18. *Tomashevskaya V.S., Yakovlev D.A.* Uslovie additivnosti informatsionno-spravochnogo kioska na osnove vremeni zagruzki stranits [The condition of the additivity of the information and reference kiosk based on the page load time], *Rossiyskiy tekhnologicheskiy zhurnal* [Russian Technological Journal], 2020, Vol. 8, No. 1 (33), pp. 27-33.
19. *Shemonchuk D.S.* Polnota i tochnost' mul'timedia kontenta dlya sushchestvenno intensifitsirovannykh mul'timedia sistem [Completeness and accuracy of multimedia content for significantly intensified multimedia systems], *Informatsionnye sistemy i tekhnologii* [Information systems and technologies], 2009, No. 3\53 (564), pp. 35-42.
20. *Shemonchuk D.S.* Razrabotka i issledovanie metodov i sredstv uluchsheniya funktsionala mul'timediynykh portal'nykh ustroystv v sfere upravleniya obrazovatel'nymi protsessami: diss. … kand. tekhn. nauk [Development and research of methods and means for improving the functionality of multimedia portal devices in the field of educational process management: cand. of eng. sc. diss.]. Moscow, 2009, 155 p.

Статью рекомендовал к опубликованию к.т.н. В.А. Холопов.

Жуков Николай Кириллович – Институт космических исследований РАН (ИКИ РАН), Российская академия наук; e-mail: nzhukov@romance.iki.rssi.ru, Москва, Россия; тел.: 89851378683.

Мордвинов Владимир Александрович – МИРЭА – Российский Технологический университет; e-mail: mordvinov@mirea.ru; Москва, Россия; кафедра инструментального и прикладного программного обеспечения; к.т.н.; профессор.

Русляков Алексей Александрович – e-mail: ruslyakov@mirea.ru; кафедра инструментального и прикладного программного обеспечения; ассистент.

Zhukov Nikolai Kirillovich – Space Research Institute RAS (IKI RAS), Russian Academy of Sciences; e-mail: nzhukov@romance.iki.rssi.ru; Moscow, Russia; phone: +79851378683.

Mordvinov Vladimir Alexandrovich – MIREA – Russian Technological University; e-mail: mordvinov@mirea.ru; Moscow Russia; the department of instrumental and applied software; cand. of eng. sc.; professor.

Ruslyakov Alexey Alexandrovich – e-mail: ruslyakov@mirea.ru; the department of instrumental and applied software; assistant.

УДК 004.896 DOI 10.18522/2311-3103-2021-5-176-192

О.Б. Лебедев, А.А. Жиглатый, Е.О. Лебедева

РАЗРАБОТКА МОДИФИЦИРАВАННЫХ МЕТОДОВ И МОДЕЛЕЙ ПОИСКОВОЙ АДАПТАЦИИ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ПЛАНИРОВАНИЯ СБИС*

В работе для решения задачи планирования СБИС разработан поисковый алгоритм на основе модифицированного метода муравьиной колонии. Задача формирования плана СБИС сводится к задаче формирования соответствующего польского выражения. Разработанный метод синтеза польского выражения включает построение дерева разрезов, выбор типов разрезов (H или V), идентификацию и ориентацию модулей. Эволюционирующая популяция разбита на пары агентов. Каждый член популяции – пара агентов, работающих совместно. При этом конструктивные алгоритмы A1 и A2, используемые агентами пары различаются. Задача, решаемая алгоритмом A1, формулируется как задача поиска взаимно однозначного отображения $F_k=M^*\rightarrow P$ множества модулей M с выбранными ориентациями, $|M^*|=|M|$ в множество P позиций шаблона Sh. Фактически решение заключается в выборе на графе G1 подмножества ребер $E^*1\in E1$, входящих в соответст-

* Работа выполнена при финансовой поддержке гранта РФФИ № 20–07–00260 А.

*вующее отображение F_k. В алгоритме А2 в качестве модели пространства поиска решений для выбора типа, последовательности и места расположения разрезов в шаблоне Sh разработан граф G2=(X, E2). $X=\{(x1_i,x2_i)|i=1,2,\ldots,n\}$ множество вершин графа G2, соответствует множеству P потенциальных позиций шаблона Sh для возможного размещения в них имен символов разрезов. Каждая потенциальная позиция $p_i \in P$ шаблона Sh моделируется двумя альтернативными вершинами $(x1_i,x2_i)$. Выбор при размещении разрезов вершины $x1_i$ указывает на то, что в позицию p_i помещен разрез типа V, выбор вершины $x2_i$ – указывает на то, что в позицию p_i помещен разрез типа H. Каждая итерация l общего алгоритма включает начальный и три основных этапа. Начальный этап заключается в следующем. Обнуляются матрицы ко-эволюционной памяти КЭП*1 и КЭП*2. На первом этапе каждая пара агентов $d_k=(a1_k, a2_k)$: – конструктивными алгоритмами А1 и А2 синтезирует свое решение $W_k=(E1_k^*,S_k)$; – формируется польское выражение Sh_k, соответствующее решению W_k; – на базе Sh_k формируется дерево разрезов T_k; – на базе T_k формируется план R_k и рассчитывается оценка решения F_k; – агенты откладывают (добавляют) феромон в ячейки матриц коллективной эволюционной памяти КЭП*1 и КЭП*2, соответствующие ребрам решения $W_k=(E1_k^*,S_k)$ в графах поиска решений G1 и G2 в количестве пропорциональном оценке решения F_k. На втором этапе феромон, накопленный в КЭП*1 и КЭП*2 агентами популяции на итерации l, добавляется в КЭП1 и КЭП2. На третьем этапе осуществляется испарение феромона на ребрах графов G1 и G2. Тестовые испытания подтвердили эффективность предложенного метода. Временная сложность алгоритма, полученная экспериментальным путем, совпадает с теоретическими исследованиями и для рассмотренных тестовых задач составляет $O(n^2)$.*

Планирование СБИС; роевой интеллект; оптимизация; муравьиный алгоритм; адаптивное поведение.

O.B. Lebedev, A.A. Zhiglatiy, E.O. Lebedeva

DEVELOPMENT OF MODIFIED METHODS AND MODELS OF SEARCH ADAPTATION FOR SOLVING THE PROBLEM OF PLANNING VLSI

*In this work, to solve the VLSI planning problem, a search algorithm has been developed based on a modified ant colony method. The task of forming a VLSI plan is reduced to the task of forming the corresponding Polish expression. The developed method for the synthesis of the Polish expression includes the construction of a tree of cuts, the choice of the types of cuts (H or V), identification and orientation of modules. The evolving population is split into pairs of agents. Each member of the population is a pair of agents working together. In this case, the constructive algorithms A1 and A2 used by the agents of the pair are different. The problem solved by Algorithm A1 is formulated as the problem of finding a one-to-one mapping $Fk=M^*\rightarrow P$ of the set of modules M with selected orientations, $|M^*|=|M|$ to the set P of positions of the template Sh. In fact, the solution consists in choosing on the graph G1 a subset of edges $E^*1 \in E1$ included in the corresponding mapping F_k. In Algorithm A2, the graph G2=(X, E2) is developed as a model of the search space for solutions for choosing the type, sequence and location of cuts in the pattern Sh. $X=\{(x1_i,x2_i)|i=1,2,\ldots,n\}$ the set of vertices of the graph G2, corresponds to the set P of potential positions of the template Sh for the possible placement of the names of the cut symbols in them. Each potential position $p_i \in P$ of the template Sh is modeled by two alternative vertices $(x1_i,x2_i)$. The choice of the vertex $x1_i$ when placing the cuts indicates that a cut of type V is placed in position p_i, the choice of vertex $x2_i$ indicates that a cut of type H is placed in position p_i. Each iteration l of the general algorithm includes an initial and three main stages. The initial stage is as follows. Co-evolutionary memory matrices are nullified CEM*1 and CEM*2 are reset to zero. At the first stage, each pair of agents $d_k=(a1_k,a2_k)$: – with constructive algorithms A1 and A2 he synthesizes his solution $W_k=(E1_k^*,S_k)$; – the Polish expression Sh_k is formed, corresponding to the solution W_k; – on the basis of Sh_k, a tree of sections T_k is formed; – on the basis of T_k, the plan R_k is formed and the estimate of the solution F_k is calculated; – agents deposit (add) the pheromone to the cells of the collective evolutionary memory (CEM) matrices CEM^*1 and CEM^*2 corresponding to the solution edges $W_k=(E1_k^*,S_k)$ in the solution search graphs G1 and G2 in an amount proportional to the solution estimate F_k. At the second stage, the pheromone accumulated in CEM^*1 and CEM^*2 by agents of the population at iteration l is added to CEM 1 and CEM2. At the third stage,*

the pheromone is evaporated on the edges of the graphs G1 and G2. Tests have confirmed the effectiveness of the proposed method. The time complexity of the algorithm, obtained experimentally, coincides with theoretical studies and it is $O(n^2)$ for the considered test problems.

