УДК 681.325

Биоинспирированные методы планирования кристалла СБИС

Б.К. Лебедев, О.Б. Лебедев

Южный федеральный университет, lebedev.b.k@gmail.com

***Аннотация* — В работе для решения задачи планирования СБИС используется новая, предложенная авторами, парадигма роевого интеллекта - *муравьиное дерево* (*trees ant colony optimization* (T-ACO)), основанная на идее непрямого обмена - *стигмержи (stigmiergy)*, позволяющая осуществлять синтез дерева. Такой подход является эффективным способом поиска решения, при котором план строится на основе различного рода деревьев.**

***Ключевые слова* — планирование СБИС, дерево, роевой интеллект, муравьиная колония, муравьиное дерево, адаптивное поведение, самоорганизация, оптимизация.**

I. Введение

Основные проблемы задачи планирования кристалла СБИС – это проблема поиска подхода к представлению решения (плана) и проблема построения оптимизационной процедуры поиска решения. План для множества модулей M представляет собой прямоугольник R с размерами (h_0, w_0), разрезанный вертикальными и горизонтальными линиями на множество областей u_i с размерами (y_i, x_i), в каждую из которых помещается соответственно модуль m_i с размерами (h_i, w_i), с соблюдением ограничений $h_i \le y_i$, $w_i \le x_i$. Площадь S_R плана R равна $h_0 w_0$. В настоящее время наибольшее распространение получил подход, при котором план строится на основе различного рода деревьев. Однако существующие способы записи и кодирования деревьев предполагают для их модификации использование нелинейных процедур.

Принято разбивать все множество представлений плана на два класса: гильотинный и негильотинный. Гильотинная структура может быть получена путем рекурсивного деления прямоугольника на две части горизонтальными и/или вертикальными разрезами. Класс негильотинных представлений плана реализуется методами: последовательной пары [1], ограниченной разрезающей решетки [2], О-дерева [3], B*-дерева [4, 7], списка угловых модулей [5], графа транзитивного замыкания [6], обобщенной польской записи (ОПЗ) [7, 8] и др.

Представление плана оказывает большое влияние на операции над модулями и сложность процесса проектирования [9]. В общем случае негильотинное представление имеет большее пространство решений и позволяет добиться более компактного расположения модулей по сравнению с гильотинным планом. Тем не менее в работе [10] показано, что решения гильотинного типа сравнимы по качеству с негильотинными. Оценка плана рассчитывается после выполнения процедуры свертки.

Основные критерии оптимизации [11–13]: площадь плана, суммарная длина проводников, задержки, энергопотребление, температура. В общем случае в качестве критерия оптимизации может использоваться аддитивная свертка приведенных показателей. Без потери общности в качестве основной цели оптимизации рассматривается минимизация общей площади $S_R = h_0 w_0$ плана R при соблюдении ограничений $h_i \le y_i$, $w_i \le x_i$: $F = S_R$.

Разработанные к настоящему времени методы планирования СБИС основываются на: целочисленном программировании; прямоугольной дуализации; иерархическом дереве; ограничениях; квантовых алгоритмах; итерационных поисковых алгоритмах; моделирование отжига; альтернативная адаптация; генетические алгоритмы (ГА); дифференциальная эволюция; моделирование роя частиц; миметические алгоритмы и др. Основными недостатками этих алгоритмов является невысокое качество результатов из-за попадания в локальные ямы, малая пригодность для задач большой размерности, плохая приспособленность для реализации на современных технических средствах. Одним из возможных методов решений этой проблемы является использование методов случайного направленного поиска, основанного на моделировании естественных процессов. В настоящее время интенсивно разрабатываются математические методы, в которых заложены принципы природных механизмов принятия решений. К таким методам можно отнести, прежде всего, методы моделирования отжига, методы эволюционного моделирования, генетические алгоритмы, методы эволюционной адаптации, алгоритмы роевого интеллекта, а также муравьиные (Ant Colony Optimization - ACO) и пчелиные (Bee Colony Optimization - BCO) алгоритмы [12-18]. Отдельные попытки [18] применения эволюционного моделирования к задаче планирования были достаточно успешными. Однако предложенные структуры алгоритмов [19-21] фактически являются “слепыми ” поисковыми структурами с присущими им недостатками: генерация решений с нарушениями, что требует дополнительного контроля; генерация большого количества подобных решений; генерация большого количества “плохих” решений.

