МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ, МОЛОДЕЖИ И СПОРТА УКРАИНЫ

ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ РАДИОЭЛЕКТРОНИКИ

ISSN 0135-1710

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ УПРАВЛЕНИЯ И ПРИБОРЫ АВТОМАТИКИ

Всеукраинский межведомственный научно-технический сборник

Основан в 1965 г.

Выпуск 154

Харьков
2011

УДК 681.3

А.А. БАРКАЛОВ, Л.А. ТИТАРЕНКО, А.Н. МИРОШКИН

ЭЛЕМЕНТАРИЗАЦИЯ ОПЕРАТОРНЫХ ЛИНЕЙНЫХ ЦЕПЕЙ УПРАВЛЯЮЩЕГО АЛГОРИТМА ПРИ РЕАЛИЗАЦИИ КОМПОЗИЦИОННЫХ УСТРОЙСТВ УПРАВЛЕНИЯ НА FPGA-МИКРОСХЕМАХ

Рассматривается влияние элементаризации операторных линейных цепей алгоритма управления на количество аппаратурных затрат при реализации композиционного устройства управления на современных микросхемах программируемой логики (Spartan3). Определяются области параметров исходных ГСА, при которых целесообразно применение методики элементаризации операторных линейных цепей алгоритма. Приводятся результаты исследований.

1. Введение

Цифровая система, реализующая принцип микропрограммного управления [1], состоит из операционного автомата (ОА) и устройства управления (УУ). Операционный автомат содержит аппаратные средства, способные выполнять конечное множество операций над данными, а устройство управления интерпретирует алгоритм обработки данных, вырабатывая последовательность управляющих сигналов, под воздействием которых операционный автомат выполняет необходимые микрооперации [2]. Для задания последовательности необходимых микроопераций в данной работе используется язык граф-схем алгоритма (ГСА). Функционирование УУ, реализующего ГСА, может быть описано моделью цифрового автомата [3]. Реализация устройства управления в базисе современных программируемых логических интегральных схем (ПЛИС) характеризуется такими показателями как количество аппаратурных затрат, тактовая частота микросхемы, потребляемая и рассеиваемая мощности. Современные ПЛИС позволяют реализовывать все более сложные схемы, однако уменьшение аппаратурных затрат остается актуальной задачей, поскольку позволяет увеличить скорость работы, а также уменьшить потребляемую и рассеиваемую мощности устройства.

Данная статья посвящена исследованию одного из подходов, направленных непосредственно на уменьшение аппаратурных затрат в схеме композиционного микропрограммного устройства управления (КМУУ) с разделением кодов в базисе ПЛИС типа программируемых вентильных матриц (Field Programmable Gate Array, FPGA).

Множество подходов к уменьшению аппаратурных затрат в устройствах управления уже известны [4-8]. Однако использование новых элементных базисов и модификация известных подходов в некоторых случаях могут давать лучшие результаты.

Базис FPGA содержит все необходимые средства для реализации сложных цифровых систем. Наличие специализированных элементов, таких как синхронные блоки встроенной памяти (Embedded Memory Block, EMB), делают базис FPGA очень удобным для реализации схем с памятью, таких как композиционные управляющие автоматы. Целый ряд работ посвящен реализации конечных автоматов с памятью в базисе FPGA [5–9].

Целью данной работы является уменьшение комбинационной части КМУУ благодаря применению методики элементаризации операторных линейных цепей алгоритма управления.

Задачей исследования является разработка методики синтеза КМУУ для ГСА с элементарными операторными линейными цепями. Вторая часть статьи содержит некоторые сведения о синтезе исследуемых структур КМУУ. Результаты исследований и некоторые графики представлены в третьей части. Выводы и направления дальнейших исследований завершают данную статью.

2. Исследуемые структуры устройств управления

Абстрактный автомат задается пятью компонентами: (X, Y, S, f, g), где $X = \{x_1, ..., x_L\}$ – входной алфавит, $Y = \{y_1, ..., y_N\}$ – выходной алфавит, S – множество внутренних состояний автомата, $f : X \times S \to S$ – функции перехода автомата, $g : X \times S \to Y$ – функции выхода. Автомат, имеющий шестую компоненту, s_0 – начальное состояние, называется детерминированным. Конечным называется автомат, имеющий конечное множество внутренних состояний. Цифровое устройство управления является детерминированным конечным цифровым автоматом.

Одним из способов реализации управляющего алгоритма в виде конечного автомата является композиционное микропрограммное устройство управления [10, 11]. Принцип организации КМУУ основан на выделении в ГСА операторных линейных цепей (ОЛЦ).

Операторной линейной цепью α_g ГСА Γ называется конечная упорядоченная последовательность $< b_{g_1}, ..., b_{g_{F_g}} >$ из F_g операторных вершин ГСА Γ, для любой пары соседних компонент которой существует дуга, соединяющая их.

КМУУ является композицией автомата с жесткой и программируемой логикой. Автомат с жесткой логикой выполняет переходы между различными ОЛЦ, а автомат с программируемой логикой адресует состояния в пределах каждой ОЛЦ. Прямая структурная таблица (ПСТ) КМУУ описывает все возможные переходы между различными ОЛЦ. Строка ПСТ включает следующие элементы: текущее состояние a_m, его двоичный код $k(a_m)$, следующее состояние a_s и его код $k(a_s)$, множество логических сигналов X, обеспечивающих переход из состояния a_m в состояние a_s, функции D возбуждения памяти регистра состояния автомата и номер h строки.

Функции перехода задаются формулой:

$$D_i = \bigvee_{h=1}^{H} (P_i^h \wedge (\bigwedge_{r=1}^{R} Q_r^h) \wedge (\bigwedge_{l=1}^{L} C_l^h x_l^h)), \; i = \overline{1, R} \qquad (1)$$

и реализуются автоматом с жесткой логикой, где D_i – функция возбуждения i-го разряда регистра состояния автомата; P_i^h – переменная, определяющая наличие функции D_i в строке h ПСТ ($P_i^h = 1$, если функция D_i присутствует в строке h, и $P_i^h = 0$ в обратном случае); Q_r^h – выход r-го разряда регистра состояния автомата в строке h ПСТ автомата ($Q_r^h = Q_r$, если на выходе r-го разряда регистра находится значение логической «1», и $Q_r^h = \overline{Q_r}$ в обратном случае); x_l^h – переменная, определяющая значение логического условия x_l в строке h ПСТ автомата ($x_l^h = x_l$, если переход в строке h выполняется при выполнении условия, и $x_l^h = \overline{x_l}$ – при невыполнении); C_l^h – переменная, определяющая наличие логического условия x_l в строке h ПСТ автомата ($C_l^h = 1$ при наличии сигнала x_l и $C_l^h = 0$ при его отсутствии); H – количество строк ПСТ автомата; R – разрядность регистра состояния автомата; L – количество сигналов во входном алфавите автомата.

Функции выхода КМУУ определяются только для каждого состояния автомата с программируемой логикой, причем выходные функции не зависят от логических условий и формируются при помощи ПЗУ, следовательно, они формально описываются как $g : S \to Y$.

Операторная вершина b_m называется *входом* ОЛЦ α_g, если существует дуга $\langle b_t, b_m \rangle$, или *выходом*, если существует дуга $\langle b_m, b_t \rangle$, где $b_t \notin \alpha_g$.

Функционирование КМУУ подразумевает естественную адресацию вершин в пределах каждой ОЛЦ α_g, т.е. для перехода $\langle b_i \to b_{i+1} \rangle$ справедливо равенство

$$A(b_{i+1}) = A(b_i) + 1, \qquad (2)$$

где $A(b_i), A(b_{i+1})$ – адреса вершин b_i и b_{i+1} соответственно, $b_i, b_{i+1} \in \alpha_g$.

Для программной реализации более удобным является алгоритм формирования ОЛЦ, основанный на математической модели произвольной операторной вершины b_m ГСА. Для описания алгоритма введем следующие обозначения:

O_m^i – количество переходов $< b_g \to b_m >$, где $b_g \in B_1$, B_1 – множество операторных вершин граф-схемы алгоритма управления;

O_m^o – количество переходов $< b_m \to b_g >$, где $b_g \in B_1$.

Алгоритм формирования ОЛЦ заключается в следующем:

1. Поиск главного входа очередной ОЛЦ по признаку $O_m^i = 0$. Отметка найденной вершины как входа ОЛЦ. Назначение ей очередного кода ОЛЦ и нулевого кода компоненты. При отсутствии вершины с $O_m^i = 0$ – переход на п. 4.

2. Если следующая вершина является условной или конечной, то текущая помечается как вершина-выход текущей ОЛЦ и выполняется переход на п. 1. Иначе переход на следующую вершину.

3. Если для текущей операторной вершины выполняется неравенство $O_m^i \geq 1$, то она может войти в одну из нескольких ОЛЦ. Такая вершина помечается для дальнейшего анализа, затем выполняется переход на п. 1. Иначе назначение текущей вершине очередного кода компоненты. Переход на п. 2.

4. При наличии вершины, которая может войти в одну из нескольких ОЛЦ, определить ее в ОЛЦ-претендент с меньшей длиной. Переход на п. 2. При отсутствии такой вершины выполнить переход на п. 5.

5. Конец.

Описанный алгоритм позволяет получить множество $C = \{\alpha_1, ..., \alpha_G\}$ минимальной мощности.

Принцип разделения кодов (Code Sharing, CS-структура КМУУ) [11, 12] реализуется следующим образом: адрес любой операторной вершины определяется как конкатенация кода ОЛЦ, которой она принадлежит, и кода данной вершины в пределах ОЛЦ (кода компоненты, порядкового номера вершины).

Такая адресация позволяет ввести в схему регистр для хранения кода текущей ОЛЦ и счетчик для хранения кода текущей компоненты ОЛЦ.

Под длиной ОЛЦ будем понимать количество входящих в нее операторных вершин: $F_g = |\alpha_g|$. Пусть $F_{max} = \max(F_1, ..., F_G)$, где G – количество ОЛЦ в ГСА, тогда разрядность регистра равна

$$R_1 = \lceil \log_2 G \rceil, \tag{3}$$

а счетчик имеет

$$R_2 = \lceil \log_2 (F_{max}) \rceil \tag{4}$$

разрядов.

Система функций выходов КМУУ с разделением кодов реализуется на ПЗУ тривиально: в памяти расположена микропрограмма из

$$M = (G-1) \cdot 2^{R_2} + F_{min} \tag{5}$$

слов, где $F_{min} = \min(F_1, ..., F_G)$. Для обеспечения равенства (5) необходимо ОЛЦ минимальной длины закодировать максимальным кодом ОЛЦ, что отнесет соответствующее содержимое управляющей памяти (УП) в область с максимальными адресами.

При унитарном кодировании микрокоманд [11] разрядность слова УП равна

$$N = |Y| + 2, \tag{6}$$

где константа «2» определяет разряды для хранения внутренних переменных y_0 и y_E.

На рис. 1, а представлена структурная схема КМУУ CS. Схема формирования адреса (СФА) реализует систему функций переходов (1), а управляющая память содержит микро-

программу, в которой находятся закодированные одним из известных способов [13] выходные функции.

По сигналу Start в регистр Рг и счетчик СТ заносится начальный адрес микропрограммы, а триггер выборки ТВ разрешает чтение из управляющей памяти. Если считанная из УП микрокоманда не соответствует выходу ОЛЦ, то одновременно с микрооперациями Y формируется сигнал y_0. Если $y_0 = 1$, то к содержимому СТ прибавляется единица и адресуется следующая компонента текущей ОЛЦ, содержимое Рг при этом остается неизменным. Если выход ОЛЦ достигнут, то $y_0 = 0$. При этом адрес входа следующей ОЛЦ вырабатывается схемой формирования адреса, после чего новый код ОЛЦ записывается в Рг, а номер компоненты – в СТ. При достижении окончания микропрограммы формируется сигнал y_E, триггер ТВ обнуляется и выборка микрокоманд прекращается.

Рассмотрим принцип элементаризации ОЛЦ. Схема формирования адреса вырабатывает код следующей ОЛЦ и номер ее компоненты. В том случае, если адресацию операторных вершин выполнить таким образом, что каждая ОЛЦ имеет ровно один вход и один выход, то переход всегда будет осуществляться на компоненту с нулевым кодом, что избавит от необходимости его формирования. При этом количество ОЛЦ может увеличиться, и СФА будет формировать адрес, разрядность которого вместо $R_1 + R_2$ (3), (4) будет составлять

$$R_1' = \lceil \log_2(G') \rceil, \qquad (7)$$

где G' – количество ОЛЦ после применения методики элементаризации. Структурная схема КМУУ с разделением кодов и элементаризацией ОЛЦ приведена на рис. 1, б.

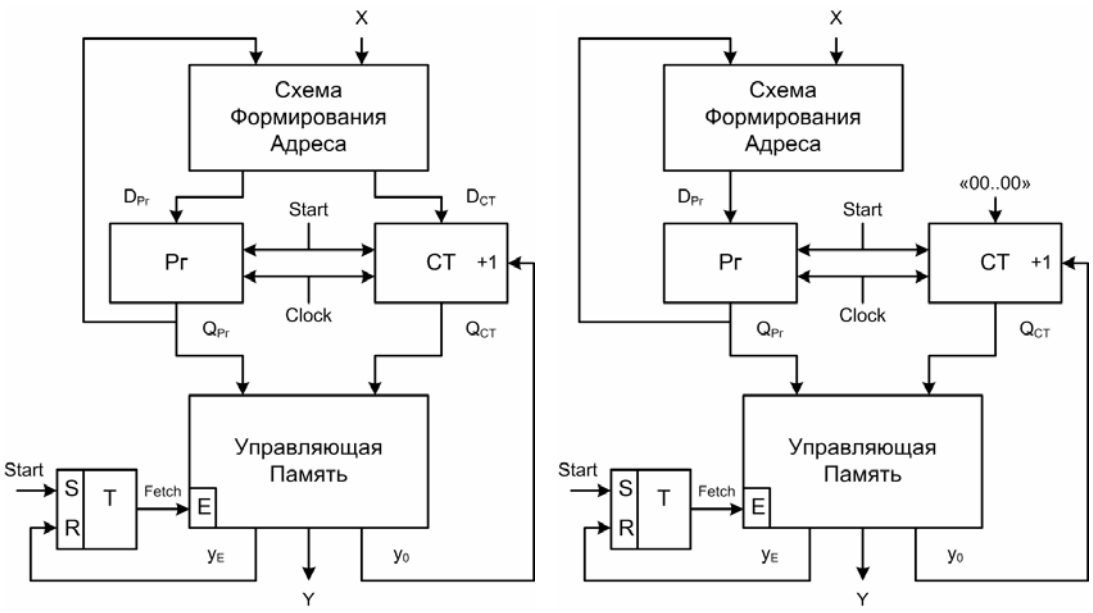


Рис. 1. Структурная схема КМУУ: а – с разделением кодов (CS);
б – с разделением кодов и элементаризацией ОЛЦ (ECS)

3. Результаты экспериментов

Для эксперимента были выбраны ГСА со следующими параметрами:
– количество вершин от 10 до 500 с шагом 10;
– доля операторных вершин от 50 до 90% с шагом 10%;
– количество микроопераций = 15;
– количество логических условий = 5.

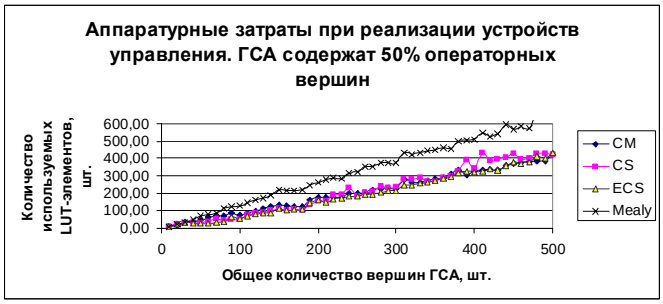
Для указанных ГСА были получены результаты имплементации КМУУ с разделением кодов без применения элементаризации ОЛЦ и с применением таковой (рис. 2-4). Также

приводится сравнительная характеристика результатов реализации тех же ГСА в виде автоматов Мили и в виде КМУУ с общей памятью [11] (СМ-структура). В качестве элементного базиса выбраны FPGA-микросхемы Spartan-6 фирмы Xilinx.

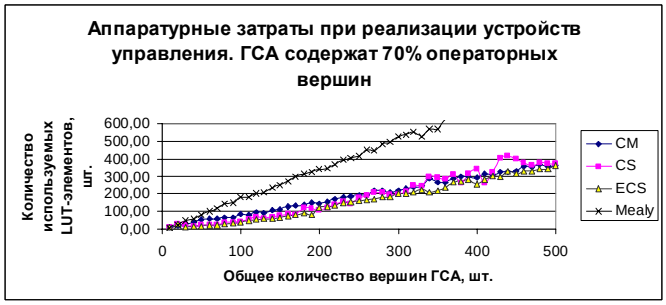
Для автоматизации процесса синтеза управляющих устройств была разработана система Генератор Автоматов (ГенА), которая по описанию алгоритма управления в формате XML генерирует VHDL-модель управляющего устройства заданной структуры. Под моделью понимается описание структурных элементов устройства на языке описания аппаратуры VHDL и при необходимости файлы прошивки ПЗУ. Далее модель автомата поступает в САПР Xilinx ISE, где проходит процесс имплементации в одну из доступных микросхем по выбору пользователя. Отчет об имплементации содержит подробную количественную информацию об использовании ресурсов микросхемы, которая и подлежит дальнейшему анализу.

При получении набора экспериментальных данных был выполнен синтез и имплементация пяти ГСА для каждой измеряемой точки.

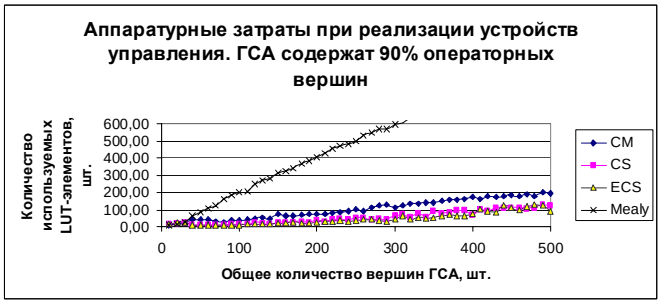
Анализ графиков на рис. 2 позволяет сделать вывод о том, что для реализации любой ГСА, параметры которой находятся в исследуемых пределах, в виде КМУУ необходимо количество аппаратуры одного порядка. Рост аппаратурных затрат для реализации алгоритма в виде автомата Мили имеет линейный характер, причем угол наклона прямой, аппроксимирующей график затрат, с ростом процента операторных вершин увеличивается, а у композиционных устройств уменьшается.



а



б



в

Рис. 2. Аппаратурные затраты при реализации автомата Мили, КМУУ с общей памятью, с разделением кодов, с элементаризацией ОЛЦ. Процент операторных вершин в ГСА: а – 50%; б – 70%; в – 90%

Для сравнения структур КМУУ между собой рассмотрим график на рис. 3, который был нормирован относительно аппаратурных затрат для КМУУ с общей памятью.

Как видно из рис. 3, структура КМУУ с разделением кодов требует меньшие аппаратурные затраты для ГСА с количеством вершин от 10 до 380, после чего определить, даст ли данная структура более выгодную в сравнении с КМУУ с общей памятью реализацию, можно лишь выполнив синтез обеих структур для каждой конкренной ГСА.

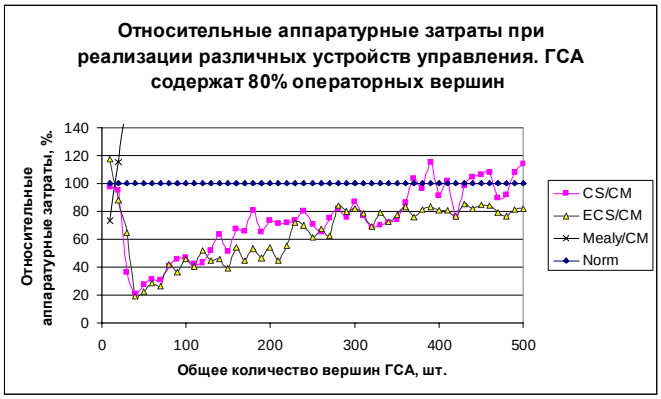


Рис. 3. Относительные аппаратурные затраты при реализации автомата Мили, КМУУ с разделением кодов, с элементаризацией ОЛЦ по отношению к значениям для КМУУ с общей памятью. ГСА содержат 80% операторных вершин

Применение элементаризации ОЛЦ управляющего алгоритма делает ECS-структуру КМУУ более экономной в плане необходимых ресурсов в сравнении с КМУУ с общей памятью на всем исследуемом отрезке параметров ГСА

Для сравнения CS- и ECS-структур КМУУ рассмотрим рис. 4, где выполнено нормирование аппаратурных затрат по отношению к КМУУ без элементаризации ОЛЦ управляющего алгоритма.

График на рис. 4 показывает, что в 26% из общего числа ГСА с 80% операторных вершин структура КМУУ без элементаризации ОЛЦ алгоритма управления оказывается эффективнее. Общее количество вершин этих ГСА сосредоточены на отрезках [10; 120] и [270; 360]. На указанных отрезках необходимо выполнять синтез обеих структур для выяснения той, которая даст меньшие аппаратурные затраты в каждом конкретном случае. Для ГСА, параметры которых не попадают в указанные отрезки, достаточно синтезировать лишь КМУУ с элементаризацей ОЛЦ.

Рис. 4. Относительные аппаратурные затраты при реализации КМУУ с использованием методики элементаризации ОЛЦ алгоритма управления и без таковой

4. Выводы

Предлагаемая методика элементаризации ОЛЦ алгоритма при реализации КМУУ с разделением кодов в базисе ПЛИС типа FPGA направлена на уменьшение числа LUT-

элементов, используемых в схеме формирования адреса микрокоманд. Методика заключается в формировании ОЛЦ с единственным входом, что позволяет уменьшить количество формируемых разрядов адреса при переходе между различными ОЛЦ с величины $R_1 + R_2$ (3), (4) до R_1' (7).

Проведенные исследования показали, что применение методики элементаризации ОЛЦ исходных ГСА может снизить количество аппаратурных затрат в среднем на 10-15% в сравнении с базовой структурой КМУУ при реализации схем устройств управления в базисе современных FPGA-микросхем. Если количество вершин ГСА принадлежит отрезкам [10; 120] или [270; 360], применение данной методики может оказаться неэффективным, однако проверить это можно лишь после синтеза всех сравниваемых структур КМУУ для конкретной ГСА.

Научная новизна предлагаемой методики заключается в установлении аналитической зависимости между параметрами входного алгоритма и количеством необходимых для реализации схемы ресурсов микросхемы FPGA. Для микросхемы Spartan-3 фирмы Xilinx аналитическая зависимость имеет вид $Q = (-0{,}043p^2 + 4{,}554p - 40{,}8) \cdot N$, где Q – количество LUT-элементов, необходимых для реализации схемы, N – общее количество вершин ГСА, p – доля операторных вершин в ней ($p \in [0.5;0.9]$). Данная зависимость может быть использована для других FPGA-микросхем фирмы Xilinx с четырехвходовыми LUT-элементами. Погрешность формулы для ГСА с количеством вершин более 40 не превышает 30%, в то время как для малых ГСА статистическая погрешность может достигать величины аппаратурных затрат.

Период синхросигнала схемы КМУУ составляет 70-110% соответствующего значения для автомата Мили. Схема автомата Мили может функционировать быстрее для интерпретации более разветвленных ГСА, а при увеличении доли операторных вершин она уступает в скорости схеме КМУУ.

Длительность формирования выходных сигналов для рассматриваемых структур КМУУ одинакова и составляет 3,6-5 нс против 7-11 нс для автомата Мили, причем для КМУУ эти значения постоянны, а в автомате Мили задержки растут при увеличении сложности схемы.

Практическая значимость методики состоит в уменьшении количества необходимых ресурсов микросхемы, а также в возможности определения этого количества до начала процесса синтеза схемы, что позволит выбрать подходящую микросхему.

Дальнейшие направления исследований связаны с анализом эффективности применения различных подходов к кодированию вершин ГСА, использования незанятых участков управляющей памяти для уменьшения аппаратурных затрат при реализации устройства управления.

Список литературы: 1. *Wilkes M.V.* The Best Way to Design an Automatic Calculating Machine (1951, July), Rept. Manchester Univ. Computer lnaug. Conf., Manchester, U.K. **2.** *Baranov S.* Logic and System Design of Digital Systems. Tallinn: TUT Press, 2008. 266 p. **3.** *Глушков В.М.* Синтез цифровых автоматов. М.: Физматгиз, 1962. 476 с. **4.** *ROM*-based finite state machine implementation in low cost FPGAs / I. Garcна-Vargas, R. Senhadji-Navarro, G. Jimйnez-Moreno, A. Civit-Balcells, P. Guerra-Gutiйrrez // (2007) IEEE International Symposium on Industrial Electronics, art. no. 4374972. P. 2342-2347. **5.** *ROM*-based FSM implementation using input multiplexing in FPGA devices / R. Senhadji-Navarro, I. Garcна-Vargas, G. Jimйnez-Moreno , A. Civit-Ballcels // (2004) Electronics Letters, 40 (20). P. 1249-1251. **6.** *An application* of functional decomposition in ROM-based FSM implementation in FPGA devices / M. Rawski, H. Selvaraj, T. Juba // (2005) Journal of Systems Architecture, 51 (6-7). P. 424-434. **7.** *Synthesis* and Implementation of RAM-Based Finite State Machines in FPGAs. / V. Sklyarov // 10th International Conference «Field-Programmable Logic and Applications: The Roadmap to Reconfigurable Computing» Proceedings, FPL 2000 Villach, Austria, August 27–30, 2000. P. 718-727. **8.** *Saving* power by mapping finite-state machines into embedded memory blocks in FPGAs. / A. Tiwari, K.A. Tomko // (2004) Proceedings - Design, Automation and Test in Europe Conference and Exhibition, 2. P. 916-921. **9.** *Garcia E.* Creating Finite State Machines Using True Dual-Port Fully Synchronous SelectRAM Blocks // Xcell Journal, 2000, Issue 38. P. 36-38. **10.** *Баркалов А.А.* Микропрограммное устройство управления как композиция автоматов с программируемой и жесткой логикой // Автоматика и вычислительная техника. 1983. №4. С.36-41. **11.** *Баркалов А.А.* Синтез устройств управления на программируемых логических устройствах. Донецк: ДонНТУ, 2002. 262 с. **12.** *Оптими-*

зация устройства управления с разделением кодов / А.А. Баркалов, Л.А. Титаренко, А.Н. Мирошкин / / Управляющие системы и машины. 2010. N 1. С. 45-50. **13.** *Синтез* микропрограммных устройств управления. Баркалов А.А., Палагин А.В. К.: ИК НАН Украины, 1997. 135 с.

Поступила в редколлегию 21.03.2011

Баркалов Александр Александрович, д-р техн. наук, проф. кафедры компьютерной инженерии ДонНТУ, проф. университета Зеленогурского (Польша). Научные интересы: цифровые устройства управления. Хобби: научная работа, спорт. Адрес: Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Gora. E-mail: A.Barkalov@iie.uz.zgora.pl.

Титаренко Лариса Александровна, д-р техн. наук, проф. кафедры ТКС ХНУРЭ, проф. университета Зеленогурского (Польша). Научные интересы: системы телекоммуникаций, цифровые устройства управления. Хобби: научная работа, спорт. Campus A, Budynek Dydaktyczny / A-2 prof. Z. Szafrana str. 2, 65-516 Zielona Gora E-mail: L.Titarenko@iie.uz.zgora.pl.

Мирошкин Александр Николаевич, аспирант кафедры компьютерной инженерии ДонНТУ. Научные интересы: цифровая схемотехника. Хобби: научная работа, спорт. Адрес: Украина, Донецкая обл., Макеевка, ул. Курская, д. 15, кв. 45.

УДК 658.512.011:681.326:519.713

НГЕНЕ КРИСТОФЕР УМЕРАХ (NGENE CHRISTOPHER UMERAH),
В.И. ХАХАНОВ, С.А. ЗАЙЧЕНКО, Е.И. ЛИТВИНОВА, О.Б. СКВОРЦОВА

МОДЕЛИ И МЕТОДЫ ВЕРИФИКАЦИИ И ДИАГНОСТИРОВАНИЯ SOC HDL-КОДА

Предлагается xor-метрика отношений объектов в векторном логическом пространстве и основанная на ней структурно-аналитическая модель тестирования цифровых систем на кристаллах. Описываются ассерционно-ориентированные модели и методы верификации и диагностирования функциональных нарушений HDL-кода, которые дают возможность существенно уменьшить время проектирования программных и аппаратных продуктов. Показывается архитектурная модель мультиматричного процессора с ограниченной системой логических команд для решения задач встроенного диагностирования.

Актуальность. Тенденция последних лет в части создания новых коммуникационных, вычислительных и информационных сервисов, полезных для человека, обращает внимание на создание все более специализированных гаджетов (gadget), обладающих существенными преимуществами перед персональными компьютерами и ноутбуками: энергопотребление, компактность, вес, стоимость, функциональные возможности, дружественность интерфейса. Практически вся десятка лучших за 2010 год специализированных изделий (Apple iPad, Samsung Galaxy S, Apple MacBook Air, Logitech Revue, Google Nexus One (HTC Desire), Apple iPhone 4, Apple TV, Toshiba Libretto W100, Microsoft Kinect, Nook Color) реализована в виде цифровых систем на кристаллах. К 2012 году рынок мобильной и беспроводной связи перейдет на 20 нм (итоги январского Саммита-2011 альянса Common Platform). Дальнейшее развитие технологий по годам: 2014 – 14 нм, 2016 – 11 нм. В 2015 году 55% сотовых телефонов станут смартфонами, планшетные компьютеры заменят ноутбуки и нетбуки. Суперфоны (Nexus-1, Google) станут той соединительной тканью, которая свяжет все остальные устройства и сервисы. Переход от вычислительных платформ к мобильным устройствам с малым форм-фактором приведет к существенному снижению энергопотребления во всем мире. Надвигается следующая волна компьютеризации под названием «интернет вещей», которая приведет к широкому распространению датчиковых сетей, включая их интеграцию в человеческое тело. Мировой рынок перечисленных выше устройств и гаджетов насчитывает сегодня порядка 3 миллиардов изделий. Для их эффективного проектирования, производства и эксплуатации создаются новые технологии и инфраструктуры сервисного обслуживания на стадиях проектирования, производства и эксплуатации. Один из возможных шагов в данном направлении представлен ниже в виде структуры публикации как технологии верификации T^v: M^t – метрика и модель процессов тестирования, H^c – HDL-код проекта, G^t – синтез графа транзакций программных блоков,

$\{M^f, M^s\}$ – построение двух моделей верификации HDL-кода (таблица функциональных нарушений и матрица активизации программных блоков), $\{D^c, D^r, D^m\}$ – разработка трех методов (анализа строк, столбцов таблицы и матрицы в целом) диагностирования функциональных нарушений, использующих механизм ассерций (ассерция – логическое высказывание, определяющее семантические ошибки программного кода), P^m – создание архитектуры мультиматричного процессора для параллельного анализа табличных данных, R – имплементация моделей методов и средств в систему Riviera компании Aldec:

$$T^v = M^t \to H^c \to G^t \to \begin{Bmatrix} M^f \to \begin{Bmatrix} D^c \\ D^r \end{Bmatrix} \\ M^s \to D^m \end{Bmatrix} \to P^m \to R.$$

Цель – существенное уменьшение времени проектирования и повышение качества цифровых систем на кристаллах за счет разработки ассерционно-ориентированной инфраструктуры, моделей и методов верификации и диагностирования HDL-кода. Информация, необходимая для поиска блоков с функциональными нарушениями, определяется в процессе моделирования(выполнения) программного кода. Эффективность проектирования цифрового изделия определяется средним и нормированным в интервале [0,1] интегральным критерием:

$$E = F(L, T, H) = \min[\frac{1}{3}(L + T + H)],\ Y = (1 - P)^n;\ L = 1 - Y^{(1-k)} = 1 - (1 - P)^{n(1-k)};$$

$$T = [(1 - k) \times H^s]/(H^s + H^a);\ H = H^a/(H^s + H^a).$$

Критерий учитывает: уровень ошибок проекта L, время верификации T, программно-аппаратную избыточность, определяемую механизмами ассерций и средствами сервисного обслуживания H. Параметр L, как дополнение к параметру Y, характеризующему выход годной продукции, зависит от тестопригодности проекта k, вероятности P существования неисправных компонентов и числа необнаруженных ошибок n. Время верификации определяется тестопригодностью проекта k [3,4], умноженной на структурную сложность аппаратно-программной функциональности, отнесенную к общей сложности проекта в строках кода. Программно-аппаратная избыточность находится в функциональной зависимости от сложности ассерционного кода и других затрат, отнесенных к общей сложности проекта. При этом программная или аппаратная, избыточность должна обеспечивать заданную глубину диагностирования ошибок функциональности и время выхода изделия на рынок, определенное заказчиком.

Задачи: 1) Создание метрики и структурно-аналитической модели тестирования цифровых систем на кристаллах. 2) Усовершенствование моделей и методов поиска функциональных нарушений на основе механизма ассерций для повышения быстродействия верификации и диагностирования HDL-кода. 3) Разработка архитектурной модели мультиматричного процессора для решения задач диагностирования.

Источники: 1. Модели формулирования и классификации задач технической диагностики [1-6]. 2. Диагностирование и верификация цифровых систем на кристаллах [9-17, 22-15]. 3. Аппаратура и матричные процессоры для ускорения процессов тестирования [18-21].

1. Модель процессов тестирования и верификации

Предлагаются технологичные и эффективные процесс-модели и методы диагностирования функциональных нарушений в программных и/или аппаратных продуктах. Используются регистровые или матричные (табличные) структуры данных, которые ориентированы на параллельное выполнение логических операций при поиске дефектных компонентов.

Проблема синтеза или анализа компонентов произвольной системы может быть сформулирована в виде взаимодействия (симметрической разности – аналог xor-операции на булеане) в кибернетическом пространстве ее модели F с входными воздействиями T и реакциями L:

$$f(F, T, L) = \varnothing \to F \Delta T \Delta L = \varnothing\ .$$

Киберпространство – совокупность взаимодействующих по метрике информационных процессов и явлений, использующих в качестве носителя компьютерные системы и сети. В частности, компонент пространства представлен k-мерным (кортежем) вектором $a=(a_1,a_2,...,a_j,...,a_k)$, $a_j=\{0,1\}$ в двоичном алфавите. Нуль-вектор есть k-мерный кортеж, все координаты которого равны нулю: $a_j=0,\ j=\overline{1,k}$.

Метрика β кибернетического (двоичного) пространства определяется единственным равенством, которое формирует нуль-вектор для xor-суммы расстояний d_i между ненулевым и конечным числом точек (объектов), замкнутых в цикл:

$$\beta=\bigoplus_{i=1}^{n} d_i=0.$$

Расстояние (по Хэммингу) между двумя объектами (векторами) a и b определяется в виде производного вектора: $d_i=d(a,b)=a_j\bigoplus_{j=1}^{k} b_j$. Иначе: метрика β векторного логического двоичного пространства есть равная нуль-вектору xor-сумма расстояний между конечным числом точек (вершин) графа, образующих цикл. Сумма n-мерных двоичных векторов, задающих координаты точек цикла, равна нуль-вектору. Данное определение метрики оперирует отношениями, что позволяет сократить систему аксиом с трех до одной и распространить ее действие на любые конструкции n-мерного киберпространства. Классическое задание метрики для определения взаимодействия одной, двух и трех точек в векторном логическом пространстве является частным случаем β-метрики при $i=1,2,3$ соответственно:

$$M=\begin{cases} d_1=0\leftrightarrow a=b; \\ d_1\oplus d_2=0\leftrightarrow d(a,b)=d(b,a); \\ d_1\oplus d_2\oplus d_3=0\leftrightarrow d(a,b)\oplus d(b,c)=d(a,c). \end{cases}$$

Метрика β кибернетического многозначного пространства, где каждая координата вектора (объекта) определена в алфавите, составляющем булеан на универсуме примитивов мощностью p: $a_j=\{\alpha_1,\alpha_2,...,\alpha_r,...,\alpha_m\},\ m=2^p$, есть равная $\varnothing$-вектору (по всем координатам) симметрическая разность расстояний между конечным числом точек, образующих цикл:

$$\beta=\bigtriangleup_{i=1}^{n} d_i=\varnothing. \qquad (1)$$

Равенство пустому вектору симметрической разности покоординатного теоретико-множественного взаимодействия (1) подчеркивает равнозначность компонентов (расстояний), формирующих уравнения, где единственная координатная операция $d_{i,j}\Delta d_{i+1,j}$, используемая, например, в четырехзначной модели Кантора, определяется соответствующей Δ-таблицей:

Δ	0	1	x	$\varnothing$
0	$\varnothing$	x	1	0
1	x	$\varnothing$	0	1
x	1	0	$\varnothing$	x
$\varnothing$	0	1	x	$\varnothing$

$\cap$	0	1	x	$\varnothing$
0	0	$\varnothing$	0	$\varnothing$
1	$\varnothing$	1	1	$\varnothing$
x	0	1	x	$\varnothing$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$

$\cup$	0	1	x	$\varnothing$
0	0	x	x	0
1	x	1	x	1
x	x	x	x	x
$\varnothing$	0	1	x	$\varnothing$

a	0	1	x	$\varnothing$
$\tilde{a}$	1	0	$\varnothing$	x

(2)

Здесь также приведены таблицы истинности для других базовых теоретико-множественных операций, далее используемых по тексту. Число примитивных символов, образующих замкнутый относительно теоретико-множественных координатных операций алфавит, может быть увеличено. При этом мощность алфавита (булеана) определяется выражением $m=2^p$, где p – число примитивных символов. Введенная метрика представляет

не только теоретический интерес, но имеет и практическую направленность на обобщение и классификацию задач технической диагностики путем создания модели xor-отношений на множестве из четырех основных компонентов. Процедуры синтеза тестов, моделирования неисправностей и поиска дефектов можно свести к xor-отношениям на графе (рис. 1) полного взаимодействия четырех вершин (функциональность, устройство, тест, дефекты) $G = \{F, U, T, L\}$.

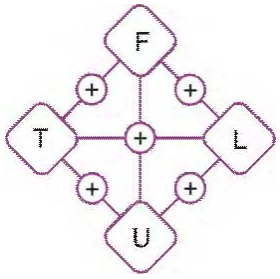


Рис. 1. Граф взаимодействия компонентов технической диагностики

Такой граф порождает четыре базовых треугольника, которые формируют 12 практически полезных триад отношений, формулирующих задачи технической диагностики:

$T \oplus F \oplus L = 0$	$T \oplus L \oplus U = 0$	$T \oplus F \oplus U = 0$	$F \oplus L \oplus U = 0$
1) $T = F \oplus L$	4) $T = L \oplus U$	7) $T = F \oplus U$	10) $F = L \oplus U$
2) $F = T \oplus L$	5) $L = T \oplus U$	8) $F = T \oplus U$	11) $L = F \oplus U$
3) $L = T \oplus F$	6) $U = T \oplus L$	9) $U = T \oplus F$	12) $U = F \oplus L$

Введение вершины U в граф взаимодействия компонентов технической диагностики расширяет функциональные возможности модели, появляются новые свойства полученной системы. Введение в структуру новой вершины должно иметь весомые аргументы в пользу ее целесообразности. Что касается представленного на рис. 1 графа, содержательно все формульно описанные задачи можно классифицировать в группы следующим образом.

Группа 1 – теоретические эксперименты (на модели функциональности), без устройства: 1) Синтез теста по модели функциональности для заданного списка неисправностей. 2) Построение модели функциональности на основе заданного теста и списка неисправностей. 3) Моделирование неисправностей функциональности на заданном тесте.

Группа 2 – реальные эксперименты (на устройстве), без модели функциональности:4) Синтез теста путем физической эмуляции дефектов в устройстве. 5) Определение списка неисправностей устройства при выполнении диагностического эксперимента. 6) Верификация теста и дефектов в эксперименте на реальном устройстве.

Группа 3 – тестовые эксперименты (верификация), без дефектов: 7) Синтез теста путем сравнения результатов моделирования модели и реального устройства. 8) Синтез функциональности по реальному устройству и заданному тесту. 9) Верификация теста и модели функциональности относительно реального устройства с существующими неисправностями.

Группа 4 – эксперименты в процессе функционирования, на рабочих воздействиях: 10) Проверка правильности поведения реального устройства на существующих или заданных дефектах. 11) Проверка работоспособности устройства относительно существующей модели в процессе функционирования. 12) Верификация функциональности и списка дефектов относительно поведения реального устройства.

Наиболее популярными задачами из перечисленного выше списка являются: 1, 3, 5, 8, 9. Можно ввести и другую классификацию типов задач, которая дает возможность определить на графе $G = (F, U, T, L)$ все концептуальные пути решения целевых проблем: синтеза тестов, определения модели функциональности, генерирования модели дефектов и проектирования устройства:

$$1)\ T = F \oplus L;\quad 4)\ F = T \oplus L;\quad 7)\ L = T \oplus F;\quad 10)\ U = T \oplus L;$$
$$2)\ T = U \oplus L;\quad 5)\ F = U \oplus L;\quad 8)\ L = T \oplus U;\quad 11)\ U = T \oplus F;$$
$$3)\ T = F \oplus U;\quad 6)\ F = T \oplus U;\quad 9)\ L = F \oplus U;\quad 12)\ U = F \oplus L.$$

Все конструкции, представленные в отношениях, обладают замечательным свойством обратимости. Компонент, вычисляемый с помощью двух других, может быть использован в качестве аргумента для определения любого из двух исходных. Поэтому здесь можно говорить о транзитивной обратимости каждой триады отношений на полном графе, когда по двум любым компонентам всегда и однозначно можно восстановить или определить третий. При этом формат представления каждого компонента должен быть одинаковым по форме и размерности (векторы, матрицы). На основе предложенной метрики и моделей тестирования далее рассмотрены более подробно методы диагностирования дефектов или функциональных нарушений.

2. Модель поиска функциональных нарушений в программных блоках

Используется уравнение пространства $f(F,T,L,U)=0 \rightarrow F \oplus T \oplus L \oplus U = 0$, которое трансформируется к виду $L = (T \oplus F) \oplus (T \oplus U)$. Диагностирование дефектов (функциональных нарушений) сводится к сравнению результатов модельного $(T \oplus F)$ и натурного $(T \oplus U)$ экспериментов, которое формирует список функциональных нарушений L, присутствующих в объекте диагностирования. Формула-модель процесса поиска блока F_i с функциональными нарушениями сводится к выбору решения посредством определения хог-взаимодействия между тремя компонентами:

$$L = F_i \leftarrow [(T \oplus F_i) \overset{p}{\underset{i=1}{\oplus}} (T \oplus U_i)] = 0.$$

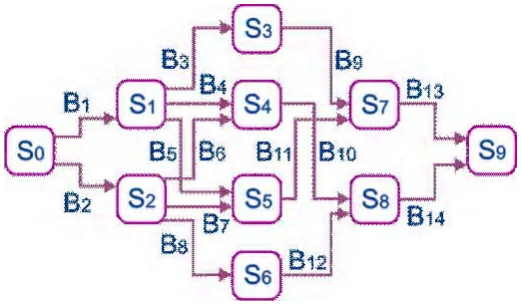
Аналитическая модель верификации HDL-кода с использованием механизма темпоральных ассерций (дополнительных линий наблюдения) ориентирована на достижение заданной глубины диагностирования и представлена в следующем виде:

$$\begin{array}{ll} M = f(F,A,B,S,T,L), & F = (A * B) \times S;\ S = f(T,B); \\ A = \{A_1, A_2, ..., A_i, ..., A_n\}; & B = \{B_1, B_2, ..., B_i, ..., B_n\}; \\ S = \{S_1, S_2, ..., S_i, ..., S_m\}; & S_i = \{S_{i1}, S_{i2}, ..., S_{ij}, ..., S_{ip}\}; \\ T = \{T_1, T_2, ..., T_i, ..., T_k\}; & L = \{L_1, L_2, ..., L_i, ..., L_n\}. \end{array} \quad (3)$$

Здесь $F = (A * B) \times S$ – функциональность, представленная графом (рис. 2) транзакций программных блоков (Code-Flow Transaction Graph – CFTG), где $S = \{S_1, S_2, ..., S_i, ..., S_m\}$ – вершины или состояния программного продукта при моделировании тестовых сегментов. Иначе граф можно идентифицировать как ABC-граф – Assertion Based Coverage Graph. Каждое состояние $S_i = \{S_{i1}, S_{i2}, ..., S_{ij}, ..., S_{ip}\}$ определяется значениями существенных переменных проекта (булевы, регистровые переменные, память). Ориентированные дуги графа представлены совокупностью программных блоков

$$B = (B_1, B_2, ..., B_i, ..., B_n),\ \overset{n}{\underset{i=1}{\cup}} B_i = B;\ \overset{n}{\underset{i=1}{\cap}} B_i = \varnothing,$$

где каждому из них может быть поставлена в соответствие ассерция $A_i \in A = \{A_1, A_2, ..., A_i, ..., A_n\}$.



$$B = (B_1B_3B_9 \vee (B_2B_7 \vee B_1B_5)B_{11})B_{13} \vee ((B_1B_4 \vee B_2B_6)B_{10} \vee B_2B_8B_{12})B_{14} =$$
$$= B_1B_3B_9B_{13} \vee B_2B_7B_{11}B_{13} \vee B_1B_5B_{11}B_{13} \vee B_1B_4B_{10}B_{14} \vee B_2B_6B_{10}B_{14} \vee B_2B_8B_{12}B_{14}.$$

Рис. 2. Пример ABC-графа для HDL-кода

Каждая дуга B_i – группа операторов кода – формирует состояние вершины $S_i = f(T, B_i)$ в зависимости от теста $T = \{T_1, T_2, ..., T_i, ..., T_k\}$. Каждой вершине может быть поставлен ассерционный монитор, объединяющий ассерции входящих в вершину дуг $A(S_i) = A_{i1} \vee A_{i2} \vee ... \vee A_{ij} \vee ... \vee A_{in}$. Вершина может иметь более одной входящей (исходящей) дуги. Множество блоков с функциональными нарушениями дано списком $L = \{L_1, L_2, ..., L_i, ..., L_n\}$.

