Жиленков Михаил Александрович – Южный федеральный университет; e-mail: MZhilenkov777@gmail.com; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 88634371651; кафедра систем автоматизированного проектирования; аспирант.

Курейчик Владимир Викторович – e-mail: vkur@sfedu.ru; кафедра систем автоматизированного проектирования; зав. кафедрой; д.т.н.; профессор.

Zhilenkov Mikhail Aleksandrovich – Southern Federal University; e-mail: MZhilenkov777@gmail.com; 44, Nekrasovskiy, Taganrog, 347928; phone: +78634371651; the department of computer aided design; graduate student.

Kureichik VladimirVictorovich – e-mail: vkur@sfedu.ru; the department of computer aided design; head of department; dr. of eng. sc.; professor.

УДК 004.896

Б.К. Лебедев, О.Б. Лебедев, Е.М. Лебедева

МОДЕРНИЗИРОВАННЫЙ МУРАВЬИНЫЙ АЛГОРИТМ СИНТЕЗА ИДЕНТИФИЦИРОВАННОГО ДЕРЕВА ГИЛЬОТИННОГО РАЗРЕЗА ПРИ ПЛАНИРОВАНИИ СБИС*

Задача планирования решается на основе модифицированной канонической парадигмы муравьиной колонии. Для представления плана используется бинарное дерево гильотинного разреза с идентифицированными вершинами. Структура дерева разрезов задается в виде польского выражения для бинарного дерева. Сформулированы основные свойства польского выражения, выполнение которых необходимо, чтобы записи соответствовало дерево разрезов. В работе рассматривается подход, при котором формирование структуры дерева разрезов муравьиным алгоритмом и разметка всех вершин дерева производится параллельно. Задача синтеза дерева разрезов с идентификацией вершин сводится к задаче формирования соответствующего польского выражения. Для отражения коллективной эволюционной памяти в течение жизни популяции муравьев и для формирования решения задачи в работе используется полный граф с альтернативными состояниями вершин. Вершины множества X_1 соответствуют множеству модулей М. Каждая вершина может находиться в одном из двух альтернативных состояний, соответствующих типу ориентации модуля. Вершины множества X_2 соответствуют разрезам. Каждая вершина может находиться в одном из двух альтернативных состояний, соответствующих типу разреза – горизонтальный или вертикальный. Ключевая проблема, которая была решена в данной работе, связана с разработкой композитной структуры пространства решений, позволяющей одновременно учитывать при построении дерева разрезов ориентацию элементов, тип разреза и метку модуля. Разработана структура модифицированного польского выражения позволяющая, учитывать ориентацию элементов, тип разреза и метку модуля. Это позволило снизить временную сложность задачи планирования и повысить качество решения. Процесс поиска решений итерационный. Каждая итерация l включает три этапа. На первом этапе муравей находит решение. Разработаны эвристики поведения муравья при перемещениях в графе поиска решений, позволяющие формировать легитимный маршрут. На втором этапе муравьи откладывают феромон, на третьем этапе осуществляется испарения феромона. Построенный агентом маршрут трансформируется в польское выражение, которое в свою очередь трансформируется в дерево разрезов с идентифицированными вершинами, а дерево разрезов трансформируется в план. Для написания программы планирования СБИС методом муравьиной колонии был использован язык C++ в среде Microsoft Visual Studio 2010 для ОС Windows, так как главный упор делался на скорость работы приложения. Для проведения экспериментов программы была использована процедура синтеза контрольных примеров с известным оптимумом $F_{опт}$ по аналогии с

* Работа выполнена при финансовой поддержке РФФИ (проект № 15–01–05297).

известным методом AFEKO – Floorplanning Examples with Known Optimal area. Исследованию подвергались примеры, содержащие до 1000 модулей. Оценкой качества служит «степень качества» – величина $F_{опт}/F$, где F – оценка полученного решения. Сравнение с известными алгоритмами показало, что при меньшем времени работы у полученных с помощью разработанного алгоритма решений отклонение целевой функции от оптимального значения меньше в среднем на 6 %. Временная сложность этого алгоритма зависит от времени жизни колонии l (число итераций), количества вершин графа n и числа муравьев m, и определяется как $O(l \cdot n^2 \cdot m)$.

Блоки; планирование; дерево разрезов; польское выражение; синтез; альтернатива; парадигма; муравьиный алгоритм.

B.K. Lebedev, O.B. Lebedev, E.M. Lebedeva

MODERNIZED ANT MARK ALGORITHM FOR THE SYNTHESIS OF THE IDENTIFIED TREE OF THE GUILOTE SECTION IN VLSI PLANNING

The planning task is solved on the basis of the modified canonical paradigm of the ant colony. To represent the plan, a binary tree of the guillotine cut with identified vertices is used. The structure of the tree of cuts is given in the form of a Polish expression for a binary tree. The basic properties of the Polish expression are formulated, the execution of which is necessary so that the records correspond to the tree of cuts. In this paper we consider the approach in which the formation of the tree structure of the sections by the ant algorithm and the marking of all tree vertices is carried out in parallel. The task of synthesizing a cut tree with vertex identification is reduced to the task of forming the corresponding Polish expression. To reflect the collective evolutionary memory during the lifetime of the ant population and to form the solution of the problem, a complete graph with alternative states of vertices is used in the work. The vertices of the set X_1 correspond to the set of modules M. Each vertex can be in one of two alternative states corresponding to the type of orientation of the module. The vertices of the set X_2 correspond to cuts. Each vertex can be in one of two alternative states, corresponding to the type of cut - horizontal or vertical. The key problem that has been solved in this paper is related to the development of a composite structure of the solution space that allows you to simultaneously take into account the orientation of the elements, the cut type and the module label when constructing the tree of sections. The structure of the modified Polish expression allowing us to take into account the orientation of the elements, the type of the cut and the label of the module is developed. This helped to reduce the time complexity of the planning task and improve the quality of the solution. The process of finding solutions is iterative. Each iteration l consists of three steps. At the first stage, the ant finds a solution. Developed heuristics of the ant's behavior when moving in the solution search graph, allowing to form a legitimate route. In the second stage, the ants lay pheromone, in the third stage the pheromone is evaporated. The agent's route is transformed into a Polish expression, which in turn is transformed into a tree of cuts with identified vertices, and the tree of cuts is transformed into a plan. The C ++ programming language was used in the Microsoft Visual Studio 2010 environment for Windows with the method of ant colony, because the main emphasis was on the speed of the application. For the experiments of the program, a procedure was used to synthesize test cases with the known optimum F_{opt}, in analogy with the known method AFEKO – Floorplanning Examples with Known Optimal area. The study was subjected to examples containing up to 1000 modules. Quality assessment is the "quality level" – the value of F_{opt}/F, where F is the estimate of the solution obtained. Comparison with known algorithms has shown that, with a shorter working time, the deviation of the objective function from the optimal value obtained by the developed algorithm is less by an average of 6 %. The time complexity of this algorithm depends on the lifetime of the colony l (the number of iterations), the number of vertices of the graph n, and the number of ants m, and is defined as $O(l \cdot n^2 \cdot m)$.

