УДК 658.512

Б.К. Лебедев, В.Б. Лебедев

ПЛАНИРОВАНИЕ НА ОСНОВЕ РОЕВОГО ИНТЕЛЛЕКТА И ГЕНЕТИЧЕСКОЙ ЭВОЛЮЦИИ*

В работе излагается метод решения задачи планирования на основе роевого интеллекта и генетической эволюции. Связующим звеном такого подхода является структура данных, описывающая в виде хромосомы решение задачи. Предложена композитная архитектура многоагентной системы бионического поиска. Рассмотрены три подхода к построению такой архитектуры.

Планирование; роевой интеллект; генетическая эволюция; хромосома; многоагентная система; бионический поиск.

B.K. Lebedev, V.B. Lebedev

PLANNING ON A BASIS SWARM INTELLIGENCE AND GENETIC EVOLUTION

In work the method of the decision of a problem of planning on a basis Swarm intelligence and genetic evolution is stated. A link of such approach is the structure of data describing in the form of a chromosome the decision of a problem. The composit architecture of multi-agent systems of bionic search is offered. Three approaches to construction of such architecture are considered.

Planning; Swarm intelligence; genetic evolution; chromosome; multi-agent systems; bionic search.

Введение. Планирование СБИС заключается в размещении на поле кристалла блоков, имеющих форму прямоугольников [1]. В результате планирования строится план кристалла, представляющий собой охватывающий прямоугольник, разделенный горизонтальными и вертикальными сегментами на непересекающиеся прямоугольники, в которые следует поместить соответствующие блоки. Основной целью оптимизации является минимизация общей площади кристалла.

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 [1,2]: линейное и квадратичное программирование [3]; имитация отжига; основанные на ограничениях [4]; сила направленная парадигма [5]; основанные на геометрической дуализации списков связей [5]; иерархические методы сверху-вниз и снизу-вверх [6]; метод кластеризации [7]; генетические алгоритмы (ГА) и на основе поисковой адаптации [8,9] и др. Анализ существующих подходов к решению поставленной задачи показал, что удачными являются подходы, основанные на методах эволюционного моделирования [10,11]. Тем не менее, в последнее время для решения различных «сложных» задач, к которым относятся и задачи планирования, всё чаще используются способы, основанные на применении методов искусственного интеллекта [13]. Особенно наблюдается стремительный рост интереса к разработке алгоритмов, инспирированных природными системами [14]. В основе большинства этих алгоритмов лежат идеи, заимствованные в природе, а также базовые постулаты

* Работа выполнена при поддержке: РФФИ (грант № 09-01-00509), г/б № 2.1.2.1652.

универсальности и фундаментальности, присущие самоорганизации природных систем.

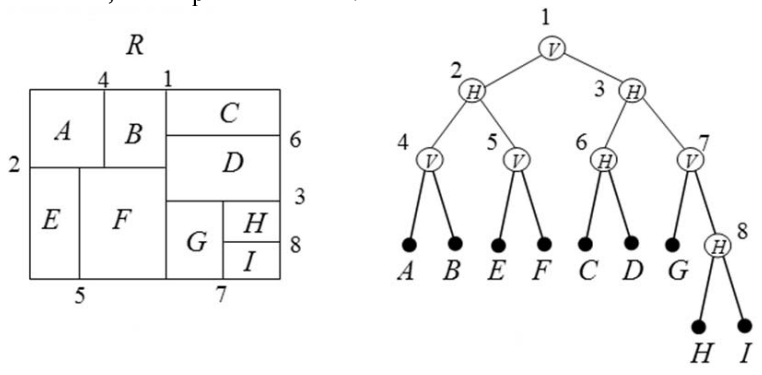
Одним из новых направлений таких методов являются мультиагентные методы интеллектуальной оптимизации, базирующиеся на моделировании коллективного интеллекта [15,16]. Коллективная система способна решать сложные динамические задачи по выполнению совместной работы, которая не могла бы выполняться каждым элементом системы в отдельности в разнообразных средах без внешнего управления, контроля или координации. В таких случаях говорят о роевом интеллекте (Swarm intelligence), как о замысловатых способах кооперативного поведения, то есть стратегии выживания. Оптимизация с использованием роя частиц (Particle Swarm Optimization, PSO) – это метод поиска, который базируется на понятии популяции, и моделирует поведение птиц в стае и косяков рыб [17,18]. Рой частиц может рассматриваться как многоагентная система, в которой каждый агент (частица) функционирует автономно по очень простым правилам. В противовес почти примитивному поведению агентов, поведение всей системы получается на удивление разумным.

