена чтения приближаются к паспортным данным жесткого диска – из-за чтения разных частей файла и значительного перемещения головки жесткого диска упреждающее чтение становится неэффективным как на уровне ОС, так и на уровне жесткого диска. Средние времена чтения для кэшированного ОС чтения немного больше, чем для некэшированного, что можно объяснить дополнительными задержками на работу потока упреждающего чтения, которая не приводит к увеличению производительности из-за чтения случайных частей файла.

Заключение

Из полученных экспериментальных результатов важен на данном этапе не столько количественный, сколько качественный вывод: использование паспортной технической производительности жестких дисков для параметризации моделей абсолютно неприемлемо. Нам представляется, что при моделировании систем с кэшированием должна быть пересмотрена сама базовая модель накопителя как обслуживающего устройства. Это требует проведения дополнительных экспериментальных исследований, в частно-

сти, изучения работы системного кэша в мультипрограммном режиме и определения влияния различных факторов на длительность операций ввода/вывода.

Литература

1. Chung L., Gray J., Worthington B., Horst R. Windows 2000 Disk IO Performance. Technical Report MS-TR-2000-55. June 2000. http://research.microsoft.com/BARC/sequential_IO/.
2. Disk Subsystem Performance Analysis for Windows. White paper. Microsoft Corporation, 2004.
3. Microsoft Developer Network Library-January 2004\Windows Development\Windows Base Services\Performance Monitoring.
4. Nagar R. Windows NT File System Internals. O'Reilly, 1997.
5. Pentakalos O., Friedman M. Windows 2000 Performance Guide. O'Reilly, 2002.
6. Shriver E. Performance modeling for realistic storage devices. Department of Computer Science, New York University, 1997.
7. Vogels W. File system usage in Windows NT 4.0. Department of Computer Science, Cornell University, 17th ACM Symposium on Operating Systems Principles (SOSP'99).
8. Основы теории вычислительных систем. / Под ред. С.А. Майорова. Учеб. пособие для вузов. М.: Высш. школа, 1978.
9. Соломон Д., Руссинович М. Внутреннее устройство Windows 2000. Мастер-класс. СПб: Питер, 2004.
10. Столлингс В. Операционные системы, 4-е издание. / Пер. с англ. М.: Издательский дом «Вильямс», 2002.
11. Феррари Д. Оценка производительности вычислительных систем. / Пер. с англ. А.И. Горлина, Ю.Б. Котова и Л.В. Ухова / Под ред. В.В. Мартынюка. М.: Мир, 1981.

АССОЦИАТИВНЫЙ ПОИСК ДАННЫХ С ПОМОЩЬЮ НЕЙРОННОЙ СЕТИ

И.А. Бессмертный, А.А.Коваль, Р.О. Белоус

В статье приведены результаты исследования возможности организации ассоциативного доступа к данным с помощью нейронных сетей Кохонена. В качестве инструментальной среды использовано средство *Neural Network Toolbox* в составе пакета *Matlab.* Приводятся результаты обучения нейронной сети на отдельной таблице данных и предлагаются способы организации пользовательского интерфейса для доступа к данным.

Введение

Информационный взрыв, в условиях которого мы живем, начиная со второй половины XX века, порождает все возрастающие объемы знаний, удвоение которых уже в 70-е годы происходило каждые пять лет [1]. Отсюда возникла проблема поиска необходимых данных, получившая название *data mining.* Этот термин очень точно отражает суть процесса – добычу сырых, т.е. неформализованных, данных.

Наиболее популярным средством организации поиска в базах данных является структурированный язык запросов SQL (*Structured Query Language*), а в сети Интернет –индексация ключевым словом и создание каталогов.

Язык SQL был разработан как универсальный язык манипулирования данными для реляционных баз данных. Он базируется на реляционном исчислении и представлении базы данных в виде совокупности таблиц-отношений. Обладая массой достоинств по сравнению с более старыми языками, SQL все же обладает рядом недостатков. Во-первых, он может оперировать только данными со строгой, заранее определенной структурой, представленными в виде отношений, что ограничивает его применимость. Во-вторых, он требует знания реляционной алгебры. В-третьих, запросы нередко достигают устрашающих размеров и с трудом поддаются визуализации [2].

Поисковые машины используют для своей работы другой метод – индексацию. Они составляют специальное удобное для поиска представление документа, называемое индексом. Можно выделить два основных класса алгоритмов индексации. Первый из них – лексическое индексирование, ставшее традиционным в поисковых системах. Оно основан на булевой алгебре и является вполне удовлетворительным для экспертов, ищущих информацию в той или иной предметной области. Второй метод – векторное индексирование – распространен меньше и позволяет осуществлять поиск по подобию. В нем документ рассматривается как вектор в неком лексическом пространстве, и релевантность документа запросу определяется в геометрическом смысле. Этот метод в своем первоначальном варианте плохо работал на запросах с малым количеством слов. Более продвинутые варианты этого алгоритма показывают гораздо лучшие результаты, но сильно проигрывают по быстродействию алгоритмам лексического индексирования. В целом, алгоритмы индексирования неплохо подходят для поиска данных в неоднородном информационном пространстве, но применимы лишь к текстовым документам и дают удовлетворительные результаты только в пределах определенной предметной области, причем пользователь должен знать, какая именно его информация интересует [4, 6].

Помимо поисковых систем, для поиска в Интернете могут использоваться каталоги. Они рассматривают представленную в виде графа (обычно дерева) структуру связанных тем, к каждой из которых относится некоторое количество документов, близких по содержанию.

Автоматизированное создание таких каталогов затруднено, поэтому они требуют большого количества работы. Кроме того, обычно структура каталогов отражает лишь

самые очевидные взаимосвязи документов. Для отражения сложной системы взаимосвязей оказывается необходимым привлекать экспертов к созданию каталога, который, в таком случае, содержит достаточно ограниченное количество документов из какой-либо одной предметной области [5].

Таким образом, недостаточность стандартных методов поиска в современных условиях лавинообразного роста количества информации очевидна. Сложность и громоздкость языков запросов, нечувствительность к контексту, значительные требования к ресурсам аппаратного обеспечения – эти недостатки присущи в той или иной мере всем общепризнанным методам поиска.

В этой работе делается попытка применить для отображения и ассоциативного поиска данных аппарат нейронных сетей, а именно – сети Кохонена. Этот метод может оказаться лишенным части или даже всех вышеперечисленных недостатков.

Основной результат

Нейронные сети Кохонена (слои и карты) являются самообучающимися и могут быть применены для кластеризации и классификации данных, поиска закономерностей и других целей. Для моделирования сети мы будем использовать пакет *Neural Network Toolbox*, входящий в систему компьютерной математики *Matlab 6* .

В первую очередь необходимо объяснить разницу между кластеризацией и классификацией. Классификация – это процесс соотнесения векторов исходного пространства входов с некоторыми конечными классами, задаваемыми пользователем. Кластеризация же есть процесс исследования входного множества векторов с целью выявления и распределения их по кластерам (классам, таксонам) в соответствии с одним или совокупностью характерных признаков, определяющих «близость» между элементами множества. В большинстве случаев выбирается признак геометрического расстояния между точками пространства входных векторов. Кластеризация выполняется автоматически, кластеры не являются заданными изначально, а формируются на основании похожести (близости) векторов. Заранее может быть задано только количество кластеров. Вообще, для кластеризации могут применяться (и применяются) методы математической статистики. Но они плохо работают на небольших выборках, кроме того, требуют предварительного определения тех признаков, на основании которых будет делаться вывод о похожести или непохожести векторов. Сети Кохонена лишены этих недостатков.

Интересующие нас объекты обладают определенным набором признаков, которые мы находим существенными для задачи. Эти признаки должны быть закодированы в числовую форму, возможно, нормированы с помощью подходящего алгоритма. После этой предварительной обработки мы получим R-мерное пространство признаков (соответственно R – количество признаков), векторы в котором группируются определенным образом. Сети Кохонена, обучаясь на векторах обучающей выборки, выделяют эти группы-кластеры. Достаточно очевидно, что количество векторов обучающей выборки должно быть больше указанного количества кластеров S. На самом деле удовлетворительные результаты получаются при количестве векторов, большем $R*S$.

Если кластеры пространства входов могут быть разделены прямыми или плоскостями (гиперплоскостями при $R>3$), то говорят о линейной разделимости кластеров. Если они могут быть разделены более сложной линией (поверхностью), то говорят о нелинейной разделимости. В случае пересекающихся кластеров говорят о вероятностной разделимости, т.е. вектор может быть отнесен к кластеру с какой-то вероятностью. Сети Кохонена не могут быть применены для решения задач с вероятностной разделимостью (по крайней мере, без некоторых модификаций). Но обычно задача оказывается

линейно или хотя бы нелинейно-разделимой либо может быть сведена к таковой за счет предварительной обработки данных.

Результат работы сети может быть представлен в наглядной графической форме для размерности пространства входов, не превышающей двух. Для трехмерного пространства визуальное представление затруднено, а для больших размерностей – и вовсе невозможно. Для решения этой проблемы можно применить два метода. Первый из них заключается в сведении R-мерного пространства к двумерному с помощью некоторого отображения: $(x, y) = f(x_1, \ldots, x_n)$, либо, в частном случае, $x = x_i$, $y = f(x_1, ..., x_{i-1}, x_{i+1}, \ldots, x_n)$, $i = 1, 2, ..., n$.

Во втором случае подразумевается использование карты Кохонена, которая обладает свойством отображать многомерное пространство признаков на плоскость (этот случай выходит за рамки данной работы).

Получив двумерную карту признаков, можно наглядно изучить структуру предметной области и быстро узнать, к какому кластеру относится интересующий нас вектор (который вовсе не обязан встречаться в обучающем множестве). Собственно, на этом и основана идея применения сетей Кохонена для ассоциативного поиска данных. Подчеркнем, что эти данные могут быть как числовыми, так и текстовыми или даже графическими [7].

Рассмотрим архитектуру слоя Кохонена (рис 1) в обозначениях системы *Matlab* [3].

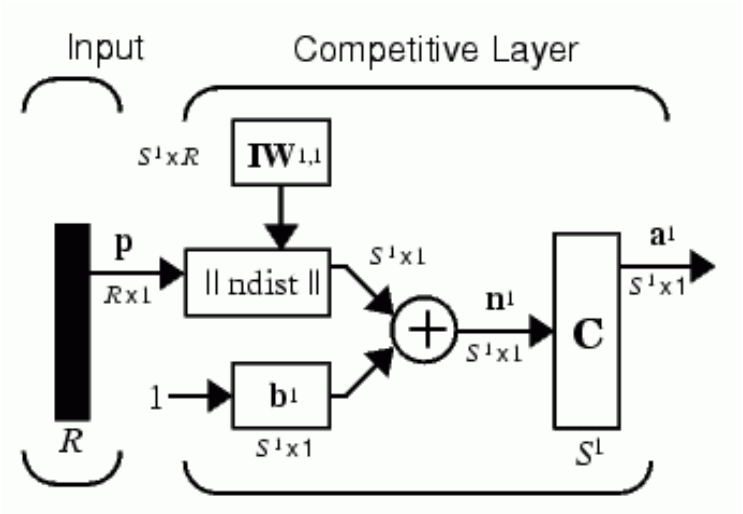


Рис 1. Архитектура слоя Кохонена

Это слой конкурирующего типа, поскольку в нем применена конкурирующая функция активации.

На вход подается вектор R-мерного пространства признаков. Вычисляется отрицательное евклидово расстояние между вектором и матрицей весов IW (вектор-строками матрицы). Вектор-строка матрицы IW с номером i задает центр i-го кластера. Затем полученный вектор расстояний суммируется с вектором смещений, и результирующий вектор подается на вход конкурирующей функции активации. Она анализирует значения элементов вектора входа $n1$ и формируем выходы нейронов, равные 0 для всех нейронов, кроме одного нейрона-победителя, имеющего на входе максимальное значение. Таким образом, вектор выхода слоя $a1$ имеет единственный элемент, равно 1, который соответствует нейрону-победителю, а остальные равны 0. Иными словами, эта функция выдает вектор размерности S (по числу нейронов в слое), у которого единице равен единственный компонент, соответствующий номеру того кластера, к центру

которого вектор входа оказался ближе всего. Все остальные компоненты этого вектора – нулевые.

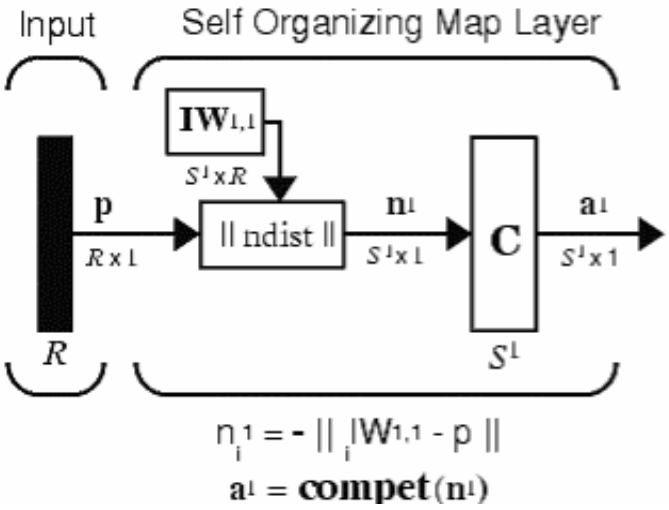


Рис 2. Архитектура самоорганизующейся карты Кохонена

Article I. Name	Family	Attendance	Progress	Sociability	Leadership
Anna	Romanova	6	6	10	8
Anna	Fokina	9	8	9	9
Maria	Kirih	9	9	8	10
Marina	Malevich	10	9	8	10
Irina	Nogina	10	10	3	3
Ksenia	Deeva	7	4	5	4
Nadezhda	Dorik	7	7	6	4
Julia	Ginzburg	10	8	7	5
Roman	Batalin	6	10	6	7
Aleksandr	Kozirev	9	9	8	10
Aleksandr	Konokradov	5	6	7	7
Kirill	Topolev	10	7	6	6
Mihail	Miheev	8	9	6	7
Andrei	Sopka	5	8	5	5
Vechyaslav	Telegin	4	6	7	3
Evgeniy	Mamontov	8	8	6	5
Dmitry	Alandarenko	2	4	4	3
Dmitry	Bovichev	1	1	3	3
Sergei	Odoevskiy	8	7	6	6

Рис. 3. Список студентов учебной группы

Самоорганизующаяся сеть в виде карты Кохонена также предназначена для задач кластеризации входных векторов. В отличие от слоя Кохонена, карта поддерживает такое топологическое свойство, когда близким кластерам входных векторов соответствуют близко расположенные нейроны. Первоначальная топология размещения нейронов в карте Кохонена соответствует размещению нейронов в узлах либо прямоугольной, либо гексагональной, либо рандомной сетки. Карта для определения нейрона-победителя использует ту же процедуру, что и в слое, однако на карте одновременно изменяются весовые коэффициенты соседних нейронов.

Архитектура самоорганизующейся карты Кохонена имеет вид, показанный на рис 2. Эта структура аналогична структуре слоя, но здесь не используются смещения. Конкурирующая функция активации возвращает 1 для элемента выхода $a1$, соответствующего победившему нейрону, все другие элементы вектора $a1$ равны 0.

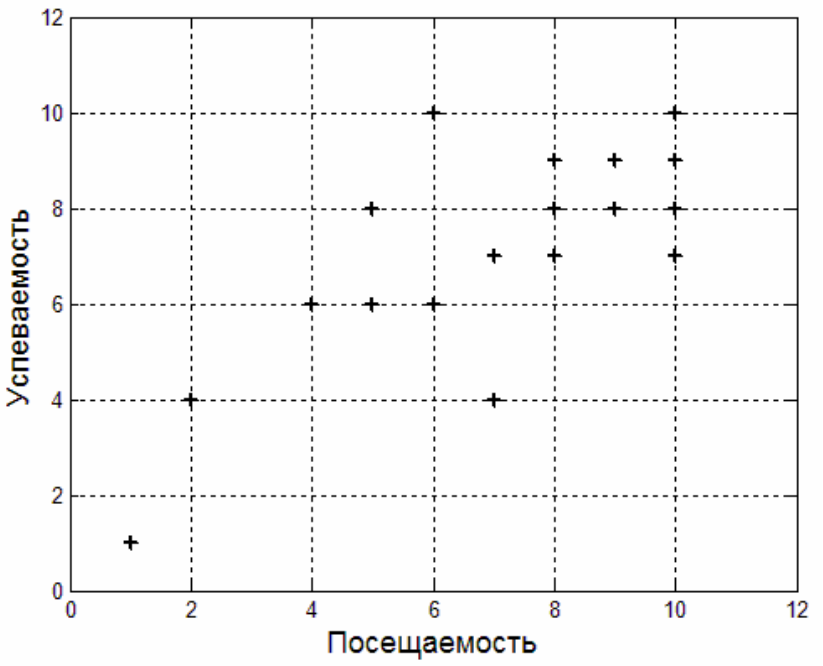


Рис. 4. Вид данных для двух размерностей

Проиллюстрируем построение нейронной сети на слое Кохонена на простейшем примере. Имеется список студентов учебной группы (рис. 3). Произведем графическое представление наших данных, рассматривая для примера проекцию на координатную плоскость «успеваемость – посещаемость» (рис. 4).

Таким образом, вектор параметров имеет размерность 2.

P = [6 9 9 10 10 7 7 10 6 9 5 10 8 5 4 8 2 1 8 ;…
6 8 9 9 10 4 7 8 10 9 6 7 9 8 6 8 4 1 7];

Сформируем слой Кохонена с 3 нейронами, потому что мы хотим создать три кластера, и диапазоном значений 0 – 10:

Net = newc([0 10;0 10], 3);

Выбор количества кластеров зависит от характера исходных данных (степени обособленности отдельных подмножеств) и может определяться опытным путем.

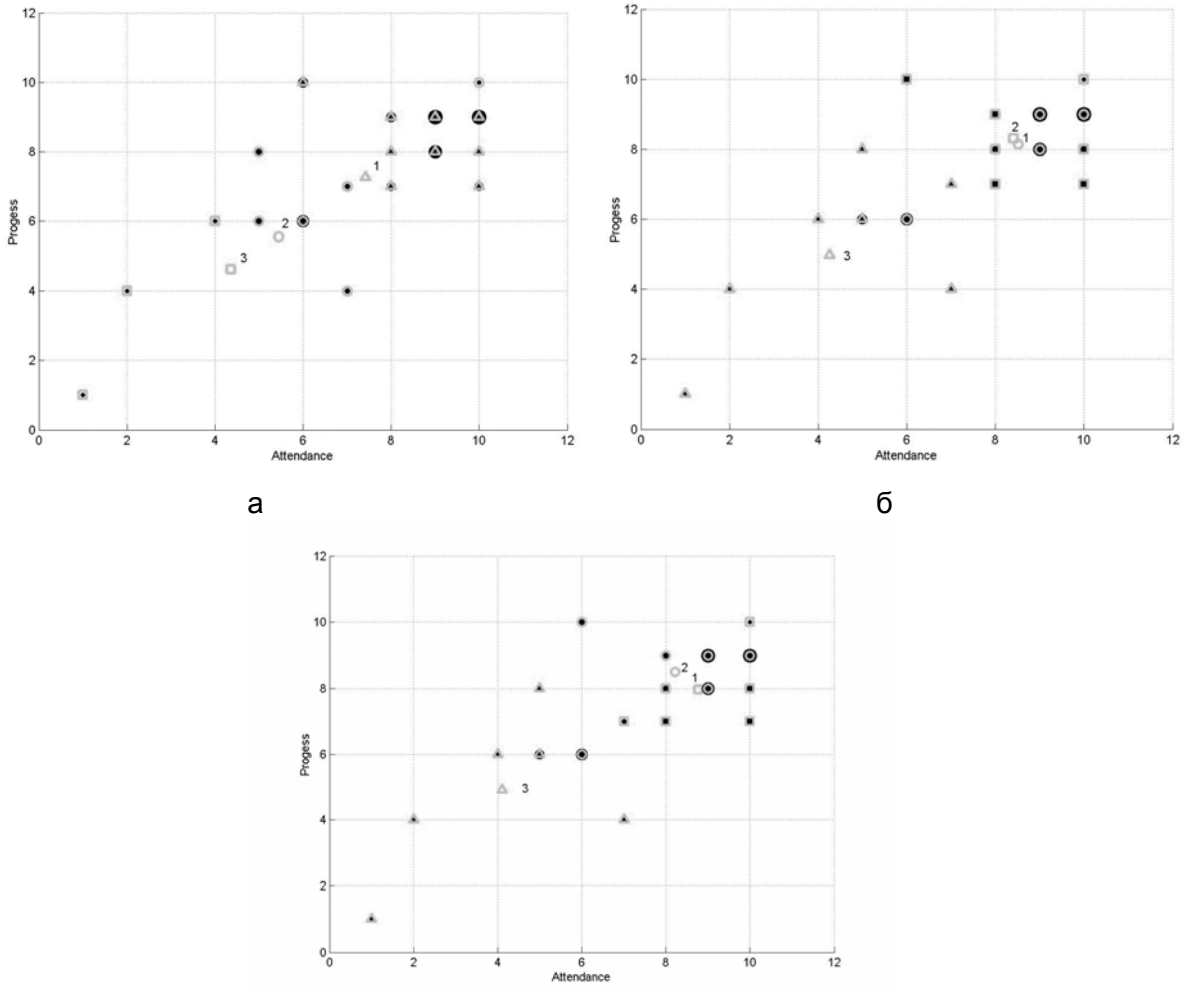
Теперь, когда сформирована самоорганизующаяся нейронная сеть, требуется обучить ее решению задачи кластеризации данных. Алгоритм обучения слоя Кохонена заключается в том, чтобы настроить нужным образом элементы матрицы весов. Он представляет собой рекуррентную операцию, обеспечивающую коррекцию строки матрицы весов добавлением взвешенной разности вектора входа и значения строки на предыдущем шаге.

Таким образом, вектор веса, наиболее близкий к вектору входа, модифицируется так, чтобы расстояние между ними стало еще меньше. Результат обучения будет заключаться в том, что «победивший» нейрон, вероятно, выиграет конкуренцию и в том случае, когда будет представлен новый входной вектор, близкий к предыдущему, и его победа менее вероятна, когда будет представлен вектор, существенно отличающийся от предыдущего.

Задавая различные значения для числа эпох обучения сети, мы можем видеть результат обработки входных данных.

```
Net.trainParam.epoch =20
Net = train(Net, P)
```

Результаты кластеризации для 20, 60 и 100 эпох обучения приведены на рис. 5. Из результатов видно, что после 60 эпох обучения кластеры и их центры фиксируются, чего нельзя сказать о более ранних стадиях.



а б

в

Рис 5. Этапы обучения сети: а – 10, б – 60 и в –100 эпох обучения

Этим самым мы провели наглядную иллюстрацию этапов развития и обучения нейронной сети. Аналогичным образом были построены кластера по всем параметрам. В результате на входе получили вектора из четырехмерного пространства. Для представления четырехмерного пространства в качестве четвертой координаты (склонность к лидерству) выступает радиус «центра точки». Внешний вид точки определяет принадлежность точки тому или иному кластеру – треугольник, квадрат или круг.

Применительно к рассматриваемому примеру произошла группировка векторов в кластеры (рис. 6), которые можно условно характеризовать следующим образом:

- умные и старательные (кластер обозначен кружками);
- умные, но необязательные (кластер обозначен квадратами);
- недалекие и необязательные (кластер обозначен треугольниками)

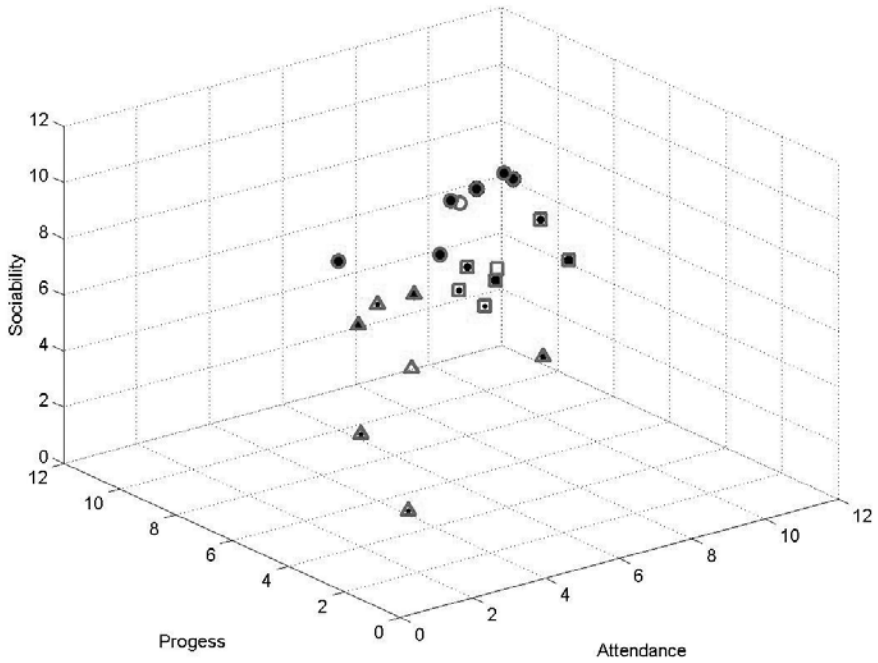


Рис. 6. Результат работы нейронной сети по кластеризации данных

Заключение

Какую пользу можно извлечь из полученных результатов для наших целей? Во-первых, все множество объектов автоматически разбито на обособленные группы. Остается только их идентифицировать и интерпретировать. Во-вторых, можно построить графический интерфейс для выборки данных, задавая область на графике, которую мы хотим охватить, например, окружность вокруг центра кластера «умные и старательные». При этом мы выбираем не самых умных и самых дисциплинированных, а самых типичных представителей данной группы. Трудно даже представить, как будет выглядеть SQL-запрос на такую выборку. В-третьих, неоднократное обращение к одним и тем же данным с помощью графического отображения способствует запоминанию их размещения на графике, упрощая последующие поиски.

Таким образом, ассоциативный доступ к данным с помощью обученной нейронной сети Кохонена лежит в области графических ассоциаций.

Приведенный пример показал, что нейронные сети дают возможность производить ассоциативный поиск в массиве неупорядоченной информации на совершенно новых принципах. В результате обучения нейронной сети весь объем данных разбивается на кластеры, содержащие связанные между собой по тем или иным признакам единицы информации, что в значительной степени упрощает поиск и увеличивает наглядность базы.

Литература

1. Громов Г.Р. Национальные информационные ресурсы: Проблемы промышленной эксплуатации. М.: Наука, 1985.
2. Глушаков С.В., Ломотько Д.В. Базы данных: учебный курс. М.: ООО «Издательство АСТ», 2000.
3. Медведев В.С., Потемкин В.Г. Нейронные сети. Matlab 6. М: «Диалог-МИФИ», 2002.
4. Шумский С.А., Яровой А.В., Зорин О.Л. Ассоциативный поиск текстовой информации». http://www.neurok.ru/

5. Липинский Ю.В. Средства информационного поиска и навигации в больших массивах неструктурированной информации. http://www.metric.ru/
6. The Spider's Apprentice. http://monash.com/spidap.html
7. Лаборатория BaseGroup. Ассоциативная память. Нейронные сети как средство добычи данных. Практическое применение нейронных сетей для задач классификации (кластеризации). http://www.basegroup.ru/

РАЗВИТИЕ АЛГОРИТМОВ СЖАТИЯ РЕЧИ

М.Б. Шалаева

В статье изложены следующие этапы исследования: сравнение современных алгоритмов кодирования речи по различным показателям, выявление тенденций развития и определения наиболее перспективных методов, выбор общих для большинства алгоритмов функциональных блоков с целью их последующей модернизации.

Введение

В настоящее время большую популярность приобрела компьютерная телефония – технология, основанная на передаче голосовых сообщений через сети, изначально предназначенные для трафика данных. Ее достоинством является снижение затрат на междугородние и международные переговоры для многих компаний и частных лиц, недостатком – относительно низкое качество синтезированной речи по сравнению с традиционными телефонными сетями общего пользования (ТфОП).

Организация этапов преобразования оказывает существенное влияние на неизбежные задержки передачи и чувствительность к потерям речевых пакетов. Совершенствование алгоритмов кодирования, наряду с сетевыми политиками, обеспечивает конкурентоспособность современных телекоммуникационных технологий.

Основная часть

Технологии преобразования речи можно разделить на две группы [1]:

- аппроксимация (кодирование) формы речевой волны;
- параметрическое компандирование речи (вокодерные преобразования).

Кодеры формы волны аппроксимируют изменение сигнала во времени. Они требуют наибольших скоростей передачи, но имеют наилучшие показатели качества воспроизведенной речи.

При параметрическом компандировании моделируется процесс речеобразования человека. В кодере из речевого сигнала вычисляются определенные параметры, передаваемые к декодеру, в котором они применяются для восстановления формы исходного сигнала. Использование исключительно параметрических методов приводит к потере натуральности звучания голоса и большой чувствительности к фоновым шумам. Вокодерные преобразования отличаются наименьшими требованиями к полосе пропускания.

Один из способов повышения эффективности использования полосы пропускания состоит в применении гибридных методов, основанных на принципах линейного предсказания и объединяющих параметрическое компандирование и кодирование формы волны. Большинство гибридных кодеров используют замкнутое кодирование (метод «анализ через синтез») на передающей стороне, что позволяет подкорректировать определенные параметры посредством сравнения результата синтеза с оригиналом. Это, безусловно, увеличивает время обработки, но обеспечивает лучшие показатели при передаче.

В табл. 1 представлены наиболее распространенные алгоритмы и области их применения [2, 3, 5–11]. Алгоритмы указаны в порядке убывания битовой скорости потока. В таблице приняты следующие сокращения:

ACELP (Algebraic Code Excited Linear Prediction) – линейное предсказание с алгебраическим возбуждением;

ADPCM (Adaptive Differential Pulse Code Modulation) – адаптивная дифференциальная импульсно-кодовая модуляция;

CS-ACELP (Conjugate Structure Algebraic Code Excited Linear Prediction) – линейное предсказание сопряженной структуры с алгебраическим возбуждением;
LD-CELP (Low Delay Code Excited Linear Prediction) – линейное предсказание с кодовым возбуждением и малой задержкой;
LPC (Linear Predictive Coding) – кодирование на основе линейного предсказания;
MP-MLQ (Multi Pulse Maximum Likelihood Quantization) – метод квантования по максимуму правдоподобия;
PCM (Pulse Code Modulation) – импульсно-кодовая модуляция;
RPE-LTP-LPC (Regular Pulse Excitation Long Time Prediction Linear Predictive Coding) – кодирование на основе линейного предсказания с долговременным предсказанием с регулярным импульсным возбуждением;
SB-ADPCM (Sub-Band Adaptive Differential Pulse Code Modulation) – адаптивная дифференциальная импульсно-кодовая модуляция с делением на поддиапазоны;
VSELP (Vector Sum Excited Linear Prediction) – линейное предсказание с векторным возбуждением.

Алгоритм	Скорость, кбит/с	Стандарт	Год	Приложение
Аппроксимация формы речевой волны				
PCM	64, 56, 48	ITU-T G.711	1960	Общественные телефоны
SB-ADPCM	64, 56, 48	ITU-T G.722	1986	Передача широкополосных сигналов
ADPCM	32	ITU-T G.721	1984	Общественные телефоны
ADPCM	40, 32, 24, 16	ITU-T G.726	1984	Общественные и цифровые беспроводные телефоны
Гибридные методы кодирования				
LD-CELP	16	ITU-T G.728	1992	Общественные телефоны, видеотелефоны
RPE-LTP-LPC	13	ETSI GSM 06.10	1992	Европейские цифровые сотовые системы
CS-ACELP	11.8, 8, 6.4	ITU-T G.729, G.729 Annex A	1997	Передача речи в сетях Frame Relay, ATM, в системах телесвязи Франции
MP-MLQ	6.3	ITU-T G.723.1	1996	Передача речи в видеотелефонии
VSELP	5.6	ETSI GSM 06.20		Европейские цифровые сотовые системы
ACELP	5.3	ITU-T G.723	1996	Передача речи в видеотелефонии
Вокодерные преобразования				
LPC-10	2.4	ANSI		Специальные системы

Таблица 1. Алгоритмы кодирования речи

Как правило, определяющими для выбора метода кодирования являются такие показатели, как:

- качество голоса по пятибалльной шкале экспертных оценок MOS (Mean Opinion Score, Рекомендация ITU-T P.800);
- задержка алгоритма;

- помехоустойчивость;
- степень ухудшения качества сигнала при квантовании QDU (Quantization Distortion Units);
- распространенность, поддержка производителями оборудования и др.

Одна из задач исследования состояла в выявлении зависимостей качества речи, задержек алгоритмов кодирования и других показателей от пропускной способности.

В табл. 2 приведены данные по соответствию качества речи, MOS, задержек передачи и типов каналов, удовлетворяющих предъявленным требованиям [4, 12, 13].

Качество	Лучшее	Хорошее	Среднее	Плохое	Стандарт
MOS	> 4.5	4–4.5	3.5–4	3–3.5	ITU-T P.800, P.830
Задержка, мс	< 150	< 250	< 350	< 450	ETSI TS 101 329
	< 150	< 260	< 400	> 400	ITU-T G.114
Тип канала	ТфОП	Спутниковый	ТфОП + спутниковый	Допустимо для VoIP	

Таблица 2. Соответствия MOS, задержек передачи и типов каналов

На рис. 1 изображены сглаженные зависимости оценок MOS от требований к битовой скорости потока, построенные автором статьи по усредненным результатам исследований ITU Study Group 15 и данным Р.Кокса (IEEE Communications Magazine, сентябрь 1997 г.)

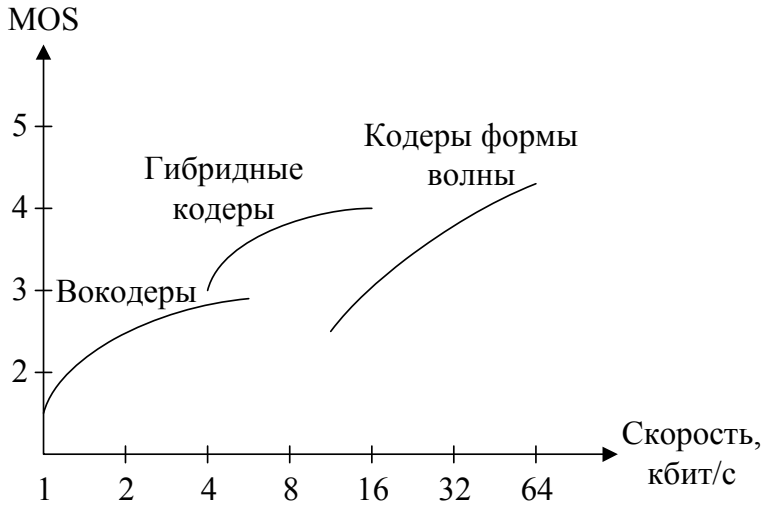


Рис. 1. Зависимость оценок MOS от скорости потока для кодеров формы волны, вокодеров и гибридных кодеров

Следует отметить, что значения MOS можно встретить во многих информационных источниках, при этом отклонения составляют не более 0.4 балла, что допустимо, поскольку «хорошая» или «плохая» связь– это субъективная оценка, зависящая от ожиданий абонентов, их капиталовложений и других факторов.

Указанные в табл. 2 значения задержек следует считать ориентировочными, можно встретить и другие, особенно у провайдеров.

На количественный показатель задержки оказывают воздействие [1]:

- алгоритмы кодирования/декодирования информации;
- сеть;
- операционная система;
- буфер устранения джиттера.

Следует отметить, что только среда передачи в среднем задерживает сигнал на 10–150 мс в зависимости от длины и типа каналов связи.

Как правило, более сложные алгоритмы кодирования обеспечивают лучшее сжатие при практически неизменном качестве речи. Становится очевидным, что для уменьшения битовой скорости, а, следовательно, и составляющих сетевой задержки, неизбежно увеличение задержки алгоритмов кодирования.

Безусловно, временная задержка кодирования зависит от быстродействия устройства, выполняющего преобразование. Поэтому представленные ниже графики, по мнению автора статьи, правильнее рассматривать с точки зрения относительных, а не абсолютных значений.

На рис. 2 изображены сглаженные зависимости общих задержек алгоритмов от битовой скорости потока. Численные значения задержек взяты из описаний рекомендаций ITU-T, ETSI и др., размещенных на сайтах www.axenet.ru, www.vocal.com.

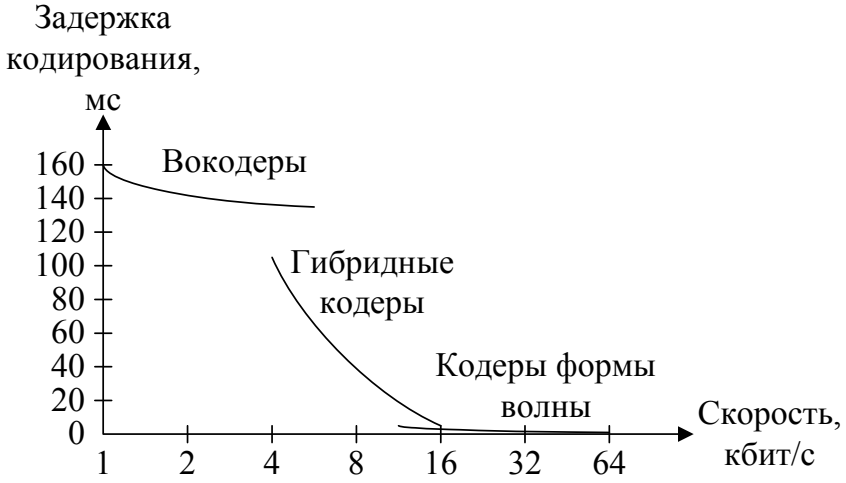


Рис. 2. Зависимость задержки кодирования от скорости потока для кодеров формы волны, вокодеров и гибридных кодеров

Задержки декодирования могут существенно изменяться в зависимости от организации буфера устранения джиттера.

Помехоустойчивость. Максимальное значение – 10 баллов. Значения для разных типов преобразований: аппроксимация формы речевой волны – 8–10 баллов, гибридные методы кодирования – 2–4 балла, вокодерные преобразования – 1 балл.

Степень ухудшения качества сигнала при квантовании. Один QDU соответствует ухудшению качества при оцифровке с использованием стандартной процедуры PCM. Согласно рекомендациям ITU-T, для международных вызовов величина QDU не должна превышать 14. Следует отметить, что передача разговора по международным магистральным каналам ухудшает качество речи, как правило, на 4 QDU. При передаче разговора по национальным сетям должно теряться не более 5 QDU. Значения QDU для некоторых алгоритмов: ADPCM (32кбит/с), LD-CELP (16кбит/с) и CS-ACELP (8кбит/с) – 3.5, ADPCM (24кбит/с) – 7. Следовательно, для качественной передачи речи процедуру компрессии/декомпрессии желательно применять в сети только один раз. В некоторых странах это является обязательным требованием регулирующих органов, предъявляемым к сетям, подключенным к ТфОП.

В результате обзора услуг провайдеров IP-телефонии можно сделать вывод, что в настоящее время наибольшую популярность приобрели алгоритмы MP-MLQ и CS-ACELP, выполненные по стандартам ITU-T G.723.1 и G.729 Annex A, соответственно.

Вторая задача исследования состояла в выявлении тенденций развития алгоритмов для определения наиболее перспективных методов. Анализ последних разработок

показал, что в первую очередь учитываются скорость алгоритма и оценка качества речи, причем именно в указанной последовательности.

Минимизация скорости привела к появлению методов, основанных на интерполяции спектрально-временных алгоритмов параметрического компандирования [1]. Но большинство разработок ведется в области гибридных методов. В последних разработках кодеров применяются:

- алгоритмы долговременного и кратковременного предсказания;
- кодовые книги, хранящие различные виды сигналов возбуждения;
- подавление пауз, которые обычно занимают до 60% длительности разговора;
- переменная скорость кодирования, учитывающая:
  - разделение сегментов речевого сигнала на основе фонетической или энергетической классификации;
  - возможность применения различных систем кодирования на разных сегментах;
- настройка на говорящего абонента.

Кроме приведенных выше выводов, была поставлена третья задача исследования, которая заключалась в выборе общих для большинства алгоритмов функциональных блоков с целью их последующей модернизации. Примеры:

- блок предсказания;
- блок спектрально-временного преобразования.

Последний блок представляет наибольший интерес, поскольку используется для решения различных задач. Спектральное представление сигнала позволяет:

- уменьшить объем передаваемых данных;
- осуществить фильтрацию – селекцию желаемой полосы частот в обрабатываемом сигнале:
  - подавлять шумы обнулением компонент на нежелательных частотах;
  - выделить достаточную полосу частот для воспроизведения разборчивой речи и особенностей (тембра) говорящего, т.е. содержащую три первых формантных частоты (как правило, используется полоса частот от 300 до 3400 Гц) [1];
- выполнить классификацию сегмента сигнала (вокализованный, невокализованный глухой, невокализованный фрикативный звук, шум) [1] и др.

Заключение

В результате были выполнены:

- сравнение современных алгоритмов кодирования речи по различным показателям;
- выявление тенденций развития и определение наиболее перспективных методов;
- выбор общих для большинства алгоритмов кодирования речи функциональных блоков с целью их последующей модернизации.

Результаты исследования послужили основой для модернизации блоков спектрально-временного преобразования, программные модели которых на настоящий момент разработаны в среде Matlab.

Литература

1. Быков С.Ф., Журавлев В.И., Шалимов И.А., Цифровая телефония, учебное пособие для ВУЗов. М.: Радио и связь, 2003.
2. ETSI Recommendation GSM 06.10. Full rate (FR) vocoder regular pulse excitation – long term prediction linear predictive coder (RPE-LTP), 1992.
3. ETSI Recommendation GSM 06.20. Half rate (HR) vocoder vector-sum excited linear prediction (VSELP), 1996.
4. ITU-T Recommendation G.114. One-way transmission time, 1996.

5. ITU-T Recommendation G.711. Pulse code modulation (PCM) of voice frequencies, 1988.
6. ITU-T Recommendation G.722. 7 kHz audio-coding within 64 kbit/s, 1988.
7. ITU-T Recommendation G.723.1. Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s, 1996.
8. ITU-T Recommendation G.726. 40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM), 1990.
9. ITU-T Recommendation G.728. Coding of speech at 16 kbit/s using low-delay code excited linear prediction, 1992.
10. ITU-T Recommendation G.729. Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction, 1996.
11. ITU-T Recommendation G.729 Annex A. Reduced complexity 8 kbit/s CS-ACELP speech codec, 1996.
12. ITU-T Recommendation P.800. Methods for subjective determination of transmission quality, 1996.
13. ITU-T Recommendation P.830. Subjective performance assessment of telephone-band and wideband digital codecs, 1996.

ОЦЕНКА КАЧЕСТВА РАБОТЫ МНОГОУРОВНЕВОЙ VOIP-СЕТИ

М.Ю. Будько

В настоящей статье рассматривается проблема оценки качества работы сети IP-телефонии, состоящей из нескольких независимых участков. В этом случае невозможно с помощью традиционных способов определить качество предоставляемых услуг. Для решения этой проблемы предлагается метод, основанный на анализе статистики состояний вызовов.

Введение

Все существующие способы оценки качества передачи речи применимы в том случае, когда можно провести измерения параметров линий связи. Их цель состоит в том, чтобы выявить влияние таких факторов, как задержка, джиттер, потеря пакетов, искажение кодеками и т.д. на качество голоса [1–5]. Основываясь на этих данных, можно определить уровень обслуживания клиентов. Однако, когда рассматривается многоуровневая Voice over IP (VoIP) сеть, ситуация меняется. В ней невозможно оценить качество каналов, по которым передается голосовая информация. Каждый оператор, предоставляющий свои услуги в рамках объединенной VoIP-сети, не распространяет информацию о своей структуре и декларирует хорошее качество связи по всем направлениям. На практике оказывается совсем не так: по одним направлениям указанный оператор работает хорошо, а по другим плохо. Таким образом, возникает задача оценки качества работы оператора по различным направлениям. В результате этого все направления должны быть поделены между различными компаниями с целью улучшения качества связи и минимизации расходов на ее осуществление.

