

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ
РАДИОЭЛЕКТРОНИКИ

ISSN 1563-0064

РАДИОЭЛЕКТРОНИКА И ИНФОРМАТИКА

Научно-технический журнал

Основан в 1997 г.

№ 2(73), апрель – июнь 2016

Выходит 4 раза в год

© Харьковский национальный
университет радиоэлектроники, 2016

Свидетельство о государственной регистрации КВ № 12097-968 ПР Т4.12.2006

СОДЕРЖАНИЕ

РАДИОТЕХНИКА

ГИБКИНА Н.В., СИДОРОВ М.В., СТАДНИКОВА А.В. ОПТИМАЛЬНОЕ ГРАНИЧНОЕ УПРАВЛЕНИЕ КОЛЕБАНИЯМИ ОДНОРОДНОЙ СТРУНЫ.....	3
---	---

ТЕЛЕКОММУНИКАЦИИ

ТВЕРДОХЛЕБ В.В. СПОСОБ КОНТРОЛЯ БИТОВОЙ СКОРОСТИ ВИДЕОПОТОКА В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ.....	12
---	----

СИСТЕМЫ И ПРОЦЕССЫ УПРАВЛЕНИЯ

НАЙДА В.В., ОКСАНИЧ А.П., ШЕВЧЕНКО И.В., КОНОХ И.С. МОДЕЛИ СИСТЕМЫ ЭМУЛЯЦИИ СЛОЖНОГО ТЕХНОЛОГИЧЕСКОГО ОБЪЕКТА УПРАВЛЕНИЯ.....	17
---	----

КОНЧАКОВСКАЯ О.С., СИДОРОВ М.В. ЧИСЛЕННЫЙ АНАЛИЗ ОДНОГО НЕЛИНЕЙНОГО ЭЛЛИПТИЧЕСКОГО УРАВНЕНИЯ, ВОЗНИКАЮЩЕГО ПРИ МОДЕЛИРОВАНИИ МИКРОЭЛЕКТРОМЕХАНИЧЕСКИХ СИСТЕМ.....	23
---	----

КОМПЬЮТЕРНЫЕ НАУКИ

СИЗОВА Н.Д., ПЕТРОВА О.О., КАМАРДИН А.С. ЭКСПЕРТНА СИСТЕМА «ІНВЕСТИЦІЙНА ПРИВАБЛИВІСТЬ ОБЛАСНИХ ЦЕНТРІВ УКРАЇНИ».....	29
---	----

КОМПЬЮТЕРНАЯ ИНЖЕНЕРИЯ И ТЕХНИЧЕСКАЯ ДИАГНОСТИКА

БАРАННИК В.В., СТАСЕВ С.Ю., ТАРНОПОЛОВ Р.В. МЕТОД ЭФФЕКТИВНОГО КОДИРОВАНИЯ РАЗДЕЛЕННЫХ ПО КОМБИНИРОВАННОЙ СХЕМЕ АЭРОФОТОСНИМКОВ	33
---	----

TAMER BANIAMER, ЕМЕЛЬЯНОВ И.В., ЛЮБАРСКИЙ М., ХАХАНОВ В.И. СИНТЕЗ Q-ТЕСТОВ ПО КУБИТНОМУ ОПИСАНИЮ ФУНКЦИОНАЛЬНОСТЕЙ.....	38
---	----

ОБРИЗАН В.И. ИНФРАСТРУКТУРА ПРОЕКТИРОВАНИЯ SOC ДЛЯ МЕТОДА МУЛЬТИВЕРСНОГО СИНТЕЗА.....	48
---	----

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ГВОЗДИНСКИЙ А.Н., СТОЛЯРЕНКО К.С. ИССЛЕДОВАНИЕ ЗАДАЧ ОПТИМАЛЬНОГО РАСПРЕДЕЛЕНИЯ ДОПУСТИМЫХ НАГРУЗОК НА СЕРВЕРЕ.....	61
---	----

ЧАЙНИКОВ С.И., СОЛОДОВНИКОВ А.С. МОДЕЛИРОВАНИЕ АРХИТЕКТУРЫ И ПОСЛЕДОВАТЕЛЬНОСТИ ВЗАИМОДЕЙСТВИЯ КОМПОНЕНТОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ИНФОРМАЦИОННОЙ СИСТЕМЫ.....	66
---	----

ГВОЗДИНСКИЙ А.Н., ЗАКУТНИЙ С.В. ИССЛЕДОВАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ МЕТОДОВ УПРАВЛЕНИЯ СИСТЕМАМИ ТРАНСПОРТНОГО ТИПА.....	73
--	----

БАРАННИК В.В., МУСИЕНКО А.П., ЖУЙКОВ Д.Б., БРАЙЛОВСКИЙ Н.Н. КЛАСТЕРИЗАЦИЯ БЛОКОВ АЭРОФОТОСНИМКА В ДВУХПРИЗНАКОВОМ СТРУКТУРНОМ ПРОСТРАНСТВЕ НА ОСНОВЕ МЕТОДА К-СРЕДНИХ В СИСТЕМЕ ОБРАБОТКИ ИНФОРМАЦИИ.....	77
---	----

РЕФЕРАТИ.....	81
---------------	----

**ОПТИМАЛЬНОЕ ГРАНИЧНОЕ
УПРАВЛЕНИЕ КОЛЕБАНИЯМИ
ОДНОРОДНОЙ СТРУНЫ**

*ГИБКИНА Н.В., СИДОРОВ М.В.,
СТАДНИКОВА А.В.*

Рассматривается задача оптимального граничного управления колебаниями однородной струны. Под оптимальным управлением в данной задаче понимается задание такого режима колебаний концов струны, при котором в конечный момент времени струна примет положение, наиболее близкое (в смысле среднеквадратической метрики) к желаемому.

Введение

Актуальность исследования. Дифференциальные уравнения в частных производных гиперболического типа описывают широкий класс физических процессов и явлений, в основе которых лежат колебания. С помощью уравнений этого типа можно моделировать поперечные колебания струны, мембраны, продольные колебания стержня, электрические колебания в проводах, акустические и электромагнитные колебания, колебания элементов стационарных конструкций, транспортных, авиационных и космических конструкций, элементов подвижных объектов, некоторые гидродинамические процессы [13].

Простейшее уравнение гиперболического типа – это одномерное волновое уравнение, которое описывает колебания струны. Использование волнового уравнения позволяет исследовать разнообразные прикладные задачи, среди которых одними из наиболее важных являются задачи управления рассматриваемыми процессами. Управляемые колебательные системы широко распространены в промышленности, технике, механике и других областях науки и производства. В частности, такие задачи связаны с гашением пульсаций потоков газа, жидкости, электричества в длинных трубопроводах, генерацией электромагнитных колебаний в волноводах и резонаторах, стабилизацией технических систем различной природы [1, 2].

Теоретическое исследование управляемости систем с распределенными параметрами, которые описываются уравнениями гиперболического типа, было проведено в [1, 6-9, 14].

Для решения этих задач используются принцип максимума [12], метод моментов [1], методы конечного управления [1], метод множителей Лагранжа [8], метод Фурье [3], метод падающих и отраженных волн [5] и другие методы.

РИ, 2016, № 2

Каждая из перечисленных работ содержит серьезные теоретические исследования и значимые результаты, которые могут быть использованы для решения задач оптимального управления колебательными процессами в различных постановках. В то же время основным недостатком большинства указанных методов является громоздкость предложенных схем решения, а иногда отсутствие их вычислительной реализации и примеров решения тестовых задач на ЭВМ.

Таким образом, разработка новых и усовершенствование существующих методов оптимального управления колебательными процессами является актуальной научной проблемой.

Цель и задачи исследования. Цель настоящего исследования – разработка математических методов оптимального управления процессом свободных колебаний однородной струны (без вынуждающей силы), которые позволят установить в конечный момент времени положение точек этой струны, наиболее близкое к заданному положению.

Для достижения поставленной цели необходимо решить следующие задачи:

- сформулировать задачу оптимального управления процессом свободных колебаний однородной струны;
- используя метод Фурье, получить решение одномерного волнового уравнения при заданных краевых и начальных условиях;
- рассмотреть аппроксимацию управляющих воздействий в виде линейной комбинации полиномов Лежандра;
- провести вычислительные эксперименты для разных параметров процесса оптимального управления конечным положением точек струны.

1. Постановка задачи

Исследуются свободные колебания однородной струны $0 \leq x \leq L$ без воздействия на нее вынуждающей силы. Функция $u = u(x, t)$ характеризует отклонение от положения равновесия точки струны с координатой x в момент времени t . Отклонение $u|_{t=0} = \varphi(x)$ и скорость точек струны $\left. \frac{\partial u}{\partial t} \right|_{t=0} = \psi(x)$ в начальный момент времени $t = 0$ считаются известными. Граничный режим для концов струны также известен.

Математическая модель описанного процесса свободных колебаний однородной струны имеет вид:

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < L, \quad t > 0, \quad (1)$$

$$u|_{t=0} = \varphi(x), \quad 0 \leq x \leq L, \quad (2)$$

$$\left. \frac{\partial u}{\partial t} \right|_{t=0} = \psi(x), \quad 0 \leq x \leq L, \quad (3)$$

$$\left(-\alpha_1 \frac{\partial u}{\partial x} + \beta_1 u\right)\Big|_{x=0} = \mu_1(t), \quad t \geq 0, \quad (4)$$

$$\left(\alpha_2 \frac{\partial u}{\partial x} + \beta_2 u\right)\Big|_{x=L} = \mu_2(t), \quad t \geq 0, \quad (5)$$

где a^2 , L , α_1 , α_2 , β_1 , β_2 – заданные константы; $\varphi(x)$, $\psi(x)$ – заданные функции из $L_2(0, L)$.

В уравнении (1) $a^2 = \frac{T}{\rho}$, где ρ – линейная плотность, характеризующая распределение масс в струне (для однородной струны $\rho = \text{const}$); T – натяжение в каждой точке струны ($T = \text{const}$ для всех x и t).

Требуется, управляя положением концов струны с течением времени $t \in [0, T]$, достигнуть к заданному моменту времени $T > 0$ такого положения струны, которое будет как можно более близким к заданному положению $y(x)$, $0 \leq x \leq L$. Формально это условие может быть записано в виде функционала качества:

$$J(\mathbf{u}) = \|u(x, T; \mathbf{u}) - y(x)\|_{L_2(0, L)}^2 = \int_0^L (u(x, T; \mathbf{u}) - y(x))^2 dx \quad (6)$$

при условии, что $u = u(x, t) = u(x, t; \mathbf{u})$ является решением начально-краевой задачи (1)–(5).

Предполагается, что $\mathbf{u} = (\mu_1(t), \mu_2(t))$ – управление, принадлежащее множеству

$$M = \{\mathbf{u} = (\mu_1(t), \mu_2(t)) \in L_2(0, T) \times L_2(0, T), \mu_1^{\min} \leq \mu_1(t) \leq \mu_1^{\max}, \mu_2^{\min} \leq \mu_2(t) \leq \mu_2^{\max}\} \quad (7)$$

почти всюду на $[0, T]$, где $\mu_1^{\min} < \mu_1^{\max}$, $\mu_2^{\min} < \mu_2^{\max}$. Возможны также и другие ограничения, накладываемые на управление \mathbf{u} исходя из физических ограничений задачи.

2. Построение оптимального управления

Первый этап решения задачи оптимального граничного управления процессом колебаний струны заключается в поиске решения задачи (1)–(5) методом Фурье в предположении, что краевые условия заданы. Сделаем замену

$$u(x, t) = w(x, t) + v(x, t), \quad (8)$$

где $v(x, t)$ – новая неизвестная функция, а

$$w(x, t) = \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \mu_2(t) + \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \mu_1(t). \quad (9)$$

Функция $w(x, t)$ выбрана так, чтобы удовлетворять неоднородным краевым условиям (4)–(5).

На основании соотношений (1) – (5) для новой неизвестной функции $v(x, t)$ получим начально-краевую задачу с однородными краевыми условиями:

$$\frac{\partial^2 v}{\partial t^2} = a^2 \frac{\partial^2 v}{\partial x^2} + f(x, t), \quad 0 < x < L, \quad t > 0, \quad (10)$$

$$v|_{t=0} = \tilde{\varphi}(x), \quad 0 \leq x \leq L, \quad (11)$$

$$\frac{\partial v}{\partial t}\Big|_{t=0} = \tilde{\psi}(x), \quad 0 \leq x \leq L \quad (12)$$

$$\left(-\alpha_1 \frac{\partial v}{\partial x} + \beta_1 v\right)\Big|_{x=0} = 0, \quad t \geq 0, \quad (13)$$

$$\left(\alpha_2 \frac{\partial v}{\partial x} + \beta_2 v\right)\Big|_{x=L} = 0, \quad t \geq 0, \quad (14)$$

где

$$f(x, t) = a^2 \left(\frac{2}{2\alpha_2 L + \beta_2 L^2} \mu_2(t) + \frac{2}{2\alpha_1 L + \beta_1 L^2} \mu_1(t) \right) - \left(\frac{x^2}{2\alpha_2 L + \beta_2 L^2} \mu_2''(t) + \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \mu_1''(t) \right), \quad (15)$$

$$\tilde{\varphi}(x) = \varphi(x) - \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \mu_2(0) - \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \mu_1(0), \quad (16)$$

$$\tilde{\psi}(x) = \psi(x) - \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \mu_2'(0) - \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \mu_1'(0). \quad (17)$$

Как известно [10], собственными значениями задачи

$$(10) - (14) \text{ являются } \lambda_n = \left(\frac{\zeta_n}{L}\right)^2, \quad n = 1, 2, \dots, \text{ где } \zeta_n -$$

n -й положительный корень трансцендентного уравнения

$$\text{ctg } \zeta = \frac{\alpha_1 \alpha_2 \zeta^2 - \beta_1 \beta_2 L^2}{(\alpha_1 \beta_2 L + \alpha_2 \beta_1 L) \zeta}, \quad (18)$$

а соответствующие им собственные функции $\Phi_n(x)$ имеют вид:

$$\Phi_n(x) = \sin(\sqrt{\lambda_n} x + \theta_n), \quad (19)$$

$$\text{где } \theta_n = \text{arctg} \frac{\alpha_1 \sqrt{\lambda_n}}{\beta_1}.$$

При этом

$$\|\Phi_n\|_{L_2(0,L)}^2 = \frac{L}{2} \left\{ 1 + \frac{(\alpha_1 \alpha_2 \zeta_n^2 + \beta_1 \beta_2 L^2)(\alpha_1 \beta_2 L + \alpha_2 \beta_1 L)}{(\alpha_1^2 \zeta_n^2 + \beta_1^2 L^2)(\alpha_2^2 \zeta_n^2 + \beta_2^2 L^2)} \right\}.$$

Решение задачи (10) – (14) будем искать в виде ряда:

$$v(x,t) = \sum_{n=1}^{\infty} T_n(t) \Phi_n(x), \quad (20)$$

где $\Phi_n(x)$ имеет вид (19).

Подставив ряд (20) в уравнение (10) и начальные условия (11) – (12), получим, что функции $T_n(t)$, $n = 1, 2, \dots$, являются решением задач Коши:

$$T_n''(t) + \lambda_n a^2 T_n(t) = f_n(t),$$

$$T_n(0) = \tilde{\varphi}_n,$$

$$T_n'(0) = \tilde{\psi}_n,$$

здесь

$$\tilde{\varphi}_n = \frac{(\tilde{\varphi}, \Phi_n)_{L_2(0,L)}}{\|\Phi_n\|_{L_2(0,L)}^2}, \quad (21)$$

$$\tilde{\psi}_n = \frac{(\tilde{\psi}, \Phi_n)_{L_2(0,L)}}{\|\Phi_n\|_{L_2(0,L)}^2}, \quad (22)$$

$$f_n(t) = \frac{(f, \Phi_n)_{L_2(0,L)}}{\|\Phi_n\|_{L_2(0,L)}^2}, \quad (23)$$

и имеют вид:

$$T_n(t) = \tilde{\varphi}_n \cos a\sqrt{\lambda_n} t + \frac{\tilde{\psi}_n}{a\sqrt{\lambda_n}} \sin a\sqrt{\lambda_n} t + \frac{1}{a\sqrt{\lambda_n}} \int_0^t f_n(\tau) \sin a\sqrt{\lambda_n}(t-\tau) d\tau, \quad n = 1, 2, \dots \quad (24)$$

Подставляя в формулу (8) соотношения (9) и (20), где $T_n(t)$ определяется соотношением (24), а $\Phi_n(x)$ – соотношением (19), получаем решение начально-краевой задачи (1) – (5):

$$u(x,t) = \sum_{n=1}^{\infty} \tilde{\varphi}_n \cos a\sqrt{\lambda_n} t \cdot \Phi_n(x) + \sum_{n=1}^{\infty} \frac{\tilde{\psi}_n}{a\sqrt{\lambda_n}} \sin a\sqrt{\lambda_n} t \cdot \Phi_n(x) + \quad (25)$$

$$+ \sum_{n=1}^{\infty} \frac{1}{a\sqrt{\lambda_n}} \int_0^t f_n(\tau) \sin a\sqrt{\lambda_n}(t-\tau) d\tau \cdot \Phi_n(x) + \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \mu_2(t) + \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \mu_1(t).$$

Аппроксимацию функций $\mu_1(t)$, $\mu_2(t)$ будем искать в виде

$$\mu_1(t) = \sum_{k=1}^{m_1} q_k Q_k(t), \quad \mu_2(t) = \sum_{j=1}^{m_2} r_j R_j(t), \quad (26)$$

где $\{Q_k\}$, $\{R_j\}$ – системы базисных функций в $L_2(0,T)$.

В этом случае

$$f(x,t) = a^2 \left(\frac{2}{2\alpha_2 L + \beta_2 L^2} \sum_{j=1}^{m_2} r_j R_j(t) + \frac{2}{2\alpha_1 L + \beta_1 L^2} \sum_{k=1}^{m_1} q_k Q_k(t) \right) - \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \sum_{j=1}^{m_2} r_j R_j''(t) - \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \sum_{k=1}^{m_1} q_k Q_k''(t),$$

$$\tilde{\varphi}(x) = \varphi(x) - \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \sum_{j=1}^{m_2} r_j R_j(0) -$$

$$- \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \sum_{k=1}^{m_1} q_k Q_k(0),$$

$$\tilde{\psi}(x) = \psi(x) - \frac{x^2}{2\alpha_2 L + \beta_2 L^2} \sum_{j=1}^{m_2} r_j R_j'(0) -$$

$$- \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} \sum_{k=1}^{m_1} q_k Q_k'(0).$$

Тогда

$$f_n(t) = \frac{1}{\|\Phi_n\|_{L_2(0,L)}^2} \left(\sum_{j=1}^{m_2} r_j R_j(t) \frac{2a^2}{2\alpha_2 L + \beta_2 L^2} \cdot \eta_n^{(1)} + \sum_{k=1}^{m_1} q_k Q_k(t) \frac{2a^2}{2\alpha_1 L + \beta_1 L^2} \cdot \eta_n^{(1)} - \right. \quad (27)$$

$$\left. - \sum_{j=1}^{m_2} r_j R_j''(t) \frac{1}{2\alpha_2 L + \beta_2 L^2} \cdot \eta_n^{(2)} -$$

$$\left. - \sum_{k=1}^{m_1} q_k Q_k''(t) \frac{1}{2\alpha_1 L + \beta_1 L^2} \cdot \eta_n^{(3)} \right), \quad n = 1, 2, \dots,$$

$$\tilde{\varphi}_n = \frac{1}{\|\Phi_n\|_{L_2(0,L)}^2} \left(\int_0^L \varphi(\xi) \Phi_n(\xi) d\xi - \sum_{j=1}^{m_2} r_j \frac{R_j(0)}{2\alpha_2 L + \beta_2 L^2} \cdot \eta_n^{(2)} - \sum_{k=1}^{m_1} q_k \frac{Q_k(0)}{2\alpha_1 L + \beta_1 L^2} \cdot \eta_n^{(3)} \right), \quad n = 1, 2, \dots, \quad (28)$$

$$\tilde{\psi}_n = \frac{1}{\|\Phi_n\|_{L_2(0,L)}^2} \left(\int_0^L \psi(\xi) \Phi_n(\xi) d\xi - \sum_{j=1}^{m_2} r_j \frac{R'_j(0)}{2\alpha_2 L + \beta_2 L^2} \cdot \eta_n^{(2)} - \sum_{k=1}^{m_1} q_k \frac{Q'_k(0)}{2\alpha_1 L + \beta_1 L^2} \cdot \eta_n^{(3)} \right), \quad n = 1, 2, \dots, \quad (29)$$

где

$$\eta_n^{(1)} = \int_0^L \Phi_n(x) dx, \quad \eta_n^{(2)} = \int_0^L x^2 \Phi_n(x) dx, \quad \eta_n^{(3)} = \int_0^L (x-L)^2 \Phi_n(x) dx.$$

Подставив (27) – (29) в (25) и вычислив полученную функцию при $t = T$, получим, что фактическое положение точек струны в конечный момент времени описывается соотношением

$$u(x, T) = \sum_{k=1}^{m_1} q_k A_k(x) + \sum_{j=1}^{m_2} r_j B_j(x) + C(x), \quad (30)$$

где

$$A_k(x) = \sum_{n=1}^{\infty} \frac{\Phi_n(x)}{\|\Phi_n\|_{L_2(0,L)}^2} \times \frac{1}{2\alpha_1 L + \beta_1 L^2} \left[-Q_k(0) \cos a\sqrt{\lambda_n} T \cdot \eta_n^{(3)} + -Q'_k(0) \frac{\sin a\sqrt{\lambda_n} T}{a\sqrt{\lambda_n}} \cdot \eta_n^{(3)} + \frac{1}{a\sqrt{\lambda_n}} \left(2a^2 \eta_n^{(1)} \int_0^T Q_k(\tau) \sin[a\sqrt{\lambda_n}(T-\tau)] d\tau - \eta_n^{(2)} \int_0^T Q'_k(\tau) \sin[a\sqrt{\lambda_n}(T-\tau)] d\tau \right) + \frac{(x-L)^2}{2\alpha_1 L + \beta_1 L^2} Q_k(T), \quad k = \overline{1, m_1}, \right. \\ \left. B_k(x) = \sum_{n=1}^{\infty} \frac{\Phi_n(x)}{\|\Phi_n\|_{L_2(0,L)}^2} \times \right.$$

$$\times \frac{1}{2\alpha_2 L + \beta_2 L^2} \left[-R_k(0) \cos a\sqrt{\lambda_n} T \cdot \eta_n^{(2)} + -R'_k(0) \frac{\sin a\sqrt{\lambda_n} T}{a\sqrt{\lambda_n}} \cdot \eta_n^{(2)} + \frac{1}{a\sqrt{\lambda_n}} \left(2a^2 \eta_n^{(1)} \int_0^T R_k(\tau) \sin[a\sqrt{\lambda_n}(T-\tau)] d\tau - \eta_n^{(2)} \int_0^T R'_k(\tau) \sin[a\sqrt{\lambda_n}(T-\tau)] d\tau \right) + \frac{x^2}{2\alpha_2 L + \beta_2 L^2} R_k(T), \quad k = \overline{1, m_1}, \right. \quad (32)$$

$$\left. C(x) = \sum_{n=1}^{\infty} \frac{\Phi_n(x)}{\|\Phi_n\|_{L_2(0,L)}^2} \times \left(\cos(a\sqrt{\lambda_n} T) \int_0^L \varphi(\xi) \Phi_n(\xi) d\xi + \frac{\sin a\sqrt{\lambda_n} T}{a\sqrt{\lambda_n}} \int_0^L \psi(\xi) \Phi_n(\xi) d\xi \right). \quad (33)$$

Таким образом, задача оптимального управления колебаниями концов однородной струны (1) – (7) сводится к задаче оптимизации:

$$J(\mu) = \int_0^L \left(\sum_{k=1}^{m_1} q_k A_k(x) + \sum_{j=1}^{m_2} r_j B_j(x) + C(x) - y(x) \right)^2 dx = \sum_{k=1}^{m_1} q_k^2 \delta_k^{(1)} + \sum_{j=1}^{m_2} r_j^2 \delta_j^{(2)} + 2 \sum_{k=1}^{m_1} \sum_{j=1}^{m_2} q_k r_j \gamma_{kj} + + 2 \sum_{k=1}^{m_1} q_k \sigma_k^{(1)} + 2 \sum_{j=1}^{m_2} r_j \sigma_j^{(2)} + \eta \rightarrow \min_{\substack{q_k, k=\overline{1, m_1}, \\ r_j, j=\overline{1, m_2}}} \quad (34)$$

где

$$\gamma_{kj} = \int_0^L A_k(x) B_j(x) dx, \quad k = \overline{1, m_1}, \quad j = \overline{1, m_2}, \\ \delta_k^{(1)} = \int_0^L A_k^2(x) dx, \quad k = \overline{1, m_1}, \\ \delta_j^{(2)} = \int_0^L B_j^2(x) dx, \quad j = \overline{1, m_2}, \\ \sigma_k^{(1)} = \int_0^L A_k(x) (C(x) - y(x)) dx, \quad k = \overline{1, m_1}, \\ \sigma_j^{(2)} = \int_0^L B_j(x) (C(x) - y(x)) dx, \quad j = \overline{1, m_2}, \\ \eta = \int_0^L (C(x) - y(x))^2 dx.$$

Задачу оптимизации (34) нужно дополнить ограничениями на управление (7) или другими.

3. Вычислительный эксперимент

Для проведения вычислительных экспериментов в задаче (1)–(7) при разных значениях m были выбраны следующие значения параметров: $L = 1$, $a = 1$, $T = 1$. Положение и скорость точек струны в начальный момент времени $t = 0$ равны $\varphi(x) = 0$ и $\psi(x) = 0$ соответственно. Считаем, что управление процессом колебаний осуществляется только за счет колебаний правого конца струны, поэтому функция $\mu_1(t)$ задана, и для нахождения оптимального управления μ необходимо определить функцию $\mu_2(t)$. Во всех следующих вычислительных экспериментах считаем, что функция $\mu_1(t)$ известна: $\mu_1(t) = 0$.

Аппроксимацию управления $\mu_2(t)$ будем искать в виде линейной комбинации смещенных полиномов Лежандра:

$$\mu_2^{(m)}(t) = \sum_{k=0}^m r_k P_k\left(\frac{2x}{L} - 1\right). \quad (35)$$

Здесь $P_k(t)$ – полином Лежандра степени k . Эти полиномы образуют полную ортогональную в $L_2(-1,1)$ систему полиномов и могут быть вычислены по формуле Родрига [10]:

$$P_k(t) = \frac{1}{2^k k!} \cdot \frac{d^k}{dx^k} [(x^2 - 1)^k].$$

На управление $\mu_2(t)$ накладывается следующее ограничение (условие согласования краевого и начального условий): $\mu_2(0) = 0$.

Случай 1. Пусть левый конец струны $x=0$ закреплен, а управление заключается в задании при $t \in (0, T]$ режима колебаний правого конца струны $x = L$. Этот случай соответствует следующим значениям параметров в краевых условиях (4)–(5): $\alpha_1 = 0$, $\beta_1 = 1$, $\alpha_2 = 0$, $\beta_2 = 1$. При этом собственные значения

$$\lambda_n = \left(\frac{\pi n}{L}\right)^2, \quad n = 1, 2, \dots, \text{ а собственные функции}$$

$$\Phi_n(x) = \sin \frac{\pi n x}{L}, \quad n = 1, 2, \dots$$

Зададим желаемое положение струны в момент времени T функцией $y(x) = x^3(1-x)$.

На рис. 1 – 4 приведены графики аппроксимаций $\mu_2^{(m)}(t)$ оптимального управления режимом колебаний правого конца струны $\mu_2(t)$ вида (35) для $m = 3, 6, 9, 12$.

Как видно из графиков, прослеживается тенденция выхода функций $\mu_2^{(m)}(t)$ на некоторое предельное

значение с ростом m . Так, в ходе вычислений получено, что $\mu_2^{(12)}(t)$ совпадает с $\mu_2^{(11)}(t)$ в норме $L_2(0, T)$ с точностью $0,15 \cdot 10^{-2}$, а значения соответствующих им функционалов отличаются на величину порядка $0,23 \cdot 10^{-18}$. Функцию $\mu_2^{(12)}(t)$ возьмем в качестве приближенного значения оптимального управления процессом колебаний правого конца струны.

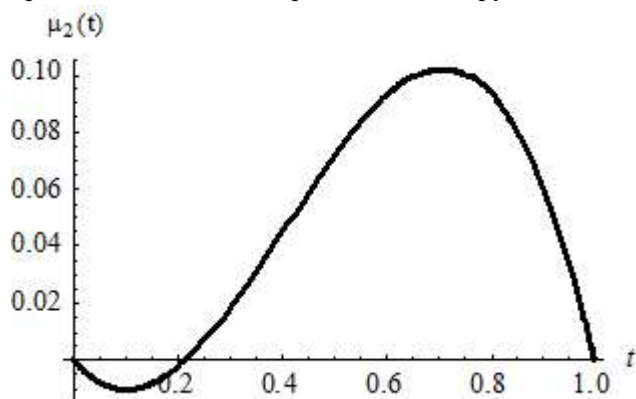


Рис. 1. График аппроксимации $\mu_2^{(3)}(t)$ оптимального управления $\mu_2(t)$

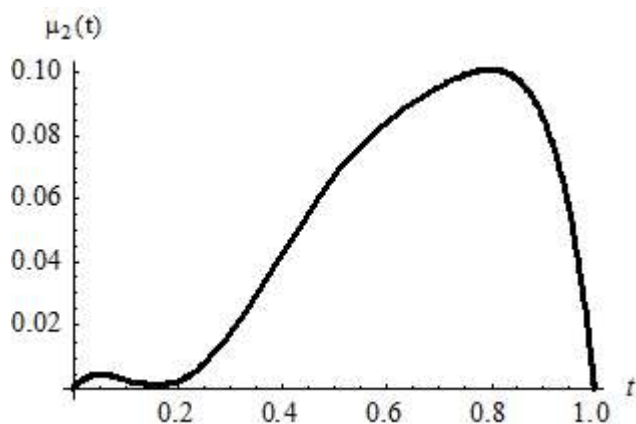


Рис. 2. График аппроксимации $\mu_2^{(6)}(t)$ оптимального управления $\mu_2(t)$

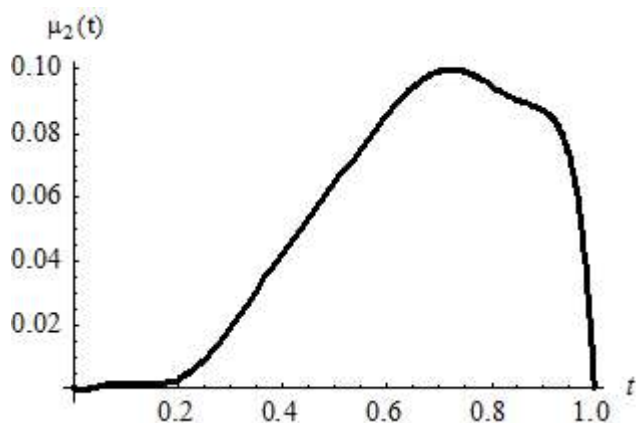


Рис. 3. График аппроксимации $\mu_2^{(9)}(t)$ оптимального управления $\mu_2(t)$

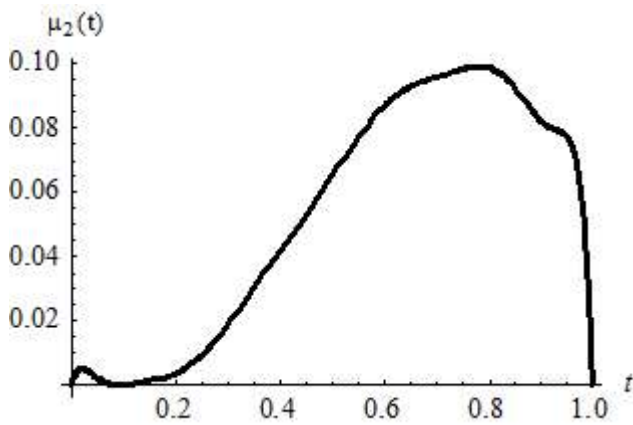


Рис. 4. График аппроксимации $\mu_2^{(12)}(t)$ оптимального управления $\mu_2(t)$

На рис. 5 приведено желаемое $y(x)$ и фактическое $u(x, T)$ положение струны в конечный момент времени $T = 1$ под воздействием управления $\mu_2^{(12)}(t)$, а на рис. 6 – соответствующий этому случаю модуль разности $|u(x, T) - y(x)|$ при $T = 1$. При этом

$$\|u(x, T) - y(x)\|_{L_2(0, L)} = 0,51 \cdot 10^{-2}.$$

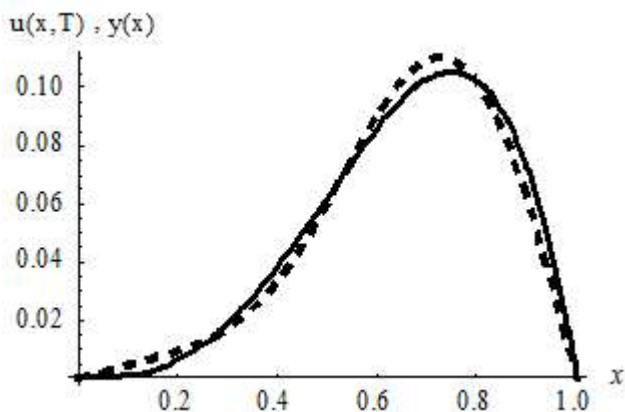


Рис. 5. График желаемого $y(x)$ (сплошная линия) и фактического $u(x, T)$ (пунктирная линия) положения точек струны при управлении $\mu_2^{(12)}(t)$ в момент времени $T = 1$

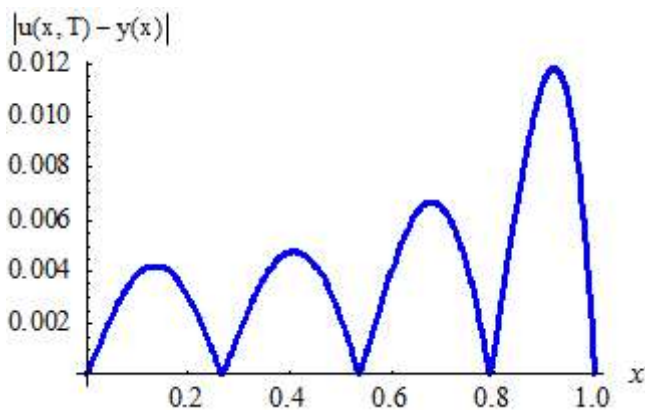


Рис. 6. График $|u(x, T) - y(x)|$

Случай 2. Пусть левый конец струны $x = 0$ закреплен, а управление заключается в задании при $t \in (0, T]$ приложенной к правому концу струны $x = L$ силы. Этот случай соответствует следующим значениям параметров в краевых условиях (4) – (5): $\alpha_1 = 0$, $\beta_1 = 1$, $\alpha_2 = 1$, $\beta_2 = 0$. При этом собственные значения $\lambda_n = \left(\frac{\pi(2n+1)}{2L}\right)^2$, $n = 0, 1, \dots$, а собственные функции $\Phi_n(x) = \sin \frac{\pi(2n+1)x}{2L}$, $n = 0, 1, \dots$

Зададим желаемое положение струны в момент времени T функцией $y(x) = x(2-x)$.

На рис. 7 – 10 приведены графики аппроксимаций $\mu_2^{(m)}(t)$ оптимального управления режимом колебаний правого конца струны $\mu_2(t)$ вида (35) для $m = 3, 6, 9, 12$. Как и в предыдущем случае, из графиков видно, что с ростом m прослеживается тенденция выхода функций $\mu_2^{(m)}(t)$ на некоторое предельное значение. При этом $\mu_2^{(12)}(t)$ совпадает с $\mu_2^{(11)}(t)$ в норме $L_2(0, T)$ с точностью $0,53 \cdot 10^{-1}$, а значения соответствующих им функционалов отличаются на величину порядка $0,19 \cdot 10^{-15}$. С указанной точностью функция $\mu_2^{(12)}(t)$ может быть использована в качестве приближенного значения оптимального управления процессом колебаний правого конца струны.

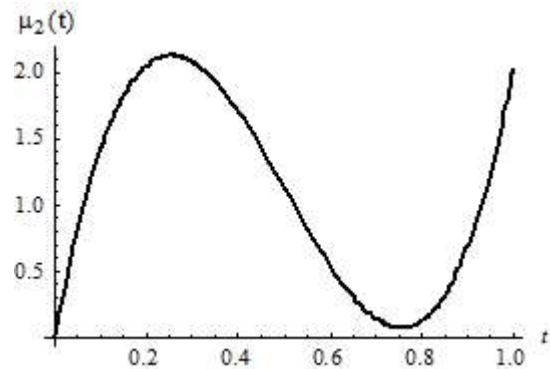


Рис. 7. График аппроксимации $\mu_2^{(3)}(t)$ оптимального управления $\mu_2(t)$

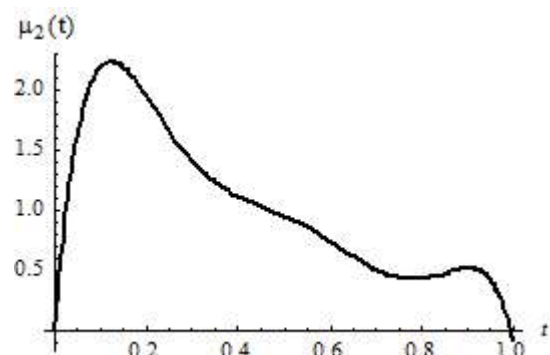


Рис. 8. График аппроксимации $\mu_2^{(6)}(t)$ оптимального управления $\mu_2(t)$

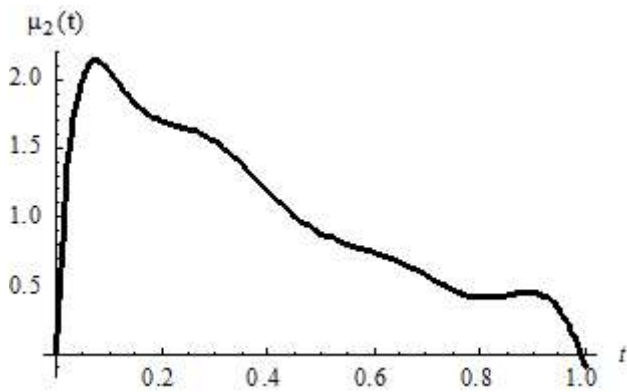


Рис. 9. График аппроксимации $\mu_2^{(9)}(t)$ оптимального управления $\mu_2(t)$

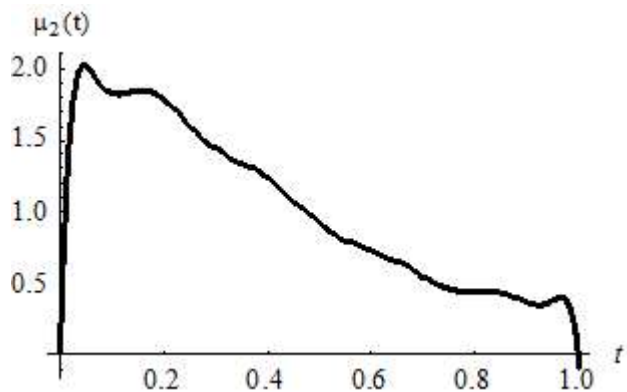


Рис. 10. График аппроксимации $\mu_2^{(12)}(t)$ оптимального управления $\mu_2(t)$

На рис. 11 приведено желаемое $y(x)$ и фактическое $u(x, T)$ положение струны в конечный момент времени $T=1$ под воздействием управления $\mu_2^{(12)}(t)$, а на рис. 12 – соответствующий этому случаю график модуля разности $|u(x, T) - y(x)|$ при $T=1$. При этом

$$\|u(x, T) - y(x)\|_{L_2(0, L)} = 0,20 \cdot 10^{-2}.$$

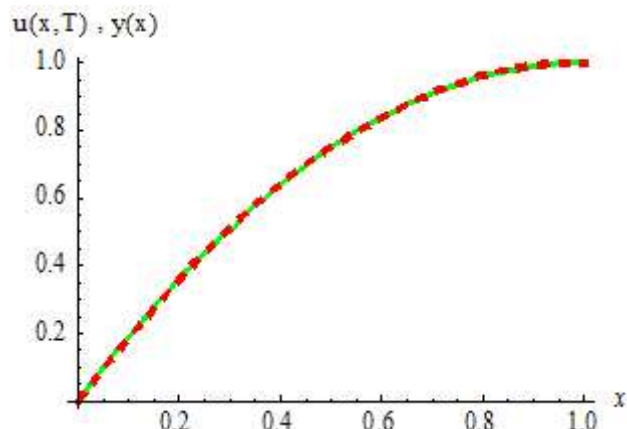


Рис. 11. График желаемого $y(x)$ (сплошная линия) и фактического $u(x, T)$ (пунктирная линия) положения точек струны при управлении $\mu_2^{(12)}(t)$ в момент времени $T=1$

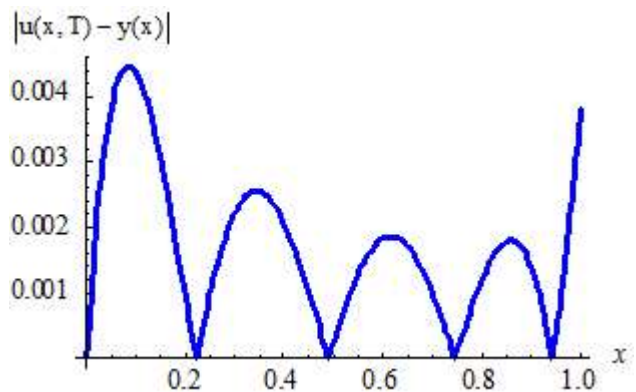


Рис. 12. График $|u(x, T) - y(x)|$

Случай 3. Считаем, что левый конец струны $x=0$ упруго закреплен, а управление заключается в задании при $t \in (0, T]$ режима колебаний правого конца струны $x=L$. Этот случай соответствует следующим значениям параметров в краевых условиях (4)-(5): $\alpha_1=1, \beta_1=1, \alpha_2=0, \beta_2=1$. При этом собственное значение $\lambda_n, n=1, 2, \dots$, определяется как n -й положительный корень уравнения $\operatorname{tg} \sqrt{\lambda} L = -\frac{\alpha_1 \sqrt{\lambda}}{\beta_1}$, а собственные функции есть $\Phi_n(x) = \sin \sqrt{\lambda_n} (x-L), n=1, 2, \dots$.

Зададим желаемое положение струны в момент времени T функцией $y(x) = x^2 - \frac{x}{2} - \frac{1}{2}$.

На рис. 13 – 16 приведены графики аппроксимаций $\mu_2^{(m)}(t)$ оптимального управления режимом колебаний правого конца струны $\mu_2(t)$ вида (35) для $m=3, 6, 9, 12$.

В этом случае также видно, что с ростом m наблюдается выход функций $\mu_2^{(m)}(t)$ на некоторое предельное значение. При этом $\mu_2^{(12)}(t)$ совпадает с $\mu_2^{(11)}(t)$ в норме $L_2(0, T)$ с точностью $0,15 \cdot 10^{-1}$, а значения соответствующих им функционалов отличаются на величину порядка $0,11 \cdot 10^{-12}$. Как и в предыдущем случае, функцию $\mu_2^{(12)}(t)$ с указанной точностью возьмем в качестве приближенного значения оптимального управления процессом колебаний правого конца струны.

На рис. 17 приведено желаемое $y(x)$ и фактическое $u(x, T)$ положение струны в конечный момент времени $T=1$ под воздействием управления $\mu_2^{(12)}(t)$, а на рис. 18 – соответствующий этому случаю график модуля разности $|u(x, T) - y(x)|$ при $T=1$. При этом

$$\|u(x, T) - y(x)\|_{L_2(0, L)} = 0,23 \cdot 10^{-2}.$$

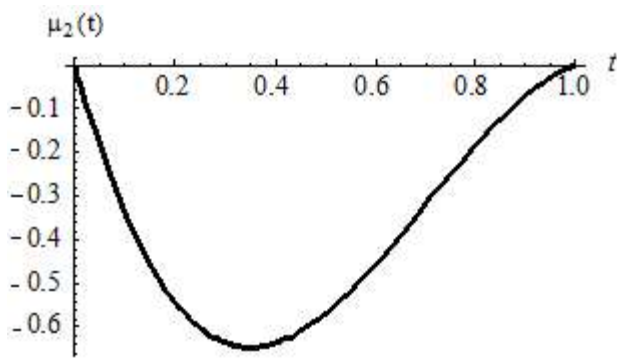


Рис. 13. График аппроксимации $\mu_2^{(3)}(t)$ оптимального управления $\mu_2(t)$

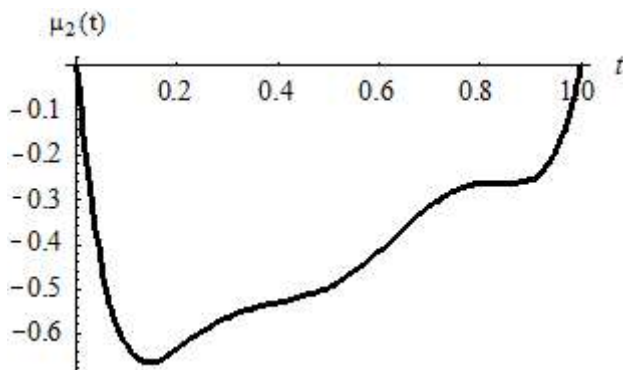


Рис. 14. График аппроксимации $\mu_2^{(6)}(t)$ оптимального управления $\mu_2(t)$

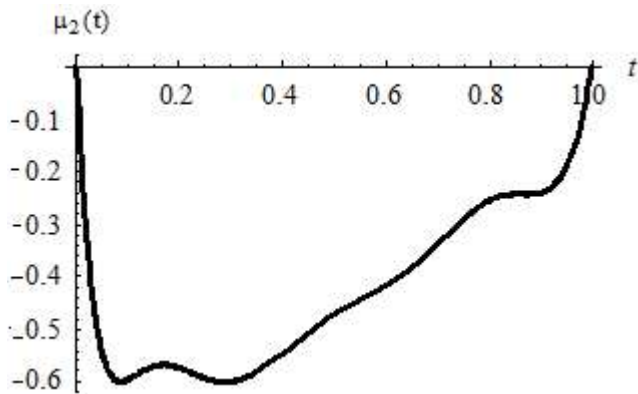


Рис. 15. График аппроксимации $\mu_2^{(9)}(t)$ оптимального управления $\mu_2(t)$

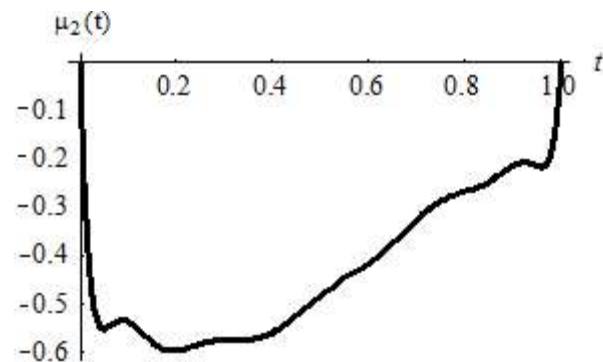


Рис. 16. График аппроксимации $\mu_2^{(12)}(t)$ оптимального управления $\mu_2(t)$

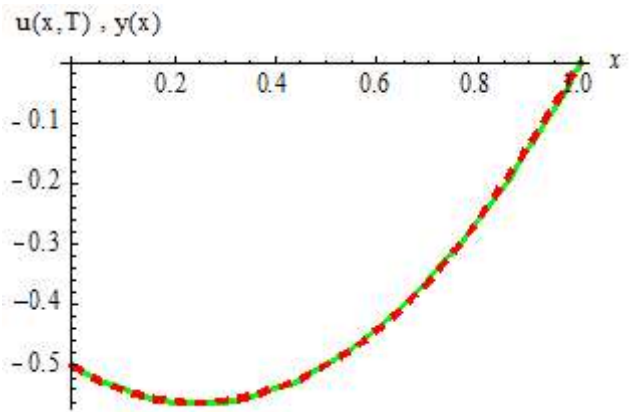


Рис. 17. График желаемого $y(x)$ (сплошная линия) и фактического $u(x, T)$ (пунктирная линия) положения точек струны при управлении $\mu_2^{(12)}(t)$ в момент времени $T=1$

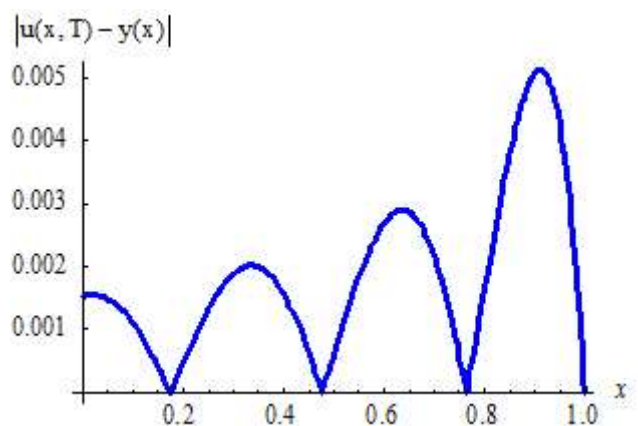


Рис. 18. График $|u(x, T) - y(x)|$

Отметим, что кроме выражения (35) можно использовать также другие виды аппроксимаций управляющего воздействия $\mu_2(t)$, например, в виде отрезка тригонометрического ряда Фурье, линейной комбинации сплайнов Шенберга [4].

Выводы

Предложен метод приближенного построения оптимального управления процессом колебаний однородной струны за счет задания режима колебаний ее концов. Вычислительные эксперименты проведены для различных режимов колебаний правого конца струны в случае, когда управляющие функции аппроксимируются полиномами Лежандра. Вычислительный эксперимент показал, что с ростом степени аппроксимирующего полинома структура оптимального управления режимом колебаний $\mu_2(t)$ выходит на некоторый предельный режим, свой для каждого типа краевых условий. Предложенный метод отличается от известных методов тем, что начально-краевая задача для волнового уравнения решается аналитически и оптимальное управление также ищется в аналитическом виде. Полученные результаты могут быть использованы при расчете оптимальных программ управления режимом колебаний в физических и технологи-

ческих системах. Этим определяется научная новизна и практическая значимость полученных результатов.