В работе излагается метод решения задачи планирования на основе роевого интеллекта и генетической эволюции [19]. Предложена композитная архитектура многоагентной системы бионического поиска.

Основные положения. Проблема планирования формулируется следующим образом [1]. 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.

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

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,а представлен план, а на рис. 1,б – соответствующее ему дерево $D=\{d_j \mid j = 1,2,\ldots,2n-1\}$ “гильотинного разреза”, листьями которого являются вершины, соответствующие блокам, а внутренние вершины соответствуют разрезам: V – вертикальный, H – горизонтальный.



а б

Рис. 1. Гильотинный разрез: а – план; б – дерево

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 r_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 R, при соблюдении ограничений (1), (2).

Формирование плана методом свертки. Задать план это: задать структуру дерева разрезов, т.е. последовательность бинарных разрезов; для внутренних вершин дерева, соответствующих разрезам, указать тип разреза H или V; пометить листья дерева номерами модулей; для модулей с фиксированными размерами указать их ориентацию.

На основе этой информации построение плана осуществляется путем последовательной бинарной свертки областей по дереву разрезов, начиная от листьев дерева. Каждой внутренней вершине дерева разрезов будет соответствовать область, полученная в результате бинарной свертки поддерева, имеющего корнем эту внутреннюю вершину. Будем считать, что разрез с номером i, разрезает вершину d_i (область u_i). Вначале свертки каждой вершине d_i, являющейся листом дерева разрезов, ставится в соответствие область r_i с размерами $x_i = h_i$, $y_i = w_i$, равными размерам модуля m_i. Пусть вершины d_i и d_j являются дочерними вершинами вершины d_k и пусть для областей u_i и u_j, соответствующих d_i и d_j, определены нижние границы их размеров (x_i, y_i), (x_j, y_j).

Процесс бинарной свертки представляет собой слияние областей u_i и u_j, формирование области u_k, определение размеров для u_k и новых размеров для u_i и u_j.

Общая структура представления решений в алгоритмах разбиения на основе роевого интеллекта и генетического поиска. В эвристических алгоритмах роевого интеллекта многомерное пространство поиска населяется роем частиц [17]. Каждая частица представляет некоторое решение. В нашем случае решение задачи планирования. Процесс поиска решений заключается в последовательном перемещении частиц в пространство поиска. Обозначим позицию частицы i в пространстве решений в момент времени t (t имеет дискретные значения) как $x_i(t)$. По аналогии с эволюционными стратегиями, рой можно трактовать как популяцию, а частицу как индивида (хромосому). Это дает возможность построения гибридной структуры поиска решения, основанную на сочетании генетического поиска с методами роевого интеллекта. Связующим звеном такого подхода является структура данных, описывающая в виде хромосомы решение задачи. Если в качестве частицы используется хромосома, то число параметров, определяющих положение частицы в пространстве решений должно быть равно числу генов в хромосоме. Значение каждого гена откладывается на соответствующей оси пространства решений. В этом случае возникают некоторые требования к структуре хромосомы и значениям генов. Значения генов должны быть независимыми друг от друга, то есть хромосомы должны быть гомологичными. В работе предлагается подход к построению структур и принципов кодирования хромосом, обеспечивающих их гомологичность и возможность одновременного использования в генетическом алгоритме и в алгоритме на основе роя частиц.

Решение кодируется набором R из трех хромосом $R=\{H1,H2,H3\}$. Хромосома $H1$ несёт информацию о разметке множества вершин M. Хромосома $H2$ содержит

информацию о структуре дерева. Хромосома *H3* содержит информацию о типах разрезов (*H* или V).

Пусть *n* – число областей плана (число вершин множества *E*). Хромосома *H1*, задающая разметку множества вершин *M*, имеет вид $H1 = \{g1_i \mid i = 1, 2, .., n-1\}$. Каждый ген $g1_i$ может принимать любое значение в интервале от *1* до $(n+1-i)$.

Например: для $n = 8;\ 1 \le g1_1 \le 8;\ 1 \le g1_2 \le 7;\ 1 \le g1_3 \le 6;\ldots;\ 1 \le g1_7 \le 2$.

