ISSN 1816-0301

ИНФОРМАТИКА

1 (53)

ЯНВАРЬ-МАРТ
2017

Адрес редакции:
220012, Минск,
ул. Сурганова, 6, к. 305
тел. (017) 284-26-22
e-mail: rio@newman.bas-net.by
http://uiip.bas-net.by

ОБЪЕДИНЕННЫЙ ИНСТИТУТ ПРОБЛЕМ ИНФОРМАТИКИ
НАЦИОНАЛЬНОЙ АКАДЕМИИ НАУК БЕЛАРУСИ

ИНФОРМАТИКА

ЕЖЕКВАРТАЛЬНЫЙ НАУЧНЫЙ ЖУРНАЛ

Издается с января 2004 г.

№ 1(53) • январь-март 2017

СОДЕРЖАНИЕ

НАУЧНОЕ НАСЛЕДИЕ

Редактор Г.Б. Гончаренко
Корректор А.А. Михайлова
Компьютерная верстка О.Б. Бутевич

Сдано в набор 25.01.2017. Подписано в печать 01.03.2017.
Формат 60×84 1/8. Бумага офсетная. Гарнитура Таймс. Ризография.
Усл. печ. л. 14,4. Уч.-изд. л. 14,1. Тираж 60 экз. Заказ 1.

Государственное научное учреждение «Объединенный институт проблем информатики Национальной академии наук Беларуси».
Свидетельство о государственной регистрации издателя, изготовителя, распространителя печатных изданий № 1/274 от 04.04.2014.
ЛП № 02330/444 от 18.12.13.
Ул. Сурганова, 6, 220012, Минск.

THE UNITED INSTITUTE OF INFORMATICS PROBLEMS
OF THE NATIONAL ACADEMY OF SCIENCES OF BELARUS

INFORMATICS

PUBLISHED QUATERLY

Issued since 2004

№ 1(53) • January-March 2017

CONTENTS

SCIENTIFIC HARITAGE

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ОБРАБОТКА ИЗОБРАЖЕНИЙ

УДК 004.942

Ю.Ф. Новик

АНАЛИЗ РЕЗУЛЬТАТОВ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ *N*-СОЛИТОННОГО РЕШЕНИЯ УРАВНЕНИЯ КОРТЕВЕГА – ДЕ ФРИЗА

Приводятся результаты компьютерного моделирования N-солитонного решения уравнения Кортевега – де Фриза при N = 1, 2, 3, 4. С помощью численного эксперимента находится свойство сохранения площади под огибающей солитонных решений уравнения Кортевега – де Фриза. Обнаруживается зависимость значения площади под огибающей N-солитона от значения параметра, входящего в соответствующее решение уравнения Кортевега – де Фриза.

Введение

Уравнение Кортевега – де Фриза (КдФ) представляет собой модельное уравнение, в котором учитываются нелинейность и дисперсия. Оно используется при описании длинноволновых возмущений малой, но конечной амплитуды, магнитогидродинамических волн в холодной плазме, а также при исследовании ионно-звуковых волн. Важную роль в понимании физических процессов, описываемых уравнением КдФ, играют его точные стационарные решения [1].

В настоящей работе будем придерживаться следующего определения: *солитоном уравнения КдФ (или солитонным решением) называют стационарное решение уравнения КдФ, которое обращается в ноль на* $\pm\infty$ *вместе со своими производными по координате и имеет вид нелинейной уединенной волны, которая распространяется без изменения своей формы* [1].

В 1971 г. Р. Хирота нашел точное *N*-солитонное решение уравнения КдФ [2], где *N* – произвольное, но конечное натуральное число. Есть мнение, что *N*-солитонное решение может быть интерпретировано как взаимодействие *N* односолитонов [3], т. е. *N*-солитон – это функция, которая описывает взаимодействие *N* одиночных солитонов $(N \geq 2)$ и является решением уравнения КдФ.

1. Уравнение КдФ и его решения

Уравнение КдФ выглядит следующим образом:

$$u_t + 6uu_x + u_{xxx} = 0. \tag{1}$$

Односолитонное решение уравнения КдФ является тривиальным случаем для рассмотрения его поведения и имеет вид [4]

$$u(x,t) = \frac{P^2}{2}\operatorname{sech}^2\frac{\eta}{2}, \tag{2}$$

где $\eta = Px + \Omega t + C$ и $\Omega + P^3 = 0.$ Произвольные параметры C и P соответственно определяют положение солитона и его амплитуду. Здесь амплитудой солитона будем называть локальный максимум функции $u(x,t_0),$ где t_0 – фиксированный момент.

На рис. 1 показана зависимость амплитуды от значения параметра P : чем больше значение параметра *P*, тем больше амплитуда.

Р. Хирота с помощью своего прямого метода (метод Хироты [4]) нашел *N*-солитонное решение уравнения КдФ:

$$u = 2(\log f)_{xx}, \tag{3}$$

где f определяется ниже для каждого *N* = 2, 3, 4.

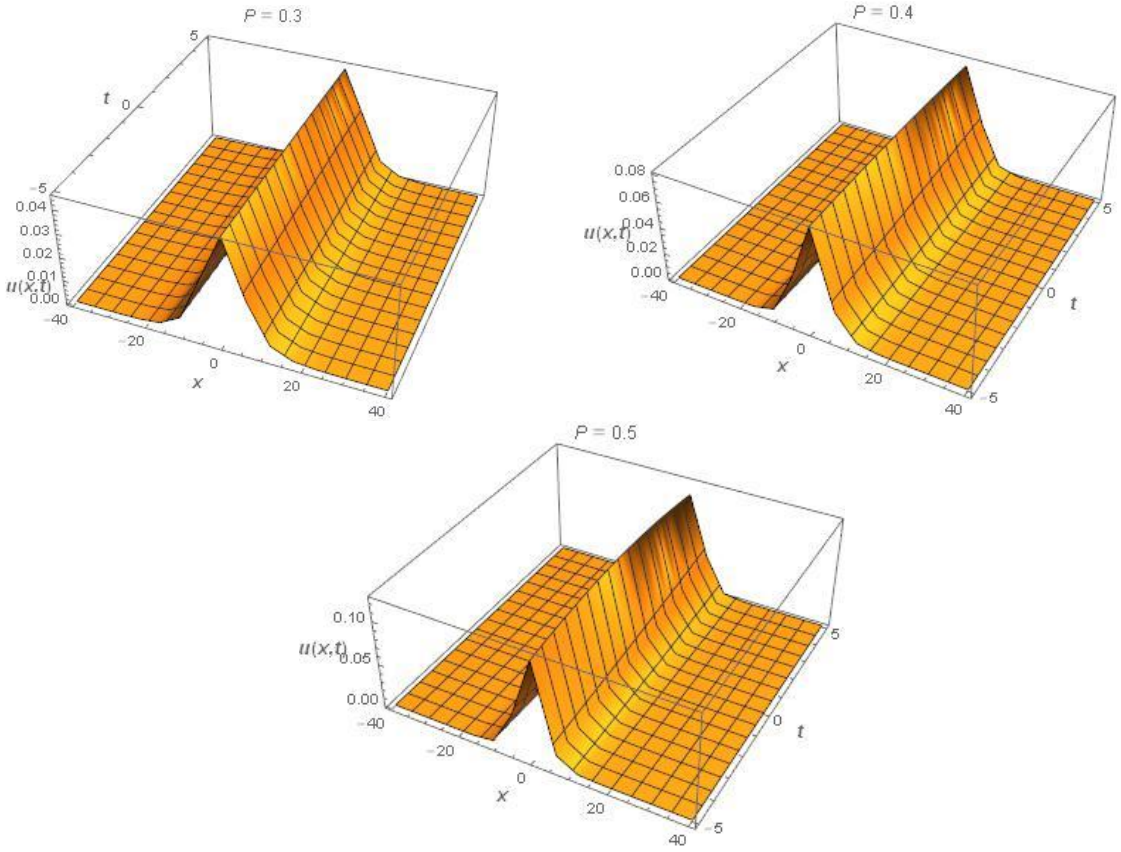


Рис. 1. Движение солитона уравнения КдФ при различных значениях P

Для двухсолитонного решения уравнения КдФ

$$f = 1 + \exp\eta_1 + \exp\eta_2 + a_{12}\exp(\eta_1 + \eta_2), \tag{4}$$

где $a_{12} = \dfrac{(P_1 - P_2)^2}{(P_1 + P_2)^2}$; $\eta_i = P_i x + \Omega_i t + C_i$, C_i – константы; P_i, Ω_i удовлетворяют нелинейному дисперсионному соотношению

$$\Omega_i + P_i^3 = 0, \quad i = 1,\ 2. \tag{5}$$

Важная величина a_{12} определяет фазовый сдвиг, т. е. изменение положения, вызванного взаимодействием двух солитонов [4].

Для трехсолитонного решения уравнения КдФ

$$\begin{aligned} f = 1 + \exp\eta_1 + \exp\eta_2 + \exp\eta_3 + a_{12}\exp(\eta_1 + \eta_2) + a_{13}\exp(\eta_1 + \eta_3) + \\ + a_{23}\exp(\eta_2 + \eta_3) + a_{12}a_{13}a_{23}\exp(\eta_1 + \eta_2 + \eta_3), \end{aligned} \tag{6}$$

где $\eta_i = P_i x + \Omega_i t + C_i$; $\Omega_i + P_i^3 = 0$; $a_{ij} = \dfrac{(P_i - P_j)^2}{(P_i + P_j)^2}$, $i, j = 1,\ 2,\ 3$.

Для четырехсолитонного решения уравнения КдФ

$$\begin{aligned} f = 1 + \exp\eta_1 + \exp\eta_2 + \exp\eta_3 + \exp\eta_4 + a_{12}\exp(\eta_1 + \eta_2) + a_{13}\exp(\eta_1 + \eta_3) + \\ + a_{23}\exp(\eta_2 + \eta_3) + a_{14}\exp(\eta_1 + \eta_4) + a_{24}\exp(\eta_2 + \eta_4) + a_{34}\exp(\eta_3 + \eta_4) + \\ + a_{12}a_{13}a_{23}\exp(\eta_1 + \eta_2 + \eta_3) + a_{23}a_{24}a_{34}\exp(\eta_2 + \eta_3 + \eta_4) + a_{13}a_{14}a_{34}\exp(\eta_1 + \eta_3 + \eta_4) + \\ + a_{12}a_{14}a_{24}\exp(\eta_1 + \eta_2 + \eta_4) + a_{12}a_{13}a_{14}a_{23}a_{24}a_{34}\exp(\eta_1 + \eta_2 + \eta_3 + \eta_4), \end{aligned} \tag{7}$$

где $\eta_i = P_i x + \Omega_i t + C_i$, $\Omega_i + P_i^3 = 0$; $a_{ij} = \dfrac{(P_i - P_j)^2}{(P_i + P_j)^2}$, $i, j = 1,\ 2,\ 3,\ 4.$

2. Моделирование двухсолитонного решения уравнения КдФ

Моделирование эволюции двухсолитонного решения (3) уравнения КдФ, где f имеет вид (4), в трехмерном случае с учетом эволюции на заданном промежутке времени при $P_1 = 1,\ P_2 = 2,\ C_1 = 0,\ C_2 = 0{,}5$ показано на рис. 2.

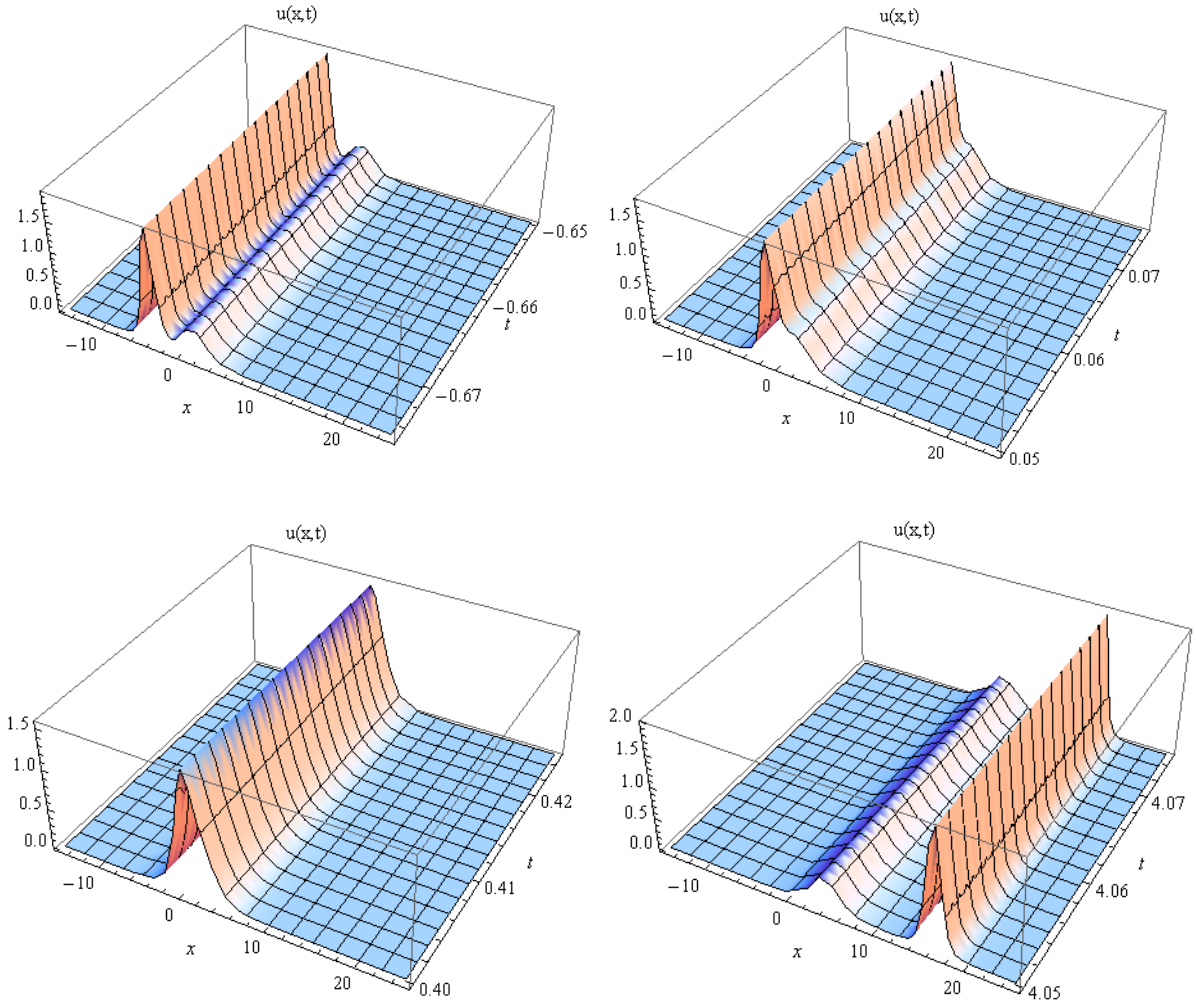


Рис. 2. Компьютерное моделирование эволюции двухсолитонного решения уравнения КдФ

Будем говорить, что солитоны достаточно отдалились друг от друга, если $u(x_0, t_0) < \varepsilon$ при $\varepsilon = 10^{-15}$, где t_0 – фиксированный момент; $x_0 = x_0^{(1)} + \dfrac{x_0^{(2)} - x_0^{(1)}}{2}$, $x_0^{(1)}, x_0^{(2)}$ – координаты, при которых $A_i = U(x_0^{(i)}, t_0)$, A_i – амплитуды меньшего и большего одиночных солитонов соответственно, $i = 1,\ 2$. В данном случае амплитуда меньшего солитона равняется 0,5, а амплитуда большего – 2.

Посчитаем площадь под огибающей каждого из двух одиночных солитонов, а также площадь под огибающей на промежутке взаимодействия данных солитонов.

Из численного эксперимента видно, что временной промежуток взаимодействия одиночных солитонов [–1,5; 2,5]. Используя результаты моделирования движения солитонов, находим, что при $t = 35$ одиночные солитоны достаточно удалились друг от друга, т. е. $u(87,9245; 35) = 0 < \varepsilon$ (рис. 3).

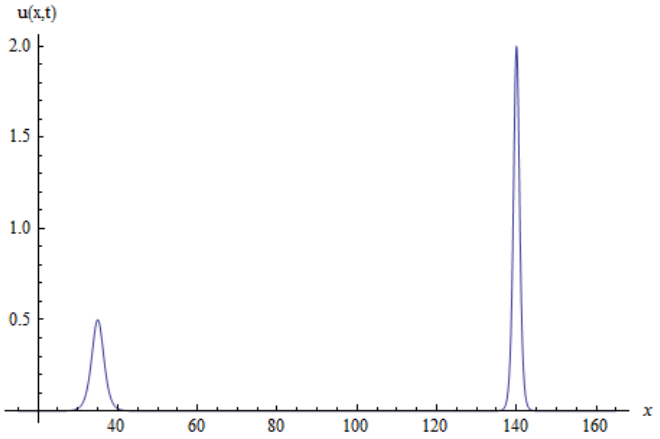


Рис. 3. Положение двухсолитонного решения уравнения КдФ при t = 35

Площадь солитонов будем считать по формуле

$$S = \int_{x_1}^{x_2} u(x,\ t_0)\, dx, \tag{8}$$

где t_0 – фиксированный момент времени; x_1, x_2 – решения уравнения $u(x, t_0) = \varepsilon$.

Находим площадь солитона с меньшей амплитудой. Корни уравнения $u(x, 35) = \varepsilon$ в окрестности точки $x_0^{(1)} = 35$ будут $x_{11} = -0,2319$, $x_{12} = 70,2539$. Подставляя их в (8), получаем $S_1 = 2$.

Оцениваем площадь солитона с большей амплитудой. Корни уравнения $u(x, 35) = \varepsilon$ в окрестности точки $x_0^{(2)} = 140,849$ будут $x_{21} = 122$, $x_{22} = 157,767$. Подставляя их в (8), получаем $S_2 = 4$.

Посчитаем площадь под огибающей во время взаимодействия солитонов по схеме, приведенной выше. Результаты вычисления представлены в табл. 1.

Таблица 1

Площадь S под огибающей двухсолитонного решения уравнения КдФ при взаимодействии в момент t_0

t_0	–1,5	–1,25	–1	–0,75	–0,5	–0,25	0	0,25	0,5	0,75	1	1,25	1,5	1,75	2	2,25	2,5
S	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6

Согласно результатам вычислений площадь под огибающей на промежутке взаимодействия солитонов равна сумме площадей одиночных солитонов $S = S_1 + S_2$, т. е. площадь под огибающей двухсолитонного решения уравнения КдФ сохраняется на всем временном промежутке для рассмотренного примера.

3. Моделирование трехсолитонного решения уравнения КдФ

Моделирование движения трехсолитонного решения (3) уравнения КдФ с учетом (6) в трехмерном случае при $P_1 = 1$, $P_2 = 1,5$, $P_3 = 0,75$ показано на рис. 4.

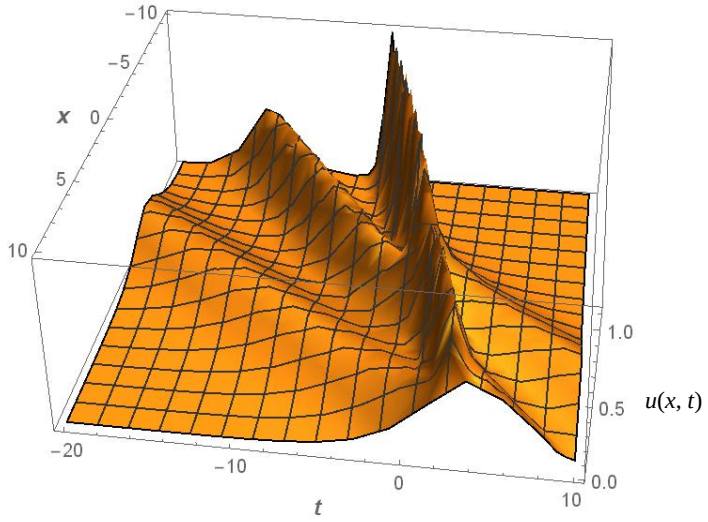


Рис. 4. Компьютерное моделирование эволюции трехсолитонного решения уравнения КдФ

Аналогично случаю при $N = 2$ вычисляем площади под огибающей трехсолитонного решения уравнения КдФ при их взаимодействии и в том случае, когда три одиночных солитона отдалились друг от друга. Промежуток взаимодействия солитонов в данном случае будет [–3; 6] (рис. 5). Результаты вычисления площади под огибающей на этом промежутке приведены в табл. 2.

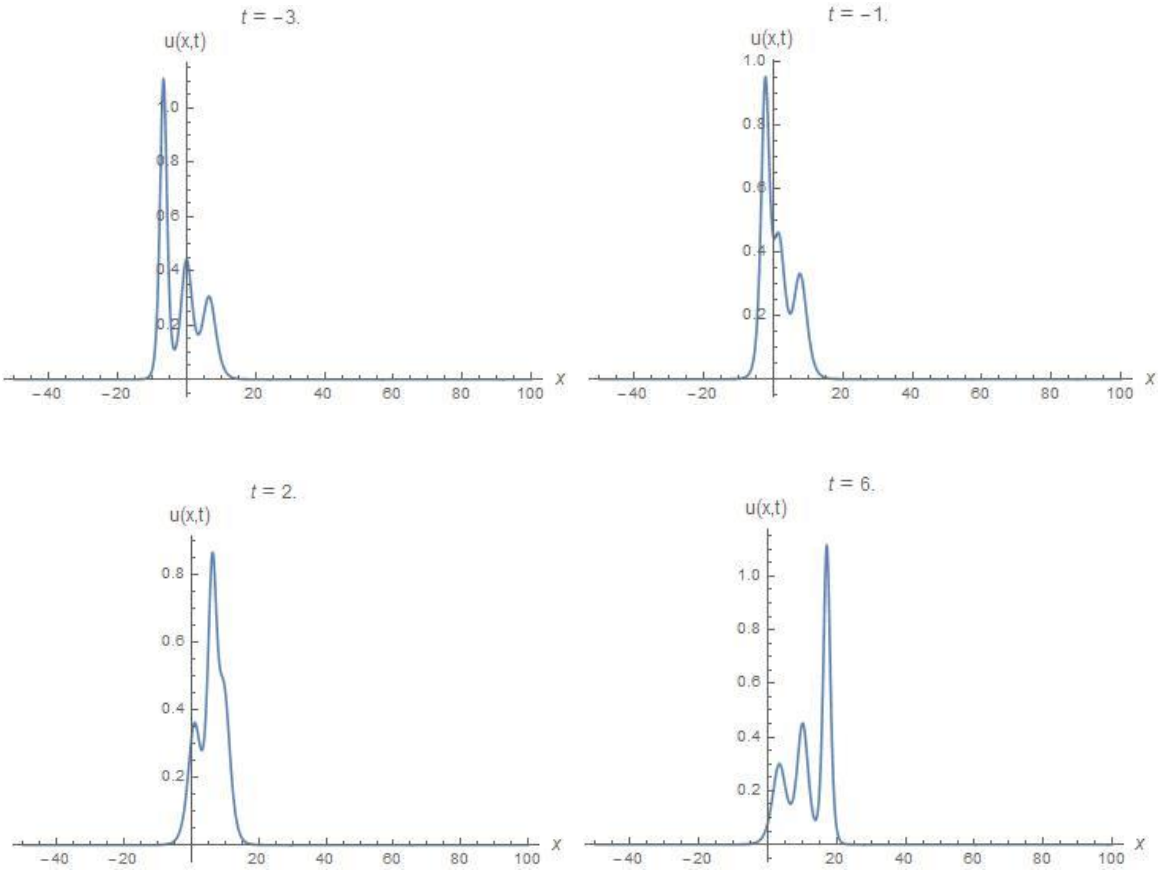


Рис. 5. Компьютерное моделирование эволюции трехсолитонного
решения уравнения КдФ при t = [–3; 6]

Таблица 2

Площадь S под огибающей трехсолитонного решения уравнения КдФ при взаимодействии в момент t_0

t_0	–3	–2,5	–2	–1,5	–1	–0,5	0	0,5	1	1,5	2	2,5	3	3,5	4	4,5	5	5,5	6
S	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5	6,5

Из численного эксперимента находим, что три солитона достаточно отдалились друг от друга в момент $t_0 = 230$, т. е. $u(182;\ 230) = 0 < \varepsilon$ и $u(377{,}5;\ 230) = 0 < \varepsilon$. Вычисляем площади трех одиночных солитонов аналогично случаю $N = 2$ и получаем $S_1 = 2,\ S_2 = 3,\ S_3 = 1{,}5$. В итоге имеем $S = S_1 + S_2 + S_3$, т. е. площадь под огибающей трехсолитонного решения уравнения КдФ также сохраняется для рассмотренного примера.

4. Моделирование четырехсолитонного решения уравнения КдФ

Согласно формулам (3), (7) было промоделировано четырехсолитонное решение уравнения КдФ (рис. 6).

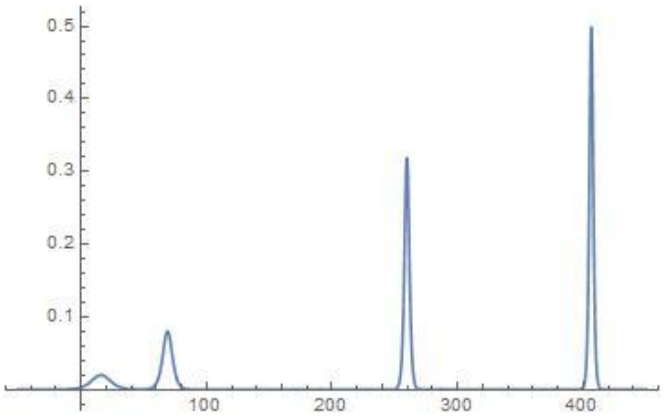


Рис. 6. Положение четырехсолитонного решения уравнения КдФ при $P_1 = 0{,}2,\ P_2 = 0{,}4,\ P_3 = 0{,}8,\ P_4 = 1$ в момент t = 400

Аналогично предыдущим случаям были найдены площади под огибающей удалившихся солитонов и площадь под огибающей на промежутке взаимодействия четырехсолитонного решения уравнения КдФ при различных значениях $P_i,\ i = \overline{1,4}$:

1. $P_1 = 0{,}2;\ P_2 = 0{,}4;\ P_3 = 0{,}8;\ P_4 = 1$ – момент, когда солитоны успели достаточно отдалиться друг от друга, $t_0 = 3000$, время взаимодействия $t = [-25;30]$. Площади под огибающей отдалившихся солитонов $S_1 = 0{,}4,\ S_2 = 0{,}8,\ S_3 = 1{,}6,\ S_4 = 2$. Площадь под огибающей во время взаимодействия $S = 4{,}8$. В результате получаем, что площадь под огибающей четырехсолитонного решения уравнения КдФ также сохраняется ($S = S_1 + S_2 + S_3 + S_4$).

2. $P_1 = 0{,}1;\ P_2 = 0{,}2;\ P_3 = 0{,}5;\ P_4 = 0{,}9$ – момент, когда солитоны успели достаточно отдалиться друг от друга, $t_0 = 20\ 000$, время взаимодействия $t = [-34;50]$. Площади под огибающей отдалившихся солитонов $S_1 = 0{,}2,\ S_2 = 0{,}4,\ S_3 = 1,\ S_4 = 1{,}8$. Площадь под огибающей во время взаимодействия $S = 3{,}4$. Заметим, что $S = S_1 + S_2 + S_3 + S_4$.

3. $P_1 = 1,\ P_2 = 1{,}2,\ P_3 = 1{,}5,\ P_4 = 2$ – момент, когда солитоны успели достаточно отдалиться друг от друга при данных $P_i,\ i = \overline{1,4}$, $t_0 = 305$, время взаимодействия $t = [-25;30]$. Площади под огибающей отдалившихся солитонов $S_1 = 2,\ S_2 = 2{,}4,\ S_3 = 3,\ S_4 = 4$. Площадь под огибающей во время взаимодействия $S = 11{,}4$ ($S = S_1 + S_2 + S_3 + S_4$).

5. Анализ результатов моделирования

С помощью численных экспериментов было показано, что площадь под огибающей *N*-солитонного решения уравнения КдФ (*N* = 2, 3, 4) во время взаимодействия и в момент отдаления сохраняется.

На рис. 1 видно, что чем больше значение параметра *P*, тем больше амплитуда солитона. Отсюда легко определить, какое $P_i,\ i=\overline{1,N},$ соответствует солитону в момент отдаления.

Обратим внимание на связь значений площади отдалившихся солитонов и значений параметров $P_i,\ i=\overline{1,N}$. При $N=3$ задавались значения параметров $P_1=1,\ P_2=1{,}5,\ P_3=0{,}75$ и были получены соответствующие площади отдалившихся солитонов $S_1=2,\ S_2=3,\ S_3=1{,}5.$ Заметим, что $S_1=2P_1,\ \ S_2=2P_2,\ \ S_3=2P_3.$ При $N=4$ были рассмотрены три различных варианта значений $P_i,\ i=\overline{1,4},$ и снова получилось, что $S_i=2P_i, i=\overline{1,4}.$

На основании вышесказанного можно выдвинуть гипотезу, что площадь под огибающей *N*-солитонного решения уравнения КдФ равна удвоенному значению параметра *P*, который входит в дисперсионное отношение $\Omega_i+P_i^3=0,\quad i=\overline{1,N}.$

Заключение

В настоящей работе показаны результаты моделирования *N*-солитонного решения уравнения КдФ при *N* = 1, 2, 3, 4. При анализе результатов моделирования было обнаружено свойство *N*-солитона уравнения КдФ сохранять площадь под огибающей на всем временном интервале. Свойство сохранения площади под огибающей является определяющей характеристикой (инвариантом) класса *N*-солитонных решений уравнения КдФ. Кроме того, обнаружена зависимость величины площади под огибающей солитона от значения параметра, входящего в данное решение: площадь под огибающей равна удвоенному значению параметра *P*.

Список литературы

1. Плазменная гелиогеофизика : в 2 т. Т. 2 / под ред. Л.М. Зеленого, И.С. Веселовского. – М. : Физматлит, 2008. – 560 с.

2. Hirota, R. Exact solution of the Korteweg – de Vries equation for multiple collisions of solitons / R. Hirota // Phys. Rev. Lett. – 1971. – № 27. – P. 1192–1194.

3. Солитоны и нелинейные волновые уравнения / Р. Додд [и др.] ; пер. с англ. – М. : Мир, 1988. – 694 с.

4. Hirota, R. The Direct Method in Soliton Theory / R. Hirota ; ed. B. Bollobas [et al.]. – N. Y. : Cambridge University Press, 2004. – 200 p.

Поступила 11.11.2016

Объединенный институт проблем информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: novik.yu.f@gmail.com

Y.F. Novik

ANALYSIS OF THE RESULTS OF COMPUTER SIMULATION *N*-SOLITON SOLUTIONS OF THE KORTEWEG – DE VRIES EQUATION

The results of computer simulation *N*-soliton solutions of the Korteweg – de Vries equation with *N* = 1, 2, 3, 4 are shown. Using numerical experiment the property of conservation of area under the envelope of soliton solutions of the Korteweg – de Vries equation is found. In addition, the dependence value of the area under the envelope of *N*-soliton on the value of parameter, included into a corresponding solution of the Korteweg – de Vries equation, is detected.

УДК 519.86

С.И. Доценко

ЗАДАЧА РАСПРЕДЕЛЕНИЯ РАСХОДОВ ПРИ РАЗВОЗКЕ ПО КОЛЬЦЕВОМУ МАРШРУТУ КАК КООПЕРАТИВНАЯ ИГРА

Рассматривается практическая задача распределения выгоды между участниками кооперативной перевозки товаров транспортным средством. Для моделирования используется аппарат теории кооперативных игр. Обсуждаются некоторые понятия кооперативной теории игр. Рассматриваются такие подходы распределения вектора стоимости и вектора получаемой в результате кооперации выгоды, как вектор Шепли, n-ядро кооперативной игры и нормированное n-ядро. Приводятся алгоритмы построения, и для рассматриваемой задачи находятся все указанные векторы распределения стоимости.

Введение

В работе рассматривается логистическая задача справедливого распределения расходов при доставке грузов нескольким потребителям на примере задачи развозки по кольцевому маршруту из Минска в пять областных центров Республики Беларусь.

На интуитивном уровне понятно, что если суммарный объем заказов не превышает грузоподъемность транспортного средства, то группе потребителей выгоднее вскладчину оплатить некоторый кольцевой маршрут, чем каждому в отдельности платить за радиальный. При получении суммарной выгоды от уменьшения длины маршрута возникает естественный вопрос, как эту выгоду справедливо разделить между потребителями. Оказывается, что математическим аппаратом такой задачи распределения расходов может служить кооперативная игра, «надстроенная» над задачей коммивояжера. К сожалению, приведенная методика применима лишь для задач с небольшим количеством потребителей, поскольку при решении задачи возникает «проклятие размерности» сразу по двум причинам. Во-первых, как известно, сама задача коммивояжера является *NP*-трудной. Во-вторых, кооперативная игра с n игроками предполагает задание характеристической функции от $2^n - 1$ аргументов, а в ходе решения вспомогательных задач каждому заданому значению соответствует ограничение в задаче линейного программирования.

Задача коммивояжера является одной из ключевых задач комбинаторной оптимизации. Она формулируется следующим образом: в графе найти замкнутый маршрут минимального веса (где под весом маршрута понимается сумма весов входящих в него ребер), в который каждая вершина входит в точности один раз.

Относительно постановки задачи коммивояжера следует сделать некоторые замечания. В произвольном графе (например, в любом дереве) может не существовать замкнутого маршрута, содержащего каждую из вершин в точности один раз. Даже если такой маршрут существует, то может существовать маршрут меньшего веса, проходящий через все вершины по крайней мере один раз, а через некоторые вершины – более одного раза. Однако в полном графе при условии соблюдения неравенства треугольника для любой тройки вершин существует замкнутый маршрут с посещением каждой вершины ровно один раз, для которого не существует другого замкнутого маршрута с посещением каждой вершины по крайней мере один раз и имеющего меньший вес.

1. Основные сведения о кооперативных играх

Кооперативная игра задается парой $\langle N, V \rangle$, где N – конечное множество игроков; n – их количество; V – отображение $2^N \to R$ из множества всех коалиций в множество действительных чисел, называемое характеристической функцией (которая ставит в соответствие каждой коалиции совместный заработок ее членов), причем $V(\varnothing) = 0$ (это, по сути, означает, что пус-

тая коалиция никогда ничего не зарабатывает). Множество всех кооперативных игр на множестве игроков N, в котором различные игры отличаются различными характеристическими функциями, обозначается через G^N. Множество всех игроков N принято называть гранд-коалицией.

Кооперативная игра называется супераддитивной, если $\left(\forall S,T \in 2^N,\ S \cap T = \varnothing\right)$ $\left(V(S)+V(T) \le V(S \cup T)\right)$, и выпуклой, если $\left(\forall S,T \in 2^N\right)\left(V(S)+V(T) \le V(S \cup T)+V(S \cap T)\right)$.

Решением кооперативной игры $\vec{x} = \left(x_1,...,x_n\right)$ называется отображение $f : G^N \to R^n$, ставящее в соответствие каждой кооперативной игре *n*-мерный вектор, *i*-я компонента которого равна платежу *i*-му игроку в данной игре. Решение игры называется эффективным, если $X\left(N\right) = V\left(N\right)$, где $X(S) = \sum_{i \in S} x_i$. Эффективность решения означает, что заработок гранд-коалиции распределяется между ее членами без потерь. Решение игры называется стабильным, если для любой коалиции $S \subset N$ выполнено условие $\sum_{i \in S} x_i \ge x(S)$, что, по сути, является условием отсутствия стимула к сепаратизму. Другими словами, никакой из коалиций *S* невыгодно выйти из гранд-коалиции, с тем чтобы получить совместный заработок и распределить его между членами *S* так, чтобы каждый из игроков получил больше, чем его доля x_i в гранд-коалиции.

Ядром игры называется множество эффективных и стабильных решений $\vec{x}$: $C(V) := \left\{\vec{x} \in R^n \mid X(N) = V(N),\ X(S) \ge V(S),\ \forall S \in 2^N\right\}$.

Рассмотрим некоторую перестановку игроков $\pi = \left(i_1,...,i_n\right)$. Пусть $\pi(i)$ – номер позиции *i*-го игрока в перестановке π; $\pi^i = \left\{ j \in N \mid \pi(j) \le \pi(i)\right\}$ – множество игроков, включающее *i* и всех, кто стоит перед ним в перестановке π. Назовем маргинальным вкладом игрока *i* в перестановку π величину $m_i^\pi = V\left(\pi^i\right) - V\left(\pi^i \setminus \{i\}\right)$. Очевидно, что для любой перестановки сумма маргинальных вкладов всех игроков равна *V*(*N*).

Вектором Шепли (ВШ) называется решение кооперативной игры, представляющее собой вектор маргинальных вкладов игроков, усредненных по всем возможным *n*! перестановкам. Компоненты ВШ также могут быть вычислены по формуле

$$\varnothing_i(V) = \sum_{S \subset N \setminus \{i\}} \frac{|S|!\left(n-1-|S|\right)!}{n!}\left(V\left(S \cup \{i\}\right) - V\left(S\right)\right). \qquad (1)$$

Оказывается, что для выпуклой игры ядро всегда непусто, а ВШ является «центром масс» ядра и, следовательно, принадлежит ему. Для невыпуклой игры ядро может оказаться пустым, а ВШ может не принадлежать ядру, даже если ядро непусто. В этом случае более приемлемыми решениями оказываются *n*-ядро (nucleolus) и nucleolus per capita (в литературе русского перевода не имеет и буквально означает «*n*-ядро на душу населения»).

Понятие *n*-ядра было впервые введено в [1]. Это точечное решение кооперативной игры, которое базируется на понятиях эксцесса и лексикографического порядка.

Определение 1. Эксцесс коалиции – это значение

$$e(x,S) = V(S) - \sum_{i \in S} x_i,\ \vec{x} \in D(V),\ S \in 2^N, \qquad (2)$$

где $D(V)$ – множество решений игры $\vec{x} = \left(x_1,...,x_n\right)$, удовлетворяющих условиям эффективности и индивидуальной рациональности, т. е. $\sum_{i \in N} x_i = V(N)$ и $x_i \ge V(i),\ i = 1,n,$ соответственно.

Другими словами, эксцесс является мерой сожаления о том, что суммарный заработок коалиции не такой большой, как хотелось бы. Если суммарный заработок членов коалиции *S* больше, чем *V*(*S*), то эксцесс будет отрицательным.

Определение 2. Вектор $\vec{x}=(x_1,...,x_n)$ лексикографически меньше, чем $\vec{y}=(y_1,...,y_n)$, если существует некоторое $k\in\{1,...,n\}$, такое, что $x_k<y_k$ и $x_i=y_i$ для всех $i<k$.

Определение 3. *n*-ядро кооперативной игры – это эффективное решение $\vec{x}=(x_1,...,x_n)$, для которого достигается лексикографический минимум на множестве векторов размерности 2^N-1 с упорядоченными в убывающем порядке компонентами, которые равны значениям эксцессов всех непустых коалиций $S\in 2^N\setminus\text{ø}$. Оказывается, что для любой кооперативной игры существует *n*-ядро. Кроме того, если ядро игры непусто, то *n*-ядро всегда принадлежит ядру.

Понятие нормированного ядра (normalized nucleolus) было введено в [2]. Оно аналогично понятию *n*-ядра с той разницей, что понятие «эксцесс» заменяется на «эксцесс на душу населения» (epc), при этом вычисленное значение эксцесса коалиции делится на количество членов коалиции. Найденный таким образом лексикографический минимум для нормализованного ядра также носит название «nucleolus per capita» [3, с. 129]:

$$\text{epc}(x,S)=\frac{V(S)-\sum_{i\in S}x_i}{|S|},\ \vec{x}\in D(V),\ S\in 2^N. \qquad (3)$$

В работе [4] было показано, что для задачи коммивояжера при условиях соблюдения неравенств треугольника и равенства расстояний в противоположных направлениях, т. е. $(\forall i,j)\left(d_{i,j}=d_{j,i}\right)$, ядро всегда непусто для трех и четырех игроков соответственно.

2. Применение аппарата кооперативных игр в задаче развозки

В качестве примера рассмотрим задачу развозки с отправной точкой в Минске с посещением пяти центров областей Республики Беларусь, а именно Витебска, Могилева, Гомеля, Бреста и Гродно. В качестве исходных данных задачи возьмем расстояния между городами по автомагистралям, вычисленные с помощью онлайн-калькулятора расстояний на сайте avtodispetcher.ru. Расстояния между городами приведены в табл. 1.

Таблица 1

Город	**Минск**	Витебск	Могилев	Гомель	Брест	Гродно
Минск	–	288	198	298	353	278
Витебск	288	–	163	336	641	568
Могилев	198	163	–	180	538	480
Гомель	298	336	336	–	536	590
Брест	353	641	641	536	–	232
Гродно	278	568	568	590	232	–

Заметим, что в табл. 1 для некоторых троек городов нарушается неравенство треугольника, хотя и незначительно. Например, *d*(Гродно – Гомель) = 590 км, что больше, чем (*d*(Гродно – Минск) = 278 км)+(*d*(Минск–Гомель) = 298 км) = 576 км. Это объясняется тем, что расстояние между городами ищется по сети дорог между центрами (нулевыми отметками), а если дорога из одного города в другой проходит через третий (например, из Гомеля в Гродно – через Минск), то маршрут строится навигатором не через центр промежуточного пункта, а в объезд.

Другими тройками городов, для которых нарушается неравенство треугольника, являются Витебск – Могилев – Гомель и Витебск – Минск – Гродно.

Чтобы избежать противоречий в дальнейших расчетах, уменьшим большее расстояние до величины суммы двух меньших так, чтобы неравенство треугольника выполнялось как равенство. Новые данные представим в табл. 2, где измененные расстояния выделены жирным шрифтом.

Таблица 2

Город	**Минск**	Витебск	Могилев	Гомель	Брест	Гродно
Минск	–	288	198	298	353	278
Витебск	288	–	163	336	641	**566**
Могилев	198	163	–	180	538	**476**
Гомель	298	336	336	–	536	**576**
Брест	353	641	641	536	–	232
Гродно	278	**566**	**476**	**576**	232	–

В роли игроков (участников кооперативной игры) выступают города, в которые осуществляется доставка, но не отправной пункт (Минск). В качестве исходной характеристической функции (ХФ), описывающей транспортные расходы коалиции городов, возьмем оптимальное решение задачи коммивояжера для множества, включающего отправной пункт и все города данной коалиции.

Так, для одноэлементных коалиций ХФ равна удвоенному расстоянию от Минска до данного города, а для двухэлементных коалиций – сумме расстояний от отправного пункта до этих городов плюс расстояние между ними. Для нахождения ХФ коалиций, состоящих из трех-пяти городов, воспользуемся сервисом решения задачи коммивояжера, представленном на сайте math.semestr.ru и основанном на методе ветвей и границ.

Исходные значения характеристической функции: *V*(Вит)=576, *V*(Мог)=394, *V*(Гом)=596, *V*(Бр)=706, *V*(Гр)=556, *V*(Вит, Мог)=649, *V*(Вит, Гом)=992, *V*(Вит, Бр)=1282, *V*(Вит, Гр)=1132, *V*(Мог, Гом)=676, *V*(Мог, Бр)=1089, *V*(Мог, Гр)=952, *V*(Гом, Бр)=1187, *V*(Гом, Гр)=1152, *V*(Бр, Гр)=863, *V*(Вит, Мог, Гом)=929, *V*(Вит, Мог, Бр)=1342, *V*(Вит, Мог, Гро)=1205, *V*(Вит, Гом, Бр)=1513, *V*(Вит, Гом, Гро)=1478, *V*(Вит, Бр, Гро)=1439, *V*(Мог, Гом, Бр)=1267, *V*(Мог, Гом, Гро)=1232, *V*(Мог, Бр, Гро)=1246, *V*(Гом, Бр, Гр)=1344, *V*(Вит, Мог, Гом, Бр)= =1520, *V*(Вит, Мог, Гом, Гр)=1485, *V*(Вит, Мог, Бр, Гр)=1499, *V*(Вит, Гом, Бр, Гр)=1670, *V*(Мог, Гом, Бр, Гр)=1424, *V*(Вит, Мог, Гом, Бр, Гр)=1677.

По данной ХФ затрат построим так называемую ХФ экономий, вычисляемую по формуле

$$W(S) = \sum_{i \in S} V(i) - V(S), \tag{4}$$

и будем вести дальнейшие рассуждения уже относительно нее. Значение ХФ *W*(*S*) коалиции равно величине экономии суммарной длины маршрута, включающего пункт отправления и все города коалиции по сравнению со случаем, когда каждый из городов коалиции использует радиальный маршрут, например *W*(Мог, Бр, Гр)=*V*(Мог)+*V*(Бр)+*V*(Гр)–*V*(Мог, Бр, Гр)=412.

Заметим, что согласно формуле (4) значение *W* от одноэлементных коалиций равно нулю, остальные значения вычислим непосредственно: *W*(Вит, Мог)=323, *W*(Вит, Гом)=180, *W*(Вит, Бр)=0, *W*(Вит, Гр)=0, *W*(Мог, Гом)=316, *W*(Мог, Бр)=13, *W*(Мог, Гр)=0, *W*(Гом, Бр)=115, *W*(Гом, Гр)=0, *W*(Бр, Гр)=399, *W*(Вит, Мог, Гом)=639, *W*(Вит, Мог, Бр)=336, *W*(Вит, Мог, Гр)= =323, *W*(Вит, Гом, Бр)=365, *W*(Вит, Гом, Гро)=250, *W*(Вит, Бр, Гр)=399, *W*(Мог, Гом, Бр)=437, *W*(Мог, Гом, Гро)=316, *W*(Мог, Бр, Гр)=412, *W*(Гом, Бр, Гр)=514, *W*(Вит, Мог, Гом, Бр)=754, *W*(Вит, Мог, Гом, Гр)=639, *W*(Вит, Мог, Бр, Гр)=735, *W*(Вит, Гом, Бр, Гр)=764, *W*(Мог, Гом, Бр, Гр)=830, *W*(Вит, Мог, Гом, Бр, Гр)=1153.

Для данной ХФ найдем ВШ. Формулу (1) можно представить в развернутом виде:

$$\varnothing_i = \frac{1}{n!}\left(\sum_{k=1}^{n} \sum_{\substack{|S|=k, \\ i \in S}} (k-1)!(n-k)!W(S) - \sum_{k=1}^{n-1} \sum_{\substack{|S|=k, \\ i \notin S}} k!(n-k-1)!W(S) \right). \tag{5}$$

Для n=5 с учетом того, что $W(S)$=0 при $|S|$=1 (т. е. для одноэлементных коалиций значения ХФ равны нулю), формула (5) приобретает вид

$$\varnothing_i = \frac{1}{120}\left(\left(6\sum_{\substack{|S|=2,\\ i\in S}} +4\sum_{\substack{|S|=3,\\ i\in S}} +6\sum_{\substack{|S|=4,\\ i\in S}} -4\sum_{\substack{|S|=2,\\ i\notin S}} -6\sum_{\substack{|S|=3,\\ i\notin S}}\right)(W(S)) + 24(W(N) - W(N\setminus i))\right). \qquad (6)$$

Компоненты ВШ, вычисленные по формуле (6) и округленные до целых значений, имеют значения $\varnothing = (199,\ 241,\ 249,\ 262,\ 202)$.

Непосредственная проверка по всем коалициям выполнения условия устойчивости $\sum_{i\in S} x_i \ge W(S)$ для ВШ показывает, что ядро данной игры непусто и ВШ принадлежит ядру.

Задачу нахождения *n*-ядра запишем как задачу нахождения лексикографического максимума величин переплат по всему набору коалиций S, $S \subseteq N, S \ne N$, на множестве всех эффективных распределений, т. е. распределений, удовлетворяющих условию $x_i \ge 0, \sum_{i\in N} x_i = W(N)$. На практике эта задача решается как последовательность вспомогательных задач линейного программирования (ЗЛП), принцип построения которой описан ниже. Для каждой коалиции S, $S \subset N$, $S \ne N$, $S \ne \varphi$, определяем величину переплаты, равную $\sum_{i\in S} x_i - W(S)$ (переплата равна величине эксцесса, взятой с обратным знаком).

Первая задача последовательности имеет вид

$$t \to \max,$$
$$\left(\sum_{i\in S} x_i - W(S) \ge t\right)(\forall S \subset N,\ S \ne N,\ S \ne \varphi),$$
$$\sum_{i\in N} x_i = W(N),\ x_i \ge 0.$$

В первой ЗЛП находится решение $\vec{x} = (x_1, ..., x_n)$. При этом решении величина переплаты *t*, которую гарантированно получат все коалиции (или, другими словами, минимальная из переплат), достигает максимума. В полученном решении оказывается, что найденное максимальное значение минимальной переплаты достигается одновременно для нескольких коалиций, при этом некоторые подмножества множества коалиций, для которых достигается найденная минимальная переплата, обязательно образуют так называемую дополняющую группу или, другими словами, являются взаимодополняющими. Значит, суммарная выплата по дополняющей группе коалиций оказывается постоянной и любое увеличение выплаты по одной из коалиций группы неизбежно ведет к уменьшению выплаты в какой-либо другой коалиции этой группы, что, в свою очередь, уменьшает (в лексикографическом смысле) вектор переплат по всем коалициям, упорядоченный в порядке возрастания. Поэтому найденные выплаты по всем коалициям дополняющей группы фиксируются, а в следующую вспомогательную ЗЛП вводятся соответствующие дополнительные ограничения.

Следующая ЗЛП находит решение, максимизирующее минимальную из переплат по всем коалициям, выплата по которым не была фиксирована ранее, и т. д.

Пронумеруем города в порядке, приведенном в табл. 1, т. е. 1 – Витебск, 2 – Могилев, 3 – Гомель, 4 – Брест, 5 – Гродно. Для данной кооперативной игры первая ЗЛП выглядит следующим образом:

$$t \to \max$$
$$x_1 - t \ge 0,\ x_2 - t \ge 0,\ x_3 - t \ge 0,\ x_4 - t \ge 0,\ x_5 - t \ge 0,\ t \ge 0,\ x_1 + x_2 - t \ge 323,$$
$$x_1 + x_3 - t \ge 180,\ x_1 + x_4 - t \ge 0,\ x_1 + x_5 - t \ge 0,\ x_2 + x_3 - t \ge 316,\ x_2 + x_4 - t \ge 13,$$
$$x_2 + x_5 - t \ge 0,\ x_3 + x_4 - t \ge 115,\ x_3 + x_5 - t \ge 0,\ x_4 + x_5 - t \ge 399,\ x_1 + x_2 + x_3 - t \ge 639,$$
$$x_1 + x_2 + x_4 - t \ge 336,\ x_1 + x_2 + x_5 - t \ge 323,\ x_1 + x_3 + x_4 - t \ge 365,\ x_1 + x_3 + x_5 - t \ge 250,$$
$$x_1 + x_4 + x_5 - t \ge 399,\ x_2 + x_3 + x_4 - t \ge 437,\ x_2 + x_3 + x_5 - t \ge 316,\ x_2 + x_4 + x_5 - t \ge 412,$$

$x_3 + x_4 + x_5 - t \ge 514,\ x_1 + x_2 + x_3 + x_4 - t \ge 754,\ x_1 + x_2 + x_3 + x_5 - t \ge 639,$
$x_1 + x_2 + x_4 + x_5 - t \ge 735,\ x_1 + x_3 + x_4 + x_5 - t \ge 764,\ x_2 + x_3 + x_4 + x_5 - t \ge 830,$
$x_1 + x_2 + x_3 + x_4 + x_5 = 1153.$

Эта и последующие ЗЛП были решены при помощи сервиса «поиск решения», являющегося надстройкой программы excel.

Решение данной задачи имеет вид (229,5, 151, 316, 399, 57,5), $t_1 = 57{,}5$. При подстановке найденного решения в ограничения получаем, что минимальная переплата t_1 имеет место для коалиций $\{5\}, \{4,\ 5\}, \{1,\ 2,\ 3\}$. Заметим, что коалиции $\{1,\ 2,\ 3\}$ и $\{4,\ 5\}$ являются взаимодополняющими и увеличение выплаты в одной и них приведет к уменьшению в другой, поэтому фиксируем величины выплат по этим коалициям, удаляем соответствующие ограничения неравенства, вводим дополнительное равенство $x_1 + x_2 + x_3 = 696{,}5$ (введение этого равенства автоматически фиксирует выплату и по коалиции $\{4,\ 5\}$) и решаем полученную ЗЛП. Решение второй ЗЛП имеет вид $\left(193\frac{1}{3},\ 259\frac{1}{3},\ 243\frac{5}{6},\ 187\frac{1}{6},\ 269\frac{1}{3}\right)$, $t_2 = 129\frac{2}{3}$. При подстановке найденного решения в ограничения видим, что минимальная переплата t_2 имеет место для коалиций $\{1,\ 2\}$, $\{1,\ 3,\ 4,\ 5\}$, $\{2,\ 3,\ 4,\ 5\}$, $\{1,\ 2,\ 3,\ 4\}$.

Рассмотрим тройку коалиций $\{1,\ 2\}$, $\{1,\ 3,\ 4,\ 5\}$, $\{2,\ 3,\ 4,\ 5\}$. Сумма выплат по трем коалициям фиксирована и равна $2W(N)$. Следовательно, увеличение выплаты по любой из трех коалиций неизбежно приведет к ее уменьшению по какой-либо из остальных, поэтому величины выплат $x_1 + x_2,\ x_1 + x_3 + x_4 + x_5,\ x_2 + x_3 + x_4 + x_5$ должны быть зафиксированы. Поскольку величина $x_4 + x_5$ была фиксирована на предыдущем шаге, фиксированными являются суммы $x_1 + x_2,\ x_1 + x_3, x_2 + x_3$. Значит, фиксированными являются сами значения $x_1,\ x_2,\ x_3$.

Фиксируя значения $x_1 = 193\frac{1}{3},\ x_2 = 259\frac{1}{3},\ x_3 = 243\frac{5}{6}$ и отбрасывая ограничения, соответствующие коалициям {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 3}, {1, 4, 5}, {2, 4, 5}, {3, 4, 5}, {1, 2, 4, 5}, {1, 3, 4, 5}, {2, 3, 4, 5}, перейдем к третьей задаче. Ее решение имеет вид $\left(193\frac{1}{3},\ 259\frac{1}{3},\ 243\frac{5}{6},\ 257,\ 199\frac{1}{2}\right)$, $t_3 = 199\frac{1}{2}$.

При подстановке найденного решения в ограничения получаем, что минимальная переплата t_3 имеет место для коалиций $\{5\}, \{1,\ 2,\ 3,\ 4\}$. Поскольку это пара взаимодополняющих коалиций, сумма выплат по каждой из них фиксирована; следовательно, фиксированы значения переменных x_4, x_5. Таким образом, уже фиксированы все переменные, поэтому решение третьей ЗЛП является *n*-ядром. Округлив значения до целых, получим nucl = (193, 259, 244, 257, 200).