Литература: 1. *Бутковский А.Г.* Методы управления системами с распределенными параметрами. М.: Наука, 1975. 588 с. 2. *Бутковский А.Г.* Теория оптимального управления системами с распределенными параметрами. М.: Наука, 1965. 474 с. 3. *Васильев Ф.П., Куржанский М.А., Разгулин А.В.* О методе Фурье для решения одной задачи управления колебанием струны // Вестник МГУ, сер. 15, вычисл. матем. и киберн. 1993. № 2. С. 3-8. 4. *Гибкина Н.В., Подусов Д.Ю., Сидоров М.В.* Оптимальное управление конечным температурным состоянием однородного стержня // Радиоэлектроника и информатика, 2014. №2. С. 9 – 15. 5. *Егоров А.И.* Управление упругими колебаниями // ДАН УССР, сер. физ.-мат. и техн. наук. 1986. № 5. С. 60 – 63. 6. *Ильин В.А., Моисеев Е.И.* Оптимизация граничных управлений колебаниями струны // УМН. 2005. Т. 60. Вып. 6 (366). С. 89 – 114. 7. *Комков В.* Теория оптимального управления демпфированием колебаний простых упругих систем. М.: Мир, 1975. 162 с. 8. *Конец М.М.* Оптимальное управление колебаниями прямоугольной мембраны // Кибернетика и вычисл. техника. 2014. Вып. 177. С. 28 – 42. 9. *Лионс Ж.-Л.* Оптимальное управление системами, описываемыми уравнениями с частными производными. М.: Мир, 1972. 416 с. 10. *Мартинсон Л.К., Малов Ю.И.* Дифференциальные уравнения математической физики. 2-е изд. М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. 368 с. 11. *Свешников А.Г., Боголюбов А.Н., Кравцов В.В.* Лекции по математической физике. М.: Наука, 2004. 416 с. 12. *Сиразетдинов Т.К.* Оптимизация систем с распределенными параметрами. М.: Наука, 1977. 480 с. 13. *Тихонов А.Н., Самарский А.А.* Уравнения математической физики. М.: Наука, 2004. 798 с. 14. *Фурсиков А.В.* Оптимальное управление распределенными системами. Теория и приложения. Новосибирск: Научная книга, 1999. 352 с.

Transliterated bibliography:

1. *Butkovskiy A.G.* Metody upravleniya sistemami s raspredelennymi parametrami. M.: Nauka, 1975. 588 s.
2. *Butkovskiy A.G.* Teoriya optimalnogo upravleniya sistemami s raspredelennymi parametrami. M.: Nauka, 1965. 474 s.
3. *Vasilev F.P., Kurzhanskiy M.A., Razgulin A.V.* O metode Fure dlya resheniya odnoy zadachi upravleniya kolebaniem strunyi // Vestnik MGU, ser. 15, vyichisl. matem. i kibern. 1993. #2. S. 3-8.
4. *Gibkina N.V., Podusov D.Yu., Sidorov M.V.* Optimalnoe upravlenie konechnym temperaturnym sostoyaniem odnorodnogo sterzhnya // Radioelektronika i informatika, 2014. #2. S. 9 – 15.

5. *Egorov A.I.* Upravlenie uprugimi kolebaniyami // DAN USSR, ser. fiz.-mat. i tehn. nauk. 1986. #5. S. 60 – 63. 6. *Ilin V.A., Moiseev E.I.* Optimizatsiya granichnykh upravleniy kolebaniyami strunyi // UMN. 2005. T. 60. Vyip. 6 (366). S. 89 – 114.

7. *Komkov V.* Teoriya optimalnogo upravleniya dempfirovaniem kolebaniy prostykh uprugih sistem. M.: Mir, 1975. 162 s.

8. *Kopets M.M.* Optimalnoe upravlenie kolebaniyami pryamougolnoy membrany // Kibernetika i vyichisl. tehnika. 2014. Vyip. 177. S. 28 – 42.

9. *Lions Zh.-L.* Optimalnoe upravlenie sistemami, opisyyivaemyimi uravneniyami s chastnyimi proizvodnyimi. M.: Mir, 1972. 416 s.

10. *Martinson L.K., Malov Yu.I.* Differentsialnyie uravneniya matematicheskoy fiziki. 2-e izd. M.: Izd-vo MG TU im. N.E. Baumana, 2002. 368 s.

11. *Sveshnikov A.G., Bogolyubov A.N., Kravtsov V.V.* Lektsii po matematicheskoy fizike. M.: Nauka, 2004. 416 s.

12. *Sirazetdinov T.K.* Optimizatsiya sistem s raspredelennymi parametrami. M.: Nauka, 1977. 480 s.

13. *Tihonov A.N., Samarskiy A.A.* Uravneniya matematicheskoy fiziki. M.: Nauka, 2004. 798 s.

14. *Fursikov A.V.* Optimalnoe upravlenie raspredelennymi sistemami. Teoriya i prilozheniya. Novosibirsk: Nauchnaya kniga, 1999. 352 s.

Поступила в редколлегию 12.04.2016

Рецензент: д-р физ.-мат. наук, проф. Колосов А.И.

Гибкина Надежда Валентиновна (N.V. Gybkina), канд. техн. наук, доцент каф. прикладной математики ХНУРЭ. Научные интересы: математическое моделирование, оптимальное управление и его приложения, математическая физика, актуарная и финансовая математика. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. (057) 7021436.

Сидоров Максим Викторович (M.V. Sidorov), канд. физ.-мат. наук, доцент каф. прикладной математики ХНУРЭ. Научные интересы: математическое моделирование, численные методы, математическая физика, теория R-функций и её приложения, стохастический анализ и его приложения. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. (057) 7021436.

Стадникова Анна Викторовна (H.V. Stadnikova), ассист. каф. прикладной математики ХНУРЭ. Научные интересы: математическое моделирование, численные методы, теория R-функций и её приложения. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. (057) 7021436.



СПОСОБ КОНТРОЛЯ БИТОВОЙ СКОРОСТИ ВИДЕОПОТОКА В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

ТВЕРДОХЛЕБ В.В.

Рассматриваются принципы построения управляющих алгоритмов для контроля битовой скорости видеопотока. Показываются эффективность подходов, основанных на трехмерном представлении трансформант кадра, позволяющих осуществлять контроль скорости с одновременным учетом итоговых значений ошибки. Приводятся методики повышения быстродействия управляющих алгоритмов.

Ключевые слова: насыщенность трансформанты, управление битовой скоростью, битовая плоскость, СКО, пропускная способность.

Keywords: saturation of transformant, bitrate control, bitplane, MSE, bandwidth.

1. Актуальность управления интенсивностью видеопотока

Современные инфокоммуникации характеризуются стремительным ростом объема передаваемых видеоданных. Происходит постоянный рост числа пользователей систем видеоконференций и сервисов трансляции потокового видео. На фоне этого, однако, увеличение пропускной способности каналов запаздывает, что является причиной частых перегрузок сетей.

В таких условиях возможность адаптируемости интенсивности видеопотока к пропускной способности канала является актуальной [1, 2].

Данная возможность способна обеспечить эффективную передачу видеопотока, предотвратить возникновение потерь и задержек передачи видеоданных.

Целью данного исследования является построение методики управления битовой скоростью видеопотока для согласования ее величины с пропускной способностью канала инфокоммуникационной сети.

Основными задачами построения метода управления контроля битовой скоростью являются: определение условий эффективной передачи видеопотока на фоне изменяющейся пропускной способности канала, построение механизма управления битовой скоростью видеопотока, а также контроля уровня ошибки [3].

2. Условия эффективной передачи видеопотока

Эффективной можно считать такую передачу видеопотока, при которой обеспечивается выполнение следующих условий:

- соответствие требованиям QoS касательно величин задержки и потерь данных;
- поддержание уровня ошибки, не превышающего заданного значения;
- обеспечение визуально приемлимого качества видео на приеме.

Таким образом, наряду с управлением битовой скоростью необходимо также обеспечить значение ошибки, в качестве которой будем рассматривать уровень среднеквадратического отклонения, на требуемом уровне [4].

Тогда условия эффективной передачи видеопотока при изменяющейся пропускной способности канала могут быть представлены следующим образом:

$$\begin{cases} R \rightarrow \min ; \\ d \leq d_{\min} . \end{cases} \quad (1)$$

3. Суть метода контроля битовой скорости

Исходный видеоквадрат F , после выполнения ДКП и преобразования цветовой модели RGB в модель YCbCr, рассматривается как множество P трансформант Y_p .

В свою очередь, каждая трансформанта Y_p представлена совокупностью $(h; w)$ -х компонент $Y_p = \|y(p)_{hw}\|$.

Каждая компонента $y(p)_{hw}$ трансформанты Y_p представлена в двоичном виде [1], на основе последовательности $\alpha(p)_{hw}^{(\mu)}$ бит (рисунок). Это эквивалентно преобразованию:

$$\|y(p)_{hw}\| \rightarrow \left\| \left\langle \alpha(p)_{hw}^{(\mu)}, \alpha(p)_{hw}^{(\mu-1)} \dots \alpha(p)_{hw}^{(0)} \right\rangle^T \right\|, \quad (2)$$

$$\alpha(p)_{hw}^{(\mu)} \in \{0, 1\}, \quad h = \overline{0, 7}; \quad w = \overline{0, 7}; \quad \mu = \overline{7, 0},$$

где $\alpha(p)_{hw}^{(\mu)}$ – бит двоичного разложения (h, w) -й компоненты p -й трансформанты.

Множество всех бит μ -го разряда p -й трансформанты составляет битовую плоскость $Y(p)_\mu$.

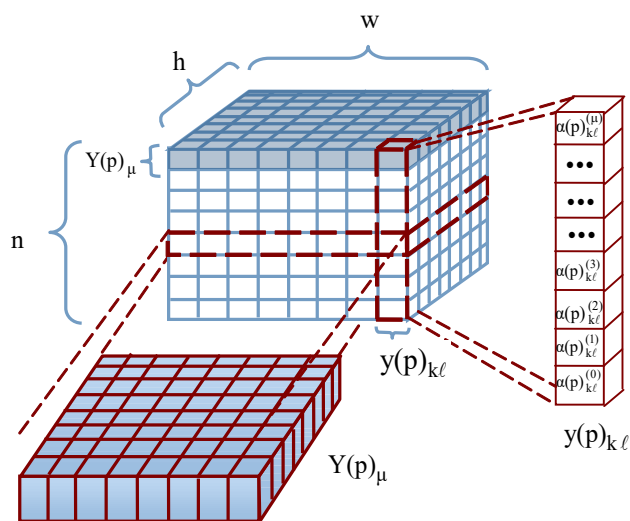
В свою очередь, совокупность двоичных представлений всех элементов матрицы Y_p составляет битовый куб $Y_p^{(3d)}$, пример которого представлен на рисунке.

При рассматриваемом способе организации данных верхний слой этого куба образуют старшие биты $\alpha(p)_{hw}^{(\mu)}$ двоичного представления.

Представление трансформанты Y_p в трехмерном пространстве позволяет осуществлять передачу данных отдельными битовыми плоскостями, аналогично подходу, который используется методом последовательного приближения технологии Progressive JPEG [2, 5].

В этом случае появляется возможность контролировать объем передаваемой информации в зависимости от требований пропускной способности B_w канала.

В зависимости от требуемого объема бит для представления кадра используются либо все n битовых плоскостей $Y(p)_\mu$ трансформанты Y_p , либо только $(n-\mu)$ битовых плоскостей, чтобы обеспечить битовую скорость R_F кадра на уровне, не превышающем требуемое значение.



Представление трансформанты Y_p в виде битового куба $Y_p^{(3d)}$

Таким образом, указанный способ представления данных обеспечивает возможность управления компрессией и может быть использован в качестве базового.

Принцип работы метода управления на базе технологии Progressive JPEG может быть описан следующим образом.

В начальный момент времени t_0 буферное устройство отправляет в канал тестовый пакет R_{start} с известной величиной.

Используя значение времени двусторонней задержки RTT , определяется величина полосы пропускания в момент t_0 :

$$c(t_0) = c RTT, \quad (3)$$

Используя значение $c(t_0)$, также вычисляется фактическое число кадров, которое необходимо поместить в буфер:

$$c(t_0) = c RTT. \quad (4)$$

При полученном значении $c(t_0)$ для соответствующего количества кадров происходит оценка битовых скоростей $R_{p\mu}$ и величины ошибки $d_{p\mu}$ по всем битовым плоскостям каждой из трансформант кадра.

На этапе, предшествующем нахождению $R_{p\mu}$ и $d_{p\mu}$ битовых плоскостей трансформант, выполняется оценка насыщенности НЧ-областей трансформант [3], как показано в следующей формуле:

$$\chi_{m,n} = \log_2 \left(\prod_{\gamma=1}^L \prod_{\lambda=1}^{\Lambda} y(p)_{\gamma\lambda} \right), \quad (5)$$

где γ – количество диагоналей НЧ-области трансформанты; λ – число элементов диагонали; $y(p)_{\gamma\lambda}$ – γ, λ -я компонента НЧ-области трансформанты.

Для кадра, состоящего из $m \times n$ трансформант, величины $\chi_{m,n}$ вычисляются по всем строкам.

Если в последовательности $\chi_{m,1}, \chi_{m,n}$ выявлены значения насыщенностей, для которых разность $|\Delta\chi| = \chi_{m,k} - \chi_{m,k+1}$ имеет несущественную величину и справедливо соотношение $\chi_{m,1} \approx \chi_{m,k} \dots \approx \chi_{m,k+1}$, то данные трансформанты составляют вектор стабилизации $S_{j,\delta}$. Индекс j при этом определяет позицию трансформанты в кадре, а δ – количество входящих в вектор стабилизации трансформант.

В пределах вектора стабилизации $S_{j,\delta}$, с учетом подобия между трансформантами, существует возможность сократить количество выполняемых арифметических операций при обработке кадра.

В частности, некоторое количество значений битовых скоростей $R_{p\mu}$ и $d_{p\mu}$ битовых плоскостей одного разряда для трансформант вектора $S_{j,\delta}$ может быть интерполировано.

При полученных значениях R_{seq} и d_{seq} для последовательности $c(t_0)$ кадров определяется разность $\Delta R = B_w - R_{seq}$ между суммарной фактической битовой скоростью кадров серии и требуемой битовой скоростью, величина которой равна $R_{seq}^{треб} = B_w$ [4].

Если $\Delta R \leq 0$, то вся последовательность $c(t_0)$ передается в буфер передатчика без дополнительной обработки [6].

В случае, когда $\Delta R < 0$, битовую скорость R_{seq} необходимо снизить на величину $|\Delta R|$ для обеспечения требуемой битовой скорости $R_{seq}^{треб}$ последовательности кадров $c(t_0)$.

Используя величину требуемой битовой скорости [7] $R^{треб}$, соотношение (1) можем представить в следующем виде:

$$\begin{cases} R \leq R^{треб}; \\ d \leq d_{min} \end{cases} \quad (6)$$

Очевидно, что ΔR определяется следующим выражением:

$$\Delta R = \sum_{i=1}^{c(t_0)} \Delta R_i, \quad (7)$$

где ΔR_i – величина, на которую необходимо снизить битовую скорость каждого кадра последовательности $c(t_0)$.

В этом случае вычисление ΔR_i производится пропорционально величинам битовых скоростей каждого кадра последовательности $c(t_0)$ [8, 9], как показано в выражении:

$$\Delta R_i = \frac{\Delta R}{c} \cdot \frac{R_i}{R_{cp}} = \frac{\Delta R R_i}{R_{seq}}, \quad (8)$$

где $c(t_0)$ – число кадров в серии; R_i – битовая скорость i -го кадра; $R_{cp} = \frac{R_{seq}}{c(t_0)}$ – средняя битовая скорость кадра в серии.

В свою очередь, требуемая битовая скорость кадра последовательности $c(t_0)$ определяется формулой:

$$R_i^{треб} = R_i - \Delta R_i. \quad (9)$$

Для эффективного учета битовой скорости трансформант в пределах кадра используется подход, учитывающий характер распределения битовой скорости в кадре:

$$R_p^{треб} = \frac{R_i^{треб}}{Q} \gamma_p, \quad (10)$$

где γ_p – коэффициент, зависящий от степени насыщенности p -й трансформанты.

После того, как для каждой трансформанты кадра найдены величины $R_p^{треб}$, определяются битовые плоскости, которые будут исключены, чтобы обеспечить значения битовых скоростей в соответствии с (10).

Для плоскостей трансформант, битовые скорости которых необходимо снизить до величины $R_p^{треб}$, определяется порядок ранжирования, при котором первыми обрабатываются битовые плоскости $Y^{(p,\mu)}$, вносящие максимальные значения $d_{p\mu}$ в общее СКО трансформанты.

В первую очередь это относится к старшим битовым плоскостям $Y^{(p,\mu)}$. Далее обрабатываются битовые плоскости в порядке снижения величин $d_{p\mu}$, вносимых ими в общее СКО.

После определения ранжирования битовых плоскостей на каждом p -м шаге вычисляются суммарные значения СКО и битовой скорости трансформант $Y_p^{(\mu)}$ и $Y_{p+1}^{(\mu)}$.

Сложение значений СКО и битовых скоростей трансформант $Y_p^{(\mu)}$ и $Y_{p+1}^{(\mu)}$ происходит попарно, с учетом порядка обработки.

Суммируются при этом только величины, имеющие одинаковые индексы ранжирования.

На каждом p -м шаге вычисления требуемой битовой скорости трансформанты определяется условно-оптимальное $R_{F,p}^*$ значение битовой скорости кадра, состоящего из трансформант, исходя из условий:

$$\begin{cases} R_{i,p}^* \in \{R_{i,p}^*\} \mid R_{i,p} \leq R_p^{треб}; \\ d_{i,p} \rightarrow \min \end{cases} \quad (11)$$

Величина $R_{F,p+1}$ для кадра F на $(p+1)$ -м шаге в этом случае будет определяться следующим способом:

$$R_{F,p+1} = \sum_{i=p}^{p+1} \sum_{u=1}^n R_{i,u}, \quad (12)$$

где u – индекс очередности обработки битовой плоскости; $R_{i,u}$ – битовая скорость u -й битовой плоскости трансформанты в порядке снижения вносимого уровня СКО.

В свою очередь, СКО на $(p+1)$ -м шаге будет определяться выражением:

$$d_{F,p+1} = \sum_{i=p}^{p+1} \sum_{u=1}^n R_{i,u}, \quad (13)$$

где $d_{i,u}$ – битовая скорость u -й битовой плоскости трансформанты.

В результате сложения битовых скоростей $Y_p^{(\mu)}$ и $Y_{p+1}^{(\mu)}$ СКО по трансформантам и на шаге результиру-

ющий порядок обхода полученного множества будет определяться суммарными значениями СКО по уменьшению.

Это позволяет более эффективно производить коррекцию битовой скорости трансформант, исключая необходимый объем битовых плоскостей для снижения битовой скорости трансформанты, не прибегая к дополнительным операциям.

Результирующая битовая скорость R_{seq} последовательности кадров определяется выражением:

$$R_{seq}^{треб} = \sum_{i=1}^c \sum_{p=1}^Q R_{i,p}^{треб}, \quad (14)$$

где $R_{p,i}^{треб}$ – полученное в соответствии с (12) значение битовой скорости p -й трансформанты i -го кадра последовательности.

Выбор на каждом p -м шаге вычисления требуемой битовой скорости трансформанты в соответствии с условиями (11) гарантирует, что уровень СКО, соответствующий полученной последовательности из $c(t_0)$ кадров, будет минимально возможным в момент времени передачи t_0 . Сформированная последовательность $c(t_0)$ кадров помещается в выходной буфер [10], размер которого рассчитывается согласно следующей формуле:

$$R_{буф} = cRTT R_{ср}. \quad (15)$$

После помещения $c(t_0)$ кадров в буфер происходит отправка всей серии кадров в канал, а также определение RTT, по величине которой находится значение полосы пропускания B_w в момент t_1 , в соответствии с (3), а также количество кадров $c(t_1)$, из которого будет состоять передача в момент t_1 .

4. Выводы

Предложена методика управления интенсивностью битовой скорости, способствующая адаптации интенсивности видеопотока в соответствии с изменяющейся пропускной способностью канала. Рассмотрены условия и способы обеспечения эффективной передачи видеопотока с использованием предложенной методики. Приведены способы эффективного распределения битовой скорости между трансформантами кадра. Показано, что при данном способе обработки кадров уровень ошибки на приемной стороне будет минимально возможным.

Литература: 1. Баранник В., Двухглавов Д., Твердохлеб В. Метод динамического управления битовой скоростью видеопотока с использованием трехмерного представления трансформант // АСУ и ПА. 2014. №176. С. 37 - 43. 2. Barannik, V., Dvuhglavov, D., Tverdokhlebl, V., Krasnorutskiy, A. Controlling of video stream bit rate using the dynamic programming method // 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2015. Lviv-PI, 2016, № 2

Polyana; Ukraine; 24 February 2015 - 27 February 2015. P. 15-17. 3. Barannik V.B. Методологические рекомендации по совершенствованию технологии снижения интенсивности кодового представления базовых кадров / В.В. Баранник, О.Ю. Отман Шади, А.А. Подорожняк // Системы обработки информации. 2014. № 8(124). С. 87-92. 4. Barannik, V.V., Kharchenko, N., Tverdokhlebl, V.V., Kulitsa, O. The issue of timely delivery of video traffic with controlled loss of quality // 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science, TCSET 2016; Lviv-Slavsko; Ukraine; 23 - 26 February 2016. P. 902-904. 5. Сэломон, Д. Сжатие данных, изображения и звука / Д. Сэломон. М.: Техносфера, 2004. 368 с. 6. Гонсалес, Р.Э. Цифровая обработка изображений/ Р.С. Гонсалес, Р.Э. Вудс. М.: Техносфера, 2006. 1072 с. 7. Ян Ричардсон. Видеокодирование. H.264 и MPEG-4 - стандарты нового поколения // Москва: Техносфера, 2005. 368с. 8. Баранник В.В. Технологическая концепция обработки базовых кадров для снижения интенсивности кодового представления / В.В. Баранник, О. Ю. Отман Шади // Радиоэлектронные и компьютерные системы. 2014. № 4. С. 25-31. 9. Баранник В.В. Методологические рекомендации по совершенствованию технологии снижения интенсивности кодового представления базовых кадров / В.В. Баранник, Отман Шади О.Ю., А.А. Подорожняк // Системы обработки информации. №8(124). 2014. С. 87-93. 10. Barannik, V., Kharchenko, N., Othman Shadi, O.Y., Musienko, A. A method to control bit rate while compressing predicted frames // 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2015. Lviv-Polyana. Ukraine. 24 - 27 February 2015. P. 36-38.

Transliterated bibliography:

1. Barannik V., Dvuhglavov D., Tverdokhlebl V. Metod dinamicheskogo upravleniya bitovoy skorostyu videopotoaka s ispolzovaniem trehmernogo predstavleniya transformant // ASU i PA. 2014. #176. S. 37 - 43.
2. Barannik, V., Dvuhglavov, D., Tverdokhlebl, V., Krasnorutskiy, A. Controlling of video stream bit rate using the dynamic programming method // 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2015. Lviv-Polyana; Ukraine; 24 February 2015 - 27 February 2015; pp. 15-17.
3. Barannik V.V. Metodologicheskie rekomendatsii po sovershenstvovaniyu tehnologii snizheniya intensivnosti kodovogo predstavleniya bazovyih kadrov / V.V. Barannik, O.Yu. Otman Shadi, A.A. Podorozhnyak // Sistemi obrobki Informatsiyi. 2014. # 8(124). S. 87-92.
4. Barannik, V.V., Kharchenko, N., Tverdokhlebl, V.V., Kulitsa, O. The issue of timely delivery of video traffic with controlled loss of quality // 13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science, TCSET 2016; Lviv-Slavsko; Ukraine; 23 - 26 February 2016. R. 902-904.
5. Selomon, D. Szhatie dannyih, izobrazheniy i zvuka / D. Selomon. M.: Tehnosfera, 2004. 368 s.
6. Gonsales R.S. Tsifrovaya obrabotka izobrazheniy/ R.S. Gonsales, R.E. Vuds. M.: Tehnosfera, 2006. 1072 s.
7. Yan Richardson. Videokodirovanie. N.264 i MPEG-4 - standartyi novogo pokoleniya // Moskva: Tehnosfera, 2005. 368s.

Поступила в редколлегию 12.05.2016

Рецензент: д-р техн. наук, проф. Безрук В.М.

8. Barannik V.V. Tehnologicheskaya kontsep-tsiya obrabotki bazovyih kadrov dlya snizheniya intensiv-nosti kodovogo predstavleniya / V.V. Barannik, O. Yu. Otman Shadi // Radioelektronnyie i kompyuternyye sistemyi. 2014. # 4. S. 25-31.

9. Barannik V.V. Metodologicheskie rekomendatsii po sovershenstvovaniyu tehnologii snizheniya intensivnosti kodovogo predstavleniya bazovyih kadrov / V.V. Barannik, Otman Shadi O.Yu., A.A. Podorozhnyak // Sistemi obrobki InformatsiYi. # 8(124). 2014. S. 87-93.

10. Barannik, V., Kharchenko, N., Othman Shadi, O.Y., Musienko, A. A method to control bit rate while compressing predicted frames // 13th International Conference: The Experience of Designing and Application of CAD Systems in Microelectronics, CADSM 2015. Lviv-Polyana. Ukraine. 24 - 27 February 2015. R. 36-38.

Твердохлеб Виталий Викторович, аспирант кафедры информационно-сетевой инженерии ХНУРЭ, ww80@mail.ru. Адрес: Украина, 61166, Харьков, пр. Науки, 14.

Tvervdokhleб Vitaliy Viktorovich, PhD student of Department of information network engineering of Kharkiv National University of Radio Electronics, ww80@mail.ru.

СИСТЕМЫ И ПРОЦЕССЫ УПРАВЛЕНИЯ



УДК681.518.24

МОДЕЛИ СИСТЕМЫ ЭМУЛЯЦИИ СЛОЖНОГО ТЕХНОЛОГИЧЕСКОГО ОБЪЕКТА УПРАВЛЕНИЯ

НАЙДА В.В., ОКСАНИЧ А.П., ШЕВЧЕНКО И.В., КОНОХ И.С.

Разрабатывается теоретико-множественная модель структуры системы эмуляции поведения сложного технологического объекта при его взаимодействии с системой управления. Предлагается нечеткая вычислительная модель для решения задачи эмуляции поведения объекта управления.

Введение

Широкое распространение сложных автоматизированных технологических комплексов, включающих в себя технологические объекты управления и автоматизированные системы управления, а также необходимость сокращения сроков разработки и внедрения приводят к необходимости использования имитационного моделирования (эмуляции) в системах отладки АСУТП. Имитационные комплексы обеспечивают отработку взаимодействия проектируемых систем управления с объектом управления и окружающей средой. Несмотря на то, что математическое моделирование значительно уменьшает стоимость исследовательских и отладочных работ, обладает хорошей повторяемостью и безопасностью, оно не позволяет полностью исключить натурные испытания и исследования качества работы систем управления при взаимодействии с реальным объектом.

Для сложных объектов, таких как корабельные системы, системы управления летательных аппаратов, электрических станций, обогатительных установок, обязательным этапом является отладка и настройка программно-аппаратных компонент на испытательных комплексах.

Известны программно-аппаратные имитационные комплексы [1], которые позволяют для информационно-управляющих систем боевых кораблей формировать внешние сигналы на основе специальных аппаратно-программных средств. Дополнительными преимуществами являются возможность тренинга обслуживающего персонала и исследование реакции системы управления в нештатных ситуациях.

В источниках [1-4] показаны характерные приложения для отладочных комплексов с имитацией электрических сигналов. Для подобных комплексов актуаль-

ным является совершенствование специальных методов идентификации объектов управления и эмуляции их поведения, которые характеризуются высокой степенью автоматизации и точностью воспроизведения осведомительных сигналов с сохранением метрологических особенностей.

Идентификация многомерных объектов в кибернетике и других областях предусматривает построение с помощью средств математики соответствующих им операторов связи между функциями входных и выходных сигналов на основе доступной информации [5]. При этом процесс создания математической модели того или иного объекта состоит из двух основных этапов:

- 1) структурная идентификация – установление переменных величин и элементов структуры модели;
- 2) параметрическая идентификация – этап, во время которого на основе наблюдения входных и выходных сигналов объекта определяются значения параметров и устанавливается окончательная структура модели путем устранения лишних элементов.

В основе современной теории идентификации лежит моделирование исследуемых объектов при помощи уравнений (дифференциальных, разностных и т. п.). При этом сложность того или иного объекта самым непосредственным образом влияет на качество построения его модели. Если для описания некоторых объектов применяется информация, которая не может быть выражена количественно – так называемая семантическая, т. е. смысловая, качественная информация, то классическая теория оказывается плохо приспособленной для таких случаев. Основные причины малой эффективности или же вообще непригодности традиционных методов моделирования к подобным ситуациям состоят в следующем:

- не все входные и выходные параметры объекта могут описываться количественно;
- между рядом входных и выходных параметров невозможно установить количественные зависимости;
- существующие способы моделирования объектов приводят к таким громоздким конструкциям, что их практическое применение оказывается невозможным;
- объект эволюционирует во времени, его структура и функции изменяются.

Преодоление таких сложностей с высокой эффективностью возможно при использовании технологий искусственного интеллекта. Известна работа [6], которая посвящена исследованиям в области идентификации нелинейных многомерных объектов нечеткими базами знаний. Этот метод идентификации ранее был научно обоснован в предыдущих работах. Для решения задачи идентификации предлагается комплексное использование нечетких баз знаний, нейронных сетей и генетических алгоритмов, объединенных под термином интеллектуальные технологии или мягкие вычисления (Soft Computing).

При моделировании объектов, которые характеризуются указанными особенностями, возникает проблема построения так называемых логико-лингвистических моделей, т.е. моделей, в которых средства обработки информации основаны на логике, а экспериментальные данные представляются в лингвистической форме [6]. Такие модели должны основываться на системах знаний об исследуемом объекте, которые представляют собой концентрацию опыта специалистов (экспертов) в данной области. Для систем знаний, в свою очередь, должен быть выбран язык представления знаний, при помощи которого в модели можно было бы адекватно воссоздавать сведения о структуре объекта, и который по возможности должен быть близким к естественному языку (в простейшем варианте – быть ее определенным подмножеством) и характеризоваться достаточной формальностью и логичностью в целях построения компактной, строгой и четкой системы знаний.

Одним из таких современных формальных аппаратов для обработки экспертной естественно-языковой информации является теория нечетких множеств. В соответствии с этой теорией модель объекта представляет собой так называемую нечеткую базу знаний в виде совокупности логических высказываний. Адекватность таких моделей к данным эксперимента определяется качеством функций принадлежности [7], при помощи которых лингвистические оценки превращаются в количественную форму. Но поскольку функции принадлежности определяются экспертными методами, адекватность нечетких моделей целиком зависит от квалификации экспертов. Иначе говоря, проблема адекватности известных нечетких моделей остается открытой. А когда привлечение экспертов для построения модели оказывается невозможным по причине их отсутствия, в таком случае возникает проблема извлечения лингвистических знаний об объекте из экспериментальных данных.

Таким образом, актуальность поднятой проблемы обусловлена тем, что для моделирования многих объектов есть смысл применять логико-лингвистические модели, дающие возможность преодолеть трудности моделирования классическими методами. Однако во многих случаях повышенная сложность того или иного объекта или его новизна (и как следствие – недостаточное освоение) делают невозможным привлечение квалифицированных экспертов для построения таких моделей. Это порождает проблему отыскания закономерностей, которые бы легли в основу системы лингвистических знаний об объекте моделирования, из имеющихся статистических (экспериментальных) данных, характеризующих исследуемый объект. Такие модели обычно являются «грубыми», поэтому не менее важен вопрос о поиске эффективных методов их тонкой настройки.

Целью исследования является разработка моделей системы эмуляции поведения сложного технологического объекта при его взаимодействии с системой управления, позволяющих разработать информационную технологию эмуляции и испытаний без разработки и использования сложных систем дифференциальных уравнений и систем имитационного моделирования.

1. Формальная постановка задачи эмуляции объекта управления и тестирования системы управления

Для записи формальной постановки задачи эмуляции объекта управления (ОУ) и тестирования СУ введем следующие обозначения:

$T = \{t_i, i = \overline{1, L}\}$ – множество дискретных моментов времени в сеансе тестирования; $U = \{u_i(t) | u_i(t) \in D^{U_i}, i = \overline{1, I}\}$, $t \in T$ – множество управляющих воздействий на объект управления (ОУ);

$D^U = \{D^{U_i}, i = \overline{1, I}\}$ – область допустимых значений управляющих воздействий;

$P = \{p_j(t) | p_j(t) \in D^{P_j}, j = \overline{1, J}\}$, $t \in T$ – множество экспериментальных последовательностей входных сигналов для ОУ; $D^P = \{D^{P_j}, j = \overline{1, J}\}$ – области допустимых значений входных сигналов для ОУ;

$S = \{S_i; S_i = f(S_{i-1}, U, P), U \in D^U, P \in D^P, t \in T\}$ –

множество состояний ОУ; $SI = \{S_i; S_i = \varphi(S_i), t \in T\}$ – множество параметров, характеризующих состояние ОУ;

$Y = \{y_m(t), m = \overline{1, M}\}$, $t \in T$, $y_m(t) = \eta_m(t, S_t)$, $t \in T$ – множество выходов ОУ; $Z = \{z_n(t), n = \overline{1, N}\}$, $t \in T$ – множество задаваемых параметров, определяющих режим сеанса тестирования;

$C = \{c_k(t) | c_k(t) \in D^{C^k}, k = \overline{1, K}\}$, $t \in T$ – множество желаемых последовательностей экспериментальных сигналов на выходе ОУ;

$D^C = \{D^{C^k}, k = \overline{1, K}\}$ – область допустимых значений желаемых экспериментальных сигналов на выходе ОУ;

$Q = \{q_g, g = \overline{1, G}\}$ – множество целей тестирования СУ.

Тогда взаимодействие системы управления и объекта управления в процессе тестирования и эмуляции можно представить рядом отображений:

$\eta: U \times P \times S \times T \rightarrow Y$ – отображение множества управляющих воздействий U , выбранных из множества экспериментальных последовательностей входных сигналов P , множества состояний ОУ S на всём интервале времени T , в котором осуществляется тестирование, во множество выходов ОУ Y ;

$\mu: U \times P \times T \rightarrow S$ – отображение множества управляющих воздействий U , выбранных из множества экспериментальных последовательностей входных сигналов P на всём интервале времени T , на множество состояний S ;

$\psi: S \times T \rightarrow SI$ – отображение множества состояний ОУ во множество параметров состояния ОУ на интервале времени T ;

$\varphi: Q \times Y \times SI \rightarrow C$ – отображение множества целей тестирования Q , множества выходов ОУ Y и множества параметров состояний ОУ SI на множество желаемых последовательностей экспериментальных сигналов на выходе ОУ.

Выбор критерия качества процесса тестирования. В процессе тестирования система тестирования и эмуляции должна фиксировать последовательности входных сигналов ОУ, последовательности выходных сигналов ОУ и оценивать степень соответствия последовательности выходных сигналов Y желаемой последовательности C .

Считая, что успешность процесса тестирования определяется степенью совпадения выходных последовательностей Y и желаемых выходных последовательностей C в различных сценариях тестирования, запишем критерий успешности процесса тестирования в виде:

$$Q_T = \sum_{k=1}^K \sum_{i=1}^N \sum_{t=0}^T (y(u_t, s_t)_{kit} - c_{kit})^2 \rightarrow \min, \quad (1)$$

где k – номер сценария тестирования; K – количество сценариев; i – номер отрезка последовательности сигналов, на котором фиксируются значения входа и выхода; N – число отрезков последовательностей.

В систему ограничений в данной задаче входят:

а) ограничения на параметры состояния и управления:

$$\begin{aligned} s(t) &\in D^S, t \in T; \\ u_i(t) &\in D^{U_i}, i = \overline{1, I}, t \in T; \\ c_k(t) &\in D^{C_k}, k = \overline{1, K}, t \in T; \end{aligned}$$

б) показатели качества функционирования ОУ, зависящие от специфики и назначения ОУ.

Ограничения могут быть жесткими и нежесткими, в последнем случае они могут быть записаны в вероятностном виде, нечетком виде.

Таким образом, сформулирована задача эмуляции сложного технологического объекта управления и тестирования системы управления.

2. Структура системы эмуляции и тестирования

Структуру системы эмуляции и тестирования (СЭТ) представим набором:

$$STS = \langle IA(\eta, \mu, \psi, \varphi), F, DB, R, MM, OMI, ICS \rangle, \quad (2)$$

где IA – интеллектуальный агент, реализующий отображения η, μ, ψ, φ ; $DB(T, P, Y, C, D)$ – база данных, содержащая отсчеты времени, входные и выходные последовательности, сигналы и ограничения; F – множество функций СЭТ; $RH F \times Q$ – отображение множества функций СЭТ на множество целей тестирования комплекса СУ–ОУ; MM – мастер-монитор, осуществляющий диспетчерские функции в ходе процесса тестирования; OMI – интерфейс с объектом управления; ICS – интерфейс с системой управления.

3. Краткая характеристика комплекса функциональных задач СЭТ

Комплекс функциональных задач СЭТ предназначен для решения задачи оптимизации параметров системы управления сложного технологического объекта в процессе отладки и испытаний.

В комплекс входят следующие задачи:

- полуавтоматическая настройка входных и выходных сигналов тестирующих процедур;
- генерация и фиксация тестовых воздействий на ОУ и фиксация откликов ОУ;
- структурирование и анализ результатов тестирования;
- формирование отчетов и рекомендаций для оператора.

Комплекс задач решается в процессе испытаний СУ в режиме Hardware-in-the-loop. Продолжительность решения задач обусловлена продолжительностью процесса тестирования и испытаний. Периодичность решения задач определяется периодичностью проектных и пуско-наладочных работ на технологических объектах. Автоматизированное решение задач может быть прекращено по желанию оператора.

В процессе решения данного комплекса задач СЭТ связана информационными потоками с системой автоматического управления ОУ и оператором, проводящим тестирование.

Входной информационный поток комплекса задач содержит:

1. Исходные данные о входных и выходных сигналах ОУ и данные о связях «вход-выход» ОУ.
2. Сигналы СУ, подаваемые на ОУ, и сигналы ОУ – реакции на тестовые воздействия.

Выходной информационный поток комплекса задач содержит структурированные массивы значений выходных сигналов ОУ и результаты их анализа.

4. Концепция функционирования СЭТ

Предлагаемый метод решения задачи эмуляции поведения объекта управления базируется на идее накопления экспериментальных временных рядов входов и выходов ОУ и расчете на их основе точек новой последовательности, которая соответствует текущему входному управляющему сигналу. При этом целесообразно использовать предварительную классификацию последовательностей путем указания для каждой из них атрибутов, содержащих информацию о форме и величине управляющего воздействия, начальном состоянии объекта, производных выходной величины. Для объектов выше второго порядка при поиске ближайших точек в выбранных последовательностях необходимо учитывать их предыдущие состояния в моменты времени, определяемые постоянными времени передаточных функций. Информация о всех последовательностях хранится в БД, содержащей следующую информацию:

- тип управляющего воздействия;
- амплитуда входного воздействия;
- дополнительная характеристика управляющего воздействия (например, скорость нарастания линейного сигнала);

- начальное состояние (начальный выход) объекта;
- значение дискреты времени;
- путь и имя файла, непосредственно содержащего экспериментальные отчеты.

Последовательности, которые возможно использовать для расчета выхода объекта управления, выбираются на основании определения евклидова расстояния по набору переменных. Сюда входят тип и амплитуда управляющего воздействия, начальный выход объекта, время с момента изменения управляющего воздействия. Исходя из этого, целесообразно исходные непрерывные экспериментальные последовательности разбить на более короткие фрагменты для сокращения времени поиска.

После выбора нескольких последовательностей, которые удовлетворяют условию поиска, необходимо определить отдельные точки в них и рассчитать интерполированные значения.

Для универсальности метода необходимо на основе указанной информации рассчитать относительные отклонения сигналов и подать их на входы интеллектуального агента, роль которого играет нечеткая вычислительная модель.

5. Нечеткая вычислительная модель

Нечеткая модель имеет четыре входные переменные из множества X :

- $x_1 = \langle \text{type} \rangle$ – тип управляющего воздействия (ступенчатое или линейное);
- $x_2 = \langle \text{impact} \rangle$ – относительная амплитуда управляющего воздействия;
- $x_3 = \langle \text{sprev} \rangle$ – прошлое относительное значение выхода объекта;
- $x_4 = \langle \text{scurrent} \rangle$ – относительное значение выхода объекта в момент времени, которое непосредственно предшествует рассчитываемой точке.

Единственная выходная переменная out определяет коэффициент, на который следует умножить текущее значение экспериментального сигнала, чтобы получить искомое значение.

Для исследуемых примеров были выбраны диапазоны лингвистических переменных и количество термов (рис. 1). База правил (таблица) содержит 23 правила, которые определяют в лингвистических терминах выход нечеткого контроллера.

Нечеткая модель на первом этапе вычисляет дискретное значение переменной out_j на основании значений степени истинности термов переменных из множества X ($type$, $impact$, $sprev$, $scurrent$) в соответствии с базой знаний (таблица):

$$\bigcup_{p=1}^P \bigcap_{i=1}^n (x_i = A_i^k) \rightarrow out_j, \quad (3)$$

где A_i^k – лингвистическая оценка (терм) параметра x_i ; k – номер терма в соответствующем терм-множестве; p – номер конъюнктивной цепочки в нечетком правиле; P_j – количество конъюнктивных цепочек в правиле с номером j ; out_j – логический вывод по текущей ситуации в правиле j .

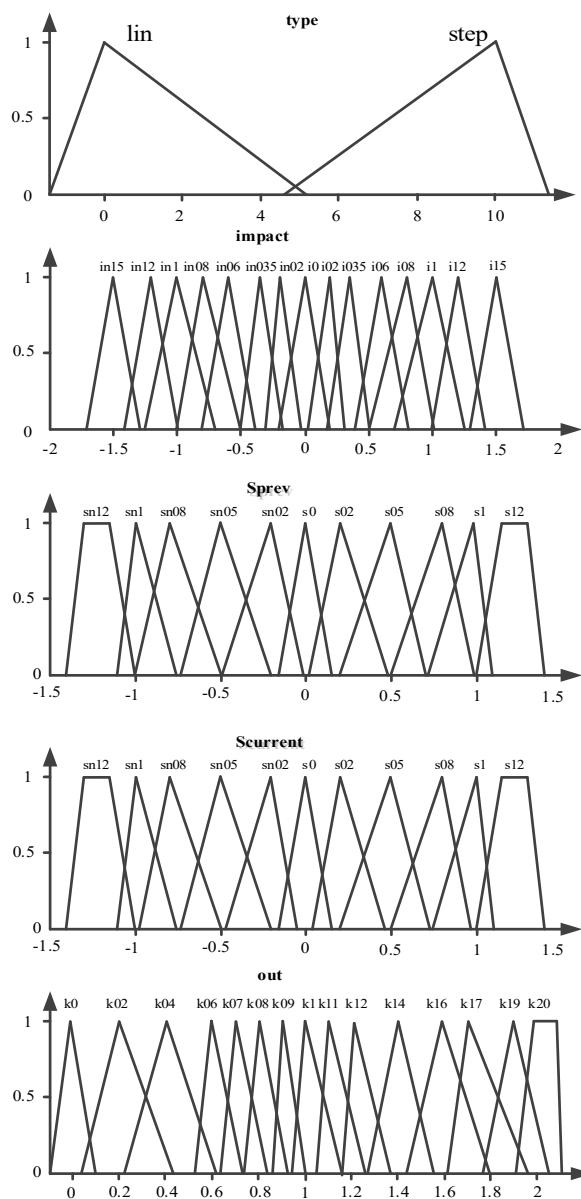


Рис. 1. Функции принадлежности входных и выходных переменных для расчета выхода объекта по экспериментальной последовательности

Преобразование агрегированных нечетких значений out_j производится центроидным методом.

Поскольку расчетная временная последовательность может быть получена путем интерполяции на основании двух имеющихся (ближайших по параметрам) последовательностей, в СЭТ используются два нечетких контроллера, реализующих описанные выше вычислительные функции.

Результаты вычислений нечетких контроллеров усредняются на основании значений расчетных коэффициентов по следующей формуле:

$$S = 0.5 \frac{(1 - |1 - k1|)}{(1 - |1 - k2|)} u1(t) + (1 - 0.5 \frac{(1 - |1 - k1|)}{(1 - |1 - k2|)}) u2(t), \quad (4)$$

где $u1(t)$, $u2(t)$ – текущие значения реальных экспериментальных сигналов; $k1$, $k2$ – расчетные коэффициенты, определенные нечеткими контроллерами для каждой последовательности в отдельности. Формула (4) описывает логику потоковых вычислений для формирования итогового результата.

№ п.п.	Экспертные правила
1	If (type is step) and (impact is i0) and (Sprev is s0) and (Scurrent is s0) then (out is k1)
2	If (type is step) and (impact is i02) and (Sprev is s0) and (Scurrent is s02) then (out is k09)
3	If (type is step) and (impact is i02) and (Sprev is sn02) and (Scurrent is s02) then (out is k08)
4	If (type is step) and (impact is i02) and (Sprev is s02) and (Scurrent is s05) then (out is k07)
5	If (type is step) and (impact is in08) and (Sprev is s0) and (Scurrent is sn08) then (out is k17)
6	If (type is step) and (impact is in1) and (Sprev is s0) and (Scurrent is sn1) then (out is k20)
7	If (type is step) and (impact is in1) and (Sprev is sn08) and (Scurrent is sn1) then (out is k20)
8	If (type is step) and (impact is i0) and (Sprev is sn1) and (Scurrent is sn05) then (out is k14)
9	If (type is step) and (impact is i0) and (Sprev is sn05) and (Scurrent is s0) then (out is k11)
10	If (type is step) and (impact is i0) and (Sprev is sn02) and (Scurrent is s0) then (out is k1)
11	If (type is step) and (impact is in1) and (Scurrent is sn1) then (out is k20)
12	If (type is step) and (impact is in08) then (out is k17)
13	If (type is step) and (impact is i0) and (Sprev is sn1) and (Scurrent is sn1) then (out is k17)
14	If (type is step) and (impact is i0) and (Sprev is sn1) and (Scurrent is sn08) then (out is k16)
15	If (type is step) and (impact is i0) and (Sprev is sn08) and (Scurrent is sn02) then (out is k12)
16	If (type is step) and (impact is i0) and (Sprev is sn08) and (Scurrent is s0) then (out is k1)
17	If (type is step) and (impact is i0) and (Sprev is sn05) and (Scurrent is s0) then (out is k1)
18	If (type is step) and (impact is i02) and (Sprev is s0) and (Scurrent is s0) then (out is k08)
19	If (type is step) and (impact is i02) and (Sprev is s0) and (Scurrent is s02) then (out is k08)
20	If (type is step) and (impact is i0) and (Sprev is s02) and (Scurrent is s02) then (out is k09)
21	If (type is step) and (impact is i0) and (Sprev is sn02) and (Scurrent is sn02) then (out is k11)
22	If (type is step) and (impact is i0) and (Sprev is sn02) and (Scurrent is s02) then (out is k08)
23	If (type is step) and (impact is i0) and (Sprev is s02) and (Scurrent is sn02) then (out is k12)
24	If (type is step) and (impact is in035) and (Scurrent is sn02) then (out is k12)
25	If (type is step) and (impact is in035) and (Scurrent is sn05) then (out is k14)
26	If (type is step) and (impact is i035) and (Scurrent is s02) then (out is k06)
27	If (type is step) and (impact is i0) and (Scurrent is s05) then (out is k06)
28	If (type is step) and (impact is i0) and (Scurrent is s02) then (out is k07)

Таким образом, выражения (3) и (4) в совокупности с базой знаний (таблица) представляют собой нечеткую вычислительную модель для решения задачи интерполяции выходного сигнала объекта управления в процессе эмуляции и тестирования.

Экспериментальная проверка нечеткой модели выполнялась путем модельных экспериментов с сигналами, формируемыми динамическими звеньями первого и второго порядков:

$$W_1(s) = \frac{3}{0,2s + 1}, \quad W_2(s) = \frac{3}{0,1s^2 + 0,2s + 1}, \\ W_3(s) = \frac{0,4s + 3}{0,05s^2 + 0,1s + 1}.$$

Для экспериментальной проверки были выбраны двухступенчатые управляющие воздействия, при которых искомая последовательность имела такие воздействия, что ее значения в одно время были близки к первой, а в другое – ко второй последовательности, полученных от одного и того же звена. Таким образом, схема проведения модельных экспериментов включает модули задания управляющих воздействий и две одинаковые передаточные функции, моделирующие экспериментальные данные. Их выход моделирует найденные точки, по которым можно определить следующее значение выхода объекта. Начальные условия – нулевые. Два нечетких контроллера, каждый из которых на основании входной информации оценивает близость искомой последовательности к экспериментальной, и формируют расчетный коэффициент. Такая вычислительная процедура обеспечивает больший вес для того результата, чей коэффициент ближе к единице, что дает преимущество более близкой последовательности и позволяет лучше учесть нелинейные свойства объекта управления, если они есть. Рассчитанное и усредненное значение управляемой переменной используется для поиска новых точек в экспериментальных последовательностях и расчетах новых значений входных лингвистических переменных.

Результаты модельных экспериментов показаны на рис. 2. Так как в наличии имеется эталонный результат, то настройку нечеткой экспертной системы можно производить по технологии гибридных нейро-нечетких систем или путем последовательного пересмотра точек переходного процесса и формулирования экспертных правил для равномерно разбитых на термы лингвистических переменных. Выполнив такой процесс для объекта первого порядка, были сформулированы первые 17 экспертных правил. Аналогичные исследования для колебательного звена второго порядка позволили добавить еще 11 правил без изменения уже найденных.