Для решения поставленной задачи предлагается метод, основанный на использовании статистики состояний вызовов в объединенной сети. Его особенностью является то, что он не требует проведение экспериментов по определению технического состояния линий связи и, следовательно, может использоваться в объединенной сети. Источником информации, на основе которого принимается решение о качестве предоставляемых услуг, является история прохождения вызовов через заданного оператора. Полученные сведения анализируются с учетом:

- организации многоуровневой сети;
- особенностей работы протоколов ответственных за регистрацию пользователя, установку соединения и передачу информации;
- опытных и аналитических данных о зависимостях в работе сети.

Структура и алгоритм работы объединенной сети

Упрощенная структура рассматриваемой VoIP сети представлена на рис. 1.

Алгоритм работы состоит в следующем.

1. На узле Gatekeeper находится таблица маршрутизации вызовов. Каждая строка таблицы указывает на отдельное направление. В строке есть три записи, определяющие операторов, через которых может быть установлено соединение.
2. При звонке узел A соединяется с узлом GK и передает запрос на вызов узла B.
3. Узел GK передает запрос оператору, указанному в таблице маршрутизации для вызываемого направления. Если оператор отказывается принять вызов, он перенаправляется следующему оператору.
4. Если ни один из трех операторов вызов не принимает, узел A получает соответствующее уведомление.

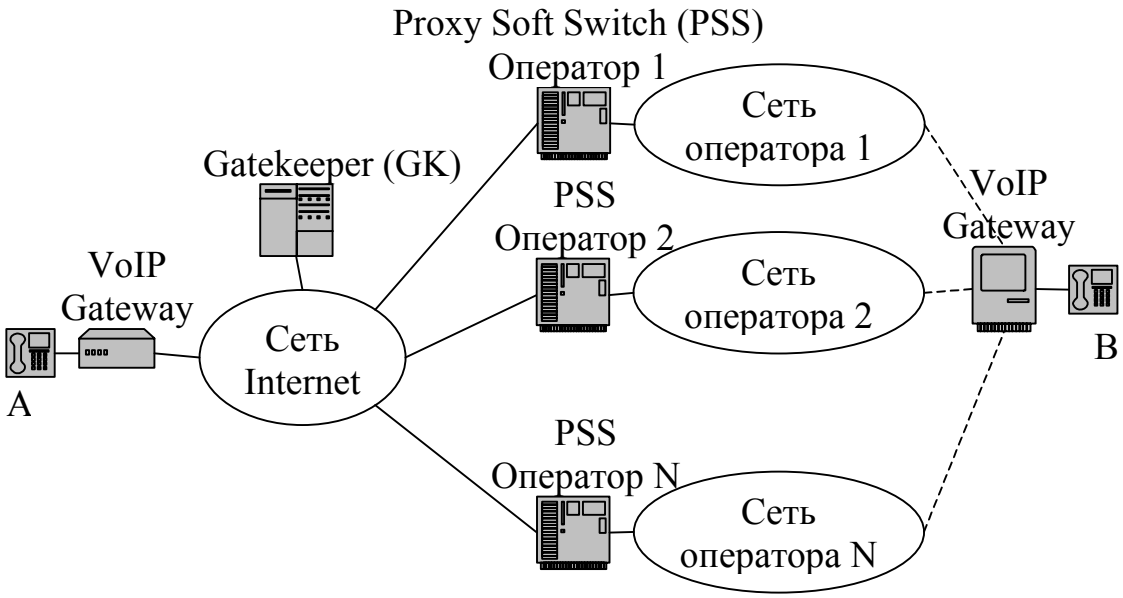


Рис. 1. Упрощенная структура рассматриваемой сети IP-телефонии

Постановка задачи исследования

Подобная схема призвана обеспечить высокий уровень обслуживания пользователей за счет автоматического выбора оператора телефонии. Основной задачей, требующей решения в этом случае, является определение того, какой из имеющихся операторов сможет предоставить лучшее качество связи при минимальной стоимости. Сложность состоит в том, что не существует оператора, одинаково хорошо работающего по всем направлениям. Следовательно, для каждого направления необходимо определять отдельный набор операторов, обеспечивающих заданное качество. Так как направлений очень много, выполнить это требование не просто. Более того, при добавлении нового оператора или изменении структуры существующего может потребоваться пересмотр всей таблицы маршрутизации. Таким образом, необходимо создание системы, способной:

- динамически изменять содержимое таблицы маршрутизации;
- автоматически подбирать оптимальную комбинацию операторов для каждого направления;
- оперативно реагировать на изменение ситуации в сети для обеспечения заданного качества обслуживания;
- учитывать при выборе такие критерии как цена, качество.

Подходы к реализации поставленной задачи

Точность работы разрабатываемого алгоритма зависит от количества показателей, которые в нем учитываются. В первом приближении можно руководствоваться всего одним критерием – длительностью соединения. Таким образом, все звонки можно разделить на две группы:

- звонки с нулевой длительностью;
- звонки с длительностью больше нуля.

На основе такой классификации можно отсечь большинство некорректных комбинаций направлений и операторов. Для достижения большей точности можно анализировать длительность разговора. Соединения с малой продолжительностью, порядка нескольких секунд, могут свидетельствовать о том, что собеседники по каким-либо

причинам были вынуждены прервать разговор в самом начале. Последующий вызов по этому же номеру будет являться только подтверждением некачественной связи на предыдущем этапе. Разрыв связи может происходить по двум причинам:

- завершение соединения, инициируемое его участниками;
- завершение соединения, вызванное оператором или техническими проблемами на линии связи.

Второй случай также будет свидетельствовать о недостатках используемого оператора. Отследить такую ситуацию можно только с помощью кодов завершения вызовов. Использование кодов завершения для определения степени адекватности таблицы маршрутизации может еще больше повысить точность анализа. Применение такого подхода осложняется тем, что различные операторы могут определять одни и те же коды завершений вызовов для различных ситуаций – например, сообщать о том, что телефон занят, в ситуациях, когда все линии, выделенные оператором, заняты или он не поддерживает это направление. Таким образом, для осуществления более точной оценки необходимо составление таблицы соответствия кодов завершения и ситуаций, возникающих в сети. Это можно осуществить только при тесном взаимодействии со службами технической поддержки операторов, что, к сожалению, не всегда возможно.

В итоге можно построить классификацию вызовов, изображенную на рис. 2.

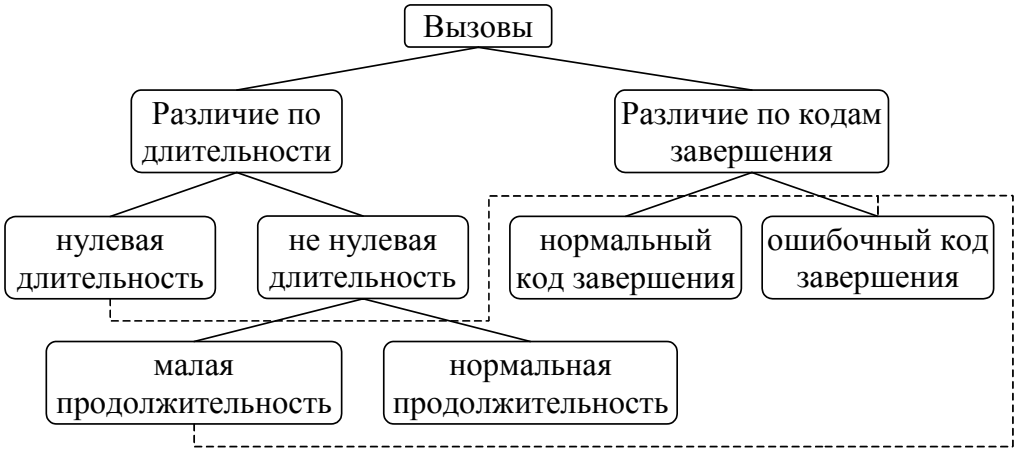


Рис. 2. Классификация вызовов

С помощью приведенной классификации можно выявить проблемные соединения, определить операторов, которые в них участвовали, и внести необходимые изменения в таблицу. При изменении маршрутизации также необходимо учитывать такой критерий, как стоимость, поэтому желательно иметь выбор между несколькими операторами.

Оценка качества работы сети на основе анализа длительности вызовов

Основные этапы анализа состояния вызовов, на основе которых изменяется таблица маршрутизации, представлены на рис. 3.

Основным в приведенной схеме является блок, оценивающий качество работы операторов в каждом направлении. На текущем этапе исследований для его реализации предлагается использовать следующие положения.

1. Оценка качества работы оператора в конкретном направлении осуществляется путем определения доли вызовов с нулевой продолжительностью по отношению к их общему числу.
2. Соединения с нулевой длительностью, имеющие код завершения 17 «Абонент занят» со стороны оператора, относятся к вызовам с положительной продолжительностью.

3. Для учета относительного изменения качества работы оператора с течением времени в памяти сохраняются результаты двух последних измерений.

Решение об изменении таблицы маршрутизации принимается в случае, если:

- доля ошибочных попыток у оператора с высоким приоритетом больше, чем у оператора с низким приоритетом;
- сумма последних трех измерений у оператора с высоким приоритетом больше, чем у остальных.

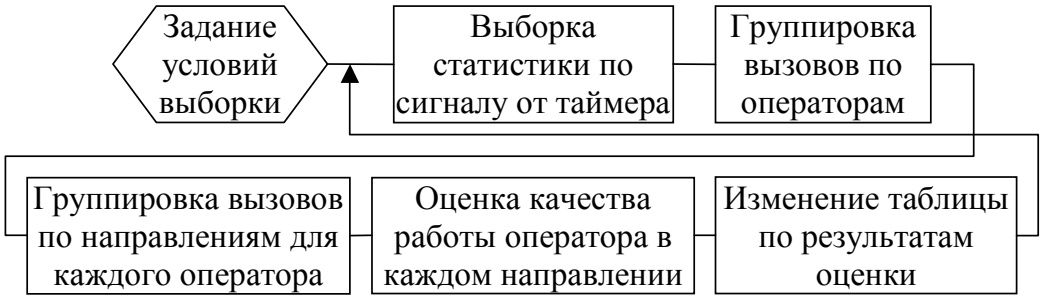


Рис. 3. Этапы анализа статистики вызовов

В указанных ситуациях осуществляется изменение приоритетов в соответствии с полученными данными. Решение о замене одного из трех операторов принимается, если сумма измерений по всем трем операторам в определенном направлении превышает единицу. Это значение свидетельствует о том, что в идеальной ситуации, когда два оператора работают без ошибок, третий простаивает.

Заключение

Предлагаемый метод статистического анализа состояния вызовов может получить широкое практическое распространение, так как в условиях повышающейся конкуренции на рынке IP-телефонии телекоммуникационные компании стремятся обеспечить лучшее качество своих услуг связи. Выше приводится один из примеров реализации предлагаемого подхода, однако уже на этом этапе он может быть использован в работе реальной сети. Задачей дальнейших исследований будет определение достоверности выбранных критериев оценки и доработка алгоритма с целью увеличения точности его работы и учета максимального количества показателей.

Литература

1. TIA, “Telecommunications IP Telephony Equipment, Voice Quality Recommendations for IP Telephony”, Арлингтон, TIA Standards and Technology Department, 2001.
2. ITU-T, “Recommendation P.861, Objective quality measurement of telephone-band (300 - 3400 Hz) speech codecs”, Женева, ITU, 1996.
3. ITU-T, “Recommendation P.11, Effect of transmission impairments”, Хельсинки, ITU, 1993.
4. ITU-T, “Recommendation G.113, Transmission impairments”, Хельсинки, ITU, 1996.
5. ITU-T, “Recommendation G.114, One-way transmission time”, Хельсинки, ITU, 1996.
6. Айдарханов Д.М. Программный коммутатор SoftSwitch в сетях IP-телефонии // Мобильные системы. 2002. №12.

ИНСТРУМЕНТИРОВАНИЕ КЛИЕНТ-СЕРВЕРНЫХ ПРИЛОЖЕНИЙ

А.В. Гирик, Б.Д. Тимченко

В статье рассматриваются современные подходы к инструментированию приложений и описывается реализация средства измерения полного времени ответа для браузера MS Internet Explorer.

Большинство современных информационных систем являются распределенными гетерогенными системами клиент-серверной архитектуры. В самом общем виде клиент-серверная информационная система может быть представлена трехкомпонентной моделью: клиентом, обобщенным сервером и соединяющей их сетью. Полное время прохождения запроса в такой системе будет суммой времен клиентской и серверной обработки и времени передачи запроса по сети. На рис. 1 приведен более детализированный пример модели многоэтапного прохождения запроса и показаны компоненты, участвующие в его обработке. Полное время прохождения запроса через все участки пути от клиента к серверу и его полной обработки всеми компонентами сервера состоит из частей, продолжительность выполнения которых приходится измерять различными методами и средствами [1], причем отдельные компоненты на пути прохождения транзакции могут быть недоступны для наблюдения и измерений.

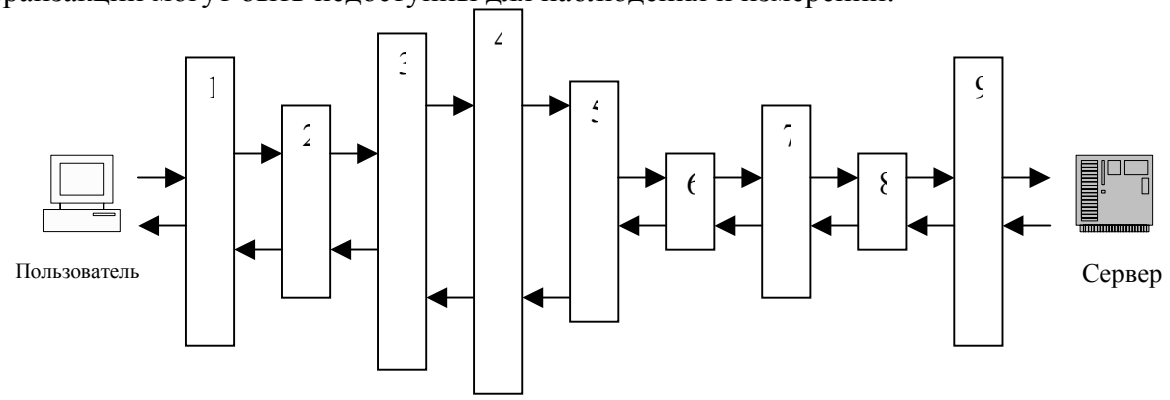


Рис. 1. Этапы обработки запроса в гетерогенной клиент-серверной системе:
1 – локальная сеть пользователя, 2 – корпоративный брандмауэр пользователя, 3 – корпоративный интернет-провайдер пользователя, 4 – Internet-магистраль, 5 – интернет-провайдер сайта, 6 – брандмауэр сайта, 7 – веб-сервер сайта, 8 – сервер приложений, 9 – сервер баз данных

Клиент-серверное взаимодействие имеет транзакционный характер, однако термин «транзакция» требует уточнения. В контексте информационных систем он определён достаточно строго [1, 2]. Информационные транзакции (*IT-transaction*) – операции обработки данных, которые выполняются СУБД или другими компонентами информационной системы. IT-транзакции обладают свойствами атомарности, непротиворечивости, изолированности и сохранности данных [3]. В более широком смысле термин «транзакция» используется в области управления бизнес-процессами, где транзакция – это логическая единица работы с точки зрения пользователя, некоторое законченное действие, выполняемое пользователем с помощью вычислительной системы. Пример бизнес-транзакции – процедура оформления заказа в системе резервирования билетов. IT-транзакции информационно обеспечивают бизнес-транзакции. Понятие бизнес-транзакции вводится для удобства оценки производительности бизнеса, в то время как понятие IT-транзакции – для оценки производительности информационной системы.

Типовые действия пользователя, работающего с информационной системой, можно представить как последовательность IT-транзакций, своеобразный «критический путь» [4], который следует анализировать с целью оценки качества обслуживания в

системе. Промежуток времени от формирования запроса к удаленной системе до получения результатов и представления их пользователю называется полным временем ответа приложения [1]. Это один из наиболее важных и наглядных показателей, характеризующих качество обслуживания и производительность информационной системы.

Существует ряд коммерческих программных пакетов, использующих этот показатель для управления производительностью приложений, например, Application Performance Manager от Tivoli, OpenView или System Insight Manager от Hewlett-Packard, Application Response Option от Computer Associates и Application Service Level Management Tools (входят в состав Enterprise Manager 10g) от Oracle. Перечисленные выше продукты дороги и ориентированны на конкретные платформы, поэтому для клиент-серверных систем среднего масштаба ощущается недостаток доступных средств. Определение полного времени ответа прямыми измерениями не только сложно в гетерогенных средах, но также приводит к проблемам хранения и обработки больших объемов данных, как в случае трассировки, так и при выборочных отсчетах (сэмплировании). По этой причине подход к определению полного времени ответа приложения на основе прямых наблюдений и комплексирования данных, собранных с различных узлов обработки, в современных высокопроизводительных системах считается бесперспективным.

Весьма активно развивается другой фундаментальный подход – инструментирование приложений, суть которого состоит в создании наблюдаемых и управляемых приложений. Обычно под инструментированием понимают внедрение сервисного (или технологического) кода в код приложения с целью получения дополнительных сведений о работе приложения (в первую очередь о производительности). Особенно часто инструментирование применяется на этапе разработки и представляет собой встраивание измерительного кода в исполняемый модуль приложения. Однако этим возможности инструментирования не ограничиваются. Возможны два различных подхода к инструментированию приложений: внедрение вызовов инструментальных функций в исходные тексты программ и внедрение инструментального кода в готовую программу.

Первый подход предусматривает использование одного из стандартов инструментирования (таких, например, как ARM или AIC) и явное размещение программистом инструментального кода в тексте программы. Достоинства этого подхода – надежность и относительная простота – в полной мере проявляются в случае, если инструментирование программы планируется и осуществляется ещё на этапе разработки. Недостатки – необходимость доступа к исходному коду и повторной сборки программы. В рамках второго подхода предполагается либо явное подключение инструментального кода к готовой программе, например, с помощью механизма расширений, либо неявное внедрение кода, например, замена библиотек, перехват вызовов функций и т.д. Следует заметить, что инструментирование готовой программы часто чревато появлением трудно обнаруживаемых ошибок.

Рассмотрим более подробно первый подход на примере ARM (Application Response Measurement) API – открытого платформенно-независимого прикладного программного интерфейса для измерения продолжительности IT-транзакций в распределенных приложениях. В исходный код программы внедряются вызовы специальных инструментальных функций, отмечающих начало и конец транзакции. Вызовы функций ARM отслеживаются специальным агентом (*logging agent*) – программой, которая записывает информацию о вызовах в журнал [5, 6]. Первая версия ARM API (опубликована в 1996 году фирмами Hewlett-Packard и Tivoli) описывала шесть функций, предназначенных для определения времени выполнения некоторого участка кода в программе: *arm_init(), arm_getid(), arm_start(), arm_update(), arm_stop()* и *arm_end()*. Последняя версия содержит уже двадцать одну функцию. Присутствует возможность устанавливать взаимосвязь между транзакциями, отслеживать дочерние транзакции и пе-

редавать агенту ARM дополнительную информацию, связанную с транзакцией (использование ресурсов, специфические данные приложения и т.д.). Стандарт ARM 4.0 позволяет также учитывать особенности обработки транзакций в многопоточных приложениях.

В стандарте ARM предусмотрены функции мониторинга, но отсутствуют возможности контроля, что ограничивает полезность этого стандарта для организации автоматизированных откликов на нарушения уровня обслуживания. Computer Associates, IBM и BMC Software предложили альтернативный стандарт – Application Instrumentation and Control (AIC). В дальнейшем разработка этого стандарта также осуществлялась The Open Group. Версия 1.0 была опубликована в 1999 году. ARM API и AIC API имеют много общего, но последний, безусловно, шире и функциональнее: помимо набора системно-независимых управляющих вызовов он поддерживает подключение дополнительных компонент безопасности [7].

На практике часто приходится иметь дело с готовыми приложениями. Например, в ОС Solaris 10 для этих целей в состав инструментальных средств включен трассировщик DTrace [9]. DTrace позволяет производить модификацию ядра операционной системы и пользовательских процессов, чтобы можно было сохранять дополнительную информацию об интересующих событиях. Для этого используются зонды (*probes*) – места или события, с которыми можно сопоставить некоторый набор действий, например, сохранение метки времени или аргумента функции, доступ к данным в адресном пространстве приложения, вызов отладчика ядра и т.д. Для написания зондов DTrace имеет специальный язык программирования D, сходный по синтаксису с C. Он позволяет составлять скрипты, которые затем можно использовать для повторения записанных в них зондов и связанных с ними действий. Зонды предоставляются набором модулей ядра, называемых поставщиками (*providers*). К сожалению, применение DTrace возможно только в ОС Solaris 10.

Рассматриваемые вопросы инструментирования особенно актуальны для клиент-серверных приложений, ориентированных на использование web-интерфейсов и не привязанных к конкретной платформе. Необходимо найти результативный и приемлемый по стоимости способ определения полного времени ответа таких приложений. Полное время ответа приложения может быть временем загрузки определенной страницы (простейший случай), временем реакции на определенную последовательность действий пользователя на странице (например, от момента заполнения формы пользователем и отправки её на сервер и до получения результата), длительностью некоторой операции, в ходе которой пользователь работает с несколькими страницами, переходит от одной странице к другой и т.д. [1].

Для частичного решения проблемы определения полного времени ответа нами предложено инструментирование браузера на клиентской машине. Все популярные браузеры (MS IE, Netscape Navigator, Opera, FireFox, …) поддерживают механизм расширений, следовательно, инструментальный код может быть подключен явным способом – это самый предпочтительный вариант в случае инструментирования готового приложения. Помимо главной задачи – измерения полного времени ответа – инструментирование браузера позволяет определить также время обдумывания (*user think time*), представляющее собой промежуток времени между получением ответа от удаленной системы на предыдущий запрос (момент окончательной загрузки страницы) и формированием нового запроса.

Инструментальное расширение, загружаемое в адресное пространство браузера, получает возможность отслеживать действия пользователя на странице (работа с меню, элементами управления – списками, кнопками и т.д.) и анализировать html-код страницы (что может быть полезно, например, в случае, если страница содержит какие-либо статистические данные – время её формирования на сервере и др.).

Предложенный подход не создает дополнительной нагрузки, позволяет учесть при измерениях логику действий пользователя и легко организовать фильтрацию измерений (по параметрам запроса, например, по имени страницы или ip-адресу, по содержанию страницы, по времени и т.д.). Подход абсолютно «прозрачен» для пользователя (который может даже не подозревать о выполняющихся измерениях). К ограничениям подхода следует отнести специфичность архитектуры подключаемых модулей для каждого браузера, необходимость перекомпиляции исходных текстов модуля при любых изменениях логики работы пользователя, архитектуры расширений и, возможно, при переходе к новой версии браузера или операционной системы и невозможность определения составляющих полного времени ответа.

Предложенный нами подход к инструментированию был применен на практике для измерения полного времени ответа при обращениях к страницам интернет-сайта на базе Oracle Portal. Клиентский уровень содержит в себе средство для формирования запросов и отправки их на обработку порталу. Уровень бизнес-логики содержит сервер приложений Oracle. На этом уровне функционируют веб-сервер Oracle, веб-кэш, собственно Oracle Portal и другие компоненты сервера приложений (блоки 7 и 8 на рис. 1). Уровень доступа к данным включает СУБД Oracle и базы данных, в которых хранится информация, используемая Oracle Portal при динамическом формировании запрошенных пользователем страниц (блок 9 на рис. 1).

Для инструментирования браузера Internet Explorer был создан подключаемый модуль (*add-on*, или *надстройка* в терминах IE), отслеживающий внутренние события браузера и действия пользователя. Расширения для IE в документации Microsoft называются Browser Helper Objects (BHO). Место инструментального расширения в архитектуре браузера показано на рис. 2.

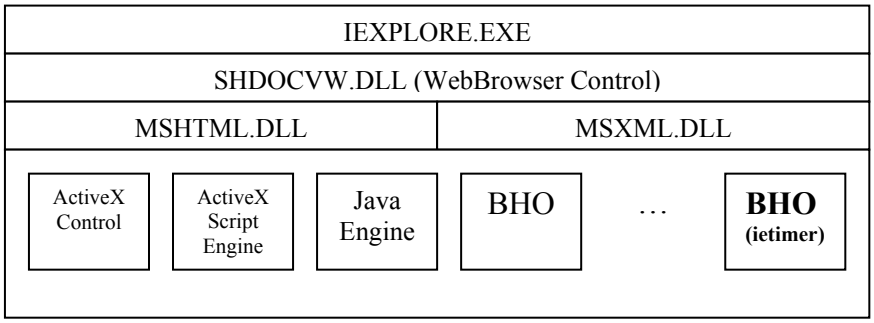


Рис. 2. Размещение инструментального расширения в браузере Internet Explorer

BHO-модуль представляет собой внутрипроцессный COM-сервер, содержащий объект, реализующий интерфейс IObjectWithSite. Для модуля, измеряющего длительность загрузки страниц, важен, прежде всего, интерфейс IDispatch, посредством которого браузер уведомляет модуль о событиях начала и окончания загрузки страницы, о возможных ошибках и некоторых других событиях. BHO реализуется в виде динамически подключаемой библиотеки.

Разработанный модуль ietimer создает отдельный журнал для каждого процесса iexplore.exe, в котором фиксируются временные метки начала и окончания запроса, адрес страницы и заголовок окна. Для измерения времени используется счетчик высокого разрешения, значение и частоту которого возвращают функции QueryPerformanceCounter() и QueryPerformanceFrequency(). В журнале также фиксируется информация об ошибках, возникающих при навигации. Модуль контролирует использование кэша IE при загрузке страниц и использует возможности механизма трассировки событий Windows, для чего создает собственного поставщика событий и генерирует события, связанные с действиями пользователя (начало и окончание запроса, ошибки и т.д.).

По классификации в [10] модуль относится к пассивным средствам мониторинга. Рассматривается возможность автоматизации модуля, т.е. превращение его в активное средство тестирования – переход по заданному множеству страниц будет осуществляться по таймеру. Подобная доработка целесообразна для облегчения сбора статистической информации. Также планируется добавить возможность настройки модуля – выбор сохраняемых данных и формат журнала, работа по расписанию и т.д.

В статье рассмотрены современные подходы к инструментированию приложений для определения полного времени ответа как альтернатива традиционным способам с применением мониторных средств. Ввиду широкого распространения web-сервисов особую важность приобретает измерение и анализ полного времени ответа приложений, использующих web-интерфейсы. В статье предложен подход к определению полного времени ответа при обращении пользователя к страницам корпоративного портала через web-интерфейс и представлена реализация для Windows-клиента, оснащенного браузером Internet Explorer, – подключаемый BHO-модуль ietimer.dll, предоставляющий базовую функциональность для измерения полного времени ответа. Главное достоинство предложенного подхода состоит в том, что инструментальное средство – в данном случае надстройка для браузера – работает с реальными запросами пользователей (т.е. с реальной нагрузкой) не на системном, а на прикладном уровне (т.е. позволяет избежать анализа низкоуровневых данных). Подход результативен и может быть развит для получения составляющих полного времени ответа приложения.

Литература

1. Norton T., End-to-End Response Time: Where to Measure? // Simalytic Solutions, 1999.
2. Norton T., End-To-End Scaling: The Response Time Pipe, CMG01, 2001.
3. Орлик С., Многоуровневые модели в архитектуре клиент-сервер. // Системы управления базами данных. 1997. №01.
4. Application Service Level Management With Enterprise Manager 10g, White Paper, 2004.
5. The Open Group, Application Response Measurement API 4.0, Technical Standard, 2003.
6. Ding Y., Performance Modeling with ARM, BGS Systems, 1997.
7. The Open Group, Application Instrumentation and Control API 1.0, Technical Standard, 1999.
8. Боркус В. Web-сервисы и транзакционные системы. // PC Week. 2004. №№ 27–32.
9. DTrace, User Manual, Sun Microsystems, 2003.
10. Yun Fu, Cherkasova Ludmila, Wenting Tang, Vahdat Amin. EtE: Passive End-to-End Internet Service Performance Monitoring, 2001.

ПРОТИВОДЕЙСТВИЕ РАСШИРЕНИЮ ПРИВИЛЕГИЙ ПУТЕМ КОНТРОЛЯ КОРРЕКТНОСТИ ОЛИЦЕТВОРЕНИЯ

С.Н. Сторожевых

В работе рассмотрены вопросы компьютерной безопасности, связанные с несанкционированным расширением привилегий, выделен класс угроз, основанных на уязвимости сервисов олицетворения, предложен подход к противодействию атакам на расширение привилегий, предложена реализация механизма контроля корректности олицетворения.

Введение

Для обеспечения корректности функционирования любой системы защиты основными условиями являются уникальная идентификация пользователей, регистрирующихся в системе, и блокирование несанкционированных попыток одних пользователей выполнять какие-либо действия от лица других (возможно обладающих более широким кругом привилегий). Другими словами, система защиты должна уметь противостоять атакам, направленным на расширение привилегий (процессу расширения возможностей текущей учетной записи пользователя до возможностей более привилегированной учетной записи, обычно суперпользователя, такой как учетная запись администратора или запись SYSTEM). После прохождения аутентификации расширение привилегий может дать пользователю возможность получить уровень прав, достаточный для осуществления несанкционированного доступа к конфиденциальным данным.

Анализ систем защиты рассматриваемых в данной статье операционных систем семейства Windows NT позволяет выявить их уязвимость к атакам такого рода. Рассматриваемые ОС предоставляют разработчикам приложений сервисы, некоторые из которых, при их некорректном использовании (ошибки в приложении), оказывают влияние на безопасность функционирования не только приложения, но и системы в целом. К подобным сервисам, например, относятся рассматриваемые в данной работе сервисы олицетворения. Олицетворение (impersonation) – средство, часто используемое в модели защиты Windows, предоставляющее возможность отдельному потоку выполняться в контексте защиты, отличном от контекста защиты процесса, т.е. действовать от лица другого пользователя. Уязвимость сервисов олицетворения привела к тому, что доля атак, направленных на расширение привилегий с использованием олицетворения, является наиболее заметной и существенно прогрессирующей в процентном отношении в последнее время [1, 2] (см. рис. 1).

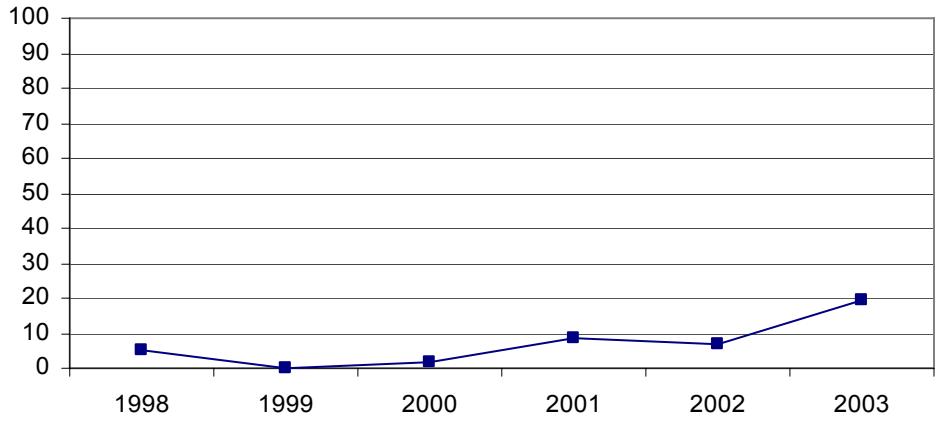


Рис. 1. Динамика изменения во времени процентной доли атак на сервисы олицетворения

В данной работе будет описан механизм действия некоторых атак, направленных на расширение привилегий, выделены общие черты их функционирования, а также предложен подход к противодействию таким атакам применительно к операционным системам семейства Windows NT.

Элементы системы безопасности ОС семейства Windows NT

Прежде чем приступить к детальному описанию процесса расширения привилегий, необходимо рассмотреть основные аспекты реализации некоторых фундаментальных механизмов защиты рассматриваемых операционных систем [3].

Для идентификации субъектов, выполняющих в системе различные действия, Windows использует так называемые идентификаторы защиты (security identifiers, SID). SID имеются у пользователей, локальных или доменных групп, локальных компьютеров, доменов и членов доменов. SID представляет собой уникальное числовое значение переменной длины, формируемое из номера версии структуры SID, 48-битного кода агента идентификатора и переменного количества 32-битных кодов субагентов. Такой подход позволяет системе различать, например, локальную учетную запись administrator и учетную запись с тем же именем с удаленной машины.

Все работающие в системе процессы и потоки выполняются в контексте защиты того пользователя, от имени которого они так или иначе были запущены. Для идентификации контекста защиты процесса или потока используется объект, называемый маркером доступа (access token). В контекст защиты входит информация, описывающая привилегии, учетные записи и группы, сопоставленные с процессом и потоком. В процессе регистрации в системе создается начальный маркер, представляющий пользователя, который входит в систему, и сопоставляет его с процессом оболочки, применяемой для регистрации пользователя. Все программы, запускаемые пользователем, наследуют копию этого маркера. Механизмы защиты в Windows используют маркер, определяя набор действий, разрешенных потоку или процессу (например, доступ к защищаемому объекту или привилегию на выключение компьютера).

Маркер может быть основным (идентифицирует контекст защиты процесса) или олицетворяющим (применяется для временного заимствования потоком другого контекста защиты – обычно другого пользователя).

Олицетворение (impersonation) – мощное средство, часто используемое в модели защиты Windows, предоставляющее возможность отдельному потоку выполняться в контексте защиты, отличном от контекста защиты процесса, т.е. действовать от лица другого пользователя. Олицетворение, например, применяется в модели программирования «клиент-сервер». При заимствовании прав сервер временно принимает профиль защиты клиента, который обращается к ресурсу. Тогда сервер может работать с ресурсом от имени клиента, а система защиты – проводить проверку его прав доступа. Обычно серверу доступен более широкий круг ресурсов, чем клиенту, и при олицетворении клиент может терять часть исходных прав доступа. И, напротив, при олицетворении сервер может получить дополнительные права. Как будет показано далее, сервисы олицетворения потенциально опасны и могут быть использованы для расширения привилегий.

Примеры реализации угроз расширения привилегий

Рассмотрим широко известные уязвимости системы защиты Windows, которые позволяют расширить привилегии, используя сервисы олицетворения (по материалам [4]).

Одна из возможностей для реализации атаки по расширению привилегий состоит в прогнозировании именованных каналов, используемых для служб диспетчера управления сервисами (Service Control Manager, SCM). Служба SCM в межпроцессной коммуникации использует прогнозируемые имена каналов для каждого запускаемого ей сервиса. Спрогнозировав имя следующего канала, можно создать такой канал до того, как канал с тем же именем будет создан службой SCM, и через SCM запустить любой сервис, который обязательно подключится к этому каналу. Теперь, воспользовавшись механизмом олицетворения через Win32-функцию *ImpersonateNamedPipeClient*, можно подменить свой профиль защиты контекстом безопасности запущенного сервиса (обычно SYSTEM).

Другой пример расширения привилегий основан на использовании сервера Internet Information Services (IIS). Процесс IIS выполняется с учетной записью SYSTEM и использует заимствование прав для обслуживания запросов. Вызов функции *ImpersonateSelf* в библиотеке ISAPI дает возможность выполнять команды под учетной записью SYSTEM. По сути, серверный поток присваивает себе маркер олицетворения просто как копию маркера доступа своего процесса.

Итак, на данный момент мы имеем два сценария расширения привилегий. Оба исправлены в последних пакетах обновлений, что, однако, не гарантирует невозможности появления подобных эксплоитов в дальнейшем, скорее, наоборот – есть основания полагать, что описанные механизмы олицетворения будут использованы в будущем при проведении атак по расширению привилегий.

Подход к противодействию угрозам расширения привилегий

Очевидно, что простейшим и, по мнению автора, отнюдь не лучшим способом является противодействие рассматриваемым атакам посредством исправления найденных ошибок в приложениях (что сегодня и имеет место на практике). Альтернативный подход, который будет рассматриваться в работе, состоит в реализации защиты предоставляемых ОС сервисов добавочными средствами. Цель реализации данного подхода – создать такие условия функционирования системы, при которых наличие ошибки в приложении не скажется на компьютерной безопасности (т.е. целью является решение задачи защиты от рассматриваемой группы атак в общем виде).

С учетом того, что средство добавочной защиты должно контролировать процедуру олицетворения, с целью разрешения только корректных, с точки зрения безопасности системы, вариантов заимствования прав, т.е. осуществлять разграничения прав, прежде всего определимся с тем, что является субъектом и объектом доступа в рассматриваемой схеме разграничений. Предлагается в качестве субъекта доступа рассматривать процесс, которым порождается поток, в качестве объекта доступа – права доступа (контекст защиты), задаваемые учетной записью пользователя, которые системой защиты предоставляются либо нет потоку, запросившему олицетворение. Другие возможные варианты – это рассматривать в качестве субъекта поток, запрашивающий операцию олицетворения (но такой механизм будет невозможно настроить), либо пользователя. Задание в качестве субъекта доступа учетной записи пользователя, вообще говоря, можно рассматривать лишь как вариант частного решения задачи (при этом все процессы, выполняющиеся в контексте защиты определенного пользователя, должны подчиняться единым правилам олицетворения). Ввиду того, что каждая программа в общем случае может затребовать собственных разрешений олицетворения для запускаемых ими потоков, именно процесс предлагается рассматривать в качестве субъекта доступа, для которого задаются разграничения.

Важным вопросом реализации предлагаемого подхода является выбор режима запуска процедуры контроля. В общем случае процедура может запускаться синхронно

(по расписанию, либо с заданным интервалом) и асинхронно – по какому-либо событию. Синхронному способу запуска присуще множество недостатков, и он, как правило, используется в случае, когда невозможен асинхронный запуск (либо в дополнение к способу асинхронного запуска).

Остановимся на рассмотрении способов асинхронного запуска. Применительно к рассматриваемому механизму можем выделить следующие события, которые в общем случае можно рассматривать как причину запуска процедуры контроля олицетворения: запрос на олицетворение, генерируемый потоком, либо запрос доступа к ресурсам. Реализация обоих способов средствами добавочной защиты имеет свои достоинства и недостатки. В первом случае механизм в части реализации представляет собой отдельный *диспетчер маркеров доступа* (рис. 2), осуществляющий перехват и обработку соответствующих запросов. Во втором случае задача контроля возлагается на драйверы контроля доступа к ресурсам (прежде всего, к файловым объектам – локальным и разделенным в сети – и устройствам; к реестру ОС), т.е. уже, как правило, существующие в системе добавочной защиты (рис. 3). Естественно, в части простоты реализации второй подход предпочтительней. С точки же зрения реализуемых возможностей защиты, естественно, предпочтительней первый подход, так как он позволяет контролировать собственно процесс олицетворения, разрешая осуществлять только корректные, с точки зрения безопасности системы, заимствования прав (т.е. в полном объеме реализует механизм управления доступом к олицетворению контекстов защиты). Второй подход же не запрещает некорректные олицетворения – он лишь не позволяет осуществить доступ к ресурсам потоку, осуществившему некорректное олицетворение.

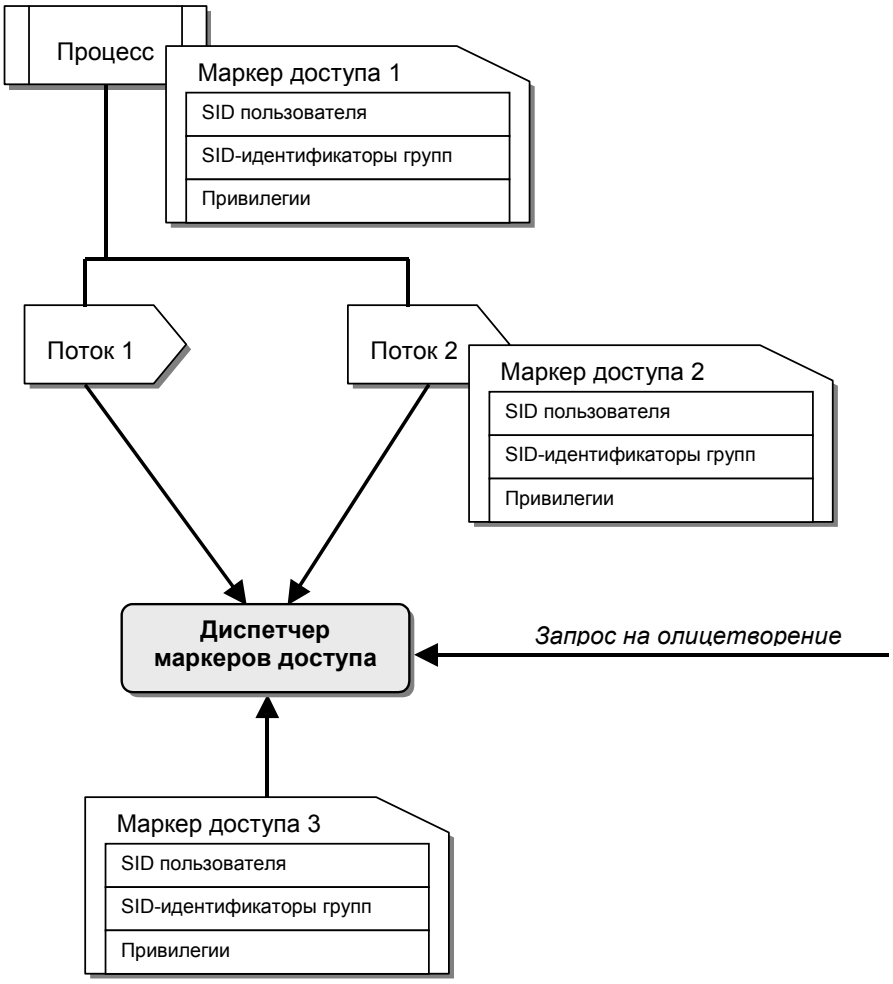


Рис. 2. Внедрение диспетчера маркеров доступа в процесс обработки запросов на олицетворение

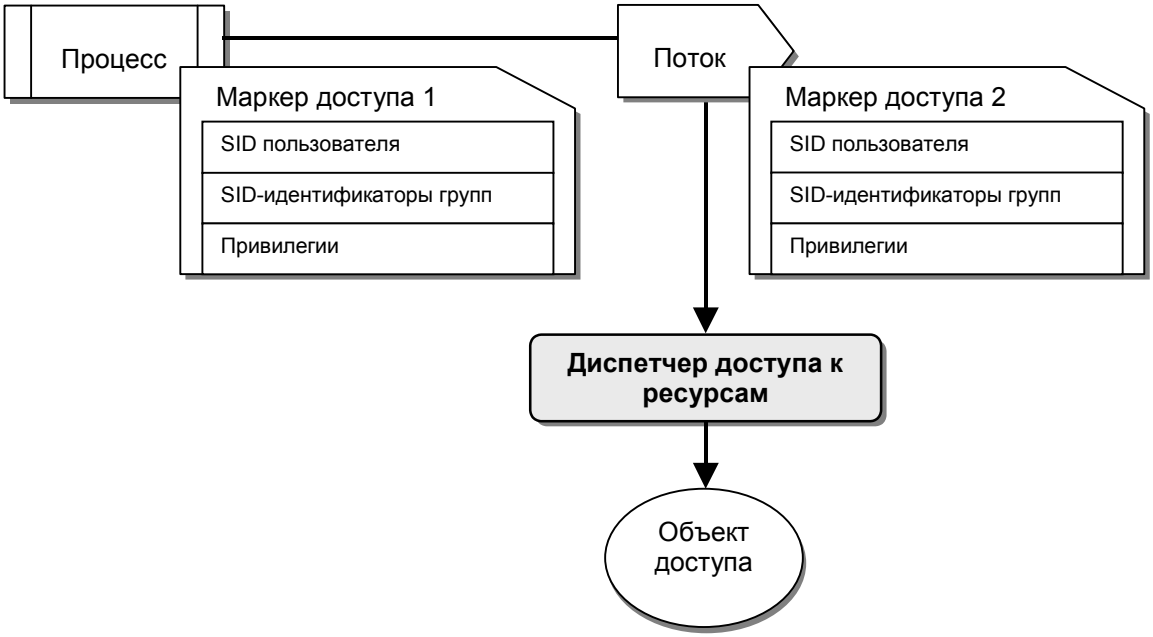


Рис. 3. Контроль корректности олицетворения при доступе к ресурсам

Реализация системы контроля корректности олицетворения

С точки зрения выбора способа реализации системы контроля корректности олицетворения вернемся к вопросу – группе каких атак мы хотим оказывать противодействие данным механизмом – это атаки на повышение привилегий. При этом мы предполагаем, что злоумышленник уже находится в системе, где может запускать собственные процессы, и цель его атаки – это получить права доступа к ресурсам более привилегированного пользователя. В этом смысле – в части противодействия атакам на повышение привилегий с целью несанкционированного доступа к данным – рассматриваемые подходы практически эквивалентны.