Задача нахождения nucleolus per capita также решается как последовательность ЗЛП, где первая задача выглядит следующим образом:

$$t \to \max,$$
$$\left(\sum_{i \in S} x_i - W(S) \ge |S| \cdot t\right)(\forall S \subset N, S \ne N),$$
$$\sum_{i \in N} x_i = W(N),\ x_i \ge 0.$$

$t \to \max$
$x_1 - t \ge 0,\ x_2 - t \ge 0,\ x_3 - t \ge 0,\ x_4 - t \ge 0,\ x_5 - t \ge 0,\ t \ge 0,\ x_1 + x_2 - 2t \ge 323,$
$x_1 + x_3 - 2t \ge 180,\ x_1 + x_4 - 2t \ge 0,\ x_1 + x_5 - 2t \ge 0,\ x_2 + x_3 - 2t \ge 316,\ x_2 + x_4 - 2t \ge 13,$
$x_2 + x_5 - 2t \ge 0,\ x_3 + x_4 - 2t \ge 115,\ x_3 + x_5 - 2t \ge 0,\ x_4 + x_5 - 2t \ge 399,\ x_1 + x_2 + x_3 - 3t \ge 639,$

$x_1 + x_2 + x_4 - 3t \ge 336,\ x_1 + x_2 + x_5 - 3t \ge 323,\ x_1 + x_3 + x_4 - 3t \ge 365,\ x_1 + x_3 + x_5 - 3t \ge 250,$
$x_1 + x_4 + x_5 - 3t \ge 399,\ x_2 + x_3 + x_4 - 3t \ge 437,\ x_2 + x_3 + x_5 - 3t \ge 316,\ x_2 + x_4 + x_5 - 3t \ge 412,$
$x_3 + x_4 + x_5 - 3t \ge 514,\ x_1 + x_2 + x_3 + x_4 - 4t \ge 754,\ x_1 + x_2 + x_3 + x_5 - 4t \ge 639,$
$x_1 + x_2 + x_4 + x_5 - 4t \ge 735,\ x_1 + x_3 + x_4 + x_5 - 4t \ge 764,\ x_2 + x_3 + x_4 + x_5 - 4t \ge 830,$
$x_1 + x_2 + x_3 + x_4 + x_5 = 1153.$

Решение данной задачи имеет вид (231, 151, 326, 422, 23), $t_1 = 23$.

При подстановке найденного решения в ограничения получаем, что минимальная переплата t_1 имеет место для коалиций $\{5\}$, $\{1, 2, 3\}$, $\{1, 2, 3, 5\}$, {1, 2, 4, 5}, {4, 5}. Заметим, что коалиции $\{1, 2, 3\}$, {4, 5} взаимодополняющие, поэтому заменяем неравенства, соответствующие этим коалициям, на равенства и переходим ко второй задаче. Ее решение имеет вид $\left(182\frac{1}{3},\ 248\frac{1}{3},\ 277\frac{1}{3},\ 373\frac{1}{3},\ 71\frac{2}{3}\right)$, $t_2 = 35\frac{1}{6}$.

При подстановке найденного решения в ограничения получаем, что минимальная переплата t_2 имеет место для коалиций {1, 2, 3, 5}, {1, 2, 4, 5}, {1, 3, 4, 5}, {2, 3, 4, 5}. Заметим, что величины выплат по коалициям {2, 3, 4, 5}, {1, 3, 4, 5} и {1, 2, 4, 5} составляют $W(N) - x_1$, $W(N) - x_2$, $W(N) - x_3$ соответственно. Поскольку величина $x_1 + x_2 + x_3$ была фиксирована при решении первой задачи, увеличение выплат по одной из трех коалиций неизбежно ведет к уменьшению выплаты по какой-либо другой. Значит, значения x_1, x_2, x_3 следует фиксировать. Фиксируя значения $x_1 = 182\frac{1}{3}$, $x_2 = 248\frac{1}{3}$, $x_3 = 277\frac{1}{3}$ и отбрасывая ограничения, соответствующие коалициям {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 3}, {1, 4, 5}, {2, 4, 5}, {3, 4, 5}, {1, 2, 4, 5}, {1, 3, 4, 5}, {2, 3, 4, 5}, перейдем к третьей задаче. Ее решение имеет вид $\left(182\frac{1}{3},\ 248\frac{1}{3},\ 277\frac{1}{3},\ 280,\ 165\right)$, $t_3 = 58\frac{1}{2}$.

При подстановке найденного решения в ограничения получаем, что минимальная переплата t_3 имеет место для коалиций {1, 2, 3, 4}, {1, 2, 3, 5}. Поскольку величины x_1, x_2, x_3 были фиксированы ранее, сумма выплат по данной паре коалиций фиксирована. Поэтому следует зафиксировать x_4, x_5. Значит, найденное решение третьей ЗЛП является искомым вектором nucleolus per capita.

Вычитая векторы экономии из вектора затрат городов при радиальных маршрутах (576, 394, 596, 706, 556), получим три способа распределения затрат в кольцевом маршруте, включающем все города: Sh=(377, 153, 347, 444, 354), nucl=(383, 135, 352, 449, 356), npc=(394, 146, 319, 426, 390).

Для наглядности полученные результаты представим в табл. 3.

Таблица 3

	Витебск	Могилев	Гомель	Брест	Гродно
Sh	377	153	347	444	354
nucl	383	135	352	449	356
npc	394	146	319	426	390

Заключение

Рассмотренный пример демонстрирует три способа распределения транспортных затрат. Каждый из способов является справедливым в определенном смысле и имеет свои преимущества и недостатки, а определить абсолютную справедливость распределения в данном случае не представляется возможным. К достоинствам распределения затрат по Шепли сле-

дует отнести простоту вычисления, к недостаткам – то, что ВШ может не принадлежать ядру, и таким образом в общем случае среди игроков, недовольных распределением, могут возникать сепаратистские тенденции, приводящие к развалу гранд-коалиции. Распределения затрат на основе nucleolus и nucleolus per capita вычисляются гораздо сложнее, зато найденный вектор распределения затрат всегда принадлежит ядру, если только ядро распределения непусто.

Список литературы

1. Schmeidler, D. The nucleolus of a charecteristic function game / D. Schmeidler // SIAM J. on Applied Mathematics. – 1969. – Vol. 17, no. 6. – P. 1163–1170.

2. Grotte, J.H. Computation of and observation on the nucleolus and central games / J.H. Grotte // M. Sc. Thesis. – N. Y. : Cornell university, 1970.

3. Moulin, H. Axioms of cooperative decision making / H. Moulin. – Cambridge university press, 1988.

4. Curiel, I. Cooperative game theory and applications / I. Curiel. – Springer US, 1997. – Vol. 16. – 194 p.

Поступила 19.01.2017

Киевский национальный университет
им. Тараса Шевченко,
Киев, пр. Глушкова, 4Д
e-mail: sergei204@ukr.net

S.I. Dotsenko

ON COSTS ALLOCATION AT CIRCULAR ROUTE CONVEYING

An applied problem of finding an optimal distribution of benefits among members of some cooperative transportation of goods is considered. The theory of cooperative games as a basic model is used and some concepts of this theory are discussed . The notions of the Shapley vector, the nucleolus and the nucleolus per capita are applied to describe optimal vectors of payoffs and value vectors. The algorithms for constructing these vectors are proposed, and all optimal value vectors for the proposed problem are found.

УДК 004.932

Г.А. Буткин, И.А. Емельянов, А.В. Тузиков

АЛГОРИТМЫ ПОСТРОЕНИЯ ДЕСКРИПТОРОВ ЛОКАЛЬНЫХ ОСОБЕННОСТЕЙ ИЗОБРАЖЕНИЙ НА БАЗЕ МНОГОКОЛЬЦЕВЫХ НЕПАРАМЕТРИЧЕСКИХ ПРЕОБРАЗОВАНИЙ

Рассматривается применение непараметрических преобразований при построении дескрипторов локальных особенностей изображений. Предлагаются подходы, в основу которых положены алгоритмы построения масштабируемых дискретных окружностей по Брезенхему, для многокольцевых непараметрических преобразований. Приводятся результаты моделирования предложенного подхода к построению локальных дескрипторов на тестовой базе по сравнению с известными дескрипторами SURF, ORB, BRISK, FREAK.

Введение

При решении целого ряда задач обработки изображений используются дескрипторы локальных особенностей изображений, которые должны быть инвариантны к различного рода искажениям и обладать высоким быстродействием [1–6]. В этой связи представляет интерес рассмотрение дескрипторов и детекторов углов, использующих непараметрические преобразования, которые по своей природе являются инвариантными к изменениям уровня освещенности [7] и при некоторых конструкциях – робастными к шумам [8], а также характеризуются низкой трудоемкостью вычисления и простотой сопоставления. Однако традиционные непараметрические преобразования [7] являются неинвариантными к повороту. Впервые идея непараметрических преобразований, инвариантных к вращению, была предложена в работах [9, 10] при рассмотрении дескрипторов типа LBP (Local Binary Patterns) для описания текстур. В этом случае непараметрические преобразования локальных особенностей изображения рассматривались на окружностях различных радиусов с заданным числом фиксируемых с помощью билинейной аппроксимации точек. Предлагаемый подход состоит в построении дескриптора локальных особенностей изображений с использованием непараметрических преобразований на множестве колец, формируемых по алгоритму Брезенхема [11] с учетом дискретного пространства пикселов. В рассматриваемом дескрипторе сначала на двух кольцах с малыми радиусами осуществляется поиск характерных точек (углов) объектов изображения. Затем добавляются дополнительные кольца с возрастающими радиусами и на них путем использования непараметрических преобразований на задаваемых точках колец строится дескриптор найденных характерных точек в виде бинарного вектора.

1. Кольцевые непараметрические преобразования

Известно [7], что непараметрические преобразования оперируют не параметрами изображения, например интенсивностью (яркостью), а отношениями между параметрами. Предположим, что p является пикселом изображения, а $I(p)$ – его интенсивностью. Обозначим через $N(p^*)$ набор пикселов на некотором кольце Брезенхема радиусом r пикселов, окружающем пиксел p^*. Число пикселов в наборе $N(p^*)$ в принципе может быть любым и произвольно расположенным по кольцу, но не может превышать число пикселов в кольце Брезенхема заданного радиуса. Непараметрическое кольцевое преобразование основывается на установлении результатов сравнения интенсивности центрального пиксела p^* с интенсивностями пикселов из его окружения $N(p^*)$.

Если $\xi(p^*, p)$ является результатом сравнения пары пикселов, то можно принять

$$\xi(p^*, p) = \begin{cases} 1, & \text{если } \mathrm{I}(p) > \mathrm{I}(p^*); \\ 0, & \text{если } \mathrm{I}(p) \le \mathrm{I}(p^*). \end{cases} \quad (1)$$

Представление в виде набора результатов сравнения пар пикселов по кольцу, начиная с первого пиксела p_1 (например, с координатами $x = r, y = 0$) и заканчивая последним пикселом в кольце p_n, будем считать классическим кольцевым непараметрическим преобразованием локальной окрестности пиксела p^*:

$$C(p^*) = U\{p, \xi(p^*, p)\}, \quad (2)$$

где $p = p_1, p_2, p_3, \ldots, p_n$.

Последовательная запись непараметрического преобразования формирует бинарную последовательность результатов сравнений пикселов кольцевого окружения $N(p^*)$ с пикселом p^*. На множестве колец различных радиусов можно сформировать обобщенный бинарный вектор (дескриптор), характеризующий рассматриваемую локальную особенность.

На рис. 1 показаны примеры дискретных аппроксимаций по алгоритму Брезенхема окружностей с радиусами r = 1, 3, 6 и 9 пикселов и их координаты (для r = 6 и 9 пикселов см. табл. 1) в первом квадранте, которые в целом образуют кольца соответственно с 8, 16, 32 и 52 пикселами. Значения координат для пикселов в других квадрантах можно легко получить, используя симметрию и отражение.

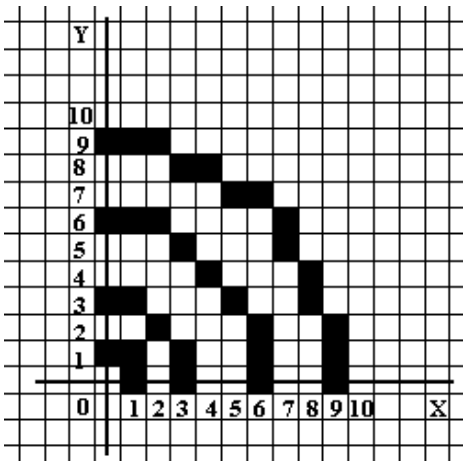


Рис. 1. Фрагменты дискретных аппроксимаций по алгоритму Брезенхема окружностей в первом квадранте с радиусами r = 1, 3, 6 и 9 пикселов

Таблица 1

Последовательности пикселов и их координаты при дискретной аппроксимации окружностей с различными радиусами по алгоритму Брезенхема

r = 6 пикселов										r = 9 пикселов													
Номер пиксела	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Координата по X	6	6	6	5	4	3	2	1	0	9	9	9	8	8	7	7	6	5	4	3	2	1	0
Координата по Y	0	1	2	3	4	5	6	6	6	0	1	2	3	4	5	6	7	7	8	8	9	9	9

В качестве описаний кольцевых непараметрических преобразований в соответствии с [7] могут рассматриваться ранговая форма (rank transform) и форма в виде последовательной записи (census transform).

Первая традиционная форма (rank transform) непараметрического преобразования определяет число пикселов в области окружения, интенсивность которых больше интенсивности центрального пиксела:

$$R(p^*) = |\{p \in N(p^*), I(p) > I(p^*)\}|. \quad (3)$$

Вторая форма непараметрического преобразования (census transform) отображается в виде записи, представляющей бинарную последовательность результатов сравнений из набора пикселов кольцевого окружения $N(p^*)$ с центральным пикселом p^*. Примеры формирования такой записи при $n=16$ показаны на рис. 2 и для разных характерных локальных особенностей изображений могут быть реализованы в виде отличающихся по содержанию строк размерностью в 16 битов.

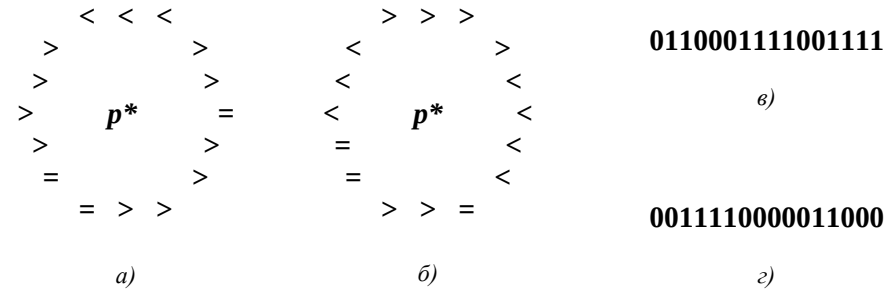


Рис. 2. Примеры формирования кольцевого непараметрического преобразования в виде census transform: *а), б)* результаты сравнения пикселов кольцевого окружения с центральным пикселом; *в), г)* сформированные соответствующие census-векторы

Форма census transform является удобным средством для сопоставления локальных особенностей характерных точек изображений по заданному порогу расстояния Хемминга:

$$D(a,b)=\sum_{i=1}^{N} a_i \mathrm{XOR} b_i\,, \tag{4}$$

где a_i и b_i – элементы сравниваемых бинарных векторов a и b длины N; XOR – логическая операция «исключающее ИЛИ».

2. Алгоритм поиска характерных точек в виде углов для дескриптора на базе непараметрических преобразований на кольцах

Алгоритм поиска углов на изображении представляется в виде двух этапов, первый из которых включает анализ присутствия признаков угла в анализируемой точке, а второй – анализ наличия самого угла [12]. На первом этапе производится анализ наличия вершины темного или светлого углов. Такой анализ в окне размером 3×3 пиксела позволяет выделить точку – претендента на угол, а если такой точки нет, то перейти к анализу следующей точки-претендента, не анализируя наличие самого угла. Если точка – претендент на вершину угла – обнаружена, то осуществляется переход на второй этап поиска угла со сторонами длиной не менее трех пикселов. Второй этап аналогичен алгоритму, реализуемому в известном детекторе углов FAST (Features from Accelerated Segment Test) [13]. Однако в предлагаемом детекторе углов имеются особенности, связанные с применением хеш-таблиц. Кроме того, применение двухэтапного алгоритма позволяет исключать из рассмотрения лежащие вблизи границы точки как ложные претенденты на углы.

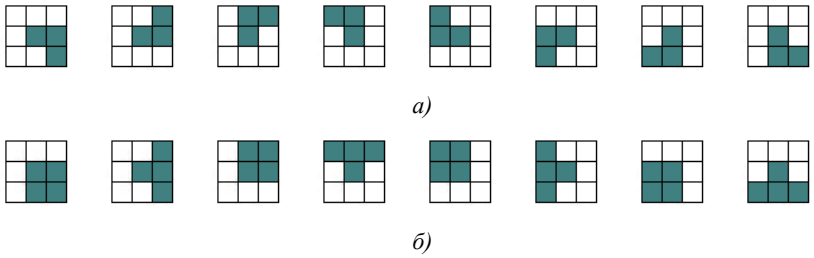
Процедура работы *первого этапа* рассматриваемого алгоритма может быть представлена в виде следующих шагов:

Шаг 1. Поиск вершины темного угла на изображении. Под вершиной темного угла подразумеваются различные сочетания между центральным пикселом и пикселами окружения в окне размером 3×3 пиксела. На рис. 3, *а* и *б* показан ряд возможных сочетаний соответственно при трех и четырех пикселах, составляющих вершину темного угла.

В окне 3×3 осуществляются непараметрические преобразования путем сравнения с учетом заданного порога σ_1 интенсивности *i*-го пиксела окружения с интенсивностью центрального пиксела I_c и формирование бинарного вектора вершины темного угла VT1:

$$\Delta I_i = I_i - (I_c + \sigma_1), \text{ где } i = 1,\ 2,\ 3,\ \ldots,\ 8. \tag{5}$$

Если $\Delta I_i > 0$, то в бинарном векторе разряд $\mathrm{VT1}(i) = 1$, в противном случае значение $\mathrm{VT1}(i) = 0$.



а)

б)

Рис. 3. Возможные сочетания начал темного угла относительно центрального пиксела в окне 3×3

Проделав цикл по $i = 1, 2, 3, \ldots, 8$, получают восьмиразрядный вектор VT1. Он представляется двоичным числом, старший разряд которого соответствует первому сравнению. При этом считается, что вершины темных углов описываются сочетаниями следующих подряд нулей (единиц) вектора VT1 в соответствии с числами 63, 126, 159, 207, 231, 243, 249, 252 (если число темных пикселов окружения равно 2) и 31, 62, 124, 143, 199, 227, 241, 248 (если число темных пикселов окружения равно 3).

Шаг 2. Бинарный вектор VT1 используется в качестве входного адреса хеш-таблицы, в которую занесены значения 2 и 3 в соответствии с числом темных пикселов окружения в вершине угла и вышеприведенными числами двоичного кода вектора VT1.

Шаг 3. Если на выходе хеш-таблицы $p = 2$ или 3, то осуществляется переход ко второму этапу алгоритма – поиску темного угла, в противном случае – к поиску наличия вершины светлого угла для анализируемой точки.

Поиск вершины светлого угла. Под вершиной светлого угла подразумеваются сочетания между центральным пикселом и пикселами окружения, которые являются инверсными по отношению к представленным на рис. 3 сочетаниям. Поэтому вычисление разности между интенсивностью центрального пиксела I_c и интенсивностью *i*-го пиксела окружения осуществляется по формуле

$$\Delta I_i = I_c - (I_i + \sigma_1), \tag{6}$$

где $i = 1, 2, 3, \ldots, 8$.

В остальном поиск вершины светлого угла аналогичен поиску вершины темного угла.

Второй этап алгоритма поиска углов базируется на анализе отношений между найденным на первом этапе пикселом вершины угла и пикселами окружения на кольце радиусом *r* = 3 пиксела, которое содержит 16 пикселов окружения. Для получения восьмиразрядного входного вектора и хэш-таблицы меньшей размерности проведено «прореживание» пикселов на кольце путем перехода от 16 к 8 пикселам. Углами будем считать число следующих подряд прореженных пикселов с большей (меньшей) интенсивностью по отношению к центральному пикселу, равное 1, 2, 3 или 4 из 8 рассматриваемых пикселов. Пошаговая процедура алгоритма поиска угла *на втором этапе* может быть представлена в следующем виде:

Шаг 1. Поиск темного угла. Сравнивается по интенсивности *i*-й пиксел на кольце с центральным пикселом I_c с учетом заданного порога σ_2 и формируется бинарный вектор VT2 по следующему правилу:

$$\Delta I_i = I_i - (I_c + \sigma_2), \tag{7}$$

где $i = 1, 2, 3, \ldots, 8$.

Если $\Delta I_i > 0$, то в бинарном векторе *i*-й разряд определяют как $\mathrm{VT2}(i) = 1$ и значение I_i суммируется в ST1, в противном случае – как $\mathrm{VT2}(i) = 0$ и значение I_i суммируется в ST2.

Проделав цикл по $i = 1, 2, 3, \ldots, 8$, получают бинарный вектор VT2 и две суммы (ST1 и ST2) соответствующих значений I_i.

Шаг 2. Использование бинарного вектора $\mathrm{VT2}$ в качестве входного адреса хеш-таблицы, в которую занесены числа 1, 2, 3 и 4 в соответствии со следующими значениями двоичного кода вектора $\mathrm{VT2}$:

число 1: 127, 191, 223, 239, 247, 251, 253, 254;

число 2: 63,126,159, 207, 231, 243, 249, 252;

число 3: 31. 62, 124, 143, 199, 227, 241, 248;

число 4: 15, 30, 60, 120, 135, 195, 225, 240.

Шаг 3. Обращение к хэш-таблице и получение на выходе числа *p*.

Если $p = 0$, то темный угол отсутствует для анализируемой точки изображения и осуществляется переход на конец алгоритма.

Если $p \neq 0$, то вычисляется величина «значимости» темного угла, которая равна модулю разности нормированных $\mathrm{ST1}$ и $\mathrm{ST2}$:

$$\mathrm{ST} = |\{[\mathrm{ST1}/(8-p)] - (\mathrm{ST2}/p)\}|. \quad (8)$$

Шаг 4. Фильтрация найденных темных углов в заданном окне: выбирается только один угол с максимальным значением значимости ST, затем осуществляется переход в конец алгоритма.

Поиск светлого угла аналогичен рассмотренному выше поиску темного угла, однако сравнение интенсивностей осуществляется по правилу

$$\Delta I_i = I_c - (I_i + \sigma_2), \quad (9)$$

где $i = 1,\ 2,\ 3,\ \ldots,\ 8.$

Фильтрация светлых углов также осуществляется в заданном окне путем выбора угла с максимальным значением значимости светлого угла.

3. Алгоритм построения многокольцевого дескриптора локальных особенностей

Как отмечалось выше, структура формируемого вокруг характерной точки дескриптора локальных особенностей изображения описывается набором колец, которые строятся по алгоритму Брезенхема описания окружности с различными радиусами на дискретно заданных точках (пикселах) плоскости изображения. В данном алгоритме на каждом шаге точка формируемого кольца шириной в один пиксел определяется путем рассмотрения трех пикселов (горизонтального, диагонального и вертикального) и выбора из них пиксела с минимальным квадратом разности между его расстоянием до точки и радиусом задаваемой окружности. Задавая на дискретном поле растрового изображения различные радиусы в целых числах (пикселах), можно получать аппроксимации окружностей (колец) с различным числом пикселов. В качестве примера локального дескриптора может быть приведен дескриптор со структурой, состоящей из четырех колец (см. рис. 1) с максимально возможным числом пикселов 8, 16, 32 и 52 соответственно. В общем случае и число колец в дескрипторе, и значения соответствующих радиусов могут быть различными. Эти параметры устанавливаются как компромиссное решение между точностью и трудоемкостью вычисления дескриптора.

Для обеспечения инвариантности описанного алгоритма к изменению масштаба необходимо увеличить радиусы анализируемых колец на величину масштаба и осуществить смещение координат точек на кольце.

Алгоритм построения локального дескриптора с изменяемым масштабом *M* показан на примере построения кольца с радиусом $r = 3$ пиксела вокруг характерной точки с координатами $X = i,\ Y = j$. В табл. 2 представлены изменения координат пикселов окружения при изменении масштаба в *M* раз.

Таблица 2

Значения координат для кольца радиусом $r = 3$ пиксела при изменении масштаба дескриптора в M раз

Номер точки	Координата по оси X	Координата по оси Y	Номер точки	Координата по оси X	Координата по оси Y
1	$i+3\times M$	$j+0\times M$	9	$i-3\times M$	$j+0\times M$
2	$i+3\times M$	$j+1\times M$	10	$i-3\times M$	$j-1\times M$
3	$i+2\times M$	$j+2\times M$	11	$i-2\times M$	$j-2\times M$
4	$i+1\times M$	$j+3\times M$	12	$i-1\times M$	$j-3\times M$
5	$i+0\times M$	$j+3\times M$	13	$i+0\times M$	$j-3\times M$
6	$i-1\times M$	$j+3\times M$	14	$i+1\times M$	$j-3\times M$
7	$i-2\times M$	$j+2\times M$	15	$i+2\times M$	$j-2\times M$
8	$i-3\times M$	$j+1\times M$	16	$i+3\times M$	$j-1\times M$

Как следует из табл. 2, при изменении масштаба в M раз радиус кольца также увеличивается в M раз и координаты пикселов окружения получают смещение на величину M по сравнению с единичным масштабом. Несложно показать, что аналогичные изменения происходят с координатами пикселов окружения для колец с другими радиусами.

При изменении масштаба также может производиться замена отдельных пикселов на масштабируемые окна (суперпикселы) размером $M\times M$ пикселов и осуществляться усреднение в них значений интенсивности (яркости) пикселов. Средняя интенсивность масштабируемого пиксела I_M с координатами i, j может быть вычислена следующим образом:

если M – нечетное число, то

$$I_M = \left(\sum_{-(M-1)/2}^{(M-1)/2}\sum_{-(M-1)/2}^{(M-1)/2} I_{(i+k),(j+l)}\right)/M^2, \qquad (10)$$

где $k = -(M-1)/2,\ \ldots, 0,\ \ldots, (M-1)/2$; $l = -(M-1)/2,\ \ldots, 0,\ \ldots, (M-1)/2$;

если M – четное число, то

$$I_M = \left(\sum_{k=-\frac{M}{2}}^{(\frac{M}{2})-1}\sum_{l=-[(\frac{M}{2})-1]}^{(\frac{M}{2})} I_{(i+k),(j+l)}\right)/M^2, \qquad (11)$$

где $k = -M/2,\ \ldots, 0,\ \ldots,\ [(M/2)-1]$; $l = -[(M/2)-1],\ \ldots, 0,\ \ldots,\ (M-1)/2$.

Из выражений (10), (11) следует:

$I_1 = I_{(i),(j)}$ при $M = 1$, $k = 0$ и $l = 0$;

$I_2 = \left(\sum_{k=-1}^{0}\sum_{l=0}^{1} I_{(i+k),(j+l)}\right)/4$ при $M = 2$, $k = -1, 0$ и $l = 0, 1$;

$I_3 = \left(\sum_{k=-1}^{1}\sum_{l=-1}^{1} I_{(i+k),(j+l)}\right)/9$ при $M = 3$, $k = -1, 0, 1$ и $l = -1, 0, 1$;

$I_4 = \left(\sum_{k=-2}^{1}\sum_{l=-1}^{2} I_{(i+k),(j+l)}\right)/16$ при $M = 4$, $k = -2, -1, 0, 1$ и $l = -1, 0, 1, 2$;

$I_5 = \left(\sum_{k=-2}^{2}\sum_{l=-2}^{2} I_{(i+k),(j+l)}\right)/25$ при $M = 5$, $k = -2, -1, 0, 1, 2$ и $l = -2, -1, 0, 1, 2$;

$I_6 = \left(\sum_{k=-3}^{2}\sum_{l=-2}^{3} I_{(i+k),(j+l)}\right)/36$ при $M = 6$, $k = -3, -2, -1, 0, 1, 2$ и $l = -2, -1, 0, 1, 2, 3$;

$I_7 = \left(\sum_{k=-3}^{3}\sum_{l=-3}^{3} I_{(i+k),(j+l)}\right)/49$ при $M = 7$, $k = -3, -2, -1, 0, 1, 2, 3$ и $l = -3, -2, -1, 0, 1, 2, 3$

и т. д.

Таким образом, алгоритм изменения масштаба дескриптора связан с изменением радиуса его колец в M раз по сравнению с единичным масштабом и смещением на M единиц координат соответствующих пикселов окружения в кольцах.

Рассмотрим структуру локального дескриптора, заданного четырьмя кольцами: радиусами в 1 и 3 пиксела, а также последующими кольцами с масштабами $M = 3$ и $M = 5$ по отношению ко второму кольцу, т. е. с радиусами в 9 и 15 пикселов соответственно. Таким образом, представленный дескриптор будет содержать 56 (8+3×16) пикселов окружения.

Затем алгоритм построения дескриптора сводится к выбору на множестве пикселов колец пар сравнения, на основании результатов которых формируется бинарный вектор. Вариантов построения пар пикселов сравнения может быть несколько. Например, это может быть традиционное сравнение центрального пиксела (характерной точки) с пикселами окружения, расположенными на кольцах. Однако такой подход обладает низкой помехоустойчивостью при изменении интенсивности центрального пиксела [14]. Вторым вариантом является выбор пар сравнения интенсивностей пикселов, расположенных диаметрально симметрично на кольцах или лежащих на концах их хорд. Может быть осуществлен случайный выбор пар сравнения пикселов, заданных на кольцах, как это делается в алгоритме BRIEF (Binary robust independent elementary features) [3]. При построении рассматриваемого дескриптора, состоящего из 56 пикселов окружения, возможно построение 1540 пар сравнения. Поэтому, задавая размерность бинарного вектора дескриптора, используем аналогично [4, 6] выбор пар сравниваемых пикселов, которые характеризуются наибольшим значением разности и наименьшей корреляцией.

Сопоставление сформированных локальных дескрипторов проводится путем вычисления расстояния Хемминга рассматриваемых бинарных векторов (реализации логической процедуры «исключающее ИЛИ» (XOR) и сравнения числа единиц несовпадения с установленными порогами). Современные архитектуры имеют только одну инструкцию (POPCNT), чтобы сравнить по XOR и подсчитать число бит в бинарном векторе [3]. Это обеспечивает высокое быстродействие при вычислении расстояния Хемминга.

4. Экспериментальные результаты работы дескрипторов на базе многокольцевых непараметрических преобразований

Оценка предложенного подхода к построению локальных дескрипторов изображений на базе многокольцевых непараметрических преобразований базировалась на проверке результатов работы предложенных дескрипторов на широко используемой базе тестовых изображений [15, 16], состоящей из восьми видов изображений (graf, ubc, wall, bikes, trees, bark, leuven, boat). Каждый вид содержит по шесть изображений с различными типами и пятью степенями искажений. В качестве примера рассматривались дескрипторы размерностью 256 битов (New256), 128 битов (New128 и New128* – произвольный выбор из 256 пар сравнения) и 64 бита (New64), построенные на четырех масштабируемых кольцах Брезенхема с общим окружением в 56 пикселов. Проводилось сравнение работы предложенных дескрипторов с известными дескрипторами SURF [2], ORB [4], BRISK [5], FREAK[6], представленными в открытой библиотеке OpenCV 2.4. В качестве критериев оценки работы дескрипторов использовались ставшие традиционными показатели точности (precision) и полноты (recall) [15, 16].

В табл. 3 и 4 и на рис. 4 и 5 приведены результаты экспериментальной проверки работы дескрипторов на отдельных тестовых изображениях в виде усредненных показателей точности и полноты соответственно.

Таблица 3

Среднее значение точности работы дескрипторов на отдельных тестовых изображениях

Изображение	SURF	ORB	BRISK	BRISK+ +FREAK	New256	New128	New128*	New64
graf	0,34	0,46	0,37	0,34	0,43	0,41	0,35	0,31
ubc	0,93	0,98	0,82	0,67	0,97	0,96	0,97	0,95
wall	0,71	0,68	0,52	0,28	0,66	0,57	0,56	0,41
bikes	0,84	0,89	0,32	0,23	0,85	0,84	0,84	0,79
trees	0,58	0,80	0,46	0,67	0,75	0,65	0,68	0,55
bark	0,63	0,68	0,44	0,18	0,55	0,49	0,54	0,34
leuven	0,87	0,86	0,83	0,73	0,85	0,80	0,81	0,77
boat	0,70	0,90	0,30	0,15	0,82	0,78	0,78	0,67

Таблица 4

Среднее значение полноты работы дескрипторов для различных тестовых изображений

Изображение	SURF	ORB	BRISK	BRISK+ +FREAK	New256	New128	New128*	New64
graf	0,22	0,18	0,11	0,15	0,18	0,17	0,15	0,15
ubc	0,82	0,82	0,32	0,37	0,82	0,80	0,81	0,77
wall	0,47	0,25	0,10	0,11	0,25	0,23	0,22	0,19
bikes	0,64	0,57	0,09	0,11	0,54	0,53	0,52	0,50
trees	0,25	0,37	0,08	0,37	0,35	0,33	0,34	0,31
bark	0,61	0,38	0,06	0,06	0,37	0,34	0,35	0,32
leuven	0,68	0,60	0,33	0,35	0,59	0,56	0,55	0,52
boat	0,53	0,41	0,06	0,07	0,42	0,41	0,41	0,35

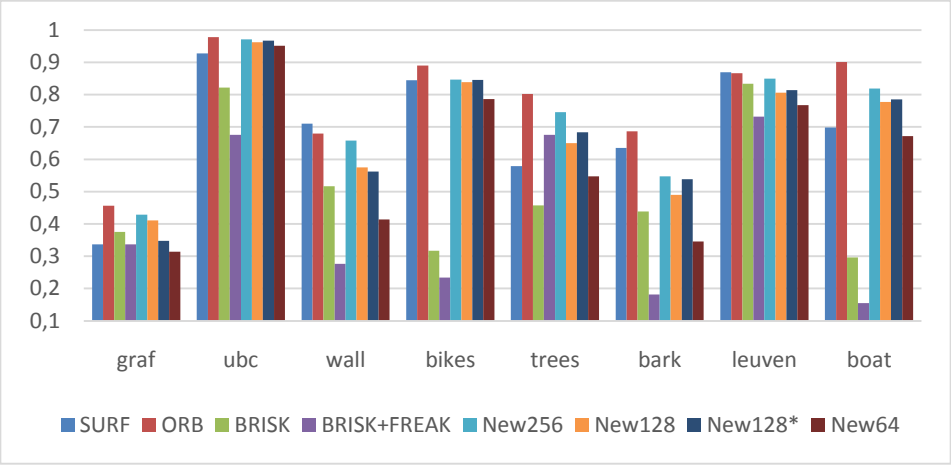


Рис. 4. Диаграммы средней точности работы дескрипторов
на отдельных тестовых изображениях

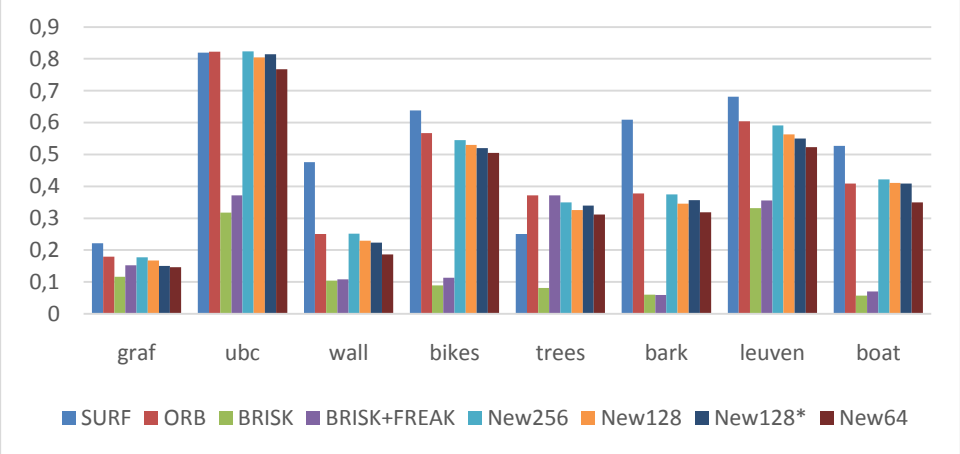


Рис. 5. Диаграммы средней полноты работы дескрипторов
на различных тестовых изображениях

В табл. 5 и на рис. 6 представлены усредненные в целом по тестовой базе изображений результаты работы дескрипторов в виде показателей точности и полноты.

Таблица 5

Средние значения точности и полноты работы дескрипторов на тестовой базе изображений

Показатели	SURF	ORB	BRISK	BRISK+ +FREAK	New256	New128	New128*	New64
precision	0,70	0,78	0,51	0,41	0,73	0,69	0,69	0,60
recall	0,53	0,45	0,14	0,20	0,44	0,42	0,42	0,39

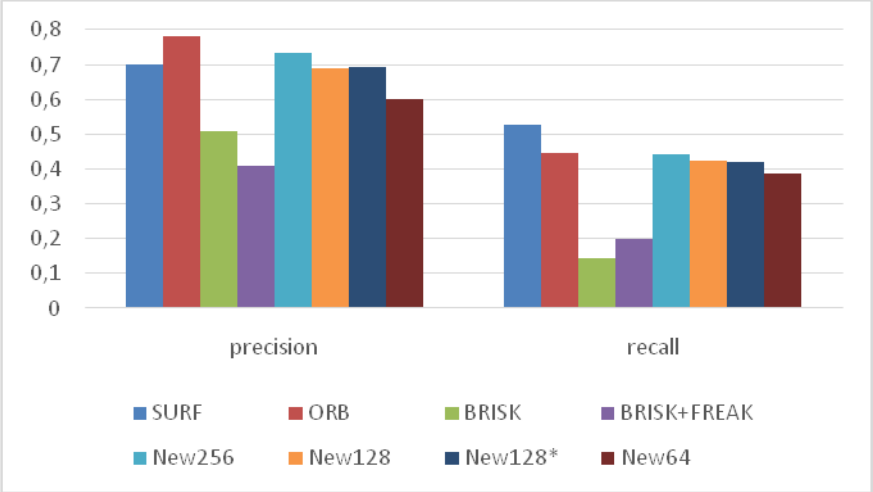


Рис. 6. Диаграммы средней точности и полноты работы дескрипторов на тестовой базе изображений

Как следует из приведенных результатов, предложенный подход к построению локальных дескрипторов на базе многокольцевых непараметрических преобразований на масштабируемых кольцах Брезенхема позволяет создавать конструкции дескрипторов, которые могут быть конкурентоспособными известным дескрипторам. Результаты экспериментальной проверки работы предложенного подхода к построению локальных дескрипторов показали, что качество работы дескриптора по критериям точности и полноты зависит от структуры обрабатываемого изображения. На рис. 7 и 8 приведены примеры графиков точности для тестовых изображений graf и ubc при различных уровнях искажений.

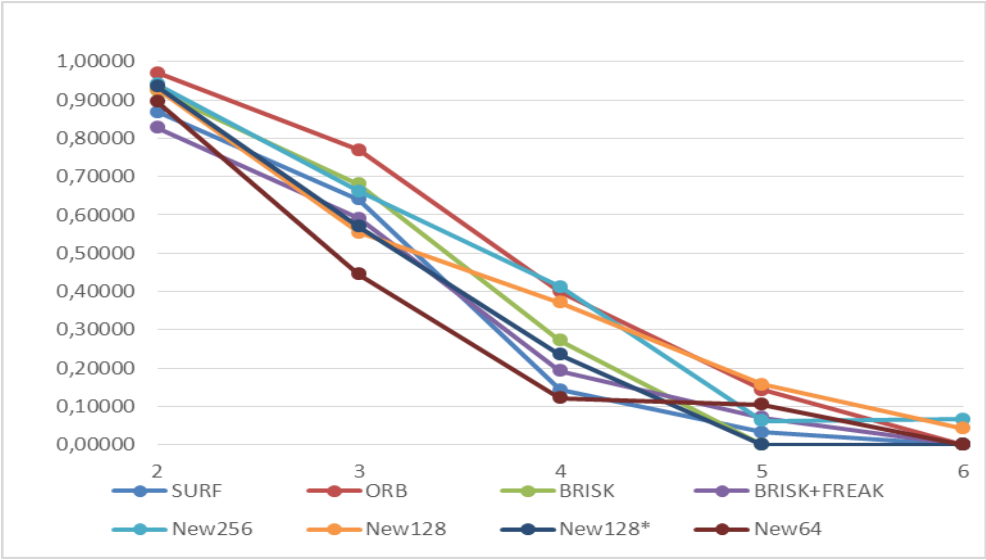


Рис. 7. Графики точности работы дескрипторов на тестовом изображении graf при пяти различных изменениях угла съемки

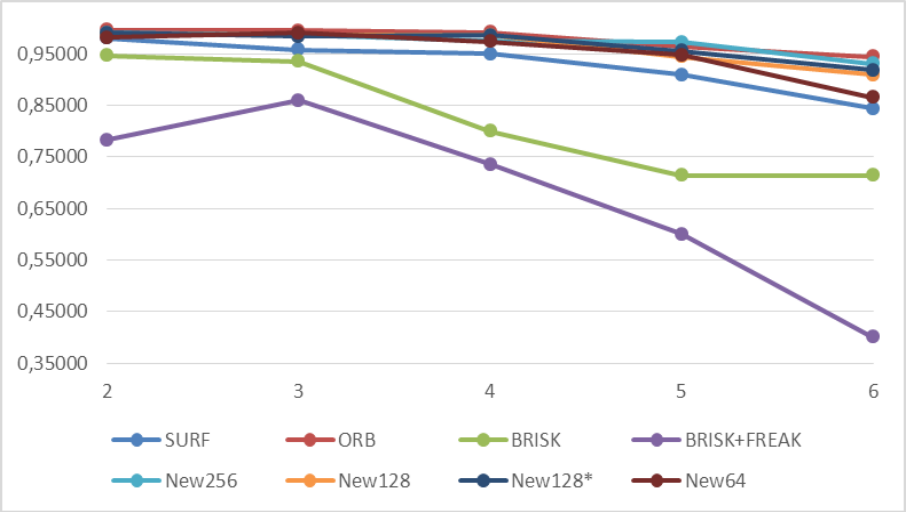


Рис. 8. Графики точности работы дескрипторов на тестовом изображении ubc при пяти различных степенях сжатия

Следует также отметить, что в общем случае значения показателей зависят от размера формируемого бинарного вектора. Однако, например, для набора изображений ubc (искажение типа «различные степени JPEG-сжатия») дескрипторы размером 256, 128 и 64 бита дают по точности примерно равные хорошие результаты. Поэтому представляет интерес исследование выбора эффективных параметров дескрипторов для работы на разнообразных структурах изображений.

Заключение

Экспериментальная проверка работы дескрипторов локальных особенностей изображений на базе многокольцевых непараметрических преобразований показала, что такие дескрипторы в зависимости от задаваемого размера бинарного вектора обладают различными значениями показателей точности и полноты, являются конкурентоспособными известным дескрипторам и могут быть использованы в качестве детекторов углов и описателей локальных особенностей изображений.

Список литературы

1. Lowe, D. Distinctive image features from scale-invariant key points / D. Lowe // Intern. J. of Computer Vision. – 2004. – Vol. 2(60). – P. 91–110.

2. Bay, H. SURF: Speeded Up Robust Features / H. Bay, T. Tuytelaars, L. van Gool // Proc. of the 9th European Conf. on Computer Vision. – Graz, Austtria : Springer LNCS, 2006. – Vol. 3951, pt. 1. – P. 404–417.

3. Brief: Binary robust independent elementary features / M. Calonder [et al.] // Proc. of the 11th European conf. on Computer vision (ECCV 2010), sept. 5–11, 2010. – Heraklion, Crete, Greece, 2010. – P. 778–792.

4. ORB: an efficient alternative to SIFT or SURF / E. Rublee [et al.] // IEEE Intern. Conf. on Computer Vision (ICCV 2011), nov. 6–13, 2011. – California, 2011. – P. 2564–2571.

5. Leutenegger, S. BRISK: Binary robust invariant scalable keypoints / S. Leutenegger, M. Chli, R.Y. Siegwart // IEEE Intern. Conf. on Computer Vision (ICCV 2011), nov. 6–13, 2011. – California, 2011. – P. 2548–2555.

6. Alahi, A. Freak: Fast retina keypoint / A. Alahi, R. Ortiz, P. Vandergheynst // Computer Vision and Pattern Recognition (CVPR), 16–21 June 2012. – Switzerland, 2012. – P. 510–517.

7. Zabih, R. Non-parametric Local Transforms for Computing Visual Correspondence / R. Zabih, J. Woodfill // Proc. of 3rd European Conf. on Computer Vision. – Sweden, 1994. – P. 150–158.

8. Butkin, G. Nonparametric transforms for describing local image features / G. Butkin, D. Zhuk, A. Tuzikov // Conf. Pattern Recognition and Information Proc. (PRIP'2011), 18–20 May 2011. – Minsk : BSUIR, 2011. – P. 209–212.

9. Ojala, T. A comparative study of texture measures with classification based on feature distributions / T. Ojala, M. Pietikainen, D. Harwood // Pattern Recognition. – 1996. – Vol. 29(1). – P. 51–59.

10. Ojala, T. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns / T. Ojala, M. Pietikäinen, T. Mäenpää // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2002. – Vol. 24, no. 7. – P. 971–987.

11. Роджерс, Д. Алгоритмические основы машинной графики / Д. Роджерс. – М. : Мир, 1989. – 512 с.

12. Буткин, Г.А. Построение детекторов углов на базе анализа непараметрических преобразований на кольцах / Г.А. Буткин, И.А. Емельянов, А.В. Тузиков // Материалы VI Белорусского космического конгресса, 28–30 октября 2014 г. – Минск : ОИПИ НАН Беларуси, 2014. – Т. 1. – С. 257–260.

13. Rosten, E. Machine Learning for High Speed Corner Detection / E. Rosten, T. Drummond // Proc. Ninth European Conf. Computer Vision. – 2006. – Vol. 1. – P. 430–443.

14. Буткин, Г.А. Об устойчивости непараметрических преобразований в задачах описания локальных особенностей изображений / Г.А. Буткин, А.В. Тузиков // Информатика. – 2011. – № 1. – С. 15–24.

15. Mikolajczyk, K. A performance evaluation of local descriptors / K. Mikolajczyk, C. Schmid // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2005. – Vol. 27, no. 10. – P. 1615–1630.

16. Miksik, O. Evaluation of Local Detectors and Descriptors for Fast Feature Matching / O. Miksik, K. Mikolajczyk // Proc. of the Intern. Conf. of Pattern Recognition (ICPR), 11–15 nov. 2012. – Tsukuba Science City, Japan, 2012. – P. 2681–2684.

Поступила 04.10.2016

Объединенный институт проблем информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: butkin@newman.bas-net.by,
bujhm24@gmail.com,
tuzikov@newman.bas-net.by

G.A. Butkin, I.A. Emelianov, A.V. Tuzikov

ALGORITHMS FOR CONSTRUCTING THE DESCRIPTORS OF LOCAL IMAGE FEATURES BASED ON MULTIRING NONPARAMETRIC TRANSFORMATION

Application of nonparametric transformations for construction of local image features descriptors is considered. An approach based on scalable Bresenham circles for multiring nonparametric transformations is proposed. A problem of finding characteristic points (corners) in images is solved using two rings of small radius. Then, considering rings of greater radius a final decision is made concerning local features under investigation. Testing results of proposed local descriptors have demonstrated their competitiveness with known descriptors.

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

УДК 681.32, 004.056

В.П. Клыбик, С.С. Заливако, А.А. Иванюк

МЕТОД УВЕЛИЧЕНИЯ СТАБИЛЬНОСТИ ФИЗИЧЕСКИ НЕКЛОНИРУЕМОЙ ФУНКЦИИ ТИПА «АРБИТР»

Предлагается метод повышения стабильности физически неклонируемой функции типа «арбитр» без увеличения затрат на аппаратное обеспечение и значительного роста времени получения ответа. Предлагается развернутая параметрическая модель формирования временной разницы тестового сигнала на входах арбитра. Проводится проверка метода на реальных устройствах программируемой логики.

Введение

Повсеместное использование в современных цифровых устройствах программируемых интегральных логических схем (ПЛИС), а также заказных специализированных сверхбольших интегральных схем (СБИС) позволило создать мощные и гибкие решения практически для всех сфер промышленности, быта, медицины, научной и военной отраслей. Увеличение количества таких устройств обусловило актуальность задачи их надежной идентификации. Некоторые способы идентификации цифровых устройств описаны в литературе [1–8]. Одним из перспективных и активно развивающихся способов является использование физически неклонируемых функций (ФНФ) для идентификации цифровых устройств.

По определению, данному в работе [9], физически неклонируемой функцией (от англ. Physical Unclonable Function, *PUF*) является характеристика физической (цифровой) системы, которая не поддается клонированию (копированию, воспроизведению) на других системах. Цифровые системы состоят из компонентов, физические параметры которых на стадии производства принимают случайные, принципиально неуправляемые значения. Наличие случайных параметров делает каждую цифровую систему уникальной и физически неклонируемой. Извлечение уникальных параметров из цифровых систем лежит в основе схемных реализаций ФНФ.

Формально ФНФ описывается значениями пар входных и соответствующих им выходных параметров, которые для цифровых ФНФ являются значениями входных сигналов запроса C(Challenge) и выходных сигналов ответа R (Response). Сама же ФНФ представляет собой функцию преобразования множества запросов C в множество ответов R:

$$R = PUF(C). \quad (1)$$

Существует множество видов ФНФ, детально описанных в статьях [10–16]. Для использования в целях идентификации важным свойством ФНФ является стабильность ответа ФНФ на многократно повторяющийся запрос при одних и тех же условиях. По этим параметрам перспективной для идентификации является ФНФ типа «арбитр» (АФНФ) [10].

Схемная реализация АФНФ состоит из генератора тестового импульса PG и множества N блоков коммутации сигнала (slice), последовательно соединенных в единую цепь и оканчивающихся арбитром (arbiter). В зависимости от управляющего сигнала входной шины запроса $C = \{CH_0, \dots, CH_{N-1}\}$ разрядности N, где CH_i – значение i-го разряда запроса, каждое звено коммутирует два тестовых импульса в прямом или перекрестном направлении (рис. 1).

В зависимости от разности времени прихода двух тестовых импульсов на входы арбитра сигнал на выходе R принимает значение логического нуля или единицы. Разность во времени обусловлена неуправляемыми девиациями производственного процесса при изготовлении интегральной схемы.

Звенья цепи могут быть реализованы на базе мультиплексоров или тристабильных буферных элементов [17]. Второй вариант позволяет достичь меньших аппаратурных затрат, но применим только в заказных СБИС.

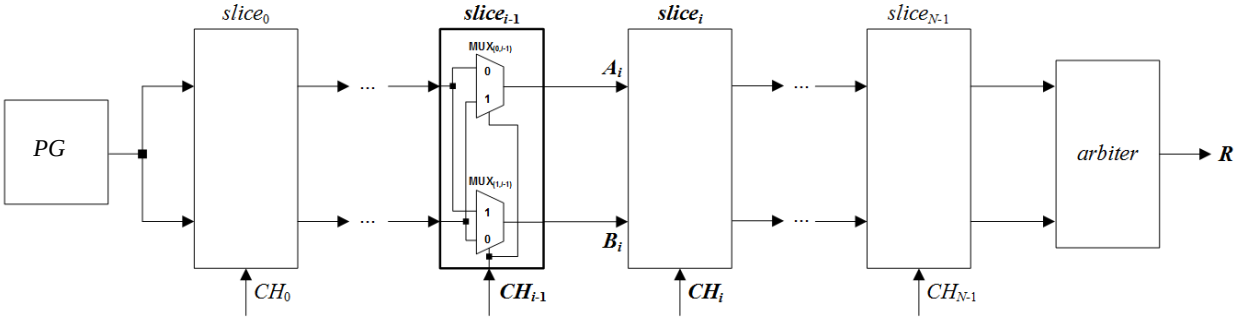


Рис. 1. Структурная схема ФНФ типа «арбитр»

Арбитр может быть реализован на базе как единичных триггеров типа *D* [10], *RS* [18], так и множественных (мультиарбитр) [19]. Одним из вариантов сложного арбитра является составная схема из четырех арбитров [20].

1. Способы стабилизации АФНФ

Варианты реализации арбитра на базе синхронного *D*-триггера или асинхронного *RS*-триггера для стабильной работы требуют наличия определенной разницы во времени прихода фронтов двух тестовых импульсов на входы арбитра. Если же такая разница меньше определенного порогового значения, зависящего от типа и технологии изготовления арбитра, то арбитр оказывается в метастабильном состоянии и значение на его выходе не всегда будет предсказуемым.

Таким образом, возможны ситуации, когда значение на выходе арбитра будет различным для одного и того же запроса при нескольких последовательных экспериментах, другими словами, ответ будет являться нестабильным или ошибочным.

Стабильность ФНФ может быть вычислена с помощью коэффициента ошибок, который определяется как среднее число ошибок (различий), приходящееся на каждый бит ответа ФНФ, полученный в серии экспериментов.

Пусть для проверки стабильности проводится E экспериментов, в результате которых получено n_0 логических нулей ($R = 0$), n_1 логических единиц ($R = 1$), а также n_X метастабильных состояний ($R = X$), под которыми понимается неопределенное значение ответа в терминах значений объектов типа std_logic языка VHDL [21]. Ненулевые значения n_X возможны только в случае реализации арбитра с помощью асинхронного *RS*-триггера.

Обозначим частоту встречаемости ответов 0, 1 и X в E экспериментах как F_0, F_1 и F_X соответственно. Эта величина может быть вычислена следующим образом:

$$F_\alpha = \frac{n_\alpha}{E}, \alpha \in \{0, 1, \mathrm{X}\}. \quad (2)$$

Эталонным ответом R_{ideal} АФНФ на некоторый запрос CH назовем наиболее часто встречающийся в E экспериментах ответ, который может быть вычислен по мажоритарному принципу:

$$R_{ideal} = \begin{cases} 0, \text{если } \max(F_0, F_1, F_X) = F_0; \\ 1, \text{если } \max(F_0, F_1, F_X) = F_1; \\ X, \text{если } \max(F_0, F_1, F_X) = F_X. \end{cases} \quad (3)$$

Обозначим через R_e ответ на запрос CH, полученный в результате эксперимента с индексом $e = 1, 2, \ldots, E$. Тогда стабильность АФНФ $S(CH)$ на запрос CH может быть вычислена как

$$S(CH) = 1 - BER(CH) = 1 - \frac{1}{E}\sum_{e=1}^{E} HD(R_{ideal}, R_e), \quad (4)$$

где *HD* – расстояние Хэмминга, методика расчета которого с учетом метастабильных состояний была представлена в работе [18].

Для ответов, полученных на $K \in [1, 2^N]$ различных запросах CH^{Ω_i} (где Ω_i – десятичное представление *N*-битного двоичного числа $CH = \{CH_{N-1}, \ldots, CH_0\}$, i = 1, 2, …, K), показатель средней стабильности может быть рассчитан с помощью соотношения

$$S_{avg} = \frac{1}{K}\sum_{i=1}^{K} S(CH^{\Omega_i}), \quad (5)$$

а показатель минимальной стабильности – с помощью соотношения

$$S_{min} = \min(S(CH^{\Omega_1}), S(CH^{\Omega_2}), \ldots, S(CH^{\Omega_K})). \quad (6)$$

В работе [18] показано, что для АФНФ с арбитром, реализованным на основе синхронного *D*-триггера и с количеством звеньев N = 128, значения стабильности S_{avg} = 0,5769 и S_{min} = 0,5648.

Существуют специальные подходы для стабилизации ответов АФНФ:

– использование дополнительных корректирующих кодов, получаемых в процессе предварительного обучения системы [22];

– применение детекторов метастабильного состояния арбитра на асинхронном *RS*-триггере [18];

– применение множественного арбитра с комбинаторным анализом выходного алфавита [19];

– мажоритарный анализ ответа для большого числа тестовых импульсов [23].

Все перечисленные подходы требуют либо дополнительных аппаратурных затрат, либо временных издержек на получение стабильного ответа.

В настоящей работе предлагается метод стабилизации АФНФ путем анализа четырех пар «запрос – ответ».

2. Параметрическая модель АФНФ

Для выявления особенностей влияния формирования запросов на ответы АФНФ была разработана детализированная параметрическая модель. Структурный блок *PG* в предлагаемой модели позволяет генерировать тестовый импульс с передним фронтом.