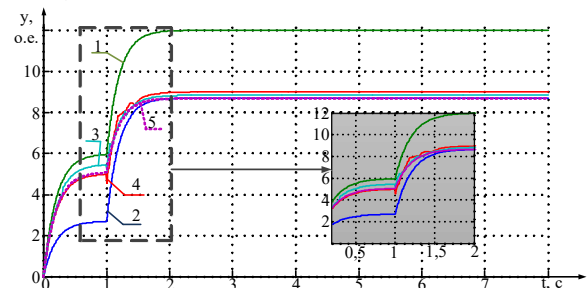


Рис. 2. Результаты модельных экспериментов по эмуляции выходного сигнала динамического звена первого порядка: 1, 2 – входные последовательности; 3, 4 – выходы нечетких контроллеров; 5 – усредненный итоговый результат

Выводы

1. Разработана формальная постановка задачи эмуляции объекта управления и тестирования системы управления для сложного технологического объекта.
2. Разработана теоретико-множественная модель структуры системы эмуляции поведения сложного технологического объекта при его взаимодействии с системой управления. Она отличается наличием интеллектуального агента, реализующего функции отображения множества управляющих воздействий, множества состояний объекта управления во множество выходов объекта, отображение множества функций системы эмуляции на множество целей тестирования, интерфейс с объектом управления и интерфейс с системой управления. Это дает возможность построения эффективной системы эмуляции и тестирования без использования имитационных моделей на основе уравнений динамики сложного объекта управления.
3. Предложена нечеткая вычислительная модель для решения задачи эмуляции поведения объекта управления, отличающаяся тем, что для вычисления реального выходного сигнала используется информация о текущем входном сигнале и нечеткая интерполяция выходного сигнала на основе имеющихся временных последовательностей входных и выходных сигналов объекта управления, что позволяет гибко моделировать поведение объекта управления без разработки и использования систем дифференциальных уравнений, а также повысить эффективность процесса испытаний и отладки системы управления.

Литература: 1. Ушаков В.С. Использование унифицированных имитационных комплексов при создании информационно-управляющих систем в судостроении // Современные технологии автоматизации. 2013. № 3. С. 102–108. 2. Ковалев С.Э. Схемная эмуляция в основе системы графического программирования / Промышленные измерения, контроль, автоматизация, диагностика (ПиКАД)". Киев. 4/2006. Режим доступа: <http://www.picad.com.ua/0406/0406.htm> 3. Ковалев С.Э. Схемная эмуляция в основе системы графического программирования (часть 2): Журнал "Промышленные измерения, контроль, автоматизация, диагностика (ПиКАД)". Киев – 1/2007. Режим доступа: <http://www.picad.com.ua/0107/0107.htm> 4. Волон Г.Я. Энергосбережение и имитационное моделирование. ОДО «Энергоэвент», Минск / Режим доступа: <http://www.energovent.com/myfiles/Articles/EnergySaving.pdf>. 5. Зарубин В.С. Математическое моделирование в технике: Учеб. для вузов / Под ред. В.С. Зарубина, А.П. Крищенко. М.: МГТУ им. Н.Э. Баумана, 2003. 496 с. (Сер. Математика в техническом университете; Вып. XXI, заключительный). 6. Митюшкин Ю. И., Мокін Б. І., Ротштейн О. П. Soft Computing: ідентифікація закономірностей нечіткими базами знань. Монографія. Вінниця: УНІВЕРСУМ. Вінниця, 2002. 145 с. 7. Девятков В.В. Системы искусственного интеллекта: Учеб. пособие для вузов. М.: Изд-во МГТУ им. Н.Э. Баумана, 2001. 352 с. 8. Конох І.С. Комп'ютеризований лабораторний комплекс для дослідження інтелектуальних цифрових систем керування електроприводом / [І. С. Конох, В. В. Найда, І. С. Гула] // Вісник КДПУ ім. М. Остроградського. Кременчук: КДПУ ім. М. Остроградського, 2008. Вип. 6 (53). Ч. 2. С. 17–21.

Transliterated bibliography:

1. Ushakov V.S. Ispolzovanie unifitsirovannyih imitatsionnyih kompleksov pri sozdanii informatsionno-upravlyayuschih sistem v sudostroenii // Sovremennyye tehnologii avtomatizatsii. 2013. # 3. S. 102–108.
2. Kovalev S.E. Shemnaya emulyatsiya v osnove sistemy graficheskogo programirovaniya / Promyishlennyye izmereniya, kontrol, avtomatizatsiya, diagnostika (PiKAD)". Kiev. 4/2006. Rezhim dostupa: <http://www.picad.com.ua/0406/0406.htm>
3. Kovalev S.E. Shemnaya emulyatsiya v osnove sistemy graficheskogo programirovaniya (chast 2): Zhurnal "Promyishlennyye izmereniya, kontrol, avtomatizatsiya, diagnostika (PiKAD)". Kiev – 1/2007. Rezhim dostupa: <http://www.picad.com.ua/0107/0107.htm>
4. Volov G.Ya. Energoberezhenie i imitatsionnoe modelirovanie. ODO «Energovent», Minsk / Rezhim dostupa: <http://www.energovent.com/myfiles/Articles/EnergySaving.pdf>.
5. Zarubin V.C. Matematicheskoe modelirovanie v tehnike: Ucheb. dlya vuzov / Pod red. V.C. Zarubina, A.P. Krischenko. M.: MGTU im. N.E. Baumana, 2003. 496 s. (Ser. Matematika v tehnichestkom universitete; Vyip. XXI, zaklyuchitelnyiy).
6. Mityushkin Yu. I., Mokin B. I., Rotshteyn O. P. Soft Computing: identyfikatsiya zakonmirnostey nechitkymy bazamy znan'. Monohrafiya. Vinnytsya: UNIVERSUM. Vinnytsya, 2002. 145 s.
7. Devyatkov V.V. Sistemy iskusstvennogo intellekta: Ucheb. posobie dlya vuzov. M.: Izd-vo MGTU im. N.E. Baumana, 2001. 352 s.
8. Konokh I.S. Komp'yuteryzovanyy laboratornyy kompleks dlya doslidzhennya intelektual'nykh tsyfrovyykh system keruvannya elektropryvodom / [I. S. Konokh, V. V. Nayda, I. S. Hula] // Visnyk KDPU im. M. Ostrohrads'koho. Kremenchuk: KDPU im. M. Ostrohrads'koho, 2008. Vyp. 6(53). Ch. 2. S. 17–21.

Поступила в редакцию 12.05.2016

Рецензент: д-р техн. наук, проф. Невлюдов И.Ш.

Найда Виталий Владимирович, аспирант кафедры информационно-управляющих систем Кременчугского национального университета им. М. Остроградского. Научные интересы: интеллектуальные информационные технологии процессов тестирования и наладки сложных систем управления. Адрес: Украина, 39600, Кременчуг, ул. Первомайская, 20, E-mail: vitalik.najda@yandex.ru

Оксанич Анатолий Петрович, д-р техн. наук, профессор, заведующий кафедрой информационно-управляющих систем Кременчугского национального университета им. М. Остроградского. Научные интересы: автоматизация сложных технологических процессов, оборудование и технологии производства материалов электронной техники. Адрес: Украина, 39600, Кременчуг, ул. Первомайская, 20, тел.: (05366) 30157. E-mail: oksanich@kdu.edu.ua

Шевченко Игорь Васильевич, д-р техн. наук, доцент, профессор кафедры информационно-управляющих систем Кременчугского национального университета им. М. Остроградского. Научные интересы: интеллектуальные информационные технологии контроля и управления в сложных технологических процессах. Адрес: Украина, 39600, Кременчуг, ул. Первомайская, 20, E-mail: ius.shevchenko@gmail.com

Конох Игорь Сергеевич, канд. техн. наук, доцент, доцент кафедры информационно-управляющих систем Кременчугского национального университета им. М. Остроградского. Научные интересы: промышленная автоматизация, интеллектуальные системы управления технологическими процессами. Адрес: Украина, 39600, Кременчуг, ул. Первомайская, 20, E-mail: kis_sau@mail.ru

ЧИСЛЕННЫЙ АНАЛИЗ ОДНОГО НЕЛИНЕЙНОГО ЭЛЛИПТИЧЕСКОГО УРАВНЕНИЯ, ВОЗНИКАЮЩЕГО ПРИ МОДЕЛИРОВАНИИ МИКРОЭЛЕКТРОМЕХАНИЧЕСКИХ СИСТЕМ

КОНЧАКОВСКАЯ О.С., СИДОРОВ М.В.

Рассматривается задача расчета микроэлектромеханической системы, математической моделью которой служит нелинейная краевая задача для эллиптического уравнения. На основании метода последовательных приближений строится алгоритм получения двусторонних приближений к точному решению задачи. Эффективность разработанного численного метода иллюстрируется сериями вычислительных экспериментов.

Ключевые слова: микроэлектромеханическая система, нелинейное эллиптическое уравнение, двусторонние приближения, инвариантный конусный отрезок, положительное решение.

Введение

Актуальность исследования. Современные исследования микросистемной техники (МСТ) сконцентрированы на создании и применении разнообразных методов математического моделирования, целью которых является описание механических (деформация), электромагнитных (проводимость, диэлектрическая и магнитная проницаемость), оптических и других свойств [5]. Создаваемые микроразмерные устройства широко применяются в различных областях науки и техники: авиационной технике, автомобилестроении, робототехнике, атомной энергетике, системах связи, медицине.

Для наиболее точных и сложных моделей основными методами решения являются численные методы, требующие проведения большого объема вычислений на ЭВМ. Эти методы позволяют добиться хорошего количественного и даже качественного результата в описании модели, однако они не лишены недостатков. Так, не всегда удается оценить погрешность полученных численных решений. Свободными от этого недостатка являются численные методы, которые дают итерационную последовательность, имеющую двустороннюю сходимость. Это позволяет апостериорно оценить погрешность приближенного решения на каждом шаге итерационного процесса.

Таким образом, разработка и совершенствование существующих средств математического моделирования и численного анализа задач, возникающих при проектировании МСТ, является актуальной научной задачей.

В данной работе рассмотрена проблема математического моделирования и численного анализа первой краевой задачи для нелинейного эллиптического урав-

нения второго порядка. Эта задача служит математической моделью работы одного микросистемного устройства. Для ее решения предлагается применить метод последовательных приближений с двусторонним характером сходимости к точному решению задачи.

Цель и задачи исследования. Целью настоящего исследования является разработка и программная реализация метода численного анализа задачи математического моделирования процессов в простейших микросистемных устройствах.

Для достижения поставленной цели необходимо:

- сформулировать задачу математического моделирования и численного анализа процессов в микросистемном устройстве;
- разработать метод решения задачи расчета микросистемного устройства;
- провести вычислительные эксперименты для различных параметров модели;
- провести анализ адекватности полученного решения.

Настоящая работа продолжает исследования, начатые в [2, 3].

1. Постановка задачи

Применение МСТ для различных областей физики обусловило разработку микроэлектромеханических (МЭМС), микрооптоэлектромеханических (МОЭМС), микроакустоэлектромеханических (МАЭМС) и других систем.

Микроактюатор (составная часть МЭМС) – это устройство, которое преобразует энергию в управляемое движение. Электростатический актюатор является одним из самых распространенных типов актюаторных элементов МСТ и состоит из подвижного и неподвижного электродов. В качестве подвижного электрода выступают упругие диэлектрические пластины и мембраны, которые покрываются тонкой металлизированной пленкой. Принцип действия данных актюаторов основан на возникновении электростатической силы между подвижным и неподвижным электродами (рис. 1). При подаче отклоняющего напряжения между электродами возникает электростатическое взаимодействие и подвижный электрод притягивается к неподвижному, вследствие чего может произойти слипание электродов. Обратный процесс называется устойчивым состоянием системы.

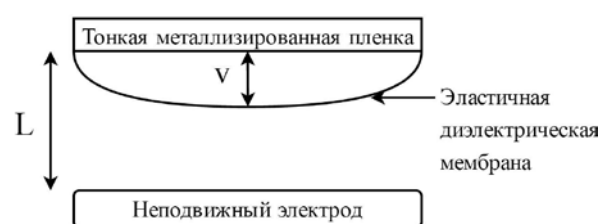


Рис. 1. Схема работы простейшей электростатической МЭМС

Рассмотрим механическую систему, состоящую из упругого тела и приложенных к нему внешних «мертвых» сил, т.е. сил, сохраняющих величину и направление при деформациях системы; тело считаем закрепленным таким образом, что его перемещения как жесткого целого исключены [1].

Полная потенциальная энергия такой консервативной силы в нагруженном состоянии определяется суммой

$$E = R + W ,$$

где R – потенциальная энергия деформации тела; W – потенциал внешних сил.

Электростатические методы активации устройств основаны на законе Кулона [9]: сила взаимодействия неподвижных зарядов q_1 и q_2 прямо пропорциональна произведению зарядов и обратно пропорциональна квадрату расстояния между ними, который в приведенных единицах измерения имеет вид

$$F = \frac{q_1 \cdot q_2}{r^2} .$$

В общем случае, когда электрические заряды распределены неравномерно по двум электродам, расстояние между электродами равно

$$r = L + v(\mathbf{x}) ,$$

где $L > 0$ – расстояние между двумя электродами при отсутствии деформаций мембраны; $v(\mathbf{x})$ – величина деформации мембраны.

Потенциал внешних сил определяется равенством

$$W = - \int_{\Omega} \frac{a \cdot V^2}{L + v(\mathbf{x})} d\mathbf{x} ,$$

здесь $a > 0$ – константа, которая определяет относительную диэлектрическую проницаемость среды; Ω – ограниченная область в \mathbf{R}^2 .

Если в системе присутствуют упругие деформации, то $v \neq 0$.

Потенциальная энергия деформации – это энергия, которая накапливается в теле при его упругой деформации.

В соответствии с законом сохранения энергии без учета её рассеивания (диссипации) потенциальная энергия деформации численно равна работе внешних сил, затраченной при упругой деформации тела. Известно, что потенциальная энергия деформаций состоит из суммы потенциальной энергии изменения формы и потенциальной энергии изменения объема.

Потенциальная энергия изменения формы – это энергия, накапливаемая за счет изменения формы элементарного объема, которая определяется как

$$P = \int_{\Omega} \frac{T}{2} |\nabla v|^2 d\mathbf{x} ,$$

где $T > 0$ – константа напряжения.

Потенциальная энергия изменения объема – это энергия, накапливаемая за счет изменения объема рассматриваемого элементарного объема (одинакового изменения всех его размеров без искажения формы), которая равна

$$Q = \int_{\Omega} \frac{D}{2} |\Delta v|^2 d\mathbf{x} ,$$

где $D = \frac{2 \cdot h^3 \cdot Y}{3(1 - \nu^2)}$; h – толщина мембраны; Y – модуль Юнга; ν – коэффициент Пуассона.

Таким образом, полная потенциальная энергия системы равна

$$E = P + Q + W = \int_{\Omega} \left\{ \frac{T}{2} |\nabla v|^2 + \frac{D}{2} |\Delta v|^2 - \frac{a \cdot V^2}{L + v} \right\} d\mathbf{x} .$$

По теореме Лагранжа состояние равновесия консервативной механической системы устойчиво тогда и только тогда, когда её полная потенциальная энергия минимальна [7]. Необходимое условие минимума функционала E (уравнение Эйлера-Остроградского) имеет вид

$$T \Delta v - D \Delta^2 v = \frac{a \cdot V^2}{(L + v)^2} \text{ в } \Omega , \quad (1)$$

$$v|_{\partial\Omega} = 0 . \quad (2)$$

Поскольку нижний электрод тонкий, то его толщиной можно пренебречь относительно его размеров ($D = 0$), и задача (1), (2) примет вид

$$T \Delta v = \frac{a \cdot V^2}{(L + v)^2} \text{ в } \Omega ,$$

$$v|_{\partial\Omega} = 0 .$$

Тогда после замены $v = -u$ получим нелинейную краевую задачу

$$-\Delta u = \frac{f(x_1, x_2)}{(L - u)^2} \text{ в } \Omega ,$$

$$u|_{\partial\Omega} = 0 ,$$

где $f(x_1, x_2) = \frac{a \cdot V^2}{T}$.

Пусть $L = 1$, $f(x_1, x_2) = \lambda$. Тогда получим задачу

$$-\Delta u = \frac{\lambda}{(1 - u)^2} \text{ в } \Omega , \quad (3)$$

$$0 < u < 1 \text{ в } \Omega, u|_{\partial\Omega} = 0. \quad (4)$$

2. Метод численного анализа

Для решения задачи (3), (4) воспользуемся методом двусторонних приближений [4, 6].

На конусе K неотрицательных в $C(\bar{\Omega})$ функций введем в рассмотрение нелинейное операторное уравнение

$$u = Tu,$$

где

$$Tu = \lambda \int_{\Omega} \frac{G(\mathbf{x}, \mathbf{s})}{(1-u(\mathbf{s}))^2} ds,$$

$G(\mathbf{x}, \mathbf{s})$ – функция Грина первой краевой задачи для оператора $-\Delta$.

Рассмотрим некоторые свойства оператора T .

Оператор T является непрерывным и монотонным на конусе K , так как

$$\forall u_1, u_2 \in K \quad u_1 \leq u_2 < 1 \Rightarrow Tu_1 \leq Tu_2.$$

Ввиду монотонности оператора T построим инвариантный конусный отрезок

$$\langle v_0, w_0 \rangle = \{u \mid v_0 \leq u \leq w_0\},$$

$$T \langle v_0, w_0 \rangle \subset \langle v_0, w_0 \rangle.$$

Для этого рассмотрим схему последовательных приближений

$$u_{n+1}(\mathbf{x}) = \lambda \int_{\Omega} \frac{G(\mathbf{x}, \mathbf{s})}{(1-u_n(\mathbf{s}))^2} ds, \quad n = 0, 1, 2, \dots \quad (5)$$

Положим $v_0 = 0$, $w_0 = \beta = \text{const} \in (0, 1)$. Тогда получим

$$w_1(\mathbf{x}) = \lambda \int_{\Omega} G(\mathbf{x}, \mathbf{s}) \frac{ds}{(1-\beta)^2} = \frac{\lambda u_0(\mathbf{x})}{(1-\beta)^2},$$

$$v_1(\mathbf{x}) = \lambda \int_{\Omega} G(\mathbf{x}, \mathbf{s}) ds = \lambda u_0(\mathbf{x}),$$

где $u_0(\mathbf{x}) = \int_{\Omega} G(\mathbf{x}, \mathbf{s}) ds$.

Необходимо выбором β достичь выполнения условия

$w_1 \leq w_0$, т.е. $\frac{\lambda u_0(\mathbf{x})}{(1-\beta)^2} \leq \beta$. Значит, β определяется системой неравенств

$$\lambda \cdot \max_{\mathbf{x} \in \bar{\Omega}} \int_{\Omega} G(\mathbf{x}, \mathbf{s}) ds \leq \beta(1-\beta)^2, \quad 0 < \beta < 1. \quad (6)$$

Пусть Ω – круг радиуса R . Функция Грина для круга радиуса R имеет вид

$$G(\mathbf{x}, \mathbf{s}) = \frac{1}{2\pi} \ln \frac{1}{r_{\mathbf{x}\mathbf{s}}} - \frac{1}{2\pi} \ln \frac{R}{\rho r_{\mathbf{x}\mathbf{s}^1}},$$

где $\mathbf{x} = (x_1, x_2)$, $\mathbf{s} = (s_1, s_2)$; точки \mathbf{s}^1 – точки, симметричные относительно окружности радиуса R ; $r_{\mathbf{x}\mathbf{s}}$, $r_{\mathbf{x}\mathbf{s}^1}$ – расстояние между точками \mathbf{x} и \mathbf{s} , \mathbf{x} и \mathbf{s}^1 соответственно; $\rho = \sqrt{s_1^2 + s_2^2}$.

Тогда $\max_{\mathbf{x} \in \bar{\Omega}} u_0(\mathbf{x}) = \frac{R^2}{4}$. Значит, система неравенств

(6) для определения β принимает вид

$$\frac{\lambda R^2}{4} \leq \beta(1-\beta)^2, \quad 0 < \beta < 1. \quad (7)$$

В работе [8] было доказано, что задача (3), (4) в единичном круге в \mathbf{R}^2 имеет при $0 < \lambda < \frac{4}{9}$ един-

ственное положительное решение, при $\frac{4}{9} \leq \lambda < 0,593\dots$

существует несколько положительных решений, а при $\lambda \geq 0,593\dots$ – задача решений не имеет. Таким образом, задача (3), (4) в круге радиуса R имеет

единственное решение при $\lambda \in \left(0, \frac{4}{9R^2}\right)$.

Вследствие монотонности оператора T справедливы неравенства $0 = v_0 \leq v_1 \leq u^* \leq w_1 \leq w_0 = \beta$.

Продолжая этот процесс, получаем

$$0 = v_0 \leq v_1 \leq \dots \leq v_n \leq \dots \leq u^* \leq$$

$$\leq \dots \leq w_n \leq \dots \leq w_1 \leq w_0 = \beta.$$

Итак, имеет место следующая теорема.

Теорема. Оператор $Tu = \lambda \int_{\Omega} \frac{G(\mathbf{x}, \mathbf{s})}{(1-u(\mathbf{s}))^2} ds$, где Ω –

круг радиуса R , имеет при $\lambda \in \left(0; \frac{4}{9R^2}\right)$ единствен-

ную неподвижную точку на конусном отрезке $\langle v_0, w_0 \rangle$, $w_0 = \beta$, $v_0 = 0$, причем β определяется неравенствами (7). Последовательные приближения, формируемые по правилу (5), двусторонне сходятся к неподвижной точке оператора T .

3. Результаты численного анализа

Вычислительный эксперимент для задачи (3), (4) был проведен в круге радиуса $R = 1$. Тогда рассматриваемая задача эквивалентна нелинейному интегральному уравнению

$$u(\mathbf{x}) = \lambda \int_{\Omega} \frac{G(\mathbf{x}, \mathbf{s})}{(1-u(\mathbf{s}))^2} ds \quad (8)$$

и имеет единственное решение при $\lambda \in \left(0; \frac{4}{9} = 0,4\right)$.

Для вычислительного эксперимента возьмем $\lambda = 0,3$. Тогда итерационный процесс решения уравнения (8) строим по схеме

$$u^{(k+1)}(x) = 0,3 \cdot \int_{\Omega} \frac{G(x,s)}{(1-u^{(k)}(s))^2} ds, \quad k = 0, 1, 2, \dots \quad (9)$$

Для определения β воспользуемся соотношениями (7). Получим $0,090710 \leq \beta \leq 0,663889$.

На рис. 2 представлены верхние (непрерывная линия) и нижние приближения (пунктирная линия) при $n = 0, 1, 2, 3, 4, 5$ в сечении $x_2 = 0$.

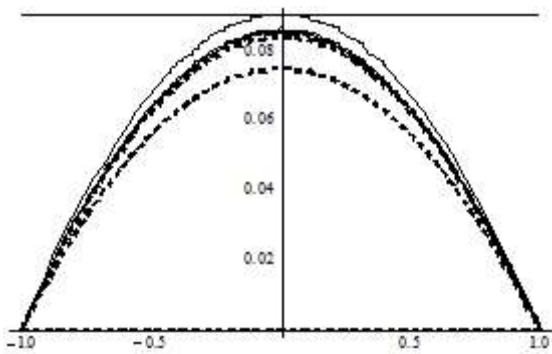


Рис. 2. Графики $w_n(x_1, 0)$ и $v_n(x_1, 0)$ при $n = 0, 1, 2, 3, 4, 5$

В табл. 1 приведены значения $w_n(x)$ и $v_n(x)$ при $n = 0, 1, 2, 3, 4, 5$ в точках области $\bar{\Omega}$ с полярными координатами (ρ_i, φ_j) , где $\rho_i = 0,2i$, $\varphi_j = \frac{\pi j}{10}$, $i = \overline{0, 4}$, $j = \overline{0, 5}$ (значения в остальных четвертях симметричны).

Как видно из табл. 1, в каждой из выбранных точек $v_n(x) \leq w_n(x)$ значения $w_n(x)$ убывают, а значения $v_n(x)$ – возрастают.

Поскольку

$$v_5 \leq u^* \leq w_5, \quad \max_{x \in \bar{\Omega}} (w_5(x) - v_5(x)) = 0,5 \cdot 10^{-5},$$

то с точностью $0,25 \cdot 10^{-5}$ имеем

$$u^* \approx u_5 = \frac{w_5 + v_5}{2}.$$

В табл. 2 приведены значения приближенного решения $u_5(x)$ с точностью $\varepsilon = 0,25 \cdot 10^{-5}$ в точках области $\bar{\Omega}$ с полярными координатами (ρ_i, φ_j) , где $\rho_i = 0,25i$,

$i = \overline{0, 2}$, $\varphi_j = \frac{\pi j}{10}$, $j = \overline{0, 5}$. Поверхность приближенного

решения $u_5(x)$ и его линии уровня представлены на рис. 3 и 4 соответственно.

Значения нормы приближенного решения задачи (3), (4) с точностью $\varepsilon = 0,25 \cdot 10^{-5}$ в зависимости от параметра λ представлены в табл. 3 и на рис. 5.

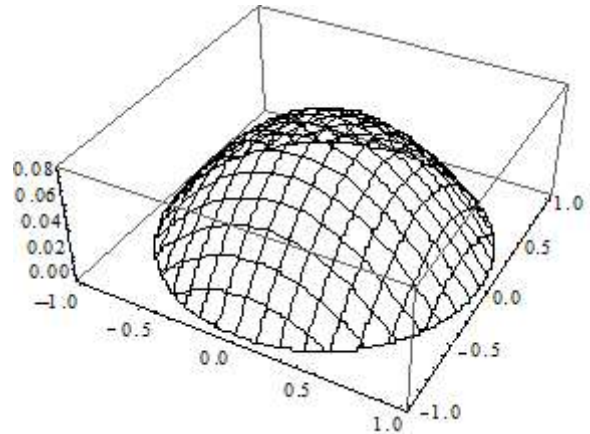


Рис. 3. Поверхность приближенного решения $u_5(x)$

Полученные результаты были доложены на XX Международном молодежном форуме «Радиоэлектроника и молодежь в XXI веке» (Харьков, ХНУРЭ, 19 – 21 апреля 2016) и на 19 Всеукраинской (14 Международной) студенческой научной конференции по прикладной математике и информатике „СНКПМІ-2016” (Львов, ЛНУ им. И.Франко, 14 – 15 апреля 2016) [2, 3].

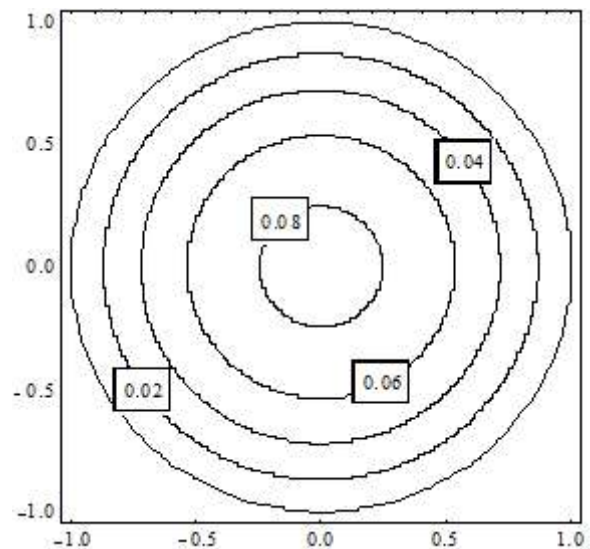


Рис. 4. Линии уровня приближенного решения $u_5(x)$

Таблица 1. Значения $w_n(\mathbf{x})$ и $v_n(\mathbf{x})$ в точках области $\bar{\Omega}$, $n = 0, 1, 2, 3, 4, 5$

$\rho \backslash \varphi$	0	$\frac{\pi}{10}$	$\frac{\pi}{5}$	$\frac{3\pi}{10}$	$\frac{2\pi}{5}$	$\frac{\pi}{2}$	n
0	0,090710	0,090710	0,090710	0,090710	0,090710	0,090710	0
	0,090710	0,090710	0,090710	0,090710	0,090710	0,090710	1
	0,086424	0,086424	0,086424	0,086424	0,086424	0,086424	2
	0,085720	0,085720	0,085720	0,085720	0,085720	0,085720	3
	0,085626	0,085626	0,085626	0,085626	0,085626	0,085626	4
	0,085614	0,085614	0,085614	0,085614	0,085614	0,085614	5
0,25	0,090710	0,090710	0,090710	0,090710	0,090710	0,090710	0
	0,084816	0,084832	0,084849	0,084848	0,084847	0,084824	1
	0,080543	0,080560	0,080578	0,080580	0,080582	0,080562	2
	0,079895	0,079912	0,079929	0,079931	0,079932	0,079912	3
	0,079809	0,079827	0,079844	0,079845	0,079847	0,079827	4
	0,079798	0,079816	0,079833	0,079835	0,079836	0,0798162	5
0,5	0,090710	0,090710	0,090710	0,090710	0,090710	0,090710	0
	0,067997	0,068044	0,068090	0,068098	0,068106	0,068034	1
	0,063980	0,064026	0,064072	0,064086	0,064100	0,064037	2
	0,063485	0,063530	0,063576	0,063589	0,063602	0,063538	3
	0,063422	0,063467	0,063513	0,063526	0,063539	0,063476	4
	0,0634139	0,063459	0,063505	0,063518	0,0635313	0,063468	5
0	0	0	0	0	0	0	0
	0,075000	0,075000	0,075000	0,075000	0,075000	0,075000	1
	0,084250	0,084250	0,084250	0,084250	0,084250	0,084250	2
	0,085438	0,085438	0,085438	0,085438	0,085438	0,085438	3
	0,085590	0,085590	0,085590	0,085590	0,085590	0,085590	4
	0,085609	0,085609	0,085609	0,085609	0,085609	0,085609	5
0,25	0	0	0	0	0	0	0
	0,070126	0,070140	0,070154	0,070153	0,070152	0,070133	1
	0,078864	0,078581	0,078598	0,078599	0,078600	0,078581	2
	0,079640	0,079657	0,079674	0,079676	0,079677	0,079657	3
	0,079777	0,079794	0,079811	0,079813	0,079814	0,079795	4
	0,079794	0,079812	0,079829	0,079830	0,079832	0,079812	5
0,5	0	0	0	0	0	0	0
	0,056221	0,056259	0,056297	0,056304	0,056310	0,056251	1
	0,062513	0,062557	0,062602	0,062615	0,062627	0,062565	2
	0,063299	0,063344	0,063389	0,063340	0,063416	0,063352	3
	0,063398	0,063444	0,063489	0,063502	0,063516	0,063452	4
	0,063411	0,063456	0,063502	0,063515	0,063528	0,063465	5

Таблица 2. Значения приближенного решения $u_5(\mathbf{x})$ в точках области $\bar{\Omega}$

$\rho \backslash \varphi$	0	$\frac{\pi}{10}$	$\frac{\pi}{5}$	$\frac{3\pi}{10}$	$\frac{2\pi}{5}$	$\frac{\pi}{2}$
0	0,085612	0,085612	0,085612	0,085612	0,085612	0,085612
0,25	0,079796	0,079814	0,079831	0,079832	0,079834	0,079814
0,5	0,063412	0,063458	0,063504	0,063517	0,063530	0,063467

Таблица 3. Значения нормы приближенного решения в зависимости от параметра λ

λ	$\ u\ _C$	λ	$\ u\ _C$	λ	$\ u\ _C$
0,02	0,005038	0,22	0,060318	0,42	0,128525
0,04	0,010153	0,24	0,066437	0,44	0,136425
0,06	0,015348	0,26	0,072687	0,46	0,144587
0,08	0,020631	0,28	0,079076	0,48	0,153037
0,10	0,026001	0,30	0,085611	0,50	0,161804
0,12	0,031462	0,32	0,092304	0,52	0,170920
0,14	0,037020	0,34	0,099164	0,54	0,180429
0,16	0,042678	0,36	0,106202	0,56	0,190377
0,18	0,048446	0,38	0,113431	0,58	0,200825
0,20	0,054323	0,40	0,120867	0,592	0,207367

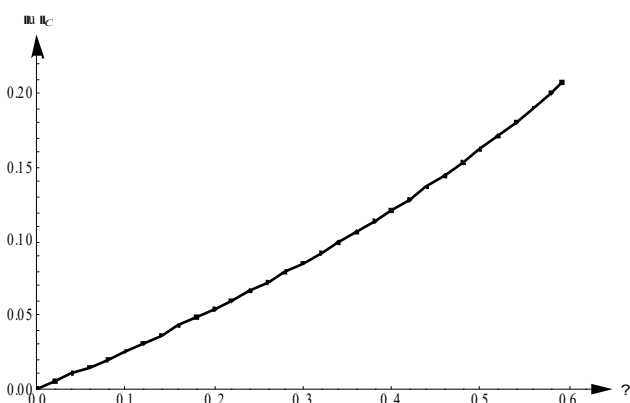


Рис. 5. Значения нормы приближенного решения в зависимости от параметра λ

Выводы

Впервые построены двусторонние приближения к решению первой краевой задачи для нелинейного эллиптического уравнения, возникающего при моделировании электростатических микроэлектромеханических систем. В ходе выполнения исследований также был разработан программный продукт в пакете Mathematica 10, с помощью которого проведен ряд вычислительных экспериментов. Результаты работы могут найти применение в научных исследованиях по физике, химии, биологии, медицине. Этим определяется научная новизна и практическая значимость работы.

Литература: 1. Алфутов Н.А. Основы расчета на устойчивость упругих систем. М.: Машиностроение, 1978. 312 с. 2. Кончаковская О.С. Численный анализ одной нелинейной краевой задачи, моделирующей микроэлектромеханическую систему // Радиоэлектроника и молодёжь в XXI веке: Материалы XX Международного молодёжного форума, 19 – 21 апреля 2016 г. Харьков: ХНУРЭ, 2016. Т. 7. С. 88 – 89. 3. Кончаковська О.С. Чисельний аналіз однієї нелінійної крайової задачі, яка моделює мікроелектромеханічну систему // Студентська наукова конференція з прикладної математики та інформатики: Наукові праці XIX Всеукраїнської (XIV Міжнародної) конференції, 14 – 15 квітня 2016 р. Львів: ЛНУ ім. І. Франка, 2016. С. 78 – 79. 4. Красносельский М.А. Положительные решения операторных уравнений. М.: ГИФМЛ, 1962. 394 с. 5. Му-

хуров Н.И., Ефремов Г.И. Электромеханические микроустройства. Минск: Беларус. навука, 2012. 257 с. 6. Онойцев В.И., Хуродзе Т.А. Нелинейные операторы в пространствах с конусом. Тбилиси: Изд-во Тбилис. ун-та, 1984. 246 с. 7. Циглер Г. Основы теории устойчивости конструкций. М.: Мир, 1971. 192 с. 8. Y. Guo and N. Ghoussoub. On the partial differential equations of electrostatic MEMS devices: stationary case. Submitted (2005). 9. F. Lin, Y. Yisong. Nonlinear non-local elliptic equation modeling electrostatic actuation. A Proceedings of The Royal Society. 05/2007; 463(2081): 1323 – 1337.

Transliterated bibliography: 1. Alfutov N.A. Osnovy rascheta na ustojchivost' uprugih sistem. M.: Mashinostroenie, 1978. 312 p. 2. Konchakovskaja O.S. Chislennyj analiz odnoj nelinejnoj kraevoj zadachi, modelirujushhej mikrojelektromehaničeskiju sistemu // Radiojelektronika i molodjozh' v XXI veke: Materialy XXI Mezhdunarodnogo molodjoznogo foruma, 19 – 21 aprelya 2016 g. Har'kov: KhNURE, 2016. T. 7. P. 88 – 89. 3. Konchakovska O.S. Chysel'nyj analiz odnijei' nelinejnoj' krajovoi' zadachi, jaka modeljuje mikroelektromehaničnu sistemu // Students'ka naukova konferencija z prykladnoi' mate-matyky ta informatyky: Naukovi praci XIX Vseukrai'ns'koi' (XIV Mizhnarodnoi') konferencii', 14 – 15 kvitnja 2016 r. L'viv: LNU im. I. Franka, 2016. P. 78 – 79. 4. Krasnosel'skij M.A. Polozhitel'nye reshenija operatornyh uravnenij. M.: GIFML, 1962. 394 p. 5. Muhurov N.I., Efremov G.I. Jelektromehaničeskie mikroustrojstva. Minsk: Belarus. navuka, 2012. 257 p. 6. Opojcev V.I., Hurodze T.A. Nelinejneje operatory v prostranstvah s konusom. Tbilisi: Izd-vo Tbilis. un-ta, 1984. 246 p. 7. Cigler G. Osnovy teorii ustojchivosti konstrukcij. M.: Mir, 1971. 192 p. 8. Y. Guo and N. Ghoussoub. On the partial differential equations of electrostatic MEMS devices: stationary case. Submitted (2005). 9. F. Lin, Y. Yisong. Nonlinear non-local elliptic equation modeling electrostatic actuation. A Proceedings of The Royal Society. 05/2007; 463(2081): 1323 – 1337.

Поступила в редколлегия 30.04.2016

Рецензент: д-р физ.-мат. наук, проф. Литвин О.Н.

Кончаковская Оксана Сергеевна, студентка гр. СА-12-1 факультета прикладной математики и менеджмента ХНУРЭ. Научные интересы: математическое моделирование и вычислительная математика, программирование. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. (057) 7021436.

Сидоров Максим Викторович, канд. физ.-мат. наук, доцент каф. прикладной математики ХНУРЭ. Научные интересы: математическое моделирование, численные методы, математическая физика, теория R-функций и её приложения, стохастический анализ и его приложения. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. (057) 7021436.

Konchakovskaya Oksana Sergeevna, student of Faculty of Applied Mathematics and Management KhNURE. Research interests: mathematical modeling and computational mathematics, programming. Address: Ukraine, 61166, Kharkov, Lenin Ave, 14, phone +38 (057) 7021436.

Sidorov Maxim Victorovich, Ph.D. in Physics and Maths, associate professor, associate professor of Department of Applied Mathematics KhNURE. Research interests: mathematical modeling, numerical methods, mathematical physics, R-functions theory and its applications, stochastic analysis and its applications. Address: Ukraine, 61166, Kharkov, Lenin Ave, 14, phone +38 (057) 7021436.



ЕКСПЕРТНА СИСТЕМА «ІНВЕСТИЦІЙНА ПРИВАБЛИВІСТЬ ОБЛАСНИХ ЦЕНТРІВ УКРАЇНИ»

СІЗОВА Н.Д., ПЕТРОВА О.О., КАМАРДІН А.С.

Пропонується розробка експертної системи мовою логічного програмування Visual Prolog для оцінки інвестування в регіони з використанням методу експертних оцінок.

Ключові слова: інвестування, метод експертних оцінок, експертна система.

Вступ

На початку 80-х років у дослідженнях зі штучного інтелекту сформувався самостійний напрям, що одержав назву «експертні системи» (ЕС). Основним призначенням ЕС є розробка програмних засобів, які при рішенні задач, важких для людини, одержують результати, що не уступають по якості й ефективності розв'язкам, одержаним людиною – експертом [1]. ЕС використовуються для рішення так званих неформалізованих задач, загальним для яких є те, що: задачі не можуть бути задані в числовій формі;

висновки не можна виразити в термінах точно визначеної цільової функції; не існує алгоритмічного розв'язку задачі; якщо алгоритмічний розв'язок є, то його не можна використовувати через обмеження ресурсів (час, пам'ять).

Експертні системи – це клас систем штучного інтелекту, призначених для отримання, накопичення, коригування знання, що надається експертами з деякої предметної області для отримання нового знання, яке дозволяє вирішувати певні задачі, віднесені до класу неформалізованих, слабо структурованих, пояснюючи хід їх рішення [2–3]. Неформалізовані задачі мають помилковість, неповноту, неоднозначність і суперечливість як вихідних даних, так і знань про задачу, яка розв'язується.

Технологічно експертна система – це пакет програм, здатний за допомогою методів штучного інтелекту аналізувати факти, що представляються користувачем, досліджувати ситуацію, процес, надавати рекомендації.

Серед функцій ЕС відмічають, як правило, такі:

- імітацію діяльності кваліфікованого експерта;
- надання допомоги недостатньо кваліфікованим фахівцям у їх діяльності в певній предметній галузі;

– компенсацію недостатньої кількості експертів у конкретній предметній галузі;

– зняття небажаних наслідків надмірної спеціалізації людини завдяки нагромадженню експертних знань;

– ефект навчання, зумовлений набуттям користувачем досвіду за період роботи з системою.

Актуальність. Більшість ЕС не цілком придатні для широкого використання. Якщо користувач не має деякого досвіду роботи з цими системами, у нього можуть виникнути серйозні труднощі. Багато ЕС доступні лише тим експертам, які створюють їх бази знань. Тому паралельно потрібно розробляти відповідний користувацький інтерфейс, який би забезпечив кінцевому користувачеві властивий йому режим роботи.

На сьогодні актуальним питанням є залучення достатнього обсягу іноземного капіталу, оскільки це стратегічно важливий напрямок подальшого розвитку будь-якої країни. Для України це один з елементів розвитку її економіки [4–5]. Приплив іноземних інвестицій може забезпечити в майбутньому повноцінне функціонування суб'єктів господарювання, зокрема обласних центрів, підвищить конкурентоспроможність національного виробництва та забезпечить збалансованість розвитку економіки країни, її галузей та регіонів.

Кожний інвестор стає перед вибором об'єкта інвестування, загальної суми інвестування та частки в окремому секторі народного господарства країни. Для прийняття правильного рішення необхідно враховувати ряд факторів, в тому числі інвестиційний клімат в регіонах і країні.

Розробники запропонованої ЕС ставили за мету полегшення прийняття рішення інвестором при виборі регіону України для розвитку інвестиційних планів.

Мета дослідження – розробка ЕС для визначення інвестиційної привабливості обласних центрів України на основі експертних оцінок.

Для досягнення поставленої мети сформульовані такі завдання:

- аналіз сучасного стану інвестування в господарюючі суб'єкти;
- створення алгоритму оцінки інвестиційної привабливості обласних центрів;
- використання експертних оцінок для побудови експертної системи;
- побудова ЕС для оцінки інвестиційної привабливості регіонів.

1. Аналіз попередніх досліджень

Однією з найбільш актуальних проблем сучасного етапу розвитку економіки кожної країни є залучення інвестицій в реальний сектор економіки. В роботі [6] відмічена необхідність грамотного формування інвестиційного плану та вибору стратегії інвестування з

урахуванням того, що кожен окремих регіон має певні розвинені сектори економіки [7].

Основними причинами, які зумовлюють необхідність використання інвестицій, є відновлення наявної матеріально-технічної бази, нарощування обсягів виробництва, освоєння нових видів діяльності та ін.

Інвестиційна діяльність будь-якого господарюючого суб'єкта спрямована на економічне зростання і максимізацію ринкової вартості підприємства. При цьому ефективне вирішення таких завдань неможливе без розробки і створення відповідного комп'ютерного програмного забезпечення інвестиційного проектування.

Аналіз існуючих публікацій показав приклади розробки експертних систем, які допомагають у вирішенні питання інвестиційного характеру. Одна з систем [8] призначена для експерта, який буде інвестиційну характеристику міста на основі великої кількості економічних показників та надає користувачу найбільш привабливий проект для інвестування.

Запропонована авторами ЕС може бути використана і недостатньо досвідченим спеціалістом у економічній теорії інвестиційної привабливості регіонів. Тому було поставлено завдання розробки програмного продукту для самого інвестора. Особливістю програми є те, що користувач додає факти, які були надані експертом, а програма працює із самим інвестором на основі цих фактів для прийняття рішення відносно інвестиційної привабливості конкретного міста або регіону. Користувачеві не потрібно самому знаходити показники для оцінки інвестиційного стану міста або регіону, що долає обмеження, які можуть бути пов'язані зі знанням економічної теорії.

2. Основний матеріал

ЕС – це програмний засіб, що використовує експертні знання для забезпечення вискоєфективного рішення неформалізованих задач у вузькій предметній області. Однією з характеристик ЕС є те, що вона застосовує досвід мислення кваліфікованих експертів у даній області знань, що приводить до точних, творчих та ефективних рішень. Крім того для ЕС характерна наявність прогностичних можливостей, системи можуть видавати відповіді на поведінку в конкретній ситуації і показувати, як зміняться ці відповіді у нових ситуаціях. Це дає змогу користувачеві оцінити можливий вплив нових фактів та зрозуміти, як вони пов'язані з рішенням.

Для аналізу інвестиційної привабливості пропонується використання методу експертних оцінок. Сутність експертних методів полягає в тому, що для прийняття рішень (оцінок явищ) залучаються висококваліфіковані спеціалісти, які мають необхідну професійну освіту, досвід та професійну інтуїцію. Експерти, як правило, виконують роль радників осіб, яким надано право приймати рішення. Робота експертів полягає в проведенні інтуїтивно-логічного аналізу проблеми, нерідко з кількісною оцінкою тверджень

та наступною формальною обробкою результатів. В розробленій ЕС знання експертів реалізовано у вигляді фактів та правил мовою логічного програмування Visual Prolog. Логічне програмування будується не за допомогою деякої послідовності абстракцій і перетворень, а на основі абстрактної моделі, що ніяк не пов'язана з якимось типом машинної моделі. Ідея використання можливостей теорії предикатів першого порядку – одна з головних переваг мови Visual Prolog для комп'ютерних наук взагалі та штучного інтелекту зокрема [9–11]. Програмування на Visual Prolog полягає у визначенні відносин та у постановці питань, які стосуються цих відносин.

У запропонованій ЕС за допомогою логічних запитань користувачеві видаються повідомлення про те, в яке місто йому бажано робити інвестиції.

При роботі ЕС були виділені основні критерії, за якими виконувався аналіз, серед них можна виділити:

- кожне місто або регіон має власні сфери інвестування, які в програмі описані у вигляді правил;
- розроблені правила, за допомогою яких програма запам'ятовує відповіді користувача і завдяки цьому може знайти рішення задачі;
- існує критерій, за яким програма розуміє, що дане місто або регіон не підходить користувачеві, та обираються критерії інших міст.

Інтерпретатор намагається уніфікувати аргументи відповідей з аргументами фактів та правил бази знань і у випадку співпадіння повертає результат правила.

Експертна система інвестиційної привабливості обласних центрів є системою, за допомогою якої може бути визначене інвестиційно привабливе місто.

Показники розраховуються за спеціальними формулами згідно з параметрами, які вираховує експерт (таблиця).

Таким чином, показник інвестиційної привабливості території є чистою економічною віддачею від вкладення інвестицій, яка визначається як розмір отриманого від інвестування доходу за мінусом втрат, викликаних проявом ризиків.

Розроблена програма побудована таким чином, що в неї вже вбудована база даних з обласними центрами країни, де для кожного міста на основі методу експертних оцінок експерт з власних міркувань і власного досвіду обрав найбільш важливі показники, проаналізував їх динаміку розвитку та сформував висновки стосовно інвестиційного клімату в означеному регіоні у вигляді пріоритетних сфер народного господарства з визначеною відсотковою ставкою доцільності вкладення капіталу інвестора в це місто. До таких характеристик можна віднести на прикладі Харкова важку промисловість, освіту та інші галузі.

Показник	Формула
Підсумкові показники інвестиційної привабливості регіону	$ИП_k = X * ИП_M + Y * ИП_O$ <p>ИП_k – комплексний показник інвестиційної привабливості регіону; ИП_M – показник інвестиційної привабливості території регіону; ИП_O – показник інвестиційної привабливості галузі регіону</p>
Показник інвестиційної привабливості регіону на територіальному рівні	$ИП_{m(o)} = EO_{m(o)} * (1 - PB_{m(o)})$ <p>EO_{m(o)} – показник, який характеризує рівень дохідності інвестованих коштів на територіальному рівні; PB_{m(o)} – показник ризику вкладення коштів на територіальному рівні.</p>
Економічна віддача	$EO_p = П / Y$ <p>EO_p = показник економічної віддачі регіону; П – сальдований фінансовий результат діяльності організації регіону, отриманий в аналізованому періоді; Y – обсяг інвестицій в основний капітал, вкладених в регіон в попередньому періоді.</p>

Користувачеві на питання треба відповідати тільки по – ні або yes – так. В результаті діалогу інвестору буде запропоноване лише те місто, яке повністю відповідає його вимогам.

Логіка програми базується на двох принципах:

– інвестор не має за мету інвестувати у конкретну галузь міста. Він дивиться на місто в цілому, а потім, якщо він обирає деяке місто, то він інвестує в нього гроші, але сам не розподіляє їх по галузях. За це відповідає міська рада, яка краще розуміється на слабких та сильних сторонах міста;

– інвестор має за мету інвестувати у конкретну галузь міста, але хоче знати про характеристики інших галузей. У цьому випадку експертна система дає повну інформацію користувачеві про вибране ним місто.

```
run(no) :- check(n).

run(yes) :- write("Устраивает ли вас одна из характеристик города:","\n"),
city(X),!, write("Рекомендуем вам город ",X, "\n"), nl,nl,check(no).

run(yes):- write("Нет подходящего города для инвестирования."), check(no).

check(no) :- retractall(_,_dbaseom),readchar(_).
```

Рис. 1. Правило для початку роботи з розробленою ЕС

В розробленій ЕС користувач додавати та видаляти факти про місто не має змоги, так як він виступає у ролі інвестора, який користується вже готовими фактами. Це зроблено для того, щоб інвестор не міг випадково замінити дані. Для редагування експертних оцінок необхідно змінювати код програми.

На початку роботи ЕС користувачеві–інвестору надаються характеристики міста або регіону. Інвестор повинен відповісти на запитання: «Чи є ці характеристики прийнятні для нього чи ні?». Програма по черзі пропонує характеристики міст, які є в списку, та якщо якась характеристика користувачеві не підходить, то програма не продовжує роботу з даним містом, а переходить до іншого. Фрагмент лістингу початку роботи з програмою наведено на рис. 1.

