Васильев Николай Олегович – Институт проблем проектирования в микроэлектронике РАН (ИППМ РАН); e-mail: vasilyev_n@ippm.ru; 124365, Москва, Зеленоград, ул. Советская, 3; инженер-исследователь.

Заплетина Мария Андреевна – e-mail: zapletina_m@ippm.ru; м.н.с.

Иванова Галина Александровна – e-mail: ivanova_g@ippm.ru; с.н.с.; к.т.н.

Щелоков Альберт Николаевич – e-mail: schan@ippm.ru; зам. директора; к.ф.-м.н.

Vasilyev Nikolay Olegovich – The Institute for Design Problems in Microelectronics (IPPM RAS); e-mail: vasilyev_n@ippm.ru; 3, Sovetskaya street, Zelenograd, Moscow, 124365, Russia; research engineer.

Zapletina Mariya Andreevna – e-mail: zapletina_m@ippm.ru; junior researcher.

Ivanova Galina Aleksandrovna – e-mail: ivanova_g@ippm.ru; senior researcher; cand. of eng. sc.

Schelokov Albert Nikolaevich – e-mail: schan@ippm.ru; Deputy Director; cand. phys. and math. sc.

УДК 004.896 DOI 10.18522/2311-3103-2020-4-125-136

Б.К. Лебедев, В.Б. Лебедев, О.Б. Лебедев

ПОИСКОВЫЙ ПОПУЛЯЦИОННЫЙ АЛГОРИТМ РАЗМЕЩЕНИЯ ЭЛЕМЕНТОВ СБИС*

В работе рассматривается поисковый популяционный алгоритм размещения компонентов СБИС. По аналогии с процессом возникновения и формирования кристаллов из вещества, процесс порождения решения путем последовательного проявления и конкретизации решения на базе интегральной россыпи альтернатив назван методом кристаллизации россыпи альтернатив. Решение Q_k задачи размещения представляется в виде биективного отображения $F_k=A\rightarrow P$, каждому элементу множества A соответствует один единственный элемент множества P и наоборот. Лежащая в основе алгоритма метаэвристика кристаллизации россыпи альтернатив выполняет поиск решений с учетом коллективной эволюционной памяти, под которой подразумевается информация, отражающая историю поиска решения и памяти поисковой процедуры. Отличительной особенностью используемой метаэвристики является учет тенденции к использованию альтернатив из наилучших найденных решений. Предложены компактные структуры данных для хранения интерпретаций решений и памяти. Алгоритм, связанный с эволюционной памятью, стремится к запоминанию и многократному использованию способов достижения лучших результатов. Разработанный алгоритм относится к классу популяционных алгоритмов. Итерационный процесс поиска решений включает три этапа. На первом этапе каждой итерации конструктивным алгоритмом формируется n_a решений Q_k. Работа конструктивного алгоритма базируется на базе показателей основной интегральной россыпи альтернатив – матрицы R, в которой хранятся интегральные показатели решений, полученных на предыдущих итерациях. Процесс назначения элемента в позицию включает две стадии. На первой стадии выбирается элемент, а на второй стадии – позиция p_j. При этом должно выполняться ограничение: каждому элементу соответствует одна позиция p_j. Рассчитывается оценка ξ_k решения Q_k и оценка полезности δ_k множества позиций P_k выбранных агентами. В работе используется циклический метод формирования решений. В этом случае наращивание оценок интегральной полезности δ_k в основной интегральной россыпи альтернатив B выполняется после полного формирования множества решений Q. На втором этапе итерации производится наращивание оценок интегральной полезности δ_k в основной интегральной россыпи альтернатив – матрице R. На третьем этапе итерации осуществляется

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

снижение оценок полезности δ_k интегральной россыпи альтернатив R на априори заданную величину δ^*. Работа алгоритма завершается после выполнения заданного числа итераций. Сравнительный анализ с другими алгоритмами решения производился на стандартных тестовых примерах (бенчмарках) корпорации IBM, при этом решения, синтезируемые алгоритмом CAF, превосходят по эффективности решения известных методов в среднем на 6%. Временная сложность алгоритма – $O(n^2)$-$O(n^3)$.

Размещение; СБИС; коллективная эволюционная память; оценка полезности; оптимизация; популяционный алгоритм; метод кристаллизации россыпи альтернатив.