Одно звено схемной реализации АФНФ может быть описано структурным блоком, который имеет следующие входные и выходные порты: A_i, B_i – входные порты для двух копий одного сигнала; C_i, D_i – выходные порты, которые коммутируются с входными портами в зависимости от значения сигнала запроса на входном порту CH_i, $i = 0, \ldots, N-1$. Условимся, что при $CH_i = 0$ происходит коммутация входного порта A_i с выходным портом C_i и входного порта B_i – с выходным портом D_i. В случае когда $CH_i = 1$, происходит перекрестная коммутация входного порта A_i с выходным портом D_i и порта B_i с портом C_i (рис. 2).

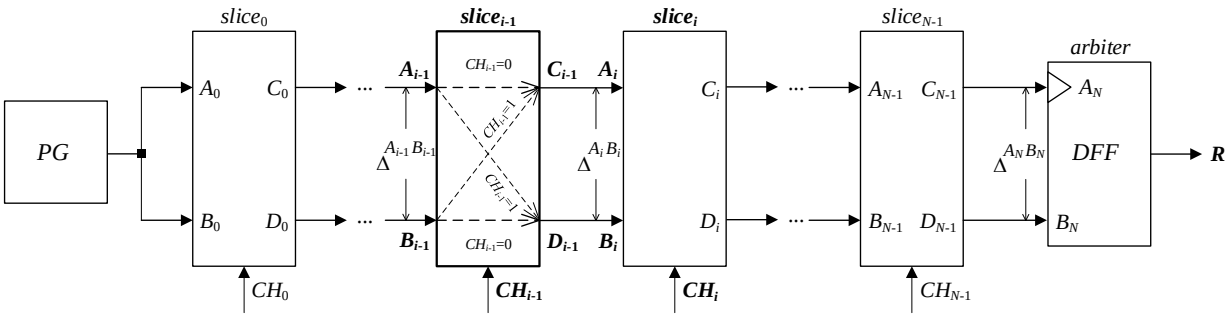


Рис. 2. Обобщенная структура ФНФ типа «арбитр»

Для рассматриваемой модели основными исследуемыми параметрами являются задержки распространения сигналов от входных портов A_i, B_i к выходным портам C_i, D_i в зависимости от значения сигнала на входе CH_i. В связи с этим введем следующее обозначение временной разницы между моментами t^p и t^g наступления фронтов сигналов на портах p и g: $\Delta^{pg} = t^p - t^g$, $p, g \in \{A_i, B_i, C_i, D_i\}$. Например, $\Delta^{A_iB_i} = t^{A_i} - t^{B_i}$ обозначает временную разницу между фронтами сигналов на входных портах A_i и B_i, наступивших в моменты t^{A_i} и t^{B_i} соответственно. Задержка распространения фронта сигнала от входного порта B_i до выходного порта C_i будет иметь обозначение $\Delta^{C_iB_i} = t^{C_i} - t^{B_i}$.

Очевидно, что $\Delta^{pg} = 0, \forall\, p = g$ и $\Delta^{pg} = -\Delta^{gp}, \forall\, p \neq g$.

Согласно определению ФНФ множества значений $\{\Delta^{C_{i-1}A_{i-1}}, \Delta^{D_{i-1}A_{i-1}}, \Delta^{C_{i-1}B_{i-1}}, \Delta^{D_{i-1}B_{i-1}}\}$ и $\{\Delta^{A_iC_{i-1}}, \Delta^{B_iD_{i-1}}\}$, $\forall\, i = 1, \ldots, N$, являются уникальными и неповторяющимися не только для схемной реализации всех звеньев одной схемы на одном полупроводниковом кристалле, но и на множестве кристаллов.

В зависимости от подаваемого значения CH_{i-1} формируются два уникальных маршрута прохождения двух тестовых импульсов от входных портов A_{i-1}, B_{i-1} до портов следующей компоненты A_i, B_i.

При использовании синхронного триггера в качестве схемы арбитра необходимо учитывать его следующие временные параметры: T_S (от англ. setup time) – время предустановки, в течение которого сигнал на входе данных B_N триггера должен оставаться стабильным перед приходом фронта сигнала синхронизации на соответствующий вход A_N; T_H (от англ. hold time) – время удержания, в течение которого сигнал на входе данных триггера должен оставаться стабильным после прихода фронта сигнала синхронизации.

Учет перечисленных параметров для схемы арбитра гарантирует появление стабильного значения ответа на выходе R. В противном случае схема арбитра может оказаться в неопределенном (метастабильном) состоянии, при котором значение ответа на выходе R будет непредсказуемым. В итоге значение ответа на выходе R зависит от результирующей разницы между фронтами сигналов $\Delta^{A_NB_N}$:

$$R = \begin{cases} 0, & \text{если } (-\Delta^{A_NB_N}) \geq T_H; \\ 1, & \text{если } \Delta^{A_NB_N} \geq T_S; \\ X, & \text{если } \Delta^{A_NB_N} < T_S \text{ и } (-\Delta^{A_NB_N}) < T_H. \end{cases} \tag{7}$$

Значение результирующей разницы $\Delta^{A_iB_i}$ формально можно выразить следующей функцией Υ от двух аргументов:

$$\Delta^{A_iB_i} = \Upsilon(\delta_{i-1}^{CH_{i-1}}, \Delta^{A_{i-1}B_{i-1}}), \tag{8}$$

где $\delta_{i-1}^{CH_{i-1}}$ – уникальная характеристика звена $slice_{i-1}$, значение которой зависит от бита запроса CH_{i-1}; $\Delta^{A_{i-1}B_{i-1}}$ – временная разница фронтов сигналов на входе звена $slice_{i-1}$.

Предположим, что $CH_{i-1} = 0$. Тогда значение $\Delta^{A_iB_i}$ можно выразить следующим образом: $\Delta^{A_iB_i} = (t^{A_i} - t^{A_{i-1}}) - (t^{B_i} - t^{B_{i-1}}) + (t^{A_{i-1}} - t^{B_{i-1}}) = \Delta^{A_iA_{i-1}} - \Delta^{B_iB_{i-1}} + \Delta^{A_{i-1}B_{i-1}}$.

Для $CH_{i-1} = 1$ предыдущее выражение принимает вид $\Delta^{A_iB_i} = (t^{A_i} - t^{B_{i-1}}) - (t^{B_i} - t^{A_{i-1}}) + (t^{B_{i-1}} - t^{A_{i-1}}) = \Delta^{A_iB_{i-1}} - \Delta^{B_iA_{i-1}} - \Delta^{A_{i-1}B_{i-1}}$.

Обозначим $\delta_{i-1}^0 = \Delta^{A_iA_{i-1}} - \Delta^{B_iB_{i-1}}$ и $\delta_{i-1}^1 = \Delta^{A_iB_{i-1}} - \Delta^{B_iA_{i-1}}$. Учитывая, что задержки распространения сигналов внутри каждого звена крайне малы и соизмеримы, можно условиться, что $\delta_{i-1}^0 = -\delta_{i-1}^1 = \delta_{i-1}$, при этом $\delta_{i-1}^1 = -\delta_{i-1}^0 = -\delta_{i-1}$, где δ_{i-1} – уникальная характеристика звена $slice_{i-1}$.

Тогда выражение (8) представим в ином виде:

$$\begin{aligned} \Delta^{A_iB_i} &= (1 - CH_{i-1}) \cdot (\delta_{i-1} + \Delta^{A_{i-1}B_{i-1}}) + CH_{i-1} \cdot (-\delta_{i-1} - \Delta^{A_{i-1}B_{i-1}}) = \\ &= (1 - 2CH_{i-1}) \cdot (\delta_{i-1} + \Delta^{A_{i-1}B_{i-1}}). \end{aligned} \tag{9}$$

Таким образом, двоичное значение CH_{i-1} определяет выбор знака для результирующей суммы $(\delta_{i-1} + \Delta^{A_{i-1}B_{i-1}})$.

Для упрощения дальнейших рассуждений введем дополнительную функцию арифметического знака:

$$Sign_{i-1} = 1 - 2CH_{i-1}. \tag{10}$$

Тогда будет справедливо следующее: при $CH_{i-1} = 0$ $Sign_{i-1} = 1$, при $CH_{i-1} = 1$ $Sign_{i-1} = -1$. При этом выражение (9), зависящее от значения CH_{i-1}, принимает вид

$$\Delta^{A_iB_i}(CH_{i-1}) = Sign_{i-1} \cdot \delta_{i-1} + Sign_{i-1} \cdot \Delta^{A_{i-1}B_{i-1}}. \tag{11}$$

Предположим, что в силу отсутствия каких-либо элементов после узла монтажного соединения (см. рис. 2), на котором происходит разделение исходного тестового импульса на два, параметр $\Delta^{A_0B_0} = 0$. Тогда выражение (11) будет определять значение $\Delta^{A_1B_1}$ исключительно исходя из уникальных для $slice_0$ параметров и значения бита запроса CH_0:

$$\Delta^{A_1B_1}(CH_0) = Sign_0 \cdot \delta_0. \tag{12}$$

Выразим значение $\Delta^{A_2B_2}$ исходя из (11) и (12):

$$\Delta^{A_2B_2}(CH_1, CH_0) = Sign_1 \cdot \delta_1 + Sign_1 \cdot \Delta^{A_1B_1} = Sign_1 \cdot \delta_1 + Sign_1 \cdot Sign_0 \cdot \delta_0. \tag{13}$$

Из (13) видно, что $\Delta^{A_2B_2}$ зависит не только от значения CH_1, но и от значения предыдущего разряда запроса CH_0. В связи с этим левую часть равенства (13) можно представить как $\Delta^{A_2B_2}(CH_1, CH_0)$.

Вычислим значение

$$\Delta^{A_3B_3}(CH_2, CH_1, CH_0) = Sign_2 \cdot \delta_2 + Sign_2 \cdot Sign_1 \cdot \delta_1 + Sign_2 \cdot Sign_1 \cdot Sign_0 \cdot \delta_0. \tag{14}$$

Обобщая, можно записать выражение

$$\begin{gathered}\Delta^{A_NB_N}(CH_{N-1}, \dots, CH_0) = \Delta^{A_NB_N}(CH^{\Omega}) = \\ = Sign_{N-1} \cdot \delta_{N-1} + Sign_{N-1} \cdot Sign_{N-2} \cdot \delta_{N-2} + \cdots + Sign_{N-1} \cdot \dots \cdot Sign_0 \cdot \delta_0.\end{gathered} \tag{15}$$

Например, для $CH^{\Omega} = \{CH_{N-1}, CH_{N-2}, CH_{N-3,} \dots, CH_0\} = \{0,\ 0,\ 0, \dots,\ 0\} = CH^0$ предыдущее выражение (15) можно представить как

$$\Delta^{A_NB_N}(CH^0) = \delta_{N-1} + \delta_{N-2} + \cdots + \delta_0 = \sum_{i=0}^{N-1} \delta_i. \tag{16}$$

Рассмотрим еще несколько примеров для различных значений запросов. Предположим, что запрос принимает значение CH^1, т. е. $CH_0 = 1$, а $CH_{\text{i}} = 0, \forall\, i = 1, \dots, N-1$:

$$\Delta^{A_NB_N}(CH^1) = \delta_{N-1} + \delta_{N-2} + \cdots - \delta_0 = \sum_{i=0}^{N-1} \delta_i - 2 \cdot \delta_0 = \Delta^{A_NB_N}(CH^0) - 2 \cdot \delta_0. \tag{17}$$

Допустим, что для CH^0 схема АФНФ вырабатывает стабильное значение ответа на выходе R (см. выражение (7)). С учетом того что значение $2 \cdot \delta_0$ крайне мало в сравнении со значением $\Delta^{A_NB_N}(CH^0)$, вероятность изменения значения ответа для CH^1 также крайне мала.

Пусть $CH^{2^{N-1}} = \{1,\ 0,\ 0, \dots,\ 0\}$, тогда

$$\Delta^{A_NB_N}\left(CH^{2^{N-1}}\right) = -\delta_{N-1} - \delta_{N-2} - \cdots - \delta_0 = -\sum_{i=0}^{N-1} \delta_i = -\Delta^{A_NB_N}(CH^0), \tag{18}$$

что означает высокую вероятность инверсии стабильного ответа R, полученного при запросе CH^0.

Рассмотрим запросы $CH^{\Omega}, CH^{\Omega'}, CH'^{\Omega}, CH'^{\Omega'}$, для которых выполняются равенства $|\Omega - \Omega'| = 1,\ |\Omega - '\Omega| = 2^{N-1},\ |\Omega - '\Omega| = 2^{N-1},\ |'\Omega' - '\Omega| = 1,\ |'\Omega' - \Omega'| = 2^{N-1}$. Выполнение указанных равенств означает, что относительно запроса CH^{Ω} для запроса $CH^{\Omega'}$ будет инвертирован младший бит запроса $LSB = CH_0$, для запроса CH'^{Ω} – старший бит запроса $MSB = CHN-1$, а для запроса $CH'\Omega'$ будут инвертированы и младший, и старший биты *LSB* и *MSB*.

Рассмотрим изменения значения $\Delta^{A_N B_N}(CH^{\Omega})$ при инверсии *LSB* и *MSB* согласно формуле (15).

При инверсии *LSB* изменяется значение запроса CH^{Ω} на $CH^{\Omega'}$ ($|\Omega - \Omega'| = 1$) и происходит изменение знака компоненты δ_0 на противоположный согласно (17).

Оценим следующую разницу:

$$\begin{gathered}\Delta(\overline{LSB}) = |\Delta^{A_N B_N}\left(CH^{\Omega}\right) - \Delta^{A_N B_N}\left(CH^{\Omega'}\right)| = \\ = |Sign_{N-1} \cdot \ldots \cdot Sign_1 \cdot \delta_0 \cdot \left(Sign_0 - (-Sign_0)\right)| = |2 \cdot \delta_0|\,.\end{gathered} \tag{19}$$

Если для произвольного запроса CH^{Ω} верно неравенство $\Omega < 2^{N-1}$, то $MSB = 0$ и $Sign_{N-1} = 1$ согласно (15), тогда

$$\Delta^{A_N B_N}\left(CH^{\Omega}\right) = \delta_{N-1} + Sign_{N-2} \cdot \delta_{N-2} + \cdots + Sign_{N-2} \cdot \ldots \cdot Sign_0 \cdot \delta_0. \tag{20}$$

Для запроса CH'^{Ω}, где $'\Omega = \Omega + 2^{N-1}$, будут верны равенства $MSB = 1$ и $Sign_{N-1} = -1$, при этом

$$\begin{gathered}\Delta^{A_N B_N}\left(CH'^{\Omega}\right) = -(\delta_{N-1} + Sign_{N-2} \cdot \delta_{N-2} + \cdots + \\ + Sign_{N-2} \cdot \ldots \cdot Sign_0 \cdot \delta_0).\end{gathered} \tag{21}$$

Таким образом,

$$\Delta^{A_N B_N}\left(CH^{\Omega}\right) = -\Delta^{A_N B_N}\left(CH'^{\Omega}\right). \tag{22}$$

Оценим модуль разности

$$|\Delta^{A_N B_N}\left(CH^{\Omega}\right) - \Delta^{A_N B_N}\left(CH'^{\Omega}\right)| = 2 \cdot |\Delta^{A_N B_N}\left(CH^{\Omega}\right)|\,. \tag{23}$$

Из формул (19)–(23) видно, что инверсия *LSB* для произвольного запроса способна внести относительно малые изменения в общее значение задержки сигналов, а инверсия *MSB* приводит к инверсии знака результата. При этом для произвольной длины путей присутствует отличная от нуля вероятность существования запросов, которые сформируют результирующую задержку $\Delta^{A_N B_N} \approx 0$ ввиду различных значений и знаков задержек на каждом звене. Последнее означает, что всегда существует подмножество запросов из множества всех возможных, генерирующее нестабильные ответы арбитра $R = X$, тогда как для получения стабильных ответов арбитра должны выполняться следующие условия:

$$R = \begin{cases} 0, \text{если } -\Delta^{A_N B_N} \gg T_H; \\ 1, \text{если } \Delta^{A_N B_N} \gg T_S. \end{cases} \tag{24}$$

Для идеального арбитра верно выполнение равенства $|T_H| = |T_S|, T_S \to 0$.

Как следствие инверсия *LSB* для произвольного запроса имеет малую вероятность инверсии ответа идеального арбитра, а инверсия *MSB* – высокую вероятность инверсии ответа идеального арбитра.

Если инверсия *MSB* произвольного запроса не привела к инверсии ответа, то можно утверждать, что $\Delta^{A_N B_N}$ для этого запроса находится в зоне метастабильности арбитра. Для реального арбитра отмечается сдвиг зоны метастабильности относительно нуля по причине наличия асимметрии при технологической реализации арбитра и выполнение неравенства $|T_H| \neq \neq |T_S|,\ T_H \neq 0, T_S \neq 0$. Например, для параметрической модели арбитра, реализованной на

FPGA Xilinx Artix-7, параметры предустановки и удержания принимают значения $T_H = -0{,}103$ нс, $T_S = 0{,}297$ нс.

Рассмотрим последовательность ответов $\{R_0, R_1, R_2, R_3\}$, где $R_0 = PUF(CH^{\Omega})$, $R_1 = PUF(CH^{\Omega'})$, $R_2 = PUF(CH'^{\Omega})$, $R_3 = PUF(CH'^{\Omega'})$.

Введем понятие сильных запросов, для которых верно $R_0 = R_1 = \overline{R_2} = \overline{R_3}$. В результате для них могут быть получены только два варианта последовательности ответов: {1,1,0,0} либо {0,0,1,1}. Все остальные запросы, сформированные по описанному выше принципу и дающие иную последовательность ответов, будем называть слабыми.

Анализируя последовательность из четырех ответов, полученных в результате подачи четырех запросов, которые сформированы описанным способом, можно выявить сильные и слабые запросы и для них с высокой вероятностью прогнозировать стабильность ответа АФНФ на любой из поданных запросов. Только две последовательности ответов {1,1,0,0} и {0,0,1,1} являются сигнатурами сильных запросов и предполагают стабильность ответа, поэтому для формирования идентификаторов цифрового устройства необходимо использовать только ответы на сильные запросы.

3. Исследование параметрической модели АФНФ

Для проверки разработанной математической модели было создано VHDL-описание схемной реализации АФНФ из $N = 10$ звеньев. Проект VHDL был реализован в виде параметрической post place&route-модели для кристалла FPGA Xilinx Artix-7. Также было создано тестовое окружение, позволяющее автоматизировать процессы формирования последовательности запросов, анализа ответа арбитра R и числового значения задержки $\Delta^{A_N B_N}$. В ходе поставленного эксперимента было сгенерировано множество всех $2^N = 1024$ запросов.

Рассмотрим график отсортированных по возрастанию значений $\Delta^{A_N B_N}$ R и значения ответа R, включая неопределенные значения X, соответствующие метастабильному состоянию арбитра, в зависимости от подаваемых запросов (рис. 3).

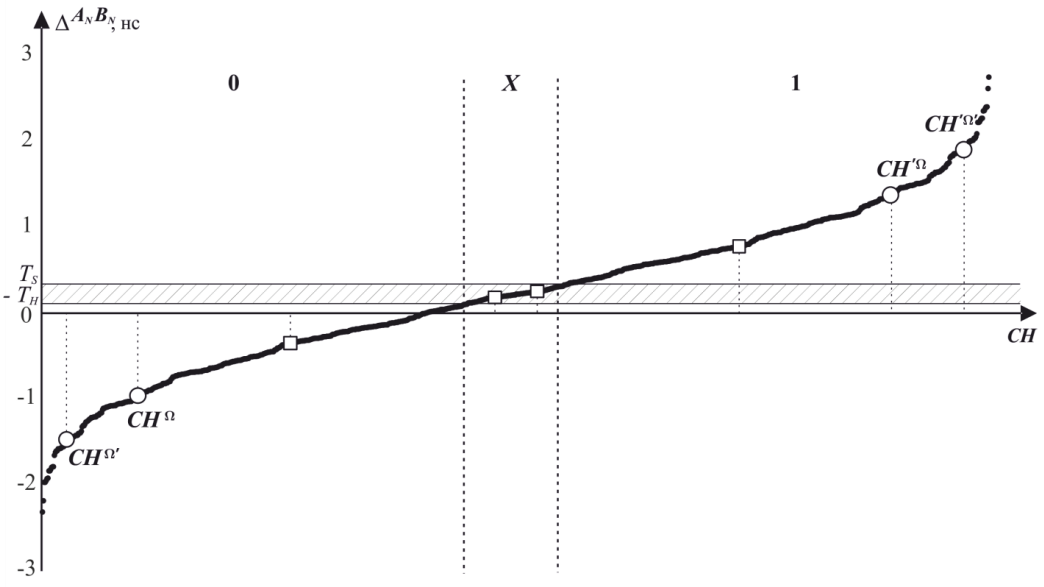


Рис. 3. Расположение групп сильных и слабых запросов

Как видно из представленного графика, зависимость $\Delta^{A_N B_N}$ от значения подаваемого запроса является нелинейной, что обусловлено наличием на звеньях задержек δ_i с разными значениями и разными знаками. Помимо этого из графика видно, что зона метастабильности реализованного арбитра смещена в сторону положительных значений относительно идеального арбитра.

На рис. 3 показана также группа слабых запросов, обозначенных символом □, и группа сильных запросов, обозначенных символом ○. Группа слабых запросов {26, 27, 538, 539} формирует на арбитре задержки {0,182, –0,346, 0,252, 0,78} нс и дает множество ответов арбитра {X, 0, X, 1}. Группа сильных запросов {56, 57, 568, 569} формирует на арбитре задержки {–0,945, –1,473, 1,379, 1,907} нс и дает множество ответов арбитра {0, 0, 1, 1}.

Разделим ось абсцисс на три региона в зависимости от ответов арбитра на соответствующий запрос. Видно, что слабые запросы расположены на графике ближе друг к другу, чем сильные, а также лежат вблизи региона метастабильности арбитра или внутри него.

Для обозначения границ метастабильного региона были выбраны значения $-T_H$ и T_S, соответствующие параметрам арбитра. Действительно, в соответствии с формулой (7) арбитр попадает в метастабильное состояние при условии $\Delta^{A_N B_N} < T_S$ и $(-\Delta^{A_N B_N}) < T_H$, которое эквивалентно выражению $-T_H < \Delta^{A_N B_N} < T_S$.

Анализ экспериментальных данных, полученных на параметрической модели АФНФ, показал преобладание сильных групп запросов: 67 % сильных и 33 % слабых.

4. Результаты экспериментов

4.1. Реализация экспериментальной установки

Для проведения натурных экспериментов была выбрана модель ПЛИС, отличная от той, на которой проводилось параметрическое моделирование. Такое решение позволяет показать, что разработанная математическая модель АФНФ может быть реализована на произвольной ПЛИС, поэтому в качестве платформы для реализации была выбрана система на кристалле ZC706 (СнК), входящая в состав платы быстрого прототипирования Xilinx Zynq-7000 [24].

Для экспериментальной проверки предлагаемого метода была реализована АФНФ с N = 130 звеньями и арбитром на базе синхронного *D*-триггера, а также асинхронного *RS*-триггера (рис. 4).

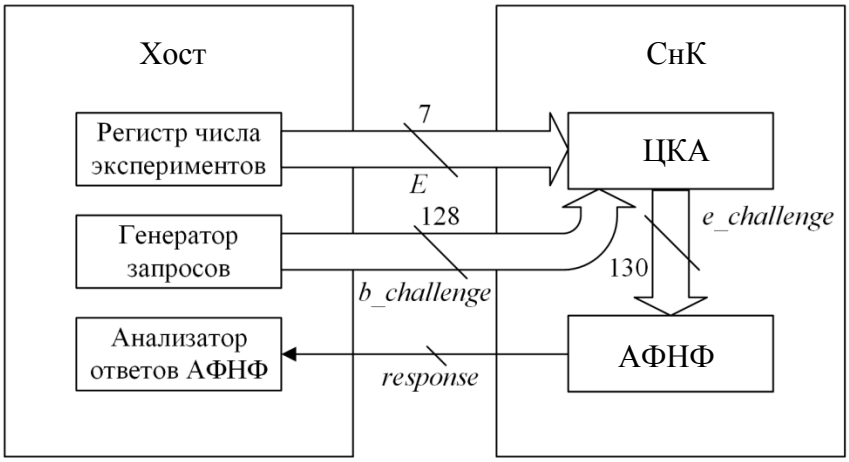


Рис. 4. Структурная схема реализованного устройства

С помощью пакета Xilinx SDK было разработано программное обеспечение на языке C для взаимодействия с АФНФ, реализованной в среде Xilinx Vivado 2014.4.1 на языке VHDL. На стороне хоста был реализован генератор псевдослучайной М-последовательности для получения 128-битных слабо коррелированных запросов, а также возможность с помощью регистра задать количество экспериментов E (от 1 до 128) по получению ответа ФНФ на фиксированный запрос. Базовый запрос (*b_challenge*) подавался на вход цифрового конечного автомата (ЦКА), который, в свою очередь, осуществлял запуск E экспериментов по получению ответов вида $\{R_0, R_1, R_2, R_3\}$.

Таким образом, экспериментальная установка позволила произвести тестирование фиксированной аппаратной конфигурации АФНФ на произвольном числе запросов.

4.2. Распределение конфигураций ответов

Для тестирования АФНФ с арбитром на базе *D*-триггера был проведен эксперимент по генерированию $K = 10\,000$ запросов, повторенных $E = 100$ раз, по описанному выше принципу. Распределение конфигураций ответов на сильные и слабые запросы показано на рис. 5.

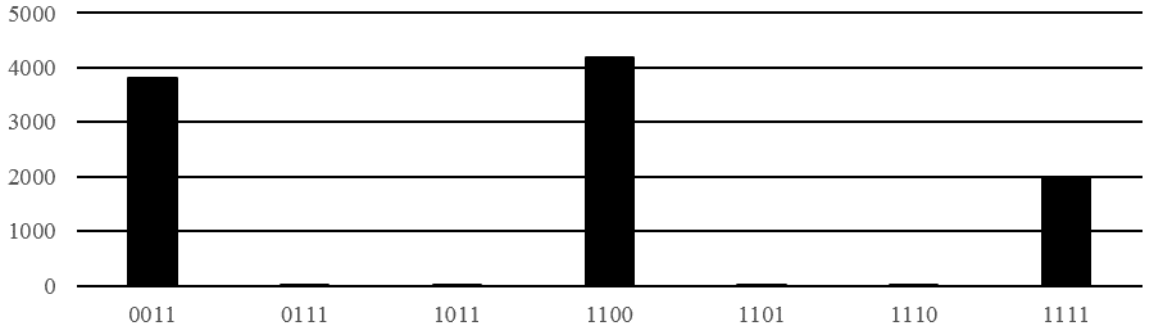


Рис. 5. Распределение конфигураций ответов на сильные и слабые запросы для АФНФ с арбитром на базе *D*-триггера

Из диаграммы видно, что ответы на сильные запросы ({0, 0, 1, 1} и {1, 1, 0, 0}) составляют порядка 80 % от всех сгенерированных. Наличие ответов вида {1, 1, 1, 1} свидетельствует о том, что регион метастабильности занимает часть региона логической единицы. Следовательно, наблюдается преобладание значений ответов {1, 1, 0, 0} над значениями {0, 0, 1, 1}.

Для проверки стабильности генерируемых ответов был реализован арбитр на базе асинхронного *RS*-триггера, с помощью которого было также сгенерировано $K = 10\,000$ запросов, повторенных $E = 100$ раз. Распределение конфигураций ответов на эти запросы показано на рис. 6.

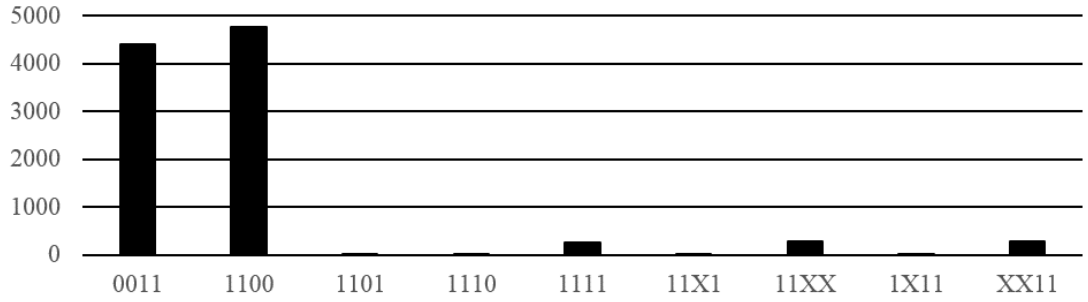


Рис. 6. Распределение конфигураций ответов на сильные и слабые запросы для АФНФ с арбитром на базе *RS*-защелки

Подобная реализация АФНФ показала несколько другое распределение запросов: 90 % сильных и 10 % слабых в силу способности арбитра, реализованного на базе асинхронного *RS*-триггера, обнаруживать метастабильное состояние. Также множество возможных ответов имеет бóльшую мощность, поскольку в него входят другие конфигурации, один или несколько ответов которых содержат метастабильное состояние *X*.

Различия в распределении сильных и слабых запросов между реализацией на ПЛИС и параметрическим моделированием объясняются принципиальной невозможностью моделирования реальных значений временных задержек и параметров арбитра. Тем не менее результаты экспериментов подтверждают тенденцию преобладания сильных запросов над слабыми.

Показатели стабильности для всех возможных ответов (S_{avg} и S_{min}), а также оценка вероятности получения стабильного ответа (P_{stable}) приведены в таблице.

Стабильность конфигураций ответов АФНФ с арбитром на базе асинхронного *RS*-триггера

$\{R_0, R_1, R_2, R_3\}$	S_{avg}	S_{min}	P_{stable}
{1, 1, 0, 0}	**0,999**	**0,758**	**0,991**
{0, 0, 1, 1}	**0,999**	**0,768**	**0,993**
$\{1, 1, X, X\}$	0,972	0,753	0,651
$\{X, X, 1, 1\}$	0,976	0,775	0,669
$\{1, 1, 1, 1\}$	0,944	0,758	0,279
$\{0, 1, 1, 1\}$	0,778	0,778	0,000
$\{1, 1, 0, 1\}$	0,740	0,733	0,000
$\{1, 1, X, 1\}$	0,774	0,763	0,000
$\{1, 1, 1, 0\}$	0,753	0,753	0,000
$\{1, 1, 1, X\}$	0,776	0,735	0,000
$\{X, 1, 1, 1\}$	0,766	0,755	0,000
$\{1, X, 1, 1\}$	0,741	0,718	0,000

Примечание: выделены конфигурации ответов, имеющие хорошую среднюю стабильность.

Понятие вероятности появления стабильного ответа P_{stable} было введено для сравнения качества ответов АФНФ, полученных с помощью предлагаемого метода и без него. Численное значение этого показателя может быть оценено с помощью выражения

$$P_{stable} = \frac{K_{stable}}{K}, \tag{25}$$

где K_{stable} – количество ответов, стабильность которых $S(CH) = 1$.

В таблице показано, что конфигурации ответов {1, 1, 0, 0} и {0, 0, 1, 1} имеют хорошую среднюю стабильность, однако минимальная находится на таком же уровне, как и при слабых запросах. Этот эффект объясняется тем, что если ответ АФНФ является нестабильным, то показатель его стабильности примерно одинаков (≈0,75) как для слабых, так и для сильных запросов, однако вероятность встретить данный ответ для сильных запросов незначительна (менее 0,01).

Таким образом, поскольку количество сильных запросов достаточно велико (9161 из 10 000), то оценка вероятности может быть признана состоятельной и такое же соотношение стабильных ответов к нестабильным будет наблюдаться и при бóльшем числе запросов.

4.3. Генерирование идентификатора

Для генерирования идентификатора был разработан следующий алгоритм:

1. Тестирование запроса.

1.1. Выдвигается гипотеза о сильном запросе: однократно подаются четыре запроса вида CH^{Ω}, CH'^{Ω}, $CH^{\Omega'}$, $CH'^{\Omega'}$ к ФНФ и считываются ответы на них $\{R_0, R_1, R_2, R_3\}$.

1.2. Из всех сгенерированных конфигураций ответов выбираются только те, которые имеют вид {0, 0, 1, 1} или {1, 1, 0, 0}, и соответствующие им запросы помечаются как сильные.

1.3. Остальные запросы помечаются как слабые.

2. Генерирование бита идентификатора.

2.1. Из сильных запросов выбирается такие, которые ранее не были использованы для генерирования идентификатора (в последовательности, определенной генератором запросов).

2.2. Ответ генерируется E раз, в результате чего проверяется его стабильность.

Был проведен эксперимент по генерированию 25 (20 на основе сильных и 5 на основе слабых запросов) 128-битных идентификаторов, отобранных из 4000 запросов. В результате средняя стабильность идентификаторов, построенных на сильных запросах, лежит в интервале [0,996, 1,000], а идентификаторов на основе слабых запросов – в интервале [0,743, 0,779]. Следовательно, предлагаемый способ генерирования идентификаторов обладает гораздо более высокой средней стабильностью (0,999), чем генерирование без учета свойств запроса (0,759).

Также был проведен эксперимент по генерированию 200 идентификаторов разной длины L (L = 1, ..., 128 бит). Результаты эксперимента показаны на рис. 7 (график выполнен в логарифмическом масштабе по оси ординат и в линейном масштабе по оси абсцисс).

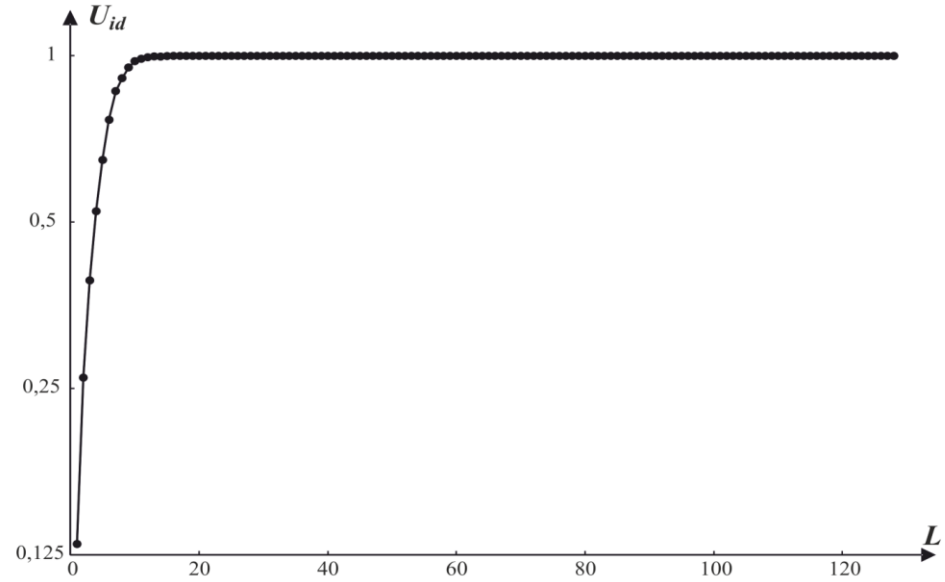


Рис. 7. Значение уникальности идентификатора U_{id} в зависимости от его длины L

Определим уникальность идентификатора как

$$U_{id} = 1 - \frac{Id_{avg}}{K_{id}}, \tag{26}$$

где Id_{avg} – среднее число идентификаторов, приходящееся на его определенное значение (например, для идентификаторов длиной 1 количество идентификаторов со значением 0 – 118, а со значением 1 – 82, следовательно, среднее значение 100); K_{id} – количество идентификаторов, которое было сгенерировано. В данном случае K_{id} = 200.

По результатам эксперимента удалось установить, что для идентификаторов длиной 16 и более все K_{id} = 200 значений являются уникальными (т. е. не повторяются), а для идентификаторов меньшей разрядности значение уникальности экспоненциально возрастает с увеличением длины. При этом расстояние Хэмминга между идентификаторами, нормированное по их длине, принимает значения в интервале [0,479, 0,495] и не зависит от разрядности идентификатора.

Таким образом, предлагаемый способ позволяет генерировать уникальные идентификаторы разрядностью 16 и более со значительно повышенной стабильностью без дополнительных аппаратурных затрат за приемлемое время.

Заключение

Разработанная математическая модель показывает, что для классической АФНФ всегда существует подмножество запросов, ответы на которые будут нестабильны. Стабильность ответа АФНФ на любой запрос можно с высокой вероятностью предсказать путем анализа ответов на три дополнительных запроса, сформированных в результате изменения *LSB* и *MSB*. Соответственно эффективное количество пар «запрос – ответ» для АФНФ из N звеньев не превышает 2^{N-2}.

Результаты параметрического моделирования на кристалле FPGA Xilinx Artix-7 показали, что множество запросов к ФНФ может быть разделено на 67 % сильных и 33 % слабых, что подтверждает выдвинутую гипотезу. Предлагаемая модель была протестирована с помощью экспериментальной установки, реализованной на плате быстрого прототипирования Xilinx Zynq-7000. Результаты экспериментов с реализацией арбитра при помощи синхронного

D-триггера показали соотношение слабых и сильных запросов 80 на 20 %, а с реализацией с помощью асинхронного *RS*-триггера – 90 на 10 % соответственно. Экспериментальные данные подтвердили гипотезу о преобладании сильных запросов над слабыми. Значения средней стабильности, а также вероятности появления стабильных ответов для сильных запросов значительно (более чем на 25 %) превышают такие же показатели, полученные для слабых запросов.

Был разработан алгоритм генерирования уникального идентификатора с помощью АФНФ, основанный на применении предложенной математической модели. Алгоритм позволяет генерировать уникальные идентификаторы разрядностью более 16 бит с повышенной стабильностью и незначительными временными издержками.

Список литературы

1. Pentium III Serial Numbers [Electronic resource]. – Mechler Enterprises, LLC., 2001. – Mode of access : http://www.pcmech.com/article/pentium-iii-serial-numbers. – Date of access : 30.05.2016.

2. Silicon physical random functions / B. Gassend [et al.] // ACM Conf. on Comp. and Comm. Security (CCS'02), N. Y., USA, November 18–22, 2002 / ACM N. Y. – N. Y., 2002. – P. 148–160.

3. Koushanfar, F. Intellectual property metering / F. Koushanfar, G. Qu, M. Potkonjak // 4th Intern. Workshop Information Hiding (IH'01), Pittsburg, USA, April 25–27, 2001. – Pittsburg, 2001. – P. 81–95.

4. Koushanfar, F. Hardware Metering / F. Koushanfar, G. Qu // Proc. IEEE Design Automation Conf. (DAC'01), Scottsdale, USA, June 18–22, 2001 / ACM N. Y. – Scottsdale, 2001. – P. 490–493.

5. Qu, G. Fingerprinting intellectual property using constraint-addition / G. Qu, M. Potkonjak // Proc. IEEE Design Automation Conf. (DAC'00), Los-Angeles, USA, June 5–9, 2000 / ACM, N. Y. – Los-Angeles, 2000. – P. 587–592.

6. A blind dynamic fingerprinting technique for sequential circuit intellectual property protection / C.-H. Chang [et al.] // IEEE Trans. Comput.-Aided Design of Integrated Circuits and Syst. – 2014. – Vol. 33, no. 1. – P. 76–89.

7. Koushanfar, F. Provably secure active IC metering techniques for piracy avoidance and digital rights management / F. Koushanfar // IEEE Trans. Inf. Forensics and Security. – 2011. – Vol. 7, no. 1. – P. 51–63.

8. Ending piracy of integrated circuits / J. Roy [et al.] // Computer. – 2010. – Vol. 43, no. 10. – P. 30–38.

9. Architecture and Design Flow for a Highly Efficient Structured ASIC / H. Man-Ho [et al.] // IEEE Transactions on VLSI Systems. – 2012. – Vol. 21, iss. 3. – P. 424 – 433.

10. A technique to build a secret key in integrated circuits for identification and authentication applications / J.W. Lee [et al.] // Intern. Symp. VLSI Circuits (VLSI'04), Honolulu, USA, June 15–19, 2004. – Honolulu, 2004. – P. 176–179.

11. Readproof hardware from protective coatings / P. Tuyls [et al.] // Cryptographic Hardware and Embedded Systems (CHES'06), Yokohama, Japan, October 10–13, 2006 / Springer, N. Y. – Yokohama, 2006. – P. 369–383.

12. Holcomb, D.E. Initial SRAM state as a fingerprint and source of true random numbers for RFID tags / D.E. Holcomb, W.P. Burleson, K. Fu // Conf. RFID Security, Malaga, Spain, July 11–13, 2007 / University of Malaga, Spain. – Malaga, 2007. – P. 1–2.

13. The butterfly PUF protecting IP on every FPGA / S.S. Kumar [et al.] // Proc. Intern. Workshop Hardware-Oriented Security and Trust (HOST'08), Anaheim, USA, June 8–10, 2008. – Anaheim, 2008. – P. 67–70.

14. Uniqueness enhancement of PUF responses based on the locations of random outputting RS latches / D. Yamamoto [et al.] // Cryptographic Hardware and Embedded Systems (CHES'11), Nara, Japan, September 28 – October 1, 2011. – Nara, 2011. – P. 390–406.

15. Ярмолик, В.Н. Физически неклонируемые функции / В.Н. Ярмолик, Ю.Г. Вашинко // Информатика. – 2011. – № 2. – С. 92–103.

16. CMOS image sensor based physical unclonable function for coherent sensor-level authentication / Y. Cao [et al.] // IEEE Trans. Circuits and Systems I: Regular Papers. – 2015. – Vol. 62, no. 11. – P. 2629–2640.

17. Ozturk, E. Physical unclonable function with tristate buffers / E. Ozturk, G. Hammouri, B. Sunar // Proc. Intern. Symp. on Circ. and Syst. (ISCAS'08), Seattle, USA, May 18–21, 2008. – Seattle, 2008. – P. 3194–3197.

18. Multi-valued arbiters for quality enhancement of PUF responses on FPGA implementation / S.S. Zalivaka [et al.] // Special Session on Cyber-Physical Systems and Security, in Proc. 21st IEEE Asia and South Pacific Design Automation Conf. (ASP-DAC 2016), Macao, China, 26–28 Jan., 2016. – Macao, 2016. – P. 533–538.

19. Клыбик, В.П. Применение физически неклонируемой функции типа арбитр для решения задачи идентификации цифровых устройств / В.П. Клыбик, А.А. Иванюк // Автоматика и вычислительная техника. – 2015. – № 3(49). – С. 24–34.

20. Иванюк, А.А. Использование физически неклонируемых функций типа «арбитр» для идентификации встроенных систем на базе ПЛИС / А.А. Иванюк // Информационные технологии и системы 2011 (ИТС 2011) : материалы Междунар. науч. конф., БГУИР, Минск, Беларусь, 26 окт. 2011 г. / редкол. : Л.Ю. Шилин [и др]. – Минск : БГУИР, 2011. – С. 270–271.

21. 1076-2008 – IEEE Standard VHDL Language Reference Manual [Electronic resource]. – IEEE., Jan. 2009. – Mode of access : http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4772740&filter=AND(p_Publication_Number:4772738). – Date of access : 30.05.2016.

22. Maes, R. PUFKY: A Fully Functional PUF-Based Cryptographic Key Generator / R. Maes, A. van Herrewege, I. Verbauwhede // Cryptographic Hardware and Embedded Systems (CHES'12), Leuven, Belgium, September 9–12, 2012. – Leuven, 2012. – P. 302–319.

23. Иванюк, А.А. Аппаратная реализация алгоритма идентификации ПЛИС на основе физически неклонируемых функций / А.А. Иванюк // Информационные технологии и системы 2013 (ИТС 2013) : материалы Междунар. науч. конф., БГУИР, Минск, Беларусь, 23 окт. 2013 г. / редкол. : Л.Ю. Шилин [и др]. – Минск : БГУИР, 2013. – С. 184–271.

24. Zynq-7000 All Programmable SoC Overview (DS190) – Xilinx [Electronic resource]. – Xilinx, Inc., 2012. – Mode of access : http://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf. – Date of access : 30.05.2016.

Поступила 23.06.2016

Белорусский государственный университет информатики и радиоэлектроники,
Минск, ул. П. Бровки, 6
e-mail: klybik@bsuir.by,
zalivako@bsuir.by,
ivaniuk@bsuir.by

V.P. Klybik, S.S. Zalivaka, A.A. Ivaniuk

RELIABILITY ENHANCEMENT METHOD FOR «ARBITER» PHYSICALLY UNCLONABLE FUNCTION

The paper presents a reliability enhancement method for an arbiter physically unclonable function (A-PUF). The proposed technique has reasonable challenge-response generation time and does not cause additional hardware overheads. A time difference of a test pulse delay has been used as a basis for A-PUF parametric model development. The proposed approach has been verified on a real programmable logic device.

УДК 50.51.17

Д.И. Черемисинов

ОТОБРАЖЕНИЕ ЛОГИЧЕСКИХ СЕТЕЙ В ЗАДАННЫЙ ТЕХНОЛОГИЧЕСКИЙ БАЗИС

Рассматривается задача синтеза многоуровневых логических сетей в базисе библиотечных элементов КМОП СБИС. Приводится описание структуры программы для решения этой задачи и форматов исходных данных. Обсуждается влияние формы исходных данных на результат решения. Программа отображения логических сетей в заданный технологический базис включена как проектная операция в программный комплекс энергосберегающего логического синтеза, предназначенного для автоматизации проектирования многоуровневых логических схем из библиотечных элементов заказных сверхбольших интегральных схем (СБИС), выполненных по КМОП-технологии.

Введение

Операция логического синтеза по заданному описанию поведения синтезируемой схемы в виде системы булевых функций строит структурное представление в виде логической сети из элементов заданной библиотеки [1]. Синтез представляет собой задачу нахождения структурной модели дискретного устройства по заданной функциональной модели.

Отображение логической сети в технологический базис (technology mapping) – это реализация комбинационной схемы с использованием элементов библиотеки, выполненных по конкретной (предоставляемой изготовителем микросхем) технологии СБИС. Библиотека, как правило, состоит из элементов различных размеров и быстродействия, реализующих примитивные булевы (AND и OR) и более сложные (исключающее ИЛИ, мультиплексор) функции. Так, исходное представление булевой функции (системы булевых функций) отображается в оптимальную схему, максимально использующую элементы библиотеки. Цель состоит в том, чтобы найти новую схему с той же функциональностью, но меньшей площади, задержки, потребляемой мощности и т. д., построенную из элементов библиотеки, а также в том, чтобы новая схема после размещения элементов и трассировки связей могла быть реализована в качестве аппаратного устройства высокого качества. Операция является неотъемлемой составляющей любого автоматизированного процесса проектирования СБИС.

Элементы пространства решений задачи технологического отображения являются некоторыми представлениями булевых функций. Установлено [2], что не существует метода построения пространства решений, содержащего все возможные представления заданной булевой функции. Все распространенные методы решения задачи синтеза используют пространство решений, построенное на основе исходного представления булевой функции. Таким образом, структура исходного представления булевой функции диктует в значительной степени структуру построенной схемы. Это и есть структурная чувствительность (structural bias [3]) инструментов синтеза, она неустранима. Ни один из известных методов синтеза не способен открыть схемы новой нестандартной структуры.

При синтезе строится многоуровневая логическая схема для данной булевой функции. Первоначально функция задается в виде таблицы истинности, ДНФ, системы уравнений или, например, в виде схемы сомнительного качества. Для нахождения лучшего решения желательно выполнять синтез несколько раз, используя различные эквивалентные формы задания исходной булевой функции [3]. Одним из способов такого синтеза является варьирование нескольких форматов задания исходной булевой функции.

В работе описывается программа технологического отображения, которая детально использует метод синтеза, основанный на структурном покрытии многоуровневой логической сети из вентилей элементами КМОП-библиотеки [1]. Эта программа отличается от известных инструментов синтеза тем, что допускает возможность применения нескольких форматов задания булевых функций. Из-за структурной чувствительности выбор формата одного из эквивалентных представлений исходной булевой функции влияет на результат синтеза. Предметом

настоящей работы являются представления булевой функции, используемые в операции технологического отображения, а не ее алгоритм. Основное внимание уделяется описанию подготовки исходных данных для программы и влиянию их формы на результат решения.

1. Структурная чувствительность технологического отображения

Элементом называется некоторое простейшее в каком-то смысле техническое устройство. Логический элемент – это такое устройство, которое описывается совокупностью двоичных физических величин, связанных функциональной зависимостью, представимой булевой функцией или системой булевых функций. Реализуемая элементом система булевых функций показывает его функциональные свойства и может рассматриваться как логическая или функциональная модель элемента.

Логической сетью или схемой называют такую модель дискретного устройства, которая отражает его внутреннее строение с точностью до функций, выполняемых элементами [2]. Сеть образуется путем соединения элементов, т. е. указываются линии связи, по которым выходные сигналы одних элементов поступают на входные полюсы других элементов. Задать логическую сеть – означает указать ее входные и выходные полюсы, структуру связей между элементами и булевы функции, реализуемые этими элементами.

Любую логическую сеть можно полностью описать в алгебраической форме системой уравнений непосредственных связей, задающих функционирование ее элементов. Все полюсы схемы обозначаются некоторыми внутренними переменными. При этом отождествляемым полюсам (соединяемым между собой) приписывается одна и та же переменная. Система уравнений непосредственных связей является формой представления исходной булевой функции в виде суперпозиции функций, заданных выражениями внутри скобок, а логическая сеть в виде иерархической структуры – гибридным структурно-поведенческим представлением (рис. 1). Алгебраическое представление в виде ДНФ, матричное и табличное задание булевой функции имеют структуру двухуровневой логической сети.

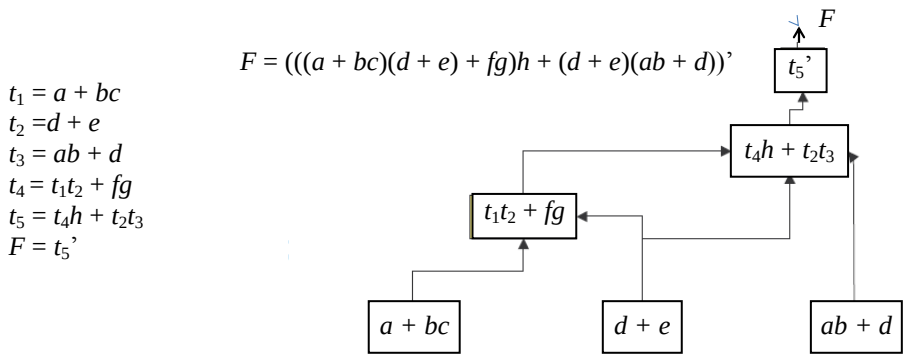


Рис. 1. Логическая сеть и ее алгебраические представления

В теории графов сетью называется помеченный ориентированный граф $G(V, E)$. В логической сети граф ациклический и множество вершин V разбито на три подмножества: первичных входов, первичных выходов, внутренних вершин. Каждая вершина помечена входной или выходной переменной либо функцией, реализуемой элементом. Для каждой вершины указаны два множества вершин, задающие отношение E: множество элементов, связанных со входами (fanin), и множество элементов, связанных с выходом (fanout). В каждой вершине множество вершин fanin (fanout) задает входящие (исходящие) дуги соответственно. Вершину i связывает с вершиной j ребро, если функция в узле j зависит явно от переменной i. Множество дуг E графа $G(V, E)$ является объединением множеств fanin и fanout всех вершин графа.

Задача синтеза ставится следующим образом [2–5]. Дано множество логических элементов L, называемое библиотекой (пусть библиотека элементов – это совокупность логических сетей, каждая из которых представляет отдельный логический элемент), и логическая сеть G

(пусть M – это множество всех логических сетей, которые можно создать на основе библиотеки L, функционально эквивалентных G). Элементы M называются отображениями исходной сети G. Цель синтеза состоит в том, чтобы найти отображение $m \in M$, оптимальное для некоторого применения сети в качестве схемы дискретного устройства. В приведенной постановке проблема синтеза представляет алгоритмически неразрешимую задачу, так как невозможно перечислить явно или неявно все элементы M. Существенное упрощение проблемы синтеза можно получить, ограничивая множество M сетями, которые структурно подобны исходному заданию G. С этим ограничением задача синтеза алгоритмически разрешима, а метод синтеза называется комбинаторным структурным подходом.

Основная идея алгоритма комбинаторного структурного синтеза проста. Предполагается, что заданная логическая сеть G уже имеет «хорошую» структуру [1, 2], т. е. оптимизирована с учетом экономии энергопотребления. Исходная сеть преобразуется в сеть, у которой все элементы с функциональными метками принадлежат одной функции (обычно двухвходовой НЕ И). Логические сети элементов библиотеки преобразуются аналогично. Этап подготовки называется детализацией сети. Отображение m строится в процессе локальной переписи исходного G. При этом в G находится подсеть N с одним выходом, которая функционально эквивалентна некоторому элементу g из L, затем N заменяется в G элементом g. В процессе покрытия каждая функция из исходной системы булевых функций представлена двумя логическими сетями: G и G–. Шаг покрытия состоит из трансформации сети G в сеть G–.

Представлением графа логической сети в памяти программы является коллекция объектов, сопоставляемых с вершинами графа. Объект вершины кроме пометки содержит и уникальный цифровой идентификатор, а также две коллекции, задающие отношение E (fanin и fanout). Коллекция объектов графа представляет собой структуру хранения данных идентификаторов других вершин. В вершинах первичных входов пуста коллекция нагрузки, а в вершинах первичных выходов – коллекция питания.

После покрытия выполняется оптимизация нагрузки элементов схемы (fanout optimisation). Логические элементы обладают определенной нагрузочной способностью – числом входов других элементов, которые может питать выход этого элемента. Алгоритм покрытия строит схему, не учитывая ограниченную нагрузочную способность элементов и ее входов. В задаче оптимизации нагрузки элементов схемы просматриваются пути распространения сигнала с выхода элемента на входы, которые этот выход питает. При необходимости в цепь распространения сигнала выхода включаются инверторы так, чтобы не нарушались ограничения нагрузочной способности [3].

В настоящее время имеется ряд коммерческих программ в составе промышленных САПР и академические программы для решения задачи технологического отображения. Разница в качестве схем, построенных известными программами, называется в [6] «шумом алгоритмов синтеза». В работе [7] исследуется качество известных алгоритмов технологического отображения, в частности показывается, что шум алгоритмов синтеза приводит к колебаниям количества элементов в схеме примерно в два раза. При этом систематического преимущества какого-либо алгоритма не выявлено. Влияние структурной чувствительности на качество по косвенным оценкам в некоторых случаях может превышать шум алгоритмов синтеза. В [5] неполный перебор эквивалентных исходных представлений назван повторным синтезом и показано, что влияние структурной чувствительности на результаты синтеза с помощью синтезаторов промышленных САПР – того же порядка, что и шум алгоритмов синтеза. Для некоторых классов схем выбор подходящего исходного представления приводит к улучшению параметров схемы на порядок [6]. Различные схемы получаются даже для структурно изоморфных описаний. Несмотря на высокое качество инструментов автоматического синтеза, влияние квалификации разработчика на результаты синтеза остается значительным, так как качество схемы зависит от выбора подходящей формы представления исходных данных.

Структурная чувствительность инструментов синтеза логических схем давно и хорошо известна разработчикам инструментов синтеза. Пользователи академических программ синтеза учитывают эту особенность, так как она выступает очевидным логическим следствием поста-

новки задачи технологического отображения. Руководящие материалы для пользователей промышленных САПР не содержат описаний используемых алгоритмов, и их пользователи склонны преувеличивать возможности инструментов.

2. Унифицированное структурно-функциональное представление схемы

Структурная модель (netlist) представляет схему в терминах взаимосвязей составляющих компонентов и не содержит ничего, кроме компонентов и цепей с их атрибутами. В своем основном значении структурное описание есть описание внутреннего устройства чего-либо. Структурное описание устройства изнутри системы характеризует ее организацию через элементы (части) и их взаимозависимости. Функциональное описание системы – это взгляд внешнего наблюдателя, определяющий его взаимодействие с окружающей средой через соотношения между входами и выходами. Структурное описание выполнимо (может моделироваться), если известны функциональные модели компонентов; функциональное описание – это описание законов функционирования, эволюции системы, алгоритмов ее поведения и моделируемо само по себе по определению [8]. Таким образом, описания поведения при помощи понятий, структуры или функции содержат фундаментальные различия.