В запропонованій ЕС підготовлена та реалізована конкретна стратегія, яка заснована на експертних оцінках, що передбачає різні експертні бали для різних груп показників з застосуванням факторів для відображення ступеня важливості аргументів у процесі виведення висновків стосовно інвестування в регіон, який розглядається. На рис. 2 наведено правило для визначення інвестиційної привабливості м. Харків (рис.2).

```
city("Харьков"):-
positive("Промышленость развита на
65/100"),
positive("Развито тяжелое
машиностроение 80/100"),
```

Рис.2 Фрагмент лістинга програми для визначення інвестиційної привабливості м. Харків

Висновок

Перевагою використання експертного підходу є можливість адаптації показників і факторів, що вивчаються, для потреб конкретних інвесторів або поглиблений аналіз привабливості певних галузей.

На основі результатів експертів щодо розвинутих сфер народного господарства обласних центрів країни розроблена експертна система для визначення інвестиційної привабливості міста. Розглянуті основні функції, які мають бути у програмі, такі як: вибір між характеристиками міста, відповіддю на питання про інвестиційну привабливість міста та повний опис характеристик, завдяки яким програма видає результат.

До переваг цієї експертної системи можна віднести:

– рішення, отримані за допомогою розробленої експертної системи, є “прозорими”, тобто можуть бути пояснені користувачеві на якісному рівні (на відміну від рішень, отриманих за допомогою числових алгоритмів, і особливо від рішень, отриманих статистичними методами);

– експертна система здатна поповнювати свої знання в ході діалогу з експертом;

– необов’язковість глибоких економічних знань у теорії інвестування та можливість вибору інвестиційно привабливого міста за допомогою готових експертних оцінок.

Таким чином, основою ЕС є сукупність знань, яка структурується для спрощення процесу прийняття рішень. ЕС «Інвестиційна привабливість обласних центрів України» може за прийнятний час знайти відповідний до запитів інвестора розв’язок та може бути використана як один із модулів загальної експертної системи, призначеної для дослідження питань інвестиційної привабливості з інших предметних областей.

Література: 1. Джексон П. Введение в экспертные системы. М.: Мир, 2002. 305 с. 2. Що таке експертна система, її функції? [Електронний ресурс]. Режим доступу: <http://www.virtual.ks.ua/essays-term-papers-and-diplomas/3506-what-is-an-expert-system-and-its-functions.html>. 3. Нейлор К. Как построить свою экспертную систему. М.: Энергоатомиздат, 1991. 288 с. 4. Левченський Д. Л. Суть та економічна природа інвестицій та інвестиційного процесу // Механізм регулювання економіки. 2011. №2. С. 131–139. 5. Грідасов В. М. Інвестування / В.М. Грідасов, С.В. Кривченко, О.Є. Ісаєва. К.: Центр навч. літ-ри, 2004. 164 с. 6. Петрова О. О., Камардін А.С. Аналіз інвестиційної привабливості // Міжнародна науково-практична конференція «Фінанси, аудит та менеджмент: аналіз тенденцій та науково-економічний розвиток», м. Львів, 15–16 квітня, 2016. С. 22–24. 7. Сізова Н.Д., Оніщук В.О. Моделювання інвестиційної привабливості з використання таксометричного аналізу // Матеріали Міжнародної науково-практичної конференції. Одеса: ОНУ ім. І.І. Мечнікова, 2016. С. 241–244. 8. Буценко Е.В. Разработка экспертной системы инвестиционного проектирования // Экономические исследования. [Електронний ресурс]. Режим доступу: <http://cyberleninka.ru/article/n/razrabotka-ekspertnoy-sistemy-investitsionnogo-proektirovaniya>. 9. Адаменко А., Кучуков А. Логическое программирование и Visual Prolog. – СПб:БХВ. Петербург, 2003. 990 с. 10. Братко И. Программирование на языке «Prolog» для искусственного интеллекта. М.:Мир,1990. 315с. 11. Дж. Малпас. Реляционный язык Prolog и его применение. М.: Наука, 1990. 304 с.

Transliterated bibliography:

1. Dzhekson P. Vvedenie v ekspertnyie sistemyi. M.: Mir, 2002. 305 s.

2. Shcho take ekspertna sistema, yiyi funktsiyi? [Elektronnyy resurs]. Rezhym dostupu: <http://www.virtual.ks.ua/essays-term-papers-and-diplomas/3506-what-is-an-expert-system-and-its-functions.html>

3. Neylor K. Kak postroit svoyu ekspertnuyu sistemu. M.: Energoatomizdat, 1991. 288 s.

4. Levchens'kyi D. L. Sut' ta ekonomichna pryroda investytsiy ta investytsiynoho protsesu //Mekhanizm rehulyuvannya ekonomiky. 2011. #2. S. 131–139.

5. Hrydasov V. M. Investuvannya / V.M. Hrydasov, S.V. Kryvchenko, O.Ye. Isayeva. K.: Tsentri navch. lit-ry, 2004. 164 s.

6. Petrova O. O., Kamardin A.S. Analiz investytsiynoyi pryvablyvosti //Mizhnarodna naukovo-praktychna konferentsiya «Finansy, audyt ta menedzhment: analiz tendentsiy ta naukovo-ekonomichnyy rozvytok», m. L'viv, 15–16 kvitnya, 2016. S. 22–24.

7. Sizova N.D., Onishchuk V.O. Modelyuvannya investytsiynoyi pryvablyvosti z vykorystannya taksometrychnoho analizu // Materialy Mizhnarodnoyi naukovo-praktychnoyi konferentsiyi. Odesa: ONU im. I.I. Mechnikova, 2016. S. 241–244.

8. Butsenko E.V. Razrabotka ekspertnoy sistemyi investitsionnogo proektirovaniya// Ekonomicheskie issledovaniya. [Elektronnyy resurs]. Rezhim dostupu: <http://cyberleninka.ru/article/n/razrabotka-ekspertnoy-sistemyi-investitsionnogo-proektirovaniya>.

9. Adamenko A., Kuchukov A. Logicheskoe programmirovaniye i Visual Prolog. SPb:BHV. Peterburg, 2003. 990 s.

10. Bratko I. Programmirovaniye na yazyike «Prolog» dlya iskusstvennogo intellekta. M.:Mir,1990. 315s.

11. Dzh. Malpas. Relyatsionnyiy yazyik Prolog i ego primeneniye. M.: Nauka, 1990. 304 s.

Надійшла до редколегії 30.04.2016

Рецензент: д-р техн. наук, проф. Гіль М.І.

Сізова Наталья Дмитрівна, докт. фіз.-мат. наук, професор каф. економічної кібернетики та інформаційних технологій Харківського національного університету будівництва та архітектури. Наукові інтереси: математичне моделювання, обчислювальні методи, математична фізика, механіка суцільних середовищ. Адреса: Україна, 20322, Харків, вул. Сумська, 40, тел. 38 050 400 92 64.

Петрова Олена Олександрівна, канд. техн. наук, доцент каф. економічної кібернетики та інформаційних технологій Харківського національного університету будівництва та архітектури. Наукові інтереси: штучний інтелект, експертні системи. Адреса: Україна, 20322, Харків, вул. Сумська, 40, тел. +38 098 8499 076.

Камардін Антон Сергійович, студент Харківського національного університету будівництва та архітектури. Наукові інтереси: вища математика, комп'ютерні науки, програмування. Захоплення та хоббі: гра на гітарі. Адреса: Україна, 61204, Харків, пр. Олександрівський, 184, кв.63, тел.+38 063 9519489.



МЕТОД ЭФФЕКТИВНОГО КОДИРОВАНИЯ РАЗДЕЛЕННЫХ ПО КОМБИНИРОВАННОЙ СХЕМЕ АЭРОФОТОСНИМКОВ

БАРАННИК В.В., СТАСЕВ С.Ю.,
ТАРНОПОЛОВ Р.В.

Предлагается метод эффективного кодирования по комбинированной схеме. Устраняется недостаток снижения минимальной границы массива верхнего квантового уровня. Указывается порядок вычисления значений для квантованных элементов массива раздельного представления компонент трансформанты. Обосновывается снижение временных затрат на доставку видовых изображений для комбинированной схемы кодирования.

Ключевые слова. Аэромониторинг, обработка видовых изображений, структурное кодирование, избыточность представления изображений.

Keywords. Aeromonitoring, processing of views images, structured coding, redundancy of image representation.

1. Введение

В настоящее время происходит активное внедрение систем наблюдения в структуру управления ведомственными организациями, особенно в Вооруженных Силах Украины, с применением беспилотных летательных аппаратов [1]. Целью аэромониторинга является получение видовых изображений местности с помощью фотографирования. Ценность содержимого данных видовых изображений выдвигает требования к снижению временных затрат на обработку и доведение данных аэрофотоснимка при необходимости соответствия восстановленного изображения исходному [2]. Для математического описания обработки аэрофотоснимков исходные изображения представляют двумерный сигнал с некоторым объемом информации [3].

В таких технологиях обработки аэрофотоснимков на базе статистического кодирования (кодов переменной длины), как JPEG, JPEG2000, значение информационной интенсивности потока данных определяется постфактум [4]. При альтернативном выборе в качестве технологии обработки структурного кодирования, служебная информация о векторе оснований при формировании вектора-столбца взвешенных коэффициентов позволяет определить максимальное

значение и соответственно длину кода-номера. Также данная схема кодирования формирует меньшее количество разрядов на кодовое представление массивов данных [5]. Значение кода при этом вычисляется на основе аналитического выражения. Для этого используются только значения элементов массива данных и значения компонент вектора оснований. Это приводит к образованию спектрального раздельно-нормированного пространства (СРП) [6].

В случае обработки аэрофотоснимка с сильно насыщенными фрагментами, которые характеризуются повышенной информационной интенсивностью, увеличиваются временные затраты на доставку данных.

При этом динамический диапазон компонент трансформанты характеризуется неравномерностью распределения. В результате этого появляется возможность квантования компонент по величинам значений в массивы раздельного представления (МРП).

Поэтому *цель работы* заключается в создании метода обработки аэрофотоснимков на основе структурного кодирования с выполнением следующих требований:

- квантование на основе выделения двухградационности диапазонов представления элементов трансформанты с помощью их матрицы признаков;
- уменьшение комбинаторной избыточности в ходе одномерного кодирования в двумерном пространстве по блочной схеме;
- устранение дополнительных затрат служебной информации.

2. Разработка метода эффективного кодирования

Первый механизм предлагаемого метода технологии компактного представления основан на квантовании массивов раздельного представления в двумерном пространстве с матрицей признаков.

1. На основе разброса $\Delta_{k\ell}$ значений элементов МРП определяется порог квантования. При этом применяется следующая формула:

$$S(x')_{\text{пор}} = \left(\sum_{k=1}^{\beta} \sum_{\ell=1}^{\beta} \Delta_{k\ell} \right) / \beta^2.$$

2. Производится квантование МРП. Принадлежность элементов МРП по массивам квантующих уровней определяется согласно следующему правилу:

– при выполнении соотношения для диапазона элементов $x'_{k\ell}$:

$$\Delta_{k\ell} \leq S(x')_{\text{пор}},$$

элемент $x'_{k\ell}$ изначально соотносят к области нижнего квантующего уровня $x'_{k\ell} \in X^{(0)}$;

– в обратном случае элемент $x'_{k\ell}$ предварительно соотносят к массиву верхнего квантующего уровня $x'_{k\ell} \in X^{(1)}$.

3. Для устранения недостатка, связанного со снижением минимальной границы в массиве верхнего квантованного уровня вплоть до нулевого уровня, когда диапазон значений элемента $x'_{k\ell}$ МРП будет выше порогового значения квантования $S(x')_{\text{пор}}$, а само значение элемента будет ниже порогового значения [2], предлагается сформировать матрицу признаков для элементов массива отдельного представления. Значение элементов матрицы определяется следующей системой неравенств:

$$\text{matrix}_{k\ell} = \begin{cases} 0, & \rightarrow \Delta_{k\ell} > S(x')_{\text{пор}} \vee x'_{k\ell} \geq S(x')_{\text{пор}}; \\ 1, & \rightarrow \Delta_{k\ell} > S(x')_{\text{пор}} \ \& \ x'_{k\ell} < S(x')_{\text{пор}}, \end{cases} \quad (1)$$

где $\text{matrix}_{k\ell}$ – элемент матрицы признаков $\text{Matrix}^{(2)}$ на позиции $(k; \ell)$.

При этом значение элементов матрицы признаков $\text{Matrix}^{(2)}$ на позиции $(k; \ell)$ передается декодеру при выполнении условия $\Delta_{k\ell} > S(x')_{\text{пор}}$.

4. Элементы массивов нижнего $x^{(0)}$ и верхнего $x^{(1)}$ квантованного уровня могут находиться на разных позициях в МРП. Для этого на следующем этапе обработки из полученных элементов создаются наиболее полные массивы для упрощения процесса кодирования. В процессе формирования подмассивов нижнего и верхнего квантованного уровня необходимо обращать внимание на следующие особенности, которые выражаются в том, что:

– заполнение массивов производится в направлении строк, так как это определяется направлением кодирования;

– при заполнении строки элементом из другой строки, который характеризуется большим значением, необходимо пересчитать разброс значений элементов для всей строки.

Это приводит к образованию двух составляющих массива отдельного представления $X^{(0)}$ и $X^{(1)}$.

5. Производится пересчет диапазона $\Delta_{k\ell}^{(0)}$ значений элементов массива нижнего квантованного уровня с учетом применения матрицы признаков. Расчет при этом происходит согласно следующей системе уравнений:

$$\Delta_{k\ell}^{(0)} = \begin{cases} \Delta_{k\ell}, & \rightarrow \text{matrix}_{k\ell} = 0; \\ S(x')_{\text{пор}}, & \rightarrow \text{matrix}_{k\ell} = 1. \end{cases} \quad (2)$$

6. Производится обработка массива верхнего квантованного уровня. При этом определяются минимальные значения v_i для каждой строки массива верхнего квантованного уровня, т.е.

$$v_i = \min_{1 \leq j \leq \beta} \{X_{ij}^{(1)}\}, \quad i = \overline{1, \alpha}. \quad (3)$$

При этом образуется динамический диапазон элементов массива $X^{(1)}$ в отдельном пространстве согласно следующей формуле:

$$\text{dist}_{ij} = \Delta_{ij}^{(1)} - v_i, \quad (4)$$

где dist_{ij} – разность между максимальным $\Delta_{ij}^{(1)}$ и минимальным v_i значениями в i -й строке массива $X^{(1)}$ верхнего квантованного уровня.

По итогам расчетов формируется массив $\bar{X}^{(1)}$ верхнего квантованного уровня в дифференциальном пространстве, элементы которого смещены на величину минимального уровня $V^{(2)}$ отдельного пространства:

$$\bar{x}^{(1)}_{ij} = x'_{ij} - v_i. \quad (5)$$

7. Происходит объединение составляющих квантованных уровней в единый массив, в итоге чего формируется композиционное представление квантованного массива отдельного представления в двумерном пространстве. Смысл композиции состоит в распределении элементов массивов $X^{(0)}$, $\bar{X}^{(1)}$ на исходные позиции в массиве X' отдельного представления. При композиции применяется информация о диапазонах $\Delta_{k\ell}$ массива РП. При объединении формируется композиционный массив X'' , $X'' = \{x''_{k\ell}\}$. Элементы данного массива составляют композиционное квантованное число в двумерном пространстве.

Упорядочивание составляющих квантованных уровней в единый массив приводит к увеличению области кодируемых элементов. Результатом этого является большее количество потенциально устраняемой структурно-комбинаторной избыточности.

Второй механизм процесса эффективного кодирования заключается в одномерном кодировании по блочной схеме. Для его выполнения необходимо определить код для композиционных квантованных с матрицей признаков чисел в двумерном пространстве.

Определение значений квантованных с матрицей признаков композиционных чисел в пространстве описывается согласно (2), (4) и (5) следующими системами уравнений для элементов нижнего квантованного уровня, верхнего квантованного уровня со значением матрицы признаков $\text{matrix}_{k\ell} = 1$ и вер-

хнего квантованного уровня со значением матрицы признаков $\text{matrix}_{k\ell} = 0$ соответственно:

$$\Delta_{k\ell} \leq S(x')_{\text{пор}} \Rightarrow \begin{cases} x''_{k\ell} := x_{k\ell}^{(0)}; \\ \Delta_{k\ell}^{(0)} := \Delta_{k\ell}; \end{cases} \quad (6)$$

$$\text{matrix}_{k\ell} = 1 \Rightarrow \begin{cases} x''_{k\ell} := x_{k\ell}'^{(0)}; \\ \Delta_{k\ell}^{(0)} := S(x')_{\text{пор}}; \end{cases} \quad (7)$$

$$\text{matrix}_{k\ell} = 0 \Rightarrow \begin{cases} x''_{k\ell} := x_{k\ell}'^{(1)} - v_i; \\ \Delta_{k\ell}^{(1)} := \text{dist}_{k\ell} := \Delta_{k\ell}^{(1)} - v_i. \end{cases} \quad (8)$$

Здесь $x''_{k\ell}$ - $(k; \ell)$ -й элемент квантованного с матрицей признаков композиционного числа в двумерном пространстве.

Процесс вычисления кода в квантованном с матрицей признаков пространстве выполняется одновременно для элементов двух квантованных уровней с учетом применения матрицы признаков и дифференциального преобразования составляющей верхнего динамического уровня. В результате этого к динамическим диапазонам элементов $x''_{k\ell}$ возможно применение трех типов коррекций относительно системы основной двумерного пространства.

Содержание кода описывает структурную сложность фрагмента аэрофотоснимка. Значение кода будет обратно пропорционально площади статических по яркости элементов фрагмента аэрофотоснимка.

При этом структурная сложность аэрофотоснимка заранее неизвестна. Это приводит к тому, что в процессе обработки разных фрагментов аэрофотоснимка будут сформированы различные значения кодов. В результате этого данные значения будут определяться структурным содержанием композиционных массивов.

При этом для фиксированной длины КМП числа соответствие между длиной его кодового слова и структурной сложностью композиционного квантованного числа может быть нарушено. Это может привести к снижению эффективности компрессии при формировании кодов для отдельных столбцов композиционного массива; к несоответствию восстановленного аэрофотоснимка исходному вследствие недостаточной длины кодового слова для отображения значения кода, который был сформирован сразу для всего композиционного массива. В результате такая обработка приводит к уменьшению коэффициента сжатия или несоответствию восстановленного аэрофотоснимка исходному.

Для устранения зависимости от:

– динамичности содержимого массивов раздельного представления;

– дополнительных затрат служебных данных на маркировку кодовых комбинаций -

предлагается формировать неравномерные кодовые комбинации для композиционных квантованных с матрицей признаков чисел (ККМПЧ), которые характеризуются одинаковой длиной.

При этом ККМП числами будут столбцы массива X'' , а длина неравномерных кодовых комбинаций будет определяться на основе накопленного произведения оснований элементов ККМП числа.

Отсюда значение кода $Z^{(\ell)}$ и длина $|Z^{(\ell)}|_2$ кодовой комбинации будут формироваться для отдельных столбцов и вычисляться согласно следующим этапам алгоритма.

Первый этап. Значение кода определяется согласно следующему выражению:

$$Z_1^{(\ell)} = \begin{cases} x_{1,\ell}^{(0)}, & \rightarrow \Delta_{1,\ell} \leq S(x')_{\text{пор}}; \\ x_{1,\ell}^{(0)}, & \rightarrow \text{mask}_{k\ell} = 1; \\ \bar{x}_{1,\ell}^{(1)}, & \rightarrow \text{mask}_{k\ell} = 0, \end{cases}$$

где $Z_1^{(\ell)}$ – значение ℓ -го кода, состоящего из одного элемента ККМПЧ.

Соответственно накопленное произведение оснований будет определяться основанием одного элемента, т.е.

$$O_1^{(\ell)} = \begin{cases} \Delta_{1,\ell}^{(0)} = \Delta_{1,\ell}, & \rightarrow \Delta_{1,\ell} \leq S(x')_{\text{пор}}; \\ \Delta_{1,\ell}^{(0)} = S(x')_{\text{пор}}, & \rightarrow \text{mask}_{k\ell} = 1; \\ \Delta_{1,\ell}^{(1)} = \text{dist}_{1,\ell}, & \rightarrow \text{mask}_{k\ell} = 0, \end{cases}$$

где $O_1^{(\ell)}$ – значение накопленного произведения для одного элемента ℓ -го ККМП числа.

После этого процесс формирования ℓ -го кода-номера для ККМПЧ продолжается.

Очередной этап заключается в добавлении следующего $(k+1; \ell)$ -го элемента, а именно

$$Z_{k+1}^{(\ell)} = \begin{cases} Z_k^{(\ell)} \Delta'_{k+1,\ell} + x_{k+1,\ell}^{(0)}, & \rightarrow \Delta'_{k+1,\ell} \leq S(x')_{\text{пор}}; \\ Z_k^{(\ell)} S(x')_{\text{пор}} + x_{k+1,\ell}^{(0)}, & \rightarrow \text{mask}_{k\ell} = 1; \\ Z_k^{(\ell)} \text{dist}_{k+1,\ell} + \bar{x}_{k+1,\ell}^{(1)}, & \rightarrow \text{mask}_{k\ell} = 0, \end{cases} \quad (9)$$

где $Z_k^{(\ell)}$, $Z_{k+1}^{(\ell)}$ - значения ℓ -го кода-номера, состоящего соответственно из k и $(k+1)$ -го элементов ККМП числа.

При этом величина $O_{k+1}^{(\ell)}$ накопленного произведения для $(k+1)$ -го количества оснований определяется согласно следующей системе:

$$O_{k+1}^{(\ell)} = \begin{cases} \Delta'_{k\ell} O_{k,\ell}^{(\ell)}, & \rightarrow \Delta'_{k\ell} \leq S(x')_{\text{пор}}; \\ S(x')_{\text{пор}} O_{k,\ell}^{(\ell)}, & \rightarrow \text{mask}_{k\ell} = 1; \\ \text{dist}_{k,\ell+1} O_{k,\ell}^{(\ell)}, & \rightarrow \text{mask}_{k\ell} = 0. \end{cases} \quad (10)$$

где $O_k^{(\ell)}$ – величина накопленного произведения для k -го количества оснований.

Общая схема алгоритма представлена на рисунке.

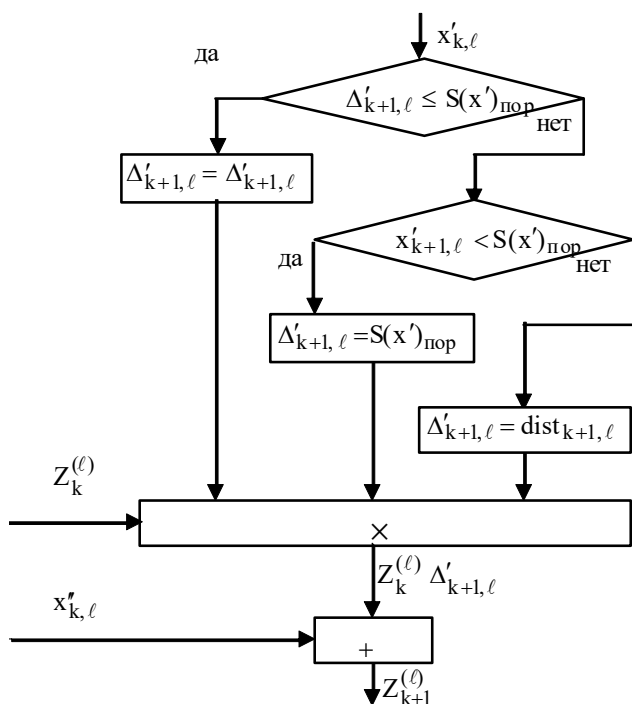


Схема алгоритма формирования кода $Z_{k+1}^{(\ell)}$

Окончание обработки определяется фактом обработки $x''_{\beta,\ell}$ элемента в ℓ -м столбце.

3. Оценка эффективности метода кодирования применительно к изменениям временных затрат на доставку данных

Для оценки эффективности разработанного метода необходимо рассмотреть изменения информационной интенсивности передаваемых данных кодера, а именно передачи матрицы признаков $Matrix^{(2)}$ из (1), минимальных значений v_i для каждой строки массива верхнего квантованного уровня из (3) и изменение длины $|Z^{(\ell)}|_2$ кода $Z^{(\ell)}$ для ℓ -го столбца.

Значение кода для ℓ -го столбца будет равно $Z^{(\ell)}$, а величина накопленного произведения – $O^{(\ell)}$. Из этого длину $|Z^{(\ell)}|_2$ кодовой комбинации, содержащей зна-

чение кода $Z^{(\ell)}$, предлагается определять на основе соотношения:

$$|Z^{(\ell)}|_2 = [\ell \log_2 O^{(\ell)}] + 1 \text{ (бит)}.$$

Вследствие мультипликативного порядка вычисления значения кода $Z^{(\ell)}$ для ℓ -го столбца и его длины $|Z^{(\ell)}|_2$ согласно (9) и (10) уменьшение внутритрансформантной структурной избыточности в абсолютных величинах превосходит затраты на передачу минимальных значений v_i для каждой строки массива верхнего квантованного уровня. Минимальные значения v_i для массива верхнего квантованного уровня в созданном методе передаются только строк. Касательно затрат на передачу матрицы признаков

$Matrix^{(2)}$ стоит отметить их физическое проявление в один бит на каждую компоненту, которая изначально была распределена в массив верхнего квантованного уровня. Передача остальной информации происходит согласно правилу выполнения структурного кодирования [5].

В результате выполнения данного алгоритма предоставляется возможность определить длину кодового слова под код ККМП числа с помощью значений оснований его элементов без применения дополнительной служебной информации. При этом происходит уменьшение внутритрансформантной структурной, вероятностно-статистической избыточности компонент трансформанты аэрофотоснимка, что приводит к уменьшению временных затрат на доведения данных видовых изображений.

4. Выводы

- 1) Разработан метод эффективного кодирования аэрофотоснимков с применением разделения фрагментов аэрофотоснимка по комбинированной схеме.
- 2) Построена концепция квантования массивов раздельного представления в двумерном пространстве с применением матрицы признаков на основе взаимного отношения значений элемента $x'_{k\ell}$ МРП, его диапазона $\Delta_{k\ell}$ и порогового значения квантования $S(x')_{\text{пор}}$.
- 3) Построен алгоритм выполнения одномерного блочного кодирования для образования кода композиционных квантованных с матрицей признаков чисел в двумерном пространстве. Это приводит к увеличению нижней границы раздельного пространства массива верхнего квантованного уровня и уменьшению разброса значений элементов массива, которые относятся к нижнему квантованному уровню. В результате этого достигается уменьшение внутритрансформантной структурной, вероятностно-статистической избыточности компонент трансформанты аэрофотоснимка.
- 4) Выполнено условие касательно недопустимости дополнительных затрат служебной информации, что

суммарно выражается в снижении временных затрат на доведение данных видовых изображений.

Литература: 1. Баранник В.В., Подлесный С.А. Анализ действия кибератак на видеоинформационный ресурс в информационно-телекоммуникационных сетях // АСУ и приборы автоматики. Вып. 169. 2014. С. 16-22. 2. Баранник В.В. Кодирование трансформированных изображений в инфокоммуникационных системах / В.В. Баранник, В.П. Поляков. Х.: ХУПС, 2010. 212 с. 3. Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов / В.Г. Олифер, Н.А. Олифер. СПб.: Питер, 2006. 958 с. 4. Красильников Н.Н. Цифровая обработка изображений. М.: Вузовская книга, 2011. 320 с. 5. Баранник В.В. Метод сжатия изображений на основе неравновесного позиционного кодирования битовых плоскостей / В.В. Баранник, Н.К. Гулак, Н.А.Королева // Радиэлектронні і комп'ютерні системи. Х.: ХНАУ "ХАГ", 2009. Вип. 1. С. 55–61. 6. Баранник В.В. Кодирование трансформированных изображений в инфокоммуникационных системах / В.В. Баранник, В.П. Поляков. Х.: ХУПС, 2010. 212 с. 7. Баранник В.В., Сидоренко Н.Ф., Шинкарев В.В. Метод композиции перфорированных неравновесных чисел // АСУ и приборы автоматики. 2009. Вып. 149. С. 30-36.

Transliterated bibliography:

1. Barannik V.V., Podlesnyj S.A. Analiz dejstvija kiberatak na videoinformacionnyj resurs v informacionno-telekomunikacionnyh setjah // ASU i pribory avtomatiki: nauchno-tehnicheskij sbornik. Vyp. 169. H.: HNURE. 2014. S. 16-22.

2. Barannik V.V. Kodirovanie transformirovannyh izobrazhenij v infokommunikacionnyh sistemah / V.V. Barannik, V.P. Poljakov. H.: HUPS, 2010. 212 s.

3. Olifer V.G. Komp'juternye seti. Principy, tehnologii, protokoly: Uchebnik dlja vuzov. / V.G. Olifer, N.A. Olifer. SPb.: Piter, 2006. 958 s.

4. Krasil'nikov N.N. Cifrovaja obrabotka izobrazhenij. M.: Vuzovskaja kniga, 2011. 320 s.

5. Barannik V.V. Metod szhatija izobrazhenij na osnove neravnovesnogo pozicionnogo kodirovanija bitovyh ploskostej / V.V. Barannik, N.K. Gulak, N.A.Koroleva // Rad?oelektronn? ? komp'jutern? sistemi. H.: HNAU "HA?", 2009. Vip. 1. S. 55–61.

6. Barannik V.V. Kodirovanie transformirovannyh izobrazhenij v infokommunikacionnyh sistemah / V.V. Barannik, V.P. Poljakov. H.: HUPS, 2010. 212 s.

7. Barannik V.V., Sidorenko N.F., Shinkarev V.V. Metod kompozicii perforirovannyh neravnovesnyh chisel // ASU i pribory avtomatiki: nauchno-tehnicheskij sbornik. Vyp. 149. H.: HNURE. 2009. S. 30-36.

Поступила в редколлегию 02.06.2016

Рецензент: д-р техн. наук, проф. Безрук В.М.

Баранник Владимир Викторович, д-р техн. наук, профессор, начальник кафедры боевого применения и эксплуатации АСУ Харьковского университета Воздушных Сил. Научные интересы: обработка и передача информации. Адрес: Украина, 61023, Харьков, ул. Сумская, 77/79.

Тарнополов Роман Викторович, преподаватель кафедры боевого применения и эксплуатации АСУ Харьковского университета Воздушных Сил. Научные интересы: обработка и передача информации. Адрес: Украина, 61023, Харьков, ул. Сумская, 77/79.

Стасев Сергей Юрьевич, аспирант ХНУРЭ. Научные интересы: технологии кодирования и обеспечения информационной безопасности государства. Адрес: Украина, 61023, Харьков, ул. Сумская, 77/79. Email: Barannik_V_V@mail.ru

СИНТЕЗ Q-ТЕСТОВ ПО КУБИТНОМУ ОПИСАНИЮ ФУНКЦИОНАЛЬНОСТЕЙ

TAMER VANI AMER, ЕМЕЛЬЯНОВ И.В.,
ЛЮБАРСКИЙ М., ХАХАНОВ В.И.

Предлагаются методы и аппаратно-программные реализации безусловного параллельного синтеза тестов на основе булевых производных для логических схем в black box форме, заданных кубитным покрытием. Дается теоретическое обоснование применения методов и оценки их эффективностей для широкого класса цифровых схем, имплементируемых в кристаллы программируемых логических устройств. Предлагаются инновационные методы взятия булевых производных и дедуктивного моделирования неисправностей для функциональных элементов, заданных кубитными покрытиями.

1. Реализация логических функциональностей на элементах памяти

Понятие адресного выполнения логических операций, реализованных на элементах памяти LUT в программируемых логических устройствах (PLD), дает потенциальную возможность создавать на кристалле только адресное пространство, максимально технологичное для встроенного восстановления работоспособности всех компонентов, участвующих в формировании функциональности [1-3]. Актуальность создания адресного пространства для всех компонентов подтверждается следующим распределением логики и памяти на кристалле, представленным на рис. 1, где после 2020 года на чипе будет только один процент логики и 99 процентов памяти.

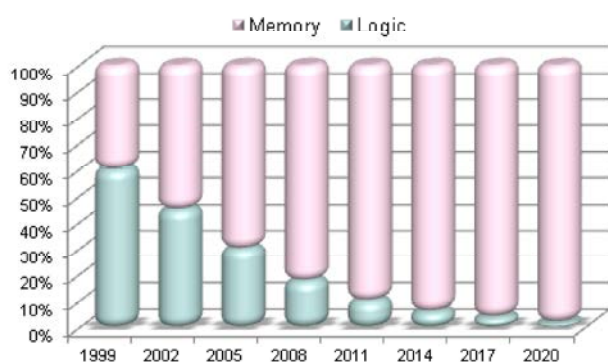


Рис. 1. Распределение памяти и логики на кристалле

Тенденция к увеличению памяти дает возможность встроенного восстановления работоспособности отказавших ячеек за счет выделенных дополнительных ресурсов для их ремонта (spare logic cells). Проблема автономного устранения дефектов (самовосстановления работоспособности) логических элементов связана с отсутствием у них адресов. Но решить ее можно, если связи между элементами логики сделать гибкими с помощью программы описания структуры, помещенной в память, которая соединит логические компоненты в схему. Кроме структуры взаимодействия элементов

память должна содержать порядок их обработки. В случае возникновения дефекта в одном из адресуемых логических элементов система встроенного тестирования восстановит его работоспособность путем переадресации на заведомо исправный аналог из ремонтного запаса. Просто решается проблема повышения качества и надежности цифровых систем на кристаллах путем создания инфраструктуры встроенного тестирования, диагностирования, оптимизации и восстановления работоспособности за счет аппаратной избыточности и уменьшения быстродействия выполнения функциональных операций [2-5].

Цель – существенное повышение быстродействия синтеза тестов и дедуктивной верификации для black box функциональностей логических компонентов с помощью компактных описаний в форме кубитных покрытий и параллельного выполнения минимального числа регистровых логических операций (shift, or, not, nxor).

Задачи:

- 1) Метод генерации тестов для black box функциональностей логических схем на основе использования кубитных покрытий и параллельного выполнения регистровых логических операций (shift, or, not, nxor).
- 2) Метод взятия булевых производных для синтеза тестов на основе использования кубитных покрытий.
- 3) Метод синтеза тестов на основе применения булевых производных, представленных векторами в формате кубитных покрытий.

Сущность исследования – разработка методов генерирования входных тестовых последовательностей и оценки их качества для функциональных логических компонентов путем параллельного выполнения регистровых логических операций (shift, or, not, nxor) над кубитным покрытием и его производными в структуре процессора кубитного моделирования.

Мотивация нового подхода для проектирования компьютерных систем обусловлена появлением облачных сервисов в рамках новой киберкультуры Internet of Things, которая представляет собой специализированные и рассредоточенные в пространстве системы, реализуемые в аппаратуре или в программном продукте. Любой компонент функциональности, равно как и структура системы, представляется векторной формой, упорядоченной по адресам, таблицы истинности, реализуемой с помощью памяти. Логические функции в традиционном исполнении reusable logic не рассматриваются. От этого частично уменьшается быстродействие, но, учитывая, что 94% SoC-кристалла составляет память [2,3], оставшиеся 6% будут имплементироваться в памяти, что не критично для большинства облачных сервисов. Практически для создания эффективных компьютерных структур следует использовать теорию, основанную на вычислительных компонентах высокого уровня абстракции: адресуемая память и транзакция.

Особенность организации данных в классическом компьютере состоит в адресации бита, байта или друго-

го компонента. Адресуемость создает проблему в обработке ассоциации неадресуемых элементов множества, которые не имеют порядка по определению. Решением может быть процессор, где образ универсума из n унитарно кодированных примитивов использует суперпозицию для формирования булеана $|B(A)| = 2^n$ всех возможных состояний [4,12].

Следовательно, структуру данных «булеан» можно рассматривать как детерминированный образ квантового кубита в алгебре логики, элементы которой унитарно кодируются двоичными векторами и обладают свойствами суперпозиции, параллелизма и перепутывания. Это дает возможность использовать предлагаемые кубитные модели для повышения быстродействия анализа цифровых устройств на классических вычислителях, а также без модификации – в квантовых компьютерах, которые появятся через несколько лет на рынке электроники.

Квантовое описание цифровых функциональных элементов. Кубит (n -кубит) есть векторная форма унитарного кодирования универсума из n примитивов для задания булеана состояний 2^{2^n} с помощью 2^n двоичных переменных. Если $n=2$, то 2-кубит задает 16 состояний с помощью четырех переменных, при $n=1$ кубит задает четыре состояния на универсуме из двух примитивов (10) и (01) с помощью двух двоичных переменных (00,01,10,11) [12]. При этом допускается суперпозиция в векторе 2^n состояний, обозначенных примитивами. Синонимом кубита при задании двоичного вектора логической функции является Q-покрытие (Q-вектор) [6,7] как унифицированная векторная форма суперпозиционного задания выходных состояний, соответствующих адресным кодам входных переменных функционального элемента. Формат структурного кубитного компонента цифровой схемы $Q^* = (X, Q, Y)$ включает интерфейс (входные и выходную переменные), а также кубит-вектор Q , задающий функцию $Y = Q(X)$, размерность которого определяется степенной функцией от числа входных линий $k = 2^n$. Новизна кубитной формы заключается в замене неупорядоченных по строкам таблиц истинности функциональных элементов векторами упорядоченных состояний выходов. Например, если функциональный примитив имеет двоичную таблицу, то ему можно поставить в соответствие кубит или Q-покрытие: $Q=(1110)$:

$$\begin{array}{ccc}
 X_1 & X_2 & Y \\
 0 & 0 & 1 \\
 0 & 1 & 1 \\
 1 & 0 & 1 \\
 1 & 1 & 0
 \end{array}
 \rightarrow Q = (1110).$$

Таким образом, дизрапторная идея, положенная в основу исследований, заключается в замене множества вход-выходных соответствий таблицы истинности кубитным вектором адресуемых выходных состояний (рис. 2).

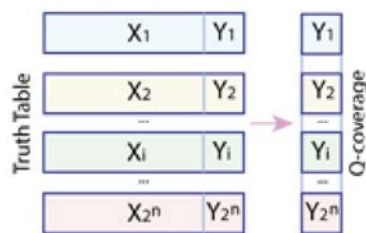


Рис. 2. Таблица истинности и кубитное покрытие

Примитивизм и компактность кубитной векторной формы (Q-coverage) диктует применение только простых параллельных регистровых операций над его содержимым: (not, shift, or, and, xor) для решения всех задач синтеза и анализа цифровых изделий.

2. Кубитный метод синтеза тестов

Предлагается метод синтеза тестов, использующий кубитные векторы или Q-покрытия функциональных примитивов цифровых устройств, который характеризуется компактностью описания данных и параллелизмом выполнения логических операций.

Q-покрытие есть векторная форма описания поведения цифрового устройства, где каждый разряд имеет адрес, формируемый двоичными состояниями его входных переменных:

$$Q = (Q_1, Q_2, \dots, Q_i, \dots, Q_n), Q_i = \{0,1\}, Q_i = Q(i), i = (X_1 X_2, \dots, X_i, \dots, X_n).$$

Q-тест есть векторная форма неявного задания тестовых последовательностей цифрового устройства, где координаты вектора формируют упорядоченную последовательность двоичных наборов, подаваемых на входные переменные по правилу:

$$Q_i = Q(i) = \begin{cases} 1 \rightarrow (X_1 X_2, \dots, X_i, \dots, X_n) = i, \\ 0 \rightarrow (X_1 X_2, \dots, X_i, \dots, X_n) = \emptyset. \end{cases}$$

Другими словами, если координата вектора $Q(i)=1$, то тестовый набор, составленный из двоичных разрядов, формирующих десятичный адрес i , подается на входы устройства. В противном случае, при нулевом значении координаты $Q(i)=0$, такой тестовый набор отсутствует.

Естественно, что размерности Q-покрытия и Q-теста всегда одинаковы, что делает возможным выполнять параллельный анализ их совместного взаимодействия при синтезе тестов для функциональных элементов по правилу [4]: $F \oplus Q \oplus T = 0, T = Q \oplus F$.

Пример 1. Необходимо сгенерировать тест проверки одиночных константных неисправностей для логического элемента 2and, заданного таблицей истинности. Для этого задается таблица влияния одиночных константных неисправностей входных переменных с числом кубов, равным количеству входов. Каждый куб неисправности имеет единицу на координате входной переменной и на выходе, а остальные координаты равны нулю. Это означает, что существует заказ: изменение входной переменной должно вызывать изменение состояния выхода.

Алгоритм синтеза теста по таблице истинности [4]:

1) На первом шаге выполняется покомбинаторная операция между таблицей истинности и каждой строкой матрицы неисправностей, которая генерирует две таблицы, по числу входных переменных, содержащие строки-кандидаты в тест:

$$\begin{bmatrix} X_1 & X_2 & Y \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} F_1 & F_2 & F_Y \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} X_1 & X_2 & \bar{Y}_1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \vee \begin{bmatrix} X_1 & X_2 & \bar{Y}_2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

2) Затем строки двух последних таблиц упорядочиваются по правилу возрастания двоичных кодов входных переменных:

$$\begin{bmatrix} X_1 & X_2 & \bar{Y}_1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \vee \begin{bmatrix} X_1 & X_2 & \bar{Y}_2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} X_1 & X_2 & Y_1' \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} X_1 & X_2 & Y_2' \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

3) После этого сравниваются полученные состояния выходов ($Y_1'Y_2'$) с вектором значений выходов исходной таблицы истинности Y по правилу операции эквивалентности (xor) – если значения одноименных координат двух кубитов равны, то результат сравнения равен 1, в противном случае – 0:

$$\begin{bmatrix} Y \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \oplus \begin{bmatrix} Y_1' \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \vee \begin{bmatrix} Y_2' \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} Y_1^t \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} Y_2^t \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

4) На последнем шаге выполняется операция логического сложения полученных результатов сравнения, которая дает Q-тест или T-вектор, единичные координаты которого идентифицируют только те двоичные входные последовательности, которые следует подавать на входы цифрового устройства для его проверки:

$$\begin{bmatrix} Y_1^t \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \vee \begin{bmatrix} Y_2^t \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} T \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} X_1 & X_2 & Y \\ \cdot & \cdot & \cdot \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

В данном примере Q-тест содержит три единицы 0111, поэтому соответствующие ему входные наборы представлены тремя векторами: 010, 100, 111.

Пример 2 иллюстрирует работу алгоритма [4] синтеза теста по таблице истинности для функционального элемента, имеющего три входных переменных. Здесь также фигурирует таблица истинности, в которой последний столбец Y является кубит-вектором. Столбцы ($F_1F_2F_3F_Y$) формируют матрицу неисправностей, сутью каждой строки которой – одномерная активизация входа на выход. Столбцы ($Y_1'Y_2'Y_3'$) получены в результате xor-взаимодействия таблицы истинности и матрицы неисправностей, которое определяет поведение функциональности при внесении на каждый вход неисправности, инверсной по отношению к ее исправному поведению. Столбцы ($Y_1^tY_2^tY_3^t$) получены путем покомбинаторного сравнения вектора выходных состояний исходной таблицы истинности с наборами ($Y_1'Y_2'Y_3'$), полученными на предыдущем шаге. Вектор T – логически объединяет наборы ($Y_1^tY_2^tY_3^t$) в тест для заданной функциональности:

X_1	X_2	X_3	Y	F_1	F_2	F_3	F_Y	Y_1'	Y_2'	Y_3'	Y_1^t	Y_2^t	Y_3^t	T
0	0	0	1	1	0	0	1	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	0	0	0	0	0
0	1	0	0	0	1	1	1	0	0	0	1	1	1	1
0	1	1	1					1	0	1	1	0	1	1
1	0	0	0					0	0	0	1	1	1	1
1	0	1	1					0	1	1	0	1	1	1
1	1	0	1					1	1	1	1	1	1	1
1	1	1	0					0	0	0	1	1	1	1

Последний столбец представляет собой тест проверки неисправностей внешних входных и выходных переменных функционального элемента $T=(10111111)$. Тест-вектор задает следующие входные наборы, которые следует подать на входы функционального элемента: 0001, 0100, 0111, 1000, 1011, 1101, 1110, чтобы проверить все дефекты заданного класса.

Формальный Q-алгоритм синтеза тестов для константных неисправностей функционального примитива на основе использования Q-покрытия содержит следующие пункты:

- 1) Инвертирование Q-покрытия: $Q_i = \bar{Q}_i, i = \overline{1, 2^n}$.
- 2) Упорядочение инверсного Q-покрытия для каждой из n входных переменных $Q_j = S_j(\bar{Q}), j = \overline{1, n}$. Данная процедура сводится к выполнению операций логического сдвига (shift) на кубитном векторе, каждая из которых имеет собственный алгоритм для рассматриваемой входной переменной, что иллюстрируется следующей схемой для трех входов (рис. 3).

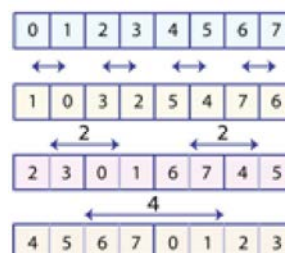


Рис. 3. Операции встречного сдвига на регистрах

Здесь вторая строка адресными индексами задает процедуру обмена данными между соседними координатами кубит-вектора путем встречного сдвига содержимого двух соседних координат, что формирует тест-вектор для первой входной переменной. Третья строка иллюстрирует обмен данными между соседними парами координат кубит-вектора путем встречного сдвига их содержимого, что формирует тест-вектор для второй входной переменной. Четвертая строка задает обмен данными между соседними тетрадами координат кубит-вектора путем встречного сдвига их содержимого, что формирует тест-вектор для третьей входной переменной.

3) Получение T-векторов для каждой из n входных переменных путем сравнения с исходным кубитным покрытием функционального элемента:

$$T_j = Q \oplus \bar{Q}_j, j = \overline{1, n}.$$

4) Получение Q-теста функциональности путем логического объединения T-векторов для каждой входной переменной:

$$T = \bigvee_{j=1}^n T_j.$$

Интегрально алгоритм безусловного синтеза тестов для функциональных элементов, заданных кубитными покрытиями, компактно может быть записан в виде следующей формулы:

$$T = \bigvee_{j=1}^n [Q \oplus \bar{Q}_j \oplus S_j(\bar{Q})].$$

Вычислительная сложность безусловного алгоритма синтеза тестов на основе последовательного использования регистровых логических операций (not – shift – xor – or) для функционального элемента, описанного кубит-вектором, представлена следующим выражением:

$$q = n + n \times 2^n + n + n = n(2^n + 3).$$

Здесь n – число переменных, первое слагаемое определяет вычислительную сложность операции 1) инверсии, второе – 2) логический сдвиг для перестановки состояний координат кубит-вектора относительно каждой из n входных переменных, третье – 3) xor-сравнение полученных тест-векторов с исходным кубит-покрытием функционального элемента, четвертое – 4) объединение тест-векторов для входных переменных.

Абстрагируясь от понятия таблицы истинности, далее предлагаем формальный безусловный алгоритм кубитного синтеза тестов для функциональных примитивов на основе Q-покрытия применительно к ранее рассмотренному примеру:

1) Инвертирование всех разрядов Q-покрытия функционального элемента, имеющего три входных переменных:

Q	1	0	0	1	0	0	1	0
\bar{Q}	0	1	1	0	1	1	0	1

2) Логический сдвиг номеров разрядов инвертированного кубитного вектора в соответствии с последовательностями номеров:

$$Q_1(X_1) = 45670123, Q_2(X_2) = 23016745, Q_3(X_3) = 10325476.$$

Здесь действует простое логическое правило: для первой входной (младшей) переменной выполняется параллельный обмен данными между соседними координатами, для второй входной переменной реализуется обмен данными, но уже между соседними парами координат, для третьей переменной выполняется обмен данными между соседними тетрадами координат вектора адресов.

Обобщение операций сдвига для функциональности, содержащей 4 переменных, представлено на рис. 4. Увеличение количества переменных принципиально не изменяет сущностей сдвига данных: встречный сдвиг двух битов, пар битов, тетрад битов, восьмерок и т.д.

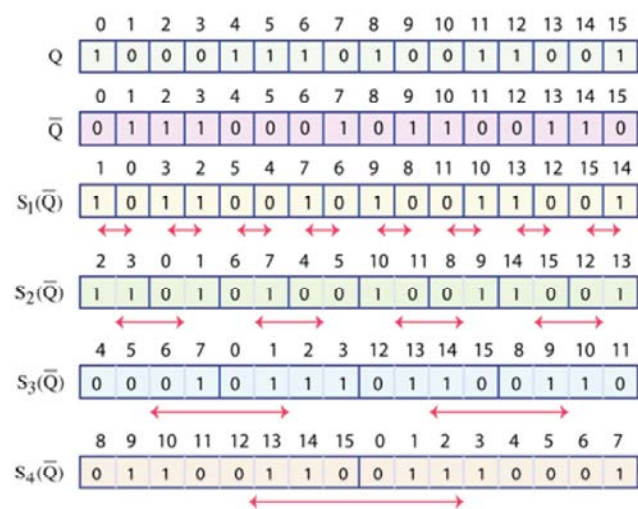


Рис. 4. Операции сдвига для четырех-входовой функциональности

Процедура встречного сдвига данных в инверсном кубитном векторе является самой времязатратной в алгоритме синтеза тестов для функциональных элементов. Поэтому ее быстроедействие будет иметь максимальное значение при аппаратной реализации сдвиговых операций.

Процедурная реализация обмена номерами для формирования тест-векторов проверки неисправностей входных переменных, применительно к рассматриваемому примеру, имеет следующий вид:

Q	0	1	2	3	4	5	6	7
Q_1	1	0	3	2	5	4	7	6
Q_2	2	3	0	1	6	7	4	5
Q_3	4	5	6	7	0	1	2	3

Интересно, что для первой переменной существует простая формула перенумерации ячеек кубит-вектора для синтеза тестов: $j = j + (-1)^j$. Для остальных переменных такой простой зависимости определить пока не удалось.