Blocks; planning; tree of cuts; Polish expression; synthesis; alternative; paradigm; ant algorithm.

Введение. Задача планирования СБИС заключается в размещении на поле кристалла блоков, полученных на этапе разбиения, имеющих заданную площадь и не имеющих фиксированных размеров [1]. Блоки 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 не налагающиеся прямоугольники, в которых следует поместить соответствующие блоки [2].

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

Принято разбивать все множество представлений на два класса: гильотинный и не гильотинный. Гильотинная структура может быть получена путем рекурсивного деления прямоугольника на две части горизонтальными и/или вертикальными разрезами (рис. 1,а). Именно такая структура используется в классической постановке. Другой класс представлений – не гильотинный – реализуется с помощью последовательной пары [3], ограниченной разрезающей решетки [4], О-дерева [5], B*-дерева [6], списка угловых модулей [7], графа транзитивного замыкания [8], обобщенной польской записи [9, 10] и др. 2 Представление плана оказывает большое влияние на операции над модулями и сложность процесса проектирования. В общем случае, не гильотинное представление имеет большее пространство решений и позволяет добиться более компактного расположения модулей по сравнению с гильотинным планом. В качестве плана кристалла наиболее часто используют план, получаемый путем рекурсивного использования “гильотинного разреза” [1, 2, 11], т.е. последовательного разрезания прямоугольников на две части (рис. 1,а). В работе [10] показано, что решения гильотинного типа сравнимы по качеству с не гильотинными. Гильотинное представление имеет ряд преимуществ, такие как небольшие затраты на кодирование и меньшее пространство поиска, приводящие к более быстрому построению плана. Более того, оно гибкое при работе с фиксированными, предварительно размещенными, гибкими и прямолинейными модулями, но в пространстве решений гильотинной структуры могут отсутствовать оптимальные решения.

Основные критерии оптимизации [1, 2, 11]: площадь кристалла; длина проводников; временные задержки; энергопотребление; температура. Основной целью оптимизации является минимизация общей площади кристалла [1, 2].

Задача планирования относится к классу *NP*. В течение последних лет были предложены различные подходы к решению проблемы планирования. Эти подходы могут быть классифицированы следующим образом: линейное и квадратичное программирование [1, 2]; имитация отжига [12, 13]; основанные на ограничениях [14]; сило-направленная парадигма [1, 2]; основанные на геометрической дуализации списков связей [1, 2]; иерархические методы сверху-вниз и снизу-вверх [15–17]; метод кластеризации [18]; генетические алгоритмы (ГА) [19, 20]; эволюционные алгоритмы [21, 22]; на основе поисковой адаптации [23]; меметические [24] и др. Анализ существующих подходов к решению поставленной задачи показал, что удачными являются подходы, основанные на методах эволюционного моделирования. Тем не менее, в последнее время для решения различных «сложных» задач, к которым относятся и задачи планирования всё чаще используются способы, основанные на применении биоинспирированных моделей [25].

В последние годы интенсивно разрабатывается научное направление, объединяющее математические методы, в которых заложены принципы природных механизмов принятия решений.

Предлагаются новые подходы к решению задачи планирования СБИС, использующие математические методы, в которых заложены принципы природных механизмов принятия решений. В работе предлагается гибридный подход, спосо-

бы и методы представления задачи планирования СБИС в виде эволюционных процессов, основанных на интеграции моделей адаптивного поведения биологических систем, композитных архитектур нахождения решений, позволяющий работать с задачами большой размерности и получать качественные результаты за приемлемое время. Для представления решения задачи планирования наряду с модифицированной польской записью используются новые парадигмы. Это позволило создать пространство решений, в рамках которого организован поисковый процесс, базирующийся на моделировании адаптивного поведения муравьиной колонии.

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.

$$l_i \le h_i/w_i \le t_i, \quad (1)$$

где h_i – это высота модуля, w_i – ширина модуля. В качестве плана кристалла будем использовать план, полученный путем рекурсивного использования “гильотинного разреза”, т.е. последовательного разрезания прямоугольников на две части. На рис. 1 представлен план, а на рис. 2 – соответствующее ему дерево $DR=\{d_j | j=1,2,\ldots,2n-1\}$ “гильотинного разреза”, листьями которого являются вершины, соответствующие блокам, а внутренние вершины соответствуют разрезам: V – вертикальный, H – горизонтальный.