VLSI planning; swarm intelligence; optimization; ant algorithm; adaptive behavior.

Введение. В связи с уменьшением топологических размеров, повышением степени интеграции СБИС и сокращением сроков проектирования возникают принципиально новые требования к средствам проектированию СБИС [1, 2].

При проектировании больших систем часто топологическая схема требуется уже на ранних стадиях проектирования, хотя еще не были спроектированы все модули, т.е. не вся информация обо всех модулях имеется в наличии, часть этой информации может оказаться неточной. Планирование – это ранняя фаза проектирования СБИС. Оно дает информацию о приблизительных значениях площади, задержки, мощности и других рабочих характеристиках [1-4]. Задача планирования СБИС заключается в размещении на поле кристалла модулей, полученных на этапе разбиения, имеющих заданную площадь и не имеющих фиксированных размеров. Модули и кристалл имеют форму прямоугольников. При планировании решаются сразу две задачи: определяется взаимно расположение блоков друг относительно друга, т.е. их размещение, а также фиксируются размеры каждого блока. В результате планирования строится план кристалла, представляющий собой охватывающий прямоугольник, разделенный горизонтальными и вертикальными сегментами на неналагающиеся друг на друга прямоугольники, в которые следует поместить соответствующие модули [3, 4].

Основные проблемы задачи планирования кристалла СБИС – это проблема поиска подхода к представлению решения (плана) и проблема построения оптимизационной процедуры поиска решения.

Основные критерии оптимизации [1, 2, 5–7]: площадь кристалла; длина проводников; временные задержки; энергопотребление; температура. Основной целью оптимизации является минимизация общей площади кристалла [1, 2, 5].

Анализ существующих подходов к решению поставленной задачи показал, что удачными являются подходы, основанные на методах эволюционного моделирования. В последнее время для решения различных «сложных» задач, к которым относятся и задачи планирования всё чаще используются способы, основанные на применении биоинспирированных моделей [5,8,9]. В работе для решения задачи планирования разработан поисковый алгоритм на основе метода муравьиной колонии. Задача формирования плана СБИС сводится к задаче формирования соответствующего польского выражения. Разработанный метод синтеза польского выражения включает построение на основе шаблона дерева разрезов, выбор типов разрезов (*H* или *V*), идентификацию и ориентацию модулей.

1. Постановка задачи планирования. Проблема планирования формулируется следующим образом [1, 2]. Имеется множество модулей $M=\{m_i|i=1,2,\ldots,n\}$. Каждый модуль характеризуется тройкой $<S_i,l_i,t_i,>$, где S_i – площадь модуля, а параметры l_i и t_i задают нижнюю и верхнюю границу значения h_i/w_i, т.е.

$$l_i \le h_i/w_i \le t_i, \tag{1}$$

где h_i – это высота модуля, w_i – ширина модуля [1,2,5]. План для множества модулей M представляет собой прямоугольник R, разрезанный вертикальными и горизонтальными линиями на множество прямоугольников блоков $B=\{b_i|i=1,2,\ldots,n\}$, в каждом из которых помещается соответственно модуль m_i [8].

Каждый прямоугольник b_i, предназначенный для размещения модуля m_i, имеет размеры x_i и y_i. При соблюдении ограничений (1) размеры прямоугольников должны также соответствовать ограничениям:

$$S_i \le x_i \cdot y_i\,,\ h_i \le y_i\,,\ w_i \le x_i. \tag{2}$$

Будем считать, что связи между модулями m_i и m_j связывают центры соответствующих прямоугольников b_i и b_j [6–11]. Обозначим через d_{ij} длину связей между m_i и m_j а через c_{ij} – стоимость связей. Тогда критерий оптимизации при планировании имеет вид:

$$F = \sum_{i=1}^{n} x_i \cdot y_i + \lambda \sum_{i,j=1}^{n} c_{ij} \cdot d_{ij},$$

при соблюдении ограничений (1), (2). Константа λ, управляет относительной важностью общей площади и взвешенной длиной связей [3,6,8,11,12]. Основная цель оптимизации – минимизация общей площади плана S_r, при соблюдении ограничений (1), (2).

Принято разбивать все множество структур плана на два класса: гильотинный и не гильотинный. Гильотинная структура может быть получена путем рекурсивного деления прямоугольника на две части горизонтальными и/или вертикальными разрезами. В качестве плана кристалла в работе используется структура, полученная путем рекурсивного использования «гильотинного разреза», т.е. последовательного разрезания прямоугольников на две части [8]. Для описания плана используются два способа: геометрическое и символьное.

Геометрическим описанием является дерево разрезов. На рис.1.a представлен план, а на рис. 1,b – соответствующее ему бинарное дерево «гильотинного разреза» $DR=(D, E)$, $D=\{d_i|j=1,2,\ldots,2n-1\}$, $E \subseteq D \times D$, листьями которого являются вершины, соответствующие прямоугольникам с размещенными в них модулями, а внутренние вершины соответствуют разрезам: V – вертикальный, H – горизонтальный. На дереве висячие вершины (листья дерева), соответствующие прямоугольникам с размещенными в них модулями, помечены именами модулей, которые входят в эти прямоугольники, цифрами помечены внутренние вершины, соответствующие разрезам, причем V – вертикальный разрез, а H – горизонтальный разрез. На основе этой информации геометрическое представление и метризация плана осуществляется путем последовательной бинарной свертки прямоугольников (блоков b_i) по дереву разрезов, начиная от листьев дерева [11]. Процесс бинарной свертки представляет собой слияние смежных прямоугольников b_i и b_j, формирование прямоугольника b_k, определение его размеров и новых размеров для b_i и b_j.

Каждой внутренней вершине дерева разрезов будет соответствовать блок b_i (прямоугольник), полученный в результате бинарной свертки поддерева, имеющего корнем эту внутреннюю вершину. Размеры блоков и описывающего прямоугольника R плана определяются путем последовательной свертки блоков по дереву разрезов, исходя из размеров модулей, помещаемых в эти блоки [6, 8]. На рис. 1,c представлен план после свертки.

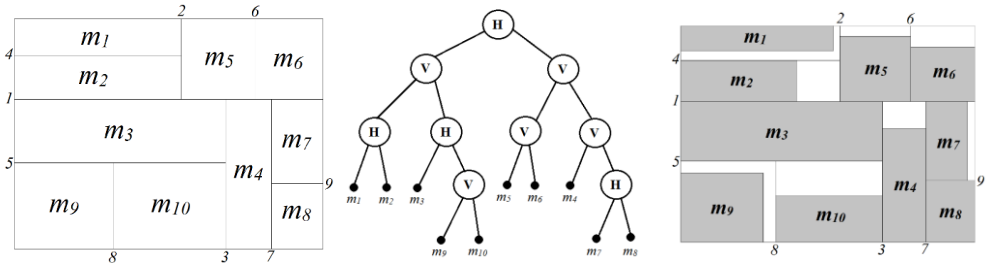

Рис. 1. a – план гильотинного разреза; b – дерево гильотинного разреза; c – план с размещенными в блоках модулями

Введём алфавит $A=\{M, TR\}$.Символьное представление структуры дерева разрезов можно задать, используя на базе алфавита A польское выражение для бинарного дерева [6, 8, 11], где множество модулей $M=\{m_i|i=1,2 \ldots,n_M\}$ соответствует листьям дерева разрезов (блокам b_i), а множество $TR=\{H, V\}$ – соответствует разрезам. Польское выражение для дерева с идентифицированными вершинами, представленного на рис.1.b, имеет вид: $D=<m_2\ \mathbf{V}\ m_3\ m_4\ \mathbf{V}\ \mathbf{H}\ m_5\ m_6\ \mathbf{H}\ m_7\ m_8\ m_9\ \mathbf{H}\ \mathbf{V}\ \mathbf{H}\ \mathbf{V}>$. Процесс восстановления дерева по польскому выражению достаточно прост [6]. Последовательно слева направо просматривается польское выражение, и отыскиваются буквы H или V, соответствующие разрезам. Каждый такой разрез объединяет два ближайших образованных на предыдущих шагах подграфа, расположенных в польской записи слева от буквы H или V. Проиллюстрируем процесс свертки с помощью скобок:

$$D=((m_1\ m_2\ V)\ (m_3\ m_4\ V)\ H)\ (m_5\ m_6\ H)\ (m_7\ (m_8\ m_9\ H)\ V)\ H)\ V.$$

Отметим основные свойства польского выражения, выполнение которых необходимо, чтобы записи соответствовало дерево разрезов. Обозначим через n_M – число элементов польского выражения, принадлежащих множеству M, а через n_R – число элементов, соответствующих разрезам.