B.K. Lebedev, V.B. Lebedev, O.B. Lebedev

SEARCH POPULATION ALGORITHM FOR VLSI ELEMENTS PLACEMENT

The paper considers a population search algorithm for the placement of VLSI components. By analogy with the process of the emergence and formation of crystals from matter, the process of generating a solution by sequential manifestation and concretization of the solution based on an integral placer of alternatives is called the method of crystallization of a placer of alternatives. The solution Q_k of the placement problem is represented as a bijective mapping $F_k = A \rightarrow P$, each element of the set A corresponds to one single element of the set P and vice versa. The metaheuristic of crystallization of a placer of alternatives underlying the algorithm searches for solutions taking into account collective evolutionary memory, which means information reflecting the history of the search for a solution and the memory of the search procedure. A distinctive feature of the metaheuristic used is that it takes into account the tendency to use alternatives from the best found solutions. Compact data structures for storing solution interpretations and memory are proposed. An algorithm associated with evolutionary memory seeks to memorize and reuse ways to achieve better results. The developed algorithm belongs to the class of population. The iterative process of finding solutions includes three stages. At the first stage of each iteration, the constructive algorithm generates n_q solutions Q_k. The work of the constructive algorithm is based on the indicators of the main integral placer of alternatives – the matrix R, which stores the integral indicators of the solutions obtained at the previous iterations. The process of assigning an item to a position involves two stages. In the first stage, the element is selected, and in the second stage, the position p_j. In this case, the restriction must be fulfilled: each element corresponds to one position p_j. The estimate ξ_k of the solution Q_k and the estimate of the utility δ_k of the set of positions P_k selected by the agents are calculated. The work uses a cyclical method of forming decisions. In this case, the accumulation of estimates of the integral utility δ_k in the main integral placer of alternatives R is performed after the complete formation of the set of solutions Q. At the second stage of the iteration, the estimates of the integral utility δ_k are increased in the main integral placer of alternatives − the matrix R. At the third stage of the iteration, the estimates of the utility δ_k of the integral placer of alternatives R are reduced by a priori a given value δ^*. The algorithm ends after the specified number of iterations has been completed. Comparative analysis with other solution algorithms was carried out on standard test examples (benchmarks) of the IBM corporation, while the solutions synthesized by the CAF algorithm exceed the solution efficiency of the known methods by an average of 6%. The time complexity of the algorithm is $O(n^2)$-$O(n^3)$.

Placement; VLSI; collective evolutionary memory; utility assessment; optimization; population algorithm; method of crystallization of alternatives placer.

Введение. Разработка поисковых алгоритмов базируется на использовании метаэвристик, заложенных в природных механизмах принятия решений [1–3]. Большинство известных алгоритмов [4–7] используют традиционные итерационные улучшающие структуры, основанные на случайном поиске. Основным недостатком, присущим этому подходу, является вхождение алгоритмов в локальный оптимум, часто далекий от глобального. В работе по аналогии с метаэвристиками на которых построены роевые алгоритмы [3], используется метаэвристика, учитывающая тенденцию к использованию альтернатив (вариантов) из наилучших найденных решений [8,9]. В процессе эволюционной коллективной адаптации методами дискриминантного анализа формируются оценки приспособленности альтер-

натив. Вероятность использования альтернативы в формируемом решении пропорциональна ее приспособленности. Совокупность данных об альтернативах и их оценках составляет *россыпь альтернатив* [8, 9]. По аналогии с процессом образования и роста кристаллов из вещества, процесс порождения решения путем последовательного проявления и конкретизации на базе интегральной россыпи альтернатив назван методом кристаллизации россыпи альтернатив. Другими словами, в процессе эволюционной коллективной адаптации решение «кристаллизуется» в состояние, определяемое совокупностью приспособленных альтернатив. Отсюда название метода оптимизации – *метод кристаллизации россыпи альтернатив* [8, 9].

Задача размещения является одним из наиболее важных этапов в процессе проектирования СБИС, поскольку качество полученного решения в значительной степени влияет на выполнение последующей трассировки соединений [10]. Задача размещения относится к классу *NP*-полных, поэтому не существует алгоритма, позволяющего получать ее решение за полиномиальное время. Проведенные исследования имеющихся подходов к ее решению показывают, что существующие алгоритмы не в полной мере отражают требования по эффективности [4], получаемые решения, с точки зрения их оптимальности неудовлетворительны. Необходимость повышения качества проектирования требует поиска новых путей и подходов к решению задач размещения. Изучение имеющихся методов и алгоритмов решения комбинаторных задач показал, что решения, чаще всего представляются списками. Основу таких алгоритмов составляет процедура формирования упорядоченного списка элементов, задающего последовательность формирования решения базовым конструктивным алгоритмом. Формально упорядоченный список – это код решения. Переход от кода (упорядоченный список) к фенотипу (решению) и вычисление значения функции пригодности выполняется процедурой декодирования. Обычно в качества декодера используются конструктивные алгоритмы. Активно оперирование с решениями, представленными в закодированном виде, развиваются в алгоритмах поисковой оптимизации (биоинспирированными, роевыми, популяционными) [11–16]. Анализ таких алгоритмов показал, что их эффективность выше, чем у классических алгоритмов. В работе рассматривается поисковый популяционный алгоритм размещения элементов СБИС, основанный на методе кристаллизации россыпи альтернатив (КРА) [11–16].