Модель HDL-кода, представленная в форме ABC-графа, отображает не только структуру программного кода, но и тестовые срезы функциональных покрытий, формируемые с помощью программных блоков, входящих в рассматриваемую вершину. Последняя определяет отношение между достигнутым на тесте пространством переменных и потенциально возможным, которое формирует функциональное покрытие как мощность состояния i-вершины графа $Q = \mathrm{card}C_i^r / \mathrm{card}C_i^p$. В совокупности все вершины графа должны составлять полное покрытие пространства состояний переменных программного кода, которое формирует качество теста, равное 1 (100%): $Q = \mathrm{card}\bigcup_{i=1}^{m} C_i^r / \mathrm{card}\bigcup_{i=1}^{m} C_i^p = 1$. Кроме того, механизм ассерций $< A, C >$, существующий на графе, позволяет выполнять мониторинг дуг (code-coverage) $A = \{A_1, A_2, ..., A_i, ..., A_n\}$ и вершин (functional coverage) $C = \{C_1, C_2, ..., C_i, ..., C_m\}$. Ассерции на дугах отвечают за диагностирование функциональных нарушений в программных блоках. Ассерции на вершинах графа дают дополнительную информацию о качестве тестов (ассерций) в целях их улучшения или дополнения. Транзакционный граф программного кода дает возможность: 1) использовать аппарат тестопригодного проектирования для оценки качества программного продукта; 2) оценить затраты на создание тестов, диагностирование и исправление функциональных нарушений; 3) оптимизировать синтез теста путем решения задачи покрытия минимальным множеством активизированных путей всех дуг (вершин). Например, минимальный тест для приведенного ABC-графа имеет шесть сегментов, которые активизируют все существующие пути:

$$T = S_0S_1S_3S_7S_9 \vee S_0S_1S_4S_8S_9 \vee S_0S_1S_5S_7S_9 \vee S_0S_2S_4S_8S_9 \vee S_0S_2S_5S_7S_9 \vee S_0S_2S_6S_8S_9.$$

Тесту можно поставить в соответствие следующую матрицу активизации программных блоков:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}
T_1	1	.	1	.	.	.	.	.	1	.	.	.	1	.
T_2	1	.	.	1	.	.	.	.	.	1	.	.	.	1
T_3	1	.	.	.	1	.	.	.	.	.	1	.	1	.
T_4	.	1	.	.	.	1	.	.	.	1	.	.	.	1
T_5	.	1	.	.	.	.	1	.	.	.	1	.	1	.
T_6	.	1	.	.	.	.	.	1	.	.	.	1	.	1

Матрица активизации иллюстрирует факт неразличимости на тесте функциональных нарушений в блоках 3 и 9, 8 и 12, которые составляют два класса эквивалентностей при наличии одной ассерции (монитора) в вершине 9. Для устранения такой неразличимости необходимо создать два дополнительных монитора в вершинах 3 и 6. В результате, 3 ассерции на вершинах $A = (A_3, A_6, A_9)$ дают возможность различить между собой все блоки программного кода. Таким образом, граф позволяет не только синтезировать оптимальный тест, но и определять минимальное число ассерционных мониторов здесь и далее в вершинах для поиска неисправных блоков с заданной глубиной диагностирования.

Увеличение числа ассерционных мониторов приводит к модификации таблицы активизации. Иначе, на заданном тесте и механизме ассерций необходимо однозначно решать задачу диагностирования функциональных нарушений программного кода с глубиной до программного блока. При этом число ассерций и тестовых сегментов должно быть мини-

мально допустимым для кодовой идентификации всех блоков: $|T|+|A| \geq \log_2|B| = \text{card}T + \text{card}A \geq \log_2 \text{card}B$. Первоначально количество мониторов-ассерций равно числу тестовых сегментов. Таблица активизации программных модулей позволяет выполнять идентификацию блоков кода с функциональными нарушениями на обобщенном векторе экспериментальной проверки (ассерционного мониторинга) $V = (V_1, V_2, ..., V_i, ..., V_n), V_i = \{0,1\}, V_i = T_i \oplus B_j, \forall j(B_{ij} = 1)$. Координата вектора $V_i = T_i \oplus B_j = 1$ идентифицирует факт «падения» тест-сегмента на подмножестве активизированных им блоков. В соответствии с вектором V, заданным на таблице активизации, с учетом приведенного выше правила вычисления его координат:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}	V
T_1	1	.	1	.	.	.	.	.	1	.	.	.	1	.	0
T_2	1	.	.	1	.	.	.	.	.	1	.	.	.	1	1
T_3	1	.	.	.	1	.	.	.	.	.	1	.	1	.	0
T_4	.	1	.	.	.	1	.	.	.	1	.	.	.	1	1
T_5	.	1	.	.	.	.	1	.	.	.	1	.	1	.	0
T_6	.	1	.	.	.	.	.	1	.	.	.	1	.	1	1

можно построить логическую функцию функциональных нарушений программного продукта, которая упрощается на основе использования координат вектора экспериментальной проверки V:

$$B = (\overline{T}_1 \vee B_1 \vee B_3 \vee B_9 \vee B_{13}) \wedge (\overline{T}_2 \vee B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (\overline{T}_3 \vee B_1 \vee B_5 \vee B_{11} \vee B_{13}) \wedge (\overline{T}_4 \vee B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge$$
$$\wedge (\overline{T}_5 \vee B_2 \vee B_7 \vee B_{11} \vee B_{13}) \wedge (\overline{T}_6 \vee B_2 \vee B_8 \vee B_{12} \vee B_{14});$$
$$\{V, T\} = (010101) \rightarrow$$
$$B = (0 \vee B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge (0 \vee B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge (0 \vee B_2 \vee B_8 \vee B_{12} \vee B_{14}) =$$
$$= (B_1 \vee B_4 \vee B_{10} \vee B_{14}) \wedge (B_2 \vee B_6 \vee B_{10} \vee B_{14}) \wedge (B_2 \vee B_8 \vee B_{12} \vee B_{14}) =$$
$$= B_1B_2 \vee B_4B_2 \vee ... \vee B_3B_6B_{12} \vee ... \vee B_{14}.$$

После преобразования конъюнктивной нормальной формы (КНФ) к дизъюнктивной нормальной форме (ДНФ) полученные термы содержат все возможные решения в виде покрытия единичных координат вектора экспериментальной проверки одиночными или кратными функциональными нарушениями программных блоков. Выбор лучшего решения связан с определением терма ДНФ минимальной длины. В данном примере оптимальным решением является терм, содержащий один блок $B = B_{14}$, который своим функциональным нарушением покрывает три единицы в векторе экспериментальной проверки $V = (010101)$. Данный факт также очевиден из сравнения двух последних столбцов матрицы активизации B.

Другое аппаратно ориентированное аналитическое решение связано с синтезом многовыходовой комбинационной схемы – дешифратора по матрице активизации программных блоков:

$$B_1 = T_1T_2T_3\overline{T}_4\overline{T}_5\overline{T}_6; B_2 = \overline{T}_1\overline{T}_2\overline{T}_3T_4T_5T_6; B_3 = \overline{T}_1T_2T_3T_4T_5T_6; \ ... \ B_{14} = \overline{T}_1T_2\overline{T}_3T_4\overline{T}_5T_6.$$

Такое устройство имеет число входов, равное количеству тестовых сегментов, а выходных линий, – равное числу программных блоков. При подаче на входы дешифратора вектора экспериментальной проверки формируется единичное значение на одном из его выходов. При этом номер выхода соответствует блоку, имеющему функциональные нарушения. Такая схема представляет интерес для создания замкнутой в цикл инфраструктуры верификации и исправления функциональных нарушений, где адрес неисправного блока может быть использован для его автоматической замены на альтернативный модуль из существующей библиотеки диверсных решений.

Определенный интерес представляет задача однозначной идентификации блока с нарушениями путем анализа матрицы активизации. Для рассматриваемого примера графа и соответствующей ему таблицы необходимо поставить три обобщенных монитора в вершинах 3, 6, 9 АВС-графа. Такое эвристическое решение требует изменения структуры матрицы активизации путем добавления разрядов в некоторых координатах таблицы B (механизме ассерций) и вектора экспериментальной проверки V:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}	V	A
T_1	1	.	11	.	.	.	.	.	01	.	.	.	1	.	00	3,9
T_2	1	.	.	1	.	.	.	.	.	1	.	.	.	1	1	9
T_3	1	.	.	.	1	.	.	.	.	.	1	.	1	.	0	9
T_4	.	1	.	.	.	1	.	.	.	1	.	.	.	1	1	9
T_5	.	1	.	.	.	.	1	.	.	.	1	.	1	.	0	9
T_6	.	1	.	.	.	.	.	11	.	.	.	01	.	1	01	6,9

В общем случае количество ассерций, различающих n функциональных блоков, соединенных последовательно, равно n-1. Для решения задачи мониторинга программного кода в целях достижения максимальной глубины поиска дефектных модулей можно использовать таблицу активизации, по которой достаточно просто установить классы эквивалентных программных блоков, которым соответствуют одинаковые столбцы. В каждом классе, имеющем мощность, равную n блоков, необходимо задать n-1 ассерционный монитор для всех блоков, исключая последний из них. При этом таблица активизации становится неоднородной по размерности числа битов в каждой координате, которая определяется минимальным числом мониторов для распознавания программных блоков, соединенных последовательно. Кроме того, в таблице активизации появляется столбец ассерционных мониторов, задающий последовательность вершин графа для снятия состояния ассерций в процессе моделирования. В случае более чем 50% существенности всех ассерционных мониторов для диагностирования большинства блоков целесообразно сделать таблицу активизации однородной на полном полученном множестве ассерций. В данном случае полученная структура превращается в таблицу функций неисправностей (ТФН) [2], где точки в координатах обозначают нулевые векторы ассерционных состояний на одном тест-сегменте:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}	B_{11}	B_{12}	B_{13}	B_{14}	V	A
T_1	101	.	101	.	.	.	.	.	001	.	.	.	001	.	000	3,6,9
T_2	001	.	.	001	.	.	.	.	.	001	.	.	.	001	001	3,6,9
T_3	001	.	.	.	001	.	.	.	.	.	001	.	001	.	000	3,6,9
T_4	.	001	.	.	.	001	.	.	.	001	.	.	.	001	001	3,6,9
T_5	.	001	.	.	.	.	001	.	.	.	001	.	001	.	000	3,6,9
T_6	.	101	.	.	.	.	.	010	.	.	.	001	.	001	001	3,6,9

Она предоставляет полную информацию о поведении программного продукта на тестовых сегментах, где реакции создаются с помощью минимального множества ассерционных мониторов, достаточного для распознавания любого программного модуля с функциональными нарушениями. Процесс-модель диагностирования сводится к одной итерации сравнения вектора экспериментальной проверки со столбцами матрицы активизации или ТФН:

$$L = L \bigvee_{j=1}^{n} B_j \leftarrow \sum_{i=\overline{1,k}}^{r=\overline{1,m}} \left(B_{ijr} \bigoplus_{i=\overline{1,k}}^{r=\overline{1,m}} V_{ir}\right) = (0 \vee \min).$$

В данном примере использование приведенного выше критерия приводит к следующему результату: $L = B_{14} \leftarrow \forall i, r(B_{i,14,r} \oplus V_{i,r} = 0)$. Сигналы несовпадения, равные 1, отсутствуют при выполнении хог-операции между вектором V и столбцом 14 матрицы активизации. Таким образом, если тест уже имеется, то для повышения глубины диагностирования

остается единственный путь, связанный с увеличением количества ассерционных мониторов, которое повышает размерность матрицы активизации до таблицы функций неисправностей в соответствии с выражением: $Q = |T| \times |B| \times |A| = \mathrm{card}T \times \mathrm{card}B \times \mathrm{card}A$. Поэтому введение каждой новой ассерции должно иметь весомые аргументы, связанные с повышением глубины поиска дефектов, поскольку добавление каждой новой ассерции увеличивает размерность ТФН в два раза.

Таким образом, граф $F = (A * B) \times S$ выгодно отличается от существующих аналогов (количество вершин равно числу переменных, дуга соответствует одной команде или множеству, но неупорядоченных в соответствии с последовательностью операторов программного блока) [4, 22-25]: 1) Имеет минимальное число дуг, равное количеству линейных блоков программного продукта. 2) Каждой дуге (вершине) ставится в соответствие ассерция, отвечающая за правильное функционирование блока (нескольких блоков). 3) Вершина, в которую входят дуги, представляет собой минимальную совокупность только тех переменных, которые модифицируются программными блоками, входящими в нее. Такие переменные легко могут быть получены путем анализа операторов программных блоков. 4) Поскольку граф есть макроструктура программного продукта, он может быть достаточно просто построен автоматически. Следовательно, имеется возможность создавать инфраструктуру (место и вид ассерций, матрицы активизации блоков и состояния ассерций) верификации, диагностирования и восстановления работоспособности программного продукта с минимальным участием человека. 5) Наличие в графе не реального, а модельного времени существенно уменьшает размерность матриц для верификации и диагностирования функциональных нарушений. 6) Наличие структурной модели программного продукта дает возможность применить алгоритмические методы синтеза тестов активизации всех вершин и дуг графа. 7) Программный блок аргумент, а вершина есть производная от него. Такая зависимость определяет линейную вычислительную сложность для синтеза графа в целях верификации программного продукта. В упомянутых выше аналогах аргументом является вершина, что приводит к квадратичной сложности задачи синтеза графа при анализе кода.

Что касается качества модели диагностирования функциональных нарушений, то она показывает эффективность использования пары (тест, ассерции) для заданной глубины диагностирования. Оценка качества модели функционально зависит от длины теста $|T|$, числа ассерций наблюдения $|A|$, количества распознаваемых блоков с функциональными нарушениями N_d на общем числе программных блоков N:

$$Q = E \times D = \frac{]\log_2 N[}{|T| \times |A|} \times \frac{N_d}{N}.$$

Эффективность диагностирования есть отношение минимального числа двоичных разрядов, необходимых для идентификации (распознавания) всех блоков к реальному количеству разрядов кода, представленному произведением длины теста на число ассерций в каждом из них. Если первая дробь оценки равна 1 и все блоки с ФН распознаются ($N_d = N$), то тест и ассерции оптимальны, что доставляет критерию качества модели диагностирования значения, равного 1. Для примера матрицы АВС-графа, представленного на рис. 2, существует 2 решения (с одной и тремя ассерциями):

$$Q_1 = \frac{]\log_2 14[}{|6| \times |1|} \times \frac{10}{14} = 0{,}5;$$
$$Q_2 = \frac{]\log_2 14[}{|6| \times |3|} \times \frac{14}{14} = 0{,}2.$$

Несмотря на то, что качество модели лучше в первом варианте из-за меньшего объема таблицы активизации, вторая модель – более предпочтительна, поскольку она имеет максимальную глубину диагностирования, когда все 14 блоков распознаются за счет добавления двух ассерций. Оценка позволяет определить минимальные затраты по

длине теста и числу ассерций для создания модели с максимальной глубиной диагностирования.

Интерес представляет оценивание качества структуры кода проекта с позиции диагностируемости (diagnosability) блоков программного кода. Цель анализа – определить количественную оценку структуры графа и места (вершины) для установки ассерционных мониторов, создающих максимальную глубину диагностирования функциональных нарушений программных блоков. Здесь важна не управляемость и наблюдаемость, как в тестопригодности (testability), а различимость программных блоков с функциональными нарушениями, в пределе представляющая ноль блоков с эквивалентными (неразличимыми) нарушениями. Такая оценка может быть полезной для сравнения графов, реализующих одинаковую функциональность. Здесь необходимо оценивать структуру графа с позиции потенциально заложенной в нем глубины поиска функциональных нарушений программного продукта. Возможным вариантом может быть диагнозопригодность ABC-графа как функция, зависящая от таких смежных дуг при каждой вершине (формирующих число N_n), одна из которых – входящая, другая – исходящая. Такие дуги составляют пути без схождений и разветвлений (N – общее количество дуг в графе):

$$D = \frac{N - N_n}{N}.$$

Каждая вершина, объединяющая 2 дуги, которые входят в число N_n, называется транзитной. Оценка N_n есть число неразличимых функциональных нарушений (ФН) программных блоков. Места потенциальной установки мониторов для различения ФН – транзитные вершины. С учетом приведенной оценки диагнозопригодности D качество модели диагностирования программного продукта принимает вид:

$$Q = E \times D = \frac{]\log_2 N[}{|T| \times |A|} \times \frac{N - N_n}{N}.$$

Правила синтеза диагнозопригодных программных продуктов: 1) Тест (testbench) должен создавать минимальное число одномерных путей активизации, покрывающих все вершины и дуги ABC-графа. 2) Базовое число мониторов-ассерций равно количеству конечных вершин графа, не имеющих исходящих дуг. 3) В каждой вершине, имеющей одну входную и одну выходную дугу, может быть размещен дополнительный монитор. 4) Параллельно независимые блоки кода имеют n мониторов и один тест или один объединенный монитор и n тестов. 5) Последовательно соединенные блоки имеют 1 тест активизации последовательного пути и n-1 монитор или n тестов и n мониторов. 6) Вершины графа, имеющие различное число входных и выходных дуг, создают условия для диагностируемости данного участка за счет одномерных тестов активизации без установки дополнительных мониторов. 7) Совокупность тестовых сегментов (testbench) должна составлять 100% покрытие функциональных режимов (functional coverage), заданных вершинами ABC-графа. 8) Функция диагнозопригодности прямо пропорциональна длине теста, числу ассерций и обратно пропорциональна двоичному логарифму от числа программных блоков:

$$D = \frac{N - N_n}{N} = f(T, A, N) = \frac{|T| \times |A|}{]\log_2 N[}.$$

Диагнозопригодность как функция, зависящая от структуры графа (программного продукта), теста и ассерционных мониторов, всегда может быть приведена к единичному значению. Для этого существует два альтернативных пути. Первый – увеличение тестовых сегментов, активизирующих новые пути, для различения эквивалентных неисправностей (ФН) без наращивания ассерций, если структура графа программных блоков имеет такой потенциал связей. Второй – размещение дополнительных ассерционных мониторов в транзитных вершинах графа. Возможен и третий, гибридный вариант, основанный на совместном применении двух перечисленных выше путей. Отношение трех компонентов (число программных блоков, мощность механизма ассерций и длина теста) при единичном значении качества модели диагностирования и диагнозопригодности формирует плоскость

оптимальных решений $D = 1 \to \frac{|T| \times |A|}{]\log_2 N[} = 1 \to]\log_2 N[= |T| \times |A|$. Она может быть полезной для выбора квазиоптимального варианта альтернативного пути достижения полной различимости на паре $|T| \times |A|$ функциональных нарушений программных блоков.

Дискретная производная как практически полезное отношение. Дискретная производная есть симметрическая разность $A \Delta B = C$ двух объектов или явлений, где координаты их параметров заданы одинаковым булеаном на универсуме примитивов.

Булева производная есть булева разность $A \oplus B = C$ двух объектов или явлений, где координаты их параметров заданы двоичным алфавитом.

Здесь существенно, что производная есть отношение, значит – объектов или явлений должно быть два. Они должны быть описаны в одинаковых алфавитах, образующих (составляющих) булеан.

Под данные определения попадают отношения двух объектов, двух компонентов объекта, двух компонентов различных объектов, компонента и объекта:

$$A \oplus B = C;$$

$$a_i \oplus a_j = a_{ij};$$

$$a_i \oplus b_i = c_i;$$

$$a_i \oplus A = A_i.$$

Здесь следует договориться, что xor-операция будет означать и симметрическую разность, если мощность алфавита описания переменных больше двух. Размерность xor-отношения может быть как форматом целого, так и его части. Она зависит от функции цели – увидеть различие в целом – формат результата максимальный или определить степень принадлежности части к целому – формат результата минимальный. Дискретная (булева) производная инвариантна по отношению к форме (аналитическая, табличная или графовая) описания процесса или явления. Цель производной – найти различия (расстояние) в объектах или явлениях (далее для уменьшения количества слов – в объектах) путем xor-сравнения. Для определения расстояния используется бета-метрика, которая вычисляет взаимодействие любого конечного числа (n = 1,2,3, …) объектов в киберпространстве, замкнутых в цикл:

$$\beta = \bigoplus_{i=1}^{n} d_i = 0.$$

Производные на аналитической форме описания (логических) объектов достаточно исследованы и представлены многочисленными публикациями. Относительно разностного анализа графовых структур здесь область исследования представлена специфическими моделями, ориентированными на решение оптимизационных задач [26]. Производная по структурным компонентам графа дает возможность решать задачи: 1) распознавания графов и его фрагментов; 2) определения путей между двумя вершинами; 3) покрытия всех вершин и дуг минимальным множеством путей; 4) диагностирования технического состояния объектов; 5) преобразования графовых структур.

3. Метод векторно-логического анализа столбцов

Методы поиска функциональных нарушений в блоках операторов кода используют предварительно построенную таблицу ФН $B = [B_{ij}]$, где строка есть отношение между тестовым сегментом и подмножеством активизированных на данном сегменте программных блоков $T_i \approx (B_{i1}, B_{i2}, ..., B_{ij}, ..., B_{in})$. Столбец таблицы формирует отношение между программным блоком и тестовыми сегментами $B_j \approx (T_{1j}, T_{2j}, ..., T_{ij}, ..., T_{pj})$, которые активизируют его. Иначе, столбец есть вектор ассерций, идентифицирующий функциональное нарушение в соответствующем блоке. На стадии моделирования определяется реакция

$m = (m_1, m_2, ..., m_i, ..., m_p)$ механизма ассерций на тест путем формирования каждого разряда $m_i = (A_1 \vee A_2 \vee ... \vee A_i \vee ... \vee A_k)$, $A_i = \{0,1\}$ как реакции ассерций на тест-сегмент T_i. Поиск ФН основан на определении xor-операции между вектором состояния ассерций и столбцов таблицы ФН $m \oplus (B_1 \vee B_2 \vee ... \vee B_j \vee ... \vee B_n)$. Выбор решения определяется вектором B_j с минимальным числом единичных координат, формирующих программные блоки с ФН, проверяемые на тестовых сегментах. Процесс диагностирования по таблице ФН на основе реакции $m = (m_1, m_2, ..., m_i, ..., m_n), m_i = \{0,1\}$ на тест сводится к методам векторно-логического анализа столбцов или строк.

Первый метод основан на применении векторной xor-операции между m-реакцией функциональности на тест, формально рассматриваемой в качестве входного вектор-столбца, и столбцов таблицы неисправностей $m \oplus (B_1 \vee B_2 \vee ... \vee B_j \vee ... \vee B_m)$. Для подсчета качества взаимодействия векторов $Q_j(m \oplus B_j)$ в целях выбора лучшего решения определяются столбцы с минимальным числом единиц результирующего вектора. Они идентифицируют и формируют дефектные блоки с функциональными нарушениями, проверяемые на тестовых наборах. Аналитическая модель процесса получения решения в виде списка блоков с ФН, присутствующих в программном продукте, представлена в следующем виде:

$$L = L \underset{j=1}{\overset{n}{\vee}} B_j \leftarrow \sum_{i=1}^{k} (B_{ij} \overset{k}{\underset{i=1}{\oplus}} m_i) = (0 \vee \min). \quad (4)$$

Здесь фигурирует вектор экспериментальной проверки, который является входным для последующего анализа таблицы ФН:

$$m = f(A,B) \oplus f^*(A,B,L) \quad (5)$$

есть результат проведения тестового эксперимента – сравнение функционалов (состояний выходов) эталонного $f(A,B)$ и реального $f^*(A,B,L)$ устройства с дефектами L на тестовых наборах A. Во втором случае, если множество дефектных блоков $L > 1$, это означает наличие эквивалентных на данном тесте и механизме ассерций, функциональных нарушений.

Процесс-модель поиска оценки лучшего решения с минимальным числом единичных координат из не менее, чем двух альтернатив, представлена на рис. 3 и имеет следующие операции. 1) Первоначально в вектор-результат Q, в котором будет сохранено лучшее решение, заносятся единичные значения во все координаты (худшее решение) и одновременно осуществляется операция slc сдвига влево с уплотнением единиц текущего вектора Q_i. 2) Выполняется сравнение двух векторов: Q и очередной оценки Q_i из списка решений. 3) Реализуется векторная операция and $(Q \wedge Q_i)$, результат которой сравнивается с содержимым вектора Q, что дает возможность изменить его, если вектор Q_i имеет меньшее число единичных значений. 4) Процедура поиска оценки лучшего решения повторяется n раз:

$$Q = Q\overline{(\vee((Q \wedge Q_i) \oplus Q))} \vee Q_i(\vee((Q \wedge Q_i) \oplus Q));$$
$$Y = \vee((Q \wedge Q_i) \oplus Q);$$
$$Q = Q\overline{Y} \vee Q_i Y.$$

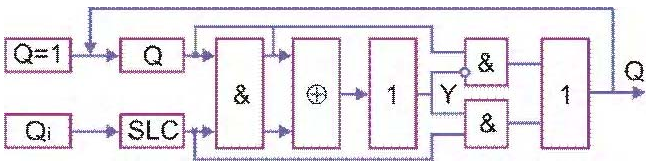
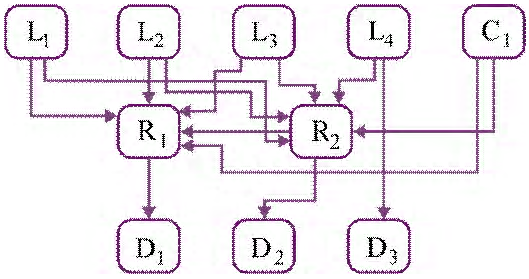


Рис. 3. Процесс-модель выбора решения

Достоинство метода векторно-логического анализа столбцов – выбор лучшего решения из всех возможных одиночных и кратных ФН. По существу, в список дефектов включают-

ся такие одиночные ФН, которые при логическом умножении на вектор экспериментальной проверки дают результат в виде вектор-столбца. Дизъюнкция всех столбцов, составляющих решение, равна вектору экспериментальной проверки $\bigvee_{j=1}^{r}(B_j \in B) = m$. Далее рассматривается пример анализа таблицы ФН блока Row_buffer, представленного на рис. 4.



Test	B_1 L_1	B_2 L_2	B_3 L_3	B_4 L_4	B_5 C_1	B_6 R_1	B_7 R_2	B_8 D_1	B_9 D_2	B_{10} D_3	m_1	m_2
T_1	1	0	0	0	0	1	0	1	0	1	0	1
T_2	0	1	0	0	0	0	1	0	1	0	1	1
T_3	0	0	1	0	0	1	1	1	0	1	0	1
T_4	0	0	0	1	0	0	1	0	1	0	1	0
T_5	0	0	0	0	1	1	1	1	0	1	0	0
T_6	1	0	0	0	0	0	1	0	1	0	1	1
T_7	0	1	0	0	0	1	0	1	0	0	0	1
T_8	0	0	1	0	0	1	0	1	0	0	0	1
T_9	0	0	1	0	0	0	1	0	1	0	1	0
T_{10}	0	0	0	1	0	1	1	1	0	0	0	0
T_{11}	0	0	0	1	0	0	0	0	0	1	0	0
T_{12}	0	0	0	0	1	0	1	0	1	0	1	0
T_{13}	0	0	0	0	1	1	1	1	0	0	0	0

Рис. 4. Row_buffer транзакционный граф и таблица ФН

На основе процедуры диагностирования (4) и таблицы ФН (см. рис. 3) можно определить дефектные компоненты методом анализа столбцов таблицы ФН. Здесь векторы m_1, m_2 формируют результаты диагностического эксперимента, выполненные по процедуре (5). Результат диагностирования одиночных и кратных ФН имеет следующий вид:

$$L^s(m_1) = m_1 \wedge (\bigvee_{j=1}^{10} B_j) = B_9 \to D_2;\ L^m(m_2) = m_2 \wedge (\bigvee_{j=1}^{10} B_j) = B_1 \vee B_2 \to L_1 \vee L_2;$$

$$Q(m_1, D_2) = 1;\ Q[m_2, (L_1 \vee L_2)] = \frac{1}{3}(\frac{4}{13} + \frac{1}{4} + 1) = 0{,}52.$$

В первом случае диагноз определен в виде одного дефектного блока D_2, присутствующего в транзакционном графе, качество решения равно 1. Во втором случае процедура диагностирования выявила наличие двух дефектных модулей $L_1 \vee L_2$, которые не смогли сформировать идеальную оценку качества. Тем не менее, решение является лучшим среди всех возможных, которое максимально приближено к вектору экспериментальной проверки по критерию принадлежности $Q[m_2, (L_1 \vee L_2)]$. Вычислительная сложность метода анализа столбцов определяется следующей зависимостью:

$$Z^c = 3n^2 + n^2 = 4n^2;\ Z^r = 3n + n = 4n.$$

Здесь первая оценка учитывает выполнение координатных операций над матрицей, размерностью $n \times n$. Вторая оценка определяет вычислительную сложность регистровых параллельных операций для подсчета критериев качества и обработки матрицы соответственно.

4. Метод векторно-логического анализа строк

Метод предназначен для определения места дефектов или ФН программного кода и состоит из двух процедур: 1) вычисление логического произведения конъюнкции строк, отмеченных единичными значениями вектора $T_i(m_i = 1)$, на отрицание дизъюнкции нулевых строк $T_i(m_i = 0)$ для одиночных дефектных блоков; 2) вычисление логического произведения дизъюнкции единичных строк на отрицание дизъюнкции нулевых строк для кратных дефектных блоков:

$$L^s = (\bigwedge_{\forall m_i = 1} T_i) \wedge (\overline{\bigvee_{\forall m_i = 0} T_i});$$
$$L^m = (\bigvee_{\forall m_i = 1} T_i) \wedge (\overline{\bigvee_{\forall m_i = 0} T_i}). \qquad (6)$$

Формулы интересны тем, что они не привязаны к критериям качества диагностирования, а оперируют лишь двумя компонентами, таблицей ФН и вектором экспериментальной проверки. Выполнение процедуры диагностирования по формулам (4) для вектора экспериментальной проверки $m_1 = (0101010010010)$, заданного в последней таблице ФН, дает результат: $L^s(m_1, T) = D_2$, который не хуже, чем ранее полученный методом анализа столбцов. Для вектора экспериментальной проверки $m_2 = (1110011100000)$ результат диагностирования имеет вид: $L^m(m_2, T) = L_1 \vee L_2$. Вычислительная сложность метода анализа строк определяется следующей зависимостью: $Z^c = n^2;\ Z^r = n.$ Первая оценка предназначена для подсчета числа координатных операций, вторая определяет вычислительную сложность процесса обработки на основе регистровых параллельных операций. Предложенные методы диагностирования ФН для программных и аппаратных продуктов есть один из наиболее существенных компонентов инфраструктуры сервисного обслуживания проектируемых изделий.

Формулы (6) могут быть модифицированы, если ввести следующие обозначения:

$$a = (\bigwedge_{\forall m_i = 1} T_i);\ b = (\bigvee_{\forall m_i = 0} T_i);\ c = (\bigvee_{\forall m_i = 1} T_i);$$
$$L^s = a\bar{b} = a \oplus ab = a(a \oplus b) = a(b \oplus 1);$$
$$L^m = c\bar{b} = c \oplus cb = c(c \oplus b) = c(b \oplus 1);$$
$$L = \begin{cases} a\bar{b} = a\bar{b} = a \oplus ab = a(a \oplus b) = a(b \oplus 1); \\ c\bar{b} = c\bar{b} = c \oplus cb = c(c \oplus b) = c(b \oplus 1) \leftarrow a\bar{b} = 0. \end{cases}$$

Любое выражение в правой части уравнений может быть использовано для определения функционального нарушения в программном или аппаратном изделии. Различие заключается в наличии или отсутствии инверсии, заменяемой xor-операцией, которая для задач диагностирования и распознавания образов зачастую является более предпочтительной. В таком случае процесс-модель диагностирования одиночных (используется a-компонент) или кратных (b-компонент) дефектов (функциональных нарушений) на основе анализа таблицы ФН будет иметь эффективную векторно-ориентированную вычислительную технологию $L = (b \oplus 1)(a \vee c)$ встроенного сервисного обслуживания программных и/или аппаратных продуктов. С позиции теории множеств это означает определение результата теоретико-множественного вычитания $L = (a \vee c) \setminus b = (a \setminus b) \vee (c \setminus b)$ в алгебрологическом векторном пространстве. Для таких операций необходим мультиматричный процессор,

строго ориентированный на параллельное выполнение нескольких логических операций над матрицами данных.

5. Матричный метод поиска функциональных нарушений в программных блоках

Метод диагностирования функциональных нарушений в программных блоках в дополнение к графу транзакций программных блоков (3) использует триаду матриц одного формата:

$$M = B \oplus A \oplus L = 0,\ L = B \oplus A \leftarrow L_{ij} = B_{ij} \oplus A_{ij} \leftarrow \{B_{ij}, A_{ij}, L_{ij}\} = \{0,1\};$$

$$B = [B_{ij}],\ A = [A_{ij}],\ L = [L_{ij}],\ i = \overline{1,n};\ j = \overline{1,m};\ \oplus = a\bar{b} \vee \bar{a}b.$$

Здесь матрицы формируют: B – активизацию блоков на тестовых сегментах в процессе моделирования; A – активность ассерций, соответствующих блокам, на тестовых сегментах также в процессе моделирования; L – дефектные блоки, полученные в результате выполнения xor-операции над двумя предыдущими матрицами. Процедура покоординатного анализа матриц использует двоичную xor-операцию, например:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	.	.	.	.	.	.	.	.
T_2	.	.	.	.	.	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	.	.	.	.	.	.	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	.	.	.	.	.	.

Полученный результат $L = B \oplus A$ в виде L-матрицы $[L_{ij}] = (T \times B \times \{0,1\})$, все координаты которой равны нулю, свидетельствует об отсутствии функциональных нарушений в программном продукте относительно предложенного плана верификации в формате (тест – функциональные блоки – активизация $[B_{ij}] = (T \times B \times \{0,1\})$, тест – ассерции – реакция $[A_{ij}] = (T \times A \times \{0,1\})$. Другой модельный эксперимент свидетельствует о наличии в программном коде функциональных нарушений $L = \{B_1, B_2, B_3, B_5, B_6\}$:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	1	.	.	1	.	.	1	.
T_2	.	0	0	.	0	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	0	.	0	.	.	0	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	0	1	1	.	.	.
$\vee A_i$	1	1	0	1	1	1	1	1

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.
$\vee L_i$	1	1	1	0	1	1	0	0

Здесь представлены результаты векторных операций над всеми строками двух таблиц $\vee L_i$=11101100 и $\vee A_i$=11011111. Их логическое умножение с предварительной инверсией первого вектора дает координаты блоков с функциональными нарушениями, отмеченные единицами. В данном примере вектор формирует только один блок (00100000)&(11101100)=(00100000). Что является основанием для уменьшения дефектных блоков? Если предположить, что проверка первого блока в соответствии с верификационным планом должна сопровождаться выявлением дефектов на первом и шестом тестах, что не выполняется, то блок 1 можно исключить из списка неисправных. Аналогично можно поступить и с модулями 2, 5, 6. Тогда скорректированный результат будет иметь только один блок с функциональными нарушениями: $L = \{B_3\}$. Процедура уточнения результата диагностирования может быть также формализована к следующему виду: $L = B_j \leftarrow B_j \wedge L_j = B_j, j = \overline{1,m}$. Если результат сравнения отрицательный $B_j \oplus L_j = 0$, то

данный факт свидетельствует как о некорректности программного кода, так и об ассерционной или тестовой несостоятельности, включая функциональное покрытие. Для диагностирования программного кода в соответствии с процесс-моделью, имеющей вид

$$L(B,T) = (B \oplus A) \rightarrow L(B) = (\overline{\underset{i=1,n}{\vee} A_i}) \wedge (\underset{i=1,n}{\vee} L_i),$$

необходимо рассмотреть следующие пункты:

1. Покрытие (coverage) – любая метрика выбора теста и определения его полноты. Покрытие кода – code coverage – метрика теста, ориентированная на гарантированное подтверждение выполнения всех строк кода. Выполняется декомпозиция программного кода на блоки $B = \{B^s, B^t\} \leftarrow B^s \cap B^t = \varnothing, B^s \cup B^t = B$. Каждый блок принадлежит к одному из двух типов: последовательность операторов без ветвления или цепь временной задержки $B_i \in \{B^s, B^t\}$. Выполняется установка ассерционных мониторов активности блоков на тесте в начале ветвления или в первом такте цепи временной задержки. В процессе моделирования ассерции формируют матрицу активизации программных блоков на каждом тест-сегменте $B_{ij} = T_i \oplus B_j \in \{0,1\}$. Если блок активен (ассерция выполнена) на тесте (testbench), значение координаты матрицы равно 1, в противном случае – $B_{ij} = 0$. Testbench – входные условия для тестирования HDL-кода и соответствующие им выходные реакции, задающие преобразования в функциональном подпространстве проверяемого устройства.

2. Функциональное покрытие – functional coverage – метрика теста, гарантирующая достижение всех существенных состояний в пространстве определения переменных и функций программного продукта. Выполняется декомпозиция функциональности программного продукта на графы: управляющий и транзакционный $F = \{F^c, F^t\} \leftarrow F^c \cap F^t = \varnothing, F^c \cup F^t = F$. Это дает возможность существенно уменьшить размерность задач, связанных с построением соответствующих покрытий, задающих область определения переменных управления и потоков данных. Создание теста и последующая верификация (coverage driven verification) используют упомянутые графы с ограничениями (constraints), взятыми из спецификации. Синтезируемый тест для графа управления обеспечивает активизацию всех логических и арифметических переменных, участвующих в инициировании транзакций программного продукта. Способ активизации переменных или синтез теста: псевдослучайная или детерминированная (алгоритмическая) генерация тестовых воздействия, а также ручное написание входных стимулов. Форма задания покрытия: сокращенная таблица истинности, булево уравнение, двоичная диаграмма решений (binary decision diagram), граф-схема алгоритма. Тест для второго графа оперирует потоками данных, которые на системном уровне не всегда следует проверять ввиду отсутствия дефектов типа коротких замыканий между переменными или константными неисправностями в них. Граф транзакций может быть использован для создания верификационного плана существенных интерфейсных параметров программного продукта. Для этого необходимо использовать интерфейсные ассерции, оперирующие глобальными переменными.

3. Матрица ассерций программных блоков имеет вид, аналогичный структуре активизации блоков $A = [A_{ij}]$. Здесь формат ассерции как логического высказывания, использующего существенные переменные программного блока $f(X) = A_{ij} = \{0,1\}$, отвечает за функционирование соответствующего активизированного на тесте модуля $B_{ij} = 1$. Высказываний в блоке может быть несколько, раздельных для повышения глубины диагностирования или объединенных по функции or. В данном (последнем) случае ассерция несет ответственность за исправное функционирование блока. Ассерция имеет два значения: 1 – блок работает исправно, 0 – существуют функциональные нарушения. Ассерции представлены двумя уровнями иерархии: интерфейсные и блоковые $A = \{A^i, A^b\}$. Первые ориентированы на проверку существенных параметров спецификации, общих для программного продукта, которые являются внешними по отношению к последнему. Вторые встраиваются в программный блок, который не имеет разветвлений. Мощность команд или строк кода – не более 20 – определяется числом операторов, размещаемых на экране. Такой блок может включать операторы задержки по времени или событию.

6. Имплементация моделей и методов в систему верификации

Практическая реализация моделей и методов верификации была интегрирована в среду моделирования Riviera компании Aldec (рис. 5). Введенные в систему модули ассерций и диагностирования усовершенствовали существующий процесс верификации, что дало возможность на 15% уменьшить общее время проектирования цифрового изделия.

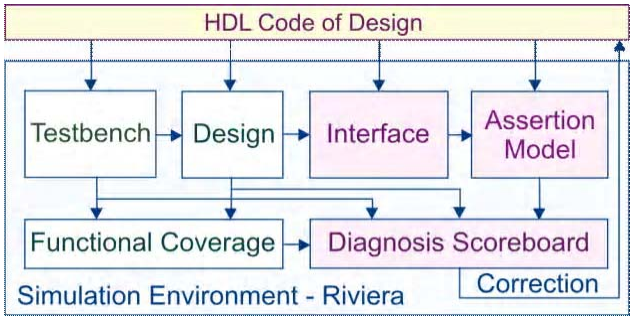


Рис. 5. Интеграция разработок в систему Riviera

Фактически применение ассерций дает возможность уменьшить объем testbench кода, а значит, существенно (x3) сократить время ручного проектирования (рис. 6), которое является наиболее дорогостоящим компонентом. Кроме того, механизм ассерций позволяет также повысить глубину диагностирования функциональных нарушений программных блоков до уровня 10-20 операторов HDL-кода.

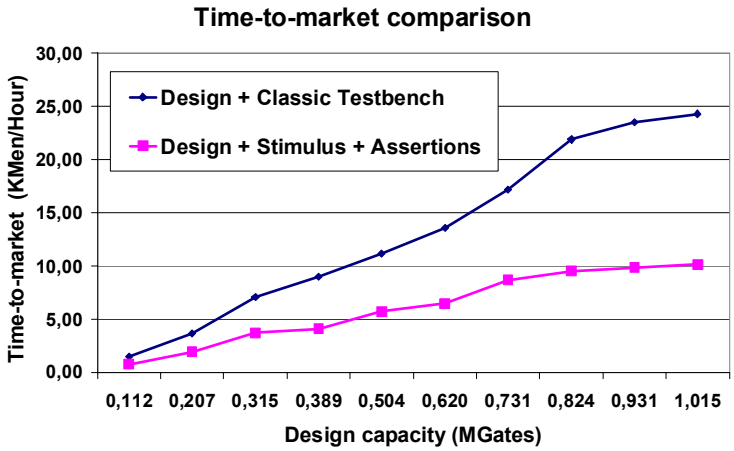


Рис. 6. Сравнительный анализ методов верификации

Благодаря взаимодействию средств моделирования и механизма ассерций, автоматически размещаемого внутри HDL-кода, появилась возможность доступа средств диагностирования к значениям всех внутренних сигналов. Это дает возможность оперативно идентифицировать место и вид функционального нарушения, а также уменьшить время обнаружения ошибок в процессе эволюции изделия при нисходящем проектировании. На основе использования практики применения ассерций для 50 реальных проектов (от 5 тыс. до 5 млн вентилей) наработаны сотни специальных решений, включенных в библиотеку верификации VTL (Verification Template Library), которая обобщает наиболее востребованные на рынке EDA (Electronic Design Automation) темпоральные ограничения при верификации широкого класса цифровых изделий. Программная реализация предложенной системы анализа ассерций и диагностирования HDL-кода входит в состав многофункциональной интегрированной среды Aldec Riviera для моделирования и верификации HDL-моделей. Высокая производительность и технологичное сочетание системы анализа ассерций с HDL-симулятором фирмы Aldec во многом обеспечивается за счет интеграции с внутренними компонентами симулятора, включая компиляторы HDL-языков. Обработка результатов работы системы анализа ассерций обеспечивается в среде Riviera представительным набором визуальных инструментов, облегчающих диагностирование и устранение

функциональных нарушений. Модель анализа ассерций также может быть реализована аппаратным способом с некоторыми ограничениями на подмножество поддерживаемых языковых конструкций. Продукт Riviera с компонентами ассерционной темпоральной верификации, позволяющими на 3-5% повысить качество проектов, в настоящее время занимает лидирующие позиции на мировом рынке электронных технологий, с числом инсталляций 5 000 в год, в 200 компаниях и университетах более чем 20 стран планеты.

7. Мультиматричный процессор бинарных операций и инфраструктура верификации

Для реализации эффективных с позиции времени и затрат вычислительных процессов, связанных с диагностированием функциональных нарушений, необходим простой по архитектуре процессор с минимальной системой команд, где в качестве операндов выступают не только булевы переменные, но и более сложные структуры, такие как регистры и матрицы. Такой процессор должен выполнять в параллельном режиме операции над всеми разрядами регулярных операндов, не требуя специальных компиляторов распараллеливания вычислительных процессов.

Мультиматричный процессор (ММП) есть такая минимальная архитектура инструкций-примитивов, где каждый из них ориентирован на параллельное выполнение только одной (and, or, xor, slc) операции над соответствующей матрицей (двумерный массив данных). Количество командно-ориентированных матриц-примитивов создает систему – гетерогенный мультиматричный процессор бинарных операций с буфером М (рис. 7).