Свойства (ограничения) польского выражения:

1. Для каждого модуля m_i возможны два способа (две ориентации (O_1 или O_2) размещения в прямоугольнике.

2. В состав выражения входят по одному разу все элементы множества $M=\{m_i|i=1, 2, \ldots, n_M\}$ с одной из меток – O_{1i} либо O_{2i}.

3. Для дерева разрезов всегда выполняется равенство $n_M=n_R + 1$.

4. Если в польском выражении провести справа от буквы H или V сечение, то слева от сечения число элементов, принадлежащих множеству M, больше числа элементов, соответствующих разрезам, минимум, на единицу.

Вышеперечисленные свойства фактически являются ограничениями, которым должна удовлетворять польское выражение.

Назовем польское выражение D, построенное на базе алфавита $A=\{M,TR\}$ легитимным, если оно удовлетворяет вышеперечисленным ограничениям 1-4. Таким образом, легитимное выражение D является символьным представлением решения задачи планирования.

2. Построение методом муравьиной колонии плана СБИС с размещением и выбором типов разрезов, размещением и ориентацией модулей. Задача формирования плана СБИС сводится к задаче формирования соответствующего польского выражения. Предварительно формируется шаблон позиций польского выражения в виде вектора $Sh=<\mathbf{o}\ \mathbf{o}\ z_1\ \mathbf{o}\ z_2\ \mathbf{o}\ z_3\ \mathbf{o}\ z_4\ \ldots\ \mathbf{o}\ z_n>$. Знаки типа (**o**) пронумерованы и соответствуют позициям для размещения в них имен модулей (листьев дерева). Между знаками (**o**) расположены пронумерованные зоны позиций $<z_1\ z_2\ z_3\ z_4\ \cdots\ z_n>$, для размещения в них имен внутренних вершин дерева (разрезов). Число потенциальных позиций n_z в зоне z_i равно номеру i зоны. Позиции p_i, предназначенные для размещения в них разрезов, объединены в общий список $P=<p_i|i=1,2,\ldots,n_p>$ и пронумерованы, в соответствии с их расположением в шаблоне Sh.

Формирование соответствующего польского выражения заключается в назначении n_M элементов, соответствующим модулям, в позиции типа (**o**) шаблона Sh, и n_R элементов типа (H или V) в зоны шаблона Sh с соблюдением рассмотренных выше свойств польского выражения (1-4). Число занятых разрезами позиций в зоне z_i, может быть равным от 0 до z_i а суммарное число занятых позиций во всех зонах равно n_R. Поиск решения сводится к поиску такого легитимного выражения Sh, которое оптимизирует показатель качества. Разработанный метод синтеза

польского выражения базируется на парадигме муравьиного алгоритма и включает построение дерева разрезов, выбор типов разрезов (*H* или *V*), идентификацию и ориентацию модулей. Для геометрической интерпретации плана выполняется процедура свертки [6–11].

Ядром муравьиного алгоритма является конструктивный алгоритм, с помощью которого каждый член популяции на каждой итерации находит решение задачи. В работе, в отличие от канонической парадигмы МА, эволюционирующая популяция разбита на пары агентов. Каждый член популяции − пара агентов, работающих совместно. Задача планирования решается каждой парой агентов двумя конструктивными алгоритмами. При этом конструктивные алгоритмы *A1* и *A2*, используемые агентами пары различаются. Алгоритм *A1* решает задачу выбора взаимного расположения и ориентации модулей в польском выражении. Алгоритм *A2* решает задачу выбора последовательности, расположения и типа разрезов в польском выражении. Для решения задачи планирования, в отличие от канонической парадигмы МА [6, 8, 11] , в качестве модели пространства поиска решений используются два графа *G1* и *G2*.

Полный двудольный граф $G1=((M1 \cup M2) \cup P, E1)$ (рис. 2) используется алгоритмом *A1* в качестве модели пространства поиска решений при размещении модулей в множестве позиций шаблона *Sh* и выборе ориентации модулей. $(M1 \cup M2)$ − множество вершин графа *G1*, соответствующих множеству модулей (первая доля графа); $M1=\{m1_i | i=1,2,\ldots,n+1\}$, $M2=\{m2_i | i=1,2,\ldots,n+1\}$. Вершины графа *G1* объединены в альтернативные пары $m_i=(m1_i, m2_i)$. Вершина $m1_i$ соответствует модулю в первой ориентации, $m2_i$ – во второй. $P=\{p_j | j=1,2,\ldots,n\}$ множество вершин графа *G1*, соответствующих множеству позиций шаблона *Sh* для размещения в них имен модулей (листьев дерева). Если модуль m_i размещается в первой ориентации, то он обозначается, как $m1_i$, а ребро, обозначается как $(m1_i, p_j)$. Если модуль m_i размещается во второй ориентации, то он обозначается, как $m2_i$, а ребро обозначается, как $(m2_i, p_j)$. *E1* множество ребер, связывающих множество вершин $(M1 \cup M2)$ с множеством вершин *P*.

Задача, решаемая алгоритмом *A1*, формулируется как задача поиска взаимно однозначного отображения $F_k=M^* \rightarrow P$ на графе *G1* (множества модулей $M^* \subset M1 \cup M2$ с выбранными ориентациями, $|M^*|=|M|$ во множество *P* позиций шаблона *Sh*. Фактически решение заключается в выборе на графе *G1* подмножества ребер $E1^* \in E1$, входящих в соответствующее отображение F_k.

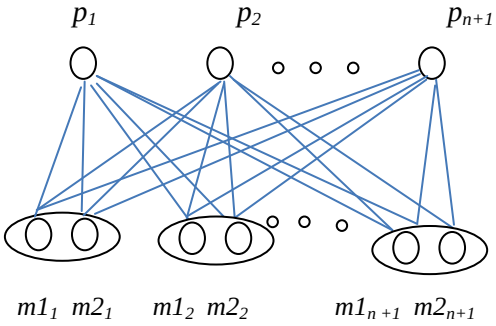

Рис. 2. Граф G1 поиска решений при размещении и выборе ориентации модулей

В качестве коллективной эволюционной памяти (КЭП) алгоритмом *A1* используется матрица КЭП1, в которой для каждого ребра хранятся две интегральные оценки степени пригодности $\delta_{ij}(O_1)$ и $\delta_{ij}(O_2)$, в зависимости от ориентации модуля (рис. 3).

В столбце $\delta_{ij}(O_1)$ таблицы КЭП1 откладывается количество феромона, соответствующее степени пригодности ребра $(m1_i,p_j)$. В столбце $\delta_{ij}(O_2)$ откладывается количество феромона, соответствующее степени пригодности ребра $(m2_i,p_j)$.

модуль	Позиции									
	p_1		p_2		p_3		p_4		p_5	
	степени пригодности δ_{ij}		степени пригодности δ_{ij}		степени пригодности δ_{ij}		степени пригодности δ_{ij}		степени пригодности δ_{ij}	
	$\delta_{ij}(O_1)$	$\delta_{ij}(O_2)$	$\delta_{ij}(O_1)$	$\delta_{ij}(O_2)$	$\delta_{ij}(O_1)$	$\delta_{ij}(O_2)$	$\delta_{ij}(O_1)$	$\delta_{ij}(O_2)$	$\delta_{ij}(O_1)$	$\delta_{ij}(O_2)$
m_1										
m_2										
…										
m_n										

Рис. 3. Матрица коллективной эволюционной памяти КЭП1

В алгоритме *A2* в качестве модели пространства поиска решений для выбора типа, последовательности и места расположения разрезов в шаблоне *Sh* разработан граф *G2=(X, E2)* (рис. 4). $X=\{(x1_i, x2_i)|i=1,2,\ldots,n\}$ множество вершин графа *G2*, соответствует множеству *P* потенциальных позиций шаблона *Sh* для возможного размещения в них символов разрезов. Каждая потенциальная позиция $p_i \in P$ шаблона *Sh* моделируется двумя альтернативными вершинами $(x1_i, x2_i)$. Выбор при размещении разрезов вершины $x1_i$ указывает на то, что в позицию p_i помещен разрез типа *V*, выбор вершины $x2_i$ – указывает на то, что в позицию p_i помещен разрез типа *H*. Граф *G2* отображается на плоскости в виде многостадийной структуры, разбитой на зоны, включающие пары вершин (рис. 4). На рис. 4 представлен граф *G2*, построенный для пяти зон.