В работе излагается алгоритм размещения элементов на основе метода кристаллизации россыпи альтернатив. С этой целью разработана единая структура данных. С учетом особенностей единой структуры данных разработан поисковый процесс, базирующийся на моделировании коллективной адаптации.

1. Постановка задачи размещения. Проблема размещения может быть сформулирована следующим образом [11–13]. Дано множество элементов (модулей) с расположенными на них терминалами (выводами). Задано множество цепей, связывающих терминалы модулей. Задано коммутационное поле (КП) с описанием позиций, в которых могут размещаться элементы. Необходимо разместить элементы на КП с оптимизацией некоторых критериев качества. Входная информация включает: описание модулей, в котором указываются форма, размеры, расположение терминалов на модулях, множество цепей, связывающих терминалы модулей, и описание КП [14]. Выходная информация включает координаты позиций на КП для размещения всех модулей.

Стандартная постановка задачи размещения элементов в позициях на коммутационном поле формулируется следующим образом [14]. Дано множество элементов $A=\{a_i|i=1,2,\ldots,n\}$ и множество позиций $P=\{p_j|j=1,2,\ldots,n_p\}$ на КП. В качестве модели схемы используется гиперграф $H=(X,E)$, где $X=\{x_i|i=1,2,\ldots,n\}$ – множество вершин, моделирующих элементы, $E=\{e_j|e_j \subset X,\ j=1,2,\ldots,m\}$ – множество гиперре-

бер, моделирующих цепи, связывающие элементы. Для размещения всех элементов необходимо выполнение условия $n_p \geq n$. Целевая функция ξ определяется выбранными критериями. Основными, известными критериями при размещении являются: минимальная суммарная длина связей, минимальная длина самой длинной связи, минимум числа возможных пересечений, минимум возможного числа изгибов соединений, минимальная площадь кристалла.

Каждая позиция p_j имеет координаты (x_j,y_j). Между множеством порядковых номеров позиций и множеством их координат устанавливается взаимно однозначное соответствие. Пусть имеется множество элементов, размещенных в позициях. Произвольное размещение элементов в позициях представляет собой перестановку (упорядоченный список) $Q=<q_j| j=1,2,...,n_p>$, где q_j – номер элемента, размещенного в j-ой позиции. Задача размещения состоит в отыскании оптимального значения функции ξ на множестве перестановок Q [11-14]. Отметим, что в общем случае может использоваться любая модель соединений и функция их оценки. И это никоим образом не влияет на структуру кодирования и декодирования решения.

2. Механизмы размещения на основе метода кристаллизации россыпи альтернатив. В методе кристаллизации россыпи альтернатив [9,10] каждое решение Q_k формируется (представляется) множеством агентов $A=\{a_i|i=1,2,...,n_a\}$, где n_a – число агентов. Каждому агенту a_i соответствует множество альтернативных состояний $S_i=\{s_{ij}|j=1,2,...,n_s\}$, где n_s – число состояний агента a_i. Каждый агент a_i может находиться в одном из альтернативных состояний. Решение Q_k определяется совокупностью альтернативных состояний множества агентов [8, 9].

В рассмотренной задаче размещения агентами являются элементы, а альтернативами – множество позиций $P=\{p_j|j=1,2,...,n_p\}$, где n_p – число позиций в которые помещаются элементы. Другими словами множество агентов $A=\{a_i|i=1,2,...,n_a\}$, соответствует множеству всех элементов. Множество альтернативных состояний $S_i=\{s_{ij}|j=1,2,...,n_p\}$ агента a_i соответствует множеству альтернативных вариантов размещения агента a_i – множеству позиций, в которые может быть помещен элемент.

Решение Q_k задачи размещения представляется в виде биективного отображения $F_k=A\rightarrow P$, каждому элементу множества A соответствует один единственный элемент множества P и наоборот. Номер позиции p_j в которую помещен элемент a_i определяется соотношением $p_j=F_k(a_i)$.

Под *россыпью альтернатив* (РА) решения в работе подразумевается структура данных, несущая информацию об альтернативах агентов в данном решении и об полезности решения, используемой в качестве коллективной эволюционной памяти (КЭП) [8].

Для хранения информации об занимаемых агентами решения Q_k альтернативных позициях (порядковых номерах), и значениях их пригодностей, составляющих коллективную эволюционную память (КЭП), используется матрица $R_k=||r_{kij}||_{m\times m}$, называемая *россыпью альтернатив* (РА) (рис. 1).