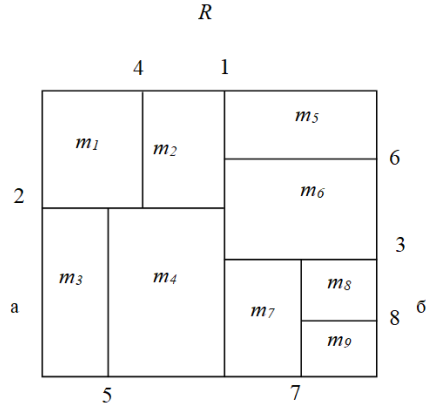

Рис. 1. План гильотинного разреза

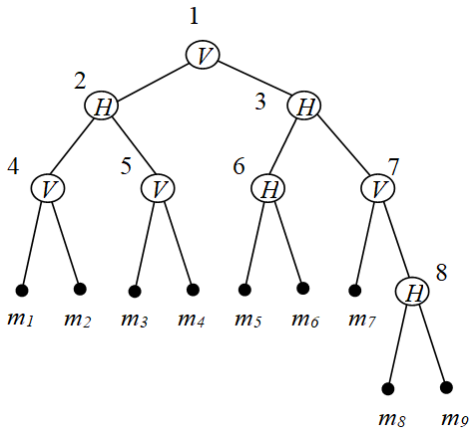

Рис. 2. Дерево гильотинного разреза

План для множества модулей M представляет собой прямоугольник R, разрезанный вертикальными и горизонтальными линиями на множество областей r_i, в каждую из которых помещается соответственно модуль m_i. На дереве цифрами помечены вершины, соответствующие разрезам, причем V – вертикальный разрез, а H – горизонтальный разрез. Буквами помечены вершины, соответствующие областям. Каждая область u_i, предназначенная для размещения модуля m_i, имеет размеры x_i и y_i. При соблюдении ограничений (1) размеры области должны также соответствовать ограничениям

$$S_i \le x_i \cdot y_i \,,\; h_i \le y_i \,,\; w_i \le x_i \,. \qquad (2)$$

Цель оптимизации – минимизация общей площади плана U, при соблюдении ограничений (1), (2).

Будем считать, что связи между модулями m_i и m_j связывают центры соответствующих областей r_i и r_j [22, 23]. 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:

$$F = \sum_{i=1}^{n} x_i \cdot y_i \; + \lambda \sum_{i,j=1}^{n} c_{ij} \cdot d_{ij} \,, \qquad (3)$$

при соблюдении ограничений (1), (2).

Константа λ, устанавливаемая пользователем, управляет относительной важностью общей площади и взвешенной длиной связей [1, 2].

Общая структура представления решений. В общем случае план формируется путем трансформации дерева разрезов с идентифицированными вершинами. Идентификатор внутренней вершины дерева, соответствующей разрезу, указывает тип разреза – H или V. Листья дерева имеют два идентификатора. Первый идентификатор указывает номер модуля. Второй идентификатор указывает ориентацию модуля.

На основе этой информации геометрическое представление и метризация плана осуществляется путем последовательной бинарной свертки областей по дереву разрезов, начиная от листьев дерева [2, 22].

Введём алфавит $A=\{M,TR\}$. Структуру дерева разрезов можно задать, используя на базе алфавита A польское выражение для бинарного дерева [22], где множество $M=\{m_i|i=1, 2, \ldots, n_x\}$ соответствует листьям дерева разрезов (областям), а множество $TR=\{\boldsymbol{H,V}\}$ – соответствует разрезам. Польское выражение для дерева с идентифицированными вершинами, представленного на рис. 2, имеет вид:

$$D=< m_1\ m_2\ \boldsymbol{V}\ m_3\ m_4\ \boldsymbol{V}\ \boldsymbol{H}\ m_5\ m_6\ \boldsymbol{H}\ m_7\ m_8\ m_9\ \boldsymbol{H}\ \boldsymbol{V}\ \boldsymbol{H}\ \boldsymbol{V} >.$$

Процесс восстановления дерева по польскому выражению достаточно прост [22, 23]. Последовательно слева направо просматривается польское выражение, и отыскиваются буквы $\boldsymbol{H}$ или $\boldsymbol{V}$, соответствующие разрезам. Каждый такой разрез объединяет два ближайших образованных на предыдущих шагах подграфа, расположенных в польской записи слева от буквы $\boldsymbol{H}$ или $\boldsymbol{V}$.

Проиллюстрируем процесс свертки с помощью скобок:

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

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 (Условия) польского выражения:

1. Для каждого модуля m_i возможны два способа (две ориентации) размещения в области u_i. Пусть первой ориентации модуля m_i соответствует обозначение – m^1_i, а при второй ориентации m^2_i.

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

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.

5. При просмотре выражения слева направо, первый элемент, соответствующий разрезам в польском выражении может появиться только после двух элементов, принадлежащих множеству *M*.

Назовем польское выражение *D*, построенное на базе алфавита $A=\{M,TR\}$ легитимным, если оно удовлетворяет вышеперечисленным условиям (1–5). Таким образом, легитимное выражение *D* является символьным представлением решения задачи планирования. Различные решения получаются путём комбинирования взаимным расположением элементов алфавита $A=\{M,TR\}$, удовлетворяющим условиям (1–5) и конкретной идентификации вершин дерева разрезов. В работе пространство решений представляется множеством легитимных выражений *D*. Поиск решения сводится к поиску легитимного выражения *D* с оптимальным значением показателя качества.

Синтез дерева разрезов с идентификацией вершин на основе модели адаптивного поведения муравьиной колонии. Общепринятый подход к решению задачи планирования алгоритмами, использующими гильотинную структуру для представления плана, предполагает формирование структуры дерева разрезов поисковыми методами, а разметку всех вершин дерева и выбор ориентации модулей осуществлять на основе механизмов генетической эволюции [5]. В работе рассматривается подход, при котором формирование структуры дерева разрезов муравьиным алгоритмом и разметка всех вершин дерева производится параллельно.

Для отражения коллективной эволюционной памяти в течение жизни популяции муравьев и для формирования решения задачи в работе используется полный граф $G=(X,U)$ с альтернативными состояниями вершин. Задача синтеза дерева разрезов с идентификацией вершин сводится к задаче формирования соответствующего польского выражения и вектора идентификаторов [22, 23].