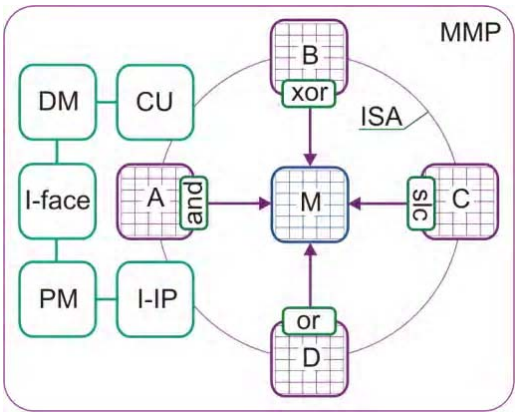


Рис. 7. Мультиматричный процессор бинарных операций

Здесь представлены стандартные блоки: памяти данных DM и программ PM, управления CU, интерфейс I-face и сервисного обслуживания I-IP, а также мультиматричный модуль процессора, включающий 4 блока памяти со встроенными в них операциями (A – and, B – xor, C – slc, D – or – shift left crowding) и буферную память M. Мультиматричный процессор (MMP) ориентирован на параллельное выполнение в данном случае одной из четырех инструкций (ISA – Instruction Set Architecture), оперирующей матрицами двоичных данных одинаковой размерности: $M = M\{and, or, xor, slc\}\{A, B, C, D\}$ с занесением результата в буфер M. Особенность ММП в том, что не ячейка матрицы имеет систему команд из четырех операций, а каждая команда имеет собственную матрицу ячеек в качестве данных для параллельной обработки, что существенно упрощает структуру управления и устройства в целом. Вся сложность ММП перенесена на структуры данных, где память матрицы имеет одну аппаратно-реализованную встроенную команду, что позволяет иметь примитивную систему управления параллельными вычислительными процессами (SIMD – Single Instruction Multiple Data), последовательностную по своей сути, а значит, нет необходимости создавать сверхсложные компиляторы, ориентированные на распараллеливание вычислительных процессов. Представленная архитектура ММП адаптируется к выполнению логических операций над операндами регистрового уровня. Прототип ММП интегрирован в плату аппаратного ускорения процессов моделирования и верификации HES™ компании Aldec.

На основе мультиматричного (-регистрового) процессора создана инфраструктура (рис. 8) верификации HDL-кода проектируемых цифровых систем на кристаллах, которая является модификацией I-IP стандарта 1500 [3, 4, 11, 14]. Здесь фигурируют четыре процесс-модели: тестирование на стадии моделирования, диагностирование функциональных нарушений, оптимизация диагноза, восстановление работоспособности.

1. Процесс-модель тестирования включает HDL-модель, механизм ассерций, testbench и coverage (покрытие). Последнее оценивает качество теста проверки всех состояний проекта. В результате моделирования синтезируется матрица активизации программных блоков B и матрица ассерционных реакций A на тестовые сегменты, которая может быть трансформирована к вектору состояния ассерций m путем применения функции or к вектор-столбцам A-матрицы:

$$\begin{cases} B = (T \oplus F); \\ m = \bigvee\limits_{j=1}^{m} A_j \leftarrow A = (T \oplus A^c). \end{cases}$$

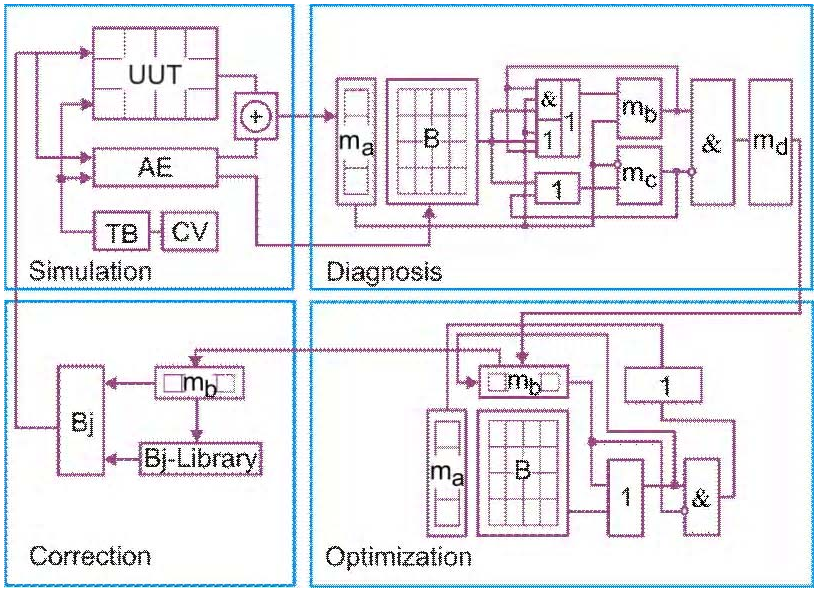


Рис. 8. Инфраструктура верификации HDL-кода

2. Два последних компонента используются во второй процесс-модели для диагностирования блоков HDL-кода. Результатом диагностирования является вектор дефектов, формирующий подмножество блоков m_d с функциональными нарушениями. При этом не исключены ошибки как в testbench, так и в ассерционных операторах, отвечающих за тестирование и мониторинг программных блоков. Тройственная неопределенность диагноза $D = \{B_j, T_i, A_{ij}\}$ характерна при отсутствии точной идентификации блока в процедуре сравнения столбцов матрицы активизации с вектором ассерционных реакций.

3. Третий блок решает задачу минимизации числа блоков, подозреваемых в наличии функциональных нарушений, до одного из них. При этом используется матрица активизации блоков и диагноз m_d, полученный в предыдущей процесс модели.

4. Модуль исправления функциональных нарушений ориентирован на ручной поиск ошибок в одном программном блоке, представленном вектором m_b. Возможен также автоматический режим исправления ошибок в блоках, если в инфраструктуре верификации предусмотрена библиотека диверсных программных модулей, имеющих аналогичные функциональности.

Предложенная инфраструктура является одним их шагов на пути создания автомата верификации программных блоков.Далее представлен пример диагностирования функцио-

нального нарушения на основе использования матрицы активизации. Вектор ассерционных реакций получен из соответствующей матрицы $A_{ij} = \{1 \to \text{failed}, 0 \to \text{passed}\}$ путем дизъюнктивного объединения содержимого каждой строки:

$$m_i = \bigvee_{j=1}^{m} A_{ij} =$$

T	m
T_1	.
T_2	1
T_3	.
T_4	1
T_5	.
T_6	.
T_7	.
T_8	1

=

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.

Последующее выполнение xor-операции между вектором ассерций и столбцами матрицы активизации блоков позволяет найти лучшее решение, которое определяется минимальным кодовым расстоянием $L = L \vee B_j \leftarrow \sum_{i=1}^{n}(B_{ij} \oplus_{i=1}^{n} m_i) = (0 \vee \min)$:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

T	m
T_1	.
T_2	1
T_3	.
T_4	1
T_5	.
T_6	.
T_7	.
T_8	1

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	1	.	.	1	.	1	1	1
T_3	.	.	.	.	.	1	1	1
T_4	.	1	.	1	1	.	1	1
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	1	1	.	.	.	1	1	1
$d(A,B_j)$	4	4	0	4	2	4	6	6

Результат диагностирования – блок 3 имеет функциональные нарушения, поскольку три ассерции «упали» на тестовых сегментах 2,4 и 8, которые в таком сочетании активизируют только блок с номером 3. Если выполнять процесс диагностирования, используя не вектор, а матрицу ассерций, то процесс поиска дефектных блоков будет иметь следующий вид:

B_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	1	.	1	.	.	1	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	1	1	1	.	.	.

$\oplus$

A_{ij}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
T_1	.	.	.	.	.	.	.	.
T_2	.	1	1	.	1	.	.	.
T_3	.	.	.	.	.	.	.	.
T_4	1	.	1	.	.	1	.	.
T_5	.	.	.	.	.	.	.	.
T_6	.	.	.	.	.	.	.	.
T_7	.	.	.	.	.	.	.	.
T_8	.	.	1	.	.	.	.	.

=

L_{ij}	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
T_1	1	.	.	1	.	.	1	.
T_2	.	.	.	.	.	.	.	.
T_3	.	.	.	.	.	1	1	1
T_4	.	.	.	.	.	.	.	.
T_5	.	1	.	1	.	.	.	1
T_6	1	.	.	.	.	.	1	1
T_7	.	1	.	.	1	1	.	.
T_8	.	.	.	1	1	.	.	.
$d(A,B_j)$	2	2	0	3	2	2	3	3

Результат диагностирования аналогичен предыдущему – блок 3 имеет функциональные нарушения, поскольку кодовое расстояние равно нулю только для столбца с номером 3.

8. Заключение

1. Представлена структурная модель отношений на множестве из четырех основных компонентов технической диагностики (функциональность, устройство, тест, дефекты), которая характеризуется полным xor-взаимодействием всех вершин графа и транзитивной обратимостью каждой триады отношений, что позволяет определить и классифицировать пути решения практических задач, включая синтез тестов, моделирование неисправностей и поиск дефектов.

2. Предложена новая модель программного продукта в форме графа блочных транзакций, а также новый матричный метод диагностирования функциональных нарушений, которые характеризуются технологичностью подготовки данных в процессе поиска некорректных блоков, что дает возможность существенно уменьшить время проектирования цифровых систем на кристаллах.

3. Усовершенствованы методы поиска функциональных нарушений, которые отличаются параллелизмом выполнения векторных операций над строками таблицы ФН, что дает возможность существенно (x10) повысить быстродействие вычислительных процедур, связанных с диагностированием и восстановлением работоспособности программных и аппаратных продуктов.

4. Разработана архитектура мультиматричного процессора, ориентированного на повышение быстродействия процедур встроенного диагностирования функциональных нарушений в программном или аппаратном изделии, которая отличается использованием параллельных логических векторных операций and, or, xor, slc, что дает возможность существенно (x10) повысить быстродействие диагностирования одиночных и/или кратных дефектов (функциональных нарушений).

5. Предложена инфраструктура верификации и диагностирования HDL-кода проектируемых цифровых систем на кристаллах, которая имеет четыре процесс-модели для тестирования, диагностирования, оптимизации и исправления ошибок, замкнутые в цикл, что дает возможность уменьшить время отладки кода в процессе создания проекта.

6. Практическая реализация моделей и методов верификации была интегрирована в среду моделирования Riviera компании Aldec. Введенные в систему модули ассерций и диагностирования усовершенствовали существующий процесс верификации, что дало возможность на 15% уменьшить общее время проектирования цифрового изделия.

Список литературы: 1.*Основы* технической диагностики / Под. ред. П.П.Пархоменко. М.: Энергия, 1976.460с. **2.** *Пархоменко П.П., Согомонян Е.С.* Основы технической диагностики (Оптимизация алгоритмов диагностирования, аппаратурные средства) / Под ред. П.П. Пархоменко.М.: Энергия. 1981. 320 с. **3.** *Инфраструктура* мозгоподобных вычислительных процессов / М.Ф. Бондаренко, О.А. Гузь, В.И. Хаханов, Ю.П. Шабанов-Кушнаренко. Харьков: Новое слово. 2010. 160 с. **4.** *Проектирование* и верификация цифровых систем на кристаллах / В.И. Хаханов, И.В. Хаханова, Е.И. Литвинова, О.А. Гузь. Харьков: Новое слово. 2010. 528с. **5.** *Семенец В.В., Хаханова И.В., Хаханов В.И.* Проектирование цифровых систем с использованием языка VHDL. Харьков: ХНУРЭ. 2003. 492 с. **6.** *Хаханов В.И., Хаханова И.В.* VHDL+Verilog = синтез за минуты. Харьков: ХНУРЭ. 2006. 264с. **7.** *Хаханов В.И.* Техническая диагностика цифровых и микропроцессорных структур: Учебник. К.: ИСИО, 1995. 242с. **8.** *Скобцов Ю.А.* Логическое моделирование и тестирование цифровых устройств Ю.А. Скобцов, В.Ю. Скобцов. Донецк: ИПММ НАН Украины, ДонНТУ, 2005. 436 с. **9.** *IEEE* Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture IEEE Std 1149.7-2009. 985 p. **10.** *Da Silva F., McLaurin T., Waayers T.* The Core Test Wrapper Handbook. Rationale and Application of IEEE Std. 1500™. Springer. 2006. XXIX. 276 p. **11.** *Marinissen E.J., Yervant Zorian.* Guest Editors' Introduction: The Status of IEEE Std 1500.IEEE Design & Test of Computers. 2009. No26(1). P.6-7. **12.** *IEEE* Std 1800-2009 IEEE Standard for System Verilog-Unified Hardware Design, Specification, and Verification Language. http://ieeexplore.ieee.org/servlet/opac?punumber= 5354133. **13.** *Marinissen E.J.* Testing TSV-based three-dimensional stacked ICs // DATE 2010. 2010. P.1689-1694. **14.** *Benso A., Di Carlo S., Prinetto P., Zorian Y.* IEEE Standard 1500 Compliance Verification for Embedded Cores // IEEE Trans. VLSI Syst. 2008. No 16(4).P. 397-407. **15.** *Ubar R., Kostin S., Raik J.* Embedded diagnosis in digital systems // 26th International Conference "Microelectronics", MIEL 2008. 2008.P. 421-424. **16.** *Elm M., Wunderlich H.-J.* Scan Chain Organization for Embedded Diagnosis // Design, Automation and Test in Europe, DATE '08. 2008. P. 468–473. **17.** *Bulent I. Dervisoglu.* A Unified DFT Architecture for Use with IEEE 1149.1 and VSIA/IEEE P1500 Compliant Test Access Controllers. Proceedings of the Design Automation Conference. 2001. P. 53-58. **18.** *Chenlong Hu, Ping Yang, Ying Xiao, Shaoxiong Zhou.* Hardware design and realization of matrix converter based on DSP & CPLD // 3rd International Conference Power Electronics Systems and Applications. 2009. P. 1-5. **19.** *Dave N., Fleming K., Myron King, Pellauer M., Vijayaraghavan M.* Hardware Acceleration of Matrix Multiplication on a Xilinx FPGA // 5th IEEE/ACM International Conference Formal Methods and Models for Codesign. 2007. P.97-100. **20.** *Loucks W.M., Snelgrove M., Zaky S.G.* A Vector Processor Based on One-BitMicroprocessors // IEEE Micro.Volume 2, Issue 1.1982. P. 53-62. **21.** *Hilewitz Y., Lauradoux C., Lee R.B.* Bit matrix multiplication in commodity processors // International Conference Application-Specific Systems, Architectures and Processors. 2008. P. 7-12. **22.** *Soon, J.L.K.; Low Ching Ling;* DEV. Design explorer for verification. Integrated Circuits, ISIC '09. Proceedings of the 2009 12th International Symposium: 2009. P.

413– 416. **23.** *Rafe V.; Rafeh R.; Azizi S.; Miralvand M.R.Z.;* Verification and Validation of Activity Diagrams Using Graph Transformation. Computer Technology and Development, 2009. ICCTD '09. 2009. P. 201- 205. **24.** *Xiaoxi Xu; Cheng-Chew Lim;* Using Transfer-Resource Graph for Software-Based Verification of System-on-Chip. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume: 27, Issue: 7. 2008. P. 1315 – 1328. **25.** *Zhongjun Du; Zhengjun Dang*. A New Algorithm Based Graph-Search for Workflow Verification. Information Engineering and Computer Science (ICIECS), 2010. 2nd International Conference: 2010. P. 1–3. **26.** *Горбатов В.А., Горбатов А.В., Горбатова М.В.* Дискретная математика. М: Высшая школа, 2006. 448с.

Поступила в редколлегию 16.03.2011

Ngene Christopher Umerah, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Зайченко Сергей Александрович, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: проектирование и верификация цифровых систем на кристаллах. Увлечения: музыка, путешествия, литература. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: sergeyz@aldec.com.

Литвинова Евгения Ивановна, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ. Научные интересы: автоматизация диагностирования и встроенный ремонт компонентов цифровых систем в пакете кристаллов. Адрес: Украина, 61166, Харьков, пр. Ленина 14, тел. 70-21-421. E-mail: kiu@kture.kharkov.ua.

Скворцова Ольга Борисовна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: автоматизация диагностирования и встроенный ремонт компонентов цифровых систем в пакете кристаллов. Адрес: Украина, 61166, Харьков, пр. Ленина 14, тел. 70-21-421. E-mail: olgas@aldec.com.

УДК 629.78.783

О.Я. НИКОНОВ, А.С. НАЗАРОВ

ИДЕНТИФИКАЦИЯ ДЕФЕКТОВ ИСПОЛНИТЕЛЬНЫХ ОРГАНОВ СИСТЕМЫ УПРАВЛЕНИЯ ОРИЕНТАЦИЕЙ МИКРОСПУТНИКА

Разрабатывается алгоритм на основе субоптимального фильтра Калмана, позволяющий по информации бесплатформенной инерционной навигационной системы идентифицировать дефекты исполнительных органов системы управления ориентацией микроспутника при наличии шумов измерений и действии возмущающих моментов. Приводится математическое моделирование процессов ориентации микроспутника.

1. Постановка задачи

В настоящее время все большую актуальность приобретают проблемы разработки высокоточных систем управления (СУ) космическими аппаратами. Одним из наиболее перспективных направлений в области создания космической техники являются микроспутники [1]. Эффективность применения микроспутников и, следовательно, эффективность решаемых ими практических задач существенным образом зависит от технических характеристик СУ аппарата, поэтому требования к точностным и эксплуатационным характеристикам СУ достаточно жесткие. Дефекты исполнительных органов системы управления ориентацией (СУО) микроспутника могут приводить к невыполнению целевой задачи и раскрутке микроспутника до недопустимых угловых скоростей. Существующие методы контроля работоспособности СУО микроспутника [2, 3] являются достаточно грубыми, чтобы выявлять дефекты исполнительных органов на фоне действия внешних возмущающих моментов (гравитационных, аэродинамических, магнитных, световых и др.). Поэтому разработка алгоритмов идентификации дефектов исполнительных органов при наличии шумов измерений и действии внешних возмущающих воздействий является актуальной задачей. В настоящей статье для построения алгоритма идентификации дефектов исполнительных органов предлагается использовать фильтр Калмана [4]. В процессе исследований микроспутник рассматривается как абсолютно твердое тело (ТТ), не содержащее каких-либо движущихся масс [2]. СУО микроспутника включает силовой гироскопический комплекс (СГК), состоящий из n одинаковых гиродинов, три пары реактивных двигателей (РД) малой тяги и три инерционных маховика (ИМ), обеспечивающих управляющие моменты вдоль главных связанных осей инерции ТТ, в дальнейшем называемых связанными. В этих условиях динамика вращения микроспутника вокруг центра масс описывается следующим векторным уравнением Эйлера:

$$I\dot{\overline{\omega}} + (\overline{\omega} x I\overline{\omega}) = \overline{M}_{УПР} + \overline{M}_{В}, \qquad (1)$$

где $\overline{\omega}$ – вектор угловой скорости ТТ в проекциях на связанные оси; I – диагональная матрица главных моментов инерции; $\overline{M}_{УПР}$ – управляющий момент; $\overline{M}_{В}$ – внешний возмущающий момент.

Наряду с динамическим уравнением рассматривается кинематическое уравнение, связывающее вектор угловой скорости $\overline{\omega}$ с углами поворота триэдра осей Oxyz относительно триэдра осей некоторой базовой системы координат (БСК), начало которой совпадает с началом координат связанной системы координат (ССК), а оси определенным образом ориентированы в инерциальном пространстве и движутся поступательно. Пусть кватернион ориентации, компоненты которого являются параметрами Родрига-Гамильтона, полностью определяют угловое положение ССК относительно БСК. Понятие кватерниона ориентации становится однозначным лишь после того, как введена последовательность поворотов твердого тела вокруг осей Ox, Oy, Oz. Для данной последовательности поворотов воспользуемся кватернионной кинематической моделью сферического типа [5]:

$$\dot{\Lambda} = \frac{1}{2}\Lambda \circ \overline{\omega}; \qquad (2)$$

$$\Lambda^*(t) = \Lambda_1(t) \circ \Lambda_2(t) \circ \Lambda_3(t), \quad (3)$$

где $\Lambda_i(t) = \cos\left(\frac{\psi_i(t)}{2}\right) + \sin\left(\frac{\psi_i(t)}{2}\right) \cdot \bar{b}_i$, $i = \overline{1,3}$.

Кватернионное кинематическое уравнение (2) представляют собой систему четырех линейных невырождающихся уравнений, которые удовлетворяют одному уравнению связи для нормирования кватерниона

$$\lambda_0^2 + \lambda_1^2 + \lambda_2^2 + \lambda_3^2 = 1.$$

Система уравнений (1) и (2) описывает угловое движение твердого тела относительно БСК. Текущее значение $\bar{\omega}$ оценивается в системе по информации измерителя угловой скорости, измеряющего интегралы от проекций вектора абсолютной угловой скорости микроспутника на оси чувствительности прибора.

Интегрируя кинематическое уравнение (2) в бортовой цифровой вычислительной машине (БЦВМ) при начальных значениях углов, а также уравнения движения центра масс микроспутника (1) при соответствующих начальных условиях, реализуют бесплатформенную инерциальную навигационную систему (БИНС). Таким образом, считаем, что текущие величины углов непрерывно вычисляются в БИНС.

Управляющий момент $\bar{M}_{УПР} = \bar{M}_{РД} + \bar{M}_{ГД} + \bar{M}_{ИМ}$ формируется в соответствии с логикой закона управления исполнительных органов СУО микроспутника (реактивных двигателей, гиродинов, инерционных маховиков) и обеспечивает заданное угловое положение микроспутника. Источником внешнего возмущающего момента $\bar{M}_{В} = \bar{M}_{АЭР} + \bar{M}_{ГРВ} + \bar{M}_{МГН} + \bar{M}_{СВТ}$ является взаимодействие микроспутника с внешней средой, приводящее к появлению воздействия на корпус внешних сил – аэродинамического, гравитационного, магнитного, светового [3].

Законы управления для исполнительных органов СУО микроспутника имеют следующий вид:

а) для реактивных двигателей

$$\bar{M}_{РД} = \{R^3 \,\|\, |M_i| \le m_i, i = \overline{1,3}\}; \quad (4)$$

б) для гиродинов

$$\bar{M}_{ГД} = -(\bar{\omega} x \bar{H}) + L(\bar{\beta}) \cdot \dot{\bar{\beta}}; \quad (5)$$

где $H(\bar{\beta})$ – собственный кинетический момент СГК; $L(\bar{\beta})$ – матрица, элементы которой являются функциями от $\bar{\beta}$ – углов прецессии ГД, $\dot{\bar{\beta}}$ – скорости прецессии гиродинов, причем $\bar{\beta} = \{R^n \,\|\, |\dot{\beta}_i| \le \dot{\beta}_{max}, i = \overline{1,n}\}$;

в) для инерционных маховиков

$$\bar{M}_{ИМ} = -(\bar{\omega} x \bar{h}) + J\dot{\bar{\Omega}}, \quad (6)$$

здесь $\bar{h}$ – собственный кинетический момент системы ИМ, *J* – осевой момент инерции ИМ; $\bar{\Omega}$ – вектор угловых скоростей вращения маховиков относительно ТТ, причем $\dot{\bar{\Omega}} = \{R^3 \,\|\, |\dot{\Omega}_i| \le \dot{\Omega}_{max}, |\Omega_i| \le \Omega_{max}, i = \overline{1,3}\}$.

Внешние возмущающие моменты в зависимости от физической природы описываются следующим образом:

а) аэродинамический возмущающий момент

$$\bar{M}_{АЭР} = \int_S \bar{r} \times d\bar{F}, \quad (7)$$

где $\bar{r}$ – радиус-вектор площадки, имеющий начало в центре масс тела; $d\bar{F}$ – вектор силы взаимодействия тела и среды;

б) гравитационный возмущающий момент

$$\bar{M}_{ГРВ} = \int_S \bar{p} \times d\bar{G}, \quad (8)$$

здесь $\bar{p}$ – радиус-вектор некоторой элементарной массы материального тела; $d\bar{G}$ – вектор силы тяжести, действующей на эту элементарную массу;

в) магнитный возмущающий момент

$$\bar{M}_{МГН} = \bar{L} \times \bar{B}, \tag{9}$$

где $\bar{L}$ – вектор магнитного момента; $\bar{B}$ – вектор магнитной индукции;

г) момент от светового давления

$$\bar{M}_{СВТ} = \int_S \bar{r} \times d\bar{f}, \tag{10}$$

$\bar{r}$ – радиус-вектор площадки, имеющий начало в центре масс микроспутника; $d\bar{f}$ – вектор суммарной силы, действующей на элемент поверхности микроспутника.

Целью настоящей работы является получение алгоритма идентификации дефектов исполнительных органов СУО микроспутника при наличии шумов измерений и действии возмущающих моментов.

Алгоритм обработки данных в бесплатформенной инерциальной навигационной системе строится с использованием субоптимального дискретного фильтра Калмана [4].

Векторные уравнения (1) и (2) в скалярном виде представляют систему из 7 уравнений:

$$\begin{cases} J_x\dot{\omega}_x - (J_y - J_z)\omega_y\omega_z = M_{упр\,x} + M_{в\,x}, \\ J_y\dot{\omega}_y - (J_z - J_x)\omega_z\omega_x = M_{упр\,y} + M_{в\,y}, \\ J_z\dot{\omega}_z - (J_x - J_y)\omega_x\omega_y = M_{упр\,z} + M_{в\,z}, \\ \dot{\lambda}_0 = -0{,}5(\lambda_1\omega_x + \lambda_2\omega_y + \lambda_3\omega_z), \\ \dot{\lambda}_1 = 0{,}5(\lambda_0\omega_x + \lambda_2\omega_z - \lambda_3\omega_y), \\ \dot{\lambda}_2 = 0{,}5(\lambda_0\omega_y + \lambda_3\omega_x - \lambda_1\omega_z), \\ \dot{\lambda}_3 = 0{,}5(\lambda_0\omega_z + \lambda_1\omega_y - \lambda_2\omega_x). \end{cases} \tag{11}$$

2. Решение задачи идентификации дефектов исполнительных органов системы управления

При получении численного решения дифференциальных уравнений движения микроспутника (11) был выбран многошаговый метод Рунге-Кутта 4-го порядка. При численной реализации необходимо выбрать шаг h численного интегрирования рассматриваемых дифференциальных уравнений или, что эквивалентно, общее количество шагов N на временном интервале решения. Для рассматриваемого интервала в 100 с было выбрано N=100000.

Для малых угловых отклонений осей ССК от БСК и при условии $I_x \approx I_y \approx I_z$ уравнения (11) запишем в виде

$$\begin{cases} J_j\dot{\omega}_j = M_{упрj} + M_{вj}, \\ \dot{\varphi}_j = \omega_j. \end{cases} \qquad (j = x, y, z)$$

Тогда для построения системы оценки вектора состояния микроспутника примем следующую модель объекта наблюдения:

$$\begin{cases} \dot{\varphi}_j = \omega_j, \\ \dot{\omega}_j = m_{в\,j} + m_j u_j, \\ \dot{m}_{в\,j} = 0, \end{cases} \tag{12}$$

где $m_j = M_{упр\,j}/J_j$ – эффективность управляющего момента; $M_{упр\,j}$ – управляющий момент; $m_{в\,j} = M_{в\,j}/J_j$ – эффективность возмущающего момента; u_j – сигнал управления; j=x, y, z.

Запишем систему уравнений (12) в стандартной векторно-матричной форме, дополнив ее уравнением измерений

$$\begin{cases} \dot{x}_j = Ax_j + Bu_j, \\ z_j = Hx_j + \xi_j, \end{cases}$$

где x_j – вектор состояния; z_j – вектор измерений; x_j – шум измерений;

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ m_j \\ 0 \end{bmatrix}, \qquad H = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \; j=x, y, z.$$

Используя критерий Калмана, несложно показать, что такая система является полностью наблюдаема $\mathrm{rank}[H^T\ A^T H^T (A^T)^2 H^T] = n = 3$, где n – порядок системы.

Реализация в бортовом вычислителе дискретного фильтра Калмана сводится к оценке вектора состояния по следующим соотношениям:

$$\begin{aligned} \hat{x}_{K+1,j} &= \Phi_{Kj}\hat{x}_{Kj} + K_{K+1,j}(z_{K+1,j} - H_j\Phi_{Kj}x_{Kj}); \\ K_{K+1,j} &= P^*_{K+1,j}H_j^T(H_jP^*_{K+1,j}H_j^T + R_{Kj})^{-1}; \\ P_{K+1,j} &= P^*_{K+1,j} - K_{K+1,j}H_jP^*_{K+1,j}; \\ P^*_{K+1,j} &= \Phi_{Kj}P_{Kj}\Phi_{Kj}^T, \end{aligned} \qquad (13)$$

где $\hat{x}_{Kj}$ – оценка вектора состояния; Φ_{Kj} – переходная матрица для вектора состояния; H_j – матрица измерений; P_{Kj} – ковариационная матрица ошибок фильтрации; P^*_{Kj} – ковариационная матрица ошибок прогноза; K_{Kj} – матричный коэффициент усиления; R_{Kj} – ковариационная матрица шумов измерения; j = x, y, z.

Работа алгоритма основана на анализе величины оцениваемого в фильтре Калмана возмущающего момента. Если математическое ожидание оценки возмущающего момента, вычисленного на некоторой временной базе, где управление равно нулю, превосходит допустимый порог, то принимается решение о дефекте исполнительного органа СУО микроспутника.

Для проверки работоспособности алгоритма проведено математическое моделирование процессов ориентации микроспутника при возникновении дефекта его исполнительного органа СУО на временном интервале 100 с. Моделирование проводилось для нескольких типов дефектов. Моменты инерции микроспутника принимались равными 10 Нмс^2 в трех каналах, величина управляющего момента принималась равной 5 Нм, а величина внешнего возмущающего момента – 0,2 Нм в каждом канале управления. Проекции начальной угловой скорости микроспутника задавались равными 0,01 рад/с в трех каналах.

Графики процессов микроспутника приведены на рис. 1-4. На них показаны компоненты векторов угловой скорости и углового ускорения вращения микроспутника, кватерниона ориентации и суммарного управляющего момента микроспутника соответственно.

Как показали результаты моделирования, понижение момента при дефекте исполнительного органа СУО микроспутника приводит к увеличению времени идентификации дефектов. Моделирование показало также, что существенное повышение уровня шумов измерений не приводит к значительному снижению чувствительности системы к выявлению дефектов.

Таким образом, предложенный на основе фильтра Калмана алгоритм позволяет идентифицировать дефекты исполнительных органов СУО микроспутника при наличии шумов измерений и действии возмущающих моментов.

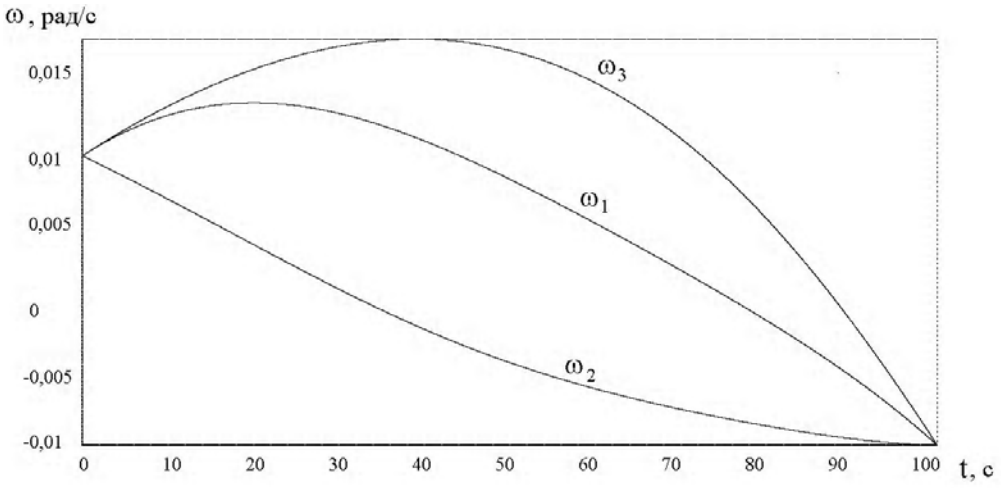


Рис. 1. Угловая скорость вращения микроспутника

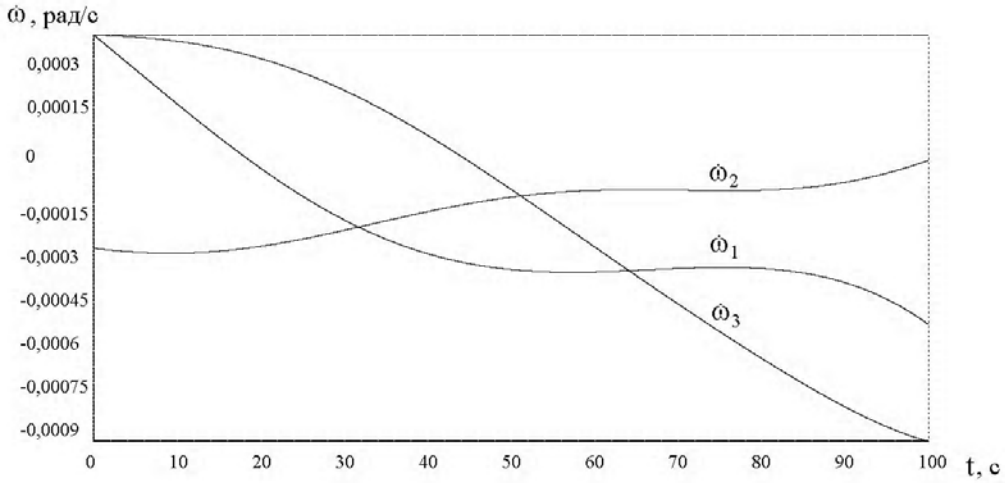


Рис. 2. Угловое ускорение вращения микроспутника

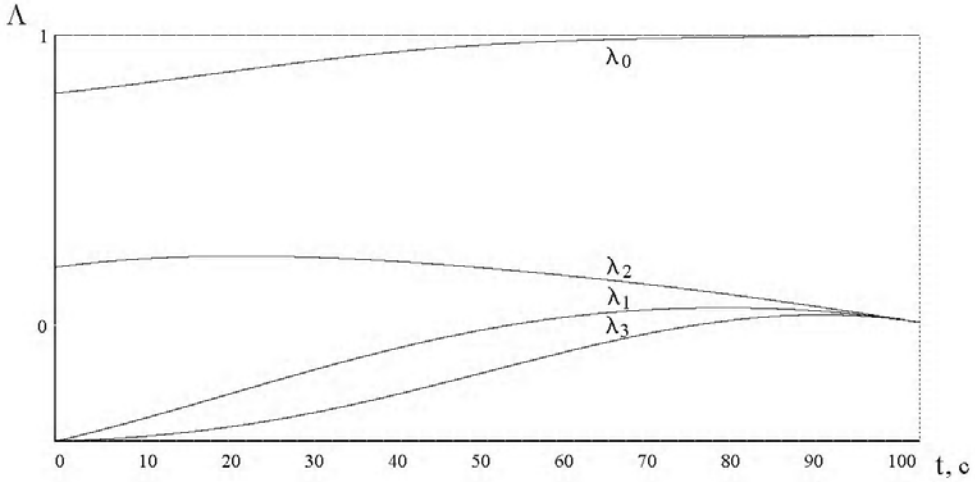


Рис. 3. Кватернион ориентации микроспутника

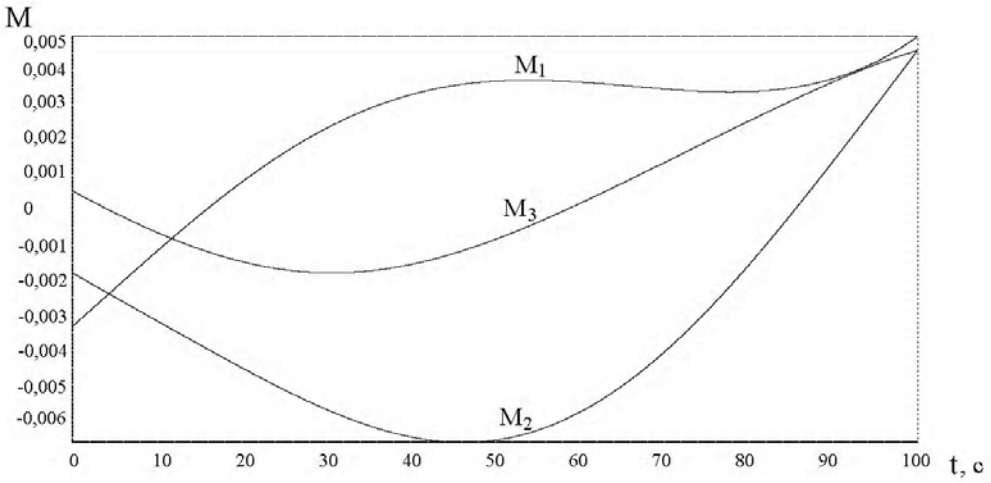


Рис. 4. Суммарный управляющий момент микроспутника

3. Заключение

Предложенный на основе фильтра Калмана алгоритм позволяет идентифицировать дефекты исполнительных органов СУО микроспутника при наличии шумов измерений и действии возмущающих моментов.

Научная новизна: разработан алгоритм на основе субоптимального фильтра Калмана, позволяющий по информации бесплатформенной инерционной навигационной системы идентифицировать дефекты исполнительных органов системы управления ориентацией микроспутника при наличии шумов измерений и действии возмущающих моментов.

Практическая значимость: разработано алгоритмическое и программное обеспечение для блока диагностики информационно-управляющих систем транспортных средств.

В дальнейших исследованиях для диагностики и контроля отказов исполнительных органов системы управления ориентацией микроспутника целесообразно использовать элементы искусственного интеллекта, такие как искусственные нейронные сети и методы эволюционного моделирования.

Список литературы: 1. *Черный И.* Интерес к малым спутникам растет / И. Черный, Л. Розенблюм // Новости космонавтики. 2008. Т.18, №3(302). С. 71. **2.** *Алексеев К.Б.* Управление космическими летательными аппаратами / К.Б. Алексеев, Г.Г. Бебенин. М.: Машиностроение, 1974. 340 с. **3.** *Раушенбах Б.В.* Управление ориентацией космических аппаратов / Б.В. Раушенбах, Е.Н. Токарь. М.: Наука, 1974. 600 с. **4.** *Калман Р.Э.* Очерки по математической теории систем / Р.Э. Калман, П.Л. Фалб, М.А. Арбиб. М.: Едиториал УРСС, 2004. 400 с. **5.** *Бранец В.Н.* Введение в теорию бесплатформенных инерциальных навигационных систем / В.Н. Бранец, И.П. Шмыглевский. М.: Наука, 1992. 280 с.

Поступила в редколлегию 12.05.2011

Никонов Олег Яковлевич, д-р техн. наук, доцент, заведующий кафедрой информатики Харьковского национального автомобильно-дорожного университета. Научные интересы: информационные технологии, системы и процессы управления, математическое моделирование. Адрес: Украина, 61002, Харьков, ул. Петровского, 25, тел.: (057) 707-37-74, e-mail: nikonov@khadi.kharkov.ua.

Назаров Алексей Сергеевич, старший преподаватель кафедры информатики Харьковского национального автомобильно-дорожного университета. Научные интересы: системы и процессы управления, математическое моделирование. Адрес: Украина, 61002, Харьков, ул. Петровского, 25, тел.: (057) 707-37-74.

УДК 004.7; 004.8; 007.85

И.Н. КЛИМОВ

МЕТОД ОПТИМИЗАЦИИ РИСКОВ ЧРЕЗВЫЧАЙНЫХ СИТУАЦИЙ ПРИРОДНОГО ПРОИСХОЖДЕНИЯ В РАСПРЕДЕЛЕННОЙ СИСТЕМЕ ЦЕНТРОВ МОНИТОРИНГА

Анализируется специфика чрезвычайных ситуаций природного происхождения. На основании проведенного анализа выдвигаются требования к системе мониторинга чрезвычайной ситуации. Вводится понятие распределенной системы центров мониторинга, предлагается метод оптимизации рисков в подобной сети

Введение

Масштабность распространения и многофакторность воздействия выбросов на биосферу, ноосферу и климат Земли определяют необходимость системного исследования многих глобальных процессов. Среди них особое место занимают процессы возникновения чрезвычайных ситуаций, как вносящие существенные риски в основы жизни и деятельности населения планеты. Системные исследования по управлению рисками и снижению воздействий чрезвычайных ситуаций природного и техногенного происхождения на качество окружающей среды и экологическую обстановку выполняются во многих странах, как в масштабе отдельной страны, так и в мировом масштабе в соответствии с решениями ООН, Европейской комиссии, Международного энергетического агентства.

Мониторинг рисков чрезвычайных ситуаций – сложная наукоемкая задача, жизненно важная для обеспечения экологической безопасности окружающей среды, снижения материальных затрат и потерь человеческих ресурсов при возникновении обстоятельств непреодолимой силы (наводнения, лавины, цунами)

Взрывообразный рост населения приводит к заселению потенциально опасных районов планеты. Также необходимо отметить, что ряд промышленностей (горнодобывающая, атомная энергетика, химическая) неизбежно связаны с рисками как природного, так и техногенного происхождения

Задача оптимизации рисков чрезвычайных ситуаций заключается в раннем обнаружении потенциальных чрезвычайных ситуаций (и в этом контексте она тесно связана с задачами мониторинга окружающей среды), в обеспечении оперативной реакции соответствующих организаций при возникновении чрезвычайных ситуаций в целях минимизации их последствий и ликвидации потенциальных будущих угроз.

Используемые на практике методы позволяют осуществлять прогнозирование развития чрезвычайных ситуаций в конкретной точке пространства, основываясь на статистических данных и данных мониторинга. К сожалению, при возникновении динамически развивающихся чрезвычайных ситуаций (выброс отравляющих веществ в окружающую среду, паводок и т.д.) мощности одного центра оказывается недостаточно для эффективного обсчета потенциальных вариантов развития ситуации, что приводит к неполноте информации, предоставляемой лицу, принимающему решения (ЛПР), и, как следствие, к ошибочным действиям для разрешения чрезвычайной ситуации. Значимость человеческого фактора при подобной системе принятия решений вырастает многократно, поскольку эксперт вынужден работать в условиях неполноты информации и опираться на слабо формализуемые критерии, что многократно увеличивает риск человеской ошибки.

Цель исследования: разработать метод оптимизации рисков чрезвычайных ситуаций, позволяющий эффективно использовать распределенную систему центров мониторинга и прогнозирования, направленный на улучшение точности прогнозирования в рамках установленных временных ограничений на время вычисления предполагаемого решения.

Задачи исследования: определить способы обмена информацией и повторного использования накопленных знаний в распредленной сети центров чрезвычайных ситуаций.

1. Анализ динамики развития чрезвычайных ситуаций природного происхождения

Чрезвычайные ситуации природного происхождения – результат непредсказуемого развития экологически опасных процессов, повлекший за собой существенные материальные либо человеческие потери. С точки зрения системного анализа экологически опасные процессы есть подмножество сложных физических систем, со всеми их свойствами – открытостью, динамичностью, уникальностью, слабой формализуемостью, многокритериальностью в смысле выбора решений, неопределенностью, вызванной: неполнотой или отсутствием знаний о природе данного процесса, ограниченной возможностью математического описания и вычислительной реализации, сложностью в применении технических средств измерения или управления, наличием стохастического или субъективного (волюнтаристского) факторов. К ним необходимо применение системного анализа, как наиболее универсального и адекватного современным требованиям средства исследования. Этот вид анализа, в свою очередь, заключается в применении методологических, математических и организационных средств, предназначенных для выявления внутренних и внешних взаимосвязей и взаимодействий между процессами-объектами – как элементами системы одно- или разнотипных и природных по происхождению, протеканию и средствам описания, оцениванию параметров, моделированию и, как следствие, – прогнозированию или научному предвидению. Что, в свою очередь, дает возможность лицу, принимающему решение, получить максимально полную, достоверную и главное, своевременную информацию о возможных или неизбежных негативных воздействиях опасных процессов на экосреду.

Системный характер исследований обусловлен самой природой подобных процессов, как это можно увидеть (рис. 1) на примере развития каскадного взаимодействия стихийных явлений. Все элементы взаимосвязаны, и возникновение одного из бедствий ведет к вероятностной экологически разрушимой цепной реакции. Существующие направления в исследовании, вплоть до сегодняшнего дня, разделялись по типам объектов и по предметной области, что в свою очередь затрудняло понимание существа влияния и взаимовлияния процесса на процесс и порождающих последствий.

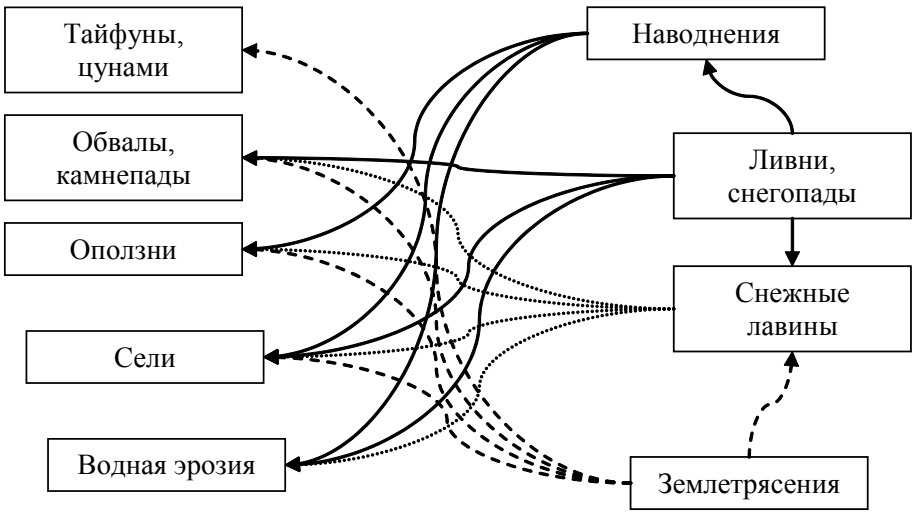


Рис. 1. Схема развития каскадного взаимодействия стихийных явлений

2. Требования к системам мониторинга рисков чрезвычайных ситуаций

Проведенный выше анализ свойств и особенностей экологически опасных процессов определяет необходимость нового подхода к информационному обеспечению процессов контроля, оценивания и прогнозирования состояния, динамики и последствий экологически опасных процессов, который должен базироваться на системной методологии исследования разнородных факторов риска и оценивания возможностей своевременного противодействия стихии и учитывать пространственный разнос и распределенность в пространстве

как условий возникновения данного вида процессов, так и результатов их воздействия на экосреду.