В работе для решения задачи планирования СБИС используется новая, предложенная авторами,

парадигма роевого интеллекта - **муравьиное дерево** (*trees ant colony optimization* (**T-ACO**)), основанная на идеях муравьиной колонии [22] и, в первую очередь, на идее непрямого обмена - *стигмержи (stigmiergy)*, позволяющая осуществлять синтез дерева.

II. НОВАЯ ПАРАДИГМА КОМБИНАТОРНОЙ ОПТИМИЗАЦИИ -- МУРАВЬИНОЕ ДЕРЕВО (*TREES ANT COLONY OPTIMIZATION* (T-ACO))

Примером использования древовидных структур является формирование плана кристалла путем рекурсивного разрезания прямоугольников на две части задаваемого деревом "гильотинного разреза". Дерево разрезов (бинарное) формируется из двух типов вершин (рис. 1). Множество вершин первого типа $M=\{m_i|i=1, 2, ..., n_m\}$, являющихся листьями дерева D, соответствуют областям, в каждую из которых помещается соответственно модуль m_i. Множество вершин второго типа $C=\{c_i|i=1, 2, ..., n_c\}$, соответствуют разрезам (H - горизонтальный или V - вертикальный). Для бинарного дерева разрезов всегда выполняется равенство $n_m = n_c+1$. Процесс синтеза плана включает два связанных этапа – синтез дерева разрезов и его свертка для формирования плана [16]. Оценка плана рассчитывается после выполнения процедуры свертки. Наиболее трудоемким является процесс синтез дерева разрезов.

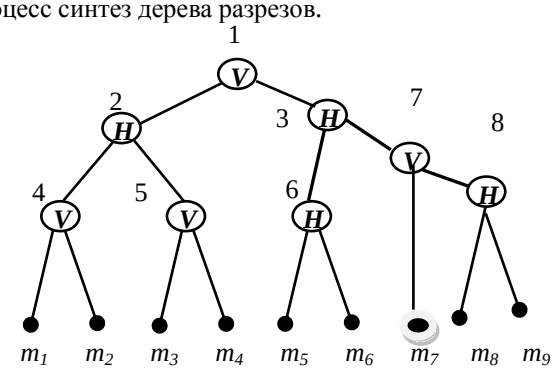

Рис. 1. Дерево разрезов

Представление оптимизационной задачи в виде парадигмы **T-ACO** опирается на два ключевых момента: формирование графа поиска решений (ГПР) и выращивание допустимых альтернативных деревьев на графе поиска решений.

Граф поиска решений $G=(X,U)$, $X=S\cup C\cup M$ - множество вершин, U - множество ребер строится следующим способом. Пусть n_c=4, а n_m =5. Вначале на вершинах множества C формируется полный граф (рис. 2). Ребра неориентированные.

Вводится стартовая вершина S, которая дугами связывается с каждой вершиной множества C (рис. 3).

Далее каждая вершина c_i связывается дугами со всеми n_m вершинами множества M (рис. 4).

Поиск решения. После построения ГПР $G=(S\cup C\cup M,U)$ на всех его ребрах U откладывается начальное количество феромона Q/v, где $v=|U|$.