Для представления структурных моделей в современных САПР используются специальные текстовые языки описания данных, называемые форматами структурных описаний. Формальной моделью структурных описаний являются двудольные графы [9].

В качестве примера на рис. 2 изображен фрагмент логической сети, а на рис. 3 – модель этого фрагмента в виде неориентированного двудольного графа. В данном фрагменте XORG – логический элемент сложения по модулю 2, GYMUX – трехвходовый элемент, DYMUX – двухвходовый элемент.

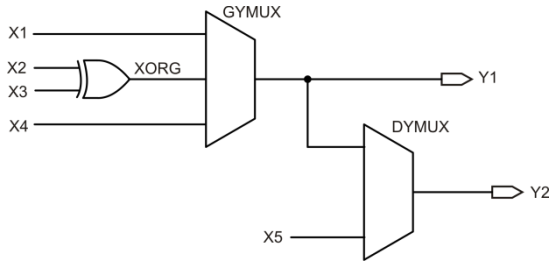


Рис. 2. Логическая сеть

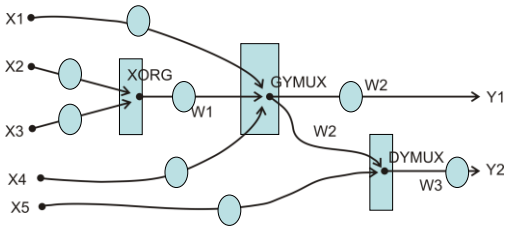


Рис. 3. Неориентированный двудольный граф соответствующей логической сети

Вершины, являющиеся выводами элемента (рис. 3), сгруппированы в прямоугольники (сами прямоугольники не являются элементами графа). Обозначения выводов предваряются именем элемента, показанным в фигуре элемента. Вершины цепей показаны кружками, обозначения (имена) выводов и некоторых цепей для упрощения опущены.

Моделью унифицированного структурно-функционального представления схемы выступает двудольный граф, одной долей которого являются порты (выводы) экземпляров элементов и порты самого устройства, а другой – цепи, соединяющие порты. Отношение графа задано в виде двух структур: списка экземпляров (instance based netlist) и списка цепей (net-based netlist).

3. Конвертация представлений схемы

Представлением графа логической сети является коллекция объектов, сопоставляемых с вершинами графа (рис. 4). Кроме пометки, объект вершины содержит уникальный цифровой идентификатор и две коллекции идентификаторов других вершин: нагрузку (fanin) и питание (fanout). Коллекция объектов графа представляет собой структуру хранения данных – двунаправленный список. В вершинах первичных входов пуста коллекция нагрузки, а в вершинах первичных выходов – коллекция питания.

Рис. 4. Структура представления логической сети

Унифицированное структурно-функциональное описание в виде диаграммы, на которой отображены интерфейсы классов, т. е. связи объектов этих классов, показано на рис. 5.

Рис. 5. Диаграмма классов структурно-функционального описания цифрового устройства

Задача конвертации представлений схем состоит в трансформации представления унифицированного структурно-функционального описания в представление логической сети и наоборот. Первое преобразование выполняется при обработке исходного формата описания схемы. Обратная трансформация (представление логической сети в представление унифицированного структурно-функционального описания) выполняется по завершении операции покрытия логической сети (рис. 6).

4. Библиотека элементов синтеза

Библиотека элементов синтеза для программы задается в формате genlib [4]. Описание каждого библиотечного элемента содержит его булеву функцию, заданную в алгебраической форме в виде текстовой строки, и стоимость элемента, влияющую на выбор элемента при

покрытии. Стоимостью библиотечного элемента обычно является число транзисторов его микросхемы (или площадь топологических базовых ячеек, размеры которых зависят от технологии изготовления КМОП-элементов).

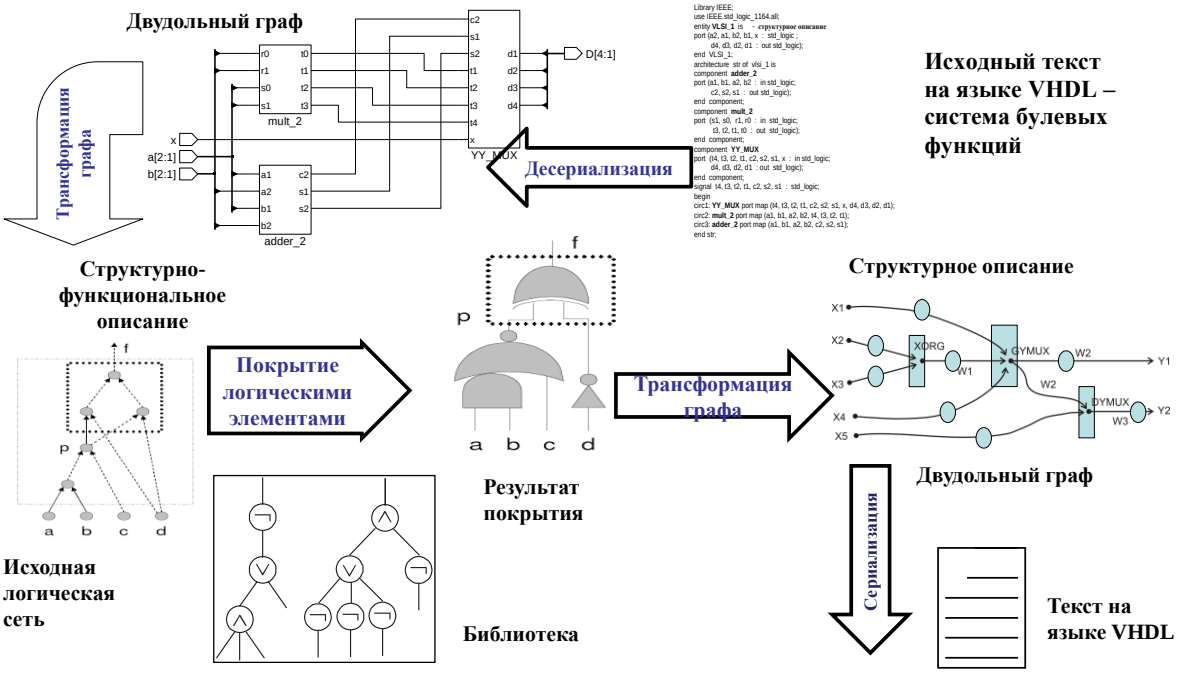


Рис. 6. Цепочка преобразований описания цифрового устройства

5. Структура программы технологического отображения

Входными данными для программы служит система ДНФ, заданная в одном из допустимых форматов. Анализаторы допустимых форматов строят унифицированное структурно-функциональное представление схемы [9]. Выбор анализатора происходит автоматически по расширению имени файла с исходным текстом (рис. 7).

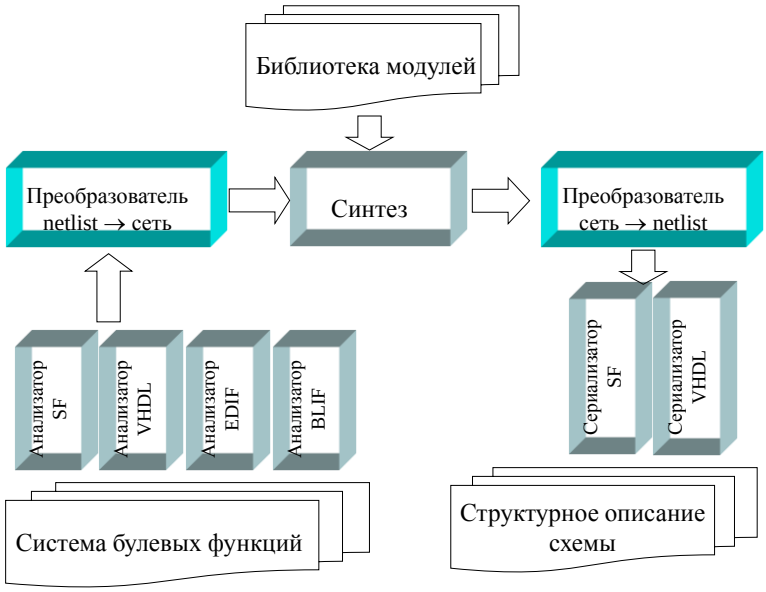


Рис. 7. Данные и операции программы технологического отображения схем из вентилей в заданный библиотечный базис

Затем это представление преобразуется в граф логической сети (см. разд. 1). Подграфу двудольного графа, составленному из вершин, которые являются портами элемента, в логической сети соответствует отдельная вершина. Анализатором формата genlib строится коллекция сетей элементов библиотеки так, что функцией каждой внутренней вершины является либо НЕ И2, либо инверсия.

Основная операция программы преобразует многоуровневую схему из элементов технологически независимого базиса (называемую исходной) в многоуровневую схему (называемую объектной) из библиотечных элементов путем локальных замен подсхем исходной сети. Предварительно (до покрытия) исходная логическая сеть подвергается детализации. Процедура покрытия содержит проверку функциональной эквивалентности исходной и объектной логических сетей.

По завершении операции покрытия логическая сеть, функционально эквивалентная исходной системе ДНФ (результат работы этапа покрытия), преобразуется в унифицированное структурно-функциональное представление схемы, которое сериализуется в требуемый формат. Текст, являющийся результатом работы программы, строится в формате SF (CONNECT) или в форме VHDL-описания структурного стиля.

6. Форматы представления исходной системы булевых функций

Допустимыми входными форматами исходной системы булевых функций являются: SF (sf, SF), VHDL (vhdl, VHDL, vhd, VHD, v), EDIF (edf, EDF, edif, EDIF), BLIF (blf, BLF, blif, BLIF), PLA (pla, PLA). (В скобках указаны расширения имени файла.)

Обрабатываются описания в формате SF, содержащие и структурные, и функциональные описания схем, в том числе файлы с несколькими описаниями различных типов. Система ДНФ задается на языке SF структурно-функциональным описанием в виде схемы из подсхем (блоков) или на функциональном уровне в матричной либо алгебраической форме. Структурное описание головной подсхемы должно быть первым в тексте и формата CONNECT. Функциональные описания листовых блоков представляют собой либо логические уравнения – скобочные формы в базисе операций И, ИЛИ, НЕ (в формате LOG), либо матричную форму представления системы ДНФ булевых функций (в формате SDF). Если головной блок описан на функциональном уровне, то вся схема состоит из единственного блока.

В графе логической сети полностью сохранится структура иерархии описания блоков. В сети, построенной по формату LOG, сохранится структура уравнений, но будет потеряна структура вхождения скобок, т. е. описание уравнения будет приведено в форму ДНФ.

Анализатор VHDL, используемый в программе, построен по грамматике VHDL «Стандарт VHDL–93». Все конструкции языка допустимы, но программой обрабатывается только подмножество языка. Обрабатываемое подмножество позволяет описывать только цифровые синхронные схемы. Исходное описание должно соответствовать структурному стилю, допускается включение операторов сигнального присваивания с тривиальной правой частью. Описания всех элементов структуры должны содержаться в тексте в виде описаний потокового стиля, состоящих из операторов сигнального присваивания. Обрабатываемые описания должны принадлежать синтезируемому подмножеству VHDL по типам сигналов.

Структурное описание головной подсхемы должно быть первым в тексте. В графе логической сети полностью сохранится структура иерархии описания блоков. В описаниях потокового стиля в уравнениях сохранится структура вхождения скобок, т. е. описание уравнения будет приведено в форму логической сети из элементов И, ИЛИ, НЕ с двумя входами.

Анализатор EDIF версии 2.0.0, используемый в программе, построен трансляцией грамматики EDIF, свободно доступной через веб-сервер Манчестерского университета в Англии. Так как EDIF является языком структурных описаний, то программа может обработать их, только если по названиям блоков можно установить булеву функцию элемента.

В структурном описании должны использоваться следующие наименования примитивных элементов: T0, T1 – константы 0, 1; Buf, Inv – буфер, инвертор; And, Xor, Or, Nor, Xnor, Nand – логические элементы. На число аргументов функции логических элементов ограничений нет. В графе логической сети полностью сохранится структура соединений примитивных элементов.

Формат BLIF (Berkeley Logic Interchange Format) предназначен для описания иерархических схем на логическом уровне в текстовой форме. Схема – это произвольная комбинационная или последовательная логическая сеть. Для каждого комбинационного элемента задана булева функция, описывающая вычисление значения единственного вывода этого элемента [4]. Программа не может обрабатывать описания, содержащие последовательностные элементы. Не обрабатываются также и иерархические описания.

Формат PLA является подмножеством формата BLIF и служит средством задания систем булевых функций в форме ДНФ.

Заключение

Программа отображения логических сетей в заданный технологический базис включена как проектная операция в программный комплекс энергосберегающего логического синтеза [10], предназначенного для автоматизации проектирования многоуровневых логических схем из библиотечных элементов заказных СБИС, выполненных по КМОП-технологии. Система энергосберегающего логического синтеза имеет развитые средства технологически независимой оптимизации, что позволяет использовать ее при синтезе логических схем не только в базисе КМОП-элементов, но и в других технологических базисах. Программный комплекс данного синтеза предназначен для проектирования схем комбинационной логики, имеющих сотни входных-выходных переменных и тысячи элементов. Исходное функциональное описание проектируемой логической схемы в переводе на эквивалентное представление в виде системы булевых функций в ДНФ может иметь до нескольких десятков переменных и функций и нескольких тысяч дизъюнкций.

Список литературы

1. Черемисинов, Д.И. Синтез комбинационных схем в базисе библиотечных элементов КМОП СБИС с учетом энергопотребления / Д.И. Черемисинов, Л.Д. Черемисинова // Информатика. – 2013. – № 4(40). – С. 91–102.

2. Закревский, А.Д. Логические основы проектирования дискретных устройств / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – М. : Физматлит, 2007. – 592 с.

3. Reducing Structural Bias in Technology Mapping / S. Chatterjee [et al.] // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2006. – Vol. 25, no. 12. – P. 2894–2903.

4. Sis: A system for sequential circuit synthesis : technical report UCB/ERL M92/41, EECS Department / E.M. Sentovich [et al.] [Electronic resource]. – University of California, Berkeley, 1992. – Mode of access : https://www.eecs.berkeley.edu/Pubs/TechRpts/1992/2010.html. – Date of access : 11.11.2016.

5. Бибило, П.Н. Логическое проектирование дискретных устройств с использованием продукционно-фреймовой модели представления знаний / П.Н. Бибило, В.И. Романов. – Минск : Беларуская навука, 2011. – 279 с.

6. Fišer, P. Sources of Bias in EDA Tools and Its Influence / P. Fišer, J. Schmidt, J. Balcárek // Proc. of the 2014 IEEE 17th Intern. Symp. on Design and Diagnostics of Electronic Circuits & Systems. – Piscataway : IEEE, 2014. – P. 258–261.

7. Бибило, П.Н. Покрытие булевой сети библиотечными элементами / П.Н. Бибило, В.Г. Лицкевич // Управляющие системы и машины. – 1999. – № 6. – С. 16–24.

8. Перегудов, Ф.П. Введение в системный анализ / Ф.П. Перегудов, Ф.П. Тарасенко. – М. : Высшая школа, 1989. – 360 с.

9. Черемисинов, Д.И. Обработка графов в программе перепроектирования FPGA / Д.И. Черемисинов // Танаевские чтения : доклады Шестой Междунар. конф. – Минск : ОИПИ НАН Беларуси, 2014. – С. 151–155.

10. Автоматизация логического синтеза КМОП-схем с пониженным энергопотреблением / П.Н. Бибило [и др.] // Программная инженерия. – 2013. – № 8. – С. 35–41.

Поступила 22.12.2016

Объединенный институт проблем информатики НАН Беларуси, Минск, Сурганова, 6
e-mail: cher@newman.bas-net.by

D.I. Cheremisinov

TECHNOLOGY MAPPING TOOL FOR VLSI CAD

Technology mapping program implements a sequential circuit using the gates of a particular technology library. It is an integral component of any automated VLSI circuit design flow. The structure of the program for solving the technology mapping problem and formats of the source and result data are presented. Models of intermediate representations of the sequential circuit and their conversions are described. Technology mapping is a stage of logic synthesis and it is viewed as the transformation of a functional (i.e., algebraic) circuit specification into a gate (i.e., netlist) specification. The program is included as project operations in the VLSI CAD system for energy-saving logical synthesis developed in the United Institute of Informatics Problems of NAS of Belarus.

ПРИКЛАДНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

УДК 658.512.2:621.9.06

Н.Н. Гущинский, Б.М. Розин

ОПТИМИЗАЦИЯ РАЗМЕЩЕНИЯ ГРУППЫ ДЕТАЛЕЙ НА МНОГОПОЗИЦИОННОМ ПОВОРОТНОМ СТОЛЕ АГРЕГАТНОГО СТАНКА

Рассматривается задача минимизации массы агрегатного станка с многопозиционным поворотным столом за счет размещения каждой детали группы на столе при их групповой последовательно-параллельной обработке. Предлагаются математическая модель и эвристический алгоритм «роя частиц» для решения данной задачи. Приводятся результаты численных экспериментов по решению предложенным методом задач этого типа.

Введение

В условиях современного производства агрегатные станки (АС) все чаще используются для серийного выпуска группы деталей. Проектирование компоновочных схем современных АС, компонуемых по блочно-модульному принципу при заданных структуре и параметрах группового технологического процесса (ГТП) обработки группы деталей, заключается в определении состава, типоразмеров основных силовых и корпусных узлов, а также параметров их взаимного расположения. Жесткие требования к срокам выпуска новой продукции, диктуемые рынком, стимулируют широкое применение средств автоматизации проектирования станочного оборудования. Решению задач автоматизации выбора компоновок различных типов станков, в том числе АС, посвящен ряд публикаций [1–7].

В работах [4–6] описана разработанная в ОИПИ НАН Беларуси подсистема «Компоновка» программного комплекса «Агрегат», предназначенная для автоматизации построения компоновок АС при серийной обработке однородных партий деталей, формируемых по блочно-модульному принципу нескольких компоновочных схем с приспособлением для установки деталей на многопозиционных поворотных столах, однопозиционных стационарных и двухпозиционных передвижных столах. Формирование компоновки при помощи этой подсистемы осуществляется в автоматизированном режиме с учетом информации о структуре и параметрах ТП, положении обрабатываемой детали на виртуальном приспособлении АС выбранного типа, а также данных об основных параметрах приспособления и инструментальной наладки. Для полученного варианта компоновки АС определяются его цена, масса, высота и занимаемая площадь.

Кроме структуры и параметров ГТП, существенное влияние на формируемую компоновку АС оказывает расположение обрабатываемых деталей на позициях обработки АС. Типоразмеры ряда унифицированных узлов и сборочных единиц АС (например, многошпиндельных коробок, силовых и крестовых столов, на которых размещаются эти коробки и силовые бабки) либо размеры оригинальных узлов (центральных и боковых станин, стоек и подставок), как правило, существенно зависят от размещения на приспособлении обрабатываемых деталей [6, 7].

Для построения рационального варианта компоновки АС в ручном режиме пользователю необходимо устанавливать на загрузочной позиции стола 3D-модели группы деталей и проводить расчеты многократно. При этом не гарантируется построение даже допустимой компоновки. На поиск же оптимальной (по выбранному критерию оптимальности) компоновки может потребоваться значительное время. В работе [7] для задачи синтеза оптимальной компоновки АС с поворотным столом для обработки однородной партии деталей был предложен метод «роя частиц». В качестве критерия оптимальности использовалась минимальная масса АС.

В настоящей работе этот метод распространен на более сложную задачу оптимизации компоновочной схемы АС с поворотным столом для обработки группы различных деталей.

1. Постановка задачи

Рассматривается задача оптимизации компоновки АС с поворотным столом для обработки группы деталей нескольких наименований посредством выбора расположения этих деталей на приспособлении АС. Особенности данной задачи по сравнению с аналогичной задачей оптимизации размещения одной детали на загрузочной позиции АС для процессов обработки однородных партий деталей следующие (см. [7]): необходимость формирования для каждой рабочей позиции силовых сборочных единиц, выполняющих обработку нескольких деталей группы различными блоками инструментов; координация размещения различных деталей на позиции с учетом обработки некоторых конструктивных элементов (отверстий, поверхностей) различных деталей одним инструментом.

Исходными данными для формирования варианта компоновки АС в подсистеме «Компоновка» наряду с параметрами ГТП являются параметры размещения с использованием графической системы на загрузочной позиции виртуального приспособления 3D-моделей каждой из деталей группы, а также начальные значения длины базы инструментов силовых узлов и длины съема инструментов, задаваемые проектантом в режиме диалога.

Формирование варианта компоновки АС, выполняющего заданный технологический процесс обработки группы деталей, осуществляется по следующей схеме:

– размещение 3D-моделей группы деталей на виртуальном приспособлении с одновременным определением соответствующих габаритных размеров приспособления (планшайбы поворотного стола) и центральной станины;

– задание начальных значений длин баз инструментов силовых узлов и длины съема инструментов;

– формирование вариантов компоновки выполняющих обработку боковых приставок;

– построение графической 3D-модели станка для выбранного варианта компоновки;

– размещение дополнительного оборудования АС;

– расчет характеристик АС;

– документирование результатов компоновки.

Далее ограничимся рассмотрением АС с приспособлением на многопозиционном поворотном делительном столе с вертикальной осью вращения и числом боковых приставок до пяти, предназначенного для обработки группы деталей n различных наименований (рис. 1).

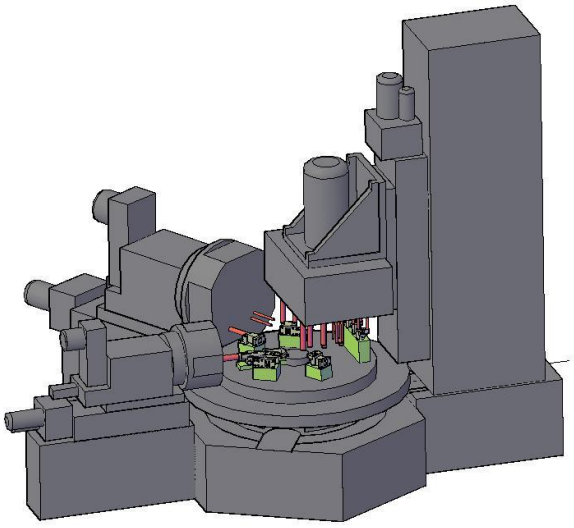


Рис. 1. АС с поворотным столом для параллельной обработки группы деталей

Следует отметить, что аналогичные задачи возникают и для АС других типов: стационарных станков, станков с приспособлением на передвижном столе и др. Из перечисленных типов АС рассматриваемый здесь тип станка представляется наиболее сложным по структуре

ограничений задачи, что делает его наиболее подходящим для исследования задачи оптимизации его компоновки.

Рассматриваемый АС с приспособлением на поворотном столе состоит из центральной станины с поворотным столом для размещения и закрепления обрабатываемых деталей и боковых приставок, осуществляющих их обработку. Для таких АС предусмотрено использование одногранных (четырехгранных с одной обработанной гранью, предназначенной для установки приставки), четырехгранных, шестигранных или восьмигранных типовых средних станин под поворотный стол. Наибольшее число боковых приставок для таких средних станин не превышает пяти.

Среди боковых приставок могут быть вертикальные и (или) горизонтальные приставки с соответственно вертикально либо горизонтально ориентированными силовыми или крестовыми столами и установленными на них силовыми бабками и многошпиндельными коробками. Предполагается, что обработка деталей группы может осуществляться в направлении осей 0*z*, 0*y* и 0*x* системы координат АС. В первом случае выполняется либо обработка отверстий детали в горизонтальной плоскости вертикально ориентированными стержневыми инструментами, либо фрезерование боковой плоскости фрезерной бабкой, установленными на вертикальном силовом столе. Во втором случае осуществляется либо фрезерование горизонтальной поверхности, либо сверление, растачивание, зенкерование и т. п. отверстий сбоку (соответствующими силовыми узлами, установленными на горизонтальном силовом столе). Для оси 0*x* выполняется фрезерование боковой поверхности фрезерной бабкой, установленной на поперечно ориентированном горизонтальном силовом столе.

Существует два типа боковых приставок. Первый тип составляют горизонтальные либо вертикальные приставки, предназначенные для обработки деталей группы (сбоку либо сверху) на одной позиции поворотного стола, второй – вертикальные приставки, предназначенные для вертикальной параллельной обработки сверху деталей группы на нескольких позициях стола. Заданы число позиций поворотного стола (вообще говоря, не совпадающее с числом граней центральной станины) и количество приставок каждого типа. Для приставок первого типа, выполняющих обработку деталей на соответствующих позициях, заданы номера граней средней станины для их размещения. Для АС этого типа расположение обрабатываемых деталей в плоскости стола существенно влияет на типоразмеры выбираемых узлов, в особенности узлов боковых приставок второго типа.

Боковые приставки второго типа (для параллельной обработки на нескольких позициях сверху) могут быть размещены на любой из предусмотренных для этого граней центральной станины АС, не занятых приставками первого типа.

В процессе поиска варианта компоновки в режиме диалога пользователем задается толщина литья каждой шпиндельной коробки, необходимая для определения ее типоразмеров для установки на соответствующем силовом столе. Основные же компоновочные параметры виртуального приспособления для размещения деталей на плоскости поворотного стола (диаметр планшайбы поворотного стола, диаметр приспособления, значения длины базы инструментов) АС определяются автоматически.

При формировании варианта компоновки учитываются характерные для рассматриваемого типа АС технологические, конструктивные и эргономические ограничения. Эти ограничения можно разделить на три группы: компоновочные, конструктивно-технологические и ограничения на размещение зажимного приспособления детали на загрузочной позиции.

Компоновочные ограничения распространяются на взаимное расположение узлов каждой приставки и узлов различных приставок между собой и связаны с обеспечением реализуемости компоновочной схемы.

К конструктивно-технологическим относятся, в частности, ограничения на минимальные межшпиндельные расстояния каждой многошпиндельной коробки (расстояния между осями пары шпинделей в плоскости, ортогональной их осям), на размещение осей шпинделей относительно границ габаритных размеров литья шпиндельной коробки и относительно направляющих кондукторной плиты [7], на параметры инструментов (например, диаметр фрезы) при обработке одним инструментом семейства элементов (фрезеруемых поверхностей) различных деталей, а также на расположение этих элементов.

Третья группа включает ограничения невыхода виртуального зажимного приспособления детали за границы сектора позиции поворотного стола. Более подробно ограничения первой и второй группы описаны в работах [4, 7].

Перечислим более детально конструктивные ограничения, учитываемые при выборе габаритов шпиндельной коробки:

– межшпиндельное расстояние должно превышать минимально допустимое, зависящее от параметров (усилий, угловых скоростей) выполняемых соответствующими инструментами технологических переходов;

– расстояния от оси шпинделя до краев литья коробки и до направляющих втулок комплектов деталей для крепления штанг кондукторных плит должны превышать заданные табличные значения [7];

– значения параметров (высоты и ширины) коробки выбираются из ряда пар типовых величин, соответствующих ее возможным типоразмерам.

Замечание. Следует обратить внимание на существенное различие влияния ограничений на минимальное межшпиндельное расстояние для шпинделей шпиндельной коробки, выполняющих обработку одной детали, и для шпинделей, обрабатывающих различные детали на одной либо различных позициях. Для первого случая нарушение такого ограничения хотя бы для одной пары шпинделей коробки приводит к ее нереализуемости (для заданной структуры технологического процесса), поскольку межшпиндельное расстояние определяется расположением осей отверстий на плоскости детали и параметрами соответствующих технологических переходов. Для второго случая межшпиндельные расстояния могут быть увеличены за счет изменения взаимного расположения деталей на столе.

Снижение массы станка достигается за счет уменьшения масс составляющих его узлов или, что эквивалентно, уменьшения их типоразмеров, существенно зависящих от расположения на станке каждой из деталей группы. При этом, основываясь на опыте разработки соответствующего семейства металлообрабатывающих АС, предполагается, что изменение типоразмеров узлов в заданном диапазоне не приводит к снижению точности и надежности обработки. Размещение на загрузочной позиции АС каждой детали группы при заданной ее ориентации однозначно определяется координатами начала связанной с деталью локальной системы координат (ЛСК) в системе координат АС, начало которой совмещено с центром поворотного стола.

Рассматриваемая в настоящей работе задача сводится к нахождению допустимых значений координат начал ЛСК каждой обрабатываемой детали группы на загрузочной позиции поворотного стола с вертикальной осью вращения, а также выбору граней средней станины для установки боковых приставок второго типа и длин баз инструментов силовых узлов, определяющих вариант компоновки АС с минимальной массой.

Предлагаемый ниже подход может быть использован также для разработки методов решения аналогичных задач для других типов АС.

2. Математическая модель

Предполагается, что начало трехмерной системы координат АС (правой декартовой) совмещено с центром поворотного стола, ее ось 0*z* является нормалью к плоскости поворотного стола, а ось 0*y* направлена вдоль биссектрисы внешнего угла сектора загрузочной позиции. Предполагается также, что расположение ЛСК каждой детали на загрузочной позиции АС отличается от расположения системы координат АС только координатами ее начала. Обработка деталей группы может осуществляться на каждой из позиций станка соответствующей боковой приставкой посредством многошпиндельной коробки либо набором силовых бабок, установленных на вертикальном либо горизонтальном силовом столе.

Введем необходимые в дальнейшем обозначения:

$x^i_{лск}$, $y^i_{лск}$, $z^i_{лск}$ – координаты начала ЛСК *i*-й детали, расположенной на загрузочной позиции, в системе координат АС;

$N_{поз}$ – число позиций АС, включая загрузочную;

$N_{гр}$ – число граней средней станины;

N_1, N_2 – числа боковых приставок первого и второго типа соответственно;

$\aleph = \{n_1, n_2, ..., n_{N_1}\}$ – номера граней, занятых приставками первого типа;

$Q = \{\chi_1, \chi_2, ..., \chi_\theta\}$ – номера свободных граней средней станины для размещения боковых приставок второго типа, $\theta = N_{гр} - N_1$, $\theta \geq N_2$;

q_r – номер грани средней станины для размещения r-й боковой приставки второго типа, $q_r \in Q$;

$\overline{Q}$ – множество значений вектора $q = (q_1, q_2, ..., q_{N_2})$, взаимно-однозначно сопоставляющего номерам $r \in \{1, 2, \ldots, N_2\}$ боковых приставок второго типа номера $q_r \in Q$ свободных граней средней станины;

$N^i_{конт}$ – число точек излома кусочно-линейной ломаной, описывающей контур i-й детали (либо ее зажимное приспособление);

Δ^l_{xi}, Δ^l_{yi} – координаты l-й точки контура i-й детали в плоскости $0xy$ ЛСК детали, $l = 1, 2, ..., N^i_{конт}$;

$(X_{лск}, Y_{лск}, Z_{лск})$ – вектор координат начал ЛСК всех деталей в системе координат АС, где $X_{лск} = (x^1_{лск}, x^2_{лск}, \ldots, x^n_{лск})$, $Y_{лск} = (y^1_{лск}, y^2_{лск}, \ldots, y^n_{лск})$, $Z_{лск} = (z^1_{лск}, z^2_{лск}, \ldots, z^n_{лск})$ – векторы координат начал ЛСК всех деталей по осям $0x$, $0y$, $0z$ системы координат АС соответственно;

b_k – длина базы инструментов силового узла k-й боковой приставки, $k = 1, 2, \ldots, N_1+N_2$;

$[\underline{b}_k, \overline{b}_k]$ – интервал возможных значений длины базы инструментов k-й боковой приставки, $k = 1, 2, \ldots, N_1+N_2$;

$m^k_{пр}(b_k, (X_{лск}, Y_{лск}, Z_{лск}))$ – масса k-й боковой приставки первого типа, $k = 1, 2, \ldots, N_1$;

$m^r_{пр}(q_r, b_r, (X_{лск}, Y_{лск}, Z_{лск}))$ – масса r-й боковой приставки второго типа, $r = 1, 2, \ldots, N_2$;

$m_{ср}(X_{лск}, Y_{лск}, Z_{лск})$ – масса средней станины АС;

$m_{пов}(X_{лск}, Y_{лск}, Z_{лск})$ – масса поворотного стола АС.

В рассматриваемой задаче управляемыми переменными являются вектор $(X_{лск}, Y_{лск}, Z_{лск})$ координат начал ЛСК деталей в системе координат АС, вектор $q = (q_1, q_2, ..., q_{N_2})$ номеров q_r, $r = 1, \ldots, N_2$, граней центральной станины для установки соответствующих вертикальных боковых приставок второго типа (для параллельной обработки сверху) и вектор $b = (b_1, b_2, \ldots, b_{N_1+N_2})$ длин баз инструментов боковых приставок.

Далее описываются наиболее существенные ограничения, которым в реальных проектных ситуациях должны удовлетворять векторы $(X_{лск}, Y_{лск}, Z_{лск})$ координат ЛСК деталей при их размещении, а также векторы q и b.

Характерной особенностью ГТП является обработка одним и тем же инструментом элементов различных деталей группы. В этом случае размещение деталей группы должно координироваться в системе координат станка. Такая координация может быть определена посредством задания начальных положений $(x^i_{лск}(0), y^i_{лск}(0), z^i_{лск}(0))$ ЛСК деталей $i = 1, 2, \ldots, n$ в системе координат АС, удовлетворяющих условиям этой координации. Тогда координация отдельных деталей группы в связи с обработкой некоторых их элементов одним инструментом может быть задана системой равенств:

$$x^i_{лск} - x^j_{лск} = x^i_{лск}(0) - x^j_{лск}(0), (i, j) \in I_x; \quad (1)$$

$$y^i_{лск} - y^j_{лск} = y^i_{лск}(0) - y^j_{лск}(0), (i, j) \in I_y; \quad (2)$$

$$z^i_{лск} - z^j_{лск} = z^i_{лск}(0) - z^j_{лск}(0), (i, j) \in I_z, \quad (3)$$

где I_x, I_y и I_z – множества пар номеров деталей i и j, которые должны быть скоординированы по x, y и z соответственно.

В частности, для операции фрезерования координаты y центров фрезеруемых боковых поверхностей и координаты z центров этих поверхностей при фрезеровании горизонтальных

поверхностей должны совпадать. Кроме того, размеры фрезеруемых поверхностей также должны быть скоординированы.

Пусть P – семейство множеств $P_j = \{p_1, p_2, \ldots, p_{k_j}\}$, $j = 1, 2, \ldots, |P|$, переходов фрезерования, выполняемых одной и той же фрезой с диаметром d_j. Каждый переход p_l, $l = 1,\ldots, k_j$, характеризуется набором $(i(p_l), x(p_l), y(p_l), z(p_l), \lambda(p_l), w(p_l))$, где $i(p_l)$ – номер детали, содержащей элемент, обрабатываемый в рамках перехода p_l; $x(p_l)$, $y(p_l)$, $z(p_l)$ – координаты центра обрабатываемого элемента в ЛСК детали; $\lambda(p_l)$ – длина фрезерования и $w(p_l)$ – ширина фрезерования. Рассматриваются случаи, когда возможна обработка фрезерованием горизонтальных поверхностей горизонтальным столом, а также аналогичная обработка боковых поверхностей вертикальным или горизонтальным столом. В первых двух случаях диаметр фрезы не может быть меньше максимальной разницы координат x обрабатываемых элементов различных деталей, а в третьем случае – максимальной разницы координат z. Это может быть обеспечено при выполнении следующих соотношений:

$$\max\{x_{лск}^{i(p_l)} + x(p_l) + w(p_l)/2 \mid l = 1, \ldots, k_j\} - \min\{x_{лск}^{i(p_l)} + x(p_l) - w(p_l)/2 \mid l = 1, \ldots, k_j\} \le d_j, \quad j = 1, 2, \ldots, |P|, P_j \in P^x; \tag{4}$$

$$\max\{z_{лск}^{i(p_l)} + z(p_l) + w(p_l)/2 \mid l = 1, \ldots, k_j\} - \min\{z_{лск}^{i(p_l)} + z(p_l) - w(p_l)/2 \mid l = 1, \ldots, k_j\} \le d_j, \quad j = 1, 2, \ldots, |P|, P_j \in P^z, \tag{5}$$

где P^x – подсемейство из P, соответствующее первым двум случаям обработки, а P^z – третьему случаю. При этом предполагается, что необходимая длина фрезерования $\lambda(p_l)$ будет обеспечиваться за счет быстрого подвода соответствующих силовых узлов.

Размещение деталей (зажимных приспособлений) в пределах сектора загрузочной позиции обеспечивается следующими соотношениями:

$$y_{лск}^i + \Delta_{yi}^l - (x_{лск}^i + \Delta_{xi}^l) \cdot \mathrm{tg}(\pi / N_{поз}) \le 0, \; l = 1, 2, \ldots, N_{конт}^i, \; i = 1, 2, \ldots, n; \tag{6}$$

$$y_{лск}^i + \Delta_{yi}^l + (x_{лск}^i + \Delta_{xi}^l) \cdot \mathrm{tg}(\pi / N_{поз}) \le 0, \; l = 1, 2, \ldots, N_{конт}^i, \; i = 1, 2, \ldots, n. \tag{7}$$

Система ограничений для выбора совокупности узлов боковой приставки первого либо второго типа соответственно может быть задана следующим образом:

$$\Omega_k(b_k, (X_{лск}, Y_{лск}, Z_{лск})) \le 0, \; k = 1, \ldots, N_1; \tag{8}$$

$$\Omega_r(q_r, b_r, (X_{лск}, Y_{лск}, Z_{лск})) \le 0, \; r = 1, \ldots, N_2, \tag{9}$$

где Ω_k, Ω_r – вектор-функции и правые части неравенств также являются векторами соответствующей левой части размерности с нулевыми компонентами.

В состав ограничений (8), (9) входят (при фиксированных структуре, параметрах ГТП и параметрах приспособления) ограничения на выбор из базы данных узлов АС (силовых бабок, шпиндельных коробок, угольников, силовых и крестовых столов, стоек, подставок, боковых станин). Эти ограничения связаны с необходимостью обеспечения при выборе силового узла требуемых в процессе обработки усилия, мощности привода, крутящего момента, величины рабочего хода, длины съема инструментов, минимальных межцентровых расстояний пар шпинделей и др. [4]. Пример системы таких ограничений для выбора многошпиндельной коробки боковой приставки второго типа приведен в работе [7].

Если множество реализаций (допустимых компоновок) боковой приставки k первого типа либо приставки r второго типа, описываемое каким-либо ограничением (8) либо (9), пусто, то полагается $m_{пр}^k(b_k, (X_{лск}, Y_{лск}, Z_{лск})) = \infty$ либо $m_{пр}^r(q_r, b_r, (X_{лск}, Y_{лск}, Z_{лск})) = \infty$ соответственно.

Кроме описанных выше ограничений на выбор узлов и сборочных единиц боковых приставок по отдельности, компоновка АС должна удовлетворять ряду ограничений, задаваемых алгоритмически и связанных с взаимодействием узлов различных приставок между собой

и с приспособлением [4]. В частности, к таким ограничениям относится запрет на взаимное пересечение 3D-моделей узлов и сборочных единиц соседних боковых приставок в их положении в конце обработки. В дальнейшем для сокращения изложения описываемое этими ограничениями множество допустимых значений вектора координат ($X_{лск}$, $Y_{лск}$, $Z_{лск}$) начал ЛСК деталей группы на загрузочной позиции приспособления для фиксированного вектора q = (q_1, q_2, ..., q_{N_2}) номеров граней вертикальных боковых приставок второго типа и вектора b = (b_1, b_2,…, $b_{N_1+N_2}$) длин баз инструментов будем обозначать $G(q, b)$:

$$(X_{лск},\ Y_{лск},\ Z_{лск}) \in G(q, b)\,. \tag{10}$$

Номера q_r граней средней станины для вертикальных боковых приставок r = 1, 2,…, N_2 второго типа выбираются из множества номеров свободных граней:

$$q \in \overline{Q}\,. \tag{11}$$

Компоненты b_k вектора b баз инструментов боковых приставок должны принимать значения из заданных диапазонов:

$$b \in \mathbf{B} = \prod_{k=1}^{N_1+N_2} [\underline{b}_k,\ \overline{b}_k]\,. \tag{12}$$

Масса $F(q,\ b,\ (X_{лск},\ Y_{лск},\ Z_{лск}))$ АС представляет собой сумму масс средней станины $m_{ср}(X_{лск}, Y_{лск}, Z_{лск})$, поворотного стола $m_{пов}(X_{лск}, Y_{лск}, Z_{лск})$, боковых приставок первого типа $m^k_{пр}(b_k, (X_{лск},\ Y_{лск},\ Z_{лск}))$, k = 1, 2, …, N_1, боковых приставок второго типа $m^r_{пр}(q_r, b_r,\ (X_{лск},\ Y_{лск},\ Z_{лск}))$, r = 1, 2, …, N_2, и вспомогательного оборудования. Поскольку состав вспомогательного оборудования не зависит от векторов ($X_{лск}$, $Y_{лск}$, $Z_{лск}$), q, b, его масса в дальнейшем без ограничения общности не учитывается.

Таким образом, задача заключается в выборе векторов ($X_{лск}$, $Y_{лск}$, $Z_{лск}$), q и b, удовлетворяющих ограничениям (1)–(12) и минимизирующих массу АС:

$$\begin{aligned} F(q,b,(X_{лск},Y_{лск},Z_{лск})) = \sum_{k=1}^{N_1} m^k_{пр}(b_k,(X_{лск},Y_{лск},Z_{лск})) + \sum_{r=1}^{N_2} m^r_{пр}(q_r,b_r,(X_{лск},Y_{лск},Z_{лск})) + \\ + m_{ср}(X_{лск},Y_{лск},Z_{лск}) + m_{пов}(X_{лск},Y_{лск},Z_{лск}) \to \min\,. \end{aligned} \tag{13}$$

Если система ограничений (1)–(12) несовместна, то полагается $F(q, b, (X_{лск}, Y_{лск}, Z_{лск})) = \infty$.

3. Метод решения

Система ограничений задачи (1)–(12) на допустимые значения векторов ($X_{лск}$, $Y_{лск}$, $Z_{лск}$) непрерывных переменных координат начал ЛСК группы обрабатываемых деталей и b значений баз инструментов, а также целочисленных переменных q_r номеров граней для размещения приставок второго типа r = 1, …, N_2 состоит из линейных либо сводящихся к линейным ограничений (1)–(3), (4)–(7), (12), условий целочисленности переменных (11) и заданных алгоритмически ограничений (8)–(10). Значения целевой функции $F(q, b, (X_{лск}, Y_{лск}, Z_{лск}))$ вычисляются алгоритмически. Можно показать, что даже только ограничения на межшпиндельные расстояния при выборе шпиндельной коробки боковой приставки второго типа порождают невыпуклую область допустимых положений деталей [7]. Таким образом, в общем случае задача (1)–(13) является сложной многоэкстремальной задачей.

Для таких задач наиболее подходящими методами решения традиционно являются эвристики и метаэвристики [8], а также методы зондирования области поиска с использованием ЛП-последовательностей (ЛП) и случайного поиска (СП) [9, 10]. Среди метаэвристик перспективными представляются метод «роя частиц» (PSO – Particle Swarm Optimization Algorithm), генетические алгоритмы, алгоритм имитации отжига, нейронные сети.

В работе для исследования был выбран PSO – эвристический метод численной оптимизации, не требующий специальных свойств оптимизируемой функции. Этот метод нашел применение в большом количестве областей [11], в частности он хорошо зарекомендовал себя при

решении задач, аналогичных исследуемой в настоящей работе [7, 12, 13]. PSO оптимизирует целевую функцию, поддерживая популяцию возможных решений, называемых частицами, и перемещая эти частицы в области поиска решений согласно итеративно вычисляемой простой формуле. Перемещения частиц учитывают наилучшее найденное в этой области положение, которое изменяется от итерации к итерации при нахождении частицами более выгодных положений. Ниже приведен алгоритм решения рассматриваемой задачи, основанный на методе PSO.

Введем следующие обозначения:

m – число частиц роя в популяции;

$F(q^\gamma, b^\gamma, (X^\gamma_{лск}, Y^\gamma_{лск}, Z^\gamma_{лск}))$ – значение целевой функции (масса АС) для γ-й частицы, γ = 1, 2, ..., m, при размещении $q^\gamma \in \overline{Q}$ боковых приставок второго типа на гранях Q и фиксированном векторе $b^\gamma \in \mathbf{B}$ длин баз инструментов;

$\overline{p}^\gamma = (\overline{X}^\gamma,\ \overline{Y}^\gamma,\ \overline{Z}^\gamma)$ – положение γ-й частицы, в котором целевая функция принимает наименьшее значение среди всех ее значений для γ-й частицы, рассмотренных к моменту текущей итерации алгоритма, γ = 1, 2, ..., m;

$\tilde{q}^\gamma = (\tilde{q}^{\gamma 1},\ \tilde{q}^{\gamma 2},\ \ldots,\ \tilde{q}^{\gamma N_2})$ – номера граней, на которых достигается наименьшее значение целевой функции в точке $\overline{p}^\gamma$;

$\tilde{b}^\gamma = (\tilde{b}^{\gamma 1},\ \tilde{b}^{\gamma 2},\ \ldots,\ \tilde{b}^{\gamma N_1+N_2})$ – длины баз инструментов, на которых достигается наименьшее значение целевой функции в точке $\overline{p}^\gamma$;

$g = (\hat{X}^{\gamma^*},\ \hat{Y}^{\gamma^*},\ \hat{Z}^{\gamma^*})$ – положение частицы роя γ^*, в котором целевая функция принимает наименьшее значение среди всех ее значений для всех частиц $\gamma \in \{1, 2, ..., m\}$, рассмотренных к моменту текущей итерации алгоритма;

$q^g = (q^1,\ q^2,\ \ldots,\ q^{N_2})$ – номера граней, на которых достигается наименьшее значение целевой функции для точки g;

$b^g = (b^1,\ b^2,\ \ldots,\ b^{N_1+N_2})$ – длины баз инструментов, на которых достигается наименьшее значение целевой функции для точки g;

R_x – множество «свободных» координат x частицы, $R_x = \{i \in \{1,\ 2,\ \ldots,\ n\} \mid (i,\ j) \notin I_x\}$;

R_y – множество «свободных» координат y частицы, $R_y = \{i \in \{1,\ 2,\ \ldots,\ n\} \mid (i,\ j) \notin I_y\}$;

R_z – множество «свободных» координат z частицы, $R_z = \{i \in \{1,\ 2,\ \ldots,\ n\} \mid (i,\ j) \notin I_z\}$;

$\upsilon^\gamma = ((\upsilon^\gamma_{xi_x} \mid i_x \in R_x),\ (\upsilon^\gamma_{yi_y} \mid i_y \in R_y),\ (\upsilon^\gamma_{zi_z} \mid i_z \in R_z))$ – вектор смещения γ-й частицы (ее скорость), γ = 1, 2, ..., m;

ω – коэффициент, учитывающий влияние на вектор смещения его значения на предыдущем шаге (инерция);

φ_p – коэффициент влияния лучшего положения отдельно взятой частицы γ на ее вектор смещения (не зависит от индекса γ);

φ_g – коэффициент влияния наилучшего найденного положения g среди всех частиц роя на вектор смещения каждой из них;

r_p, r_g – вспомогательные коэффициенты, рассматриваемые как равномерно распределенные на интервале [0, 1] случайные величины;

$\overline{\eta}$ – предельное число итераций алгоритма;

T_{CPU} – процессорное время работы алгоритма, с;

$\overline{T}_{CPU}$ – максимально допустимое время работы алгоритма, с;

$[x^i_{\min}, x^i_{\max}]$ – отрезки возможных значений $X^\gamma_{лск}$, i = 1, 2, …, n, γ = 1, 2, ..., m;

$[y^i_{\min}, y^i_{\max}]$ – отрезки возможных значений $Y^\gamma_{лск}$, i = 1, 2, …, n, γ = 1, 2, ..., m;

$[z^i_{\min}, z^i_{\max}]$ – отрезки возможных значений $Z^\gamma_{лск}$, i = 1, 2, …, n, γ = 1, 2, ..., m.

Прямое произведение отрезков $[x^i_{\min},\ x^i_{\max}]\times[y^i_{\min},\ y^i_{\max}]$ должно включать все возможные значения координат начал ЛСК, для которых координаты $x^\gamma_i+\Delta^l_{xi},\ y^\gamma_i+\Delta^l_{yi}$ всех точек контура i-й детали содержатся внутри сектора загрузочной позиции поворотного стола, i = 1, 2, …, n, l = 1, 2, ..., $N_{конт}$. Отрезки $[z^i_{\min},\ z^i_{\max}]$ определяются пределами допустимой высоты приспособления для детали.

Сокращение размеров отрезков $[x^i_{\min},\ x^i_{\max}]$, $[y^i_{\min},\ y^i_{\max}]$, определяющих пространство поиска, позволяет повысить эффективность алгоритма решения задачи. Определение этих отрезков зависит от числа $N_{поз}$ позиций поворотного стола АС, координат $\Delta^l_{xi},\ \Delta^l_{yi}$ точек контура i-й детали в ЛСК детали, l = 1, 2, ..., $N_{конт}$, и расположения начала координат ЛСК. Возможные подходы к такому сокращению предложены в [7].

Описываемый ниже алгоритм является модификацией алгоритма PSO [7] для размещения одной детали на поворотном столе АС, предназначенного для обработки однородных партий деталей.

Алгоритм PSO

Итерация η = 0

1. Генерируются m точек $p^{\gamma(0)}=(X^{\gamma(0)},\ Y^{\gamma(0)},\ Z^{\gamma(0)})$, $\gamma=1,\ ...,\ m$, начал ЛСК деталей, координаты $x^\gamma_{i_x}$, $y^\gamma_{i_y}$, $z^\gamma_{i_z}$ которых равномерно распределены на заданных отрезках $[x^{i_x}_{\min},\ x^{i_x}_{\max}]$, $i_x\in R_x$, $[y^{i_y}_{\min},\ y^{i_y}_{\max}]$, $i_y\in R_y$, $[z^{i_z}_{\min},\ z^{i_z}_{\max}]$, $i_z\in R_z$, и $x^{i_x}_{лск}=x^{j_x}_{лск}+x^{i_x}_{лск}(0)-x^{j_x}_{лск}(0)$, $(i_x, j_x)\in I_x$, $y^{i_y}_{лск}=y^{j_y}_{лск}+y^{i_y}_{лск}(0)-y^{j_y}_{лск}(0)$, $(i_y, j_y)\in I_y$, $z^{i_z}_{лск}=z^{j_z}_{лск}+z^{i_z}_{лск}(0)-z^{j_z}_{лск}(0)$, $(i_z, j_z)\in I_z$, соответственно.

2. За положение $\overline{p}^\gamma$ частицы γ принимается его начальное значение:

$$\overline{p}^\gamma=(\overline{X}^\gamma,\ \overline{Y}^\gamma,\ \overline{Z}^\gamma)=(X^{\gamma(0)},\ Y^{\gamma(0)},\ Z^{\gamma(0)}),\ \gamma=1,\ ...,\ m.$$

3. Определяется начальное значение $\upsilon^{\gamma(0)}=((\upsilon^{\gamma(0)}_{xi_x}\,|\,i_x\in R_x),\ (\upsilon^{\gamma(0)}_{yi_y}\,|\,i_y\in R_y),\ (\upsilon^{\gamma(0)}_{zi_z}\,|\,i_z\in R_z))$ вектора скорости частицы γ посредством выбора случайных равномерно распределенных чисел из $\left[-(x^{i_x}_{\max}-x^{i_x}_{\min}),(x^{i_x}_{\max}-x^{i_x}_{\min})\right]$, $\left[-(y^{i_y}_{\max}-y^{i_y}_{\min}),(y^{i_y}_{\max}-y^{i_y}_{\min})\right]$, $\left[-(z^{i_z}_{\max}-z^{i_z}_{\min}),(z^{i_z}_{\max}-z^{i_z}_{\min})\right]$, γ = 1, 2, ..., m, соответственно.

4. Если частица $\overline{p}^\gamma$ допустима хотя бы для одной пары $q\in\overline{Q}$, $b\in\mathbf{B}$, то вычисляется значение целевой функции $F(\tilde{q},\ \tilde{b},\ \overline{p}^\gamma)=\min\{F(q, b, (X^{\gamma(0)}, Y^{\gamma(0)}, Z^{\gamma(0)}))|\ q\in\overline{Q},\ b\in\mathbf{B}\}$, иначе полагается $F(\tilde{q},\ \tilde{b},\ \overline{p}^\gamma)=\infty$, $\gamma=1,\ ...,\ m$.

5. Среди всех значений $F(\tilde{q},\ \tilde{b},\ \overline{p}^\gamma)$ находится наименьшее и определяются соответствующая точка $g=(\hat{X}^{\gamma^*},\ \hat{Y}^{\gamma^*},\ \hat{Z}^{\gamma^*})$ и соответствующие ей векторы q^g и b^g:

$$F(q^g, b^g, g)=\min\{F(\tilde{q},\ \tilde{b},\ \overline{p}^{\gamma(0)})|\ \gamma=1, 2, ..., m\};$$
$$(q^g, b^g, g)=\arg\{F(q^g, b^g, g)\}.$$

η = 1.

Итерация η

1. Для каждой частицы γ = 1, 2, ..., m:

1.1. Генерируются случайные числа r_p и r_g из интервала [0, 1].

1.2. Вычисляется новая скорость $\upsilon^{\gamma(\eta)}=((\upsilon^{\gamma(\eta)}_{xi_x}\,|\,i_x\in R_x),\ (\upsilon^{\gamma(\eta)}_{yi_y}\,|\,i_y\in R_y),\ (\upsilon^{\gamma(\eta)}_{zi_z}\,|\,i_z\in R_z))$ частицы γ по формулам

$$\upsilon^{\gamma(\eta)}_{xi_x}=\omega\cdot\upsilon^{\gamma(\eta-1)}_{xi_x}+\varphi_p\cdot r_p\cdot(\overline{x}^\gamma_{i_x}-x^\gamma_{i_x})+\varphi_g\cdot r_g\cdot(\hat{x}^\gamma_{i_x}-x^\gamma_{i_x});$$

$$\upsilon^{\gamma(\eta)}_{yi_y}=\omega\cdot\upsilon^{\gamma(\eta-1)}_{yi_y}+\varphi_p\cdot r_p\cdot(\overline{y}^\gamma_{i_y}-y^\gamma_{i_y})+\varphi_g\cdot r_g\cdot(\hat{y}^\gamma_{i_y}-y^\gamma_{i_y});$$

$$\upsilon^{\gamma(\eta)}_{zi_z}=\omega\cdot\upsilon^{\gamma(\eta-1)}_{zi_z}+\varphi_p\cdot r_p\cdot(\overline{z}^\gamma_{i_z}-z^\gamma_{i_z})+\varphi_g\cdot r_g\cdot(\hat{z}^\gamma_{i_z}-z^\gamma_{i_z}).$$

1.3. Определяется новое положение $p^{\gamma(\eta)} = (X^{\gamma(\eta)}, Y^{\gamma(\eta)}, Z^{\gamma(\eta)})$ частицы γ переносом ее из положения $p^{\gamma(\eta-1)}$ на итерации η–1 на вектор $\upsilon^{\gamma(\eta)}$:

$$x_{i_x}^{\gamma(\eta)} = x_{i_x}^{\gamma(\eta-1)} + \upsilon_{xi_x}^{\gamma(\eta)},\ i_x \in R_x;$$

$$y_{i_y}^{\gamma(\eta)} = y_{i_y}^{\gamma(\eta-1)} + \upsilon_{yi_y}^{\gamma(\eta)},\ i_y \in R_y;$$

$$z_{i_z}^{\gamma(\eta)} = z_{i_z}^{\gamma(\eta-1)} + \upsilon_{zi_z}^{\gamma(\eta)},\ i_z \in R_z;$$

$$x_{i_x}^{\gamma(\eta)} = x_{j_x}^{\gamma(\eta)} + x_{лск}^{i_x}(0) - x_{лск}^{j_x}(0),\ (i_x, j_x) \in I_x;$$

$$y_{i_y}^{\gamma(\eta)} = y_{j_y}^{\gamma(\eta)} + y_{лск}^{i_y}(0) - y_{лск}^{j_y}(0),\ (i_y, j_y) \in I_y;$$

$$z_{i_z}^{\gamma(\eta)} = z_{j_z}^{\gamma(\eta)} + z_{лск}^{i_z}(0) - z_{лск}^{j_z}(0),\ (i_z, j_z) \in I_z.$$

1.4. Если точка $(X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})$ допустима хотя бы для одной пары q, b, то вычисляется значение $F(q^{\gamma(\eta)},\ b^{\gamma(\eta)}, (X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})) = \min\{F(q, b, (X^{\gamma(\eta)}, Y^{\gamma(\eta)}, Z^{\gamma(\eta)}) \mid q \in \bar{Q},\ b \in \mathbf{B})\}$, иначе полагается $F(q^{\gamma(\eta)},\ b^{\gamma(\eta)}, (X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})) = \infty$.