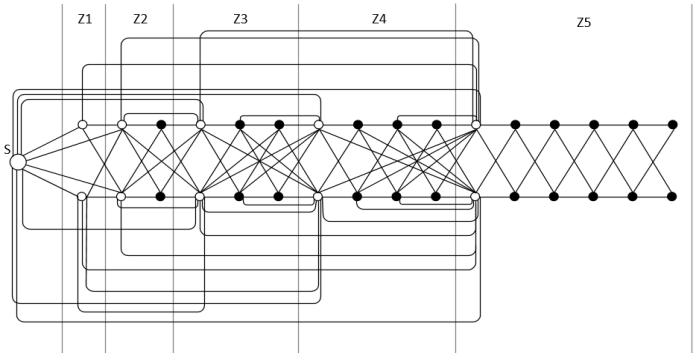

Рис. 4. Модель пространства поиска решений для выбора типа, последовательности и места расположения разрезов в шаблоне Sh

На рис. 5 представлены идентификаторы вершин графа *G2*.

Зона	1		2				3					
Позиция	1		1		2		1		2		3	
Разрез	V	H	V	H	V	H	V	H	V	H	V	H
Идентификатор	11V	11H	21V	21H	22V	22H	31V	31H	32V	32H	33V	33H
№ Вершины	1	2	3	4	5	6	7	8	9	10	11	12
Опорные вершины	1		3				7					

Зона	4								5									
Позиция	1		2		3		4		1		2		3		4		5	
Разрез	V	H	V	H	V	H	V	H	V	H	V	H	V	H	V	H	V	H
Идентификатор	41V	41H	42V	42H	43V	43H	44V	44H	51V	51H	52V	52H	53V	53H	54V	54H	55V	55H
№ Вершины	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Опорные вершины	13										23							

Рис. 5. Матрица идентификаторов графа G2

Каждая позиция p_i в зоне шаблона *Sh* моделируется парой вершин, на плоскости расположенных одна над другой. Верхняя вершина пары соответствует состоянию позиции, при котором в нее помещается разрез типа *V*, нижняя вершина пары соответствует состоянию позиции, при котором в нее помещается разрез типа *H*. В соответствии со структурой шаблона *Sh* в зоне z_i располагается *i* пар вершин, являющиеся кандидатами для размещения в них разрезов. В работе используется упорядоченная идентификация вершин. Вершины проиндексированы в том же порядке в каком соответствующие им позиции расположены в шаблоне *Sh*. Множество вершин *X* разбито на подмножества в зависимости от того в какой зоне шаблона *Sh* расположены потенциальные позиции, соответствующие вершинам.

Для отображения взаимного пространственного расположения вершин графа, зоны и вершины в зонах расположены на плоскости в линейку, в соответствии с расположением позиций шаблона *Sh* на плоскости для размещения в них символов разрезов.

Вершины первой пары в каждой зоне называются опорными. На рис. 4 опорные вершины выделены кружками.

Ориентированные ребра, связывающие вершины графа, формируются следующим образом.

Стартовая вершина *O* маршрута *S* связана ориентированными ребрами (на рис. 4 слева направо) со всеми опорными вершинами.

Каждая опорная вершина связана ориентированными ребрами (на рис.4 слева направо) со всеми опорными вершинами, расположенными на рис.4 справа.

Каждая вершина зоны z_i (опорная или простая) связана ориентированными ребрами (на рис.4 слева направо) с первой (опорной) парой вершин в зоне $z_{i+1.}$

Каждая вершина зоны z_i (опорная или простая) связана с соседней справа парой простых вершин зоны z_i ориентированными ребрами (на рис.4 слева направо).

Алгоритмом А2 решается задача построения маршрута *S* на графе *G2,* начинающегося со стартовой вершины *O* и включающего вершины, соответствующие позициям на шаблоне *Sh* для размещения в них разрезов, с указанием типа, в том порядке, в котором они расположены в шаблоне *Sh*.

В качестве коллективной эволюционной памяти (КЭП2) алгоритма *A2* служит матрица смежности графа *G2*. Значением элемента x_{ij} матрицы смежности графа *G2* является степень пригодности ребра, связывающего вершины x_i и x_j. Степень пригодности ребра определяется количеством отложенного феромона на ребре.

	11V	11H	21V	21H	22V	22H	31V	31H	32V	32H	33V	33H	…
11V	X	X											…
11H	X	X											…
21V			X	X									…
21H			X	X									…
22V					X	X							…
22H					X	X							…
31V							X	X					…
31H							X	X					…
32V									X	X			…
32H									X	X			…
33V											X	X	…
33H											X	X	…
…													…

Рис. 6. Матрица смежности графа G2, используемая в качестве коллективной эволюционной памяти (КЭП2)

Процесс поиска решений итерационный. Эволюционирующая популяция разбита на пары агентов. Каждый член популяции – пара агентов, работающих совместно. При этом конструктивные алгоритмы *A1* и *A2*, используемые агентами пары различаются. Алгоритмы разработаны на основе метода муравьиной колонии. Каждая итерация *l* общего алгоритма включает начальный и три основных этапа.

Начальный этап. Обнуляются вспомогательные матрицы КЭП*1 и КЭП*2.

На первом этапе каждая пара агентов $d_{k=}(a1_k,\ a2_k)$:

- конструктивными алгоритмами *A1* и *A2* на базе графов *G1* и *G2* и коллективной эволюционной памяти – матриц КЭП1 и КЭП2 синтезирует свое собственное решение $W_k=(E1_k^*,S_k)$;
- формируется польское выражение Sh_k, соответствующее решению W_k;
- на базе Sh_k формируется дерево разрезов T_k;
- на базе T_k формируется план R_k и рассчитывается оценка решения $F_{k.}$;
- агенты пары $d_{k=}(a1_k,\ a2_k)$ откладывают (добавляют) феромон в ячейки вспомогательных матриц коллективной эволюционной памяти КЭП*1 и КЭП*2, соответствующие ребрам решения $W_k=(E1_k^*,S_k)$ в графах поиска решений *G1* и *G2* в количестве пропорциональном оценке решения F_k.

На втором этапе после построения всеми парами агентов решений W_k феромон, накопленный в КЭП*1 и КЭП*2 агентами популяции на итерации *l* добавляется в КЭП1 и КЭП2.

На третьем этапе осуществляется испарение феромона на ребрах графов *G1* и *G2* (в КЭП1 и КЭП2).

В работе используется циклический (ant-cycle) метод муравьиных систем. В этом случае феромон откладывается в матрицы КЭП1 и КЭП2 после полного формирования всеми агентами решений на текущей итерации. Для накопления феромона, откладываемого агентами за одну итерацию, используются вспомогательные дублирующие матрицы КЭП*1 и КЭП*2. Другими словами в процессе выполнения итерации агенты по очереди откладывают феромон в КЭП*1 и КЭП*2, а после завершения итерации феромон, накопленный агентами в КЭП*1 и КЭП*2, откладывается в КЭП1 и КЭП2.

Отметим, что оценка решения задачи планирования вычисляется после полного формирования решения $W=(E1^*,S)$ парой агентов $d_{k=}(a1_k,\ a2_k)$ и последующей свертке дерева разрезов [1].

На начальном этапе всем элементам матриц памяти КЭП1 и КЭП2 присваивается значение $\Theta=\xi/\varepsilon$, где $\varepsilon=|E|$.

3. Первый конструктивный алгоритм *A1*. В работе для удобства изложения используется общее обозначение модуля, без указания типа ориентации, как – m_i.

С помощью первого конструктивного алгоритма *A1* решается задача выбора ориентации и назначения множества модулей $M=\{m_i|i=1,2,\ldots,n+1\}$ в множество позиций $P=\{p_j|j=1,2,\ldots,n\}$ шаблона *Sh*. Поиск решения осуществляется первыми агентами $a1_k$ каждой пары $d_k=(a1_k,\ a2_k)$ на графе *G1*. Задача сводится к поиску на полном двудольном графе $G1=((M1\cup M2)\cup P,E1)$ с использованием КЭП1 взаимно однозначного соответствия между множествами вершин *M* и *P*. Формируемое соответствие между *M* и *P* описывается множеством ребер $E1^*\subset E1$.