$R_k=$

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	6	0	0	0	0
a_2	0	0	0	0	6	0
a_3	6	0	0	0	0	0
a_4	0	0	6	0	0	0
a_5	0	0	0	0	0	6
a_6	0	0	0	6	0	0

Рис. 1. Россыпь альтернатив R_k

Представим структуру данных для отображения решения Q_k, формируемого множеством агентов A, в виде совокупности векторов $R_k=\{R_{ki}|i=1,2,\ldots,n_a\}$. Каждый вектор $R_{ki}=\{r_{kij}|j=1,2,\ldots,n_p\}$ соответствует агенту a_i и фактически является строкой матрицы R_k. Размерность вектора R_{ki} определяется числом возможных состояний агента a_i. В векторе R_{ki} только один элемент r_{kij} соответствующий состоянию s_{ij}, в котором находится агент a_i, имеет значение, отличное от нуля и это значение равно $\delta_k=f(\xi_k)$, где ξ_k – оценка решения Q_k, δ_k – оценка полезности этого решения. Остальные элементы вектора R_{ki} имеют нулевые значения. Таким образом, в векторе R_{ki} хранится информация о состоянии, реализованном агентом a_i в решении Q_k, и об оценке полезности δ_k этого состояния.

Пример. Пусть шестью агентами сформировано решение Q_k=<2, 5, 1, 3, 6, 4>. В решении Q_k агентами реализованы следующие состояния (альтернативы): a_1 в состоянии s_{12}, $a_2 - s_{25}$, $r_3 - s_{31}$, $a_4 - s_{43}$, $a_5 - s_{56}$, $a_6 - s_{64}$, которым соответствуют множество Θ_k ячеек r_{kij} матрицы R_k: Θ_k=={r_{k12}, r_{k25}, r_{k31}, r_{k43}, r_{k56}, r_{k64}}. Элементам множества Θ_k присваивается значение оценки полезности δ_k=6. Тогда россыпь альтернатив R_k для решения Θ_k имеет вид, представленный на рис. 1.

Пусть для решения задачи размещения построено множество решений $Q=\{Q_k|k=1,2,\ldots,n_q\}$:

Q_1=<2,5,1,3,6,4>; Q_2=<2,4,1,6,3,5>; Q_3=<2,4,1,6,3,5>; Q_4=<5,2,4,3,6,1>; Q_5=<1,4,2,5,3,6>; Q_6=<5,6,3,4,2,1>. Для каждого решения Q_k рассчитаны оценки ξ_k и δ_k и сформирована индивидуальная россыпь альтернатив R_k. δ_1=6, δ_2=8, δ_3=4, δ_4=2, δ_5=9, δ_6=5.

Посредством объединения всех россыпей формируется интегральная россыпь альтернатив R альтернатив:

$$\begin{gathered} R=||r_{ij}||_{m\times m}=\cup R_k|k=1,2,\ldots,n_q\}; \\ R_k=||r_{kij}||_{m\times m}=\{R_{ki}|i=1,2,\ldots,n_a\}; \\ R_{ki}=\{r_{kij}|j=1,2,\ldots,n_p\}; \\ r_{kij}=\sum_k(r_{kij}),\ k=\{1,n_q\}. \end{gathered} \quad (1)$$

Фактически r_{kij} является суммарным значением полезностей решений, в которых агентом a_i была реализована альтернатива s_{ij}.

Произведем интеграцию решений и построим интегральную россыпь альтернатив множества решений $Q=\{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6\}$. Агентами в каждом из решений реализованы следующие альтернативы (табл. 1):

Таблица 1

Реализация альтернатив

Q	Агенты						δ
	a_1	a_2	a_3	a_4	a_5	a_6	
Q_1	2	5	1	3	6	4	δ_1=6
Q_2	2	4	1	6	3	5	δ_2=8
Q_3	2	4	1	6	3	4	δ_3=4
Q_4	5	2	4	3	6	1	δ_4=6
Q_5	2	4	2	5	3	6	δ_5=9
Q_6	5	6	3	4	2	1	δ_6=5

На рис. 2. представлены россыпи альтернатив $\{R_1, R_2, R_3\ R_3, R_5, R_6\}$ множества решений Q.

Произведем расчет значений элементов r_{ij} интегральной россыпи альтернатив по формуле 1. Интегральная россыпь альтернатив представлена на рис. 3.