Исходя из этого, на данном этапе для реализации был выбран второй способ – контроль корректности олицетворения, осуществляемый при доступе к защищаемым ресурсам – файловым объектам и устройствам (локальным и разделенным в сети), реестру ОС. Данный механизм реализован КСЗИ «Панцирь» для ОС Windows 2000/XP/2003 (интерфейс его настройки приведен на рис.4) и апробирован.

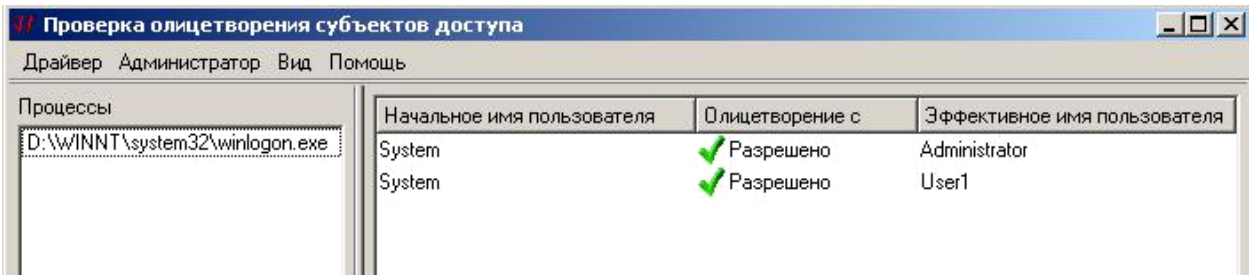


Рис. 4. Интерфейс настройки механизма контроля олицетворения, реализованный в КСЗИ «Панцирь» для ОС Windows 2000/XP/2003

Помимо основной задачи предотвращения рассмотренного класса угроз расширения привилегий, данный механизм позволяет реализовать и принципиально новые свойства защиты. Например, при авторизации пользователя при входе в систему потоки, запускаемые процессом *winlogon*, олицетворяются с учетной записью авторизуемого пользователя. При использовании данного механизма можно управлять тем, какие пользователи могут входить в систему. Для настроек, представленных на рис. 2, это

только пользователи «Administrator» и «User1». Наличие иных учетных записей, заведенных в системе, не даст возможности войти под ними в систему.

Заключение

Итак, в ходе работы был осуществлен анализ текущей статистики угроз компьютерной безопасности, на основании которого выделен отдельный класс угроз расширения привилегий. Установлены причины уязвимости ОС семейства Windows NT, позволяющие осуществлять атаки, направленные на расширение привилегий. Сделан вывод, что основная причина кроется в предоставлении ОС незащищенных сервисов олицетворения, позволяющих потокам, запущенных от имени одного пользователя, временно заимствовать профили защиты других пользователей без прохождения дополнительной идентификации (т.е. без ввода имени пользователя и пароля). В качестве иллюстрации в работе были рассмотрены примеры реализации конкретных атак, использующих эти уязвимости. В результате был предложен подход к противодействию угрозам расширения привилегий, заключающийся в контроле процедуры олицетворения, с целью разрешения только корректных, с точки зрения безопасности системы, вариантов заимствования прав. Были предложены способы реализации обозначенного подхода с точки зрения запуска процедуры контроля и выбран способ, подразумевающий контроль корректности олицетворения при доступе к ресурсам. Итогом работы стал механизм контроля олицетворения, реализованный в КСЗИ «Панцирь» для ОС Windows 2000/XP/2003.

Литература

1. http://www.microsoft.com/technet/security. Различные ресурсы по информационной безопасности продуктов Microsoft Corporation.
2. http://www.ntbagtraq.com. Различные ресурсы по информационной безопасности.
3. Соломон Д., Руссинович М. Внутреннее устройство Microsoft Windows 2000. Мастер-класс. / Пер. с англ. СПб.: Питер; М.: Издательско-торговый дом «Русская редакция», 2001. 752 стр.: ил.
4. Скембрей Д., Мак-Клар С. Секреты хакеров. Безопасность Windows 2000 – готовые решения. / Пер. с англ. М.: Издательский дом «Вильямс», 2002. 464 стр.: ил.

ИССЛЕДОВАНИЕ БЕЗОПАСНОСТИ ПРОТОКОЛА HTTP

А.Е. Изюмов

В данной статье рассматриваются проблемы защищенности протокола HTTP, варианты возможных атак, которые могут быть проведены с использованием уязвимостей протокола, и способы защиты.

Введение

HTTP (Hyper Text Transfer Protocol) протокол впервые был зафиксирован в RFC (Request For Comments – RFC 1945) в 1996 году, но в то же время его начали использовать примерно с 1993 года. Этот протокол был разработан для поддержки отображения разнообразной информации посредством стандартного клиентского интерфейса. Его способность динамично взаимодействовать с информационным содержимым в пределах HTTP сессии привела к созданию множества мультимедиа приложений, скриптов и программ, использующих HTTP протокол в качестве удобного инструмента для управления и передачи данных. Вследствие его широкой распространенности он довольно часто подвергается атакам. Так, например, по данным сайта SANS Institute's Internet Storm Center (Incidents.org), занимающегося вопросами интернет-безопасности и проблемами обнаружения атак, 80 порт (порт протокола HTTP) последние пять лет входит в пятерку наиболее атакуемых портов.

В свете подобных обстоятельств можно считать задачу обеспечения безопасности веб-серверов и мер по предотвращению несанкционированного доступа к ним первоочередной для большинства организаций и частных лиц, интересы которых могут быть затронуты в случае успешного проникновения хакеров.

В данной работе рассматриваются атаки на протокол HTTP и его уязвимости, после чего на основе полученного исследования предлагается система мер по обеспечению комплексной защиты веб-сервера от атак.

Уязвимости протокола HTTP

Изначально протокол HTTP разрабатывался в качестве протокола, ориентированного на высокую производительность при обмене электронной информацией, и поддерживает очень небольшое количество встроенных возможностей контроля за безопасностью.

Рассмотрим, какие возможности протокола HTTP могут быть использованы в качестве уязвимостей, а также различные виды атак, которым он может быть подвергнут. Атаки можно объединить в четыре основных группы – атаки на протокол, атаки с использованием кэша HTTP, атаки на веб-приложения и атаки на доверительную модель протокола HTTP.

Атаки на протокол HTTP

Перехват информации, содержащейся в пакетах HTTP. Информация, передаваемая в пакетах и заголовках HTTP, может быть использована для получения данных о веб-сервере или клиенте и, впоследствии, для выбора типа атаки.

Атакующий может получить от веб-сервера следующие типы информации:

- информация об учетных записях пользователей;
- коммерческая информация;
- информация о хосте (сервер или клиент).

Взлом учетных записей пользователей может привести к получению доступа и привилегий к другим ресурсам. Например, взлом учетной записи домена NT/2000, свя-

занной с веб-сервером, может открыть доступ к ресурсам домена, если сервер имеет с ним доверительные отношения.

Базовая авторизация доступа. Используемое кодирование Base-64 вместе со слабым способом авторизации дает возможность применить атаку прямым перебором или с помощью словаря (см. рис. 1).

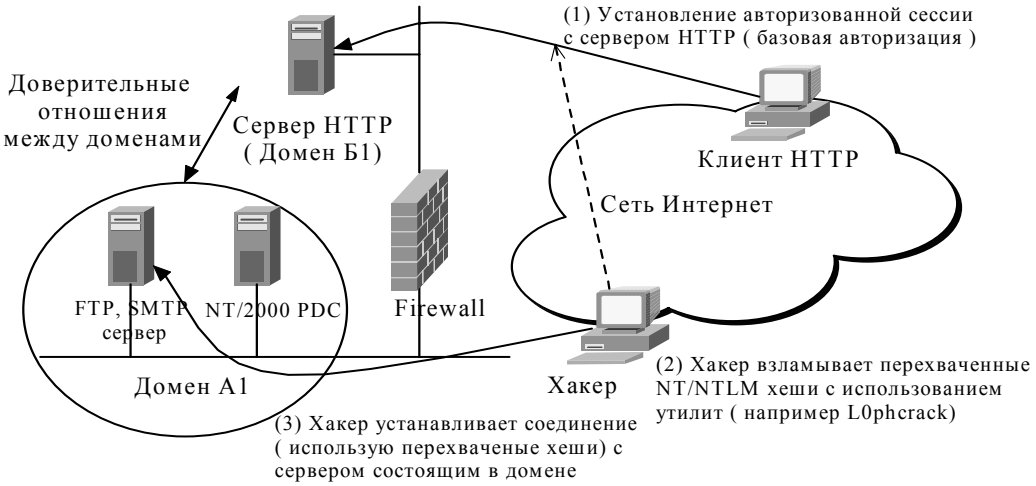


Рис. 1. Схема атаки на механизм базовой авторизации

Авторизация доступа на основе дайджестов подвержена атакам прямым перебором и с помощью словаря, но при этом сильно усложняет задачу хакерам из-за необходимости взлома дайджеста для успешного проведения авторизации.

Уязвимости методов протокола HTTP. Методы POST, PUT позволяют вносить изменения напрямую в файловой системе сервера на уровне привилегий файловой системы, назначенных веб-серверу. Метод GET также дает возможность проникать на сервер, особенно при использовании уязвимостей приложений и скриптов на веб-сервере.

Уязвимости кэша HTTP

В большинстве случаев типы уязвимостей, которым подвержен кэш HTTP, совпадают с уязвимостями для веб-серверов, но в то же время из-за некоторой специфики кэш подвержен следующим типам атак:

- атаки вида Cache Poisoning (компрометирование кэша);
- атаки вида Man-in-the-middle (перехват и подмена данных);
- несанкционированный доступ к данным кэша или его мониторинг;
- отказ в обслуживании (DoS).

Атаки на приложения

Атаки вида **переполнение буфера** (привилегированный доступ к серверу, отказ в обслуживании) и **Directory Traversal** направлены на использование уязвимостей при интерпретации веб-сервером закодированных последовательностей передаваемых данных от клиентского браузера. Запустить команду cmd.exe из поддиректории сервера IIS можно по коду http://target/msadc/..%c0%af../..%c0%af../..%c0%af../wint/system32/cmd.exe?/c+dir, где "/..%c0%af../," обзначает "../" в кодировке Unicode.

Атаки типа **отказ в обслуживании** (DoS) на уровне приложений направлены на использование уязвимостей определенных версий сервера HTTP с целью уменьшения доступных серверных ресурсов или остановки сервера.

Атаки на доверительную модель HTTP

Атаки на HTTP сессию (Session ID Hacking). Протокол HTTP не позволяет сохранять информацию о состоянии между сессиями. Поэтому веб-сайты используют такие механизма сохранения состояния, как cookies, динамические или статические поля URL и скрытые теги для контроля над сессиями. Большинство из этих механизмов обеспечивает авторизацию и аутентификацию пользователя и могут быть использованы при проведении атак подмены (Session ID Hijacking – см. рис. 2) или повтора сессии (Session Replay).

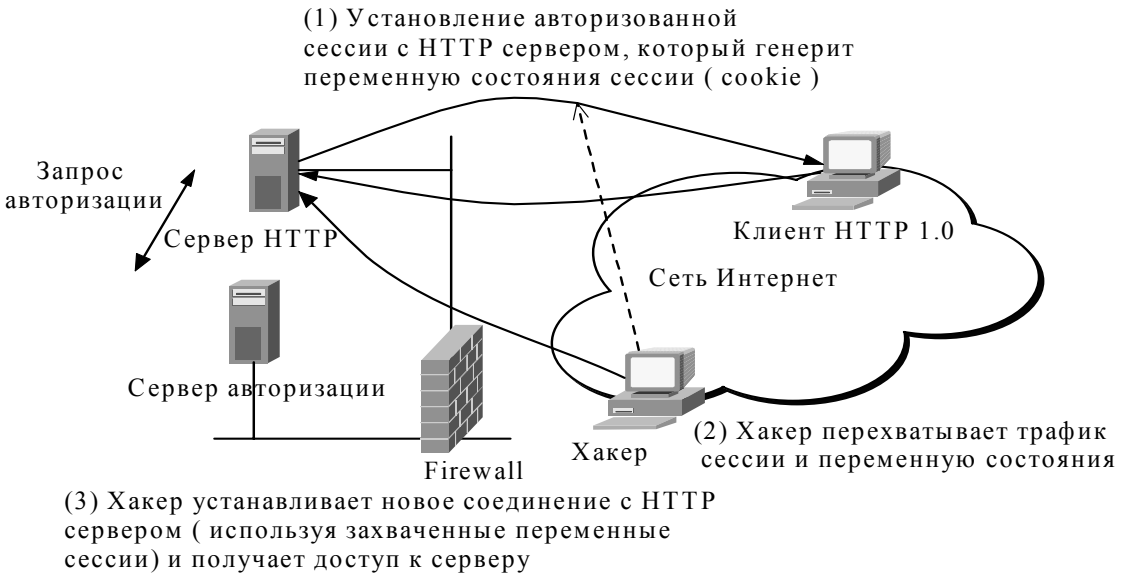


Рис. 2. Схема атаки Session ID Hijacking

HTTP Spoofing, HTTP Redirection. Данные атаки используются для перенаправления HTTP клиентов на сайт злоумышленника с целью перехвата информации или для совершения взлома сервера.

Атаки вида Man-in-the-middle (перехват и подмена данных). Данный вид атаки включает в себя перехват и изменения пакетов данных с целью внедрения системы злоумышленника в TCP сессию.

Стратегия защиты

Рассмотрим предлагаемый комплекс мер обеспечения безопасности протокола HTTP.

Настройка кэша (или кэширующего прокси). Прежде всего, для ускорения доступа к статическому контенту, следует использовать «серверный кэш», т.е. расположить сервер кэша перед веб-сервером. Настройки безопасности кэша включают в себя следующее:

- *Управление доступом и авторизацией.* Доступ может предоставляться на основе групп, имен пользователей, IP-адресов, HTTP-методов.
- *Фильтрация URL.* Ограничение доступа к определенным ресурсам, а также для защиты кэша путем ограничения типов URL и содержимого для кэширования.
- *Фильтрация контента.* Защита содержимого кэша путем фильтрации типов контента для кэширования, например по MIME-типу или расширению файлов.
- *Время жизни, обновления и устаревания кэша.* Контроль времени кэширования для защиты против атак cache poisoning.
- *Ограничения на размер файлов в кэше.*

- *Распределенное кэширование.* Обеспечивает лучшую устойчивость при DoS атаках.
- *Фильтрация заголовков.* Кэш-прокси могут подавлять передачу некоторых HTTP заголовков с целью защиты серверного контента и от сбора информации о сервере.
- *Организация межсетевой защиты и сокрытие.* Располагая кэш-прокси перед группой веб-серверов, и заставляя пользователей извне запрашивать данные из кэш-прокси, мы скрываем присутствие конечных веб-серверов и серверов-приложений.

Отключение уязвимых методов HTTP. Помогает снизить вероятность компрометации веб-сервера.

Фильтрация заголовков. В большинстве случаев фильтрация заголовков HTTP осуществляется на кэширующем прокси, тем не менее, при поддержке сервером данной возможности, фильтрацию можно производить и на сервере.

Внедрение авторизации доступа HTTP на основе дайджестов. Данный вид авторизации предоставляет лучшую безопасность, нежели базовая авторизация. Поддерживается с версии HTTP 1.1.

Выравнивание сетевой нагрузки и резервные сервера. Устройства выравнивания сетевой нагрузки могут обеспечить защиту от HTTP атак на отказ в обслуживании (см. рис. 3).

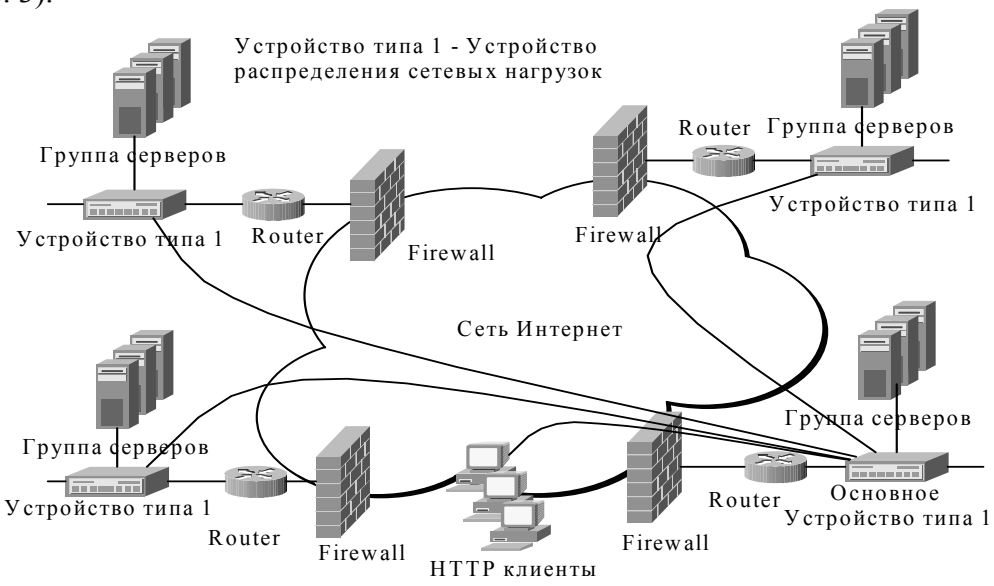


Рис. 3. Схема сети с распределенной нагрузкой

Мониторинг сети и сервера HTTP, обнаружение атак. Устройства мониторинга веб-серверов можно разделить на следующие категории.

- *Встроенные средства мониторинга и измерений*. Администраторы могут использовать логи операционной системы и веб-сервера для отслеживания данных об авторизации, статистики соединений, запрашиваемых объектов, потребления ресурсов сервера.
- *Системы обнаружения атак (IDS) – на уровне сети и на уровне хоста.* IDS хостов могут использоваться для мониторинга файлов лога, ресурсов, файловой системы на случай отказа в обслуживании (DoS) или проникновения на сервер. Сетевые IDS могут отслеживать использование сетевых ресурсов или проверять сетевой трафик на предмет злоумышленного кода или попытку взлома HTTP сервера.

Следует обратить внимание на следующие атаки.

- *Прослушивание веб-трафика.* Злоумышленник, перехватывающий трафик, может быть выявлен с использованием соответствующей утилиты.

- *Взлом учетных записей пользователей.* Данный вид атаки может быть обнаружен при многократных попытках получить доступ к системе под одним и тем же логином или при попытке перехвата идентификаторов авторизации сессии.
- *Отказ в обслуживании (DoS).* Анализ показателей производительности может указать на возможную попытку DoS атаки или на нехватку ресурсов в системе.
- *Атаки переполнением буфера.* Могут быть обнаружены при нехватке системных ресурсов, или нестабильной работе системы. Администраторы должны отслеживать такие же показатели, как и для DoS атак.
- *Атаки на сессию и на "обход" директорий.* Данные атаки могут быть обнаружены IDS хостов (HIDS – Host IDS) или сетевыми IDS (NIDS – Network IDS) с помощью сигнатур для известных уязвимостей.

Системные заплатки и обновления безопасности.

Безопасность финансовых транзакций. Помимо TLS (Transport Layer Security) и SSL (Secure Socket Layer), существуют и другие технологии обеспечения безопасности финансовых транзакций. Наиболее известной можно считать SET (Secure Electronic Transaction).

Контроль доступа со стороны сервера включает следующие настройки:

- Контроль доступа и авторизации, включающий в себя контроль доступа к определенным URI (Unified Request Identificator), виртуальным серверам и хостам.
- Контроль доступа к контенту, который может определять доступ к определенному скрипту, запускаемому файлу или файловому содержимому.
- Контроль доступа к URI, ограничивающий доступ к определенным URI на веб-сервере.

Настройка системы и сервисов для повышения безопасности. При настройке HTTP сервера администраторам нужно обратить внимание на следующее.

- Контроль доступа для файловой системы, веб-директорий, CGI (Common Gateway Interface) директорий, файлов лога, и индивидуальных скриптов и исходных файлов.
- Запуск HTTP сервера с правами пользователя. Что ограничивает возможности доступа к файловой и операционным системам.
- Ведение лога для веб-транзакций. HTTP сервер должен вести логи, куда должны заноситься все ошибки и успешно запрошенные данные. Предпочтительно, чтобы логи периодически архивировались на удаленном сервере.
- Отключение неиспользуемых скриптов, и компонентов для уменьшения возможностей для проникновения.

Использование TLS или SSL для повышения защищенности передаваемых данных.

Заключение

В ходе работы проводилось исследование проблем безопасности, затрагивающих работу веб-сервера, были выделены в группы и структурированы атаки на протокол HTTP, рассмотрены имеющиеся в протоколе уязвимости. По итогам исследования можно сделать вывод, что протокол HTTP сам по себе является небезопасным. По результатам исследования была предложена система мер по обеспечению стабильной и безопасной работы веб-сервера, представляющая собой комплексный подход для предотвращения широкого спектра специфичных для протокола атак.

Литература

1. HTTP Authentication: Basic and Digest Access Authentication (RFC 2660, E. Rescorla, A. Schiffman, Aug. 1999)

2. Hypertext Transfer Protocol – HTTP/1.1 (RFC 2617, J. Franks, P. Hallam-Baker,J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart, June 1999).
3. Information Security Management Handbook, 5th Edition, Harold F. Tipton, Micki Krause (Auerbach Publications, ISBN: 0-8493-1997-8).
4. Internet security : cryptographic principles, algorithms, and protocols, Man Young Rhee (Wiley, ISBN 0-470-85285-2).
5. Web Security (A Stepby-Step Reference Guide), Lincoln D. Stein (Addison Wesley, ISBN 0-201-63489-9).
6. Web Hacking: Attacks and Defense, Stuart McClure, Saumil Shah, Shreeraj Shah (Addison Wesley, ISBN 0-201-76176-9).

ИССЛЕДОВАНИЕ БЕЗОПАСНОСТИ ПРОТОКОЛА FTP

А.В. Иванов

Эта статья связана с протоколом FTP – протоколом передачи файлов. В ней рассматриваются схемы работы протокола, приводится классификация удаленных атак, а также меры противодействия уязвимостям протокола.

Введение

В статье рассматриваются аспекты безопасности протокола FTP. Актуальность данной проблемы связана с широким использованием данного протокола. В настоящее время протокол является одним из двух наиболее популярных средств обмена файлами.

Проблемы его безопасности обусловлены временем принятия данного стандарта (1985 год) и особенностями его работы. Со времени создания протокол ни разу не дорабатывался с учетом требований безопасности, хотя рекомендации по этому вопросу выпущены были.

В настоящее время протокол часто используется как один из этапов удаленной атаки, на котором либо происходит сбор информации о системе и сети, в которой находится атакуемая система, либо вредоносный код записывается на атакуемый узел с целью его локального выполнения.

В рамках статьи проводится обзор протокола FTP, описание его работы и особенностей, классификация удаленных атак. Далее рассматриваются атаки, относящиеся непосредственно к протоколу, и способы защиты от них. В заключение приводится описание комплексного подхода к противодействию уязвимостям протокола FTP.

Протокол FTP

FTP (File Transfer Protocol или «протокол передачи файлов») – один из старейших протоколов в Internet, входящий в его стандарты, известные как RFC (Request for Comments). Его начало было положено в 1971 году в стандарте RFC 114, который описывал первые механизмы передачи файлов между двумя узлами сети. После этого он претерпел множество изменений, а в 1985 году был принят стандарт RFC 959, который считается на данный момент основной версией.

Стандарт RFC 959 определяет 4 основных функции протокола FTP:

1. предоставление файлов для общего доступа;
2. упрощение обмена данными и непрямого использования удаленных компьютеров посредством программных средств;
3. устранение различий в представлении данных между узлами сети различных архитектур;
4. надежная и эффективная передача данных.

В общем случае сетевой протокол предполагает участие двух узлов сети, между которыми происходит обмен информацией, и наличие одного канала связи, по которому этот обмен происходит. В случае протокола FTP используется два канала связи: канал управления обменом и канал передачи данных.

Схема работы протокола FTP показана на рис. 1. Сначала по запросу клиента формируется канал управления, который в дальнейшем используется для передачи команд от клиента и откликов от сервера. Канал передачи данных формируется сервером по команде клиента. Этот канал не должен существовать постоянно на протяжении всей FTP-сессии и может формироваться и ликвидироваться по мере необходимости. Канал управления может быть закрыт только после завершения информационного обмена. Канал передачи данных может работать в двух режимах: активном и пассивном.

При использовании активного режима клиент указывает с помощью FTP-команды PORT адрес и порт узла, с которым сервер создает соединение с 20-го порта для обмена данными. Пассивный режим включается FTP-командой PASV. Ответ на эту команду содержит адрес и порт сервера, к которому должен подключиться клиент для начала обмена данными. Порт выбирается произвольно из верхнего диапазона номеров (больше 1024).

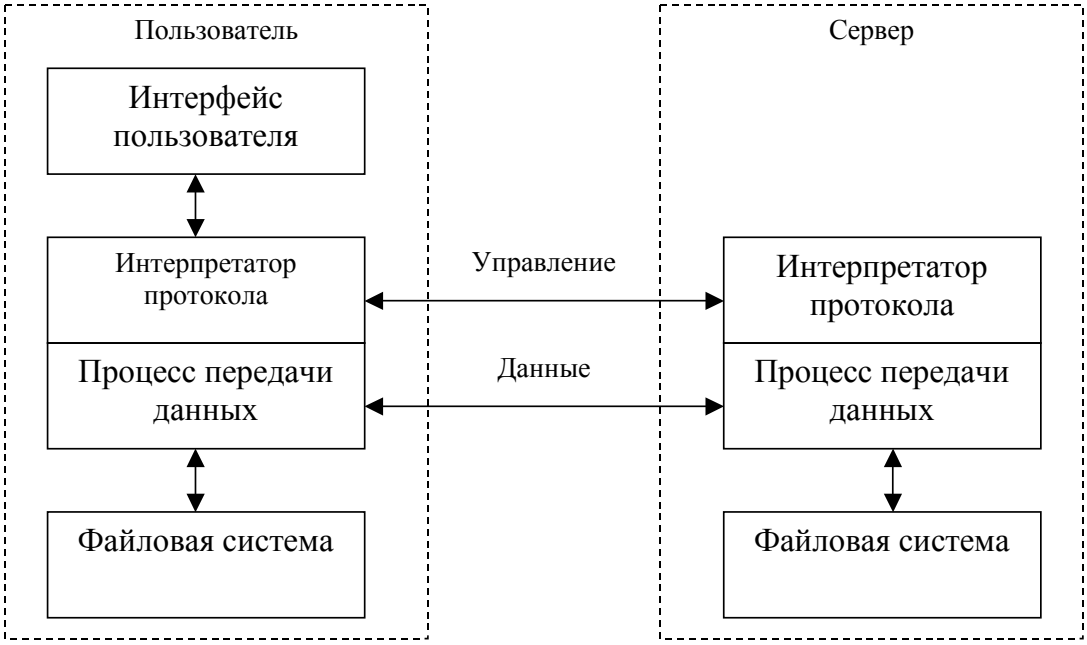


Рис.1. Схема работы протокола FTP

Использование раздельных каналов управления и данных позволяет ввести третьего участника обмена данными. Такая схема взаимодействия приведена на рис. 2.

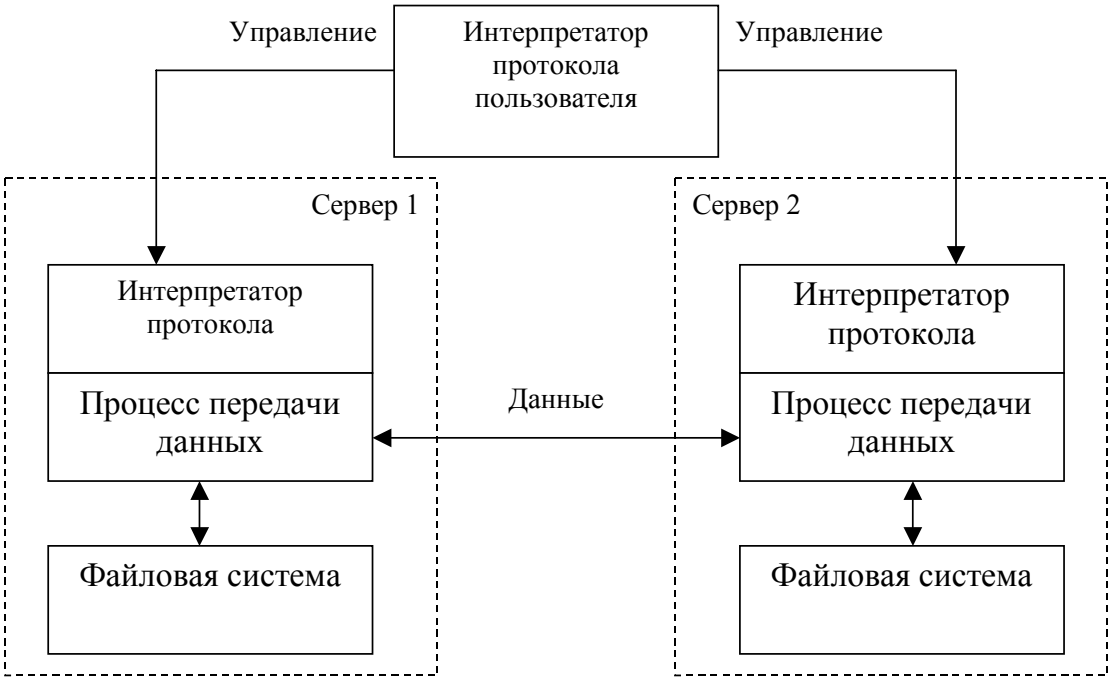


Рис. 2. Соединение с двумя разными серверами и передача данных между ними

В этом случае пользователь организует канал управления с двумя серверами и прямой канал данных между ними. Команды управления идут через пользователя, а данные – напрямую между серверами. Один из серверов должен работать в пассивном

режиме, другой – в активном. Клиент, переведя какой-либо сервер в пассивный режим, получает адрес и порт, который потом используется для указания адреса узла, с которым должен соединиться другой сервер, работающий в активном режиме.

Уязвимости протокола FTP

Наиболее существенным недостатком протокола является передача всей информации, а также имен и паролей пользователей, в открытом виде. Это обусловливает невозможность использования данного протокола для передачи конфиденциальной информации без использования сторонних программных и аппаратных средств. Если злоумышленник имеет доступ к каналу связи, через который проходят эти данные, необходимо использовать шифрование. Это характерный случай пассивного воздействия атаки – состояние сервера не изменяется, политика безопасности не нарушается, но существует доступ к необходимой информации.

В протоколе не определены действия, противодействующие подбору паролей. После неправильного пароля клиенту предоставляется возможность ввести его повторно, а соединение не разрывается. Также не существует ограничений на количество повторов. В результате атака, направленная на подбор паролей, может продолжаться сколь угодно долго, а отсутствие задержек при ответах сервера повышает эффективность.

Стандарт протокола FTP определяет код ответа 530 на команду USER, когда имя пользователя не допускается. Если имя пользователя верно и необходимо ввести пароль, то возвращается код ответа 331. Это является уязвимостью с точки зрения защиты имен пользователей. По ответам сервера можно судить о существовании определенного имени в базе пользователей и продолжить подбор пароля.

Следующие две уязвимости связаны с пассивным режимом протокола и возможностью участия в соединении третьего узла.

При использовании пассивного режима передачи данных, при котором сервер указывает клиенту, на какой порт необходимо соединиться для начала передачи, возможна установка соединения с другого компьютера. Если реальный клиент уже выбрал необходимый для скачивания файл и имеет необходимый доступ, то возможна кража от его имени. Злоумышленник, зная особенности выбора портов FTP-сервером для организации пассивного режима, повышает вероятность успеха атаки. Для этого необходимо пытаться устанавливать соединения с портами, и в случае успеха файл будет украден. Точно также можно записать файл на сервер от имени зарегистрированного пользователя, установив соединение с портом сервера, ожидающим начало файла.

Атака «FTP-bounce» использует команду PORT для соединения с третьим компьютером. Схема атаки изображена на рис. 3. В команде указывается адрес и порт узла-жертвы, с которым сервер устанавливает ответное соединение.

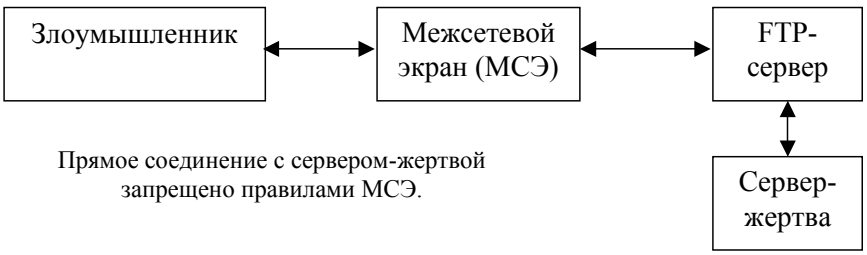


Рис. 3. Схема атаки FTP-Bounce

Данную уязвимость можно использовать для:

- скрытого сканирования другого узла, поскольку сканирование будет проходить от имени FTP-сервера, а не атакующего. В некоторых условиях это позволяет обойти

списки доступов, так как FTP-сервер может находиться в подсети жертвы или иметь с ним доверительные отношения;

- обхода межсетевых экранов. Если некий порт внутреннего сервера закрыт межсетевым экраном, а доступ к FTP-серверу есть, то можно от его имени установить соединение с портом, который недоступен. Далее происходит посылка специально сформированного файла, содержащего инструкции протокола службы, с которой было установлено соединение. В результате имитируется прямое соединение с вводом необходимой последовательности команд.

В отличие от атак на протокол, использующих особенности или недоработки протокола, приведенные выше, существуют также атаки на приложения – серверы, реализующие этот протокол. В основном для этого используются некорректно сформированные команды. В общем виде целью данных атак является:

1. нарушение работоспособности системы, которое включает в себя аварийный останов или зацикливание программы-сервера, избыточное использование системных ресурсов (например, оперативной памяти или процессорного времени);
2. повышение привилегий доступа к системе;
3. выход за границу FTP-каталога%
4. выполнение команд ОС с привилегиями FTP-сервера.

Противодействие уязвимостям протокола FTP

Уязвимости протокола по большей части обусловлены его особенностями и отсутствием способов защиты передаваемой информации. Для повышения безопасности необходимо использовать сторонние средства, а также тщательно продумывать взаимодействие узлов сети посредством протокола FTP.

Проблема передача всей информации в открытом виде решается либо использованием средств шифрования, где это возможно, либо защитой каналов связи от несанкционированного доступа. Остальные проблемы можно решить с помощью фильтрации.

Для защиты имен пользователей от подбора фильтр должен подменять ответ 530 FTP-сервера, формируемый при получении команды USER с несуществующим именем, на ответ 331, который подтверждает существование имени пользователя. Для защиты паролей от перебора необходимо настроить FTP-сервер таким образом, чтобы соединения закрывались после некоторого количества попыток ввода пароля. В фильтр можно также заносить количество указанных USER-PASS команд с отрицательным ответом сервера и, при их недопустимом значении, закрывать порт для данного адреса на определенный промежуток времени, для чего необходимо взаимодействие с межсетевым экраном. Также необходимо обеспечить паузу перед ответом на каждый неправильный пароль, что позволит существенно затормозить их перебор. Однако при этом остается возможность перебора с использованием параллельных соединений с FTP-сервером. Для противодействия необходимо устанавливать ограничения на количество одновременно обслуживаемых подключений, что, в свою очередь, делает возможным атаки типа «Отказ в обслуживании».

Для предотвращения кражи файлов при пассивном режиме необходима фильтрация по IP-адресу. Адрес клиента, который инициировал пассивный режим, будет адресом назначения пакета с ответом на команду PASV. При использовании такой фильтрации становится невозможным обмен между двумя серверами, инициируемый клиентом, поскольку сервер, переведенный в активный режим, будет иметь адрес, отличный от адреса клиента, а пакеты от него будут фильтроваться. По этой причине необходимо определить, будет ли существовать такой обмен и можно ли его избежать, и только после этого осуществлять фильтрацию.

Возможность атаки «FTP-Bounce» также обусловливается особенностями протокола. Способ решения – такой же, как и для проблемы кражи файлов. Также возможна фильтрация пакетов или конфигурация FTP-сервера, чтобы при получении команды PORT с номером порта меньше 1024 выдавалась ошибка. Это нужно для того, чтобы предотвратить атаку на другие стандартные сервисы от имени FTP-сервера. Протокол FTP не должен использовать номера портов меньше 1024 для передачи данных

Наиболее значительная группа уязвимостей связана с атаками на приложения. Это связано с большим разнообразием программных средств, реализующих FTP-серверы, и их версий. Существует 2 способа обработки некорректных запросов.

1. Вести базу некорректных запросов и фильтровать пакеты в соответствии с ней. Этот вариант плох тем, что база уязвимостей растет, а быстродействие фильтра падает. Также необходимо отслеживать новые уязвимости и новые версии программного обеспечения для обновления этой базы.
2. Не фильтровать некорректные запросы вообще, а отслеживать состояние FTP-сервера. В этом случае должны быть обеспечены отказоустойчивость операционной системы к сбоям FTP-сервера и возможность корректного перезапуска приложений в ней.

Второй способ является комплексным и включает в себя, помимо фильтрации, следующее:

- настройку FTP-сервера для работы под отдельной учетной записью с ограниченными привилегиями. В этом случае получение прав самого сервера в результате использования его уязвимости ничего не даст;
- отслеживание состояния FTP-сервера, а в случае аномалий в его поведении – корректный перезапуск. Это должно осуществляться средствами ОС (контроль за выполнением процесса и используемых ресурсах) и сторонними средствами (контроль за возможностью соединения с FTP-сервером по сети);
- запрет на исполнение программ, записанных клиентами на FTP-сервер. Это необходимо по причине частого использования протокола FTP как средства для записи злоумышленником вредоносного кода на сервер-жертву, который впоследствии запускается как локальный.

Заключение

В статье были рассмотрены аспекты безопасности протокола FTP, являющегося популярным средством передачи файлов по сети. Было выяснено, что протокол по своей природе не является безопасным, и необходимо использование комплексных средств защиты. Этими средствами являются: правильное построение модели файлового обмена, корректная настройка программного обеспечения, использование алгоритмов шифрования везде, где это доступно, применение фильтрации при взаимодействии с межсетевыми экранами и использование сторонних средств для контроля за состоянием программного обеспечения FTP-сервера. Применение этих средств позволит уменьшить риск атак с использованием этого протокола.

Литература

1. Postel J., Reynolds J., File Transfer Protocol (FTP), RFC 959, ISI, 1985.
2. Allman M., Ostermann S., FTP Security Considerations, Ohio University, 1999.
3. Леонов Д.Г., Лукацкий А.В., Медведовский И.Д., Семьянов П.В. Атака из Internet. М.: СОЛОН-Р, 2002.
4. Лукацкий А.В. Обнаружение атак. / 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2003.

5 ИНФОРМАЦИОННО-УПРАВЛЯЮЩИЕ СИСТЕМЫ

ВЛИЯНИЕ АППАРАТНОЙ ОРГАНИЗАЦИИ НА ТЕХНОЛОГИЮ ПРОГРАММИРОВАНИЯ ВО ВСТРОЕННЫХ СИСТЕМАХ

Р.Р. Ковязин

Современные технологии постепенно проникают в область встроенной техники. Для применения этих технологий нужны специалисты, владеющие ими. В статье рассматриваются вопросы, связанные с выбором технологии программирования при разработке встроенных систем. Оценивается применимость различных технологий программирования.

Введение

Профессия программиста сейчас чрезвычайно популярна. Персональные компьютеры уверенно входят в нашу жизнь, и люди, умеющие не только общаться с ними, но и задавать их функциональность, очень перспективны. Во множестве вузов страны готовят специалистов по различным направлениям, которые потом могут участвовать в разработке вычислительных систем, быть программистами, проектировщиками, архитекторами, тестерами и т.п. Несмотря на большой поток выпускников вузов – будущих программистов, среди них очень мало людей, пригодных к участию в разработке программного обеспечения для встроенных систем (ВсС). Специалистов с подходящей специализацией должны выпускать кафедры с направлением «Информатика и вычислительная техника», но по объективным причинам они с этим не справляются [1]. По оценкам специалистов время, проведенное выпускником вуза на работе, по истечению которого от него можно ожидать прогнозируемых результатов (оценить профессиональную пригодность), составляет около полугода для разработчиков систем на основе персональных компьютеров, локальных сетей и Internet. Для разработчика ВсС «подготовительный период» составляет не менее двух лет. Причиной этого является то, что подготовка специалистов в этой области включает в себя базовые знания, а современные тенденции и технологии разработки ВсС в нее не входят.

10–15 лет назад разработчиков ВсС можно было условно поделить на программистов и схемотехников, при этом они не сильно отличались друг от друга. Вычислительная и коммуникационная мощность электронных компонент позволяла реализовывать простые (по сегодняшним меркам) аппаратные решения. Программное обеспечение плохо переносилось на другие аппаратные платформы. Технический прогресс позволил значительно усложнить аппаратную базу и программное управление, реализованное на ее основе. Это потребовало повышения квалификации разработчиков ВсС, которое, в свою очередь, разделило эти группы разработчиков и ухудшило взаимодействие между ними. Усовершенствование элементной базы предоставило больше возможностей программному обеспечению, сделало возможным использовать сложные алгоритмы, сложные структуры данных и управления, обрабатывать большие объемы информации. Все это в совокупности вывело ВсС на новый уровень [2]. Современные ВсС имеют сложную организацию, могут использовать операционные системы реального времени, объединяться в контроллерные сети (Embedded Net), использовать современные и высокоскоростные технологии связи.

Существует потребность в специалистах, владеющих современными технологиями, которые могут применить их во ВсС. Поскольку таких специалистов вузы не готовят, то набирать их можно среди традиционных программистов с последующим переучиванием. Как было сказано выше, требуется не менее двух лет, чтобы разработчик

перестал принимать глупые (деструктивные) решения, делающие систему ненадежной, небезопасной (даже на этапе разработки и тестирования) и неэффективной. Если программирование персонального компьютера приучает к программированию с неограниченными вычислительными ресурсами, то программирование ВсС обладает следующими свойствами:

- надежное программирование;
- вычислительные ресурсы существенно ограничены;
- важен реальный масштаб времени;
- значительное число используемых аппаратных платформ.

Используя те или иные методики при разработке программного обеспечения, следует отдавать себе отчет в том, какие ресурсы потребуются для его исполнения. В данной работе представлено введение в концепцию, позволяющую унифицировать подход к многообразию используемых во ВсС программируемых устройств, выделяются их классы и показываются ограниченность применимости различных технологий программирования.

Вычислители и технологии программирования

В программировании широко используется термин *уровень программирования*. Рассмотрим платформу IBM PC с процессором Intel x86. Благодаря долгому развитию у нее существует множество уровней программирования, начиная от программирования на ассемблере и заканчивая библиотеками MFC, VCL и др. под платформу WIN32. Чем выше уровень программирования, тем меньше в нем остается средств непосредственного управления процессором и аппаратными ресурсами системы, тем больше в нем «удобств», предоставляемых операционной системой, различными службами, драйверами, библиотеками функций и т.п. Существуют интерпретирующие ядра (виртуальные машины), например, Java, Smalltalk, Perl, в которых аппаратный процессор вообще подменяется программным. Несмотря на то, что процессор один, его программирование на разных уровнях существенно отличается.

Очевидно, что уровни программирования определяются используемыми инструментальными средствами. Для унифицированного рассмотрения программируемых устройств, использующихся во ВсС, введем понятие *вычислитель* – устройство или комплекс устройств, имеющее уровень программирования. Наличие нескольких уровней программирования означает наличие нескольких вычислителей. Традиционное разделение программы на прикладной и системный уровень равносильно введению в систему нового вычислителя. Разработчик программы формирует управление ресурсами вычислителя, а не использующегося в системе процессора, который является самым низкоуровневым вычислителем. Разработчику не всегда предоставляется возможность программировать этот вычислитель [3]. Внутреннее устройство вычислителя, его компоненты, реализация, число внутренних уровней программирования и т.п. важны только для разработчика этого вычислителя, но не для разработчика, пользующегося им. В основе всех вычислителей лежат программируемые устройства: микроконтроллеры, микропроцессоры и программируемые логические интегральные схемы (ПЛИС).