В общем случае поиск решения задачи осуществляется коллективом муравьев $A=\{a_k|k=1, 2, ..., n_k\}$. Построение дерева муравьями начинается со стартовой вершины S. Процесс поиска решений итерационный. На каждой итерации алгоритма муравьиного дерева каждый муравей a_k выращивает в графе $G=(M\cup C,U)$ свое конкретное, бинарное дерево, которое является решением задачи. Каждая итерация l включает три этапа. На первом этапе муравей выращивает дерево, на втором этапе откладывает феромон, на третьем этапе осуществляется испарение феромона. В работе используется циклический (ant-cycle) метод муравьиных систем. В этом случае феромон откладывается агентом на ребрах после полного формирования решения. На первом этапе каждой итерации каждый k-й муравей выращивает свое собственное бинарное дерево D_k. Бинарное дерево выращивается на базе ГПР последовательно от корня до листьев.

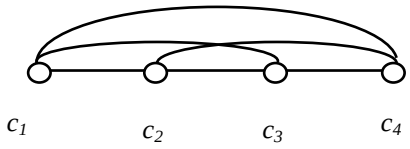

Рис. 2. Первый этап построения ГПР

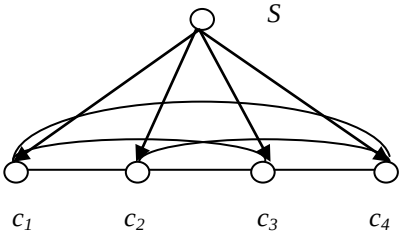

Рис. 3. Второй этап построения ГПР

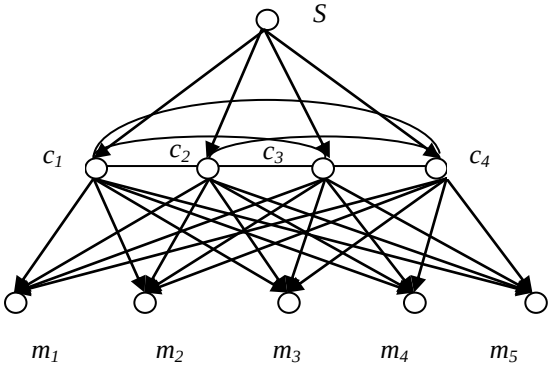

Рис. 4. Третий этап построения ГПР

Пошаговый процесс выращивания дерева на базе ГПР начинается со стартовой вершины S. В этом случае на первом шаге муравьем осуществляется выбор корневой вершины $c^0\in C$, смежной вершине S. При втором подходе в качестве начальных (корневых) вершин у выращиваемых муравьями деревьев используются вершины множества C общим числом n_c. Через некоторое число итераций можно сменить способ выбора начальных (корневых) вершин. Другими словами вначале муравьи используют второй способ выбора корневых вершин. А затем все муравьи

процесс выращивания дерева начинают со стартовой вершины S, поскольку через некоторое количество итераций выясняется, что использование некоторых вершин $c^0 \in C$ в качестве начальных не приводит к построению оптимальных деревьев, в связи с этим нет смысла это делать.

На каждом шаге выращивания дерева выбирается одна из еще не связанных вершин ГПР, которая связывается ребром с одной из уже ранее выбранных и вошедших в состав дерева вершин. На вершины накладываются следующие ограничения. Вершины множества M могут быть в строящемся дереве только дочерними (листьями), в каждую вершину $m_j \in M$ входит только одно ребро ГПР. Вершины множества C должны быть связаны с двумя дочерними вершинами из множества $C \cup M$ и сами в свою очередь являются дочерними для некоторых вершин множества C. Изначально каждая вершина $c_i \in C$ обладает двумя *вакансиями* для связи с двумя дочерними вершинами и одной *вакансией* для связи с родительской вершиной. В процессе построения дерева вакансии вершин $c_i \in C$ последовательно заполняются. Процесс завершается после заполнения всех вакансий.