Декодирование хромосомы *H1* производится с использованием опорного вектора $B^1 = < b^1_i \mid i = 1, 2, .., n >$ число элементов которого равно *n*, а их значения лежат в интервале от *1* до *n*. Пусть для *n=8* имеется хромосома *H1=<3,5,3,4,4,2,2>*, и пусть имеется опорный вектор $B^1 = <1,2,3,4,5,6,7,8>$.

Рассматриваем по порядку гены хромосомы и в соответствии с их значениями выбираем элементы в опорном векторе и записываем их в порядке выборки в вектор *Q*.

Значение $g1_1 = 3$. Выбираем в B^1 элемент b^1_j $(j=g1_1=3,\ b^1_3=3)$ и записываем его на первое место формируемого вектора *Q*, т.е. $q_1 = b^1_3 = 3$.

Удаляем элемент b^1_3 из B^1 и получаем вектор $B^2=<1,2,4,5,6,7,8>$, содержащий 7 элементов. Следующим выбирается $g1_2$, $g1_2= 5$. Отыскиваем элемент b^2_5 вектора B^2, $b^2_5 = 6$. Следовательно, $q_2 = 6$. Удаляем из B^2 элемент b^2_5, получаем вектор $B^3 = <1,2,4,5,7,8>$. Далее:

$g1_3 = 3$	$b^3_3 = 4$	$q_3 = 4$	$B^4 = < 1,2,5,7,8 >$
$g1_4 = 4$	$b^4_4 = 7$	$q_4 = 7$	$B^5 = < 1,2,5,8 >$
$g1_5 = 4$	$b^5_4 = 8$	$q_5 = 8$	$B^6 = < 1,2,5 >$
$g1_6 = 2$	$b^6_2 = 2$	$q_6 = 2$	$B^7 = < 1,5 >$
$g1_7 = 5$	$b^7_2 = 5$	$q_7 = 5$	$B^8 = < 1 >$

$q_8 = b^8_1 = 1$. В итоге получаем вектор $Q = <3,6,4,7,8,2,5,1>$, задающий разметку множества вершин *M*.

Рассмотрим структуру хромосомы *H2*. Введём алфавит $A = \{X, \bullet\}$. Структуру дерева разрезов можно задать, используя на базе алфавита *A* польское выражение для бинарного дерева, где знак *X* соответствует листьям дерева разрезов (областям), а знак • – соответствует внутренним вершинам дерева (разрезам). Польское выражения для дерева, представленного на рис. 1,b, имеет вид: $X\ X \bullet X\ X \bullet \bullet X\ X \bullet X\ X\ X \bullet \bullet \bullet \bullet$.

Процесс восстановления дерева по польскому выражению достаточно прост. Последовательно, слева направо, просматривается польское выражение и отыскиваются буквы типа •, соответствующие разрезам. Каждый такой разрез объединяет два ближайших образованных на предыдущих шагах подграфа, расположенных в польской записи слева от знака •. Проиллюстрируем процесс свертки с помощью скобок: $((X\ X \bullet)(X\ X \bullet) \bullet)((X\ X \bullet)\ (X\ (X\ X \bullet) \bullet) \bullet) \bullet$.

Отметим основные свойства польского выражения, для выполнения которых необходимо, чтобы записи соответствовало дерево разрезов. Обозначим через n_x – число элементов польского выражения типа *X*, а через $n_\bullet$ – число элементов типа •. Для дерева разрезов всегда выполняется равенство $n_x = n_\bullet + 1$. Если в польском выражении провести справа от знака • сечение, то слева от сечения число знаков *X* больше числа знаков •, по крайней мере, на единицу. Первый знак • в польском выражении (при просмотре слева направо) может появиться только после двух знаков *X*. Пронумеруем позиции между знаками *X*:

$$XX^{1}X^{2}X^{3}X^{4}\ldots X^{Nx-1}.$$

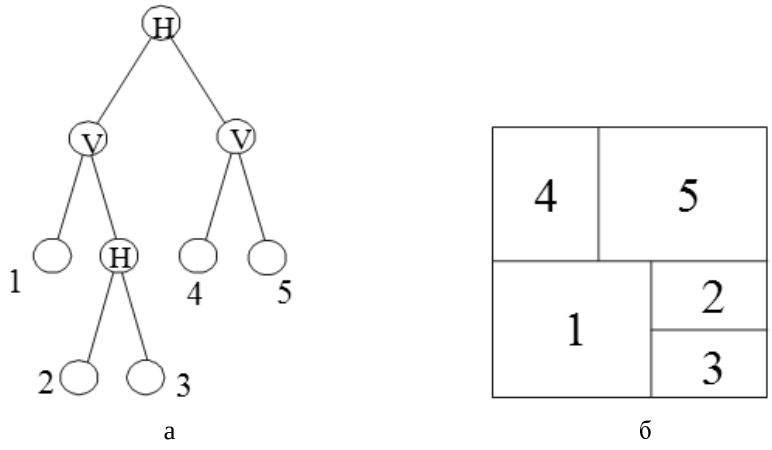
Максимальное число знаков •, которое может появиться в позиции, равно номеру позиции. Если польское выражение соответствует выше перечисленным свойствам, то ему соответствует дерево разрезов.

Хромосома *H2* имеет вид: $H2=\{g2_i|i=1,2,..,n_\bullet\}$. В результате декодирования хромосомы строится польское выражение. Число генов в хромосоме равно $n_\bullet$, т.е. числу знаков •. Значение гена $g2_i$ колеблется в пределах от i до $n_\bullet$, т.е. $i \le g2_i \le n_\bullet$. Значение гена указывает номер позиции в польском выражении, в которую необходимо поместить знак •.

Например, пусть для $n_\bullet = 4$ имеется хромосома $H2 = < 4,2,2,4 >$. Польское выражение, соответствующее этой хромосоме, имеет вид: $XX\ X \bullet\ \bullet X\ X \bullet\ \bullet$.

С помощью хромосомы $H3 = \{g3_i \mid i = 1, 2, .., n_\bullet\}$ задаются типы разрезов (*H* или *V*). Число генов *H3* равно $n_\bullet$. Значением гена является *0* или *1*, при этом *0* – соответствует *H*, а *1 – V*. Разметка внутренних вершин (определение типа разрезов) осуществляется последовательно в порядке расположения знаков • в польском выражении. Например, *H3 = <1,0,0,1>*. *H1 = <1,1,1,1,1>*. Соответствующий *H1* вектор *Q* имеет вид: *Q = <1,2,3,4,5 >*.

Пусть *H1 = <2,1,3,1>*, B^1 *= <2,1,4,5,3>*, *H2 = <4,2,2,4 >* и *H3 = <1,0,0,1 >*. Соответствующий хромосоме *H1* вектор *Q* имеет вид: *Q = <1,2,3,4,5 >*. Польское выражение, соответствующее *H2* имеет вид *X X X •• X X ••*. Модифицированное с учётом *H3* польское выражение примет вид: *X X X H V X X V H*. Дерево разрезов имеет вид, представленный на рис. 2,а, а план имеет вид, представленный на рис. 2,б.



а б

Рис. 2. Пример: а – дерево разрезов, б – план

Размеры областей и описывающего прямоугольника плана определяются путем последовательной свертки областей по дереву разрезов, исходя из размеров модулей, помещаемых в эти области.

Процесс декодирования (построения фенотипа) по хромосомам *H1, H2, H3* заключается в построении вектора *Q* по хромосоме *H1* и модифицированного польского выражения по *H2, H3*. Каждая из перечисленных процедур имеет оценку трудоемкости *O(n)*, где *n* – число модулей. Таким образом, общая оценка трудоемкости равна *O(n)*. Пространственная сложность для одного решения также имеет оценку *O(n)*.

Алгоритм планирования на основе роевого интеллекта. Основу поведения роя частиц составляет самоорганизация, обеспечивающая достижение общих целей роя на основе низкоуровневого взаимодействия. Каждая частица связана со всем роем, может взаимодействовать со всем роем и она тяготеет к лучшему решению роя. Процесс поиска решений итерационный. На каждой итерации каждая частица перемещается в новую позицию. Новая позиция определяется как

$$x_i(t+1)= x_i(t)+ v_i(t+1),$$

где $v_i(t+1)$ – скорость перемещения частицы из позиции $x_i(t)$ в позицию $x_i(t+1)$. Начальное состояние определяется, как $x_i(0)$, $v_i(0)$. Приведенная формула представлена в векторной форме. Для отдельного измерения j пространства поиска формула примет вид:

$$x_{ij}(t+1)= x_{ij}(t)+ v_{ij}(t+1), \quad (3)$$

где $x_{ij}(t)$ – позиция частицы i в измерении j; $v_{ij}(t+1)$ – скорость частицы i в измерении j.