Технология программирования (ТП) применительно ко ВсС является технологией реализации программного управления средствами вычислителей системы. В каждом из них может использоваться своя ТП. ТП объединяет в себе множество необходимых для создания программ понятий. Их можно разделить на несколько категорий:

- вычислительные модели;
- методики (шаблоны и типовые решения);
- языки программирования;
- инструментальные средства.

Все эти категории связаны друг с другом, между ними можно вводить разные отношения следствия. ТП объединяет в себе способ, процесс и средства алгоритмизации.

Даже в рамках одного проекта в коллективе разработчиков могут присутствовать те, кто разработали вычислитель, и те, кто им пользуется. Назовем их условно «разработчиками» и «пользователями». В таблице показаны особенности разработки программного обеспечения, зависящие от наличия разработчиков этих двух категорий.

«Разработчики»	«Пользователи»	Описание
Отсутствуют	Отсутствуют	Вырожденный случай. Вычислитель не является программируемым.
Отсутствуют	Присутствуют	Программное обеспечение создается для вычислителя, разработанного сторонней фирмой. Это самый распространенный вариант использования вычислителей во ВсС.
Присутствуют	Отсутствуют	Разрабатываемая ВсС допускает пользовательское программирование, является программируемым вычислителем. Она может существовать как самостоятельная система или как подсистема.
Присутствуют	Присутствуют	Сложная система, в разработке программного обеспечения которой участвует несколько групп разработчиков. При этом одна группа пользуется результатами работы другой группы.

Таблица. Варианты организации разработчиков программного обеспечения ВсС

Из таблицы можно понять, что самым низкоуровневым вычислителем является тот, который был разработан сторонними фирмами. Для него создается система драйверов, операционная среда, операционная система. Они обеспечивают новые ресурсы (функциональность, данные) – формируют новый вычислитель, которым будут управлять более высокие уровни программного обеспечения.

Классификация вычислителей

Выбор вычислителей является важным вопросом в организации программного управления. Существует множество классификаций вычислителей. Характеристики и классы существующих вычислителей, с одной стороны, являются ограничителями ТП, используемых для создания ВсС. С другой стороны, они влияют на создание новых, более эффективных ТП в области ВсС. Ниже будет показано несколько классификаций, оказывающих наибольшее влияние на выбор ТП. С более полным списком классификаций можно ознакомиться в статье «Выбор технологии программирования встроенных систем», готовящейся к печати в журнале «Компоненты и технологии».

Программное управление. В рамках программного управления вычислитель позволяет манипулировать своими ресурсами. Можно выделить характерные способы доступа к ресурсам вычислителя. На их основе построим классы программного управления или классы программ вычислителей.

- *Последовательное* программное управление. Ресурсы вычислителя доступны лишь в определенных комбинациях. Для управления всеми необходимыми ресурсами требуется последовательность комбинаций. Допустимые комбинации назовем *инструкциями* вычислителя. В любой момент времени ядро выполняет лишь одну инструкцию.

- *Последовательно-параллельное* программное управление. Одновременно доступно несколько комбинаций ресурсов вычислителя, что позволяет организовать несколько последовательностей инструкций (программных потоков). Управление числом последовательностей и взаимодействие между ними должно быть обеспечено на уровне инструкций, происходить явно, на программном уровне.
- *Параллельное* программное управление. Все ресурсы вычислителя доступны в любой комбинации и в любой момент времени.

Вычислители классифицируются на основе классов программ. Последовательные программы соответствуют принципам программного управления фон Неймана. Большинство существующих программируемых вычислителей, используемых при разработке ВсС, являются последовательными. Последовательно-параллельные вычислители менее распространены в области ВсС и реализуются, например, в различных API (WIN32 API, POSIX и др.). Если параллелизм доступа к ресурсам вычислителя достигается неявно, то он не считается параллельно-последовательным. Вычислителями с параллельным программным управлением в определенном смысле являются ПЛИС.

Основным ограничением, вносимым вычислителями с различным программным управлением, являются доступные вычислительные модели. Языки для создания последовательных и параллельных программ различаются множеством вычислительных моделей, реализованных в них. Среди всех поддержанных моделей можно выделить базовые и производные. В последовательных и параллельных программах отличаются базовые модели, хотя некоторые производные совпадают. Для параллельных программ базовой вычислительной моделью является модель дискретных событий, а для последовательных программ – синхронный автомат [4]. Принципиальные отличия в базовых моделях программ ведут к тому, что обычно разработчики специализируются на разработке программ только одного из классов. Специализация в разработке программ обоих классов в соответствии с составляющими ТП влечет за собой необходимость владения характерными вычислительными моделями, методиками, языками и инструментальными средствами обоих классов.

Разрядность вычислителя. Разрядность вычислителя определяет, какие данные им обрабатываются наиболее оперативно (*разрядность данных*), какие объемы данных вычислитель в состоянии обработать (*объем адресного пространства данных*) и насколько сложное программное управление в нем можно реализовать (*объем адресного пространства кода*).

Превышение пределов разрядной сетки влечет за собой трудности, в большинстве случаев скрываемые инструментальным обеспечением и незаметные на программном уровне. Результатом является значительное увеличение размера кода и замедление работы с «большими» данными.

- *4-битные* вычислители решают специфические задачи и обычно используются в сложных электронных компонентах ВсС (контроллеры ЖКИ, инфракрасные приемопередатчики и т.п.). Программное управление алгоритмически простое и низкоскоростное.
- *8-битные* вычислители используются в электронных компонентах, как и 4-битные. Кроме этого, их используют для решения низкоскоростных коммуникационных задач, задач логического управления небольшой и средней сложности, управления частью периферии контроллера. Программа обычно имеет однородную структуру [3]. Ресурсы вычислителя ограничены и достаточно универсальны.
- *16-битные* вычислители используются для коммуникационных задач средней сложности, на них строят стеки протоколов различных интерфейсов. В программе появляется структура: операционные среды, виртуальные машины и т.п. Объем ресурсов вычислителя увеличился по сравнению с предыдущим классом. Для организации более сложного программного управления появляется специализация по

классу решаемых задач: логическое управление, математические вычисления, сигнальная обработка, коммуникации и т.п.

- *32-битные* вычислители обладают высокой производительностью и используются для сложных вычислений, обработки больших объемов информации и высокоскоростных коммуникационных протоколов. У программ таких вычислителей усложняется структура (многозадачные операционные системы, планировщики, подсистемы управления ресурсами, API и т.п.).

Распространенность архитектуры. Использование процессоров определенной архитектуры определяет доступный арсенал инструментальных средств. Разработчики инструментальных средств стараются вывести свои продукты на высокий уровень, равняясь на широко распространенные инструментальные средства известных фирм (Microsoft, Metroworks). Хотя инструментальная поддержка процессоров разных архитектур существенно отличается, ее отдельные компоненты можно сделать единообразными для всех архитектур: среду разработки, редактор, интерфейс компилятора, компоновщика и т.п. Выделим следующие классы архитектур:

- *Специализированные архитектуры* обычно разрабатываются для специфических нужд, когда существующие архитектуры не могут быть использованы. Этот подход обычно применяется при создании вычислителей в рамках ВсС.
- *Архитектуры ограниченного применения* имеют инструментальную поддержку, но она обычно осуществляется разработчиками процессора. Со временем процессоры с этими архитектурами могут распространиться, и их инструментальная поддержка расширится.
- *Широко распространенные архитектуры* имеют хорошую инструментальную поддержку, большое число готовых решений. Во ВсС нашли применение архитектуры x86, ARM, MCS-51, AVR, PIC и др. Инструментальные средства создаются как разработчиками процессора, так и сторонними фирмами (Keil Software, HI-TECH и др.)
- Разработка программ ведется на специальной вычислительной машине, называемой инструментальной. Процессор ВсС может обладать *архитектурой инструментальной* машины. Это дает существенное преимущество в отладке программ и расширяет число инструментальных средств, но затрудняет адаптацию под конкретную систему, поскольку она все-таки не полностью совпадает с инструментальной.

Инструментальные средства для каждой архитектуры постепенно развиваются, но, к сожалению, не всегда в одном направлении. Разработчик, работающий с различными архитектурами, вынужден работать с разными инструментальными средствами, пользовательскими интерфейсами, компонентами, знать и учитывать их свойства.

Распространенность архитектуры в первую очередь влияет на инструментальную поддержку вычислителя и документирование.

Заключение

Ограничения, накладываемые вычислителем на реализацию программного обеспечения, и его преимущества по сравнению с другими вычислителями нужно учитывать на этапе проектирования и реализации ВсС. Особое внимание нужно этому уделить во время выбора программируемых устройств. У разработчиков аппаратной составляющей, как и у разработчиков программного обеспечения, есть критерии выбора программируемых устройств и организации их взаимодействия с периферией системы. Важно, чтобы в выборе компонент системы, определяющих, какие ТП будут использоваться, принимали участие обе группы разработчиков. В противном случае в архитектуру могут быть заложены ненадежность аппаратной составляющей, дорогое производство и тестирование, сложное программирование и трудность смены профиля системы,

способность обрабатывать небольшие объемы информации и другие свойства, препятствующие повторному использованию и развитию системы на рынке BcC.

Литература

1. B. Graaf et al. Embedded Software Engineering: The State of Practice. IEEE Software, 2003.
2. E. Lee. What's Ahead for Embedded Software? 2000.
3. Ковязин Р., Платунов А. Программирование микроконтроллерных систем. // Электронные компоненты. 2003. №4.
4. L. Lavagno, A. Sangiovanni-Vincentelli, E. Sentovich. Models of Computation for Embedded System Design. 1998.

ЦЕЛИ ПОДГОТОВКИ СПЕЦИАЛИСТОВ В ОБЛАСТИ КОРПОРАТИВНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

П.В. Кустарев

В статье анализируются актуальные задачи и проблемы подготовки корпоративных специалистов по информационным технологиям (ИТ). Рассмотрены требования к составу и качеству подготовки со стороны основных заинтересованных групп этого процесса: работодателей, студентов и преподавателей, констатируется серьезное несоответствие взглядов. На базе выполненного анализа сформулированы цели и предложены некоторые общие принципы подготовки ИТ-специалистов, отвечающие необходимости устранения указанных противоречий и общему повышению эффективности процесса.

Введение

Современная ситуация такова, что наряду с бурным ростом спроса на специалистов в области информационных и компьютерных технологий наблюдается не столько недостаток выпускников вузов в этом секторе, сколько несоответствие подготовки специалистов и требований реального производства. Такое положение вещей наблюдается, видимо, по всем направлениям: в проектировании программных средств различного уровня и назначения, аппаратных средств вычислительной техники, в алгоритмизации, в создании и обслуживании информационных систем.

Одним из наиболее крупных по охвату задач и персонала является сектор корпоративных компьютерных систем. Такая «массовость» определяет наибольшее количество публичных нареканий по качеству подготовки специалистов данного направления (часто применяются термины ИТ-специалисты, ИТ-системы, ИТ-сектор и т.п.).

Основываясь на опыте постоянного взаимодействия со специалистами отрасли, с преподавателями и студентами (в частности, кафедры вычислительной техники СПбГУ ИТМО), автор считает, что первопричина проблем заключается в несоответствии задач подготовки, видимых в вузе и на производстве. Необходимо переориентировать и реорганизовать подготовку специалистов таким образом, чтобы минимизировать данные противоречия, сохранив при этом значимость, потенциал и долговременную перспективу образования. Задачами данного небольшого исследования является поиск упомянутых противоречий, анализ причин их возникновения с последующей выработкой некоторых общих подходов, позволяющих продвинуться на пути их устранения.

Корпоративный ИТ-сектор

Сектор информационных технологий (ИТ-сектор) организаций определяется как системообразующая компьютеризированная инфраструктура. Различные специализированные информационные системы – бизнес-планирования, бухгалтерские, документооборота, системы автоматизированного проектирования (САПР) и другие – также можно рассматривать как системообразующие, однако они являются прикладными, иерархически расположены над инфраструктурой и поэтому не отнесены к сфере проектной ответственности ИТ-сектора. Внедрением подобных комплексов обычно занимаются специалисты по информационным системам: экономическим или техническим. Корпоративные ИТ-специалисты в данном случае ответственны только за общетехническую поддержку. СУБД занимают пограничное положение и, если не сегодня, то в ближайшие годы перейдут на уровень инфраструктуры, аналогично тому, как это сделали в свое время коммуникационные системы.

Анализ организации компьютерной инфраструктуры в различных организациях, в том числе на производственных предприятиях, на предприятиях коммунальных служб, телекоммуникационных фирмах, в фирмах-разработчиках программного обеспечения и других, позволяет сделать вывод о схожести технических задач и средств компьютер-

ной инфраструктуры вне зависимости от отрасли и размера предприятия, и принципиальных различиях в методах управления ИТ-сектором. Если малые организации довольствуются одним или несколькими системными администраторами – универсалами, то крупные фирмы содержат независимые группы системных аналитиков, внедрения, эксплуатации, развития. Эти моменты необходимо учитывать при подготовке специалистов.

С самого начала следует определиться, что в качестве стратегического направления системы высшего образования рассматривается выпуск специалистов, удовлетворяющих требованиям потенциальных работодателей. В этом смысле вузы не отличаются от других фирм: работодатель выступает в роли или фактически является «клиентом» вуза. Производство «некачественного» продукта – специалистов – приведет к банкротству вуза.

Важным моментом является, что государство следует рассматривать как приоритетного клиента, предпринимать мероприятия по повышению его заинтересованности в данной категории специалистов.

Требования к подготовке ИТ-специалистов

Ниже выполнена попытка оценить требования к молодым специалистам, как они видятся различными заинтересованными группами людей. Автору представляется, что сектор подготовки корпоративных ИТ-специалистов очень показателен в этом плане: так как мощная (как минимум, объемная) компьютерная инфраструктура на сегодня характерна для большинства предприятий, то достаточно широка и, хочется надеяться, объективна выборка мнений.

Технические специалисты по ИТ – системные администраторы, начальники отделов обслуживания компьютерных систем, проектировщики систем и сетей – основными преимуществами работников видят технические навыки, позволяющие им качественно и, что важно, самостоятельно выполнять определенный спектр работ. При этом в большинстве случаев требуется определенная универсальность работника: он должен уметь и проложить кабель, и установить программное обеспечение, и подобрать оборудование. К сожалению, отечественные специалисты этого уровня часто упускают профессиональные навыки в системотехническом проектировании. Как следствие – компьютерные системы далеко не всегда сбалансированы, их эффективность низка и не отвечает уровню вложений. Низкая рентабельность ведет к падению интереса к этому направлению со стороны топ-менеджеров предприятий, т.е. тех, кто выделяет деньги.

Следующая группа – технические менеджеры. К ним отнесены руководители по информационным технологиям, технические директора, иногда менеджеры по персоналу. Это основная группа «клиентов» для вузов, так как именно они устанавливают правила и принимают решения по найму тех или иных категорий работников. Основными профессиональными требованиями здесь является базовая подготовка по требуемой специализации, опыт реальной практической деятельности в этой или близкой области, стремление и способность к самостоятельному обучению. Такие оценки, как виртуозное обращение с тем или иным программным пакетом или опыт работы с определенным оборудованием, иногда могут повысить оценку кандидата на место, но в очень незначительной степени. Люди, дошедшие до руководителей такого уровня, понимают: зная принцип и имея голову, деталям можно обучиться очень быстро. Еще раз отмечу, что не снижается ценность практических навыков, просто снимаются требования к деталям.

Другая группа качеств, важных для менеджмента фирмы не менее, а возможно и более, чем профессиональные навыки – это способность и стремление к командной ра-

боте, осознание ее необходимости, ответственность работника. В этой связи часто упоминают навыки руководства (группой, проектом), хотя это мнение является спорным.

Представленные выше оценки во многом подтверждаются информацией, полученной с сайтов по поиску работы. Обычно присутствует типовой перечень менеджерских требований: опыт, командная работа, ответственность и обязательность (а какая без этого командная работа!). Указывается предметная область (системное администрирование, проектирование сетей), иногда перечисляются программные пакеты и технические средства, однако это чаще затрагивает только второстепенные должности.

Наконец, третья группа мнений высказана самими преподавателями вузов. В статье они приводятся, чтобы сравнить то, что ожидают «клиенты», с тем, что для них готовят. Мнения преподавателей делятся на несколько групп. Первая проповедует принцип «научить думать и учиться». При многих положительных свойствах такого подхода усматривается проблема неспособности многих студентов перейти на этот метауровень образования. В итоге они не получают конкретных профессиональных навыков и знаний и остаются неспособными их самостоятельно приобрести. Вторая группа преподавателей предлагает сконцентрироваться на востребованных знаниях и навыках, чаще практического свойства. Иногда это сводится к безудержной погоне за новейшими технологиями. Третья группа ориентируется на преподавание теоретических основ, оставляя практические навыки на самостоятельное обучение.

В заключении покажем взгляд самих студентов и выпускников вузов. Видимые ими требования к подготовке специалистов в большинстве случаев мало отличаются от аналогичных требований первой группы (это не удивительно, ведь именно технические специалисты еще вчера были студентами). Упоминаются, прежде всего, разнообразные практические навыки в выбранной профессиональной области. Отличительной чертой студента являются значительные личные амбиции, явные и скрытые.

Из вышесказанного видно, что требования того, «кто покупает» (менеджеры), с одной стороны, и, с другой стороны, тех «кто выпускает» (преподаватели), а также тех, «кто пользуется» (технические специалисты), и тех, «кого выпускают», во многом не совпадают. В чем причина таких противоречий и на кого ориентироваться при определении стратегического направления в подготовке специалистов?

Сначала немного о причинах. Скорее всего, они кроются в различной должностной мотивации, определяемой решаемыми задачами и мерой ответственности. Уровень менеджмента сконцентрирован на создании устойчивой, слаженной, развивающейся, эффективной фирмы. Технические детали и решения интересуют хорошего менеджера лишь в качестве косвенных характеристик эффективности коллектива. Главное свойство работника – способность адаптироваться под решение разнообразных задач в рамках единого производственного механизма.

Технический специалист видит цель своей деятельности в получении эффективных и интересных решений, обычно имеет устоявшуюся инструментально-методическую среду работы, заинтересован, чтобы новый член коллектива был его единомышленником, укреплял, а не разрушал эту привычную рабочую среду. Отсюда вытекают требования, достаточно жестко проводимые этой группой.

Несмотря на кажущийся нивелированным подход менеджера (иногда говорят, что работники воспринимаются не более чем шестеренки), именно он дает максимальные степень свободы и перспективы роста для специалиста, так как в минимальной степени регламентирует техническую деятельность рамками эффективности работы коллектива. Однако этому препятствует неготовность молодого специалиста к функционированию в подобной среде. Реальное состояние дел (по оценке менеджеров различных организаций) таково, что выпускники, с одной стороны, не имеют навыков и опыта командной работы, иногда необоснованно амбициозны, с другой стороны, зачастую имеют слабую базовую техническую подготовку и не могут выполнять даже несложные задачи.

Цели и принципы подготовки ИТ-специалистов

На базе выполненного анализа попробуем сформулировать главные цели и базовые принципы подготовки специалистов корпоративного ИТ-сектора. Общую цель подготовки можно достаточно узко определить следующим образом: выпуск специалистов востребованных, в дальнейшем развивающих свое направление, тем самым, увеличивая рынок образовательных услуг в нем. В рамках приведенного определения детализируем составляющие процесса вузовского образования.

Первой, с высшим приоритетом, идет подготовка специалистов под требования «клиентов» – менеджеров, т.е. имеющих базовую подготовку, наделенных установкой самообразования и с развитыми навыками коллективной работы. Наряду с этим не следует забывать о реалиях жизни. Выпускник вольется в первую очередь в коллектив «технических специалистов», а значит, как минимум первое время должен говорить на «их языке». В этой связи второй составляющей останется так называемая «тренировка», т.е. овладение в минимальном объеме современными техническими и программными средствами.

В завершении перечислим ряд базовых принципов, отвечающих указанным целям подготовки. Они известны давно, однако хотелось бы еще раз отметить их значимость, в том числе и в области ИТ-подготовки.

- Принцип базовых знаний, подразумевающий преподавание студенту обязательного, достаточно формального набора знаний и навыков, создающих информационную и методическую опору дальнейшего самостоятельного развития.
- Принцип ценностности знаний, заключающийся в получении знаний в процессе осмысления проблемы. Данный подход является тренировкой способности самообразования, личностной активности будущего работника.
- Принцип личностно-ориентированного (неформального) образования, позволяющий раскрывать и эффективно использовать индивидуальные особенности и способности студента. В рамках реализации этого принципа наиболее важными видятся психологические и педагогические навыки преподавателя.
- Принцип коллективно-обучающей деятельности. Его суть – обучение работе в команде. В рамках такого подхода важным является формирование групповой ответственности, обучение позиционированию специалиста в профессии и в коллективе, способам консолидации личных и производственных интересов, целям и методам карьерного роста.
- Принцип деятельного образования, подразумевающий обучение на базе реальной практической деятельности как на производственной базе, так и на аудиторной базе университета.

Заключение

Представленный выше анализ не является полным и окончательным. Это только начальный шаг, попытка осознать сложную и разностороннюю проблему. Автор имеет собственный оформленный взгляд на возможные пути реорганизации процесса обучения в рамках указанных принципов, однако размеры статьи не позволяют представить их в полном объеме.

Литература

1. Государственный образовательный стандарт высшего профессионального образования. Направление подготовки дипломированного специалиста 654600 – Информатика и вычислительная техника. – Москва, 2000.

2. Проблемы экономического и гуманитарного образования в техническом вузе. Научно-технический вестник СПб ГИТМО (ТУ). Выпуск 7. / Под ред. В.И. Подлесных. СПб: СПб ГИТМО(ТУ), 2003. 270 с.
3. Подготовка научных кадров: методики, технологии, результаты. Научно-технический вестник СПб ГИТМО (ТУ). Выпуск 10. / Под ред. Ю.А. Гатчина. СПб: СПб ГИТМО(ТУ), 2003. 199 с.

ПОЛУЧЕНИЕ АРХИТЕКТУРЫ ПРОГРАММНОГО ПРОДУКТА ПО ЕГО ИСХОДНОМУ ТЕКСТУ

А. С. Ахапкин

В статье приводится определение архитектуры и описание процесса реверс инжиниринга. Предлагается стратегия построения архитектуры программного продукта по его исходному тексту с использованием вспомогательных средств, позволяющих получить функциональную декомпозицию системы.

Введение

Стремительное увеличение сложности программ привело к увеличению ошибок в программах, локализация и исправление которых требует большого количества временных и материальных затрат. Таким образом, существенно возросла роль разработки архитектуры программы на этапе проектирования. Этот этап определяет все дальнейшее будущее программы как надежной, безошибочной и успешно решающей поставленные перед ней задачи.

Получение архитектуры уже законченной программы может стать важным этапом, обеспечивающим более глубокое понимание ее работы и облегчающим ее дальнейший анализ, оценку эффективности и определения вариантов использования этой программы. Одним из способов получения архитектуры программы может являться анализ ее исходных текстов. На сегодняшний день не существует формальных способов перехода от исходных текстов программы к ее архитектуре. В связи с этим особую актуальность приобретает задача разработки стратегии осуществления таких переходов.

Понятие архитектуры

Рассмотрим общее понятие архитектуры программного продукта. Одно из определений архитектуры может выглядеть следующим образом. Архитектура программного продукта позволяет получить ясное представление обо всей системе и включает в себя такие данные, как:

- данные об организации программной системы;
- данные о ее структурных элементах, их интерфейсах, а также их поведении, которое определяется кооперациями, в которых участвуют элементы;
- данные о составе структурных элементов и элементов поведения наиболее крупных подсистем.

Также архитектура программного продукта имеет отношение к способам использования, функциональности, производительности, гибкости, возможности повторного использования, экономическим, технологическим ограничениям, компромиссам и вопросам эстетики [1].

Данное выше определение архитектуры используется компанией Rational Rose. Оно достаточно громоздко и не раскрывает всей сути этого термина. Более коротко архитектуру можно охарактеризовать как концепцию работы системы, объясняющую внутреннюю структуру и поведение программы, однако и это короткое определение не позволяет раскрыть весь смысл этого важнейшего понятия.

Архитектура играет важнейшую роль в следующих аспектах:

- в понимании системы;
- в организации ее разработки;
- в повторном использовании кода;
- в дальнейшем развитии системы.

Процесс реверс-инжиниринга

Главный процесс реверс-инжиниринга – анализ исходных текстов исследуемой системы и распознавание компонент исходных текстов и отношений между ними.

Подходы, облегчающие понимание программного обеспечения, могут рассматривать исходные тексты в различных абстрактных формах, таких как: исходный текст; текст, прошедший препроцессорную обработку; лексические признаки текста; его синтаксические деревья; абстрактные синтаксические деревья с таблицами символов; графы потоков данных/управления. Более абстрактная форма влечет за собой дополнительный синтаксический и семантический анализ, который в большей степени согласуется со значением и поведением кода и в меньшей степени – с его формой и структурой. Различные уровни анализа служат разным целям понимания программы. Например, текст, прошедший препроцессорную обработку, теряет значительную часть информации об именованных константах, встроенных (inline) функциях и подключениях файлов.

Более подробно с такими подходами реверс-инжиниринга, как текстовый анализ, синтаксический анализ и семантический анализ, можно ознакомиться в [2].

Средства получения функциональной декомпозиции системы

Анализ исходных текстов некоторого программного продукта с целью получения его функциональной декомпозиции может осуществляться, например, пакетом программ Portable Bookshelf, использующим технику синтаксического анализа реверс-инжиниринга. С помощью этого пакета над исходными текстами программного продукта выполняются следующие действия:

- извлечение факторов из исходных текстов. Здесь и далее под факторами понимаются простейшие сущности программы, такие как имена переменных, имена функций и имена заголовочных файлов [3];
- кластеризация в подсистемы;
- отображение сгенерированной структуры программного продукта.

Функциональная декомпозиция системы упрощает понимание архитектуры программного продукта, но не может быть к ней приравнена. Полученный результат предлагается использовать для дальнейшего анализа системы путем его привязки к распространенным вычислительным моделям, таким, например, как SDF (Synchronous Dataflow) или FSM (Finite State Machines). Такая привязка возможна в случае иерархического представления программного продукта, в этом случае в основу каждого уровня иерархии может лечь определенная вычислительная модель.

Описание реализации

Будем считать базой для описания архитектуры следующие вычислительные модели [4–6]:

- CT – модель с непрерывным временем (continuous-time modeling);
- DE – модель дискретных событий (discrete-event modeling);
- FSM – конечные автоматы (finite state machines);
- DPN – сети процессов потоков данных (dataflow process networks);
- SDF – синхронные потоки данных (synchronous dataflow);
- SR – синхронные реактивные процессы (synchronous reactive).

Программная система может быть описана иерархически, на каждом уровне иерархии в качестве закона может использоваться своя вычислительная модель.

Стратегия получения архитектуры программы по ее исходным текстам состоит из трех частей – работа с исходными текстами, синтез архитектуры и ее проверка.

Первая часть стратегии – работа с исходными текстами системы в рамках подхода текстового анализа реверс-инжиниринга и ее функциональной декомпозицией. Целью этого этапа работы является выделение механизмов и их связей. Под механизмом будем понимать элемент архитектуры, реализующий часть структуры и функциональности исследуемой программы. Как правило, механизмы находятся на более высоком уровне абстракции, чем, например, структуры и функции языка С. Важное свойство механизма – его инвариантность к реализации. Пример механизма – интерпретатор пакетов. Результатом этапа работы с исходными текстами являются факты:

- перечень найденных механизмов и их связей;
- полученная из комментариев к исходным текстам дополнительная, относящаяся к механизмам информация.

Вторая часть стратегии – синтез архитектуры. Для осуществления корреляции между результатами предыдущего этапа и архитектурой воспользуемся методом экспертных оценок. Происходит оценка кода экспертом и сопоставление кусков кода и возможных механизмов. Результатом использования этого метода является связь вычислительных моделей с определенными уровнями иерархии программной системы и генерация нескольких архитектур, из которой для проверки выбирается более простая, «красивая» или симметричная архитектура. Такую выбранную архитектуру необходимо проверить на соответствие исходной программной системе.

Проверка осуществляется путем имитационного моделирования программной системы. Имитационное моделирование осуществляется с помощью программного комплекса PTOLEMY. В процессе моделирования результаты его сравниваются с результатами работы реальной программы. Сопоставимые результаты работы реальной системы и ее модели – критерий верности модели и архитектуры системы. В случае существенного расхождения в результатах работы системы и ее модели осуществляется возврат ко второму этапу стратегии или проверка другого, ранее сгенерированного, варианта до тех пор, пока результаты работы реальной системы и ее модели не будут сопоставимы.

Заключение

Функциональная декомпозиция программной системы, получаемая с помощью описываемых в статье средств, а также работа с исходными текстами по предложенной в статье стратегии позволяют синтезировать архитектуру исследуемой программной системы. Проверка архитектуры предложенными в статье средствами позволяет определить ее соответствие исходной системе. Изложенный в статье материал может служить основой для дальнейшего анализа в области архитектурного проектирования и других смежных областях.

Литература

1. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения. Питер, 2002.
2. E. Bass, Len, R. De Mori, M. Gentleman, J. Henshaw, "Investigating Engineering Technologies: The CAS Program Understanding Project", 1994.
3. R. Holt, “Portable Bookshelf Tools”, 1997.
4. S. S. Bhattacharyya, P. K. Murthy, E.A. Lee, “Software Synthesis from Dataflow Graphs”, Kluwer Academic Press, Norwood, Mass, 1996.
5. Edward A. Lee, Thomas M. Parks, “Dataflow Process Networks”, 1995.
6. Luciano Lavagno, Alberto Sangiovanni-Vincentelli, Ellen Sentovich, “Models of Computation for Embedded System Design”, 1998.

РАСПРЕДЕЛЕНИЕ ПРЕРЫВАНИЙ ПО УРОВНЯМ В МИКРОПРОЦЕССОРНЫХ СИСТЕМАХ

В.И. Скорубский, Ф.В. Хмылко

Рассматривается одна из ключевых задач, решаемых при программировании микропроцессорных систем реального времени. Предлагается алгоритм распределения классов прерываний по уровням с учетом абсолютных приоритетов и заданных ограничений. Решение этой задачи является одним из этапов программной или аппаратной инициализации системы прерывания.

Введение

С развитием и повышением степени интеграции вычислительных и управляющих устройств область применения микропроцессорных систем реального времени существенно расширилась и охватывает системы управления объектами и технологическими процессами, системы контроля, измерения и обработки изображений.

К реальному времени традиционно относятся все события, связанные с контролем временных соотношений, учитывающих продолжительность внешних (в объекте управления или измерения) и внутренних (в микропроцессорной системе) процессов. Принципиальным ограничением операционных систем реального времени (ОСРВ) является «жесткое» программное исполнение алгоритмов управления прерываниями. Следствием является увеличение времени реакции на случайные воздействия на вычислительный процесс внешних и внутренних событий.

Реактивность систем реального времени (время реакции) является одним из важнейших показателей, по которым сравниваются ОСРВ.

Системы прерываний наиболее распространенных 8-битовых микроконтроллеров (MCU) подчиняются жестким дисциплинам – используются абсолютные приоритеты и не допускаются многоуровневые повторные прерывания. Кроме того, средства программирования приоритетов с распределением по уровням могут быть использованы только на этапах разработки программ и не позволяют изменять данное распределение в динамике с учетом реальной обстановки в момент возникновения событий.

Возникновение событий носит стохастический характер, что учитывается в аналитических расчетах с использованием моделей массового обслуживания. В большинстве случаев предполагается идеализированная модель равномерного, стационарного или нормального распределения временных интервалов событий. При этом все характеристики усредняются и имеют отношение к реальным условиям работы системы с погрешностью, близкой к 100%. Вследствие этого, при всей оптимистичности оценок работоспособности системы в реальных условиях, события, нерегулярные по своей природе, происходят в случайные моменты времени и задерживаются до окончания программной обработки ранее принятых событий. Система прерываний, использующая регистры запросов прерываний, позволяет сохранить запросы разных классов. Однако запросы одного класса будут пропадать, и, следовательно, может быть утеряна информация о состоянии объектов или измерения.

Реализовать необходимую динамику распределения приоритетов при работе систем реального времени можно с использованием вспомогательных, внешних по отношению к микроконтроллерам аппаратных средств, имеющих существенно (на 2–3 порядка) более высокую разрешающую способность.

Построенные на основе ПЛИС с высоким уровнем интеграции при тактовых частотах в сотни МГц средства предварительного анализа и сохранения запросов прерываний (препроцессоры обработки прерываний) могут обеспечить более высокую надежность работы систем реального времени.

Метод распределения прерываний по уровням

Определим требования к системам реального времени (СРВ), исходя из следующих соображений.

1. Интегрированная совокупность аппаратных и программных средств микропроцессорной системы (МПС) должна гарантировать обработку любого наступившего события и принятие решения за время $t_П = t_з + t_о + t_р$, где $t_з$ – задержка входа в прерывание, $t_о$ – время обработки, $t_р$ – время выхода из прерывания, $t_П$ время реакции.

2. Обработка события должна осуществляться так, чтобы гарантировалось сохранение причинно-следственных связей между событиями – зависимые события должны обрабатываться в том порядке, в каком они происходят.

Точность сохранения этих связей зависит от интервала квантования временной шкалы событий T. Уменьшить данный интервал можно, учитывая среднее время обработки событий и формируя приоритеты с учетом их размещения по возрастанию – традиционная дисциплина обслуживания событий (задач) в пакетном (Batch) режиме ОСРВ. При наличии поддерживающих аппаратных средств в ПЛИС можно выполнить более тонкий расчет с учетом как порядка поступления событий в процессе их обработки, так и априорной информации о значимости порядка поступления событий.

Таким образом, свойства i-ого события определяем вектором (p, $t_П$, $t_З$, $t_О$, T), где параметр p – динамический приоритет в кванте времени, допускающем совпадение запросов прерываний с точностью $\tau=1/f$, (f – например, частота синхронизации ПЛИС), T – минимальное время между двумя запросами.

Кроме того, при обработке прерываний нобходимо учесть следующую систему неравенств: во-первых,

$$\Sigma\lambda < \mu,$$

где $\lambda = 1/T$ – интенсивность поступления запросов, $\mu = 1/t_Т$ – интенсивность обработки запросов, т.е. сумма интенсивностей запросов должна быть меньше интенсивности обработки; во-вторых,

$$\sum_{n=1}^{i} t_П\ (i) < T\ ,$$

т.е. реакция на последовательность запросов меньше минимального времени задержки каждого события T.

К задачам предварительной обработки событий относятся:

- накопление запросов прерываний по каждому каналу, пока микроконтроллер занят обработкой текущего прерывания;
- упорядочение и выделение запроса с максимальным приоритетом и формирование запроса к MCU;
- ожидание и фиксация момента завершения обработки прерывания – по счетчику времени или сигналу обратной связи перед выполнением команды возврата из прерывания;
- контроль допустимого времени ожидания завершения прерывания и формирование сброса, если сигнал завершения не поступает (функция сторожевого таймера при обработке прерываний).

В общем случае прерывания в MCU структурированы и распределены по группам и классам внутри групп, что позволяет упростить аппаратуру. Распределение сигналов по уровням прерываний может быть выполнено на этапе компиляции программы и учтено при инициализации системы обработки прерываний.

Рассмотрим механизм распределения сигналов по уровням и классам прерываний в случае программной реализации препроцессора.

Абсолютные приоритеты представим частично упорядоченным множеством классов прерываний $S=\{s_i\}$, где s_i – подмножества – классы прерываний, между классами

выбирается отношение строгого порядка (<), т.е. создается приоритет классов $s_1 < s_2 < s_3 \ldots < s_i$. Аппаратно фиксировано M уровней прерывания, в которых реально могут быть распределены классы. При этом задан жесткий порядок обработки (приоритет) классов прерываний $R=\{ri\}$, определяемый пользователем или условиями работы прикладной системы.

Приоритеты $r_i > r_j$ учитывают динамику работы системы – количество зафиксированных и необработанных запросов прерываний класса i или j, время ожидания запросов и время реакции и может быть определено логическим анализом.. В общем случае отношение приоритета не транзитивно.

Определим приоритеты ($r_i > r_j$) матрицей смежности $R*R$. Тогда классы прерываний i и j смежны (несовместимы), если $s_i < s_j$ и $r_i > r_j$. Построим ориентированный граф $G = \{X, S\}$, где $S = \{S_i\}$ – вершины (классы прерываний), $X = \{X_{ij}\}$ – ориентированные дуги (i, j), обозначающие порядок, реализуемый приоритетами различных уровней прерывания, которые несовместимы с абсолютными приоритетами.

Совместимые между собой классы сохраняют абсолютные приоритеты и входят в одно подмножество совместимости. Совместимые могут располагаться на одном уровне приоритета, а несовместимые должны располагаться на разных уровнях.

Система $C=\{\Psi_i\}$ устойчивых подмножеств совместимости образует покрытие множества R. Если покрытие содержит m подмножеств совместимости, то достаточно не более m уровней прерывания при распределении всех сигналов по уровням. Очевидно, это не всегда возможно. Тогда может быть использована стратегия наименьшего риска, при которой выбирается подмножество из Ψ, содержащее максимальное количество классов и устраняющее максимальное число отношений несовместимости.

Пример решения этой задачи

Микроконтроллер SAB515 разрешает N=13 сигналов прерываний, которые попарно объединяются в 7 классов и могут быть распределены по M=4 уровням прерывания. В одном классе прерывания упорядочены по абсолютным приоритетам. Сигналы из одного класса не могут быть распределены по различным уровням. Таким образом, при сравнении несовместимость сигналов в рамках одного класса не учитывается.

Граф несовместимости сигналов представлен на рис.1.

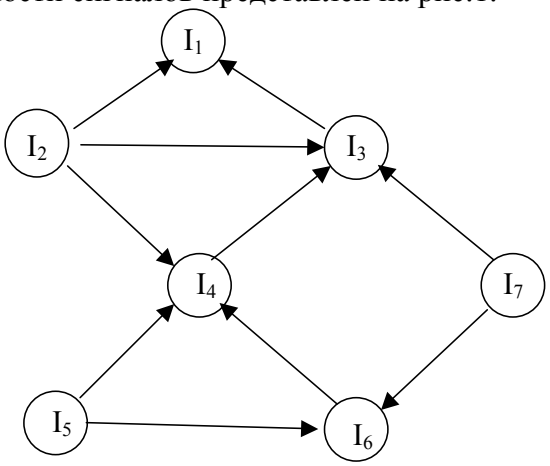


Рис. 1. Граф несовместимости классов прерываний

Разбиение графа по классам совместимости находим по следующим правилам:

- выбирается вершина с минимальным коэффициентом связности K_i (количеством смежных вершин) и включается в подмножество Ψ_j, $j = 1 \ldots m$;
- отмечаются связи, устраняемые при исключении данной вершины из G;

- из оставшихся вершин выбирается L-ая, не связанная с отмеченными дугами графа G и имеющая минимальный коэффициент связности K_L за вычетом отмеченных дуг, и включается в Ψ_j;
- после просмотра всех вершин графа переходим к формированию следующего подмножества Ψ_{j+1} и продолжаем применять правила, пока все вершины графа G не будут размещены в подмножествах разбиения.

Получена система подмножеств $C = \{\Psi_1, \ldots, \Psi_m\}$, которая соответствует распределению сигналов по m уровням. Если $m>M$, то минимальные по числу вершин подмножества Ψ_j объединяются с максимальным Ψ_k. Объединение оценивается по минимальным приращениям числа несовместимых классов или по числу дуг, связывающих вершины в Ψ_k. Таким образом, некоторые приоритеты из $R=\{r_i\}$ могут быть не реализуемы. Для примера, приведенного на рис.1, система подмножеств S имеет вид:

$\Psi_1=\{t_1, t_7, t_5\}$; $\Psi_2=\{t_2, t_6\}$;

$\Psi_3=\{t_3\}$; $\Psi_4=\{t_4\}$.

На следующем шаге необходимо перенумеровать подмножества в $C=\{\Psi_1, \ldots, \Psi_m\}$ и определить приоритеты подмножеств. С этой целью строится граф связности G_m подмножеств, которые обозначаются как вершины графа (см. рис.2). Выбирается вершина (подмножество $\Psi_{i)}$ с максимальным числом исходящих ребер (приоритет $t_i > t_k$) и минимальным числом входящих ($t_i < t_p$), назначается наименьший номер вершине Ψ_i

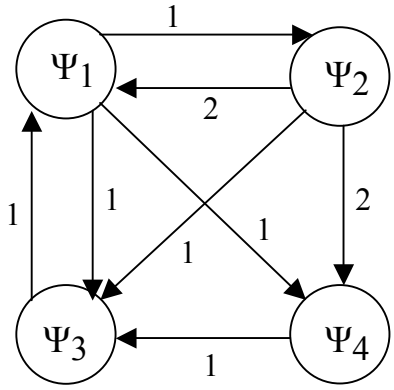


Рис. 2. Граф несовместимости подмножеств прерываний, распределяемых по уровням приоритетов

Для вершины Ψ_1 соответствующая пара значений связности (2, 3). Для вершины Ψ_2 соответствующая пара значений (5, 1) . Для вершины Ψ_3 соответствующая пара значений (1, 3). Для вершины Ψ_4 соответствующая пара значений (1, 3). Выбираем для Ψ_2 номер 0.

После устранения вершины Ψ_2 из графа G_m для вершины Ψ_1 соответствующая пара значений связности (2,1), для вершины Ψ_3 соответствующая пара значений (1, 2), для вершины Ψ_4 соответствующая пара значений (1, 1). Выбираем номер 1 для вершины Ψ_1.

После исключения вершины Ψ_1 вершина Ψ_3 характеризуется парой (0, 1), вершина Ψ_4 – парой (1, 0). Номер 2 с наибольшим приоритетом назначается вершине Ψ_4, а номер 3 с наименьшим – вершине Ψ_3.

В результате получаем оптимальное распределение классов прерывания по уровням $s_2 < s_1 < s_4 < s_3$, которое может быть зафиксировано при инициализации системы прерываний.

Заключение

Сформулирована задача расчета программируемых приоритетов прерываний с учетом абсолютных и динамических приоритетов. Препроцессор сохраняет запросы

прерываний, формирует запросы для обработки в MCU в соответствии с этими приоритетами.

Рассмотрен алгоритм распределения запросов прерывания по уровням в микропроцессорной системе, который может быть использован на этапе инициализации системы прерывания в MCU и зафиксирован аппаратно в ПЛИС препроцессора.

Предполагается, что применение в качестве препроцессора обработки прерываний ПЛИС позволит повысить надежность систем реального времени, связанную с конечным временем обработки и случайным характером распределения запросов во времени.

Литература

1. Операционная система ОСРВ СМ ЭВМ. М.: Финансы и статистика, 1987.
2. Описание SAB515 – материалы Infineon, Siemens.
3. Харари Ф. Теория графов. М.: Мир, 1973.

6 ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ, АВТОМАТИЗАЦИЯ ЛОГИЧЕСКОГО ПРОЕКТИРОВАНИЯ И ЗАЩИТА ИНФОРМАЦИИ

ПРОБЛЕМНЫЕ ВОПРОСЫ СОСТОЯНИЯ И РАЗВИТИЯ НОРМАТИВНО-ПРАВОВОГО ОБЕСПЕЧЕНИЯ ТЕХНИЧЕСКОЙ ЗАЩИТЫ ИНФОРМАЦИИ В АВТОМАТИЗИРОВАННЫХ СИСТЕМАХ УПРАВЛЕНИЯ И ИНФОРМАЦИОННО-ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

А.Д. Катаржнов, Е.А. Проценко

В статье рассматриваются положение дел и проблемные вопросы нормативно-правового обеспечения технической защиты информации в АСУ и ИТКС, и пути их решения.

Современный этап развития Российской Федерации характеризуется возрастающей ролью информационной сферы, представляющей собой совокупность информации, информационной инфраструктуры, субъектов, осуществляющих сбор, формирование, распространение и использование информации, а также системы регулирования возникающих при этом общественных отношений.

Анализ положения дел в области информатизации и защиты информации в России показывает, что информационные технологии принципиально изменили объем и важность информации, обращающейся в технических средствах ее хранения, обработки и передачи. В настоящее время одним из главных ресурсов Российской Федерации, основной ее экономического потенциала становится информация и информационные технологии. При этом всеобщая компьютеризация основных сфер деятельности привела к появлению широкого спектра внутренних и внешних угроз, нетрадиционных каналов утечки информации и несанкционированного доступа к ней.