С учетом введенных правил встречного сдвига ячеек кубит-вектора реализация данного пункта алгоритма для $Q=11010110$ представлена таблицей:

$$\begin{aligned} Q &= 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0 \\ \bar{Q} &= 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1 \\ \bar{Q}_1 &= 1\ 0\ 0\ 1\ 0\ 0\ 1\ 0 \\ \bar{Q}_2 &= 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0 \\ \bar{Q}_3 &= 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0 \end{aligned}$$

Таким образом, каждая переменная делает разбиение кубит-вектора на группы упорядоченных последовательностей. Первая (младшая) переменная создает изменения состояний по правилу: прибавить 1 к текущему четному адресу ячейки: $j=j+1$, отнять 1 у текущего нечетного адреса ячейки: $j=j-1$. Для второй переменной рассматриваются уже пары ячеек, которые обрабатываются по правилу: прибавить 2 к текущему четному адресу пары ячеек: $j=j+2$, отнять 2 у текущего нечетного адреса пары ячеек: $j=j-2$. Для третьей переменной рассматриваются уже тетрады ячеек, которые обрабатываются по правилу: прибавить 4 к текущему четному адресу тетрады ячеек: $j=j+4$, отнять 4 у текущего нечетного адреса тетрады ячеек: $j=j-4$.

3) Сравнение с помощью операции эквивалентности полученных в данном случае трех инвертированных и переупорядоченных Q -векторов с исходным Q -покрытием функционального элемента:

Q	1 1 0 1 0 1 1 0
\bar{Q}_1	0 0 1 0 1 0 0 1
$S_1(\bar{Q})$	1 0 0 1 0 0 1 0
$S_2(\bar{Q})$	1 0 0 0 0 1 1 0
$S_3(\bar{Q})$	0 0 0 1 0 1 1 0
$Q \oplus S_1(\bar{Q})$	1 0 1 1 1 0 1 1
$Q \oplus S_2(\bar{Q})$	1 0 1 0 1 1 1 1
$Q \oplus S_3(\bar{Q})$	0 0 1 1 1 1 1 1

4) Дизъюнкция полученных трех векторов формирует Q -тест, единичные координаты которых определяют тестовые наборы для проверки всех одиночных константных неисправностей внешних входов и выходов:

Q	1 1 0 1 0 1 1 0
\bar{Q}	0 0 1 0 1 0 0 1
$S_1(\bar{Q})$	1 0 0 1 0 0 1 0
$S_2(\bar{Q})$	1 0 0 0 0 1 1 0
$S_3(\bar{Q})$	0 0 0 1 0 1 1 0
$T_1 = Q \oplus S_1(\bar{Q})$	1 0 1 1 1 0 1 1
$T_2 = Q \oplus S_2(\bar{Q})$	1 0 1 0 1 1 1 1
$T_3 = Q \oplus S_3(\bar{Q})$	0 0 1 1 1 1 1 1
$T = T_1 \vee T_2 \vee T_3$	1 0 1 1 1 1 1 1

Учитывая существенность встречных регистровых сдвигов, ниже предлагается универсальный алгоритм получения перестановок кубитных фрагментов в зависимости от номера входной переменной.

Последовательный алгоритм встречного сдвига данных в кубитных покрытиях для получения Q -тестов входных переменных имеет три вложенных цикла:

1) Задание i -номера входной переменной или шага 2^i для встречного сдвига данных в кубите: $i = \overline{1, n}$.

2) Формирование цикла обработки кубита с уже заданным шагом 2^i (2,4,8,16...), кратным степени двойки: $j = \overline{0, 2^n - 1, 2^i}$. В зависимости от номера входной переменной i формируются следующие последовательности индекса $j=f(i)$: [(0,2,4,6...), (0,4,8,12...), (0,8,16,32...)].

3) Задание цикла встречного сдвига данных для пары соседних групп: $t = j, j + 2^{i-1} - 1$. Значения индекса $t=f(i,j)$: обработка кубита первой переменной – (0,0; 2,2; 4,4...), второй переменной – (0,1; 4,5; 8,9...), третьей переменной – (0,3; 8,11; 16,19...). Выполнение операций сдвига посредством использования буферного регистра B (рис. 5):

$$(B_t = Q_{i,t+j}) \rightarrow (Q_{i,t+j} = Q_{i,t}) \rightarrow (Q_{i,t} = B_t)$$

Конец алгоритма сдвига данных в регистрах.

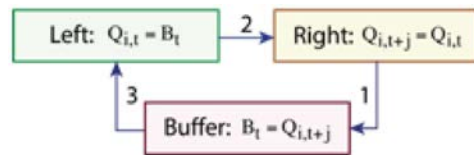


Рис. 5. Встречный сдвиг соседних частей кубита

Вычислительная сложность последовательного алгоритма для обработки кубитного покрытия функциональности, имеющей n входных переменных, равна $q = 3n2^n$.

Пример 3. Построить тест для проверки функциональности, заданной логическим уравнением: $Y = \bar{X}_1 \bar{X}_2 \vee \bar{X}_1 X_2$. Результаты выполнения алгоритма синтеза теста по заданному кубит-вектору представлены в следующем виде:

Q	0 0 1 1
\bar{Q}_1	1 1 0 0
$S_1(\bar{Q})$	0 0 1 1
$S_2(\bar{Q})$	1 1 0 0
$T_1 = Q \oplus S_1(\bar{Q})$	1 1 1 1
$T_2 = Q \oplus S_2(\bar{Q})$	0 0 0 0
$T = T_1 \vee T_2$	1 1 1 1

Результирующий тест содержит 4 входных последовательности, каждая из которых проверяет неисправности для входной переменной X_1 . Однако тест для переменной X_2 не существует, поскольку $T_2=0000$. Это означает, что переменная X_2 не может быть проверена, а значит она не является существенной. Действительно, преобразование исходной функции

$$Y = \bar{X}_1 \bar{X}_2 \vee \bar{X}_1 X_2 = \bar{X}_1 (\bar{X}_2 \vee X_2) = \bar{X}_1$$

в сторону минимизации исключает переменную X_2 , как избыточную или несущественную. Таким образом, метод генерации тестов на основе кубитного покрытия дает дополнительную возможность опреде-

лять существенность переменных и минимизировать (уменьшать размерность) Q-вектор путем уменьшения количества переменных.

Пример 4. Построить тест для проверки функциональности, заданной логическим уравнением: $Y = \bar{X}_1 X_2 \vee X_1 \bar{X}_2$. Результаты выполнения алгоритма синтеза теста по заданному кубит-вектору на основе последовательного выполнения четырех регистровых логических операций (not, shift, nxor, or) представлены в виде:

Q	0 1 1 0
\bar{Q}_1	1 0 0 1
$S_1(\bar{Q})$	0 1 1 0
$S_2(\bar{Q})$	0 1 1 0
$T_1 = Q \oplus S_1(\bar{Q})$	1 1 1 1
$T_2 = Q \oplus S_2(\bar{Q})$	1 1 1 1
$T = T_1 \vee T_2$	1 1 1 1

Полученный тест содержит 4 входных последовательности, каждая из которых проверяет неисправности для входных переменных X_1, X_2 . Таким образом, метод генерации тестов на основе кубитного покрытия позволяет сгенерировать тест для функционального компонента, но не предоставляет инструмента сделать его минимальным.

Пример 5. Синтезировать тест для проверки функциональности, заданной логическим уравнением: $Y = \bar{X}_1 X_2 \bar{X}_3 \vee X_1 \bar{X}_2 \bar{X}_3 \vee X_1 X_2 X_3 \vee \bar{X}_1 X_2 X_3$. Результаты выполнения алгоритма синтеза теста по Q-вектору на основе последовательного выполнения четырех регистровых логических операций (not, shift, nxor, or) представлены в виде:

Q	0 0 1 1 1 0 0 1
\bar{Q}	1 1 0 0 0 1 1 0
$S_1(\bar{Q})$	1 1 0 0 1 0 0 1
$S_2(\bar{Q})$	0 0 1 1 1 0 0 1
$S_3(\bar{Q})$	0 1 1 0 1 1 0 0
$T_1 = Q \oplus S_1(\bar{Q})$	0 0 0 0 1 1 1 1
$T_2 = Q \oplus S_2(\bar{Q})$	1 1 1 1 1 1 1 1
$T_3 = Q \oplus S_3(\bar{Q})$	1 0 1 0 1 0 1 0
$T = T_1 \vee T_2 \vee T_3$	1 1 1 1 1 1 1 1

На рис. 6 показан секвенсор синтеза тестов для функциональных элементов, заданных кубитными покрытиями. Он содержит модуль управления, который распределяет четыре синхроимпульса, создавая цикл генерации теста, включающий 4 параллельные операции, последовательно выполняемые: 0) Начальная стартовая операция, инициируемая сигналом Start, предполагает загрузку в регистр кубитного покрытия. 1) Затем синхроимпульс Clk N активирует выполнение операции инверсии Not над содержимым регистра кубитного покрытия. 2) Синхросигнал Clk S активирует выполнение операций встречного сдвига над содержимым, в данном случае трех, регистров, получая следующие результаты: S_1 (not Q), S_2 (not Q), S_3 (not Q). 3) Затем синхросигнал Clk C инициирует

выполнение параллельных операций сравнения (в данном случае для трех регистров) полученных кандидатов в тест с начальным кубитным покрытием: $nxor(notQ, S_1)$, $nxor(notQ, S_2)$, $nxor(notQ, S_3)$. 4) Синхросигнал Clk U активирует выполнение or-операции над кандидатами в тест, в данном случае реализацию логического объединения содержимого трех регистров:

$$T = or[nxor(notQ, S_1), nxor(notQ, S_2), nxor(notQ, S_3)].$$

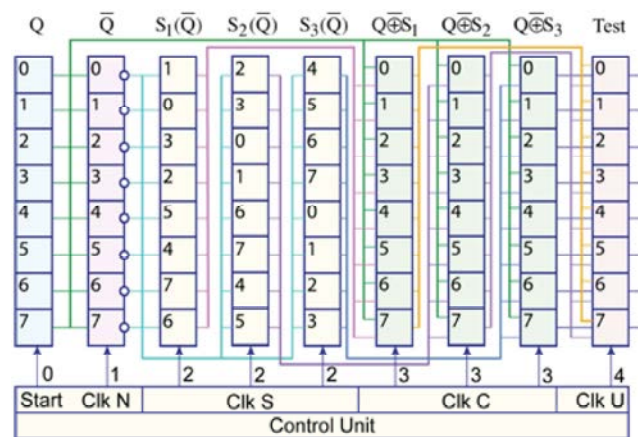


Рис. 6. Секвенсор синтеза тестов

Если не экономить на аппаратуре, то быстродействие тестового генератора можно довести до четырех автоматных тактов $q=4$ параллельного выполнения логических регистровых операций, что дает возможность встраивать секвенсор в BIST-инфраструктуру цифровых систем на кристаллах для online тестирования функциональных элементов. При этом суммарное количество 2^n -разрядных регистров будет равно $N(n)=1+1+n+n+1=(2n+3)$, а общий объем регистровой памяти в битах будет равен $N(R)=(2n+3) 2^n$, где n – число входных переменных функционального элемента.

Сложить навстречу сдвинутые соседние тетрады кубита для вычисления производной по первой переменной. Сложить навстречу сдвинутые соседние пары кубита для вычисления производной по второй переменной. Сложить навстречу сдвинутые соседние биты кубита для вычисления производной по третьей переменной.

3. Вычисление булевых производных для Q-синтеза тестов

Рассмотрим метод взятия булевых производных по кубитному покрытию для создания условий активизации входных переменных при синтезе кубитных тестов. Проведем аналогию между двумя формами булевых функций для взятия производных: аналитической и векторной. Исследование метода выполним на примерах логических функций:

- $f(x) = x_1 \vee x_1 \bar{x}_2$.
- $f(x) = x_1 x_2 \vee \bar{x}_1 x_3$.
- $f(x) = \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3$.

Вопросы, подлежащие решению: 1) Определение производных первого порядка по аналитической и кубитной форме задания логической функции. 2) Верификация полученных условий активизации путем их моделирования на одной из форм описания функциональности. 3) Синтез тестов активизации переменных логической функции на основе вычисления производных.

Пример 6. Определить все производные первого порядка по кубитной форме логической функции $f(x) = x_1 \vee x_1 \bar{x}_2$.

Применение формулы вычисления по аналитическому выражению

$$\frac{df(x_1, x_2, \dots, x_i, \dots, x_n)}{dx_i} \cdot f(x_1, x_2, \dots, x_i = 0, \dots, x_n) \cdot f(x_1, x_2, \dots, x_i = 1, \dots, x_n)$$

определяет булеву производную первого порядка как сумму по модулю два нулевой и единичной остаточных функций.

Для рассматриваемой функции получается:

$$\begin{aligned} \frac{df(x_1, x_2)}{dx_1} \cdot f(0, x_2) \cdot f(1, x_2) \\ \cdot (0 \cdot 0 \bar{x}_2) \cdot (1 \cdot 1 \bar{x}_2) \cdot 0 \cdot 1 \cdot 1 \end{aligned}$$

$$\begin{aligned} \frac{df(x_1, x_2)}{dx_2} \cdot f(x_1, 0) \cdot f(x_1, 1) \\ \cdot (x_1 \cdot x_1 \cdot \bar{0}) \cdot (x_1 \cdot x_1 \cdot \bar{1}) \\ \cdot (x_1 \cdot x_1 \cdot 1) \cdot (x_1 \cdot x_1 \cdot 0) \\ \cdot (x_1 \cdot x_1) \cdot (x_1 \cdot 0) \cdot x_1 \cdot x_1 \cdot 0 \end{aligned}$$

Нулевая величина производной означает отсутствие условий активизации для переменной x_2 , что дает основания считать ее несущественной, следовательно, убрать из числа переменных, формирующих функциональность. Далее предлагаются аналогичные преобразования для кубитного покрытия функции, заданного вектором:

x_1	x_2	Y
0	0	0
0	1	0
1	0	1
1	1	1

= (0011)

Производная по таблице истинности может вычисляться путем поочередного задания всех нулей и единиц в координатах столбцов, соответствующих каждой переменной:

x_1	x_2	Y	Y_1^0	Y_1^1	Y_2^0	Y_2^1	Y_2^2	Y_2^3
0	0	0	0	1	1	0	0	0
0	1	0	0	1	1	0	0	0
1	0	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1	0

Таким образом, производные по первой и второй переменной, записанные в формате кубитного покрытия,

равны: 1111 и 0000. Это означает, что производная по первой переменной равна 1, а по второй равна 0.

Однако такой результат можно получить более формально и технологично, не рассматривая входные наборы таблицы истинности, используя только логические операции встречного сдвига и последующей хог-операции над разрядами кубитного покрытия $\{a, b\} = a \oplus b$, где a, b – соседние подвекторы кубита $Q=(a, b)$:

Y	Y_1'	Y_2'
0	1	0
0	1	0
1	1	0
1	1	0

Иначе, для первой переменной необходимо хог-сложить, сдвинутые относительно друг друга две половинки первого столбца, а результат записать в обе сдвигаемые симметричные области $\{a, b\} = a \oplus b$, $(11, 11) = 00 \oplus 11$. Для второй переменной следует рассматривать пары соседних координат столбца, а общий результат записывать в каждые сдвигаемые симметричные области-биты $\{a, b\} = a \oplus b$: $(0, 0) = 0 \oplus 0 = 0$, $(0, 0) = 1 \oplus 1 = 0$. Таким образом, результат суммирования будет общим для каждой пары взаимодействующих подвекторов, размерность которых определяется номером рассматриваемой переменной, от 0 до $2^n - 1$.

Пример 8. Определить все производные первого порядка по аналитической форме логической функции $f(x) = x_1 x_2 \vee \bar{x}_1 x_3$. Для рассматриваемой функции выполняются следующие вычисления:

$$\begin{aligned} \frac{df(x_1, x_2, x_3)}{dx_1} \cdot f(0, x_2, x_3) \cdot f(1, x_2, x_3) \\ \cdot (0 \cdot x_2 \cdot \bar{0} \cdot x_3) \cdot (1 \cdot x_2 \cdot \bar{1} \cdot x_3) \\ \cdot (0 \cdot 1 \cdot x_3) \cdot (x_2 \cdot 0 \cdot x_3) \\ \cdot x_3 \cdot x_2 \cdot x_2 \bar{x}_3 \cdot \bar{x}_2 x_3 \end{aligned}$$

$$\begin{aligned} \frac{df(x_1, x_2, x_3)}{dx_2} \cdot f(x_1, 0, x_3) \cdot f(x_1, 1, x_3) \\ \cdot \bar{x}_1 x_3 \cdot (x_1 \cdot x_3) \\ \cdot \bar{x}_1 x_3 (x_1 \cdot x_3) \cdot \bar{x}_1 x_3 \overline{(x_1 \cdot x_3)} \\ \cdot (x_1 \cdot \bar{x}_3)(x_1 \cdot x_3) \cdot \bar{x}_1 x_3 \bar{x}_1 \bar{x}_3 \cdot x_1 \end{aligned}$$

$$\begin{aligned} \frac{df(x_1, x_2, x_3)}{dx_3} \cdot f(x_1, x_2, 0) \cdot f(x_1, x_2, 1) \\ \cdot x_1 x_2 \cdot (\bar{x}_1 \cdot x_2) \\ \cdot x_1 x_2 (\bar{x}_1 \cdot x_2) \cdot x_1 x_2 \overline{(\bar{x}_1 \cdot x_2)} \\ \cdot (\bar{x}_1 \cdot \bar{x}_2)(\bar{x}_1 \cdot x_2) \cdot x_1 x_2 x_1 \bar{x}_2 \cdot \bar{x}_1 \end{aligned}$$

Для трех переменных получены 4 условия активизации, которые соответствуют четырем логическим путям в схемной структуре дизъюнктивной формы данной функции.

Вычисление трех производных первого порядка по таблице истинности дает следующий результат:

X ₁	X ₂	X ₃	Y	Y ₁ ⁰	Y ₁ ¹	Y ₁ '	Y ₂ ⁰	Y ₂ ¹	Y ₂ '	Y ₃ ⁰	Y ₃ ¹	Y ₃ '
0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	1	1	1	0	1	1	1	0	0	1	1
0	1	0	0	0	1	1	0	0	0	0	1	1
0	1	1	1	1	1	0	1	1	0	0	1	1
1	0	0	0	0	0	0	0	1	1	0	0	0
1	0	1	0	1	0	1	0	1	1	0	0	0
1	1	0	1	0	1	1	0	1	1	1	1	0
1	1	1	1	1	1	0	0	1	1	1	1	0

Если исключить из рассмотрения таблицу истинности, а использовать кубитное покрытие, то на содержательном уровне процесс вычисления производных будет иметь следующий вид:

Y	Y ₁ '	Y ₂ '	Y ₃ '
0	0	0	1
1	1	0	1
0	1	0	1
1	0	0	1
0	0	1	0
0	1	1	0
1	1	1	0
1	0	1	0

Кубитный метод взятия булевой производной:

- 1) Определить кубитный вектор функциональности.
- 2) Выполнить хог-операцию для сдвинутых навстречу друг другу соседних частей кубита: биты, пары, тетрады.
- 3) Результат записать в обе соседние части: биты, пары, тетрады.

Для рассматриваемого примера выполнение процедуры взятия производной по первой переменной имеет вид: {a,b}=a⊕b, (0110,0110)=0101⊕0011. Для получения производной по второй переменной нужно последовательно хог-сложить соседние пары кубит-вектора Y, а общий результат записать в каждую пару: {a,b}=a⊕b: (00,00)=01⊕01, (11,11)=00⊕11. Для получения производной по третьей переменной необходимо последовательно хог-сложить соседние биты кубит-вектора Y, а общий результат записать в каждый бит соседей: {a,b}=a⊕b: (1,1)=0⊕1, (1,1)=0⊕1, (0,0)=0⊕0, (0,0)=0⊕0.

Естественно, что кубит-производная по любой входной переменной, как вектор, обладает относительной симметрией равенства подвекторов по построению: производная первой переменной имеет симметричное равенство двух тетрад, производная второй переменной имеет симметричное равенство каждых соседних пар, производная третьей переменной имеет симметричное равенство каждых соседних битов.

Пример 8. Определить все производные первого порядка по таблице истинности логической функции трех переменных: $f(x) = \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3$.

Результат взятия производных по таблице истинности [4] приведенной функциональности на основе выполнения операций над столбцами входных переменных представлен в виде:

x ₁	x ₂	x ₃	Y	Y ₂ ⁰	Y ₂ ¹	Y ₂ '
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	1	1	1	0
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	1	0	1	1

· x₂x₃.

x ₁	x ₂	x ₃	Y	Y ₂ ⁰	Y ₂ ¹	Y ₂ '
0	0	0	1	1	0	1
0	0	1	0	0	0	0
0	1	0	0	1	0	1
0	1	1	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	0	1	1
1	1	0	0	1	0	1
1	1	1	1	0	1	1

· $\bar{x}_1 \bar{x}_3$ · x₁ \bar{x}_3 · x₁x₃ · \bar{x}_3 · x₁x₃.

x ₁	x ₂	x ₃	Y	Y ₃ ⁰	Y ₃ ¹	Y ₃ '
0	0	0	1	1	0	1
0	0	1	0	1	0	1
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	1	0	1
1	1	0	0	0	1	1
1	1	1	1	0	1	1

· $\bar{x}_1 \bar{x}_2$ · x₁ \bar{x}_2 · x₁x₂ · \bar{x}_2 · x₁x₂.

Как альтернатива, ниже приведены результаты выполнения процедур взятия производных первого порядка для трех переменных по кубическому покрытию (таблице истинности) функциональности:

X ₁	X ₂	X ₃	Y	Y ₁ ⁰	Y ₁ ¹	Y ₁ '	Y ₂ ⁰	Y ₂ ¹	Y ₂ '	Y ₃ ⁰	Y ₃ ¹	Y ₃ '
0	0	0	1	1	1	0	1	0	1	1	0	1
0	0	1	0	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	1	0	1	0	1	0
0	1	1	0	0	1	1	0	0	0	0	1	0
1	0	0	1	1	1	0	1	0	1	1	0	1
1	0	1	0	0	0	0	0	1	1	1	0	1
1	1	0	0	0	0	0	1	0	1	0	1	1
1	1	1	1	0	1	1	0	1	1	0	1	1

Далее представлен исключительно простой процесс вычисления производных на кубитном покрытии без рассмотрения таблицы истинности, что на формальном уровне имеет следующий вид:

Y	Y ₁ '	Y ₂ '	Y ₃ '
1	0	1	1
0	0	0	1
0	0	1	0
0	1	0	0
1	0	1	1
0	0	1	1
0	0	1	1
1	1	1	1

Интерпретация полученных кубит-производных. Естественно, что производные есть функции, заданные векторами. Они могут быть записаны в аналитической форме (ДНФ) по единичным значениям переменных, формирующих адреса ячеек кубит-вектора:

$$Y_1' = 011 \vee 111 = \bar{X}_1 X_2 X_3 \vee X_1 X_2 X_3 = X_2 X_3 (\bar{X}_1 \vee X_1) = X_2 X_3.$$

$$Y_2' = 000 \vee 010 \vee 100 \vee 101 \vee 110 \vee 111 = \bar{X}_1 \bar{X}_2 \vee X_1.$$

$$Y_3' = 000 \vee 001 \vee 100 \vee 101 \vee 110 \vee 111 = \bar{X}_1 \bar{X}_2 \vee X_1.$$

Минимизация булевых функций, соответствующих производным, приводит к аналитическим выражениям, где отсутствуют переменные, по которым берется производная. Таким образом, все результаты по вычислению производных от трех форм (аналитическая, табличная, векторная) задания функции являются идентичными. Наиболее технологичным является метод взятия производной по кубитному покрытию. Он имеет меньшую вычислительную сложность в силу компактного представления функциональности. Использование аналитической формы предполагает существенное повышение сложности алгоритмов, связанной с применением законов булевой алгебры и минимизации функций, что ограничивает ее применение для решения практических задач.

Для сравнения далее коротко описан метод получения теста $T = [T_{ij}]$, $i = \bar{1, k}$; $j = \bar{1, n}$ комбинационной функциональности, заданной кубическим покрытием или таблицей истинности, который содержит пункты [4]:

$$1) f'(x_i) = f(x_1, x_2, \dots, x_i = 0, \dots, x_n) \oplus f(x_1, x_2, \dots, x_i = 1, \dots, x_n);$$

$$2) T = \bigcup_{i=1}^n [f'(x_i) * (x_i = 0) \vee (x_i = 1)];$$

$$3) T_{ij} = T_{i-1, j} \leftarrow T_{ij} = X; T_{1j} = 1 \leftarrow T_{1j} = X;$$

$$4) T = T \setminus T_i \leftarrow T_i = T_{i-r}, r = \bar{1, i-1}, i = \bar{2, n}.$$

1) Вычисление производных по всем n переменным функциональности путем использования кубического покрытия. 2) Объединение всех условий (векторов) активизации в таблицу, где каждому вектору путем конкатенации (*) ставится в соответствие изменение переменной, по которой была взята производная, что означает удвоение числа тестовых наборов по

отношению к общему количеству (k) условий активизации. 3) Минимизация тестовых векторов путем удаления повторяющихся входных последовательностей. Рис. 7 иллюстрирует таблицы процесса получения теста в соответствии с пунктами 2-3 алгоритма для функциональности $f(x) = \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3$, представленной схемной структурой:

x ₁	x ₂	x ₃	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

x ₁	x ₂	x ₃	Y
0	1	1	0
1	0	0	1
1	1	0	0
1	1	1	1

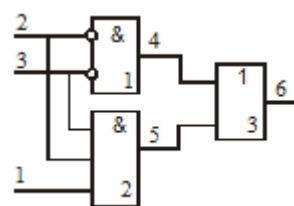


Рис. 7. Получение теста для схемной структуры булевой функции

Как альтернатива упомянутому выше, далее предлагается технологически простой метод синтеза тестов на основе взятия производных по кубитным покрытиям функциональных элементов, без рассмотрения состояний входных переменных.

- 1) Исходное задание логической функциональности кубитным покрытием.
- 2) Выполнение операций встречного сдвига частей кубит-вектора и последующего по координатному хогу суммирования для получения векторов производных для каждой входной переменной.
- 3) Логическое объединение векторов производных, формирующее тест-вектор, равный по размеру кубитному покрытию.
- 4) В случае необходимости получения минимального теста решается задача покрытия (уже на матрице кубит-производных) путем нахождения минимального числа пар единичных координат кубит-вектора всех переменных, где пара единиц должна проверять одиночные константные неисправности каждого входа. Процедура выбора пары единиц в кубит-производной определяется наличием двух представителей, по одному от любой четной и любой нечетной частей вектора.

Далее в таблице представлены результаты синтеза теста для функциональности, заданной уравнением:

$$f(x) = \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3.$$

Данной логической функции от трех переменных соответствует кубитное покрытие: (10001001):

X_1	X_2	X_3	Y	Y'_1	Y'_2	Y'_3	T		X_1	X_2	X_3	Y
0	0	0	1	0	1	1	1					
0	0	1	0	0	0	1	1		X_1	X_2	X_3	Y
0	1	0	0	0	1	0	1		0	1	1	0
0	1	1	0	1	0	0	1		1	1	1	1
1	0	0	1	0	1	1	0		0	1	0	0
1	0	1	0	0	1	1	0		0	0	0	1
1	1	0	0	0	1	1	0		0	0	1	0
1	1	1	1	1	1	1	1					

С учетом выполнения пункта 3 алгоритм синтеза дает минимальный тест проверки входных переменных, который содержит 5 наборов, что представлено столбцом T , а также продублировано в явном виде правой таблицей.

Интерес представляет тот факт, что результат выполнения процедуры взятия производной по кубит-вектору уже содержит тест активизации каждой переменной. Объединенный тест проверяет все константные неисправности входных переменных, а также может быть использован для диагностирования неисправностей, поскольку для существенных входов все производные-векторы будут различными. Фактически взятие производной по переменной на кубит-покрытии формирует Q -тест, не больше и не меньше.

4. Заключение

Ниже представлены формулировки научной новизны и практической значимости описанных исследований.

- 1) Впервые разработан метод и секвенсор безусловного синтеза тестов для функциональных логических компонентов, который характеризуется параллельным выполнением регистровых логических операций (shift, or, not, nxor) над кубитным вектором и его производными, что дает возможность существенно уменьшить время генерирования входных наборов и тестирования устройства в режиме embedded online.
- 2) Впервые разработан метод взятия производных для генерации тестов функциональных компонентов, который характеризуется параллельным выполнением регистровых логических операций (shift, or, not, nxor) над кубитным вектором, что дает возможность существенно уменьшить время генерирования входных наборов и тестирования устройства за счет аппаратной избыточности.
- 3) Практическая значимость исследований заключается в возможности облачной реализации быстродействующего метода синтеза тестов и моделирования неисправностей для функциональных логических компонентов на основе параллельного выполнения регистровых логических операций (shift, or, not, nxor) над кубитным вектором и его производными, что дает возможность генерировать входные наборы и оценивать их качество в режиме online. Кроме того, облачный микросервис синтеза тестов и моделирования дефектов для функциональных логических компо-

нентов может быть востребован для учебных и научных целей в процессах синтеза и анализа цифровых архитектур.

4) Предложенный метод синтеза тестов для функциональностей на основе кубитного покрытия может быть использован в качестве встроенного BIST-компонента для сервисного обслуживания SoC на основе стандарта граничного сканирования IEEE 1500 SECT или в качестве облачного online сервиса тестирования аппаратных модулей посредством IP-протокола.

5) Дальнейшие исследования в данной области будут направлены на создание программно-аппаратных генераторов тестов, симуляторов неисправностей, исправного поведения, алгоритмов диагностирования и библиотечных решений, встроенных в инфраструктуру кристаллов и/или облачные сервисы, использующих кубитное описание функциональности логического компонента.

Литература: 1. *Рябцев В.Г., Муамар Д.Н.* Методы средства визуализации алгоритмов тестов диагностирования запоминающих устройств // Электронное моделирование 2010. Т. 32, № 3. С. 43-52. 2. *Zorian Y., Shoukourian S.* Test solutions for nanoscale Systems-on-Chip: Algorithms, methods and test infrastructure // Ninth International Conference on Computer Science and Information Technologies Revised Selected Papers, Yerevan, 2013. P. 1-3. doi: 10.1109/CSITechnol.2013.6710371. 3. *Tshagharyan G., Harutyunyan G., Shoukourian S. and Zorian Y.* Overview study on fault modeling and test methodology development for FinFET-based memories // 2015 IEEE East-West Design & Test Symposium (EWDTS), Batumi, 2015. P. 1-4. doi: 10.1109/EWDTS.2015.7493149. 4. *Проектирование и тестирование цифровых систем на кристаллах* / В. И. Хаханов, Е. И. Литвинова, И. В. Хаханова, О. А. Гузь. Харьков : ХНУРЭ, 2009. 484 с. 5. *Abramovici M., Breuer M.A. and Friedman A.D.* Digital systems testing and testable design.- Computer Science Press. 1998. 652 p. 6. *Кубитные* структуры данных вычислительных устройств / В. И. Хаханов, Ваджеб Гариби, Е. И. Литвинова, А. С. Шкиль // Электронное моделирование. 2015. Т. 37, № 1. С. 76-99. 7. *Кубитные* технологии анализа и диагностирования цифровых устройств / В. И. Хаханов, Тамер Бани Амер, С. В. Чумаченко, Е. И. Литвинова // Электронное моделирование. 2015. Т. 37, № 3. С. 17-40. 8. *Автоматизированное проектирование* цифровых устройств / С.С. Бадулин, Ю.М. Барнаулов и др. / Под ред. С.С. Бадулина. М.: Радио и связь. 1981. 240 с. 9. *Michael A. Nielsen & Isaac L. Chuang.* Quantum Computation and Quantum Information. Cambridge University Press. 2010. 676 p. 10. *Mikio Nishihara.* Quantum Computing. An Overview. Higashi-Osaka: Kinki University, 2010. 53p. 11. *Курш А.Г.* Курс высшей алгебры. М.: Наука. 1968. 426с. 12. *Бондаренко М.Ф., Хаханов В.И., Литвинова Е.И.* Структура логического ассоциативного мультипроцессора // Автоматика и телемеханика. 2012. № 10. С. 71-92. 13. *Molnar L. and Gontean A.* Fault simulation methodes // 2016 12th IEEE International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 2016. P. 194-197. 14. *Hadjithеоphanous S., Neophytou S.N. and Michael M.K.* Scalable parallel fault simulation for shared-memory multiprocessor systems // 2016 IEEE 34th VLSI Test Symposium (VTS). Las Vegas NV. 2016. P. 1-6.

15. *Pomeranz Irith, Reddy Sudhakar M.* Aliasing Computation Using Fault Simulation with Fault Dropping // IEEE Transactions on Computers. 1995. P. 139-144. 16. *Ubar R., Kusaar J., Gorev M. and Devadze S.* "Combinational fault simulation in sequential circuits," 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon. 2015. P. 2876-2879. 17. *Gorev M., Ubar R. and Devadze S.* "Fault simulation with parallel exact critical path tracing in multiple core environment," 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble. 2015. P. 1180-1185. 18. *Pomeranz I.* "Fault simulation with test switching for static test compaction," 2014 IEEE 32nd VLSI Test Symposium (VTS), Napa, CA. 2014. P. 1-6. 19. *Mirkhani S. and Abraham J. A.* "EAGLE: A regression model for fault coverage estimation using a simulation based metric". 2014 International Test Conference, Seattle, WA. 2014. P. 1-10.

Поступила в редколлегию 17.06.2016

Рецензент: д-р техн. наук, проф. Кривула Г.Ф.

УДК681.326:519.713

ИНФРАСТРУКТУРА ПРОЕКТИРОВАНИЯ SoC ДЛЯ МЕТОДА МУЛЬТИВЕРСНОГО СИНТЕЗА

ОБРИЗАН В.И.

Предлагается программно-аппаратная реализация моделей, методов и структур данных для проектирования цифровых систем на кристаллах, которая включает процедуры создания спецификации, синтеза, тестирования, моделирования и верификации на основе инфраструктуры, учитывающей промышленные средства компаний Aldec и Xilinx. Рассматриваются вопросы тестирования программных продуктов на реальных цифровых проектах создания IP-Core как примитивов для реализации цифровых систем на кристаллах.

Введение. Общая характеристика исследования

Цель — разработка и тестирование инфраструктуры проектирования цифровых систем на кристаллах, которая характеризуется параллельным выполнением мультиверсного синтеза функциональности, обеспечивающей существенное уменьшение времени создания проекта в условиях ограничения на аппаратные затраты.

Задачи:

1. Разработка метода мультиверсного синтеза управляющих и операционных автоматов в заданной инфраструктуре проектирования, ориентированных на архитектурные решения в метрике, минимизирующего время выполнения функциональности за счет распараллеливания операций при ограничении на аппаратные затраты.
2. Программная реализация моделей и методов мультиверсной разработки операционных устройств в рамках интегрированной системы проектирования функ-

Tamer Bani Amer, аспирант ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Емельянов Игорь Валерьевич, н.с. кафедры АПВТ ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Любарский Михаил, соискатель кафедры АПВТ ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ, IEEE Senior Member, IEEE Computer Society Golden Core Member. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326. E-mail: hahanov@icloud.com.

циональных и архитектурных решений SoC на основе использования продуктов верификации и синтеза компаний Aldec и Xilinx.

3. Тестирование и верификация программных модулей инфраструктуры проектирования цифровых систем на кристаллах, а также определение эффективности предложенных моделей, методов и структур данных при создании реальных компонентов цифровых изделий.

1. Организация системы автоматизированного проектирования

По своей сути программа синтеза C++ в VHDL является машиной по трансформации исходного описания в результирующее. Таких трансформаций происходит несколько. Входная модель представлена на языке C++. Это алгоритмическое описание, которое решает поставленную перед инженером задачу.

Первый этап преобразования – *синтаксический анализ*. На этом этапе во входном описании из потока символов выделяются лексические элементы: ключевые слова, операторы и лексемы. В результате этого этапа получается синтаксическая модель исходного алгоритмического описания.

Второй этап преобразования – *трансформации на уровне синтаксической модели*. Они применяются для получения моделей с меньшим количеством состояний и занимающих меньше аппаратных ресурсов. К таким трансформациям относятся: вычисление констант, удаление недостижимого кода, встраивание функций, операции над циклами (развертки, свертки, распараллеливания).

Третий этап преобразования – *построение граф-схемы алгоритма*. В этой модели алгоритм представлен в виде отношений вершин двух типов: операций (арифметических, логических или ввода/вывода) и ветвений.

15. *Pomeranz Irith, Reddy Sudhakar M.* Aliasing Computation Using Fault Simulation with Fault Dropping // IEEE Transactions on Computers. 1995. P. 139-144. 16. *Ubar R., Kusaar J., Gorev M. and Devadze S.* "Combinational fault simulation in sequential circuits," 2015 IEEE International Symposium on Circuits and Systems (ISCAS), Lisbon. 2015. P. 2876-2879. 17. *Gorev M., Ubar R. and Devadze S.* "Fault simulation with parallel exact critical path tracing in multiple core environment," 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble. 2015. P. 1180-1185. 18. *Pomeranz I.* "Fault simulation with test switching for static test compaction," 2014 IEEE 32nd VLSI Test Symposium (VTS), Napa, CA. 2014. P. 1-6. 19. *Mirkhani S. and Abraham J. A.* "EAGLE: A regression model for fault coverage estimation using a simulation based metric". 2014 International Test Conference, Seattle, WA. 2014. P. 1-10.

Поступила в редколлегию 17.06.2016

Рецензент: д-р техн. наук, проф. Кривуля Г.Ф.

УДК681.326:519.713

ИНФРАСТРУКТУРА ПРОЕКТИРОВАНИЯ SoC ДЛЯ МЕТОДА МУЛЬТИВЕРСНОГО СИНТЕЗА

ОБРИЗАН В.И.

Предлагается программно-аппаратная реализация моделей, методов и структур данных для проектирования цифровых систем на кристаллах, которая включает процедуры создания спецификации, синтеза, тестирования, моделирования и верификации на основе инфраструктуры, учитывающей промышленные средства компаний Aldec и Xilinx. Рассматриваются вопросы тестирования программных продуктов на реальных цифровых проектах создания IP-Core как примитивов для реализации цифровых систем на кристаллах.

Введение. Общая характеристика исследования

Цель — разработка и тестирование инфраструктуры проектирования цифровых систем на кристаллах, которая характеризуется параллельным выполнением мультиверсного синтеза функциональности, обеспечивающей существенное уменьшение времени создания проекта в условиях ограничения на аппаратные затраты.

Задачи:

1. Разработка метода мультиверсного синтеза управляющих и операционных автоматов в заданной инфраструктуре проектирования, ориентированных на архитектурные решения в метрике, минимизирующего время выполнения функциональности за счет распараллеливания операций при ограничении на аппаратные затраты.
2. Программная реализация моделей и методов мультиверсной разработки операционных устройств в рамках интегрированной системы проектирования функ-

Tamer Bani Amer, аспирант ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Емельянов Игорь Валерьевич, н.с. кафедры АПВТ ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Любарский Михаил, соискатель кафедры АПВТ ХНУРЭ. Научные интересы: квантовые вычисления, тестирование и диагностика цифровых систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326.

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ, IEEE Senior Member, IEEE Computer Society Golden Core Member. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. +3805770-21-326. E-mail: hahanov@icloud.com.

циональных и архитектурных решений SoC на основе использования продуктов верификации и синтеза компаний Aldec и Xilinx.

3. Тестирование и верификация программных модулей инфраструктуры проектирования цифровых систем на кристаллах, а также определение эффективности предложенных моделей, методов и структур данных при создании реальных компонентов цифровых изделий.

1. Организация системы автоматизированного проектирования

По своей сути программа синтеза C++ в VHDL является машиной по трансформации исходного описания в результирующее. Таких трансформаций происходит несколько. Входная модель представлена на языке C++. Это алгоритмическое описание, которое решает поставленную перед инженером задачу.

Первый этап преобразования – *синтаксический анализ*. На этом этапе во входном описании из потока символов выделяются лексические элементы: ключевые слова, операторы и лексемы. В результате этого этапа получается синтаксическая модель исходного алгоритмического описания.

Второй этап преобразования – *трансформации на уровне синтаксической модели*. Они применяются для получения моделей с меньшим количеством состояний и занимающих меньше аппаратных ресурсов. К таким трансформациям относятся: вычисление констант, удаление недостижимого кода, встраивание функций, операции над циклами (развертки, свертки, распараллеливания).

Третий этап преобразования – *построение граф-схемы алгоритма*. В этой модели алгоритм представлен в виде отношений вершин двух типов: операций (арифметических, логических или ввода/вывода) и ветвений.

Четвертый этап преобразования – *построение автоматной модели*. Она представлена в виде вершин с операциями и переходов между ними.

Пятый этап преобразования — *синтез операционного и управляющего автоматов*. В результате этого этапа получается структурная модель, определенная на множестве логических элементов, регистров, арифметико-логических устройств и элементов памяти.

Шестой этап преобразования – *сохранение модели операционного и управляющего автомата в VHDL-код*. Результатом этого этапа является набор исходных файлов на языке VHDL, которые описывают устройство на уровне регистровых передач. Эта VHDL-модель реализует исходный алгоритм, изначально составленный на языке C++.

2. Тестирование системы

Тестирование системы (рис. 1) состоит из нескольких этапов:

- разработка схемы тестирования компилятора;
- подготовка тестов;
- подготовка эталонов;
- выполнение тестов и анализ результатов.

Были подготовлены тесты для различных частей компилятора: синтаксический анализатор; построитель граф-схемы алгоритма; построитель цифрового автомата; построитель VHDL-модели синтезируемого устройства.

При подготовке тестов к синтаксическому анализатору учитывались следующие особенности компилятора:

- работа препроцессора;
- поддержка базовых типов языка C++;
- поддержка базовых конструкций языка C++.

Тест для компилятора — это исходный текст на языке C++. Примитивный тест показан в листинге 1. Например, этот тест проверяет успешную декларацию и инициализацию объектов типа `int`, декларацию функции, арифметическую операцию «сложение».

Листинг 1. Примитивный тест для компилятора

```

1. int f1()
2. {
3.     int a = 2, b = 3;
4.     return a + b;
5. }
```

При выполнении теста сохраняется возвращаемое значение функции в файл для последующего сравнения с эталоном. Эталоны были подготовлены с использованием компилятора Microsoft Visual Studio. Было предположено, что он не содержит ошибок.

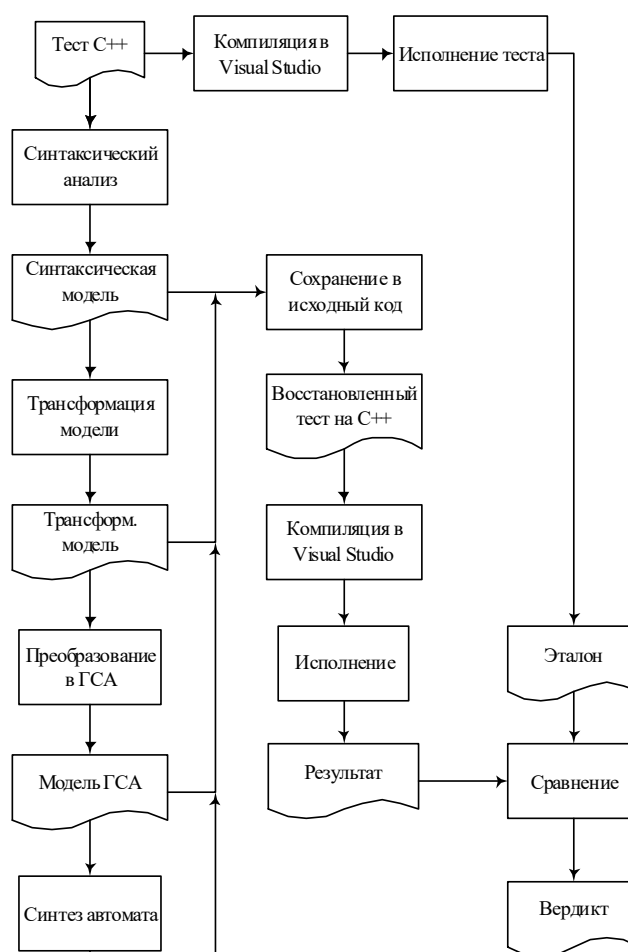


Рис. 1. Схема тестирования различных внутренних моделей компилятора

3. Сравнение с аналогами

В американском патенте № 6 226 776 «System for Converting Hardware Designs in High-Level Programming Language to Hardware Implementations» [1] предлагается система автоматизированного проектирования, которая конвертирует алгоритмическое описание на языке ANSI C в аппаратные реализации на ПЛИС или на кристаллах жесткой логики. В патенте предложены многочисленные примеры преобразований алгоритмов из языка C++ в синтезируемое подмножество языка Верилог. Рассмотрим пример, показанный в листинге 2.

Программа высокоуровневого синтеза, предложенная в этой работе, произвела оптимизированную ГСА, как показано на рис. 2.

При компиляции примера были сделаны следующие оптимизации: встраивание функций, выявление параллелизма на уровне микроопераций. Таким образом, после синтеза мы получили модель автомата Мили с 7-ю состояниями. В патенте показано, что в результате конвертации этого примера получается 16 состояний автомата.

Листинг 2. Исходный алгоритм на языке C++

```

6. int sum1 (int n)
7. {
8.     int i, sum = 0;
9.     for (i = 0; i < n; i++)
10.        sum += i;
11.    return sum;
12. }
13.
14. int sum2 (int array[], int size)
15. {
16.     int i, sum = 0;
17.     for (i = 0; i < size; i++)
18.        sum += array[i];
19.    return sum;
20. }
21.
22. int f1()
23. {
24.     int i;
25.     int array[10];
26.     int size = sizeof(array)/sizeof(*array);
27.     for (i = 0; i < size; i++)
28.        array[i] = i * 2;
29.     return sum1(size) + sum2(array, size);
30. }

```

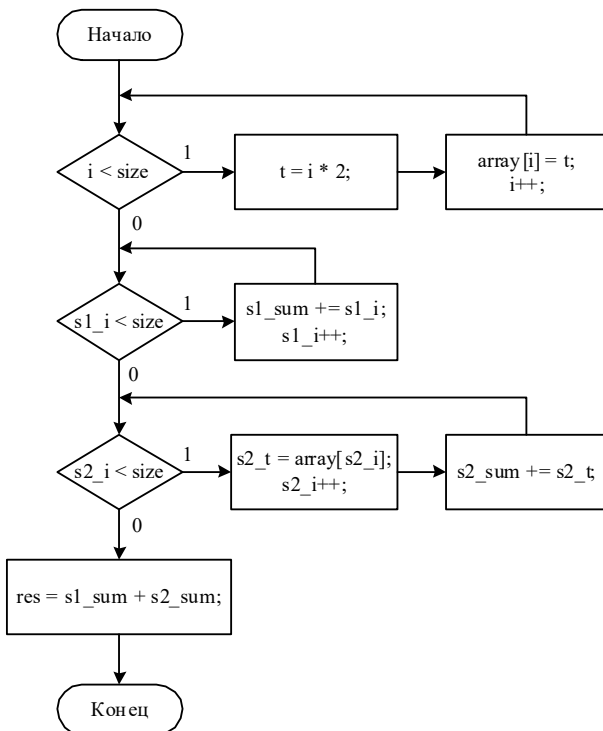


Рис. 2. СГСА, полученная в результате синтеза

4. Проектирование системы на кристалле

Рассмотрим систему на кристалле, показанную на рис. 3.

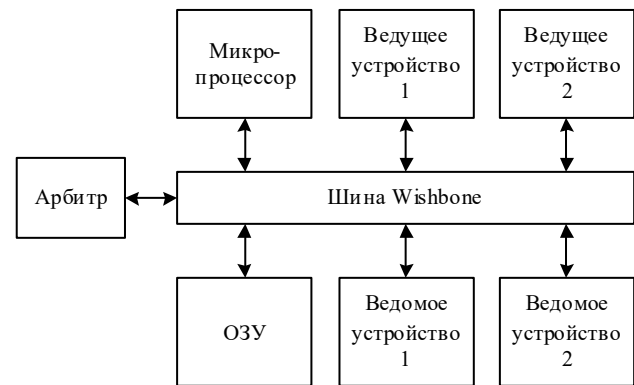


Рис. 3. Архитектура тестовой системы на кристалле

В данной тестовой системе на кристалле используется микропроцессор OpenRISC 1200 [2]. OpenRISC — это 32-разрядный RISC-микропроцессор с открытым исходным кодом. Модель процессора распространяется на правах свободной лицензии, что позволяет его изучать, моделировать, развивать и использовать в коммерческих системах без уплаты авторских отчислений. OpenRISC сделан на основе гарвардской архитектуры, где предполагается раздельное хранение инструкций и данных. Микропроцессор имеет пятиступенчатый конвейер, тем не менее, большинство инструкций могут быть выполнены за один такт.

В этой системе используется шина Wishbone на тех же правах.

Для компиляции исходных текстов программ на языке C применяется компилятор gcc.

При запуске системы микропроцессор начинает читать инструкции по адресу 256.

5. Общая организация системы проектирования

На рис. 4 показана структура процесса проектирования с использованием пространства проектных решений. На начальных этапах осуществляется ввод спецификации проекта на системном уровне с применением языков C++ и SystemC. Ввод моделей осуществляется как в текстовом виде, так и в графическом с использованием блочных диаграмм и содержательных граф-схем алгоритмов. При определении спецификации указываются функциональные и нефункциональные требования. Первые описывают закон функционирования модели, а также входные воздействия на систему и ожидаемые ответные реакции. Среди нефункциональных требований можно выделить следующие:

- ограничения на получаемые проектные решения: стоимость реализации, быстродействие, производительность, аппаратные затраты, затраты оперативной памяти, энергопотребление;

– доступные архитектуры: типы интегральных схем, элементная база, а также их характеристики; типы встроенных микропроцессоров.

Исходные тексты входных спецификаций должны соответствовать международным стандартам C++ [3] и SystemC [4].

На следующем этапе осуществляется синтаксический анализ исходной модели. В случае обнаружения ошибок создается отчет, на основании которого инженер может исправить ошибки и повторить трансляцию модели. Если исходные тексты модели не содержат ошибок, то строится синтаксическая модель проектируемой системы.