Введем обозначения:

$f_i(t)$ – текущее значение целевой функции частицы i в позиции $x_i(t)$;

$x^*_i(t)$ – лучшая позиция частицы i, которую она посещала с начала первой итерации, а $f^*_i(t)$ – значение целевой функции в этой позиции (лучшее значение частицы i);

$F(t)$ – лучшее значение целевой функции среди частиц роя в момент времени t, а $x(t)$ – позиция с этим значением.

Тогда скорость частицы i на шаге $(t+1)$ в измерении j вычисляется как

$$v_{ij}(t+1)=w\cdot v_{ij}(t)+ k_1\cdot rnd(0,1)\cdot(x^*_{ij}(t)- x_{ij}(t)) + k_2\cdot rnd(0,1)\cdot(x_j(t) - x_{ij}(t)), \quad (4)$$

где $rnd(0,1)$ – случайное число на интервале (0,1); (w,k_1, k_2) – некоторые коэффициенты. Формула для расчета скорости составлена из трех компонентов.

Предыдущая скорость $v_{ij}(t)$ выступает в роли памяти частицы об ее перемещениях в прошлом и является инерционным компонентом

Значение второго компонента, называемого когнитивным, прямо пропорционально текущему расстоянию частицы от ее наилучшей позиции, которая была найдена с момента старта ее жизненного цикла. Когнитивный компонент выступает в роли индивидуальной памяти о наиболее оптимальных позициях данной частицы.

Значение третьего компонента, называемого социальным, прямо пропорционально текущему расстоянию частицы от наилучшей позиции роя в момент времени t. Благодаря социальному компоненту частица имеет возможность передвигаться в оптимальные позиции, найденные соседними частицами. В нашем случае позиция $x_i(t)$ задается с помощью хромосом $H1$, $H2$, $H3$, структура которых рассмотрена выше и объединенных в одну хромосому $H_i(t) = H1 \cup H2 \cup H3$. Число осей пространства поиска равно числу генов в объединенной хромосоме $H_i(t)$. Отметим, что скорость $v_i(t)$ имеет ту же структуру, что и хромосома $H_i(t)$. Позиция $x_i(t)$ то есть хромосома $H_i(t)$ является решением, а скорость $v_i(t+1)$ рассматривается как средство изменения хромосомы, то есть решения.

Отличительной особенностью позиции $x_i(t) = H_i(t)$ является то, что возможные значения гена g_{ij} зависят от локуса и меняются в интервале $\alpha_j \le g_{ij} \le \beta_j$. Обозначим через $xc_i(t)$ и $vc_i(t)$ позицию и скорость для которых соблюдаются указанные выше ограничения. Рассмотрим методику получения $xc_i(t)$ и $vc_i(t)$. Расчет выполняется в два этапа. Вначале рассчитывается $v_{ij}(t+1)$.

$$v_{ij}(t+1)=w\cdot vc_{ij}(t)+ k_1\cdot rnd(0,1)\cdot(xc^*_{ij}(t)-xc_{ij}(t)) + k_2\cdot rnd(0,1)\cdot(xc_j(t)-xc_{ij}(t)). \quad (5)$$

Затем по $v_{ij}(t+1)$ рассчитывается $vc_{ij}(t+1)$.

$$vc_{ij}(t+1)=\begin{cases}1, \text{ если } 0{,}5\le vc_{ij}(t+1);\\ 0, \text{ если } -0{,}5<vc_{ij}(t+1)<0{,}5;\\ -1, \text{ если } vc_{ij}(t+1)\le -0{,}5.\end{cases} \quad (6)$$

Новая позиция вычисляется следующим образом.