1.5. Если $F(q^{\gamma(\eta)},\ b^{\gamma(\eta)},\ (X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})) < F(\tilde{q}^{\gamma},\ \tilde{b}^{\gamma},\ \bar{p}^{\gamma})$, то полагается $\bar{p}^{\gamma} = (X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})$, $\tilde{q}^{\gamma} = q^{\gamma(\eta)}$, $\tilde{b}^{\gamma} = b^{\gamma(\eta)}$ и $F(\tilde{q}^{\gamma},\ \tilde{b}^{\gamma},\ \bar{p}^{\gamma}) = F(q^{\gamma(\eta)},\ b^{\gamma(\eta)}, (X^{\gamma(\eta)}, Y^{\gamma(\eta)}, Z^{\gamma(\eta)}))$.

1.6. Если $F(\tilde{q}^{\gamma},\ \tilde{b}^{\gamma},\ \bar{p}^{\gamma}) < F(q^{g},\ b^{g},\ g)$, то полагается $g = \bar{p}^{\gamma}$, $q^{g} = \tilde{q}^{\gamma}$, $b^{g} = \tilde{b}^{\gamma}$ и $F(q^{g}, b^{g}, g) = F(\tilde{q}^{\gamma},\ \tilde{b}^{\gamma},\ \bar{p}^{\gamma})$.

2. Если выполняется любое из условий $\eta = \bar{\eta}$ либо $T_{CPU} \geq \bar{T}_{CPU}$, то переход к п. 3. Иначе полагается $\eta = \eta + 1$ и переход к п. 1.

3. Если $F(q^{g}, b^{g}, g) < \infty$, то (q^{g}, b^{g}, g) – решение задачи. В противном случае решение задачи не найдено.

Таким образом, итерации повторяются до тех пор, пока не выполнится хотя бы одно из условий остановки работы алгоритма: либо выполнится заданное количество итераций, либо истечет максимально допустимое время работы алгоритма.

В процессе работы алгоритма целесообразно проверять допустимость ограничений задачи последовательно, что повышает его эффективность за счет исключения лишних проверок заведомо недопустимых точек.

Также для повышения эффективности алгоритма недопустимые точки следует разделить на два типа: строго недопустимые, для которых не существует компоновочной схемы АС, и условно недопустимые, которые могут быть трансформированы в допустимые. Относительно условно недопустимых точек используются следующие способы их трансформации в допустимые:

1. Пусть для сгенерированного вектора $(X^{\gamma(\eta)},\ Y^{\gamma(\eta)},\ Z^{\gamma(\eta)})$ не выполняются ограничения (4)–(7).

Обозначим через dx^i, dy^i и dz^i искомые отклонения от значений $x_{лск}^i$, $y_{лск}^i$ и $z_{лск}^i$ соответственно. Отклонения, обеспечивающие их минимальную общую сумму при выполнении ограничений (4)–(7), могут быть получены в результате решения следующей задачи:

$$\sum_{i=1}^{n} |\, dx^i \,| + \sum_{i=1}^{n} |\, dy^i \,| + \sum_{i=1}^{n} |\, dz^i \,| \to \min; \tag{14}$$

$$y_{лск}^i + \Delta_{yi}^l + dy^i - (x_{лск}^i + \Delta_{xi}^l + dx^i) \cdot \mathrm{tg}(\pi / N_{поз}) \leq 0,\ l = 1, 2, \ldots,\ N_{конт}^i,\ i = 1, 2, \ldots, n; \tag{15}$$

$$y_{лск}^i + \Delta_{yi}^l + dy^i + (x_{лск}^i + \Delta_{xi}^l + dx^i) \cdot \mathrm{tg}(\pi / N_{поз}) \leq 0,\ l = 1, 2, \ldots,\ N_{конт}^i,\ i = 1, 2, \ldots, n; \tag{16}$$

$$dx^i - dx^j = 0,\ (i, j) \in I_x; \tag{17}$$

$$dy^i - dy^j = 0,\ (i, j) \in I_y; \tag{18}$$

$$dz^i - dz^j = 0,\ (i, j) \in I_z; \tag{19}$$

$$\max\{dx^{i(p_l)} + x_{лск}^{i(p_l)} + x(p_l) + w(p_l)/2 \mid l = 1,\ \ldots,\ k_j\} -$$

$$-\min\{dx^{i(p_l)} + x_{лск}^{i(p_l)} + x(p_l) - w(p_l)/2 \mid l = 1,\ \ldots,\ k_j\} \le d_j,\ j = 1, 2, \ldots, |P|,\ P_j \in P^x; \tag{20}$$

$$\max\{dz^{i(p_l)} + z_{лск}^{i(p_l)} + z(p_l) + w(p_l)/2 \mid l = 1,\ \ldots,\ k_j\} -$$

$$-\min\{dz^{i(p_l)} + z_{лск}^{i(p_l)} + z(p_l) - w(p_l)/2 \mid l = 1,\ \ldots,\ k_j\} \le d_j,\ j = 1, 2, \ldots, |P|,\ P_j \in P^z. \tag{21}$$

Задача (14)–(21) может быть преобразована в задачу линейного программирования путем введения дополнительных переменных δx^i, δy^i, δz^i, замены целевой функции (14) функцией

$$\sum_{i=1}^{n} \delta x^i + \sum_{i=1}^{n} \delta y^i + \sum_{i=1}^{n} \delta z^i \to \min, \tag{22}$$

введения дополнительных ограничений

$$\delta x^i - dx^i \ge 0,\ \ \delta x^i + dx^i \ge 0,\ i = 1, 2, \ldots, n; \tag{23}$$

$$\delta y^i - dy^i \ge 0,\ \ \delta y^i + dy^i \ge 0,\ i = 1, 2, \ldots, n; \tag{24}$$

$$\delta z^i - dz^i \ge 0,\ \ \delta z^i + dz^i \ge 0,\ i = 1, 2, \ldots, n, \tag{25}$$

а также замены ограничений (20), (21) соответственно следующими ограничениями:

$$dx^{i(p_{k'})} + x_{лск}^{i(p_{k'})} + x(p_{k'}) + \lambda(p_{k'})/2 - x_{лск}^{i(p_{k''})} - x(p_{k''}) - dx^{i(p_{k''})} + \lambda(p_{k''})/2 \le d_j,$$
$$p_{k'}, p_{k''} \in P_j,\ k', k'' \in \{1, 2, \ldots, k_j\},\ k' < k'',\ (k', k'') \notin I_x,\ j = 1, 2, \ldots, |P|,\ P_j \in P; \tag{26}$$

$$dx^{i(p_{k'})} + x_{лск}^{i(p_{k'})} + x(p_{k'}) + \lambda(p_{k'})/2 - x_{лск}^{i(p_{k''})} - x(p_{k''}) - dx^{i(p_{k''})} + \lambda(p_{k''})/2 \le d_j,$$
$$p_{k'}, p_{k''} \in P_j,\ k', k'' \in \{1, 2, \ldots, k_j\},\ k' > k'',\ (k', k'') \notin I_x,\ j = 1, 2, \ldots, |P|,\ P_j \in P^x; \tag{27}$$

$$dz^{i(p_{k'})} + z_{лск}^{i(p_{k'})} + z(p_{k'}) + \lambda(p_{k'})/2 - z_{лск}^{i(p_{k''})} - z(p_{k''}) - dz^{i(p_{k''})} + \lambda(p_{k''})/2 \le d_j,$$
$$p_{k'}, p_{k''} \in P_j,\ k', k'' \in \{1, 2, \ldots, k_j\},\ k' < k'',\ (k', k'') \notin I_z,\ j = 1, 2, \ldots, |P|,\ P_j \in P^z; \tag{28}$$

$$dz^{i(p_{k'})} + z_{лск}^{i(p_{k'})} + z(p_{k'}) + \lambda(p_{k'})/2 - z_{лск}^{i(p_{k''})} - z(p_{k''}) - dz^{i(p_{k''})} + \lambda(p_{k''})/2 \le d_j,$$
$$p_{k'}, p_{k''} \in P_j,\ k', k'' \in \{1, 2, \ldots, k_j\},\ k' > k'',\ (k', k'') \notin I_z,\ j = 1, 2, \ldots, |P|,\ P_j \in P^z. \tag{29}$$

Решая преобразованную задачу (22)–(29), можно получить из недопустимой относительно ограничений (4)–(7) точки ($x_{лск}^i$, $y_{лск}^i$, $z_{лск}^i$), $i = 1, \ldots, n$, допустимую ($x_{лск}^i + dx^i$, $y_{лск}^i + dy^i$, $z_{лск}^i + dz^i$), $i = 1, \ldots, n$.

Если задача (22)–(29) не имеет решения, то не имеет решения и исходная задача (1)–(13).

Следует отметить, что ограничения (26)–(29) для $(k', k'') \in I_x$ и $(k', k'') \in I_z$ выполняются тогда и только тогда, когда они выполняются для начальных положений $(x_{лск}^i(0), y_{лск}^i(0), z_{лск}^i(0))$, $i = 1, 2, \ldots, n$. Таким образом, эти ограничения могут быть проверены заранее до решения задачи (1)–(13).

2. Пусть для фиксированных векторов ($X_{лск}$, $Y_{лск}$, $Z_{лск}$), q и b пересекаются две шпиндельные коробки на соседних горизонтальных силовых столах, расположенных на гранях r и r+1 центральной станины (рис. 2), в их положении в конце обработки (не выполняется усло-

вие (12)). Это нарушение может быть устранено либо за счет изменения циклограммы станка, если это позволяет цикловое время (требуемая производительность), либо за счет увеличения длин баз b_r, b_{r+1} инструментов силовых узлов соответствующих боковых приставок. В первой ситуации пересечение будет устранено посредством разведения коробок по времени нахождения в конечном положении каждой из коробок. В другой ситуации увеличение базы инструментов (если оно допустимо) может привести к увеличению конструктивных параметров узлов соответствующей приставки (например, хода силового стола), что, как правило, приводит к увеличению его типоразмера и, соответственно, массы. Но при этом вектор ($X_{лск}$, $Y_{лск}$, $Z_{лск}$) может стать допустимым.

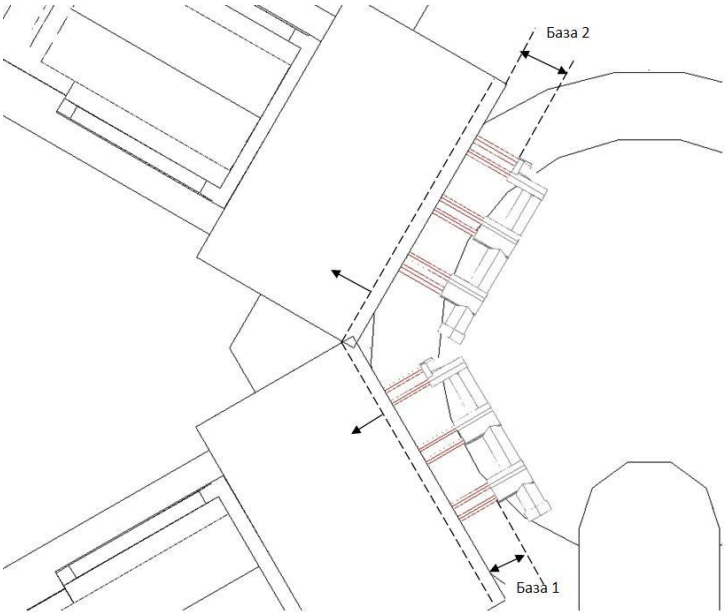


Рис. 2. Пересечение шпиндельных коробок соседних боковых приставок в проекции «в плане» при их положении в конце обработки. Штриховыми линиями указаны размеры требуемого увеличения величин базы инструментов для устранения пересечения

Замечание. Наличие пересечения шпинделей соседних многошпиндельных коробок указывает на недопустимость вектора ($X_{лск}$, $Y_{лск}$, $Z_{лск}$), поскольку оно может быть устранено только изменением положения деталей.

В следующем разделе приведены результаты вычислительных экспериментов по решению реальных проектных задач при различных значениях управляющих параметров.

4. Результаты численных экспериментов

Описанный в разд. 3 алгоритм PSO реализован на языке C++. Эксперименты проводились на ПК с ОС Windows 7 на базе процессора Intel Core с частотой 1,86 ГГц и оперативной памятью 4 ГБ. Характеристики 61 решаемой задачи представлены в табл. 1, где $K = \{k_1, k_2, \ldots, k_{|P|}\}$, {} – пустое множество.

Таблица 1

Характеристики задач

Задача	n	$N_{гр.}$	N_1	N_2	ℵ	Q	$\|I_x\|$	$\|I_y\|$	$\|I_z\|$	$\|P\|$	K	P_x	P_z
1	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
2	10	4	2	0	{1,3}	{}	4	3	3	0	{}	{}	{}
3	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
4	10	6	4	0	{1,2,3,4}	{}	4	3	3	0	{}	{}	{}
5	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}

Продолжение табл. 1

Задача	n	$N_{гр.}$	N_1	N_2	$\aleph$	Q	$\|I_x\|$	$\|I_y\|$	$\|I_z\|$	$\|P\|$	K	P_x	P_z
6	10	6	5	0	{1,2,3,4,5}	{}	4	3	3	0	{}	{}	{}
7	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
8	10	6	3	0	{1,2,3}	{}	4	3	3	0	{}	{}	{}
9	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
10	10	4	2	0	{1,2}	{}	4	3	3	0	{}	{}	{}
11	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
12	10	6	5	0	{1,2,3,4,5}	{}	4	3	3	0	{}	{}	{}
13	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
14	10	4	2	0	{1,2}	{}	4	3	3	0	{}	{}	{}
15	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
16	10	6	3	0	{1,2,3}	{}	4	3	3	0	{}	{}	{}
17	6	4	3	0	{1,2,3}	{}	2	5	2	1	{5}	{}	{1}
18	6	6	4	0	{1,2,3,4}	{}	0	0	0	0	{}	{}	{}
19	6	6	3	0	{1,3,4}	{}	2	5	2	1	{5}	{}	{1}
20	6	6	4	0	{1,3,4,5}	{}	0	0	0	0	{}	{}	{}
21	4	4	2	1	{1,2}	{3}	2	2	2	0	{}	{}	{}
22	6	8	3	0	{2,4,5}	{}	2	5	2	1	{5}	{}	{1}
23	6	8	4	0	{2,4,5,6}	{}	0	0	0	0	{}	{}	{}
24	6	4	2	1	{1,2}	{3}	2	2	2	0	{}	{}	{}
25	5	6	4	0	{1,2,3,4}	{}	3	4	3	1	{5}	{1}	{}
26	5	4	3	0	{1,2,3}	{}	3	4	3	1	{4}	{1}	{}
27	5	4	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
28	5	4	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
29	5	6	4	0	{1,2,3,4}	{}	3	4	3	1	{5}	{1}	{}
30	5	4	3	0	{1,2,3}	{}	3	4	3	1	{4}	{1}	{}
31	5	4	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
32	5	4	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
33	5	4	3	0	{1,2,3}	{}	3	4	3	1	{4}	{}	{1}
34	5	4	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
35	5	4	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
36	10	6	5	0	{1,2,3,4,5}	{}	6	8	7	1	{9}	{1}	{}
37	10	4	2	0	{1,3}	{}	4	3	3	0	{}	{}	{}
38	5	6	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
39	5	6	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
40	5	6	4	0	{1,2,3,4}	{}	3	4	3	1	{5}	{1}	{}
41	5	6	3	0	{1,2,3}	{}	3	4	3	1	{4}	{1}	{}
42	5	6	4	0	{1,2,3,4}	{}	3	4	3	1	{5}	{1}	{}
43	5	6	3	0	{1,2,3}	{}	3	4	3	1	{4}	{1}	{}
44	5	6	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
45	5	6	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
46	5	6	4	0	{1,2,3,4}	{}	3	4	3	1	{5}	{1}	{}
47	5	6	3	0	{1,2,3}	{}	3	4	3	1	{4}	{1}	{}
48	5	6	2	0	{1,2}	{}	1	1	1	0	{}	{}	{}
49	5	6	3	0	{1,2,3}	{}	3	3	3	0	{}	{}	{}
50	6	6	2	1	{1,2}	{3,4,5}	2	2	2	0	{}	{}	{}
51	6	6	2	1	{1,2}	{3,4,5}	1	1	1	0	{}	{}	{}
52	6	6	2	1	{1,2}	{3,4,5}	2	2	2	0	{}	{}	{}
53	6	6	2	1	{1,2}	{3,4,5}	2	2	2	0	{}	{}	{}
54	6	6	2	1	{4,5}	{1,2,3}	2	2	2	0	{}	{}	{}
55	6	6	2	1	{4,5}	{1,2,3}	1	1	1	0	{}	{}	{}

Окончание табл. 1

Задача	n	$N_{гр.}$	N_1	N_2	$\aleph$	Q	$\lvert I_x\rvert$	$\lvert I_y\rvert$	$\lvert I_z\rvert$	$\lvert P\rvert$	K	P_x	P_z
56	6	6	2	1	{3,4}	{1,2,5}	2	2	2	0	{}	{}	{}
57	6	6	2	1	{3,4}	{1,2,5}	2	2	2	0	{}	{}	{}
58	6	4	2	1	{1,3}	{2}	2	2	2	0	{}	{}	{}
39	6	4	2	1	{1,3}	{2}	1	1	1	0	{}	{}	{}
60	6	4	2	1	{1,2}	{3}	2	2	2	0	{}	{}	{}
61	6	4	2	1	{1,2}	{3}	2	2	2	0	{}	{}	{}

Задачи с одинаковыми параметрами соответствуют различным вариантам технологического процесса.

Эксперименты проводились при различных значениях управляющих параметров. В табл. 2 приведены результаты для m = 10, $\overline{\eta}$ = 50, $\overline{T}_{CPU}$ = 600 с. В этой таблице F_0 – значение целевой функции (масса станка в кг) для начальных положений ($x^i_{лск}(0)$, $y^i_{лск}(0)$, $z^i_{лск}(0)$) ЛСК деталей i = 1, 2, …, n; F_1^* – найденное значение целевой функции для ω = 0,729, φ_p = 1,49445, φ_g = 1,49445 [14]; F_2^* – для ω = 0,729, φ_p = 2,05, φ_g = 2,05 [15]; F_3^* – для ω = 0,5, φ_p = 1,5, φ_g = 3,5 [7]; F_4^* – для ω = 0,5, φ_p = 4, φ_g = 1,5 [7] с корректировкой сгенерированных координат (решение задачи (22)–(29)) и выбором вектора b. Значения F_5^* соответствуют выбору вектора b без корректировки сгенерированных координат, а F_6^* – корректировке сгенерированных координат без выбора вектора b для ω = 0,729, φ_p = 1,49445, φ_g = 1,49445. Значения $F_{СП}^*$ и $F_{ЛП}^*$ соответствуют значениям, полученным при использовании алгоритмов случайного поиска и ЛП-последовательностей для 500 положений деталей с корректировкой сгенерированных координат и выбором вектора b.

Жирным шрифтом выделены наилучшие значения массы станка.

Таблица 2

Результаты решения задач

Задача	F^0	F_1^*	F_2^*	F_3^*	F_4^*	F_5^*	F_6^*	$F_{СП}^*$	$F_{ЛП}^*$
1	–	17155,9	**17001,7**	17155,9	17097,6	17249,6	17155,9	17272,6	17230,4
2	–	**7813,41**	**7813,41**	**7813,41**	**7813,41**	–	**7813,41**	**7813,41**	**7813,41**
3	20479,2	17428,4	17447,7	17357,2	18921	18824,5	**17305,8**	17852,6	17854,9
4	–	**10756,2**	–	–	–	–	–	13594,6	14291,7
5	–	17375	**17347,5**	17374,9	17376,8	17561,5	17356,8	17870,1	19010,5
6	–	12535,3	**12312,8**	12935,8	13793,5	-	13840,6	13793,5	13793,5
7	–	18761,8	18707,9	**18638,4**	18652,5	19398,9	18761,8	19101,9	18892,9
8	–	**12305,9**	–	–	–	–	–	15024,4	13367,8
9	16769,3	16251,4	**16245,5**	16280,5	16257,5	16622,7	16251,4	16448,6	16454
10	–	**9486,13**	9556,1	9556,1	9556,1	–	**9486,13**	9556,1	9556,1
11	–	**18842,7**	19196,9	18968,8	18958,8	18925,7	**18842,7**	19574,5	19746,9
12	–	12563,1	12494,1	**12491**	–	–	–	15171,4	15353,6
13	16769,3	16251,4	**16245,5**	16280,5	16257,5	16622,7	16251,4	16448,6	16454
14	–	**9676,54**	11505,9	11110,8	–	–	–	12116,1	14200,6
15	–	18757	18703,2	**18633,6**	18647,7	19394,1	18757	19097,1	18888,1
16	–	**12444,7**	**12444,7**	**12444,7**	**12444,7**	–	–	**12444,7**	**12444,7**
17	–	6213,37	6634,92	6696,24	**6154,19**	8992,34	6213,37	7038,45	6938,07
18	–	12037,7	13069,9	–	13325,2	–	–	11428,1	**10814,6**
19	–	8150,34	6758,94	**6679,19**	7221,25	8202,53	8150,34	7525,12	6891,17
20	–	12042,7	13069,9	–	13325,2	–	–	11433,1	**10819,6**
21	–	11762,9	**10664,6**	10921,3	11767,5	17988,2	12785	19993,3	–

Продолжение табл. 2

Задача	F^0	F_1^*	F_2^*	F_3^*	F_4^*	F_5^*	F_6^*	$F_{СП}^*$	$F_{ЛП}^*$
22	–	9954,67	8730,99	10280,5	**8651,45**	–	9954,67	8825,38	8748,23
23	–	**11664,9**	–	–	–	–	–	15217,8	12560,7
24	–	19255,3	19323,1	20451	–	–	**18900,2**	21160,9	21555,1
25	16130,4	**14310,9**	14442,8	14444	14492,8	14379	14363	14621,1	14692,6
26	13173	**12287**	12287,1	12287,2	12288,1	12288	12287	12302,2	12291
27	7750,3	7078,17	7025,73	**6407,38**	7078,17	7094,02	7078,17	7447,09	7389,73
28	9813,63	**7594,72**	**7594,72**	**7594,72**	7636,1	7594,72	**7594,72**	7762,55	8210,5
29	16130,4	**14396,1**	14469	14444	14492,8	14379	14363	14621,1	14692,6
30	13173	**12287**	12287,1	12287,2	12288,1	12288	12287	12302,2	12291
31	7750,3	**6402,95**	7122,51	6423,97	7201,15	7094,02	7078,17	7447,09	7389,73
32	7737,98	**7594,72**	**7594,72**	**7594,72**	**7594,72**	7631,88	**7594,72**	7737,98	7737,98
33	7211,67	**6992,09**	**6992,09**	**6992,09**	**6992,09**	**6992,09**	**6992,09**	7039	7107,53
34	8193,46	**6549,11**	7142,44	7380,92	8143,71	7488,34	7365,95	7578,39	7521,03
35	7689,26	7662,5	7655,06	7662,38	7655,06	7661,16	**7655,06**	7689,3	7689,3
36	–	17372	**17211,3**	17211,4	17212,1	17561,5	17368,1	17670,9	17607,5
37	7872,1	**7865,04**	7872,1	7872,1	7872,1	7872,1	7872,1	7872,1	7872,1
38	–	7840,27	7840,27	7840,27	7840,27	8157,4	**7770,68**	7840,27	7840,27
39	7248,96	**7191,15**	7196,57	7196,57	7196,57	7196,57	7248,96	7196,57	7196,57
40	–	14392,5	14449,3	14469,3	14508,1	14400,9	**14360,7**	14655,1	14510,7
41	–	11012,5	11012,5	11012,8	11012,6	11013	11012,5	11026,9	11034,5
42	–	14490,3	14441,8	14386,9	14975,5	14799,5	**14360,7**	14655,1	14510,7
43	14394,7	**14264,1**	14264,5	14264,3	14264,7	14264,8	**14264,1**	14286,9	14303,4
44	–	11147,6	11149,5	11147,6	**7117,61**	–	–	7899,29	8000,18
45	–	7773,19	7773,19	7773,2	**7773,18**	7773,86	7812,91	7834,44	7849,04
46	17565,9	14430,7	14454,5	14444	14492,8	15779,5	**14363**	14621,1	14692,6
47	14442,4	12913,4	12913	12936,1	12919,2	**12911,8**	12913,4	12951,7	12966,8
48	–	**7005,72**	7011,18	7019,88	**7005,72**	–	7008,97	7063,8	7063,8
49	–	7742,46	7742,47	7742,46	**7742,45**	7746,9	7746,9	7805,76	7820,59
50	20183,8	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**
51	19907,6	19907,6	19738,1	19451,1	**18460**	19907,6	19907,6	19907,6	19907,6
52	16845,8	16845,8	16845,8	**16394,9**	16845,8	16845,8	16845,8	16845,8	16845,8
53	15207,6	**13729,9**	13792,1	15063,8	15207,6	15207,6	15207,6	15207,6	15207,6
54	20183,8	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**	**20183,8**
55	19907,6	19035,4	19457,9	18927,3	**18459,9**	19907,6	19907,6	19573	19907,6
56	16608,1	16608,1	16533,5	**16143,3**	16608,1	16608,1	16608,1	16608,1	16514,5
57	15207,5	**13671,3**	13792,1	15063,8	15207,6	15207,6	13719,8	15207,6	15207,6
58	17605,3	**17605,3**	**17605,3**	**17605,3**	**17605,3**	**17605,3**	**17605,3**	**17605,3**	**17605,3**
59	18888,2	**18888,2**	**18888,2**	**18888,2**	**18888,2**	**18888,2**	**18888,2**	**18888,2**	**18888,2**
60	17196,8	**17196,8**	**17196,8**	**17196,8**	**17196,8**	**17196,8**	**17196,8**	**17196,8**	**17196,8**
61	15597,6	**14070,1**	15597,6	15597,6	15597,6	–	15597,6	15597,6	15597,6

В табл. 3 приводятся результаты сравнительного анализа эффективности протестированных алгоритмов. Здесь NSOL – число решенных задач; NOPT – число решенных задач с наилучшим значением массы; NOPT1 – число решенных задач с абсолютно наилучшим значением массы; АО – среднее отклонение массы полученного решения от наилучшего, %; MNO – минимальное отклонение массы полученного решения от наилучшего, %; МХО – максимальное отклонение массы полученного решения от наилучшего, %; АОТВ – среднее отклонение массы полученного решения от наилучшего без учета максимального отклонения, %; MNO1 – минимальное (большее нуля) отклонение массы полученного решения от наилучшего, %; МХО1 – второе большее отклонение массы полученного решения от наилучшего, %.

Таблица 3

Сравнение результатов решения задач

METH	NSOL	NOPT	NOPT1	AO	MNO	MXO	AOTB	MNO1	MXO1
PSO1	61	29	12	2,72	0,00	56,62	1,82	0,16	22,03
PSO2	58	19	7	3,41	0,00	56,65	2,48	0,37	20,85
PSO3	56	17	7	3,11	0,00	56,62	2,14	0,16	18,83
PSO4	55	18	7	3,35	0,00	24,35	2,96	0,17	23,21
PSO5	46	8	1	5,96	0,00	68,67	4,56	0,11	46,12
PSO6	52	23	7	2,93	0,00	22,03	2,55	0,35	19,88
СП	61	7	0	7,07	0,00	87,47	5,73	0,12	30,46
ЛП	60	9	2	5,45	0,00	46,75	4,75	0,20	32,87

Только два алгоритма смогли решить все задачи: PSO с параметрами $\omega = 0{,}729$, $\varphi_p = 1{,}49445$, $\varphi_g = 1{,}49445$ (PSO1) и СП. При этом СП ни разу не получил решение, лучшее по сравнению с другими алгоритмами, а для PSO1 число таких задач составило 12. Кроме того, для PSO1 получено наибольшее число наилучших решений и минимальное среднее отклонение значения массы АС по сравнению с другими алгоритмами.

Для пяти задач (50, 54, 58, 59, 60) ни один из алгоритмов не смог улучшить значение целевой функции, полученное для исходных положений деталей.

В табл. 4 приводятся результаты попарного сравнения PSO1 с другими алгоритмами. Характеристики NSOL, NOPT, NOPT1, AO, MXO, AOTB, MNO1 и MXO1 вычислены для сравниваемых алгоритмов. Первое значение в ячейке соответствует характеристике PSO1, а второе – характеристике алгоритма из столбца METH.

Таблица 4

Сравнение PSO1 с другими алгоритмами

METH	NSOL	NOPT	NOPT1	AO	MXO	AOTB	MNO1	MXO1
PSO2	61/58	43/32	29/18	0,86/1,43	20,59/18,91	0,53/1,12	0,16/0,11	14,02/11,24
PSO3	61/56	46/29	32/15	0,86/1,49	22,03/14,82	0,51/1,24	0,41/0,18	10,47/12,70
PSO4	61/55	48/26	35/13	1,62/2,15	56,62/24,35	0,71/1,74	0,34/0,43	15,06/12,47
PSO5	61/46	58/12	49/3	0,00/4,36	0,12/52,92	0,00/3,28	0,12/0,22	0,02/44,73
PSO6	61/52	51/37	24/10	0,09/1,36	1,88/12,47	0,06/1,14	0,10/0,35	0,90/10,86
СП	61/61	56/15	46/5	1,19/5,51	41,12/69,97	0,53/4,43	5,33/0,12	12,80/30,46
ЛП	61/60	55/16	45/6	1,55/4,51	39,34/46,75	0,92/3,79	0,53/0,14	18,27/32,87

Результаты экспериментов показали, что лучшим алгоритмом в среднем является PSO1. Его можно рекомендовать использовать в случае ограниченного времени для поиска варианта компоновки АС. Если же ресурс времени достаточный, то целесообразно применять наряду с PSO1 и другие варианты метода PSO (PSO2 – PSO6).

Заключение

В работе исследована задача оптимизации массы АС с поворотным столом для многоинструментальной обработки группы различных деталей за счет их размещения на загрузочной позиции станка, выбора граней расположения боковых приставок и величин баз инструментов. Выбор грани для размещения боковой приставки, предназначенной для обработки деталей сверху параллельно на нескольких позициях стола, осуществляется полным перебором граней, не занятых горизонтальными приставками. Для оптимизации размещения деталей группы предложен эвристический алгоритм «роя частиц» и подобраны наилучшие параметры этого алгоритма для ряда тестовых примеров. Проведено экспериментальное сравнение предложенного метода с известными методами СП и ЛП-последовательностей, показавшее его превосходство в среднем по значению целевой функции.

В дальнейшем предполагается расширить постановку задачи на случай возможных различных ориентаций ЛСК деталей относительно системы координат АС, а также учесть возможность обработки деталей боковыми приставками вдоль произвольной оси, а не только параллельно осям системы координат АС.

Список литературы

1. Гебель, Х. Компоновка агрегатных станков и автоматических линий / Х. Гебель. – М. : Машгиз, 1959. – 189 с.

2. Хомяков, В.С. Автоматизированное проектирование компоновок металлообрабатывающих станков / В.С. Хомяков, И.И. Давыдов // Станки и инструмент. – 1990. – № 5. – С. 4–7.

3. Аверьянов, О.И. Модульный принцип построения станков с ЧПУ / О.И. Аверьянов. – М. : Машиностроение, 1987. – 232 с.

4. Комплекс алгоритмов и программных средств формирования компоновок агрегатных металлорежущих станков / Г.М. Левин [и др.] – Минск, 2005. – 50 с. – (Препринт / Нац. академия наук Беларуси, Объед. ин-т проблем информатики НАН Беларуси ; № 4).

5. Автоматизация проектирования агрегатных металлорежущих станков : учеб. пособие / Г.М. Левин [и др.] ; Витебский гос. технол. ун-т. – Витебск, 2008. –122 с.

6. Розин, Б.М. К синтезу компоновок агрегатных станков для групповой обработки / Б.М. Розин, В.Е. Зданович // Информатика. – 2014. – № 4(44). – С. 100–116.

7. Гущинский, Н.Н. Оптимизация размещения детали на многопозиционном поворотном столе агрегатного станка / Н.Н. Гущинский, В.Е. Зданович, Б.М. Розин // Информатика. – 2015. – № 4. – С. 57–72.

8. Скобцов, Ю.А. Метаэвристики / Ю.А. Скобцов, Е.Е. Федоров – Донецк : Ноулидж, 2013. – 426 с.

9. Пантелеев, А.В. Методы оптимизации в примерах и задачах : учеб. пособие / А.В. Пантелеев, Т.А. Летова. – 2-е изд. – М. : Высш. шк., 2005. – 544 с.

10. Соболь, И.М. Выбор оптимальных параметров в задачах со многими критериями / И.М. Соболь, Р.Б. Статников. – М. : Дрофа, 2006. – 175 с.

11. Poli, R. Analysis of the Publications on the Applications of Particle Swarm Optimisation / R. Poli // J. of Artificial Evolution and Applications. – 2008. – Vol. 2008. – 57 p.

12. Shahrajabian, H. Multi-constrained optimization in ball-end machining of carbon fiber-reinforced epoxy composites by PSO / H. Shahrajabian, M. Farahnakian // Cogent Engineering. – 2015. – Vol. 2, iss. 1. – 14 p.

13. Wei, L. A particle swarm optimization approach to a multi-objective reconfigurable machine tool design problem / L. Wei, L. Ming // Proc. of the 2006 IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, 16–21 July 2006 / IEEE Press. – Vancouver, 2006. – P. 2222–2229.

14. Clerc, M. The particle swarm – explosion, stability, and convergence in a multidimensional complex space / M. Clerc, J. Kennedy // IEEE Transactions on Evolutionary Computation. – 2002. – Vol. 6. – P. 58–73.

15. Pedersen, M.E.H. Simplifying particle swarm optimization / M.E.H. Pedersen, A.J. Chipperfield // Applied Soft Computing. – 2010. – Vol. 10, no. 2. – P. 618–628.

Поступила 06.01.2017

Объединенный институт проблем
информатики НАН Беларуси,
Минск, ул. Сурганова, 6
e-mail: gyshin@newman.bas-net.by,
rozin@newman.bas-net.by

N.N. Guschinsky, B.M. Rozin

OPTIMIZING THE PLACEMENT OF A BATCH OF WORK-PIECES AT A MULTI-POSITION ROTARY TABLE OF TRANSFER MACHINE

The problem of minimizing the mass of rotary transfer machine by placing a batch of work-pieces at the rotary table is considered. To solve this problem the mathematical model and heuristic PSO algorithm are proposed. The results of numerical experiments for series of real problems are reported. The experiments revealed that the PSO algorithm on average is more effective for the solution of the problem compared to methods of random search and LP-search.

УДК 65.011.56; 007.51

И.В. Шатохин

КОРПОРАТИВНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ МАТЕРИАЛЬНО-ТЕХНИЧЕСКИМ ОБЕСПЕЧЕНИЕМ

Предлагается подход к построению автоматизированной централизованной системы управления материально-техническим обеспечением предприятий корпорации. Рассматриваются функциональные требования к средствам автоматизации для различных этапов соответствующего бизнес-процесса: заявочной кампании, организации закупок и их контрактации, планирования и организации отгрузок материально-технических ресурсов в адрес предприятий корпорации.

Введение

Многие предприятия на сегодняшний день объединены в корпорации с централизованной службой материально-технического обеспечения (ЦСМТО), которая имеет центральную базу производственно-технического обеспечения, включающую складской комплекс. Запасы, хранимые на центральной базе, числятся на балансе головной организации холдинга. Предприятия, входящие в корпорацию (далее – дочерние организации), также имеют собственные склады с запасами материально-технических ресурсов (МТР), но которые числятся уже на балансе самих дочерних организаций.

Цикл работ по материально-техническому обеспечению начинается с этапа расчета потребностей дочерних организаций в МТР по различным направлениям деятельности (ремонтно- и производственно-эксплуатационные нужды, капитальное строительство и др.) на определенный плановый период с детализацией по службам главных специалистов, объектов строительства, ремонта и т. д. Обычно плановый период – это следующий год с разбивкой по кварталам и месяцам. Рассчитанную потребность назовем брутто-потребностью. Далее определяется нетто-потребность путем вычитания из брутто-потребности объемов запасов на складах, которые могут быть вовлечены в счет ее обеспечения, и объемов закупки МТР, осуществляемых дочерними организациями самостоятельно в соответствии с корпоративными регламентами. В зависимости от нетто-потребности формируются заявки на МТР в ЦСМТО, которые проходят процесс согласования на соответствие лимитам финансирования. Источники их покрытия определяются за счет наличных запасов центральной базы материально-технического обеспечения, неиспользованных запасов, находящихся на балансе дочерних организаций, закупки у сторонних поставщиков.

На требуемые объемы закупки профильными специалистами ЦСМТО формируются заказные спецификации, которые передаются в подразделение, отвечающее за организацию и проведение торгов в соответствии с национальным законодательством [1] и корпоративными регламентами [2, 3]. По результатам торгов с победителями заключаются договоры на поставку МТР. Далее поступившие по договорам закупки от поставщиков МТР приходуются на склад и проводятся по регистрам бухгалтерского учета головной организации холдинга с последующей передачей их в дочерние организации по заявкам.

С течением времени дочерние организации в силу производственных обстоятельств пересчитывают свои потребности, что влечет за собой подачу ими в ЦСМТО корректирующих заявок на МТР (отмену или корректировку ранее поданных заявок в сторону уменьшения или увеличения объемов потребности в МТР, появление новых потребностей), которые требуют проведения всего вышеописанного цикла работ. В связи с этим основной задачей ЦСМТО является своевременное и полное обеспечение заявок дочерних организаций на МТР, решить которую можно благодаря автоматизированной системе управления бизнес-процессом (рис. 1).

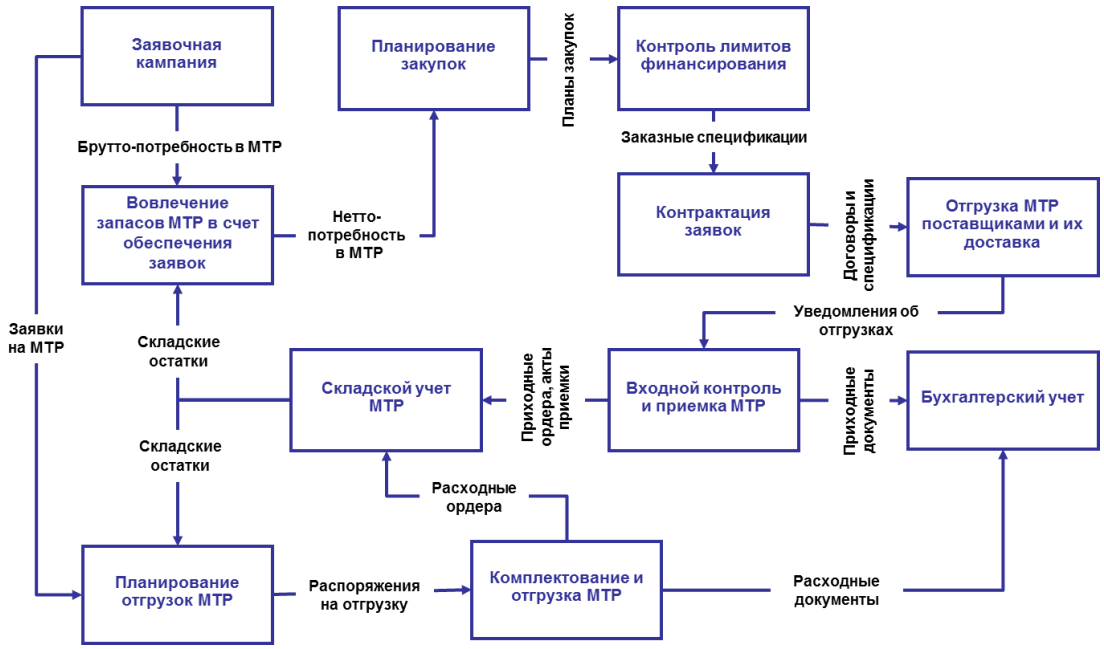


Рис. 1. Схема бизнес-процесса материально-технического обеспечения

1. Заявочная кампания

Определим, что под заявкой на МТР понимается заявка дочерней организации на МТР одного наименования в определенном количестве и с определенным сроком поставки, характеризующаяся уникальным идентификационным кодом (UIN). Другими словами, это минимальная единица информации, вводимая пользователями в период заявочной кампании, которой в дальнейшем оперирует система автоматизированного управления. Чаще всего ввод заявок в базу данных производится удаленно через веб-портал или путем импорта данных из Excel-таблиц. Выбор способа ввода зависит от организационно-территориальной структуры и технической готовности заказчика. На данном этапе обеспечивается возможность объединения заявок одной дочерней организации, если требуется комплектная поставка различных МТР.

Для введенных заявок автоматически генерируется ее идентификационный код по формуле

$$UIN = \text{префикс} + \text{предыдущий номер} + 1,$$

где префикс – год, на который подается заявка; предыдущий номер – наибольший номер заявки, ранее зарегистрированной в базе данных. Нумерация заявок в новом году начинается с единицы.

На основе введенных заявок в автоматизированной системе управления материально-техническим обеспечением формируется регистр, назначением которого является накопление информации о текущем состоянии заявок по всем этапам их жизненного цикла (рис. 2). Жизненный цикл заявки состоит из следующих основных этапов:

– согласования;
– вовлечения наличных запасов МТР в счет ее обеспечения;
– контрактации;
– отгрузки МТР поставщиком;
– приемки и входного контроля качества МТР;
– оприходования поступивших МТР на склад и их хранения;
– отгрузки МТР в адрес дочерней организации со склада.

Средства системы поддерживают расширение перечня этапов, отображаемых в регистре заявок, в зависимости от специфики и требований автоматизируемого предприятия за счет внутреннего механизма настройки шаблона регистра. В частности, такая необходимость возни-

кает при решении задач комплектации объектов строительства, где дополнительно контролируются этапы передачи МТР подрядчикам, их использования и списания в производстве.

Каждому этапу ставятся в соответствие определенные хозяйственные операции. Факт операции фиксируется вводом в систему подтверждающего ее электронного документа определенного типа, например спецификации договора, уведомления об отгрузке, приходной накладной, складского ордера и т. п. Номенклатурные позиции товарной спецификации электронного документа должны быть связаны с соответствующими заявками на МТР. Это позволяет рассчитать в регистре заявок значения показателей, отражающих степень исполнения заявки на каждом из этапов.

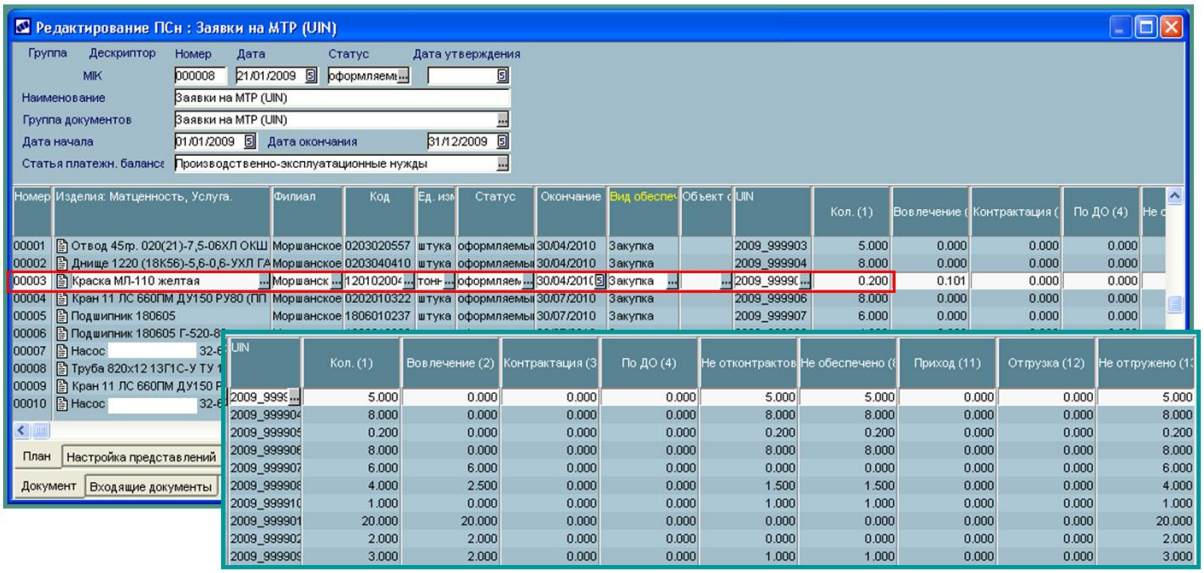


Рис. 2. Регистр заявок

На стадии планирования закупок производится расчет плановых цен закупки МТР по различным правилам: цене последнего прихода данного МТР; средней цене закупок данного МТР за определенный период; цене аналогичной заявки на данный МТР, введенной ранее, и др. Наличие функции расчета цен позволяет подготовить исходные данные для автоматизированного контроля лимитов финансирования закупок МТР по соответствующим статьям бюджета расходов предприятия, что является одним из важнейших этапов согласования заявок.

Согласование заявок обеспечивается встроенными в систему механизмами поддержки документооборота (doc-flow), передающими заявку в зависимости от ее статуса по определенному маршруту между ответственными пользователями. При этом существуют возможности внесения пользователями необходимой поясняющей информации и распорядительных указаний, а также перевода заявки в статус согласно принятому решению. Текущий статус заявки отражается в регистре заявок.

2. Контрактация и обеспечение заявок

Процесс начинается с вовлечения наличных складских запасов МТР в счет обеспечения заявок. Эта операция состоит в изменении аналитики целевого учета [4] невостребованных запасов под заявки прошлых лет, излишков запасов и других под заявки планируемого года. Целевой учет материальных запасов и их движения производится в разрезе уникальных номеров заявок (UIN) на МТР, направлений деятельности, потребителей МТР (дочерних организаций), объектов ремонта, строительства и т. д. Средства целевого учета обеспечивают:

– настройку типов объектов учета. Цель ее состоит в определении перечня аналитических признаков, каждый из которых сопоставляется с конкретным справочником информационных объектов системы (рис. 3);

– ввод объектов целевого учета необходимого типа в позициях товарных спецификаций электронных документов;

– документирование операций изменения аналитики складского учета запасов МТР, т. е. перевод свободных запасов в резерв под конкретные объекты целевого учета и обратно, а также изменение аналитического разреза резервирования запаса (перемещение резерва с одного объекта целевого учета на другой);

– копирование аналитики целевого учета запасов в бухгалтерские проводки, формируемые по первичным электронным документам согласно учетной политики предприятия.

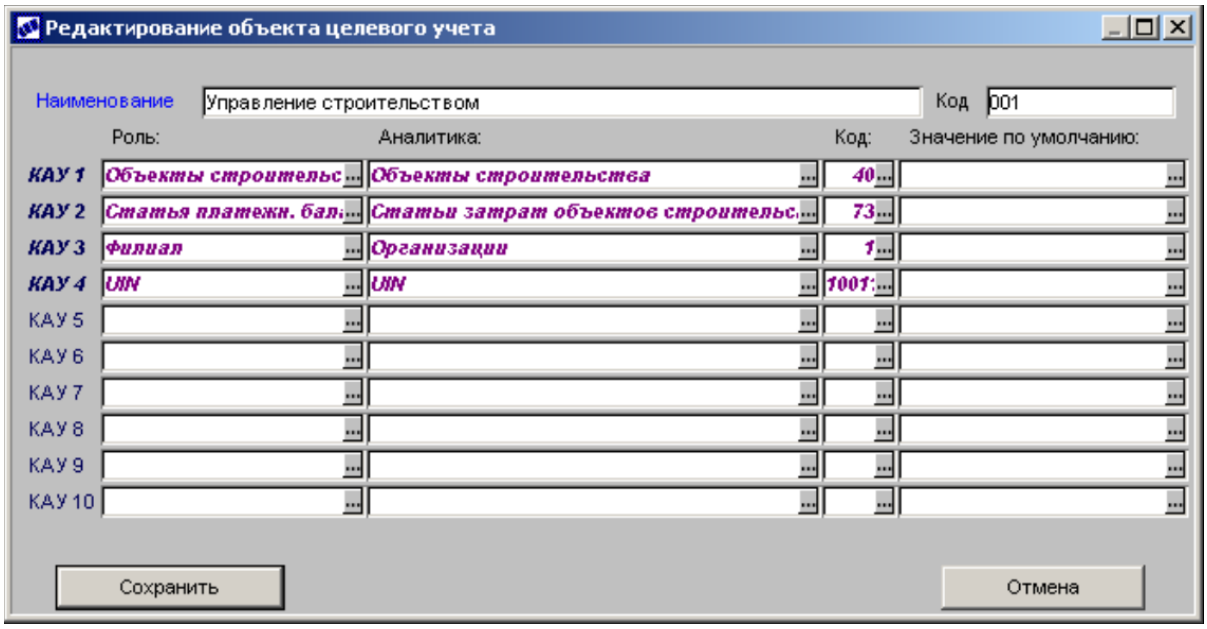


Рис. 3. Настройка типа объекта целевого учета

Операция вовлечения запасов, впрочем как и все последующие, осуществляется профильными специалистами ЦСМТО. Поэтому необходимым свойством системы является профилирование пользователей и поддержка их взаимозаменяемости. В результате профилирования каждый специалист службы снабжения работает только со своими профильными номенклатурными позициями [4]. Взаимозаменяемость подразумевает передачу прав работы с номенклатурой между пользователями, например, на время отпуска одного из них.

При вовлечении запасов может производиться замена заказанных МТР на их аналоги, которые отличаются от заявленных наименованием, номенклатурным номером или учетной единицей измерения и по своим физико-химическим свойствам применимы в хозяйственной деятельности предприятия вместо заявленного МТР. Для обеспечения работы с аналогами используется такой инструмент, как справочник замен. Он содержит информацию о том, какие МТР могут заменяться на их аналоги с указанием коэффициента автоматического пересчета количества. Однако в ряде случаев, связанных со спецификой свойств ряда номенклатурных позиций (например, металлопроката, трубной, кабельной продукции и т. п.), применение теоретических коэффициентов пересчета при замене материалов не имеет практического смысла. Поэтому инструментарий системы предоставляет пользователю возможность в конкретных случаях самостоятельно определять соотношения между количеством заказанного и заменяющего его материала исходя из опыта, знаний и здравого смысла [5]. Результаты вовлечения наличных запасов учитываются в регистре заявок пересчетом показателей обеспечения для соответствующего этапа жизненного цикла заявки, что позволяет рассчитать нетто-потребность, которая удовлетворяется за счет закупки.

Контрактация – процесс обеспечения заявок на МТР договорными объемами закупки и подтверждения их поставщиками. В процессе контрактации профильные специалисты ЦСМТО посредством функции системы агрегируют заявки на МТР в заказные спецификации.

На данном этапе пользователем определяется способ доставки грузов (самовывоз или с участием транспортно-экспедиционных организаций). Заказные спецификации передаются в подразделение, ответственное за организацию и проведение торгов в виде тендеров, конкурсов и т. д. Результаты проведения торгов фиксируются в информационной системе путем ввода данных из спецификаций заключенных договоров с привязкой их товарных позиций к заявкам на МТР. Система поддерживает возможность ведения договоров различных видов (прямых договоров закупки, агентских, консигнационных и др.), что влияет на последующую организацию и учет состояния взаиморасчетов с контрагентами за поставку МТР. В соответствии с оговоренным способом доставки обеспечивается формирование заказов на доставку и экспедирование грузов.

Своевременное начало работ по организации закупок с оповещением ответственного пользователя определяется системой исходя из типовой длительности циклов закупки (например, 30, 60, 90 или 180 дней) соответствующей номенклатуры. Точкой отсчета является требуемая дата исполнения заявки. Под длительностью цикла понимается промежуток времени с момента начала работ по организации закупки до момента поступления ее на склад предприятия. Типовая длительность цикла определяется обычно экспертным путем и складывается из норм времени на подготовку и проведение торгов, длительности производственного цикла поставщика и доставки грузов до склада заказчика.

Согласно жизненному циклу заявки на МТР средства системы обеспечивают учет отгруженных поставщиком и находящихся в пути товаров, формирование распоряжений складу на приемку МТР, их оприходование на склад в разрезе заявок на МТР с автоматическим расчетом соответствующих показателей в регистре заявок. Дополнительной опцией является формирование пропусков на транспортные средства для их допуска на территорию складского комплекса на разгрузку.

При создании автоматизированной системы управления следует учитывать, что на любом из этапов может возникать работа с аналогами МТР. В этом случае при связывании позиций спецификации любого хозяйственного документа (спецификации договора, уведомления об отгрузке, приходной накладной) с заявками на МТР требуется использовать инструменты, аналогичные применяемым на этапе вовлечения запасов. Необходимой функцией системы является закрытие заявок в случае поставки МТР с наличием отрицательного или положительного отклонения от заявленного количества, возможность которого определяется условиями поставки по договору (насыпные, наливные грузы) или свойствами номенклатуры (например, для трубной продукции подобное отклонение возникает по весу из-за наличия допусков на физические размеры трубы). Возникший при этом дефицит или профицит рассматривается системой в дальнейшем как штатная ситуация.

На всех этапах привязка позиций спецификации электронного документа осуществляется автоматически, если реквизиты позиции (наименование МТР, единица измерения и т. д.) совпадают с соответствующими реквизитами заявки на МТР. В противном случае она осуществляется вручную. Результатом формирования подобных связей документов между собой выступает автоматическое резервирование приходуемых в системе на склад МТР под заявки дочерних организаций, являющихся объектами целевого учета.

Важную роль в рамках бизнес-процесса управления материально-техническим обеспечением играет подсистема автоматизации складского учета. Наиболее существенными функциональными требованиями к ней в контексте заявленной проблематики являются:

– учет запасов по складам (основного или ответственного хранения);

– учет запасов, находящихся в подотчете материально-ответственных лиц;

– партионный учет МТР;

– учет МТР в нескольких единицах измерения;

– учет запасов в разрезе аналитики их целевого назначения (заявок на МТР (UIN), направлений деятельности, объектов ремонта и строительства, дочерних организаций и т. п.);

– учет запасов материалов и комплектующих изделий согласно их срокам годности и гарантийного хранения;

– упреждающая индикация окончания сроков годности МТР, гарантийных сроков их хранения или истечения сроков действия сертификатов поставщиков;

– ведение истории продлений гарантийных сроков хранения;

– оприходование покупных комплектов частями по мере их поступления. Под комплектом здесь понимается МТР одного наименования, состоящий из нескольких составных частей, доставка которых производится несколькими транспортными единицами и может быть распределена во времени;

– печать складских бирок;

– наличие средств управления статусами запасов, отражающих их текущее состояние;

– учет выработки ресурса для МТР многократного использования.

При поступлении МТР могут проходить входной контроль, результаты которого фиксируются переводом запасов в соответствующий статус. Забракованные МТР блокируются. Дальнейшая работа с ними обеспечивается функциями принятия решений, которые определяют алгоритмы (сценарии) последующей обработки данных. Примеры наиболее употребительных сценариев: использование (отклонения являются допустимыми), возврат поставщику, неиспользование до особого распоряжения, списание в металлолом.