Состояние процесса построения дерева на шаге t описывается следующими параметрами. Подмножества $C_1(t) \subset C$ и $M_1(t) \subset M$ включают вершины уже вошедшие в состав строящегося дерева на $t-1$ предыдущих шагах. Подмножества $C_2(t) \subset C$ и $M_2(t) \subset M$ включают вершины, которые еще не вошли в состав строящегося дерева. $C_1(t) \cup C_2(t) = C$, $M_1(t) \cup M_2(t) = M$. Причем, если $C_2(t) \cup M_2(t) \neq \varnothing$, то вершины множества $C_1(t)$ должны обладать хотя бы одной вакансией.

Процедура включения на шаге t вершины в состав строящегося дерева производится с соблюдением условия *достаточности вакансий*, суть которого заключается в том, что если число вершин еще не вошедших в состав строящегося дерева больше единицы (т.е. $|C_2(t) \cup M_2(t)| > 1$), то после включения одной из таких вершин в множество $C_1(t) \cup M_1(t)$ вновь образованное множество $C_1(t+1)$ должно обладать хотя бы одной вакансией. Условие *достаточности вакансий* служит для исключения преждевременного завершения строительства дерева.

Для каждой вершины $x_j \in X_2(t) = C_2(t) \cup M_2(t)$ определяется набор ребер $U_{jk}(t)$, связывающих x_j с вершинами множества $X_1(t) = C_1(t) \cup M_1(t)$, каждое из которых с соблюдением перечисленных выше ограничений и условия достаточности может войти в состав строящегося бинарного дерева. Для каждого ребра $u_i \in (\cup U_{jk}(t))$ определяется параметр f_{ik} - суммарный уровень феромона на этом ребре.

Вероятность P_{ik} включения ребра u_i в формируемое дерево $D_k(t)$ определяется следующим соотношением

$$P_{ik} = f_{ik} / \sum_i (f_{ik}).$$

Агент с вероятностью P_{ik} выбирает одно из ребер, которое включается в строящееся дерево $D_k(t)$. Процесс построения дерева продолжается до тех пор, пока множество $X_2(t)$ не станет пустым.

На втором этапе итерации каждый муравей откладывает феромон на рёбрах построенного им дерева.

Количество феромона $\tau_k(l)$, откладываемое муравьем a_k на каждом ребре построенного им дерева D_k, пропорционально базовому (опорному) количеству феромона Δ и определяется следующим образом

$$\tau_k(l) = \Delta / F_k(l),$$

где l-номер итерации, $F_k(l)$– целевая функция для решения (площадь плана), соответствующего дереву D_k, Чем меньше $F_k(l)$, тем больше феромона откладывается на ребрах построенного дерева и, следовательно, тем больше вероятность выбора этих ребер при построении деревьев на следующей итерации.

После того, как каждый агент сформировал решение и отложил феромон, на третьем этапе происходит общее испарение феромона на ребрах графа G в соответствии с формулой

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

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

Время работы этого алгоритма зависит от времени жизни колонии l (число итераций), количества модулей n и числа агентов m, и определяется как $(l*c^2*m)$.

III. Алгоритмы планирования СБИС на основе парадигмы муравьиного дерева

Алгоритм поведения муравьиной колонии

1. Формируется множество вершин первого типа $M=\{m_i \mid i=1,2,\ldots,n_m\}$ и множество вершин второго типа $C=\{c_i \mid i=1,2,\ldots,n_c\}$.

2. Строится граф поиска решений $G=(M \cup C,U)$.

3. На всех ребрах графа поиска решений $G=(M \cup C,U)$ откладывается начальное количество феромона $\Omega = Q/v$, где $v=|U|$.

4. В общем случае поиск решения задачи осуществляется коллективом муравьев $A=\{a_k \mid k=1, 2, \ldots, n_k\}$.

5. Выращивание дерева муравьи начинают со стартовой вершины S.

6. Задается число итераций n_l.

7. $l=1$. (l – номер итерации).

8. $k=1$. (k- номер муравья в группе).