При расширении процессов информатизации в ключевых сегментах информационной инфраструктуры России все больше функций передается автоматизированным системам управления (АСУ) и информационно-телекоммуникационным системам (ИТКС), включая органы государственной власти, учреждения и организации связи, транспортные и топливно-энергетические комплексы, финансово-кредитные системы, предприятия с вредными производствами, объектами жизнеобеспечения и коммунального хозяйства крупных населенных пунктов. Нарушения в работе указанных систем могут привести к техногенным катастрофам, экологическим, экономическим, социальным и другим чрезвычайным ситуациям и катастрофам с необратимыми последствиями, большими человеческими жертвами и материальным ущербом.

Качественное обеспечение этих процессов требует создания огромных баз данных, соответствующих хранилищ и коммуникаций для обмена данными, вовлекает в процесс управления большие группы специалистов, что, в свою очередь, создает дополнительные проблемы. Наиболее важной из них является возрастающая уязвимость АСУ и ИТКС в указанных сегментах информационной инфраструктуры государства в отношении угроз безопасности информации, показанных на рис. 1.[1]

В соответствии с Доктриной информационной безопасности Российской Федерации [2] необходимо решить ряд задач:

- повысить безопасность информационных систем, включая сети связи;

- интенсифицировать развитие отечественного производства аппаратных и программных средств защиты информации;
- обеспечить защиту сведений ограниченного доступа;
- расширять международное сотрудничество России в области развития и безопасного использования информационных ресурсов, противодействия угрозе информационного противоборства в информационной сфере.

Основными сферами реализации угроз безопасности информации для Российской Федерации являются:

- сфера экономики;
- сфера внутренней политики;
- сфера правоохранительной и судебной деятельности;
- сфера предупреждения и ликвидации чрезвычайных ситуаций;

Основными из существующих угроз являются:

- нарушение технологии обработки информации;
- перехват информации в сетях передачи данных и на линиях связи;
- несанкционированный доступ к информации, находящейся в банках и базах данных.

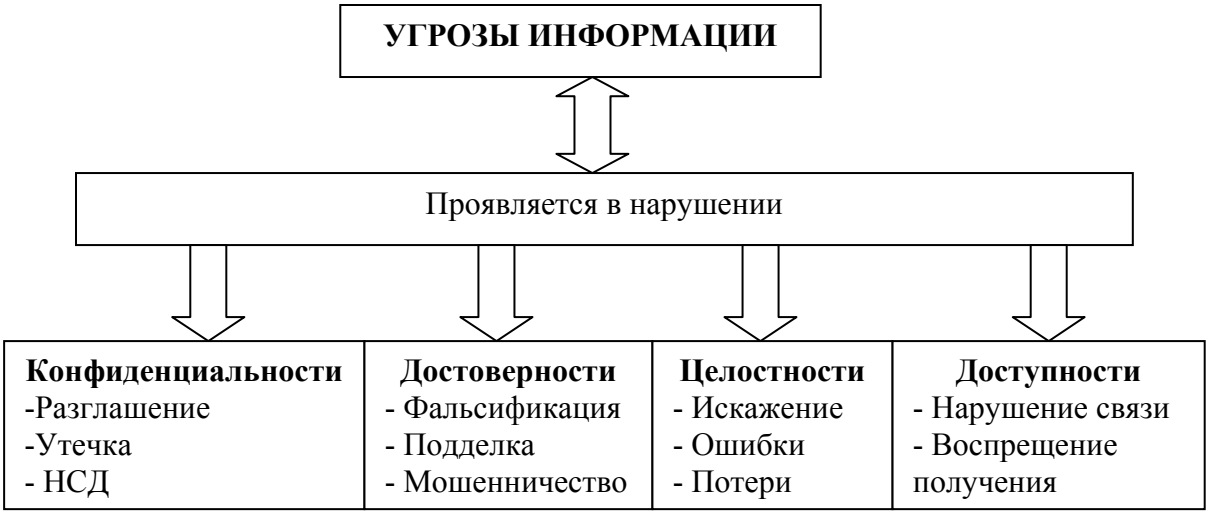


Рис .1. Угрозы информации

Реализация угроз безопасности информации относительно большинства информационных телекоммуникационных систем, АСУ технологическими процессами потенциально-опасных объектов, ведомственных сетей связи может привести к возникновению чрезвычайных ситуаций от локального до трансграничного масштаба, создать угрозу жизни людей и повлиять на экологическую безопасность Российской Федерации.

Оценка защищенности АСУ потенциально-опасными объектами и ИТКС в ключевых сегментах информационной структуры показывает следующее.

- Состояние и организация защиты АСУ и ИТКС в указанных сегментах информационной структуры требует безотлагательного решения проблемных вопросов с целью обеспечения безопасности и устойчивости их функционирования:

  - нет четкого определения широкого перечня внутренних и внешних угроз, нетрадиционных каналов утечки информации и несанкционированного доступа к ней, а также к АСУ и ИТКС;

  - имеет место неопределенность в правовом статусе и в уровне требований по защите информации конфиденциального характера и другой информации, циркулирующей в АСУ и ИТКС, критически важных сегментов информационной инфра-

структуры Российской Федерации, а также недостаточна правоприменительная практика в этой области;

- в ряде органов власти и организаций обработка информации ограниченного доступа в автоматизированных системах осуществляется с нарушением требований федеральных законов, руководящих и нормативных документов ФСТЭК России (ранее Гостехкомиссия России) в области обеспечения безопасности информации. Защита информации в основном сводится к использованию встроенных механизмов защиты операционных систем, резервному копированию, архивированию баз данных и использованию антивирусных программ. При этом внедрение АСУ потенциально-опасных объектов осуществляется без учета возможных угроз их функционированию, в том числе угроз информационного терроризма во всех его проявлениях;

- недостаточно финансирование мероприятий по обеспечению информационной безопасности государства;

- отсутствуют федеральные и региональные целевые программы в области информационной безопасности, прежде всего по защите АСУ и ИТКС в ключевых сегментах информационной инфраструктуры государства;

- существующая на государственном уровне межведомственная разобщенность ведут к дублированию выполняемых работ по информационной безопасности России.

- Уровень информационной безопасности этих систем значительно отстает от масштабов и темпов возрастания внутренних и внешних угроз.
- В случае реализации угроз информационной безопасности со стороны внешних или внутренних источников угроз возможны утечка конфиденциальной и технологической информации, ее искажение, нарушение целостности и доступности, что может привести к возникновению чрезвычайных ситуаций и нанести ущерб национальным интересам государства.
- Снижение эффективности системы образования и воспитания, недостаточное количество квалифицированных кадров в области обеспечения информационной безопасности.
- Недостаточное методическое руководство в организации работ по защите информации в органах власти и организациях со стороны вышестоящих ведомств и органов исполнительной власти.

Анализ состояния нормативно-правового обеспечения технической защиты информации (ТЗИ) показывает, что в настоящее время в Российской Федерации эту систему формирует:

- 26 государственных и межгосударственных стандартов ГОСТ и ГОСТ Р;
- 17 руководящих документов Гостехкомиссии России, не включая документы с грифом ДСП;
- 50 стандартов в оборонных отраслях промышленности и Министерстве обороны России.

Состояние указанных нормативных документов отстает от современного уровня развития информационных технологий и требуемого нормативного обеспечения, не соответствует современному уровню развития международной нормативно-правовой базы и не учитывает международный опыт и практику разработки нормативных документов в области информационной безопасности.

Чтобы сократить это отставание, в период с 1999 года по настоящее время ФСТЭК России совместно с другими министерствами и ведомствами внесли ряд предложений по совершенствованию нормативно-правовой и методологической базы в данной области. На основе международного стандарта ISO/IEC 15408:1999 «Критерии оценки безопасности информационных технологий» (Common Criteria for IT Security Evaluation) [3] были утверждены три государственных стандарта ГОСТ Р ИСО/МЭК

15408-1-2002 [4], 15408-2-2002 [5], 15408-3-2002 [6], которые определяют критерии оценки безопасности информационных технологий и позволяют определить требования по формированию заданий по безопасности в соответствии с положениями международных стандартов, и как ранее отмечалось, призваны предотвратить угрозы:

- несанкционированного раскрытия информации (обеспечение конфиденциальности);
- обеспечения достоверности информации;
- несанкционированного модифицирования и/или уничтожения информационно-программных ресурсов (обеспечение целостности);
- обеспечения своевременного и санкционированного получения информации (обеспечение доступности).

В развитие данных стандартов по заказу ФСТЭК был разработан ряд нормативных и методических документов, в которых зафиксированы современные решения и методологии:

- руководство по разработке профилей защиты (типовой набор сведений, которым должны удовлетворять продукты и/или системы определенного класса) и заданий по безопасности (совокупность требований к конкретной разработке) [7];
- руководство по регистрации профилей защиты [8];
- руководства по разработке семейств профилей защиты [9, 10].

  - Контролируемый доступ - Меточная защита - Многоуровневые операционные системы в средах, требующих средней робастности - Межсетевые экраны корпоративного уровня - Межсетевые экраны провайдерского уровня - Одноуровневые операционные системы в средах, требующих средней робастности - Клиентские операционные системы;

  - Системы управления базами данных - Средство защиты ресурсов компьютера от несанкционированного доступа на начальном этапе его загрузки - Средства построения виртуальных локальных вычислительных сетей - Средства построения виртуальных частных вычислительных сетей;

  - Билинговые системы - Средства доверенной загрузки ЭВМ - Системы обнаружения вторжений - Удостоверяющие центры - Инфраструктуры открытых ключей - Защита от несанкционированного доступа к информации.

  Данные профили защиты разработаны с учетом профилей, выработанных в других странах.
- Разработан и утвержден ГОСТ Р 34.10-94 [11], определяющий процессы формирования и проверки электронной цифровой подписи, который в скором будущем должен стать межгосударственным стандартом.
- Разработан проект классификатора техники защиты информации (средства защиты информации, средства контроля эффективности защиты информации, средства и системы управления) [12].
- Разработан проект государственного стандарта, определяющего общие положения по формированию системы управления качеством при разработке, изготовлении, внедрении и эксплуатации техники защиты информации [12].

В стадии разработки находятся следующие документы:[10]:

- концепция обеспечения безопасности изделий информационных технологий;
- положение по организации разработки, испытаний, производства и эксплуатации безопасных информационных технологий;
- базовая модель угроз безопасности ИТ;
- положение об оценке и сертификации ИТ;
- методология (типовые методики) оценки по общим критериям;
- проект «Программы комплексной стандартизации в области защиты информации, составляющей государственную тайну», расширяющий требования ФЗ «О техниче-

ском регулировании» [13] и ФЗ «О государственной тайне» [14] путем создания более 30 национальных стандартов и рекомендаций по стандартизации:

- общий технический регламент «Безопасность информации, составляющей государственную тайну»;
- общий технический регламент «Безопасность техники защиты объектов как носителей информации, составляющей государственную тайну»;
- специальный технический регламент «Безопасность информации, составляющей государственную тайну, для информационных и телекоммуникационных технологий»;
- специальный технический регламент «Безопасность информации, составляющей государственную тайну, для систем управления военного назначения».

В заключении следует отметить, что указанный пакет документов и профилей защиты должен составить целостную систему нормативно-методических документов по оценке безопасности информационных технологий, внедрение которых позволит обеспечить единую классификацию и кодирование техники защиты информации, что снизит разобщенность разработчиков и изготовителей, согласует их работу по разработке систем качества и обеспечит получение доступа на международный рынок сертифицированных продуктов и продукции, так как сложно обеспечивать национальную и тем более информационную безопасность Российской Федерации, используя информационные продукты и услуги, в разработке и предоставлении которых, потенциально могут участвовать спецслужбы иностранных государств.

Также возникает необходимость:

- законодательно определить статус информации, циркулирующей в АСУ и ИТКС в ключевых сегментах информационной инфраструктуры Российской Федерации;
- законодательно закрепить ответственность за выдачу сертификата на несоответствующие требованиям информационные продукты;
- разработать нормативы и методические документы, устанавливающие критерии отнесения информации к основным сегментам информационной инфраструктуры России (в том числе с учетом возможного ущерба в результате нарушения работоспособности АСУ и ИТКС);
- федеральным органам исполнительной власти, уполномоченным в области защиты информации, разработать целевые программы в области информационной безопасности, прежде всего по защите АСУ и ИТКС в сегментах информационной инфраструктуры государства, и принять соответствующие нормативные документы, конкретизирующие необходимые требования по их технической защите;
- пересмотреть устаревшие нормы и стандарты, а также федеральные целевые программы по развитию информационной сферы государства и уточнить их разделы в области информационной безопасности;
- усилить работу по контролю за привлечением к разработке и внедрению АСУ в защищенном исполнении и средств защиты информации только организаций, имеющих лицензию федерального органа исполнительной власти, уполномоченного в области защиты информации.

Литература

1. Ярочкин В.И. Информационная безопасность: Учебник для студентов вузов. М.: Академический проект; Гаудеамус, 2-е изд. 2004.
2. Доктрина информационной безопасности Российской Федерации» (утв. Президентом РФ 09.09.2000 N Пр-1895) // Российская газета, № 187, 28.09.2000.

3. Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model; Part 2: Security functional requirements; Part 3: Security assurance requirements. – ISO/IEC 15408-1-2-3-1999.
4. ГОСТ Р ИСО/МЭК 15408-1-2002. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 1. Ведение и общая модель. М.: ИПК Издательство стандартов, 2002.
5. ГОСТ Р ИСО/МЭК 15408-2-2002. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 2. Функциональные требования безопасности. М.: ИПК Издательство стандартов, 2002.
6. ГОСТ Р ИСО/МЭК 15408-3-2002. Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Часть 3. Требования доверия к безопасности. М.: ИПК Издательство стандартов, 2002.
7. Руководящий документ. Руководство по разработке профилей защиты и заданий по безопасности. Гостехкомиссия России, 2003. http://www.gostexkom.ru/_ispo.htm
8. Руководящий документ. Безопасность информационных технологий. Руководство по регистрации профилей защиты. Гостехкомиссия России, 2003. http://www.gostexkom.ru/_ispo.htm
9. Руководящий документ. Безопасность информационных технологий. Руководство по формированию семейств профилей защиты. Гостехкомиссия России, 2003. http://www.gostexkom.ru/_ispo.htm
10. Калайда И.А. Состояние и перспективы развития нормативно-методической базы в области безопасности информационных технологий (08.04.2004). http://www.infoforum.ru/detail.php?pagedetail=895
11. ГОСТ Р 34.10-94 Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма. Госстандарт России.
12. Отчет об итогах работы Всероссийского научно-исследовательского института стандартизации за 2003 год. http://www.vniistandart.com/general/inst/2003.shtml?05
13. Федеральный закон от 27 декабря 2002 г. N 184-ФЗ «О техническом регулировании» // Российская газета. 31 декабря 2002. № 245.
14. Закон РФ от 21 июля 1993 г. N 5485-1 «О Государственной тайне» // Российская газета. 21 сентября 1993.
15. Галатенко В.А. Основы информационной безопасности / Под редакцией члена-корреспондента РАН В.Б. Бетелина. М.: ИНСТИТУТ.РУ «Интернет-Университет Информационных Технологий», 2003.
16. Галатенко В.А. Стандарты информационной безопасности / Под редакцией члена-корреспондента РАН В.Б. Бетелина. М.: ИНСТИТУТ.РУ «Интернет-Университет Информационных Технологий», 2003.

К ИССЛЕДОВАНИЮ МОДЕЛИ АДАПТИВНОЙ ЗАЩИТЫ СИСТЕМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Ф.Г. Нестерук, Л.Г Нестерук, Г.Ф. Нестерук, С.И. Воскресенский

Эволюция систем информационных технологий (ИТ) привела к разработке технических систем, в которых присутствуют биоподобные процессы наследования, адаптации, развития и отбора. Предложена модель адаптивной защиты ИТ-систем. На базе модели адаптивной защиты обсуждаются технические и экономические аспекты обеспечения информационной безопасности ИТ-систем.

Введение

В технических системах оправдывает себя использование специфики организации биосистем [1, 2], которые характеризуются иерархией средств жизнеобеспечения, эволюционными процессами и встроенными механизмами адаптивной памяти, иммунитета, защиты информации и избыточности.

Адаптивные механизмы памяти, позволяющие накапливать жизненный опыт, связаны с распределенными избыточными информационными полями нейронных комплексов нервной системы [3]. Реализация адаптивных механизмов памяти в распределенных информационных полях нейронных сетей (НС) является основной предпосылкой эволюции ИТ-систем, т.е. придания им атрибутов наследования, адаптации, развития и отбора.

Основной архитектурной особенностью биосистем является внутрисистемный характер механизмов защиты (МЗ), реализованный в иерархии ИТ-системы. Поэтому при моделировании искусственных систем следует учитывать, что функции защиты информации должны быть встроенными функциями проектируемой ИТ-системы для реализации адаптивных механизмов иммунной системы и информационных полей нервной системы.

Обеспечение информационной безопасности современных ИТ-систем по возможностям уступает биологическим прототипам, поэтому разработка и моделирование адаптивных СИБ, основанных на биосистемной аналогии, актуальны.

В работе рассмотрена модель адаптивной защиты ИТ-систем, учитывающая взаимосвязь событий: источник угрозы – фактор (уязвимость) – угроза (действие) – последствия (атака). В процессе эксплуатации защищаемой системы при изменении поля угроз проявляются ранее скрытые уязвимости, не отраженные в исходной модели, и возникает возможность причинения ущерба хозяйствующему субъекту. Поэтому модель ИТ-системы должна обладать свойством адаптивности, а методика проведения инвестирования в систему информационной безопасности (СИБ) – предусматривать оценку потенциального ущерба от реализации угроз, в соответствии с которой определять, какие механизмы защиты необходимо включать в конкретные уровни СИБ для предотвращения ущерба [4].

Предлагаются к обсуждению технические и экономические аспекты обеспечения информационной безопасности ИТ-систем, а именно: решение задачи четкой и нечеткой классификации угроз по признакам атак и МЗ на поле известных угроз; разработка показателей защищенности и оценок эффективности инвестиций в СИБ; разработка инструментальных средств и методики их применения для оптимизации затрат на СИБ.

Модель адаптивной системы информационной безопасности

Связующим звеном модели является методика оценки защищенности ИТ-системы, которая координирует взаимосвязь классификаторов угроз и механизмов защиты (в виде нейронных сетей – НС, нечетких НС, систем нечетких предикатных пра-

вил), структурной модели СИБ, инструментальных средств расчета показателей защищенности и рейтинга ИТ-системы [5, 6].

Адаптивность позволяет при ограниченных затратах на организацию СИБ обеспечить заданный уровень безопасности ИТ-системы за счет реакции на изменение поля угроз. Не менее важным качеством является возможность накопления и передачи опыта системой информационной безопасности, который овеществляется в виде информационно-полевой компоненты иерархии СИБ. Распределенные информационные поля нейронных и нейро-нечетких сетей иммунного и рецепторного уровней иерархии СИБ аккумулируют знания в процессе развития ИТ-системы и адаптации к изменению поля угроз и могут передаваться в последующие версии ИТ-системы (процесс наследования).

В соответствии с заданием на проектирование системы защиты выбирается структурная модель СИБ в виде иерархии уровней механизмов защиты (МЗ). Как правило, ограничиваются минимальным комплектом МЗ, достаточным для отражения угроз информационным ресурсам, оговоренных в спецификации на проектирование ИТ-системы.

Опыт экспертов представляется матрицами экспертных оценок, на базе которых формируются системы нечетких предикатных правил для классификации угроз по признакам атак и МЗ на поле угроз. Системы нечетких предикатных правил для последующей адаптации и анализа представляются в виде нейро-нечетких классификаторов, которые обучают на некотором подмножестве входных векторов признаков атаки. Параллельно обучают четкие нейросетевые классификаторы таким образом, чтобы число образуемых кластеров равнялось числу правил в системе нечетких предикатных правил.

Для исходных матриц экспертных оценок производят расчет показателей защищенности и рейтинга ИТ-системы [5, 6], которые используются методикой оценки защищенности ИТ-системы для анализа и коррекции как матриц экспертных оценок, так и функциональных параметров нейросетевых и нейро-нечетких классификаторов (систем нечетких предикатных правил).

Информация в адаптивной СИБ хранится и может передаваться в поколениях (тиражирование и последующие модификации ИТ-системы) в виде распределенных адаптивных информационных полей НС: 1) поля известных угроз иммунных уровней защиты и 2) поля жизненного опыта рецепторных уровней защиты. Процесс адаптации первых связан с решением задач классификации, кластеризации, приводящих к расширению информационного поля известных угроз на нижних уровнях иерархии СИБ. Изменение перечня известных угроз ИБ отражается на верхних уровнях иерархии СИБ в соответствующей модификации информационного поля жизненного опыта, реализованного в виде специализированных структур нечетких НС, которые, в свою очередь, описываются системами нечетких предикатных правил. Процесс адаптации вторых связан с обучением нечетких НС, которые адекватно корректируют систему нечетких предикатных правил, ставящую в соответствие известным угрозам механизмы защиты информации.

Нейро-нечеткий классификатор

В качестве нечетких классификаторов можно использовать нейро-нечеткие сети [7]. Нейро-нечеткий классификатор m-мерных нормализованных векторов (например, угроз) $\boldsymbol{X}$ с нечеткими координаторами $(\widetilde{x}_1,...,\widetilde{x}_m)$ будем представлять в виде трехслойной нечеткой НС (рис. 1).

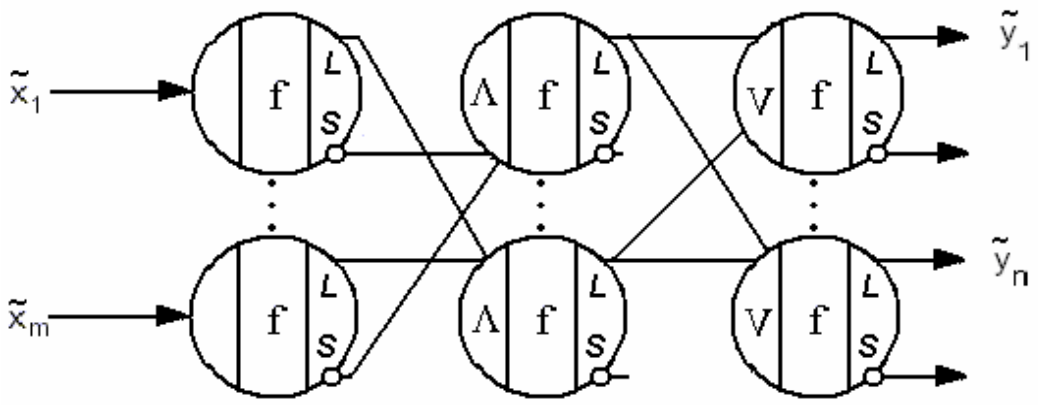


Рис. 1. Структура нейро-нечеткого классификатора адаптивной СИБ

В полном пространстве предпосылок $X = \{\tilde{x}_1,..., \tilde{x}_m\}$ максимальное число входных нечетких векторов задается всевозможными сочетаниями координат $\tilde{x}_i, \quad i = 1...m$. Каждому входному вектору из пространства *X* можно поставить в соответствие нечеткий формальный нейрон (ФН) классификатора, выполняющий операцию логического вывода, например, *min*. Отображение множества результатов логического вывода в полное пространство заключений $Y = \{\tilde{y}_1,..., \tilde{y}_n\}$ можно реализовать через операцию композиции, и каждому выходному вектору (например, механизмов защиты - МЗ) из пространства *Y* можно поставить в соответствие нечеткий ФН классификатора, выполняющий операцию, к примеру, *max*.

В классификаторе первый слой содержит *m*, по числу координат входного вектора угроз, нечетких ФН с комплементарными (взаимодополняющими) нечеткими связями (НЧС), формирующих *m* пар нечетких высказываний вида: $\tilde{x}_i$ *есть S*, $\tilde{x}_i$ *есть L*, $i = 1...m$, где *S* – small, *L* - large; средний слой содержит до 2^m нечетких ФН, выполняющих операцию логического вывода (например, *min*) над сочетаниями нечетких высказываний – НВ первого слоя НС для формирования системы нечетких классификационных заключений; выходной слой содержит *n*, по числу координат выходного вектора МЗ, нечетких ФН, выполняющих операцию композиции (например, *max*) над нечеткими классификационными заключениями второго слоя НС для формирования *n*-мерных векторов ***Y*** выходных нечетких заключений $(\tilde{y}_1,..., \tilde{y}_n)$.В первом слое нечеткие ФН в соответствии с заданным типом функции принадлежности μ формируют комплементарные пары значений истинности для входных нечетких переменных – НП $\tilde{x}_i, \quad i = 1...m,$ координат входного вектора ***X***.

При заданном значении координаты вектора ***X*** на отрезке области определения каждому значению входной НП соответствует значение ординат функций принадлежности *S* (small) и *L* (large), которые в сумме дают 1 (рис. 2). Другими словами, каждый нечеткий ФН 1-го слоя реализует пару «частично противоположных» нечетких высказываний, которые через комплементарную пару нечетких связей подаются на средний слой НС. Нечеткая связь представляет собой совокупность локальных связей на выходе ФН, веса которых отражают распределение истинности в НМ, соответствующем некоторой функции принадлежности. Пара функций принадлежности, например *S* и *L,* образует 2 нечеткие связи, составляющие одну комплементарную нечеткую связь.

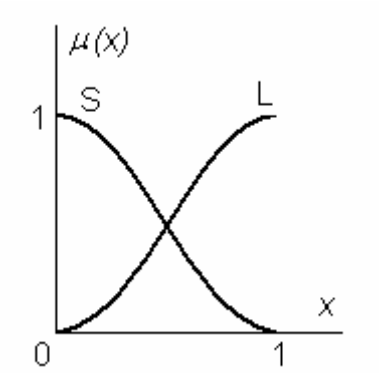


Рис. 2. Комплементарная пара функций принадлежности

Если во втором слое НС содержится максимальное число нечетких ФН «И», то промежуточный вектор нечетких заключений будет содержать всевозможные нечеткие классификационные заключения, которые следуют из векторов входных посылок.

Третий слой НС образован из нечетких ФН «ИЛИ» (по числу нечетких заключений $\tilde{y}_j$, $j=1...n$) и формирует вектор выходных нечетких заключений в соответствии с заданной экспертами системой нечетких предикатных правил.

Для рассмотренной модели разработаны комплекс показателей защищенности ИТ-системы и инструментальные средства [8], использующие полученные оценки для инвестиционного анализа СИБ, которые используются в ИТ-системах для определения (путем моделирования) положения МЗ, включение которых в иерархию СИБ предотвратит появление ущерба, превышающего допустимый для хозяйствующего субъекта уровень.

Показатели применимы для оценки защищенности систем по полю известных угроз и по подмножеству поля угроз: нарушения целостности, конфиденциальности, доступности информации. На рис. 3 приведены рейтинговые оценки автоматизированных систем (АС) по классам защищенности РД ФСТЭК. Здесь Rm и Re – рейтинговые оценки защищенности ИТ-системы в разрезе, соответственно, механизмов защиты и эшелонов (уровней иерархии) СИБ. Анализ приведенных зависимостей подтверждает адекватность показателей защищенности, используемых в модели, известным оценкам защищенности АС [9].

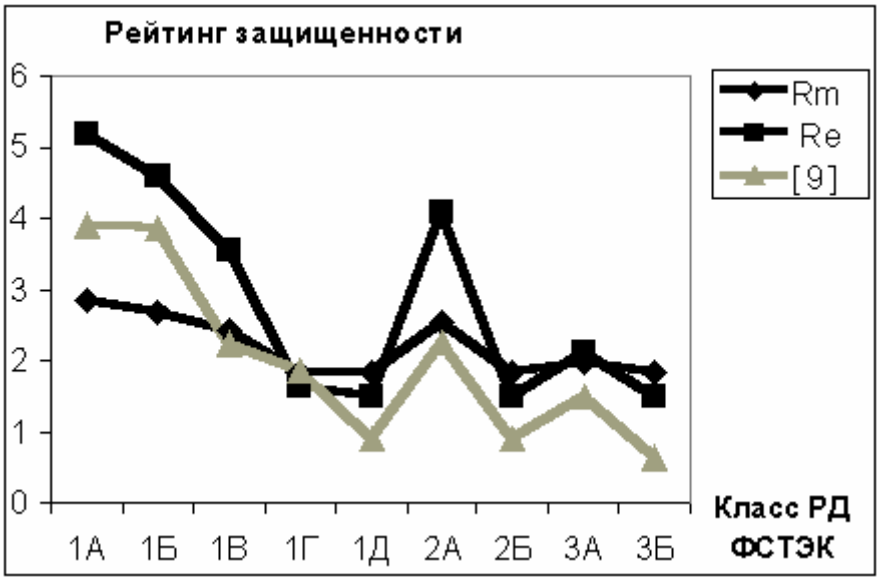


Рис. 3. Рейтинговые оценки АС по классам защищенности РД ФСТЭК

Проведена сравнительная оценка альтернативных инвестиционных проектов (модернизации конкретной СИБ путем введения комплексных средств защиты информации «Кобра» и «Спектр-Z»), учитывающая показатели чистого приведенного эффекта, сроков окупаемости и индекса рентабельности проектов модернизации СИБ (рис. 4).

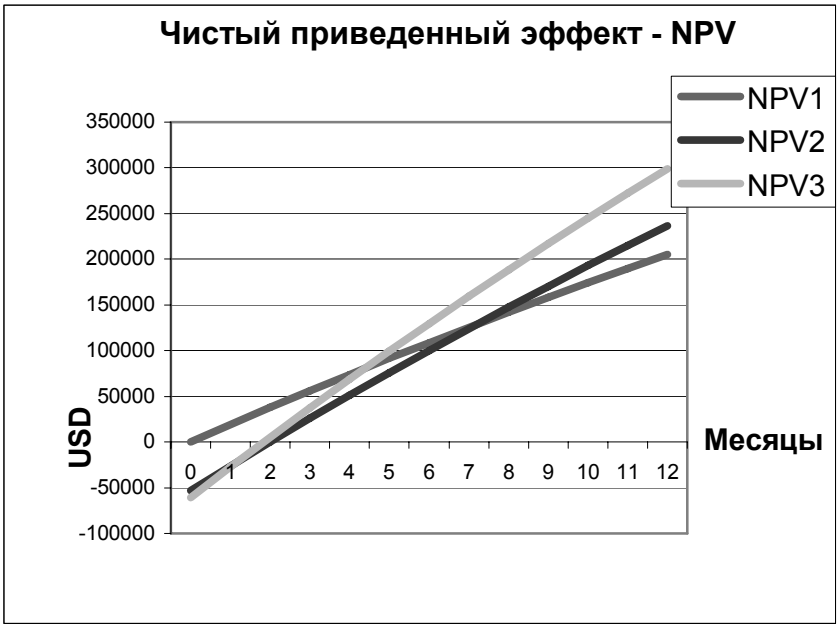


Рис. 4. Сравнение альтернативных проектов модернизации СИБ

Анализ зависимостей позволяет сделать вывод, что системам информационной безопасности, характеризующимся более высокими значениями рейтинговых показателей (на рис. 4 – проект NPV3), свойственны более короткие сроки окупаемости затрат.

В работе рассмотрены модель адаптивной защиты ИТ-систем, а также технические и экономические аспекты обеспечения информационной безопасности ИТ-систем, включая решение задачи четкой и нечеткой классификации для иммунного и рецепторного уровней СИБ, разработку комплекса показателей защищенности и эффективности инвестиций в СИБ, инструментальных средств, применяемых для оптимизации затрат на СИБ.

Литература

1. Осовецкий Л.Г., Нестерук Г.Ф., Бормотов В.М. К вопросу иммунологии сложных информационных систем // Изв. вузов. Приборостроение. 2003. Т.46, № 7. С. 34–40.
2. Нестерук Г.Ф., Осовецкий Л.Г., Нестерук Ф.Г. О применении нейро-нечетких сетей в адаптивных системах информационной защиты. // В сб. «Нейроинформатика-2005»: VII Всероссийская научно-техническая конференция. 26–28 января 2005. М.: МИФИ, 2005. Ч. 1. С.71–79.
3. Мелик-Гайназян И.В. Информационные процессы и реальность. М.: Наука, 1998. 192 с.
4. Нестерук Л.Г., Нестерук Т.Н. Применение нейро-нечетких сетей для анализа инвестиционных процессов // В сб. «Нейроинформатика-2005»: VII Всероссийская научно-техническая конференция. 26–28 января 2005. М.: МИФИ, 2005. Ч.2. С. 221–228.
5. Нестерук Ф.Г., Осовецкий Л.Г., Жигулин Г.П., Нестерук Т.Н. Разработка комплекса показателей для оценки информационных ресурсов и безопасности иерархиче-

ских систем // РИ-2004: Материалы IX Санкт-Петербургской международной конференции «Региональная информатика - 2004». СПб: СПИИ РАН, 2004.
6. Нестерук Г.Ф., Осовецкий Л.Г., Нестерук Ф.Г. К оценке защищенности систем информационных технологий // Перспективные информационные технологии и интеллектуальные системы. 2004. № 1.
7. Fuller R. Neural Fuzzy Systems. - Abo: Abo Akademi University, 1995.
8. Нестерук Г.Ф., Куприянов М.С., Осовецкий Л.Г., Нестерук Ф.Г. Разработка инструментальных средств для оценки защищенности информационных систем с применением механизмов нечеткой логики и нейронных сетей // SCM'2004: Сборник докладов VII Международной конференции по мягким вычислениям и измерениям. Т.1. СПб.: СПбГЭТУ, 2004.
9. Осовецкий Л., Шевченко В. Оценка защищенности сетей и систем // Экспресс-электроника. 2002. № 2–3. С.20–24.

ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ТРИАНГУЛЯЦИИ ДЕЛОНЕ ПРИ ПОСТРОЕНИИ ПОВЕРХНОСТЕЙ НЕОГРАНИЧЕННОГО РАЗМЕРА

В.Н. Блохин, В.Е. Бочков

Рассматривается задача построения трехмерных моделей поверхностей. Описывается подход к построению триангуляции Делоне, позволяющей производить триангуляцию над неограниченным объемом исходных данных. В качестве алгоритма триангуляции используется инкрементальный алгоритм с разбиением исходного множества точек. Описывается реализация приложения TriOr для проведения триангуляции средствами СУБД Oracle.

Создание дискретных векторных моделей поверхностей, основанных на множествах исходных точек, является весьма распространенной задачей, возникающей в большом количестве прикладных областей науки. В настоящий момент существует довольно много различных подходов к решению этой задачи. Однако все их можно классифицировать по следующему существенному критерию: степень точности полученной модели. При этом наиболее простым в реализации подходом, результатом использования которого является приближенная модель, является моделирование поверхности методом построения регулярной матрицы. Подходы, которые позволяют добиться точного соответствия полученной модели исходному множеству вершин, как правило, основаны на применении диаграмм Воронова или обратной задачи – триангуляции Делоне. На данный момент применение триангуляции Делоне является наиболее оптимальным методом для построения поверхностей.

По сравнению с другими видами моделирования этот метод позволяет добиться наиболее точной модели. Формальное описание задачи Делоне будет изложено далее.

Существенным недостатком методов, основанных на решении задачи Делоне, является проблема обработки больших объемов исходных данных: большие временные затраты, а также высокие требования к объему используемой оперативной памяти осложняют применение триангуляции. Экстенсивным путем решения описанных недостатков является повышение производительности используемых при расчетах компьютеров, однако задача преодоления верхней границы объема исходных данных будет присутствовать постоянно.

В данной работе мы будем рассматривать задачу моделирования поверхностей применительно к моделям рельефа местности. Будет дана краткая аннотация приложения TriOr, предназначенного для проведения триангуляции, а также оценка производительности инкрементального алгоритма с разбиением.

Формальное определение триангуляции Делоне

Задачу моделирования поверхности можно сформулировать следующим образом: пусть дана поверхность S, определенная в n-мерном пространстве (на практике моделирование поверхностей производится в трехмерном пространстве), и множество N точек, принадлежащих поверхности S. Требуется построить поверхность S', где Ni принадлежит S', такую, чтобы для произвольной точки M, принадлежащей S', выполнялось следующее условие: нормаль, отложенная из M на S, имеет минимальную длину. Другими словами, необходимо получить функцию F такую, что $S' = F(N)$.

Задача триангуляции исходного дискретного множества точек была впервые сформулирована советским математиком Делоне [1]: Триангуляцией набора N, состоящего из n точек, называется сетка примыкающих друг к другу треугольников, таких, что вершинами любого треугольника являются точки из набора N, и ребра треугольников не могут пересекаться в других точках, не принадлежащих N. Точки и ребра, принадлежащие выпуклой оболочке $CH(S)$, называются граничными, остальные – внутрен-

ними. Обозначим общее количество внутренних точек через v. Основным свойством триангуляции является то, что при любом способе триангуляции общее число треугольников остается постоянным и равным $K= n+v-2$.

Триангуляция удовлетворяет условию Делоне, если описанная окружность (сфера в случае триангуляции в трехмерном пространстве), проведенная вокруг любого треугольника из набора K, не содержит внутри других точек набора N. Отсутствие попадания точки точно на любую описанную окружность приводит к единственной триангуляции удовлетворяющей условию Делоне.

Отметим также еще два важных свойства триангуляции Делоне [2]:

- максимальная сумма минимальных углов всех составляющих треугольников среди всех возможных триангуляций исходного множества;
- минимальная сумма радиусов описанных вокруг треугольников окружностей среди всех возможных триангуляций.

Как нетрудно заметить, результатом триангуляции всегда будет являться планарный граф, все внутренние области которого являются треугольниками. Если учесть, что на практике при построении моделей поверхности рельефа местности используются отметки высот местности, которые, по сути, являются точками экстремума рельефа, то применение трехмерной триангуляции Делоне позволяет построить каркасную модель рельефа местности.

Именно триангуляция Делоне позволяет получить модель, обладающую наивысшим качеством аппроксимации к исходной поверхности. Более того, применение диаграмм Воронова [3] совместно с методом кубических сплайнов [4] позволяет провести детализацию триангуляции внутри уже полученных треугольников с последующей интерполяцией высот в полученных вершинах, что, в свою очередь, значительно повышает степень аппроксимации полученной поверхности к исходной.

Выбор оптимального алгоритма

В настоящее время описано достаточно большое количество алгоритмов триангуляции, большинство из них очень подробно освещены в монографии А.В. Скворцова [2]. В рамках данной работы мы рассмотрим лишь те особенности реализации алгоритмов, которые могут существенно повлиять на выбор алгоритма, наиболее оптимального для обработки неограниченных множеств исходных точек.

Наиболее существенным для нас признаком классификации алгоритмов следует признать необходимость проведения так называемой начальной триангуляции, не удовлетворяющей условию описанной окружности, с последующим приведением к триангуляции Делоне. Такой подход характерен для семейства итеративных алгоритмов. Именно это не позволяет в полной мере использовать индексацию исходного множества точек как средство повышения производительности вычислений.

Данного недостатка лишены так называемые прямые инкрементальные алгоритмы. Данный класс алгоритмов основан на пошаговом построении треугольников, удовлетворяющих условию Делоне, что позволяет обойтись без рекурсивного возврата к обработанным вершинам. Однако существенным недостатком инкрементальных алгоритмов является необходимость перебора всего множества необработанных вершин на каждой итерации алгоритма.

Таким образом, использование данного алгоритма, с одной стороны, позволяет упростить процесс вычислений, но, с другой стороны, в базовой реализации не позволяет использовать его при больших объемах исходных точек. Для использования инкрементального алгоритма необходимо предварительное разбиение исходного множества, позволяющее сократить обрабатываемых вершин.

Реализация инкрементального алгоритма в приложении TriOr. Исследование производительности

При выборе средств для практической реализации задачи моделирования поверхностей методом триангуляции Делоне авторы руководствовались следующими критериями:

- возможность работы как с тестовыми массивами, так и с реальными исходными данными (в качестве исходных данных для построения моделей земного рельефа предполагалось использовать цифровые векторные карты);
- возможность пространственного разбиения исходного множества точек на области, содержащие подмножества исходного массива;
- «прозрачная» реализация приложения с возможностью дальнейшей модернизации алгоритма триангуляции или его полной заменой;
- простота представления полученной модели.

Также было необходимо учитывать, что хранение исходных цифровых векторных карт осуществлялось в хранилище пространственных данных СУБД Oracle.

После проведения анализа существующих средств разработки приложений было выработано следующее решение:

- приложение должно быть полностью реализовано средствами языка Java;
- запуск и выполнение приложения должны осуществляться непосредственно внутри СУБД Oracle;
- полученные данные должны визуально представляться в виде трехмерных моделей формата VRML (Virtual Reality Modeling Language).

Обоснуем изложенное решение. Применение СУБД Oracle в качестве хранилища пространственных данных позволяет использовать методы пространственной индексации, предусмотренные в СУБД, для разбиения исходного множества точек на области обработки. Однако оптимальное использование индексации достигается лишь при проведении многократных запросов к исходному множеству. Внешняя реализация алгоритма триангуляции по отношению к СУБД влечет повышение трафика обмена данными между приложением и базой данных.

СУБД Oracle позволяет интегрировать написанные на языке Java приложения в состав средств вычислений базы данных. Это позволяет минимизировать затраты времени на получение данных, максимально использовать встроенный инструментарий СУБД и, следовательно, максимально повысить производительность приложения.

Формат VRML для публикации полученных поверхностей был выбран в связи с тем, что на данный момент он является фактически единственным открытым стандартом хранения и визуализации трехмерных моделей. Помимо этого, существует расширение стандарта VRML – Geographic VRML, позволяющее без каких-либо преобразований координат и потерь точности визуализировать трехмерные модели земной поверхности неограниченной площади и детальности.

Все вышеизложенные аргументы были учтены при разработке приложения TriOr. При разбиении был применен алгоритм пространственного индексирования методом построения квадродерева. Облако точек разбивается на кубы, включающие подмножества точек, равные по количеству элементов. Допускается регулирование точности разбиения, а также размеров кубов.

Процесс триангуляции начинается в произвольном кубе и производится последовательно во всех кубах-соседях. При поиске вершин треугольников производится перебор точек, находящихся в одном кубе с рассматриваемой вершиной. Алгоритм допускает нахождение вершин за пределами данного куба для вершин, лежащих вблизи его границ, после неудачи внутри рассматриваемой области. Именно поэтому большую роль играет метод индексации исходного облака точек.

Приложение TriOr было апробировано на множествах исходных точек, полученных из узлов изолиний рельефа, и отметок высот, извлеченных из цифровых векторных карт различных масштабов.

Экспериментальным путем было определено, что для карт масштабов 1:25000 и менее можно сделать допущение и перейти от проверки условия Делоне в трехмерном пространстве к проверке на плоскости (от описываемого шара к описываемой окружности). Это связано с шагом рельефа (относительные высоты между горизонталями), который в данных масштабах пренебрежимо мал по сравнению с расстояниями между точками в плане. Такое допущение абсолютно исключено при моделировании поверхности на основе карт масштабов 1:1000 и крупнее и может привести к отклонениям полученной модели от реальной поверхности.

Были проведены сравнительные исследования производительности алгоритма с применением разбиения и без него. Для исследований использовался компьютер Pentium4, оснащенный процессором 1600 Mhz и 1 Гб ОЗУ. При обработке множества, состоящего из 10000 элементов, получены следующие результаты: без использования разбиения – 6 минут; с разбиением – 4 минуты. При обработке множества, состоящего из 500000 элементов, получены следующие результаты: без использования разбиения – отказ, связанный с нехваткой оперативной памяти на 5 часе работы; с разбиением – 2 часа 30 минут.

Данные результаты говорят о высокой производительности алгоритма и о возможности его использования при моделировании поверхностей неограниченного размера.

Заключение

При экспериментальных исследованиях мы практически ушли от построения моделей в трехмерном пространстве. Это допущение было сделано для скорейшего подтверждения повышения производительности при применении разбиения.

Авторы считают необходимым апробировать приложение TriOr при моделировании трехмерных объектов на основе данных лазерного сканирования [5]. Лазерное сканирование на данный момент является самой современной технологией проведения геодезических изысканий. В процессе проведения работ по лазерному сканированию непосредственно формируется трехмерное облако точек, отображающее объектовый состав и рельеф местности. Количество элементов такого облака может доходить до миллиардов, а учет вертикальной составляющей строго обязателен. Применение материалов лазерного сканирования в качестве исходных данных позволит объективно оценить производительность приложения при трехмерной триангуляции.