Вначале рассчитывается

$$x_{ij}(t+1)= xc_{ij}(t)+ vc_{ij}(t+1). \quad (7)$$

Затем определяется позиция с целочисленным значением, заключенным в заданном диапазоне:

$$xc_{ij}(t+1)=\begin{cases}\beta_j, \text{ если } \beta_j\le xc_{ij}(t+1);\\ x_{ij}(t+1), \text{ если } \alpha_j \le x_{ij}(t+1)\le\beta_j.;\\ \alpha_j, \text{ если } xc_{ij}(t+1)\le\alpha_j.\end{cases} \quad (8)$$

Схема работы роевого алгоритма планирования включает следующие шаги:

1. В соответствии с постановкой задачи разбиения и исходными данными формируется структура данных частицы (хромосома) и устанавливаются диапазоны значений для каждого измерения (оси) пространства поиска.

2. Создается исходная «случайная» популяция частиц, $t=0$. Для каждой частицы случайным образом задается начальная позиция $xc_i(0)$ и начальная скорость перемещения $vc_i(0)$. С этой целью в каждой формируемой хромосоме, соответствующей позиции $xc_i(0)$, каждому гену, лежащему в локусе j, случайным образом присваивается целочисленное значение, лежащее в допустимом диапазоне. Генам хромосомы, соответствующей скорости перемещения $vc_i(0)$ задаются сравнительно малые значения.

3. $t= t+1$.

4. Рассчитывается целевая функция $f_i(t)$ для каждой частицы.

5. Для каждой частицы определяются лучшая позиции $xc^*_i(t+1)$, которую она посещала с начала первой итерации, и значение целевой функции $f^*_i(t+1)$ в этой позиции.

7. Определяются лучшая позиция роя на шаге t и значение целевой функции $F(t)$ в этой позиции.

8. Лучшие частицы с точки зрения целевой функции объявляется «центром притяжения». Векторы скоростей всех частиц устремляются к этим центрам. Чем дальше частица находится от центра, тем большим ускорением она обладает. По формулам (5,6) для всех частиц рассчитываются скорости приращения.

9. Рассчитываются новые позиции частиц в пространстве решений.

10. Шаги 3-9 итерационно повторяются заданное число раз.

11. Последний «центр тяжести» соответствует найденному локальному оптимуму.

Таким образом, согласно алгоритму роя после случайной инициализации популяции частиц для каждой из них вычисляется значение целевой функции $f_i(t+1)$. Если оно окажется лучше, чем $f^*_i(t)$, то $f^*_i(t+1)= f_i(t+1)$, в противном случае, $f^*_i(t+1)= f^*_i(t)$. Далее, среди $f_i(t+1)$ выбирается лучшее значение $F(t)$, а затем вычисляются, согласно приведенным выше формулам, новые значения скоростей частиц и их новые позиции в пространстве решений. Итерационный процесс повторяется.

Отметим, что формула (3) фактически является оператором (назовем его роевым), с помощью которого изменяется текущее решение.

Гибридизация роевого интеллекта с генетическим поиском. Предлагается композитная архитектура многоагентной системы бионического поиска для решения задачи планирования на основе роевого интеллекта и генетической эволюции [19]. Рассмотрены три подхода к построению такой архитектуры.

Первый и наиболее простой подход к гибридизации заключается в следующем. С начала поиск решения осуществляется генетическим алгоритмом [10]. Затем на основе популяции, полученной на последней итерации генетического поиска, формируется популяция для роевого алгоритма. В формируемую популяцию включаются лучшие, но отличные друг от друга хромосомы. При необходимости полученная популяция доукомплектовывается новыми индивидами. После этого дальнейший поиск решения осуществляется роевым алгоритмом.

При втором подходе метод роя частиц используется в процессе генетического поиска и играет роль аналогичную генетическим операторам. В этом случае на каждой итерации генетического алгоритма синтез новых хромосом, с одной стороны, осуществляется с помощью кроссинговера и мутации, а, с другой стороны, с помощью операторов метода роя частиц по формулам (7,8) и модифицированной формуле. Модифицированная формула получается путем удаления в формуле (4) второй компоненты и имеет вид

$$v_{ij}(t+1)=w\cdot vc_{ij}(t)+ k\cdot rnd(0,1)\cdot(xc_j(t) - xc_{ij}(t)). \qquad (9)$$

Третий подход является объединением первого и третьего подходов.

Оценка временной сложности операторов роя частиц не превышает оценки временной сложности генетических операторов. Оценка временной сложности генетического алгоритма не превышает оценки временной сложности алгоритма роя частиц. В связи с этим общая оценка временной сложности при любом подходе к гибридизации не превышает оценки временной сложности генетического алгоритма и лежит в пределах $O(n^2)$-$O(n^3)$.