Синтаксическая модель пригодна для широкого класса задач автоматизированного проектирования. Среди них: декомпозиция проекта на программную и аппаратную части, высокоуровневая оптимизация, логический синтез и компиляция, построение проверяющих тестов.

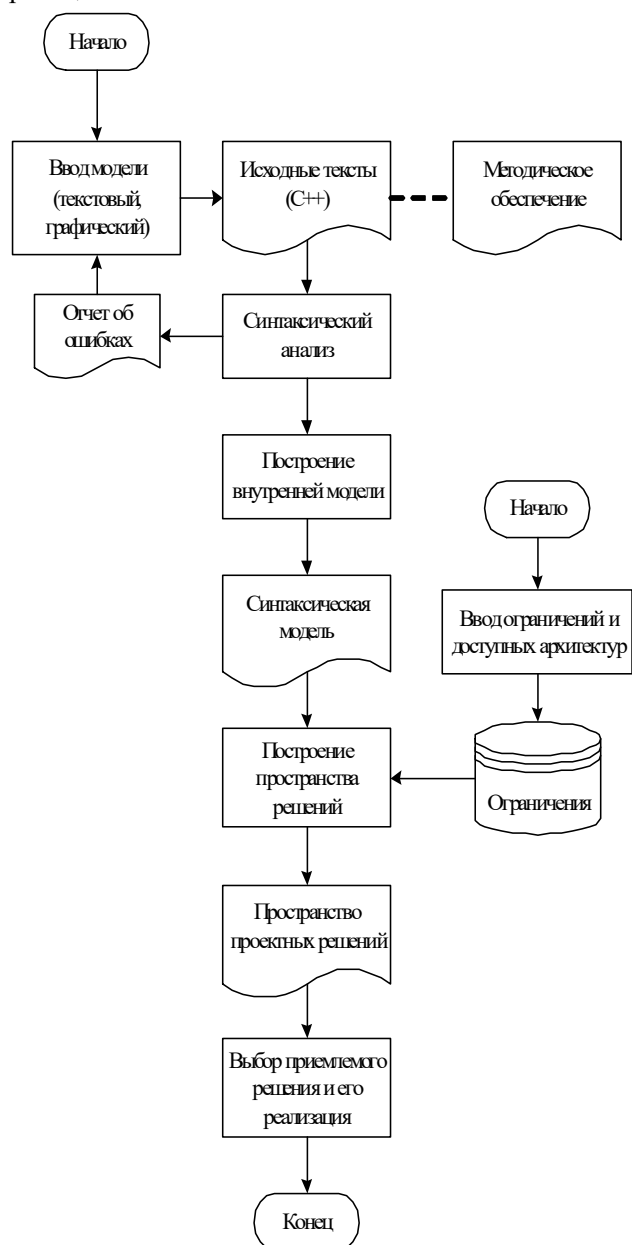


Рис. 4. Структура процесса проектирования [5] с использованием пространства проектных решений РИ, 2016, № 2



Рис. 5. Процесс построения пространства проектных решений

6. Структурная декомпозиция проекта

Для успешной реализации проекта необходимо решить задачу структурной декомпозиции проекта на программную и аппаратную части. Введем несколько определений, которые широко используются в языке SystemC.

Определение 1. Модуль — класс C++, унаследованный от класса *sc_module*, объявленного в библиотеке SystemC. В языке VHDL понятию модуля соответствует пара *entity* и *architecture*. Модуль имеет определенный интерфейс — множество входных и выходных сигналов определенного типа, а также множество параллельно взаимодействующих процессов с соответствующими списками чувствительности. Процесс в SystemC аналогичен конструкции *process* в языке VHDL или *always* в языке Verilog.

Определение 2. Иерархия модулей — множество экземпляров модулей, созданных при компиляции SystemC модели. На этом множестве определены

отношения «А есть дочерний модуль Б», «А есть родительский модуль Б».

Определение 3. Модуль верхнего уровня — экземпляр модуля, который не установлен ни в один другой модуль. В случае несвязанной иерархии модулей может быть несколько модулей верхнего уровня.

Рассмотрим блочную диаграмму проекта *sc_main*, показанного на рис. 6 (исходный текст модели на языке SystemC может быть найден в [6]). Блочная диаграмма отражает иерархию модулей, а также отношения модулей между собой. Здесь модули *m_producer* и *m_consumer* являются дочерними по отношению к модулю *testbench*, который в свою очередь является дочерним к модулю *sc_main*.

Проект цифровой системы, описанный на языке SystemC, удобно представить в виде корневого дерева, определенного на множестве иерархии модулей проекта. Это дерево отображает отношение «быть установленным в». Корень дерева — модуль верхнего уровня, в котором установлены все другие экземпляры модулей низших уровней. На рис. 7 показано корневое дерево для проекта *sc_main*.

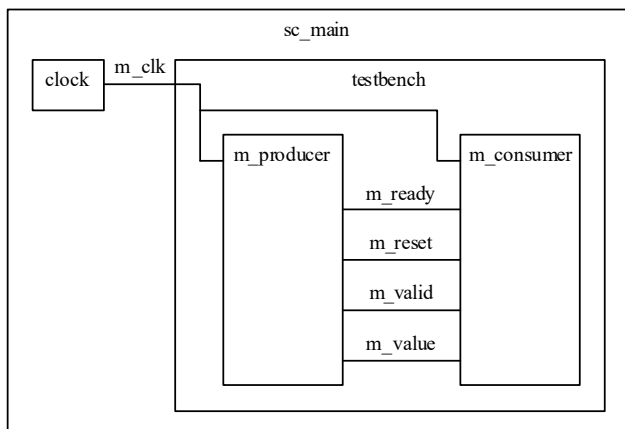


Рис. 6. Блочная диаграмма проекта *sc_main*

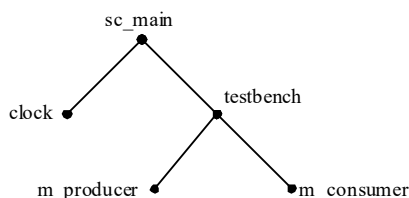


Рис. 7. Представление иерархии модулей в виде корневого дерева

Для реализации каждого модуля есть две альтернативы: аппаратная реализация и программная. Количество конфигураций при таком условии равно $k = 2^n$, где n — количество модулей в иерархии. Это количество может быть сокращено за счет того, что не все модули могут быть логически синтезированы в аппаратные структурные модели. Таким образом, количество конфигураций может быть определено

$k = 2^m, m \leq n$, где m — количество модулей, для которых возможно построить корректную аппаратную модель из SystemC описания. Например, для модели на рис. 4 $k = 32$.

7. Методы структурных и алгоритмических трансформаций

Существует компромисс между объемом аппаратных ресурсов и временем выполнения той или иной задачи. Аппаратура обладает громадными возможностями в плане выполнения функций с большим уровнем параллелизма. При распараллеливании функция может быть выполнена быстрее, но для этого требуются дополнительные аппаратные затраты.

Методы структурной оптимизации заключаются в применении различных трансформаций моделей или описаний, которые приводят к улучшению временных или аппаратных характеристик устройства.

Методы могут быть применены на различных уровнях абстракции. На алгоритмическом уровне объектом оптимизации становится алгоритм работы устройства. На микроархитектурном уровне объектом оптимизации выступает структурная модель операционного автомата, а также граф управляющего автомата. Базовые сведения о минимизации условных и операторных вершин автоматов могут быть найдены в работах С.И. Баранова [7].

Векторизация — такая трансформация цикла, при которой циклические действия выполняются сразу над множеством операндов, а не над одним. В классификации Флинна такой организации вычислений соответствует модель SIMD — Single Instruction, Multiple Data — одна операция, множество операндов.

Циклы — типичная горячая точка в приложении. Даже быстрая операция или функция, выполненная миллион раз, замедлит работу системы. Кроме полезной нагрузки: тела цикла, имеются и накладные расходы на организацию цикла: счетчик итераций (регистр), операция инкремент/декремент, ветвление. При векторизации цикла используется операция «развертка цикла». Операнды объединяются в векторы с определенным числом элементов, называемым *степень векторизации цикла*. Как правило, операция выполняется за один такт над всеми элементами вектора.

Рассмотрим пример, показанный в листинге 3. Здесь идет последовательное суммирование элементов двух векторов, результат записывается в третий вектор.

Листинг 3. Последовательная обработка элементов массива в цикле

```
31. for (int i = 0; i < N; i++)
32.     c[i] = a[i] + b[i];
```

На рис. 8 схематически показано выполнение арифметических операций в оригинальном цикле (а) и в векторизованных циклах (б, в). В первом случае за одну итерацию выполняется действие над одной парой операндов, а во втором и третьем случаях — над двумя и четырьмя парами операндов.

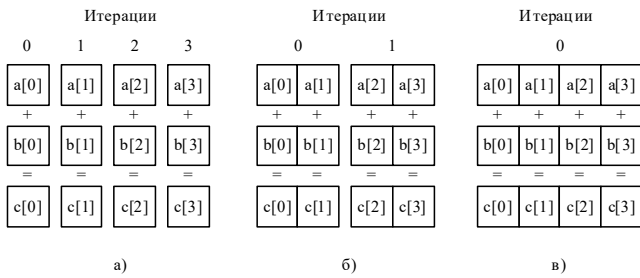


Рис. 8. Выполнение операций: а – за четыре итерации в оригинальном цикле; б – за две итерации в цикле со степенью векторизации 2; в – за одну итерацию в цикле со степенью векторизации 4

На рис. 9 приведена содержательная ГСА для этого примера. Таким образом, цикл включает в себя четыре состояния автомата. Для выполнения программы суммирования векторов из N элементов потребуется $k = 4 \times N$ тактов работы устройства. Аппаратные затраты: 4 регистра ($a1, b1, c1, i$), 2 сумматора, функция сравнения «меньше».

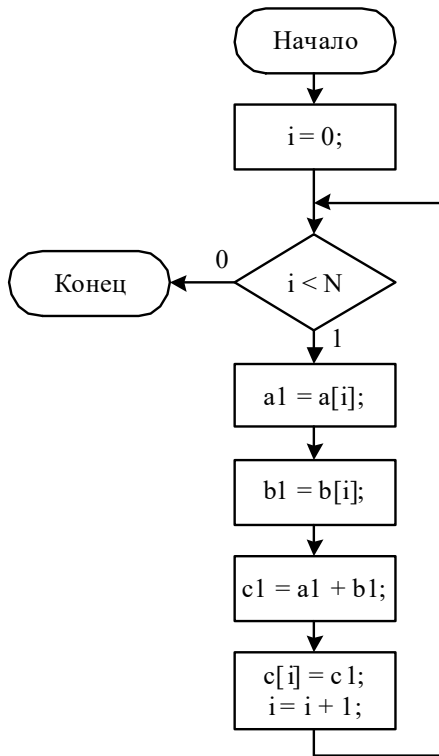


Рис. 9. СГСА для примера в листинге 3

Рассмотрим пример, показанный в листинге 4. Осуществлена трансформация «векторизация цикла». Здесь на одном витке цикла выполняется две операции сложения. Важное условие — N должно быть кратно степени векторизации.

Листинг 4. Векторизованный цикл из листинга 1

```

33. for (int i = 0; i < N; i += 2) {
34.     c[i] = a[i] + b[i];
35.     c[i+1] = a[i+1] + b[i+1];
36. }

```

На рис. 10 показана СГСА для примера из листинга 4. Цикл включает в себя 7 состояний автомата. Для выполнения программы суммирования векторов из N элементов потребуется $k = (7 \times N) / 2$ тактов работы устройства. Аппаратные затраты: 7 регистров ($a1, a2, b1, b2, c1, c3, i$), 3 сумматора, функция сравнения «меньше».

В этом примере видно, что в теле цикла присутствует четыре операции чтения из памяти и две операции записи в память, которые должны выполняться последовательно. Две операции сложения могут быть выполнены параллельно. Можно подсчитать, что цикл со степенью векторизации 1, в котором содержится s операций (выполняются последовательно) и p операций (выполняются параллельно), может быть выполнен за k тактов, что определяется формулой:

$$k = \frac{(s + 1) \times N}{1} \quad (1)$$

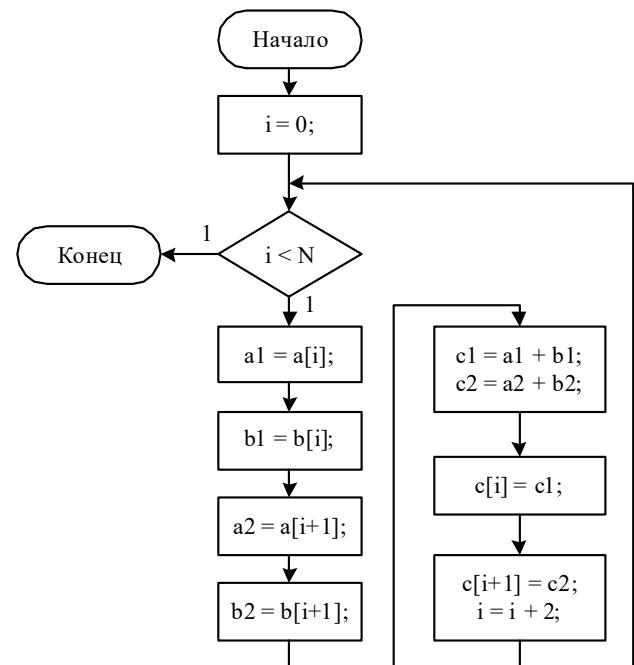


Рис. 10. СГСА для примера из листинга 4

Рассмотрим различные оптимизации на микроархитектурном уровне. *Слияние (объединение) микроопераций* – такая трансформация ГСА, при которой микрооперации в различных операторных блоках (рис. 11, а) объединяются в один операторный блок (рис. 11, б). Такая трансформация возможна, если микрооперации u_1 и u_2 могут выполняться параллельно и их одновременное выполнение приводит к тому же результату, что и последовательное. Допускается объединение произвольного количества микроопераций на линейном участке ГСА. При такой трансформации сокращается количество операторных вершин, что приводит к сокращению количества состояний управляющего автомата Мура и сокращению количества тактов в цикле. Эта трансформация не влияет на аппаратные затраты.

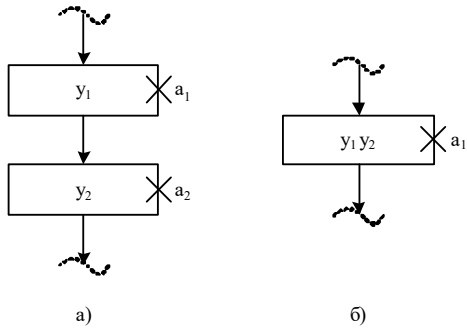


Рис. 11. Слияние микроопераций:
а – исходный подграф; б – трансформированный подграф

В том случае, если между двумя операторными вершинами есть входящая дуга (рис. 12, а), то слияние операций происходит по следующим правилам. Необходимо объединить операции y_1 и y_2 в состояние a_1 по правилам, указанным выше. Микрооперацию y_2 следует внести в цепь до схождения дуг (рис. 12, б). Такое преобразование не приводит к изменению количества состояний в автомате и не влияет на аппаратные затраты. Выигрыш заключается в сокращении количества тактов в цикле.

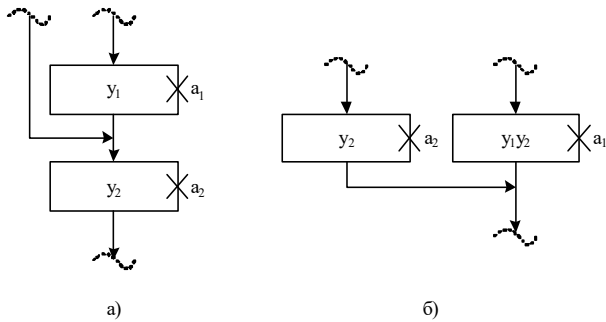


Рис. 12. Слияние микроопераций:
а – исходный подграф; б – трансформированный подграф

Рассмотрим более сложный случай объединения операторов. В исходном подграфе на рис. 13, а микрооперации y_1 и y_2 могут быть выполнены параллельно. В цепи между состояниями a_1 и a_2 присутствует условный оператор и разветвление. В том случае, если y_1 и y_2 будут объединены в состояние a_1 , то в операторе альтернативной ветви необходимо поместить микрооперацию y_2' (читается «игрек два штрих»), которая имеет обратное действие микрооперации y_2 .

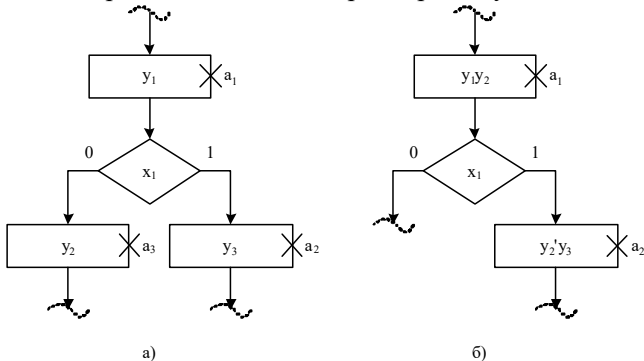


Рис. 13. Сложный случай объединения операторов:
а – исходный подграф; б – трансформированный подграф

Вычисление констант — такая трансформация операционного автомата, при которой вносится аппаратная избыточность, вычисляющая константную функцию от переменной (рис. 14). При такой трансформации сокращается количество состояний автомата, а также длина цикла в тактах. При этом возрастает время распространения сигнала от регистра к выходам.

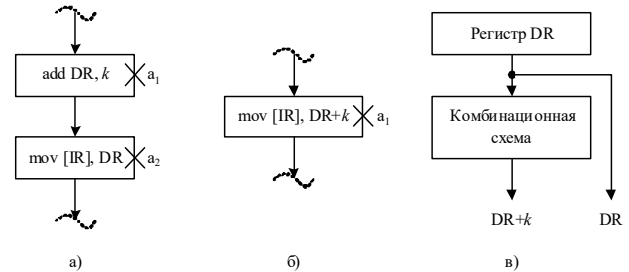


Рис. 14. Вычисление констант: а – исходный подграф; б – трансформированный подграф; в – структурная реализация

На рис. 15 показана оптимизированная ГСА, в которой объединены микрооперации и вычислено константное выражение. Количество состояний сокращено с 5 до 3. Самый короткий цикл a_2 составляет один такт, самый длинный a_2a_3 – два такта.

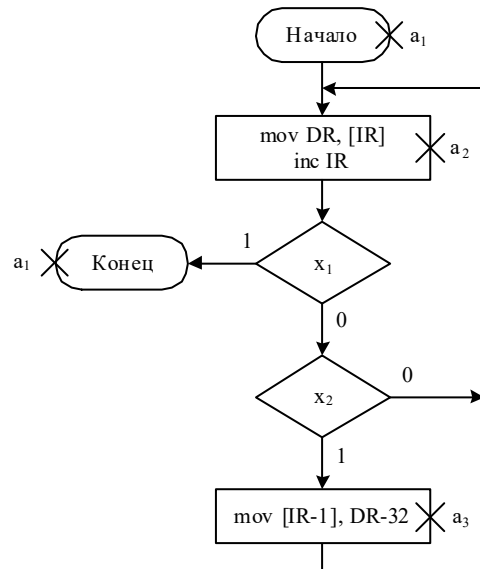


Рис. 15. Оптимизированная ГСА

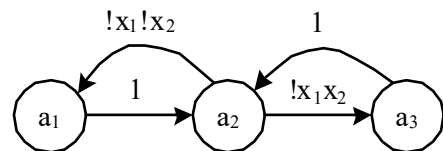


Рис. 16. Оптимизированный граф переходов

8. Преобразование несинтезируемых конструкций

Синтаксис языка C++ чрезвычайно многообразен. Существует большое множество библиотек и при-

ёмов программирования на C++, которые не могут быть преобразованы в схему на уровне регистровых передач на текущий момент. Например, популярными являются функции и классы ввода-вывода информации: `scanf`, `printf`, `std::cin`, `std::cout`, а также операции с файлами, которые не имеют прямого отображения в логическую схему. В настоящий момент программы синтеза обнаруживают такие конструкции на ранних стадиях компиляции исходных файлов, а затем либо игнорируют эти функции, либо сообщают о присутствии несинтезируемых конструкций и останавливают процесс синтеза.

В некоторых случаях целесообразно подготовить замену такой несинтезируемой конструкции, чтобы продолжить верификацию модели, пока не будет готова синтезируемая замена. На рис. 17 показан график проекта, в котором верификация не может быть начата до тех пор, пока не будут подготовлены синтезируемые замены несинтезируемым конструкциям.

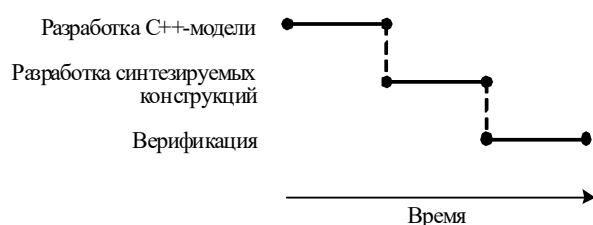


Рис. 17. График проектирования устройства

Существует решение этой проблемы, которое заключается в использовании виртуальных прототипов на языке SystemC. Виртуальный SystemC-прототип — это модель на языке SystemC, интерфейс которой строго соответствует проектируемому устройству, а архитектура выполнена на высоком уровне описания. Обычно SystemC-прототипы не синтезируются, а используются для верификации системы в целом на ранних этапах проектирования.

На рис. 18 показан график проекта, в котором несинтезируемые конструкции заменены SystemC-прототипами. Из-за быстрого перехода от C++ к SystemC модели верификация всей цифровой системы может быть начата раньше, чем будет закончена разработка синтезируемых замен.

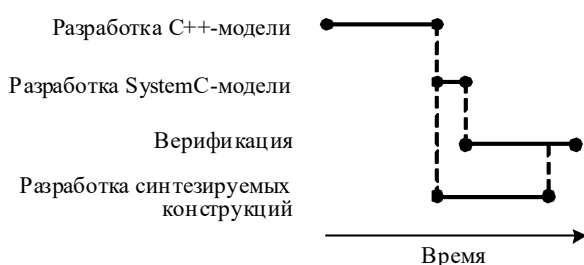


Рис. 18. График проектирования устройства с использованием SystemC-прототипов

Основные этапы преобразования следующие.

1. Определить фрагмент кода, который не может быть синтезирован.
2. Определить информационные зависимости этого фрагмента от смежного кода.
3. Выделить этот фрагмент кода в отдельную C++-функцию.
4. Разработать SystemC-прототип на основе этой C++-функции.
5. Выполнить синтез C++ модели в структурную VHDL модель.
6. Интегрировать SystemC модель в структурную VHDL модель.

В результате преобразования получается система, показанная на рис. 19. На этапе верификации будет использована SystemC-модель (рис. 19, а). После того, как будет готов структурный эквивалент, будет использована VHDL-модель компонента (рис. 19,б).

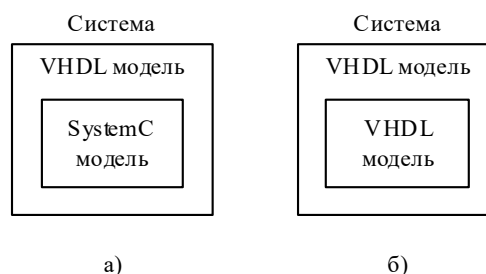


Рис. 19. Иерархия структурной модели: а – с использованием SystemC-модели компонента; б – с использованием VHDL-модели компонента

Рассмотрим пример, показанный в листинге 5. В этом примере создается объект статической памяти, объемом 1024 ячейки, которые инициализируются псевдослучайными значениями. Предположим, что функция `rand()` не имеет структурного эквивалента на этот момент.

Листинг 5. Пример программы с несинтезируемой конструкцией

```

37. int main()
38. {
39.     const int size = 1024;
40.     int mem[size];
41.     for (int i = 0; i < size; i++)
42.         mem[i] = rand();
43.     return 0;
44. }

```

Введем несколько определений.

Определение 4. Транзакция — модельное событие высокого уровня, которое включает в себя множество событий низшего уровня. В транзакции соблюдается строгая последовательность и временные интервалы между событиями. Обычно транзакция — это целостная операция, например, чтения или записи

данных. Синтаксически транзакция выглядит как вызов функции или метода класса.

Определение 5. Транзакционная модель – такая модель системы, где все события между элементами происходят на уровне транзакций.

Определение 6. Транзактор – элемент системы, который преобразовывает транзакцию к множеству событий модели низшего уровня, например, регистрового или вентильного. Транзактор также выполняет обратное преобразование – с регистрового и вентильного уровней на уровень транзакций. Транзактор реализуется с помощью класса, имеющего два интерфейса: высокого уровня и уровня регистровых передач или даже вентильного.

Здесь и далее будет использоваться нотация для моделей системного уровня, как показано на рис. 20. Нотация взята из [8]. Интерфейс системного уровня соответствует понятию интерфейса языка C++ — совокупность абстрактных методов, которые реализует модуль.

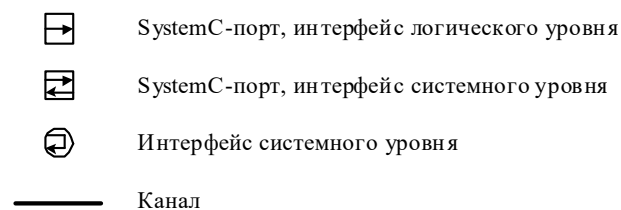


Рис. 20. Нотация моделей системного уровня

Представим эту программу в виде транзакционной модели [9] на языке SystemC, как показано на рис. 21. Необходимо сделать декомпозицию на два модуля: а) Main — основной, где происходят все вычисления; б) Rand — модуль, где будет производиться вычисление функции rand(). Модуль Main будет иметь один порт абстрактного типа rand_if (см. листинг 5).

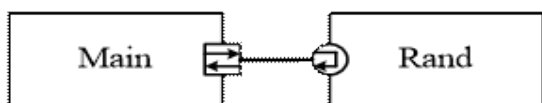


Рис. 21. Система из двух модулей

Создание абстрактного интерфейса необходимо для того, чтобы можно было создавать различные модули, которые наследуют один и тот же интерфейс.

Листинг 6. Абстрактный интерфейс для модуля rand

```

45. struct rand_if : public sc_interface {
46.     virtual int rand_f() = 0;
47. };
  
```

Модифицированный модуль Main показан в листинге 7. Здесь создан класс-обертка на языке SystemC.

Листинг 7. Модуль Main

```

48. SC_MODULE(Main) {
49.     sc_port<rand_if> rand_m;
  
```

```

50.     SC_CTOR(Main) {
51.         SC_THREAD(process);
52.     }
53.     void process() {
54.         for (int i = 0; i < size; i++)
55.             // подстановка вместо функции
56.             mem[i] = rand_m->rand_f();
57.         sc_stop();
58.     }
59. private:
60.     const int size = 1024;
61.     int mem[size];
62. };
  
```

Реализация модуля rand показана в листинге 8.

Листинг 8. Реализация модуля Rand.

```

63. SC_MODULE(Rand), public rand_if {
64.     virtual int rand_f() {
65.         return rand();
66.     }
67. };
  
```

Для успешной интеграции виртуального прототипа в систему на вентильном уровне необходимо детализировать интерфейсы системы. Для этого между модулями Main и Rand нужно поместить два транзактора: а) H2L (High Level To Low Level), который преобразует высокоуровневый интерфейс rand_if к протоколу на уровне логических сигналов; б) L2H (Low Level to High Level) — транзактор, обратный H2L. По своей сути Rand — это генератор случайных чисел, который генерирует очередное число по запросу. Для этого в систему введен модуль Clock — генератор синхриимпульсов. Таким образом, при детализации системы в ней появилось понятие времени (рис.22). Мы получили временную транзакционную модель из безвременной транзакционной модели.

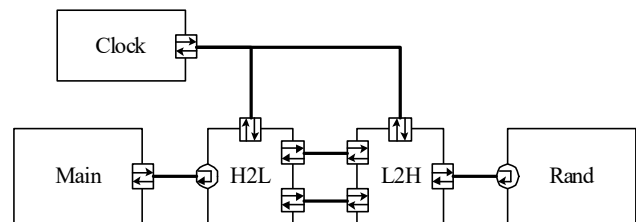


Рис. 22. Детализированная система

Иерархия классов на языке UML показана на рис. 23. Из диаграммы классов видно, что модули Rand и H2L имеют идентичный интерфейс rand_if. Это дает возможность свободно подключать любой из них к порту модуля Main.

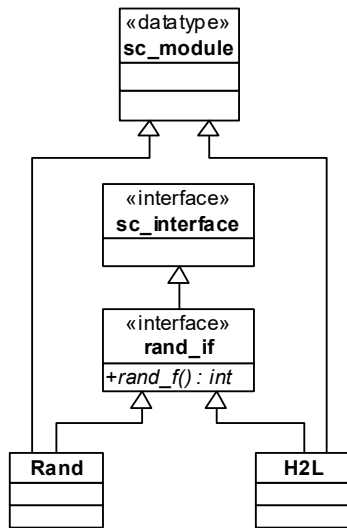


Рис. 23. Иерархия классов: множественное наследование от класса `sc_module` и интерфейса `rand_if`

Построим содержательную граф-схему алгоритма для основного процесса модуля `Main` (рис. 24).

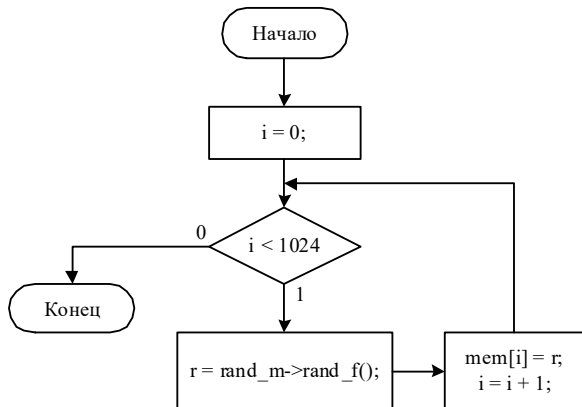


Рис. 24. СГСА для процесса `process` модуля `Main`

В листинге 9 показана реализация транзактора [10] `L2H`. Логика работы транзактора следующая: он реагирует на передний фронт синхронизации `clk`, и если сигнал разрешения `en` установлен в единицу, то выставляет на выход `output` значение функции `rand_f()` модуля, подключенного к порту `rand_m`. Важно отметить то, что интерфейс транзактора (сигналы `clk`, `enable`, `output`) уже меняться не будут. Эти же сигналы будут использованы при установке реального синтезируемого модуля `Rand`, когда такой модуль будет доступен. То же самое касается и протокола работы этого транзактора.

Листинг 9. Реализация транзактора `L2H`

```

68. SC_MODULE(L2H) {
69.     sc_port<rand_if> rand_m;
70.     sc_in<bool> clk;
71.     sc_in<en> enable;
72.     sc_out<int> output;
73.     SC_CTOR(L2H) {
74.         SC_METHOD(process);
  
```

```

75.         sensitive << clk.posedge();
76.     }
77.     void process() {
78.         if(en)
79.             output = rand_m->rand_f();
80.     }
81. };
  
```

На рис. 25 показана временная диаграмма работы транзактора `L2H`.

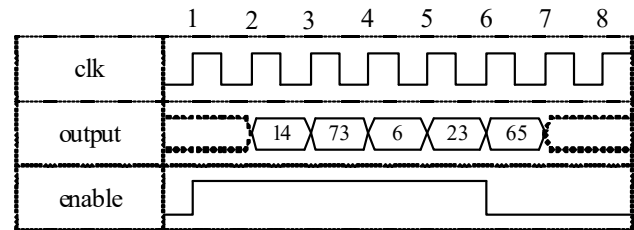


Рис. 25. Временная диаграмма работы транзактора `L2H`

Следующим этапом интеграции виртуального прототипа является аппаратный синтез модуля `Main` и его слияние с транзактором `L2H`. После этого шага система примет вид, как показано на рис. 26. Модуль `Main-HW` представляет собой аппаратную реализацию модуля `Main` (см. листинг 7).

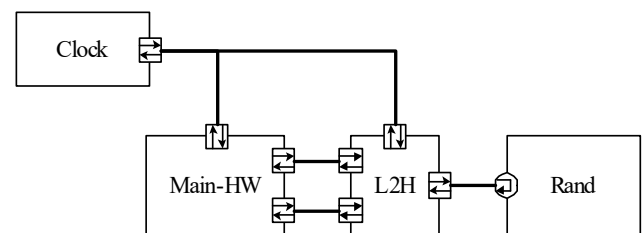


Рис. 26. Слияние транзактора `L2H` и модуля `Main` в модуль `Main-HW`

СГСА для этого модуля, с условием подключения к транзактору `L2H`, показана на рис. 27. Подразумевается, что сигналы `en` и `output` — это соответствующие сигналы транзактора `L2H`.

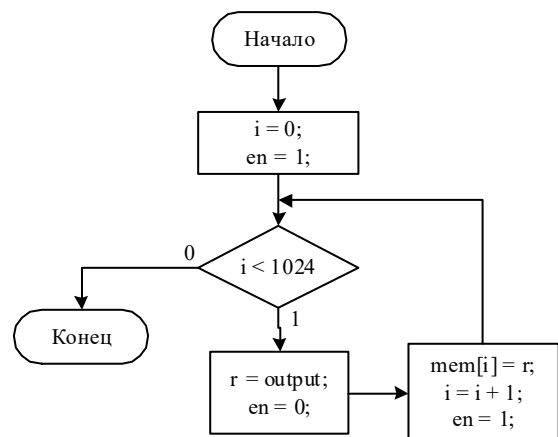


Рис. 27. Модифицированная СГСА для модуля `Main-HW`

9. Результаты практического синтеза мультиверсного метода проектирования

Было проведено сравнение производительности разработанной аппаратной модели и программной реализации на микропроцессоре экспериментальным путем. Оценивались следующие характеристики: временные затраты на проектирование, быстродействие, энергопотребление, площадь на кристалле.

Было выбрано три самые распространённые реализации: последовательностная, параллельная, конвейерная [11].

Последовательностной называется реализация, в которой все микрооперации упорядочены во времени. Преимущество применения такой реализации в том, что используется меньшая площадь на кристалле, меньшее энергопотребление, недостаток – низкая производительность.

Параллельной называется реализация, в которой параллельно выполняются две и более микрооперации за один такт работы. Особенности: высокое энергопотребление, высокая производительность и площадь на кристалле.

Конвейерной называется реализация, в которой за один такт работы выполняется две и более микрооперации на разном этапе выполнения. Особенности: высокая производительность, занимает больше места на кристалле и имеет высокое энергопотребление.

Для аддитивной (мультипликативной) оценки эффективности методов все значения были интегрально оценены. При интегральной оценке самое эффективное значение принималось за единицу, а остальные высчитывались по следующей формуле:

$$y_i = x_i / x_{\max} \quad (2)$$

где x_{\max} – самое эффективное значение; x_i – значение, которое было получено в ходе эксперимента.

Эта оценка хорошо показала, в какой реализации какой метод более эффективен. В последовательностной реализации по всем критериям эффективным является автоматический метод. В параллельной реализации по площади на кристалле эффективнее работает ручной метод, а по всем остальным критериям выигрывает автоматический метод. В конвейерной реализации ручной метод выигрывает по энергопотреблению, во всем остальном эффективнее автоматический метод.

После этого была сделана аддитивная (мультипликативная) оценка эффективности каждой по отдельности реализации и общая аддитивная (мультипликативная) оценка эффективности метода.

Общий вид аддитивной модели следующий:

$$y_i = \prod_{i=1}^n x_i \quad (3)$$

где y_i – коэффициент каждого инженера для аддитивной оценки, x_i – коэффициент по каждому критерию, n – количество критериев.

Общий вид мультипликативной модели имеет вид:

$$y_i = \prod_{i=1}^n x_i \quad (4)$$

здесь y_i – коэффициент каждого инженера для мультипликативной оценки, x_i – коэффициент по каждому критерию, n – количество критериев.

На рис. 28-30 представлены результаты аддитивной оценки каждой реализации.

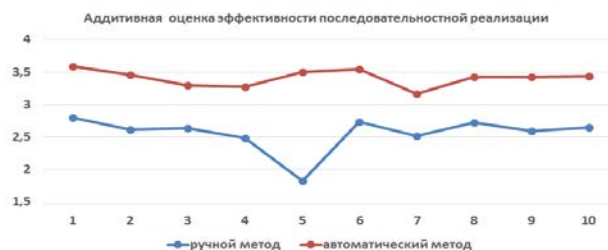


Рис 28. Аддитивная оценка эффективности последовательностной реализации

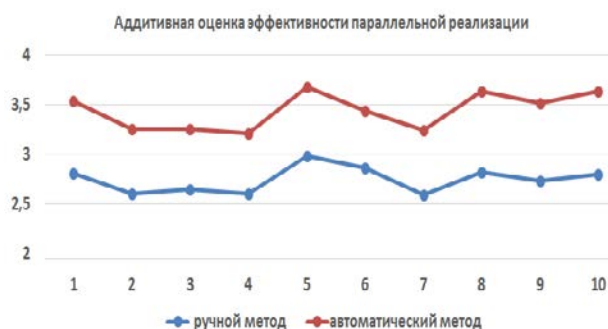


Рис 29. Аддитивная оценка эффективности параллельной реализации



Рис 30. Аддитивная оценка эффективности конвейерной реализации

На рис. 31-33 представлены результаты мультипликативной оценки каждой реализации.

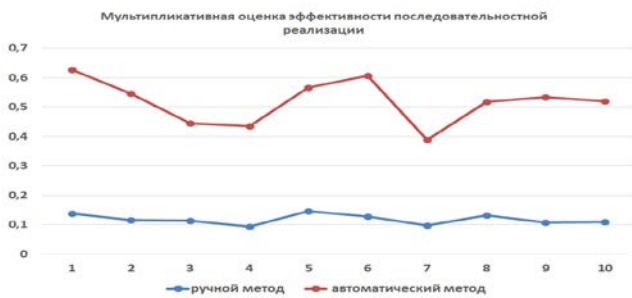


Рис. 31. Мультипликативная оценка эффективности последовательной реализации

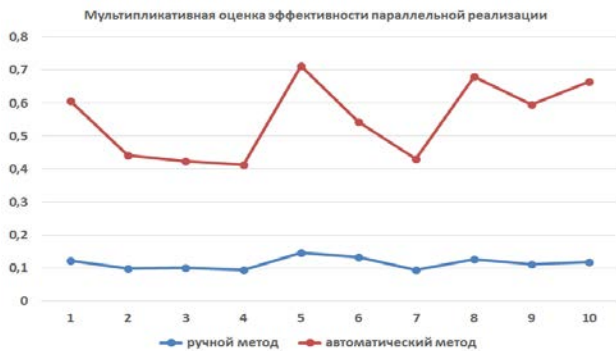


Рис. 32. Мультипликативная оценка эффективности параллельной реализации



Рис. 33. Мультипликативная оценка эффективности конвейерной реализации

В результате исследования получили, что в случае аддитивной оценки автоматический метод лучше в 1,301292012 раз (рис. 34).

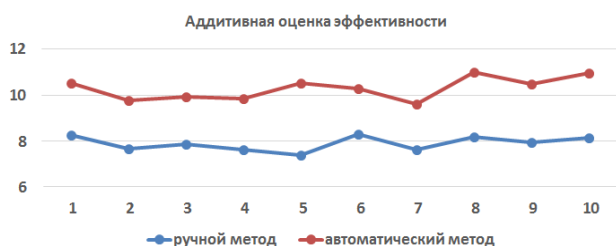


Рис. 34. Аддитивная оценка эффективности

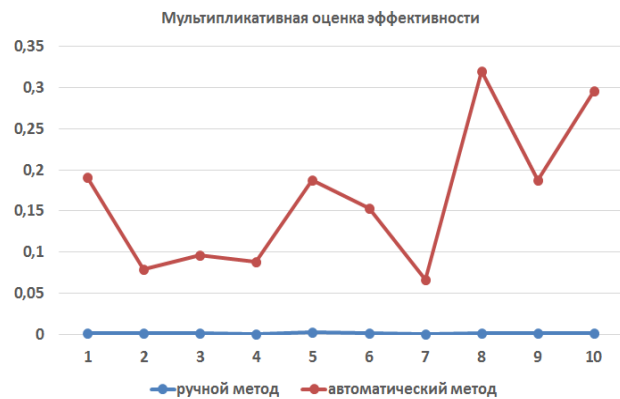


Рис. 35. Мультипликативная оценка эффективности

При мультипликативной оценке автоматический метод лучше в 133,2182966 раз. На рис. 35 видно, что значения ручного метода практически стремятся к нулю, в то время как в автоматическом методе значения стремятся к 1. Это доказывает, что автоматический метод действительно эффективнее.

Выводы

Разработаны программно-аппаратные реализации моделей, методов и структур данных для проектирования цифровых систем на кристаллах, которые включают процедуры создания спецификации, синтеза, тестирования, моделирования и верификации на основе предложенной инфраструктуры, учитывающей промышленные средства компаний Aldec и Xilinx. Рассмотрены вопросы тестирования разработанных программных продуктов на реальных цифровых проектах создания IP-Core как примитивов для реализации цифровых систем на кристаллах.

При этом достигнута *цель* – разработаны и верифицированы инфраструктурные модули проектирования цифровых систем на кристаллах, которые характеризуются параллельным выполнением мультиверсного синтеза функциональности, обеспечивающей существенное уменьшение времени создания проекта в условиях ограничения на аппаратные затраты.

Решены следующие задачи:

1. Разработан и описан метод мультиверсного синтеза управляющих и операционных автоматов в заданной инфраструктуре проектирования, ориентированных на архитектурные решения в метрике, минимизирующий время выполнения функциональности за счет распараллеливания операций при ограничении на аппаратные затраты.
2. Выполнена программная реализация моделей и методов мультиверсной разработки операционных устройств в рамках интегрированной системы проектирования функциональных и архитектурных решений SoC на основе использования продуктов верификации и синтеза компаний Aldec и Xilinx.
3. Выполнено тестирование и верификация программных модулей инфраструктуры проектирования цифровых систем на кристаллах, а также определена

эффективность предложенных моделей, методов и структур данных при создании реальных компонентов цифровых изделий.

Литература: 1. Патент 6 226 776 США, G 06 F 17/50. System for Converting Hardware Designs in High-Level Programming Language to Hardware Implementations / Yuri V. Panchul, Donald A. Soderman, Denis R. Coleman; заявитель и патентообладатель Synetry Corporation (США); заявл. 16.09.1997, опубл. 01.05.2001. <http://www.google.com/patents/about?id=BM4GAAAAEBAJ&dq=6226776> **2.** OR1200 OpenRISC Processor. http://opencores.org/wiki/index.php?title=OR1K_CPU_Cores&oldid=1404 **3.** Язык программирования C++. Международный стандарт ISO/IEC 14882. Второе издание, 15 октября 2003 г. American National Standards Institute, New York. **4.** Jayaram Bhasker. A SystemC Primer. 2002. **5.** Лисьяк В.В., Лисьяк Н.К. Обзор европейских производителей программного обеспечения САПР РЭА // Известия ЮФУ. Технические науки. 2008. №9 С.81-86. **6.** <https://github.com/systemc/systemc-2.2.0>. **7.** Баранов С.И. Синтез микропрограммных автоматов (граф-схемы и автоматы). Ленинград: Издательство 'Энергия'. Ленинградское отделение, 1979. **8.** SystemC: From the Ground Up, Second Edition. Black, D.C., Donovan, J., Bunton, B., Keist, A. Springer. 2010. P. 279. **9.** Weiwei Chen, Rainer Dömer, Weiwei Chen, Rainer Dömer. Transaction Level Models with Relaxed Timing. 2011. **10.** Tareq Hasan Khan. Automatic Generation of Transactors in SystemC. Concordia University. 2007. **11.** Ed Tam. A Microarchitectural Survey of Next Generation Microprocessors. 1997.

Transliterated bibliography:

1. Patent 6 226 776 SShA, G 06 F 17/50. System for Converting Hardware Designs in High-Level Programming Language to Hardware Implementations / Yuri V. Panchul, Donald A. Soderman, Denis R. Coleman; zayavitel i patentoobladatel Synetry Corporation (SShA); zayavl. 16.09.1997, opubl. 01.05.2001. <http://www.google.com/patents/about?id=BM4GAAAAEBAJ&dq=6226776>

2. OR1200 OpenRISC Processor. http://opencores.org/wiki/index.php?title=OR1K_CPU_Cores&oldid=1404

3. Yazyik programmirovaniya S . Mezhdunarodnyiy standart ISO/IEC 14882. Vtoroe izdanie, 15 oktyabrya 2003 g. American National Standards Institute, New York.

4. Jayaram Bhasker. A SystemC Primer. 2002.

5. Lisyak V.V., Lisyak N.K. Obzor evropeyskikh proizvoditeley programmogo obespecheniya SAPR REA // Izvestiya YuFU. Tehnicheskie nauki. 2008. #9 S.81-86.

6. <https://github.com/systemc/systemc-2.2.0>.

7. Baranov S.I. Sintez mikroprogrammnyih avtomatov (grafshemyi i avtomaty). Leningrad: Izdatelstvo 'Energiya'. Leningradskoe otdelenie, 1979.

8. SystemC: From the Ground Up, Second Edition. Black, D.C., Donovan, J., Bunton, B., Keist, A. Springer. 2010. P. 279.

9. Weiwei Chen, Rainer Dtsmer, Weiwei Chen, Rainer Dtsmer. Transaction Level Models with Relaxed Timing. 2011.

10. Tareq Hasan Khan. Automatic Generation of Transactors in SystemC. Concordia University. 2007.

11. Ed Tam. A Microarchitectural Survey of Next Generation Microprocessors. 1997.

Поступила в редколлегию 11.02.2016

Рецензент: д-р техн. наук, проф. Кривуля Г.Ф.

Обризан Владимир Игоревич (Obrizan Vladimir), старший преподаватель кафедры АПВТ ХНУРЭ. Научные интересы: компьютерная инженерия. Адрес: Украина, 61166, Харьков, пр. Науки, 14.



ИССЛЕДОВАНИЕ ЗАДАЧ ОПТИМАЛЬНОГО РАСПРЕДЕЛЕНИЯ ДОПУСТИМЫХ НАГРУЗОК НА СЕРВЕРЕ

ГВОЗДИНСКИЙ А.Н., СТОЛЯРЕНКО К.С.

Для минимизации возможных сбоев и корректной работы с данными исследуется алгоритм, с помощью которого решаются задачи для распределения допустимых нагрузок на сервере.

1. Введение

Актуальность исследования. Последние десятилетия XX века отмечены событиями, существенным образом трансформировавшими современную технологическую реальность. Речь идет об активном вхождении в жизнь общества новейших информационных технологий, произошедшем в результате бурного развития электроники, а также необходимости использования автоматизации бизнес-процессов как в коммерческой, так и некоммерческой деятельности предприятия. Внедрение IT-технологий и автоматизация рабочих мест кадров необходима для того, чтобы всегда оставаться конкурентоспособным на рынке как внешнем, так и внутреннем. Внедрение таких технологий и их использование дают массу преимуществ. Это может быть и быстрый поиск информации во внутренних базах или хранилищах, и исключение дублирования информации. К преимуществам можно отнести:

- увеличение объемов обработки информации;
- снижение времени на ее обработку;
- повышение качества обработки;
- автоматическое создание отчетов.

Все это автоматизируется с помощью серверов. Слово «сервер» произошло от английского глагола *serve*, что означает «служить, обслуживать», т.е. серверные устройства обслуживают пользователей и являются аппаратными устройствами и программными средствами. Как аппарат сервер представляет собой компьютер, который обслуживает другие компьютеры, а также принтеры, факсы и иные технические средства. Пользователи формируют для сервера задачи, а он их быстро и без ошибок решает с помощью определенных методов. В наше время невозможно себе представить работу банков, различных предприятий и научно-исследовательских учреждений без объединения информационных ресурсов в единую сеть, т.е. без сер-

веров. Даже факс можно отправить с помощью компьютера, а следовательно, и с помощью сервера. Поэтому нужно понимать всю важность и ценность серверов, их возможности и преимущества, а также различные назначения.

На любом предприятии используют серверы, создают многопользовательские центры, которые объединяют информацию всех работников-пользователей, что обеспечивает моментальный и очень удобный доступ к ней. Сервер обрабатывает огромный массив данных и предоставляет обмен информацией между всеми участниками одного проекта. Работа в учреждении или на предприятии становится более эффективной и слаженной, так как увеличивается скорость выполнения задач и повышается ее надежность. Соответственно уменьшается количество ошибок и просчетов. С помощью серверов можно объединить также и материальные ресурсы, такие как факсы, принтеры и др. Поэтому понятно, что все это приводит только к огромной экономии времени и денег. Благодаря серверу мы общаемся по электронной почте, в сбербанках сотрудники распечатывают нам документы, а в супермаркетах кассиры очень быстро нас обслуживают. Понятно, что совокупность серверов, которые обеспечивают доступ к удаленным собраниям ресурсов, дает такое многообразие наполнения Интернета. Все сайты в Интернете хранятся на серверах. А доступ к серверам предоставляют хостинговые компании. Все это называется виртуальным хостингом. Безусловно, чем надежнее, быстрее и безопаснее работает сервер, тем дороже обходится и серверное оборудование, а также программное обеспечение. Поэтому сервер является очень выгодным компонентом при работе в сети. Статистика гласит о том, что с каждым годом Интернет разрастается все быстрее и быстрее. В 2012 году web вырос более чем в два раза: год начался с 330 млн зарегистрированных веб-сайтов, а к декабрю эта цифра возросла к 743 млн. Сайт «internetlivesstats.com» в режиме онлайн фиксирует каждый новый сайт, блог сообщение и твиты, которые появляются во всемирной паутине. Количество созданных сайтов к 2016 году перевалило за 1 млрд.

Сущность исследования заключается в том, чтобы найти оптимальное распределение допустимых нагрузок серверных приложений или служб на сервере, учитывая критерии передачи данных для точной их передачи и хранения.