9. (*Алгоритм муравья)* (см. ниже).

Муравей a_k строит на графе $G=(M\cup C,U)$ свое решение (дерево) $D_k(l)$.

10. Трансформация между представлением $D_k(l)$ и планом $p_k(l)$. Расчет значения критерия $F_k(l)$ для $p_k(l)$.

11. Если $k<n_k$, то $k=k+1$ и переход к пункту 9.

12. Каждый муравей a_k на ребрах построенного им дерева $D_k(l)$ откладывает феромон в количестве

$$\tau_k(l)=\Delta/F_k(l). \qquad (4)$$

Общее количество феромона $h_i(l)$, отложенного на ребре $u_i \in D_k(l)$ после выполнения l итераций, определится как

$$h_i(l)= h_i(l-1)+ \sum_{k|u_i \in D_k(l)} \tau_k(l)$$

13. После того, как каждый агент отложил феромон, происходит общее испарение феромона на ребрах графа $G=(M\cup C,U)$ в соответствии с нижеприведенной формулой

$$h_i= h_i(1- \rho), \qquad (5)$$

где ρ – коэффициент обновления, h_i – суммарное количество феромона, отложенного муравьями на ребре $u_i \in U$ графа $G=(M\cup C,U)$.

14. Выбор лучшего решения, полученного на протяжении всех выполненных итераций.

15. Если все итерации выполнены, то конец работы алгоритма, в противном случае $l=l+1$ и переход к пункту 8.

Алгоритм поведения муравья

1. Агент a_k начинает выращивание дерева $D_k(l)$ в графе $G=(M\cup C,U)$ со стартовой вершины S.

2. Формируется множество вакансий $V=\{v_i|i=1, 2, ..., n_c\}$, соответствующих множеству внутренних вершин $C=\{c_i|i=1, 2, ..., n_c\}$. Всем элементам v_i присваивается значение равное двум.

3. $t=1$. Выбор случайным образом корневой вершины $c_i \in C$.

Формирование множества $X_1(t)$, которое включает вершины уже вошедшие в состав строящегося дерева на $t-1$ предыдущих шагах:

$$C_1(t)=\{c_i\},\ M_1(t)=\varnothing,\ X_1(t) = C_1(t)\cup M_1(t).$$

Формирование множества $X_2(t)$ свободных вершин

$$C_2(t)= C\backslash c_i,\ M_2(t)=M,\ X_2(t) = C_2(t)\cup M_2(t).$$

4. Формирование множества $U^*(t)\subset U$ ребер, связывающих вершины множества $X_1(t)$ с вершинами множества $X_2(t)$.

5. *(Соблюдение условия достаточности вакансий).*

Если $|C_2(t)\cup M_2(t)| >1$), то производится исключение из $U^*(t)$ всех ребер, нарушающих условие достаточности вакансий. Ребро $u_v \in U^*(t)$ нарушает условие достаточности вакансий, если после его включения в состав строящегося дерева число вакансией вновь образованного множества $X_1(t+1)$ станет равным нулю.

После удаления таких ребер образуется множество $U_o^*(t) \subset U^*(t)$.

6. Для каждого ребра $u_i \in U_o^*(t)$ определяется суммарный уровень феромона f_{ik} на этом ребре.

7. Вероятность P_{ik} включения ребра u_i в формируемое дерево $D_k(t)$ определяется следующим соотношением

$$P_{ik}=f_{ik}/\sum_i f_{ik}.$$

8. Агент с вероятностью $P_{ik}(t)$ выбирает одно из ребер $u_i \in U_o^*(t)$, которое включается в формируемое дерево $D_k(t)$ и исключается из $U^*(t)$.

Вершина $x_i \in X_2(t)$ и инцидентная ребру u_i исключается из множества $X_2(t)$ и включается в множество $X_1(t)$.

9. Если $X_2(t) \neq \varnothing$, то $t=t+1$ и переход к пункту 4.