Основные ограничения формируемого множества ребер $E1^*$, заключаются в следующем:

- каждое ребро $(m_i,\ p_j)\in E1^*$, с одной стороны, инцидентно только одной вершине $m_i\in M$, с другой стороны – инцидентно одной и только одной вершине $p_j\in P$;

♦ формирование каждым агентом $a1_k$ множества ребер $E1^*$осуществляется последовательно по шагам на базе множества $E1$ ребер полного двудольного графа $G1$ (пошагово).

В памяти агента $a1_k$ имеется:

♦ список вершин $M_{1k}(t) \in M$, уже размещенных в позициях $P_{1k}(t) \subset P$;

♦ список вершин $M_{2k}(t) \in M$, оставшихся неразмещенными, $M_{1k}(t) \cup M_{2k}(t)=M$;

♦ список свободных позиций $P_{2k}(t) \subset P$, $P_{1k}(t) \cup P_{2k}(t)=P$.

♦ $\delta_{ij}(t)$ – количество феромона, на каждом ребре (m_i, p_j) графа $G1$.

В зависимости от ориентации модуля − O_1 или O_2 каждому ребру $(m1_i, p_j)$ соответствует оценка степени пригодности $\delta_{ij}(O_1)$, ребру $(m2_i, p_j)$ − $\delta_{ijj}(O_2)$,

Приводится в начальное состояние память агента $a1_k$:

$$M_{1k}(1)=\varnothing;\ M_{2k}(1)=M;\ P_{1k}(1)=\varnothing;\ P_{2k}(1)=P;\ E1^*_k(1)=\varnothing.$$

Формирование агентом $a1_k$ решений $E1^*_k$ производится на множестве ребер $E1$ графа $G1=(M \cup P, E1)$.

На шаге t формируется:

♦ множество вершин $M_{1k}(t) \in M$, уже размещенных в позициях $P_{1k}(t) \subset P$;

♦ множество вершин $M_{2k}(t) \in M$, оставшихся неразмещенными, $M_{1k}(t) \cup M_{2k}(t)=M$;

♦ список свободных позиций $P_{2k}(t) \subset P$, $P_{1k}(t) \cup P_{2k}(t)=P$.

Для каждой пары вершин $m_i \in M_{2k}(t)$ определяется набор ребер $U_i(t)$, $|U_i(t)|=|P_{2k}(t)|$, связывающих m_i с вершинами множества $P_{2k}(t)$. Пусть $U1_i(t)$ − множество ребер $(m1_i, p_j)$, инцидентных вершине $m1_i$, а $U2_i(t)$ − множество ребер $(m2_i, p_j)$, инцидентных вершине $m2_i$.

На основе данных КЭП1 для каждой вершины m_i подсчитываются суммарное количества феромона $\varphi 1_{ki}$, размещенного на ребрах $e_{ij} \in U1_i(t)$, и $\varphi 2_{ki}$, размещенного на ребрах $e_{ij} \in U2_i(t)$: $\varphi 1_{ki}=\sum_j \delta_{ij}(O_1) | e_{ij} \in U1_i(t)$, $\varphi 2_{ki}=\sum_j \delta_{ij}(O_2) | e_{ij} \in U2_i(t)$.

Для каждой вершины m_i подсчитываются общее суммарное количество феромона $\Omega_{ki}= \varphi 1_{ki} + \varphi 2_{ki}$.

Среди вершин $m_i \in M_{2k}(t)$ отыскивается вершина m_α с максимальным значением Ω_{ki}.

Если $(\varphi 1_{k\alpha})_{max} > (\varphi 2_{ki})_{max}$, то агент $a1_k$ среди ребер $U1_\alpha(t) \subset U_i(t)$, инцидентных выбранной вершине m_α, с вероятностью $\Psi_\alpha(t)=\delta_{\alpha j}(O_1)(t)/\varphi 1_{ki}$ выбирает ребро $e_{\alpha j}=(m1_\alpha, p_j)$, которое включается в формируемое агентом $a1_k$ множество ребер $E1_k^*(t+1)$.

Если $(\varphi 2_{k\alpha})_{max} > (\varphi 1_{ki})_{max}$, то агент $a2_k$ среди ребер $U2_\alpha(t) \subset U_i(t)$, инцидентных выбранной вершине m_α, с вероятностью $\Psi_\alpha(t)=\delta_{\alpha j}(O_2)(t)/\varphi 1_{ki}$ выбирается ребро $e_{\alpha j}=(m2_\alpha, p_j)$, которое включает в формируемое агентом $a1_k$ множество ребер $E1_k^*(t+1)$.

Далее, выполняются постпроцедуры.

$E1_k^*(t+1)=E1_k^*(t+1) \cup e_{\alpha j}$.

$M_{1k}(t+1)=M_{1k}(t) \cup m_\alpha$; $M_{2k}(t+1)=M_{2k} \setminus m_\alpha$.

$P_{1k}(t+1)=P_{1k} \cup p_j$; $P_{2k}(t+1)=P_{2k} \setminus p_j$.

Переход к следующему шагу.

Процесс формирования агентом $a1_k$ множества $E1_k^*$завершается при $M_{2k}(t)=P_{2k}(t)=\varnothing$.

4. Второй конструктивный алгоритм *A2*. Второй конструктивный алгоритм $A2$ решает задачу выбора типа разреза, зоны на шаблоне Sh и места в зоне для размещения в нем разреза.

Поиск решения осуществляется вторыми агентами $a2_k$ каждой пары $d_k=(a1_k, a2_k)$ на разработанной модели (граф *G2)* пространства поиска решений, учитывающей специфику решаемой задачи.

Задача сводится к поиску на графе *G2=(X,E2),* с использованием КЭП2, маршрута S_k, включающего вершины, соответствующие позициям на шаблоне *Sh,* для размещения в них разрезов, с указанием типа, в том порядке, в котором они расположены в шаблоне *Sh.*

Формирование каждым агентом $a2_k$ маршрута S_k осуществляется на графе *G2* последовательно по шагам (пошагово), начиная с начальной вершины *O.*

Агенты обладают памятью. Вершины графа *G2* могут находиться в одном из двух состояний – активном и пассивном, которое меняется в процессе построения маршрута. Для отображения состояния вершин используется массив $\Theta=\{\theta_i|i=1,2,\ldots,n\}$ текущего состояния вершин графа *G2.* Если θ_i=1, то вершина x_i в активном состоянии и может служить кандидатом для размещения в ней разрезов, если θ_i=0, то вершина x_i в пассивном состоянии и не является кандидатом для размещения в ней разреза. На каждом шаге *t* в памяти агента $a2_k$ хранится:

- число зон и разрезов N_z формируемого польского выражения;
- число N_r и список разрезов (вершин графа *G2),* уже вошедших в состав формируемого маршрута $S_k(t)$;
- последняя вершина *end(t-1)* вошедшая в состав маршрута $S_k(t-1)$ на шаге *(t-1);*
- параметры вершины *end(t-1)*: z_e – номер зоны, в которой размещена вершина *end(t-1),* τ_e – тип вершины *end(t-1)*: $\tau_e=1$ – опорная или τ_e=0 простая вершина*;*

Приводится в начальное состояние память агента $a2_k$:

Элементы массива $\Theta=\{\theta_i|i=1,2,\ldots,n\}$ приводятся в активное состояние, т.е. $(\forall i)[\theta_i=1]$; фиксируется: число зон и разрезов N_z; $t=0$; $N_r(t)=0$; $S_k(t)=\varnothing$; $end(t)=O$; $z_e=0$; $\tau_e=0$.

Первой процедурой на шаге *t* формируется список ребер U^k_e графа *G2*, которые исходят из вершины *end(t-1)* и входят в находящиеся в активном состоянии вершины множества X^k_e.

На базе списка X^k_e формируется список $XK^k_e \subset X^k_e$ вершин – кандидатов для выбора очередной позиции, не приводящих к нарушению ограничений польского выражения.

Формирование XK^k_e, производится в соответствии со структурой графа *G2*, описанной выше (см. рис. 4). В XK^k_e включаются:

1) все, если такие есть, опорные вершины, входящие в X^k_e;

2) ближайшая в зоне z_e к вершине *end(t-1)* пара простых вершин, если такая есть.

Далее, среди ориентированных ребер, входящих в вершины множества XK^k_e, выбирается ребро e_{max}, на котором в графе *G2* отложено максимальное количество феромона и вершина x_{max}, в которую входит ребро e_{max}. Ребро включается в маршрут $S_k(t)$. Выполнение постпроцедуры: $N_r(t)=N_r(t)+1$; $S_k(t)=S_k(t-1)\cup e_{max}$.