Литература

1. Делоне Б.Н. О пустоте сферы. Изв. АН СССР. ОМЕН. 1934. № 4. С. 793–800.
2. Скворцов А.В. Триангуляция Делоне и ее применение. Томск. Издательство ТГУ. 2002.
3. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение: Пер. с англ. Москва: Мир, 1989.
4. Синицын С.И. Гибридный рекурсивно-инкрементный алгоритм построения триангуляции Делоне. Материалы конференции GraphiCon. Нижний Новгород. 2001.
5. Черновцев А.А. К вопросу о точности сканирующих систем. // Информационный бюллетень ГИС-Асооциации. Москва. 2004. №4(46).

СТРУКТУРИРОВАНИЕ ПРОГРАММ И ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ НА МНОЖЕСТВО ЛИНЕЙНЫХ И УСЛОВНЫХ ВЕРШИН

А.Г. Зыков, О.Ф. Немолочнов, В.И. Поляков, А.В. Сидоров

В работе предложена модель вычислительного процесса программы в виде структурированного графа и методы построения графо-аналитической модели. Данная модель позволяет в дальнейшем применить математический аппарат исчисления кубических комплексов для решения различных задач анализа вычислительных процессов, порождаемых программами, таких как, например, тестирование или верификация.

Введение

Современные требования к качеству программных изделий заставляют разработчиков уделять все большее внимание технологическому процессу разработки, а также вопросам тестирования и верификации программ. Разработка моделей и эффективных методов анализа программ является актуальной проблемой в условиях ужесточения требований к качеству программного продукта. Предложенное в данной работе представление программы в виде структурированного графа позволяет перейти от программного кода к математической модели более высокого уровня, которая может использоваться для проведения различных исследований программного кода с применением математического аппарата исчисления кубических комплексов [1, 2].

Неструктурированный граф программы

Программа, реализующая некоторый вычислительный процесс, может быть представлена в виде графа [3]. Каждая команда есть вершина графа, а дуги графа есть переходы между командами. В таком графе дуги образуются либо в порядке следования операторов, либо с помощью условной и безусловной передачи управления, нарушающей естественный ход выполнения программы. На графе выделяются начальная и конечная вершины, содержащие, соответственно, точки входа и выхода из программы. Будем называть такой граф неструктурированным графом программы.

Рассмотрим некоторый язык программирования, состоящий из набора команд, которые можно условно разделить на три группы:

$<команда> = < kLin > | < kJump > | < kIfJump >,$

где *kLin* – любая команда, не являющаяся командой передачи управления; *kJump* – команда безусловной передачи управления; *kIfJump* – команда условной передачи управления. Каждая команда имеет текущий адрес (или уникальную метку, которая обязательно должна присутствовать у команд, на которые возможен условный или безусловный переход):

$AD_i : <команда>,$

где AD_i – текущий адрес команды (или метка).

Неструктурированный граф программы (*GP*), написанной на таком языке, будет определен следующими соотношениями:

$GP = <VG, DG>,$

где *VG* – множество вершин, *DG* – множество дуг;

$VG = \{ \{ < kLin > \} \cup \{ < kJump > \} \cup \{ < kIfJump > \} \};$

$DG = \{ \{ v_1 \} \cup \{ v_2 \} \cup \{ v_3 \} \}.$

Здесь $\{v_1\}$ – множество дуг, соответствующих линейному порядку выполнения команд; $\{v_2\}$ – множество дуг, соответствующих командам безусловного перехода; $\{v_3\}$ – множество дуг, соответствующих командам условного перехода (по две дуги для каждой команды).

Структурированный граф программы

Неструктурированный граф, определенный указанными выше соотношениями, является полной и наиболее подробной моделью программы. Однако, в силу своей подробности, он является малоинформативным. Для повышения информативности графа некоторые группы команд можно объединить и воспринимать в дальнейшем как единые неделимые вершины нового графа. Процесс объединения вершин графа программы будем называть структурированием, а граф, получаемый в результате структурирования, – структурированным графом программы (*SGP*). В результате структурирования графа программы из рассмотрения исчезнут как отдельные вершины-команды, так и дуги, последовательно связывающие их между собой.

В качестве вершин структурированного графа будем использовать линейные (*Lin*) и условные (*IfLin*) вершины. Линейная вершина содержит одну точку входа и одну точку выхода. Условная вершина имеет одну точку входа и две точки выхода, задающие адреса ветвления, в зависимости от выполнения или невыполнения условия, задаваемого в вершине. В общем случае, условная вершина может содержать не только одну команду условного перехода, но и несколько команд, формирующих само условие:

$SGP = <VG_i, DG_j>,$

где VG_i - множество вершин, DG_j - множество дуг;

$VG_1 = \{ < Lin > \} \cup \{ < IfLin > \}.$

Линейные вершины

Чтобы сформулировать правила выделения линейных вершин, опишем свойства, которыми должна обладать любая линейная вершина. Линейная вершина должна содержать линейную последовательность команд, в которой нет входов и выходов в нее и из нее, отличных от начального входа и конечного выхода.

Исходя из вышесказанного, можно сформулировать условия начала и окончания линейной вершины. Началом линейной вершины должна быть одна из следующих команд:

1. команда входа в программу (например, из операционной системы);
2. команда, которая следует непосредственно за командами условной (*kIfJump*) или безусловной передачи управления (*kJump*);
3. команда, на которую замыкаются ветви, формируемые командами условной (*kJump*) и безусловной передачи управления (*kJump*).

Окончанием линейной вершины должна быть одна из следующих команд:

1. команда безусловной передачи управления (*kJump*);
2. команда, предшествующая команде, на адрес которой замыкаются ветви, формируемые командами условной (*kIfJump*) и безусловной передачи управления (*kJump*);
3. команда выхода из программы, например, возврат в операционную систему;
4. команда, предшествующая команде начала условной или линейной вершин.

Четвертое условие нуждается в уточнении. Дело в том, что в условную вершину, как было отмечено выше, может включаться некоторая линейная часть, формирующая само условие. Определение линейной части условной вершины должно производиться по некоторому набору эмпирических правил. Например, это может быть максимально длинная последовательность команд без записи промежуточных результатов. Кроме того, некоторые длинные линейные последовательности могут быть искусственно разбиты на составляющие, например, по записи глобальных переменных, реализующих те или иные линейные формулы, т.е. формулы без интервальных условий.

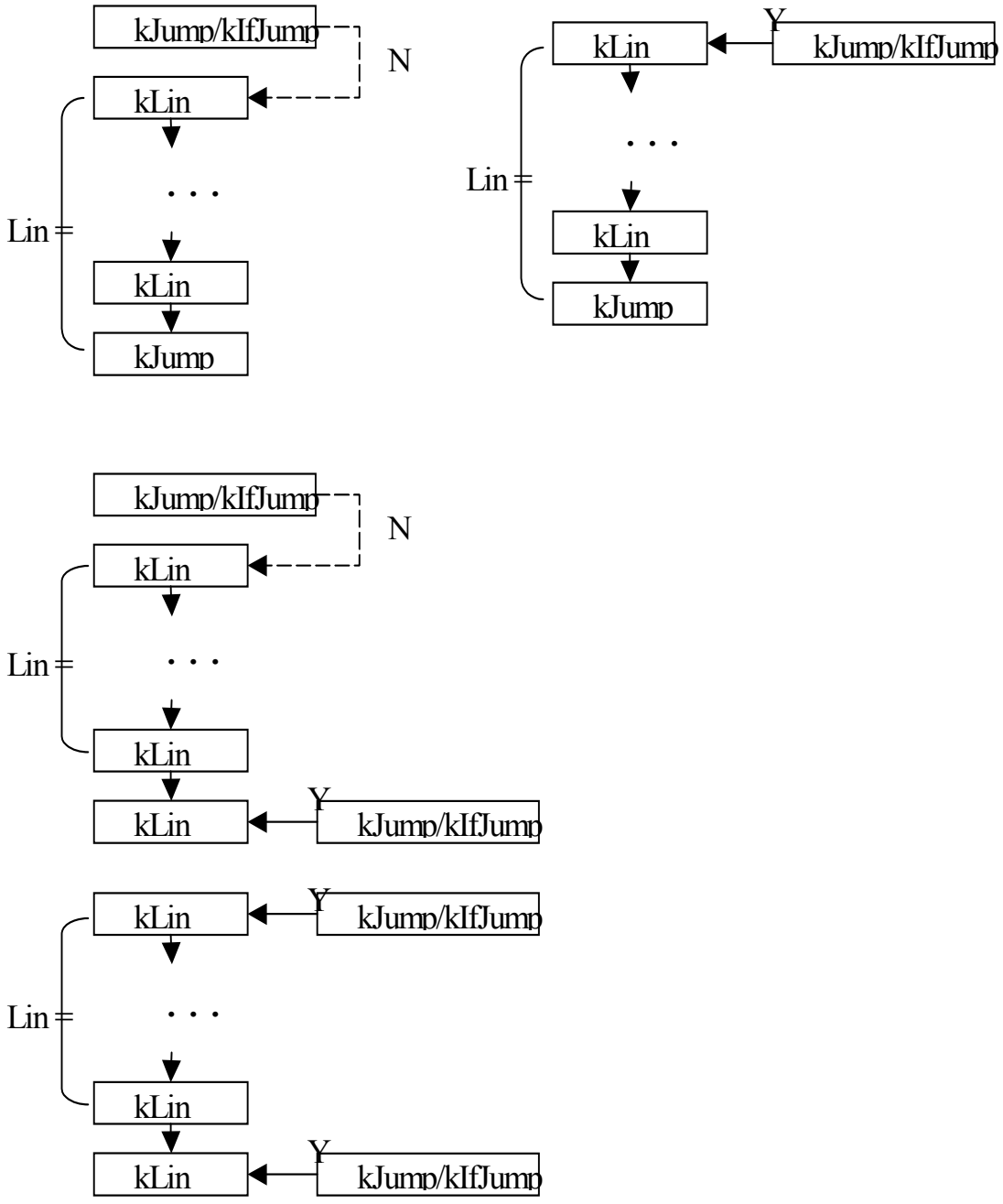


Рис. 1. Схемы формирования линейных вершин по коду программы

Условные вершины

Условная вершина в общем случае состоит из линейной части, образуемой из естественной последовательности команд, и команды условного перехода. В частном случае это может быть просто одна команда условного перехода. Таким образом, условную вершину следует формировать исходя из ее окончания.

Итак, любая условная вершина кончается на адресе команды условного перехода.

Началом условной вершины, точнее, ее линейной части, должен служить адрес:

1. команды, являющейся входом в программу;
2. команды, которая следует непосредственно за командами условной (*kIfJump*) или безусловной передачи управления (*kJump*);

3. команды, на которую есть передача управления, т.е. на ней сходятся ветви из других частей программы;
4. команды, которой предшествует запись локальной или глобальной переменной.

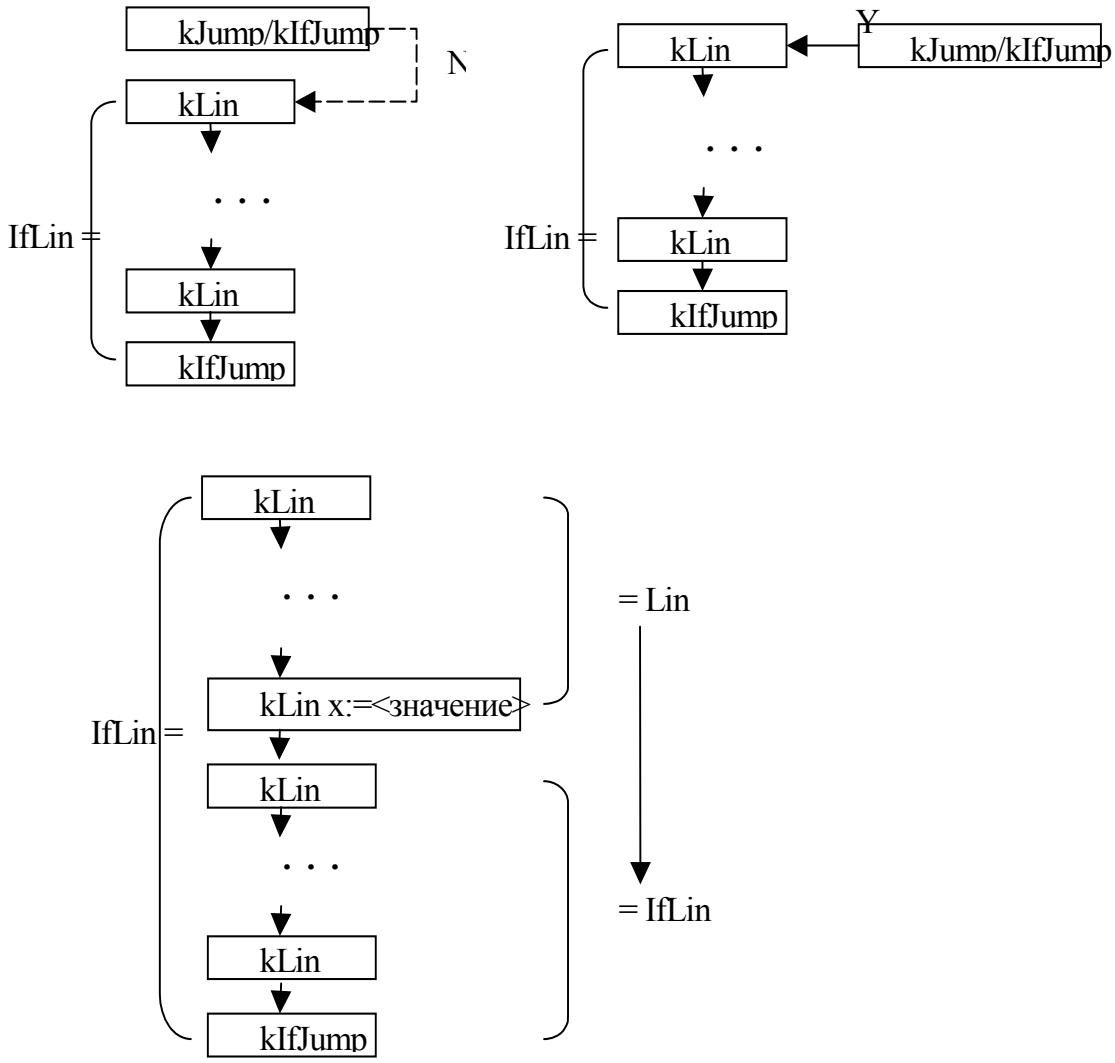


Рис. 2. Схемы формирования условных вершин по коду программы

Четвертое условие является эвристическим и позволяет искусственно ограничивать “глубину” условия по количеству переменных и формул, включенных в условие, и не является принципиальным. В рассматриваемом правиле, значение переменной, записываемое в память, может иметь или не иметь отношение к условию. Если оно не используется при формировании условия, то оно является детерминированной точкой окончания предыдущей линейной вершины, и, следовательно, команда, следующей за ней, однозначно является началом условной вершины. Если же значение переменной, записываемое в память, используется в условии, то на данной команде искусственно обрывается линейная вершина по приведенным выше соображениям об ограничении на длину формулы условия. На рис.2 показаны все варианты формирования условных вершин по коду программы.

Построение условных и линейных вершин

Заметим, что условные вершины имеют детерминированное окончание. Их число равняется числу команд условной передачи управления. Начало условной вершины не

является строго детерминированным вследствие п.4 списка команд, с которых может начинаться условная вершина. Линейные вершины, напротив, имеют детерминированное начало и не имеют детерминированного окончания из-за п.4 списка команд-окончаний линейной вершины.

Таким образом, временно исключив из рассмотрения п.4 списка команд, с которых может начинаться условная вершина, и п.4 списка команд-окончаний линейной вершины, можем построить множество условных вершин {*IfJump*} и множество линейных вершин {*Lin*}, содержащих вершины с детерминированным началом и окончанием. Далее из множества {*If Lin* } следует выделить подмножество {*~If Lin* } вершин, начинающихся с команды записи первой локальной или глобальной переменной, и подмножество {*~Lin*} вершин, оканчивающихся на этой команде записи.

Машинное представление структурированного графа программы

Учитывая специфику графов программ, в том числе и структурированных, их машинно-ориентированное описание целесообразно представлять в виде так называемых схемных списков. Слабые связи между вершинами делают представление графа программы в виде схемного списка более компактным, чем матричные или табличные представления.

При формировании списка целесообразно совместить прямой и обратный списки, чтобы иметь возможность обходить граф в любом направлении. Кроме того, для вспомогательных целей имеет смысл создать список по дугам, его можно использовать, например, для отметок активизации путей обхода графа.

Практическая реализация

На основе вышеизложенного была разработана подсистема построения структурированного графа программы. На ее вход подается исполнительный код или код программы на языке ассемблера. Это может быть программа целиком или какая-либо ее часть. Результатом работы подсистемы является файл формата DOT (формат описания графов, разработанный компанией AT&T), содержащий описание структурированного графа исходной программы. Разработанная подсистема используется в учебном процессе, в курсе изучения языков программирования высокого уровня и языка ассемблер, а также является частью создаваемой на кафедре информатики и прикладной математики СПбГУ ИТМО автоматизированной системы верификации и тестирования программных продуктов.

Заключение

Структурирование графа программы позволяет:

- уменьшить его размерность по числу вершин и дуг;
- перейти на основе выделенных условных вершин к построению булева графа;
- построить комплексные кубические покрытия переменных, вычисляемых программой.

Дальнейшее структурирование графа вычислительного процесса программы предлагается проводить путем объединения условных и линейных вершин в циклы, процедуры, параллельные структуры и другие возможные блоки, что позволит в конечном итоге осуществить формирование граф-схем программ и автоматизировать верификацию, контроль и диагностику программного продукта.

Литература

1. Проектирование цифровых вычислительных машин / Под ред. С.А.Майорова. М.: Высш. шк., 1972. 344 с.
2. Немолочнов О.Ф. Методы технической диагностики. Методические указания к курсовой работе. Л.: ЛИТМО, 1976.
3. Немолочнов О.Ф., Зыков А.Г., Лаздин А.В., Поляков В.И. Верификация в исследовательских, учебных и промышленных системах / Научно-технический вестник СПб ГИТМО(ТУ). Выпуск 11. Актуальные проблемы анализа и синтеза сложных технических систем / Гл. ред. В.Н. Васильев. СПб.: СПб ГУ ИТМО , 2003. С 146–151

МОДЕЛЬ ПОТОКОВ ДЛЯ ОПИСАНИЯ ВЗАИМОДЕЙСТВИЯ КОРПОРАЦИЙ В ИНФОРМАЦИОННЫХ СЕТЯХ

А.А. Оприщенко, Л.Г. Осовецкий

Введение

Функционирование сложных ИТ и ТКС, на построение которых затрачиваются значительные ресурсы и средства, может быть оправдано только при условии получения от их использования соответствующей экономической отдачи. Адекватное выполнение функций сложных ИТ связано не только с вероятностью наличия ошибок при их создании, но и все в большей степени с влиянием угроз внешнего воздействия с целью нарушения штатного функционирования систем ввиду наличия других систем, конкурирующих с исходной. Ситуация схожа с популяцией живых организмов, использующих общие ресурсы для существования и развития.

Макромодель эволюции ИТ

Популяционная макромодель базируется на представлении современных ИТ как взаимосвязанной системы функционирующих и конкурирующих единиц. В качестве основного показателя эффективной жизнедеятельности членов такой популяции выступает их экономическая эффективность.

Динамичность сложной системы заключается в непрерывности процессов взаимодействия субъектов друг с другом и ресурсами системы. Некоторые ресурсы могут одновременно использоваться несколькими различными субъектами, в результате происходит борьба за ресурсы, следствие борьбы – естественный отбор. Постоянные процессы борьбы за ресурсы – еще один признак динамичностисложной системы.

Модель взаимодействия двух корпораций

В работе [2] описан класс моделей динамических развивающихся информационных систем. Рассмотрим модель такой системы, взаимодействующей с другой подобной системой и с внешней средой. Пусть рассматриваются только две функции системы первая (внутренняя), обеспечивающая его существование, и вторая (внешняя), являющаяся внешним результатом функционирования системы. В качестве такой системы может выступить корпорация. Окружение ее составляют другие корпорации и кооперации, обменивающиеся веществом (субъектами, элементами) и информацией.

Материальное, энергетическое и информационное обеспечение этих функций назовем продуктами соответственно первого (I) и второго (II) рода. РС с номером i ($РС_i$), i=1, 2, представим в виде двух подсистем A_i^k, k=1, 2. В системы A_i^k поступают продукты k-го рода из $РС_{3-i}$ и из внешней среды. Кроме этих продуктов, в системах A_i^k при помощи определенных частей ранее появившихся продуктов I рода создаются новые продукты k-го рода (дальше вместо слов «количество продуктов» употребляется «продукты»).

Пусть $f(t)$ — скорость поступления в момент времени t из внешней среды продуктов I и II рода (общего ресурса) в обе РС; $z_i(t)\, f(t)$ – доля $f(t)$, поступающая в момент времени t в $РС_i$; $0 \le z_i \le 1$, $z_1 + z_2 = 1$; $x_i^k(t)\, z_i(t)\, f(t)$ – доли $z_i(t)\, f(t)$, поступающие в системы A_i^k, $0 \le x_i^k \le 1$, $x_i^1 + x_i^2 = 1$, $[u_i^k(t)]'_t$ – скорости поступления в системы A_i^k в момент времени t продуктов k-го рода из $РС_{3-i}$; $v_{i1}^k(t)$ и $v_{i1}^k(t)$ – скорости создания новых продуктов k-го рода в момент времени t в $РС_i$, используемых в дальнейшем соответственно в $РС_i$ и $РС_{3-i}$; $m_i^k(t)$ — скорости появления (т.е. создания и поступления) в $РС_i$ в момент времени t новых продуктов k-го рода; величины f, m_i^k предполагаются одной размерно-

сти; $\underline{y}_i^k(\underline{t})m_i^1\,\Delta t$ – доли продуктов из $m_i^1\,\Delta t$, используемых в дальнейшем для производства продуктов k-го рода в системах A_i^k, $0 < y_i^k < 1$, $y_i^1 + y_i^2 = 1$, $p_i^k(t)$ – количества функционирующих (создающих) продуктов I рода в момент времени t в системах A_i^k; $P_i(t) = p_i^1(t) + p_i^2(t)$; $a_i^k(t)$ – максимальные моменты времени, ранее которых не появился в $РС_i$ ни один продукт из $p_i^k(t)$. $a_i^k(t) < t$; $b_i^k(t)$ – максимальные моменты времени, ранее которых не появился в $РС_i$ ни один продукт из $u_i^k(t)$, $b_i^k(t) < t$; $\lambda_i^k(t, \tau, \Delta\tau)\,\Delta\tau$ – относительные доли продуктов из $y_i^k(\tau)\,m_i^1(\tau)\,\Delta\tau$, участвующих в создании продуктов k-го рода в момент времени t в системах A_i^k; $\alpha_i^k(t, \tau, \Delta\tau)$ – количества вновь созданных продуктов k-го рода в единицу времени, начиная с момента t, приходящихся на одну единицу продукта из $\lambda_i^k(t, \tau, \Delta\tau)\,y_i^k(\tau)\,m_i^1(\tau)\,\Delta\tau$; $v_i^k(t, \tau, \Delta\tau)$ – относительные доли продуктов из $v^k_{3\text{-}i,2}(\tau)\,\Delta\tau$, поступивших в $РС_i$ в момент времени t. Так как v_{i1}^k и v_{i2}^k по определению неотрицательны, то можно положить $v_{i1}^k = \mu_i^k v_i^k$, $v_{i2}^k = (1 - \mu_i^k)\,v_i^k$, $v_i^k = v_{i1}^k + v_{i2}^k$, $0 \le \mu_i^k \le 1$. Пусть t_0 — начало моделирования, $t \ge t_0$. На заданном интервале времени $[c_{i0}, t_0]$, $c_{i0} = \min\limits_{k=1,2} \alpha_k^i$, $(b_i^k(t_0))$, функции $m_i^1(\tau) \equiv m_{i0}^1(\tau)$ и $y_i^1(\tau) \equiv y_{i0}^1(\tau)$ заданы, а также заданы $a_i^k(\tau) \equiv a_{i0}^k(\tau)$ на $[b_i^k(t_0), t_0]$. В дальнейшем для любой функции $\varphi(t, \tau, \Delta\tau)$ полагаем $\varphi(t, \tau) = \lim\limits_{\Delta\tau \to 0} \phi(t, \tau, \Delta\tau)$.

Теорема 1. Если функции $\alpha_i^k\,\lambda_i^k\,y_i^k\,m_i^1$, $\lambda_i^k\,y_i^k\,m_i^1$, $v_i^k(1 - \mu_i^k)\,v_i^k$, $i, k = 1, 2$, интегрируемы, то

$$m_i^k(t) = \mu_i^k \int \alpha_i^k(t, \tau)\,\lambda_i^k(t, \tau)\,y_i^k(\tau)\,m_i^1(\tau)d\tau + x_i^k(t)\,z_i^k(t)f(t) +$$
$$+ d/dt \int v_i^k(t, \tau)(1 - \mu_{3\text{-}i}^k(\tau))d\tau \int \alpha_{3\text{-}i}^k(\tau, u)\,\lambda_{3\text{-}i}^k(\tau, u)\,y_{3\text{-}i}^k(u)\,m_{3\text{-}i}^1(u)du,$$
$$P_i(t) = \sum \int \lambda_i^k(t, \tau)\,y_i^k(\tau)\,m_i^1(\tau)d\tau, \qquad (1)$$
$$x_i^1 + x_i^2 = y_i^1 + y_i^2 = z_1 + z_2 = 1,\ 0 \le x_i^k, y_i^k, z_i, \lambda_i^k, \mu_i^k, v_i^k \le 1,$$
$$a_i^k(t) < t,\ b_i^k(t) < t,\ t \ge t_0 > a_i^k(t_0) > c_{i0},\ i, k = 1,2.$$

Обозначим через $C^{(0)}$, C, $C^{(1)}$ пространства соответственно кусочно-непрерывных, непрерывных и непрерывно дифференцируемых функций.

Теорема 2. Пусть $R_i(a_i^1(t), a_i^2(t)) = r_i(t - a_i^1(t)) + a_i^2(t) - t = 0$ и на своих областях определения $x_i^k\,z_i f, y_i^k, \lambda_i^k \in C^{(0)}$; $\alpha_i^k, \mu_i^k \in C$; $P_i, r_i, b_i^k, v_i^k \in C^{(1)}$; $i, k = 1, 2$. Если:

1) $m_{i0}^1(\tau) > 0$, $\tau \in [c_{i0}, t_0)$, $-\infty < c_{i0} < t_0$;
2) функции $\lambda_i^1\,P_i$ положительны, а $\alpha_i^k, \lambda_i^k, \mu_i^k, v_i^k, r_i, f$ – неотрицательны;
3) $0 < y_i^1 \le 1$;
4) $da_i^1(t)/dt \ge 0$;
5) $dr_i(a)/da \ge 0$, $0 < r_i(a) \le c_i a$, $c_i = \text{const} > 0$;
6) $b_i^k(t) \ge b_i^k(t_0)$, $t - b_i^1(t) \ge \text{const} > 0$, $t \ge t_0$,

то задача отыскания функций m_i^k, a_i^k при заданных функциях $\alpha_i^k, \lambda_i^k, \mu_i^k, v_i^k, x_i^k z_i f, y_i^k, b_i^k$, $P_i, r_i, I, k = 1, 2$ из соотношений (1) имеет единственное решение на любом конечном отрезке $[t_0, T]$, причем $m_i^k \in C^{(0)}$, $a_i^k \in C^{(1)}$, $i, k = 1, 2$.

Замечание 1. Функции в (1) должны удовлетворять определенным ограничениям. Например, при $a_i^k \equiv a_i$, $\mu_i^k = \lambda_i^k \equiv 1$, $k = 1, 2$, необходимым условием выполнения условия 4 теоремы 2 является

$$\max \alpha_i^1(t, \tau)\,y_i^1(\tau) \ge (P'_i(t) - x_i^1(t)\,z_i(t)\,f(t))/P_i(t),$$

а достаточным –

$$\min \alpha_i^1(t, \tau)\,y_i^1(\tau) \ge (P'_i(t) - x_i^1(t)\,z_i(t)\,f(t))/P_i(t),\ i = 1, 2.$$

Замечание 2. В соотношениях (1) a_i^k обычно зависят от всех элементов модели (а также могут зависеть и от некоторых параметров внешней среды, не отмеченных ранее). Теорему 2 нетрудно обобщить на этот случай (при этом подынтегральные выражения в (1) для $k = 1$ должны удовлетворять условию Липшица по искомым функциям m_i^1 равномерно по t).

Поставим следующие задачи взаимодействия РС. С учетом связей (1) и заданных α_i^k, $z_i f$, λ_i^k, T, v_i^k, b_i^k, P_i, r_i, i, $k = 1, 2$, определить:

1) $\Phi_1 = \max \sum \int m_i^2(t)\, dt$,
2) $\Phi_2 = \max \min \int (m_1^2(t) - m_2^2(t))\, dt$, $\mu_i^k \equiv 1$, i, $k = 1, 2$.

Теорема 3. Пусть выполнены условия теоремы 2 и $a_i^k \equiv a_i$, $\mu_i^k = \lambda_i^k \equiv 1$, $k = 1, 2$.

А) Если $T - t_0$ достаточно мало, то:

1) в случае невозрастания $\alpha_i(t, \tau)$ с ростом t и неубывания $\alpha_i(t, \tau)$ с ростом τ, i, k= 1,2 Φ_1 и Φ_2 достигаются при $x_i^1 \equiv 0$, $y_i^1 = y^1{}_{\text{imin}}$, $y^1{}_{\text{imin}}(t)$ – минимально допустимая в силу ограничений типа неравенства $y_i^1(t)$, $I = 1, 2$;

2) в случае непрерывности справа $x_i^2 z_i f$, i= 1, 2, в момент t_0, необходимым (достаточным) условием выполнения $\Phi_2 > 0$ является: в момент t_0 сумма скорости поступления внешнего ресурса в подсистему A_1^2 и произведения средней производительности α_1^2 на число функционирующих продуктов I рода в подсистеме A_1^2 не меньше (больше), чем соответствующая сумма для второй РС.

Б) Пусть $T - t_0$ достаточно велико, причем:

1) порядки экспоненциального возрастания α_i^k по τ больше порядков их экспоненциального убывания по t;

2) порядок экспоненциального роста $P'_i(t)$ меньше, чем порядок роста α_i^1 по τ, но не меньше, чем порядок убывания α_i^1 по t ;

3) $(P'_i - x_i^1 z_i f)'_i$ неотрицательна и порядка $P'_i(t)$;

4) относительная скорость снижения производительности α_i^1 продуктов I рода в момент t по отношению к производительности α_i^1 момента τ, $\tau \le t$, имеет экспоненциальный порядок по $\tau - t$ не меньше, чем порядок убывания α_i^1 по t;

5) $\alpha_i^1(t, \tau)\, y^1{}_{\text{imin}}(\tau) \ge \alpha_i^1(t, a_i(t_0))\ y_{i0}^1(a_i(t_0))$, $\tau \in [t_0, T]$, (см. [1, с. 274–276]), i, $k = 1, 2$. Тогда Φ_1, и Φ_2 достигаются при $x_i^1 \equiv 1$, y_i^1 отличной от минимально допустимой $y^1{}_{\text{imin}}$ на большей части отрезка $[t_0, T]$ и при $x_i^1 \equiv 0$, $y_i^1 = y^1{}_{\text{imin}}$ в конце отрезка $[t_0. T]$, $i = 1, 2$.

Заключение

При помощи этой модели можно определять степень связи любых двух корпораций (двух субъектов в корпорации), ценность информации, определять прогноз развития корпорации на адаптивном участке, а при использовании дополнительных краевых условий устойчивости – прогнозировать достижение участка неопределенности (бифуркации) в развитии корпорации. Таким образом, можно прогнозировать критические изменения для корпорации, которые могут быть вызваны внешними разрушающими воздействиями.

Литература

1. Осовецкий Л.Г., Немолочнов О.Ф., Твердый Л.В., Беляков Д.А. Основы корпоративной теории информации. СПб: ИТМО, 2004. 84 с.
2. Глушков В.М., Иванов В.В., Яненко В.М. Моделирование развивающихся систем. М.: Наука, 1983. 351 с.
3. Михалевич В.С., Иванов В.В., Яненко В.М. Методологические вопросы моделирования развивающихся систем. Киев : Об-во «Знание» УССР, 1984. 16 с.
4. Бердышев Г.Д. Строение, функции и эволюция генов. Киев, 1980.
5. Молекулярные основы геносистематики. / Под ред. Антонова А.С. М.: Изд-во Моск. ун-та, 1980.
6. Рефф, Рудольф и др. Эмбрионы, гены и эволюция. Москва, 1986.
7. Геном человека. Ч. 1, 2. Москва, 1994.

МОДЕЛИРОВАНИЕ СИСТЕМЫ ОПТИЧЕСКОГО ПОИСКА

А.В. Кучер, А.В. Демин, В.С. Кулагин

В данной статье представлен алгоритм имитационного моделирования поисковой системы, работающей в оптическом диапазоне. Рассмотренный алгоритм может быть применен как при выборе параметров поисковой системы на этапе проектирования, так и для моделирования реальной системы.

Введение

В настоящее время существует ряд работ, посвященных проблемам поиска, но проблема построения модели системы поиска и оптимизации вероятностно-временных характеристик системы на этапе проектирования системы с использованием имитационного моделирования в целом так и не раскрыта. В работах [1, 2] рассматривается модель системы поиска и наведения как единого целого, что ограничивает применение данной модели для использования принятия проектного решения в других классах систем, использующих систему поиска как подсистему.

Таким образом, целью данной работы является построение имитационной модели, позволяющей осуществить поиск схемотехнического решения поисковой системы. В данной работе предложенный подход построения имитационной модели будет рассматриваться на примере системы оптического поиска.

Постановка задачи

Задача поиска объекта в заданной области Ω n-мерного евклидового пространства возникает тогда, когда необходимо обнаружить объект и определить его местоположение в этой области с помощью системы поиска.

Параметрами системы поиска, влияющими на качество и время обнаружения, являются:

- область поиска (Ω) – область, в которой производится поиск объекта;
- поле анализа (для системы оптического поиска величина поля зрения системы 2ω) – часть области, воспринимаемая системой ($2\omega \le \Omega$);
- разрешающая способность системы (α) –минимально регистрируемая разница между двумя соседними информационными признаками;
- поисковые усилия – время просмотра, затрачиваемое на просмотр области поиска.

Под поиском будем понимать процесс обработки информации наблюдателем, разворачивающийся во времени и направленный на решение задачи обнаружения объекта относительно выбранной системы отсчета при наличии помех и за конечное время.

Наблюдатель получает информацию о находящихся в области поиска объектах путем просмотра области поиска. Просмотр может осуществляться [4]:

– последовательно (путем сканирования);
– параллельно – по нескольким каналам одновременно.

Поиск осуществляется в соответствии с некоторым планом, представляющим собой правило выбора последовательности действий – алгоритм поиска. Выделяют следующие алгоритмы поиска [4]:

- равномерный – распределение поисковых усилий по просматриваемому пространству происходит равномерно. При этом тщательность и скорость просмотра любого участка области поиска одинакова и определяется уровнем расходуемых поисковых усилий;
- априорный – наличие предварительных сведений о возможном местоположении искомого объекта указывает на то, что тщательность просмотра отдельных участков

области при поиске должна соизмеряться с этими сведениями, т.е распределение поисковых усилий должны быть неравномерным;

- апостериорный – все пространство поиска просматривается равномерно, причем уровень усилий, приходящих на поле анализа, недостаточен для обнаружения объекта с заданной достоверностью, но позволяет сделать некоторое заключение о вероятности наличия или отсутствия объекта в данной ячейке. По результатам равномерного просмотра отбираются ячейки, сигнал в которых превышает выбранный порог. На второй стадии просматриваются только отобранные ячейки и среди них выявляются те, где сигнал превысил некоторый другой порог, и т.д.

Если рассматривать поиск как процесс получения и обработки информации, то общая схема этого процесса может быть представлена в виде рис.1.

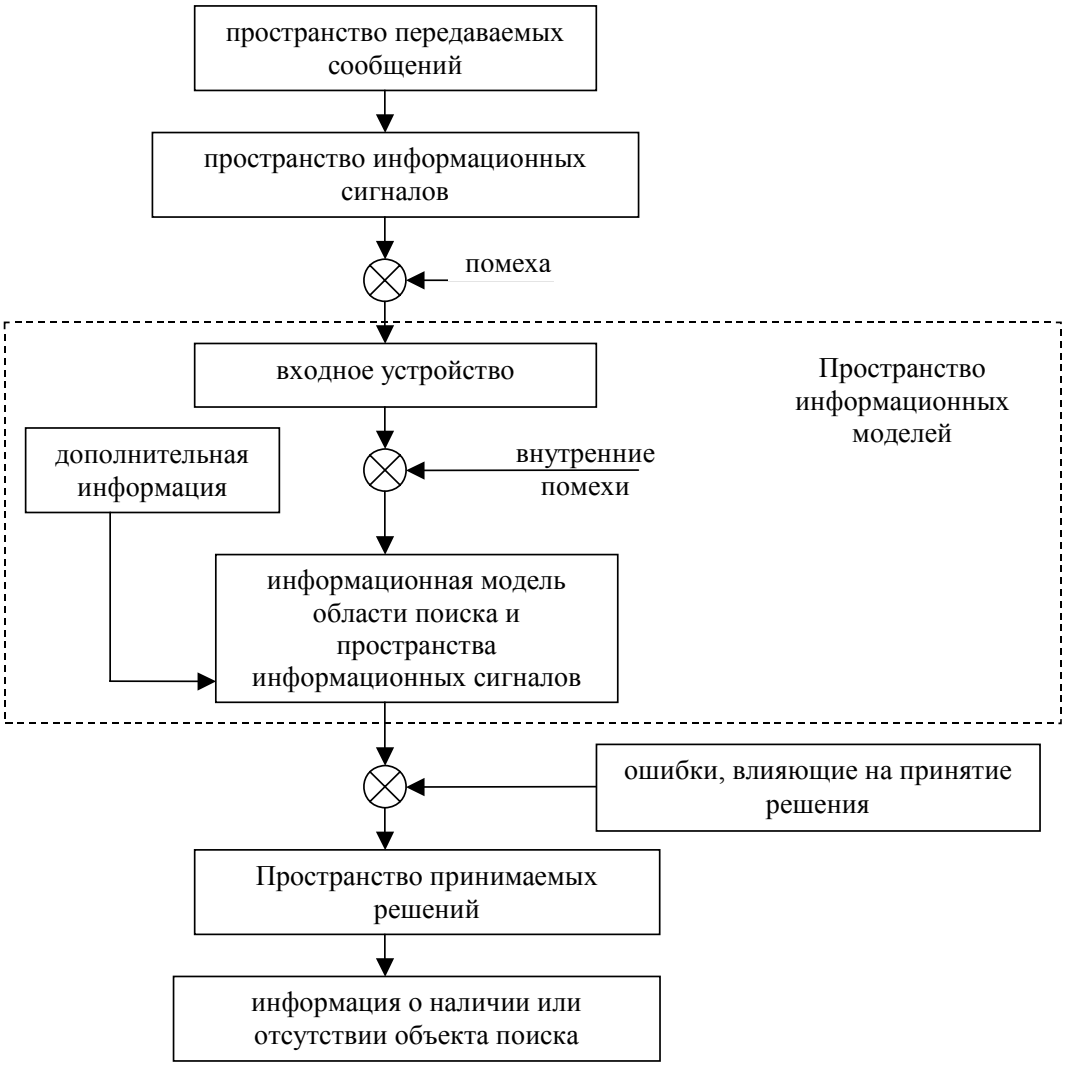


Рис. 1. Общая схема получения и обработки информации при поиске

Вероятностные характеристики системы оптического поиска

С точки зрения событийности система поиска должна принимать следующие решения:

- объект обнаружен в поле зрения;
- объект не обнаружен в поле зрения.

Но, поскольку поиск – процесс вероятностный, а также в случае, когда решение об обнаружении или не обнаружении принимается на пределе «чувствительности» правила принятия решения, то возможны еще два варианта событий:

- объект не обнаружен к заданному сроку, но он есть в поле зрения – пропуск цели;
- объект не находится в поле зрения, но проходит информация о его обнаружении – ложная тревога;

Таким образом, для системы оптического поиска характерны следующие три вероятностные характеристики:

- вероятность правильного обнаружения объекта;
- вероятность ложного обнаружения;
- вероятность пропуска объекта.

Для моделирования системы оптического поиска будем использовать имитационное моделирование.

Сущность имитационного моделирования

Имитационное моделирование, натурное или математическое – это вид моделирования, для которого характерно имитирование некоторых возможных условий функционирования подсистем моделируемой системы. Совокупность результатов имитации подсистем позволяет с той или иной степенью достоверности представить поведение системы в целом.

При построении системы исследователя интересует прежде всего вычисление функционала, характеризующего поведение объекта имитации. Наиболее важный функционал – это показатель эффективности системы, с помощью которого исследователь может оценить различные принципы управления системой, сравнение вариантов схемотехнических решений, определение степени влияния изменений параметров и начальных условий имитации ее поведениях [5].

Алгоритм моделирования системы оптического поиска

Определим исходные данные для моделирования.

1) Величину поля зрения системы определим в следующем виде:

$\|2\omega\| = n$.

2) Введем операцию разбиения поля зрения на ячейки. Количество ячеек определяется по формуле

$$K = \left]\frac{2\omega}{\alpha}\right[+ 1 .$$

где α – разрешающая способность системы, а] [– целая часть от числа.

3) Обозначим размеры объекта SizeX, SizeY (в ячейках). Размеры объекта определяются в диапазоне 3×3 – K × K. Минимальный размер объекта равен трем, так как по критерию Джонсона об опознании объекта необходимо минимум три точки для опознания.

4) Область поиска представляет собой пространственно-временное множество.

4.1) Пространственное множество определим массивом чисел. Для упрощения модели будем рассматривать поиск объекта на плоскости. Если X, Y – координаты области поиска, то

$$\Omega = \left\{\left]\frac{X}{2\omega}\right[+ 1\right\} \times \left\{\left]\frac{Y}{2\omega}\right[+ 1\right\} = \mathrm{N_x} \times \mathrm{N_y},$$

где

$N_x = \left]\frac{X}{2\omega}\right[+ 1$, $N_y = \left]\frac{Y}{2\omega}\right[+ 1$.

Разобьем область поиска на поля зрения и проиндексируем поля зрения так, как показано на рис. 2. Индексы поля зрения определяют местоположение поля зрения в области, где n_{ij} – поле зрение, $i = 1..N_y$, $j = 1..N_x$.

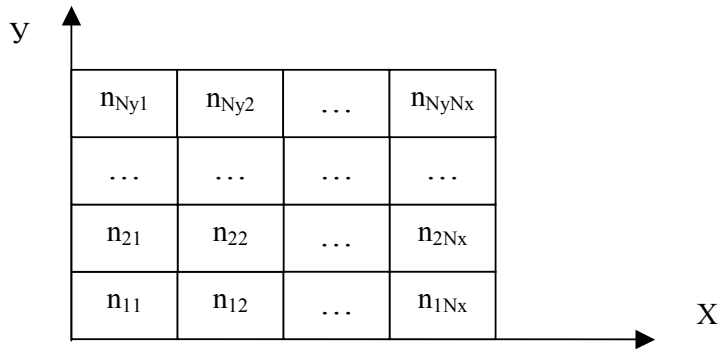


Рис. 2. Разбиение области поиска на поля зрения

Каждое поле зрения разобьем на ячейки и присвоим индексы ячейкам так, как показано на рис. 3.