Необходимость учитывать данные факторы вынуждает предъявить специфические требования к структуре и свойствам информационной системы мониторинга, системного анализа и прогнозирования динамики экологически опасных процессов. Это требование практически очевидно - для исследования свойств, специфики и динамики процессов, пространственно распределенных на больших площадях. Требуется пространственно распределенная система сбора и обработки первичной информации о разнотипных факторах риска и свойствах процессов в различных масштабах исследуемого пространства. Вместе с тем, подобные системы имеют определенные особенности в процессах функционирования и управления. Представляется целесообразным при разработке структуры и моделей информационной системы мониторинга, системного анализа экологически опасных процессов принять, что:

– общая стратегия принятия решений в сложной иерархической системе соответствует принципам централизованного управления в динамике ЧЭС;

– централизованное управление является наиболее эффективным видом управления в динамике ЧЭС, что подтверждает практика многих стран;

– система централизованного контроля экологической обстановки и управления в динамике ЧЭС принята в Украине.

Основная идея предлагаемого подхода – на основе параллельного анализа разнородных факторов риска обеспечить системно согласованную взаимосвязь различных информационных процессов формирования, обоснования и реализации решений по своевременному противодействию негативному влиянию. Цель подхода – повысить обоснованность и достоверность принимаемых решений в условиях неопределенности, неполноты, неточности и противоречивости исходной информации о факторах риска. Стратегия подхода – обеспечить рациональное использование априорной и текущей информации о факторах риска как при оценивании и прогнозировании экологически опасных процессов, так и при разработке решений и мероприятий по предотвращению и (или) минимизации их нежелательных последствий .

3. Структура информационной системы мониторинга и оптимизации рисков чрезвычайных ситуаций

На основе предлагаемого подхода разработана иерархическая структура информационной системы мониторинга и оптимизации рисков чрезвычайных ситуаций (рис. 2).

Состав и последовательность функциональных элементов иерархической структуры информационной системы определяется последовательностью информационных процессов формирования и обоснования решений при оценивании и прогнозировании динамики экологически опасных процессов. Первоочередной задачей в этой последовательности является сбор и формирование исходной информации об окружающей среде и экологической обстановке с требуемыми уровнями полноты, достоверности и своевременности. Это достигается на основе непрерывного контроля, системного анализа и прогнозирования разнотипных и разноприродных факторов риска, действие которых является основными причинами и источниками экологически опасных процессов. Процедуры решения рассматриваемой задачи реализуются на иерархическом уровне 1 «Система анализа состояния и динамики факторов риска». В любом регионе обязательно выполняется анализ природных и техногенных процессов и факторов риска. В зависимости от специфики природных, производственных и других процессов в конкретном регионе дополнительно может выполняться анализ и других факторов риска, например, анализ и мониторинг динамики уровня воды в реках, анализ и мониторинг динамики загрязнения воздуха, грунтовых вод, водоемов специфическими веществами и т.п. Целью такого анализа и мониторинга является своевременное предупреждение потенциальных объектов и субъектов о степени и уровне риска нештатных и критических ситуаций. Результаты решения данной задачи служат основой для анализа и прогнозирования состояния и динамики развития экологически опасных процессов, что составляет содержание второй задачи.

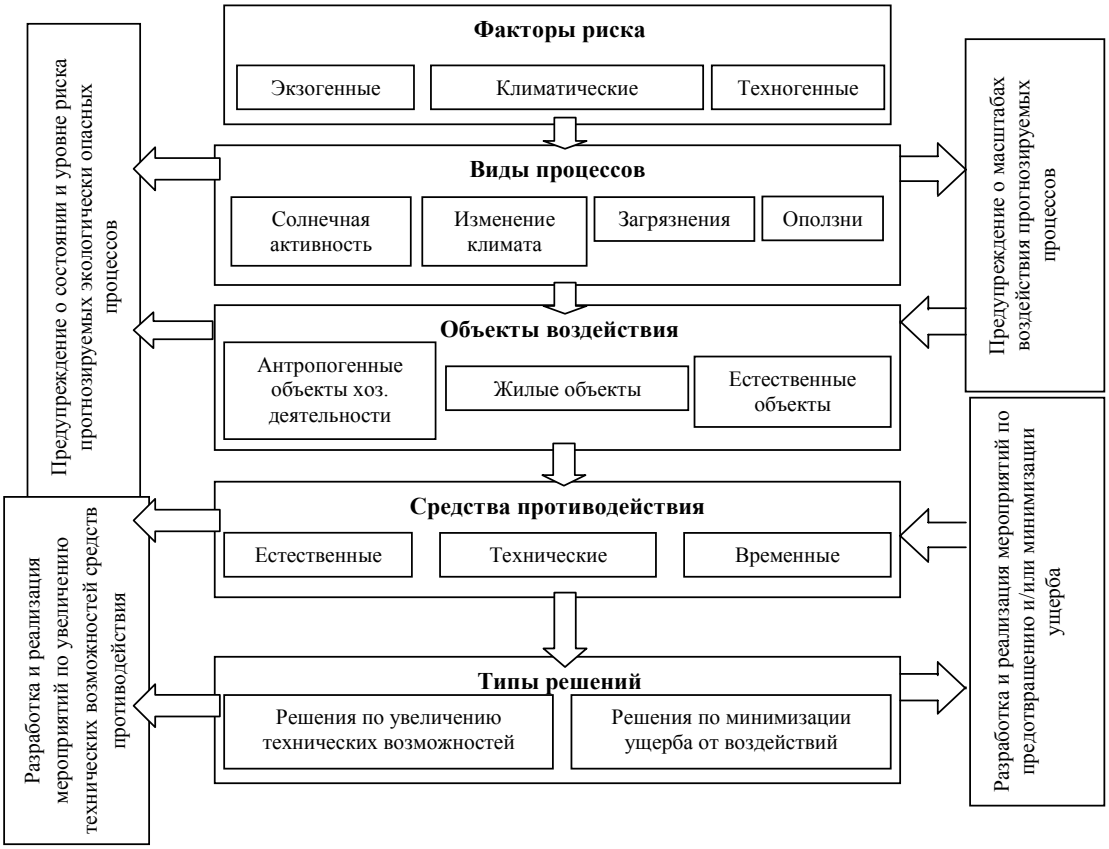


Рис 2. Иерархическая структура информационной системы мониторинга и оптимизации рисков

Ее суть - анализ и прогнозирование состояния и динамики данных процессов. Процедуры решения данной задачи реализуются на иерархическом уровне 2 «Система анализа состояния и динамики экологически опасных процессов». Результатом ее решения является оперативная информация о состоянии и динамике развития разного рода экологически опасных процессов. Полученная на этапе 2 оперативная информация о состоянии и динамике процессов используется в рассматриваемой последовательности информационных процессов формирования и обоснования решений, как исходная информация для следующих задач: оценивание возможных результатов воздействия прогнозируемых экологически опасных процессов; оценивание возможности различных средств противодействия прогнозируемым процессам. Процедуры решения этих задач реализуются в соответствующих системах на иерархических уровнях 3 и 4 «Система анализа и прогнозирования рисков воздействия процессов», «Система анализа и прогнозирования возможностей средств противодействия». Решения данных задач должны быть взаимно согласованы по целям, срокам, ресурсам и ожидаемым результатам, так как только при таком подходе к их решению возможно обеспечить рациональное использование ограниченных ресурсов в условиях неустранимого временного ограничения на цикл формирования и реализацию решений в динамике ЧЭС. Отсюда следует, что данные задачи должны решаться параллельно в целях с одной стороны, оценивания и прогнозирования риска и возможных последствий воздействия экологически опасных процессов на объекты хозяйственной деятельности, жилые, культурные и другие объекты, а с другой стороны – оценивания и прогнозирования возможностей средств защиты по противодействию разрушительным воздействиям экологически опасных процессов. Результаты решения этих задач являются исходной информацией для уровня 5 «Система формирования решений». Данный уровень информационной системы параллельно формирует и обосновывает:

– решения и мероприятия по предотвращению и (или) минимизации ущерба от негативных воздействий;

– решения и мероприятия по увеличению, в случае необходимости, технических возможностей средств противодействия.

Указанные решения и мероприятия разрабатываются при двух существенно различающихся условиях: в динамике повседневного мониторинга и контроля экологической обстановки и в динамике конкретной ЧЭС

4. Математическая модель мониторинга и оптимизации рисков чрезвычайных ситуаций

Центральным элементом модели является понятие ситуации. В рамках исследования понятие ситуации формулируется следующим образом:

$$\mathrm{Sit} = \{s_1 ... s_n, \mathrm{Sit}'\}, 0 \le s_i \le 1, i = 1...n,\ s_i \in S, \quad (1)$$

где $s_1 ... s_n$ – нормализованное множество значений сенсоров, установленных на объекте мониторинга (под сенсорами здесь и далее будем понимать не только аналоговые и цифровые приборы, но и точки зрения экспертов, нормализованные на единичной шкале); Sit' – данные про ситуацию, полученные от других систем монторинга, а S – множество существующих типов датчиков, описанных в базе знаний. Примером информации, полученной от других систем мониторинга, может быть, к примеру, описание погодных условий от ближайшей метеостанции. Для определения чрезвычайной ситуации введем функцию E, выходным значением которой является численная оценка текущего состояния окружающей среды:

$$E(\mathrm{Sit}) = \max e_i(\mathrm{Sit}_i), i = 1..n, \quad (2)$$

где e_i – функция оценки уровня риска группы показаний датчиков Sit_i, причем $\mathrm{Sit}_i \subseteq \mathrm{Sit}$. Вид функции e_i зависит от специфики исследуемой системы, но все функции семейства e обладают следующими свойствами:

– результатом функции является числовое значение в диапазоне 0...1, описывающее меру близости к чрезвычайной ситуации в пространстве возможных значений функции e_i;

– для компоненты ситуации Sit_i существует ровно одна функция e_i, имеющая зависимость от Sit_i

Процесс оценки рисков состоит из серии последовательных измерений $t_1, t_2 ... t_n$, причем интервал между измерениями t_k и t_{k+1} определяется по формуле:

$$\Delta t_{k+1} = \min(\Delta t_{\max}, \Delta t_k - k \times (E(\mathrm{Sit}_{k+1}) - E(\mathrm{Sit}_k)), \quad (3)$$

где $\Delta t_{\max}$ – верхняя граница допустимого периода измерения; Δt_k – интервал между предыдущими измерениями; k – фиксированный коэфициент влияния риска. Благодаря формуле (3) обеспечивается процесс адаптации системы к динамике изменения риска окружающей среды – с ростом риска интенсивность измерений растет, а также обеспечивается интерактивная работа системы. Однако необходимо заметить, что при высоких значениях интегрального риска вычисление функции e_i, используемой в (2), может занимать время, сравнимое с Δt_k, что приводит к ограничению на время отклика системы ΔK в пределах $\sum_{i=1}^{n}(t(e_i(\mathrm{Sit}_i)))\Delta t_{\max}$, где $t(a)$ – функция, описывающая величину процессорного времени, необходимую для вычисления функции a.

В общем виде принцип работы системы мониторинга и оптимизации рисков можно описать следующим образом:

$$\left.\begin{matrix}\mathrm{Sit}' \\ s_1 ... s_n\end{matrix}\right\} \xrightarrow{\Delta S} \mathrm{Sit} \xrightarrow{\Delta K} E(\mathrm{Sit}) \xrightarrow{\Delta U} M(\mathrm{Sit}), \quad (4)$$

где ΔS – фактическая задержка получения показаний мониторинга, ΔK вычисляется с помощью (3), а ΔU – задержка, связанная с принятием решения по оптимизации рисков в условиях ситуации Sit. Существующие задержки в системе соотносятся следующим обра-

зом: $\Delta S << \Delta K << \Delta U$. Такое соотношение обусловлено экспоненциальным ростом информации для обработки, необходимой на каждом шаге принятия решений.

Уменьшение ΔK в (4) позволяет улучшить контроль за развитием чрезвычайной ситуации и является очень важным в процессе оптимизации рисков. Классический подход к решению этой проблемы включает увеличение вычислительных мощностей системы мониторинга, что приводит к следующим негативным последствиям:

– увеличение рисков отказов из-за усложенения системы мониторинга;

– рост экономических затрат как для создания, так и для поддержки существующих систем;

– рост объемов обрабатываемых данных.

Альтернативным решением этой проблемы является постройка сети ситуационных центров, которая позволяет проводить вычисления функции e_i параллельно. Однако вид функций e_i для разных центров мониторинга существенно отличается из-за специфики расположения компонентов мониторинга (например угла наклона для мониторинга лавин, глубина измерения сейсмической активности для землетрясений и т.п.). Совокупность фиксированных факторов, описывающих специфику конкретного центра мониторинга, будем называть геоинформационным контекстом и обозначать буквой G. При условии использования контекста формула (4) приобретает вид:

$$\max_{i=1..n}(t(e_i(Sit_i, G))) \le \Delta K \le \max_{i=1..n}(t(e_i(Sit_i, G))) + \Delta t_{max} . \qquad (5)$$

Необходимо учесть, что приведенная оценка времени решения задачи мониторинга является оптимистической и достигается только при наличии большого количества центров мониторинга. Однако построенная система имеет следующие преимущества:

– простота смены архитектуры системы, возможность ее расширения без изменения основополагающей концепции;

– робастность модели – отказ одного узла приводит к росту времени ΔK , а не к отказу всей системы;

– эффективное использование вычислительных ресурсов из-за равномерного распределения задач мониторинга среди узлов, которые в данный момент не нагружены.

Выводы

Предложенный метод позволяет эффективно решить задачу комплексного мониторинга рисков чрезвычайных ситуаций на большой площади.

Впервые предложен метод прогнозирования возникновения чрезвычайной ситуации на основе распределенной базы знаний и статистической оценки на основе наблюдений за определённый промежуток времени, что позволяет снизить время прогнозирования за счет использования распределенных вычислений и моделей, оптимизированных под распределенные вычисления. *Впервые* предложен метод оптимизации рисков чрезвычайной ситуации с использованиям кластерной архитектуры с учетом геоинформационного контекста.

В дальнейших исследованиях необходимо провести детальную формализацию структуры предлагаемой сети в целях построения эффективного инструментария для проектирования и оценки необходимого количества узлов системы мониторинга.

Список литературы: 1. *Белов П.Г.* Системный анализ и моделирование опасных процессов в техносфере / П.Г.Белов, М.: Академия, 2003. 512 с. **2.** *Drenick K.* A mathematical theory of organization. Acad. Press. 1986. 340 p. **3.** *Гражданкин А.И.* Экспертная система оценки техногенного риска опасных производственных объектов/А.И. Гражданкин, П.Г. Белов//Безопасность труда в промышленности. 2000. № 11. С.6-10.

Климов Илья Николаевич,аспирант кафедры информатики. Научные интересы: системы реального времени, мониторинг. Адрес: Украина, 61189, Харьков, ул. Мира 118, кв. 55, тел.: 8(0572)99-28-55, e-mail:illya.klymov@gmail.com.

УДК 004.7; 004.8; 007.85

Я. ДАЮБ

МОДЕЛЬ ОПТИМИЗАЦИИ ПРИНЯТИЯ РЕШЕНИЙ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ РИСКОВ ПРИ ПЕРЕВОЗКЕ ГРУЗОВ ПОВЫШЕННОЙ ОПАСНОСТИ

Анализируется информационная среда при перевозке грузов повышенной опасности, при этом производится моделирование информационной среды для поддержки принятия решений. Вводится понятие микроситуации. Строится модель принятия решения на основании количественных и качественных характеристик микроситуации.

Введение

В настоящее время задача управления транспортными объектами повышенной опасности в условиях неопределенности возникновения транспортных происшествий является все более актуальной. Данная задача включает моделирование маршрутных перевозок, оптимизацию принятия решений в условиях неопределенности, разработку информационной среды для поддержки принятия решений в области перевозок грузов повышенной опасности.

Организация безопасных маршрутов опасных грузов в случае возникновения чрезвычайных ситуаций (ЧС) является важнейшей задачей управления на транспорте, которая требует переосмысления методов и подходов к решению данной задачи, а также применения новейших достижений в области информационных технологий.

Цель исследования – разработать динамическую модель транспортной задачи, направленную на принятие оперативных, тактических и стратегических решений, снижение потерь и рисков при возникновении чрезвычайных ситуаций на маршруте прохождения объектов с опасными грузами.

1. Постановка динамической транспортной задачи

Решение задачи направлено на изменение маршрута движения опасных грузов (ОГ) в связи с возникновением на пути следования чрезвычайных ситуаций в целях минимизации затрат времени, материальных ресурсов, минимальных рисков и обеспечения прогнозирования развития сценариев (ЧС).

Для оценки эффективности предлагаемых решений за базовый критерий можно принять математическое ожидание $M_{<i...j>}[Y]$ количества потерь от ЧС при перевозке ОГ. Таким образом, решение динамической транспортной задачи минимизации рисков может быть сведено к минимизации математического ожидания количества потерь $M_{<i...j>}[Y]$. Данный критерий также может быть использован для оценки рисков от возможных ЧС на маршруте, представленном транспортным графом $G(V,E)$ в течение заданного интервала времени t. Таким образом, динамическая транспортная задача многокритериальной оптимизации с учетом рисков в течение заданного интервала времени t на маршруте $<i...j>$ может быть представлена следующим образом:

$$
\begin{gathered}
M_{<i...j>}[Y]=\sum_{l\in<i...j>} M_l[Y]\to\min,\\
t(<i...j>)\to\min,\\
\max P(l)\to\min,\\
l\in<i...j>,\\
M_l[Y]=\sum_{a=1}^{m}\sum_{b=1}^{k} Q^1_{ab}(\tau)Y^1_{ab}+\sum_{a=1}^{m}\sum_{b=1}^{k} Q^2_{ab}(\tau)Y^2_{ab}+\sum_{v=1}^{n} Q_V(\tau)Y_V,\\
<i...j>\in G(V,E);\\
\forall l\in E: P(l)=Q^1_{ab}+Q^2_{ab},
\end{gathered}
\qquad (1)
$$

где $a=\overline{1,m}$ – количество типов возможных ЧС: авария (a=1), несчастный случай (a=2), пожар (a=3) и т.д. – форм получения прямых и непрямых потерь ресурсов (людей, времени, различных средств); $b=\overline{1,k}$ – количество предвидимых сценариев возникновения и развития разных типов событий (как правило, с наибольшей вероятностью и наиболее тяжелыми последствиями); Q^1_{ab}, Q^2_{ab}, – вероятность возникновения в момент времени τ события конкретного вида и Y^1_{ab}, Y^2_{ab} обусловленный им размер прямых и непрямых потерь; $v=\overline{1,n}$ – число непрерывных видов ЧС и/или систематических вредных энергетических (шум, вибрация, электромагнитные излучения) и материальных (отравляющие вещества, отходы) технологических выбросов; Q_v – вероятность появления в момент времени τ каждого типа непрерывных или систематических вредных выбросов; Y_v – размер возможных от них прямых и непрямых потерь.

2. Моделирование информационной среды для поддержки принятия решений при перевозке грузов повышенной опасности

Процесс принятия решения по управлению транспортными потоками состоит из ряда фаз: стратегической, тактической и оперативной. В данной статье рассматривается оперативная фаза управления – возникновение чрезвычайной ситуации, влекущей за собой невозможность следования установленному маршруту.

Процесс принятия решений по управлению транспортными потоками в случае возникновения чрезвычайной ситуации $S_r = S(x)$, где $x=(x_1, x_2, ..., x_p)$ – количественные и качественные параметры условий транспортирования (характеристики транспорта, дороги, метеорологических условий, параметры, которые характеризуют ЧС), связан со следующими видами информации: множеством микроситуаций $\{S_i\}, i=\overline{1,m}, S_i = S(x_\alpha), \alpha=\overline{1,p}$, отражающих априорную информацию о подобных чрезвычайных ситуациях $\{S_j\}, j=\overline{1,n}$, для ликвидации которых были приняты наиболее удачные решения в смысле удовлетворения заданным критериям оценки принятых мер [1]; множеством решений $\{\gamma_\alpha\}$ $\alpha=\alpha_1, \alpha_2, ..., \alpha_i, ..., \alpha_m$ и множеством возможных сценариев возникновения и развития чрезвычайных ситуаций $\{S_j\}, j=\overline{1,n}$. Каждая совокупность $x=(x_1, x_2, ..., x_p)$ характеризует какую-либо физическую сторону микроситуации, например, маршрут, скорость движения, количество типов опасных грузов (рис.1).

Рис. 1. Основные этапы обработки информации для принятия решений

Модель оптимизации принятия решений в условиях неопределенности рисков возможных сценариев возникновения ЧС представлена формулой:

$$\min_{D(\gamma/S_i,\bar{C})} R = \min_{D(\gamma/S_i,\bar{C})} \sum_{\alpha=1}^{r} \sum_{i=1}^{r} q(S_i,\gamma_\alpha) P(\gamma_\alpha/S_i), \quad (2)$$

где $D(\gamma(\gamma_i,\bar{C})$ – структура модели, представленная набором решающих правил; $\bar{C}=(c_1,c_2,...,c_m)$ – параметры модели; $P(\gamma_\alpha/S_i)$ – вероятность принятия решения $\{\gamma_\alpha\}$ в возможном сценарии возникновения ситуации; (S_i,γ_α) – элементы повышенного риска, возникающие при принятии решения $\{\gamma_\alpha\}$ в ситуации S_k; $q(S_k)$ – априорная вероятность ситуации S_k [1].

Критерий эффективности (2) существенно зависит как от свойств признаков $\{x\}$, по которым принимаются решения $\{\gamma\}$, так и от алгоритмов принятия решений $D(\gamma/x)$.

Если бы никакой информации о признаках не поступало, то процесс принятия решений можно охарактеризовать следующими начальными рисками:

$$R^0 = \sum_{l=1}^{r} \sum_{k=1}^{r} q(S_k)\Pi(S_k,\gamma_l). \quad (3)$$

Пусть имеется в наличии совокупность информативных признаков $\{x\}=\{\bigcup_{i=1}^{N} x_i\}$, которые можно в дальнейшем использовать для распознавания возможных сценариев возникновения чрезвычайных ситуаций, и известны совместные распределения $W(x/S)$ этих признаков в различных ситуациях. Тогда разность между начальным R^0 и минимальным риском, полученным путем выбора оптимального алгоритма распознавания $D(\gamma/x)$, можно рассматривать как «ценность информации», которую несут N признаков:

$$F(\bigcup_{i=1}^{N} x_i) = R_0 - \min_{D(\gamma/\{x\})} R\{\prod q(s), W(x/S)\}. \quad (4)$$

Выбор решения γ_i в момент $t[n]$ фиксирует состояние системы в момент $t[n+1]$. Этим самым модель приобретает способность к прогнозированию. Тогда для ближайшей к эталонной микроситуации S_{et} априорная микроситуация $\{S_j\}$, $j=\overline{1,n}$ будет соответствовать наилучшему решению $\gamma(t_j)$ в момент времени t_j при $i \le j < i = n$. Модель $\bar{C}(t)$ предлагает возможный сценарий развития чрезвычайной ситуации S_k, которой соответствует $\gamma(t_j)$.

В соответствии с этим сценарием принимаются решения $\gamma_i(t_i)$ о проведении мероприятий $\Gamma_l \in \{\Gamma\}$, результатом которых возможно получение решения:

$$\{\gamma_1^*(t_j),\gamma_2^*(t_j),\ldots,\gamma_n^*(t_j)\} = \{\gamma_m^*(t_j)\} \in \{\gamma(t_j)\}.$$

Поскольку $\gamma_p(t) \in \{\gamma(t)\}$, где $p=\overline{1,n}$, то значения решения $\{\gamma_m^*(t_j)\}$, полученного на $(j-1)$-м такте экстраполяции, изменятся на $\gamma(t_{j+1})$.

При оперативном управлении состав критериев и ограничений зависит от конкретного сценария возникновения чрезвычайной ситуации S_r и выбора решения γ_r. Поэтому каждое решение γ_r полностью определяется тройкой $\gamma_r \Rightarrow <\{f_i^r\},\bar{a}^r,\{\bar{A}^r\}>$, где $\{f_i^r\}$ – множество критериев, используемых ЛПР в сценарии возникновения чрезвычайной ситуации S_r; $\bar{a}^r$ – вектор весов или приоритеты критериев; $\{\bar{A}^r\}$ – ограничения, имеющиеся в сценарии возникновения чрезвычайной ситуации S_r [3,4].

Построение вектора весов критериев $\bar{a}^r\{\overline{a_1}^r, \overline{a_2}^r, \ldots, \overline{a_i}^r\}$ в сценарии возникновения чрезвычайной ситуации S_r осуществляется с помощью алгоритма обучения. На этапе обучения ЛПР реализует процедуру преобразования, т.е. построения своей функции предпочтения в ситуации S_r: $F^r = \sum_i a_i^r f_i^r(X)$, где $\sum_i a_i^r = 1$.

Решение задачи в сценарии возникновения чрезвычайной ситуации S_r сводится к следующей задаче математического программирования: определить

$$\operatorname{extr}_X \sum_i a_i^r f_i^r(X) \text{ при } \{\overline{A}^r\} \quad . \tag{5}$$

Информационная технология, направленная на оценку рисков перевозки опасных грузов, включает в себя схему взаимодействия банка данных сценариев ЧС с базами данных (БД) при управлении перевозками опасных грузов с минимальными рисками. Множество решений $\{\gamma\}$ определяет допустимые действия над реализациями конкретной БД для преобразования её в другую реализацию, и может рассматриваться как функции, определенные на множестве состояний базы данных.

3. Построение модели ранжирования и классификации микроситуаций

Система представляет собой ориентированный граф, имитирующий дорожно-транспортную сеть, транспортное средство, перемещающееся по ней, погодные условия и объекты повышенной опасности. Множество возможных решений состоит из всевозможных вариантов скоростного режима, направлений движения внутри дуг и соответственно выбираемых дуг (если транспорт находится в точке перемены маршрута). Множество сценариев S представляет собой всевозможные конечные наборы последовательных микроситуаций, где начальной микроситуацией является нахождение транспортного средства в стартовой точке с текущими параметрами системы, а конечной, соответственно, либо нахождение в конечной точке, либо попадание в чрезвычайную микроситуацию. Чрезвычайной считается такая микроситуация, в которой уровень риска превышает допустимые значения.

Таким образом, каждой микроситуации ставится в соответствие некоторый конечный набор решений $\{\gamma\}$, каждое из которых даёт нам конечное число возможных сценариев дальнейшего развития ситуации. В соответствии с полученными сценариями выделяется оптимальное решение, которое характеризуется множеством критериев, вектором весов и ограничениями множества возможных решений во время поиска оптимального.

Сценарии и вероятности перехода из микроситуации в микроситуацию определяются либо экспертно, либо путем использования статистических данных о предыдущих маршрутах и микроситуационных последовательностях.

Вектор весов строится с помощью алгоритма обучения.

Вектор ограничений задаётся исходя из ограничений на управляемые параметры системы (такие, как скоростной режим, направление движения и тому подобные).

Выделим критерии, которыми пользуется ЛПР при выборе оптимального решения. Имеем три основных: риск чрезвычайной ситуации, риск дорожно-транспортного происшествия и риск ограбления. Все три критерия зависят от параметров системы и могут быть вычислены для каждой микроситуации и, соответственно, для каждого решения. Помимо параметров системы в вычислении критериев используются априорные вероятности всех трёх видов рисков в виде интервальных оценок [5].

Поскольку решается задача $\operatorname{extr}_X \sum_i a_i^r f_i^r(X)$ при $\{\overline{A}^r\}$, рассмотрим детально вектор критериев. Обозначим критерии f_A, f_E, f_R для ДТП, чрезвычайной ситуации и ограбления соответственно. Априорные вероятности – Q_A, Q_E, Q_R соответственно. Вектор значений погодных переменных обозначим $\overline{Wt}$, кумулятивный коэффициент посещений, отвечающий за статистику прохождения транспортом данной точки K_R, время t и пройденный

путь обозначим s. Кроме того, введём понятие объекта ЧС. Данный тип объектов будет представлять собой некоторую зону, внутри которой он будет оказывать влияние на параметры микроситуаций согласно экспертно определённой функции в зависимости от собственной специфики и от априорных параметров микроситуации. Обозначим его E.

Таким образом, мы описали все факторы, которые будут формировать отношение предпочтения для множества решений в каждой из микроситуаций.

Каждая микроситуация имеет набор параметров, значимость их зависит от конкретного сценария возникновения ЧС выбора решений S_r. Перечислим их: качество дорожного покрытия, сложность участка дороги, наличие объектов повышенной опасности, время суток, вектор погодных условий (температура, давление, влажность и тому подобные), окружающий рельеф, уровень преступности, загруженность трассы, видимость, состояние дорожного покрытия. Эти параметры различным образом связаны друг с другом и используются для расчёта критериев.

Находясь в микроситуации S_r, мы решаем следующую задачу математического программирования:

$$\underset{\{\gamma_r\}}{\text{extr}} = \alpha_A f^r{}_A(\overline{\beta_A} \times (Q_A, E, t, s, \overline{Wt})) + \alpha_E f^r{}_E(\overline{\beta_E} \times (\overline{Wt}, E, Q_E)) + \alpha_R f^r{}_R(\overline{\beta_R}(Q_R, K_R)) \quad (6)$$

при ограничениях $\{\overline{A^r}\}$. Здесь $\overline{\beta_A}, \overline{\beta_E}, \overline{\beta_R}$ – векторы весов факторов.

Декомпозировав факторы до параметров, получаем следующую задачу:

$$\underset{\{\gamma_r\}}{\text{extr}} = \overline{\alpha}(f_A(\beta_A(Q_A, OHD, E, \overline{Wt}, Rd, t, s)), f_E(Q_E, OHD, E, \overline{Wt})), f_R(\beta_R(Q_R, Hist, CR)). \quad (7)$$

Здесь Rd, Rq – сложность участка дороги и качество дорожного покрытия соответственно; OHD – объекты повышенной опасности; Hist – история посещений; CR – уровень преступности.

Таким образом, мы выделили критерии выбора решений и конкретизировали задачу (6) до задачи (7).

Введем на множестве микроситуаций метрику. Пусть $\overline{Par}$ есть вектор всех значимых параметров из (7). Тогда расстояние между двумя микроситуациями может быть получено как взвешенная с вектором весов $\overline{M}$ евклидова метрика между векторами их параметров. Исходя из метрики, можно построить множество экстремальных микроситуаций и оценить степень чрезвычайности каждой конкретной ситуации на основании расстояния между рассматриваемой и эталонной экстремальной ситуацией, что поможет ограничить область поиска решений на более ранних этапах.

4. Прогнозирование рисков развития ЧС в транспортной сети

Для решения задачи прогнозирования рисков предлагается рассмотреть транспортную сеть с потенциально опасными участками. В качестве математической модели такой сети логично использовать граф, заданный матрицей инцидентности. Для упрощения модели будем считать все дуги двунаправленными. Вероятность возникновения опасной ситуации (ОС) не зависит от направления движения по дуге.

Предполагается, что априорные вероятности возникновения ОС, рассматриваемых автономно, известны или могут быть вычислены путем статистического анализа имеющихся данных. Таким образом, перед ЛПР ставится задача оценки дополнительной вероятности возникновения опасных ситуаций, рассматриваемых как результат возникновения других ЧС.

Информационная технология расчета рисков возникновения ЧС при транспортировке грузов повышенной опасности для транспортной сети QN_i включает в себя следующие шаги.

Шаг 1. Получение обобщенных данных по грузопотокам. Эти данные включают в себя:

– объемы входных и выходных потоков по конечным элементам транспортной сети (источники и получатели грузов) за единицу времени. Для получения достоверного прогноза рекомендуется брать этапы не меньше 1 года;

– матрицу инцидентности сети;

– маршруты транспортировки грузов либо список узлов, обслуживание которых включает в себя элементы повышенного риска.

Шаг 2. Расчет фиктивных потоков по каждой дуге сети на основании данных, полученных на шаге 1. На этом этапе производится оценка – может ли транспортная сеть при данной конфигурации обеспечить надежное обслуживание/транспортировку грузов.

Шаг 3. Формирование списка возможных сценариев ЧС для входного и выходного грузопотока каждого узла сети.

Далее, для каждого из возможных сценариев возникновения ЧС, каждого маршрута и каждой дуги производятся следующие расчеты согласно шагам 4-13.

Шаг 4. Получение характеристик транспортного потока:

m – количество типов ОГ в выходном потоке;

n – количество типов ОГ во входном потоке;

mc – количество известных сценариев возникновения ЧС в выходном потоке;

nc – количество известных сценариев возникновения ЧС во входном потоке.

Обращаем внимание, что $m \le mc, n \le nc$, так как одному грузу может соответствовать более одного сценария возникновения ЧС: $N_1 \ldots N_m$ – объемы перевозок соответствующих типов ОГ в единицу времени в выходном потоке; $M_1 \ldots M_n$ – объемы перевозок соответствующих типов ОГ в единицу времени во входном потоке.

Шаг 5. Получение либо расчет на основе статистики априорных вероятностей возникновения ЧС для различных грузов в выходном ($QN_1 \ldots N_{mc}$) и входном ($QM_1 \ldots QM_{nc}$) потоках.

Шаг 6. Получение статистически значимых параметров транспортной сети. Значимость параметров может определяться исходя из специфики задачи прогнозирования. В целях упрощения задачи будем рассматривать только универсальные для любой задачи характеристики:

– L – протяженность транспортной сети;

– $V_1 \ldots V_m$ – средние скорости перевозки соответствующих грузов в выходном потоке;

– $W_1 \ldots W_n$ – скорости перевозки грузов во входном потоке.

Шаг 7. Для каждого опасного груза рассчитываются скорости его перемещения в окрестности моделируемого узла для:

– элементов k - го типа ОГ в выходном потоке: $\tau_k^{вых} = \frac{L}{V_k}, k=1 \ldots m$;

– элементов l -го типа ОГ во входном потоке: $\tau_l^{вых} = \frac{L}{W_l}, l=1 \ldots n$.

Шаг 8. Построение прямой и обратной связи «ОГ – сценарий ЧС» для выходного и входного потоков: $k=1 \ldots m$ – индекс типов ОГ выходного потока; $R(k)$ – множество индексов сценариев в результате ЧС с грузами k -го типа выходного потока ($1 \le R(k) \le mc$); $l=1 \ldots n$ – индекс типов ОГ входного потока; $R(l)$ – множество индексов сценариев в результате ЧС с грузами l -го типа входного потока ($1 \le R(l) \le nc$); $i=1 \ldots mc$ – индекс сценариев (вариантов развития ЧС) выходного потока; $X(i)$ – индекс типа опасного груза, вызывающего i -й сценарий ЧС в выходном потоке ($1 \le X(i) \le m$); $j=1 \ldots nc$ – индекс сценариев (вариантов развития ЧС) входного потока; $Y(j)$ – индекс типа опасного груза, вызывающего j -й сценарий ЧС в выходном потоке($1 \le Y(j) \le n$).

Шаг 9. Расчет плотности ОГ для имеющихся потоков (под плотностью будем понимать среднее количество элементов ОГ, находящихся на дуге)

$$n_k = \frac{N_k * \tau_k^{вых}}{T_ч}.$$

Таким образом, подсчитывается среднее количество элементов ОГ k-го типа выходного потока, одномоментно находящегося на дуге, $k=1\ldots m$, $m_l=\frac{M_l*\tau_l^{вх}}{T_ч}$.

Аналогично оценивается среднее количество элементов ОГ l-го типа входного потока, одномоментно находящегося на дуге, $l=1\ldots n$; $T_ч$ – количество часов в принятой единице измерения времени. Для обеспечения эффективного измерения рекомендуется проводить анализ статистики за срок не менее одного года.

Шаг 10. Расчет вероятности возникновения i-го типа ЧС в выходном потоке в результате возникновения ЧС j-го типа в одном из элементов входного потока (каскадирование чрезвычайных ситуаций, возникновение ЧС на одном из элементов маршрута приводит к каскадному нарушению маршрута):

$$\Omega_{ij}=2QN_j\int_0^{t_{max}}P_{ij}(S(t)/V_i)dt,$$

где $t_{max}=\frac{l_{ij}^{max}}{V_{X(i)}+W_{Y(j)}}$ – время нахождения в зоне взаимовлияния до встречи элементов встречных потоков; l_{ij}^{max} – предельное расстояние взаимовлияния i-го типа ЧС в выходном потоке в при появлении во входном потоке ЧС j-го типа; $S(t)=l_{ij}^{max}-(V_{X(i)}+W_{Y(j)})t$, $S(t)$ – расстояние между элементами в конкретный момент времени; P_{ij} – условная вероятность возникновения i-го типа ЧС при возникновении ЧС j-го типа на расстоянии $S(t)$ между ними.

Шаг 11. Оценка вероятности возникновения ЧС i-го типа в элементе входного потока производится аналогично шагу 10.

Шаг 12. Расчет вероятности возникновения ЧС i-го типа в выходном потоке вследствие возникновения группы (серии) ЧС во входном потоке: $P_i^{каск}=\sum_{j=1}^{n}\sum_{j\in S(j)}2m_{Y(j)}\Omega_{ij}$, где $S(j)$ – множество сценариев ЧС, соответствующих текущему элементу j входного потока.

Шаг 13. Определение уточненной вероятности возникновения ЧС i-го типа для конкретного выходного потока:

$$QN_i^*=QN_i+\sum_{j=1}^{n}\sum_{j\in S(j)}2m_{Y(j)}\cdot 2QN_{ij}\int_0^{\frac{l_{ij}}{V_{X(i)}+W_{Y(j)}}}P_{ij}(S(t)/V_i)dt,$$

где QN_i – априорная вероятность, полученная путем статистического анализа; $\sum_{j=1}^{n}\sum_{j\in S(j)}$ – суммирование по всем сценариям развития ЧС; $2m_{Y(j)}t$ – среднее количество элементов входного потока с соответствующими j-ми сценариями реализации опасных факторов

Априорная вероятность QN_i может содержать неопределенности различных видов. Это обуславливается неполнотой знаний, погрешностью измерений, малым интервалом ретроспективного наблюдения. Вследствие этого будем предполагать, что элемент QN_i в реальной практике оценивается неравенством: $QN_i^{min}\le QN_i\le QN_i^{max}$ и вырождается из интервальной вероятности в детерминированное значение при $QN_i^{min}=QN_i^{max}$. Необходимо заметить, что распределение значений внутри этого интервала может варьироваться и может характеризоваться: статистической информацией (мат. ожидание, дисперсия, функция плотности распределения вероятности); нечеткой информацией в виде принадлежности функции к некоторому нечеткому множеству; отсутствием подобной информации (интервальное значение).

Применение интервального подхода позволяет оценить не только «среднюю» вероятность наступления ЧС, но и разброс оценок данной вероятности. Эта информация может быть учтена системой для повышения точности прогнозирования.

5.Оценка эффективности

Для оценки еффективности подобных процедур была проведена проверка моделей. Для оценки эффективности моделей использовалась векторная карта дороги, которая имела 25 отдельных элементов, каждый имел примерно 50000 узлов (перекрестков). Каждые связанные два узла имеют информацию о типе дорожного покрытия (асфальтобетон, многорядный автобан, автострада, асфальтированная дорога, грунтовая дорога). При этом учитывается количество чрезвычайных ситуаций на каждом узле, освещенность в течение суток, средняя скорость транспортных средств. Были сформулированы несколько тестовых задач:

1. Для тестирования стратегического принятия решения при перевозке опасного груза оценивался оптимальный маршрут протяженностью около 2000 км.
2. Построение оптимального маршрута при перевозке опасного груза на расстоянии около 20 км (тактическое принятие решения).
3. Оперативное изменение маршрута при возникновении чрезвычайной ситуации на пути следования для задачи 2.

Риск на пути следования оценивался с учетом экспертных оценок влияния факторов, которые можно выделить для данной карты (линейная зависимость на величину риска). Для задач 1 и 2 основным количественным критерием потерь при их решении было отклонение от предлагаемой наилучшей длины на 20%, а максимальная величина риска на пути следования – не больше заданного значения. Эталонным было взято решение группы экспертов (согласование нескольких экспертов не рассматривалось). Результаты испытания представлены в таблице.

Критерий	Экспертный результат	Предлагаемые модели	Номер задачи
Длина маршрута	2078 км	2107 км	1
Максимальная величина риска на пути следования	0.0875	0.0841	1
Длина маршрута	20100 м	18400 м	2
Максимальная величина риска	0.1015	0.0872	2

Наибольшую эффективность предлагаемые модели дают для тактического управления при построении маршрутов в условиях большого количества пересечений и наличия альтернатив. При построении маршрутов для первой задачи эффективность модели почти совпадает с экспертными решениями. Это объясняется тем, что при большой протяженности маршрута (около 2000 км и более) граф маршрута имеет мало альтернатив и оценки вариантов близки. Безусловно, что риск экспоненциально растет при большом числе альтернатив и экспертная оценка имеет больше ошибок. Для оперативного управления необходимо практически мгновенно принимать решения. Если сравнивать модель многокритериальной оптимизации и «муравьиноподобную модель» (при количестве итераций – 2000) [2], то результаты поиска оптимального маршрута будут близкими (рис. 2).

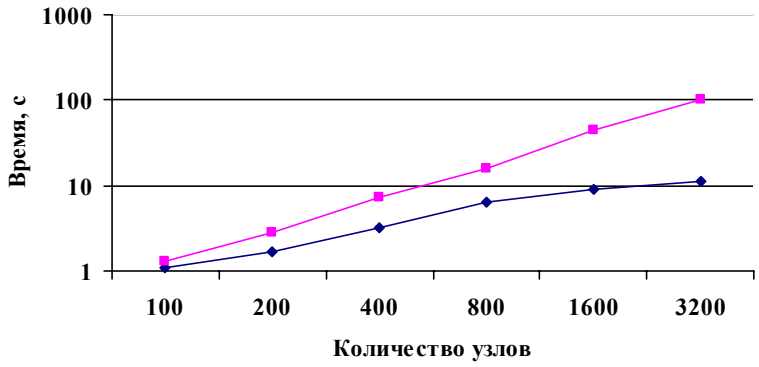


Рис. 2. Поиск оптимального маршрута

Выводы

В работе впервые выполнена постановка динамической транспортной задачи управления относительно принудительного изменения запланированного маршрута в случае возникновения чрезвычайной ситуации на маршруте транспортировки опасных грузов; для ее решения предложена модель представления транспортной сети, модель принятия решений и модель с использованием аппарата микроситуаций в комбинации с «муравьиноподобной моделью», которые обеспечили при минимальных рисках оптимизацию решения, с минимальными затратами времени и материальных ресурсов для условий неопределенности рисков возможных сценариев, что позволило реализовать прогнозирование развития сценариев ситуаций и сократить время принятия управляющих тактических или стратегических решений относительно изменения маршрута.

Впервые предложен метод оперативного поиска управляющих решений по изменению маршрута, учитывающий на начальном этапе истории подобных чрезвычайных ситуаций в виде множества априорных микроситуаций, и реализует на следующем этапе активно управляемую эволюционную стратегию с направленной эволюцией в комбинации с «муравьиноподибной моделью», что позволило обеспечитьпринятия более эффективных оперативных решений по управлению перевозкой опасных грузов за минимальное время.

Получил дальнейшее развитие метод расчета рисков возникновения чрезвычайных ситуаций при транспортировке опасных грузов, который в отличие от существующих предлагает на первом этапе учитывать обобщенные данные по грузопотоках, в следующем - конфигурацию транспортной сети, геоинформационную привязку, что позволяет рассчитать безопасную скорость движения, количество опасных грузов и вероятность возникновения «каскадного» расширение чрезвычайных ситуаций.

Список литературы: 1. *Кузёмин А.Я., Фастова Д.В., Дяченко О.Н.* Геоинформационная система для классификации и прогнозирования лавинной опасности // АСУ и приборы автоматики. 2006. Вип. 136. С. 87-91. **2.** *Кузёмин А.Я, Левыкин В.М.* Разработка инструментальных средств обеспечения принятия решений для предупреждения и управления в чрезвычайных природных ситуациях // АСУ и приборы автоматики. 2007. №139. С. 31-38. **3.** *Кузёмин А.Я.* Обобщённая модель и прогнозирование чрезвычайных природных ситуаций // АСУ и приборы автоматики. Харьков, 2007. №141. С. 21-29. **4.** *Kuzemin A., Sorochan M., Yanchevkiy I., Torojev A.* The use of situation representation when searching for solutions in computer aided design systems International Journal // Information Theories & Applications. Bulgaria. 2005. Vol. 11, №1 P. 101-107. **5.** *Kuzemin A., Fastova D., Yanchevsky I.* Methods of adaptive extraction and analysis of knowledge of knowledge-base construction and fast decision making // International Journal on Information Theories & Applications. Bulgaria. 2005. Vol.12. №1. P. 93 – 99.

Поступила в редколлегию 11.02.2011

Даюб Ясер, аспирант кафедры информатики ХНУРЭ. Научные интересы: системный анализ, оптимизация. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 702-15-15.

УДК 658.52.011.56

А.И. КОВАЛЕВ

ПОСТРОЕНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННЫМИ ЗАКАЗАМИ НА ОСНОВЕ АРХИТЕКТУРЫ ОПЕРАЦИОННЫХ СИСТЕМ. ЧАСТЬ 1

Экспертная система управления заказами в логистической сети предприятия как система программной поддержки гибкого производства реализует архитектуру операционных систем. В работе описываются функции, требования к управлению и построение системы. Управление заказами включает одноразовые операции (состав заказов, набор операций, блок управления заказами, разбивка заказов по участкам/службам предприятия) и многоразовые операции (запуск, приостановка, блокирование, разблокирование заказов), а также планирование и взаимодействие заказов. В первой части рассматривается менеджмент заказов, программная поддержка процесса управления заказами, экспертная система управления заказами и ее функции.