Цели исследования заключаются в минимизации числа допустимых ошибок на сервере, а также минимизации времени и затрат на обработку запросов приложений (служб); распределении допустимых нагрузок серверных приложений (служб) между *m* серверами компьютерной сети. Для достижения данных целей необходимо решить задачи анализа алгоритмов оптимизации, выбрать конкретный алгоритм, который бы мог справиться с проблемой.

Индустрия электронной коммерции продолжает развиваться, и сегодня все новые компании начинают

общаться со своими клиентами с помощью Web. Высокопроизводительный сайт, способный предоставлять материалы быстро и без сбоев, не только помогает привлекать новых клиентов, но и становится важнейшей предпосылкой успешной деятельности предприятий электронной торговли.

2. Постановка задачи и математическая модель

Предприятие, работа которого зависит от компьютеров и компьютерной сети, нуждается в наличии серверов, для которых нужно определить оптимальность их работы. Каждый сервер имеет характеристики производительности. При обнаружении неполадки одного из серверов выходом из ситуации будет взаимозаменяемость их при обработке сетевых приложений (служб). Таким образом, для того, чтобы выполнить задачу, нужно составить план распределения приложений (служб) на серверы с заданными ограничениями при условии:

- минимум времени на обработку запросов;
- минимум затрат на обработку запросов приложений (служб);
- минимум ошибок.

Исходя из общей математической модели задачи распределения нагрузок, с учетом всех особенностей поставленной перед нами задачи построим конкретную математическую модель, с помощью которой можно будет вычислить экстремумы по трем условиям:

- минимум времени на обработку запросов:

$$F(x) = 60x_1 + 25x_2 + 15x_3 \rightarrow \min \quad (1)$$

при условиях:

$$0.3x_1 + 0.5x_2 + 0.4x_3 \leq 300; \quad (2)$$

$$0.5x_1 + 0.7x_2 + 0.8x_3 \leq 800; \quad (3)$$

$$1.3x_1 + 0.1x_2 + 1.2x_3 \leq 200; \quad (4)$$

$$x_1 \geq 500; x_2 \geq 300; \quad (5)$$

- минимум затрат на обработку запросов приложений (служб):

$$F(x) = 55x_1 + 30x_2 + 8x_3 \rightarrow \min \quad (6)$$

при условиях:

$$0.8x_1 + 0.5x_2 + 0.3x_3 \leq 250; \quad (7)$$

$$0.3x_1 + 0.5x_2 + 0.6x_3 \leq 500; \quad (8)$$

$$0.8x_1 + 0.1x_2 + 1.1x_3 \leq 150; \quad (9)$$

$$x_1 \geq 400; x_2 \geq 150; \quad (10)$$

- минимум ошибок:

$$F(x) = 55x_1 + 30x_2 + 8x_3 \rightarrow \min \quad (11)$$

при условиях:

$$0.6x_1 + 0.7x_2 + 0.4x_3 \leq 200; \quad (12)$$

$$0.6x_1 + 0.75x_2 + 0.8x_3 \leq 700; \quad (13)$$

$$1.4x_1 + 0.5x_2 + 1.4x_3 \leq 300; \quad (14)$$

$$x_1 \geq 550; x_2 \geq 350; \quad (15)$$

Целевые функции $F(x)$, представленные в (1), (6), (11), выражают возможные значения нагрузок, при которых сервер может корректно обрабатывать, передавать, принимать данные, определенные пользователем. Таким образом, для получения оптимального плана необходимо минимизировать целевые функции $F(x)$ с соблюдением всех условий задачи, которые накладывают ограничения на $F(x)$.

В выражениях (1)-(15) описана задача для нахождения экстремумов линейных функций в условиях системы линейных ограничений.

3. Исследования методов решения

Исходя из полученной обобщенной математической модели решения, изложенных в постановке задач, для решения обычно используют математические методы поиска экстремумов. Среди них наиболее развитыми и широко используемыми являются методы математического программирования, методы эволюционной оптимизации (генетические алгоритмы) и классические итерационные методы достижения экстремумов функций.

К математическому программированию относятся: 1. Линейное программирование. 2. Нелинейное программирование. 3. Теория графов. 4. Целочисленное программирование. 5. Стохастическое линейное программирование. 6. Задачи теории массового обслуживания. 7. Динамическое программирование. 8. Геометрическое программирование.

Ввиду того, что задача распределения нагрузок на сервер имеет вид задачи на нахождение экстремумов функций и состоит из трех систем уравнений, что описано в ее условиях, и данные три функции цели сводятся к нахождению минимумов, то решать задачу нужно универсальным методом линейного программирования – симплекс-метод и стохастический Байесов принцип.

Симплекс-метод. Данный метод решения задачи линейного программирования является универсальным. С его помощью можно решить любую задачу такого типа. В его основу положена идея последовательного улучшения получаемого решения.

Геометрический смысл данной задачи состоит в том, чтобы последовательно перейти от одной вершины многогранника ограничения нагрузок на сервер к соседней таким образом, чтобы целевая функция имела наилучшее значение до тех пор, пока не будет найдено оптимальное решение распределения нагрузок на сервер. В нашем случае – тах нагрузок.

Процесс нахождения оптимального плана предполагает реализацию 3-х элементов:

- 1) определение первоначального допустимого базисного решения задачи распределения нагрузок;
- 2) правило перехода к лучшему решению для нахождения \max нагрузок;
- 3) критерий проверки оптимальности для решения данной задачи.

Эта задача касается анализа линейного программирования. Но существуют еще модели стохастического характера, которые можно свести к детерминированным линейным моделям большей размерности; при этом увеличение размерности в реальных задачах настолько значительно, что нахождение оптимального решения подчас оказывается практически невозможным. Стоит рассмотреть стохастический Байесов принцип. Данный метод в условиях частичной неопределенности предполагает, что возможным состояниям природы можно приписать вероятность их наступления и, определив математическое ожидание выигрыша для каждого решения, выбрать то, которое обеспечит наибольшее значение выигрыша.

При этом предполагается, что ситуация, в которой принимается решение, характеризуется следующими обстоятельствами:

- вероятности появления состояния F_j известны и не зависят от времени;
- решение реализуется (теоретически) бесконечно много раз;
- для малого числа реализаций решения допускается некоторый риск.

При достаточно большом количестве реализаций среднее значение постепенно стабилизируется. Поэтому при полной (бесконечной) реализации какой-либо риск практически исключён.

4. Алгоритмы решения задачи

Для конкретных методов решений стоит сформулировать следующие примеры:

Симплексный метод включает в себя ряд этапов и может быть сформулирован в виде алгоритма.

Шаг 1. Формулировка ЗЛП (формирование целевой функции и системы ограничений распределения нагрузок на сервер). Из этого следует, что решается задача на отыскание максимума. Следует привести общую постановку данной задачи для достижения минимума времени на обработку запросов:

$$F(x) = 60x_1 + 25x_2 + 15x_3 \rightarrow \min \quad (16)$$

при условии, что:

$$0.3x_1 + 0.5x_2 + 0.4x_3 \leq 300; \quad (17)$$

$$0.5x_1 + 0.7x_2 + 0.8x_3 \leq 800; \quad (18)$$

$$1.3x_1 + 0.1x_2 + 1.2x_3 \leq 200; \quad (19)$$

$$x_1 \geq 500; x_2 \geq 300; \quad (20)$$

В этих выражениях x_1, x_2, x_3 – критерии, по которым рассчитываются нагрузки на серверы.

Шаг 2. Приведение задачи к канонической форме, что является переводом функции ограничений в систему уравнений. Для этого нужно вводить дополнительные неотрицательные переменные. Следует знать, что эти переменные должны иметь тот же знак, что и свободные члены системы ограничений. Если это условие не выполняется, то нужно использовать метод искусственного базиса (М-метод).

В результате получим каноническую форму задачи для распределения нагрузок приложений на сервере:

$$F(x) = 60x_1 + 25x_2 + 15x_3 \rightarrow \min \quad (21)$$

при тех же ограничениях:

$$0.3x_1 + 0.5x_2 + 0.4x_3 - x_4 = 300; \quad (22)$$

$$0.5x_1 + 0.7x_2 + 0.8x_3 - x_5 = 800; \quad (23)$$

$$1.3x_1 + 0.1x_2 + 1.2x_3 - x_6 = 200; \quad (24)$$

В результате этого можно получить систему ограничений данной задачи по 3-м критериям.

Шаг 3. Построение исходной симплекс-таблицы, в результате чего можно получить первоначальный план решения задачи. Он показан в табл. 1.

Таблица 1

	C_j		$C_1=60$	$C_2=25$	$C_3=15$	$C_4=0$	$C_5=0$	$C_6=0$
Базис	C_i	$P_0=X_0$	P_1	P_2	P_3	P_4	P_5	P_6
P_4	0	300	0,3	0,5	0,4	-1	0	0
P_5	0	800	0,5	0,7	0,8	0	-1	0
P_6	0	200	1,3	0,1	1,2	0	0	-1
F_j		0	0	0	0	0	0	0
Δ_j		-	-60	-25	-15	0	0	0

Итак, в левом столбце записываются основные (базисные) переменные, в первой строке перечисляются все переменные задачи. Крайний правый столбец содержит свободные члены системы ограничений b_1, b_2, \dots, b_m . В последней строке таблицы, которая называется оценочной, записываются коэффициенты целевой функции, а также ее значение. В рабочую область таблицы занесены коэффициенты a_{ij} при переменных системы ограничений.

Шаг 4. Проверка условия $c_{ij} \geq 0$. Если условие не действительно – осуществляется переход к шагу 5, если да – задача решена. Таким образом, на данном шаге проверяется наличие положительных элементов в последней строке симплексной таблицы.

Шаг 5. Выбор разрешающего столбца и строки. Столбец выбирается в соответствии со следующим условием:

$$\theta = x_j / x_{ik}, x_{ik} \geq 0, \quad (25)$$

где k – номер разрешающего столбца.

Таким образом, при определении разрешающего столбца просматривается выбранная строка симплексной таблицы и в ней отыскивается положительный наибольший элемент. Для данной задачи этим элементом будет $x_k = 1.3$.

Шаг 6. Проверка условий:

– если $x_k \geq 0$, то опорный план и есть решением задачи;

– $x_k < 0$ следует рассмотреть шаг 7.

Проверять первый раз это условие не стоит, так как изначально опорный план задачи распределения ресурсов не имеет отрицательных значений, поэтому нужно переходить к шагу 7.

Шаг 7. Пересчет элементов симплекс-таблицы и переход к новому базисному решению. Для элементов разрешающей строки используются следующие формулы:

$$x'_{rj} = x_{rj} / x_{rk}, \quad (26)$$

$$x'_{ij} = x_{ij} - x_{ri} * x_{ik} / x_{rk}, \quad (27)$$

$$j = \overline{0, (r + m)}, \quad (28)$$

где r – номер разрешающей строки; k – номер разрешающего столбца; x'_{rj}, x'_{ij} – новые значения пересчитываемых элементов.

По окончании пересчета осуществляется возврат к шагу 4.

Если в результате выполнения шага 4 были получены положительные значения строки j с, то оптимальный план задачи был достигнут.

Таким образом, с помощью симплекс-метода было найдено решение задачи распределения нагрузок приложений (служб) на сервере.

Базовый план выглядит так:

$$x^* = \|112.9; 532.26; 0; 0; 370.97; 0\|.$$

В результате получили значение целевой функции:

$$F(x) = 25 * 532.26 + 0 * 370.97 + 60 * 112.9 = 20080.$$

Допустимые нагрузки на данном сервере с ограничением на минимальное время выполнения действий (приложений, служб) на сервере могут составить не больше 20080 запросов.

Аналогично вычислениям на минимальное время выполнения служб можно получить значение критерия на минимум затрат при обработке запросов приложений (служб).

Оптимальный план будет выглядеть так:

$$x^* = \|156.5; 250; 0; 0; 328.13; 0\|,$$

$$F(x) = 30 * 250 + 0 * 328.13 + 55 * 156.5 = 16093.$$

Минимальное время выполнения служб будет при подаче 16093 запросов на сервер.

При достижении минимума ошибок во время выполнения обработки информации на сервере с помощью служб или приложений будут другие данные:

$$x^* = \|0; 0; 214.29; 114.29; 528.57; 0\|,$$

$$F(x) = 0 * 114.29 + 0 * 528.57 + 30 * 214.29 = 6428.$$

Минимум ошибок будет достигнуто при выполнении 6428 запросов.

Критерий Байеса-Лапласа применяют к ситуации, в которой принимается решение о распределении нагрузок приложений на сервер.

Расчет критерия Байеса-Лапласа выглядит так:

$$K_i^{BL} = \sum_j p_j u_{ij}, \quad (29)$$

$$i^{BL} = \arg \max_i (K_i^{BL}). \quad (30)$$

Шаг 1. Определение вероятностей состояний сервера для данной задачи. Они будут равны $p_1=0.3$; $p_2=0.5$; $p_3=0.2$.

Шаг 2. Составление матрицы ценности.

Соответственно она будет состоять из альтернативных решений, которые включают в себя состояния сервера и решений.

Это показано в табл. 2.

Таблица 2

Альтернативные решения	Состояния сервера		
	1	2	3
Решение 1	10000	15366	9870
Решение 2	8750	14800	7650
Решение 3	9630	15560	9320

Шаг 3. Определение критериев по Байесу-Лапласу. Значения критериев будут равны:

$$K_1^{BL} = 0.3 * 10000 + 0.5 * 15366 + 0.2 * 9870 = 12657;$$

$$K_2^{BL} = 11555;$$

$$K_3^{BL} = 12533;$$

Шаг 4. Определение наилучшего решения – максимального. Для данной задачи теперь можно определить, что наилучшим решением является первое:

$$i^{BL} = \arg \max(12657, 11555, 12533) = 1,$$

$$K^{BL} = 1.$$

Таким образом, найдено решение для распределения нагрузок на сервере: из трех решений первое оказалось наилучшим по критерию Байеса-Лапласа.

5. Выводы

В ходе выполнения работы был рассмотрен метод оптимизации распределения допустимых нагрузок приложений (служб) на сервер с использованием определенных условий. Основным инструментом при решении данной задачи стало математическое моделирование – формальное описание изучаемого явления и исследование с помощью математического аппарата. Была составлена математическая модель задачи. Для нахождения решений выбран симплекс-метод и стохастический принцип Байеса-Лапласа.

Научная новизна. Важность использования сервера, а также регулярной проверки наличия доступа к нему в век информационных технологий и глобального присутствия любого бизнеса в Интернете трудно переоценить. При неработающем хостинге, кроме недоступности некоторого сайта посетителям, могут быть и более серьезные проблемы. Если робот поисковой системы попадает на необрабатываемый сайт, то он может быть исключен из выдачи поисковика. Соответственно есть возможность в потере очень многих посетителей и потенциальных клиентов. Таким образом, данное понятие означает тот временной промежуток, в течение которого сервер работает без неполадок.

Практическая значимость. Полученные научные результаты данного исследования имеют большое практическое значение для разработки плана нагрузок с точки зрения минимизации ошибок, времени на обработку запросов приложений (служб) и затрат работы технического оборудования.

Вопрос обслуживания сервера очень часто встает перед разработчиками или владельцами крупных высоко посещаемых проектов, ведь масштабов обычного виртуального хостинга уже не хватает, а VIP хостинг дорогой и не выгоден с точки зрения эффективности. В этой связи приходится арендовать свой сервер, однако его необходимо администрировать, управлять, защищать и так далее. Серверы используют различные компании в своих офисах для обеспечения эффективной работы персонала, финансовой и бухгалтерской службы, которые могут работать в одной программе 1С, например, чтобы всегда иметь свежие

обновленные данные. Серверы в компаниях обеспечивают выход сотрудников в Интернет, регламентируют доступ к определенным ресурсам, чтобы персонал занимался работой, а не социальными сетями и другими развлечениями.

Работа сотрудников любой фирмы во многом зависит от эффективности работы сервера. Хорошо отлаженный сервер увеличивает производительность и эффективность работы компании в целом в несколько раз. Сбои на сервере могут привести не только к отсутствию доступа в сеть Интернет, но и к потере ценных данных, простою в работе, потере заявок, просрочке выполнения работ и многое другое. Сегодня почти все компании, нацеленные на долгосрочную и плодотворную деятельность, не могут обойтись без развитой компьютерной сети, а значит и без сервера, который эту сеть обслуживает и организует. Здесь очень важно подумать не только об установке и настройке сервера, но и о его дальнейшем качественном и квалифицированном обслуживании.

Литература: 1. Гвоздинский А.Н., Губин В.А., Шергин В.Л. Методы оптимизации в организационном управлении», ХНУРЭ, 2014. 395с.2. Бондаренко М.Ф. Оптимізаційні задачі в системах прийняття рішень: підручник / М.Ф. Бондаренко, А.М. Гвоздинський. Харків: ХТУРЕ, 1998. 219 с.

Transliterated bibliography:

1. Gvozdzinskiy A.N., Gubin V.A., Shergin V.L. Metody optimizatsii v organizatsionnom upravlenii», HNURE, 2014. 395s.
2. Bondarenko M.F. Optimizatsiyni zadachi v sistemah priynyattya rishen: pdruchnik / M.F. Bondarenko, A.M. Gvozdzinskiy. Harkiv: NTURE, 1998. 219 s.

Поступила в редколлегию 23.06.2016

Рецензент: д-р техн. наук, проф. Пуятин Е.П.

Гвоздинский Анатолий Николаевич, канд. техн. наук, профессор кафедры искусственного интеллекта ХНУРЭ. Научные интересы: оптимизация процедур принятия решения в сложных системах управления. Адрес: Украина, 61166, Харьков, ул. акад. Ляпунова, 7, кв. 9, тел. 702-38-33.

Столяренко Константин Сергеевич, студент группы СА-14-1 кафедры прикладной математики ХНУРЭ. Научные интересы: проектирование клиент-серверных приложений под мобильную платформу iOS, разработка баз данных и их комплексное взаимодействие. Адрес: Украина, 61072, Харьков, пер. Отакара Яроша, 12А, кв. 155, тел. +38066-706-58-69.

МОДЕЛИРОВАНИЕ АРХИТЕКТУРЫ И ПОСЛЕДОВАТЕЛЬНОСТИ ВЗАИМОДЕЙСТВИЯ КОМПОНЕНТОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ИНФОРМАЦИОННОЙ СИСТЕМЫ

ЧАЙНИКОВ С.И., СОЛОДОВНИКОВ А.С.

Предлагается графовая модель архитектуры программного обеспечения информационной системы и автоматная модель проверки выполнения ограничений к нему. Графовая модель описывает архитектуру с избыточной функциональностью и используется для конфигурирования рабочих мест на основе требований конечного пользователя. Выполнимость требований проверяется на базе автоматной модели, которая также используется для управления вычислительными процессами.

1. Обзор существующих методов автоматизированного синтеза программного обеспечения информационной системы

Для процесса проектирования программного обеспечения (ПО) за основу может быть взята одна из существующих технологий проектирования (SADT, IDEF, SSADM, Meris), которые используют однотипные этапы жизненного цикла (ЖЦ) информационных систем (ИС). В рамках ЖЦ для обеспечения автоматизации процесса синтеза необходимо формализованное описание как самой предметной области (ПрО), так и структуры ПО вместе с требованиями к конечному продукту. Использование такого подхода [1] обосновывает создание инструментария – так называемого генератора проектов – на базе совокупности формальных документов, адекватно отражающих ПрО, который позволяет на конечных этапах генерировать программный код системы и выполнять технологическую сборку.

Разработки в данном направлении велись с 80-х годов [2], при этом методы имеют различные недостатки: например, использование генерируемых скриптов вместе с текстами программного кода для сборки программ проекта по исходным текстам в соответствии с выбранной платформой, в связи с чем необходимо дополнительное ПО, анализирующее полученные скрипты. Дополнительное ПО (или программные инструментари), позволяющее осуществлять синтез программы, может работать в диалоговом режиме и быть основанным на: 1) запоминании состояния задачи на любом этапе в целях восстановления ее состояния до заданного момента времени; 2) проверке исходных данных задачи до ее решения, в процессе решения и после него; 3) оперативном вводе исправлений в исходных данных; 4) предоставлении пользователю возможности многосеансной работы. Указанные особенности являются преимуществом и позволяют оптимизировать процессы синтеза. Такой функ-

ционал позволяет пользователю прорабатывать различные стратегии решения задачи, минимизировать время ожидания решения задачи за счет снижения количества ошибок, приводящих к сбою вычислительного процесса (ВП), и разбивать последовательность действий пользователя на этапы (сеансы) с длительными временными перерывами между ними. Однако для осуществления синтеза программного продукта с помощью подобных программных инструментов требуется развитие проблемно-ориентированного языка.

В рамках использования диалоговых систем автоматизированного синтеза ПО и построения ВП обычно используются графовые модели вычислений [3, 4], которые позволяют формировать ВП на основе маршрутов, выделяемых на графе. В дальнейшем развитие диалоговых систем обозначилось в сторону проектирования диалога на базе естественного языка и получило развитие путем применения искусственного интеллекта и баз знаний.

На современном этапе существует также подход к автоматизированной генерации ПО, основывающийся на использовании алгебраических спецификаций – системах алгоритмических алгебр (САА) В.М. Глушкова [5] и методе диалогового конструирования синтаксически правильных программ [6]. Особенность заключается в совместном использовании трех представлений алгоритма при его конструировании: аналитического (САА-формула), естественно-лингвистического и графового (граф-схемы). Такое направление получило развитие в представлении алгоритмов в виде САА-М-схем со смещением акцента в сторону разработки инструментария автоматизированного преобразования параллельных алгоритмов [7] с использованием технологий MPI (Message Passing Interface) и многопоточности в Java. Недостатком этих методов является то, что их реализация зависит от конкретного языка программирования.

Выделяют также технологию автоматического синтеза ПО с использованием онтологии прецедентов [8], в которой прецеденты определяются как пара $\langle S, P \rangle$, где S — спецификация и P — соответствующая ей программа. Такое онтологическое описание позволяет накапливать опыт разработки, выполнять автоматическую классификацию программ на основе их спецификаций и построение программ путем адаптации известных решений.

Другое направление в автоматическом синтезе ПО получило развитие с разработкой технологии генетических алгоритмов. Они позволяют определять структуру управляющего автомата, который в свою очередь является системой вложенных и взаимовызываемых автоматов [9]. Такой подход реализует технологию автоматного программирования и обладает такими преимуществами как автоматизация процесса верификации, документированности, упрощение процедуры внесения изменений.

Автоматный подход широко применяется не только в синтезе программного кода [10], но и в управлении поведением самой системы, запущенной на выполнение [11].

Использование архитектурных шаблонов и проблемно-ориентированных языков описания архитектуры применяется в развитии технологии построения композитных приложений [12, 13]. Примером служит инструментально-технологическая платформа CLAVIRE, использующая проблемно-ориентированный язык EasyFlow и позволяющая автоматизировать формирование архитектурных шаблонов, на основе которых выполняется генерация сценариев запуска приложения в распределенной среде [14].

С усложнением предметных областей увеличивается сложность аппаратного обеспечения и ПО. Повышаются требования к эффективности и производительности ПО (стандарт ISO/IEC 25041:2012). Пока ПО по скорости развития отстает от аппаратного, поэтому требует использования особых технологий по оптимизации своей работы.

Сервисно-ориентированные технологии организации кластерных вычислений являются на данный момент наиболее перспективными. Они порождают новое ответвление – так называемые облачные технологии. Однако в данном направлении присутствует существенная проблема – обеспечение безопасности данных, находящихся во владении сторонних организаций. К известным типам угроз (сетевые атаки, вредоносное ПО, уязвимости в приложениях и ОС) при использовании облачных технологий добавляются сложности, связанные с контролем среды (гипервизора), трафика между гостевыми машинами и разграничением прав доступа [15].

2. Архитектура программного инструментария для автоматизированного синтеза программного обеспечения

Каждое из данных направлений требует своих подходов, эффективных для определенного класса задач и архитектур вычислительных систем, а также детального изучения и развития. Поэтому современные технологии проектирования ПО должны быть сориентированы на перспективу развития аппаратного обеспечения и иметь возможность адаптации их принципов к многопроцессорным и распределенным архитектурам.

На основе приведенного анализа можно выделить следующие методы синтеза ПО ИС:

- 1) ручной;
- 2) автоматизированный (в том числе с использованием метода диалогового конструирования);
- 3) автоматический.

При этом в рамках автоматизированного и автоматического методов наиболее широко используются следующие средства формализации:

- 1) логико-алгебраические спецификации;
- 2) автоматные модели;
- 3) графовые модели;
- 4) ADL-языки;
- 5) онтологические средства.

Перечисленные средства формализации могут применяться в рамках синтезирующего (на базе модели вычислений, отображающей понятия и отношения ПрО и программной спецификации), композиционного (на базе функций и операций композиции в логико-математической системе) и сборочного программирования (на базе модели сборки в виде ориентированного нагруженного графа) [12, 13].

Что касается методов проектирования и разработки ИС с использованием наиболее распространенного сборочного подхода, на данный момент существует следующая обобщающая классификация этих методов [13]:

- 1) модульно-ориентированный;
- 2) объектно-ориентированный;
- 3) компонентно-ориентированный;
- 4) метод генерации;
- 5) сервисно-ориентированный.

Каждый следующий метод является развитием предыдущего. Реализация этих методов основывается на использовании хранилищ готовых решений, компонент повторного использования, а также на наличие существенных проблем – обеспечение межмодульного интерфейса при сборке ИС [13] и существование конфликта между нефункциональными требованиями к компонентам.

В целях повышения эффективности синтеза ПО ИС с помощью CASE-средств и предоставления возможности управления действиями программы указывается на необходимость разделения программной системы на управляющий объект и объект управления. Управляющим объектом может служить некая исполнительная система или диспетчер. В большинстве случаев в составе инструментальных средств, позволяющих обеспечивать автоматизацию процессов сборки ПО, определяют компоновщик, формирующий общий программный код на основе атомарных фрагментов и алгоритма сборки, и так называемый диспетчер, который получается на выходе процесса сборки наряду с целевой ИС (рис. 1) [16].

Такой диспетчер позволяет управлять функционированием данной системы. Кроме того, при указанной архитектуре компоновщик может быть сориентирован на повторное использование программных модулей, компонентов, ПО. Может применяться версионность сборок, а архитектура ИС может быть расширена до включения в состав нескольких компаний-разработчиков ПО, что влечет за собой необходимость разработки процессов повторного применения

программных компонентов и механизмов управления этими процессами.

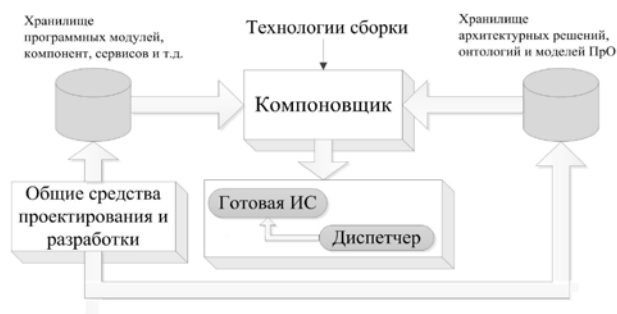


Рис. 1. Концептуальная структура инструментальных средств, реализующих сборку ИС

В основу работы компоновщика взята графовая модель архитектуры ПО ИС, которая содержит информацию о правилах взаимодействия программных модулей между собой. Работа над формированием этой графовой модели и вводом соответствующей информации осуществляется с помощью разработанного программного инструментария, поддерживающего разработчика в рамках процессов анализа требований к программным средствам, проектирования архитектуры, конструирования и комплексирования программных средств ЖЦ, создания ПО. При этом проверка корректности стыковки программных модулей и управление их взаимодействием ложится на автоматную модель ВП, которая синтезируется при помощи автоматного метода проверки выполнения ограничений к формируемому ПО. Разработчик в визуальной или в табличной форме описывает архитектуру будущего ПО с помощью графовой модели, на основе которой формируется автоматная модель ограничений формируемого ПО. Конечные автоматы (КА) данной модели при помощи архитектурных шаблонов реализуются в виде классов управляющих автоматов на базе объектно-ориентированных языков программирования и позволяют управлять реализацией ВП программы. Зададим формально информацию, получаемую из графовой модели архитектуры ПО ИС.

3. Модель архитектуры программного обеспечения информационной системы

Упомянутые инструментарии реализуют CASE-технологии, поддерживающие и автоматизирующие работы на стадиях ЖЦ, тем самым ускоряя процессы проектирования и разработки систем и повышая их качество. CASE-средства обладают достаточно большими преимуществами [17]: качество разрабатываемого ПО за счет средств автоматического контроля и генерации, повышают уровень технологической поддержки процессов разработки и сопровождения ПО, улучшают производительность и качество продукции при соблюдении стандартов и документирования. При использовании CASE-средств обнаруживают высокий уровень отдачи от инвестиций в инструментарий, возможность повторного использования компонентов разработки и поддержку адаптивности и сопровождения ПО. CASE-средства позволяют снизить

время создания системы, получая ее прототип и оценку на ранних стадиях проектирования. Позволяют осуществлять коллективную разработку ПО в режиме реального времени.

Данные достоинства характеризуют CASE-средства как высокоэффективный инструмент. Тем не менее, в зависимости от специфики и сложности конкретной ПРО всегда требуются усовершенствования методов, заложенных в этот инструментарий.

Автоматизация процесса синтеза с помощью CASE-средств базируется на упорядоченной и структурированной информации о необходимых программных библиотеках, участвующих в формировании программы, об автоматных моделях, встраиваемых в структуру формируемого ПО, о формализованном представлении архитектуры ПО в виде графовой модели и программных спецификаций, регламентирующих правила стыковки программных модулей и функционирование компоновщика ПО. Все эти данные должны сохраняться на каждой стадии проектирования и разработки ПО, чем обосновано использование специальных форматов данных, обеспечивающих хранение и использование необходимой информации для разработчика в файле проекта. Такая информация необходима для того, чтобы обеспечить взаимосвязь программных компонентов, управляющего КА и графовой модели. Учитывая, что разработчику, который использует программный инструментарий автоматизированного синтеза ПО, требуется визуально отображать данные графовой модели на экранных формах, а также обеспечить взаимосвязь данных графовой модели архитектуры ПО и автоматной модели проверки выполнения ограничений, определим формально кортеж G_{sp} . Этот кортеж специфицирует данные графовой модели и структуру файла проекта:

$$G_{sp} = \langle N, S, VC, AM, CG, D \rangle, \quad (1)$$

где N – множество координат вершин, размещаемых на рабочей области экранной формы; S – множество стадий проекта, фиксирующих текущее состояние процесса компиляции ПО; VC – множество характеристик вершин графовой модели, описывающих наличие компонент сильной связности; AM – множество атрибутов, определяющих взаимосвязь вершин в соответствии с матрицей смежности графовой модели; CG – множество атрибутов, определяющих взаимосвязь между вершинами в соответствии с матрицей смежности конденсированного графа; D – множество атрибутов, определяющих характеристики функций, которые сопоставляются с вершинами графовой модели в соответствии с отображением «вершина-функция».

Для кортежа (1) множество координат вершин определяется следующим образом:

$$N = \{m_1^N, m_2^N, \dots, m_k^N\}, \quad m_i^N = \{vx, vy\}, \quad (2)$$

где vx, vy – координаты i -й вершины. Стадии проекта определяются как

$$S = \{s_1, s_2, \dots, s_r\}, \quad s_i = \{\text{True}, \text{False}\}, \quad (3)$$

где s_i – логическая переменная, обозначающая текущую активную стадию проекта. Это может быть, например, стадия, на которой осуществляется ввод матрицы смежности графовой модели или стадия проверки корректности введенной информации.

Множество характеристик вершин графовой модели определяется так:

$$VC = \{m_1^{VC}, m_2^{VC}, \dots, m_n^{VC}\}, \quad (4)$$

здесь $m_i^{VC} = \{isSubG, Comp\}$ – подмножество, состоящее из двух элементов: $isSubG$ – логическая переменная, которая определяет наличие вложенного подграфа для супервершины (т.е. той вершины, которая образовалась после процесса конденсации и заменила собой часть графа $G = \langle V, X \rangle$; подмножество $Comp$ – перечисляет номера вершин, принадлежащих компоненте сильной связности. Если задать $G_i^{SC} = \langle V^{SC}, X^{SC} \rangle$ (где V^{SC} – множество вершин, а X^{SC} – множество ориентированных дуг) как i -ю компоненту сильной связности $G_i^{SC} \subset G$, обнаруживаемую при одноименном поиске, то тогда множество $Comp$ будет совпадать с множеством V^{SC} , т.е. $Comp = V^{SC}$, где $V^{SC} \subset V$.

Множество атрибутов, позволяющих задать матрицу смежности ориентированного графа в списочном виде, определяется как

$$AM = \{m_1^{AM}, m_2^{AM}, \dots, m_k^{AM}\}, \quad (5)$$

где каждый атрибут $m_i^{AM} = \{V_{adj}\}$ представляет собой список вершин $v_j \in V_{adj}$, в которые идет дуга $x_{ij} \in X$ из вершины $v_i \in V$, при этом $V_{adj} \subset V$.

Матрица смежности конденсированного графа задается в списочном виде с помощью множества

$$CG = \{m_1^{CG}, m_2^{CG}, \dots, m_p^{CG}\}, \quad (6)$$

где каждый атрибут m_i^{CG} , по аналогии с (5), содержит список вершин $v_j \in V_{adj}^{CG}$, в которые идет дуга из вершины $v_i \in V$, но, в отличие от (5), дополняется подмножеством состояний ВП ST^{CG} , соответствующего i -й вершине, и подмножеством характеристик архивных данных AR^{CG} . Элемент множества CG определяется таким образом:

$$m_i^{CG} = \{V_{adj}^{CG}, ST^{CG}, AR^{CG}\}, \quad (7)$$

где $V_{adj}^{CG} \subset V$, $ST^{CG} = \{0, 1, 2\}$, $AR^{CG} = \{0, 1, 2\}$.

Другими словами, если задать переменную e^{st} , которая принимает значения из подмножества ST^{CG} , и переменную e^{ar} , принимающую значения из подмножества AR^{CG} , то

$$e^{st} = \begin{cases} 0, & \text{если ВП не запущен;} \\ 1, & \text{если ВП запущен;} \\ 2, & \text{если ВП выполнен.} \end{cases}$$

и также

$$e^{ar} = \begin{cases} 0, & \text{если архив отсутствует;} \\ 1, & \text{если запущен процесс архивации;} \\ 2, & \text{если архив существует.} \end{cases}$$

Разделение информации об исходном и конденсированном графе продиктовано необходимостью сохранять информацию о слоях графовой модели и обеспечивать быстрый доступ к нужному уровню.

Множество функций, сопоставляемых с вершинами, определяется таким образом:

$$D = \{p_1^D, p_2^D, \dots, p_q^D\}, \quad (8)$$

где

$$p_i^D = \{\text{UID}, \text{PName}, \text{PMName}, \text{PUPath}, \text{PDes}, \text{SPath}\} \quad (9)$$

– подмножество элементов, необходимых для описания программных компонент или модулей, которые эти функции выполняют. Для этого определяем UID как уникальный идентификатор ВП; PName – имя процесса, используемое компоновщиком, при определении логической связи между вершиной графовой модели и программным модулем; PMName – реальное имя программного модуля (или библиотеки); PUPath – физический путь к файлу программного модуля; SPath – физический путь к файлу спецификации, которая определяет основные характеристики модуля и является ссылкой на спецификацию модуля Sp_i ; PDes – дополнительные сведения об элементе. По сути, атрибут PName является синтаксическим уникальным именем программного модуля, а PMName – реальным физическим именем (семантическим) процесса или модуля, на которое могут ссылаться несколько синтаксических имен.

Разработчик, описывая архитектуру ПО, сопоставляет вершины графовой модели архитектуры ПО ИС с программными модулями, которые выполняют свои функции, и имеет дело прежде всего с атрибутами PName, PMName, PUPath и SPath. Уникальное значение атрибута UID формируется автоматически, а присваивание значения атрибуту PDes является опциональным, т.е., в том случае, если требуются дополнительные данные по программному модулю. Значения PName и PMName теоретически могут совпадать, но на практике это не будет правильным подходом, поскольку очень часто можно найти однотипные функ-

ции для различных подсистем ПО, которые могут быть выполнены одним программным модулем, хранят один экземпляр модуля, используя в других случаях ссылки на него, как это общепринято для стандартной практики использования динамических или статических библиотек. Однако применение различных имен (синтаксических) для одного и того же программного модуля при формировании кортежа P_i^D (9) продиктовано желанием упростить представление и обработку графовой модели, уменьшив количество связей между вершинами, чрезмерное количество которых могло бы привести к появлению дополнительных компонент сильной связности, а значит и супервершин.

Документ, предоставляющий проектировщику необходимую информацию для синтеза ПО, может быть согласован по формату файла с форматом одного из XML-стандартов (например, языка GraphML).

Используя объектно-ориентированный подход, задается принцип взаимодействия программных модулей. Этот принцип базируется на интерфейсах класса, который содержит сигнатуры его компонентов (запросов и команд), а также семантические свойства: предусловия и постусловия запросов и команд, инварианты класса. Для объектно-ориентированного подхода предлагается использовать шаблоны КА (автомата Мили) при разработке управляющего программного модуля (диспетчера) (рис. 2) [18].

Беря во внимание шаблоны проектирования КА и заявленные требования к исполнителю для объектно-ориентированных программ, зададим автоматную модель проверки выполнения ограничений к формируемому ПО.

Для этого примем допущение, что V – является множеством вершин подграфа G , который определяется для текущей задачи, запускаемой пользователем. Обозначим текущий номер яруса буквой L ; $I_{\max}(L)$ – количество вершин на ярусе L ; $L_{\max}(G)$ – количество ярусов для подграфа G . Обозначим вычислительный процесс, соответствующий вершине v_i , как $P(V_i)$.

Обозначим вершину $v_i \in V$, которая находится на ярусе L , как $V_i(V, L)$. Также обозначим количество входных дуг (полустепень захода) для вершины $v_i \in V$ как $DEG_IN(V_i)$. А вершину v_k , из которой существует направленная дуга (v_k, v_i) , как V_k . На этом основании зададим автомат $A_{ForwardSM}$ (рис.3).

ЛОМeЛОМeД Для диаграммы автомата $A_{ForwardSM}$ (см.рис.3) заданы распараллеливающие и синхронизирующие переходы (табл. 1), обеспечивающие ввод автомата одновременно в несколько состояний (табл. 2). Основные задачи, которые возложены на описанный КА, это:

- 1) формирование подграфа задачи;
- 2) проверка состояния задачи;
- 3) проверка результата на корректность;
- 4) изменение состояния задачи.

Объектом управления для автомата $A_{ForwardSM}$ является программный модуль $ForwardUnit$, контролирующий прохождение вычислительного процесса по подграфу.

4. Выводы

Графовая модель архитектуры ПО ИС, содержащая информацию, необходимую для формирования архитектуры ПО с избыточной для заданной ПрО функциональностью, позволяет получить гибкую конфигурацию системы в условиях эволюционно изменяющихся требований. Модель позволяет оценить его архитектуру до непосредственного построения и обеспечить динамическое конфигурирование рабочих мест пользователей, что является ее преимуществом. Также предложена автоматная модель проверки выполнения ограничений к формируемому ПО. Модель используется для проектирования исполнителя ВП, моделируя сценарии взаимодействия программных модулей и сравнивая результаты моделирования с нефункциональными требованиями к ПО.

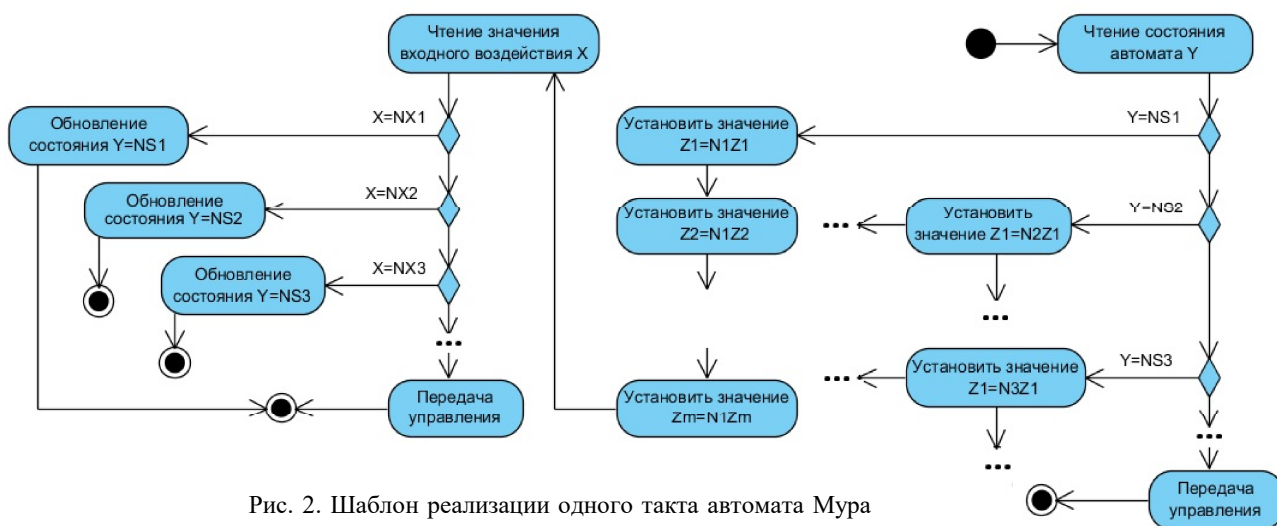


Рис. 2. Шаблон реализации одного такта автомата Мура

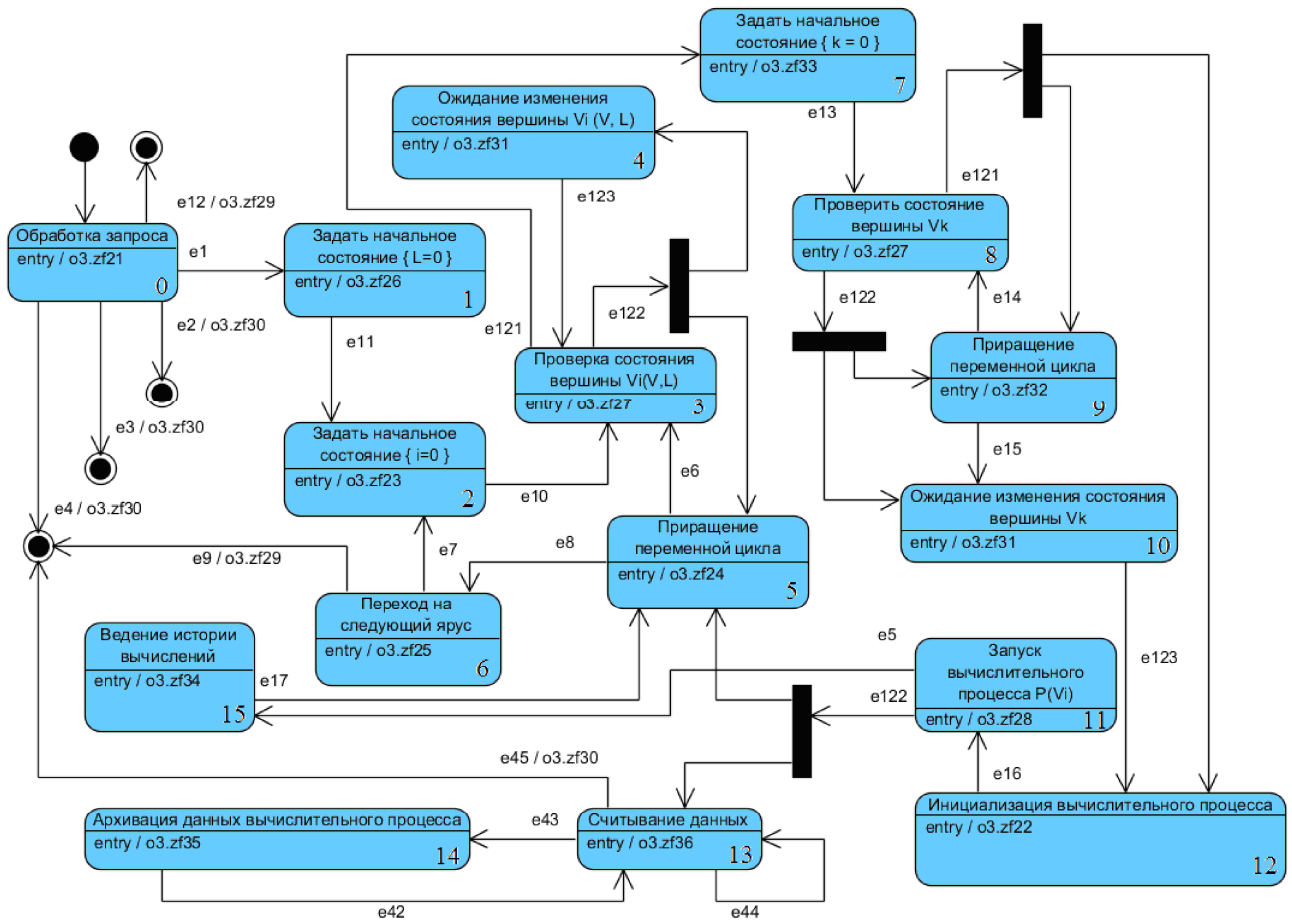


Рис. 3. Диаграмма переходов для вложенного конечного автомата AForwardSM

Таблица 1

Переходы конечного автомата

Обозначение	Наименование
e1/ e2/ e3	Получено разрешение/ Отмена действия/ Действие выполнено
e4/ e5	Процедура выполняется /Ошибка выполнения
e6	Номер вершины меньше максимального для текущего яруса
e7	Номер текущего яруса меньше количества ярусов
e8	Вершина принадлежит множеству вершин текущего яруса
e9	Достигнут последний ярус ЯПФ подграфа задачи
e10/ e11	Возврат к исходному ВП/ Возврат к исходному ярусу
e12	Сформирован ответ на запрос пользователя
e13	Получено значение полустепени захода для текущей вершины
e14	Обработка входных данных текущей вершины
e15	Множество входящих дуг исчерпано для данной вершины
e16/ e43	Заданы условия и параметры ВП/ Данные считаны
e44/ e45	Время ожидания не превышено / истекло
e121/e122/e123	ВП не запущен / выполняется / выполнен

Таблица 2

Состояния конечного автомата

Обозначение	Наименование
o3.zf21	Инициализация подграфа вычислительного процесса
o3.zf22	Инициализация ВП для автомата AForwardSM
o3.zf23	Инициализация переменных цикла
o3.zf24/ o3.zf25	Переход к следующему ВП / ярусу ЯПФ подграфа задачи
o3.zf26	Возвращение на исходный ярус ЯПФ подграфа задачи
o3.zf27/o3.zf28/o3.zf31	Проверка состояния ВП/ запуск ВП/ ожидание ВП
o3.zf32 /o3.zf33	Переход к следующему / исходному ВП
o3.zf34	Сохранение log-записи
o3.zf35/ o3.zf36	Архивация данных/получение данных

Литература: 1. *Генератор проектов* – средство автоматизации проектирования прикладных информационно-вычислительных систем / Ю.А. Флёрв, Л.Л. Вышинский, И.Л. Гринёв, А.А. Логинов, и др. // Автоматизация проектирования инженерных и финансовых информационных систем средствами «Генератора проектов». М.: ВЦ РАН. 2010. С. 3-15. **2.** *Инструментальные средства САПР* / Л.Л. Вышинский, И.Л. Гринёв, В.И. Шиленко, Н.И. Широков / Задачи и методы автоматизированного проектирования в авиационной промышленности. М.: ВЦ АН СССР. 1991. С.52-70. **3.** *Перевозчикова О.Л.* Диалоговые системы / О.Л. Перевозчикова, Е.Л. Ющенко. Ин-т кибернетики им. В.М. Глушкова. К.: Наук. думка, 1990. 184 с. **4.** *Jiannong C.* GOP: A graph-oriented programming model for parallel and distributed systems / Cao Jiannong, Alvin T.S. Chan, Yudong Sun // Parallel and distributed computing. 2005. P.21-36. **5.** *Дорошенко Е.А.* Формализованное проектирование эффективных многопоточных программ / Е.А. Дорошенко, К.А. Жереб, Е.А. Яценко // Проблемы программирования. 2007. № 1. С. 17–30. **6.** *Алгеброалгоритмические модели и методы параллельного программирования* / Ф.И. Андон, Е.А. Дорошенко, Г.Е. Цейтлин., Е.А. Яценко. Киев: Академперіодика, 2007. 631 с. **7.** *Шкулипа И. Ю.* Автоматизированный синтез программ на основе САА-М-схем / И. Ю. Шкулипа, С. Д. Погорельный // УСиМ. 2010. №4. С. 58-63. **8.** *Корухова Ю. С.* Автоматический синтез программ с использованием онтологии прецедентов / Ю. С. Корухова, Н. Н. Фастовец // Программные системы и инструменты: тематический сборник. 2011. Т. 12. С.203-215. **9.** *Автоматический синтез системы управления мобильным роботом для решения задачи «Кегельринг»* / С.А. Алексеев, А.И. Калинин, В.О. Клебан, А.А. Шалыто // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. 2011. № 2 (72). С. 26-31. **10.** *Канжелев С. Ю.* Автоматическая генерация кода программ с явным выделением состояний / С.Ю. Канжелев // Мат. конф. «Software Engineering Conference (Russia) – 2006» (SEC (R)). 2006. С. 60–63. **11.** *Салмре И.* Программирование мобильных устройств на платформе .NET Compact Framework / Иво Салмре, пер. с англ. М.: Изд. дом «Вильямс». 2006. 736 с. **12.** *Лаврищева Е.М.* Сборочное программирование. Основы индустрии программных продуктов / Е.М. Лаврищева, В.Н. Грищенко. Киев: Наук. думка, 2009. 372с. **13.** *Лаврищева Е.М.* Software Engineering компьютерных систем. Парадигмы, технологии и CASE-средства программирования / Е. М. Лаврищева. К.: Наук. думка. 2013. 283 с. **14.** *Интеллектуальные технологии распределенных вычислений для моделирования сложных систем* / Марьин С. В., Ларченко А. В., Ковальчук С. В., Князьков К. В., и др. // Науч.-техн. вестн. СПбГУ ИТМО. 2010. Вып. № 70. С. 123–124. **15.** *Сергеев Ю.* Управление доступом к виртуальной инфраструктуре с помощью продукта HyTrust / Ю. Сергеев // Jet Info Информационный бюллетень. 2012. №3 (224). 44 с. **16.** *Малышкин В. Э.* Параллельное программирование мультимикросистем / В.Э. Малышкин, В.Д. Корнеев. Новосибирск: Новосибирский государственный технический университет. 2006. 452 с. **17.** *Калянов Г. Н.* CASE-технологии: консалтинг в автоматизации бизнес-процессов / Г. Н. Калянов. М.: Горячая линия-Телеком. 2002. 320 с. **18.** *Поликарпова Н. И.* Автоматное программирование / Н. И. Поликарпова, А. А. Шалыто. СПб.: «Питер». 2008. 167 с.