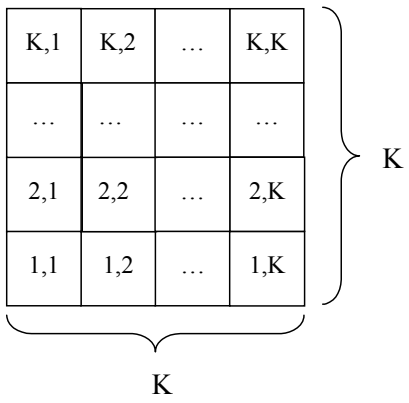


Рис. 3. Разбиение поля зрения на ячейки

Таким образом, пространственное множество будет представлено массивом чисел. Количество элементов массива определяется по формуле $N_x \times N_y \times K^2$. Область поиска будет представлена в виде двумерного массива

$$A = \begin{vmatrix} a_{11} & a_{12} & \ldots & a_{1Nx} \\ a_{21} & a_{22} & \ldots & a_{2Nx} \\ \ldots & \ldots & \ldots & \ldots \\ a_{Ny1} & a_{Ny2} & \ldots & a_{NyNx} \end{vmatrix},$$

где a_{ij} – элемент массива, представляющий собой разбиение поля зрения на ячейки в виде двумерного массива чисел B размером $K \times K$:

$$a_{ij} = B = \begin{vmatrix} b_{11} & b_{12} & \ldots & b_{1K} \\ b_{21} & b_{22} & \ldots & b_{2K} \\ \ldots & \ldots & \ldots & \ldots \\ b_{K1} & b_{K2} & \ldots & b_{KK} \end{vmatrix}.$$

4.2) Временное множество определим следующим образом: время анализа в ячейке определим в виде разбиения элемента массива на числа, меньшие по значению этого элемента массива и сумма которых равна значению этого элемента. Каждый элемент разбиения представляет собой одну условную единицу затраченного времени анализа.

$$(b_{ml})_{ij} = \sum_{s=1}^{q} (\frac{b_{ml}}{q}).$$

Влияние затраченных поисковых усилий на «качество» обнаружения цели можно задать виде округления элементов разбиения. Чем меньше количество разбиений, тем «грубее» округления.

Модель системы оптического поиска состоит из 7 подпрограмм. Структурная схема модели оптического поиска представлена на рис. 4.

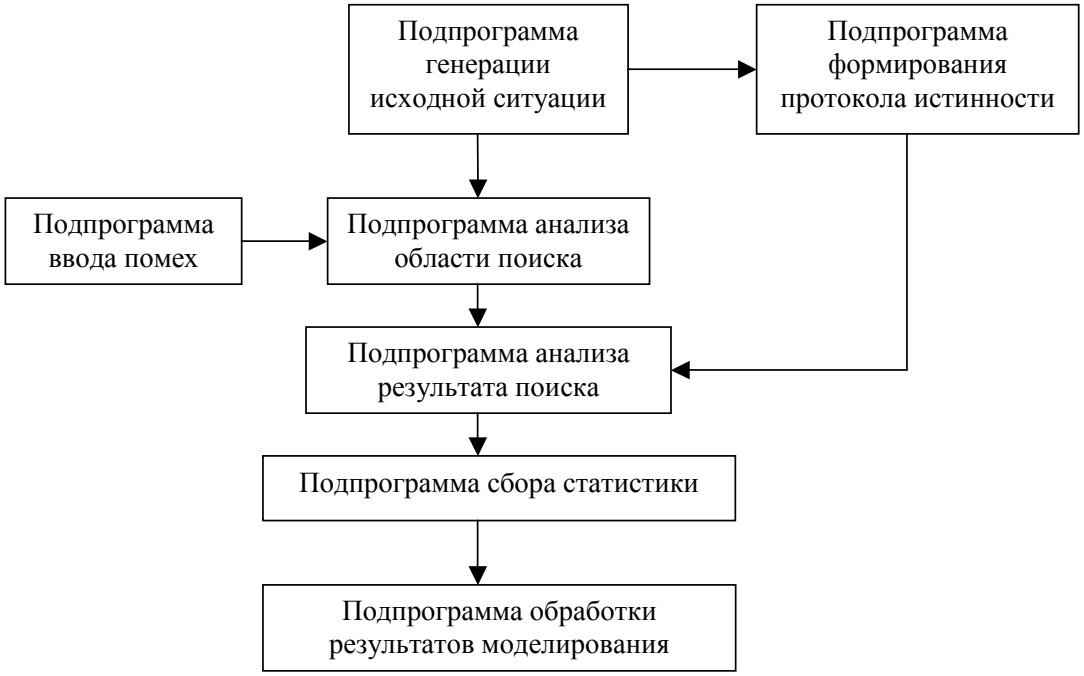


Рис. 4. Структурная схема модели системы оптического поиска

Подпрограмма генерации исходной ситуации выполняет следующие этапы:

a) Инициализируются элементы массива значением, равным M, где M – целое положительное число. Число M «задает» уровень фона в каждой ячейки области поиска, постоянное в пространстве за отведенное время

$$(b_{ml})_{ij} = M,\ i = 1..N_y,\ \ j = 1..N_x,\ \ m,l = 1..K.$$

b) Задается стратегия поиска. Для этого необходимо для всех ячеек области поиска задать количество элементов разбиения элемента массива – $(q_{ml})_{ij}$, $i = 1..N_y,\ \ j = 1..N_x,\ \ m,l = 1..K$. Значение q зависит от выбранной стратегии поиска и соответствует следующим условиям:

- q = const (равномерный поиск) для всех ячеек области поиска;
- $q \neq$ const (априорный поиск);
- 1-й шаг – q = const, 2-й шаг – $q \neq$ const (апостериорный поиск);

На значение q также влияет заданное пороговое время анализа области поиска $t_{\text{п}}$.

c) С помощью генератора случайных чисел генерируется одно из двух событий:

- объект находится в области поиска;
- объект не находится в области поиска;

Эта процедура осуществляется следующим образом: с помощью генератора случайных чисел генерируется два числа – NumberX в диапазоне (1 – X) – Nx и NumberY в диапазоне (1 – X_y) – Ny, где X – целое число, $X_x \in 1 -]0.1\times Nx[$, а $X_y \in 1 -]0.1\times Ny[$. Эти числа (NumberX и NumberY) являются индексами элемента массива A и определяют местоположение объекта поиска в области. Событию «объект находится в области поиска» соответствует выполнение обоих условий:

$1 \le NumberX \le Nx$;

$1 \le NumberY \le Ny$.

В противном случае считается, что сгенерировано событие «объекта поиска в области поиска нет» и нижеследующие этапы подпрограммы генерации исходной ситуации не выполняются.

d) Определим случайным образом два числа – задание местоположения объекта (индексы элемента массива B) в поле зрения с индексами NumberX и NumberY– iBegin, jBegin. Для осуществления нахождения объекта поиска в одном поле зрения $iBegin \in 1 \div K\text{-}SizeX$, а $jBegin \in 1 \div K\text{-}SizeY$.

e) Определим случайным образом индексы ячеек в поле зрении, в которых будет изменено числовое значение элемента массива на значение 2M. Индексы ячеек должны быть в области расположения объекта, исходя из его размеров. Так i= iBegin,…, iBegin +SizeX. Аналогично для j= jBegin, …, jBegin + SizeY. Количество операций изменения ячеек равно SizeX×SizeY. Если значение в ячейке с индексами *i*, *j* (*i*, *j* сгенерированы на данной итерации) уже изменено (не равно значению фона M), то в ячейке изменения не производятся. Измененные элементы и представляют собой объект поиска.

Подпрограмма формирования протокола истинности регистрирует в протоколе истинности сведения о наличии или отсутствии объекта поиска в заданной области поиска, сгенерированные подпрограммой генерации исходной ситуации.

Подпрограмма ввода помех случайным образом генерирует в области поиска ложный объект. Подпрограмма запускается в работу случайным образом, на основе случайно сгенерированного числа, и либо производит какие-нибудь изменения, либо нет. В результате внесенных помех в область поиска появляются следующие события:

- ложная тревога;
- пропуск цели.

Подпрограмма анализа области поиска выполняет сканирование поля зрения на наличие-отсутствие объекта поиска в заданной области поиска с учетом выбранной стратегии. Подпрограмма запускает счетчик фактически затраченного времени анализа области поиска. В каждом поле зрении вычисляется сумма элементов массива (соответствующий полю зрения массив B), необходимая для анализа поля зрения. После вычисления суммы элементов поля зрения модифицируется счетчик времени, потраченного на анализ. Для принятия решения о наличии-отсутствии объекта поиска в поле зрения устанавливается пороговое значение суммы элементов поля зрения. Пороговое значение зависит от разрешающей способности системы и размеров объекта и обозначается $\sum_{tv}$. По проведенным исследованиям алгоритма вбрасывания чисел в область нахождения объекта (см.п. «e» подпрограммы генерации исходной ситуации) было получено, что математическое ожидание количества измененных элементов равно 0.75× SizeX×SizeY. Тогда $\sum_{tv}$ будет вычисляться по формуле:

$$\sum_{tv} = K^2 \times M + 0.75 \times SizeX \times SizeY \times M_2,$$

где M_2 – значение элемента массива, соответствующее объекту в ячейке без фона. Как отмечалось ранее, элемент массива принимает значение, равное 2M, если в ячейке находится объект, тогда

$M_2 = 2M - M =>$

$\sum_{tv} = K^2 \times M + 0.75 \times SizeX \times SizeY \times M = M (K^2 + 0.75 \times SizeX \times SizeY) =$

$=M ((\left[\frac{2\omega}{\alpha}\right[+ 1)^2 + 0.75 \times SizeX \times SizeY).$

Таким образом, если сумма элементов поля зрения больше порогового значения суммы $\sum_{tv}$, то система поиска дает ответ о наличии объекта поиска в поле зрения, в противном случае – о его отсутствии:

$\sum_{m=1}^{K} \sum_{l=1}^{K} (b_{ml})_{ij} \quad i = 1..N_y, \quad j = 1..N_x >= \sum_{tv}$ – объект есть;

$\sum_{m=1}^{K} \sum_{l=1}^{K} (b_{ml})_{ij} \quad i = 1..N_y, \quad j = 1..N_x < \sum_{tv}$ – объекта нет.

Ответом системы является индексы поля зрения (обозначим их NumX и NumY соответственно) с обнаруженным объектом поиска (в противном случае NumX и NumY меньше 1) и фактически затраченное время анализа области поиска - $t_ф$.

Подпрограмма анализа результата поиска на основании критерия оценки ситуации определяет событие, анализируя результаты работы подпрограммы анализа области поиска и протокола истинности. Определим содержательно критерий оценки ситуации.

A. Объекта нет – индексы поля зрения с объектом, сгенерированные подпрограммой генерации исходной ситуации, и полученные подпрограммой анализа области поиска, меньше единицы и фактическое время анализа области поиска меньше заданного порогового времени анализа.
B. Объект есть – индексы поля зрения с объектом, сгенерированные подпрограммой генерации исходной ситуации, и полученные подпрограммой анализа области поиска, лежат в диапазоне области поиска и совпадают и фактическое время анализа области поиска меньше заданное порогового времени анализа.
C. Ложная тревога – индексы поля зрения с объектом, сгенерированные подпрограммой генерации исходной ситуации, меньше единицы, а полученные подпрограммой анализа области поиска, лежат в диапазоне области поиска, и фактическое время анализа области поиска меньше заданного порогового времени анализа.
D. Объект пропущен – индексы поля зрения с объектом, сгенерированные подпрограммой генерации исходной ситуации, лежат в диапазоне области поиска, а индексы поля зрения, полученные подпрограммой анализа области поиска, меньше единицы, и фактическое время анализа области поиска меньше заданное порогового времени анализа. Второй вариант – индексы исходного и полученного местоположений совпадают и лежат в диапазоне области поиска, но фактическое время анализа области поиска больше заданного порогового времени анализа.

Тогда критерий оценки можно записать так:

A. NumX < 1, NumY < 1, NumberX < 1, NumberY < 1;
B. NumberX = NumX, NumberY = NumY, $t_ф <= t_п$;

при условии NumX, NumberX $\in$ 1÷ Nx; NumY, NumberY $\in$ 1÷ Ny;

C. NumberX <1, NumX $\in$ 1÷ Nx, NumberY <1 NumY $\in$ 1÷ Ny, $t_ф <= t_п$;
D. NumberX $\in$ 1÷ Nx, NumberY $\in$ 1÷ Ny, NumX < 1, NumY < 1, $t_ф <= t_п$

или

NumberX = NumX, NumberY = NumY, $t_ф > t_п$

при условии

NumX, NumberX $\in 1 \div$ Nx; NumY, NumberY $\in 1 \div$ Ny;

или в табличной форме (табл. 1).

	Событие	Исходные индексы лежат в области	Полученные индексы лежат в области	Проверка совпадения индексов	Время анализа меньше заданного
a.	Объекта нет	–	–	–	+
b.	Объект есть	+	+	+	+
c.	Ложная тревога	– (+)	+	–	+
d.	Пропуск цели	+ (+)	– (+)	– (+)	+ (–)

Таблица 1. Критерий оценки ситуации

Подпрограмма сбора статистики накапливает статистику событий, полученную в процессе моделирования. Подпрограмма обработки результатов моделирования вычисляет оценки по ситуациям и достоверности ответа для системы и выдает результаты пользователю. Результаты моделирования представлены в табл. 2.

Поле зрения	Разрешающая способность	Поисковые усилия на яч. (в условных единицах)	Вероятность достоверного ответа	Вероятность ложной тревоги	Вероятность пропуска цели
2	0.1	2	0.87	0.07	0.06
2	0.1	5	0.9	0.08	0.05
2	0.1	8	0.86	0.11	0.03
4	0.1	2	0.92	0.03	0.05
4	0.1	5	0.9	0.06	0.07
4	0.1	8	0.89	0.05	0.06

Таблица 2. Результаты моделирования

Заключение

Разработанная модель системы оптического поиска позволяет получить вероятностные оценки по ситуациями и достоверного ответа. В настоящее время готовится натурный эксперимент для проверки адекватности разработанной модели.

Литература

1. Демин А.В., Копорский Н.С., Немолочнов О.Ф. Вероятностная модель режимов работы систем поиска и наведения. // Известия вузов. Приборостроение. 1999. Т.42. № 5–6. С.14–18.
2. Демин А.В., Копорский Н.С., Немолочнов О.Ф., Чиченова Е.А. Имитационное моделирование технических объектов. // Известия вузов. Приборостроение. 2003. Т.46. № 2. С.73–79.
3. Здор С.Е., Широков В.Б. Оптический поиск и распознавание. М.: Наука, 1973. 240 с.
4. Кожухов А.К. Основы теории поиска объектов. М.: МАИ, 1999. 76 с.
5. Максимей И.В. Имитационное моделирование на ЭВМ. М.: Радио и связь, 1988. 232 с.

АНАЛИЗ СРЕДСТВ АВТОМАТИЗАЦИИ УПРАВЛЕНИЯ ОРГАНИЗАЦИЕЙ

А.Е. Лашкевич, Т.А. Павловская

В статье рассматриваются способы автоматизации управления современными предприятиями. Описывается понятие комплексной автоматизированной системы и ее назначение. Приводится краткий аналитический обзор и классификация наиболее востребованных на российском рынке комплексных автоматизированных систем.

Введение

В современных условиях эффективность деятельности предприятия во многом определяется эффективностью его системы управления. Необходимость оперативного реагирования на конъюнктуру рынка и быстро меняющуюся экономическую ситуацию требует перестройки внутренней микроэкономики предприятия, постановки управленческого учета, оптимизации процессов управления.

В процессе управления предприятиями необходимо эффективное решение комплекса задач, основными их которых являются управление финансами, управление производством, управление сбытом и снабжением, управление внутренними службами, управление кадрами. В зависимости от особенностей и масштаба предприятия каждый из выделенных видов может включать в себя значительное число отдельных задач. В общем случае на предприятии можно выделить как минимум три вида учета:

1) оперативный учет – обеспечивает сбор первичной информации и является основным поставщиком данных для остальных видов учета;

2) управленческий учет – направлен на получение информации для детального анализа деятельности предприятия, прогнозирования и принятия решений;

3) бухгалтерский учет – обеспечивает получение необходимой бухгалтерской отчетности.

В общем виде процесс управления во всех сферах деятельности можно представить в виде так называемой «петли управления», включающей циклическую последовательность следующих этапов: прогноз – планирование – контролируемая деятельность по реализации планов – учет и анализ результатов – коррекция прогнозов и планов (рис. 1) [1].

Рис. 1. Обобщенная схема управления

Процесс управления предприятием реализуется в рамках системы управления предприятием. Автоматизируя локальные задачи управления (островная автоматизация), зачастую можно столкнуться с проблемой несвязности информационных потоков, поступающих от управляющих подразделений. Анализ таких материалов требует существенных затрат времени и далеко не всегда дает руководителю желаемые результаты для интегрированной оценки и оперативного принятия тактических и стратегических управленческих решений. Попытка же объединить имеющиеся системы, т.е. «построить мосты между островами», как правило, не удается вследствие того, что системы приобретали независимо друг от друга. Следовательно, комплексная автоматизация управления предприятием реализуется в рамках комплексной системы автоматизации.

В настоящее время стоимость программных продуктов и их внедрения довольно высока. Если же рассмотреть все остальные затраты, связанные с процессом автомати-

зации, то суммы получатся еще больше. Исходя из этого, руководители должны быть уверены, что деньги на автоматизацию будут потрачены не зря [3].

Понятие комплексной автоматизированной системы

Комплексная система автоматизации (КСА)[1] - это система управления финансово-хозяйственной деятельностью предприятия, обеспечивающая принятие обоснованных управленческих решений на основе качественной и достоверной информации, получаемой с помощью современных управленческих и информационных технологий. Она обеспечивает ведение оперативного, бухгалтерского и управленческого учета и строится на основе единого информационного пространства, охватывая и координируя всю совокупность управленческих процессов предприятия [1].

Целью данной системы является обеспечить:

1) высшее руководство – информацией для стратегического планирования, финансово-экономического прогнозирования и анализа хозяйственной деятельности;

2) руководство среднего уровня – информацией для оперативного планирования и координации подконтрольных ему функций;

3) рядовых сотрудников – эффективными инструментами для выполнения должностных функций, регистрации фактов хозяйственной деятельности и принятия оперативных решений.

КСА предприятия является сегодня одной из важнейших составляющих успешного развития бизнеса. Она представляет собой информационную систему, в которой оперативно накапливаются и обрабатываются данные о текущей финансово-хозяйственной деятельности предприятия. Она помогает устранить многие недостатки в управлении, например: разобщенность управленческих и информационных технологий, несоответствие систем планирования и контроля, неэффективность управления затратами, неэффективность использования финансовых ресурсов.

Подходы к созданию ко мплексной автоматизированной системы

Существует несколько типовых подходов к созданию КСА, схема их классификации приведена на рис.2. Рассмотрим подробнее особенности применения каждого из них.

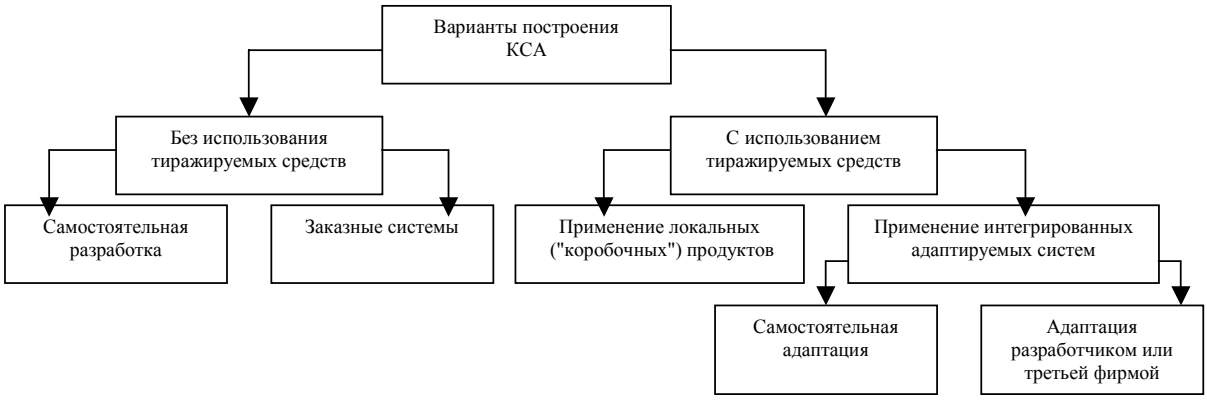


Рис. 2. Подходы к построению КСА

Самостоятельная разработка предполагает разработку КСА собственными силами, без привлечения сторонних организаций и приобретения тиражируемого прикладного программного обеспечения [1]. Потенциально эффективным этот подход может быть для крупных предприятий, имеющих большой коллектив разработчиков, уже обладающих опытом разработки и внедрения комплексных систем автоматизации. В

[1] В литературе и проспектах эти системы часто называются – корпоративная информационная система (КИС).

большинстве же случаев для предприятий такой подход будет наиболее дорогим, длительным по срокам реализации и рискованным в смысле достижения поставленных целей. Это объясняется тем, что разрабатывающие систему сотрудники будут оторваны от своих прямых обязанностей по эксплуатации уже функционирующих программ, проект может сорваться из-за ухода одного-двух специалистов или нехватки сил для построения действительно мощной системы.

Заказные системы предполагают разработку системы, полностью соответствующей особенностям конкретного предприятия, что является их основным преимуществом. В потенциале этот подход характеризуется сравнительно меньшей стоимостью и меньшими сроками реализации, чем самостоятельная разработка [1]. Сегодня «заказная» разработка фактически сводится к неявному использованию тиражируемых систем, которые имеются в распоряжении исполнителя. Следовательно, имеет смысл познакомиться с явным применением тиражируемых средств, поскольку эти варианты могут быть дешевле при той же функциональности и надежнее в связи с применением широко апробированных решений.

При использовании тиражируемых (коробочных) продуктов вы приобретаете программы автоматизации различного хозяйственного учета. Основным преимуществом данного подхода является сравнительно низкая стоимость программ, простота, небольшие сроки их освоения, хороший сервис по сопровождению. Такие системы хорошо апробированы, однако с помощью них практически невозможно построить КСА. Это объясняется недостаточной функциональностью таких продуктов. Использование коробочных продуктов целесообразно на малых и средних предприятиях на начальных стадиях автоматизации.

Адаптированные интегрированные системы основываются на тщательно проработанном и предназначенном для тиражирования программном ядре [1]. Это ядро изначально функционально ориентировано на возможность обеспечения комплексной автоматизации управленческого и других видов учета, данные которых необходимы для КСА. Таким образом, наличие этого ядра, с одной стороны, в потенциале обеспечивает интегрированным системам такие преимущества тиражируемых систем, как использование апробированных решений, а с другой – устраняет недостаточный уровень функциональности и проблемы совместимости «коробочных» продуктов. Эти системы содержат гибкие средства настройки характеристик и возможностей создаваемой КСА на особенности бизнеса конкретной организации. КСА, построенные с использованием данного подхода, отличаются сравнительно небольшим временем разработки, эффективностью решения задач автоматизации управления и сравнительно простотой модификации при изменении организационной структуры предприятия или существующих бизнес-процессов.

Рынок адаптируемых интегрированных систем комплексной автоматизации в России

	ЛОКАЛЬНЫЕ СИСТЕМЫ	СРЕДНИЕ ИНТЕГРИРОВАННЫЕ СИСТЕМЫ	КРУПНЫЕ ИНТЕГРИРОВАННЫЕ СИСТЕМЫ
ОСОБЕННОСТИ ПЕРИОД ВНЕДРЕНИЯ	Простое, коробочный вариант. Две недели	Поэтапное. Один и более месяцев	Поэтапное, сложное. Более 9-12-ти месяцев
ФУНКЦИОНАЛЬНАЯ ПОЛНОТА	Учетные системы (по направлениям)	Комплексный учет и управление финансами	Комплексное управление: учет, управление, производство
СООТНОШЕНИЕ ЗАТРАТ: ЛИЦЕНЗИЯ / ВНЕДРЕНИЕ / ОБОРУДОВАНИЕ	1 / 0,5 / 2	1 / 2 / 1	1 / 1-5 / 1
ОРИЕНТИРОВОЧНАЯ СТОИМОСТЬ, тысяч долл.	до 10	10-100	100-500 и более

Таблица 1. Укрупненная классификация систем

На рис. 3 приведена качественная характеристика эффективности применения выделенных видов систем в зависимости от величины предприятия.

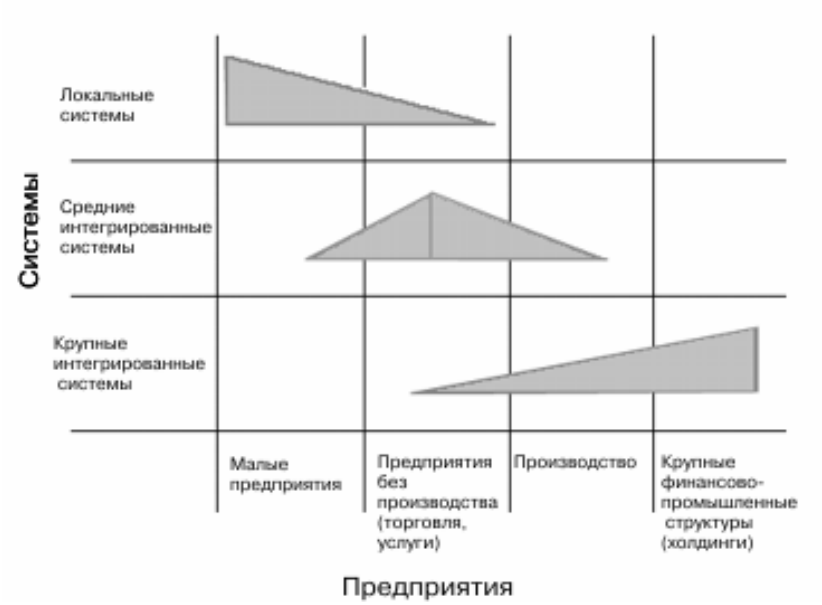


Рис. 3. Эффективность применения систем

Локальные системы

Локальные «коробочные» системы достаточно успешно справляются с решением отдельных задач учета на предприятии, но, как правило, не предоставляют целостной информации для автоматизации управления. Преимуществом этих систем являются сравнительно низкая цена и простота внедрения. Примерами таких систем являются «Инфо-Бухгалтер» (фирма «Информатик») и «Турбо-Бухгалтер» (фирма «Диц»). К этому же классу модно отнести некоторые продукты фирмы «1С» («1С:Бухгалтерия» и др.), а также программы десятков других производителей. Эти программы обладают возможностями адаптации к особенностям предприятия, а некоторые из них, например, «Турбо-Бухгалтер», представляют собой так называемые программы-конструкторы, обладающие расширенными адаптационными возможностями. Тем не менее, для применения таких программ в качестве основы для комплексной автоматизации они не годятся в силу указанных ранее недостатков «коробочных» продуктов.

Крупные интегрированные системы

На сегодняшний день это наиболее функционально развитые и соответственно наиболее сложные и дорогие системы, в которых реализуются западные стандарты управления MRP II и ERP. На российском рынке этот вид систем представлен в основном продуктами западных фирм: SAP, Oracle, Baan, PeopleSoft и Platinum [2]. Ориентация этих продуктов на уровень доходов предприятия приведена в табл. 2.

ФИРМА	ПРОДУКТ	ОРИЕНТАЦИЯ НА КЛИЕНТОВ С ДОХОДАМИ млн. долл.
SAP	R3 (Accelerated Solutions)	до 600
Oracle	Oracle Applications	до 500
BAAN	Baan Midmarket Solutions	до 350
PeopleSoft	PeopleSoft Select	до 250

Табл. 2. Ориентация западных крупных интегрированных систем на уровень доходов предприятия.

Средние интегрированные системы

Средние интегрированные системы обладают ограниченным (специализированным) функционалом. Они вполне конкурентоспособны на отечественном рынке в своей области специализации с крупными западными системами, при этом их стоимость значительно ниже, чем крупных. В этом виде систем доминируют отечественные фирмы-разработчики [2]. Примером могут служить системы «Галактика», «Инфософт», «NS2000» и «ABACUS Financial», фирмами-производителями которых соответственно являются корпорация «Галактика», фирма «Инфософт», фирма «Никос-Софт» и фирма «Омега». К этой же группе можно отнести систему управления предприятием фирмы «Парус», интегрированную систему управления «БЭСТ ПРО» фирмы «Интеллект-сервис», систему комплексной автоматизации финансово-хозяйственной деятельности предприятия AVACCO фирмы AVACCO SOFT; систему управления бизнесом «Монополия» фирмы «ФОРМОЗА СОФТ» и некоторые другие системы, в частности, «"1С:Предприятие» фирмы «1С».

Среди продуктов данного вида можно выделить три группы:

1) системы, представляющие собой переходный вариант от традиционного «коробочного» варианта к средней интегрированной системе, например «1С: Предприятие»;

2) известные интегрированные системы, присутствующие давно на рынке, например, «Галактика», «Парус», «БЭСТ» и др., мигрирующие по мере развития к системам класса MRP и MRP II;

3) новые интегрированные системы, появившиеся на рынке недавно, например AVACCO и Монополия.

Преимуществом систем, давно появившихся на рынке, является то, что их решения многократно апробированы, значит, более надежны. С другой стороны, это влечет за собой необходимость совместимости с предыдущими версиями, в результате чего затрудняется использование новых современных технологий. В новых системах этот недостаток отсутствует.

Обзор возможностей и применений КСА, разработанных отечественными производителями

Как уже говорилось, отечественные тиражируемые КСА принадлежат к классу средних интегрированных систем. Отчасти это связано с тем, что данная отрасль сформировалась относительно недавно и не набрала соответствующие обороты. С другой стороны, они находят достаточно потребителей, так как в современной экономической ситуации предприятия, ориентированные на международные стандарты, составляют небольшой процент от основной массы. Рассмотрим наиболее популярные из них.

Система «1С:Предприятие» предназначена для комплексной автоматизации экономической деятельности предприятий различных направлений деятельности и форм собственности. Она позволяет организовать в единой системе эффективный бухгалтерский учет, кадровый, оперативный торговый учет, а также расчет заработной платы.

В комплект поставки входят компоненты «Бухгалтерский учет», «Оперативный учет» и «Расчет», работающие с единой конфигурацией[2] [4].

Компонент «Бухгалтерский учет» может быть использован для реализации любой схемы бухгалтерского учета и предоставляет гибкие возможности учета, возможность как ручного, так и автоматического ввода бухгалтерских операций, вывода, хранения и печати различных документов, а также формирования и печати разнообразных отчетов.

[2] Под конфигурацией в "1С" понимается конкретный набор объектов и прав пользователя для работы с ними, соответствующие им интерфейсы и экранные формы, структуры информационных массивов, алгоритмы обработки информации и специализированная настройка

Для автоматизации складского учета и торговли реализован компонент «Оперативный (торговый) учет», который может быть использован для учета движения средств в реальном времени. Компонент «Расчет» предназначен для расчета заработной платы и ведения кадрового учета, позволяет автоматизировать проведение других сложных периодических расчетов.

В поставку также входят отдельные конфигурации, реализующие автоматизацию бухгалтерского учета (типовая конфигурация), оперативного торгового учета (конфигурация «Торговля + Склад»), расчета заработной платы и кадрового учета (конфигурация «Зарплата +Кадры»).

Система может поставляться также в сетевом варианте, который обеспечивает одновременную работу нескольких пользователей с единой информационной базой. Очень важным преимуществом «1С:Предприятия» является открытость системы [4].

Предполагается, что такая система будет использоваться в первую очередь для автоматизации деятельности торговых предприятий и организаций. [4]

Система "Галактика" осуществляет информационное обеспечение руководителей различных уровней и категорий – от высшего менеджмента до руководителей подразделений, служб и участков. Численность сотрудников предприятий, внедривших систему «Галактика», составляет от нескольких десятков до 25000 человек. Для крупных компаний, имеющих филиалы и территориально удаленные подразделения, реализована возможность оперативного удаленного доступа и информационного обмена. Специфика конкретного предприятия (корпорации) учитывается с помощью более 300 параметров настройки. Структура системы позволяет вести параллельный многоплановый учет в нескольких стандартах (Россия, GAAP, IAS, HGB и др.) для любого количества филиалов или подразделений предприятия. Кроме того, обладая средствами экономического анализа, система позволяет построить схему налогообложения и определить структуру платежей с целью избежания налоговых переплат и штрафов.

В составе системы «Галактика» реализованы несколько так называемых контуров управления, которые включают в себя функциональные модули системы, среди них: контур административного управления, контур управления персоналом, контур «Бухгалтерского учет», контур оперативного управления, контур администрирования.

Контур управления производством позволяет автоматизировать техническую подготовку производства, технико-экономическое планирование, учет фактических затрат на предприятиях различных отраслей промышленности: машиностроения и приборостроения, легкой, пищевой, химической, горнорудной промышленности, черной и цветной металлургии [5].

Контур отраслевых и специализированных решений включает решения для автотранспортных предприятий; предприятий розничной торговли; компаний, оказывающих услуги по ремонту изделий заказчика; организаций, где необходимо вести учет специальной и форменной одежды [5].

Система «Галактика» предназначена для автоматизации управления в корпорациях со сложной структурой, финансово-промышленных группах, а также на отдельных промышленных и торговых предприятиях.

Система «Парус». Корпорация «ПАРУС» предлагает комплексные решения для автоматизации финансово-управленческой деятельности крупных предприятий, работающих в различных отраслях экономики [6].

Система «Парус» построена на базе СУБД ORACLE и MS Office. В основу решений ПАРУСа положен модульный принцип при взаимосвязи всех подсистем с единой базой данных, что обеспечивает возможность автоматизации полного цикла управления предприятием [6]. «Парус» автоматизирует четыре основные бизнес-направления (бизнес-сферы) финансово-хозяйственной деятельности предприятия: управление финан-

сами, логистика, управление производством, управление персоналом посредством реализованных соответствующих подсистем.

Особенности системы «Парус»:

- мультивалютность и мультивариантность учета (подсистема «Управление финансами»);
- учет товаров по партиям с точностью до модификаций и упаковок (подсистема «Логистика»).
- специализированные приложения: для предприятий нефтяного комплекса, для предприятий энергетики и электрификации, для предприятий связи.

Данная система представляет собой программный комплекс для автоматизации управления предприятием любого размера и структуры.

Система AVACCO предназначена для решения основных задач большинства предприятий – планирования, учета и управления – в различных областях деловой деятельности, в том числе торговле, производстве и бюджетной сфере. Система изначально позиционируется как конструктор, на базе которого может быть реализован самый широкий спектр функциональных возможностей. Система построена по трехуровневой архитектуре «клиент-сервер» [7].

Основой системы AVACCO является сервер приложений AVACCO Server, представляющий собой набор функциональных модулей, которые и определяют возможности системы. Ядро сервера позволяет дополнительно устанавливать необходимые модули даже после ввода системы в эксплуатацию.

В стандартный комплект AVACCO Server входят следующие группы модулей: «Сервер бизнес-процессов», «Финансы», «Склад», «Сервисы» – модули, призванные облегчить работу разработчиков и стандартизировать возможности системы. Например, файловое хранилище позволяет хранить произвольные файлы в базе данных, что применяется при работе VBA и Интернет-клиента, а сервис папок – иерархическая структура, позволяющая хранить любые объекты системы в структурированном виде любой вложенности; модуль «Администрирование» – набор модулей для администратора системы: модуль «Базы данных» (БД), «Конфигуратор модулей», позволяющий настраивать функциональное наполнение каждой БД, установить новый модуль или поставить новую версию существующего, «Пользователи и группы», «Права доступа».

Система базируется на открытых стандартах (используются технологии COM, OLE, ActiveX и ASP), которые позволяют ей взаимодействовать с разнообразными программными продуктами (например, семейство Microsoft Office, программы фирмы 1С) [7].

Анализ средств автоматизации управления

Чтобы выбрать способ создания КСА, позволяющей эффективно решить задачи автоматизации управления бизнесом, необходимо не только иметь представление о рынке систем, но и знать реальные возможности и потребности компании. При этом необходимо выбрать оптимальное сочетание таких параметров, как эффективность от внедрения КСА, универсальность КСА, квалификация персонала, использующего КСА. Так, при увеличении универсальности системы уровень квалификации персонала должен быть выше, причем снижается эффективность ее применения. Для повышения эффективности необходимо выбрать менее универсальную специализированную КСА, при этом требования к квалификации сотрудников снизятся.

Отечественные комплексные системы автоматизации обладают развитой функциональностью, существуют различные варианты их исполнения, в том числе применительно к разным формам и/или отраслям хозяйственной деятельности предприятий. Немаловажной характеристикой предлагаемых средств автоматизации является то, что

они учитывают специфику российского бизнеса. К тому же отечественные КСА обладают низкой стоимостью по сравнению с зарубежными аналогами.

В действительности даже самые современные системы автоматизации управления предприятием, как отечественные, так и зарубежные, управленческих решений сегодня самостоятельно не принимают. Автоматизация сегодня – это автоматизация разных областей учета, документооборота и т.д. с целью оперативной подготовки информации для принятия руководителями различных уровней обоснованных управленческих решений, так как для принятия управленческих решений на верхнем уровне необходимо учитывать очень большое число трудно формализуемых факторов.

Заключение

Необходимость оперативного реагирования на конъюнктуру рынка и быстро меняющуюся экономическую ситуацию требует перестройки внутренней микроэкономики предприятия, постановки управленческого учета, оптимизации процессов управления. Эффективность деятельности предприятия во многом определяется эффективностью его системы управления. Процесс управления предприятием реализуется в рамках системы управления предприятием в зависимости от особенностей и масштаба предприятия. Комплексная система автоматизации предприятия обеспечивает ведение оперативного, бухгалтерского и управленческого учета и строится на основе единого информационного пространства, охватывая и координируя всю совокупность управленческих процессов предприятия. Она представляет собой информационную систему, в которой оперативно накапливаются и обрабатываются данные о текущей финансово-хозяйственной деятельности предприятия, и если эта система выбрана и реализована правильно, она помогает устранить многие недостатки в управлении.

Существует несколько подходов к построению комплексной системы управления – разработка своими силами, заказ на разработку такой системы, использование готовых решений. Проанализировав свои требования к комплексной автоматизированной системе, а также задачи, для решения которых она будет применяться, и оценив свои возможности для ее создания, российский руководитель может выбрать наиболее рациональное решение.

На современном российском рынке представлен широкий ассортимент программного обеспечения, применяемого для автоматизации управления, как российского производства, так и зарубежных фирм-производителей.

Средние интегрированные системы обладают ограниченным (специализированным) функционалом. Они вполне конкурентоспособны на отечественном рынке в своей области специализации с крупными западными системами, при этом их стоимость значительно ниже, чем крупных. В этом виде систем доминируют отечественные фирмы-разработчики.

В зависимости от размера своего предприятия, профиля его деятельности, финансовых возможностей и целей, которые он преследует, руководитель может использовать соответствующий программный продукт, как основу для построения комплексной автоматизированной системы. Эти системы позволяют руководителю принимать обоснованные управленческие решения на основе качественной и достоверной информации, а также обеспечить персонал эффективными инструментами для выполнения должностных функций.

Литература

1. Петров Ю.А., Шлимович Е.Л., Ирюпин Ю.В. Комплексная автоматизация управления предприятием. / Информационные технологии - теория и практика М.: Финансы и статистика, 2001. 160 с.

2. Баронов В.В., Калянов Г.Н., Попов Ю.И. и др. Автоматизация управления предприятием М.: ИНФРА-М, 2000. 238 с. (Секреты менеджмента).
3. Филиппенко И. Выбор ПО для автоматизации управления [Электронный ресурс]. Режим доступа: http://www.cfin.ru/itm/selectsoft.shtml.
4. WWW.1C.RU Фирма «1С» [Электронный ресурс]. – Режим доступа: http://www.1c.ru
5. Корпорация «Галактика». Автоматизация управления предприятием [Электронный ресурс]. Режим доступа: http://www.galaktika.ru.
6. Корпорация ПАРУС – корпоративные системы управления для предприятий и государственных структур [Электронный ресурс]. Режим доступа: http://www.parus.ru.
7. Компания AVACCO SOFT [Электронный ресурс]. Режим доступа: http://www.avacco.ru.

7 СИСТЕМЫ ОРИЕНТАЦИИ И НАВИГАЦИИ

ОСОБЕННОСТИ КОНСТРУИРОВАНИЯ ЧУВСТВИТЕЛЬНОГО ЭЛЕМЕНТА МИКРОМЕХАНИЧЕСКОГО ГИРОСКОПА

М.И. Евстифеев, А.А. Унтилов

Рассмотрены особенности конструирования чувствительного элемента микромеханического гироскопа. Приведены требования к параметрам конструкции. Проведен анализ влияния технологических особенностей материала и инструментальных погрешностей изготовления.

Введение

В настоящее время все большее внимание уделяется разработке микромеханических гироскопов (ММГ), выполненных с использованием технологий микроэлектронной промышленности. Несмотря на невысокую точность, ММГ находят применение в автомобильной промышленности, при создании нового поколения навигационного оборудования, в разработках робототехнических устройств и спортивного снаряжения, в медицинской промышленности и при выпуске товаров широкого потребления [1]. Принцип работы большинства ММГ аналогичен осцилляторным вибрационным гироскопам и основан на измерении сил Кориолиса от внешней угловой скорости Ω (рис. 1). Для увеличения выходного сигнала и повышения чувствительности в ММГ, как правило, используется резонансная настройка первичных f_1 и вторичных f_2 колебаний инерционной массы, при этом необходимо, чтобы амплитуда первичных колебаний инерционного тела и добротность по оси вторичных колебаний поддерживались на максимальном уровне, а собственная частота была минимальной [2]. Устранение противоречия между достижением высокой добротности и, соответственно, максимальной чувствительности и обеспечением требуемой полосы пропускания возможно при использовании интегрирующих свойств ММГ [3].

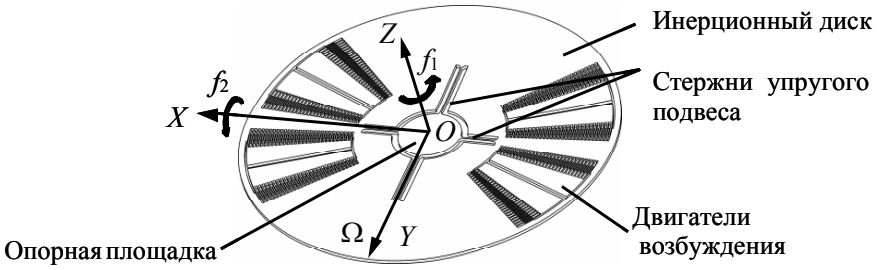


Рис. 1. Конструкция ротора ММГ: f_1 и f_2 – первичные и вторичные колебания; Ω – измеряемая угловая скорость основания

Постановка задачи

Разрабатываемые микромеханические гироскопы (ММГ) отличаются от традиционных гироскопических устройств рядом существенных факторов, которые следует принимать во внимание при проектировании. Проектирование ММГ требует решения ряда специфических задач конструкторско-технологического характера [4, 5], среди которых:

- построение расчетной схемы, адекватно описывающей характеристики подвеса;

- оптимизация параметров конструкции, обеспечивающая требуемые соотношения между собственными частотами колебательной системы и соответствующие формы колебаний;
- подбор материалов с необходимыми физическими характеристиками;
- достижение минимального порога чувствительности;
- обеспечение прочности конструкции в процессе эксплуатации;
- поиск способов уменьшения влияния технологических, температурных и иных факторов на точность и стабильность характеристик приборов.

При выполнении расчетов и разработке конструкции чувствительного элемента можно выделить следующие особенности проектирования упругого подвеса ММГ: неопределенность механических характеристик материала, использование планарных конструкций, учет возможной точности изготовления упругих элементов, оценка нелинейности упругих характеристик.

Особенности используемых материалов

При изготовлении инерционной массы ММГ толщиной несколько десятков микрометров преимущественно используется монокристаллический кремний. Среди преимуществ монокристаллического кремния следует отметить низкие внутренние потери, позволяющие достичь высокой добротности осцилляторов; высокий модуль Юнга, сравнимый с модулем Юнга стали; предел текучести, превышающий предел текучести стали [6, 7]. Однако существенным недостатком этого материала является анизотропия свойств (в том числе механических) в зависимости от кристаллографических направлений [8]. На рис. 2 приведено изменение модуля Юнга и коэффициента Пуассона на пластине с ориентацией (100).