1. Постановка задачи

Основная задача современной производственной стратегии заключается в поиске возможностей сочетать гибкость предприятия, выпускающего продукцию на заказ, со стоимостными преимуществами, характерными для сборочных линий и непрерывного производства. В настоящее время такое сочетание возможно только в условиях полной автоматизации производственной системы. С этой целью на современных предприятиях применяются системы гибкого автоматизированного производства. Чтобы обеспечить бесперебойную работу таких систем, в них используются сложнейшие системы автоматизированного управления [1]. Совершенствование и, одновременно, упрощение таких систем является актуальной научно-технической задачей. В настоящей работе представлена основа автоматизированной системы управления, использующая методы искусственного интеллекта и архитектуру операционных компьютерных систем.

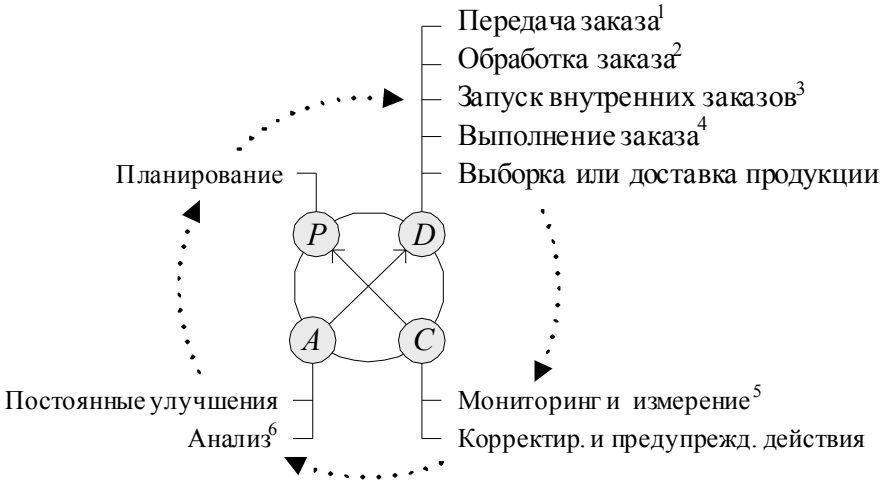
2. Менеджмент заказов

Процессы связи с потребителями, включающие, (см. ДСТУ ISO 9001): а) информацию о продукции; б) прохождение запросов, контракта (заказа), включая поправки; в) обратную связь с потребителями, являются объектами изучения в логистике. Управление заказами можно рассматривать как первую ключевую логистическую активность, так как с приема и обработки заказов начинается логистический менеджмент. Общий процесс управления заказами разделяют на два подпроцесса [2]: 1) управление заказами в логистической сети предприятия (т.е. внешних заказов, в дальнейшей терминологии *заказы-родители*); 2) управление внутренними заказами (в дальнейшей терминологии *заказы-дети*). Управление заказами в логистической сети предприятия не связано непосредственно с изготовлением готовой продукции, но способствует более эффективной реализации процедур выполнения заказов. Общую процедуру выполнения заказов представляют в виде функционального или логистического цикла, включающего несколько этапов (рисунок).

Результаты мониторинга используются для проверки соответствия целям долгосрочного планирования. Результаты анализа применяются для совершенствования выполнения заказов-родителей.

Логистический цикл характеризуется такими оперативными показателями: 1) скорость; 2) бесперебойность; 3) гибкость; 4) уровень брака / устранение недостатков [3]. *Скорость* прохождения логистического цикла измеряется временем от получения заказа до его исполнения (доставки потребителю). Хотя скорость обслуживания, несомненно, играет важную роль, но не меньшее, если не большее значение имеет бесперебойность операций. *Бесперебойность* означает способность предприятия придерживаться ожидаемых сроков исполнения заказа на протяжении многих логистических циклов, т.е. это – постоянное соблюдение условий поставок на протяжении длительного времени. Бесперебойность представляет собой ключевое качество логистики. *Гибкость* означает способность предприя-

тия удовлетворять исключительные запросы заказчиков. Компетентность предприятия в логистике непосредственно связана с тем, насколько успешно оно справляется с неожиданными обстоятельствами. Показатель (отношение) *уровень брака / устранение недостатков* предусматривает возможность срывов и недостатков, а следовательно, и особые действия, направленные на их исправление. Тем самым такие планы, разработанные с осознанием того факта, что ни один план не может полностью исключить сбоев в текущей деятельности, гарантируют высокий уровень сервиса.



Этапы логистического цикла выполнения заказов (*PDCA*: *Plan-Do-Check-Akt*): [1] Ряд действий, которые осуществляются с момента, когда заказчик размещает (посылает) заказ, и до момента, когда поставщик его получает. [2] Процедура приема заказа и трансформации требований заказчика применительно к условиям производителя. [3] Заказы на закупку материалов, инструмента, на проектирование и изготовление технологической оснастки (если применимо); заказы изготовителям-смежникам для выполнения нужных работ. [4] Время выполнения заказа включает время ожидания постановки заказа на выполнение и время его выполнения, складывающееся из технологического времени, времени межоперационных простоев. [5] Постоянная прослеживаемость, измерение оценочных показателей процесса. Частота измерений показателей должна обеспечивать своевременную идентификацию изменений этих показателей. [6] Деятельность, предпринимаемая для установления пригодности, адекватности, результативности процедуры выполнения заказов

3. Программная поддержка процесса управления заказами

Оказалось возможным соприкосновение стратегии производства «на заказ» с вычислительными алгоритмами. Стандарт *MRP II* (планирование ресурсов производства) позволил развить такую технологию, ориентированную на применение корпоративных информационных систем. Эта технология, в том числе, предоставляет возможность осуществлять моделирование в целях ответа на вопросы типа «что будет, если …» [4]. Интерактивность систем на базе стандарта *MRP II* обеспечивается заложенным в него блоком моделирования. Другими словами, существует возможность проигрывания вероятных ситуаций на предмет исследования их влияния на результаты деятельности предприятия. Такая методология управления производством может быть отнесена к системам поддержки принятия решений, так как позволяет просчитывать последствия, хотя и не выдает никаких практических вариантов преодоления возникающих проблем.

Новые технологии, описываемые комплексом стандартов и рекомендаций *CSRP* (планирование ресурсов, синхронизированное с заказчиком), призваны удовлетворить индивидуальные нужды и ожидания, отвечать на эти нужды товарами, которые предоставляют уникальную ценность для каждого заказчика [5]. Концепция *CSRP* предусматривает запись специфических требований к продукту и преобразование их в детальные инструкции по производству и планированию. Создается список материалов и комплектующих для производства, автоматически определяются производственные маршруты, материалы планируются и заказываются и, наконец, создается рабочий заказ.

Однако в стратегии производства «на заказ» существуют процедуры, которые требуют реализации интеллектуальных функций человека (диспетчера), не описываемых алгоритмически. Это – удовлетворение исключительных запросов заказчиков, действия в условиях срывов и недостатков. Это также процедуры, характерные для универсального производства, в том числе, в следующих случаях:

– оптимальная переналадка технологического оборудования в целях его эффективного использования при производстве разнородной продукции;

– при одновременной обработке многих партий или даже единиц изделий необходимо контролировать исполнение и приоритет каждого конкретного заказа;

– оптимизация использования технологического оборудования при существенных различиях его загрузки по времени;

– координирование заказов и ресурсов (материал, персонал, оснастка).

К интеллектуальным задачам, возлагаемым на диспетчера, относят [6]: 1) распознавание ситуации; 2) подготовка решения в сложных ситуациях, например, в случае неожиданных обстоятельств или особых действий; 3) утверждение решений. Имеется возможность описания таких процессов, проходящих в неопределенных внешних условиях – в виде системы объектов и прерываний (заимствовано из методов программирования в компьютерной области [7]). В данном случае процесс можно рассматривать как ответные действия программы на события, определяемые человеком. В указанном подходе объектно-событийного описания процессов рассматриваются объекты, участвующие в процессе (заказы), и события, которые порождают действия, вносящие изменения в объекты. Этот метод полезен в тех случаях, когда известны действия (возможности) процесса, но не известно, какие именно действия и в какой последовательности необходимо будет совершить для получения ожидаемого результата.

Для программной поддержки процесса управления заказами, способной решать неформальные задачи, накапливать опыт, обучаться, надо создать базу знаний для экспертной системы. В работе [8] отмечается необходимость создания новых знаний (интеллектуальных активов), в которых нуждаются организации, чтобы выполнять индивидуальные, отличающиеся друг от друга заказы потребителей. Указан механизм создания интеллектуального капитала – наблюдение за работой своих уникальных специалистов и обобщение их опыта. Наблюдения за работой настоящих мастеров своего дела обобщаются в виде инструмента, делающего доступным их опыт сотруднику предприятия, который обладает средней квалификацией.

Таким инструментом может быть экспертная система. Экспертные системы содержат объекты, их свойства, правила, механизмы вывода. Рассматриваемая предметная область – логистика, управление заказами. Чтобы создать экспертную систему в предметной области, собираются свойства объектов (данные). Подвергнутые обработке и удобно представленные данные есть информация. Эта информация, характеризующая состояние объектов, заносится в базу знаний и ее можно использовать для интеллектуальной поддержки принятия решений по управлению заказами.

Правила и механизм вывода вместе составляют механизм планирования выполнения заказов (МПВЗ). МПВЗ использует эвристические правила и его можно назвать эвристическим механизмом планирования. Главное отличие эвристических правил: они формулируются не на основании обычных, признанных знаний, а на основании практических знаний экспертов. Эвристика (эвристическое правило, эвристический метод) представляет собой некоторое произвольное правило, стратегию, хитрость, упрощение или любое другое средство, которое ограничивает объем поиска решений [9]. Большинство задач, с которыми успешно справляются экспертные системы, являются эвристическими по своей природе – они требуют использования эмпирических правил для получения приемлемого решения.

Экспертная система предполагает способность и готовность быстро снабжать потребителей точной информацией о текущей логистической деятельности и прочих обстоятельствах. Еще одним важным критерием качества обслуживания, которое может обеспечить экспертная система, является непрерывное совершенствование (в смысле достижения оперативных целей с наименьшим уровнем брака). Один из способов добиться этого –

учиться на допущенных ошибках и постоянно совершенствовать экспертную систему, чтобы избежать их повторения в будущем.

4. Экспертная система управления заказами (ЭСУЗ)

Существенные особенности ЭСУЗ как системы программной поддержки процесса управления заказами [10]:

– *Реализация защитных механизмов*. Заказы не должны иметь самостоятельного доступа к распределению ресурсов, т.е. не должны выполняться заказы, использующие ресурсы несанкционированно.

– *Наличие прерываний*. Внешние прерывания оповещают ЭСУЗ о том, что произошло асинхронное событие, связанное с выполнением другого заказа. Внутренние прерывания возникают, когда выполнение программы привело к исключительной ситуации, требующей вмешательства диспетчера, например, обращение к отсутствующим ресурсам: нет результатов химического анализа или входного контроля закупленного материала; нет расчета потребности в материалах, самого материала (не указана замена); нет необходимых станков (вышли из строя, не указаны другие возможности, например, внешний субподрядчик, способный выполнить нужную работу).

Исключительные ситуации можно разделить на исправимые и неисправимые. К исправимым (оперативные ошибки) относят такие исключительные ситуации, как отсутствие нужного расчета материала, данных химического контроля электролитического раствора и т.п. После устранения причины исправимой исключительной ситуации можно продолжить выполнение заказа. Возникновение в процессе работы ЭСУЗ исправимых исключительных ситуаций является нормальным явлением. Неисправимые исключительные ситуации обычно возникают в результате ошибок в планировании.

– *Интерфейс* между пользователями (диспетчер, начальники производственных цехов и участков[1], заказчики) в ЭСУЗ организован при помощи набора системных вызовов (кроме прерываний).

– *Организация очереди* из заказов – планирование.

– *Сохранение структуры данных*. Для переключения работы с одного заказа на другой возникает потребность в сохранении содержимого структуры данных, необходимых для выполнения заказа (для обеспечения правильности продолжения работы).

– *Стратегия управления ресурсами*: упорядочивание процедур покупки, замены и выборки материалов и комплектующих со складов; изменений в персонале, оборудовании. В целях чтения, создания и удаления записей о ресурсах (персонал, рабочие смены, материалы, оборудование, оснастка, др.) должны быть предусмотрены соответствующие системные вызовы для работы в интерактивном режиме с уполномоченными людьми. Ресурсы организовывают в виде каталогов: основной (ресурсы предприятия), каталоги производственных участков, склада материалов и комплектующих изделий.

– *Удобство применения*. Возможность одновременной обработки многих заказов одной компьютерной системой (ЭСУЗ) должна обеспечивать пользователям – начальникам производственных участков легко и эффективно контролировать заказы в интерактивном режиме; диспетчеру – управлять всеми заказами предприятия; заказчикам – возможность «следить» за заказом, определять его фазу.

5. Функции ЭСУЗ

Необходимость в правильном функционировании в процессе управления заказами возникает потому, что в распоряжении диспетчера имеется несколько методов решения. Это администрирование должно быть таким, чтобы метод, являющийся (по некоторому эвристическому критерию) наиболее обещающим, использовался в первую очередь [9]. Термин «администрирование» говорит об иерархической структуре системы решения задач, которая в вычислительных машинах обеспечивается операционной системой. Поэтому экспертную систему управления заказами можно рассматривать как операционную систему, управляющую (в интерактивном режиме) основными действиями пользователей и обеспечивающую запуск заказов, а также взаимодействие с пользователями. В частности, ЭСУЗ выполняет следующие функции:

---

1 Далее – участков

– управление памятью: создание банков данных, которые можно запрашивать с различными целевыми установками* ;

– управление запуском заказов (приоритетность, прерывания, блокирование);

– управление оборудованием и оснасткой (мониторинг наличия, работоспособного состояния, ремонта);

– учет использования сырьевых материалов;

– управление взаимодействием заказов;

– диспетчеризация доступа (ЭСУЗ посредничает при всех обращениях субъектов к объектам);

– диспетчеризация заказов (контроль графика выполнения работ на участках и своевременного выпуска продукции);

– диспетчеризация материально-технического обеспечения, в том числе, контроль выполнения планов снабжения и выдача рекомендаций для принятия мер в случае их нарушения;

– контроль за приемкой продукции.

Во всех перечисленных функциях эвристический метод управления полезен при установлении порядка, который, будучи не обязательно оптимальным, с большой вероятностью должен оказаться намного лучше, чем случайно выбранный порядок. В теории искусственного интеллекта известна основная эвристика обучения, которая формулируется так: в новой ситуации попытайся использовать методы, подобные тем, которые лучше всего работали в аналогичных уже известных ситуациях. Основная эвристика обучения кажется очевидной, однако ее применение связано с выбором подходящего объективного критерия подобия задач (а следовательно, и подобия ситуаций), а также подобия решений. Эта проблема далеко не тривиальна и решается, как правило, эвристически. Обучение – одна из существенных особенностей экспертной системы. Под обучением в контексте искусственного интеллекта понимается не столько накопление данных, сколько приобретение и накопление на опыте необходимых навыков пользователями ЭСУЗ.

Список литературы: 1. *Чейз Ричард Б.* Производственный и операционный менеджмент / Ричард Б. Чейз, Николас Дж. Эквилайн, Роберт Ф. Якобс /Пер. с англ. М.: Издательский дом «Вильямс», 2004. 704 с. 2. *Сергеев В.И.* Логистика в бизнесе: Учебник / В.И. Сергеев. М: ИНФРА-М, 2001. 608 с. **3.** *Бауэрсокс Доналд Дж.* Логистика. Интегрированная цепь поставок / Доналд Дж. Баурсокс, Дейвид Дж. Клосс; пер. с англ. М.: ЗАО «Олимп – Бизнес», 2008. 640 с. **4.** *Гаврилов Д.А.* Управление производством на базе стандартов MRP II / Д.А. Гаврилов. СПб.: Питер, 2008. 416 с. **5.** *Колесников С.Н.* Стратегии бизнеса: управление ресурсами и запасами / С.Н. Колесников. М.: Статус-Кво 97, 2000. 245 с. **6.** *Дружинин В.В.* Системотехника / В.В. Дружинин, Д.С. Канторов. М.: Радио и связь, 1985. 200 с. **7.** *Галеев В.И.* Кухня процессного подхода / В.И. Галлеев, К.В. Пичугин // Методы менеджмента качества. 2003. №4. С.12-21. 8. *Коулсон-Томас К.* Качество в обществе, основанном на знаниях / К. Коулсон-Томас // Европейское качество. 2003. №1-2. С. 30-49. **9.** *Эндрю А.* Искусственный интеллект / А. Эндрю /Пер. с англ. под ред. и с предисл. Д.А. Поспелова. М.: Мир, 1985. 264 с. 10. *Карпов В.Е.* Введение в операционные системы [Электронный ресурс] / В.Е. Карпов, К.А. Коньков, В.П. Иванников. 2001-2003. Режим доступа: *http:/ cs.mipt.ru/docs/courses/osstud/os.html.*

Поступила в редколлегию 28.02.2011

Ковалев Алексей Иванович, канд. техн. наук, начальник отдела управления проектами ОАО ЭК «Хмельницкоблэнерго». Научные интересы: управление предприятиями, оценивание деятельности, международные стандарты на системы управления, качество управления, информационные системы. Адрес: Украина, 29016, Хмельницкий, ул. Храновского 11а, тел.: (0382) 78-78-23.

---

* Например, статистические данные о заказах могут использоваться для планирования загрузки мощностей, планирования работы отделов, а также для планирования инновационной деятельности предприятия (разработка новых технологий, закупка оборудования).

УДК 658.512.011:681.326:519.713

TIECOURA YVES, С.В. ЧУМАЧЕНКО, В.И. ХАХАНОВ

АКТУАЛЬНЫЕ ПРОБЛЕМЫ АНАЛИЗА КИБЕРПРОСТРАНСТВА

Предлагаются процесс-модели получения квазиоптимального детерминированного многозначного решения в n-мерном логическом векторном кибернетическом пространстве. Рассматриваются теоретические основы векторных логических вычислений для получения решения, оцениваемого неарифметическим критерием качества взаимодействия объектов в пространстве. Формулируются проблемы создания индивидуального киберпространства, распознавания образов, принятия решений, вакцинации программных продуктов, актуальные для рынка информационных технологий. Общность метрики оценивания качества решения на основе xor-операции объединяет достаточно разнородные по составу проблемы анализа информации.

1. Введение

Киберпространство (cyberspace) – метафорическая абстракция, используемая в философии и в компьютерах, – является виртуальной реальностью, которая представляет Ноосферу или Второй мир «внутри» компьютеров и сетей. Это определение взято из Википедии и согласуется с многочисленными публикациями, авторами которых в абсолютном большинстве являются философы. Возможно, уже настало время взглянуть на определение и сущность киберпространства с позиции технологической и математической культуры. Основная аксиома, заложенная в определении, – виртуальность пространства.

Виртуальность (virtus — потенциальный, возможный) – вымышленный, воображаемый объект или явление, не присутствующий в реальном мире, а созданный игрой воображения человеческой мысли, либо смоделированный при помощи других объектов. Реальные компьютеры создают «нереальное пространство», а точнее, Второй мир, который может и хочет стать первым. Не хватает малого – целенаправленного самосовершенствования компьютерного сообщества, способного на равных с человеком участвовать в создании и формировании киберпространства. Информация – реальность и сущность пространства, компьютер – ее носитель или форма.

Киберпространство – совокупность взаимосвязанных компонентов в виде информационных процессов или явлений с носителем в форме компьютера, взаимодействие которых между собой оценивается соответствующей метрикой.

Метрика – способ измерения расстояния в пространстве процессов или явлений. Что такое расстояние в киберпространстве? Самое простое есть самое верное, это – кодовое расстояние по Хэммингу между парой векторов, обозначающих процесс или явление. Производная (булева) – степень изменения или различия двух объектов. Киберпространство хорошо тем, что понятие близости объектов означает, насколько они отличаются друг от друга. Как измерить различия? Сравнить – применить xor-операцию, проще не бывает. Жизнь в киберпространстве уже имеет свои законы. Сохранение своей индивидуальности важно не только в Первом мире. Клоны только с первого взгляда есть хорошее дело в обоих мирах. Но если человека размножить, то прообраз просто потеряется в своих копиях. Примитивно, но мало кто пожелает делить славу, почет, деньги с незаслужившими это копиями. Но хорошо делить ответственность за деструктивные действия оригинала. Аналогично, копировать и продавать клоны своего продукта есть путь к богатству. Но это может сделать и не автор продукта, продавая нелицензионные копии, по сути, занимаясь воровством. В пределе – клонирование есть плохо в обоих мирах. Альтернатива уже сейчас прослеживается в киберпространстве, называемая, возможно, не совсем удачно, облачными вычислениями. Благодаря развитию телекоммуникаций, можно утверждать, что все объекты (субъекты) в киберпространстве находятся рядом в индивидуальном компьютере. Это означает, что необходимо иметь программу (функциональность) в одном экземпляре, которая доступна всему миру, первому и второму благодаря облачной технологии. Защитить одну копию – продукт на собственном сервере – намного проще задача,

чем сделать это для сотен и тысяч клонов. Аргументом в пользу появления облачных технологий является несанкционированное снятие защиты с лицензионной копии программного продукта и дальнейшее его использование, в том числе и на продажу. Востребованность продукта есть успех в бизнесе, здесь все как в первом мире – востребованность индивидуума – основа его состоятельности, признания, богатства. Таким образом, можно сделать некоторые выводы. 1) Второй мир стремится к повторению Первого – клонов не будет, но будет доступность каждого продукта, как компонента киберпространства для всех субъектов обоих миров. 2) Каждый субъект Первого мира будет иметь в идеале один образ (сайт) во Втором мире. 3) Аутентичность – желаемый идеал для всех конструктивных (законопослушных) субъектов обоих миров. 4) Сайт не олицетворяет сущности образа, который должен быть для индивидуума (прообраза) больше, чем отражение в зеркале. Это – история, действительность, метрические и медицинские данные, привычки, бизнес, быт, отдых, будущее, контакты, пристрастия. 5) Такой образ – индивидуальное киберпространство – должно быть сверхнадежно закрыто от несанкционированного доступа. 6) Защищать от деструктивных компонентов и действий нужно будет не компьютер, а субъект (программный продукт) киберпространства на основе предлагаемой технологии вакцинации. Повысить иммунитет субъекта за счет внедрения в программу агента – вакцинной избыточности, взаимодействующей с сервером Лаборатории Касперского (ЛК) – значит сохранить работоспособность продукта. Сервер ЛК генерирует облако услуг по защите субъектов киберпространства, которое в пределе может покрыть всю цифровую планету. 7) Расстояние (производная) – бинарное отношение объектов в киберпространстве есть степень изменения, различия в процессах или явлениях, определяемое с помощью (векторной) xor-операции. Производную можно применять ко всему, что шевелится или нет. Лишь бы было выполнено единственное условие – бинарность (два состояния в процессе, два объекта в пространстве, два явления в природе, два адреса, две части целого). Равно как и утверждение – сравнивать (измерять, соизмерять, оценивать, распознавать) можно все, что угодно, лишь бы этого было не менее двух. Семантический изоморфизм глагола «сравнивать» можно далее распространять на действия: принимать решение, тестировать, диагностировать, идентифицировать, управлять, прогнозировать,

Резюме – нужен мощный мультиматричный процессор (ММП), где каждая команда (and, or, xor, slc) обрабатывает параллельно и предельно быстро только одну бинарную операцию на матрицах (двумерные массивы данных). Количество командно-ориентированных матриц-примитивов создает систему – гетерогенный мультиматричный процессор бинарных операций с буфером M (рис. 1).

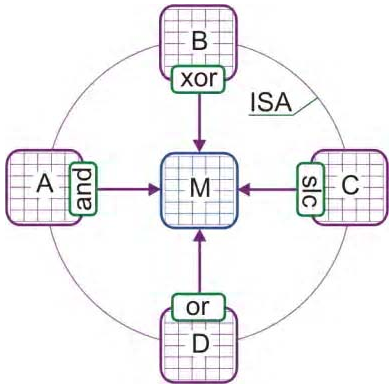


Рис. 1. Мультиматричный процессор бинарных операций

Мультиматричный модуль процессора включает 4 блока памяти со встроенными в них операциями (A – and, B – xor, C – slc – shift left crowding, D – or) и буферную память M. Модуль ориентирован на параллельное выполнение в данном случае одной из четырех инструкций (ISA – Instruction Set Architecture), оперирующих матрицами двоичных данных одинаковой размерности: $M = M\{and, or, xor, slc\}\{A, B, C, D\}$ с занесением результата в буфер M. Особенность ММП в том, что не ячейка матрицы имеет систему команд из

четырех операций, а каждая команда имеет собственную матрицу ячеек в качестве данных для параллельной обработки, что существенно упрощает структуру управления и устройства в целом. Вся сложность ММП перенесена на структуры данных, где память матрицы имеет одну аппаратно-реализованную встроенную команду, что позволяет иметь примитивную систему управления параллельными вычислительными процессами (SIMD – Single Instruction Multiple Data), последовательностную по своей сути, а значит, нет необходимости создавать сверхсложные компиляторы, ориентированные на распараллеливание вычислительных процессов.

Здесь каждый матричный процессор выполняет одну операцию, встроенную в запоминающие элементы матрицы. Но возникают ситуации, когда матричный уровень (M-level) задания данных избыточен для выполнения, например операций над булевыми (B-level) или регистровыми (R-level) переменными. Для такого случая необходимо иметь иерархию по уровням представления данных, что иллюстрируется рис. 2.

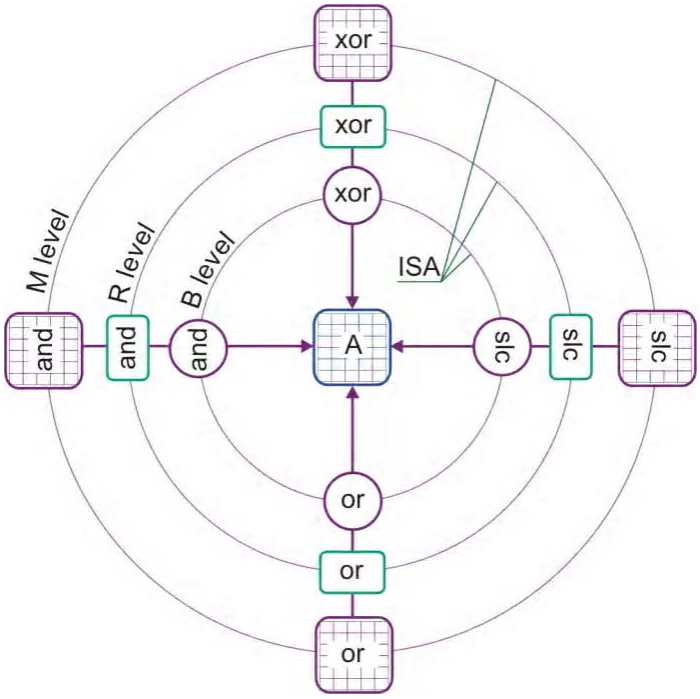


Рис. 2. Мультиуровневый по данным процессор

Здесь система команд (ISA) одна и та же, но операнды – различные по информационному объему, буфер A={B,R,M} представлен компонентами всех уровней иерархии. Мультиуровневая структура данных (Hierarchical SIMD) эффективна только для регистровых и матричных вычислений (переменных), где наблюдается высокий уровень параллельной обработки данных. Формат элементарной вычислительной процедуры для одной из операций имеет вид: $A = A\,\{\text{and}, \text{or}, \text{xor}, \text{slc}\}\,\{B, R, M\}$.

Источники: 1. Актуальные задачи анализа объектов в киберпространстве [1-3] 2. Метрика киберпространства [4, 5]. 3. Аппаратура и матричные процессоры [6-13, 14-17, 18-28].

2. Транзитивное замыкание дружественного для пользователя пространства

Многообразие сетевых структур (глобальные, корпоративные, социальные, профессиональные, технологические, культурные, бытовые) доставляет не только позитивное влияние на качество работы и отдыха, но и сильнейший прессинг каждому пользователю со стороны киберпространства. Вывод напрашивается сам собой. Необходимо создавать дружественное индивидуальное киберпространство (ИКП) с фильтрами, настроенными на повышение качества жизни каждого человека. Вход в киберпространство должен быть инвариантным по отношению к средству общения (компьютер, мобильный телефон, смар-

тфон, планшет, Apple iPad, Samsung Galaxy S, Apple MacBook Air, Logitech Revue, Google Nexus One (HTC Desire), Apple iPhone 4, Apple TV, Toshiba Libretto W100, Microsoft Kinect, Nook Color). Это означает, что индивидуальное пространство в будущем не должно быть связано с персональным компьютером и зависимо от него. ИКП позиционируется как внешний компонент по отношению к средству управления вычислительными процессами и отображения информации (monitored remote control). При этом вычислительные процессы реализуются на мощных серверах, а пользователь имеет технологию облачных вычислений для всех сфер человеческой деятельности. Такое упрощение функций персонального компьютера и необходимость создания внешнего индивидуального пространства, подключенного к вычислительным ресурсам планеты, ставит новые проблемы разработки средств защиты и сервисного обслуживания ИКП путем проектирования фильтров, структур данных, библиотек и технологий измерения взаимодействия процессов и явлений в киберпространстве. В частности, средства защиты информации следует создавать не только для программных и аппаратных продуктов, компьютеров и сетей, но и для ИКП.

Цель – существенное повышение качества сервисов, доставляемых со стороны программных, аппаратных изделий, сетевых структур и уменьшение стоимости эксплуатационных расходов за счет создания стандартизованных инфраструктур сервисного обслуживания, обеспечивающих дружественную эксплуатацию, тестирование и устранение функциональных нарушений.

Задачи: 1) Разработка математического аппарата анализа кибернетического пространства, ориентированного на создание моделей и методов сервисного обслуживания программных и аппаратных продуктов. 2) Создание процесс-моделей и критериев взаимодействия вредоносных компонентов с программными кодами полезных функциональностей. 3) Описание актуальных для рынка проблем, подлежащих решению в векторном логическом пространстве, определяемом бета-метрикой взаимодействия объектов.

Общность задач синтеза и анализа основана на использовании равенства нулю триады равноценных компонентов, соединенных операцией xor $m \oplus A \oplus Q = 0$, формулирующей условия позитивного решения проблемы. Здесь первый компонент m – входные условия, второй A – эталонные модели процесса или явления, третий Q – результат взаимодействия первых двух, который может вырождаться в критерий качества отношения или принятия решения, оценку распознавания объектов или образов. В общем случае третий компонент Q есть критерий качества решения многих задач синтеза или анализа, имеющий единый с первыми двумя компонентами формат данных – вектор или матрицу. При синтезе входные условия m есть спецификация, а модель A представлена элементами, которые создают условия покрытия специфицированной функциональности. При этом полнота покрытия оценивается третьим компонентом Q, исполняющим роль критерия качества. Следует помнить, что xor-триада, равная нулю, есть условие или метрика оценивания взаимодействия объектов в киберпространстве. В то время как стратегия, технология, модель или алгоритм вычислительных процессов, приводящих к заданной цели, есть самостоятельная проблема для каждой области знаний, которая в данном случае остается за кадром. Аналогично можно использовать триадное уравнение для тестирования знаний студентов, равно как и для создания валидных тестовых заданий (T – тест, S – объект или студент):

$$T \oplus S \oplus Q = 0;\ T \oplus S_i = \min_i Q_i \to x \oplus m \oplus Q = 0,$$

$$T \oplus S_1 = Q_1 \approx \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \oplus \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & 1 \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & 1 & 1 \\ . & 1 & . & 1 \\ 1 & . & . & . \end{bmatrix},$$

$$T \oplus S_2 = Q_2 \approx \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \oplus \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & 1 \\ . & 1 & . & 1 \\ . & . & . & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & 1 & 1 \\ . & . & . & . \\ 1 & . & . & . \end{bmatrix},$$

$$T \oplus S_3 = Q_3 \approx \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \oplus \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & 1 & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix}.$$

В медицине процесс идентификации нарушений сводится к сравнению эталона органа или организма с существующими состояниями упомянутых компонентов конкретного человека. В результате формируется критерий качества, указывающий на критические места организма, подлежащие коррекции:

$$T \oplus S = Q \approx \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \oplus \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & 1 \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & 1 & 1 \\ . & 1 & . & 1 \\ 1 & . & . & . \end{bmatrix}.$$

Восстановление зафиксированных функциональных сбоев возможно и путем воздействия на органы человека через каналы связи информационными символами или образами: пищевыми, тактильными, звуковыми, вербальными, вкусовыми, обонятельными, зрительными. Возможность такой коррекции основана на влиянии (позитивном или негативном) всех указанных компонентов на здоровье человека в процессе жизненного цикла. К 40 годам каждый человек в силу своих привычек формирует собственную карту входных переменных, которые создают, как правило, функциональные нарушения отдельных органов. Путем сравнения реальной карты переменных человека с идеальной для него можно выработать формулу коррекции нарушений и предложить далее более приемлемую карту (библиотеку) переменных, отражающую позитивные и негативные для организма входные воздействия. Второй компонент библиотеки необходим для создания специальных фильтров, ставящих под запрет доставку человеку деструктивной для него информации. Любое входное воздействие m сравнивается с фильтрами позитивных и негативных воздействий $A = A^p \vee A^n$. Критерий качества определяет инверсную функцию принадлежности входного воздействия к одному из фильтров:

$$m \oplus A = Q;\ A = A^p \vee A^n;$$

$$m = \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \oplus (A^p \vee A^n) = \left(\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & 1 \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix} \vee \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & . & . & . \\ 1 & . & . & . \end{bmatrix} \right) = Q(A^p \vee A^n) = \left(\begin{bmatrix} . & . & . & . \\ . & . & 1 & 1 \\ . & 1 & . & 1 \\ 1 & . & . & . \end{bmatrix} \vee \begin{bmatrix} . & . & . & . \\ . & . & . & . \\ . & 1 & . & 1 \\ . & . & . & 1 \end{bmatrix} \right).$$

В данном случае входное воздействие является негативным для индивидуума, поскольку $Q(A^p) \succ Q(A^n)$. Предложенная модель должна стать частью индивидуального кибернетического пространства или компьютера-друга, который, прежде всего, должен ежедневно заботиться о состоянии организма путем наблюдения и управления его поведением с целью выработать привычки относительно правильных входных переменных и корректировать возникающие несоответствия относительно идеальной карты, уникальной для каждого человека, состояния организма. Доказательством возможности коррекции состояния организма может служить эмоциональная зависимость от той информации, которую ему навязывают дома, на работе, в транспорте, по Интернету, телевидению и печати. Информация (образ) может исцелить (известную идею разделяет доктор В.А.Садырин) или убить человека, возвысить или опустить, активизировать или сделать его пассивным. Очень важен авторитет источника, от которого исходит информация. Одни и те же слова по-разному воспринимаются человеком, если они произносятся двумя персонами, имеющими для приемника различный рейтинг. Отсюда вывод – авторитетный для пациента врач уже исцеляет его своим образом и словами без медикаментозных препаратов. Для социума, группы людей, человека существует система позитивных и негативных образов, символов и знаков, которые необходимо найти для создания индивидуального компьютера, способного поддерживать человека на высоком уровне творческой и физической активности путем торможения отношения к деструктивам и активизации – при наличии позитивных образов. Все перечисленное уже имеется в распоряжении спецслужб и психоаналитиков, которые могут манипулировать человеком для решения не всегда гуманистических проблем. Цель, актуальная на рынке сегодня, – сделать каждого человека планеты более счастливым путем информационной коррекции его отношения к действительности, с помощью индивидуального киберпространства или компьютера. Создать дружественный уголок киберпространства лучше с позиции его инвариантности по отношению к типам интер-

фейса <человек–пространство>, которые могут быть представлены любым доступным прибором: компьютер, мобильный телефон, смартфон. Даже при утере физического интерфейса (гаджета) киберпространство каждого индивидуума сохраняется. Как построить такой приватный уголок? Для этого необходимо создать изменяющуюся в пространстве и во времени идеальную компьютерную модель (коррекции) поведения человека, которая способна к самосовершенствованию в процессе жизненного цикла индивидуума, постоянно изучая поведение последнего на уровне всех возможных каналов его реакций на окружающую действительность. Кто формирует здоровье нации, группы людей и человека? Не только пища и экология, и не столько перечисленные компоненты. Но авторитеты для каждого индивидуума: руководитель страны, города, предприятия, учреждения, отдела, группы. Чем ближе к индивидууму и выше рейтинг руководителя, тем сильнее его негативное влияние на подчиненного отрицательными образами, словами, символами. Не сотвори себе кумира! Равно как и наоборот, позитивный знак или образ приумножают силы и активность подчиненных на решение сложных задач. С другой стороны, каждый из нас ежедневно должен заботиться о своем имидже успешного, креативного, жизнерадостного и здорового человека, который своим образом, приятным для окружающих, должен порождать в них аналогичные качества.

3. Технология вакцинации программных продуктов

Востребованной на рынке является решение проблемы создания теории, методов и архитектуры параллельного анализа информации, представленной в виде аналитических, графовых и табличных форм ассоциативных отношений для поиска, распознавания, диагностирования деструктивных компонентов и принятия решений в n-мерном векторном дискретном пространстве. Здесь целесообразно использовать векторно-логические процесс-модели актуальных прикладных задач, в том числе – диагностирование вирусов и восстановление работоспособности программно-аппаратных компонентов компьютерных систем, качество решения которых оценивается неарифметической метрикой взаимодействия булевых векторов.

Решение проблемы ориентировано на поиск, распознавание, диагностирование деструктивных компонентов аппаратно-программными методами в дискретном кибернетическом пространстве. Общность представленной теории синтеза и анализа кибернетического пространства основана на использовании равенства нулю триады равноценных компонентов, соединенных операцией xor $m \oplus A \oplus Q = 0$, формулирующей условия решения проблемы. Здесь первый компонент m – входной код программы, второй A – эталонные модели деструктивов, третий Q – результат взаимодействия первых двух, который может вырождаться в критерий качества отношения или принятия решения, оценку распознавания объектов или образов.

Цель – существенное повышение качества программных продуктов и уменьшение стоимости эксплуатационных расходов за счет их вакцинации путем введения в код встроенной программной избыточности в виде инфраструктуры сервисного обслуживания, обеспечивающей тестирование, диагностирование и устранение вредоносных компонентов, классифицируемых в библиотеках.

Объект исследования – кибернетическое пространство, представленное информацией, ее носителями и преобразователями, а также деструктивными компонентами, наносящими вред функциональностям, улучшающим качество жизни человека.

Субъект исследования – инфраструктура сервисного обслуживания в виде встроенной программной избыточности, работающей в реальном масштабе времени, обеспечивающей тестирование, диагностирование и устранение вредоносных компонентов, описанных в соответствующих библиотеках.

Мотивация: 1) Отсутствие на рынке антивирусной защиты встроенных средств тестирования, диагностирования и удаления вредоносных компонентов, составляющих инфраструктуру сервисного обслуживания, подобной тому, как в цифровых системах на кристаллах существуют стандарты граничного сканирования, а в программных продуктах – ассерционная избыточность, ориентированные на встроенное тестирование дефектов и ошибок с последующим восстановлением работоспособности аппаратных или программных изде-

лий. 2) Наличие теоретических разработок, связанных с технологией алгеброгологического векторного анализа информационных данных, ориентированных на высокое быстродействие решения и оценивания задач распознавания образов, принятия решений и тестирования объектов [1-9]. 3) Наличие образцовой производственной и маркетинговой инфраструктуры (Лаборатория Касперского), способной поддержать проект создания технологии вакцинации программных продуктов и авторитетно предложить его рынку информационных технологий. 4) Миниатюризация цифровых и телекоммуникационных систем (телефоны, смартфоны, IP-фоны, планшеты) требует постоянной защиты от несанкционированного доступа путем внедрения встроенных антивирусных средств, контролирующих информационный обмен.

Задачи: 1) Разработка математического аппарата анализа кибернетического пространства, ориентированного на создание моделей и методов сервисного обслуживания программных продуктов для тестирования, диагностирования и устранения вредоносных компонентов. 2) Создание типовых процесс-моделей и критериев взаимодействия вредоносных компонентов с программными кодами полезных функциональностей. 3) Разработка технологии анализа структуры программного кода для определения критических точек и установки в них ассерционных операторов наблюдения и управления в процессе его функционирования. 4) Создание инфраструктуры сервисного обслуживания функциональных программ для встроенного тестирования, диагностирования и устранения вредоносных компонентов из программного кода функциональности на основе использования библиотеки деструктивных элементов. 5) Тестирование и верификация встроенной инфраструктуры сервисного обслуживания функциональностей, защищающей программный код от вредоносных компонентов.

Ожидаемые результаты и их рыночная привлекательность: 1) Инфраструктура встроенной защиты программного кода от несанкционированной модификации, приводящей к изменению функциональности. 2) Избыточность инфраструктуры программного кода, которая автоматически синтезируется на стадии проектирования и верификации, составляет не более 5% от специфицированной функциональности. 3) Рыночная привлекательность инфраструктуры, определяемая многообразием программных продуктов, умноженной на уровень продаж каждого изделия, составляет в год порядка одного миллиарда экземпляров. 4) Стоимость создания инфраструктуры для программного продукта составляет 5–10% затрат от разработки функционального кода. Если уровень продаж – не менее 500 копий, то затраты на создание встроенного антивируса вполне окупаемы в течение года. 5) Внедрение запатентованной технологии вакцинации программных продуктов при их рождении может принести компании порядка миллиарда долларов. 6) Маркетинговая проблема глобальной компании (Лаборатория Касперского) заключается в убеждении разработчиков программных продуктов имплементировать существующие внешние антивирусы вовнутрь кода полезной функциональности.

4. Инфраструктура программного продукта

Проблема создания эффективной инфраструктуры кибернетического пространства (Cyber Space), а также саморазвивающейся информационно-компьютерной экосистемы (ИКЭС) планеты особенно важна для глобальных компаний, таких как Лаборатория Касперского, Google, Microsoft. Экосистема – совокупность популяций, взаимодействующих между собой и окружающей их средой неопределённо долгое время, имеющая сходство относительно протекающих в них энергетических (информационных) процессов.

Кибернетическое пространство как объект природы также подвержено влиянию деструктивных компонентов, влияющих на работоспособность субъектов, которыми являются компьютеры, системы и сети. Поэтому сейчас и в будущем важной проблемой остается стандартизация пространства и специализация всех взаимодействующих субъектов, включая негативные, как неотъемлемую часть экосистемы. Данная акция есть постоянно действующая во времени, цель которой – не отставать, но на один шаг опережать появление новых вредоносных компонентов. путем создания инфраструктуры кибернетического пространства, обеспечивающей функционирование компьютерной экосистемы планеты и качество жизни каждого человека, включающей следующие модули:

1) Наблюдение за состоянием внутренних и выходных линий компьютерных систем в процессе функционирования, верификации и тестирования штатных функциональных блоков на основе использования стандартов граничного сканирования.

2) Тестирование функциональных модулей путем подачи проверяющих наборов от различных тестовых генераторов, ориентированных на проверку вредоносов или исправного поведения.

3) Диагностирование отказов и вредоносов путем анализа информации, полученной на стадии тестирования и использования специальных методов встроенного поиска деструктивов на основе стандартов граничного сканирования или ассерционной избыточности, ориентированной на обнаружение вредоносного кода, что позволит идентифицировать и устранять деструктивы программных продуктов без использования внешних средств. Таким образом, можно будет обходиться без сложных внешних программ моделирования, тестирования и диагностирования путем прививки каждого программного изделия тестопригодной интеллектуальной избыточностью кода на стадии его создания. При этом следует использовать предикат узнавания буквы a: $x^a = x \oplus a$, который оперирует не только булевыми, но регистровыми и матричными переменными, что делает его практически значимым в формальной записи уравнений диагноза или распознавания:

$$x^a \approx x \oplus a = 0 \vee \min_i Q_i \to x \oplus a \oplus Q = 0;$$

$$x^m \approx x \oplus m = 0 \vee \min_i Q_i \to x \oplus m \oplus Q = 0;$$

$$T \oplus S = Q \approx \begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}\begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & . & . \\ . & 1 & . & 1 \\ 1 & . & . & 1 \end{bmatrix} \Delta \begin{bmatrix} 1 & . & 1 & 1 \\ . & 1 & 1 & 1 \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & 1 & 1 \\ . & 1 & . & 1 \\ 1 & . & . & . \end{bmatrix}.$$

На основе предиката узнавания m-образа любой сложности, природы и формы можно создавать достаточно компактные уравнения предикатов, формирующие интеллектуальные решения в области распознавания образов, принятий решений, тестирования знаний и технических объектов, диагностирования (узнавания) вирусных компонентов в программном коде.

4) Восстановление работоспособности функциональных модулей после фиксации отрицательного результата тестирования и определения места и вида вредоноса при выполнении фазы диагностирования внутренними средствами, создающими инфраструктуру функциональности.

5) Измерение сигнатуры, основных характеристик и параметров функционирования изделия на основе встроенных средств, позволяющих производить временные и функциональные измерения для идентификации состояния программного продукта.

6) Надежность и отказоустойчивость функционирования изделия в процессе эксплуатации, которая достигается диверсификацией функциональных блоков, их дублированием и восстановлением работоспособности в реальном масштабе времени, а также встроенными средствами сервисного обслуживания, работающими в реальном масштабе времени.