R_1=

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	6	0	0	0	0
a_2	0	0	0	0	6	0
a_3	6	0	0	0	0	0
a_4	0	0	6	0	0	0
a_5	0	0	0	0	0	6
a_6	0	0	0	6	0	0

R_2 =

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	8	0	0	0	0
a_2	0	0	0	8	0	0
a_3	8	0	0	0	0	0
a_4	0	0	0	0	0	8
a_5	0	0	8	0	0	0
a_6	0	0	0	0	8	0

R_3=

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	4	0	0	0	0
a_2	0	0	0	4	0	0
a_3	4	0	0	0	0	0
a_4	0	0	0	0	0	4
a_5	0	0	4	0	0	0
a_6	0	0	0	0	4	0

R_4=

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	0	0	0	2	0
a_2	0	2	0	0	0	0
a_3	0	0	0	2	0	0
a_4	0	0	2	0	0	0
a_5	0	0	0	0	0	2
a_6	2	0	0	0	0	0

R_5=

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	9	0	0	0	0	0
a_2	0	0	0	9	0	0
a_3	0	9	0	0	0	0
a_4	0	0	0	0	9	0
a_5	0	0	9	0	0	0
a_6	0	0	0	0	0	9

R_6 =

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	0	0	0	0	5	0
a_2	0	0	0	0	0	5
a_3	0	0	5	0	0	0
a_4	0	0	0	5	0	0
a_5	0	5	0	0	0	0
a_6	5	0	0	0	0	0

Рис. 2. Россыпи альтернатив решений {R_1, R_2, R_3, R_4, R_5, R_6}

R=

P	p_1	p_2	p_3	p_4	p_5	p_6
a_1	9	18	0	0	7	0
a_2	0	2	0	21	6	5
a_3	18	9	5	2	0	0
a_4	0	0	8	5	9	12
a_5	0	5	13	0	0	8
a_6	7	0	0	6	12	14

Рис. 3. Интегральная россыпь альтернатив R

Работа алгоритма размещения базируется на использовании коллективной эволюционной памяти (КЭП), под которой подразумевается любой вид информации, которая отражает прошлую историю развития и хранится независимо от индивидуумов. Алгоритм, связанный с эволюционной памятью, стремится к запоминанию и многократному использованию способов достижения лучших результатов. КЭП алгоритма размещения представляет собой набор статистических показателей, отражающих для каждого фрагмента решения число δ, показывающее полезность фрагмента при построении решений на предыдущих итерациях алгоритма. Основу многоагентной системы составляет самоорганизация, обеспечивающая достижения общих целей коллектива на основе низкоуровневого взаимодействия. Коллектив не имеет централизованного управления, и его особенностями являются наличие непрямого обмена информацией. Непрямой обмен – стигмержи (stigmergy), представляет собой разнесённое во времени взаимодействие, при котором один агент изменяет некоторые области КЭП, а другие агенты позже используют уже измененную информацию в этих областях, когда в неё попадают [1, 2].

Разработанный алгоритм относится к классу популяционных алгоритмов. Всем элементам основной матрицы россыпи альтернатив R присваивается начальное значение оценки полезности δ. Процесс поиска решений итерационный. Каждая итерация l включает три этапа.

На первом этапе каждой итерации конструктивным алгоритмом (КА) формируется n_q решений Q_k. Работа конструктивного алгоритма базируется на базе показателей основной интегральной россыпи альтернатив – матрицы $R=||r_{ij}||_{m\times m}$, структура которой описана выше, и в которой хранятся интегральные показатели решений, полученных на предыдущих итерациях.

Формирование каждого решения Q_k выполняется посредством выбора каждым агентом a_i позиции p_j. Каждое решение Q_k является биективным отображением $F_k=A\rightarrow P$ и формируется путем последовательного назначения элементов в позиции [14]. Процесс назначения элемента в позицию включает две стадии. На первой стадии выбирается элемент, а на второй стадии − позиция p_j. При этом должно выполняться ограничение: каждому элементу множества A соответствует один единственный элемент множества P и наоборот. Рассчитывается оценка ξ_k решения Q_k и оценка полезности δ_k множества позиций P_k выбранных агентами в решении Q_k. Запоминание показателей построенных решений осуществляется в два этапа с использованием основной R и промежуточной R^* матриц россыпей альтернатив. Формируется набор $\Theta_k=\{r_{kij}\}$ ячеек матрицы R^*, соответствующих альтернативам, выбранным агентами, для отображения решения Q_k в матрице R^*. После построения очередного решения Q_k его результаты заносятся в промежуточную россыпь альтернатив − матрицу R^*: все элементы набора Θ_k в матрице R^* увеличиваются на величину δ_k. В работе используется циклический метод формирования решений. В этом случае наращивание оценок интегральной полезности δ_k в основной интегральной россыпи альтернатив R выполняется после полного формирования множества решений Q на итерации, и внесения показателей этих решений в промежуточную матрицу россыпи альтернатив R^*.

На втором этапе итерации производится наращивание оценок интегральной полезности δ_k в основной интегральной россыпи альтернатив – матрице R, путем добавления матрицы R^* к матрице R. На третьем этапе итерации осуществляется снижение оценок полезности δ_k интегральной россыпи альтернатив R на априори заданную величину δ^*. Работа алгоритма завершается после выполнения заданного числа итераций.

3. Базовый алгоритм размещения методом кристаллизации россыпи альтернатив. Предварительно формируется структура основной – R и промежуточной R^*матриц россыпей альтернатив.