10. Конец работы алгоритма.

Временная сложность этого алгоритма зависит от времени жизни колонии l (число итераций), количества вершин графа n и числа муравьев m.

На основе бинарного дерева разрезов осуществляется построение плана путем последовательной бинарной свертки областей по дереву разрезов, начиная от листьев дерева [15]. Каждой внутренней вершине дерева разрезов будет соответствовать область, полученная в результате бинарной свертки поддерева, имеющего корнем эту внутреннюю вершину. Процесс бинарной свертки представляет собой слияние двух дочерних областей u_i и u_j, формирование родительской области u_k, определение размеров для u_k и новых размеров для u_i и u_j. В структуре бинарного дерева гильотинного разреза используются два инфиксных оператора V и H. Запись $u_k = u_i\ H\ u_j$ означает, что области u_i и u_j сливаются по горизонтали в одну область u_k. Если $u_k = u_i\ V\ u_j$, то области u_i и u_j сливаются по вертикали (рис. 5). В простейшем случае области при слиянии сохраняют свою ориентацию и размеры. В работе [15] рассмотрена адаптивная процедура выбора ориентации и размеров областей при свертке, минимизирующая площадь плана. В представлении «обобщенная польская запись» [16] используется дополнительный оператор @. Оператор @ называют угловым, он помещает модуль или подплан в угол, сформированный другими модулями. При наличии операторов только горизонтального (H-) и вертикального (V-) разрезов нельзя использовать мертвую зону. Угловой оператор @ задействует угол, образованный модулями u_i и u_j. Временная сложность трансформации в план – $O(n)$. Таким образом, дерево в зависимости от его структуры и типов операторов

свертки может быть использовано для различного вида представлений плана.

Рассмотренная парадигма муравьиного дерева варьируя значениями управляемых параметров, таких как число и типы вершин, значения и типы вакансий, виды операторов слияния и некоторых операторов управления позволяет синтезировать дерево любой требуемой конфигурации.

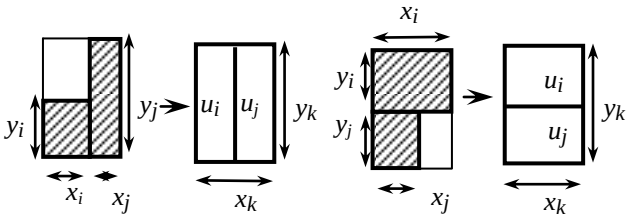

Рис. 5. Слияние двух дочерних областей u_i и u_j, и т.д.

IV. Экспериментальные исследования

Проведение экспериментов преследовало две цели: исследование эффективности и качества механизмов алгоритма муравьиного дерева для решаемых задач и эффективности предложенных алгоритмов. Для исследования механизмов алгоритма муравьиного дерева была использована процедура синтеза **контрольных примеров с известным оптимумом** по аналогии с известным методом PEKU (Placement Examples with Known Upper bounds on wirelength) [22, 23]. Размеры схем сгенерированного набора: Ex.1 на 30 блоков; Ex.2 – 40; Ex.3 – 60; Ex.4 – 90; Ex.5 – 110.

Первой целью являлось исследование влияния управляющих операторов, таких как: объем популяции, количество итераций, начальное количество феромона Ω, откладываемое на ребрах графов и т.д.

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

Анализ результатов показал, что увеличение популяции M больше 90 нецелесообразно, так как это не приведет к заметному изменению качества.

При исследовании сходимости алгоритма для каждого эксперимента запоминался номер генерации, после которой не наблюдалось улучшения оценки. В каждой серии из 10 испытаний определялись минимальный и максимальный номера генерации. Кроме этого рассчитывалось среднее значение числа генераций, после которого не наблюдалось улучшения оценки. Для каждой серии испытаний определялось лучшее решение, которое фактически являлось оптимальным. В результате экспериментов установлено, что при объеме популяции $M=90$ алгоритм сходится в среднем на 120 итерации. При этом отклонения в сторону увеличения этой оценки составляли до 10%, а в сторону уменьшения до 35%.