В процессе обработки указанных пользователем сценариев система автоматически переводит запасы МТР в соответствующие статусы с формированием необходимых электронных документов (писем о приглашении представителей поставщика для освидетельствования и ремонта, накладных на возврат МТР поставщику, актов на списание и т. п.).

Организация отгрузок МТР со склада в дочерние организации производится на основании плана отгрузок, который формируется по регистру заявок на МТР с отражением плановых и фактических показателей исполнения заявок, а также их обеспеченности складскими запасами центральной базы производственно-технического обеспечения (рис. 4). Обеспеченность заявок на МТР определяется по соответствующей аналитике целевого учета складских запасов. На основе информации, представленной в плане, ответственные специалисты ЦСМТО оформляют требования на отгрузку МТР и резервируют под них складские остатки. Возможность включения в распоряжение тех или иных позиций из складских запасов определяется статусом последних. При создании распоряжений обеспечивается контроль комплектности отгрузки МТР, если в заявках дочерних организаций были соответствующие указания. На данном этапе существует возможность автоматизированного формирования пропусков на транспортные средства для их постановки под погрузку.

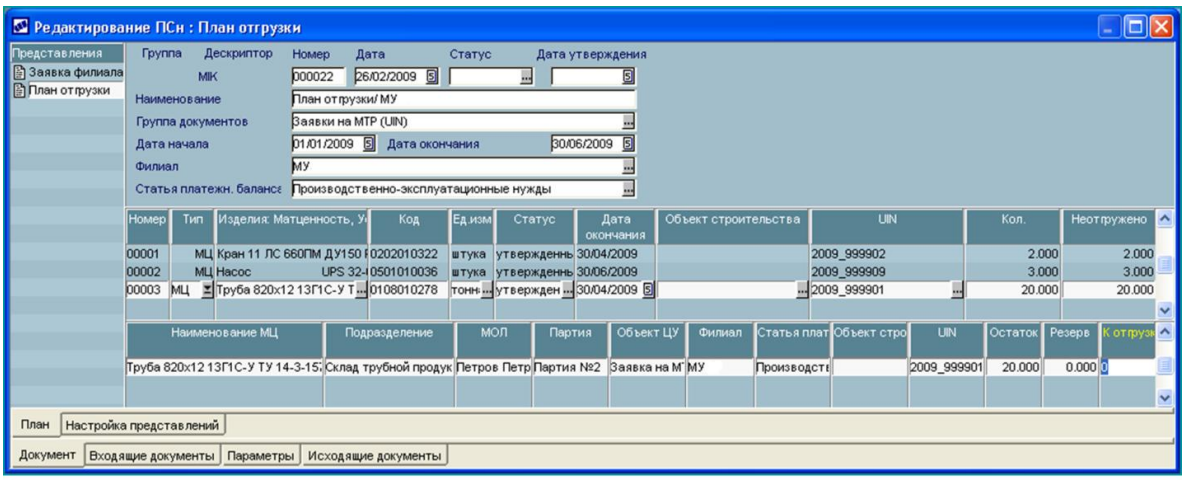


Рис. 4. План отгрузки МТР и его обеспеченность складскими запасами

Сформированные требования на отгрузку становятся доступны ответственному работнику склада, который организует комплектование и погрузку МТР в транспортные средства для их доставки заказчикам. Для распоряжений на отгрузку поддерживается механизм управления документооборотом. Это необходимо для оперативной корректировки распоряжения специалистом ЦСМТО в случае возникновения нештатных ситуаций на складе. Допустим, физический доступ на месте хранения к указанной в распоряжении партии МТР оказался заблокирован и нужно произвести отпуск из доступной партии. Для этого в распоряжение должны быть внесены соответствующие изменения сотрудником, оформившим документ.

Формирование товарно-транспортных накладных осуществляется в системе по распоряжению на отгрузку. Привязка позиций товарных спецификаций распоряжения и расходных накладных с заявками на МТР, а также пересчет соответствующих показателей в регистре заявок на МТР производятся автоматически.

Следует учесть, что в силу свойств отдельных видов номенклатуры отпуск МТР в дочерние организации может производиться сверх плановых объемов. Например, под заявки двух дочерних организаций было закуплено и принято к учету по 30 т труб, но в силу наличия положительного допуска на физические размеры часть труб оказалась тяжелее. В результате в адрес первого заказчика получается отпустить 32 т (из них 2 т за счет запасов второй организации). Подобная ситуация приводит к возникновению дефицита в обеспечении заявки второго заказчика. Для таких случаев в системе предусмотрена функция, которая производит переучет отпускаемого излишка в разрезе аналитики целевого учета запасов и информирование о возникновении дефицита специалистов ЦСМТС с целью принятия ими мер по его устранению.

Операция списания МТР (проведение расходной накладной) со склада приводит к пересчету фактических показателей плана отгрузки и отклонений от него.

По результатам деятельности ЦСМТС выполняется цветовая раскраска позиций, отражающая факт исполнения графика обеспечения заявок относительно требуемой даты их исполнения (таблица).

Пример раскраски

Цвет	Описание
Белый	Срок поставки МТР не наступил
Зеленый	Поставка МТР произведена вовремя
Желтый	До окончания срока поставки МТР осталось N дней. Значение N определяется настройкой системы
Красный	Поставка МТР произведена с задержкой

Сформированные в процессе первичные электронные документы являются основой для последующего автоматизированного формирования проводок по соответствующим счетам бухгалтерского учета.

Заключение

Рассмотренный в работе подход применялся при создании автоматизированных систем управления материально-техническим снабжением на базе корпоративной информационной системы Галактика ERP (Enterprise Recourses Planning) для предприятий нефтегазовой и горнодобывающей отраслей. В результате внедрения системы удалось сократить сверхнормативные и неликвидные складские запасы, высвободить оборотные средства предприятий, повысить исполнительскую дисциплину сотрудников служб снабжения, исключить дублирование закупок. Дочерние организации получили возможность оперативного контроля состояния исполнения своих заявок, например, через витрину данных, источником информации для которой служит регистр заявок. Рассмотренный бизнес-процесс может быть реализован и на базе иных ERP-систем. Цена вопроса при этом будет заключаться в объеме необходимых доработок их функционала для контроля жизненного цикла заявок в разрезе необходимых аналитических признаков. Достоинством выбранной ERP-платформы является наличие инструментария настройки целевого учета запасов. Он в значительной степени сокращает трудоемкость работ по созданию и внедрению автоматизированных систем управления материально-техническим обеспечением с учетом специфических особенностей конкретного заказчика.

Список литературы

1. О закупках товаров, работ, услуг отдельными видами юридических лиц : Федеральный закон Российской Федерации от 18 июля 2011 г. № 223-ФЗ // Российская газета. – 2011. – № 159. – С. 12.

2. Эффективное управление корпоративными закупками. Опыт РАО «ЕЭС России» / под. общ. ред. Г.А. Суходольского. – М. : Вершина, 2007. – 376 с.

3. Линдерс, М. Управление снабжением и запасами. Логистика / М. Линдерс, Х. Фирон ; пер. с англ. – СПб. : ООО «Полиграфуслуги», 2006. – 768 с.

4. Шатохин, И.В. Информационная система управления предприятием как цепочкой поставок // Информатика. – 2016. – № 4(52). – С. 20–26.

5. Шатохин, И.В. От заявки до результата / И.В. Шатохин // Белорусы и рынок. – 2011. – № 6(941). – С. 17.

Поступила 27.01.2017

УП «Топ Софт»,
Минск, ул. Сурганова, 28В
e-mail: IVS@galaktika.by

I.V. Shatokhin

THE CORPORATE INFORMATION SYSTEM OF LOGISTICS MANAGEMENT

The corporate information system of logistics management is discussed. Its main tasks and functions are described. The system is operated in several enterprises.

УДК 681.396.36

Д.И. Рабченок

МЕТОДИКА СИНТЕЗА ИНФОРМАЦИОННОЙ МОДЕЛИ БОЕВОЙ ОБСТАНОВКИ

Рассматривается методика структурно-параметрического синтеза информационной модели на командном пункте войск противовоздушной обороны. Используется метод диагональных матриц событий, позволяющий сформировать эффективные структуру, параметры и связи между элементами информационной модели. Разрабатываются динамические прототипы из элементов информационной модели. Проверяется адекватность информационной модели и оценивается ее эффективность при помощи интегрального показателя качества деятельности человека-оператора.

Введение

Одной из важнейших задач автоматизированного управления, направленной на повышение эффективности боевых действий и сокращение цикла управления, является формирование таких информационных моделей (ИМ), которые способны отражать реальные события в требуемом районе и при этом соответствуют задачам и возможностям оператора, а также обеспечивают оптимальный информационный баланс представляемой ему информации.

Разработка и совершенствование ИМ боевой обстановки являются достаточно трудоемкими задачами. Это обусловлено большим количеством неизвестных, взаимосвязанных и трудно формализуемых параметров. Кроме того, способы представления информации выбираются с целью облегчения выполнения оператором действий, а действия, в свою очередь, определяются способами представления информации.

1. Понятийный аппарат

Пропускная способность оператора характеризует его возможности по обработке поступающей информации в сложной человекомашинной организационно-технической системе. Для оценки пропускной способности оператора выбрана скорость выполнения им определенной последовательности действий как реакции на появление события. Под событием понимается появление отслеживаемого оператором признака изменения состояния описываемой ИМ системы.

Динамично меняющаяся обстановка характеризуется скоротечностью изменения ситуации в области информационного поля оператора, что, в свою очередь, требует от оператора высокой скорости обработки информации; существенно ограниченным временем анализа визуальной информации; высокой ответственностью оператора, особенно при управлении сложными системами, например автоматизированными системами управления войск противовоздушной обороны (АСУ ПВО).

Информационное поле – это часть ИМ, используемая в конкретный промежуток времени непосредственно для реализации определенной части алгоритма деятельности оператора.

Объем зрительного восприятия – количество элементов ИМ, которое одновременно попадает в зону, ограниченную углом зрения 10° в горизонтальной и вертикальной координатных плоскостях матрицы средства отображения информации.

Под ошибкой оператора понимается неправильное выполнение или невыполнение предписанных ему действий либо произвольное нарушение предписанного алгоритма деятельности. В общем случае ошибками можно назвать невыполнение требуемого или выполнение лишнего (несанкционированного) действия, неправильное или несвоевременное выполнение требуемого действия.

Состояние системы – это совокупность отслеживаемых оператором параметров описываемой ИМ системы, влияющая на выполнение оператором соответствующего алгоритма деятельности.

Стимул – сложный сигнал с несколькими опознавательными признаками.

2. Описание методики проектирования информационной модели

Решение задачи проектирования ИМ может быть достигнуто последовательным приближением ИМ к оптимальному варианту с оценкой качества проектирования на каждой итерации. Ниже рассмотрим основные положения методики решения данной задачи, которая представлена на рис. 1 в виде итерационного алгоритма.

Для определения перечня функций боевого управления и составления на его основе алгоритмов деятельности оператора предлагается использовать:

– существующие документы на образцы вооружения и военной техники (руководства по боевой работе, описания подсистем отображения информации, руководства по эксплуатации и т. п.);

– экспертные методы с привлечением специалистов по разработке и эксплуатации образцов вооружения и военной техники, для которых проектируется ИМ;

– современные рекомендации по результатам анализа специализированных источников информации.

После составления перечня функций боевого управления подбираются элементы визуализации ИМ для реализации алгоритмов деятельности оператора. Подбор целесообразно проводить исходя из сложившихся стереотипов в отрасли АСУ ПВО с привлечением экспертов.

Затем следует прототипирование статических отображений наборов элементов для соответствующих алгоритмов деятельности оператора. Прототипирование заключается в создании эскизов фрагментов ИМ, представляющих собой наборы групп объединенных элементов отображения-управления, которые необходимы для реализации оператором функций боевого управления. Таким образом формируется опорная структура ИМ в статике, подобная существующим ИМ и учитывающая сложившиеся в отрасли АСУ ПВО стереотипы.

Следующий этап проектирования ИМ направлен на совершенствование опорной структуры ИМ в сторону улучшения показателей качества последней.

Увеличение скорости переработки информации оператором возможно при условии уменьшения количества информации, предъявляемой последнему в процессе функционирования АСУ. В свою очередь, уменьшение количества информации подразумевает предварительный анализ критических ситуаций (событий), прогнозируемых в каждом алгоритме деятельности оператора, причин их возникновения, минимального набора элементов ИМ, необходимого для их устранения, а возможно, и целесообразных управляющих воздействий со стороны ЭВМ. Данный процесс целесообразно реализовывать методом диагональной матрицы событий [1] как хорошо себя зарекомендовавшим при решении задач подобного рода.

Анализ показал [2], что при выявлении событий, связанных с целевым назначением описываемой ИМ системы, необходимо использовать положения теории игр, при помощи которых устанавливается целевое назначение описываемой ИМ системы (например, не допустить нахождения цели в опасной близости от прикрываемого объекта для АСУ ПВО). Такой выбор связан с тем, что игровой подход в наибольшей степени удовлетворяет решению задачи построения ИМ АСУ в условиях, когда исходные данные носят преимущественно стохастический характер с известными или неизвестными (либо в принципе не существующими) законами распределения.

Результатом применения метода диагональной матрицы событий является общая структурная схема событий, позволяющая определить алгоритм деятельности оператора как его реакцию на любое из возникших в системе сложных событий и избежать трудоемких экспериментальных исследований деятельности оператора, связанных с инвариантностью его реакции на сложные события.

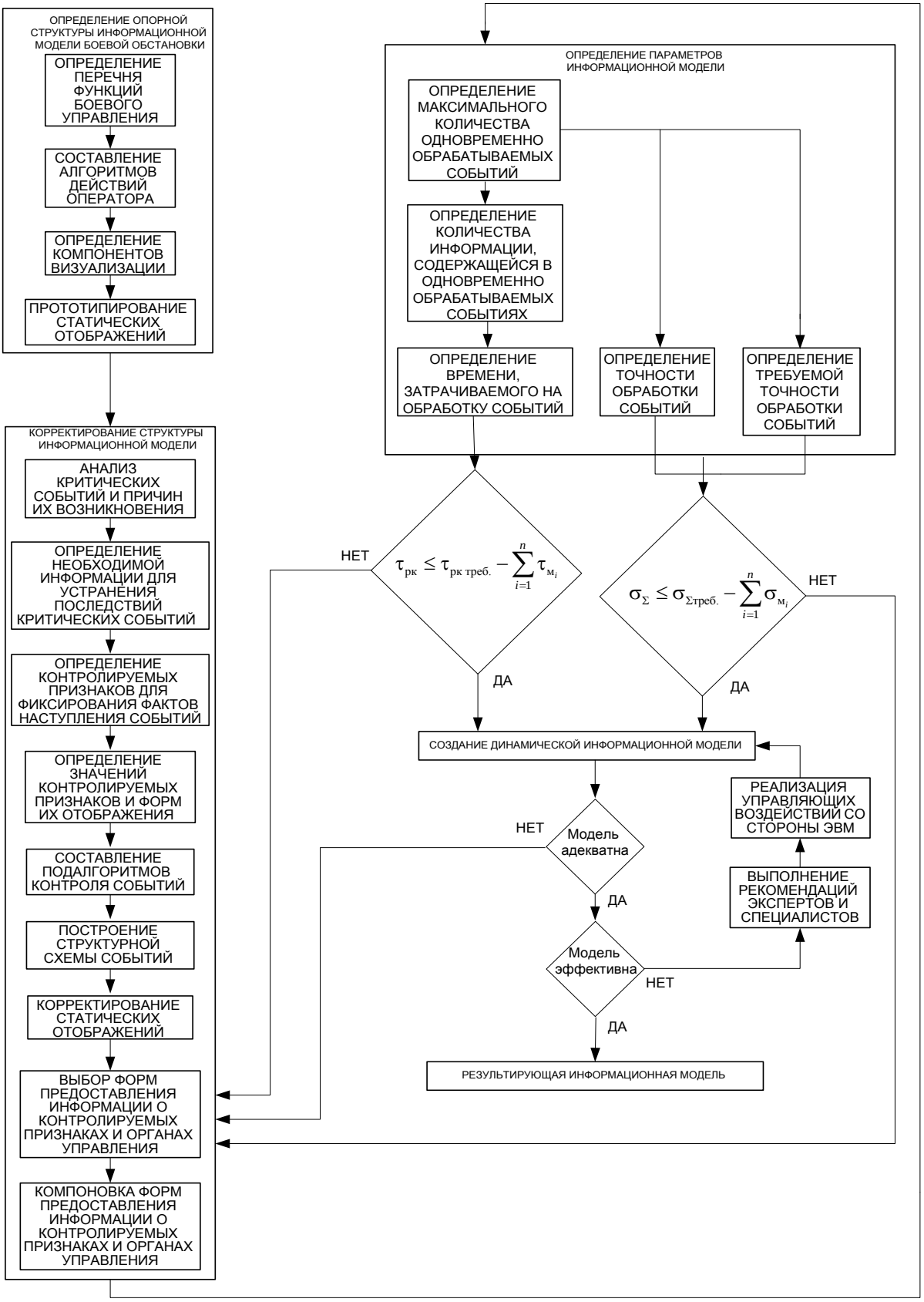


Рис. 1. Алгоритм решения задачи проектирования ИМ боевой обстановки

На основе общей структурной схемы событий корректируется опорная структура ИМ в статике в сторону уменьшения количества отображаемых элементов.

Выбор форм представления оператору информации о контролируемых признаках, а также органов управления для реализации управляющих воздействий на средствах отображения и управления ИМ может быть произведен согласно одному из следующих вариантов, приведенных в порядке возрастания трудозатрат:

– из существующей ИМ, использующейся на образце военной техники;

– в соответствии с действующими нормативно-техническими документами в области предъявления требований к системам отображения и управления в части, касающейся ИМ;

– из вновь созданного разработчиком, ранее неизвестного или известного, но не применявшегося в данном контексте набора форм с привлечением экспертов;

– комбинацией вышеуказанных вариантов.

Для оценки информационной нагрузки на оператора необходимо определить максимальное количество одновременно обрабатываемых оператором событий.

При установлении максимального количества событий, обрабатываемых оператором АСУ ПВО, надо исходить из предположения, что появление события, отображаемого ИМ, инициировано в первую очередь воздушной целью. Управление в такой системе должно осуществляться в соответствии с принципом гарантированного результата, который описывается критерием Вальда:

$$U^* = \max_{U \in \boldsymbol{U}} \min_{R \in \boldsymbol{R}} \left\{ J(X, X_f, U, R, T) \right\}, \qquad (1)$$

где в общем случае X – текущее состояние описываемой ИМ системы; X_f – требуемое конечное состояние данной системы ($f = \overline{1,F}$, F – множество состояний системы); U – выбранное управление; R – множество известных состояний внешней среды; T – длительность интервала времени, выделенного на завершение процесса управления; $\max\limits_{U \in \boldsymbol{U}}$ – наилучшая реализация управляющей стратегии со стороны оператора; $\min\limits_{R \in \boldsymbol{R}}$ – наихудшая для оператора реализация стратегии со стороны внешней среды.

Данный принцип является выражением стратегии крайнего пессимизма со стороны оператора, поскольку ориентирует на ситуацию, наихудшую из возможных.

Применительно к АСУ ПВО максимальное количество обрабатываемых событий соответствует наихудшей для оператора реализации стратегии со стороны природы $\min\limits_{R \in \boldsymbol{R}}$ (1), а наилучшей реализацией стратегии оператора $\max\limits_{U \in \boldsymbol{U}}$ в данном случае будет такая ответная реакция последнего, которая не приведет к возникновению критических ситуаций в описываемой ИМ системе.

Множество R определяется возникающими в системе событиями. Текущее состояние системы X соответствует возникновению в ней определенных событий, а требуемое конечное состояние системы X_f соответствует отсутствию в ней критических ситуаций. Для определения верхнего предела величины T целесообразно использовать модель массированного ракетно-авиационного удара предполагаемого противника как наиболее вероятную в рамках реализации противодействия современному агрессору, а также инструкции по боевому применению АСУ ПВО, для которой проектируется ИМ.

Параметры ИМ определяют время реакции оператора и точность его действий. Определение параметров предлагается производить при помощи аналитической модели взаимодействия оператора с ИМ на автоматизированном рабочем месте [3]. Рассмотрим последовательность действий в рамках синтеза ИМ:

1. Определение времени, затрачиваемого оператором на обработку события, в том числе и на серию ответных реакций.

Исходя из предположения, что этап контроля при решении задачи обработки информа-

ции оператором функционально осуществляет ЭВМ, время реакции оператора на событие определяется выражением

$$\tau_{\text{рк}} = \tau_{\text{в}} + \tau_{\text{р}} + \tau_{\text{п}} + \tau_{\text{м}},$$

где $\tau_{\text{в}}$ – время восприятия информации, $\tau_{\text{р}}$ – время принятия решения, $\tau_{\text{п}}$ – время поиска нужного органа управления, $\tau_{\text{м}}$ – время осуществления моторного акта.

2. Сопоставление времени обработки события с требуемым значением величины T формулы (1).

При помощи следующего выражения можно задать требования к пропускной способности человекомашинной системы через время обработки оператором информации:

$$\tau_{\text{рк}} \le \tau_{\text{рк треб.}} - \sum_{i=1}^{n} \tau_{\text{м}_i}, \qquad (2)$$

где $\tau_{\text{рк треб.}}$ – требуемое время обработки информации оператором, $\tau_{\text{м}_i}$ – время задержки информации в *i*-м звене машины, n – число машинных звеньев.

3. Определение точности действий оператора.

При независимости погрешности деятельности оператора и погрешности представления информации устройством отображения суммарная среднеквадратическая ошибка системы «оператор – устройство отображения» будет определяться как

$$\sigma_{\Sigma} = \sqrt{\sigma_{\text{уо}}^2 + \sigma_{\text{оп}}^2},$$

где $\sigma_{\text{уо}}$ – среднеквадратическая погрешность воспроизведения информации устройством отображения (обычно задается в пределах от 0,1 до 0,5 % [4]), $\sigma_{\text{оп}}$ – среднеквадратическая погрешность деятельности оператора.

Величина $\sigma_{\text{уо}}$ ввиду своей незначительности существенного вклада в суммарную среднеквадратическую ошибку человекомашинной системы не вносит; кроме того, она может быть значительно уменьшена за счет применения более прецизионных средств отображения информации. Величина $\sigma_{\text{оп}}$ вносит наиболее весомый вклад в суммарную ошибку системы «оператор – устройство отображения», при этом она не может быть менее некоторого минимального значения $\sigma_{\text{оп min}}$.

Выражение для $\sigma_{\text{оп}}$ включает погрешности, связанные с восприятием информации, принятием решения, выбором органа управления, управляющим воздействием, и имеет вид

$$\sigma_{\text{оп}} = \sqrt{\sigma_{\text{с}}^2 + \sigma_{\text{л}}^2 + \sigma_{\text{оу}}^2 + \sigma_{\text{мт}}^2},$$

где среднеквадратическая погрешность $\sigma_{\text{с}}$ обусловлена ошибкой восприятия оператором информации, $\sigma_{\text{л}}$ связана с ошибкой оператора при принятии им решения, $\sigma_{\text{оу}}$ – с ошибкой выбора органа управления, $\sigma_{\text{мт}}$ – с ошибочными действиями оператора при осуществлении управляющих воздействий. Выражение справедливо при независимых слагаемых подкоренного выражения, что подтверждается статистическими исследованиями, посвященными оценке качества деятельности оператора [5].

4. Определение требуемой точности действий оператора.

Требуемую точность действий оператора предлагается определять при помощи методики оценки эффективности АСУ по качеству [6] как наиболее родственной с учетом следующего.

Процессы целераспределения и формирования данных, целеуказания и его отработки огневыми средствами представляют собой наиболее существенный этап управления ходом боевых действий, влияющий в максимальной степени не только на оперативность, но и на боевую

эффективность этих действий. Вероятность успешного целеуказания количественно характеризует его качество. В свою очередь, вероятность успешного целеуказания $P_{\text{цу}}$ зависит от вероятности безошибочной обработки информации на командном пункте $P_{\text{обр.}}$, принятия командным пунктом правильного решения на боевые действия $P_{\text{реш.}}$, безошибочного наведения огневого средства на цель $P_{\text{нав.}}$, обнаружения цели боевым расчетом $P_{\text{обн.}}$.

Тогда имеет смысл выражение

$$P_{\text{цу}} = P_{\text{обр.}} \cdot P_{\text{реш.}} \cdot P_{\text{нав.}} \cdot P_{\text{обн.}},$$

которое справедливо при условии независимости четырех составляющих, определяющих успешность целеуказания.

Полагая, что обрабатываемая на командном пункте информация поступает из единого информационного поля, а средство отображения информации достаточно прецизионно, можно считать, что $P_{\text{обр.}} \approx 1$. Также будем полагать, что оператор имеет достаточные опыт и навыки в принятии решений на боевые действия и необходимая для принятия решения информация присутствует в ИМ, и примем $P_{\text{реш.}} \approx 1$. Вполне обоснованно можно принять $P_{\text{обн.}} \approx 1$, так как рубежи целеуказания определяются после обнаружения и завязки трасс целей. Таким образом, $P_{\text{цу}} \approx P_{\text{нав.}}$, при этом

$$P_{\text{нав.}} \approx P_{\varepsilon} \cdot P_{\beta} \cdot P_{\text{Д}},$$

где P_{ε} и P_{β} – вероятности попадания цели в сектор поиска по углу места и азимуту соответственно, $P_{\text{Д}}$ – вероятность попадания цели в строб по дальности. Данное выражение будет справедливо для независимых сомножителей.

Можно провести аналогию между процессом наведения огневого средства, состоящим из операций поиска в заданной области пространства и отождествления цели, с одной стороны, и поиском с отождествлением объектов ИМ оператором – с другой.

Как отмечалось ранее, процессы наведения огневого средства и поиска цели оператором имеют родственную природу. В первом случае используется алгоритм поиска, который является упорядоченным перебором элементов пространства. Во втором случае также происходит упорядоченный перебор элементов ИМ по определенному алгоритму поиска. Поэтому можно заменить угломестную и азимутальную координаты наведения огневого средства вертикальной и горизонтальной координатами матрицы средства отображения информации соответственно. При этом координата по дальности при двухмерной системе позиционирования информации на экране не учитывается. Тогда выражение для вероятности успешного обнаружения оператором искомого элемента ИМ будет иметь вид

$$P_{\text{обн.опер.}} \approx P_{\text{верт.}} \cdot P_{\text{гориз.}}.$$

Для двухмерных случаев перемещения оператором курсора манипулятора между прямыми или искривленными границами области информационного поля ИМ целесообразно использовать наименьшее из значений размеров поля для перемещения по горизонтали или вертикали. На сегодняшний день большинство средств отображения информации военного назначения и прямоугольного исполнения имеют горизонтальную ориентацию, т. е. вертикальная составляющая меньше горизонтальной. Запишем выражение для вероятности успешного обнаружения оператором искомого элемента ИМ:

$$P_{\text{обн.опер.}} \approx K_{\text{пропорц.}} \cdot P^2_{\text{верт.}},$$

где $K_{\text{пропорц.}}$ – коэффициент пропорциональности сторон средства отображения информации (для прямоугольных матриц). В случае вертикальной ориентации средства отображения информации данная формула примет вид

$$P_{\text{обн.опер.}} \approx K_{\text{пропорц.}} \cdot P^2_{\text{гориз.}}.$$

Предположим, что ошибка поиска в вертикальной плоскости распределена по нормальному закону. Тогда выражение для нахождения $P_{\text{верт.}}$ примет вид

$$P_{\text{верт.}} = \int_{-\Delta\varphi_\varepsilon}^{+\Delta\varphi_\varepsilon} \frac{1}{\sqrt{2\pi}\cdot\sigma_\varepsilon}\cdot e^{\frac{(\varepsilon-m_\varepsilon)^2}{\sigma_\varepsilon^2}}\partial\varepsilon = \Phi\left(\frac{\Delta\varphi_\varepsilon - m_\varepsilon}{\sigma_\varepsilon}\right) + \Phi\left(\frac{\Delta\varphi_\varepsilon + m_\varepsilon}{\sigma_\varepsilon}\right),$$

где ε – координата в вертикальной плоскости, m_ε – систематическая ошибка в вертикальной плоскости, σ_ε – среднеквадратическая ошибка в вертикальной плоскости, $\Phi(u)$ – функция Лапласа.

В обобщенном виде функцию $P_{\text{верт.}}$ можно построить, используя выражение

$$P_{\text{верт.}} = \Phi\left(\frac{l - m_\varepsilon}{\sigma_\varepsilon}\right) + \Phi\left(\frac{l + m_\varepsilon}{\sigma_\varepsilon}\right),$$

где l – полуширина доверительного интервала изменения координаты ε.

В этом случае функция $P_{\text{верт.}}$ при $m_\varepsilon = K_1 \cdot l$, $\sigma_\varepsilon = K_2 \cdot l$, $l = 1$, $K_1 = K_2 = \{0,2; 0,4; 0,6; 0,8; 1,0; 1,2; 1,4; 1,6; 1,8; 2,0; 3,0\}$ для различных m_ε и σ_ε будет такой, как показано на рис. 2.

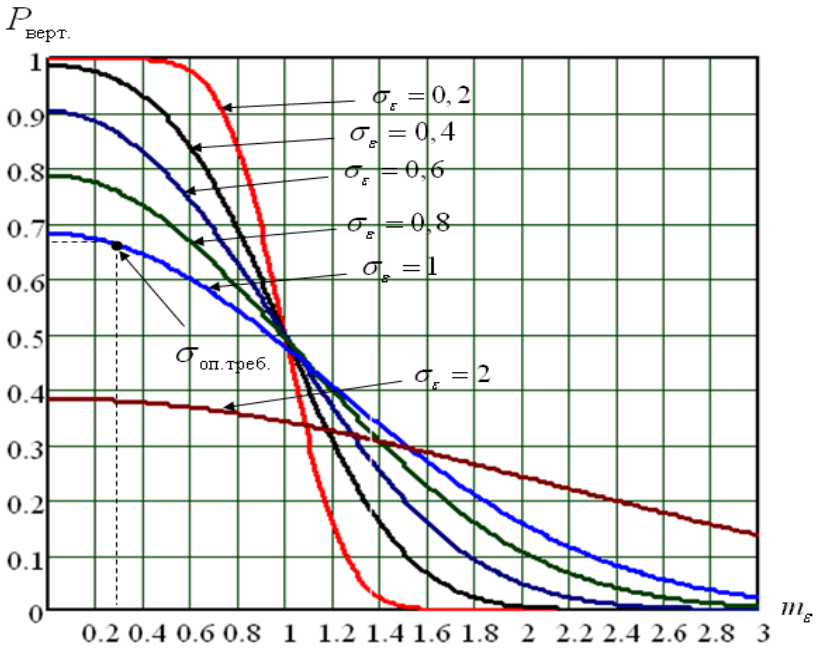


Рис. 2. Семейство кривых $P_{\text{верт.}(m_\varepsilon)}$ при различных $\sigma_{\text{м}_i}$

Семейство кривых на рис. 2 позволяет по заданной вероятности успешного обнаружения оператором искомого элемента ИМ определить предельно допустимые точностные характеристики ИМ m_ε и σ_ε. В рассматриваемом случае $\sigma_{\text{оп треб.}}$ будет определяться по графику, представленному на рис. 2, следующим образом.

Под m_ε подразумевается систематическая ошибка поиска, вносимая средством отображения информации. Выбранное значение откладывается по оси абсцисс, значение вероятности

обнаружения – по оси ординат. На пересечении указанных значений (на рис. 2 – пунктирные линии) и будет находиться искомая величина $\sigma_{\text{оп треб.}}$.

5. Сопоставление точности действий оператора с требуемым значением погрешности обработки информации в системе.

Действия 4 и 5 позволяют задать требование к точности обработки информации оператором следующим образом:

$$\sigma_{\text{оп}} \le \sigma_{\text{оп треб.}} - \sum_{i=1}^{n} \sigma_{\text{м}_i}, \qquad (3)$$

где $\sigma_{\text{оп треб.}}$ – требуемая погрешность обработки информации оператором, $\sigma_{\text{м}_i}$ – погрешность обработки информации в i-м звене машины, n – число машинных звеньев.

В случае невыполнения требования, описываемого выражением (2), осуществляются действия [3], направленные на уменьшение времени реакции оператора и проводимые на этапе выбора форм представления оператору информации о контролируемых признаках и органах управления:

− повышение заметности событий;

− улучшение соответствия «стимул – реакция на события»;

− уменьшение количества условных обозначений, используемых в ИМ;

− упрощение логической и математической обработки, уменьшение порядка данных, обрабатываемых оператором;

− разделение когнитивных блоков информации паузами, сопровождающимися отсутствием событий;

− корректирование объема зрительного восприятия оператора;

− использование простых геометрических фигур в качестве условных обозначений, применяемых в ИМ;

− увеличение линейных размеров элементов ИМ до значений, ограниченных объемом зрительного восприятия;

− уменьшение расстояния до смежных элементов ИМ, используемых при решении однородных задач оператором.

В случае невыполнения требования, описываемого выражением (3), на основании анализа эмпирической информации производятся изменения форм ИМ и их компоновки.

Рассмотренные выше действия в рамках синтеза ИМ носят итерационный характер и производятся до тех пор, пока не выполнятся неравенства (2) и (3).

Для проверки адекватности ИМ разрабатываются динамические прототипы из ее элементов на основании результатов компоновки и комплексирования. Прототипы должны характеризоваться высокой достоверностью, детализацией, визуальной точностью и высокой степенью близости к конечному виду ИМ. Разработка динамических прототипов производится с помощью специализированных средств прототипирования, например кросс-платформенного инструмента динамического прототипирования ИМ GUI Machine, который является русифицированным аналогом таких известных программ, как Axure RP, GUI Design Studio, MS Expression Blend, и отличается простотой, функциональной наполненностью, интуитивностью, высокой скоростью и удобством создания прототипов ИМ. Продукт ориентирован на создание интерактивных прототипов сложных десктоп- и веб-приложений без написания программного кода в средах Windows, Mac OS и Linux.

Проверка адекватности динамических прототипов ИМ осуществляется непосредственно с участием реальных операторов и экспертов на основе метода фокус-групп [7], при этом фиксируются их замечания и предложения по улучшению ИМ.

Сущность проверки адекватности динамических прототипов ИМ состоит в выявлении степени их соответствия некоторому эталону. Проверку рекомендуется выполнять несколькими методами. Прямые методы проверки подразумевают натурный эксперимент, позволяющий получить истинные значения показателей:

− по средним значениям откликов динамических прототипов ИМ и системы, описываемой ИМ;

− дисперсиям отклонений откликов динамических прототипов ИМ от среднего значения откликов системы, описываемой ИМ;

− максимальному значению отклонений откликов динамических прототипов ИМ от откликов системы, описываемой ИМ.

Косвенные методы проверки применяются в случаях, когда прямые методы невозможны или нецелесообразны. Для этого необходимо использовать проверку согласованности результатов моделирования с результатами, полученными на существующих ИМ. Причинами неадекватности, как правило, являются ошибки в организации и проведении опытов (например, неконтролируемое изменение не учтенных при проектировании ИМ факторов), погрешности в задании исходных данных и измерении результатов, большой размах варьирования параметров моделирования.

Если по результатам проверки адекватности выявляются недопустимые расхождения между системой, описываемой ИМ, и ее динамическими прототипами, необходимо изменить формы представления информации о контролируемых признаках и органах управления. Затем следует изменить компоновку форм, заново определить параметры ИМ, проверить выполнение точностных и временны́х условий и адекватность динамических прототипов. Если результаты проверки удовлетворительны, необходимо оценить эффективность динамических прототипов по отношению к существующим аналогам.

Проверка эффективности производится непосредственно с участием экспертов и специалистов в рамках полунатурного моделирования. Далее решается двухкритериальная задача оценки эффективности синтезированной и аналогичной существующей динамических ИМ при помощи интегрального показателя, описываемого выражением [5]

$$\eta^* = \frac{\sum_{i=1}^{k_1} X_i^{(\eta_1)} \cdot K_i^{(\eta_1)}}{k_1} + \frac{\sum_{j=1}^{k_2} \left[M_j^{(\Delta\eta_2)} + 3\sigma_j^{(\eta_2)} \right] \cdot K_j^{(\eta_2)}}{k_2},$$

где k_1 – число заданных основных параметров, определяющих точность выдерживания режима в определенных точках; k_2 – число заданных основных параметров, определяющих временны́е характеристики данного режима управления; $X_i^{(\eta_1)}$ – моментные отклонения основных параметров управляемого процесса в определенных точках; $M_j^{(\Delta\eta_2)}$, $3\sigma_j^{(\eta_2)}$ – статистические показатели, отражающие средние отклонения j параметров от заданных значений и их вариантность; $K_i^{(\eta_1)}$, $K_j^{(\eta_2)}$ – весовые коэффициенты, соответствующие заданным точностным и временны́м параметрам управляемого процесса; $(\eta_1), (\eta_2)$ – индексы, отражающие принадлежность входящих в данную формулу параметров к точностным и временны́м характеристикам управля емого человеком процесса соответственно.

Значения параметра η^* для синтезированной и существующей ИМ сравниваются, и делается вывод о степени превосходства одной ИМ над другой, либо об отсутствии данного превосходства, либо его незначительности, свидетельствующей об уровне эффективности синтезированной ИМ по сравнению с существующей. В случае отсутствия превосходства (низкой эффективности) синтезированной ИМ устраняются замечания и реализуются рекомендации экспертов и специалистов, полученные в ходе оценки адекватности, а также целесообразные управляющие воздействия со стороны ЭВМ (определяются в процессе реализации метода диагональной матрицы событий [1]). Если эффективность синтезированных динамических прототипов ИМ удовлетворительна, формируется результирующая ИМ.

3. Проверка адекватности предлагаемой методики проектирования информационной модели

Согласно предлагаемой методике структурно-параметрического синтеза ИМ боевой обстановки на автоматизированном рабочем месте лица, принимающего решение, был синтезирован фрагмент ИМ боевой обстановки для АСУ ПВО дивизионного (бригадного) уровня управления. Путем сравнения синтезированного фрагмента с уже существующей аналогичной ИМ была проверена адекватность предлагаемой методики. Фрагмент ИМ отражал основные составляющие боевой работы лица, принимающего решение в рассматриваемой АСУ, и позволял имитировать этапы ведения боевой работы по отражению воздушного налета предполагаемого противника.

Проверка адекватности фрагмента ИМ проводилась непосредственно с участием экспертов и операторов, имеющих опыт работы в системах подобного рода. При оценке адекватности фрагмента ИМ в рамках прямого метода проверки использовалось полунатурное моделирование боевой работы оператора. Затем оценивалась близость теоретических результатов и практических реализаций к существующим образцам военной техники. Оценка проводилась с использованием известного и хорошо себя зарекомендовавшего в данной предметной области критерия согласия, на выбор которого кроме специфики использования и мощности повлияли также объем и качество выборки, тип шкалы измерения и закон распределения выборки.

Генеральная совокупность событий во фрагменте ИМ представляла собой множество событий с соответствующим множеством значений времени реакции оператора на эти события (выборка состояла из 43 возможных операций).

Для оценки выборок синтезированного фрагмента и существующей ИМ использовался статистический пакет анализа данных приложения Microsoft Office Excel 2010. В табл. 1 приведена описательная статистика, а на рис. 3 изображены гистограммы частот попадания данных (значений времени реакции оператора) в указанные интервалы значений для синтезированного фрагмента и существующей ИМ.

Таблица 1

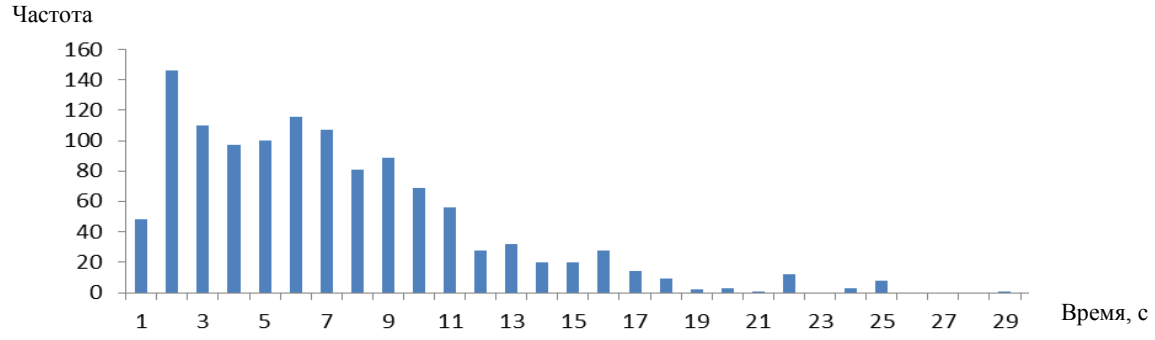
Описательная статистика синтезированного фрагмента и существующей ИМ

Статистическая величина	Значение для синтезированного фрагмента ИМ	Значение для существующей ИМ
Математическое ожидание	6,690 675	6,699 308
Среднеквадратическое отклонение	4,732 586	4,680 335
Дисперсия	22,397 371	21,905 541
Медиана	5,915	6,175
Мода	6	6,7
Асимметричность	1,208 036	1,134 102
Количество элементов выборки	1200	1200
Уровень надежности (при 99, 0 %)	0,352 465	0,348 574

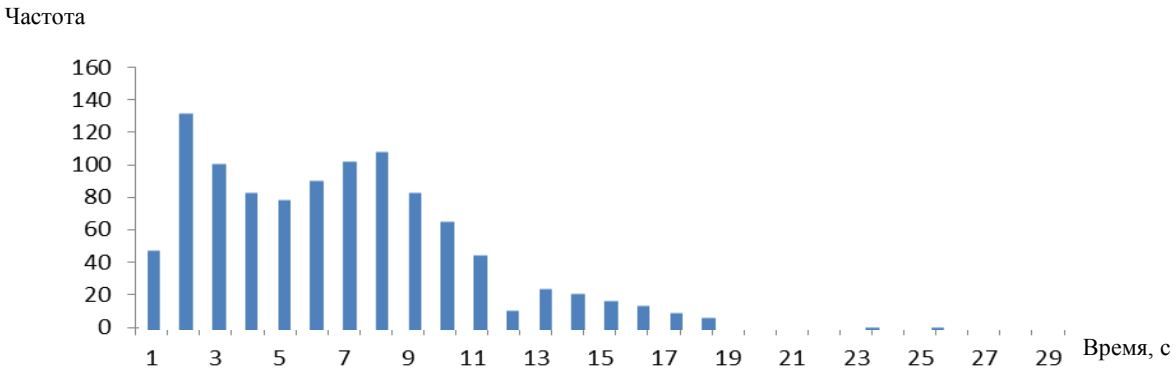
Анализ описательной статистики показал схожесть значений статистических величин синтезированного фрагмента и существующей ИМ и, как следствие, подобие законов распределения времени выполнения действий в последних. Наличие положительной асимметрии подтвердило тяготение законов распределения к гамма-распределению как наиболее характерному для действий оператора в человекомашинных системах.

Гистограммы, отражающие приближенные графики статистических функций распределения, подобны визуально и свидетельствуют о схожести законов распределения времени выполнения операций в синтезированном фрагменте и существующей ИМ.

При помощи критерия χ^2 проверялась гипотеза об отсутствии различий между двумя эмпирическими распределениями, полученными в ходе полунатурного моделирования.



а)



б)

Рис. 3. Частоты попадания данных в интервалы временны́х значений: *а)* для синтезированного фрагмента ИМ; *б)* для существующей ИМ

Вычисление χ^2 проводилось при помощи выражения [8 с. 149]

$$\chi^2 = 4 \cdot \sum^{k} \frac{f1 \cdot f1}{f1 + f2} - 2 \cdot N, \qquad (4)$$

где $f1$ – частоты распределения в синтезированном фрагменте, $f2$ – частоты распределения в существующей ИМ, N – число элементов в каждой выборке, k – число интервалов разбиения. Число степеней свободы $\nu = (k-1) \cdot (c-1)$, где c – число столбцов.

В табл. 2 приведены эмпирические данные и предварительные расчеты, необходимые для получения значения χ^2. Путем подстановки значения из таблицы в выражение (4) было получено $\chi^2 = 28,248\ 342\ 29$. Число степеней свободы $\nu = 28$. С помощью табличных данных [8, с. 307] были найдены критические значения χ^2 для уровней статистической значимости $p \le 0,05$ и $p \le 0,01$ при указанном числе степеней свободы:

$$\chi^2 = \begin{cases} 42,557 \text{ для } p \le 0,05; \\ 49,558 \text{ для } p \le 0,01. \end{cases}$$

После построения оси значимости (рис. 4) было установлено, что полученные различия (выделены овалом) попали в зону незначимости. Таким образом, была отклонена гипотеза о различии распределений двух выборок и принята гипотеза об их подобии.

В пользу адекватности синтезированного фрагмента ИМ говорит и тот факт, что суммы ошибок, допущенных испытуемыми в обоих фрагментах, отличались менее чем на 1 %: 139 ошибок в синтезированном фрагменте ИМ против 138 в существующей ИМ.

Таблица 2

Эмпирические данные и расчеты

Время выполнения, с	Частоты		$f1 \cdot f1$	$f1 + f2$	$\frac{f1 \cdot f1}{f1 + f2}$
	$f1$	$f2$			
1	48	54	2304	102	22,588 235 29
2	146	142	21 316	288	74,013 888 89
3	110	110	12 100	220	55
4	97	92	9409	189	49,783 068 78
5	100	87	10 000	187	53,475 935 83
6	116	99	13 456	215	62,586 046 51
7	107	111	11 449	218	52,518 348 62
8	81	118	6561	199	32,969 849 25
9	89	91	7921	180	44,005 555 56
10	69	73	4761	142	33,528 169 01
11	56	52	3136	108	29,037 037 04
12	28	16	784	44	17,818 181 82
13	32	30	1024	62	16,516 129 03
14	20	27	400	47	8,510 638 298
15	20	22	400	42	9,523 809 524
16	28	20	784	48	16,333 333 33
17	14	15	196	29	6,758 620 69
18	9	12	81	21	3,857 142 857
19	2	4	4	6	0,666 666 667
20	3	3	9	6	1,5
21	1	4	1	5	0,2
22	12	4	144	16	9
23	0	5	0	5	0
24	3	2	9	5	1,8
25	8	6	64	14	4,571 428 571
26	0	0	0	0	
27	0	0	0	0	
28	0	0	0	0	
29	1	1	1	2	0,5
30	0	0	0	0	

Иными словами, вероятности успешных действий оператора в обоих фрагментах достаточно близки друг к другу (0,884 в синтезированном фрагменте ИМ против 0,885 в существующем).

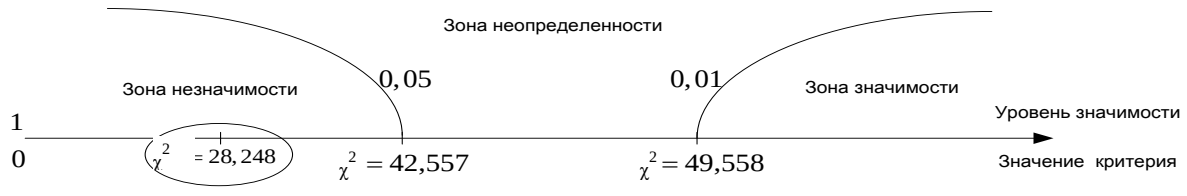


Рис. 4. Ось значимости

Таким образом, адекватность синтезированного фрагмента ИМ подтвердила состоятельность предлагаемой методики синтеза ИМ на рабочем месте оператора командного пункта войск ПВО.

Заключение

Время обработки информации в АСУ является одним из основополагающих критериев при разработке ИМ динамично меняющейся обстановки. При этом разработчик ИМ должен обеспечить не просто высокую, а гарантированную скорость обработки информации оператором. Безошибочность действий оператора наряду со скоростью выполнения действий также является важным критерием при компоновке элементов ИМ динамично меняющейся обстановки.

Предлагаемый методический аппарат дает возможность решить задачу структурно-параметрического синтеза ИМ. Синтезированная ИМ будет характеризоваться соблюдением эффективного баланса между степенью загруженности лица, принимающего решение, и возможностями последнего по реализации возложенных на него функций.

Использование предлагаемой методики позволит повысить эффективность проектируемых человекомашинных систем в ситуациях, связанных с высокой нагрузкой оператора (например, на рабочем месте лица, принимающего решение, в АСУ ПВО при большой плотности налета) и улучшить на 7–9 % [9] качество управления за счет оптимизации ИМ как составляющей подсистемы отображения информации автоматизированного рабочего места оператора.

Список литературы

1. Галактионов, А.И. Представление информации оператору (исследование деятельности человека – оператора производственных процессов) / А.И. Галактионов. – М. : Энергия, 1969. – 139 с.

2. Рабченок, Д.И. Анализ подходов к построению информационной модели боевой обстановки в условиях неопределенности / С.В. Кругликов, Д.И. Рабченок // Вестн. Воен. акад. Респ. Беларусь. – 2012. – № 1. – С. 64–71.

3. Кругликов, С.В. Адаптивные информационно-управляющие системы специального назначения: теория и практика синтеза / С.В. Кругликов. – Минск : ВА РБ, 2014. – 233 с.

4. Береза, А.С. Основы построения комплексов технических средств АСУ ПВО / А.С. Береза. – Харьков : ХВУ, 1993. – 386 с.

5. Шибанов, Г.П. Количественная оценка деятельности человека в системах человек – техника / Г.П. Шибанов. – М. : Машиностроение, 1983. – 263 с.

6. Кругликов, С.В. Автоматизация процессов организационного управления силами и средствами / С.В. Кругликов. – Минск : ВАРБ, 2007. – 229 с.

7. Белановский, С.А. Методика и техника фокусированного интервью : учеб.-метод. пособие / С.А. Белановский. – М. : Наука, 1993. – 352 с.

8. Ермолаев, О.Ю. Математическая статистика для психологов : учебник / О.Ю. Ермолаев. – М. : Московский психолого-социальный институт Флинта, 2002. – 336 с. – (Библиотека психолога).

9. Герасимов, Б.М. Системное проектирование средств отображения информации в АСУ / Б.М. Герасимов, Б.М. Егоров. – Киев : КВИРТУ ПВО, 1983. – 348 с.

Поступила 08.11.2016

Военная академия Республики Беларусь,
Минск, пр. Независимости, 220
e-mail: dimedrolus1798@tut.by

D.I. Rabchonak

METHOD FOR SYNTHESIS OF INFORMATION MODEL OF COMBAT SITUATION

The technique of structural and parametric synthesis of the information model of the combat situation on the workstation of the decision maker is considered. The used method of diagonal matrix of events allows to form an effective structure, parameters and connections between the elements of the information model. The prototypes of dynamic elements of the information model are developed. The adequacy of the information model is tested. The effectiveness of the information model using the integral index of the quality of the human operator work is evaluated.

УДК 004.3

Л.И. Кульбак

РАСЧЕТ ПОКАЗАТЕЛЕЙ НАДЕЖНОСТИ НЕВОССТАНАВЛИВАЕМЫХ ОБЪЕКТОВ С УЧЕТОМ ПОГРЕШНОСТЕЙ ИСХОДНЫХ ДАННЫХ

Приводится методика расчета показателей надежности невосстанавливаемых объектов с учетом погрешностей исходных данных. Рассматриваются объекты, надежность которых обеспечивается путем структурного резервирования с ограниченной кратностью. К числу таких объектов относится бортовая аппаратура малогабаритных космических аппаратов.

Введение

Под погрешностью показателя надежности (ПН) объекта принято понимать отклонение реального значения ПН объекта от его истинного значения. Задача оценки ПН объектов с учетом погрешности исходных данных частично была решена в работе [1], посвященной интервальной оценке расчетных значений ПН объектов. В ней рассматривалась вероятностная модель и интервальная оценка ПН представлялась в виде числового интервала $[X_{CP} - \Delta X, X_{CP} + \Delta X]$, в который с определенной вероятностью α попадает истинное значение ПН. Здесь X_{CP} – среднее значение ПН, ΔX – отклонение ПН от среднего значения (погрешность). Объекты, имеющие различные виды структурного резервирования, не рассматривались.

Представляется целесообразным продолжить работу [1] с учетом различных видов структурного резервирования в невосстанавливаемых объектах и перейти от вероятностной модели погрешностей к детерминированной.

1. Модель оценки погрешностей при расчете ПН объекта

Суть расчета ПН объекта состоит в вычислении функции (ПН объекта) по ее аргументам (ПН комплектующих объект элементов). Как и при любом расчете, его характеристикой является погрешность (точность) вычислений.

Согласно теории погрешностей [2], если зависимость ПН объекта от ПН его элементов выражается функцией

$$R = f(\lambda_1, \lambda_2, ..., \lambda_m), \tag{1}$$

то результаты расчета ПН объекта следует вычислять по формулам

$$R = R_{CP} \pm \Delta R; \tag{2}$$

$$R_{CP} = f(\lambda_{CP.1}, \lambda_{CP.2}, ..., \lambda_{CP.m}); \tag{3}$$

$$\Delta R \le \sum_{i=1}^{m} \left| \frac{\partial f}{\partial \lambda_i} \right| \Delta\lambda_i, \tag{4}$$

где R – ПН объекта;

R_{CP} – среднее значение ПН объекта;

ΔR – погрешность определения ПН R;

$f(\lambda_1, \lambda_2, ..., \lambda_m)$ – функциональная зависимость ПН объекта R от ПН его элементов λ_i;

$\lambda_{CP.1}, \lambda_{CP.2}, ..., \lambda_{CP.m}$ – средние значения ПН элементов объекта;

$\frac{\partial f}{\partial \lambda_i}$ – первая производная от функции f по аргументу λ_i, вычисленная в точке $\lambda_{i\,CP}$;

$\Delta\lambda_i$ – погрешность определения λ_i.

2. Структурная модель надежности невосстанавливаемого объекта

В качестве модели надежности невосстанавливаемого объекта нормативная документация по расчету надежности [3] рекомендует использовать структурные схемы надежности (ССН).

Установлено, что подавляющее число структур объектов, показатели надежности которых подлежат расчету, можно привести к последовательной ССН (рис. 1).

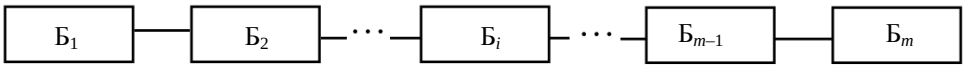


Рис. 1. Обобщенная последовательная ССН объекта

Метод свертки основан на последовательном преобразовании ССН объекта и сведении ее к основному соединению элементов. Покажем применение данного метода на примере ССН (рис. 2).

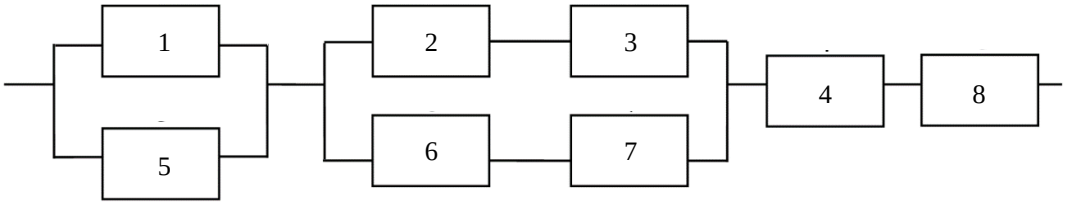


Рис. 2. Последовательно-параллельная ССН объекта

Метод свертки состоит из нескольких этапов.

На *первом этапе* рассматриваются все параллельные соединения, которые заменяются эквивалентными блоками с соответствующим ПН. В рассматриваемом примере такими параллельными блоками являются 1 и 5.

На *втором этапе* рассматриваются все последовательные соединения, которые заменяются эквивалентными блоками (рис. 3). Здесь последовательными элементами являются 2 и 3, 6 и 7, 4 и 8.