Transliterated bibliography:

1. *Generator projektov – sredstvo avtomatizatsii projektirovaniya prikladnykh informatsionno-vychislitelnykh sistem* / Yu.A. Fl'orov, L.L. Vyishinskiy, I.L. Grin'ov, A.A. Loginov, i dr. // Avtomatizatsiya projektirovaniya inzhenernykh i finansovykh informatsionnykh sistem sredstvami «Generators projektov». М.: VTs RAN. 2010. S. 3-15. **2.** *Instrumentalnyiye sredstva SAPR* / L.L. Vyishinskiy, I.L. Grin'ov, V.I. Shilenko, N.I. Shirokov // Zadachi i metody avtomatizirovannogo projektirovaniya v aviastroenii. М.: VTs AN SSSR. 1991. S.52-70.

3. *Pervezchikova O.L.* Dialogovyye sistemy / O.L. Pervezchikova, E.L. Yuschenko. In-t kibernetiki im. V.M. Glushkova. K.: Nauk. dumka, 1990. 184 s. **4.** *Jiannong C.* GOP: A graph-oriented programming model for parallel and distributed systems / Cao Jiannong, Alvin T.S. Chan, Yudong Sun // Parallel and distributed computing. 2005. P.21-36. **5.** *Doroshenko E.A.* Formalizovannoe proektirovanie effektivnykh mnogopotchnykh programm / E.A. Doroshenko, K.A. Zhereb, E.A. Yatsenko // Problemy programmirovaniya. 2007. #1. S. 17–30. **6.** *Algebroalgoritmicheskie modeli i metodyi parallelnogo programmirovaniya* / F.I. Andon, E.A. Doroshenko, G.E Tseytlin., E.A. Yatsenko. Kiev: Akadempriodika, 2007. 631 s. **7.** *Shkulipa I. Yu.* Avtomatizirovannyiy sintez programm na osnove SAA-M-shem / I. Yu. Shkulipa, S. D. Pogorelyiy // USiM. 2010. #4. S. 58-63. **8.** *Koruhova Yu. S.* Avtomaticheskii sintez programm s ispolzovaniem ontologii pretsedentov / Yu. S. Koruhova, N. N. Fastovets // Programmiyie sistemy i instrumentyyi: tematicheskii sbornik. 2011. T. 12. S.203-215. **9.** *Avtomaticheskii sintez sistemyi upravleniya mobilnyim robotom dlya resheniya zadachi «Kegelring»* / S.A. Alekseev, A.I. Kalinichenko, V.O. Kleban, A.A. Shalyito // Nauchno-tehnicheskii vestnik Sankt-Peterburgskogo gosudarstvennogo universiteta informatsionnykh tehnologiy, mehaniki i optiki. 2011. # 2 (72). S. 26-31. **10.** *Kanzhelev S. Yu.* Avtomaticheskaya generatsiya koda programm s yavnyim vyideleniem sostoyaniy / S.Yu. Kanzhelev // Mat. konf. «Software Engineering Conference (Russia) – 2006» (SEC (R)). 2006. S. 60 – 63. **11.** *Salmre I.* Programmirovaniye mobilnykh ustroystv na platforme .NET Compact Framework / Ivo Salmre, per. s angl. М.: Izd. dom «Vilyams». 2006. 736 s. **12.** *Lavrisheva E.M.* Sborochnoe programmirovaniye. Osnovy industrii programnykh produktov / E.M. Lavrisheva, V.N. Grischenko. Kiev: Nauk. dumka, 2009. 372s. **13.** *Lavrisheva E.M.* Software Engineering kompyuternykh sistem. Paradigmy, tehnologii i CASE-sredstva programmirovaniya / E. M. Lavrisheva. K.: Nauk. dumka. 2013. 283 s. **14.** *Intellektualnyie tehnologii raspredelennykh vychisleniy dlya modelirovaniya slozhnykh sistem* / Marin S. V., Larchenko A. V., Kovalchuk S. V., Knyazkov K. V., i dr. // Nauch.- tehn. vestn. SPbGU ITMO. 2010. Vyip. # 70. S. 123–124. **15.** *Sergeev Yu.* Upravlenie dostupom k virtualnoy infrastrukture s pomoschyu produkta HyTrust / Yu. Sergeev // Jet Info Informatsionnyiy byulleten. 2012. #3 (224). 44 s. **16.** *Malyishkin V.E.* Parallelnoe programmirovaniye multikompyutero / V.E. Malyishkin, V.D. Korneev. Novosibirsk: Novosibirskiy gosudarstvennyiy tehicheskii universitet. 2006. 452 s. **17.** *Kalyanov G. N.* CASE-tehnologii: konsalting v avtomatizatsii biznes-protsessov / G. N. Kalyanov. М.: Goryachaya liniya-Telekom. 2002. 320 s. **18.** *Polikarpova N. I.* Avtomatnoe programmirovaniye / N. I. Polikarpova, A. A. Shalyito. SPb.: «Piter». 2008. 167 s.

Поступила в редколлегию 10.05.2016

Рецензент: д-р техн. наук, проф. Гребенник И.В.

Чайников Сергей Иванович, канд. техн. наук, профессор кафедры системотехники, Харьковский национальный университет радиоэлектроники. Адрес: Украина, 61166, Харьков, пр. Науки, 14, E-mail: chajnikov@kture.kharkov.ua

Солодовников Андрей Сергеевич, ассистент кафедры медицинской и биологической физики и медицинской информатики, Харьковский национальный медицинский университет. Адрес: Украина, 61166, Харьков, пр. Науки, 14, E-mail: andrey.sldv@rambler.ru

ИССЛЕДОВАНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ МЕТОДОВ УПРАВЛЕНИЯ СИСТЕМАМИ ТРАНСПОРТНОГО ТИПА

ГВОЗДИНСКИЙ А.Н., ЗАКУТНИЙ С.В.

Исследуются алгоритмы, с помощью которых возможно решить задачи составления оптимального количества задействованного транспорта и перечня выполнения рейсов самолетов для минимизации времени выполнения полного цикла полетов.

1. Введение

Состояние проблемы. Под названием “транспортная задача” объединяется широкий круг задач с единой математической моделью. Они относятся к задачам оптимизации и могут быть решены различными способами. Однако матрица системы ограничений транспортной задачи настолько своеобразна, что для ее решения разработаны специальные методы. Эти методы позволяют найти начальное опорное решение, а затем, улучшая его, получить оптимальное решение.

Транспортная задача заключается в том, что необходимо найти оптимальный план перевозок груза с баз потребителям.

Существует два типа транспортных задач: по критерию стоимости (план перевозок оптимален, если достигнут минимум затрат на его реализацию) и по критерию времени (план оптимален, если на его реализацию затрачивается минимум времени).

Многие важные модели линейного программирования часто содержат тысячи операций (переменных) и сотни ограничений, в связи с чем применение эффективных алгоритмов становится не только выгодным, но и просто необходимым. Например, составление сложного комплекса технологических работ, выполнение крупных деталей оборудования, системы связи.

Транспортная задача – это один из первых примеров оптимизации на линейных сетях. В настоящее время эта задача стала типовой для промышленных фирм, имеющих несколько предприятий, складов, рынков сбыта и оптовых баз. Модель применяется главным образом при решении плановых задач. В этом случае стратегические решения сводятся к выбору транспортных маршрутов, по которым продукция различных предприятий доставляется на несколько складов или в различные конечные пункты назначения.

Актуальность рассмотренной задачи состоит в том, что в наше время составление расписания очень важно с точки зрения его правильности и количества задействованного транспорта (ресурсов) при его выполнении. Очень актуальным является вопрос о составлении оптимального плана (расписания) с точки зрения минимизации времени простоя (отдыха) ресурсов и использования оптимального их количества.

Сущность исследования заключается в поиске наиболее эффективных математических, алгоритмических и программных методов для решения задач транспортного типа со спецификой рассматриваемой задачи [1].

Цель исследования заключается в минимизации времени выполнения циклов рейса и оптимальном использовании транспортных ресурсов. Для достижения цели решаются такие задачи: анализ алгоритмов, которые бы помогли решить поставленную задачу; выбор адаптированного алгоритма; составление подсистемы, которая на основе адаптированного алгоритма, выводит перечень рейсов для выполнения полета за минимально оптимальное время; определение количества требуемых ресурсов для выполнения составленного расписания.

2. Постановка задачи

Авиакомпания хочет организовать полеты «туда» и «обратно» так, чтобы минимизировать время простоя. Каждый самолет, который выполняет рейс на авиалинии, перевозит пассажиров и груз. Во время рейса используется некоторое количество ресурсов (горючее, смазочные материалы, затраты на обслуживание), которые составляют в стоимостном выражении себестоимость рейса. Необходимо исследовать распределение самолетов по авиалиниям, при котором выполняются запланированные показатели перевозок при минимальной общей их стоимости [2].

3. Построение математической модели

Транспортная задача линейного программирования в настоящее время широко распространена в теоретических обработках и практическом применении на транспорте и в промышленности. Особенно важное значение она имеет в деле рационализации поставок важнейших видов промышленной и сельскохозяйственной продукции, а также оптимального планирования грузопотоков и работы различных видов транспорта (автобусов, самолетов, маршрутных такси). Введем следующие обозначения: n – количество типов самолетов, $n = 5$; m – количество авиалиний, $m = 3$.

Индекс будем использовать для указания авиалиний и для указания типов самолетов: вместимость самолета (чел.); грузоподъемность самолета (t); план перевозки пассажиров за месяц на i -й авиалинии (тыс. чел./мес.); план перевозку грузов за месяц на i -й авиалинии (t /мес.); максимальное количество рейсов j -го типа самолета на i -й авиалинии за один месяц; имеющееся в наличии у авиакомпании количество самолетов; коэффициент исправности.

Целевая функция, задающая критерий минимизации, имеет вид:

$$F(x) = \sum_{i=1}^m \sum_{j=1}^n x_{ij} c_{ij} \rightarrow \max, \quad (1)$$

где $F(x)$ задает общую стоимость перевозок за один месяц.

Определим ограничения, присутствующие в условии задачи: $s_{ij} = q_{ij}u_i, \forall ij$ – объем перевозок пассажиров (тыс. чел./мес.); $o_{ij} = q_{ij}v_i, \forall ij$ – объем перевозок грузов (т/мес.)

Хотя самолет перевозит и пассажиров и груз одновременно, ограничения по выполнению перевозок пассажиров и груза можно рассматривать отдельно, так как вместительность и грузоподъемность самолета независимы друг от друга:

$$\left\{ \begin{array}{l} \sum_{j=1}^n x_{ij}s_{ij} \geq a_i \\ \sum_{i=1}^m x_{ij}o_{ij} \geq b_i \end{array} \right\} i = \overline{1, m} \quad (2)$$

или

$$\left\{ \begin{array}{l} \sum_{j=1}^n x_{ij}u_{ij}q_{ij} \geq a_i \\ \sum_{i=1}^m x_{ij}v_{ij}q_{ij} \geq b_i \end{array} \right\} i = \overline{1, m} \quad (3)$$

Также учитываются ограничения на количество самолетов каждого типа, которые имеются у авиакомпании, и ограничения неотрицательных искомым переменных. Учитывая тот факт, что некоторые самолеты могут иметь неисправности, эти ограничения имеют вид:

$$\left\{ \begin{array}{l} \sum_{i=1}^m x_{ij} \leq p_{ij}k_{ij} \quad j = \overline{1, n} \\ x_{ij} \geq 0 \quad \forall ij \end{array} \right\} i = \overline{1, m} \quad (4)$$

Заметим, что суммарный возможный объем перевозок всеми самолетами превышает суммарный план и, следовательно, требование целочисленности можно опустить. При этом дробные значения x_{ij} будут означать частичное использование j -го типа самолета на i -й авиалинии.

Полная математическая модель поставленной задачи выражается в виде [3]:

$$F(x) = \sum_{i=1}^m \sum_{j=1}^n x_{ij}c_{ij} \rightarrow \max \quad (5)$$

при условиях

$$\left\{ \begin{array}{l} \sum_{j=1}^n x_{ij}u_{ij}q_{ij} \geq a_i \\ \sum_{i=1}^m x_{ij}v_{ij}q_{ij} \geq b_i \\ \sum_{i=1}^m x_{ij} \leq p_{ij}k_{ij}, \quad j = \overline{1, n} \\ x_{ij} \geq 0, \quad \forall ij \end{array} \right\} i = \overline{1, m} \quad (6)$$

4. Исследование и разработка методов решения

Для составления плана придуман не один алгоритм, но так как транспортные задачи такого вида являются глубоко вырожденными, т.е. один рейс (работа) должен выполняться только одним исполнителем, то не всякий алгоритм это сможет решить.

Симплекс-метод хорошо справляется с t -задачами, но так как эта задача транспортного типа особого назначения, т.е. на каждый заказ назначается только один исполнитель, то она глубоко вырождена и симплекс-метод не применяем.

Венгерский метод основан на некоторых довольно трудных и нетривиальных комбинаторных свойствах матрицы. Этот метод дает хорошие результаты, но его довольно трудно программировать. Преобразования матриц происходят таким образом, чтобы найти в них нули, которые были бы единственными в соответствующем столбце, они и будут искомым планом.

Метод Мака имеет преимущество более простого интуитивного обоснования, ведь это – логический процесс. Этот метод основан на идее выбора в каждой строке минимального элемента. Вообще говоря, минимальные элементы строк не распределены по всем столбцам матрицы. Здесь используется идея сложения (или вычитания) одного и того же значения по всем элементам строк по столбцам. Если в каждом столбце имеется подчеркнутый элемент, работа алгоритма закончена. По элементам, которые соответствуют оптимальному выбору, может быть вычислена соответствующая стоимость.

При написании программы для решения транспортной задачи использовался метод потенциалов. Идея его состоит в следующем. Для любой свободной клетки транспортной таблицы существует единственный цикл, положительная вершина которого лежит в этой свободной клетке, а все остальные – в базисных. Если цена такого цикла отрицательна, то план можно улучшить перемещением перевозок по данному циклу. Количество единиц груза, которое можно переместить, определяется минимальным значением перевозок, стоящих в отрицательных вершинах цикла (если переместить большее число единиц груза, возникнут отрицательные перевозки). Если циклов с отрицательной ценой нет, то это означает, что дальнейшее улучшение плана невозможно, т.е. оптимальный план найден.

Транспортная задача характеризуется широтой применения, а также ее универсальностью (к данному типу задач могут быть сведены другие задачи линейного программирования). Для постановки транспортной задачи необходимо знать запасы A_i каждого i -го поставщика (количество поставщиков равно m), потребности B_j j -го получателя (количество получателей равно n), затраты на перевозку продукции (C_{ij}) от i -го поставщика к j -му получателю. Предполагается, что транспортные расходы пропорциональны перевозимому количеству продукции, т.е. перевозка X единиц продукции вызывает расходы $X \cdot C_{ij}$. Транспортная задача является задачей определения плана

перевозок $(X) = X_{ij}$, где X_{ij} – количество единиц продукции, поставляемой по коммуникации i, j :

$$u_1 u_2 \dots u_m, v_1 v_2 \dots v_n. \quad (7)$$

Целевой функцией можно считать суммарную стоимость всех перевозок. Результатом решения транспортной задачи является оптимальный план перевозок продукции от поставщиков к потребителям, при котором затраты будут минимальными:

$$u_i + v_j \leq c_{ij}, j = \overline{1, n}, i = \overline{1, m}. \quad (8)$$

Был разработан программный продукт, куда введены исходные данные: вместимость и грузоподъемность самолетов, количество самолетов разных типов, максимальное количество рейсов, стоимость эксплуатации одного самолета на авиалиниях:

$$T(u, v) = \sum_{i=1}^m a_i u_i \sum_{j=1}^n b_j v_j \rightarrow \max. \quad (9)$$

Исходя из вычисленных оценок, в которых содержатся положительные оценки, делаем вывод о том, что данный опорный план не является оптимальным [4]. Номер ведущего столбца вычисляется из соотношения, так как рассматривается задача на минимизацию.

Дальнейшее решение предполагает использование итераций симплекс-метода для нахождения оптимального плана перевозок:

$$\begin{aligned} &(30X_1 + 50X_2 + 28X_3) + (45X_4 + 55X_5 + 58X_6) + (55X_7 + 28X_8 + 43X_9) + (45X_{10} + 63X_{11} + 80X_{12}) + (65X_{13} + 76X_{14} + 39X_{15}) \Rightarrow \min; \\ &576X_1 + 352X_4 + 656X_7 + 600X_{10} + 1200X_{13} \geq 3500; \\ &480X_2 + 352X_5 + 820X_8 + 600X_{11} + 600X_{14} \geq 3000; \\ &576X_3 + 440X_6 + 820X_9 + 400X_{12} + 900X_{15} \geq 600; \\ &12X_1 + 8X_4 + 24X_7 + 30X_{10} + 32X_{13} \geq 80; \\ &10X_2 + 8X_5 + 30X_8 + 30X_{11} + 16X_{14} \geq 70; \\ &14X_3 + 10X_6 + 30X_9 + 20X_{12} + 24X_{15} \geq 65; \\ &X_1 + X_2 + X_3 \leq 9; \\ &X_4 + X_5 + X_6 \leq 9; \\ &X_7 + X_8 + X_9 \leq 8; \\ &X_{10} + X_{11} + X_{12} \leq 10,008; \\ &X_{13} + X_{14} + X_{15} \leq 4,002; \\ &X_i \geq 0; \end{aligned}$$

Для достижения оптимального плана потребовалось 12 итераций по улучшению опорного плана.

Как видно из решения, оптимальный план будет таким:

$$X = (0,666667; 8,333334; 0; 9; 0; 0; 0; 0; 8; 0; 0; 0; 0; 0; 0).$$

Минимальное значение функции примет значение:

$$F(x^*) = 2323,67 \text{ млн. грн. / мес.}$$

Дробные значения искоемых переменных означают неполную загрузку самолетов (например, в конце

месяца). Следовательно, необходимо округлить полученные значения в большую сторону:

$$X = (1; 8; 0; 9; 0; 0; 0; 0; 8; 0; 0; 0; 0; 0; 0).$$

Увеличившиеся затраты эксплуатации примут значение:

$$F(x^*) = 2324 \text{ млн. грн. / мес.}$$

В полученном результате не используются все типы самолетов, а только наиболее выгодные для каждой линии.

5. Эволюционные методы оптимизации

Наиболее актуальными среди эволюционных методов являются генетические алгоритмы (ГА). Они есть поисковые алгоритмы, основанные на механизмах натуральной селекции и натуральной генетики. Реализуют «выживание сильнейших» среди рассмотренных структур, формируя и изменяя поисковый алгоритм на основе моделирования эволюции поиска. В каждой генерации новое множество искусственных последовательностей создается, используя части старых и добавляя новые части с «хорошими свойствами». ГА – это не просто случайный поиск. Он эффективно использует информацию, накопленную в процессе эволюции.

Цель ГА двояка: абстрактно и формально объяснить адаптацию процессов в естественных системах; спроектировать искусственные программные системы, которые содержат механизмы естественных систем. Центральная система поиска в ГА – поиск баланса между эффективностью и качеством для выживания в различных условиях. ГА отличаются от других оптимизационных и поисковых процедур следующим: работают в основном не с параметрами, а с закодированным множеством параметров; осуществляют поиск из популяции точек, а не из единственной точки; используют целевую функцию, а не ее различные приращения для оценки информации; используют не детерминированные, а вероятностные правила.

ГА берет множество натуральных параметров оптимизационной проблемы и кодирует их как последовательность конечной длины в некотором конечном алфавите. В естественных системах общая генетическая упаковка называется генотип. В натуральных системах организм формируется посредством связи генетической упаковки с окружающей средой и называется фенотип. В естественной терминологии хромосомы состоят из генов, которые могут иметь числовые значения, называемые «аллели».

Начальное условие задачи – распределение транспортных средств. Необходимо закодировать это условие в хромосому. Для этого план перевозок нужно представить в виде развернутой строки. Генами в данном случае будут выступать элементы ячеек плана. При генерации популяции будем использовать случайные значения генов. Размер популяции для матрицы стоимостей размерности m на n элементов необходимо

брать в районе $m * n$. Количество генов в хромосоме будет равняться $m * n$.

Фитнесс-функцией (целевой функцией) для популяции является суммарная стоимость перевозки всех грузов:

$$f(k) = \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} . \quad (10)$$

Для каждой сгенерированной особи считаем значение целевой функции, после чего вычисляем отношение ее приспособленности к суммарной приспособленности популяции:

$$Ps(i) = \frac{f(k)}{\sum_{i=1}^k f(k)} . \quad (11)$$

Полученное значение определяет вероятность выбора особи для дальнейших операций – кроссинговера и мутации. К хромосомам с наилучшей фитнес-функцией ($f(k) \rightarrow \min$) применяется оператор кроссинговера, который заключается в попарном «скрещивании» выбранных на предыдущем этапе особей. Например, имея хромосомы $A = 10,1,1,1,2,8,7$ и $B = 1,2,3,4,5$, после применения к ним оператора кроссинговера мы получим хромосомы $C = 10,1,1,2,4,5$ и $D = 1,2,3,8,7$. После скрещивания применяем к отобраным хромосомам оператор мутации. Он заключается в случайном изменении одной или нескольких позиций в хромосоме. Например, хромосома $A = 10,1,1,2,8,7$ после мутации может принять вид $A = 10,8,1,2,1,7$.

Оператор кроссинговера применяется с вероятностью 60%, мутации – 40%. Как ни странно, именно вероятностный подход позволяет генетическим алгоритмам достигать оптимума в решаемой задаче.

Применив разработанный алгоритм к задаче транспортного типа, получили следующий результат: на каждой итерации особи в популяции улучшали свою целевую функцию, стремясь к оптимальному решению, и на четырнадцатой итерации было получено решение задачи, в то время как при использовании метода потенциалов оптимальное решение было найдено только на двадцать второй.

Рассмотренные вычислительные алгоритмы были исследованы и апробированы для различных типов транспортных задач [4,5].

6. Выводы

В результате выполнения работы решена поставленная задача распределения самолетов по авиалиниям. В качестве метода решения был выбран симплекс-метод. Рассмотрены понятия эволюционного подхода к решению задач и, в частности, разработан алгоритм решения оптимизационной задачи транспортного типа на основе генетических алгоритмов. В ходе исследования сделаны выводы о том, что несмотря на сложность программной реализации, новый алгоритм является более производительным и позволяет находить оптимальное решение за более короткий срок.

Научная новизна. В ходе исследования впервые применен эволюционный подход к рассматриваемому классу задач. Таким образом, создан первый алгоритм решения оптимизационных задач транспортного типа на основе генетических алгоритмов.

Практическим значением полученных результатов является то, что разработанным методом можно решать транспортные задачи на любом предприятии транспортного типа. В отличие от уже существующих методов решения новый алгоритм является более производительным и не требует закупки более дорогостоящей вычислительной техники.

Литература: 1. Гвоздинский А.Н. Разработка информационной подсистемы управления воздушным транспортом / А.Н. Гвоздинский, Е.А. Гольцев // Радиоэлектроника и информатика. 2010. №2. С. 47-51. 2. Гвоздинський А.М. Методи оптимізації в системах прийняття рішень: навчальний посібник / А.М. Гвоздинський, Н.А. Якімова, В.О. Губін // Харків : ХНУРЕ, 2006 С.325. 3. Гвоздинский А.Н. Эволюционный подход к решению оптимизационных задач транспортного типа // А.Н. Гвоздинский, С.В. Мельник // АСУ и приборы автоматки. 2009. №147. С. 76-80. 4. Гвоздинский А.Н. Об одном подходе к решению задач составления расписания в системах управления объектами транспортного типа. / А.Н. Гвоздинский, А.А. Куликова // АСУ и приборы автоматки. 2010. №150. С. 101-106. 5. Гвоздинский А.Н. Исследование и разработка методов решения задач закрепления потребителей за поставщиками с учетом возврата транспортных средств. / А.Н. Гвоздинский, М.А. Гаврюшенко // АСУ и приборы автоматки. 2010. №150. С.106-111.

Transliterated bibliography:

1. Gvozdinskiy A.N. Razrabotka informatsionnoy podsistemy upravleniya vozduzhnyim transportom/ A.N. Gvozdinskiy, E.A. Goltsev // Radioelektronika i informatika. 2010. #2. S. 47-51.
2. Hvozdyns'kyy A.M. Metody optymizatsiyi v systemakh pryunyattya rishen': navchal'nyy posibnyk / A.M. Hvozdyns'kyy, N.A. Yakimova, V.O. Hubin // Kharkiv : KhNURE, 2006 S.325.
3. Gvozdinskiy A.N. Evolyutsionniy podhod k resheniyu optimizatsionnykh zadach transportnogo tipa // A.N. Gvozdinskiy, S.V. Melnik // ASU i priboryi avtomatiki. 2009. #147. S. 76-80.
4. Gvozdinskiy A.N. Ob odnom podhode k resheniyu zadach sostavleniya raspisaniya v sistemah upravleniya ob'ektami transportnogo tipa. / A.N. Gvozdinskiy, A.A. Kulikova // ASU i priboryi avtomatiki. 2010. #150. S. 101-106.
5. Gvozdinskiy A.N. Issledovanie i razrabotka metodov resheniya zadach zakrepleniya potrebiteley za postavschikami s uchetom vozvrata transportnykh sredstv. / A.N. Gvozdinskiy, M.A. Gavryushenko // ASU i priboryi avtomatiki. 2010. #150. S.106-111.

Поступила в редколлегию 23.06.2016

Рецензент: д-р техн. наук, проф. Путятин Е.П.

Гвоздинский Анатолий Николаевич, канд. техн. наук, профессор кафедры искусственного интеллекта ХНУРЭ. Научные интересы: оптимизация процедур принятия решений в сложных системах управления. Адрес: Украина, 61166, Харьков, ул. акад. Ляпунова, 7, кв. 9, тел. 32-69-08.

Закутний Сергей Валерьевич, студент группы СА-14-1 ХНУРЭ. Научные интересы: программирование, математическая статистика. Адрес: Украина, 61202, Харьков, ул. Целиноградская, 58.

КЛАСТЕРИЗАЦИЯ БЛОКОВ АЭРОФОТОСНИМКА В ДВУХПРИЗНАКОВОМ СТРУКТУРНОМ ПРОСТРАНСТВЕ НА ОСНОВЕ МЕТОДА К-СРЕДНИХ В СИСТЕМЕ ОБРАБОТКИ ИНФОРМАЦИИ

*БАРАННИК В.В., МУСИЕНКО А.П.,
ЖУЙКОВ Д.Б., БРАЙЛОВСКИЙ Н.Н.*

Обосновывается необходимость обеспечения своевременной доставки цифровых снимков с использованием беспилотных бортовых средств аэромониторинга. Показывается направление для повышения эффективности процесса кластеризации блоков аэрофотоснимка на основе метода К-средних в двухпризнаковом структурном пространстве в системе обработки информации. Описывается возможность использования алгоритма К-средних для определения степени семантической насыщенности распределенных блоков аэрофотоснимка в кластерах по признаковым характеристикам. Такой подход позволяет сравнить результаты кластеризации блоков аэрофотоснимка, которые получены автоматическим способом дешифрирования (без участия оператора), с результатами, полученными дешифровщиком, основываясь на визуальном анализе.

Ключевые слова: кластеризация, метрика, блок, аэрофотоснимок, обработка, технология.

1. Введение

Современное развитие беспилотных летательных аппаратов свидетельствует о повышении интереса к получению своевременной и достоверной информации в системе аэромониторинга [1-5]. Здесь ключевым моментом является использование цифровых аэрофото-снимков, регистрируемых в процессе полета беспилотным летательным аппаратом (БПЛА). Однако возникает ряд вопросов, связанных с необходимостью одновременного решения задач относительно обработки, передачи, а также доставки цифровых аэрофотоснимков в реальном времени в центр обработки информации [6].

Один из подходов к решению таких задач заключается в использовании методов обработки цифровых изображений с учетом выделения структурно-значимой информации о характеристиках объектов. В случае обработки цифровых аэрофотоснимков на борту БПЛА необходимо предварительно выделить на снимке семантически насыщенные блоки, которые в дальнейшем будут передаваться с сохранением наибольшей информативности, с заданным качеством, а также с заданной пропускной способностью бортовых каналов связи. Поэтому актуальной задачей является необходимость обеспечения своевременной доставки цифровых аэро-фотоснимков с использованием беспилотных бортовых средств аэромониторинга,

с сохранением наибольшей информативности блоков аэрофотоснимка.

Анализ публикаций по цифровой обработке изображений показал, что наиболее семантически значимой является информация о контурах, границах объектов [5-8, 9]. При этом не учитывается семантическая структурно-значимая составляющая обрабатываемых изображений. Это позволяет снизить время обработки на этапе кодирования, но при этом вносятся значительные искажения в исходную семантическую структуру [10], что приводит к неправильному дешифрированию снимков и усложнению выделения значимых фрагментов.

Решением такого противоречия является применение технологии кластерного анализа с использованием методов обрабатывающих множество исходных данных – блоков аэрофотоснимка. Одной из реализаций такого подхода является способ кластеризации блоков аэрофотоснимка – К-средних. Среди многих методов кластерного анализа [11 - 13] выбор алгоритма К-средних связан с простотой реализации, высокой скоростью обработки исходных данных (в нашем случае множество блоков аэрофотоснимка). Из особенностей данного способа можно выделить то, что процесс кластеризации блоков аэрофотоснимка может происходить не по одной характеристике (показателю) блока, а по многим признаковым характеристикам. Однако ввиду ограничений, связанных с бортовой мощностью аппаратуры обработки и передачи данных БПЛА, в нашем случае предлагается использовать два показателя. Очевидно, что применение двух признаковых показателей в процессе кластеризации блоков аэрофотоснимка даст необходимый результат.

Алгоритм К-средних позволит более эффективно обрабатывать большую совокупность аэрофотоснимков за счет предварительного распределения блоков аэрофотоснимка в признаковом пространстве. Поэтому цель исследований заключается в создании способа кластеризации блоков аэрофотоснимка в двухпризнаковом структурном пространстве на основе метода К-средних в системе обработки информации.

2. Основное содержание исследований

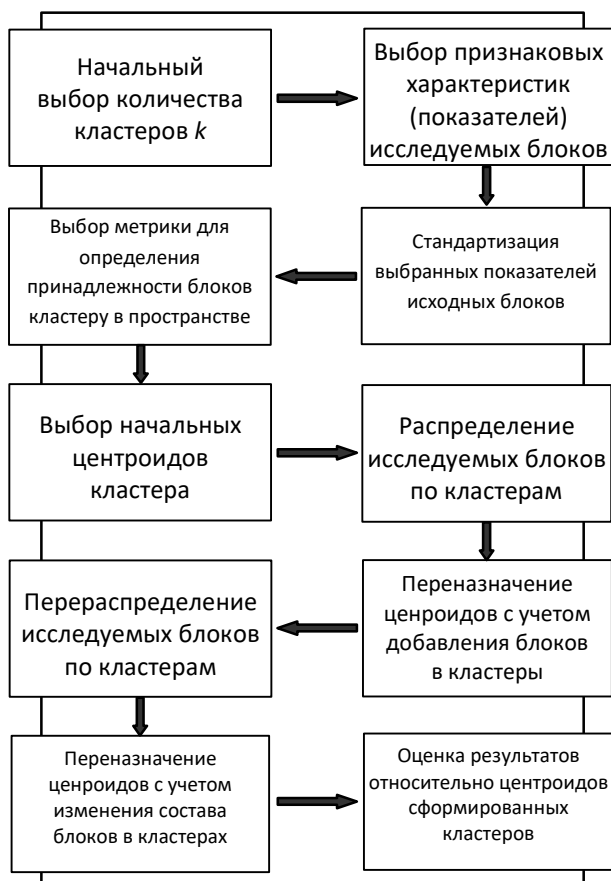
Поскольку нам неизвестно, к каким классам семантической насыщенности относятся исследуемые блоки аэрофотоснимка, то применение способа кластеризации на основе алгоритма К-средних позволит классифицировать характерные блоки аэрофотоснимков по степени семантической насыщенности. В конечном итоге, результативность процесса дешифрирования аэрофото-снимков будет зависеть от использования наиболее эффективного метода кластеризации [11, 14].

Проведем кластеризацию блоков аэрофотоснимка методом К-средних, для того чтобы:

– во-первых, получить распределение блоков аэрофотоснимков по кластерам;

– во-вторых, результаты кластеризации блоков аэрофотоснимка, которые получены автоматическим способом дешифрирования (без участия оператора), сравнить с результатами, полученные дешифровщиком, который основывается на визуальном анализе.

Этапы метода кластеризации блоков аэрофотоснимка, на основе алгоритма К-средних, представлены на рисунке.



Этапы метода кластеризации блоков аэрофотоснимка

Первым этапом предлагается выбрать количество кластеров $k=3$. Это объясняется тем, что в рамках исследования рассматриваются блоки $b(p_{k,\ell})$ аэрофотоснимка разных типов (контурные, текстурные, однородные), в которых учитывается степень насыщенности, а именно: слабонасыщенные, средненасыщенные и сильнонасыщенные [13].

На этапе выбора признаков (показателей) исследуемых блоков необходимо определить те показатели, которые будут наиболее точно характеризовать исследуемые блоки. В нашем случае для проведения эксперимента были выбраны показатель насыщенности $p_{k,\ell}^{(1)}$ и структурный показатель $p_{k,\ell}^{(2)}$. Ввиду того, что данные показатели разные по величине и по способу получения, необходимо произвести нормировку исходных значений показателей. Применение нормирования приведет к тому, что данные показатели станут равнозначными с точки зрения определения меры принадлежности исследуемых бло-

ков $b(p_{k,\ell})$ аэрофотоснимка в кластере в двухпризнаковом структурном пространстве $P(2)_j$.

Третьим этапом, при построении кластеризации, необходимо выбрать метрику относительно принадлежности исследуемых блоков кластеру. В нашем случае на этом этапе процесс кластеризации заключается в определении минимального расстояния $d(2)_{\min}^{(\gamma)}$ между блоком $b(p'_{k,\ell})$ аэрофотоснимка и центроидом $C_j^{(\gamma)}$ (центром) кластера.

Центроиды кластеров необходимы для вычисления на каждом шаге процесса кластеризации меры близости элементов. При этом начальные центроиды кластеров выбираются с учетом нормированного показателя $p'_{k,\ell}$ начальных блоков $b(p'_{k,\ell})$ аэрофотоснимков. В этом случае значения нормированных показателей $p'_{k,\ell}$ блоков каждого кластера должны быть наиболее показательны, удалены друг от друга (отличаться в наибольшей степени).

В качестве метрики определения расстояния $d(2)_{\xi}^{(\gamma)}$ между блоком аэрофотоснимка $b(p'_{k,\ell})$ и начальным центроидом $C_j^{(\gamma)}$ j -го кластера по нормированному показателю $p'_{k,\ell}$ – используется мера близости $D(2)_{Ev}$ евклидово расстояние:

$$d(2)_{\xi}^{(\gamma)} = \sqrt{|p'_{k,\ell} - \bar{p}_{\xi}^{(\gamma)}|^2} = |p'_{k,\ell} - \bar{p}_{\xi}^{(\gamma)}|, \quad (1)$$

где $\bar{p}_{\xi}^{(\gamma)}$ – среднее значение по нормированному показателю среди всех элементов ξ -го кластера на γ -м шаге процесса кластеризации в двухпризнаковом структурном пространстве $P(2)_j$.

Далее по полученному расстоянию принимается решение, к какому кластеру отнести блок.

На четвертом этапе проводится переназначение центроида $C_j^{(\gamma+1)}$ для j -го кластера на $(\gamma+1)$ -м шаге процесса кластеризации.

В нашем случае центроид $C_j^{(\gamma+1)}$ на $(\gamma+1)$ -м шаге процесса кластеризации при добавлении i -го элемента к j -му кластеру переназначается с учетом своего текущего значения $\bar{p}_j^{(\gamma)}$ и нормированного показателя $p'_{i,j}$ элемента, являющегося наиболее близким к значению центроида $C_j^{(\gamma)}$. Происходит уточнение сред-

него значения $\overline{p}_j^{(\gamma+1)}$ на $(\gamma+1)$ -м шаге процесса кластеризации между текущим значением $\overline{p}_j^{(\gamma)}$ центроида и нормированным показателем $p'_{i,j}$ элемента $m_{i,j}^{(\gamma)}$, что представлено выражением:

$$\overline{p}_j^{(\gamma+1)} = \frac{\overline{p}_j^{(\gamma)} + p'_{i,j}}{2}. \quad (2)$$

Каждый последующий шаг, с учетом изменения состава кластера, предусматривает переназначение центроида. В результате этого, элемент $m_{i,j}^{(\gamma)}$, принадлежащий одному кластеру, может перераспределиться в другой кластер.

Обобщенный критерий эффективности всего процесса кластеризации позволяет оценить степень уменьшения ошибки относительно внутрикластерной близости центроида $C_j^{(\gamma)}$ и блоков $b(p'_{k,\ell})$ аэрофотоснимков в двухпризнаковом структурном пространстве $P(2)_j$ и может быть рассчитан как сумма квадратичных ошибок, что представлено выражением:

$$E(2)^{(\gamma)} = \left[\sum_{j=1}^k \sum_{i=1}^{n_j^{(\gamma)}} (p'_{i,j} - \overline{p}_j^{(\gamma)})^2 \right], \quad (3)$$

где k – число кластеров; $n_j^{(\gamma)}$ – количество элементов $m_{i,j}^{(\gamma)}$, которые принадлежат j -му кластеру; при этом $n_j^{(\gamma+1)} = n_j^{(\gamma)} + 1$.

Эффективность процесса кластеризации аэрофотоснимка определяется как минимизация функции $E(2)^{(\gamma)}$:

$$E(2)^{(\gamma)} \rightarrow \min.$$

Таким образом, чем ближе блоки $b(p'_{k,\ell})$ аэрофотоснимка относительно центроида сформированного кластера, тем более качественно проведен процесс кластеризации. При этом значение функционала $E(2)^{(\gamma)}$ будет стремиться к нулевой отметке.

3. Выводы

1. Усовершенствован метод кластеризации фрагментов аэрофотоснимка в двухпризнаковом структурном пространстве на основе метода К средних в системе обработки информации. Данная технология позволяет отобрать блоки аэрофотоснимка в двухпризнаковом структурном пространстве по кластерам, с учетом степени семантической насыщенности исследуемых

блоков. Особенностью усовершенствованного метода является то, что среди выделенных признаков характеристик для кластеризации блоков аэрофотоснимка использовались показатель степени насыщенности блока $p_{k,\ell}^{(1)}$ и структурный показатель блока $p_{k,\ell}^{(2)}$.

2. Проведенные практические расчеты показали, что эффективность процесса кластеризации достигается при использовании не менее двух признаков показателей, характеризующих блок, а процесс распределения блоков по кластерам завершен на второй итерации работы разработанного способа.

3. Преимущество усовершенствованного метода заключается в том, что при несложной вычислительной реализации он позволяет достаточно точно распределить блоки аэрофотоснимка по кластерам.

Усовершенствованный метод дает возможность:

- более эффективно обрабатывать множество аэрофотоснимков за счет предварительного распределения блоков аэрофотоснимка в двухпризнаковом структурном пространстве;

- отбирать кластеризованные блоки аэрофото-снимка по степени дешифровочной насыщенности. Это позволит дешифровать его с учетом выделенных значимых признаков (контуры, границы объектов, текстура и однородные участки местности).

Литература: 1. *Мосов С.* Аэрокосмическая разведка в современных военных конфликтах: монография / С.Мосов. К.: Изд. дом. "Румб", 2008. 248 с. 2. *Баранник В.В.* Метод повышения информационной безопасности в системах видеомониторинга кризисных ситуаций / В.В. Баранник, Ю.Н. Рябуха, О.С. // Монография. Черкассы, 2015. 143 с. 3. *Баранник В.В.* Метод селекции кадрового потока в системах критического аэромониторинга для повышения безопасности государственного информационного ресурса / В.В. Баранник, Ю.Н. Рябуха, С.С. Бульба // Авиационно-космическая техника и технологи. 2015. № 3. С. 111-118. 4. *Баранник В.В.* Концептуальный метод повышения безопасности дистанционного видеoinформационного ресурса в системе аэромониторинга кризисных ситуаций на основе интеллектуальной обработки видеокладов / В.В. Баранник, Ю.Н. Рябуха // Радиоэлектронные компьютерные системы. 2015. № 3. С. 19-21. 5. *Баранник В.В.* Методология совершенствования обработки видеoinформации, для повышения эффективности сервиса предоставления дистанционных видеослужб, при управлении в кризисных ситуациях / В.В. Баранник, Ю.Н. Рябуха, А.А. Красноруцкий, В.Ж. Яценко // АСУ и приборы автоматики. №170. 2015. С. 12-20. 6. *Баранник В.В.* Метод повышения доступности видеoinформации аэромониторинга / В.В. Баранник, О.С. Кулица // Радиоэлектронные и компьютерные системы. №3. 2013. С. 17-20. 7. *Гонсалес Р.С., Вудс Р.Э.* Цифровая обработка изображений / Р.С. Гонсалес, Р.Э. Вудс. М.: Техносфера, 2006. 1072 с. 8. *Кашкин В.Б.* Цифровая обработка аэрокосмических изображений. Красноярск: ИПК СФУ, 2008. 121 с. 9. *Красильников Н.Н.* Цифровая обработка изображений / Н. Н. Красильников. М.: Вузовская книга, 2011. 320. 10. *Баранник В.В.* Метод кластеризации фрагментов аэрофо-

тоснимков в спектрально-частотном пространстве // Баранник В.В., Мусиенко О.П., Яливец К.С. // Научно-технический журнал. 2016, №29(1). 2016. С. 23-30. **11. Blizorukov M. G.** Количественные методы анализа многомерных величин / Blizorukov M. G. Урало-Сибирский институт бизнеса. Издательство АМБ, 2006. 68 с. **12. Barannik V.** The Evaluation Method Of Coding Efficiency Of Basic Frames Of The Video Stream In Infocommunication / V.V. Barannik, A.P. Musienko // IEEE Second International Scientific-Practical Conference ["IEEE Problems of Infocommunications. Science and Technology, PICS&T'2015"], (Kharkiv, Ukraine, October 13-15, 2015) / Kharkiv: 2015. P. 223-225. **13. Barannik V.** Methodological base for representation transformants in equilibrium uneven-diagonal / V.V. Barannik, Sergii Shulgin, A.P. Musienko // 2015 1st International Conference ["Advanced Information and Communication Technologies-2015 (AICT'2015)"], (Lviv, Ukraine, October 29 - November 1, 2015) / Lviv: 2015. P. 138-140. **14. Barannik V.V.** Methodological basis for determining the energy significance of the structural unit of a video frame based on the estimation of low-frequency components of the matrices of the DCT blocks of the luminance component / V.V. Barannik, Dmitry Komolov, A.P. Musienko, R.V. Tarnopolov // XIVth International Conference ["Modern Problems of Radio Engineering, Telecom-munications and Computer Science, TCSET'2016"], (Lviv-Slavske, Ukraine, February 22-26, 2016) / Lviv-Slavske: 2016. P. 572-574.

Transliterated bibliography:

1. Mosov S. Aerokosmicheskaya razvedka v sovremennykh voennykh konfliktakh: monografiya / S.Mosov. K.: Izd. dom. "Rumb", 2008. 248 s.
2. Barannik V.V. Metod povyisheniya informatsionnoy bezopasnosti v sistemah videomonitoringa krizisnykh situatsiy / V.V. Barannik, Yu.N. Ryabuha, O.S. // Monografiya. Cherkassy, 2015. 143 s.
3. Barannik V.V. Metod selektsii kadrovogo potoka v sistemah kriticheskogo aeromonitoringa dlya povyisheniya bezopasnosti gosudarstvennogo informatsionnogo resursa / V.V. Barannik, Yu.N. Ryabuha, S.S. Bulba // Aviatsonno-kosmicheskaya tehnika i tehnologii. 2015. # 3. S. 111-118.
4. Barannik V.V. Kontseptualnyy metod povyisheniya bezopasnosti distantsionnogo videoinformatsionnogo resursa v sisteme aeromonitoringa krizisnykh situatsiy na osnove intellektualnoy obrabotki videokadrov / V.V. Barannik, Yu.N. Ryabuha // Radioelektronnyye kompyuternyye sistemy. 2015. # 3. S. 19-21.
5. Barannik V.V. Metodologiya sovershenstvovaniya obrabotki videoinformatsii, dlya povyisheniya effektivnosti servisa predostavleniya distantsionnykh videouslug, pri upravlenii v krizisnykh situatsiyah / V.V. Barannik, Yu.N. Ryabuha, A.A. Krasnorutskiy, V.Zh. Yaschenok // ASU i pribory avtomatiki. #170. 2015. S. 12-20.
6. Barannik V.V. Metod povyisheniya dostupnosti videoinformatsii aeromonitoringa / V.V. Barannik, O.S. Kulitsa // Radioelektronnyye i kompyuternyye sistemy. #3. 2013. S. 17-20.
7. Gonsales R.S., Vuds R.E. Tsifrovaya obrabotka izobrazheniy / R.S. Gonsales, R.E. Vuds. M.: Tehnosfera, 2006. 1072 s.
8. Kashkin V.B. Tsifrovaya obrabotka aerokosmicheskikh izobrazheniy. Krasnoyarsk: IPK SFU, 2008. 121 s.
9. Krasilnikov N.N. Tsifrovaya obrabotka izobrazheniy / N. N. Krasilnikov. M.: Vuzovskaya kniga, 2011. 320.
10. Barannik V.V. Metod klasterizatsii fragmentov aerofotosnimkov v spektralno-chastotnom prostranstve //

Barannik V.V., Musienko O.P., Yalivets K.S. // NaukoEmnI tehnologiyi. 2016, #29(1). 2016. S. 23-30.

11. Blizorukov M. G. Kolichestvennyye metody analiza mnogomernykh velichin / Blizorukov M. G. Uralo-Sibirskiy institut biznesa. Izdatelstvo AMB, 2006. 68 s.

12. Barannik V. The Evaluation Method Of Coding Efficiency Of Basic Frames Of The Video Stream In Infocommunication / V.V. Barannik, A.P. Musienko // IEEE Second International Scientific-Practical Conference ["IEEE Problems of Infocommunications. Science and Technology, PICS&T'2015"], (Kharkiv, Ukraine, October 13-15, 2015) / Kharkiv: 2015. P. 223-225.

13. Barannik V. Methodological base for representation transformants in equilibrium uneven-diagonal / V.V. Barannik, Sergii Shulgin, A.P. Musienko // 2015 1st International Conference ["Advanced Information and Communication Technologies-2015 (AICT'2015)"], (Lviv, Ukraine, October 29 - November 1, 2015) / Lviv: 2015. P. 138-140.

14. Barannik V.V. Methodological basis for determining the energy significance of the structural unit of a video frame based on the estimation of low-frequency components of the matrices of the DCT blocks of the luminance component / V.V. Barannik, Dmitry Komolov, A.P. Musienko, R.V. Tarnopolov // XIVth International Conference ["Modern Problems of Radio Engineering, Telecom-munications and Computer Science, TCSET'2016"], (Lviv-Slavske, Ukraine, February 22-26, 2016) / Lviv-Slavske: 2016. P. 572-574.

Поступила в редколлегию 11.06.2016

Рецензент: д-р техн. наук, проф. Безрук В.М.

Баранник Владимир Викторович, д-р техн. наук, профессор, начальник кафедры Харьковского национального уни-верситета Воздушных Сил имени Ивана Кожедуба. Научные интересы: системы, технологии преобразования, кодирования, семантической обработки изображений. Адрес: Украина, 61023, Харьков - 23, ул. Сумская, 77/79, тел. 8 050-3038971.

Barannik Vladimir Viktorovich, Doctor of Science (eng.), Professor, chief of chair, Kharkiv National University of Aircraft of the name of Ivan Kozhedub. Scientific interests: systems, tech-nologies of transformation, encoding, defence and information transfer, semantic processing of images. Address: Ukraine, 61023, Kharkiv - 23, Sumskaya street , 77/79, tel. 8 050-3038971. E-mail: Barannik_V_V@mail.ru.

Мусиенко Александр Павлович, адъюнкт кафедры Харьковского национального университета Воздушных Сил им. Ивана Кожедуба. Научные интересы: кодирование и се-мантическая обработка изображений. Адрес: Украина, 61023, Харьков, ул. Сумская, 77/79, тел. 8 093-7103877.

Musienko Olexandr Pavlovich, adjunct department, Kharkiv National University of Aircraft of the name of Ivan Kozhedub. Scientific interests: semantic processing of images. Address: Ukraine, 61023, Kharkiv, Sums'ka street, 77/79. tel. 8 093-7103877.

Жуйков Дмитрий Борисович, канд. техн. наук, доцент, заместитель начальника факультета Харьковского национального университета Воздушных Сил им. Ивана Кожедуба. Ад-рес: Украина, 61023, Харьков, ул. Сумская, 77/79

Браиловский Николай Николаевич, доцент кафедры кибербезопасности и защиты Киевского национального уни-верситета имени Т.Г.Шевченко. Научные интересы: кодиро-вание и семантическая обработка изображений. Адрес: Київ, 01033, вул. Володимирська, 60.

Brailovskyi Mykola Mykolaevich, associate professor of the cyber security and defense department of Kyiv National University of the name of Taras Shevchenko. Scientific interests: semantic processing of images.