Фиксация последней вершины *end(t),* вошедшей в состав маршрута $S_k(t)$ на шаге *(t)* и ее параметров: z_e, τ_e.

Во всех зонах последняя в активном состоянии пара простых вершин переходит в пассивное состояние. Переход к следующему шагу.

Процесс формирования агентом $a2_k$ маршрута S_k. завершается после распределения по позициям всех разрезов.

После построения агентами пары $d_k=(a1_k, a2_k)$ решения $W=(E1^*,S_k)$, оно последовательно трансформируется в польскую запись, дерево разрезов, а затем алгоритмом свертки в план $R_k(l)$ для которого рассчитывается оценка решения $F_k(l)$. Оценки, полученные всеми агентами на одной итерации суммируются и вносятся в матрицы коллективной эволюционной памяти. После испарения феромона в матрицах КЭП выполняется следующая итерация МА.

5. Эволюционный алгоритм планирования.

1. Задается число модулей и их геометрические параметры.

2. Задается критерий оптимизации.

3. В соответствии с исходными данными формируются шаблон позиций польского выражения в виде вектора $Sh=<\mathbf{o}\ \mathbf{o}\ \underline{1}\ \mathbf{o}\ \underline{2}\ \mathbf{o}\ \underline{3}\ \mathbf{o}\ \underline{4}\ \ldots\ \mathbf{o}\ \underline{n_R}>$.

4. В соответствии с исходными данными формируются две модели пространства поиска решений: два графа поиска решений *G1* и *G2*.

5. Задаются значения управляющих параметров.

6. Задается число пар агентов N_a в популяции.

7. Задается число итераций – N_l.

8. Формируются матрицы коллективной эволюционной памяти КЭП1 и КЭП2. На всех ребрах исходных графов поиска решений *G1* и *G2* (в соответствующих ячейках КЭП1 и КЭП2) откладывается начальное количество феромона.

9. *l=1. (l – номер итерации)*

10. Обнуление элементов матриц КЭП*1 и КЭП*2.

Начало работы 1 конструктивного алгоритма A1.

11. 7. *k=1. (k – номер агента).*

12. Формируются начальные значения параметров памяти для агента $a1_k$ популяции Z_1. $t=0$. $M_{1k}(t)=\varnothing$. $M_{2k}(t)=M$. $P_{1k}(t)=\varnothing$. $P_{2k}(t)=P$. $E1^k(t)=\varnothing$.

13. $t=t+1$. *(t – номер шага)*.

14. Для каждой пары вершин $m_i \in M_{2k}(t)$ определяется набор ребер $U_i(t)$, $|U_i(t)|=|P_{2k}(t)|$, связывающих m_i с вершинами множества $P_{2k}(t)$. Пусть $U1_i(t)$ − множество ребер $(m1_i, p_j)$, инцидентных вершине $m1_i$, а $U2_i(t)$ − множество ребер $(m2_i, p_j)$, инцидентных вершине $m2_i$.

15. На основе данных КЭП1 для каждой вершины m_i подсчитываются суммарное количества феромона $\varphi 1_{ki}$, размещенного на ребрах $e_{ij} \in U1_i(t)$, и $\varphi 2_{ki}$, размещенного на ребрах $e_{ij} \in U2_i(t)$: $\varphi 1_{ki}=\sum_j \delta_{ij}(O_1) | e_{ij} \in U1_i(t)$, $\varphi 2_{ki}=\sum_j \delta_{ij}(O_2) | e_{ij} \in U2_i(t)$.

16. Для каждой вершины m_i подсчитываются общее суммарное количество феромона $\Omega_{ki}=\varphi 1_{ki} + \varphi 2_{ki}$.

17. Среди вершин $m_i \in M_{2k}(t)$ отыскиваются вершина m_α с максимальным значением Ω_{ki}.

18. Если $(\varphi 1_{k\alpha})_{max} > (\varphi 2_{ki})_{max}$, то агент $a1_k$ с вероятностью $\Psi_\alpha(t)=\delta_{\alpha j}(O_1)(t)/\varphi 1_{ki}$ включает ребро $e_{\alpha j}=(m1_\alpha, p_j)$, инцидентное вершине m_α, в формируемое агентом множество $E1_k^*(t+1)$.

Если $(\varphi 2_{k\alpha})_{max} > (\varphi 1_{ki})_{max}$, то агент $a2_k$ с вероятностью $\Psi_\alpha(t)=\delta_{\alpha j}(O_2)(t)/\varphi 2_{ki}$ включает ребро $e_{\alpha j}=(m2_\alpha, p_j)$, инцидентное вершине m_α, в формируемое агентом множество $E1_k^*(t+1)$.

19. Выполнение постпроцедур после шага *t*.

$E1_k^*(t+1)=E1_k^*(t+1) \cup e_{\alpha j}$.

$M_{1k}(t+1)=M_{1k}(t) \cup m_\alpha$; $M_{2k}(t+1)=M_{2k} \backslash m_\alpha$.

$P_{1k}(t+1)=P_{1k} \cup p_j$; $P_{2k}(t+1)=P_{2k} \backslash p_j$.

20. Если $M_{2k}(t+1)=\varnothing$, т.е все модули получили назначение и ориентацию агентом $A1_k$, то переход к 21 (алгоритму *A2*), иначе переход к 14. Переход к следующему шагу.

Процесс формирования агентом $a1_k$ множества $E1_k^*$ завершается при $M_{2k}(t)=P_{2k}(t)=\varnothing$.

21. *Начало работы 2 конструктивного алгоритма.(2й агент $a2_k$).*

22. Формируются начальные значения параметров памяти для агента $a2_k$ популяции Z_1. Фиксируется: число зон и разрезов N_z. $t=0$. $N_r=0$. $S_k(t-1)=\varnothing$. $end(t)=\varnothing$., z_e, τ_e.

Все элементы массива $\Theta=\{\theta_i | i=1,2,\ldots,n\}$ приводятся в активное состояние, т.е. $(\forall i)[\theta_i=1]$.

23. $t=t+1$. *(t – номер шага).*

24. Формируется список ребер $U^k_e(t)$ графа *G2,* исходящих из вершины *end(t-1)* и список находящихся в активном состоянии вершин множества X^k_e, в которые входят ребра списка $U^k_e(t)$.

25. На базе списка $X^k_e(t)$ формируется список $XK^k_e(t) \subset X^k_e(t)$ вершин – кандидатов для выбора очередной позиции, не нарушающих ограничений польского выражения.

26. Среди ребер, входящих в вершины множества $XK^k_e(t)$ выбирается ребро e_{max}, на котором в графе *G2* отложено максимальное количество феромона, Это ребро включается в маршрут $S_k(t)$.

27. Выполнение постпроцедур после шага *t.* $N_r=N_r+1$. $S_k(t)=S_k(t-1) \cup e_{max}$.

Фиксация последней вершины *end(t),* вошедшей в состав маршрута $S_k(t)$ на шаге *(t)* и ее параметров: z_e, τ_e. Во всех зонах последняя в активном состоянии пара простых вершин переходит в пассивное состояние.

28. Если $N_r < N_z$, то переход к 19 (алгоритму *A2*) , иначе переход к 24.

29. Формируется польское выражение $Sh_k(l)$, соответствующее решению $W_k(l)=(E1_k^*, S_k)$, полученному парой агентов $d_{k=}(a1_k, a2_k)$.

30. На базе W_k формируется дерево разрезов $T_k(l)$.

На базе $T_k(l)$ алгоритмом свертки формируется план $R_k(l)$ и рассчитывается оценка решения $F_k(l)$. Расчет количества феромона пропорционального оценке $F_k(l)$ $\tau_k(l)= Q/ F_k(l)$.

31. Агенты откладывают (добавляют) феромон в ячейки матриц коллективной эволюционной памяти КЭП*1 и КЭП*2, соответствующие ребрам решения $W_k=(E1_k,S_k)$ в графах поиска решений *G1* и *G2* в количестве пропорциональном оценке решения $F_k(l)$.

32. Если $k<N_a$, то $k=k+1$ и переход к 12, иначе переход к 33.

33. Сложение матриц КЭП1 и КЭП*1, КЭП2 и КЭП*2.

34. Выполняется процедура испарения феромона на ребрах графов поиска решений *G1* и *G2*.

35. Если $l<N_l$, то $l=l+1$ и переход к 7, иначе переход к 36.

36. Конец работы алгоритма.