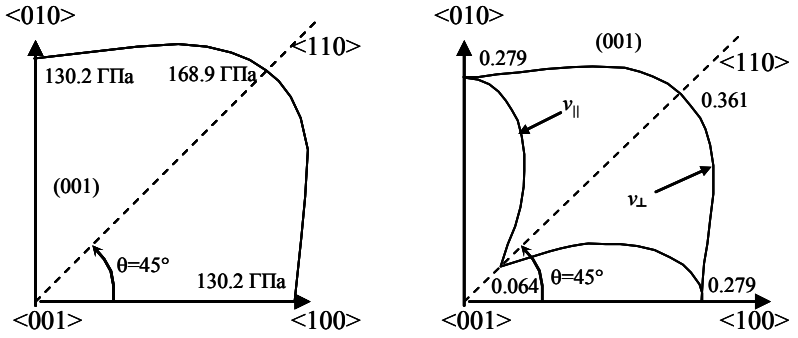


Рис. 2. Изменение модуля Юнга (а) и коэффициента Пуассона (б) на пластине с ориентацией (100).

При проектировании необходима информация о расположении упругого подвеса на пластине-заготовке, в противном случае ошибки в расчете собственных частот могут оказаться недопустимо большими. Исследования показывают, что соотношение между частотами первичных и вторичных колебаний может изменяться на 10% при одинаковой геометрии в зависимости от кристаллографического направления [9].

Несмотря на высокие конструкционные характеристики, монокристаллический кремний хрупок, вследствие чего следует избегать конструкций, в которых чувствительный элемент может испытывать удары о корпус или подложку.

При расчете параметров следует уделить внимание таким вопросам, как подбор материалов с одинаковыми коэффициентами теплового расширения [10], учет изменения модуля упругости и добротности вследствие изменения температуры [6,11], разра-

ботка адекватной модели тепловых погрешностей прибора с целью их компенсации [12], разработка систем терморегулирования [13].

Температурные зависимости модуля упругости и добротности приводят к погрешностям прибора вследствие изменения геометрических размеров и упругих характеристик конструкции. При этом происходит изменение частотных свойств и добротности, зазоров в электрических датчиках прибора и соответственно энергетических характеристик. В конструкции появляются температурные градиенты и возникают внутренние механические напряжения, вследствие которых возможно появление перекосов и нарушения геометрии. Изменение собственных частот и соотношение между ними при изменении температуры для рассматриваемой конструкции приведено на рис. 3 [14].

$$\Delta f = \frac{f_2 - f_1}{f_2} \cdot 100\%$$

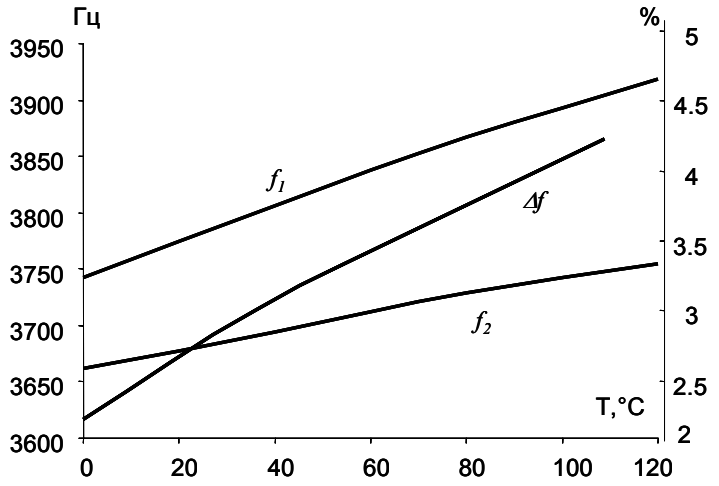


Рис. 3. Изменение собственных частот чувствительного элемента ММГ f_1, f_2 и соотношения между ними Δf при изменении температуры

Особенности технологии изготовления

Планарная технология, используемая при изготовлении ММГ, накладывает определенные ограничения на выбор конструктивной схемы. Вследствие технологических особенностей кремниевые ММГ представляют собой плоские конструкции, у которых толщина в 100–200 раз меньше длины. Это обстоятельство затрудняет обеспечение требуемых упругих характеристик и исключает использование пространственных конструктивных схем.

Точность расчетных моделей связана с учетом возможной точности изготовления упругих элементов подвеса. В конструкциях ММГ вследствие группового построения технологического процесса, большой серийности производства и малых размеров обеспечить допуски изготовления на уровне долей процента от геометрического размера невозможно и экономически нецелесообразно. Технологические допуски приводят к расстройке резонанса и резкому изменению масштабного коэффициента [15]. Для ММГ с резонансной настройкой необходима регулировка частот путем использования электрической системы создания положительной или отрицательной жесткости.

Отклонение от номинальных значений размеров чувствительного элемента приводит к статическому и динамическому дисбалансам, перекосам ротора, несовпадению кристаллографических направлений с осями прибора. Помимо расстройки резонанса, такие погрешности приводят к появлению «нулевого» сигнала.

Проблемы стабильности характеристик ММГ связаны в основном с непостоянством геометрических размеров, упругих свойств и добротности. Коэффициент добротности для данного класса приборов является метрологическим параметром, от которого зависит стабильность показаний прибора. Диссипация энергии колебаний связана с потерями на газовое трение, поверхностными потерями, потерями из-за связанности различных типов колебаний, термоупругим демпфированием и прочим [6]. Снижение потерь от взаимодействия с газовой средой требует вакуумирования внутренней полости элемента [16]. Одним из принципиальных вопросов разработки конструкции является использование технологии вакуумирования на уровне вафли, что позволяет снизить стоимость и повысить надежность прибора [17]. При конструировании ММГ следует принимать во внимание особенности технологии изготовления и исключать возможность адгезионного слипания элементов конструкций вследствие воздействия температур в процессе сварки и вакуумных ударов при герметизации.

Особенности конструкции чувствительного элемента

Одной из проблем разработки конструкции упругих подвесов является обеспечение линейности характеристик жесткости по осям первичных и вторичных колебаний. Для подвесов, состоящих из прямых стержней (см. рис. 1), восстанавливающая сила подвеса является нелинейной и содержит кубические члены жесткости подвеса. Это приводит к возникновению неустойчивых колебательных режимов и возможности перехода системы из одного положения в другое без каких-либо дополнительных внешних воздействий [18]. Для ММГ компенсационного типа нелинейность упругого подвеса оказывает существенное влияние только на первичные колебания.

Одним из существенных факторов, влияющих на параметры разрабатываемого ММГ, является порог чувствительности [19]. Обеспечение высокой точности измерения угловой скорости основания требует создания высокодобротных, колеблющихся с большой амплитудой осцилляторов и уменьшения практически до нуля механических и электрических шумов, что может быть достигнуто путем тщательного проектирования всех элементов прибора.

Условия эксплуатации ММГ предполагают наличие интенсивных инерционных воздействий – высокочастотной вибрации и ударов. При проектировании упругого подвеса ММГ следует учитывать моменты сил упругости, возникающие из-за деформаций конструкции в условиях угловой и поступательной вибрации. Наибольшее влияние на показания ММГ с угловыми движениями инерционного тела оказывает воздействие поступательной вибрации на частоте, равной половине частоты вторичных угловых колебаний [20]. Это требует разработки равножестких конструкций упругих подвесов, что затрудняется использованием планарной технологии.

Конструкции ММГ должны иметь собственные частоты выше диапазона вибрации и выдерживать ударные воздействия величиной в десятки тысяч g. Большая прочность конструкций обусловлена малой массой инерционного тела и высокими частотами колебаний подвеса на уровне килогерц. При конструировании необходимо предусматривать установку упоров – ограничителей перемещения элементов конструкции для предотвращения замыкания контактных проводников.

Использование трехмерного моделирования и конечно-элементного анализа позволяет значительно сократить время разработки и расчета нового варианта конструкции ММГ, избежать итераций методом «проб и ошибок» [14]. Тем не менее, точность расчета конечна и составляет несколько процентов от номинального значения частоты.

Проблемы расчета конструкций связаны с необходимостью создания расчетных моделей, описывающих системы с распределенными параметрами, и использования программ конечно-элементного анализа [21]. Для разработки конструкции и решения

обозначенных выше проблем могут быть использованы как универсальные (ANSYS, Nastran, Pro/MECHANICA), так и специализированные программы (CoventorWare, MEMSCap).

Литература

1. Пешехонов В.Г. Гироскопы начала XXI века // Гироскопия и навигация. 2003. № 4. С.5–18.
2. Кучерков С.Г., Шадрин Ю.В. К вопросу о выборе конструктивных параметров микромеханического кольцевого гироскопа вибрационного типа / Материалы III конференции молодых ученых «Навигация и управление движением». СПб.: ЦНИИ «Электроприбор», 2001. С.94–101.
3. Кучерков С.Г. Использование интегрирующих свойств вибрационного микромеханического гироскопа с резонансной настройкой для построения датчика угловой скорости компенсационного типа. //.Гироскопия и навигация. 2002. №2. С.12–18.
4. Северов Л.А., Пономарев В.К., Панферов А.И., Сорокин А.В., Кучерков С.Г., Лучинин В.В., Корляков А.В.. Микромеханические гироскопы: конструкции, характеристики, технологии, пути развития. // Известия вузов. Приборостроение. 1998. Т.41. №1–2. С.57–73.
5. Лестев А.М., Попова И.В., Евстифеев М.И. и др. Особенности микромеханических гироскопов. // Микросистемная техника. 2000. №4. С.16–18.
6. Duwel A. et al. Experimental Study of thermoelastic damping in MEMS gyros// Sensor and Actuators, 103, 2003,. pp.70-75.
7. Петерсен К.Э. Кремний как микромеханический материал. // ТИИЭР. 1982. Т.70. №5. С.5–49.
8. Kim J., Cho D., Muller R.S. Why is (111) silicon a better mechanical material for MEMS? / Proceedings of Transducers 2001: 11th International Conference on Solid State Sensors and Actuators, Munich, Germany, June 2001, pp. 662-665.
9. Унтилов А.А. Влияние анизотропии монокристаллического кремния на характеристики микромеханического гироскопа. / Навигация и управление движением: Материалы докладов VI конференции молодых ученых «Навигация и управление движением»/Под общ. ред. академика РАН В.Г.Пешехонова СПб.: ГНЦ РФ ЦНИИ «Электроприбор», 2005. С.154–161.
10. Ahn Y., Guckel H. Thermoelastic effect of silicon for strain sensing. J. Micromech. Microeng. 11 (2001) pp. 443–451.
11. Александров Л.Н., Зотов М.И. Внутреннее трение и дефекты в полупроводниках. Новосибирск: Наука, 1979.
12. Джашитов В.Э., Панкратов В.М. Математические модели теплового дрейфа гироскопических датчиков инерциальных систем. СПб.: ГНЦ РФ ЦНИИ «Электроприбор», 2001.. 150 с.
13. Барулина М.А., Джашитов В.Э., Панкратов В.М. Математические модели систем терморегулирования микромеханических гироскопов. // Гироскопия и навигация. 2002. №3. С. 48–59.
14. Евстифеев М.И., Унтилов А.А. Особенности проектирования чувствительного элемента микромеханического гироскопа / III международный симпозиум «Аэрокосмические приборные технологии» сборник материалов. СПб. 2-4 июня 2004. С.297–298.
15. Евстифеев М.И., Унтилов А.А. Требования к точности изготовления упругого подвеса микромеханического гироскопа. // Гироскопия и навигация. 2003. №2. С.24–31.

16. Кучерков С.Г. Определение необходимой степени вакуумирования рабочей полости осциллятора микромеханического гироскопа. // Гироскопия и навигация.. 2002. № 1. С. 52–56.
17. Goldberg H., Selvakumar A. Issues & challenges of MEMS wafer-level packaging// MicroNano, №9, 2003, p.8–9.
18. Davis W.O., Pisano A.P. Nonlinear Mechanics of Suspension Beams for a Micromachined Gyroscopes//Modeling and Simulation of Microsystems 2001, pp.270–273.
19. Ayazi F., Najafi K. A HARPSS Polysilicon Vibrating Ring Gyroscope // IEEE, Journal of Microelectromechanical Systems. 2001. Vol.10. №2. P.169–179.
20. Евстифеев М.И. Погрешности микромеханического гироскопа на вибрирующем основании. // Гироскопия и навигация. 2002. №2. С.19–25.
21. Евстифеев М.И., Кучерков С.Г., Унтилов А.А. Шадрин Ю.В., Шалобаев Е.В. Использование мехатронного подхода при анализе систем компьютерного проектирования микромеханических гироскопов. // Мехатроника, автоматизация, управление. 2004.. №2. С.31–37.

МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ РЕШЕНИЯ ЗАДАЧИ ОРИЕНТАЦИИ В СРЕДЕ ГСНС

В.В.Серегин, В.И. Ющенко

Обобщаются результаты исследований проблемы определения ориентации подвижного объекта по информации от спутниковых навигационных систем, опубликованные в периодической печати. Сравниваются результаты использования фазовых и частотных измерений. Даются рекомендации по их совместной обработке с целью повышения эффективности получаемой информации.

Введение

Современное направление развития систем ориентации и навигации можно характеризовать все более проникающим объединением инерциальных методов и технологий навигационных спутников. В области инерциальных чувствительных элементов устойчивая тенденция состоит в использовании малогабаритных, сравнительно дешевых измерителей [1]. Такие элементы, как известно, относятся к среднему, а чаще низкому классу точности. Построение систем ориентации и навигации с применением таких инерциальных измерительных модулей возможно [2] только на основе комплексирования вырабатываемой ими информации с информацией от других типов измерителей. В настоящее время основным и наиболее точным источником такой информации признаны глобальные спутниковые навигационные системы (ГСНС) типа GPS и ГЛОНАСС.

Традиционным можно считать использование информации о координатах объекта, полученной от ГСНС, при комплексной обработке навигационной информации. В современных интегрированных инерциально-спутниковых системах навигации в алгоритмы обработки также вводят информацию о скорости объекта, вычисленной в аппаратуре потребителя ГСНС. В журнальных статьях и в материалах научных конференций эти проблемы освещены достаточно подробно, (см., например.[3]). В аппаратуре потребителя ГСНС географические координаты и составляющие скорости объекта вычисляются путем обработки кодовых сигналов, принимаемых от навигационных спутников.

Однако в связи с повышением качества работы ГСНС и совершенствованием аппаратуры потребителя появилась возможность получать дополнительную информацию за счет обработки несущей частоты сигналов. В частности, измерение фазы несущей частоты позволяет существенно повысить точность измерения псевдодальностей и, как следствие, вычисленных координат. Качественно новым направлением использования информации, содержащейся в несущей частоте, является определение ориентации объекта [4, 5]. При включении этой информации в алгоритм начальной выставки инерциальной навигационной системы (ИНС) возможно существенное уменьшение времени готовности системы, особенно на подвижном объекте. Кроме того, появляется возможность непрерывной калибровки инерциальных измерительных модулей в реальных условиях. Особенно актуально комплексирование этих данных об ориентации объекта для решения задачи курсоуказания в системах, в которых не может быть реализован режим гирокомпасирования из-за низкой точности инерциальных чувствительных элементов.

В настоящее время уже имеются опытные разработки интегрированных систем, в которых используется информация об угловом положении объекта, полученная от много антенной аппаратуры потребителя ГСНС [3]. Проведенные испытания показали перспективность положенных в их основу решений. Однако теоретические основы определения ориентации объекта по информации, содержащейся в несущей частоте, проработаны недостаточно полно и не позволяют эффективно использовать ее, особенно на

объектах с высокой динамикой угловых движений. Поэтому необходимо обобщить уже имеющиеся результаты исследований этой проблемы, наметить перспективные направления развития и дать рекомендации по наиболее эффективному комплексированию различных видов информации об угловом движении объекта.

Информационная среда

Основными параметрами несущей частоты сигналов, принимаемых от навигационных спутников, являются частота и фаза колебаний. При этом частота отличается от известной частоты излучаемого сигнала на величину доплеровского сдвига, обусловленного относительным перемещением объекта и спутника. Фаза колебаний определяется псевдодальностью между ними. Эти параметры и определяют информационную среду, в которой решается задача определения ориентации объекта.

Суть интерферометрического принципа, реализуемого при использовании фазы несущей частоты, состоит в измерении разности фаз сигналов, принимаемых от спутника на разнесенные антенны (см. рис.1). В дальнейшем для краткости будем называть это **фазовым методом.**

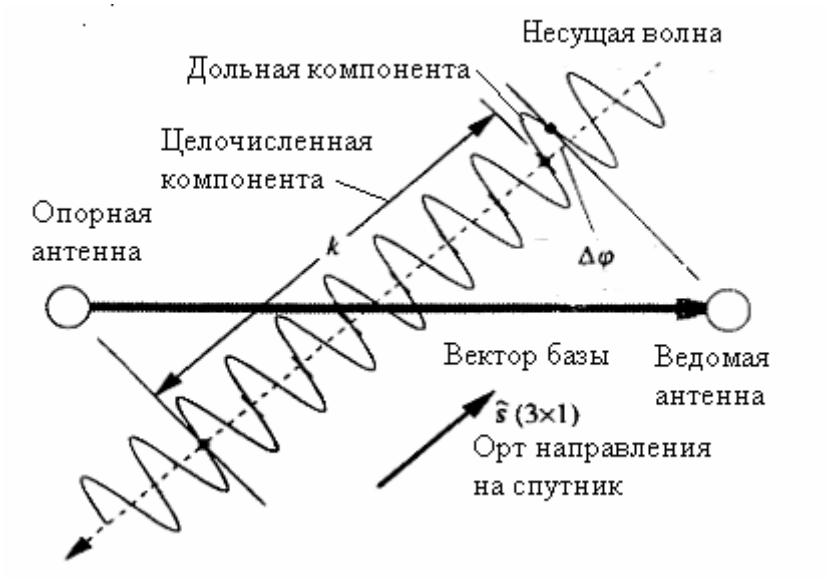


Рис. 1. Фазовый метод

Из рис.1 видно, что полная разность фаз сигналов, принимаемых опорной и ведомой антеннами, может быть представлена в виде

$$\varphi_{ij} = \Delta\varphi_{ij} + \lambda \cdot k_{ij} + \varepsilon_{ij}\ , i = 1\ldots m, j = 1\ldots n, \qquad (1)$$

где $\Delta\varphi_{ij}$ – дольная часть длины волны, λ – длина несущей волны, k_{ij} – целое число волн, ε_{ij} – ошибка измерения, i – номер вектора базы, j – номер навигационного спутника. В единицах длины эта разность фаз может быть выражена также через модуль вектора базы $\overline{b}^{\,i}$

$$\varphi_{ij} = \left|\overline{b}^{\,i}\right| \cdot \cos\beta_{ij}\ , \qquad (2)$$

где β_{ij} – угол между i-м вектором базы и ортом направления от антенны на j-ый спутник. Из сравнения (1) и (2) вытекает, что по величине разности фаз φ_{ij} можно вычислить ориентацию вектора базы в плоскости, проходящей через базу антенн и местоположение спутника, относительно орта $\overline{s}^{\,j}$. Проблема состоит в том, что непосредственно может быть измерена только дольная часть фазы. Присутствие в (1) целого числа

длин волн приводит к неоднозначности фазовых измерений, разрешение которой представляет сложную самостоятельную задачу. В литературе, например, в [6], предложены различные методы исключения этой неоднозначности, и здесь они рассматриваться не будут.

Как отмечено выше, вторым информационным параметром является доплеровский сдвиг частоты. Результаты измерения разности частот сигналов, принимаемых опорной и ведомой антеннами (см. рис. 2), дают разность доплеровских сдвигов частот в каждой антенне и могут быть использованы для решения задачи ориентации. Будем называть этот подход **частотным методом.**

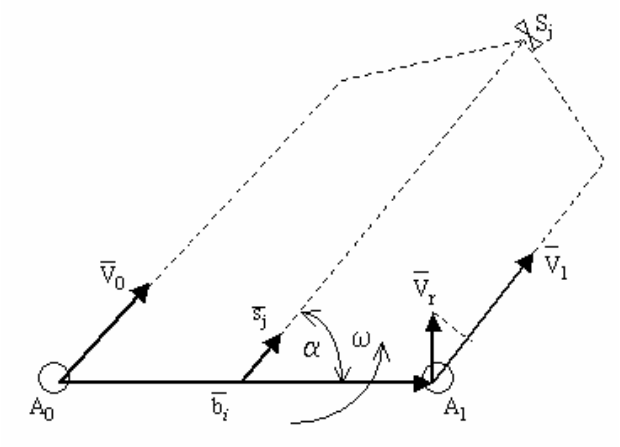


Рис. 2. Частотный метод

На рис.2 изображена плоскость, проходящая через вектор базы $\bar{b}^i$ и навигационный спутник S_j, и приняты обозначения: A_0, A_1 – опорная и ведомая антенны соответственно, $\bar{V}_0, \bar{V}_1$ - скорости относительного движения фазовых центров антенн (ФЦА) и спутника, ω - скорость поворота ведомой антенны относительно опорной из-за углового движения объекта, $\bar{V}_r$ - линейная скорость ФЦА антенны A_1, обусловленная этим движением, $\bar{s}^j$ - орт направления из центра базы антенн на спутник.

В [7] получено соотношение для разности доплеровских частот F_1 и F_0, принимаемых антеннами A_1 и A_0, в виде

$$F_{ij}^r = F_1 - F_0 = f_s \frac{\bar{V}_r^i \cdot \bar{s}^j}{c} = \frac{\left|\bar{\omega}_{ij} \times \bar{b}^i\right|}{\lambda} \sin \alpha_{ij} \ , \qquad (3)$$

где f_s, λ – частота и длина волны сигнала, излучаемого навигационным спутником, $\bar{\omega}_{ij}$ - проекция вектора угловой скорости объекта на нормаль к плоскости, содержащей вектора $\bar{b}^i$ и $\bar{s}^j$, c – скорость распространения электромагнитных колебаний. Непосредственно из (3) видно, что, измерив разность частот F_{ij}^r, можно вычислить угловое положение вектора базы относительно орта направления на данный навигационный спутник. При наличии избыточной информации, получаемой от нескольких спутников, по разности доплеровских частот можно вычислить, кроме угла, также угловую скорость объекта. Ограничением на применение частотного метода является требование, чтобы угловая скорость объекта была отлична от нулевого значения.

Определение ориентации

В задачах управления и/или стабилизации объекта его ориентация определяется относительно некоторой базовой системы координат (СК), в качестве которой часто

используется географическая СК *NHE*: ось *N* направлена по полуденной линии на север, ось *H* совпадает с внешней нормалью к поверхности референц-эллипсоида. Ориентация объекта задается взаимным расположением базовой СК и связанной СК *XYZ*: ось *X* – по продольной оси объекта, ось *Z* – по нормали к продольной оси вверх. Начала этих СК можно совместить, например, с фазовым центром опорной антенны, что не снижает общности решения задачи.

После решения в аппаратуре потребителя ГСНС основной навигационной задачи по определению координат объекта, используя также известные координаты данного спутника, можно вычислить направляющие косинусы (составляющие) орта $\bar{s}^j$ в базовой СК. Отметим, что важным методологическим фактором в данном случае является возможность определить проекции орта на оси базовой СК без привлечения дополнительной внешней информации, а используя только информацию, полученную в среде ГСНС. Очевидно, что для того, чтобы найти величины проекций вектора базы $\bar{b}^i$ на оси базовой СК, необходимо располагать его ориентацией относительно направлений на два навигационных спутника и иметь априори известную длину базы антенн, а именно:

$$\begin{aligned}
&\bar{b}^i \cdot \bar{s}^j = b_N^i \cdot s_N^j + b_H^i \cdot s_H^j + b_E^i \cdot s_E^j;\\
&\bar{b}^i \cdot \bar{s}^{j+1} = b_N^i \cdot s_N^{j+1} + b_H^i \cdot s_H^{j+1} + b_E^i \cdot s_E^{j+1};\\
&\bar{b}^i \cdot \bar{b}^i = \left(b_N^i\right)^2 + \left(b_H^i\right)^2 + \left(b_E^i\right)^2,
\end{aligned} \qquad (4)$$

где $b_Б^i$ (*Б=N,H,E*) – проекции вектора $\bar{b}^i$ на оси географической СК; $s_Б^j, s_Б^{j+1}$ (*Б=N,H,E*) - направляющие косинусы соответствующих ортов на те же оси. В том случае, когда можно одновременно наблюдать три навигационных спутника (минимальное число в созвездии, необходимое для решения основной задачи навигации), вместо (4) имеем матричное уравнение

$$\begin{Vmatrix} \bar{b}^i \cdot \bar{s}^j \\ \bar{b}^i \cdot \bar{s}^{j+1} \\ \bar{b}^i \cdot \bar{s}^{j+2} \end{Vmatrix} = \begin{Vmatrix} s_N^j & s_H^j & s_E^j \\ s_N^{j+1} & s_H^{j+1} & s_E^{j+1} \\ s_N^{j+2} & s_H^{j+2} & s_E^{j+2} \end{Vmatrix} \cdot \begin{Vmatrix} b_N^i \\ b_H^i \\ b_E^i \end{Vmatrix}. \qquad (5)$$

Компоненты b_C^i (*C=X,Y,Z*) векторов баз в связанной СК можно считать точно известными. Поэтому после определения компонент $b_Б^i$ (*Б=N,H,E*) в географической СК задача ориентации решается на основе **метода векторного согласования** [8]. Суть метода состоит в том, что компоненты вектора $\bar{b}$, известные или измеренные в двух различных СК, связаны между собой через матрицу преобразования от одной СК к другой. В данном случае имеем

$$\begin{Vmatrix} b_X^i & b_Y^i & b_Z^i \end{Vmatrix}^T = A_{СБ} \cdot \begin{Vmatrix} b_N^i & b_H^i & b_E^i \end{Vmatrix}^T, \qquad (6)$$

где $A_{СБ}$ - матрица перехода от географической СК к связанной. Если матрица $A_{СБ}$ параметризована с помощью углов курса *K* и качки ϑ, γ, то эти углы вычисляются непосредственно по элементам матрицы. Следует иметь в виду, что для однозначного определения матрицы $A_{СБ}$ из (6) необходимо располагать как минимум двумя неколлинеарными векторами. При решении задачи ориентации объекта – это два вектора баз антенн, развернутые друг относительно друга на некоторый угол.

В [7] приведены результаты исследований, согласно которым измерения доплеровского сдвига частот несущего сигнала позволяют определять как углы ориентации. так и угловые скорости объекта. Для этого достаточно **методом наименьших квадратов** (МНК) решить векторно-матричное уравнение, составленное для трехбазовой антенной системы,

$$\|\Delta\omega\| + \|\omega\| \cdot \left\|(\Delta A_{СБ})^T \cdot A_{СБ}\right\| = \left\|\overline{V}_r^1 \quad \overline{V}_r^2 \quad \overline{V}_r^3\right\| \cdot \left\|\overline{b}^1 \quad \overline{b}^2 \quad \overline{b}^3\right\|^{-1} \cdot A_{СБ} - \|\omega\|, \qquad (7)$$

где $\|\omega\|$ – кососимметрическая матрица из составляющих вектора $\overline{\omega}$ в географической СК, $\Delta\omega, \Delta A_{СБ}$ – приращения элементов вычисляемых матриц. Система 9 скалярных уравнений, полученных из (7), решается методом последовательных приближений.

В [9] было показано, что по результатам частотных измерений можно непосредственно вычислить компоненты векторов баз антенн в географической СК. Для этого необходимо решить матричное уравнение, полученное для трех ортогональных векторов баз, с учетом наличия в нем избыточной информации:

$$\begin{Vmatrix} \|-b_Б^1\| & \|\omega\| & 0 & 0 \\ \|-b_Б^2\| & 0 & \|\omega\| & 0 \\ \|-b_Б^3\| & 0 & 0 & \|\omega\| \\ 0 & (\overline{b}^2)^T & (\overline{b}^1)^T & 0 \\ 0 & (\overline{b}^3)^T & 0 & (\overline{b}^1)^T \\ 0 & 0 & (\overline{b}^3)^T & (\overline{b}^2)^T \\ 0 & (\overline{b}^1)^T & 0 & 0 \\ 0 & 0 & (\overline{b}^2)^T & 0 \\ 0 & 0 & 0 & (\overline{b}^3)^T \\ 0 & (\overline{d}^1)^T & (\overline{d}^2)^T & (\overline{d}^3)^T \end{Vmatrix} \cdot \begin{Vmatrix} \Delta\overline{\omega} \\ \Delta\overline{b}^1 \\ \Delta\overline{b}^2 \\ \Delta\overline{b}^3 \end{Vmatrix} = \begin{Vmatrix} \overline{V}_r^1 - \overline{\omega} \times \overline{b}^1 \\ \overline{V}_r^2 - \overline{\omega} \times \overline{b}^2 \\ \overline{V}_r^3 - \overline{\omega} \times \overline{b}^3 \\ C_{12} - (\overline{b}^1)^T \cdot \overline{b}^2 \\ C_{13} - (\overline{b}^1)^T \cdot \overline{b}^3 \\ C_{23} - (\overline{b}^2)^T \cdot \overline{b}^3 \\ C_{11} - (\overline{b}^1)^T \cdot \overline{b}^1 \\ C_{22} - (\overline{b}^2)^T \cdot \overline{b}^2 \\ C_{33} - (\overline{b}^3)^T \cdot \overline{b}^3 \\ C_{123} - (\overline{b}^1)^T \cdot (\overline{b}^2 \times \overline{b}^3) \end{Vmatrix}, \qquad (8)$$

где $\Delta\overline{b}^i$ $(i = 1,2,3), \Delta\overline{\omega}$ – приращения вычисляемых элементов матрицы; C_{ij} $(i, j=1,2,3)$ – константы, характеризующие вектора баз и их взаимное положение; $\overline{d}^i$ $(i = 1,2,3)$ - векторы, взаимные к векторам баз антенн в их проекциях на оси географической СК. Далее, используя известные направляющие косинусы орта $\overline{s}^j$, можно пересчитать векторы баз на направление от объекта на спутник и выделить целое число длин волн, содержащееся в модуле вектора. Это дает возможность исключить неоднозначность фазовых измерений за одну измерительную эпоху, что имеет большое значение для определения ориентации объектов с высокой динамикой угловых движений.

Сравнение методов. Использование информации, полученной фазовым или частотным методом, для определения ориентации объекта имеет как преимущества, так и недостатки в сравнении друг с другом. Для наглядности эти свойства сведены в табл. 1. Из их анализа следует вывод, что наилучшие результаты будут при совместном использовании информации, полученной этими двумя методами.

Совместная обработка фазовых и частотных измерений. Каждый из методов преобразования информации несущей волны обладает как положительными качествами, так и отрицательными. Для повышения эффективности использования информации естественно применить процедуру их совместной обработки. Возможные варианты такой процедуры показаны на рис. 3 и 4. Различаются они, в основном, алгоритмом обработки результатов измерения доплеровских сдвигов частоты.

В первом из вариантов (см. рис.3) линейная скорость $\overline{V}_r$ ведомой антенны, полученная путем обработки разностей доплеровских частот, используется для вычисления ориентации вектора базы относительно орта $\overline{s}^j$ направления на данный спутник. Спроецировав вектор базы на орт, находим целое число длин волн, содержащееся в этой проекции. При этом для повышения точности вычислений можно применить какой-либо фильтр, выделяющий именно целое число k_{ij}. Это решает задачу исключения

неоднозначности фазовых измерений. Кроме того, доплеровские частоты, измеренные во всех антеннах, используются в алгоритме вычисления угловых скоростей объекта.

Характеристика	Метод фазовых измерений	Метод частотных измерений
1	2	3
Источник первичной информации	Разность фаз несущего сигнала НКА ведомой антенны относительно опорной	Разность доплеровских сдвигов частот несущего сигнала НКА, принятого ведомой и опорной антеннами
Вторичная информация	Угловое положение базы антенны относительно направления на НКА	Разность линейных скоростей ФЦА в проекциях на оси географической СК
Определяемые параметры	Угловая ориентация объекта в географической СК	Угловые скорости и углы, определяющие ориентацию объекта в географической СК
Причина получения неоднозначного решения	Неизвестное число целых длин волн в разности псевдодальностей антенн и НКА при единичном измерении	Сохранение ориентации вектора линейной скорости при одновременном изменении направления векторов базы и угловой скорости на противоположное
Влияние динамики углового движения объекта	Снижает точность определения ориентации, увеличивает время обработки информации	Повышает точность определения ориентации и угловых скоростей

Таблица 1. Сравнение методов определения ориентации объекта

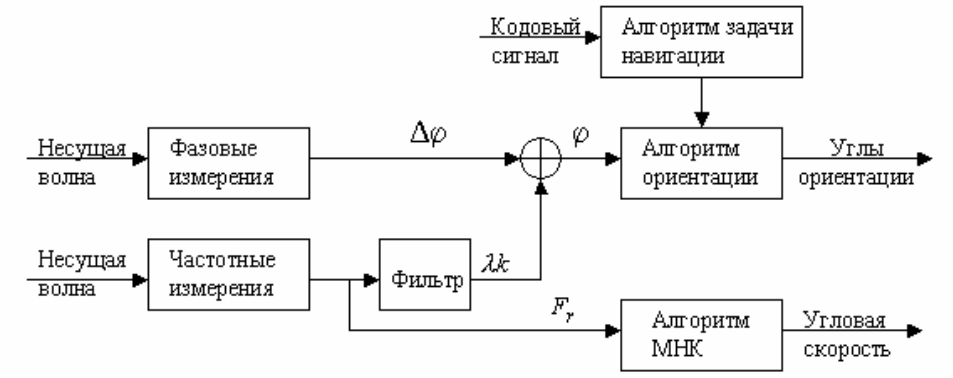


Рис. 3. Совместная обработка фазовых и частотных измерений. Вариант 1

В другом варианте (см. рис. 4) на основе уравнения (9) в географической СК вычисляются компоненты векторов баз всех антенн и угловые скорости объекта. Затем, как отмечалось выше, необходимо полученные компоненты каждого вектора базы спроецировать на орты направлений на соответствующие спутники и выделить в этих проекциях целое число длин волн. Далее задача определения углов ориентации решает-

ся так же, как в предыдущем случае. Можно предположить, что в данной процедуре вычисление компонент векторов баз будет выполнено с большей точностью за счет привлечения большего объема информации, поступающей от всего созвездия спутников, находящихся в зоне видимости объекта.

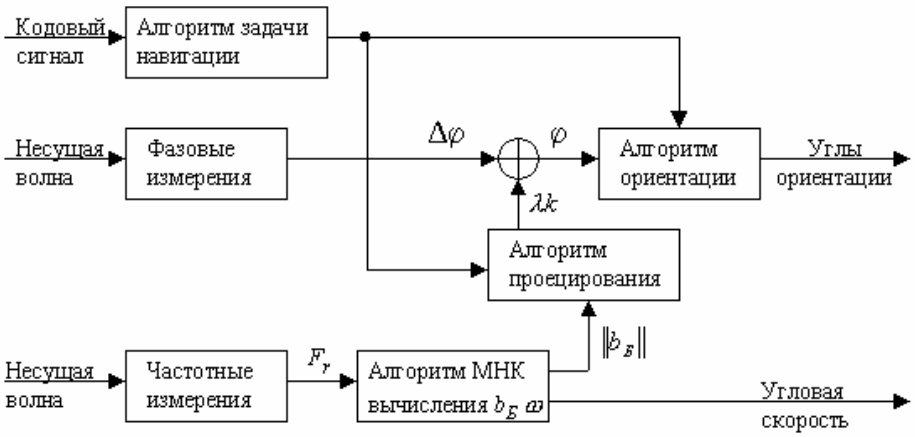


Рис. 4. Рис. 3. Совместная обработка фазовых и частотных измерений. Вариант 2

Заключение

Подводя итоги изложенного, можно сделать следующие выводы.

1. Информационная среда глобальных спутниковых навигационных систем ГЛОНАСС и NAVSTAR содержит координатно-временные, фазовые и частотные параметры необходимые и достаточные для автономного решения задачи определения движения объекта по углам и угловым скоростям.
2. Применение метода векторного согласования обеспечивает вычисление взаимной ориентации связанной и географической систем координат по минимальному составу рабочего созвездия навигационных спутников при наличии на объекте много антенной аппаратуры потребителя, образующей систему из двух неколлинеарных векторов баз, при условии априорно известных длинах баз и ориентации векторов относительно связанной системы координат.
3. Комплексная обработка фазовой и частотной информации, содержащейся в несущих колебаниях, дает возможность оптимизировать определение углов и угловых скоростей для подвижных объектов без ограничений на динамику их углового движения. Разработка оптимальной процедуры требует дальнейших исследований характеристик погрешностей обрабатываемой информации.

Литература

1. Ландау Б.Е. Современные тенденции развития чувствительных элементов инерциальных навигационных систем. // Навигация и управление движением. Сб. докладов I НТК молодых ученых. СПб.: ЦНИИ «Электроприбор», 1999. С. 87–97.
2. Пешехонов В.Г. Проблемы и перспективы современной гироскопии. // Изв. вузов. Приборостроение. 2000. Т. 43. № 1-2. С. 48–56.
3. Интегрированные инерциально-спутниковые системы навигации. Сб. статей и докладов./Под общей ред. акад.РАН В.Г.Пешехонова. СПб.: ГНЦ РФ-ЦНИИ «Электроприбор», 2001. 234 с.

4. Использование системы NAVSTAR для определения угловой ориентации объектов./ В.Н. Абросимов, В.И. Алексеева, Ю.А. Гребенко и др. // Зарубежная радиоэлектроника. 1989. № 1. С. 46-53.
5. Lachapelle G. Attitude Determination. // AGARD Lecture Series 207 - NATO.1996.P.10.
6. Степанов О.А., Кошаев Д.А. Исследование методов решения задачи ориентации с использованием спутниковых систем // Гироскопия и навигация. 1999, № 2. С.30–55.
7. Серегин В.В., Ющенко В.И. Алгоритмы обработки информации, получаемой многоантенной аппаратурой потребителей GPS. // Гироскопия и навигация. 1999. №3. С. 93–100.
8. Липтон А. Выставка инерциальных систем на подвижном основании. М.: Наука, 1971. 167 с.
9. Серегин В.В., Ющенко В.И. Одномоментное разрешение неоднозначности фазовых измерений при определении ориентации подвижного объекта. // Гироскопия и навигация. 2004. №4. С. 88.

НАШИ АВТОРЫ

Акунов Таалайбек Абакирович – кандидат технических наук, докторант кафедры систем управления и информатики

Алексеев Ростислав Александрович – аспирант кафедры систем управления и информатики

Амоскин Игорь Вячеславович – студент

Ахапкин Андрей Станиславович – студент

Белоус Роман Олегович – студент

Бессмертный Игорь Александрович – кандидат технических наук, доцент кафедры вычислительной техники

Блинников Андрей Алексеевич – студент

Блохин Валентин Николаевич – кандидат технических наук, доцент кафедры информатики и прикладной математики

Бобцов Алексей Алексеевич – кандидат технических наук, докторант кафедры систем управления и информатики

Бочков Владимир Евгеньевич – начальник отдела системного и прикладного программного обеспечения ООО «Нефтегазгеодезия», аспирант кафедры информатики и прикладной математики

Будько Михаил Юрьевич – магистрант

Бушуев Александр Борисович – кандидат технических наук, доцент кафедры информатики и прикладной математики

Воскресенский Станислав Игоревич – аспирант кафедры безопасных информационных технологий

Гирик Алексей Валерьевич – магистрант кафедры вычислительной техники

Гришин Михаил Викторович – аспирант кафедры вычислительной техники

Громов Геннадий Юрьевич – старший преподаватель кафедры вычислительной техники

Демин Анатолий Владимирович – доктор технических наук, профессор кафедры информатики и прикладной математики

Дорожкин Антон Константинович – аспирант кафедры вычислительной техники

Дорожкин Сергей Константинович – аспирант кафедры вычислительной техники,

Дорохин Денис Алексеевич – программист Центра информационных систем

Дударенко Наталия Александровна – аспирант кафедры систем управления и информатики

Евстифеев Михаил Илларионович – студент кафедры информационно-навигационных систем

Зотеев Василий Сергеевич – магистрант кафедры вычислительной техники

Зыков Анатолий Геннадьевич – кандидат технических наук, доцент кафедры информатики и прикладной математики

Поляков Владимир Иванович – кандидат технических наук, доцент кафедры вычислительной техники

Иванов Алексей Владимирович – студент кафедры вычислительной техники

Иванов Сергей Евгеньевич – кандидат физ.-мат. наук, ассистент кафедры теоретической и прикладной механики

Изюмов Александр Евгеньевич – магистрант

Катаржнов Александр Демьянович – кандидат технических наук, доцент кафедры безопасных информационных технологий

Кириллов Владимир Васильевич – кандидат технических наук, профессор кафедры вычислительной техники

Коваль Александр Александрович – студент

Ковязин Рустам Раисович – магистр математики, старший программист ООО «ЛМТ», аспирант кафедры вычислительной техники

Кремлев Артем Сергеевич – аспирант кафедры систем управления и информатики

Кривошеев Александр Геннадьевич – кандидат физ.-мат. наук, доцент кафедры теоретической и прикладной механики

Кустарев Павел Валерьевич – кандидат технических наук, доцент кафедры вычислительной техники

Кучер Алексей Владимирович – ассистент кафедры информатики и прикладной математики

Лашкевич Анастасия Евгеньевна – аспирант кафедры информатики и прикладной математики

Мельников Андрей Александрович. – кандидат технических наук, докторант кафедры систем управления и информатики.

Мельников Виталий Геннадиевич – кандидат технических наук, заведующий кафедрой теоретической и прикладной механики

Мельников Геннадий Иванович – доктор физ.-мат. наук, профессор кафедры теоретической и прикладной механики

Мирошник Илья Васильевич – доктор технических наук, профессор кафедры систем управления и информатики

Немолочнов Олег Фомич – доктор технических наук, профессор, заведующий кафедрой информатики и прикладной математики

Нестерук Геннадий Филиппович – кандидат технических наук, доцент кафедры безопасных информационных технологий

Нестерук Леся Геннадиевна – ассистент кафедры информатики Санкт-Петербургского государственного университета экономики и финансов

Нестерук Филипп Геннадьевич – аспирант кафедры безопасных информационных технологий

Николаев Николай Анатольевич – аспирант

Ожиганов Александр Аркадьевич – доктор технических наук, профессор кафедры вычислительной техники

Осипцева Ольга Святославовна – аспирант кафедры систем управления и информатики

Павловская Татьяна Александровна – кандидат технических наук, профессор кафедры информатики и прикладной математики

Проценко Евгений Александрович – аспирант кафедры безопасных информационных технологий

Серегин Валерий Васильевич – доктор технических наук, профессор кафедры информационно-навигационных систем

Сидоров Александр Валерьевич – аспирант кафедры безопасных информационных технологий

Скатин Андрей Владимирович – аспирант кафедры вычислительной техники

Скорубский Владимир Иванович – кандидат технических наук, доцент.

Слита Ольга Валерьевна – аспирант кафедры систем управления и информатики

Сторожевых Сергей Николаевич – аспирант кафедры: вычислительной техники

Сударчиков Сергей Алексеевич – кандидат технических наук, доцент ИКВО

Тимченко Борис Дмитриевич – кандидат технических наук, доцент кафедры: вычислительной техники

Тропченко Александр Ювенальевич – доктор технических наук, профессор кафедры вычислительной техники

Тропченко Андрей Александрович – кандидат технических наук, доцент кафедры: вычислительной техники

Унтилов Александр Алексеевич – сотрудник ГНЦ РФ-ЦНИИ «Электроприбор»

Ушаков Анатолий Владимирович – доктор технических наук, профессор кафедры систем управления и информатики,

Федосов Михаил Михайлович – магистрант.

Харитонова Анастасия Евгеньевна – аспирант кафедры вычислительной техники

Хмылко Федор Вадимович – инженер ООО «МБ-Инфо»

Черемухин Виктор Станиславович– программист Центра информационных систем
Шалаева Марина Борисовна – студентка кафедры вычислительной техники

Шефов Владимир Владимирович – магистрант кафедры вычислительной техники

Ющенко Владимир Ильич – заведующий лабораторией кафедры информационно-навигационных систем

СОДЕРЖАНИЕ

Научно-технический вестник СПбГУ ИТМО. Выпуск 19. Программирование, управление и информационные технологии / Главный редактор д.т.н., проф. В.Н. Васильев. – СПб: СПбГУ ИТМО, 2005. 252 с.

НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК СПбГУ ИТМО
Выпуск 19

Программирование, управление и информационные технологии

Главный редактор
доктор технических наук, профессор
В.Н. Васильев
Дизайн обложки М.А. Петров
Редакционно-издательский отдел СПбГУ ИТМО
Зав. РИО Н.Ф. Гусарова
Лицензия ИД № 00408 от 05.11.99.
Подписано в печать 20.12.05.
Заказ 884. Тираж 100 экз.