В связи с этим предложенная инфраструктура кибернетического пространства, метрика его измерения и процесс-модели анализа и синтеза субъектов дают возможность создавать эффективные решения компьютерных изделий, ориентированных на быстрый поиск, распознавание, диагностирование не только позитивных, но и негативных субъектов. Конкретно, предложенная инфраструктура может решать задачи: 1) Описание многообразия деструктивных компонентов кибернетического пространства. 2) Формализация процессов взаимодействия триады компонентов <программа, деструктивности, тесты>. 3) Диагностирование и устранение деструктивных компонентов. 4) Создание и эффективное использование базы деструктивных данных. 5) Создание быстродействующих интеллектуальных саморазвивающихся средств сервисного обслуживания и защиты кибернетического пространства.

Инфраструктура верификации и тестирования проектов цифровых систем и программных продуктов есть актуальная проблема для рынка информационных технологий. Ее

комплексное решение, основанное на наличии исходной информации в качестве спецификации, включает следующие пункты: 1. Синтез транзакционного графа (ТГ) HDL-кода, где вершины есть примитивы или структуры памяти, а дуги – операторы или строки HDL-кода. 2. Оценивание тестопригодности проекта на основе анализа ТГ и определение критических точек, покрываемых механизмом ассерций для наблюдения (управления) процесса верификации. 3. Оценивание диагностируемости проекта по ТГ, как количественной характеристики глубины поиска функциональных нарушений (ФН) в коде HDL-модели. Улучшение оценки диагностируемости путем введения дополнительных ассерций, позволяющих довести упомянутую характеристику до требований пользователя. 4. Определение функционального покрытия (functional coverage) для синтеза теста и оценивания его качества в виде полноты покрытия. 5. Синтез таблицы функциональных нарушений (ТФН), как матрицы контрдостижимостей вершин и дуг, задающих структуры памяти HDL-кода и блоки операторов соответственно, проверяемых на тестовых сегментах. 6. Создание методов и средств диагностирования ФН на основе векторного параллельного анализа строк и столбцов ТФН. 7. Разработка методов и средств восстановления работоспособности HDL-кода за счет введения избыточности в проект в виде ТГ, механизма ассерций и использования моделей и методов, направленных на: синтез ТГ, оценивание тестопригодности и диагностируемости, построение ТФН, анализ ТФН для поиска функциональных несоответствий.

5. Проблема оценивания качества решения

1. Можно ли уйти от численной оценки качества двух или нескольких проектов (решений)? Ответ позитивный. Процесс-модель поиска оценки лучшего решения (рис. 3) имеет следующие пункты. 1) Первоначально в вектор-результат Q, в котором будет сохранено лучшее решение, заносятся единичные значения во все координаты (худшее решение) и одновременно осуществляется операция slc сдвига влево и уплотнения всех единиц текущего вектора Q_i. 2) Затем выполняется сравнение двух векторов: Q и очередной оценки Q_i из списка решений. 3) После этого реализуется векторная операция and $(Q \wedge Q_i)$, результат которой сравнивается с содержимым вектора Q, что дает возможность изменить его, если вектор Q_i имеет меньшее число единичных значений. 4) Процедура поиска оценки лучшего решения повторяется n раз.

$$Q = Q\overline{(\vee((Q \wedge Q_i) \oplus Q))} \vee Q_i(\vee((Q \wedge Q_i) \oplus Q));$$

$$Y = \vee((Q \wedge Q_i) \oplus Q);$$

$$Q = Q\overline{Y} \vee Q_i Y.$$

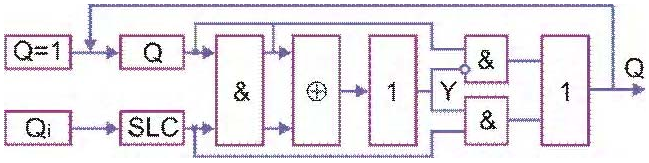


Рис. 3. Процесс-модель выбора решения

2. Можно ли найти замену фаззи-логике, чтобы уйти от численного интервала и арифметических операций? Исходя из сущности фаззи-логики, как гибрида чисел в интервале [0,1] и логических (теоретико-множественных) операций, можно предложить более рациональную алгебраическую структуру, в которой нет «тяжеловесных» арифметических операций и чисел. Лингвистическая или логическая переменная определена любым фиксированным количеством градаций, кратным степени двойки (2, 4, 8, 16, 32, 64,…). Логические операции над лингвистическими переменными определяются многозначными таблицами или диаграммой Хассе. Последняя дает возможность ранжировать все состояния лингвистической переменной для определения операций пересечения и объединения в виде быстрого поиска по графу минимума или максимума функций принадлежности, заданных кодами-состояниями двух любых вершин. Для определения функции принадлежности некоторого входного компонента необходимо иметь эталон модели, относительно которой путем сравнения формируется степень соответствия объекта эталону. Вместо численного интервала [0,1]

для задания функции принадлежности используется вектор двоичных или многозначных переменных:

$$\mu(m \in A) = (a_1, a_2, ..., a_i, ..., a_n), a_i = \{0,1\} \vee \{\alpha_1, \alpha_2, ..., \alpha_i, ..., \alpha_k\}.$$

Вектор может вырождаться в одну переменную, равно как и усложняться до матричной структуры, если она есть технологичная форма для описания эталона и входного объекта. Таким образом, функция принадлежности должна быть идентичной по структуре анализируемым компонентам. Нуль-вектор (-матрица, -переменная) будет означать полное совпадение входного объекта с эталоном.

Пример 1. Имеются три подмножества температур: C = {9, 10, 11}, W = {11, 12, 13}, H={13, 14, 15}. Определить функции принадлежности для входного подмножества m = {10,11,12}. Подмножества представлены тремя векторами:

$$C = (1110000),\ W = (0011100),\ H = (0000111).$$

Функции принадлежности определяются выполнением xor-операции над входным вектором $m = (0111000)$ и тремя эталонами:

$$\mu(m \in C) = m \oplus C = (0111000) \oplus (1110000) = (1001000);$$
$$\mu(m \in W) = m \oplus W = (0111000) \oplus (0011100) = (0100100);$$
$$\mu(m \in H) = m \oplus H = (0111000) \oplus (0000111) = (0111111).$$

Для выбора лучшего решения следует использовать схему, представленную на рис. 3, которая путем попарного сравнения определит, что минимальную оценку имеет $\mu(m \in C)$, как первое решение, которое будет встречено в процессе вычислений, из двух оптимальных $\mu(m \in C), \mu(m \in W)$, имеющих одинаковое и минимальное число несовпадений или единиц, равное двум. Кроме того, векторная оценка принадлежности дает еще дополнительную информацию – по каким координатам или переменным произошло несовпадение, что полностью скрыто в скалярном интервальном оценивании функций принадлежности.

Если считать достоинством непустое пересечение значений лингвистических переменных, которое позволяет относить результат к двум или нескольким подмножествам, то для целей определения принадлежности результата логической операции к одному из заданных подмножеств можно использовать диаграмму Хассе, как идеальный инструмент выполнения теоретико-множественных операций пересечения и объединения. На рис. 4 представлена диаграмма, содержащая 4 примитива: 00, 01, 10, 11. Здесь нет «тяжелых» арифметических операций, но только логические, которые имеют высокое быстродействие, что является несомненным преимуществом использования диаграммы для определения результата и принадлежности входного объекта к одному из заданных подмножеств.

Пример 2. Построить диаграмму карьерной иерархии. Для этого используется четыре непересекающихся примитива культур: эстетическая, математическая, технологическая и социальная: $A = \{A_e = Q, A_m = E, A_t = H, A_s = J\}$. Все виды теоретико-множественного взаимодействия данных примитивов представлены диаграммой Хассе на рис. 4, которые задают 4 уровня состоятельности человека. Каждая вершина есть лингвистическая переменная, содержащая от нуля до четырех значений. Вычисление операций (объединения, пересечения) над любыми двумя вершинами или символами сводится к нахождению ближайшей общей вершины (верхнего, нижнего) уровня. Определение функции принадлежности сводится к вычислению площади взаимодействия между m и примитивами: $\mu(m \in A) = m \oplus \{A_e, A_m, A_t, A_s\}$ или m и производными от примитивов – любой вершиной диаграммы Хассе.

Для перевода теоретико-множественной модели в плоскость булевой векторной алгебры, где основные операции есть and, or, not, необходимо выполнить кодирование всех многозначных символов двоичными векторами:

∅	Q	E	H	J	O	I	A	B	S	P	C	F	L	V	Y
0000	1000	0100	0010	0001	1010	0101	1100	0011	1001	0110	0111	1011	1101	0111	1111

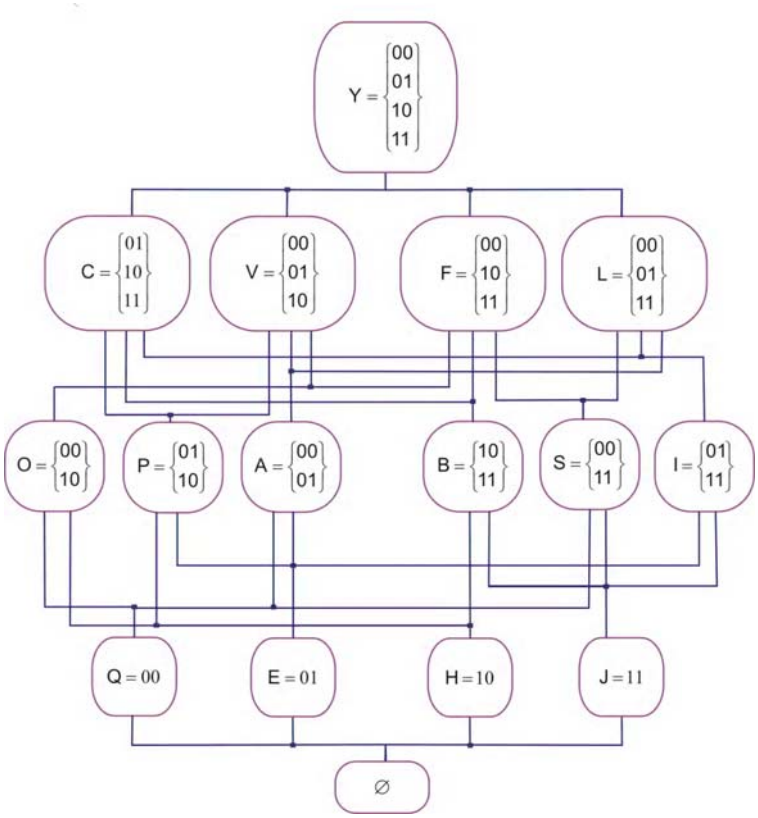


Рис. 4. Диаграмма Хассе для четырех примитивов

Алгебра векторной логики дает возможность существенно повысить быстродействие выполнения операций по сравнению с теоретико-множественными, которые технологически трудно сделать векторными и параллельными процедурами. На кодах символов шестнадцатеричного алфавита выполняются все операции алгебры логики, относительно которых кодовый алфавит является замкнутым, как и исходный теоретико-множественный булеан. В общем случае вектор объекта $A=(A_1,A_2,...,A_i,...,A_k)$ киберпространства может содержать разнородные лингвистические переменные, каждая из которых может быть адекватно и однозначно представлена вектором из n переменных в логическом булевом пространстве $n=\log_2|A_i|$. Для понимания человеком лингвистические переменные, заданные в многозначном алфавите, более предпочтительны. Однако все вычислительные процессы необходимо ориентировать на векторы-коды значений упомянутых переменных в булевом пространстве.

Мощность универсума примитивов определяется моделью проблемы, подлежащей решению. Для универсума, содержащего 2 примитива, оптимальная структура будет представлена в виде четырехзначной диаграммы $A=\{A_m=0, A_t=1, A_x=X=\{0,1\}, \varnothing\}$, изображенной на рис. 5.

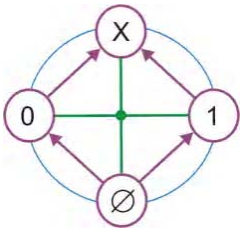


Рис. 5. Диаграмма Хассе для двух примитивов

Восьмизначный алфавит на универсуме из трех элементов представлен следующим множеством всех подмножеств: $A=\{\varnothing,\{a\},\{b\},\{c\},\{a,b\},\{a,c\},\{b,c\},\{a,b,c\}\}$, которому соответствует диаграмма, представленная на рис. 6.

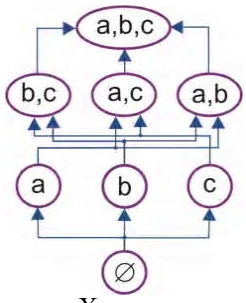


Рис. 6. Диаграмма Хассе для трех примитивов

Такая диаграмма может моделировать процессы, связанные с определением принадлежности входного воздействия одному из компонентов библиотеки. Например, заданы три перекрывающихся интервала роста человека, названные: низкий (1,2,3), средний (3,4,5), высокий (5,6,7). Векторное задание каждого из интервалов на универсуме из 7 двоичных переменных Y=(1,2,3,4,5,6,7) имеет следующий вид: $A_1 = (1110000)$; $A_2 = (0011100)$; $A_3 = (0000111)$. Полная модель логического взаимодействия данных примитивов относительно операций объединения и пересечения определяется трехпримитивной диаграммой Хассе (см. рис. 4), где представлены все возможные (a, b, c, ab, bc) и невозможные (ac, abc) сочетания для решения задачи вычисления принадлежности входного слова m одному из восьми состояний вершины. Чтобы вычислить функции принадлежности m=(1101001) для трех примитивов графа Хассе, нужно применить операцию xor:

$$\mu(m \in A_1) = m \oplus A_1 = (1101001) \oplus (1110000) = (0011001);$$

$$\mu(m \in A_2) = m \oplus A_2 = (1101001) \oplus (0011100) = (1110101);$$

$$\mu(m \in A_3) = m \oplus A_3 = (1101001) \oplus (0000111) = (1101110).$$

Вывод: входной сигнал или условие m имеет минимум различий с объектом под номером 1.

Таким образом, функция принадлежности может быть идентифицирована скалярной величиной или одной переменной, градуированной мощностью примитивных символов диаграммы Хассе; одномерным вектором двоичных или многозначных переменных; двумерной матрицей двоичных или многозначных переменных. Если площадь взаимодействия между m и A формирует функцию принадлежности $\mu(m \in A) = m \oplus A$, то некоторые варианты принадлежности не могут быть адекватно представлены интервальной метрикой. Например, рис. 7 имеет скалярную неразличимость в интервальной арифметике первого и третьего, второго и четвертого вариантов отношений между m и A.

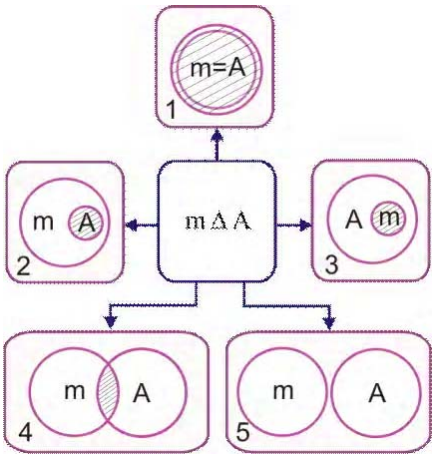


Рис. 7. Типы взаимодействия двух объектов

6. Распознавание образов на основе xor-операции

Если разделить синтаксис и семантику постановки технологической задачи, то многообразие интересных для практики проблем можно свести к универсальной в некотором смысле модели или форме. Так, множество задач можно привести к модели треугольника, когда обобщенная структура представляется уравнением (входные образы, эталоны, критерии сходства): $m \oplus A \oplus Q = 0$, где по любым двум известным компонентам следует найти третий. В формате триады компонентов модели описания процесса или явления простое решение задачи распознавания образов может быть получено, если воспользоваться вектором переменных $m = (m_1, m_2, ..., m_i, ..., m_n)$, где каждая из них $m_i \in B(Y), Y = (Q = 00, E = 01, H = 10, J = 11)$ определена, в общем случае, более чем двумя значениями или булеаном на универсуме примитивов B(Y) = {Q, E, H, J, O = {Q,H}, I = {E,J}, A = {Q,E}, B = {H,J}, S = {Q,J}, P = {E,H}, C = {E,H,J}, F = {Q,H,J}, L = {Q,E,J}, V = {Q,E,H}, Y = {Q,E,H,J}, U}. Тогда взаимодействие двух векторов по правилам теоретико-множественной операции симметрической разности $m_i \Delta A_i = (m_i \cap \tilde{A}_i) \cup (\tilde{m}_i \cap A_i)$, например, будет иметь следующий вид: $m\Delta A = (SPQF)\Delta(QEAL) = (JHEP)$. Графическая интерпретация векторов и результата операции оценивания их сходства имеет следующий вид:

$$m\Delta A = (SPQF)\Delta(QEAL) = (JHEP) = \begin{matrix}00\\01\\10\\11\end{matrix}\begin{bmatrix}1&.&1&1\\.&1&.&.\\.&1&.&1\\1&.&.&1\end{bmatrix}\Delta\begin{bmatrix}1&.&1&1\\.&1&1&1\\.&.&.&.\\.&.&.&1\end{bmatrix}=\begin{bmatrix}.&.&.&.\\.&.&1&1\\.&1&.&1\\1&.&.&.\end{bmatrix}.$$

Здесь вектор-столбец (1..1) идентифицирует или кодирует пару элементов {00, 11} одним символом S, а точки обозначают отсутствие соответствующих примитивов, в данном случае {01, 10}. Естественно, что сложение всех матриц путем применения операции симметрической разности дает матрицу пустых значений:

$$(SPQF)\Delta(QEAL)\Delta(JHEP) = (\varnothing\varnothing\varnothing\varnothing) \rightarrow$$

$$\rightarrow \begin{matrix}00\\01\\10\\11\end{matrix}\begin{bmatrix}1&.&1&1\\.&1&.&.\\.&1&.&1\\1&.&.&1\end{bmatrix}\Delta\begin{bmatrix}1&.&1&1\\.&1&1&1\\.&.&.&.\\.&.&.&1\end{bmatrix}\Delta\begin{bmatrix}.&.&.&.\\.&.&1&1\\.&1&.&1\\1&.&.&.\end{bmatrix}=\begin{bmatrix}.&.&.&.\\.&.&.&.\\.&.&.&.\\.&.&.&.\end{bmatrix}.$$

Выполнение операции симметрической разности можно представить набором тождеств:

$$\Delta = \begin{cases} 0 \leftarrow (m = 0 \wedge A = \varnothing) \vee (m = 1 \wedge A = x) \vee m = x \wedge A = 1) \vee (m = \varnothing \wedge A = 0); \\ 1 \leftarrow (m = 0 \wedge A = x) \vee (m = 1 \wedge A = \varnothing) \vee m = x \wedge A = 0) \vee (m = \varnothing \wedge A = 1); \\ x \leftarrow (m = 0 \wedge A = 1) \vee (m = 1 \wedge A = 0) \vee m = x \wedge A = \varnothing) \vee (m = \varnothing \wedge A = x); \\ \varnothing \leftarrow (m = 0 \wedge A = 0) \vee (m = 1 \wedge A = 0) \vee m = x \wedge A = x) \vee (m = \varnothing \wedge A = \varnothing). \end{cases}$$

Они регулируют все отношения в четырехзначном алфавите Кантора $A = \{0, 1, x = \{0,1\}, \varnothing\}$ для определения оценки взаимодействия или сходства двух объектов, например:

$$m\Delta A = (01xx)\Delta(10x1) = (xx00) = \begin{matrix}0\\1\end{matrix}\begin{bmatrix}1&.&1&1\\.&1&1&1\end{bmatrix}\Delta\begin{bmatrix}.&1&1&.\\1&.&1&1\end{bmatrix}=\begin{bmatrix}1&1&.&1\\1&1&.&.\end{bmatrix}.$$

Очевидно, что значность алфавита увеличивает число строк, спрятанных в многозначном векторе, которые можно одновременно анализировать в процессе распознавания образов, описанных упорядоченной совокупностью логических векторов.

С помощью свертываемых в многозначный вектор единичных матриц можно легко создавать эталоны символов, цифр, знаков, геометрических фигур, в дальнейшем используемые для создания более сложных образных эталонов. Некоторые из них (N,I,C,O) представлены ниже:

$$A = \begin{matrix}00\\01\\10\\11\end{matrix}\begin{bmatrix}1&.&.&1\\1&1&.&1\\1&.&1&1\\1&.&.&1\end{bmatrix}\begin{bmatrix}.&1&1&1\\.&.&1&.\\.&.&1&.\\.&1&1&1\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&.&.&.\\1&.&.&.\\1&1&1&1\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&.&.&1\\1&.&.&1\\1&1&1&1\end{bmatrix}.$$

Распознавание некорректно написанного тестового символа C на множестве эталонов дает матричный результат:

$$\begin{bmatrix}1&1&1&1\\1&1&.&.\\1&1&.&.\\1&1&1&1\end{bmatrix}\Delta\begin{bmatrix}1&.&.&1\\1&1&.&1\\1&.&1&1\\1&.&.&1\end{bmatrix}\begin{bmatrix}.&1&1&1\\.&.&1&.\\.&.&1&.\\.&1&1&1\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&.&.&.\\1&.&.&.\\1&1&1&1\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&.&.&1\\1&.&.&1\\1&1&1&1\end{bmatrix}=$$

$$=\begin{bmatrix}.&1&.&.\\.&.&.&1\\.&1&1&1\\.&1&.&.\end{bmatrix}\begin{bmatrix}1&.&.&.\\1&1&1&.\\1&1&1&.\\1&.&.&.\end{bmatrix}\begin{bmatrix}.&.&.&.\\.&1.&.&.\\.&1.&.&.\\.&.&.&.\end{bmatrix}\begin{bmatrix}.&.&.&.\\.&1&.&1\\.&1&.&1\\.&.&.&.\end{bmatrix}\rightarrow$$

$$\mu=\min(\mu_1,\mu_2,\mu_3,\mu_4)=\begin{bmatrix}.&.&.&.\\.&1.&.&.\\.&1.&.&.\\.&.&.&.\end{bmatrix}\rightarrow\begin{bmatrix}1&1&1&1\\1&.&.&.\\1&.&.&.\\1&1&1&1\end{bmatrix}\rightarrow C.$$

В аналитической форме последнее выражение можно записать в следующем виде: $\mu=\min(\mu_1,\mu_2,\mu_3,\mu_4)=[\varnothing P\varnothing\varnothing]\rightarrow[YSSS]\rightarrow C$, что позволяет идентифицировать входное изображение как букву С.

Таким образом, функция распознавания входного образа m относительно имеющихся в наличии эталонов множества А может быть представлена в следующем виде:

$$\mu(m,A)=A_i[\min(m\overset{n}{\underset{1=1}{\Delta}}A_i)].$$

Иначе, функционал, определяющий максимально сходный с эталоном образ, обращает критерий качества взаимодействия в минимальное число пиксельных или координатных несовпадений. В двоичном варианте описания образов симметрическая разность вырождается в хог-функцию:

$$\mu(m,A)=A_i[\min(m\overset{n}{\underset{1=1}{\oplus}}A_i)].$$

Практическая значимость и проблемы для решения:

1. Аналитическая и матричная оценки степени сходства прообразов или взаимодействия объектов любой природы вычисляются параллельно, путем обработки векторов или матриц, использующих операцию симметрической разности, вычисляемую за один временной такт благодаря использованию мультипроцессора.

2. Наглядность матричной оценки в фрагментах изображения, где существуют различия, дает возможность корректировать не только анализируемый объект, но и эталонные примитивы на стадии создания библиотеки образов.

3. Технология распознавания образов, основанная на матричном (векторном) представлении изображений, ориентирована на использование матричных параллельных вычислений путем применения видеопроцессора или неарифметического мультипроцессора.

4. Переход от матричного к аналитическому образу и наоборот обеспечивает как визуализацию процесса распознавания образов, так и высокое быстродействие параллельного выполнения основной и единственной векторной операции симметрической разности (исключающее ИЛИ для двоичного алфавита).

5. Модели и метод векторно-матричного распознавания образов допускают сканируемое по галсам распознавание масштабных изображений окном допустимой для пользователя размерности.

6. Векторно-логическое представление изображения дает возможность существенно минимизировать объем памяти для его хранения, если соседние векторы имеют незначительные отличия друг от друга.

7. Нормализация изображений для последующего сравнения образов сводится к выполнению операций сдвига по трем степеням свободы и масштабированию к размерности эталонных примитивов, представленных в библиотеке.

8. Матричный образ, представленный двоичным кодом, является одним из возможных вариантов применения технологии распознавания, в то время как все объекты в кибернетическом пространстве, по сути, есть цифровые образы, к которым применима универсальная модель векторно-логического процесса распознавания, предложенная выше.

9. Стандартизация моделей образов. Разбиение образа сеткой на сегменты позволяет масштабировать рисунки любой размерности для их приведения к матрице одной размерности, имеющей одинаковое число координат, которые идентифицируются двоичными или многозначными значениями, составляющими сигнатуру изображения. Это дает возможность стандартизировать не только модели образов, пригодные для сравнения, но и модели критериев сходства, которые имеют такой же формат данных. Иначе, модели входных образов, эталонов и критериев должны иметь одинаковый формат данных. Имея два любых из упомянутых компонентов, можно легко восстановить третий. Например, взаимодействие матриц одной размерности теста, функциональности и дефектов в техническом изделии может быть представлено тремя полезными равенствами:

$$T \oplus M \oplus F = 0 \rightarrow \begin{cases} T = M \oplus F; \\ M = T \oplus F; \\ F = T \oplus M. \end{cases}$$

В общем случае уравнение взаимодействия трех объектов:

$$m \oplus A \oplus Q = 0 \rightarrow \begin{cases} Q = m \oplus A; \\ m = A \oplus Q; \\ A = m \oplus Q \end{cases}$$

– образа, эталона и критерия качества отношения между ними – формулирует, а значит, решает сотни практически полезных задач, включая: принятие решений, распознавание образов, тестирование и диагностику процессов и явлений.

Области применения векторно-логической технологии анализа процессов или явлений: 1) распознавание текстов в регистрационных картах пересечения границы; 2) идентификация личности по фотографиям, близким к стандартным снимкам для получения визовых документов; 3) поиск аналогов в Интернете по заданным образам; 4) сортировка изображений в базе данных по классам и признакам; 5) дактилоскопия и создание классифицированной библиотеки спецслужб; 6) распознавание и классификация программных вирусов; 7) распознавание почерков и идентификация личности по существенным признакам написания букв; 8) идентификация наземных целей и движущихся объектов (самолеты, корабли, машины); 9) синтез или корректировка образов по характерным существующим признакам; 10) применение технологии векторно-матричного (-логического) распознавания для управления робототехнических комплексов; 11) распознавание образов обоняния, вкуса, звуковых, тепловых и радиочастотных излучений; 12) распознавание лингвистических конструкций и примитивов, а также их оценивание при сравнении с эталонами; 13) динамическая визуализация эффектов последовательного и мягкого трансформирования одного образа в любой другой.

7. Заключение

1. Кибернетическое пространство как объект природы подвержен влиянию деструктивных средств, влияющих на работоспособность субъектов, которыми являются компьютеры, системы и сети. Поэтому важной проблемой остается стандартизация и спецификация пространства и всех взаимодействующих субъектов, включая негативные. Данная акция есть постоянно действующая во времени, цель которой – не отставать, но на один шаг опережать появление новых деструктивных компонентов, что обеспечит функционирование компьютерной экосистемы планеты и качества жизни человечества.

2. В связи с этим инфраструктура кибернетического пространства, метрика его измерения и процесс-модели анализа и синтеза субъектов дают возможность создавать эффек-

тивные решения для компьютерных изделий, ориентированных на быстрый поиск, распознавание, диагностирование не только позитивных, но и негативных субъектов. Конкретно, инфраструктура призвана решать задачи: 1) Описание многообразия деструктивных компонентов кибернетического пространства. 2) Формализация процессов взаимодействия триады компонентов <программа (аппаратура), функциональные нарушения, тесты>. 3) Диагностирование и устранение функциональных нарушений. 4) Создание и эффективное использование базы деструктивных данных или функциональных нарушений. 5) Создание быстродействующих интеллектуальных саморазвивающихся средств сервисного обслуживания и защиты кибернетического пространства.

3. Компьютер, как субъект экосистемы, имеет собственный путь развития во взаимодействии с остальным миром. Ученым не следует заниматься проблемой очеловечивания компьютера. Но компьютер нуждается в помощи человека в части наделения его способностью к саморазвитию (творчеству) путем создания стандартизованной инфраструктуры информационно-компьютерного пространства, включающей опыт человечества, оформленный в виде всемирной библиотеки стандартизованной математической, технологической и интеллектуальной культуры.

4. Информационно-компьютерная экосистема планеты есть материальная структура, ориентированная на повышение качества жизни каждого человека. При этом персонализация компьютера трансформируется в индивидуализацию, а далее – в личное киберпространство, что имеет целью создать интеллектуального и верного «друга», способного обслуживать человека 24 часа в сутки по всем сервисам, связанным с бытом, бизнесом, отдыхом, здоровьем.

5. Информационное векторное логическое пространство как подмножество метрического регулирует взаимодействие конечного числа объектов с помощью введенных определений, аксиом тождественности, симметрии и транзитивности треугольника. При этом последнее свойство вырождается в строгое равенство, что дает возможность потенциально уменьшить на треть объемы двоичной информации о взаимодействии объектов, благодаря свертке любого замкнутого логического пространства в нуль-вектор.

6. Рыночная привлекательность обозначенных проблем заключается в ориентации математической и технологической культуры на создание метрики кибернетического пространства, инфраструктуры сервисного обслуживания в виде моделей, методов и средств, включающих быстродействующие мозгоподобные компьютеры, критерии качества взаимодействия объектов в пространстве при поиске, распознавании и принятии решений, а также новые сервисы со стороны кибернетического пространства и индивидуального интеллектуального компьютера (киберпространства).

7. Актуальные проблемы создания саморазвивающейся информационно-компьютерной экосистемы: 1) Синтез минимального количества элементарных функций, способных покрыть все коды изображения для минимизации информационного объема, передаваемого по каналу. 2) Уменьшение объемов информации и минимизация структур путем использования принципа дополнения в избыточности замкнутых алфавитов. 3) Формирование решений в причинно-следственном покрытии на основе максимизации критерия качества взаимодействия вектор-спецификации и библиотечных примитивов, которые должны быть упорядочены по разделам или кодам и представлены в форме двоичного дерева. 4) Разработка формальных методов синтеза функциональностей по таблице истинности, которая в совокупности составляет спецификацию проекта. 5) Создание новых методов сжатия данных, основанных на использовании метрики векторно-логического пространства, для передачи информации по телекоммуникационным системам. 6) Разработка новых методов криптоанализа на основе использования метрики векторно-логического пространства и принципа свертки кодов расстояний в нулевую сумму.

Список литературы: 1. *Babulak E.* Future Global Office // 12th International Conference “Computer Modelling and Simulation”. 2010. P. 352–356. **2.** *Caplan K., Sanders J.L.* Building an international security standard // IT Professional. Vol. 1, Issue 2. 1999. P. 29-34. **3.** *Qishi Wu, Ferebee D., Yunyue Lin, Dasgupta D.* Visualization of security events using an efficient correlation technique // Computational Intelligence in Cyber Security, CICS ’09. 2009. P. 61-68. **4.** *Инфраструктура* мозгоподобных вычислительных процессов / М.Ф. Бондаренко, О.А. Гузь, В.И. Хаханов, Ю.П. Шабанов-Кушнаренко. Харьков: Новое Слово. 2010.

160 с. **5.** *Хаханов В.И., Чумаченко С.В.* Модели пространств в научных исследованиях// Радиоэлектроника и информатика. 2002. №1. С. 124-132. **6.** *IEEE* Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture IEEE Std 1149.7-2009. 985 p. **7.** *Da Silva F., McLaurin T., Waayers T.* The Core Test Wrapper Handbook. Rationale and Application of IEEE Std. 1500™. –Springer.– 2006. XXIX. 276 p. **8.** *Marinissen E.J., Yervant Zorian.* Guest Editors' Introduction: The Status of IEEE Std 1500.IEEE Design & Test of Computers. 2009. No26(1). P.6-7. **9.** *IEEE* Std 1800-2009 IEEE Standard for System Verilog-Unified Hardware Design, Specification, and Verification Language. http://ieeexplore.ieee.org/servlet/opac?punumber= 5354133. **10.** *Marinissen E.J.* Testing TSV-based three-dimensional stacked ICs // DATE 2010. 2010. P.1689-1694. **11.** *Benso A., Di Carlo S., Prinetto P., Zorian Y.* IEEE Standard 1500 Compliance Verification for Embedded Cores // IEEE Trans. VLSI Syst. 2008. No 16(4). P. 397-407. **12.** *Ubar R., Kostin S., Raik J.* Embedded diagnosis in digital systems // 26th International Conference "Microelectronics", MIEL 2008. 2008. P. 421-424. **13.** *Elm M., Wunderlich H.-J.* Scan Chain Organization for Embedded Diagnosis // Design, Automation and Test in Europe, DATE '08. 2008. P. 468–473. **14.** *Проектирование* и тестирование цифровых систем на кристаллах. Verilog & System Verilog / В.И. Хаханов, Е.И. Литвинова, О.А. Гузь. Харьков: ХНУРЭ. 2009. 484с. **15.** *Проектирование* и верификация цифровых систем на кристаллах / В.И. Хаханов, И.В. Хаханова, Е.И. Литвинова, О.А. Гузь. Харьков: Новое слово. 2010. 528с. **16.** *Семенец В.В., Хаханова И.В., Хаханов В.И.* Проектирование цифровых систем с использованием языка VHDL. Харьков: ХНУРЭ. 2003. 492 с. **17.** *Хаханов В.И., Хаханова И.В.* VHDL+Verilog = синтез за минуты. Харьков: ХНУРЭ. 2006. 264с. **18.** *Zorian Yervant.* Guest Editor's Introduction: Advances in Infrastructure IP // IEEE Design and Test of Computers. 2003. P.49-55. **19**. *Bulent I. Dervisoglu.* A Unified DFT Architecture for Use with IEEE 1149.1 and VSIA/IEEE P1500 Compliant Test Access Controllers. Proceedings of the Design Automation Conference. 2001. P. 53-58. **20.** *Bergeron J.* Writing Testbenches using SystemVerilog. Springer US.2006. 414 p. **21.** *Shibata T.* Implementing brain-like systems using nano functional devices // Ultimate Integration of Silicon, ULIS 2009. 2009.P. 131-134. **22.** *Soliman M.I., Al-Junaid A.F.* Codevelopment of Multi-level ISA and hardware for an efficient matrix processor // International Conference Computer Engineering & Systems.2009. P. 211-217. **23.** *Senning C., Studer C., Luethi P., Fichtner W.* Hardware-efficient steering matrix computation architecture for MIMO communication systems // IEEE International Symposium Circuits and Systems. 2008. P. 304-307. **24.** *Pedram Ardavan, Daneshtalab Masoud, Sedaghati-Mokhtari Nasser, Fakhraie Sied Mehdi.* A High-Performance Memory-Efficient ParallelHardware for Matrix Computation in Signal Processing Applications // International Symposium Communications and Information Technologies. ISCIT '06. 2006. P. 473-478. **25.** *Chenlong Hu, Ping Yang, Ying Xiao, Shaoxiong Zhou.* Hardware design and realization of matrix converter based on DSP & CPLD // 3rd International Conference Power Electronics Systems and Applications. 2009. P. 1-5. **26.** *Dave N., Fleming K., Myron King, Pellauer M., Vijayaraghavan M.* Hardware Acceleration of Matrix Multiplication on a Xilinx FPGA // 5th IEEE/ACM International Conference Formal Methods and Models for Codesign. 2007. P.97-100. **27.** *Loucks W.M., Snelgrove M., Zaky S.G.* A Vector Processor Based on One-BitMicroprocessors // IEEE Micro. Volume 2, Issue 1. 1982. P. 53-62. **28.** *Hilewitz Y., Lauradoux C., Lee R.B.* Bit matrix multiplication in commodity processors // International Conference Application-Specific Systems, Architectures and Processors. 2008. P. 7-12.

Поступила в редколлегию 16.02.2011

Tiecoura Yves, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

Чумаченко Светлана Викторовна, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: ri@kture.kharkov.ua.

Хаханов Владимир Иванович, декан факультета КИУ ХНУРЭ, д-р техн. наук, проф. кафедры АПВТ ХНУРЭ. Научные интересы: техническая диагностика цифровых систем, сетей и программных продуктов. Увлечения: баскетбол, футбол, горные лыжи. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326. E-mail: hahanov@kture.kharkov.ua.

УДК. 519. 71

В.С. МИХАЙЛЕНКО, Р.Ю. ХАРЧЕНКО

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТЕОРИЙ ПРИНЯТИЯ РЕШЕНИЙ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ И НЕЧЕТКОЙ ЛОГИКИ НА ПРИМЕРЕ НАСТРОЙКИ ПИ-РЕГУЛЯТОРА

Рассматриваются возможности применения различных вариантов математического аппарата в АСУ ТП сложных объектов с ПИ-регулированием. Анализируются и моделируются на ЭВМ как модели традиционных теорий, так и модели с применением альтернативного направления – теории нечеткой логики. Результаты проведенной работы показали актуальность рассмотрения нечеткого подхода в условиях неопределенности при разработке интеллектуальных систем управления.

1. Введение

Традиционные автоматизированные системы управления технологическими процессами (АСУ ТП) строятся, как правило, на основе линейных моделей объектов. Полученные таким образом регуляторы являются устойчивыми по отношению к заложенным в их основу моделям реальных технологических процессов. Однако часто методы преобразования Лапласа и линеаризации, применяемые к нелинейным, динамическим, нечетко определенным объектам не дают ожидаемых результатов устойчивого управления и желаемого качества регулирования. А с увеличением сложности структуры объекта и выполняемых им функций становится невозможно использовать классические методы управления. Стремление преодолеть обостряющееся противоречие между усложнением создаваемых систем и традиционными подходами к их проектированию и обслуживанию в настоящее время определило одно из возможных новых направлений развития АСУ ТП, связанное с применением интеллектуальных технологий. На основе результатов научных работ [7] сформировалось несколько самостоятельных подходов к решению данной проблемы, основанных на использовании методов искусственного интеллекта, а именно:

– применение технологии экспертных систем (ЭС), предполагающей программно-алгоритмическую реализацию интеллектуальных функций на основе применения знаний;

– применение технологии нейро – нечетких сетевых структур, предполагающей аппаратно-программную реализацию интеллектуальных функций.

Использование технологии экспертных систем позволяет существенно повысить гибкость исполнительного уровня управления, что обеспечивается следующими факторами:

– возможностью работать с несколькими алгоритмами управления и адаптации, осуществляя их обоснованный выбор на основе текущей (иногда неполной и противоречивой) информации о функционирующей системе и используя знания, которые содержатся в ЭС;

– способностью к обучению и корректировке знаний; при этом содержимое ЭС всегда может быть расширено и модифицировано, что обеспечивает адаптацию системы к изменениям как в самом объекте, так и в текущих целях управления.

Необходимость разработки интеллектуальных технологий для АСУ ТП представляется *актуальной* с позиции усовершенствования, а также диктуется потребностями в уменьшении времени на настройку систем, без снижения качества и надежности.

Например, при управлении сложной системой, функционирующей в условиях неопределенности, значения контролируемых параметров объекта регулирования входят за установленные диапазоны в силу влияния неконтролируемых возмущений. В этом случае, эксперту-наладчику АСУ ТП необходимо производить дистанционное управление сложным объектом либо перенастраивать регулятор на оптимальные настройки. И для определения настроечных параметров регулятора при управлении объектом в условиях неопределенности эксперту необходимо произвести выбор математического аппарата.

Целью исследования является нахождение значений настроек ПИ – регулятора на основе аппарата теории принятия решений и теории нечеткой логики и проведение сравни-

тельного анализа полученного результата по критерию времени регулирования. Наилучшее решение может быть использовано в супервизорной АСУ для выдачи рекомендаций персоналу при коррекции настроек или для разработки системы адаптивного управления. При этом необходимо решить следующие задачи:

1. Определить диапазон изменения значений канала внешнего возмущения и соответствующий ему диапазон значений настроек регулятора для компенсации такового.
2. Провести расчет по определению оптимальных настроек с помощью методов теории принятия решений в условиях неопределенности и теории нечеткой логики.
3. Повести апробацию полученных значений с помощью компьютерного эксперимента.
4. Провести сравнительный анализ результатов в целях получения переходного процесса с минимальным временем регулирования.

Рассмотрим научно-производственную задачу: предположим, оператору – наладчику АСУ ТП известен диапазон изменения значений параметров объекта и в силу технологических условий основным критерием при выборе новых настроек регулятора является время регулирования переходного процесса. Таким образом, перед наладчиком стоит задача выбора настройки типового ПИ- регулятора для получения минимального времени регулирования при нахождении объекта в условиях неопределенности, например, подверженного нестационарным внешним возмущениям.

2. Использование теории принятия решений в условиях неопределенности

Для решения поставленной задачи рассмотрим один из научных подходов создания ЭС – теорию принятия решений в условиях неопределенности с использованием следующих критериев для анализа ситуации [2] :

1. Критерий Лапласа.
2. Минимаксный критерий (Вальда).
3. Критерий Сэвиджа.
4. Критерий Гурвица.

Параметры системы представим в виде табл. 1

Таблица 1

Состояния системы управления

$v(a_1, s_1)$	$v(a_1, s_2)$	$v(a_1, s_n)$
$v(a_2, s_1)$	$v(a_2, s_2)$	$v(a_2, s_n)$
$v(a_m, s_1)$	$v(a_m, s_2)$	$v(a_m, s_n)$

Элемент a представляет i-е возможное решение (настройки ПИ - регулятора), s_j элемент – j-е состояние системы (значение возмущения). Результат, связанный с решением a_i (настройкой регулятора K_p – коэффициент пропорциональности, $T_и$ – постоянная интегрирования) и состоянием s_j (значением возмущения T_N) составляет $v(a_i,s_j)$ – время регулирования (T_p). Представленные критерии отличаются по степени консерватизма, который проявляет эксперт. Рассмотрим их.

1. Критерий Лапласа использует оптимистическое предположение, что вероятности всех состояний системы (возмущений) равны между собой $P(s_1) = P(s_n) = 1/n$.

По Лапласу

$$K_P = \min\left\{\frac{1}{n}\sum_{j=1}^{n} v(a_i, s_j)\right. . \qquad (1)$$

2. Минимаксный критерий сводится к выбору минимального значения альтернативы из максимальных:

$$K_P = \min_{a_i}\{\max_{s_i} v(a_i, s_j)\} . \qquad (2)$$

3. Критерий Сэвиджа рассчитывается с учетом матрицы потерь и дальнейшего использования (1):

$$K_P = r(ai, sj) = v(ai, sj) - \min_{a_i} \{v(a_i, s_j)\}. \quad (3)$$

4. Критерий Гурвица основан на учете показателя оптимизма $\alpha \in [0,1]$:

$$K_P = \min_{a_i} \{\alpha \min_{s_j} v(a_i, s_j) + (1-\alpha) \max_{s_j} v(a_i, s_j)\}. \quad (4)$$

3. Первый компьютерный эксперимент

Для анализа эффективности предложенного подхода проведем компьютерный эксперимент в программе MatLab (Simulink) [3] (рис.1).

Объект представлен последовательным соединением инерционных звеньев с запаздыванием находящегося под действием канала возмущения в виде инерционного звена $W(s) = K_N/(T_N(s)+1)$. Задание $z = 2$.

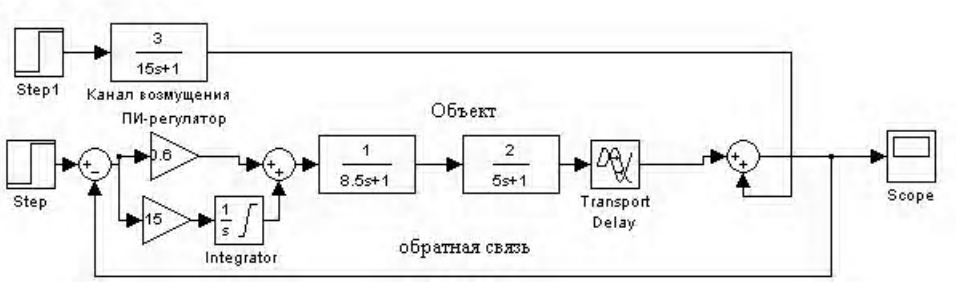


Рис. 1. Автоматическая система регулирования с ПИ – регулятором

Постоянная времени возмущения T_N варьируется в диапазоне (15 – 80 с), где (s_1=15с, s_2=40 с, s_3=80 с), при этом точное значение T_N в текущий момент неизвестно. Задача сводится к определению оптимальных настроек K_p, $T_и$ ($a_1 = (K_{p1}=0,3, T_и = 12)$; $a_2 = (K_{p2}=0,6, T_и=15)$; $a_3 = (K_{p3}=0,8, T_и=20)$ в целях получения минимального значения времени регулирования $v(a_i,s_j) = T_p$. При этом в зависимости от неопределенной ситуации (случайного T_N) можно ожидать следующие значения (табл.2).

Таблица 2

Значения состояния системы

Настройки/возмущения	s_1	s_2	s_3
a_1	60	130	300
a_2	50	125	225
a_3	70	130	225

Проанализируем данную ситуацию с точки зрения четырех рассмотренных выше критериев:

1.Лапласа. При одинаковых вероятностях возникновения T_N=P(s_j) = 1/3, ожидаемые значения T_p для различных возможных решений находят согласно (1):

$M(a_1) = 163,3$ с; $M(a_2) = 133,3$ с; $M(a_3) = 141,7$ с. Оптимальным является альтернатива а2.

2. Минимаксный критерий согласно (2) сводится к сведению к минимуму ожидаемого результата и позволяет получить альтернативу - a_2 или a_3.

3. Критерий Сэвиджа (3) позволяет получить табл. (3).