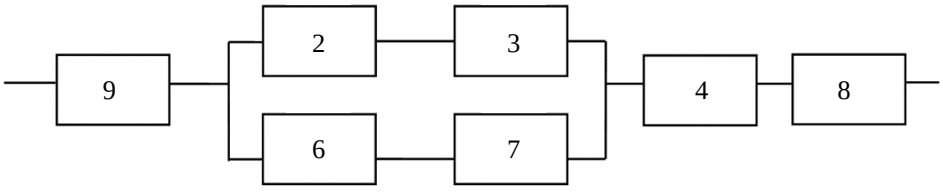


Рис. 3. ССН объекта после первого этапа свертки

На *третьем этапе* вновь рассматриваются параллельные соединения, которые заменяются эквивалентными блоками (рис. 4). В данном примере такими параллельными блоками являются 21 и 22.

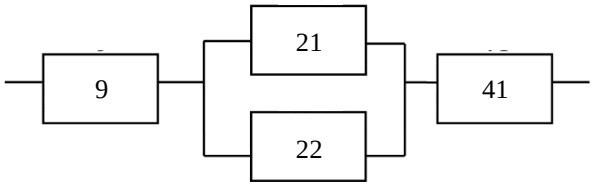


Рис. 4. ССН объекта после второго этапа свертки

ССН на рис. 5 является сверткой исходной ССН на рис. 1. Таким образом, для дальнейшего исследования методики расчетного значения ПН объекта с учетом погрешности можно использовать ССН, изображенную на рис. 1.

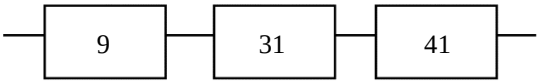


Рис. 5. ССН объекта после третьего этапа свертки

3. Анализ надежности блоков ССН

3.1. Общие положения

По аналогии с [4] примем, что ССН объекта может состоять из блоков простых структур, без резервирования элементов, и сложных, состоящих из резервируемых различными способами цепочек элементов.

В отличие от [4] анализ надежности блоков ССН проводится при условии их использования в следующих режимах эксплуатации:

функционирования по назначению – Р1;

хранения в аппаратуре в промежутках между функционированием – Р2;

комплексном (функционирование с перерывами хранения) – Р3.

В модели надежности объекта приняты некоторые допущения и ограничения:

– восстановление работоспособности объекта осуществляется автоматически путем использования резервных элементов, которые не подлежат восстановлению;

– элементы, подключающие резерв при замещении, имеют пренебрежимо малую интенсивность отказов;

– время, расходуемое на подключение резерва, пренебрежимо мало и не учитывается как время потери работоспособности объекта;

– кратности резервирования принимаются не более двух;

– исследованию подлежит лишь одно свойство надежности – безотказность;

– отказы элементов объекта являются независимыми событиями;

– в качестве показателей безотказности элементов используются интенсивность отказов λ и вероятность безотказной работы $P(t)$;

– в качестве показателей безотказности составных частей объекта используются вероятность безотказной работы $P(t)$ и интенсивность отказов Λ;

– закон распределения наработки до отказа элементов объекта принимается экспоненциальный с параметром λ;

– интенсивности отказов элементов объекта задаются в виде граничных значений: нижнего λ_H и верхнего λ_B.

3.2. Простой блок ССН

Простые блоки ССН объединяют элементы с одинаковыми значениями интенсивности отказов. Определим ПН таких блоков при их работе в различных режимах эксплуатации с учетом погрешностей ПН элементов.

Режим Р1. Среднее значение интенсивности отказов простого блока ССН в режиме функционирования $\Lambda_{П.Р1.СР}$ вычисляется по формуле

$$\Lambda_{П.Р_1.СР} = n\lambda_{Ф.СР}, \tag{5}$$

где n – количество элементов в простом блоке ССН;

$\lambda_{Ф.СР}$ – среднее значение интенсивности отказов элементов блока ССН в режиме функционирования.

Среднее значение вероятности безотказной работы простого блока ССН в режиме функционирования в интервале наработки t $P_{П.Р1.СР}(t)$ вычисляется по формуле

$$P_{П.Р1.СР}(t) = \exp(-\Lambda_{П.Р1.СР}t) = \exp(-n\lambda_{Ф.СР}t) \approx 1 - n\lambda_{Ф.СР}t\,. \quad (6)$$

Отклонение интенсивности отказов простого блока ССН в режиме функционирования от своего среднего значения $\Delta\Lambda_{П.Р1}$ находится из выражения

$$\Delta\Lambda_{П.Р1} = \frac{\partial\Lambda_{П.Р1}}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = n\Delta\lambda_{Ф}\,, \quad (7)$$

где $\Delta\lambda_{Ф}$ – отклонение интенсивности отказов элемента блока ССН в режиме функционирования от своего среднего значения.

Отклонение вероятности безотказной работы простого блока ССН в режиме функционирования от своего среднего значения $\Delta P_{П.Р1}(t)$ вычисляется по формулам

$$\Delta P_{П.Р1}(t) = \frac{\partial P_{П.Р1}(t)}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = \exp(-n\lambda_{Ф.СР}t)nt\Delta\lambda_{Ф} = P_{П.Р1.СР}(t)\Delta_{П.Р1}; \quad (8)$$

$$\Delta_{П.Р1} = nt\Delta\lambda_{Ф}, \quad (9)$$

где $\Delta_{П.Р1}$ – относительное отклонение вероятности безотказной работы простого блока ССН в режиме функционирования от своего среднего значения.

Режим Р2. Среднее значение интенсивности отказов простого блока ССН в режиме хранения $\Lambda_{П.Р2.СР}$ находится следующим образом:

$$\Lambda_{П.Р2.СР} = n\lambda_{Х.СР} = nb\lambda_{Ф.СР}; \quad (10)$$

$$\lambda_{Х.СР} = b\lambda_{Ф.СР}, \quad (11)$$

где $\lambda_{Х.СР}$ – среднее значение интенсивности отказов элементов блока ССН в режиме хранения;

b – коэффициент снижения интенсивности отказов функционирующего элемента блока ССН до интенсивности его отказов при хранении в составе аппаратуры.

Среднее значение вероятности безотказной работы простого блока ССН в режиме хранения в интервале наработки t $P_{П.Р2.СР}(t)$ вычисляется по формуле

$$P_{П.Р2.СР}(t) = \exp(-\Lambda_{П.Р2.СР}t) = \exp(-nb\lambda_{Ф.СР}t) \approx 1 - nb\lambda_{Ф.СР}t\,. \quad (12)$$

Отклонение интенсивности отказов простого блока ССН в режиме хранения от своего среднего значения $\Delta\Lambda_{П.Р2}$ вычисляется по формуле

$$\Delta\Lambda_{П.Р2} = \frac{\partial\Lambda_{П.Р2}}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = nb\Delta\lambda_{Ф}\,, \quad (13)$$

где $\Delta\lambda_{Ф}$ – отклонение интенсивности отказов элемента простого блока ССН в режиме функционирования от своего среднего значения.

Отклонение вероятности безотказной работы простого блока ССН в режиме хранения от своего среднего значения $\Delta P_{П.Р2}(t)$ вычисляется следующим образом:

$$\Delta P_{П.Р2}(t) = \frac{\partial P_{П.Р2}(t)}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = \exp(-nb\lambda_{Ф.СР}t)nbt\Delta\lambda_{Ф} = P_{П.Р2.СР}(t)\Delta_{П.Р2}; \quad (14)$$

$$\Delta_{П.Р2} = nbt\Delta\lambda_{Ф}, \quad (15)$$

где $\Delta_{П.Р2}$ – относительное отклонение вероятности безотказной работы простого блока ССН в режиме хранения от своего среднего значения.

Режим Р3. Простой блок ССН при его использовании в течение времени t в комплексном режиме можно представить в виде структурной схемы надежности, состоящей из двух простых блоков. При этом один блок характеризует работу блока в режиме функционирования в течение времени t_1, а второй блок – в режиме хранения в течение времени t_2. Выразим времена использования блока ССН в различных режимах через коэффициент интенсивности эксплуатации. Очевидно, что

$$t_1 = t\kappa_{ИЭ},\ t_2 = t(1-\kappa_{ИЭ}),\ t_1 + t_2 = t, \tag{16}$$

где $\kappa_{ИЭ}$ – коэффициент интенсивности эксплуатации блока ССН (доля общего времени использования блока ССН в режиме функционирования).

В этом случае среднее значение вероятности безотказной работы простого блока ССН в комплексном режиме вычисляется по формуле

$$P_{П.Р3.СР}(t) = P_{П.Р1.СР}(t)P_{П.Р2.СР}(t). \tag{17}$$

Подставив компоненты в формулу (9), получим

$$P_{П.Р3.СР}(t) = \exp(-n\lambda_{Ф.СР}t\kappa_{ИЭ})\exp[-nb\lambda_{Ф.СР}t(1-\kappa_{ИЭ})] \approx 1 - n\lambda_{Ф.СР}[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]t, \tag{18}$$

где $P_{П.Р3.СР}(t)$ – среднее значение вероятности безотказной работы простого блока ССН в комплексном режиме в интервале времени t.

Среднее значение интенсивности отказов блока ССН при использовании его в комплексном режиме $\Lambda_{П.Р3.СР}$ вычисляется по формуле

$$\Lambda_{П.Р3.СР} = n\lambda_{Ф.СР}[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]. \tag{19}$$

Отклонение интенсивности отказов простого блока ССН в комплексном режиме от своего среднего значения $\Delta\Lambda_{П.Р3}$ вычисляется по формуле

$$\Delta\Lambda_{П.Р3} = \frac{\partial\Lambda_{П.Р3}}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = n[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]\Delta\lambda_{Ф}, \tag{20}$$

где $\Delta\lambda_{Ф}$ – отклонение интенсивности отказов элемента блока ССН в режиме функционирования от своего среднего значения.

Отклонение вероятности безотказной работы простого блока ССН в комплексном режиме от своего среднего значения $\Delta P_{П.Р3}(t)$ вычисляется по формулам

$$\Delta P_{П.Р3}(t) = \frac{\partial P_{П.Р3}(t)}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} = \exp\{-n\lambda_{Ф.СР}t[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]\}nt[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]\Delta\lambda_{Ф} = \\ = P_{П.Р3.СР}(t)\Delta_{П.Р3}; \tag{21}$$

$$\Delta_{П.Р3} = nt[\kappa_{ИЭ} + b(1-\kappa_{ИЭ})]\Delta\lambda_{Ф}, \tag{22}$$

где $\Delta_{П.Р3}$ – относительное отклонение вероятности безотказной работы простого блока ССН в комплексном режиме от своего среднего значения.

3.3. Сложные блоки ССН

Примем, что сложные блоки ССН представляют собой одиночный элемент или последовательную цепочку элементов с однократным нагруженным или ненагруженным резервом из аналогичных элементов с одинаковыми значениями интенсивности отказов. Определим ПН таких блоков при их работе в различных режимах эксплуатации с учетом погрешностей ПН элементов.

Режим Р1. Среднее значение вероятности безотказной работы сложного блока ССН с нагруженным однократным резервом в режиме функционирования $P_{C.P1.H.CP}(t)$ согласно [5] вычисляется по формуле

$$P_{C.P1.H.CP}(t) = 2\exp(-\lambda_{\Phi.CP}t) - \exp(-2\lambda_{\Phi.CP}t) \approx 1-(\lambda_{\Phi.CP}t)^2. \quad (23)$$

Среднее значение интенсивности отказов сложных блоков ССН с нагруженным однократным резервом в режиме функционирования $\Lambda_{C.P1.H.CP}$ находится следующим образом:

$$\Lambda_{C.P1.H.CP}(t) = \frac{2\lambda_{\Phi.CP}\exp(-\lambda_{\Phi.CP}t)[1-\exp(-\lambda_{\Phi.CP}t)]}{1-[1-\exp(-\lambda_{\Phi.CP}t)]^2} \approx \frac{2\lambda_{\Phi.CP}^2 t}{1+\lambda_{\Phi.CP}t}. \quad (24)$$

Отклонение интенсивности отказов сложного блока ССН с нагруженным однократным резервом от своего среднего значения в режиме функционирования $\Delta\Lambda(t)_{C.P1.H}$ вычисляется по формуле

$$\Delta\Lambda_{C.P1.H} = \frac{\partial\Lambda_{C.P1.\text{H}}}{\partial\lambda_{\Phi.CP}}\Delta\lambda_{\Phi} \approx \frac{2\lambda_{\Phi.CP}t(2+\lambda_{\Phi.CP}t)}{(1+\lambda_{\Phi.CP}t)^2}\Delta\lambda_{\Phi}. \quad (25)$$

Отклонение вероятности безотказной работы сложного блока ССН с нагруженным однократным резервом от своего среднего значения в режиме функционирования $\Delta P_{C.P1.H}(t)$ вычисляется по формулам

$$\Delta P_{C.P1H}(t) = \frac{\partial P_{C.P1.H.CP}(t)}{\partial\lambda_{\Phi.CP}}\Delta\lambda_{\Phi.CP} = 2\exp(-\lambda_{\Phi.CP}t)t\ \Delta\lambda_{\Phi} - \exp(-2\lambda_{\Phi.CP}t)2\ t\Delta\lambda_{\Phi} =$$
$$= 2t\ \Delta\lambda_{\Phi}\ [\exp(-\lambda_{\Phi.CP}\ t) - \exp(-2\lambda_{\Phi.CP}t)] = 2t\ \Delta\lambda_{\Phi}\ [\exp(-\lambda_{\Phi.CP}t) - \quad (26)$$
$$-\exp(-2\lambda_{\Phi.CP}t)]\ \frac{P_{C.P1.H.CP}(t)}{P_{C.P1.H.CP}(t)} = P_{C.P1.H.CP}(t)\Delta_{C.P1.H};$$

$$\Delta_{C.P1.H} = 2t\left[\frac{1-\exp(-\lambda_{\Phi.CP}t)}{2-\exp(-\lambda_{\Phi.CP}t)}\right]\Delta\lambda_{\Phi} \approx \frac{2\lambda_{\Phi.CP}t^2}{1+\lambda_{\Phi.CP}t}\Delta\lambda_{\Phi}. \quad (27)$$

Под ненагруженным резервом блока понимается состояние, когда на резервный элемент блока не подается электропитание. В действительности резервный элемент подвергается нагрузке, характерной для режима хранения. Представляется целесообразным учесть данное положение. Учет можно осуществить, если представить, что ненагруженный режим заменяется режимом с облегченной нагрузкой, при которой интенсивность отказа резерва равнялась интенсивности отказов в режиме хранения.

Среднее значение вероятности безотказной работы сложного блока ССН с ненагруженным однократным резервом в режиме функционирования $P_{C.P1.HH.CP}(t)$ с учетом отказа резервного элемента при хранении согласно [5] вычисляется по формуле

$$P_{C.P1.HH.CP}(t) = \frac{1}{b}[1+b-\exp(-b\lambda_{\Phi.CP}t)]\exp(-\lambda_{\Phi.CP}t) \approx 1-(\lambda_{\Phi.CP}t)^2. \quad (28)$$

Среднее значение интенсивности отказов сложного блока ССН с ненагруженным однократным резервом в режиме функционирования с учетом отказа резервного элемента при хранении $\Lambda_{C.P1.HH.CP}$ следует определить по формуле

$$\Lambda_{C.P1.HH.CP}(t) = -\frac{dP_{C.P1.HH.CP}(t)}{d(t)\ P_{C.P1.HH.CP}(t)}, \quad (29)$$

при этом

$$\frac{dP_{C.P1.HH.CP}(t)}{d(t)} = -\frac{\lambda_{\Phi.CP}(1+b)}{b}\{\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]\}; \quad (30)$$

$$\Lambda_{C.P1.HH.CP}(t) = \frac{\lambda_{\Phi.CP}(1+b)\{\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]\}}{(1+b)\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]} \approx \lambda_{\Phi.CP}{}^2 t(1+b). \quad (31)$$

Отклонение интенсивности отказов сложного блока ССН с ненагруженным однократным резервом от своего среднего значения в режиме функционирования с учетом отказа резервного элемента при хранении $\Delta\Lambda_{C.P1.HH}$ вычисляется следующим образом:

$$\Delta\Lambda_{C.P1.HH} = \frac{\partial\Lambda_{C.P1.HH}(t)}{\partial\lambda_{\Phi.CP}}\Delta\lambda_{\Phi} \approx 2(1+b)\lambda_{\Phi.CP}t\Delta\lambda_{\Phi}. \quad (32)$$

Отклонение вероятности безотказной работы сложного блока ССН с ненагруженным однократным резервом от своего среднего значения в режиме функционирования с учетом отказа резервного элемента при хранении $\Delta P_{C.P1.HH}(t)$ вычисляется по формулам

$$\Delta P_{C.P1.H}(t) = \frac{\partial P_{C.P1.H}(t)}{\partial\lambda_{\Phi.CP}}\Delta\lambda_{\Phi.CP} = \frac{(1+b)t}{b}\{\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]\}\Delta\lambda_{\Phi} = \\ = P_{C.P1.HH}(t)\Delta_{C.P1.HH}; \quad (33)$$

$$\Delta_{C.P1.HH} = \frac{(1+b)t\{\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]\}}{(1+b)\exp(-\lambda_{\Phi.CP}t) - \exp[-\lambda_{\Phi.CP}t(1+b)]}\Delta\lambda_{\Phi}. \quad (34)$$

Режим Р2. Характеристики сложного блока ССН в режиме хранения вычисляются по формулам характеристик нагруженного сложного блока ССН с заменой интенсивности $\lambda_{\Phi.CP}$ на $\lambda_X = b\lambda_{\Phi.CP}$.

Среднее значение вероятности безотказной работы сложного блока ССН с однократным резервом в режиме хранения $P_{C.P2.CP}(t)$ вычисляется по формуле (23) с заменой $\lambda_{\Phi.CP}$ на $\lambda_X = b\,\lambda_{\Phi.CP}$:

$$P_{C.P2.CP}(t) = 2\exp(-b\lambda_{\Phi.CP}t) - \exp(-2b\lambda_{\Phi.CP}t) \approx 1 - (b\lambda_{\Phi.CP}t)^2. \quad (35)$$

Среднее значение интенсивности отказов сложного блока ССН с однократным резервом в режиме хранения $\Lambda_{C.P2.CP}$ вычисляется по формуле (24) с заменой $\lambda_{\Phi.CP}$ на $\lambda_X = b\lambda_{\Phi.CP}$:

$$\Lambda_{C.P2.CP}(t) = \frac{2b\lambda_{\Phi.CP}\exp(-b\lambda_{\Phi.CP}t)[1-\exp(-b\lambda_{\Phi.CP}t)]}{1-[1-\exp(-b\lambda_{\Phi.CP}t)]^2} \approx \frac{2b^2\lambda_{\Phi.CP}^2 t}{1+b\lambda_{\Phi.CP}t}. \quad (36)$$

Отклонение интенсивности отказов сложного блока ССН с однократным резервом от своего среднего значения в режиме хранения $\Delta\Lambda_{C.P2}$ вычисляется по формуле (25) с заменой $\lambda_{\Phi.CP}$ на $\lambda_X = b\lambda_{\Phi.CP}$:

$$\Delta\Lambda_{C.P2} = \frac{\partial\Lambda_{C.P2}(t)}{\partial\lambda_{\Phi.CP}}\Delta\lambda_{\Phi} \approx \frac{4b\lambda_{\Phi.CP}t(1-b\lambda_{\Phi.CP}t)}{1+b\lambda_{\Phi.CP}t}. \quad (37)$$

Отклонение вероятности безотказной работы сложного блока ССН с однократным резервом от своего среднего значения в режиме хранения $\Delta P_{C.P2}(t)$ вычисляется по формуле (26) с заменой $\lambda_{\Phi.CP}$ на $\lambda_X = b\lambda_{\Phi.CP}$:

$$\Delta P_{C.P2}(t) = \frac{\partial P_{C.P2}(t)}{\partial \lambda_{\Phi.CP}} \Delta\lambda_{\Phi.CP} = 2\exp(-b\lambda_{\Phi.CP}t)bt\,\Delta\lambda_{\Phi} - \exp(-2b\lambda_{\Phi.CP})2\,bt\Delta\lambda_{\Phi.CP} =$$
$$= 2bt\,\Delta\lambda_{\Phi}\,[\exp(-b\lambda_{\Phi.CP}\,t) - \exp(-2b\lambda_{\Phi.CP}t)] = 2t\,b\Delta\lambda_{\Phi}\,[\exp(-b\lambda_{\Phi.CP}t) - \quad (38)$$
$$-\exp(-2b\lambda_{\Phi.CP}t)]\;\frac{P_{C.P2.CP}(t)}{P_{C.P2.CP}(t)} = P_{C.P2.CP}(t)\Delta_{C.P2};$$

$$\Delta_{C.P2} = 2bt\left[\frac{1-\exp(-b\lambda_{\Phi.CP}t)}{2-\exp(-b\lambda_{\Phi.CP}t)}\right]\Delta\lambda_{\Phi} \approx \frac{2b\lambda_{\Phi.CP}t^2}{1+b\lambda_{\Phi.CP}t}\Delta\lambda_{\Phi}. \quad (39)$$

Режим Р3. Сложный блок ССН, как и простой блок, при использовании в течение времени t в комплексном режиме можно представить в виде ССН, состоящей из двух простых блоков. При этом один блок характеризует работу блока в режиме функционирования как нагруженный резерв или ненагруженный резерв в течение времени t_1, а второй блок характеризует работу блока в режиме хранения в течение времени t_2 ($t_1+t_2 = t$). В этом случае среднее значение вероятности безотказной работы сложного блока ССН, работающего с нагруженным резервом в комплексном режиме, вычисляется по формуле

$$P_{C.H.P3.CP}(t) = P_{C.H.P1.CP}(t)P_{C.H.P2.CP}(t). \quad (40)$$

Подставив компоненты в формулу (40), получим

$$P_{C.H.P3.CP}(t) = [2\exp(-\lambda_{\Phi\ .CP}t\kappa_{ИЭ}) - \exp(-2\lambda_{\Phi.CP}t\kappa_{ИЭ})]\{\,2\exp[-b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})] -$$
$$-\exp[-2b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]\,\} \approx 1-(\lambda_{\Phi.CP}t\kappa_{ИЭ})^2]\{1-[b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})^2]\}. \quad (41)$$

Среднее значение интенсивности отказов сложного блока ССН, работающего с нагруженным резервом в комплексном режиме, вычисляется по формуле

$$\Lambda_{C.H.P3.CP}(t) = \Lambda_{C.H.P1.CP}(t_1) + \Lambda_{C.H.P2.CP}(t_2). \quad (42)$$

Подставив компоненты в формулу (42), получим

$$\Lambda_{C.H.P3.CP}(t) = \frac{2\lambda_{\Phi.CP}\exp(-\lambda_{\Phi.CP}t\kappa_{ИЭ})[1-\exp(-\lambda_{\Phi.CP}t\kappa_{ИЭ})]}{1-[1-\exp(-\lambda_{\Phi.CP}t\kappa_{ИЭ})]^2} +$$
$$+\frac{2b\lambda_{\Phi.CP}\exp[-b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]\{1-\exp[-b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]\}}{1-\{1-\exp[(-b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]\}^2} \approx \quad (43)$$
$$\approx 2\lambda^2_{\Phi.CP}t\{\kappa_{ИЭ}(1-\lambda_{\Phi.CP}t\kappa_{ИЭ}) + b^2[1-b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]\}.$$

Отклонение вероятности безотказной работы сложного блока ССН с нагруженным однократным резервом от своего среднего значения в комплексном режиме $\Delta P(t)_{C.P3.H}$ вычисляется по формулам

$$\Delta P(t)_{C.P3.H} = \frac{\partial P_{C.P3.H.CP}(\mathrm{t})}{\partial \lambda_{\Phi.CP}}\Delta\lambda_{\Phi} \approx 2\lambda_{\Phi.CP}bt^2[1-(\lambda_{\Phi.CP}t\kappa_{ИЭ})^2] +$$
$$+2\lambda_{\Phi.CP}\kappa^2_{ИЭ}t^2\{1-[b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]^2\}\frac{P_{C.P3.H.CP}(t)}{P_{C.P3.H.CP}(t)}\Delta\lambda_{\Phi.CP} = P_{C.P3.H.CP}(t)\Delta_{C.H}; \quad (44)$$

$$\Delta_{C.H} = \frac{2\lambda_{\Phi.CP}bt^2[1-(\lambda_{\Phi.CP}t\kappa_{ИЭ})^2] + 2\lambda_{\Phi.CP}\kappa^2_{ИЭ}t^2\{1-[b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})]^2\}}{[1-(\lambda_{\Phi.CP}t\kappa_{ИЭ})^2]\{1-[b\lambda_{\Phi.CP}t(1-\kappa_{ИЭ})^2]\}}\Delta\lambda_{\Phi} \approx \quad (45)$$
$$\approx 2\lambda_{\Phi.CP}t^2(b+\kappa^2_{ИЭ})\Delta\lambda_{\Phi}.$$

Отклонение интенсивности отказов сложного блока ССН с нагруженным однократным резервом от своего среднего значения в комплексном режиме $\Delta\Lambda(t)_{С.Р3.Н}$ находится согласно выражению

$$\Delta\Lambda_{С.Р3.Н}(t)=\frac{\partial\Lambda_{С.Р3.Н.СР}(t)}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф}\approx\frac{2\lambda_{Ф.СР}t_1(2+\lambda_{Ф.СР})}{(1+t\lambda_{Ф.СР})^2}+\frac{2b\lambda_{Ф.СР}t_2(2+b\lambda_{Ф.СР})}{(1+b\lambda_{Ф.СР})^2}. \tag{46}$$

В случае сложного блока ССН с ненагруженным резервом среднее значение вероятности безотказной работы в комплексном режиме $P_{С.НН.Р3.СР}(t)$ вычисляется по формуле

$$P_{С.НН.Р3.СР}(t)=P_{С.НН.Р1.СР}(t)P_{С.Р2.СР}(t). \tag{47}$$

Подставив компоненты в формулу (47), получим

$$\begin{aligned}P_{С.НН.Р3.СР}(t)=\frac{1}{b}\{[1+b-\exp(-b\lambda_{Ф.СР}t\kappa_{ИЭ})]\exp(-\lambda_{Ф.СР}t\kappa_{ИЭ})\}\{[1+b-\\-\exp(-b\lambda_{Ф.СР}t(1-\kappa_{ИЭ}))]\exp[-\lambda_{Ф.СР}bt(1-\kappa_{ИЭ})]\}.\end{aligned} \tag{48}$$

Среднее значение интенсивности отказов сложного блока ССН, работающего с ненагруженным резервом в комплексном режиме $\Lambda_{С.НН.Р3.СР}(t)$, определяется по формуле

$$\Lambda_{С.НН.Р3.СР}(t)=\Lambda_{С.НН.Р1.СР}(t_1)+\Lambda_{С.Р2.СР}(t_2). \tag{49}$$

Подставив компоненты в формулу (49), получим

$$\begin{aligned}\Lambda_{С.НН.Р3.СР}(t)&=\frac{\lambda_{Ф.СР}(1+b)\{\exp(-\lambda_{Ф.СР}t\kappa_{ИЭ})-\exp[-\lambda_{Ф.СР}t\kappa_{ИЭ}(1+b)]\}}{(1+b)\exp(-\lambda_{Ф.СР}t\kappa_{ИЭ})-\exp[-\lambda_{Ф.СР}t\kappa_{ИЭ}(1+b)]}+\\&+\frac{2b\lambda_{Ф.СР}\exp[-b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})]\{1-\exp[-b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})]\}}{1-\{1-\exp[b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})]^2\}}\approx\\&\approx\lambda^2t\kappa_{ИЭ}(1+b)+\frac{2b^2\lambda^2_{Ф.СР}t(1-\kappa_{ИЭ})}{1+b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})}.\end{aligned} \tag{50}$$

Отклонение вероятности безотказной работы сложного блока ССН с ненагруженным однократным резервом от своего среднего значения в комплексном режиме $\Delta P_{С.Р3.НН}(t)$ вычисляется по формулам

$$\begin{aligned}\Delta P_{С.Р3.НН}(t)&=\frac{\partial P_{С.Р3.НН.СР}(t)}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф}\approx\{[1-\frac{3}{2}b^4\lambda^3_{Ф.СР}t^4\kappa^2_{ИЭ}(1-\kappa_{ИЭ})^2]-\\&-3b^2\lambda_{Ф.СР}t^2(1-\kappa_{ИЭ})^2[1+\frac{(b\lambda_{Ф.СР}t\kappa_{ИЭ})^2}{2}]\}\frac{P_{С.Р3.НН}(t)}{P_{С.Р3.НН}(t)}\Delta\lambda_{Ф}=P_{С.Р3.НН}(t)\Delta_{С.НН};\end{aligned} \tag{51}$$

$$\Delta_{С.НН}=\frac{[1-\frac{3}{2}b^4\lambda^3_{Ф.СР}t^4\kappa^2_{ИЭ}(1-\kappa_{ИЭ})^2]-3b^2\lambda_{Ф.СР}t^2(1-\kappa_{ИЭ})^2[1+\frac{(b\lambda_{Ф.СР}t\kappa_{ИЭ})^2}{2}]}{[1+\frac{(b\lambda_{Ф.СР}t\kappa_{ИЭ})^2}{2}]\{1-\frac{3}{2}[b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})]^2\}}\Delta\lambda_{Ф}. \tag{52}$$

Отклонение интенсивности отказов сложного блока ССН с ненагруженным однократным резервом от своего среднего значения в комплексном режиме $\Delta\Lambda(t)_{С.Р3.НН}$ находится по формуле

$$\Delta\Lambda(t)_{C.РЗ.НН} = \frac{\partial\Lambda(t)_{C.РЗ.НН.СР}}{\partial\lambda_{Ф.СР}}\Delta\lambda_{Ф} \approx [2\lambda_{Ф.СР}t\kappa_{ИЭ} + \\ + \frac{4b^2\lambda_{Ф.СР}t(1-\kappa_{ИЭ})[1+b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})] - 2b^3\lambda^2_{Ф.СР}t(1-\kappa_{ИЭ})^2}{[1+b\lambda_{Ф.СР}t(1-\kappa_{ИЭ})]^2}]\Delta\lambda_{Ф}. \quad (53)$$

4. Анализ надежности объекта в целом по его ССН

Как указывалось ранее, в ССН объекта могут входить простые и сложные блоки с различными видами резервирования, при этом блоки ССН работают в комплексном режиме. В таком случае формула расчета среднего значения интенсивности отказов объекта примет следующий вид:

$$\Lambda_{ОБ.СР}(t_{ОБ}) = \sum_{i=1}^{m_1}\Lambda_{П.i.СР} + \sum_{j=1}^{m_2}\Lambda_{С.Н.j.СР}(t_{ОБ}) + \sum_{v=1}^{m_3}\Lambda_{С.НН.v.СР}(t_{ОБ}), \quad (54)$$

где $\Lambda_{ОБ.СР}$ – среднее значение интенсивности отказов объекта;

$\Lambda_{П.i.СР}$ – среднее значение интенсивности отказов i-го простого блока ССН объекта, вычисляемое по формуле (19);

$\Lambda_{С.Н.j.СР}$ – среднее значение интенсивности отказов j-го сложного блока ССН объекта с нагруженным резервом, вычисляемое по формуле (43);

$\Lambda_{С.НН.v.СР}$ – среднее значение интенсивности отказов v-го сложного блока ССН объекта с ненагруженным резервом, вычисляемое по формуле (50);

m_1 – количество простых блоков в ССН объекта;

m_2 – количество блоков с нагруженным резервом в ССН объекта;

m_3 – количество блоков с ненагруженным резервом в ССН объекта;

$$m_1 + m_2 + m_3 = m. \quad (55)$$

Отклонение интенсивности отказов объекта от своего среднего значения $\Delta\Lambda_{ОБ}$ определяется по формуле

$$\Delta\Lambda_{ОБ} = \frac{\partial\Lambda_{ОБ.СР}(t_{ОБ})}{\partial\lambda_{Ф.СР}} = \sum_{i=1}^{m_1}\frac{\partial\Lambda_{П.i.СР}}{\partial\lambda_{Ф.i.СР}}\Delta\lambda_{Ф.i} + \sum_{i=1}^{m_2}\frac{\partial\Lambda_{С.Н.i.СР}(t_{ОБ})}{\partial\lambda_{Ф.i.СР}}\Delta\lambda_{Ф.i} + \\ + \sum_{i=1}^{m_3}\frac{\partial\Lambda_{С.НН.i.СР}(t_{ОБ})}{\partial\lambda_{Ф.i.СР}}\Delta\lambda_{Ф.i}, \quad (56)$$

где $\frac{\partial\Lambda_{П.i.СР}}{\partial\lambda_{Ф.i.СР}}$, $\frac{\partial\Lambda_{С.Н.i.СР}}{\partial\lambda_{Ф.i.СР}}$, $\frac{\partial\Lambda_{С.НН.i.СР}}{\partial\lambda_{Ф.i.СР}}$ вычисляются по формулам (20), (46), (53) соответственно.

Граничные значения интервальной оценки интенсивности отказов объекта находятся по формулам

$$\Lambda_{ОБ.Н} = \Lambda_{СР.ОБ} - \Delta\Lambda_{ОБ}; \quad (57)$$

$$\Lambda_{ОБ.В} = \Lambda_{СР.ОБ} + \Delta\Lambda_{ОБ}, \quad (58)$$

где $\Lambda_{ОБ.Н}$, $\Lambda_{ОБ.В}$ – соответственно нижнее и верхнее значения интервала интенсивности отказов объекта.

Для обобщенной ССН формула расчета среднего значения вероятности безотказной работы объекта имеет следующий вид:

$$P_{ОБ.СР}(t) = \prod_{i=1}^{m_1}P_{П.i.СР}(t)\prod_{j=1}^{m_2}P_{С.Н.j.СР}(t)\prod_{v=1}^{m_3}P_{С.НН.v.СР}(t), \quad (59)$$

где $P_{ОБ.СР}(t)$ – среднее значение вероятности безотказной работы объекта за наработку t объекта;

$P_{П.i.СР}(t)$ – среднее значение вероятности безотказной работы i-го простого блока ССН за наработку t объекта, вычисляемую по формуле (18);

$P_{С.Н.j.СР}(t)$ – среднее значение вероятности безотказной работы сложного j-го блока ССН с нагруженным резервом за наработку t объекта, вычисляемую по формуле (41);

$P_{С.НН.v.СР}(t)$ – среднее значение вероятности безотказной работы сложного v-го блока ССН с ненагруженным резервом за наработку t объекта, вычисляемую по формуле (48).

Отклонение вероятности безотказной работы объекта от своего среднего значения $\Delta P_{ОБ}(t_C)$ определяется по формуле

$$\Delta P_{ОБ}(t) = \sum_{i=1}^{m_1} \frac{\partial P_{ОБ}(t)}{\partial P_{\text{П.i}}(t)} \Delta P_{П.i}(t) + \sum_{j=1}^{m_2} \frac{\partial P_{ОБ}(t)}{\partial P_{С.Н.j}(t)} \Delta P_{С.Н.j}(t) + \sum_{v=1}^{m_3} \frac{\partial P_{ОБ}(t)}{\partial P_{С.НН.v}(t)} \Delta P_{С.НН.v}(t). \quad (60)$$

Заметим, что

$$\frac{\partial P_{ОБ}(t)}{\partial P_i(t)} = \prod_{j=1, j\neq i}^{m} P_j(t) = \frac{P_{ОБ}(t)}{P_i(t)}. \quad (61)$$

После подстановки в (60) соответствующих значений из (61) и соответствующих значений из (21), (44) и (51) получим

$$\Delta P_{ОБ}(t) = P_{ОБ.СР}(t)\left(\sum_{i=1}^{m_1} \Delta_{П.i} + \sum_{j=1}^{m_2} \Delta_{С.Н.j} + \sum_{v=1}^{m_3} \Delta_{С.НН.v}\right). \quad (62)$$

Нижнее и верхнее граничные значения интервальной оценки вероятности безотказной работы объекта вычисляются соответственно по формулам

$$P_{ОБ.Н}(t) = P_{ОБ.СР}(t) - \Delta P_{ОБ}(t); \quad (63)$$

$$P_{ОБ.В}(t) = P_{ОБ.СР}(t) + \Delta P_{ОБ}(t). \quad (64)$$

5. Пример расчета ПН объекта с учетом погрешностей ПН элементов

Произведем расчет ПН объекта с ССН, приведенной на рис. 6, с учетом погрешностей ПН элементов объекта и отсутствия отказов при хранении в аппаратуре (b = 0). ССН объекта состоит из двух простых блоков ССН 1 и ССН 3, двух сложных блоков ССН 2 с нагруженным резервом и ССН 4 с ненагруженным резервом.

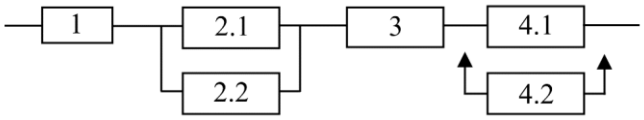


Рис. 6. Структурная схема надежности объекта

Исходными данными для расчета являются:

– количество блоков в ССН m = 4;

– количество элементов в блоках ССН: n_1 = 3, n_2 = 1, n_3 = 4, n_4 = 1;

– интенсивности отказов элементов блоков ССН: нижние значения $\lambda_{1.H} = 1{,}0 \times 10^{-6}$ 1/ч, $\lambda_{2.H} = 2{,}0 \times 10^{-6}$ 1/ч, $\lambda_{3.H} = 2{,}5 \times 10^{-6}$ 1/ч, $\lambda_{4.H} = 1{,}0 \times 10^{-6}$ 1/ч; верхние значения $\lambda_{1.B} = 1{,}2 \times 10^{-6}$ 1/ч, $\lambda_{2.B} = 2{,}5 \times 10^{-6}$ 1/ч, $\lambda_{3.B} = 3{,}1 \times 10^{-6}$ 1/ч, $\lambda_{4.B} = 1{,}3 \times 10^{-6}$ 1/ч;

– коэффициенты интенсивности эксплуатации блоков ССН: $\kappa_{ИЭ.1}$ = 0,70, $\kappa_{ИЭ.2}$ = 0,80, $\kappa_{ИЭ.3}$ = 0,60, $\kappa_{ИЭ.4}$ = 1,00;

– наработка объекта t = 8760 ч = 1 год.

Промежуточные и окончательные данные расчетов приведены в таблице.

Элементы ССН	$\Lambda_{CP}(t) \times 10^{-6}$ 1/ч	$\Delta\Lambda(t) \times 10^{-6}$ 1/ч	$P_{CP}(t)$	$\Delta(t)$
Блок 1	2,310	0,210	0,979 764	0,001 8396
Блок 2	1,210	0,112	0,999 888	0,000 0001
Блок 3	6,720	0,720	0,941 133	0,006 3100
Блок 4	0,575	1,223	0,997 764	0,028 1942
ССН в целом	10,805	1,617	0,921 991	0,028 1942

Результаты вычислений: $\Lambda_H = 9{,}188 \times 10^{-6}$ 1/ч, $\Lambda_{CP} = 10{,}805 \times 10^{-6}$ 1/ч, $\Lambda_B = 12{,}422 \times 10^{-6}$ 1/ч; $P_H(t) = 0{,}895\,996$, $P_{CP}(t_{CAC}) = 0{,}921\,9907$, $P_B(t_{CAC}) = 0{,}947\,9854$.

Заключение

Настоящая работа представляет собой методику расчета ПН объектов с учетом погрешностей ПН их составных частей, формирующих ПН объектов. При этом рассматриваются только невосстанавливаемые объекты, ПН которых обеспечиваются путем использования структурного резервирования с ограниченной кратностью. К числу таких объектов можно отнести бортовую аппаратуру малогабаритных космических аппаратов

Модель надежности объектов, рассмотренная в работе, учитывает погрешности ПН исходных данных, приводящих к погрешности ПН объекта, и работу блоков ССН в режимах непрерывного функционирования, хранения в составе аппаратуры и комбинированном (чередование двух предыдущих).

В работе впервые получены формулы расчета ПН блока ССН со структурным ненагруженным резервом с учетом отказов резервного ненагруженного звена. Применение методики продемонстрировано на примере.

Список литературы

1. Кульбак, Л.И. Интервальная оценка расчетных показателей надежности объекта / Л.И. Кульбак // Информатика. – 2014. – № 1(41). – С. 25–45.

2. Калинкин, В.И. Теория погрешностей / В.И. Калинкин // Материалы лекционного курса «Вычислительная математика» [Электронный ресурс]. – 2016. – Режим доступа : http://icm.muctr.ru/study/maths/lecture_2.pdf. – Дата доступа : 06.07.2016.

3. Менеджмент риска. Структурная схема надежности и булевы методы : ГОСТ Р 51901.14–2007 (МЭК 61078:2006). – Введ. 27.12.07. – М. : Стандартинформ, 2008. – 28 с.

4. Кульбак, Л.И. Оценка надежности бортовой аппаратуры малых космических аппаратов в процессе их полета / Л.И. Кульбак, В.Б. Алюшкевич, С.А. Золотой // Информатика. – 2015. – № 4(48). – С.109–118.

5. Козлов, Б.А. Краткий справочник по расчету надежности радиоэлектронной аппаратуры / Б.А. Козлов, И.А. Ушаков. – М. : Сов. радио, 1966. – 472 с.

Поступила 27.07.2016

Объединенный институт проблем информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: lkulbak@yandex.ru

L.I. Kulbak

RELIABILITY PARAMETERS CALCULATION OF NON-RESTORABLE OBJECTS WITH REGARD TO ERRORS IN THE INITIAL DATA

The technique of reliability parameters calculation of non-restorable objects with regard to errors in the initial data is proposed. The objects, the reliability of which is ensured by structural redundancy with limited multiplicity, are considered. Such objects include an on-board equipment of small spacecrafts.

УДК 004.5; 004.4:004.7

М.К. Буза, О.М. Кондратьева

ПРОГРАММНАЯ СРЕДА ПРОЕКТИРОВАНИЯ ПАРАЛЛЕЛЬНЫХ ПРИЛОЖЕНИЙ С ОБЩЕЙ ПАМЯТЬЮ

Исследуются средства поддержки разработки параллельных программ на языке C/C++. Предлагаются методика и программная среда, которые могут быть использованы для автоматизации процесса проектирования параллельных приложений.

Введение

Разработка эффективных программ для параллельных компьютеров является сложной задачей. В настоящее время ведется значительная работа по упрощению процесса проектирования и реализации параллельных приложений. Существующие средства разработки параллельных программ весьма разнообразны [1]. Представляет интерес разработка инструментальных систем, облегчающих создание и проектирование параллельных программ, а именно объектно-ориентированных библиотек и фреймворков.

При выборе инструментария для разработки параллельной программы хотелось бы руководствоваться возможностью не только создания эффективной программы как главной цели распараллеливания, но и сохранения ее эффективности при переносе с одного целевого компьютера на другой. Однако перечисленные критерии имеют во многом противоречивый характер [2].

В настоящей работе представлена программная среда, которая может быть использована для проектирования параллельных программ с общей памятью (многопоточных программ) на языке C/C++.

1. Средства поддержки проектирования параллельных приложений

Параллельная программа может быть написана на специальном языке или расширении существующего языка, а также на стандартном языке с использованием библиотеки. Одним из преимуществ последнего варианта является то, что среда разработки привычна для программиста. В настоящее время существует ряд кроссплатформенных многопоточных библиотек для языка программирования C++: Boost Thread, C++ Standard Library Thread, Parallel Patterns Library, Threading Building Blocks [3]. Перечисленные программные продукты представляют собой объектно-ориентированные библиотеки и позволяют использовать преимущества объектно-ориентированного и обобщенного программирования языка C++ при разработке параллельных приложений.

Известный пример обобщенного программирования – стандартная библиотека шаблонов C++. Она предоставляет пользователю высококачественные алгоритмы и структуры данных. В стандарте языка C++, принятом в 2011 г. (C++11), появилась развитая поддержка многопоточности [4]. Возможно, многопоточные C++11-программы и уступают по эффективности низкоуровневым средствам, однако явно превосходят их по скорости создания и мобильности.

Одним из механизмов, которые могут увеличить качество и скорость разработки параллельной программы, служат паттерны параллельного программирования. Паттерн в разработке программного обеспечения – это повторимая архитектурная конструкция, представляющая собой решение проблемы проектирования в рамках некоторого часто возникающего контекста [5]. Обычно паттерн является не законченным образцом, который может быть прямо преобразован в код, а лишь примером решения задачи, который можно использовать в различных ситуациях. И если в известной книге [6], посвященной паттернам параллельного программирования, представлены примеры решения задач, то в настоящее время речь идет уже о «алгоритмических скелетах», которые являются высокоуровневыми моделями программирования для

параллельных и распределенных вычислительных систем [7]. В качестве примеров можно назвать такие известные паттерны, как master-slave, pipe, for, while, if, map, divide-conquer.

Объектно-ориентированные библиотеки часто включают реализации различных паттернов. Обычно говорят, что библиотека содержит алгоритмы. Например, Parallel Patterns Library от Microsoft содержит for, for_each, map и другие алгоритмы, а Threading Building Blocks от Intel – for, reduce, do, scan, while, pipeline, sort.

Таким образом, можно констатировать, что параллельное программирование уже накопило общие приемы и превратилось из искусства в технологию. Оно может стать простым и безопасным с помощью предварительно запрограммированных паттернов. Для проектирования параллельной программы потребуется взять готовый и проверенный временем шаблонный код и адаптировать к нему свой алгоритм.

Настоящее исследование направлено на то, чтобы процесс проектирования параллельных программ стал простым и понятным для разработчика. Цель работы – предложить доступную методику проектирования параллельных программ и создать программный инструмент для ее поддержки. В отличие от существующих аналогов здесь используются более общие модели многопоточных программ и существенно упрощается процесс проектирования. При этом предложенная методика ориентирована не только на создателей параллельного программного обеспечения, но и на студентов при обучении их проектированию параллельных программ.

2. Состав среды проектирования

Рассмотрим компоненты разработанной среды проектирования параллельных приложений (рис. 1).

Рис. 1. Компоненты среды проектирования

Общие паттерны параллельных программ. Авторы представляют паттерн как каркас программы, реализующей некоторую модель создания и функционирования потоков. К рассматриваемым моделям относятся: модель делегирования, модель с равноправными узлами, конвейер, изготовитель-потребитель [8]. Каждая модель характеризуется собственной декомпозицией работ, которая определяет, кто отвечает за создание потоков, при каких условиях они создаются, как потоки взаимодействуют. При разработке параллельной программы предлагается выбрать подходящий паттерн. Решение задач, связанных с коммуникацией, синхронизацией и балансировкой нагрузки, уже реализовано в выбранном паттерне.

Специализированные паттерны параллельных программ. Каждый специализированный паттерн является конкретизацией общего паттерна для решения определенного множества задач. Например, имеющиеся паттерны для решения матричных задач реализуют различные общие паттерны, а также позволяют определить способ распределения данных (полосы или бло-

ки, непрерывный или циклический), размерность матрицы (одномерная, двухмерная, трехмерная), вариант заполнения матрицы (генератор случайных чисел, из файла).

База данных параллельных приложений содержит готовые решения разнообразных задач параллельного программирования: матричных вычислений, решений систем линейных уравнений, сортировки, обработки графов, уравнений в частных производных, многоэкстремальной оптимизации.

Потокобезопасные контейнеры. Для многопоточных программ существует проблема разделения данных между потоками. Операция «читать – модифицировать – писать» должна быть атомарной в качестве общего решения этой проблемы. Предлагаемая среда проектирования содержит потокобезопасные контейнеры: вектор, очередь и стек. Названные контейнеры используются в паттернах параллельных программ, например в моделях, которые работают с очередью заданий. Пользователи могут напрямую включать их в свои программы.

Мастер создания остова (каркаса) приложения представляет собой оконный интерфейс пользователя, посредством которого он выбирает и настраивает паттерн параллельной программы.

Скрипты для проведения экспериментов автоматизируют запуск приложений, сохраняют замеры времени и рассчитывают показатели эффективности.

3. Методика проектирования параллельных приложений

В основу предлагаемой методики проектирования параллельных программ положена PCAM (Partitioning, Communication, Agglomeration, Mapping) [9], которая включает следующие этапы:

– разделение вычислений на независимые части;

– выделение информационных зависимостей;

– масштабирование набора подзадач;

– распределение подзадач между вычислительными элементами (в многопоточных программах обычно выполняется операционной системой).

Предлагаемая среда проектирования поддерживает все этапы разработки параллельной программы. Поскольку выделение информационных зависимостей и разделение вычислений на независимые части существенно зависят от рассматриваемой задачи, соответствующие этапы частично реализуются при создании паттернов, а частично − при адаптации (конкретизации) паттерна к решаемой задаче. Рассмотрим процесс разработки с использованием среды (рис. 2).

Рис. 2. Взаимодействие пользователя со средой проектирования

Первым этапом разработки параллельного приложения является *создание остова приложения*. Для этого используется мастер, который включает следующие шаги:

Шаг 1. Определение типа приложения:

C++11 (технология высокого уровня) – паттерны, написанные на стандарте C++11 языка C++, в который включена поддержка параллелизма [4];

WinAPI (технология низкого уровня) – встроенные потоки операционной системы Microsoft Windows [10];

PThread (технология низкого уровня) – потоки POSIX Thread Library [10];

Author Thread (технология высокого уровня) – собственная объектно-ориентированная библиотека, которая инкапсулирует интерфейс низкого уровня.

Шаг 2. Выбор общего паттерна параллельной программы. Предлагаются общие паттерны, реализующие следующие модели создания и функционирования потоков:

– модель делегирования в двух вариантах: управляющий поток создает новый поток для каждого нового запроса и управляющий поток создает пул потоков, которые обрабатывают все запросы;

– модель с равноправными узлами в двух вариантах: общий поток входных данных и собственный входной поток у каждого потока;

– конвейер;

– модель «изготовитель – потребитель».

Шаг 3. Пользователь может для решения своей задачи подобрать специализированный паттерн, который реализует выбранную на предыдущем шаге модель для определенного класса задач. Предлагаются варианты:

– матричные задачи;

– анализ данных из файлов.

Шаг 4. Выбор конкретного приложения из базы данных параллельных приложений, если оно подходит для решения задачи пользователя. Это приложение удовлетворяет параметрам выбора, установленным на предыдущих шагах.

Шаг 5. Установка дополнительных параметров. Предлагается включить в приложение замеры времени работы, определить количество потоков, задать способ подготовки входных данных для тестирования параллельной программы: ручной или автоматический (генерация файлов).

После завершения работы мастера создания остова приложения получаем каркас или уже законченное многопоточное приложение. Этапы проектирования (декомпозиция вычислений, их координация и масштабирование) нашли свое отражение в тех паттернах, которые выбирал пользователь. Каркасы используют в качестве параметра количество потоков, обеспечивая зависимость правил масштабирования от количества вычислительных элементов. Также есть возможность использовать как высокоуровневые, так и низкоуровневые средства.

Второй этап проектирования параллельного приложения – *ручная конкретизация остова приложения*. Полученный с помощью мастера каркас приложения вручную дорабатывается пользователем. Трудоемкость этого этапа существенно зависит от того выбора, который был сделан на первом этапе. Если пользователь на четвертом шаге подобрал подходящую программу из базы данных параллельных приложений, то ручная конкретизация не требуется. Если подходящая программа не была найдена, но на третьем шаге был выбран специализированный паттерн, то требуется определить функцию, которую будет выполнять каждый поток. Если пользователь выбрал только общий паттерн, то требуется максимально трудоемкая доработка: определить функцию потока, конкретизировать типы данных. Ясно, что ручная конкретизация сильно зависит от типа приложения.

Третий этап проектирования – *проведение вычислительных экспериментов* с целью определения эффективности разработанной программы. Для этого выполняют последовательную и параллельную версии программы, измеряют время выполнения и определяют показатели эффективности. При отсутствии последовательной версии программы можно воспользоваться параллельной версией, установив количество потоков равным единице.

Одной из главных характеристик параллельной программы является ускорение:

$$S_p(n) = T_1(n) / T_p(n),$$

где $T_1(n)$ − время выполнения последовательной версии программы; $T_p(n)$ − время выполнения параллельной программы на p-процессорной вычислительной системе.

Еще одна важная характеристика параллельной программы – эффективность:

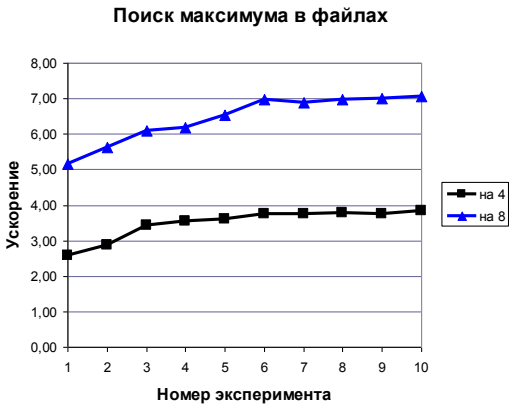
$$E_p(n) = T_1(n) / (pT_p(n)) = S_p(n) / p.$$

Для помощи пользователю написаны скрипты, которые автоматизируют запуск многопоточной программы с различным количеством потоков и для различных входных данных, сохранение замеров времени в файле и вычисление показателей эффективности.

Результаты экспериментов можно просмотреть как в текстовом, так и графическом виде. В качестве примера на рис. 3 показан фрагмент данных, которые были получены при решении задачи поиска максимума в файлах. Для каждого теста (входного набора данных) указываются результаты выполнения в виде списка значений «время_выполнения (количество_потоков)».

```
Тест 1:     3,1(1),    1,2(4),    0,6(8)
Тест 2:    12,2(1),    5,3(4),    2,7(8)
Тест 3:    31,1(1),    9,1(4),    5,1(8)
Тест 4:   148,7(1),   42,0(4),   24,0(8)
Тест 5:   298,0(1),   82,5(4),   45,6(8)
Тест 6:   600,1(1),  160,0(4),   86,0(8)
Тест 7:  1197,0(1),  318,0(4),  174,0(8)
Тест 8:  2420,0(1),  639,0(4),  347,0(8)
Тест 9:  4837,0(1), 1290,0(4),  692,0(8)
Тест 1:  9760,0(1), 2551,0(4), 1380,0(8)
```

а)



б)

Рис. 3. Результаты эксперимента: *а)* фрагмент текстового файла;
б) графики ускорений для четырех и восьми потоков

Четвертый этап проектирования – *анализ результатов*. Процесс проектирования имеет итеративную природу. Если в результате анализа эффективности параллельной программы разработчик будет неудовлетворен результатом, то он может перепроектировать свою программу, вернувшись на предыдущие этапы. Существует важное преимущество предлагаемого подхода – легкость перехода от одной модели функционирования потоков к другой, эти изменения выполняются автоматически.

4. Апробация и области применения среды проектирования

Исследование среды проектирования проводилось посредством использования предлагаемого инструмента для проектирования некоторых распространенных задач параллельного программирования. В таблице представлены основные результаты экспериментов – среднее время, затраченное на разработку параллельного приложения для решения типовой задачи с использованием разных средств проектирования. Разработка приложений осуществлялась студентами третьего курса факультета прикладной математики и информатики, специализация «Прикладная информатика».

На новом уровне был продолжен эксперимент по распараллеливанию сложной прикладной программы − пакета моделирования методом молекулярной динамики XMD (Molecular

Dynamics for Metals and Ceramics). В работе [11] к названному пакету были добавлены многопоточные версии функций вычисления сил в случае полупроводниковых ковалентных материалов. С помощью среды проектирования легко были получены несколько вариантов распараллеливания: модели с равноправными узлами и модели делегирования. Эти модели обычно дают показатели эффективности, близкие по значению. Однако существует вероятность того, что на разных вычислительных системах могут быть получены существенно разные показатели эффективности.

Среднее время разработки параллельного приложения, ч

Задача	Используемая модель	Встроенные потоки	Общий паттерн	Специализированный паттерн
Вычисление значения определенного интеграла	Модель с равноправными узлами	2,5	1,25	0,7
Поиск максимума в матрице	Модель делегирования	2,5	1,25	0,7
Умножение матриц	Модель с равноправными узлами	2,5	1,25	0,7
Внутренняя сортировка	Модель с равноправными узлами	3	2	1
Внешняя сортировка	Модель делегирования	4,5	2,5	1,5
Внешняя сортировка	Конвейер	5	3	1,7

Полученные результаты иллюстрируют снижение трудоемкости разработки, а также подтверждают, что абстрагирование и использование паттернов позволяют ускорить и улучшить процесс проектирования параллельного кода.