Поиск решений осуществляется на полном графе поиска решений $G=(X,U)$, где $X=X_1 \cup X_2$. Вершины множества $X_1=\{x_{1i}|i=1,2,\ldots, n_M\}$ соответствуют множеству модулей $M=\{m_i|i=1,2,\ldots,n\}$. Каждая вершина $x_{1i} \in X_1$ может находиться в одном из двух альтернативных состояний (α или β), соответствующих типу ориентации модуля. Если модуль m_i размещается в первой ориентации, то x_{1i} находится в первом состоянии – (α). Если модуль m_i размещается во второй ориентации, то x_{1i} находится во втором состоянии – (β). Вершины множества $X_2=\{x_{2i}|i=1,2,\ldots, n_R\}$ соответствуют разрезам. Каждая вершина $x_{2i} \in X_1$ может находиться в одном из двух альтернативных состояний, соответствующих типу разреза. Если разрез горизонтальный (*H*), то x_{2i} находится в первом состоянии (α). Если разрез вертикальный (*V*), то x_{2i} находится во втором состоянии (β).

Если ориентация модулей зафиксирована, то в состав графа поиска решений $G=(X,U)$ вместо множества вершин X_{12} войдет X_1.

Задача формулируется как задача поиска минимального по стоимости маршрута на графе поиска решений $G=(X,U)$. Отличительная особенность заключается в том, что при построении маршрута одновременно с выбором внутренней вершины $x_i \in X$ осуществляется выбор состояния этой вершины. Обозначим как θ_i состояние вершины x_i. $\theta_i=\{\alpha, \beta\}$.

Другими словами, для каждой пары взаимоисключающих (альтернативных) состояний (значений) вершины в процессе прокладки маршрута выбирается один альтернативный вариант из двух возможных. Будем обозначать вершину x_i в первом состоянии как x^{α}_i, а во втором как x^{β}_i.

В общем случае поиск решения задачи планирования осуществляется коллективом муравьев $Z=\{z_k|k=1,2,\ldots,l\}$. На каждой итерации муравьиного алгоритма каждый муравей z_k строит свое конкретное решение задачи планирования. Решением является маршрут в графе $G=(X,U)$, включающий n_M вершин, соответствующих модулям, и n_R вершин, соответствующих разрезам, построенный в соответствии с условиями польского выражения (1–5). Отметим, что для каждой из вершин $x_{1i} \in X_1$ и $x_{2i} \in X_2$, входящих в маршрут, выбраны альтернативные состояния $\theta_i=\{\alpha, \beta\}$. В этом случае построенный маршрут представляется в виде легитимного выражения D.

Для равномерного распределения муравьев и создания равных стартовых условий в качестве начальных вершин у формируемых муравьями маршрутов используются вершины множества X_{12} общим числом n_M . Другими словами число решений формируемых муравьями на каждой итерации равно n_M.

Моделирование поведения муравьёв в задаче планирования связано с распределением феромона на ребрах и вершинах графа G. Для учета оценок состояний вершины x_i вводится два счетчика α_i и β_i. На начальном этапе на всех ребрах графа G и на всех счетчиках вершин откладывается одинаковое (начальное) количество феромона – ε. Параметр ε задается априори. Процесс поиска решений итерационный. Каждая итерация l включает три этапа. На первом этапе муравей находит решение, на втором этапе откладывает феромон, на третьем этапе осуществляется испарения феромона. В работе используется циклический (ant-cycle) метод муравьиных систем. В этом случае ферромон откладывается агентом на ребрах после полного формирования решений популяцией. На первом этапе каждой итерации каждый муравей z_k формирует свой собственный маршрут M_k. Процесс построения маршрута M_k пошаговый. На каждом шаге t агент применяет вероятностное правило выбора следующей вершины для включения ее формируемый маршрут $M_k(t)$. Для этого формируется множество вершин $X_k(t) \in X$, таких, что каждая из вершин $x_i \in X_k(t)$ может быть добавлена в формируемый маршрут $M_k(t)$ с соблюдением условий (1–5). Пусть $e_k(t)$ - последняя вершина маршрута $M_k(t)$. Агент просматривает все вершины $x_i \in X_k(t.)$

Для каждой вершины $x_i \in X_k(t)$ рассчитывается параметр f_{ik} – суммарный уровень феромона на ребре графа G, связывающего x_i с вершиной $e_k(t)$, а также суммарные уровни α_{ik} и β_{ik} феромона в счетчиках первого и второго состояний вершины x_i. Вероятность включения вершины $x_i \in X_k(t)$ с заданным состоянием θ_i в формируемый маршрут $D_k(t)$ определяется следующими соотношениями:

$$\begin{aligned} P^{\alpha}_{ik} &= (f_{ik}+\alpha_{ik}) / \textstyle\sum_i [(f_{ik}+\alpha_{ik})+(f_{ik}+\beta_{ik})], \text{ если } x_i \text{ в состоянии } \alpha\ (\theta_i=\alpha); \\ P^{\beta}_{ik} &= (f_{ik}+\beta_{ik}) / \textstyle\sum_i [(f_{ik}+\alpha_{ik})+(f_{ik}+\beta_{ik})], \text{ если } x_i \text{ в состоянии } \beta\ (\theta_i=\beta), \end{aligned} \quad (4)$$

где $i|\ x_i \in X_k(t)$.

В соответствии с рассчитанным с помощью соотношений (4) распределением вероятности, агент случайным образом выбирает одну из вершин $x_i \in X_k(t)$ в некотором состоянии θ_i, которая включается в маршрут $D_k(t)$. Построенный агентом z_k на итерации l маршрут D_k трансформируется в польское выражение. Польское выражение трансформируется в дерево разрезов с идентификацией вершин, которое в свою очередь трансформируется в план, для которого рассчитывается оценка $F_k(l)$. $F_k(l)$ рассматривается в качестве оценки маршрута D_k.

На втором этапе итерации, каждый муравей откладывает феромон на рёбрах и счетчиках вершин построенного маршрута. Если $x_i \in D_k(t)$ в первом состоянии ($\theta_i=\alpha$), то феромон откладывается на счетчике α_i , если $x_i \in D_k(t)$ во втором состоянии, то феромон откладывается на счетчике β_i.

Количество феромона $\tau_k(l)$, откладываемое муравьем z_k на каждом ребре и счетчике вершины, включенных в построенный маршрут D_k, определяется следующим образом:

$$\tau_k(l)= \delta\, Q / F_k(l), \quad (5)$$

где l – номер итерации, δ – коэффициент отложения, Q_i – базовое количество феромона, откладываемое муравьем на ребрах маршрута D_k, $F_k(l)$ – целевая функция для решения, полученного муравьем z_k на l-ой итерации. Чем меньше $F_k(l)$, тем больше феромона откладывается на ребрах и счетчиках вершин построенного маршрута и, следовательно, тем больше вероятность выбора этих ребер и состояний вершин при построении маршрутов на следующей итерации.