Таблица 3

Расчеты по Сэвиджу

Настройки/возмущения	s_1	s_2	s_3
a_1	10	5	5
a_2	0	0	0
a_3	20	5	0

Используя (2), получаем альтернативу a_2.

4. Критерий Гурвица (4). При б = 0,5 получаем табл. 4.

Таблица 4

Расчеты по Гурвицу

Альтернатива	Min строки	Max строки	α (Min строки)+ (1-α) (Max строки)
a_1	60	300	300-240α
a_2	50	225	225-175α
a_3	70	225	225-155α

Подставляя α, получаем в качестве альтернативного решения стратегию a_2.

Заключение по первому эксперименту. Сравнительный анализ результатов четырех критериев показывает, что оптимальным является альтернатива a_2 ($K_{р2}$=0,6, $T_и$=15), т.е. согласно теории принятия решений в условиях неопределенности, при данных настройках ПИ- регулятор будет поддерживать минимальное время регулирования при действии на объект неконтролируемых или случайных возмущений.

4. Использование теории нечеткой логики

Рассмотрим альтернативное научное направление – теорию нечеткой логики и нечетких множеств [4-6], позволяющую создавать нечеткие экспертные системы (НЭС), которые имитируют рассуждения эксперта – наладчика АСУ ТП. Данный подход предполагает использование знаний экспертов об объекте управления, представляемых в виде правил, выраженных на естественном языке. При описании объекта используются лингвистические переменные, определяющие состояние объекта. Дальнейшие процедуры формализации направлены на получение так называемых нечетких множеств, определяющих параметры объекта управления (этап фаззификации). А расчет значений управления производится с помощью применения операций - t-норм к нечетким множествам. t-нормы, или триангулярные нормы, реализуют логические операции “И”, “ИЛИ”, “НЕ”, а также операции минимума или максимума над нечеткими множествами [6]. Последним этапом является обратное преобразование значений управления в виде нечеткого множества в четкое значение выхода регулятора (дефаззификация). Базовыми типами такого рода регуляторов являются контроллеры Мамдани и Сугено или экспертные регуляторы.

Подстройку регулятора можно выполнить на основе правил, которые используются для ручной настройки. Эти правила получены из опыта, теоретического анализа и численных экспериментов. Они сводятся к следующему:

– увеличение пропорционального коэффициента увеличивает быстродействие и снижает запас устойчивости;

– с уменьшением интегральной составляющей ошибка регулирования с течением времени уменьшается быстрее;

– уменьшение постоянной интегрирования уменьшает запас устойчивости;

Перечисленные правила применяются также для регуляторов, использующих методы экспертных систем и нечеткой логики. Ручную настройку с помощью правил удобно выполнять с применением интерактивного программного обеспечения (СКАДА) на компьютере, временно включенном в контур управления. Для оценки реакции системы на изменение настроек, внешних воздействий или шумов измерений подают эмулированные воздействия и наблюдают реакцию на них. После определения наилучших настроек значения коэффициентов регулятора записывают в память ПИ-контроллера. Используя опыт эксперта – наладчика автоматизированных систем регулирования, представим его рассуждения в виде продукционных правил вида: ЕСЛИ возмущения высокие (V) И коэффициент пропорциональности регулятора высокий (V), И постоянная интегрирования регулятора высокая ТО время регулирования будет среднее (S); или ЕСЛИ s_1 = V И a_1 = V, ТО $v(a_1,s_1)$ = S.

5. Разработка нечеткой экспертной системы по определению оптимальных настроек ПИ – регулятора

Для разработки НЭС воспользуемся программой MatLab (FLT) (рис. 2) и встроенным в программу алгоритмом Мамдани [6], с последующим этапом фаззификации входных и выходной переменных (рис.3-6).

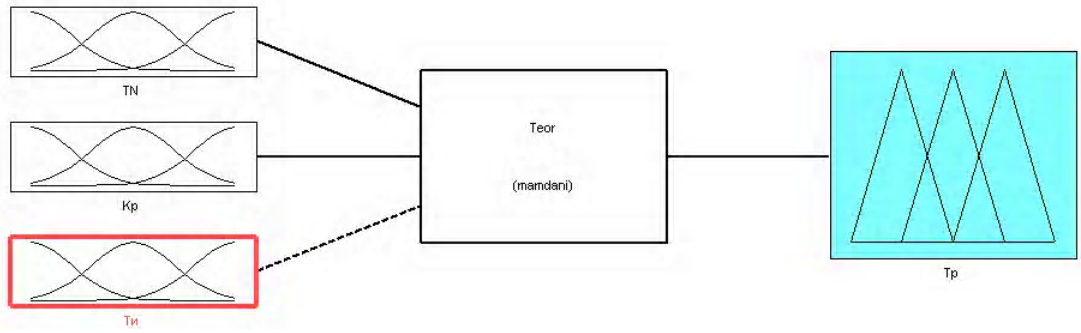


Рис. 2. Нечеткая система анализа информации

Входными параметрами НЭС являются T_N , $K_р$, $T_и$; выходной параметр – $T_р$. Типы функций принадлежности определены исходя из рекомендаций [6] и качественных рассуждений о процессе управления объектом.

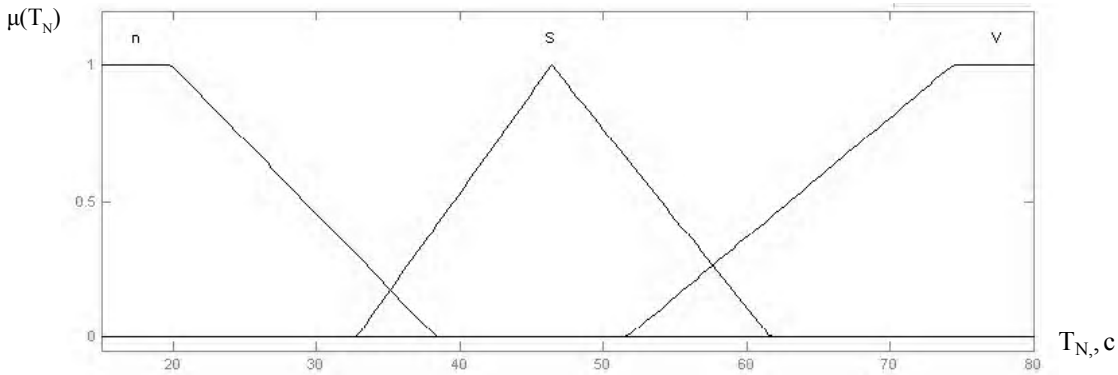


Рис. 3. Функции принадлежности T_N

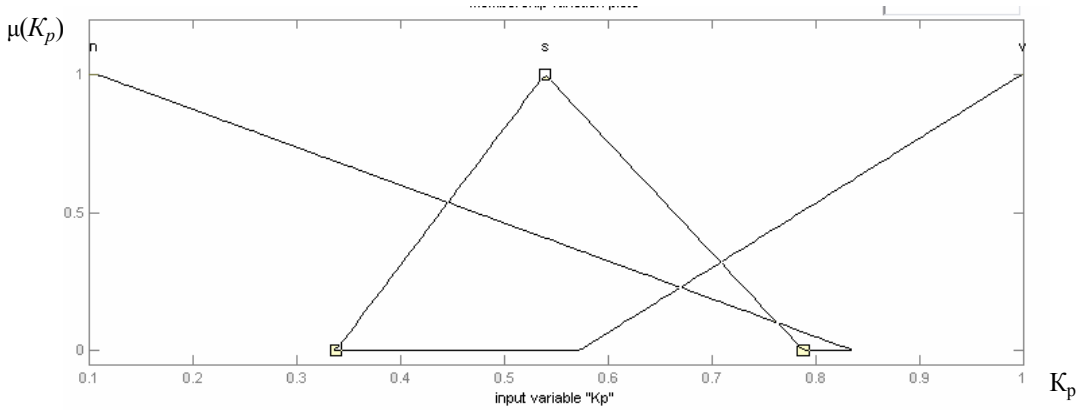


Рис. 4. Функции принадлежности $K_р$

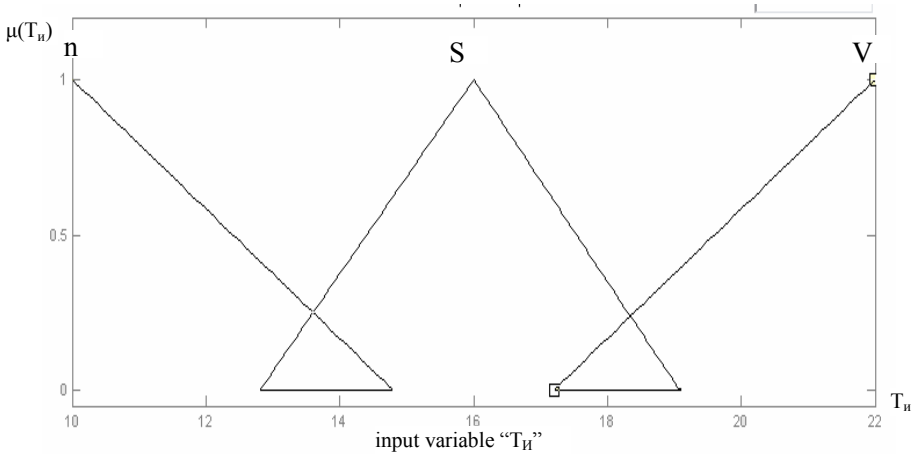


Рис. 5. Функции принадлежности $T_и$

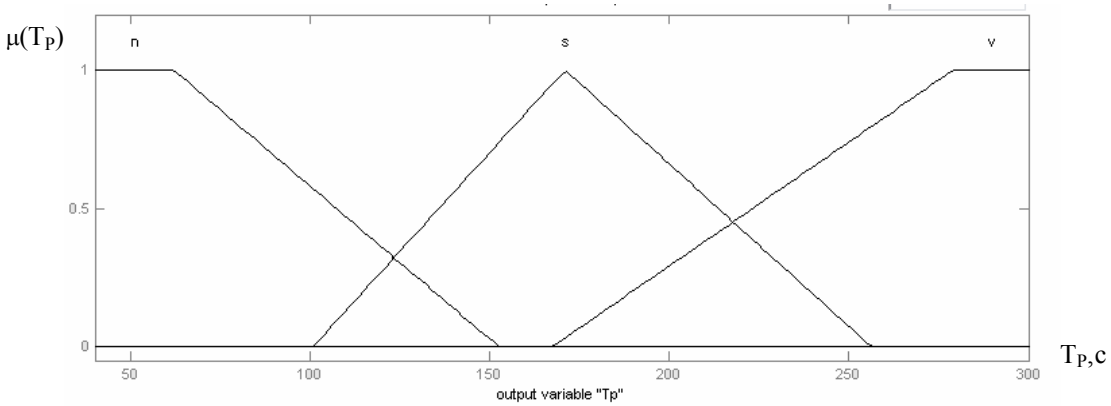


Рис. 6 . Функции принадлежности T_p

Фрагмент базы правил, имитирующую рассуждения эксперта, представлен на рис 7.

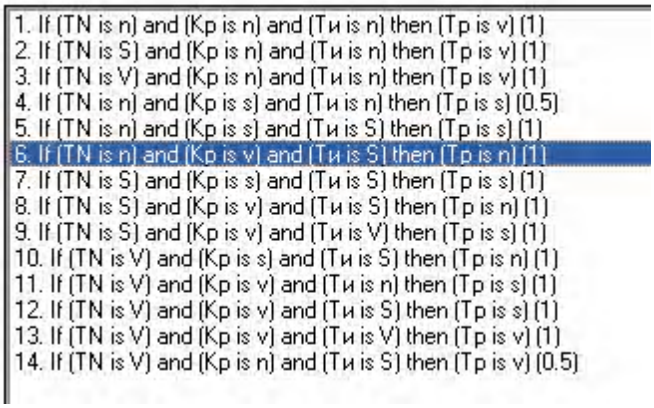


Рис. 7. Фрагмент базы правил НЭС в окне программы (FLT)

Программа FTL позволяет оператору АСУ ТП просматривать альтернативные настройки (a_i) на всем диапазоне возмущений T_N (рис.8).

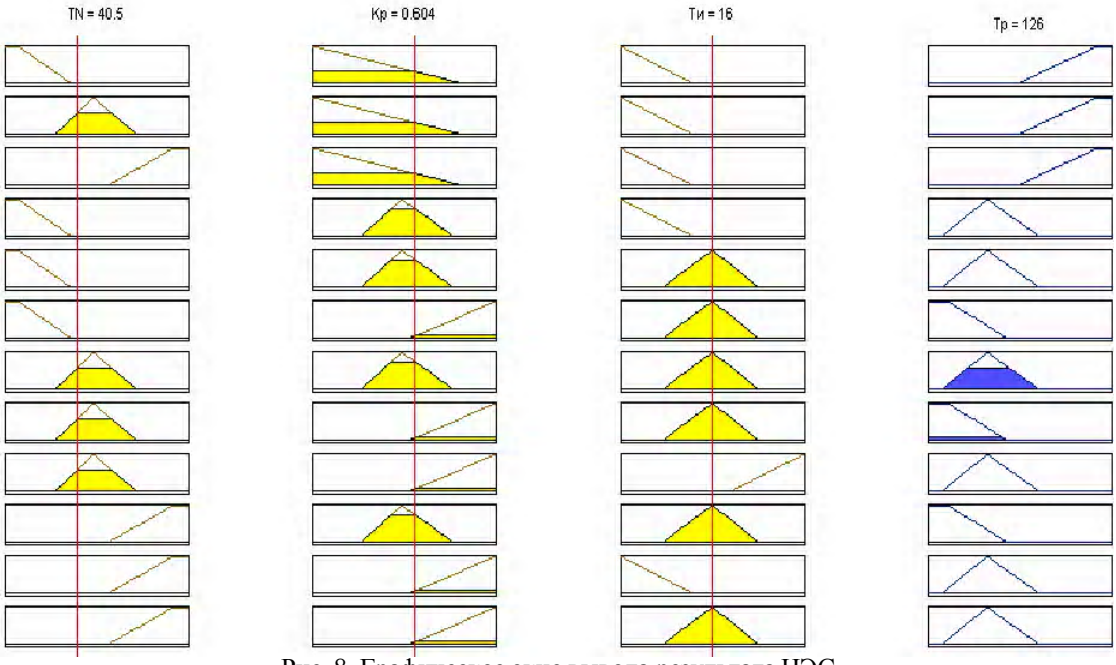


Рис. 8. Графическое окно вывода результата НЭС

Заключение по пункту 5. Как видно из рис.8 представленный результат $К_р$ = 0,6, $Т_и$ = 16 близок к альтернативе $a_{2,}$ определенной с помощью теории принятия решений (первый эксперимент). Таким образом, сравнительный анализ результата двух научных подходов демонстрирует их схожесть и позволяет рекомендовать полученные значения для адаптации ПИ -регулятора

6. Второй компьютерный эксперимент

Для проверки рекомендованного значения a_2 и осуществления сравнительного анализа с другими значениями настроек проведем компьютерный эксперимент с подстановкой произвольного значения T_N = 75 и настроек (a_1, a_2) (рис.9).

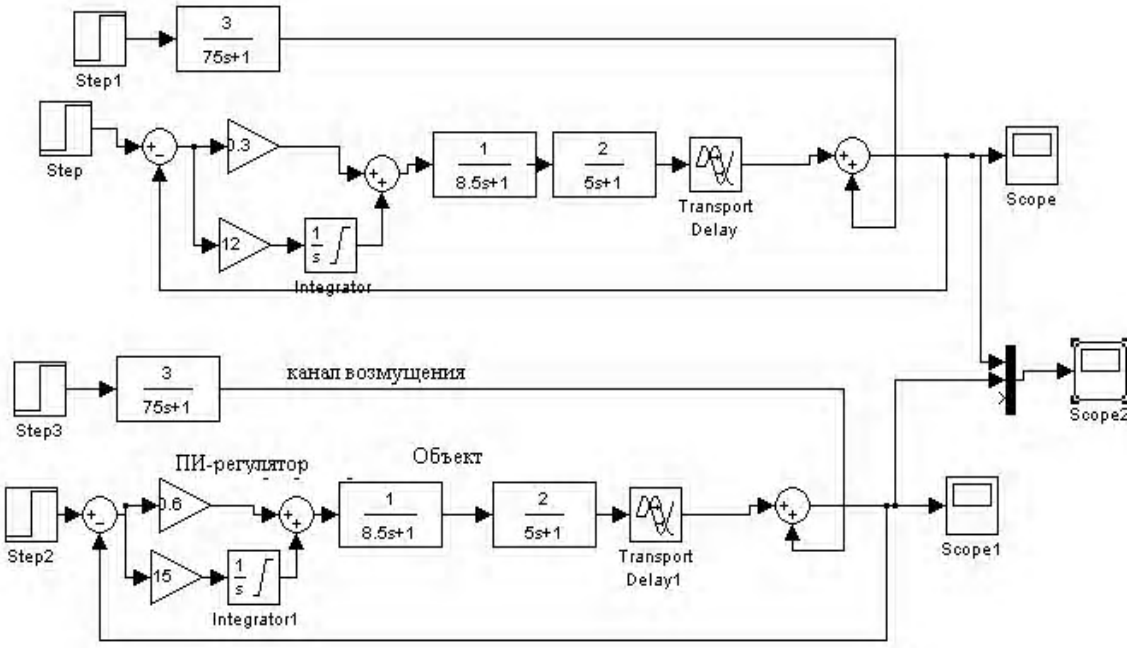


Рис. 9. АСР с ПИ – регуляторами при влиянии на объект внешнего возмущения

Результат эксперимента (переходные процессы регулирования) представлен на рис.10.

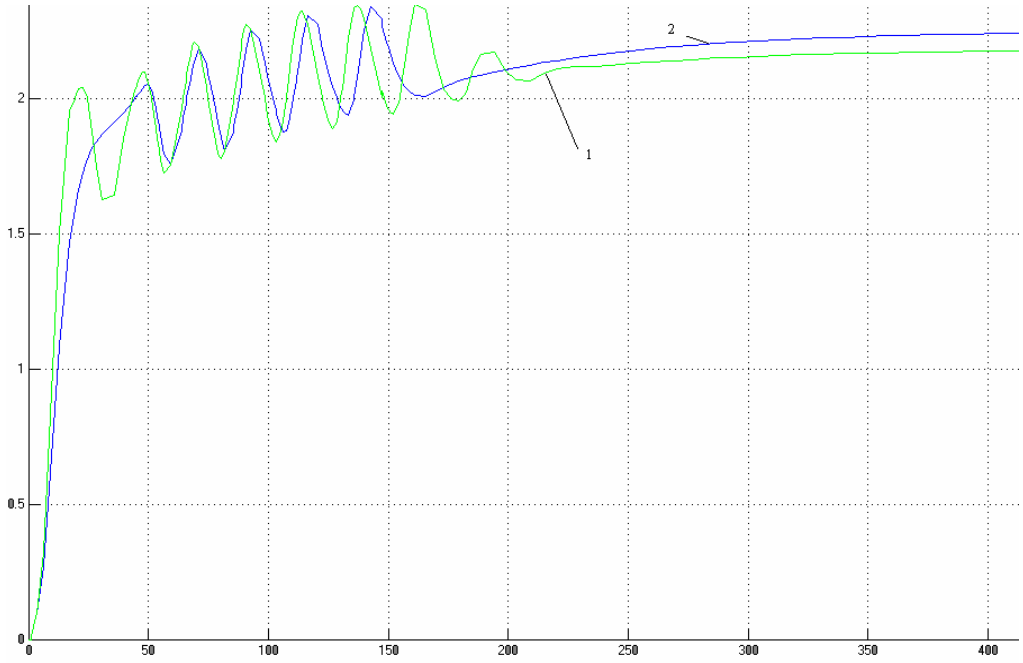


Рис. 10. Переходные процессы регулирования

7. Заключение

Как видно из рис. 10, альтернатива (a_2) демонстрирует лучший результат с $T_p = 225$ с в отличие от альтернативы a_1 с $T_p = 275$ с. Альтернатива a_3 показывает худший результат по сравнению с a_2 и имеет $T_p = 240$ с. Аналогичным образом можно рассматривать и определение настроек для ПИД – регулятора, а также моделировать параметрические возмущения.

Научная новизна заключается в установлении, что теория нечеткой логики, с подходом моделирования опыта эксперта, альтернативна теории принятия решений в условиях неопределенности и в отличие от последней позволяет оператору вести супервизорное управление без использования расчетов значений критериев по четырем методам.

Практическая значимость состоит в возможности совместного использования нескольких научных подходов при разработке интеллектуальных АСУ ТП. Это позволит выбрать наилучший алгоритм управления (альтернативу) в случае совпадения результатов, избежать ошибок в управлении, вызванных человеческим фактором, и повысит качество управления в целом.

Список литературы: 1. *Блюмин С.Л., Шуйкова И.А.* Модели и методы принятия решений в условиях неопределенности. Липецк: ЛЭГИ. 2001. **2.** *Тынкевич М.А.* Экономико – математические методы (исследование операций). Кемерово :КГТУ. 2000. **3.** *Дьяконов В.П.* Simulink 5/6/7: Самоучитель. М.: ДМК-Пресс, 2008. **4.** *Борисов А.Н., Алексеев А.В., Меркурьева Г.В.* и др. Обработка нечеткой информации в системах принятия решений. М: Радио и связь. 2002. **5.** *Алиев Р.А., Церковный А.Э., Мамедова Г.А.* Управление производством при нечеткой исходной информации. М.: Энергоиздат. 1991. 234 с. **6.** *Леоненков А. Ю.* Нечеткое моделирование в среде Matlab и fuzzyTech. С. Птб.: БХВ, 2003. 720. **7.** *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И.Д. Рудинского. М.: Горячая линия – Телеком, 2006. 452 с.

Поступила в редколлегию 19.03.2011

Михайленко Владислав Сергеевич, канд. техн. наук, доц. Одесской государственной академии холода. Научные интересы: интеллектуальные системы управления. Адрес: Украина, 65082, Одесса, ул. Дворянская, 1/3, тел. 0634531509, vlad_mihailenko@mail.ru.

Харченко Роман Юрьевич, ст. преп. Одесской национальной морской академии. Научные интересы: интеллектуальные системы управления. Адрес: Украина, 65029, Одесса, ул. Дидрихсона, 8, тел. 0964535937, romannn30@gmail.com.

УДК 681.325.53:37 : 004.5

Н.Я. КАКУРИН, Е.В. БОЧАРОВ, В.В. ВАРЕЦА, К.В. ПОЛЕЖАЕВ, Ю.С. ЗАМАЛЕЕВ

ПРОГРАММНОЕ СРЕДСТВО ДЛЯ АНАЛИЗА ПРЕОБРАЗОВАНИЙ ЧИСЕЛ В ПРЕОБРАЗОВАТЕЛЯХ КОДОВ ПАРАЛЛЕЛЬНОГО ТИПА

Рассматривается назначение и возможности программного средства, моделирующего процессы преобразования чисел в преобразователях кодов, которые функционируют по методу накопления эквивалентов. Приводится сравнение последовательной и параллельной стратегий преобразования.

1. Постановка задачи

Достоинствами преобразователей кодов по методу накопления эквивалентов (ПК НКЭ) являются малые аппаратурные затраты (небольшое число корпусов ИС или число вентилей) и возможность изменения соотношения между быстродействием и аппаратурными затратами за счет выбора числа шагов преобразования, их значений и стратегии использования различных шагов преобразования [1-3].

Такая стратегия может быть последовательной или параллельной.

При последовательной стратегии показания разрядных счетчиков, значения которых равны или превышают значение шага, уменьшают на значение этого шага.

Если же во всех преобразуемых разрядах значения цифр оказываются меньше этого шага, происходит переход на меньший шаг, и далее ведется уменьшение значений разрядов на значение меньшего шага и т. д., пока не будут обнулены все разрядные счетчики.

Последовательная стратегия по сравнению с параллельной структурно реализуется достаточно просто. Аппаратурные затраты на построение основного блока ПК – формирователь элементов – будут наибольшими.

Для оценки сверху числа тактов преобразования целых чисел с последовательной стратегией в многошаговых ПК выполняют по формулам (1):

$$N_1^{цел} = K\text{-}1;$$

$$N_2^{цел} =](K\text{-}1)/a[+a\text{-}1; \qquad (1)$$

$$N_3^{цел} =](K\text{-}1)/b[+](b\text{-}1)/a[+a\text{-}1;$$

$$N_4^{цел} =](K\text{-}1)/c[+](c\text{-}1)/b[+](b\text{-}1)/a[+a\text{-}1,$$

где К – основание системы счисления на входе; a, b, c – соответственно второй, третий и четвертый шаги преобразования (первый шаг всегда равен 1); $N_1^{цел}$, $N_2^{цел}$, $N_3^{цел}$, $N_4^{цел}$ – соответственно максимальное число тактов преобразования одно-; двух-; трех-; четырехшагового ПК целых чисел;] [- означает округление до меньшего целого.

Для значений шагов преобразования должны выполняться следующие ограничения:

$$1< a< b;\ a < b < c;\ b < c \leq K\text{-}1.$$

Недостатками использования максимальных значений $N_i^{цел}$ $(i = \overline{1,4})$ для оценки быстродействия являются завышенные требования, которые вызывают необходимость применения в структуре ПК более быстродействующих ИС.

В целях реальной оценки быстродействия разработан программный пакет ”Transformation”, позволяющий промоделировать процесс преобразования чисел для последовательной стратегии и определить конкретное число тактов преобразования, а не оценку сверху.

Основными задачами работы являются:

– рассмотрение структуры и функционирования ПК НКЭ;

– анализ структуры и возможностей программного средства в режиме преобразования чисел;

– сравнительная оценка последовательной и параллельной стратегий по числу тактов преобразования.

2.Структура и функционирование ПК параллельного типа

Увеличения быстродействия ПК НКЭ можно достичь или за счет увеличения числа шагов в наборе до трех 1, a, b (b – третий шаг) при сохранении принципа последовательного использования шагов преобразования (вначале *b*, затем *a* и в заключение шаг 1), или путем параллельного (одновременного) вычитания различных шагов из различных разрядных цифр. Усиление локального параллелизма достигается благодаря возможности одновременного использования различных шагов преобразования в различных старших разрядах числа [4].

Выражение для формулы максимального числа тактов преобразования двухшагового ПК с параллельным применением шагов преобразования получать в явном виде затруднительно. Определить же значение максимального числа тактов преобразования можно путем моделирования процесса преобразования для различных наборов шагов и анализа результатов моделирования.

Результаты моделирования для К=10 и наборов шагов (1, 2) и (1,4) даны в табл. 1.

Таблица 1

Последовательная стратегия использования шагов		
Номер такта	К=10. Набор шагов (1,2)	К=10. Набор шагов (1,4)
	9 8 7 6 5 4 3 2 1 0	9 8 7 6 5 4 3 2 1 0
1	7 6 5 4 3 2 1 0 1 0	5 4 3 2 1 0 3 2 1 0
2	5 4 3 2 1 0 1 0 1 0	1 0 3 2 1 0 3 2 1 0
3	3 2 1 0 1 0 1 0 1 0	0 0 2 1 0 0 2 1 0 0
4	1 0 1 0 1 0 1 0 1 0	0 0 1 0 0 0 1 0 0 0
5	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0
Параллельная стратегия использования шагов		
Номер такта	К=12. Набор шагов (1,4)	К=12. Набор шагов (1,5)
	11 10 9 8 7 6 5 4 3 2 1 0	11 10 9 8 7 6 5 4 3 2 1 0
1	7 6 5 4 3 2 1 0 2 1 0 0	6 5 4 3 2 1 0 3 2 1 0 0
2	3 2 1 0 2 1 0 0 1 0 0 0	1 0 3 2 1 0 0 2 1 0 0 0
3	2 1 0 0 1 0 0 0 0 0 0 0	0 0 2 1 0 0 0 1 0 0 0 0
4	1 0 0 0 0 0 0 0 0 0 0 0	0 0 1 0 0 0 0 0 0 0 0 0
5	0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0

Из табл.1 следует, что набор шагов (1,4) для К=10 и параллельной стратегии обеспечивает преобразование числа не более чем за 4 такта. Для нахождения всех других наборов шагов следует провести моделирование для всех возможных наборов, что можно сделать с помощью программного средства “Transformation”.

Структурная схема двухшагового ПК НКЭ с параллельным использованием шагов приведена на рис.1.

Закон функционирования эквивалентов для 2-шагового ПК НКЭ с числом резисторов n=3, основанием К=10 и набором шагов (1,4) приведен в табл. 2.

Таблица 2

Номер набора	Состояние триггеров		Общий вид эквивалента	Десятичный код эквивалента	Двоичный код эквивалента S_2		
	Второй группы	Первой группы					
Z_i	$Д_3Д_2Д_1$	$C_3C_2C_1$	S	S_{10}	$У_{12}У_{11}У_{10}У_9$	$У_8У_7У_6У_5$	$У_4У_3У_2У_1$
0	00	00	X_0	X_0	Трансляция младшей тетрады		
1	00	01	K^1	10	0000	0000	1010
2	00	10	K^2	10	0000	0110	0100
3	00	11	$K^2 + K^1$	110	0000	0110	1110
4	01	01	aK	40	0000	0010	1000
5	01	11	$K^2 + aK$	140	0000	1000	1100
6	10	10	aK^2	400	0001	1001	0000
7	10	11	$aK^2 + K$	410	0001	1001	1010
8	11	11	$aK^2 + aK$	440	0001	1011	1000

Логика управления в двухшаговом ПК параллельного типа выполнена так, чтобы запретить возможность вычитания шага 1, если в этом разряде имеется возможность вычитания шага a. И наоборот, если значение разряда x_m находится в пределах $1 \le x_m < a$, то следует разрешить опрос вентиля, управляющего вычитанием 1 из разрядного счетчика, хранящего x_m.

Этот принцип управления реализуется в двухшаговом ПК параллельного типа путем замены n входового элемента ИЛИ-НЕ блоком инверторов (инверторы 17_1, 17_2), вход каждого из которых связан с единичным выходом соответствующего триггера старшего регистра состояний $РГ_4$, а выход соответствующего инвертора соединяется с управляю-

щими входами схем И-НЕ 8_1, И-НЕ 8_2, на информационные входы которых поступают сигналы с единичных выходов триггеров этого разряда младшего регистра состояний $РГ_3$ (рис. 1).

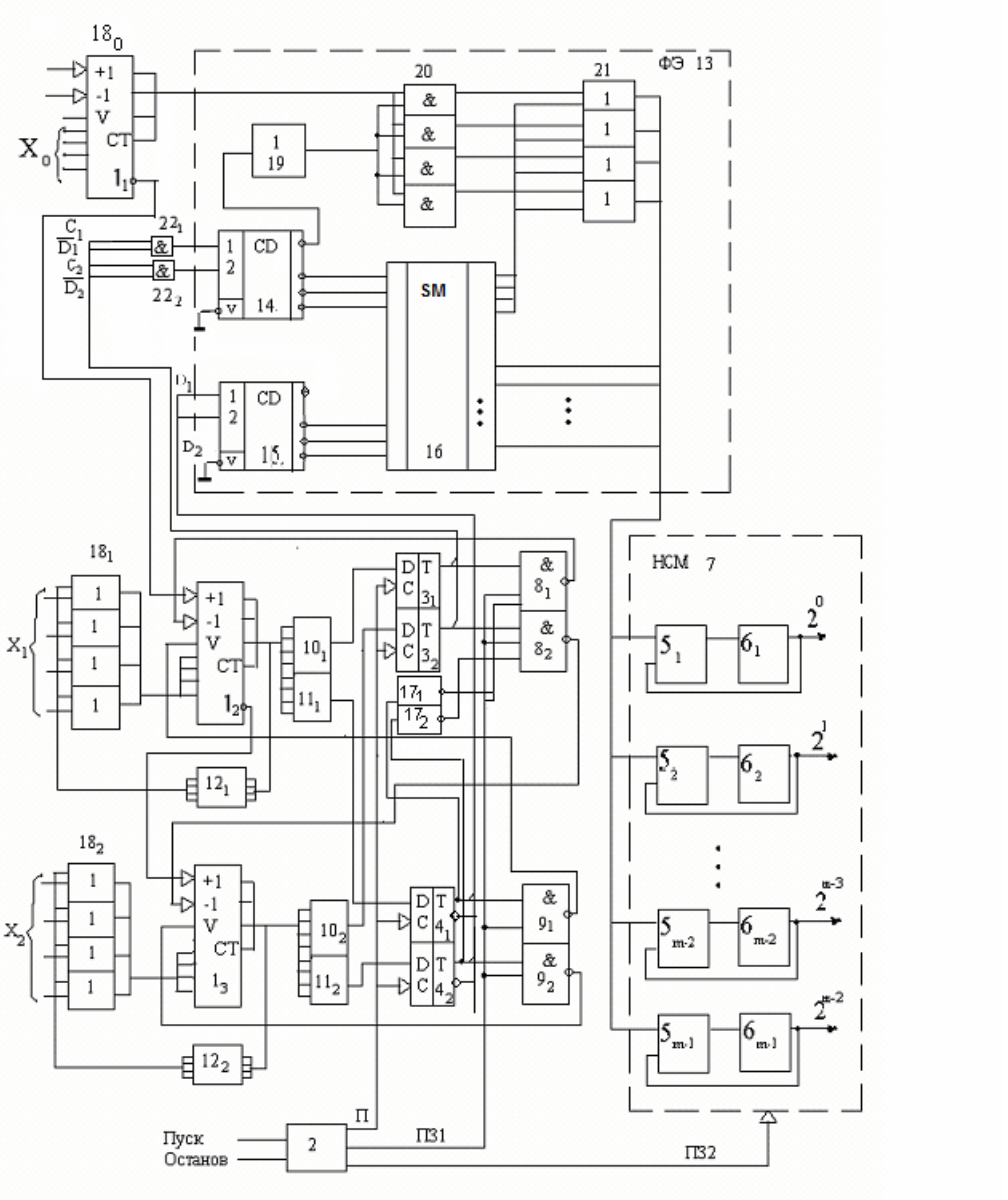


Рис. 1. Структура и функционирование двухшагового ПК параллельного типа

В целях оптимизации аппаратурных затрат на реализацию ФЭ (затраты на построение которого пропорциональны $c \cdot 2^P$, где P – число разрядов в блоке) он разбивается на M блоков. Это целесообразно при $n \geq 5$ [5]. Закон функционирования 2-шагового моноблочного ФЭ на текущем h-м шаге преобразования в общем виде можно описать выражением:

$$S_m = \sum_{i=(m-1)P+1}^{i=mP} \gamma_i(h) K^{i-1} \cdot R_i(h), \; (i = \overline{1, mP}), \qquad (2)$$

где m-номер блока ФЭ $m = \overline{1, M}$; i – номер разряда в блоке; n – общее число преобразованных разрядов. Очевидно, что $M = n / P$, а коэффициент $y_i(h)$ при степени основания K^{i-1} может принимать три значения в зависимости от состояния триггеров D_i и C_i (3):

$$\gamma_i(h)=\begin{cases} a, D_i \neq 0; C_i \neq 0; \\ 1, D_i = 0; C_i \neq 0; \\ 0, D_i = 0; C_i = 0. \end{cases} \quad (3)$$

Коэффициент $R_i(h)$ учитывает вхождение компоненты i-го разряда в (2) для блока m и находится из выражения:

$$R_i(h)=\begin{cases} 0, D_i = C_i = 0; \\ 1, D_i \cup C_i = 1. \end{cases} \quad (4)$$

В зависимости от текущего шага преобразования $h(h=\overline{1,H})$, $y_i(h)$ и $R_i(h)$ могут изменяться, т.е. являются динамически изменяющимися, что относится также и к величине S_m. Число блоков M разбиения ФЭ, как правило, является делителем числа входных разрядов n. В этом случае блоки содержат одинаковое число разрядов p. Чтобы это условие выполнялось всегда, младший преобразуемый разряд x_0 следует не транслировать (передавать без изменений) через вентили И20 и схемы ИЛИ21, а включить в состав p разрядов блока 1. Степень K в формуле (2) соответствует этому случаю.

3. Назначение и структура программного средства «Transformation»

К назначениям программного средства относятся:

– возможность имитации работы преобразователя кодов по методу накопления эквивалентов;

– возможность работы в двух режимах: режиме преобразования целых и дробных чисел, режиме статистики преобразования целых и дробных чисел.

При разработке данного программного продукта было выбрана наиболее оптимальная по скорости проектирования и эффективности визуальная среда программирования QT для Windows XP/7, программа реализована на языке C++.

Для нормальной работы данной программы необходимы следующие технические средства: процессор не менее 1Ггц, 512 мб ОЗУ, 50 мю дискового пространства, Windows XP/ Vista/7.

Разработанное программное обеспечение сформировано как независимый модуль и для своего функционирования не требует дополнительного программного обеспечения. На рис. 2 представлена структурная схема алгоритма режима преобразования чисел

Режим преобразования чисел используется для преобразования числа из K-ичной системы счисления в двоичную. Программа выдает не только окончательный результат преобразования, но и промежуточные результаты в виде таблицы: состояние счетчика, величину эквивалента и состояние накапливающего сумматора на каждом такте преобразования. Имеется возможность задавать различные число шагов преобразования от 1 до 8, основание системы счисления на входе преобразователя – от 3 до 16, разрядность числа – от 2 до 12, веса шагов преобразователя.

Для того чтобы произвести расчет в данном режиме, необходимо запустить программу путем активизации исполняемого модуля transformation.exe.

После запуска откроется главное меню программы (рис.3) , где можно ввести исходные данные. Программа имеет вкладочный интерфейс, благодаря которому работать с программным средством просто и удобно.

В этом окне имеются пять текстовых полей для ввода данных, три из которых снабжены кнопками инкремента и декремента. С помощью этих элементов управления необходимо задать исходные данные для режима преобразования чисел: количество шагов преобразователя, основание системы счисления на входе, разрядность преобразуемого числа, собственно само число, которое необходимо преобразовать, и значения весов преобразователя. Программа содержит ряд проверок, которые предотвращают ввод некорректных данных.

Когда все значения будут введены в окно, следует нажать кнопку ОК для запуска преобразователя. После того, как программа произведет расчет, на экране высветится окошко с результатами преобразования введенного числа (рис.3).

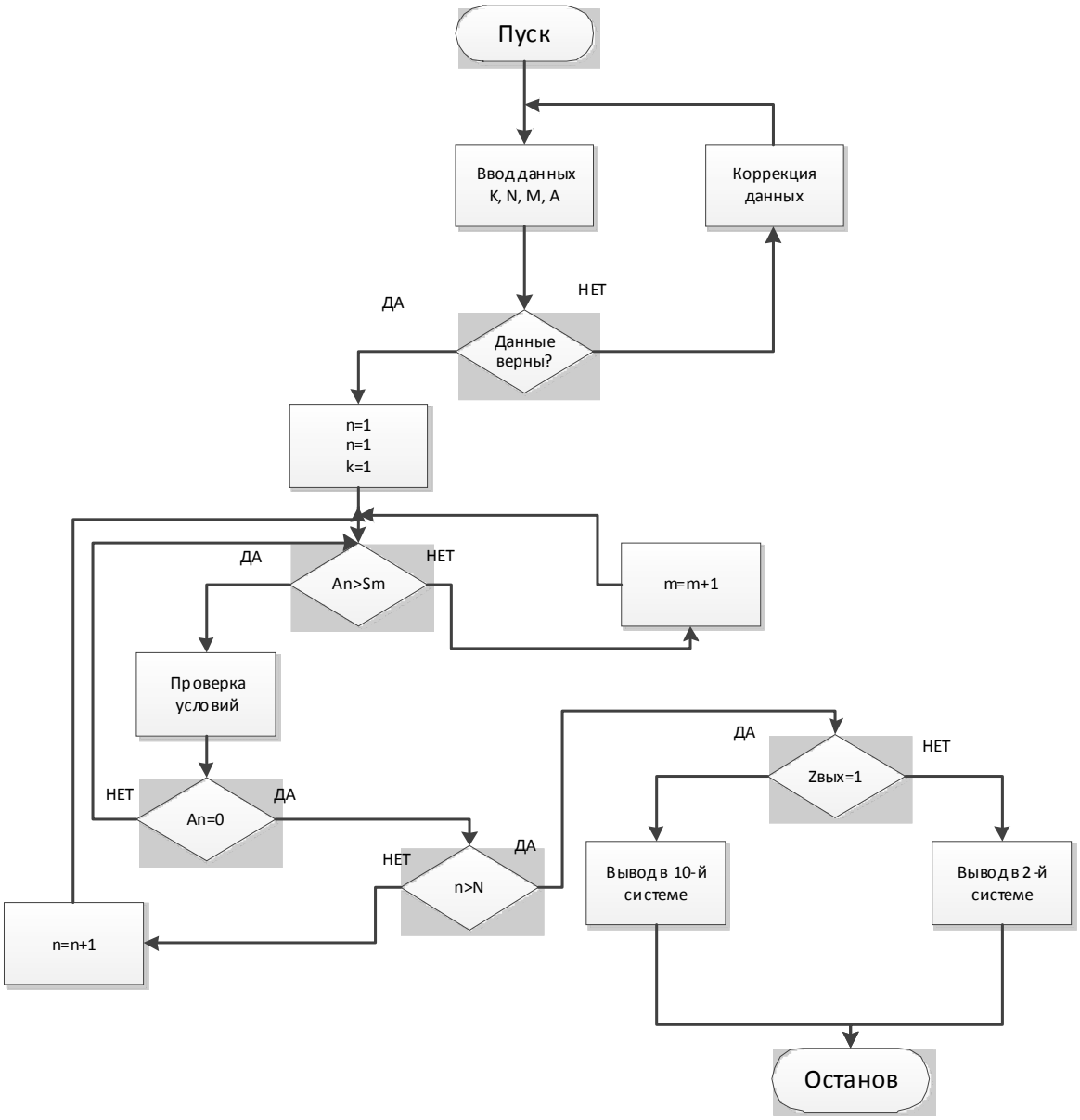


Рис. 2. Структурная схема алгоритма режима преобразования чисел

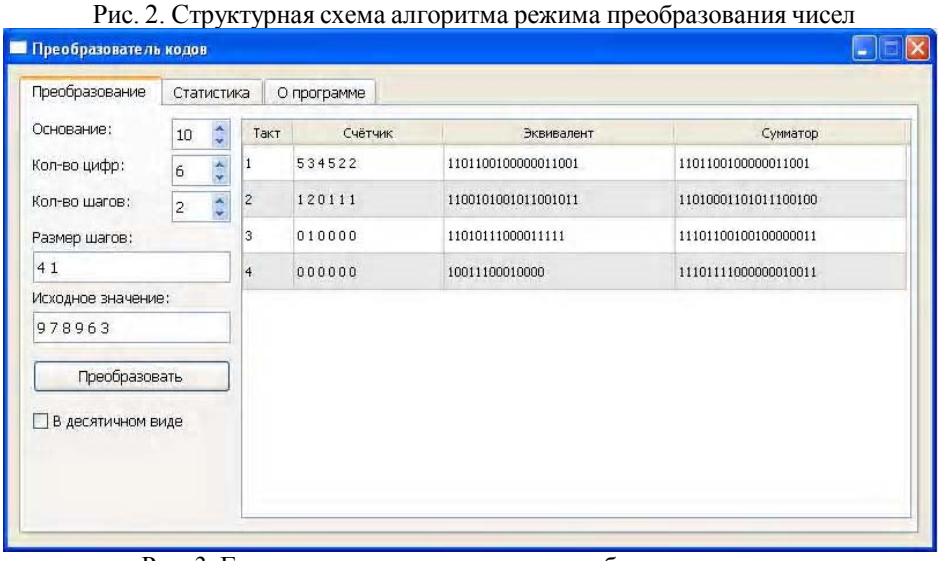


Рис. 3. Главное окно с результатами преобразования числа

Программа отображает значения эквивалентов и состояния накапливающего сумматора в десятичном виде. Для того чтобы увидеть двоичные эквиваленты, необходимо отметить поле «В десятичном виде», после чего в окошке с результатами отобразится двоичное представление данного числа. Код функции, реализующей преобразование из двоичного в десятичный для отображения в программе, представлен ниже:

```
    int showRadix = ui->checkBoxDecimal->checkState() == Qt::Checked ? 10 : 2;
do
{
   registerValue = 0;
   equivalentValue = 0;
   for (int i = 0; i < digitsCount; ++i)
   {
      quint8 digit = registerValues.at(i).toInt(),
           equivalent = 0;
      for (int j = 0; j < stepsCount; ++j)
      {
         quint8 step = stepsSizes.at(j).toInt();
         if (step <= digit && step > equivalent)
            equivalent = step;
      }
      digit -= equivalent;
      registerValues.replace(i, QString("%0").arg(digit));
      registerValue += digit * power(radix, digitsCount - i - 1);
      equivalentValue += equivalent * power(radix, digitsCount - i - 1);
   }
   sum += equivalentValue;
   item = new QStandardItem(QString::number(row + 1, 10));
   model.setItem(row, 0, item);
   item = new QStandardItem(registerValues.join(" "));
   model.setItem(row, 1, item);
   item = new QStandardItem(QString::number(equivalentValue, showRadix));
   model.setItem(row, 2, item);
   item = new QStandardItem(QString::number(sum, showRadix));
   model.setItem(row, 3, item);
   ++row;
   }
```

В алгоритме реализован ряд проверок, в частности проверка на совпадение количества разрядов размерности регистров. С помощью разработанного программного средства для различных пар шагов (1, a) 2-шагового ПК НКЭ получены значения числа тактов преобразования (столбец 2 в табл.3). В столбце 1 табл. 3 приведены значения числа тактов преобразования для последовательной стратегии использования шагов.