Обобщенные средства разработки, как правило, порождают не самый эффективный код. Этот недостаток может быть ослаблен на этапе ручной конкретизации остова приложения.

Универсальные средства программирования обычно ориентированы на ограниченный класс приложений. В том случае если приложение не вписывается в модель программирования, принятую в системе разработки, программист не может ею воспользоваться. Эта проблема решается программистом за счет введения собственных расширений.

Рассмотрим возможные способы применения разработанной системы. Наиболее органичным способом является использование ее при обучении технологиям параллельного программирования, в первую очередь при освоении тем «Распараллеливание последовательных программ» и «Технологии параллельного программирования с использованием шаблонов» [12].

Применение разработанной среды проектирования повышает культуру программирования. То, что может написать начинающий программист, существенно отличается от параллельных приложений, созданным опытным разработчиком. Примером может служить «простая реализация параллельной версии аккумулятора» [4].

Заключение

В настоящей работе представлена авторская инструментальная система, предназначенная для проектирования параллельных программ с общей памятью на языке C/C++. При проектировании программ используются готовые модели, что делает процесс написания программ более простым. Использованы преимущества объектно-ориентированного и обобщенного программирования.

Разработанная программная среда автоматизирует процесс проектирования и исследования эффективности параллельной программы, избавляет программиста от рутинной работы и позволяет писать многопоточный код проще и с меньшим количеством ошибок. Поддерживается возможность получать оценки применения различных моделей при решении конкретных задач.

Полученные при апробации среды результаты иллюстрируют снижение трудоемкости разработки, а также подтверждают, что избранная методика позволяет ускорять и улучшать процесс проектирования параллельного кода, при этом несущественно влияя на эффективность исполнения созданных программ.

Разработанная среда может быть использована при обучении технологиям параллельного программирования, позволяя выбирать и оценивать тот или иной подход для создания собственных эффективных параллельных приложений.

Проект выполнен при частичной поддержке ГПНИ «Информатика и космос».

Список литературы

1. Средства разработки параллельных программ [Электронный ресурс]. – Режим доступа : https://parallel.ru/tech/tech_dev. – Дата доступа : 15.06.2016.

2. Воеводин, В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. – СПб. : БХВ – Петербург, 2002. – 608 с.

3. List of C++ multi-threading libraries [Electronic resource]. – Mode of access : https://en.wikipedia.org/wiki/List_of_C%2B%2B_multi-threading_libraries. – Date of access : 15.06.2016.

4. Уильямс, Э. Параллельное программирование на C++ в действии. Практика разработки многопоточных программ / Э. Уильямс. – М. : ДМК Пресс, 2012. – 672 с.

5. Шаблон проектирования [Электронный ресурс]. – Режим доступа : https://ru.wikipedia.org/wiki/Шаблон_проектирования. – Дата доступа : 15.06.2016.

6. Mattson, T.G. Patterns for Parallel Programming / T.G. Mattson, B.A. Sanders, B.L. Massingill. – Addison Wesley, 2002. – 300 p.

7. Algorithmic skeleton [Electronic resource]. – Mode of access : https://en.wikipedia.org/wiki/Algorithmic_skeleton. – Date of access : 15.06.2016.

8. Хьюз, К. Параллельное и распределенное программирование с использованием C++ / К. Хьюз, Т. Хьюз. – М. : Вильямс, 2004. – 672 с.

9. Foster, I. Designing and Building Parallel Programs: Concepts and Tools for Parallel / I. Foster. – Addison Wesley, 1995. – 430 p.

10. Буза, М.К. Проектирование программ для многоядерных процессоров : учеб.-метод. пособие для студентов факультета прикладной математики и информатики / М.К. Буза, О.М. Кондратьева. – Минск : БГУ, 2013. – 48 с.

11. Буза, М.К. Параллельная реализация метода молекулярной динамики на многоядерном процессоре / М.К. Буза, О.М. Кондратьева // Информатика. – 2011. – № 1(29). – С. 44–51.

12. Технологии параллельного программирования II (2012871) [Электронный ресурс]. – Режим доступа : http://open.ifmo.ru/wiki/Технологии_параллельного_программирования_II_(2012871). – Дата доступа : 15.06.2016.

Поступила 08.08.2016

Белорусский государственный университет,
Минск, пр. Независимости, 4
e-mail: bouza@bsu.by,
kondratjeva@bsu.by

M.K. Bouza, O.M. Kondratjeva

SOFTWARE FOR DESIGNING PARALLEL APPLICATIONS

The object of research is the tools to support the development of parallel programs in C/C ++. The methods and software which automates the process of designing parallel applications are proposed.

НАУЧНОЕ НАСЛЕДИЕ

УДК 681.3

П.Н. Бибило, Ю.В. Поттосин, Л.Д. Черемисинова

О НАУЧНОМ НАСЛЕДИИ ЧЛЕНА-КОРРЕСПОНДЕНТА А.Д. ЗАКРЕВСКОГО

Посвящается научному наследию Аркадия Дмитриевича Закревского, который стоял у истоков зарождения кибернетики в Советском Союзе и являлся основателем одной из самых известных школ логического проектирования в Советском Союзе и мире.

В ноябре 2016 г. лаборатории логического проектирования Объединенного института проблем информатики НАН Беларуси исполнилось 45 лет. Она была создана в 1971 г. доктором технических наук Аркадием Дмитриевичем Закревским под названием «лаборатория системного программирования и логического синтеза». Научные исследования, которыми руководил Аркадий Дмитриевич в этих двух областях науки, были начаты задолго до 1971 г., еще в Томском государственном университете.

Начало научных исследований. Томский период

Научную деятельность Аркадий Дмитриевич Закревский начал в середине XX в. с момента широкого распространения и бурного развития вычислительной техники в Советском Союзе. Тогда было налажено серийное производство электронных вычислительных машин «Урал», одна из которых была установлена в Томском государственном университете. Это была первая и в то время единственная ЭВМ в Сибири. Ознакомившись с устройством «Урала», А.Д. Закревский предложил свой вариант равносильной ЭВМ, обладающей в десятки раз меньшим количеством аппаратуры. Дело в том, что оперативная память «Урала» размещалась на магнитном барабане, который вращался со скоростью 6000 оборотов в минуту. За один оборот машина выполняла одну операцию: арифметическую или логическую. Арифметическое устройство работало по принципу параллельно-последовательного действия, требовавшему использования многоразрядного сумматора и многоразрядных регистров для хранения промежуточных результатов. Аркадий Дмитриевич предложил располагать упомянутые регистры на том же магнитном барабане и использовать принцип последовательного действия. Операции выполнялись бы за один оборот барабана, а для их выполнения достаточно было нескольких триггеров. Соответствующий проект был опубликован в статье [1], где для описания устройств широко применялся аппарат булевой алгебры. Студентам этот материал давался как пример использования булевых функций в проектировании дискретных устройств. Данный проект не был реализован, поскольку к тому времени появились более быстродействующие, надежные и компактные устройства оперативной памяти, чем магнитный барабан.

В основе элементной базы логических схем в то время лежали схемы, состоящие из полупроводниковых диодов и сопротивлений. Например, логические схемы упомянутой ЭВМ «Урал» насчитывали свыше тысячи полупроводниковых диодов. Статья [2] посвящена оптимизации таких схем. В ней описан метод, применение которого позволило заметно сократить число диодов в конкретной схеме.

С развитием вычислительной техники выросла потребность в использовании методов логического проектирования. Следует отметить широту охвата проблем логического проектирования в научных исследованиях А.Д. Закревского. Его работы 1960-х гг. посвящены довольно разнообразным задачам из этой области: теории автоматов, теории булевых функций, синтезу комбинационных схем в заданном базисе логических элементов, надежности логических схем, системному программированию. Многие из полученных результатов являются пионерскими, они не потеряли актуальности и оригинальности и в настоящее время.

Оптимизация функциональных описаний логических схем. В одной из первых работ Аркадия Дмитриевича [3] была решена задача минимизации булевых функций в классе дизъюнктивных нормальных форм, которая занимала центральное место в оптимизации комбинационных логических схем. Помимо, собственно, метода минимизации в статье предлагался инструментарий, который значительно облегчал восприятие булева пространства и позволял эффективно решать различные задачи, связанные с булевыми функциями. Систематизации построений в булевом пространстве, полезных при решении широкого класса логических задач, позднее была посвящена статья [4]. Особое место в задаче минимизации булевых функций занимало рассмотрение не полностью определенных функций. Ранее проблема их использования в практике логического проектирования в советской и зарубежной литературе практически не обсуждалась. В работе [5] предложен метод минимизации слабо определенных функций, т. е. функций, значения которых заданы на небольшой части булева пространства. В отличие от общего подхода к минимизации не полностью определенных функций предложенный метод частично избавлял от рассмотрения обширной области булева пространства, где значения функции не определены.

Значительным вкладом в теорию логического проектирования стали работы А.Д. Закревского по методам решения оптимизационных задач синтеза последовательностных схем. Предложенный метод последовательных сокращений для решения задачи минимизации числа состояний конечного автомата [6] позволял получить автомат, реализующий исходный автомат с числом состояний, близким к минимальному, а во многих случаях и равным ему. При этом затраты времени на реализацию данного метода оценивались полиномом третьей степени от числа состояний исходного автомата. Идея использовать модель конечного автомата для разложения сложного процесса проектирования управляющей системы на относительно простые действия [7] представляла собой огромный шаг на пути автоматизации проектирования.

Решение логико-комбинаторных задач. Значительное место в научном наследии А.Д. Закревского занимала проблема формализации и решения логико-комбинаторных задач, имеющих приложение в логическом проектировании и дискретной математике в целом. Методика решения (и программирования) таких задач, основанная на сокращении перебора вариантов при решении, изложена в статье [8]. В частности, одна из обширного списка рассмотренных логико-комбинаторных задач о покрытии, к которой сводятся многие оптимизационные задачи дискретной математики, была сформулирована на абстрактном уровне [9] и были даны алгоритмы ее решения в различных постановках [10].

Перечисленные работы в дальнейшем образовали алгоритмическую базу для создания автоматизированных систем логического проектирования.

Надежность логических схем. Не осталась без внимания А.Д. Закревского и проблема надежности логических схем. Метод повышения надежности логических схем представлен в статье [11], где предложено использовать результаты техники передачи сигналов без потери информации при наличии помех в канале связи. При этом небольшое усложнение схемы позволило обеспечить ее устойчивость к одиночным неисправностям.

Аркадий Дмитриевич также предложил общий подход к логическому проектированию схем, преобразующих информацию в коде Хэмминга, в рамках которого была решена задача синтеза многовыходных логических схем, функционально-устойчивых относительно выхода из строя подсхем, реализующих отдельные функции. При решении этой задачи впервые был применен помехоустойчивый код Хэмминга. Для случая контактных схем разработаны методы синтеза схем, продолжающих исправно функционировать при ложных замыканиях отдельных контактных пар.

Еще в конце 1950-х гг. А.Д. Закревский заинтересовался проблемами криптографии. Его статья, где был описан подход, использующий теорию автоматов для решения задач шифрования сообщений, не была в то время опубликована, так как представленные в ней результаты объявили «совершенно секретными». Однако именно она дала начало научному направлению, которое с тех пор развивается в Томском университете. Статья [12] была опубликована после снятия грифа секретности.

Автоматизация логического проектирования. В середине XX в. работы по автоматизации логического проектирования часто сводились к созданию специализированных вычислительных устройств, реализующих некоторые конкретные алгоритмы логического синтеза. По сравнению с универсальными ЭВМ такие устройства являются более эффективными, но не обладают возможностью варьирования параметров этих алгоритмов. В работе [13] высказана идея совмещения достоинств специализированных логических машин с преимуществами программного управления универсальной ЭВМ путем создания к ней специальной приставки. Приставка была названа *L*-машиной, а система «универсальная ЭВМ плюс приставка» – *L*-системой. Программа для *L*-системы, названная *L*-программой, состояла из команд двух видов: *U*-команды, реализуемой непосредственно на ЭВМ, и *L*-команды, реализуемой с помощью *L*-машины. Основными объектами, операции над которыми должны производиться на *L*-машине, считались булевы функции, и пространством их представления было принято множество вершин многомерного булева гиперкуба. Конструктивно *L*-машина состояла из блока управления, предназначенного для управления реализацией *L*-команд на *L*-машине, и блока многомерных полей, совмещающего функции хранения и преобразования информации. Структура многомерного поля подобна булеву гиперграфу, и каждое такое поле способно хранить информацию о булевой функции. Одно из многомерных полей, названное основным, играло роль сумматора, подобного сумматору одноадресной ЭВМ. На нем выполнялись *L*-команды и фиксировался результат. В качестве примера приведено решение задачи декомпозиции булевой функции на *L*-машине. Моделирование *L*-системы на универсальной ЭВМ доказало ее эффективность.

Однако увеличение быстродействия и возможностей памяти ЭВМ показало, что более перспективным направлением, чем создание специализированных машин и приставок, являлась разработка специализированных языков программирования для универсальных ЭВМ. Такой язык, язык ЛЯПАС (Логический Язык Представления Алгоритмов Синтеза), и был предложен А.Д. Закревским. В своей первой монографии [14] он описал этот язык и показал его возможности на примерах описаний алгоритмов решения разнообразных задач синтеза дискретных автоматов. Позже был издан сборник статей [15], описывающих сам язык ЛЯПАС, его реализацию на ЭВМ и серию алгоритмов решения задач логического проектирования. Кроме Аркадия Дмитриевича авторами статей являлись и его ученики. Сборник был переведен на английский язык и издан за рубежом [16], что свидетельствует о мировом признании научного направления, открытого А.Д. Закревским.

Научные исследования. Минский период

В 1971 г. доктор технических наук А.Д. Закревский и группа его учеников переехали в Минск, где 26 ноября 1971 г. в Институте технической кибернетики АН БССР была организована лаборатория системного программирования и логического синтеза. В 1972 г. Аркадий Дмитриевич избирается членом-корреспондентом АН БССР. С 1971 по 2014 г. им были опубликованы 24 монографии.

Автоматизация синтеза дискретных автоматов. В Минске Аркадий Дмитриевич продолжил заниматься теоретическими вопросами автоматизации проектирования дискретных управляющих устройств. Монография «Алгоритмы синтеза дискретных автоматов» [17] явилась фундаментальным научным трудом, где был предложен и развит комплексный подход к проблеме автоматизации проектирования дискретных устройств, основанный на формализации задач и раскрытии их логико-комбинаторной природы, разработке точных и приближенных алгоритмов и программ решения задач теории графов, минимизации булевых функций, а также анализа и синтеза конечных автоматов. В книге описан и язык ЛЯПАС, ориентированный на проблемную область автоматизации проектирования дискретных устройств, приведены

тексты программ на данном языке, составившие библиотеку базовых программ решения комбинаторных задач логического проектирования. Книга стала своеобразным «источником задач» для многочисленных учеников А.Д. Закревского, которые продолжили развитие его идей и довели их до практического использования. Результатом коллективной работы явилась монография под редакцией Аркадия Дмитриевича «Синтез асинхронных автоматов на ЭВМ» [18], посвященная описанию системы «Автомат-74» и реализованным в ней методам проектирования.

Система «Автомат-74» была одной из первых в СССР систем сквозного проектирования конечных автоматов, в которой использовался классический подход к синтезу логических схем, реализующих дискретные (конечные) автоматы. Основные этапы этого подхода:

– минимизация числа состояний полностью и частично определенных автоматов;

– противогоночное кодирование состояний и получение булевых функций возбуждения элементов памяти (триггеров);

– совместная минимизация систем булевых функций в классе дизъюнктивных нормальных форм (ДНФ);

– построение логической схемы в базисе элементов И–НЕ, И–ИЛИ–НЕ;

– разбиение и покрытие схемы логическими элементами серии К133 малой степени интеграции.

Система была реализована при помощи программирующей системы ЛЯПАС-71 на ЭВМ М-222. Исходные данные вводились с перфокарт, программы выполнялись по модулям согласно этапам синтеза, диалог отсутствовал, результаты промежуточных вычислений выдавались на печать. Классические методы Квайна – МакКласки минимизации булевых функций были развиты для случая совместной минимизации систем не полностью определенных (частичных) булевых функций. Оптимизация логических схем осуществлялась на основе факторизации логических выражений, задающих совместно минимизированные ДНФ.

Решение логических уравнений. Основной вклад создателя лаборатории логического проектирования в логику отражает монография «Логические уравнения» [19] (1975). Здесь проведена классификация и разработаны методы решения логических уравнений из выделенных классов, предложен комбинаторный подход к решению уравнений и компактному представлению множества всех корней логических уравнений, заданных в форме ДНФ, на основе которого разработаны практические методы быстрого поиска одного, всех или заданного числа корней. В книге рассмотрены как тривиальные методы решения уравнений, так и методы перемножения ДНФ, лексикографического перебора, минимизации дерева поиска, основанные на сокращенном переборе аргументов и уравнений системы, локальной редукции, метод «отраженных волн» и распространения констант и др. Классическими задачами, основанными на решении уравнений частного вида в виде ДНФ и конъюнктивных нормальных форм (КНФ), являются задачи анализа булевой матрицы на вырожденность минимизации булевых функций и проверки выполнимости КНФ, которые лежат в основе основных подходов к формальной верификации логических схем. Решение последней задачи настолько важно в настоящее время, что регулярно проводится чемпионат мира между эффективными программами решения задач выполнимости КНФ большой размерности. В книге представлены также полисиллогизмы – суждения, содержащие более двух посылок. Заметим, что полную систему силлогизмов описал Аристотель, это и составило основное содержание логики, изучавшейся на протяжении двух тысячелетий в университетских курсах.

Логические уравнения оказались удобной формальной моделью, пригодной для решения разнообразных задач. К решению таких уравнений в лаборатории логического проектирования были сведены задачи декомпозиции булевых функций и систем функций, при этом различные виды декомпозиции потребовали учета различных логических условий. Сюда можно отнести и задачи минимизации площади транзисторных структур с разрывами шин, технической диагностики и другие из многих проблемных областей.

Решение больших систем логических уравнений. В более поздних работах А.Д. Закревского был выделен класс больших систем логических уравнений (БСЛУ), который имел большое практическое значение для решения задач проектирования, диагностики, распознавания, защиты информации и др. БСЛУ в монографии «Решение больших систем логических уравнений» [20] (2009) характеризуются сотнями переменных и тысячами уравнений. Важной их осо-

бенностью является то, что каждое из уравнений содержит относительно небольшое число (не более десятка) переменных. Данное ограничение позволило предложить оригинальные высокоэффективные методы решения БСЛУ, в основе которых лежат минимизация дерева поиска, сокращенный перебор аргументов и уравнений системы, распространение констант, силлогизмы, коллапсирование уравнений, локальная редукция и др. Выделен практически важный подкласс БСЛУ с малым числом корней. Разработаны методы решения типичных для диагностики и распознавания задач проверки системы на выполнимость, нахождения единственного корня системы.

Приведена классификация систем линейных логических уравнений (СЛЛУ), и решены для них две трудные и практически важные задачи: нахождения кратчайшего решения неопределенной СЛЛУ и общего корня максимального числа уравнений противоречивой системы. Разработана серия эффективных методов их решения, обобщающих метод Гаусса канонизации системы путем рандомизированного построения множества эквивалентных канонических форм, решаемых параллельно, что обеспечивает снижение уровня перебора и сокращение времени решения на несколько порядков. Показана эффективность предложенных методов для криптоанализа машины Hagelin M-209, используемой немцами во время Второй мировой войны. Отдельно изучался класс линейных логических уравнений. Предложенные методы решения БСЛУ были реализованы программно, проведено экспериментальное исследование программ, подтвердившее их высокую эффективность.

Автоматизация программирования. В 1970–1980-е гг. на базе предложенного А.Д. Закревским языка ЛЯПАС были разработаны языки ЛЯПАС-71 и ЛЯПАС-М, которые содержали более развитые системы команд. Они легли в основу высокоэффективных по компактности и быстродействию систем программирования, реализованных для отечественных ЭВМ типа БЭСМ-6, СМ-4, «Минск-32», ЭВМ серии ЕС и ПЭВМ ЕС-1840. Монография «Система программирования ЛЯПАС-М» [21] зафиксировала новый стандарт ЛЯПАС-М, сменивший вторую версию языка – ЛЯПАС-71. В языке ЛЯПАС-М символика была приближена к стандартным алфавитам отечественных устройств отображения информации, включены новые операции над символами и двухмерными логическими массивами, названными комплексами. Созданные программирующие системы ЛЯПАС-М для ЭВМ серии ЕС содержали развитый набор программных средств, достаточных для разработки сложных комплексов автоматизации проектирования.

Следует отметить, что язык ЛЯПАС продолжает развиваться в Томском государственном университете, опубликовано описание новой версии ЛЯПАС-Т, заключающейся в увеличении длины операндов и расширении множества элементарных операций над ними.

Логическое проектирование. Развитие микроэлектроники привело к появлению сверхбольших интегральных схем (СБИС) и вызвало широкое применение в 1990-е гг. регулярных матричных структур, в частности программируемых логических матриц (ПЛМ), реализующих системы ДНФ булевых функций. ПЛМ выпускались в виде отдельных микросхем и применялись в качестве макроэлементов в составе заказных СБИС. Основы проектирования дискретных устройств на базе ПЛМ были заложены А.Д. Закревским в монографии «Логический синтез каскадных схем» [22], где он предложил матричный аппарат, впервые сформулировал и решил многие задачи анализа, синтеза и диагностики матричных схем. Критериями оптимизации при синтезе схем в базисе ПЛМ(t, q, s) с ограниченными параметрами (t, s и q – числа входных, выходных полюсов и промежуточных шин соответственно) являлись сложность (число микросхем) и быстродействие схемы. При реализации ДНФ функции с числом n входных, m выходных переменных и k конъюнкциями на ПЛМ с ограниченными параметрами учет ограничений $k \le q$, $m \le s$ достаточно просто осуществлялся с помощью декомпозиций «по промежуточным шинам» и «по выходам». При этом использовались возможности монтажной логики («проводного» ИЛИ) для выходных полюсов микросхем. Декомпозиция «по промежуточным шинам» сводилась к дизъюнктивному разложению системы ДНФ (или системы булевых функций в общем случае), а декомпозиция «по выходам» – к разбиению системы функций на подсистемы с ограниченным числом функций.

В монографии [22] предложены новые виды декомпозиции систем ДНФ: стандартная, комбинированная, параллельная, матричной факторизации, группирования ортогональных конъюнкций. На их основе разработаны декомпозиционные методы синтеза одноярусных и многоярусных сетей в базисе элементарных матричных схем и ПЛМ. Построение одноярусных схем возможно для систем ДНФ с короткими термами, длина которых не превышает число t входных полюсов ПЛМ.

Синтез одноуровневых схем в базисе ПЛМ представляет собой сложную комбинаторную задачу покрытия матриц T^x, B^f, задающих систему ДНФ(n, k, m) множеством k конъюнкций, соответствующих k строкам n-столбцовой троичной матрицы T^x, и распределением конъюнкций по ДНФ системы, задаваемым m столбцами k-строчной булевой матрицы B^f. Пара матриц покрывается парами подматриц T_i^x, $B_i{}^f$ ограниченных размеров. Так, матрица T^x покрывается подматрицами T_i^x, имеющими не более t столбцов и не более q строк; матрица B^f – подматрицами $B_i{}^f$, имеющими не более s столбцов и соответствующее T_i^x число строк. Подматрицы каждой пары формируются на одном и том же множестве строк матриц T^x, B^f, и каждая подматрица T_i^x, $B_i{}^f$ определяет одну ПЛМ(t, q, s) строящейся сети.

В том случае когда система ДНФ не удовлетворяла условию одноярусной реализации (длина некоторых конъюнкций превышала число t входных полюсов ПЛМ), синтез сети из ПЛМ сводился к задаче построения многоярусных сетей, решаемой на основе декомпозиции по входам «большой» ПЛМ, соответствующей реализуемой системе ДНФ. Для этого случая, в частности, был предложен оригинальный метод тождественных отображений в пространстве промежуточных переменных. Особенностью данного метода является то, что исходная «большая» ПЛМ разбивается на две взаимосвязанные ПЛМ, одна из которых (первая от входов сети) имеет t входов и, следовательно, удовлетворяет условию реализации одноярусной сетью базовых ПЛМ с ограниченными параметрами, а вторая не удовлетворяет и должна быть, в свою очередь, декомпозирована аналогичным путем. Метод тождественных отображений позволяет итеративным способом строить сети из элементов с ограниченным числом входов. В этом методе получение промежуточных переменных разложения сведено к комбинаторной задаче кодирования интервалов таким образом, что пересекающиеся части интервалов были закодированы ортогональными кодами.

Содержание данной фундаментальной монографии не исчерпывалось построением сетей ПЛМ. В ней были рассмотрены также методы минимизации площади одной ПЛМ, реализуемой в виде макроэлемента, причем исходное описание представляло собой систему полностью определенных, частичных (не полностью определенных) булевых функций, заданных на наборах и интервалах значений входных переменных. Рассмотрены проблемы комбинаторного поиска, минимизации систем и реализации на ПЛМ дискретных устройств, заданных моделями секвенциальных и микропрограммных автоматов, а также проблемы диагностики матричных схем.

Логическое распознавание. Одним из направлений деятельности лаборатории, которой руководил А.Д. Закревский, наряду с логическим проектированием являлось логическое распознавание. Основные результаты по этой теме отражены в монографии «Логика распознавания» [23] (1988). Здесь излагается логический подход к проблеме распознавания в пространстве дискретных признаков, использующий общий аппарат для индуктивного вывода на этапе извлечения знаний из данных и дедуктивного вывода на этапе предсказания целевых свойств частично наблюдаемого объекта. Разработаны метод выявления импликативных закономерностей в булевом пространстве признаков и построения соответствующей базы знаний, а также оригинальные методы дедуктивного вывода на этапе распознавания. Предложенные методы обобщены на случай многозначных признаков с использованием введенного аппарата конечных предикатов.

Теоретические основы логики распознавания нашли отражение в созданной в лаборатории экспертной системе «ЭКСИЛОР» (ЭКспертной СИстеме ЛОгического Распознавания в пространстве дискретных признаков). В ней распознавание сведено к решению логических уравнений над конечными предикатами – двухзначными (0,1) функциями многозначных (конечно-значных) аргументов. Знания о предметной области поданы в виде импликативных закономерностей и трактуются как информация о невозможных сочетаниях значений признаков. Закономерности выражены в форме конъюнктов-запретов или дизъюнктов. Выявленные зако-

номерности, имеющие место для объектов распознавания, вводились в систему «ЭКСИЛОР» экспертом или получались путем индуктивного вывода на основании выборки данных, после чего рассматривались в качестве аксиом при дедуктивном выводе (распознавании). Система распознавания снабжена средствами объяснения вывода. Главными информационными блоками системы, реализованной на ПЭВМ, являлись база знаний, задающая закономерности, и база данных, в которой содержатся описания объектов в пространстве признаков. Подход, заложенный в основу создания ЭКСИЛОР, был использован при разработке распознающих экспертных систем в конкретных проблемных областях. В данном направлении А.Д. Закревский вел работу в сотрудничестве с Ю.Н. Печерским – ученым Академии наук Молдавской ССР.

Параллельные алгоритмы логического управления. Наряду с усложнением элементной базы – появлением больших и сверхбольших интегральных схем – возросла также сложность тех технических объектов, для которых потребовалась разработка систем логического управления: станков, поточных линий, робототехнических комплексов. Автоматизация проектирования дискретных устройств и систем связана с соответствующими языковыми средствами, используемыми на различных этапах проектирования для описания основных объектов. В новых обстоятельствах потребовались соответствующие языковые средства для описания процессов логического управления, характеризуемых параллельностью и асинхронностью. Ответом на запросы практики стала монография «Параллельные алгоритмы логического управления» [24] (1999), где предложена модель алгоритма дискретного управления параллельными процессами – А-сеть. На ее основе были разработаны языки АЛУ и ПРАЛУ, позволяющие иерархически описывать широкий класс параллельных алгоритмов логического управления и объединяющие в себе сильные стороны языков, применяемых в конструкторской практике, и сетей Петри. Разработана алгоритмическая база для решения задач конструирования устройств управления, основанная на оригинальных моделях и декомпозиции проблемы проектирования на ряд формализованных задач, для которых предложены конкурирующие методы решения.

Язык ПРАЛУ, представленный в [24], опирается на мощный формализм сетей Петри. Особенностями языка ПРАЛУ являются логическая стройность, простота, компактность получаемых описаний, использование двоичных (булевых) переменных для задания входов и выходов устройства управления, алгоритм функционирования которого задан на языке ПРАЛУ.

Определено понятие корректности параллельного алгоритма управления на языке ПРАЛУ; показано, что ее проверка в значительной степени сводится к проверке живости и безопасности введенной формальной модели – α-сети; разработан ряд эффективных методов проверки этих свойств. Впервые предложены быстрые алгоритмы диагностики и проверки корректности алгоритмов управления ряда важных типов, а также методы моделирования сетями Петри алгоритмов логического управления, анализа ординарных и операционных сетей Петри.

Разработаны методы равносильных преобразований, композиции и декомпозиции алгоритма на языке ПРАЛУ, позволяющие упростить их или получить описание заданного типа, полезного при проектировании специализированных систем управления, в частности мультипроцессорных систем; предложены методы синтеза логических схем, реализующих ПРАЛУ-описания, на ПЛМ с памятью в виде регистра RS-триггеров. Промежуточными моделями при схемной реализации ПРАЛУ-описаний являлись автоматные модели параллельного алгоритма управления – параллельный и секвенциальный автоматы.

Формализованы отношения эквивалентности и реализации конечных автоматов, выделены реализации доопределением, кодированием и расширением. Разработан комплекс точных и приближенных, матричных и графовых методов минимизации и декомпозиции полных и частичных автоматов, кодирования внутренних состояний синхронного и асинхронного автоматов, получения булева автомата. Предложены секвенциальная модель алгоритма управления и формы задания секвенциальных автоматов, полезные при решении задач анализа и синтеза, отношения эквивалентности и реализации между ними. Введено понятие инерционного и частичного секвенциальных автоматов, разработаны методы их анализа и реализации на матричных СБИС.

Основной оптимизационной задачей при схемной реализации параллельного автомата и переходе к секвециальному автомату стала задача кодирования состояний параллельного ав-

томата, а критерием оптимизации явилось число кодирующих переменных и простота реализации получаемого секвециального автомата. Введено понятие интервального вытесняющего кода частичных состояний, сформулированы требования, обеспечивающие совместимость кодов параллельных состояний, а также разработан комплекс матричных, итерационных, декомпозиционных и блочных методов кодирования состояний параллельных синхронных автоматов.

Язык ПРАЛУ является входным языком отечественных систем автоматизированного проектирования (САПР) цифровых устройств, разработанных в лаборатории логического проектирования. Важно отметить особенности схемной реализации параллельных ПРАЛУ-алгоритмов. Во-первых, возможна как синхронная, так и асинхронная реализация; во-вторых, при схемной реализации ПРАЛУ-описаний параллельного алгоритма осуществляется глобальное кодирование состояний соответствующего параллельного автомата. Именно это и отличает данный подход от развиваемых в зарубежных САПР подходов, где синтез осуществляется заменой локальных алгоритмических конструкций языков VHDL, Verilog соответствующими логическими выражениями либо установкой в получаемую схему элементов памяти. Кроме того, для ПРАЛУ можно проверять семантические свойства (безызбыточность, восстанавливаемость, непротиворечивость, устойчивость, самосогласованность), в то время как при моделировании VHDL-описаний проверяется лишь синтаксическая корректность. Следует заметить, что это возможно из-за ограниченности множества операций: допускаются лишь операции ожидания и установки значений (0,1) булевых переменных.

Полиномиальная реализация булевых функций. Монография «Полиномиальная реализация частичных булевых функций и систем» [25] (2001) заполнила теоретический пробел в области оптимизации полиномиальных представлений булевых функций – полиномов Жегалкина и Рида – Маллера для частичных функций и систем. Предложенные алгоритмы поиска минимальной полиномиальной реализации системы частичных булевых функций базировались на теории линейных векторных пространств. Методы обобщены также на случай k-значной логики.

Задачи экономного представления булевых функций полиномами Жегалкина, Рида – Маллера фиксированной полярности или общего типа (при оптимальном синтезе AND/EXOR-схем) сведены к составлению и решению матричных уравнений с использованием методов теории линейных векторных пространств. Разработан комплекс эффективных методов на основе векторных и матричных представлений, включающий методы реализации частичных булевых функций полиномами Жегалкина, в частности метод реализации булевых функций на основе лестничного алгоритма минимизации полинома и быстрый приближенный метод синтеза реализующего полинома Жегалкина. Эти методы обобщены для систем частичных булевых функций: предложен метод минимизации таких систем в классе полиномов Жегалкина и метод реализации систем многозначных частичных функций поляризованными полиномами Рида – Маллера. На основе предложенных методов оптимизации полиномиальных представлений булевых функций разработаны подходы к диагностированию константных неисправностей в EXOR-схемах.

Логико-комбинаторный подход к решению задач проектирования. А.Д. Закревским был предложен общий методологический подход к решению задач алгоритмического, логического и топологического проектирования СБИС [26–28], в рамках которого проблема проектирования рассматривалась как комплексная. Она требовала сначала теоретического решения на основе выяснения логико-комбинаторной природы решаемых задач в их точной постановке и только после этого – проведения разработки эффективных точных и приближенных алгоритмов их решения, программной реализации, экспериментального исследования программ и выбора лучших для включения в создаваемые системы проектирования. При этом критерии оптимизации от этапа к этапу менялись и отражали прямо либо косвенно основные критерии оптимизации логических схем: сложность (площадь кристалла), быстродействие, тестопригодность и энергопотребление.

Был определен набор комбинаторных задач дискретной математики, к решению которых сводилось подавляющее большинство задач проектирования и которые были представимы в терминах логических векторов и матриц. Предложена техника вычислений в булевом пространстве, базирующаяся на сокращенном обходе дерева поиска, декомпозиции и редуцирова-

нии возникающих ситуаций с максимальным усечением неперспективных ветвей дерева поиска. Разработан комбинаторный базис логического проектирования – комплекс эффективных методов и программ решения комбинаторных задач над логическими матрицами, имеющих множество полезных практических интерпретаций. Это были задачи о кратчайшем покрытии, минимальном дизъюнктивном коде и базисе, минимальном диагностическом тесте, минимальной и плотной упаковках, а также задачи над графами: поиск клик и кратчайших путей в графе, проверка графа на связность, раскраска вершин графа, кратчайшее покрытие графа полными двудольными подграфами и др.

Сотрудниками лаборатории логического проектирования также уделялось большое внимание комбинаторному обеспечению проектирования: развивались методы решения логических уравнений, комбинаторных задач теории графов, задач над булевыми и троичными матрицами, методы оптимизации различных форм представлений булевых функций и систем булевых функций.

Основы логического проектирования дискретных устройств. Монография «Логические основы проектирования дискретных устройств» была издана в трех книгах в ОИПИ НАН Беларуси и переиздана в России одной книгой [29]. В первой книге [30] представлены базовые понятия по теории множеств и теории графов, логике высказываний и предикатов, булевой алгебре, а также комбинаторные задачи и методы комбинаторного поиска, булевы функции и формы их представления. Вторая книга [31] посвящена методам оптимизации представлений полностью и частично определенных булевых функций и систем таких функций, ДНФ и полиномиальных представлений Жегалкина и Рида – Маллера, в ней приведены методы декомпозиции (функционального разложения) полностью и не полностью определенных булевых функций. В третьей книге [32] рассмотрены задачи проектирования систем логического управления: синтеза комбинационных схем и схем с памятью на различной элементной базе (библиотечных логических элементах, транзисторных матрицах и ПЛМ). В качестве исходных описаний (заданий на проектирование) могли выступать конечные автоматы, граф-схемы алгоритмов, секвенциальные автоматы, параллельные автоматы, а также параллельные алгоритмы управления на языке ПРАЛУ. Трилогия была переведена на английский язык и издана в Эстонии при содействии проректора Таллиннского технического университета А. Кееваллика. Все монографии подвели своеобразный итог полученным в лаборатории результатам по схемной реализации устройств логического управления и часто до сих пор рекомендуются в качестве учебного пособия для теоретической подготовки студентов вузов в области логического проектирования.

Важной чертой научного творчества А.Д. Закревского явилось то, что он достаточно часто возвращался к уже решенным задачам и находил новые, более эффективные методы их решения. Прежде всего это касается классических задач минимизации и функционального разложения булевых функций. Последняя задача часто называется также задачей простой декомпозиции булевой функции, когда требуется найти двухблочное разбиение (либо покрытие) множества аргументов булевой функции на два подмножества, по которым проводится разложение. Первая статья по данной проблеме была написана в 1964 г., а затем в 2007 г. Аркадием Дмитриевичем были предложены новые алгоритмы ее решения.

Вычисления в многомерном булевом пространстве. Прогресс в области интегральных технологий привел к удешевлению элементов памяти, что позволило использовать в вычислительных системах память большого объема. Основными проблемами в микроэлектронике стали быстродействие схем и их энергопотребление. Отражением возможностей использования памяти большого объема при решении оптимизационных логико-комбинаторных задач явилась монография «Вычисления в многомерном булевом пространстве» [33] (2011). Большой заслугой А.Д. Закревского стало кардинальное изменение развиваемого им на протяжении всей своей научной деятельности подхода к решению логико-комбинаторных задач, который существенным образом ориентировался на ограниченный объем памяти компьютеров. Обычно память была небольшого объема, это приходилось учитывать при создании эффективных алгоритмов и программ. Однако многие комбинаторные задачи упростились, когда появилась «большая» память, а вместе с этим и возможность оперирования векторами большой размерности, в качестве которых выступали столбцы значений функций на наборах значений аргументов из таблиц

истинности булевых функций и их систем. Аркадий Дмитриевич ввел операции над булевыми функциями в таком представлении и переформулировал на этот язык многие оптимизационные задачи: минимизацию булевых функций, нахождение простого функционального разложения и др. Это позволило увеличить скорость решения задач практической размерности, когда число аргументов булевых функций достигало трех десятков. Были проведены соответствующие вычислительные эксперименты.

Научная и организационная деятельность А.Д. Закревского

Особенностью научной деятельности А.Д. Закревского явилось то, что, получив результаты в каком-то направлении, он публиковал сначала научные статьи и выступал с докладами на конференциях, а затем готовил монографии. В перечисленных монографиях представлены далеко не все результаты, которые были опубликованы в научных статьях. Хотелось бы привести некоторые из них, которые по своей значимости не уступают монографиям. Например, в статьях [34, 35] сформулированы принципы создания диалоговых систем проектирования, в [26, 27] поставлены важные теоретические задачи, возникающие в области логического проектирования и искусственного интеллекта. В последние годы своей жизни А.Д. Закревский занимался проблемами сокращения энергопотребления схем с памятью, которые реализуют конечные автоматы [36, 37].

С первых дней работы в Институте технической кибернетики АН БССР А.Д. Закревский много внимания уделял организации научных исследований. Каждый научный сотрудник его лаборатории проводил исследования в своей области системного программирования или логического проектирования, поэтому большое значение придавалось обсуждению получаемых результатов и расширению научного кругозора. В январе 1972 г. Аркадий Дмитриевич организовал городской научный семинар «Логическое проектирование», которым бессменно руководил до 2014 г. Им было принято, что любая научная работа, которую планировалось заслушать на этом семинаре, проходила сначала этап рецензирования и, несмотря на свою занятость, Аркадий Дмитриевич, как правило, знакомился с каждой из них. Рецензирование рукописи, выступление автора и обсуждение материала на этом семинаре часто было более серьезным процессом, чем рассмотрение соответствующей статьи в редакции журнала, куда ее планировалось направить. При оценке доклада решающее мнение было за руководителем семинара, и оно никогда не подвергалось сомнению. Внимательно читая, а по сути, рецензируя подготовленные научные работы, А.Д. Закревский был прекрасно осведомлен о том, чем занимается каждый его сотрудник или аспирант, насколько он продвинулся в решении поставленной задачи.

Начиная с 1975 г. лаборатория издавала ежегодный сборник научных трудов «Алгоритмы решения логико-комбинаторных задач». Под таким названием было выпущено шесть изданий. Сначала они нумеровались, затем в 1980 г. сборник был выпущен без номера, а после стали выходить выпуски того же сборника статей, но под разными названиями. С появлением персональных компьютеров с 1990 г. стал выходить ежегодник «Логическое проектирование». Всего под редакцией А.Д. Закревского было выпущено семь ежегодных сборников под одним названием. Сборники были гордостью Аркадия Дмитриевича, в них аккумулировались алгоритмы и даже программы, а включение статьи в сборник было возможным только после получения рекомендации семинара. В 2004 г. в ОИПИ НАН Беларуси начал издаваться периодический журнал «Информатика», где имеется раздел «Логическое проектирование».

После распада Советского Союза (1991) перестали проводиться практически все периодические научные конференции. В 1995 г. силами сотрудников лаборатории была организована и проведена Международная конференция «Автоматизация проектирования дискретных устройств» (International Conference on Computer-Aided Design of Discrete Devices – CAD DD). Всего за период 1995 – 2010 гг. было проведено семь конференций с таким названием. Аркадий Дмитриевич был бессменным председателем программных комитетов конференции, оценивал каждый из докладов лично, рекомендовал рецензентов, участвовал в распределении докладов по секциям и составлении планов работы секций.

В биобиблиографии [38], изданной к 80-летию А.Д. Закревского, содержится хронологический перечень научных трудов за период с 1956 по 2007 г. Всего в ней перечислено 565 на-

именований, в том числе 54 сборника научных трудов и других изданий, в которых он был редактором либо соредактором. В книге [39] имеется глава, посвященная научной, научно-организационной, педагогической и общественной деятельности А.Д. Закревского, где указывается, что он является автором более 540 научных работ, научным руководителем 34 кандидатов и 8 докторов наук. Около 30 заметок и статей было издано о его жизни и научной деятельности. Аркадий Дмитриевич всегда был твердым и бескомпромиссным искателем научной истины и настоящим лидером созданной им научной школы.

Помимо научных работ Аркадий Дмитриевич подготовил две книги воспоминаний: «Поет морзянка» о своей работе радистом в экспедициях по изысканию проведения железнодорожных путей и «В Томском университете» о своей студенческой жизни и научной работе в Томском государственном университете.

Фундаментальность научных идей А.Д. Закревского, создание им научной школы, развитие его идей в области логического проектирования учениками позволили коллективу лаборатории логического проектирования получить впечатляющие научные результаты мирового уровня. Достаточно сказать, что за 45 лет существования лаборатории логического проектирования заведующим и сотрудниками лаборатории было опубликовано 53 книги (44 монографии и 9 учебных пособий для студентов вузов) и более 1000 научных статей.

Список литературы

1. Закревский, А.Д. Формализация синтеза электронной цифровой вычислительной машины / А.Д. Закревский // Труды СФТИ. – Вып. 40 : Вычислительная техника, автоматика, теория информации. – Томск : Изд-во Томск. ун-та, 1961. – С. 64–72.

2. Закревский, А.Д. Метод синтеза диодных логических схем / А.Д. Закревский // Там же. – С. 73–88.

3. Закревский, А.Д. Визуально-матричный метод минимизации булевых функций / А.Д. Закревский // Автоматика и телемеханика. – 1960. – № 3. – С. 369–373.

4. Закревский, А.Д. Вычисления в булевых пространствах / А.Д. Закревский // Логическая структура научного знания. – М. : Наука, 1965. – С. 292–310.

5. Закревский, А.Д. Алгоритмы минимизации слабо определенных булевых функций / А.Д. Закревский // Кибернетика. – 1965. – № 2. – С. 53–60.

6. Закревский, А.Д. К синтезу последовательностных автоматов / А.Д. Закревский // Труды СФТИ. – Вып. 40 : Вычислительная техника, автоматика, теория информации. – Томск : Изд-во Томск. ун-та, 1961. – С. 89–94.

7. Закревский, А.Д. Операторный метод синтеза алгоритмических систем / А.Д. Закревский // Изв. ВУЗов. Радиофизика. – 1959. – № 2. – С. 306–315.

8. Закревский, А.Д. О сокращении переборов при решении некоторых задач синтеза дискретных автоматов / А.Д. Закревский // Изв. ВУЗов. Радиофизика. – 1964. – № 1. – С. 166–174.

9. Закревский, А.Д. Покрытия множеств / А.Д. Закревский // Труды СФТИ. – Вып. 48 : Автоматизация синтеза дискретных автоматов. – Томск : Изд-во Томск. ун-та, 1966. – С. 69–72.

10. Закревский, А.Д. Оптимизация покрытий множеств / А.Д. Закревский // Логический язык для представления алгоритмов синтеза релейных устройств. – М. : Наука, 1966. – С. 136–147.

11. Закревский, А.Д. Функциональная устойчивость релейных схем / А.Д. Закревский // Труды СФТИ. – Вып. 40 : Вычислительная техника, автоматика, теория информации. – Томск : Изд-во Томск. ун-та, 1961. – С. 112–126.

12. Закревский, А.Д. Метод автоматической шифрации сообщений / А.Д. Закревский // Прикладная дискретная математика. – 2009. – № 2(4). – С. 127–137.

13. Закревский, А.Д. Машина для решения логических задач типа синтеза релейных схем / А.Д. Закревский // Синтез релейных структур : тр. Междунар. симп. по теории релейных устройств и конечных автоматов (ИФАК). – М. : Наука, 1965. – С. 346–356.

14. Закревский, А.Д. Алгоритмический язык ЛЯПАС и автоматизация синтеза дискретных автоматов / А.Д. Закревский; МВ и ССО РСФСР, СФТИ им. В.А. Кузнецова при Томском гос. ун-те ; [науч. ред. В.П. Тарасенко]. – Томск : Изд-во Томск. ун-та, 1966. – 266 с.

15. Логический язык для представления алгоритмов синтеза релейных устройств / Под ред. М.А. Гаврилова. – М. : Наука, 1966. – 342 с.

16. LYaPAS, A Programming Language for Logic and Coding Algorithms / Ed. by M. Gavrilov and A. Zakrevskij. – N. Y., London : ACM Monograph Series, 1969.

17. Закревский, А.Д. Алгоритмы синтеза дискретных автоматов / А.Д. Закревский. – М. : Наука, 1971. – 512 с.

18. Синтез асинхронных автоматов на ЭВМ / [авт.-сост.: А.Д. Закревский, Л.И. Балаклей, Н.А. Елисеева и др.] ; под общ. ред. А.Д. Закревского. – Минск : Наука и техника, 1975. – 184 с.

19. Закревский, А.Д. Логические уравнения / А.Д. Закревский. – Минск : Наука и техника, 1975. – 96 с. ; изд. 2-е. – М. : УРСС, 2003. – 96 с.

20. Закревский А.Д. Решение больших систем логических уравнений / А.Д. Закревский. – Минск : ОИПИ НАН Беларуси, 2009. – 96 с. ; Solving Large Systems Logical Equations / A.D. Zakrevskij. – Tallinn : TUT Press, 2013. – 114 p.

21. Закревский, А.Д. Система программирования ЛЯПАС-М / А.Д. Закревский, Н.Р. Торопов ; АН БССР, Ин-т техн. кибернетики. – Минск : Наука и техника, 1978. – 240 с.

22. Закревский, А.Д. Логический синтез каскадных схем / А.Д. Закревский. – М. : Наука, 1981. – 416 с.

23. Закревский, А.Д. Логика распознавания / А.Д. Закревский ; АН БССР, Ин-т техн. кибернетики. – Минск : Наука и техника, 1988. – 118 с. ; изд. 2-е, доп. – М. : УРСС, 2003. – 140 с.

24. Закревский, А.Д. Параллельные алгоритмы логического управления / А.Д. Закревский. – Минск : Ин-т техн. кибернетики НАН Беларуси, 1999. – 202 с. ; изд. 2-е. – М. : УРСС, 2003. – 200 с.

25. Закревский, А.Д. Полиномиальная реализация частичных булевых функций и систем / А.Д. Закревский, Н.Р. Торопов. – Минск : Ин-т техн. кибернетики НАН Беларуси, 2001. – 200 с ; изд. 2-е. – М. : УРСС, 2003. – 200 с.

26. Закревский, А.Д. Комбинаторика логического проектирования / А.Д. Закревский // Автоматика и вычислительная техника. – 1990. – № 2. – С. 68–79.

27. Закревский, А.Д. Комбинаторные задачи над логическими матрицами в логическом проектировании и искусственном интеллекте / А.Д. Закревский // Зарубежная радиоэлектроника: успехи современной радиоэлектроники. – 1998. – № 2. – С. 59–67.

28. Boolesche Gleichungen. Theorie, Anwendung, Algorithmen / ed. mit D. Bochmann, A. Zakrevskij und Ch. Posthoff. – Berlin : VEB Verlag Technik, 1984.

29. Закревский, А.Д. Логические основы проектирования дискретных устройств / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – М. : Физматлит, 2007. – 590 с.

30. Закревский, А.Д. Основы логического проектирования: в 3 кн. / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – Минск : ОИПИ НАН Беларуси, 2004. – Кн. 1: Комбинаторные алгоритмы дискретной математики. – 225 с. ; Combinatorial algorithms of discrete mathematics. – Tallinn : TUT Press, 2008. – 192 p.

31. Закревский, А.Д. Основы логического проектирования : в 3 кн. / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – Минск : ОИПИ НАН Беларуси, 2004. – Кн. 2 : Оптимизация в булевом пространстве. – 240 с. ; Optimization in Boolean space. – Tallinn : TUT Press, 2009. – 241 p.

32. Закревский, А.Д. Основы логического проектирования : в 3 кн. / А.Д. Закревский, Ю.В. Поттосин, Л.Д. Черемисинова. – Минск : ОИПИ НАН Беларуси, 2006. – Кн. 3 : Проектирование устройств логического управления. – 254 с. ; Design of Logical Control Devices. – Tallinn : TUT Press, 2009. – 304 p.

33. Закревский А.Д. Вычисления в многомерном булевом пространстве / А.Д. Закревский. – Минск : ОИПИ НАН Беларуси, 2011. – 106 с. ; Combinatorial Calculations in Many-Dimensional Boolean Space / A.D. Zakrevskij ; ed. A. Keevallik. – Tallinn : TUT Press, 2012. – 121 p.

34. Закревский, А.Д. Автоматизация логического синтеза дискретных устройств / А.Д. Закревский // Кибернетика. – 1975. – № 4. – С. 100–108.

35. Закревский, А.Д. Принципы построения диалоговых систем проектирования и их реализация в системе логического синтеза / А.Д. Закревский // Оптимизация систем сбора, передачи

и обработки информации в человекомашинных системах. – Минск : Ин-т техн. кибернетики АН Беларуси, 1980. – С. 3–6.

36. Закревский, А.Д. Энергосберегающее кодирование состояний конечного автомата. Метод квадратов / А.Д. Закревский // Информатика. – 2005. – № 4. – С. 105–113.

37. Закревский, А.Д. Нахождение режима максимального энергопотребления логической схемы / А.Д. Закревский // Прикладная дискретная математика. – 2012. – № 2(16). – С. 100–104.

38. Биобиблиография ученых Беларуси. Член-корреспондент Аркадий Дмитриевич Закревский (к 80-летию со дня рождения) / НАН Беларуси, Объед. ин-т проблем информатики, ЦНБ им. Я. Коласа. – Минск : ОИПИ НАН Беларуси, 2008. – 91 с.

39. Кибернетика и информатика в Национальной академии наук Беларуси: очерки развития / Объед. ин-т проблем информатики НАН Беларуси ; науч. ред.: С.В. Абламейко, А.В. Тузиков, О.И. Семенков. – Минск : Тэхналогія, 2015. – 348 с.

Поступила 28.11.2016

Объединенный институт проблем
информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: bibilo@newman.bas-net.by

P.N. Bibilo, Yu.V. Pottosin, L.D. Cheremisinova

ON SCIENCE HERITAGE OF CORRESPONDING MEMBER A.D. ZAKREVSKIJ

Arkadij Dmitrievich Zakrevskij was at the beginnings of cybernetics origin in the Soviet Union. He is the founder of one of the known schools of logical design in the Soviet Union and all the world. This paper is devoted to the science heritage of A.D. Zakrevskij.

ПРАВИЛА ДЛЯ АВТОРОВ

1. Статьи принимаются в редакцию через электронную систему подачи по адресу **http://jinfo.bas-net.by** в формате файлов текстовых редакторов Microsoft Word 97 и Word 2000 для Windows. Основной текст статьи набирается с переносами шрифтом Times New Roman 11 пт, интервал между строками – одинарный, абзацный отступ 1 см, поля по 2,5 см со всех сторон.

2. Статья должна иметь индекс УДК (универсальная десятичная классификация).

3. Название статьи, фамилии всех авторов и аннотация должны быть переведены на английский язык. Для каждого из авторов приводится развернутое название учреждения с полным почтовым адресом, а также номер телефона и электронный адрес (e-mail) для связи с редакцией.

4. Формулы, иллюстрации, таблицы, встречающиеся в статье, должны быть пронумерованы в соответствии с порядком цитирования в тексте. Ссылки на рисунки и таблицы в тексте обязательны. Необходимо избегать повторения одних и тех же данных в таблицах, графиках и тексте статьи.

Рисунки должны быть выполнены с хорошим разрешением в масштабе, позволяющем четко различать надписи и обозначения. Подрисуночные подписи с расшифровкой всех позиций, представленных на рисунке, набираются шрифтом гарнитуры основного текста, размер символов 9 пт. Цветные иллюстрации печатаются только в том случае, когда это необходимо для понимания излагаемого материала.

5. Набор формул выполняется в формульных редакторах Microsoft Equation или Math Type и должен быть единообразным по применению шрифтов и знаков по всей статье.

Прямо (_) набираются: греческие и русские буквы; математические символы (sin, lg, ∞); символы химических элементов (C, Cl, $CHCl_3$); цифры (римские и арабские); векторы; индексы (верхние и нижние), являющиеся сокращениями слов.

Курсивом (~) набираются: латинские буквы – переменные, символы физических величин (в том числе и в индексе).

6. Сокращения в тексте статьи (за исключением единиц измерения) могут быть использованы только после упоминания полного термина. Единицы измерения физических величин следует приводить в Международной системе СИ.

7. Литература приводится автором общим списком в конце статьи. Ссылки на литературу в тексте идут по порядку и обозначаются цифрой в квадратных скобках. Ссылаться на неопубликованные работы не допускается. С примерами оформления библиографического описания в списке литературы можно ознакомиться в приложении 2 к *Инструкции по оформлению диссертации, автореферата и публикаций по теме диссертации* на сайте Высшей аттестационной комиссии Республики Беларусь **http://vak.org.by**.

8. Поступившие в редакцию статьи направляются на рецензирование специалистам. Основным критерием целесообразности публикации является новизна и информативность статьи. Если по рекомендациям рецензента статья возвращается автору на доработку, а переработанная рукопись вновь рассматривается редколлегией, датой поступления считается день получения редакцией ее окончательного варианта. Статьи не по профилю журнала возвращаются авторам после заключения редколлегии.

9. Статьи, направляемые на доработку, должны быть возвращены в исправленном виде с ответами на все вопросы.

10. Редакция журнала предоставляет возможность первоочередного опубликования статей, представленных лицами, которые осуществляют послевузовское обучение (аспирантура, докторантура, соискательство) в год завершения обучения.

11. Авторы несут ответственность за направление в редакцию статей, уже опубликованных ранее, или статей, принятых к публикации другими изданиями.

12. Редакция оставляет за собой право на редакционные изменения, не искажающие основное содержание статьи.

Журнал «Информатика» включен Высшей аттестационной комиссией Республики Беларусь в список научных изданий для опубликования результатов диссертационных исследований.

Индексы

00827

для индивидуальных подписчиков

008272

для предприятий и организаций

ИНФОРМАТИКА 1 (53) январь–март 2017