Сравнение оценок решений, полученных алгоритмом муравьиного дерева на тестовых примерах, с известным оптимумом показало, что у 60% примеров полученное решение было оптимальным, у 15% примеров решения были на 5% хуже оптимального, а у 25% примеров решения были хуже не более, чем на 2%.

Для сравнения качества решений задачи планирования кристалла СБИС применялись стандартные тесты для оценки разработанных алгоритмов [24-25]. Эксперименты проводились на MCNC (Microelectronics Center of North Carolina) образцах плат. 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. Для сравнения были выбраны современные планировщики [11, 16]. Представленный алгоритм находит решения, по площади превосходящие результаты существующих алгоритмов (табл. 1). Сравнение результатов произведено по показателю "мертвая зона". Показатель "мертвая зона" определяется как отношение свободной от модулей площади плана к общей площади плана в процентах. В колонках [11] и [16] приведены результаты планировщиков из работ [11] и [16]. В колонке [МД] приведены результаты, полученные алгоритмом муравьиного дерева. Было замечено, что уменьшение площади, как правило, приводило к уменьшению длины соединений. Следует отметить, что экспериментальная временная сложность алгоритма на одной итерации при фиксированных значениях управляющих параметров составляет $O(nlgn)$, а временная сложность существующих алгоритмов [3-15] - $O(n^2)$, где n – число модулей.

При больших размерностях временные показатели разработанного алгоритма превосходят показатели сравниваемых алгоритмов при лучших значениях целевой функции.

Таблица 1

Сравнения результатов

Тест	Мертвая зона (%)		
	[16]	[11]	МД
apte	2,11	2,34	2,01
xerox	3,37	3,67	3,15
hp	3,39	3,86	3,22
ami33	3,54	4,38	3,5
ami49	4,07	4,84	4,01

Предлагаемый алгоритм планирования позволяет лучше упаковывать модули по сравнению с планировщиками аналогичного уровня (качество упаковки улучшилось в среднем на 3–6%).

При больших размерностях временные показатели разработанного алгоритма превосходят показатели

сравниваемых алгоритмов при лучших значениях целевой функции.

V. Заключение

В работе для решения задачи планирования СБИС используется новая, предложенная авторами, парадигма роевого интеллекта - муравьиное дерево (*trees ant colony optimization* (T-ACO)), основанная на идеях муравьиной колонии и, в первую очередь, на идее непрямого обмена - *стигмержи (stigmiergy),* позволяющая осуществлять синтез дерева. Такой подход является эффективным способом поиска решения, при котором план строится на основе различного рода деревьев. Большинство существующих способов записи и кодирования деревьев предполагают для их модификации использование нелинейных процедур. Как правило, данные о структуре дерева и именах модулей разделены. Например, в генетических алгоритмах одна хромосома служит для кодирования структуры дерева, а другая для имен модулей. Представления, синтезируемые муравьиным деревом, наследуют простоту обычной польской записи, линейную оценку временной сложности декодирования, позволяют работать как с гильотинными, так и с негильотинными планами.

Литература

[1] Sengupta D. et al. Sequence pair based voltage island floorplanning // IEEE International Green Computing Conference. – 2011. – pp. 1–6.

[2] 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.

[3] Guo P.N. Floorplanning Using a Tree Representation // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 20, No. 2, FEBRUARY 2001. – pp. 281–289.

[4] Chang Y.-C. B*-trees: A New Representation for Non-slicing Floorplans // proc. DAC, 2000. – pp. 458–463.

[5] 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.

[6] 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 Volume 13 Issue 2, February 2005. – pp. 288-292.

[7] Lin C.-T. et al. GPE: A New Representation for VLSI Floorplan Problem // IEEE International Conference on Computer Design (ICCD'02), 2002. – pp. 42–44.