Таблица 3

Второй шаг	Основание системы счисления К																			
	12		11		10		9		8		7		6		5		4		3	
a	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
2 3 4 5	6 5 5 6	6 5 5 5	6 5 5 6	5 4 4 5	5 5 5 5	5 4 4 5	5 4 5 5	4 4 4 4	4 4 4 5	4 3 4 4	4 4 4 5	3 3 3 4	3 3 3 4 5	3 3 3 4	3 3 3 4	2 2 2 3	2 3	2 2	2	1
6 7 8 9 10 11	6 7 8 9 10 11	6 6 7 8 9 10	6 7 8 9 10	5 6 7 8 9	6 7 8 9	5 6 7 8	6 7 8	5 6 7	6 7	5 6	6	5								

Из анализа табл.3 следует, что для К=11 и К=10 при параллельной стратегии использования шагов достигается уменьшение числа тактов преобразования по сравнению с последовательной на 20%, для К=8 и К=7- на 25%, для К=5 – на 33%, и для К=3 на 50%. Для значений К=12, 9, 6, 4 выигрыш отсутствует.

Выводы

1. Рассмотрена структура и функционирование ПК НКЭ с параллельной стратегией использования шагов.

2. Проанализирована математическая модель моноблочного многоразрядного ФЭ для параллельной стратегии, на базе которой может выполняться построение законов функционирования ФЭ различных блоков.

3. Предложено новое программное средство “Transformation”, позволяющее моделировать процесс преобразования чисел в ПК НКЭ и выполнять его анализ при различных основаниях и наборах шагов.

Практическая значимость результатов заключается в возможности проведения детального анализа в целях нахождения оптимальных наборов шагов, а также значений внутренних и выходных результатов как в десятичной системе счета, так и в двоичной.

.**Список литературы: 1.** *Самофалов К.Г., Корнейчук В.И., Тарасенко В.П.* Цифровые ЭВМ.Киев: Вища школа, 1989. 424 с. **2.** *А.С. 1647908* 5НОЗМ 7/12. Преобразователь двоично-К-ичного кода в двоичный код /*Н.Я.Какурин, Ю.К. Кирьяков, А.Н. Макаренко* // Открытия, изобретения. 1991 №17. с. 262-263. **3.** А.С. 1783618 5НОЗМ 7/12. Преобразователь двоично-К-ичного кода в двоичный код /*Н.Я.Какурин, А.Н. Макаренко, Д.Ю. Исхаков,В.А. Толмацкий* // Открытия, изобретения. 1984 .№44. С. 250. **4.** *Какурин Н.Я., В.В. Вареца, С.Н. Коваленко.* Параллельная стратегия использования шагов в двухшаговых преобразователях кода //АСУ и приборы автоматики. 2007. Вып.141. С.29-36. **5.** *Голян В.В., Какурин Н.Я., Макаренко А.Н., Замалеев Ю.С., Николаев А.А.* Двухкритериальный системный синтез преобразователей кодов по методу накопления эквивалентов//АСУ и приборы автоматики. 2005. Вып. 133. С.102-107.

Поступила в редколлегию 28.02.2011

Какурин Николай Яковлевич, канд.техн.наук, профессор кафедры АПВТ ХНУРЭ. Научные интересы: прикладная теория цифровых автоматов, автоматизация проектирования цифровых устройств. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326.

Бочаров Евгений Витальевич, студент группы КИ-07-6 ХНУРЭ. Научные интересы: автоматизация проектирования цифровых устройств, проектирование программного обеспечения. Адрес: Украина, 61166, Харьков, пр.Ленина, 14, тел. 70-21-326.

Вареца Виталий Викторович, аспирант кафедры АПВТ ХНУРЭ. Научные интересы: проектирование программного обеспечения, автоматизация проектирования цифровых устройств. Адрес: Украина, 61166, Харьков, пр.Ленина, 14, тел. 70-21-326.

Полежаев Кирилл Вадимович, студент группы СИ-07- ХНУРЭ. Научные интересы: автоматизация проектирования цифровых устройств, проектирование программного обеспечения. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-326.

Замалеев Юрий Салихович , канд. техн. наук, доцент кафедры ПМ и ИТ Харьковской национальной академии городского хозяйства. Научные интересы: моделирование сложных систем управления, программирование, теория цифровых интегрирующих структур. Адрес: Украина, 61204, Харьков, пр. Победы, 68-А, кв.17, тел. 336-77-84.

УДК 004.896

Р. АЛЬХАЙЕК, С.Г. УДОВЕНКО, А.А. ШАМРАЕВ

НЕЧЕТКОЕ ЦИФРОВОЕ УПРАВЛЕНИЕ КОМПРЕССОРНОЙ СТАНЦИЕЙ ГАЗОТРАНСПОРТНОЙ СИСТЕМЫ

Рассматривается подход к управлению газоперекачивающими агрегатами компрессорных станций газотранспортной системы с помощью нечетких регуляторов. Предлагается структура системы цифрового управления, основанная на применении внутренних нечетких моделей. В качестве компенсационного регулятора используется инверсная нечеткая модель объекта управления. Приводятся результаты моделирования системы нечеткого управления компрессорной станцией, подтверждающие эффективность применения инверсных нечетких моделей для поддержания максимального или заданного давления газа на выходе компрессорной станции.

1. Введение

В настоящее время при управлении газоснабжением возникают новые технические и экономические задачи, решить которые без создания эффективной системы автоматизированного управления технологическими процессами с применением современных вычислительных средств невозможно.

Конечная цель создания автоматизированной системы управления технологическими процессами транспорта газа состоит в повышении эффективности работы газотранспортной системы за счет оперативного планирования, централизованного контроля и управления режимами работы основных технологических объектов газопровода в реальном времени.

Оптимальный режим эксплуатации магистральных газопроводов определяется работой компрессорных станций. Задачи управления технологическими процессами (ТП) компрессорной станции (КС) состоят в регулировании расхода, давления и температуры газа. Управление газоперекачивающими агрегатами (ГПА) осуществляется автономными автоматическими системами регулирования. Существуют регуляторы давления газа, многоканальные измерители и регуляторы температуры. В настоящее время регулирование осуществляется преимущественно с использованием микроконтроллерных ПИД-регуляторов [1]. Для организации эффективного функционирования систем управления компрессорной станцией целесообразно применять системы с элементами искусственного интеллекта [2]. Модель нечеткого управления компрессорной станцией, как и модель любой системы управления с нечетким описанием параметров, строится на основе формализации субъективных знаний экспертов.

Рассмотрим возможность и целесообразность применения для управления ТП компрессорной станции моделей с нечетким описанием исходных данных и нечетким логическим выводом. Режим работы компрессорной станции зависит от типов, схемы соединения, оборотов ГПА, температуры и состава газа.

Предположим, что существует инверсная система, способная в соответствии с заданной желаемой траекторией вырабатывать управляющий сигнал, который позволяет добиться сходимости управляемого выхода ГПА (заданного давления газа) к желаемой траектории. В этом случае синтез регулятора можно свести к проблеме определения инверсной системы. Идея использования инверсных моделей для класса линейных и нелинейных систем получила развитие в работах [3, 4]. Представляется целесообразным рассмотреть возможность применения нечетких систем Такаги-Сугено в качестве инверсных моделей ТП КС. Однако при наличии неопределенностей и возмущений реализация подобной схемы управления в открытом контуре может привести к низкому качеству работы, синтезируемой системы и даже к неустойчивым режимам. *Целью данной работы* является исследование возможности и целесообразности использования инверсных нечетких компенсационных регуляторов в замкнутых системах цифрового управления с внутренней нечеткой моделью, отражающей динамические свойства технологических процессов компрессорной станции газоперекачивающей системы.

2. Принцип инверсного управления с внутренней нечеткой моделью

Определение нелинейной модели объектов цифрового управления может быть трудной задачей, особенно для сложных процессов (в частности, для ТП КС). Альтернатива решению этой проблемы заключается в использовании аппроксимирующих возможностей нечетких систем для представления нелинейных динамических процессов. Синтез регулятора при этом может основываться на инверсии нечеткой модели. При этом может быть использован принцип управления, основанный на структуре с внутренней моделью (ВМ-структуры). Идея применения ВМ-структур получила развитие преимущественно для управления линейными объектами. ВМ-структуру можно преобразовать в эквивалентную ей классическую структуру. В работе [4] показано, что дискретная передаточная функция цифрового регулятора для линейной системы, соответствующего классической структуре, определяется следующим соотношением:

$$R(z)=\frac{Q(z)}{1-Q(z)G_0(z)}, \qquad (1)$$

где $G_0(z)$ – дискретная передаточная функция модели объекта управления; $Q(z)$ – дискретная передаточная функция инверсной модели объекта.

Если объект управления является минимально-фазовым, а используемая модель достаточно точной, то регулятор с дискретной передаточной функцией вида (1), обеспечивает устойчивость и приемлемые характеристики качества управления даже при наличии возмущений на входе и выходе управляемого объекта. Это делает привлекательным использование ВМ-структур для создания системы управления ГПА, работающего в условиях постоянно действующих возмущений. Рассмотрим возможность применения нелинейных ВМ-структур с использованием нечетких внутренних моделей объекта управления.

Структура системы инверсного управления с внутренней нечеткой моделью (ИУВНМ) приведена на рис. 1.

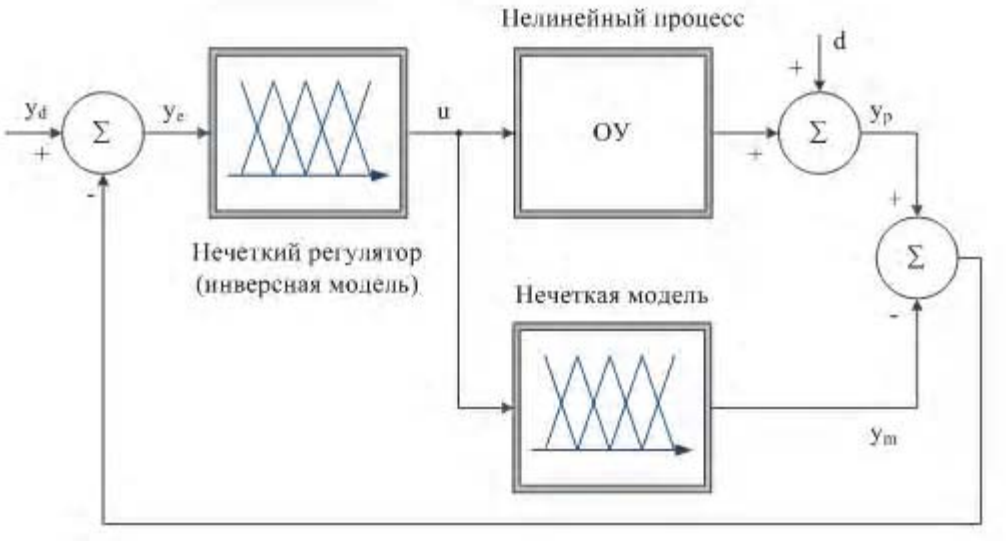


Рис. 1. Система инверсного управления с внутренней нечеткой моделью

Управление y_d, вырабатываемое компенсационным регулятором (инверсной нечеткой моделью), подается одновременно на процесс и его модель. Сравнение выходов процесса y_p и модели y_m позволяет генерировать сигнал ошибки, который используется для изменения входа регулятора y_e в целях улучшения управления u. Если модель является идеальным представлением процесса, то сигнал ошибки равен сигналу возмущения d. В этом случае управляющая структура эквивалентна схеме открытого контура управления. На практике процесс и его модель редко бывают полностью идентичны. В этом случае

обратный сигнал представляет собой сумму сигнала возмущения и ошибки моделирования. Очевидно, при условии устойчивости процесса, если в качестве регулятора выбрана инверсная нечеткая модель, то контур ИУВНМ реализует идеальную передачу задающего сигнала и выхода процесса независимо от объекта. Регулятор, синтезированный по такому принципу, осуществляет качественное управление, несмотря на возмущения. Однако, если ошибка рассогласования выходов модели и процесса является значительной, то возникает опасность потери устойчивости в контуре управления, что вызывает необходимость применения фильтрации сигнала обратной связи для обеспечения определенной робастности по отношению к ошибкам моделирования.

На практике инверсия модели не всегда возможна (например, когда ОУ является неминимально-фазовым). В ряде работ были предложены методы решения этой проблемы. В частности, в [4] предлагается метод частичной инверсии модели. Он состоит в декомпозиции модели на две части, содержащие соответственно устойчивые и неустойчивые нули. В этом случае регулятор выбирается как инверсия устойчивой части исходной модели, а для неустойчивой части используется коэффициент передачи, равный единице.

Рассмотрим подход к инверсному нечеткому управлению нелинейными процессами, основанный на достижении в каждый момент линейности ARMA-модели для каждой элементарной нечеткой ячейки. Преимуществом этого подхода является возможность синтеза регулятора, основанного только на параметрах линейных разностных моделей. При этом модель и ее инверсия (регулятор) являются линейными. В этом случае можно легко анализировать обратимость модели по ее нулям. Метод, используемый для обеспечения линейности каждой элементарной нечеткой ячейки, основан на принципе декомпозиции, подробно рассмотренном в [5].

В соответствии с этим принципом выход, генерируемый каждой элементарной нечеткой подсистемой i, может быть аппроксимирован линейной системой следующим образом:

$$[y(k+1)]_i = c_0^{(i)} + \sum_{p=1}^{n} c_p^{(i)} y(k-p+1) + \sum_{q=1}^{m} d_q^{(i)} u(k-q+1), \qquad (2)$$

где $i \in \{1,\ldots,\prod_{p=1}^{n}(N_p-1)\times\prod_{q=1}^{m}(M_q-1)\}$; n,m – количество строк и столбцов в элементарных ячейках; c,d – настраиваемые параметры модели.

В этом случае глобальная нечеткая модель аппроксимируется набором линейных систем. Если мы обозначим:

$$C^{(i)}(z) = 1 - \sum_{p=1}^{n} c_p^{(i)} z^{-p}; \; D^{(i)}(z) = \sum_{q=1}^{m} d_q^{(i)} z^{-q+1}, \qquad (3)$$

то каждая линейная система может быть представлена в виде:

$$[y(k+1)_i = \frac{c_0^{(i)}}{C^{(i)}(z)} + \frac{D^{(i)}(z)}{C^{(i)}(z)} u(z)]. \qquad (4)$$

Этот принцип разделения проиллюстрирован на рис.2.

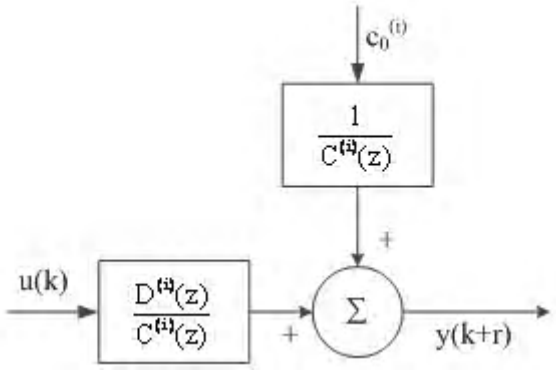


Рис. 2. Иллюстрация принципа разделения блоков элементарной подсистемы

Таким образом, эквивалентная система управления может быть представлена структурой, приведенной на рис. 3. В каждом такте управления для набора [y(k),y(k-1),u(k)], определяемого по сформированным заранее функциям принадлежности и базам правил нечеткой системы, активизируется соответствующая линейная подсистема (элементарная нечеткая ячейка). Значение выходного сигнала регулятора формируется по результатам инверсии активизированной элементарной ячейки и применения полученного линейного уравнения (дефаззификации Такаги-Сугено). Для инверсии глобальной нечеткой системы, представленной совокупностью элементных нечетких подсистем, необходимо осуществить инверсию каждой из этих подсистем. С практической точки зрения важно также определить, все ли подсистемы необходимо инвертировать или некоторые из них можно исключить из рассмотрения. Вычисления по рассмотренному выше алгоритму предполагают, что для каждой подсистемы, содержащей одну входную переменную, известны значения остальных (n-1) входов. В случае монотонности набора правил решение задачи инверсии является единственным.

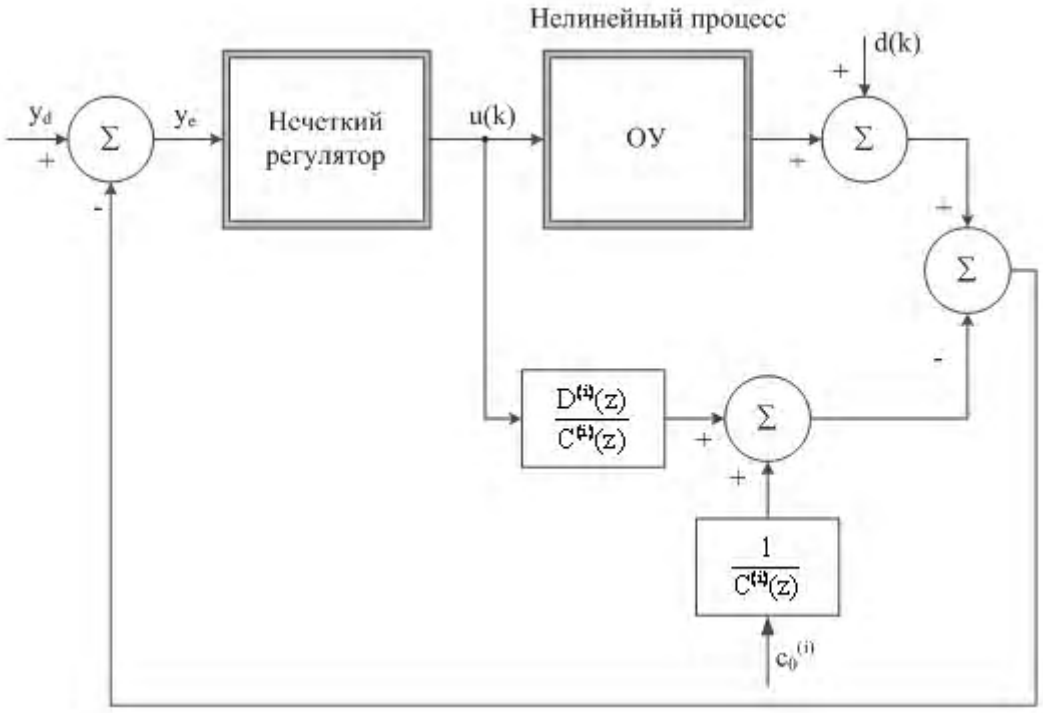


Рис. 3. Эквивалентная схема инверсного управления с внутренней нечеткой моделью

3. Моделирование нечеткого цифрового регулятора для управления ГПА

Для моделирования системы нечеткого управления ГПА КС использовались возможности программной среды MATLAB, в частности, пакета для моделирования динамических систем Simulink и пакета синтеза нечетких систем управления Fuzzy Logic Toolbox.

Для разных диапазонов выходного давления (или степени сжатия газа) динамические свойства ГПА могут изменяться (в силу нелинейности статической зависимости по каналу «скорость оборотов вала ГПА – выходное давление»). В связи с этим было проведено разделение общего диапазона изменения выходного давления на 5 интервалов, что позволило после выбора функций принадлежности сформировать 5 линейных подсистем (и, соответственно, 5 элементарных нечетких ячеек). Правила определения выходов для каждой из этих подсистем приведены в таблице.

В соответствии с таблицей структура системы нечеткого управления ГПА КС имеет вид, представленный на рис. 4. Очевидно, что такая структура позволяет в реальном времени после определения текущего состояния системы фиксировать номер элементарной нечеткой ячейки и формировать соответствующее изменение скорости оборотов вала ГПА.

Таблица выходов элементарных подсистем модели ГПА

Номер подсистемы	Элементарная нечеткая ячейка	Выход y(k+1)
1	(1,1)	$0{,}30y(k)+0{,}999u(k)+1{,}603$
2	(2,1)	$-0{,}258y(k)+0{,}999u(k)+1{,}078$
3	(3,1)	$-0{,}574y(k)+0{,}999u(k)+1{,}052$
4	(4,1)	$0{,}193y(k)+0{,}999u(k)+0{,}453$
5	(5,1)	$0{,}158y(k)+0{,}999u(k)+0{,}511$

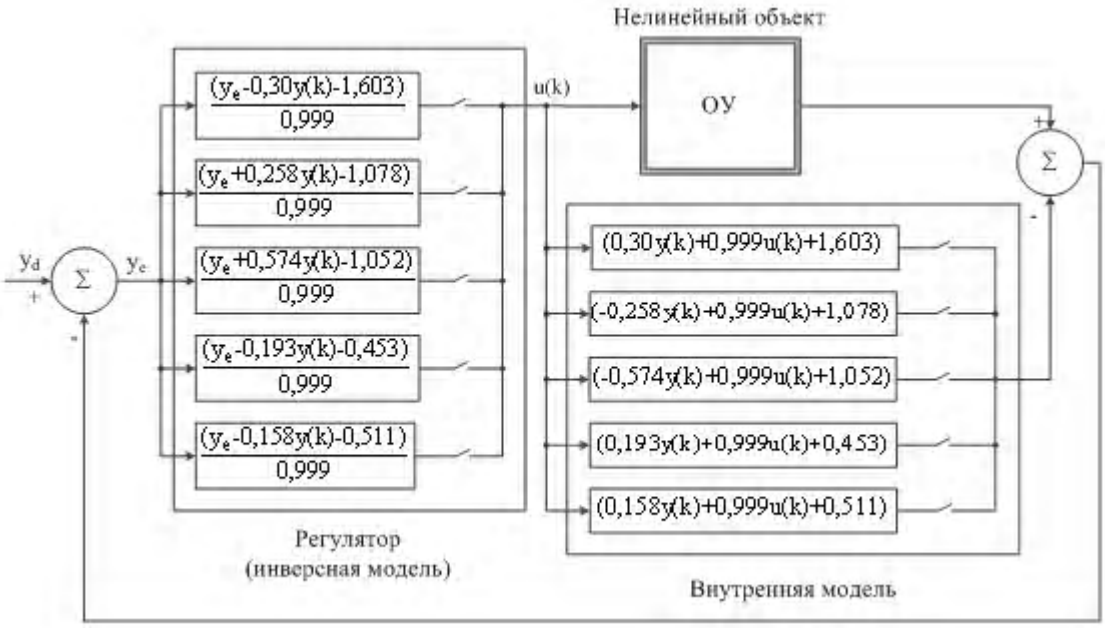


Рис. 4. Структура системы нечеткого управления ГПА с внутренней моделью

Общая схема моделирования системы управления ГПА с нечетким регулятором представлена на рис. 5. В предложенном цифровом SISO-регуляторе реализован описанный выше метод нечеткого управления динамическим процессом, основанный на использовании инверсии внутренней нечеткой модели.

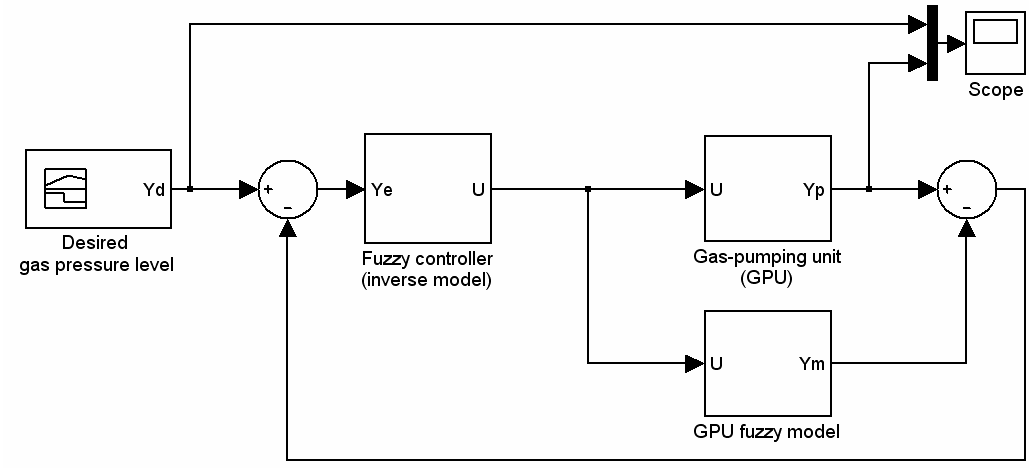


Рис. 5. Схема моделирования системы управления ГПА с нечетким регулятором

На рис. 6 приведены результаты моделирования работы системы при скачкообразном изменении задающего воздействия от 8,0 до 8,5 МПа (диапазон элементарной ячейки (3,1))

и последующем его понижении до 8,0 МПа. Изменение режима работы ГПА осуществляется на верхнем уровне автоматизированной системы управления технологическими процессами КС.

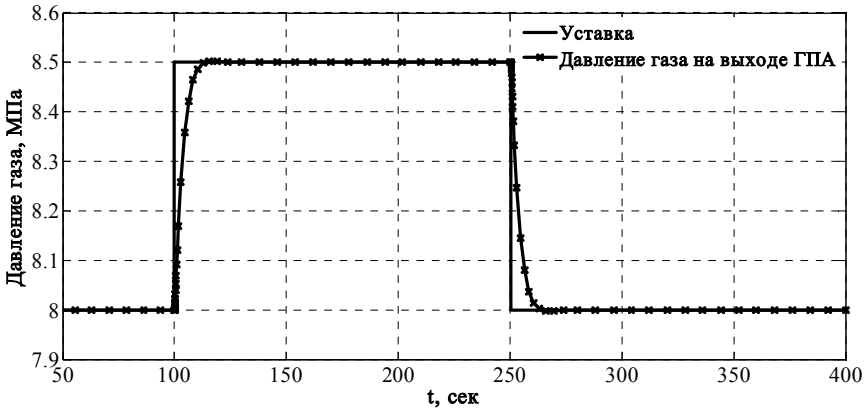


Рис. 6. Результаты моделирования

Результаты моделирования подтверждают работоспособность предложенного нечеткого регулятора в составе системы управления ГПА КС, что свидетельствует о возможности практического использования описанных в данной статье результатов.

4. Выводы

Научная новизна полученных результатов заключается в системном подходе к синтезу системы управления ГПА КС с учетом действующих возмущений и существующих ограничений. Предложенный подход к управлению ГПА КС, основанный на применении инверсной внутренней нечеткой модели, состоящей из набора элементарных нечетких ячеек, позволяет синтезировать регулятор, который сохраняет работоспособность в условиях постоянно действующих возмущений как на входе, так и на выходе объекта управления. *Практическая значимость* заключается в теоретическом и экспериментальном подтверждении возможности и целесообразности применения нечетких ВМ-структур для создания систем цифрового управления технологическими процессами компрессорных станций газотранспортной системы. *Перспективным* представляется развитие теоретического обоснования предложенного подхода и тестирование полученных результатов для различных типов возмущений, действующих на газоперекачивающие агрегаты.

Список литературы: 1. *Ісаков А.Т., Хохряков М.В., Фланчик Б.С. та ін.* Експлуатація і технічне обслуговування газорозподільних станцій магістральних газопроводів (довідник). Київ: Росток, 2003.411 с. **2.** *Боженюк А.В., Шадрина В.В.* Нечеткая классификация ситуаций и принятие решений в системах магистрального транспорта.// Известия ТРТУ - Таганрог: Изд-во ТРТУ, 2006. № 10 (65). С. 9-12. **3.** *Babuska R., Sousa J., Verbruggen H.B.* Modelbased design of fuzzy control systems. Proceedings of the International Conference EUFIT'95, vol.1. Aachen, Germany. P. 1115-1119. **4.** *Пегат А.* Нечеткое моделирование и управление. М.: БИНОМ. Лаборатория знаний, 2011. 798с. **5.** *Альхайек Р., Удовенко С.Г.* Модифицированный метод построения инверсной нечеткой модели объекта цифрового управления / Системи управління, навігації та зв'язку. 2009. Вип.4(12). С.130–134.

Поступила в редколлегию 09.03.2011

Альхайек Ранем, аспирант кафедры электронных вычислительных машин ХНУРЭ. Научные интересы: нечеткая идентификация нелинейных систем, нейро-нечеткое управление. Адрес: Украина, 61166, Харьков, пр. Ленина, 14.

Удовенко Сергей Григорьевич, д-р техн. наук, профессор кафедры электронных вычислительных машин ХНУРЭ. Научные интересы: управление стохастическими процессами, методы вычислительного интеллекта. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-354.

Шамраев Анатолий Анатольевич, канд. техн. наук, доцент кафедры электронных вычислительных машин ХНУРЭ. Научные интересы: нейро-нечеткое управление, разработка и оптимизация микроконтроллерных систем. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 70-21-354.

УДК 519.7

А.Н. ГВОЗДИНСКИЙ, В.А. МАЛЫШКИН, С.В. ТКАЧЁВ

ПРИМЕНЕНИЕ МЕТОДОВ ЭВОЛЮЦИННОЙ ОПТИМИЗАЦИИ ДЛЯ РЕШЕНИЯ ЗАДАЧ ПРОИЗВОДСТВЕННОГО ПЛАНИРОВАНИЯ

Рассматривается многокритериальная задача принятия решений с использованием методов эволюционной оптимизации в системах производственного планирования.

1. Введение

В последнее время проявляется тенденция использования естественных аналогов при создании моделей технологий методик алгоритмов для решений тех или иных задач, стоящих перед человечеством. В большинстве случаев применение естественных аналогов дает положительные результаты. Как правило, это объясняется тем, что аналог, взятый из природы, совершенствовался в течение многих лет эволюции и имеет в данный момент самую совершенную в своем роде структуру.

В настоящее время наиболее известным представителем эволюционных алгоритмов является генетический алгоритм (ГА), который содержит все основные генетические операции. Он получен в процессе обобщения и имитации в искусственных системах свойств живой природы:

– приспособляемость к изменениям среды;
– естественный отбор;
– наследование потомками наиболее « ценных» свойств родителей и т.д.

С его помощью можно улучшить работу поисковых систем, которые требуют обработки больших массивов информации.

Среди основных трудностей в использовании генетического алгоритма -возможность эффективно сформулировать задачу, определить рациональный выбор функции приспособленности и хромосом, которые описывают особей популяции , являются эвристическими и под силу только специалисту.

Актуальность исследования. Одной из особенностей предлагаемого метода является отход от традиционной схемы «размножения», используемой в большинстве реализованных ГА-max и повторяющих классическую схему, предложенную Голландом. Классическая схема предполагает ограничение численности потомков путем использования так называемой вероятности кроссовера. Такая модель придает величине, соответствующей численности потомков, вообще говоря, недетерминированный характер. Мы предлагаем отойти от вероятности кроссовера и использовать фиксированное число брачных пар на каждом поколении, при этом каждая брачная пара «дает» двух потомков. Такой подход хорош тем, что делает процесс поиска более управляемым и предсказуемым в смысле вычислительных затрат.

Используя генетическую информацию хромосомных наборов родителей в качестве генетических операторов получения новых генотипов «потомков», мы применяли два типа кроссоверов - одно и двухточечный. Вычислительные эксперименты показали, что даже для простых функций нельзя говорить о преимуществе того или иного оператора. Более того, было показано, что использование механизма случайного выбора одно- или двухточечного крассовера для каждой конкретной брачной пары иногда оказывается более эффективным, чем детерминированный подход к выбору кроссоверов , поскольку достаточно трудно априорно определить, который из двух операторов более подходит для каждого конкретного ландшафта приспособленности. Использование же случайного выбора преследовало цель, прежде всего, сгладить различия этих двух подходов и улучшить показатели среднего ожидаемого результата. Для всех представленных тестовых функций так и произошло. Метод случайного выбора оказался эффективнее худшего. Кроме того, в ряде случаев (функции Griewanka'a , 10-мерная функция Растригина) применение случайного механизма в выборе кроссовера дала лучшие результаты по сравнению с детерминированными подходами.

Повышение эффективности поиска при использовании случайного выбора операторов кроссовера вызвало необходимость применения аналогичного подхода при реализации процесса мутагинеза новых особей, однако в этом случае преимущество перед детерминированным подходом не так очевидно в силу традиционно малой вероятности мутации (в наших экспериментах вероятность мутации составляла 0.001 – 0.01).

Цель работы: исследовать возможность использования методов эволюционной оптимизации для решения многокритериальных задач в производственных системах управления.

Задачи исследования. В рамках данной статьи рассмотреть 4 метода эволюционной оптимизации производственного планирования

Сущность исследования. В качестве основного аппарата для разработки систем производственного планирования используется многокритериальная оптимизация. Из множества качественных показателей для оценки взяты основные, среди которых можно выделить следующие:

1. Величина прибыли, получаемой предприятием, определяется с помощью соотношения:

$$F_1(X) = \sum_{j=1}^{n} C_j^{(1)} X_j \to \max, X_j \in Q = \{1,2,...,n\}, \quad (1)$$

где Q – множество видов продукции, выпускаемых предприятием.

2. Показатель качества выпускаемой продукции задается соотношением:

$$\sum_{i=1}^{S} P_i * X_i \to \max. \quad (2)$$

Для конкретных значений функция цели примет вид:

$$F_2(X) = 10x_1 + 12x_2 + 8x_3 + 16x_4 + 11x_5 \to \max. \quad (3)$$

3. Минимизация себестоимости

$$\sum_{i=1}^{S} C_i * X_i \to \min. \quad (4)$$

4. Минимизация производственного времени:

$$\sum_{i=1}^{S} T_i * X_i \to \min. \quad (5)$$

Тогда задача исследования может быть сформулирована таким образом.

Определить оптимальный план $X^{(0)} \in Q$ производства продукции, удовлетворяющий указанным критериям (1) – (4).

Ограничениями на выпуск продукции различных видов служат производственные ресурсы $b_1, b_2, \ldots, b_m$. С учетом норм затрат ресурсов на единицу каждого типа продукции указанные ограничения можно записать в виде:

$$AX \le B^T; \quad (6)$$

$$X \ge 0, \quad (7)$$

где

$$B^T = \{b_1, b_2, ..., b_m\}, \quad (8)$$

$A=\{a_{ij}\}$, $i=\overline{1,m}$, $j=\overline{1,n}$ – матрица норм затрат ресурсов на единицу каждого типа продукции.

Выражение (4) описывает условия, которые необходимо учесть в годовой производственной программе. Строкам матрицы А соответствуют все виды ресурсов (группы машин, запасы материалов), рассматриваемые в задачах. Соответствующие строкам матрицы А компоненты вектора В указывают ограничения видов ресурсов или объёмов производства, которые установлены для годовой производственной программы предприятия. Неравенство (5) представляет собой обычные условия неотрицательности, вытекающие из смысла задачи.

Общая постановка задачи состоит в следующем. Нам нужно определить вектор $X^{(0)}$, обеспечивающий компромисс между величиной прибыли (1), валовым объемом (2) и минимальной себестоимостью (3), который удовлетворяет ограничениям минимизации производственного времени (5).

Один из возможных методов решения состоит в том , что вначале находятся три оптимальных вектора производства $x^{(i)}$,$i=\overline{1,4}$, каждый из которых соответствует одному из локальных критериев (1) – (4) . Затем определяется выпуклая линейная комбинация $X^{(0)}$, представляющая собой оптимальную (компромиссную) программу относительно указанных критериев :

$$X^{(0)} = \nu x^{(1)} + \nu_2 x^{(2)} + \nu_3 x^{(3)}, \tag{9}$$

$$\sum_{i=1}^{3} \nu_1 = 1, \nu \geq 0 . \tag{10}$$

2. Расчёт показателей производства

2.1. Максимизация прибыли.

Расчет показателей качества продукции относится к задачам линейной оптимизации. В общем виде её можно записать так:

$$\sum_{i=1}^{S} (P_i - C_i) * X_i \rightarrow \max . \tag{11}$$

Эту задачу обычно решают симплекс-методом.

Идея симплекс-метода состоит в последовательном продвижении по базисам опорных планов вплоть до получения оптимального решения или доказательства неразрешимости задачи. При этом значение целевой функции должно увеличиваться.

2.2. Определение валового объема выпускаемой продукции.

Для решения этой задачи с использованием ГА в качестве общей математической модели использовали формулу:

$$\sum_{i=1}^{S} P_i * X_i \rightarrow \max . \tag{12}$$

Для конкретных значений функция цели примет вид:

$$F_2(x) = 10x_1 + 12x_2 + 8x_3 + 16x_4 + 11x_5 \rightarrow \max . \tag{13}$$

2.3. Третьей функцией цели представим минимизацию себестоимости, которая имеет общий вид:

$$\sum_{i=1}^{S} C_i * X_i \rightarrow \min . \tag{14}$$

Запишем эту функцию с конкретными значениями:

$$F_3(X) = 3x_1 + 4x_2 + 4x_3 + 2x_4 + x_5 \rightarrow \min . \tag{15}$$

2.4. И, наконец, в роли четвертой функции цели будет выступать минимизация производственного времени:

$$\sum_{i=1}^{S} T_i * X_i \rightarrow \max , \tag{16}$$

$$F_4(X) = 2x_1 + 2x_2 + x_3 + x_4 + 3x_5 \rightarrow \max . \tag{17}$$

Чтобы достичь поставленной задачи, мы должны найти оптимальное решение для каждой функции цели. Для этого будем использовать вместо традиционных методов оптимизации, таких как математическое программирование, методы эволюционной оптимизации.

Остановимся на применении генетических алгоритмов следующих видов.

Для решения задачи ,представленной моделями (1) – (2), используем генетический алгоритм типа метода муравьиных колоний [1].

1. Основу поведения муравьев составляет самоорганизация, механизмы которой обеспечивают теоретически оптимальное поведение. Принципы его состоят в достижении системой некоторой глобальной цели в результате низкоуровневого взаимодействия ее элементов. Здесь имеется в виду использование системой только локальной информации, при этом исключается любое централизованное управление и обращение к внешнему образу системы.

Муравьиный алгоритм применяется следующим образом: в начальный момент времени, в который входит эта функция базы знаний, находится количество муравьев, равное числу кластеров, куда входит эта функция. При этом каждый муравей имеет строгую принадлежность тому кластеру, из которого он начал свое движение. Принадлежность кластеру проявляется в том, что муравей более восприимчив к феромону, оставленному муравьями из «своего» кластера:

$$F_1(X) = \sum_{j=1}^{n} C_j^{(1)} X_j \to \max, X_j \in Q = \{1,2,...,n\}, \quad (18)$$

где Q – множество видов продукции, выпускаемых предприятием.

2. При переходе из одной функции в другую муравей оставляет на связи, соединяющей эти функции, определенное количество феромона. Для того чтобы избежать схождения маршрута движения всех муравьев к одному циклу , используется испарение феромона:

$$\sum_{i=1}^{S} P_i * X_i \to \max . \quad (19)$$

Для конкретных значений функция цели примет вид:

$$F_2(X) = 10x_1 + 12x_2 + 8x_3 + 16x_4 + 113x_5 \to \max . \quad (20)$$

Муравьиный алгоритм применяется на двух этапах анализа знаний системы. Вначале он запускается на пространственной (многомерной) модели базы, после чего на основании его работы делаются первоначальные выводы. Затем модель упрощается : удаляются некоторые связи между функциями , отдельные функции объединяются в более крупные структурные единицы , структура знаний отображается на двумерное пространство. После этого алгоритм запускается на упрощенной плоской модели знаний.

3. Для решения задач, представленных моделями (3) – (4), воспользуемся генетическими алгоритмами [2]. Оптимизировать работу нефтяных трубопроводов; распределять инструменты в металлообрабатывающих цехах; осуществлять оптимизации является одной из основных областей применения ГА. Генетические алгоритмы имитируют процесс естественного отбора в природе. Для решения задачи, более оптимального с точки зрения некоторого критерия, все решения описываются набором чисел или величин нечисловой природы. Поиск оптимального решения похож на эволюцию популяции индивидов, которые представлены наборами их хромосом. В этой эволюции действуют три механизма, представленных на рисунке. Можно выделить следующие механизмы.

– отбор сильнейших наборов хромосом, которым соответствуют наиболее оптимальные решения;

– скрещивание – получение новых индивидов при помощи смешивания хромосомных наборов отобранных индивидов;

– мутации – преобразование хромосомы, случайное изменение одного или нескольких генов (чаще = одного).

В результате смены поколений вырабатывается такое решение поставленной задачи, которое уже нельзя дальше улучшать.

Для рассмотрения данной задачи используем минимизацию себестоимости:

$$\sum_{i=1}^{S} C_i * X_i \to \min . \quad (21)$$

4. Наиболее популярное приложение генетического алгоритма – оптимизация многопараметрических функций. Реальные задачи формируются , как поиск оптимального значения сложной функции , зависящей от некоторых n- выходных параметров . Сила ГА – в способности манипулировать одновременно многими параметрами. В одних случаях получено точное решение функции , в других – решением считается любое значение , лучшее некоторой заданной величины.

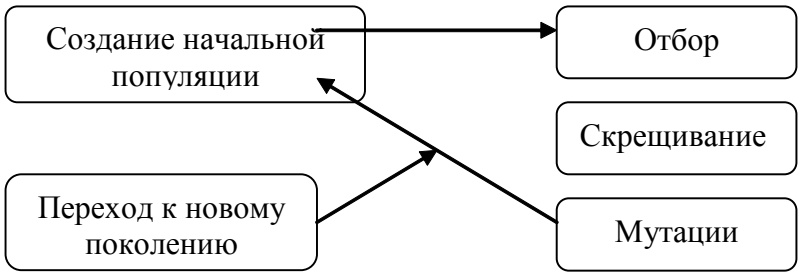


Схема генетического алгоритма

Общую схему генетического алгоритма лучше понять, когда рассматривается задача минимизации производственного времени :

$$\sum_{i=1}^{S} T_i * X_i \to \min . \qquad (22)$$

Общую схему генетического алгоритма можно записать следующим образом.

1. Выбрать начальную популяцию $S_k(0)\{S_{k1},S_{k2},\ldots S_{kn}\}$, где N – длина цепочки. Считать это $f^* = \max \{f(S_k) \mid S_k \in S_k(t)\}$,t:=0.
2. Пока не выполнен критерий остановки , делать следующее :

– выбрать родителей S_{k1},S_{k2} из популяции $S_k(t)$ (отбор);
– построить новое решение S_k по S_{k1},S_{k2} (скрещивание);
– модифицировать S_k (мутация);
– если $f^* < f(S_k)$, то $f^* := f(S_k)$;
– обновить популяцию и считать t:= t+1.

3. Поиск оптимального компромиссного решения

После решения локальных задач оптимизации следует установить «меру оценки», которая укажет отклонение значений целевых функций при выборе единичного оптимального вектора производства от оптимальных значений остальных целевых функций .

Скалярная характеристика выбирается по формуле:

$$a_{ij} = \frac{F_j(X_j) - F_j(X_j)}{F_j(X_j)}, (i, j = \overline{1,3}) , \qquad (23)$$

где $F_j(x_j)$- значение локальной функции цели j-й задачи, вычисленной в результате подстановки решения 1-й задачи. Значение a_{ij} характеризует «качество» оптимальной производственной программы X_i относительно показателя $F_j(x) \to$ extr и представляет собой модуль потерь относительно этого показателя , если выполняется программа X_i вместо X_j.

Строкам матрицы $A=\|-a_{ij}\|$ соответствуют 3 оптимальных вектора производства $x_n,(n=\overline{1,3})$, столбцам 3 – целевые функции $F_j(x) \to \text{extr},(j = \overline{1,3})$.

Матрица А может условно рассматриваться как матрица платежей матричной игры двух лиц X и F с нулевой суммой, которая определена множеством чистых стратегий $\{x_1,x_2,x_3\}$ первого игрока и множеством чистых стратегий второго игрока.

4. Выводы

При разработке проектов сложных систем, в частности автоматических систем управления АСУ, перед проектировщиком возникает проблема принятия решений при наличии одновременно нескольких показателей качества.

Поэтому разработка методов принятия решений при нескольких критериях оптимальности и в условиях неопределенности по-прежнему остается одной из главных задач исследования операций.

Исследование операций как наука располагает разнообразными средствами моделирования целенаправленной деятельности. Существующие и развиваемые подходы к анализу прикладных программ проникают в новые области автоматизированного управления . Полученные результаты не только позволяют рационально расходовать ограниченные ресурсы , но и развивают наши представления о возможностях изучаемой науки.

Научная новизна: результатом проведенного исследования является решение многокритериальной задачи с использованием генетических алгоритмов.

Практическая значимость: результатом применения предложенных методов является нахождение оптимальных по задаваемым критериям показателей деятельности предприятия.

Список литературы: 1. *Гвоздинский А.Н. , Клименко Е.Г.* Методы аналитической обработки информации // Радиоэлектроника и информатика. 2000. №4. С. 111-112. **2.** *Гвоздинский А.Н. , Клименко Е.Г.* Приминение генетических алгоритмов для решения оптимизационых задач. 7-я Международная конфиренция «Теория и техника передачи , приёма и обработки информации»; Сб. научных трудов. Харьков: ХНУРЭ. 2001. С.390-391. **3.** *Штовба С.Д.* Муравьиные алгоритмы // Exponenta Pro. Математика в приложениях. 2003.№4. С.58- 134.

Поступила в редколлегию 17.02.2011

Гвоздинский Анатолий Николаевич, канд. тех. наук, профессор кафедры искусственного интеллекта ХНУРЭ. Научные интересы: оптимизация процедур принятия решений в сложных системах управления. Адрес: Украина , 6166, Харьков, ул. Академика Ляпунова, 7,кв.9. тел. 702-38-23.

Малышкин Владимир Александрович, студент, бакалавр специальности интеллектуальные системы принятия решений, факультет КН ХНУРЭ. Адрес: Украина , 6166, Харьков, ул. Академика Ляпунова, 7, кв. 164, тел. 0956605746 e-mail: royallifeua@gmail.com.

Ткачёв Сергей Владимирович, студент, бакалавр специальности интеллектуальные системы принятия решений, факультет КН ХНУРЭ. Адрес: Украина , 61202,Харьков, ул. Целиноградская, 36, к.306, тел. 0633006304 e-mail: serejjjke-2h@yandex.ru.