После того, как каждый агент сформировал решение и отложил феромон, на третьем этапе происходит общее испарение феромона на ребрах и счетчиках вершин полного графа G в соответствии с формулой (6).

$$f_{ik} = f_{ik}(1 - \rho), \quad (6)$$

где ρ – коэффициент обновления.

После выполнения всех действий на итерации находится агент с лучшим решением, которое запоминается. Далее осуществляется переход на следующую итерацию.

Временная сложность этого алгоритма зависит от времени жизни колонии l (число итераций), количества вершин графа n и числа муравьев m, и определяется как $O(l \cdot n^2 \cdot m)$.

Алгоритм планирования на основе метода муравьиной колонии формулируется следующим образом.

1. В соответствии с исходными данными формируется полный граф поиска решений G с альтернативными состояниями вершин.
2. Для каждой вершина графа поиска решений G формируются счетчики состояний α_i , β_i.
3. Задается число N_a агентов и вершин графа G, в которые они помещаются.
4. Задаются значения управляющих параметров:
   ε – начальная доза феромона,
   Q_i – базовая доза феромона,
   ρ – коэффициент обновления,
   N_l – число итераций.
5. На всех ребрах и счетчиках состояний вершин графа G откладывается начальное количество феромона ε.
6. l=1. (l – номер итерации).
7. k=1. (k – номер агента).
8. Агентом z_k строится на графе G легитимный маршрут $M_k(l)$.
9. Осуществляется трансформация маршрута $M_k(l)$ в представление плана в виде польской записи.
10. Представление плана в виде польской записи трансформируется в дерево разрезов $T_k(l)$.
11. Дерево разрезов $T_k(l)$ с помощью алгоритма свертки трансформируется в план $R_k(l)$ и находится значение целевой функции $F_k(l)$ плана, которая рассматривается как оценка маршрута $M_k(l)$.
12. Если $k<N_k$, то $k=k+1$ и переход к п. 8, иначе переход к пункту 13.

13. На ребрах и счетчиках вершин каждого найденного маршрута в графе G откладывается феромон. Если $x_i \in M_k(t)$в первом состоянии, то феромон откладывается на счетчике α_i, если $x_i \in D_k(t)$во втором состоянии, то феромон откладывается на счетчике β_i. Количество феромона $\tau_k(l)$, откладываемое муравьем z_k пропорционально $F_k(l)$.
14. Испарение феромона на ребрах и счетчиках вершин графа G.
15. Выбор лучшего решения, полученного на протяжении всех выполненных итераций.
16. Если все итерации выполнены, то конец работы алгоритма, в противном случае переход к пункту 5 для выполнения очередной итерации.

Отметим, что построенный агентом маршрут трансформируется в польское выражение, которое в свою очередь трансформируется в дерево разрезов с идентифицированными вершинами, а дерево разрезов трансформируется в план.

Временная сложность этого алгоритма зависит от времени жизни колонии l (число итераций), количества вершин графа n и числа муравьев m, и определяется как $O(l \cdot n^2 \cdot m)$.

Экспериментальные исследования. Для написания программы планирования СБИС методом муравьиной колонии **(ПМК)** был использован язык C++ в среде Microsoft Visual Studio 2010 для ОС Windows, так как главный упор делался на скорость работы приложения.

Тестирование проводилось на ЭВМ с процессором Intel Core 2 Duo T6600 2200 МГц, 4 Гб ОЗУ под управлением операционной системы Windows 7.

Целью экспериментальных исследований является выявление зависимостей и влияния на качество планирования и размещения различных комбинаций управляющих параметров и структур.

Эксперименты показали, что начальное количество феромона Ω должно быть в 10-12 раз больше среднего количества значения феромона $\tau_k(l)$, откладываемого муравьями на каждой итерации.

Для нахождения наилучшего сочетания таких параметров, как объем популяции M и количество итераций T, экспериментальные исследования проводились следующим образом.

Для проведения экспериментов программы **ПМК** была использована процедура синтеза контрольных примеров с известным оптимумом $F_{опт}$ по аналогии с известным методом AFEKO – Floorplanning Examples with Known Optimal area [26, 27]. Исследованию подвергались примеры, содержащие до 1000 модулей.

Оценкой качества служит ***«степень качества»*** – величина $F_{опт}/F$, где F – оценка полученного решения. На основе обработки экспериментальных исследований была построена средняя зависимость степени качества от числа итераций (рис. 3) и от размера популяции (рис. 4). Из графика видно, что в среднем на 130-ой итерации решение близко к оптимальному.

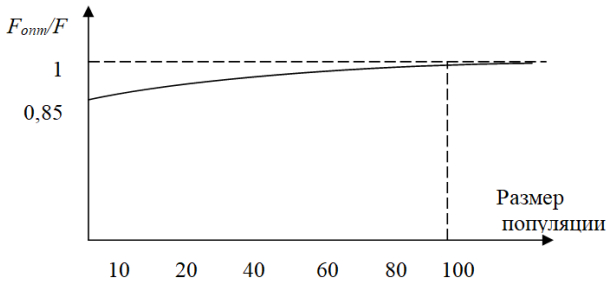

Рис. 3. Зависимость качества алгоритма планирования от размера популяции

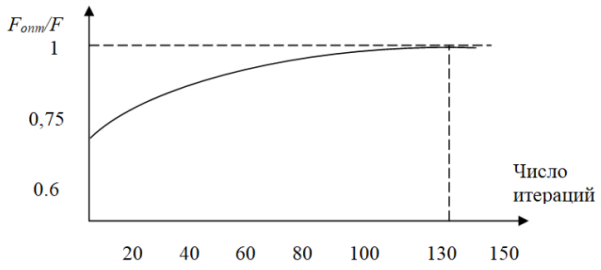

Рис. 4. Зависимость степени качества алгоритма планирования от числа итераций

Сравнительный анализ разработанных алгоритмов на основе ACO и T-ACO **с** другими алгоритмами планирования производился на стандартных тестовых примерах и схемах (бенчмарках) [28, 29]. Данные примеры представляют собой стандартные промышленные цепи (блоки, схемы, ИС, БИС, СБИС). Сравнение производилось по значению площади.