Введем обозначения*:*

A(t) – множество агентов еще не размещенных в списке Q_k на шаге *t;*

P(t) – множество свободных позиций в списке n_p на шаге *t*;

N_l – число итераций;

n_k – число решений;

l – номер итерации;

k – номер решения*;*

ξ_{min} – минимальное значение оценки размещения Q_k*;*

δ – априорное (начальное) значение оценки полезности альтернатив.

1. Задаются параметры N_k, N_l, δ. $Q_{min}=\varnothing$. $\xi_{min}=1000$.

2. $l=1$.

3. Всем элементам основной матрицы россыпи альтернатив *R(l)* присваивается начальное значение оценки полезности – δ.

4. $k=1$.

5. Элементы промежуточной матрицы россыпи альтернатив *R*(l)* обнуляются.

6. *Формирование конструктивным алгоритмом решения Q_k на базе матрицы R(l).*

7. Список Q_k процедурой декодирования трансформируется в решение Q_k.

8. Расчет оценки ξ_k решения и оценки полезности δ_k решения Q_k.

9. Формирование набора Θ ячеек для отображения решения Q_k в матрице $R^*(l)$.

10. Если $\xi_k<\xi_{min}$, то $\xi_{min}=\xi_k$, $Q_{min}=Q_k$.

11. Все элементы набора Θ_k в матрице$R^*(l)$ увеличиваются на величину δ_k.

12. Если $(k<n_k)$, то $k=k+1$ и переход к 5, иначе переход к 13.

13. Сложение матриц $R(l)$ и $R^*(l)$. $R(l+1)=R(l)+R^*(l)$.

14. Снижение всех интегральных оценок полезности r_{ij} интегральной россыпи альтернатив $R(l)$ на величину δ^*.

15. Если $(l<N_l)$, то $l=l+1$ и переход к 4, иначе переход к 16.

16. Конец работы алгоритма. Фиксация и вывод лучшего решения.

4. Конструктивный алгоритм формирование решения Q_k на базе интегральной россыпи альтернатив *R(l)*.

1. Приводится в начальное состояние память:

$A(t)=A$;

$P(t)=P$;

$Q_k(t)=\varnothing$;

$t=1$.

2. Выбор очередного агента. Для каждого не размещенного $a_i \in A(t)$ в соответствующем векторе $R_i(l)$ –строке матрицы $R(l)$ отыскивается элемент r_{ij}^* с максимальным значением σ^*_i суммарной оценки полезности назначения a_i в p_j.

3. Среди множества не размещенных агентов $A(t)$ с максимальными значениями σ^*_i с вероятностью:

$$\Psi_i = \sigma^*_i / \sum_i (\sigma^*_i), (i|a_i \in A(t)),$$

пропорциональной σ^*_i, выбирается агент $a^*_i \in A(t)$.

4. На второй стадии – выбор позиции. Пусть μ_j значение суммарной оценки полезности назначения выбранного на первой стадии агента $a^*_i \in A(t)$ в позицию $p_j \in P(t)$. Среди множества свободных позиций $P(t)$ с вероятностью:

$$\Psi_j=\mu_j/\sum_j(\mu_j), (j|p_j \in P(t))$$

пропорциональной значению оценки полезности μ_j выбирается позиция p^*_j.

5. Агент a_i^* помещается в позицию p_j^* списка $Q_k(t)$.

6. Если $t<n_a$, то $t=t+1$ и переход к 6, иначе переход к 7.

7. $A(t)= A(t)\setminus a_i^*$; $P(t)= P(t)\setminus p_j^*$. Переход к 2.

8. Конец работы алгоритма.

5. Экспериментальные исследования. Для формирования тестов были использованы процедуры синтеза контрольных примеров с известным оптимумом F_{opt} по аналогии с известными методами PEKU (Placement Examples with Known Upper bounds on wirelength) [15–16].

Разработанный алгоритм CAF сравнивался с современными алгоритмами размещения KraftWerk [6], Capo 8.6 [21] и гибридным биоинспирированным трехуровневым алгоритмом ГТА [18].

Экспериментальные исследования проводились на наборе тестовых схем PEKU [23, 22]. Размещение оценивалось по суммарной длине соединений. Результаты экспериментов приведены в табл. 2.

Таблица 2

Результаты экспериментов

№ теста	Kraft Werk	Capo8.6	ГТА	CAF
	F, (e6)	F, (e6)	F, (e6)	F, (e6)
1	5,02	4,97	4,94	4,74
2	11.45	10,32	10,11	9,82
3	16,2	14,06	13,84	13,8
4	18,98	18,13	18,02	17,9
5	24,7	21,96	21,68	21,51
6	38,1	36,06	35,96	35.01
7	31,7	30,28	29,98	29.76
8	39,43	37,86	37,74	37,30
9	66,6	61,25	61,05	59,32
10	58,2	56,45	56,23	54,84

По представленным в таблице результатам можно заключить, что по качеству решения на тестовых примерах PEKU в среднем алгоритма CAF превосходит известные алгоритмы KraftWerk на 6 %, Capo8.6 на 4 % и ГТА) на 2 %.