6. Экспериментальные исследования. Экспериментальные исследования программы **ПМК** проводились на контрольных примерах с известным оптимумом $F_{опт}$, синтезированные известным методом AFEKO − Floorplanning Examples with Known Optimal area [14–22].

При проведении испытаний фиксировалось число итерации, при которых алгоритм находил лучшее решение решению. Сравнения результатов производилась по показателю $F_{опт}/F$ – ***«степень качества»,*** где F – оценка полученного решения. На основе обработки экспериментальных исследований была построена средняя зависимость степени качества от числа итераций и от размера популяции (рис. 7). Анализ результатов показал, что лучшие решения были получены в пределах 120–130 итерации. Решение близкое к оптимальному, было получено среднем на 125 итерации. В результате анализа показателей качества, полученных разработанным алгоритмом на тестовых примерах с известным оптимумом установлено, что у 70 % примеров значение показателя оптимально, у 15 % примеров – на 5 % хуже, а у 15 % – хуже не более, чем на 2 %.

Так же были проведены экспериментальные исследования, характеризующие зависимость времени выполнения алгоритма от размера задачи. Для определения временной сложности алгоритма при проведении тестовых испытаний исследовалась зависимость продолжительности работы алгоритма от количества блоков. Общая оценка лежит в пределах $O(n^2)$-$O(n^3)$.

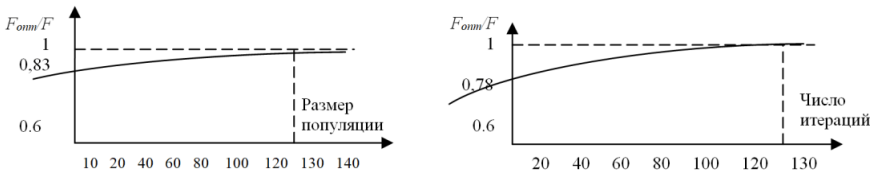

Рис. 7. Зависимость качества алгоритма от размера популяции и числа итераций

Для сравнения качества разработанного алгоритма планирования использовались стандартные тесты (бенчмарки), используемые разработчиками для оценки разработанных алгоритмов [12, 13]. Стандартные тесты (бенчмарки) созданы на MCNC (Microelectronics Center of North Carolina) образцах элементов. Были использованы пять типов корпусов, в которых количество элементов варьировалось от 9 до 49. Самый большой корпус – для платы ami49 на 49 элементов, с 42 интерфейсами ввода-вывода, 408 связями и 931 ножкой. Сравнение производилось по значению площади.

Для сравнения были выбраны наиболее известные поисковые алгоритмы планирования: АД – адаптивный [3]; МА – меметический [14]; ГЕН – генетический [15]; ГА – гибридный [16]; ПМК – разработанный муравьиный Эксперименты разработанного алгоритма проводились на ЭВМ типа IBM PC с процессором Pentium, а результаты у сравниваемых алгоритмов получены на базе платформы Ultra1-Spark.

Результаты разработанного алгоритма по площади превосходят результаты существующих алгоритмов (табл.1). Эксперименты показали, что при больших размерностях временные показатели разработанных алгоритмов превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции, отклонение целевой функции от оптимального значения меньше в среднем на 6 %.

Таблица 1

Сравнение результатов работы алгоритмов

Бенч марка	АЛГОРИТМ				
	АД	ГЕН	МА	ГА	**ПМК**
apte	49425136,4	49425136,4	49396354	49396354	49196354
xerox	24954567,8	24954567,8	24873723	24895479	23957366
hp	12232485	12232485	12176343	12194845	11393236
ami33	3510173,5	3530173,4	3520854	3538838	34190074
ami49	136133614,4	136133614,4	134285950	137343334	128591325

АД – адаптивный; ГЕН – генетический; МА – меметический; ГА – гибридный; **ПМК** – муравьиный.

Заключение. В работе для решения задачи планирования СБИС разработан поисковый алгоритм на основе модифицированного метода муравьиной колонии. Задача формирования плана СБИС сведена к задаче соответствующего польского выражения. Предложена методика формирования польского выражения на основе шаблона, заключающаяся в назначении элементов, соответствующих модулям и разрезам, в позиции шаблона *Sh*, с соблюдением свойств польского выражения. Поиск решения сводится к поиску такого легитимного выражения *Sh*, которое оптимизирует показатель качества.

Разработанный метод синтеза польского выражения базируется на парадигме муравьиного алгоритма и включает построение дерева разрезов, выбор типов разрезов (*H* или *V*), идентификацию и ориентацию модулей. Для геометрической ин-

терпретации плана выполняется процедура свертки [6]. Ядром муравьиного алгоритма является конструктивный алгоритм, с помощью которого каждый член популяции на каждой итерации находит решение задачи. В работе, в отличие от канонической парадигмы МА, эволюционирующая популяция разбита на пары агентов. Каждый член популяции – пара агентов, работающих совместно. Задача планирования решается каждой парой агентов двумя конструктивными алгоритмами. При этом конструктивные алгоритмы *A1* и *A2*, используемые агентами пары различаются. Алгоритм *A1* решает задачу выбора взаимного расположения и ориентации модулей в польском выражении. Алгоритм *A2* решает задачу выбора последовательности, расположения и типа разрезов в польском выражении. Для решения задачи планирования методом, отличным от канонической парадигмы МА, разработаны модели пространства поиска решений – два графа *G1* и *G2*. Полный двудольный граф *G1* используется алгоритмом *A1* в качестве модели пространства поиска решений при размещении и выборе ориентации модулей в множестве позиций шаблона *Sh*. В алгоритме *A2* в качестве модели пространства поиска решений для выбора типа, последовательности и места расположения разрезов в шаблоне *Sh* разработан граф $G2=(X, E2)$. $X=\{(x1_i. x2_i)|i=1,2,\ldots,n\}$ множество вершин графа *G2*, соответствует множеству *P потенциальных* позиций шаблона *Sh* для возможного размещения в них разрезов. Матрица смежности графа *G2*, используется в качестве коллективной эволюционной памяти. Значением элемента x_{ij} матрицы смежности графа *G2* является степень пригодности ребра, связывающего вершины x_i и x_j.

Тестовые испытания подтвердили эффективность предложенного метода. Временная сложность алгоритма, полученная экспериментальным путем, совпадает с теоретическими исследованиями и для рассмотренных тестовых задач составляет $O(n^2)$.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Курейчик В.М., Лебедев Б.К., Лебедев В.Б.* Планирование сверхбольших интегральных схем на основе интеграции моделей адаптивного поиска // Известия РАН. Теория и системы управления. – 2013. – № 1. – С. 84-101.
2. *Норенков И.П.* Основы автоматизированного проектирования: учебник. – М.: Изд-во МГТУ имени Н.Э. Баумана, 2006. – 336 с.
3. *Kureichik V.M., Lebedev B.K., Lebedev O.B.* Hybrid evolutionary algorithm of planning VLSI // Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference. – Portland, 2010. – P. 821-822.
4. *Pritha B., Megha S., Susmita S.-K.* Floorplanning for Partially Reconfigurable FPGAs // IEEE Trans. Comput.-Aided Des. Integr. Circuits Sys. – 2011. – No. 1. – P. 8-17.
5. *Карпенко А.П.* Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой: учебное пособие. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2014. – 448 с.
6. *Лебедев О.Б.* Модели адаптивного поведения муравьиной колонии в задачах проектирования. – Таганрог. Изд-во ЮФУ, 2013. – 199 с.
7. *Лебедев Б.К., Лебедев О.Б., Лебедев В.Б.* Гибридизация роевого интеллекта и генетической эволюции на примере размещения // Электронный журнал “Программные продукты, системы и алгоритмы». – Тверь: Изд-во «Центрпрограммсистем», 2017. – № 4.
8. *Лебедев О.Б.* Планирование СБИС на основе метода муравьиной колонии // Известия ЮФУ. Технические науки. – 2010. – № 7. – С. 67-73.
9. *Mourelle M.* Swarm intelligent systems. – Berlin: Heidelberg: Springer Verlag, 2006. – 217 p.
10. *Qi L. et al.* Simulated annealing based thermal-aware floor planning // International Conference on Electronics, Communications and Control. – 2011. – P. 463-466.
11. *Лебедев Б.К., Лебедев О.Б.* Биоинспирированные методы планирования кристалла СБИС // Тр. VI Всероссийской научно-технической конференции «Проблемы разработки перспективных микро- и наноэлектронных систем». Сборник трудов. – М.: ИППМ РАН, 2012. – С. 171-176.