Для сравнения экспериментальных данных алгоритмов планирования были выбраны наиболее известные эволюционные алгоритмы: ГЕН – генетический [19]; ЭА – эволюционный [21]; ГА – гибридный [20]; МА – меметический [24]. Отметим, что результаты у сравниваемых алгоритмов получены на базе платформы Ultra1-Spark, а эксперименты разработанных алгоритмов проводились на ЭВМ типа IBM PC с процессором Pentium.

Представленная программа **ПМК** на основе парадигмы **ACO** находит решения, превосходящие результаты существующих алгоритмов по площади (табл. 1). Экспериментальная временная сложность представленных алгоритмов на одной итерации при фиксированных значениях управляющих параметров – $O(n)$, а временная сложность существующих алгоритмов составляет $O(n^2)$. Эксперименты показали, что при больших размерностях временные показатели разработанный алгоритм превосходит показатели сравниваемых алгоритмов при лучших значениях целевой функции.

Таблица 1

Результаты сравнения работы алгоритмов

Бенч марк	АЛГОРИТМ				
	ГЕН	ЭА	ГА	МА	**ПМК**
apte	49425136,4	49425136,4	49396354	49396354	49396354
xerox	24954567,8	24954567,8	24895479	24873723	24957366
hp	12232485	12232485	12194845	12176343	12393236
ami33	3530173,4	3510173,5	3538838	3520854	3490074
ami49	136133614,4	136133614,4	137343334	134285950	138591325

ГЕН – генетический; ЭА – эволюционный; ГА – гибридный; МА – меметический; **ПМК** – муравьиная колония

Сравнение с известными алгоритмами показало, что при меньшем времени работы у полученных с помощью разработанного алгоритма решений отклонение целевой функции от **оптимального** значения меньше в среднем на 6%.

Заключение. Задача планирования решается на основе модифицированной канонической парадигмы муравьиной колонии. Для отражения коллективной эволюционной памяти в течение жизни популяции муравьев и для формирования решения задачи в работе используется полный граф $G=(X,U)$ с альтернативными состояниями вершин.

Общепринятый подход к решению задачи планирования алгоритмами, использующими дерево разрезов для представления плана, предполагает формирование структуры дерева разрезов поисковыми методами, а разметку всех вершин дерева и выбор ориентации модулей осуществлять на основе механизмов генетической эволюции.

Ключевая проблема, которая была решена в данной работе, связана с разработкой композитной структуры пространства решений, позволяющей одновременно учитывать при построении дерева разрезов ориентацию элементов, тип разреза и метку модуля. Для отражения коллективной эволюционной памяти в течение жизни популяции муравьев и для формирования решения задачи в работе используется полный граф с альтернативными состояниями вершин. Разработана структура модифицированного польского выражения, позволяющая, учитывать ориентацию элементов, тип разреза и метку модуля.

Разработаны эвристики поведения муравья при перемещениях в графе поиска решений, позволяющие формировать легитимный маршрут. Это позволило снизить временную сложность задачи планирования и повысить качество решения.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Kahng A.B.* Classical Floorplanning Harmful // ISPD 2000. – P. 207-213.
2. *Lin C.-T. et al.* GPE: A New Representation for VLSI Floorplan Problem // IEEE International Conference on Computer Design (ICCD'02). – 2002. –P. 42-44.
3. *Sengupta D. et al.* Sequence pair based voltage island floorplanning // IEEE International Green Computing Conference. – 2011. – P. 1-6.
4. *Nakatake S. et al.* The channeled-BSG: a universal floorplan for simultaneous place/route with IC applications // International Conference on Computer-Aided Design, November 8-12, 1998. – P. 418-425.
5. *Guo P.N.* Floorplanning Using a Tree Representation // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – February 2001. – Vol. 20, No. 2. – P. 281-289.
6. *Nirmala R.D.G., Rajaram S.* Performance Driven VLSI Floorplanning with B*Tree Representation Using Differential Evolutionary Algorithm // Trends in Network and Communications in Computer and Information Science. – 2011. – Vol. 197, Part 2. – P. 445-456.
7. *Wang R. et al.* An Improved P-admissible Floorplan Representation Based on Corner Block List // Proceedings of the 2005 Asia and South Pacific Design Automation Conference. – 2005. – P. 1115-1118.
8. *Lin J.-M., Chang Y.-W.* TCG: A Transitive Closure Graph-Based Representation for Non-Slicing Floorplans // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. – February 2005. – Vol. 13, Issue 2. – P. 288-292.
9. Sassone T.P.G. and Lim S.K. A novel geometric algorithm for fast wire-optimized floorplanning // In Proc. Int. Conf. Comput. Aided Design. – 2003. – P. 74-80.
10. *Lai M., and Wong D.F.* Slicing Tree Is a Complete Floorplan Representation // In Proc. DATE. – 2001. – P. 228-232.
11. *Cheng L., Wong D.F.* Floorplan Design for Multi-million Gate FPGAs // DAC. – 2004. – P. 292-313.
12. *Fang J.-P. et al.* A Parallel Simulated Annealing Approach for Floorplanning in VLSI // Proceedings of the 9th International Conference on Algorithms and Architectures for Parallel Processing. – 2009. – P. 291-302.
13. *Qi L. et al.* Simulated annealing based thermal-aware floorplanning // International Conference on Electronics, Communications and Control (ICECC). – 2011. – P. 463-466.
14. *Young F.Y. et al.* Slicing floorplans with boundary constraints // IEEE Transactions on Computer-Aided Design. – 1999. – P. 1385-1389.
15. *Chen T.-C., Chang Y.-W., and Lin S.-C.* A new multilevel framework for large-scale interconnect-driven floorplanning // IEEE Trans. Comput.-Aided Design Integrat. Circuits Syst. – 2008. – Vol. 27, No. 2. – P. 286-294.
16. *Singhal L. and Bozorgzadeh E.* Multilayer floorplanning for reconfigurable designs // IET Comput. Digit. Tech. – 2007. – Vol. 1, No. 4. – P. 276-294.