При совместной работе алгоритмов в рамках общего подхода вероятность получения оптимального решения составило 0,9. Среднее отклонение неоптимальных решений составило 1%. Результаты экспериментов разработанного алгоритма сравнивались по MCNC тестам [12] с алгоритмами [4-7]. Представленный роевой алгоритм в сочетании с генетическим алгоритмом синтеза дерева разрезов находит решения по площади превосходящие результаты сравниваемых алгоритмов в среднем на 3-5%. с меньшими временными затратами для задач большой размерности, кроме того, он оптимизирует длину соединений.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Naveed Sherwani.* Algorithms for VLSI Physical Design Automation. Kluwer academic publishers. Boston /Dordrecht/ London. 1995.
2. *Sarrafzadeh M. and Wong C.K.* An Introduction to VLSI Physical Design. New York: McGraw Hill. 1996.
3. *Kahng A.B.* “Classical Floorplanning Harmful”, *ISPD 2000*, pp. 207-213.
4. *Cong J. en all.* “Microarhitecture Evolution With Physical Planning”, *DAC*, 2003, P. 32-35.
5. *Adyaetall S.N.* “Unification of Partioning, Plasement and Floorplanning”. *DAC*, 2004, P. 550-557.
6. *Liu I., Chen H., Chou T., Aziz A., Wong D.F.* “Integrated Power Supply Planning and Floorplanning”, *IEEE Tranc.on CAD*, 2004.
7. *Cheng L., Wong D.F.* “Floorplan Design For Multi-million Gate FPGAs”, DAC, 2004, P. 292-313.

8. *Лебедев В.Б.* Планирование СБИС методом адаптивного поиска // Известия ТРТУ. – Таганрог: Изд-во ТРТУ, 2000, №2. – С. 168-177.
9. *Лебедев Б.К.* Адаптация в САПР. – Таганрог: Изд-во ТРТУ, 1999.
10. *Лебедев Б.К.* Методы поисковой адаптации в задачах автоматизированного проектирования СБИС: монография. – Таганрог: Изд-во ТРТУ, 2000.
11. *Mazumder P., Rudnick E. Genetic Algorithm For VLSI Design, Layout & Test Automation*. India, Pearson Education, 2003.
12. MCNC Electronic and Information Technologies (Online). Available: www.mcnc.org.
13. *МакКоннелл Дж.* Основы современных алгоритмов. – М.: Техносфера, 2004.
14. *G. Di Caro, F. Ducatelle, L. M. Gambardella*. AntHocNet: An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16(5):443-455, 2005.
15. *Курейчик В.М., Лебедев Б.К., Лебедев О.Б.* Поисковая адаптация: теория и практика. – М.: Физматлит, 2006.
16. *Engelbrecht A.P. Fundamentals of Computational Swarm Intelligence.* John Wiley & Sons, Chichester, UK, 2005.
17. *Clerc M.* Particle Swarm Optimization. ISTE, London, UK, 2006.
18. *Poli R.* Analysis of the publications on the applications of particle swarm optimisation. Journal of Artificial Evolution and Applications, Article ID 685175, 10 pages, 2008.
19. *Емельянов В.В., Курейчик В.М., Курейчик В.В.* Теория и практика эволюционного моделирования. – М.: Физматлит, 2003.

Лебедев Борис Константинович
Технологический институт федерального государственного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: lbk@tsure.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 8(8634)371-743.
Кафедра систем автоматизированного проектирования; профессор.

Лебедев Владимир Борисович
Технологический институт федерального государственного образовательного учреждения высшего профессионального образования «Южный федеральный университет» в г. Таганроге.
E-mail: lbk@tsure.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 8(8634)371-743.
Кафедра систем автоматизированного проектирования; доцент.

Lebedev Boris Konstantinovich
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education “Southern Federal University”.
E-mail: lbk@tsure.ru.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: 8(8634)371-743.
Department of Computer Aided Design; professor.

Lebedev Vladimir Borisovich
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education “Southern Federal University”.
E-mail: lbk@tsure.ru.
44, Nekrasovskiy, Taganrog, 347928, Russia.
Phone: 8(8634)371-743.
Department of Computer Aided Design; associate professor.