12. *Ерошенко И.Н.* Разработка генетического алгоритма кластерного планирования СБИС // Известия ЮФУ. Технические науки. – 2010. – № 7. – С. 54-60.
13. *Sherwani N.A.* Algorithms for VLSI Physical Design Automation. – Third Edition, Kluwer Academic Publisher. – USA: 2013. –572 p.
14. *Potti S., Pothiraj S.* GPGPU Implementation of Parallel Memetic Algorithm for VLSI Floorplanning Problem // Trends in Computer Science, Engineering and Information Technology, Communications in Computer and Information Science. – 2011. – P. 432-441.
15. *Ерошенко И.Н.* Методы адаптации генетических алгоритмов к задаче планирования СБИС // Тр. конгресса по интеллектуальным системам и информационным технологиям «IS-IT'11». – М.: Физматлит, 2011. – С. 138-145.
16. *Chen J., Zhu W.* A hybrid genetic algorithm for VLSI floorplanning // IEEE International Conference on Intelligent Computing and Intelligent Systems. – 2010. – P. 128-132.
17. *Cong J., Nataneli G., Romesis M., Shinnerl J.* An Area-Optimality Study of Floorplanning // Proceeding of the International Symposium on Physical Design. – Phoenix, AZ, 2004. – P. 78-88.
18. *Cong J., Romesis M., Xie M.* UCLA Optimality Study Project. http://cadlab.cs.ucla.edu/ ~pubbench. 2004.
19. MCNC Electronic and Information Technologies (Online). Available: www.mcnc.org.
20. hMetis [Online]. Available: http://www-users.cs.umn.edu/karypis/metis/hmet300. HB Floorplan Benchmarks [Online]. Available: http://cadlab.cs.ucla.edu/ cpmo/HBsuite.html.
21. IBM-PLACE 2.0 benchmark suits [http://er.cs.ucla.edu/benchmarks/-ibm-place2/bookshelf/ ibm-place2-all-bookshelf-nopad.tar.gz].
22. *Adya S.N.* ISPD02 IBM-MS Mixed-size Placement Benchmarks [http://vlsicad.eecs.umich. edu/BK/ISPD02bench/].

REFERENCES

1. *Kureychik V.M., Lebedev B.K., Lebedev V.B.* Planirovanie sverkhbol'shikh integral'nykh skhem na osnove integratsii modeley adaptivnogo poiska [Planning of very large-scale integrated circuits based on the integration of adaptive search models], *Izvestiya RAN. Teoriya i sistemy upravleniya* [News RAS. Theory and Control Systems], 2013, No. 1, pp. 84-101.
2. *Norenkov I.P.* Osnovy avtomatizirovannogo proektirovaniya: uchebnik [Basics of computer-aided design: textbook]. Moscow: Izd-vo MGTU imeni N.E. Baumana, 2006, 336 p.
3. *Kureichik V.M., Lebedev B.K., Lebedev O.B.* Hybrid evolutionary algorithm of planning VLSI, *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference*. Portland, 2010, pp. 821-822.
4. *Pritha B., Megha S., Susmita S.-K.* Floorplanning for Partially Reconfigurable FPGAs, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Sys*., 2011, No. 1, pp. 8-17.
5. *Karpenko A.P.* Sovremennye algoritmy poiskovoy optimizatsii. Algoritmy, vdokhnovlennye prirodoy: ucheb. posobie [Modern search engine optimization algorithms. Algorithms inspired by nature: textbook]. Moscow: Izd-vo MGTU im. N.E. Baumana, 2014, 448 p.
6. *Lebedev O.B.* Modeli adaptivnogo povedeniya murav'inoy kolonii v zadachakh proektirovaniya [Models of adaptive behavior of an ant colony in design problems]. Taganrog. Izd-vo YuFU, 2013, 199 p.
7. *Lebedev B.K., Lebedev O.B., Lebedev V.B.* Gibridizatsiya roevogo intellekta i geneticheskoy evolyutsii na primere razmeshcheniya [Hybridization of swarm intelligence and genetic evolution on the example of placement], *Elektronnyy zhurnal "Programmnye produkty, sistemy i algoritmy»* [Electronic journal "Software products, systems and algorithms"]. Tver': Izd-vo «TSentrprogrammsistem», 201, No. 4.
8. *Lebedev O.B.* Planirovanie SBIS na osnove metoda murav'inoy kolonii [VLSI planning based on the ant colony method], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2010, No. 7, pp. 67-73.
9. *Mourelle M.* Swarm intelligent systems. Berlin: Heidelberg: Springer Verlag, 2006, 217 p.
10. *Qi L. et al.* Simulated annealing based thermal-aware floor planning, *International Conference on Electronics, Communications and Control*, 2011, pp. 463-466.
11. *Lebedev B.K., Lebedev O.B.* Bioinspirirovannye metody planirovaniya kristalla SBIS [Bioinspired methods of VLSI crystal planning], *Tr. VI Vserossiyskoy nauchno-tekhnicheskoy konferentsii «Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh sistem». Sbornik*

trudov [Proceedings of the VI All-Russian scientific and technical conference «Problems of the development of promising micro- and nanoelectronic systems». Collection of works]. Moscow: IPPM RAN, 2012, pp. 171-176.
12. *Eroshenko I.N.* Razrabotka geneticheskogo algoritma klasternogo planirovaniya SBIS // *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2010, No. 7, pp. 54-60.
13. *Sherwani N.A.* Algorithms for VLSI Physical Design Automation. Third Edition, Kluwer Academic Publisher. USA: 2013, 572 p.
14. *Potti S., Pothiraj S.* GPGPU Implementation of Parallel Memetic Algorithm for VLSI Floorplanning Problem, *Trends in Computer Science, Engineering and Information Technology, Communications in Computer and Information Science*, 2011, pp. 432-441.
15. *Eroshenko I.N.* Metody adaptatsii geneticheskikh algoritmov k zadache planirovaniya SBIS [Methods for adapting genetic algorithms to the VLSI planning problem], *Tr. kongressa po intellektual'nym sistemam i informatsionnym tekhnologiyam «IS-IT'11»* [Proceedings of the Congress on Intelligent Systems and Information Technologies «IS-IT'11»]. Moscow: Fizmatlit, 2011, pp. 138-145.
16. *Chen J., Zhu W.* A hybrid genetic algorithm for VLSI floorplanning, *IEEE International Conference on Intelligent Computing and Intelligent Systems*, 2010, pp. 128-132.
17. *Cong J., Nataneli G., Romesis M., Shinnerl J.* An Area-Optimality Study of Floorplanning, *Proceeding of the International Symposium on Physical Design*. Phoenix, AZ, 2004, pp. 78-88.
18. *Cong J., Romesis M., Xie M.* UCLA Optimality Study Project. Available: http://cadlab.cs.ucla.edu/ ~pubbench. 2004.
19. MCNC Electronic and Information Technologies (Online). Available: www.mcnc.org.
20. hMetis [Online]. Available: http://www-users.cs.umn.edu/karypis/metis/hmet300. HB Floorplan Benchmarks [Online]. Available: http://cadlab.cs.ucla.edu/ cpmo/HBsuite.html.
21. IBM-PLACE 2.0 benchmark suits [http://er.cs.ucla.edu/benchmarks/-ibm-place2/bookshelf/ ibm-place2-all-bookshelf-nopad.tar.gz].
22. *Adya S.N.* ISPD02 IBM-MS Mixed-size Placement Benchmarks [http://vlsicad.eecs.umich.edu/BK/ISPD02bench/].

Статью реуемендовал к опубликованию д.ф.-м.н., профессор А.П. Карпенко.

Лебедев Олег Борисович – Южный федеральный университет; e–mail: lebedev.ob@mail.ru; г. Таганрог, Россия; тел.: 89085135512; кафедра систем автоматизированного проектирования; доцент.

Жиглатый Артемий Александрович – e-mail: artemiy.zhiglaty@gmail.com; тел.: 89185916819; кафедра систем автоматизированного проектирования; аспирант.

Лебедева Екатерина Олеговна – e-mail: lbedevakate@mail.ru; тел.: 89289591426; кафедра систем автоматизированного проектирования; аспирант.

Lebedev Oleg Borisovich – Southern Federal University; e-mail: lebedev.ob@mail.ru; Taganrog, Russia; phone: +79085135512; the department of computer aided design; associate professor

Zhiglatiy Artemy Alexandrovich – e-mail: artemiy.zhiglaty@gmail.com; phone: +79185916819; the department of computer aided design; graduate student.

Lebedeva Ekaterina Olegovna – e-mail: lbedevakate@mail.ru; phone: +79289591426; the department of computer aided design; graduate student.