17. *Pritha Banerjee, Megha Sangtani, and Susmita Sur-Kolay*. Floorplanning for Partially Reconfigurable FPGAs // IEEE Trans. Comput.-Aided Des. Integr. Circuits Sys. – 2011. – Vol. 30, No. 1. – P. 8-17.
18. *Chuan-Wen Chiang*. Ant Colony Optimization for VLSI Floorplanning with Clustering Constraints // Journal of the Chinese Institute of Industrial Engineers. – 2009. – Vol. 26, Issue 6. – P. 440-448.
19. *Lin C.-T. et al.* An efficient genetic algorithm for slicing floorplan area optimization // Proceedings of the International Symposium on Circuits and Systems. – 2002. – P. 879-882.
20. *Chen J., Zhu W.* A hybrid genetic algorithm for VLSI floorplanning // IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS). – 2010. – P. 128-132.
21. *Shanavas .I H. et al.* Evolutionary Algorithmical Approach for VLSI Floorplanning Problem // International Journal of Computer Theory and Engineering. – 2009. – Vol. 1, No. 4. – P. 461-464.
22. *Лебедев Б.К., Лебедев В.Б.* Планирование СБИС на основе эволюционной адаптации // Известия ТРТУ. – 2006. – № 9 (64). – С. 93-97.
23. *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.
24. *Tang, Maolin and Yao, Xin.* A memetic algorithm for VLSI floorplanning // IEEE Transactions On Systems, Man, And Cybernetics – Part B: Cybernetics. – 2007. – Vol. 37 (1).
25. *Лебедев Б.К., Лебедев О.Б.* Биоинспирированные методы планирования кристалла СБИС // МЭС-2014. VI Всероссийская научно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем - 2014»: Сборник трудов. – М.: ИППМ РАН, 2012. – С. 171-176.
26. *Alpert C.J.* In Proceeding of the International Symposium on Physical Design // The ISPD98 circuit benchmark suite. – 1998. – P. 85-90.
27. *Cong J., Romesis M. и Xie M.* UCLA Optimality Study Project. http://cadlab.cs.ucla.edu/ ~pubbench. 2004.
28. MCNC Electronic and Information Technologies (Online). Available: www.mcnc.org.
29. 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.

REFERENCES

1. *Kahng A.B.* Classical Floorplanning Harmful, *ISPD 2000*, pp. 207-213.
2. *Lin C.-T. et al.* GPE: A New Representation for VLSI Floorplan Problem, *IEEE International Conference on Computer Design (ICCD'02)*, 2002,Ppp 42-44.
3. *Sengupta D. et al.* Sequence pair based voltage island floorplanning, *IEEE International Green Computing Conference*, 2011, pp. 1-6.
4. *Nakatake S. et al.* The channeled-BSG: a universal floorplan for simultaneous place/route with IC applications, *International Conference on Computer-Aided Design, November 8-12, 1998*, pp. 418-425.
5. *Guo P.N.* Floorplanning Using a Tree Representation, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, February 2001, Vol. 20, No. 2, pp. 281-289.
6. *Nirmala R.D.G., Rajaram S.* Performance Driven VLSI Floorplanning with B*Tree Representation Using Differential Evolutionary Algorithm, *Trends in Network and Communications in Computer and Information Science*, 2011, Vol. 197, Part 2, pp. 445-456.
7. *Wang R. et al.* An Improved P-admissible Floorplan Representation Based on Corner Block List, *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, 2005, pp. 1115-1118.
8. *Lin J.-M., Chang Y.-W.* TCG: A Transitive Closure Graph-Based Representation for Non-Slicing Floorplans, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, February 2005, Vol. 13, Issue 2, pp. 288-292.
9. *Sassone T.P.G. and Lim S.K.* A novel geometric algorithm for fast wire-optimized floorplanning, *In Proc. Int. Conf. Comput. Aided Design*, 2003, pp. 74-80.
10. *Lai M., and Wong D.F.* Slicing Tree Is a Complete Floorplan Representation, *In Proc. DATE*, 2001, pp. 228-232.
11. *Cheng L., Wong D.F.* Floorplan Design for Multi-million Gate FPGAs, *DAC*, 2004, pp. 292-313.