[8]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. .

[9] Pritha Banerjee, Megha Sangtani, and Susmita Sur-Kolay. “Floorplanning for Partially Reconfigurable FPGAs,” *IEEE* Trans. Comput.-Aided Des. Integr. Circuits Sys., vol. 30, no. 1, pp. 8-17, January 2011.

[10] Lai M., and Wong D.F. Slicing Tree Is a Complete Floorplan Representation. // In Proc. DATE, 2001. – pp. 228–232.

[11] Ma Q., Young E.F.Y. Multivoltage Floorplan Design // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume: 29 No: 4. – 2010. – pp. 607 – 617.

[12] Лебедев О.Б. Планирование СБИС на основе метода муравьиной колонии // Известия Южного федерального университета. Технические науки. – 2010. Т.108. №7. – С.67-73.

[13] Лебедев Б.К., Лебедев В.Б. Размещение на основе метода пчелиной колонии // Известия Южного федерального университета. Технические науки. 2010. Т. 113. № 12. С. 12–19.

[14]. Лебедев В.Б. Построение связывающих сетей на основе роевого интеллекта и генетической эволюции //Сборник научных трудов XIV Всероссийской научно-технической конференции "НЕЙРОИНФОРМАТИКА-2012”. Ч.2 М.: Изд-во Физматлит, 2012. - С. 93-103.

[15]Лебедев Б.К., Лебедев В.Б. Планирование на основе роевого интеллекта и генетической эволюции Известия Южного федерального университета. Технические науки. 2009.Т.93.№4.С.25–33.

[16] Ерошенко И.Н. Планирование кристалла СБИС с учетом энергопотребления// МЭС-2012. V Всероссийсканаучно-техническая конференция «Проблемы разработки перспективных микро- и наноэлектронных систем - 2012». Сборник трудов.- М.:ИППМ РАН, 2012. С.257-263

[17] Лебедев В.Б. Планирование СБИС методом адаптивного поиска. //Известия ТРТУ. Таганрог. Изд-во ТРТУ, 2000, №2, с. 168-177.

[18].Лебедев Б.К. Лебедев В.Б. Адаптивная процедура выбора ориентации модулей при планировании СБИС //Проблемы разработки перспективных микроэлектронных систем - 2006. Сборник научных трудов /под общ. ред. А.Л. Стемпковского. – М.: ИППМ РАН, 2006, с. 120-125.

[19] Курейчик В.М., Лебедев Б.К., Лебедев О.Б. Гибридный алгоритм разбиения на основе природных механизмов принятия решений //Искусственный интеллект и принятие решений. – Москва: Изд-во Институт системного анализа РАН, 2012. С. 3-15

[20] Jackey Z. Yan, Chris Chu //DeFer: Deffered Decision Making Enabled Fixed-Outline Floorplanning Algorithm //*IEEE Trans. Comput.-Aided Des. Integr. Circuits Sys.*. 2010. Vol. 29. №. 3. Pp. 367-381.

[21] Tang Maolin. Yao Xin. A memetic algorithm for VLSI floorplanning//*IEEE Trans. on Systems, Man, And Cybernetics—Part B: Cybernetics*. 2007. № 37(1). Pp. 278-292.

[22]. M. Dorigo and T. Stützle. Ant Colony Optimization. MIT Press, Cambridge, MA, 2004.

[23] *Chang C.C., Cong J., ie M.* Optimality and scalability study of existing placement algorithms //*In Proc.s of the Asia South Pacific Design Automation Conference.* 2003. Pp. 621–627.

[24]. MCNC Electronic and Information Technologies (Online). Available: www.mcnc.org (дата обращения: 05.06.2014).

[25]. HB Floorplan Benchmarks [Online]. Available: http://cadlab.cs.ucla.edu/cpmo/HBsuite.html (дата обращения: 05.06.2014).