Сравнительный анализ с другими алгоритмами размещения производился на стандартных тестовых примерах (бенчмарках) корпорации IBM [19, 20]. В этом случае сравнение разработанного алгоритма CAF с известными методами QPlace [22], mPG [23], A4 [7] и производилось на тестовых схемах корпорации IBM. Результаты экспериментов приведены в табл. 3. Алгоритм CAF в среднем превосходит программные средства Qplace, mPG и A4.

Таблица 3

Результаты сравнения

№ теста	QPlace	mPG	Ap1	CAF
	F, (e6)	F, (e6)	F, (e6)	F, (e6)
1	4,94	4,97	5,06	4,94
2	10,10	10,32	10,87	10,11
3	13,81	14,06	15,32	13,84
4	18,01	18,13	18,94	18,02
5	21,62	21,96	23,77	21,68
6	35,91	36,06	37,24	35,96
7	29,92	30,28	31,29	29,98
8	37,71	37,86	38,93	37,74
9	61,01	61,25	62,52	61,05
10	56, 22	56,47	57,38	56,26

Еще одной важной задачей проведенного тестового исследования является определение сходимости разработанного алгоритма. Тестирование определило, что разработанный алгоритм размещения методом кристаллизации россыпи альтернатив (**CAF**) в среднем имеет сходимость на 125 итерации. Проведенное тестирование показало, что вероятность того, что будет найдено оптимальное решение, в среднем равняется – 0.9. В заключении отметим, что предложенный алгоритм CAF способствует формированию решений, не уступающими полученным известным методами, а иногда качество решений выше в среднем на 6%.

Заключение. Задача размещения – является одной из широко востребованных задач синтеза топологии СБИС. В работе рассматривается поисковый популяционный алгоритм размещения компонентов СБИС. Лежащая в основе алгоритма метаэвристика кристаллизации россыпи альтернатив выполняет поиск решений с учетом коллективной эволюционной памяти (КЭП), под которой подразумевается информация, отражающая историю поиска решения и памяти поисковой процеду-

ры. Отличительной особенностью используемой метаэвристики является учет тенденции к использованию альтернатив из наилучших найденных решений. Предложены компактные структуры данных для хранения интерпретаций решений и памяти. Разработанный конструктивный алгоритм с временной сложностью *O(n)* для размещения интегрирован с итерационной структурой поиска. Алгоритм использует представление решения задачи размещения в виде упорядоченного списка. Переход от списка к решению задачи размещения производится с помощью декодера. Для перехода от упорядоченного списка к решению разработана процедура декодирования с временной сложностью *O(n).* Рассмотрены главные принципы анализа и выбора альтернатив в процессе адаптации популяции. К преимуществам алгоритма относится: невысокая трудоемкость, временная сложность алгоритма – $O(n^2)$; возможность диверсификации областей применения; алгоритм отличается быстрой сходимостью и высокой степенью эффективности.

Эффективность предложенного подхода подтверждена в процессе тестовых испытаний.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Карпенко А.П.* Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой: учеб. пособие. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2014. – 448 с.
2. *Лебедев О.Б.* Модели адаптивного поведения муравьиной колонии в задачах проектирования. – Таганрог. Изд-во ЮФУ, 2013. – 199 с.
3. *Лебедев Б.К., Лебедев О.Б., Лебединский А.Е.* Гибридный биоинспирированный алгоритм размещения базовых стандартных библиотечных элементов при проектировании топологии полузаказной СБИС // Известия ЮФУ. Технические науки. – 2019. – № 3 (205). – С. 97-110.
4. *Курейчик Вл.Вл., Курейчик В.В.* Интегрированный алгоритм размещения фрагментов СБИС // Известия ЮФУ. Технические науки. – 2014. – № 7 (156). – С. 84-91.
5. *Лебедев Б.К., Лебедев О.Б., Лебедев В.Б.* Гибридизация роевого интеллекта и генетической эволюции на примере размещения // Программные продукты, системы и алгоритмы. – 2017. – № 4.
6. *Caldwell A.E.* Can Recursive Bisection Alone Produce Routable Placements // DAC. – 2000. – P. 477-482.
7. *Mourelle M.* Swarm intelligent systems. – Berlin: Heidelberg: Springer Verlag, 2006. – 217 p.
8. *Лебедев Б.К., Лебедев В.Б.* Оптимизация методом кристаллизации россыпи альтернатив // Известия ЮФУ. Технические науки. – 2013. – № 7 (144). – С. 11-17.
9. *Лебедев Б.К., Лебедев В.Б.* Метод кристаллизации россыпи альтернатив // Сб. научных трудов VII Международной научно-практической конференции “Интегрированные модели и мягкие вычисления в искусственном интеллекте”. Т. 2. – М.: Изд-во Физматлит, 2013. – С. 903-912.
10. *Аюпов А.Б., Марченко А.М.* Метод оптимизации трассируемости в аналитическом алгоритме размещения // Информационные технологии. – 2008. – № 2. – С. 12-17.
11. *Норенков И.П.* Основы автоматизированного проектирования: учебник. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2006. – 336 с.
12. *Sherwani N.A.* Algorithms for VLSI Physical Design Automation // Third Edition, Kluwer Academic Publisher. – USA: 2013. – 572 p.
13. *Alpert C.J.* Handbook of Algorithms for Physical design Automation // Auerbach Publications Taylor & Francis Group. – USA, 2009. – 349 p.
14. *Лебедев Б.К., Лебедев В.Б., Лебедев О.Б.* Методы, модели и алгоритмы размещения: монография. – Ростов-на-Дону: Изд-во ЮФУ, 2015. –150 с.
15. *Cong J., Romesis M., Xie M.* UCLA Optimality Study Project. – http://cadlab.cs.ucla.edu/~pubbench. 2004.
16. *Cong J., Romesis M., Xie M.* Optimality, Scalability and Stability Study of Partitioning and Placement Algorithms // Proc. of the International Symposium on Physical Design. – Monterey, CA, 2003. – P. 88-94.