12. *Fang J.-P. et al.* A Parallel Simulated Annealing Approach for Floorplanning in VLSI, *Proceedings of the 9th International Conference on Algorithms and Architectures for Parallel Processing*, 2009, pp. 291-302.
13. *Qi L. et al.* Simulated annealing based thermal-aware floorplanning, *International Conference on Electronics, Communications and Control (ICECC)*, 2011, pp. 463-466.
14. *Young F.Y. et al.* Slicing floorplans with boundary constraints, *IEEE Transactions on Computer-Aided Design*, 1999, pp. 1385-1389.
15. *Chen T.-C., Chang Y.-W., and Lin S.-C.* A new multilevel framework for large-scale interconnect-driven floorplanning, *IEEE Trans. Comput.-Aided Design Integrat. Circuits Syst*., 2008, Vol. 27, No. 2, pp. 286-294.
16. *Singhal L. and Bozorgzadeh E.* Multilayer floorplanning for reconfigurable designs, *IET Comput. Digit. Tech.*, 2007, Vol. 1, No. 4, pp. 276-294.
17. *Pritha Banerjee, Megha Sangtani, and Susmita Sur-Kolay*. Floorplanning for Partially Reconfigurable FPGAs, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst*., 2011, Vol. 30, No. 1, pp. 8-17.
18. *Chuan-Wen Chiang*. Ant Colony Optimization for VLSI Floorplanning with Clustering Constraints, *Journal of the Chinese Institute of Industrial Engineers*, 2009, Vol. 26, Issue 6, pp. 440-448.
19. *Lin C.-T. et al.* An efficient genetic algorithm for slicing floorplan area optimization, *Proceedings of the International Symposium on Circuits and Systems*, 2002, pp. 879-882.
20. *Chen J., Zhu W.* A hybrid genetic algorithm for VLSI floorplanning, *IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, 2010, pp. 128-132.
21. *Shanavas .I H. et al.* Evolutionary Algorithmical Approach for VLSI Floorplanning Problem, *International Journal of Computer Theory and Engineering*, 2009, Vol. 1, No. 4, pp. 461-464.
22. *Lebedev B.K., Lebedev V.B.* Planirovanie SBIS na osnove evolyutsionnoy adaptatsii [Planning VLSI on the basis of evolutionary adaptation], *Izvestiya TRTU* [Izvestiya TSURE], 2006, No. 9 (64), pp. 93-97.
23. *Kureychik V.M., Lebedev B.K., Lebedev V.B.* Planirovanie sverkhbol'shikh integral'nykh skhem na osnove integratsii modeley adaptivnogo poiska [Planning of super large integral schemes based on integration of adaptive search models], *Izvestiya RAN. Teoriya i sistemy upravleniya* [Izvestiya RAN. Theory and Control Systems], 2013, No. 1, pp. 84-101.
24. *Tang, Maolin and Yao, Xin.* A memetic algorithm for VLSI floorplanning, *IEEE Transactions On Systems, Man, And Cybernetics – Part B: Cybernetics*, 2007, Vol. 37 (1).
25. *Lebedev B.K., Lebedev O.B.* Bioinspirirovannye metody planirovaniya kristalla SBIS [A memetic algorithm for VLSI floorplanning], *MES-2014. VI Vserossiyskaya nauchno-tekhnicheskaya konferentsiya «Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh sistem - 2014»: Sbornik trudov* [MES-2014. VI All-Russia scientific and technical conference "Problems of development of promising micro- and nanoelectronic systems - 2014". Collection of works]. Moscow: IPPM RAN, 2012, pp. 171-176.
26. *Alpert C.J.* In Proceeding of the International Symposium on Physical Design, *The ISPD98 circuit benchmark suite*, 1998, pp. 85-90.
27. *Cong J., Romesis M. u Xie M.* UCLA Optimality Study Project. Available at: http://cadlab.cs.ucla.edu/~pubbench. 2004.
28. MCNC Electronic and Information Technologies (Online). Available at: www.mcnc.org.
29. hMetis [Online]. Available at: http://www-users.cs.umn.edu/karypis/metis/ hmet300. HB Floorplan Benchmarks [Online]. Available at: http://cadlab.cs.ucla.edu/ cpmo/HBsuite.html.

Статью рекомендовала к опубликованию д.т.н., профессор А.Г. Коробейников.

Лебедев Борис Константинович – Южный федеральный университет; e-mail: lebedev.b.k@gmail.com; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 89282897933; кафедра систем автоматизированного проектирования; профессор.

Лебедев Олег Борисович – e –mail: lebedev.ob@mail.ru; тел.: 89085135512; кафедра систем автоматизированного проектирования; доцент.

Лебедева Елена Михайловна – e-mail: lebedev.ob@mail.ru; тел.: 89081702418; кафедра систем автоматизированного проектирования; аспирант.

Lebedev Boris Konstantinovich – Southern Federal University; e-mail: lebedev.b.k@gmail.com; 44, Nekrasovsky, Taganrog, 347928, Russia; phone: +79282897933; the department of computer aided design; professor.

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

Lebedeva Elena Mikhaylovna – e-mail: lebedev.ob@mail.ru; phone: +79081702418; the department of computer aided design; graduate student.

УДК 004.896

Э.В. Кулиев, С.Н. Щеглов, Е.А. Пантелюк, Н.В. Кулиева

АДАПТИВНЫЙ АЛГОРИТМ СТАИ СЕРЫХ ВОЛКОВ ДЛЯ РЕШЕНИЯ ЗАДАЧ ПРОЕКТИРОВАНИЯ*

Данная статья связана с решением одной из ключевых задач этапа автоматизированного конструкторского проектирования – размещения компонентов сверхбольших интегральных схем. В последнее время началось исследование возможностей применения и разработка алгоритмов, инспирированных природными системами, для эффективного принятия решения в задачах САПР. При этом постоянно возникает конфликт между сложностью САПР и требованиями принятия эффективных решений в реальном масштабе времени. Данные проблемы не могут быть полностью решены распараллеливанием процесса принятия решений, увеличением числа операторов, пользователей и т.д. Одним из возможных подходов к решению этой проблемы является использование новых технологий на стыке информатики, бионики и автоматизации проектирования. В этой связи разработка новых принципов и подходов принятия эффективных решений в задачах проектирования и управления имеет важное экономико-социальное значение и является, в настоящее время, актуальной и важной. В статье описывается алгоритм живой природы, который основывается на примере стаи серых волков. Приведена постановка задачи размещения элементов схем ЭВА на множестве заданных позиций дискретного рабочего поля. Представлена модифицированная технология разработки инспирированных природой алгоритмов. Показаны основные шаги работы алгоритма поведения стаи серых волков применительно к задаче размещения. Приведены сравнительные результаты вычислительных экспериментов. Основной целью исследования является оценка возможности применения интегрированных методов, инспирированных природными системами, для решения задач конструкторского проектирования САПР на примере использования алгоритма поведения стаи серых волков в живой природе.

Ноевой алгоритм; генетический алгоритм; целевая функция; окрестность; стая серых волков.

E.V. Kuliev, S.N. Sheglov, E.A. Pantelyuk, N.V. Kulieva

ADAPTIVE ALGORITHM OF THE PACK OF GREY WOLVES FOR SOLVING DESIGN OBJECTIVES

This article is related to the solution of one of the key tasks of the automated design stage – a placement of components of super-large integrated circuits. Recently, started have been a study of application possibilities and the development of algorithms inspired by natural systems for effective decision making in CAD tasks. At the same time, there is a constant conflict between the

* Работа выполнена при финансовой поддержке РФФИ (проект № 15–07–06415) и внутреннего гранта ЮФУ по теме №2.6432.2017/БЧ.