17. *Zaporozhets D.Y., Zaruba D.V., Kureichik V.V.* Hierarchical approach for VLSI components placement // Advances in Intelligent Systems and Computing. – 2015. – P. 79-87.
18. *Kureichik Vl., Kureichik V., Leschanov D., Zaruba D.* Hybrid approach for VLSI fragments placement // Advances in Intelligent Systems and Computing. – 2018. – P. 349-358.
19. IBM-PLACE 2.0 benchmark suits. – http://er.cs.ucla.edu/benchmarks/-ibm-place2/bookshelf/ibm-place2-all-bookshelf-nopad.tar.gz.
20. *Adya S.N.* ISPD02 IBM-MS Mixed-size Placement Benchmarks. – http://vlsicad.eecs.umich.edu/BK/ISPD02bench/.
21. *Roy J.A., Papa D.A., Markov I.L.* Capo: Congestion-Driven Placement for Standard-cell and RTL Netlists with Incremental Capability // In: Modern Circuit Placement. Series on Integrated Circuits and Systems. Springer, Boston, MA Springer Science + Business Media, LLC, 2007. – P. 123-146.
22. *Adya S.N.* Consistent placement of macro-blocks using floor planning and standard-cell placement // In Proc. Intl. Symp. on Physical Design. – 2002. – P. 12-17.
23. *Wang M., Yang X., Sarrafzadeh M.* Dragon 2000: Standard-cell Placement Tool for Large Industry Circuits. – ICCAD, 2000. – P. 260-263.

REFERENCES

1. *Karpenko A.P.* Sovremennye algoritmy poiskovoy optimizatsii. Algoritmy, vdokhnovlennye prirodoy: ucheb. posobie [Modern search engine optimization algorithms. Algorithms inspired by nature: textbook]. Moscow: Izd-vo MGTU im. N.E. Baumana, 2014, 448 p.
2. *Lebedev O.B.* Modeli adaptivnogo povedeniya murav'inoy kolonii v zadachakh proektirovaniya [Models of adaptive behavior of an ant colony in design problems]. Taganrog. Izd-vo YuFU, 2013, 199 p.
3. *Lebedev B.K., Lebedev O.B., Lebedinskiy A.E.* Gibridnyy bioinspirirovannyy algoritm razmeshcheniya bazovykh standartnykh bibliotechnykh elementov pri proektirovanii topologii poluzakaznoy SBIS [Hybrid bioinspired algorithm for placing basic standard library elements in the design of the topology of a semi-custom VLSI], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2019, No. 3 (205), pp. 97-110.
4. *Kureychik Vl.Vl., Kureychik V.V.* Integrirovannyy algoritm razmeshcheniya fragmentov CBIS [Integrated algorithm for placement of VLSI fragments], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2014, No. 7 (156), pp. 84-91.
5. *Lebedev B.K., Lebedev O.B., Lebedev V.B.* Gibridizatsiya roevogo intellekta i geneticheskoy evolyutsii na primere razmeshcheniya [Hybridization of swarm intelligence and genetic evolution on the example of placement], *Programmnye produkty, sistemy i algoritmy* [Software products, systems and algorithms], 2017, No. 4.
6. *Caldwell A.E.* Can Recursive Bisection Alone Produce Routable Placements, *DAC*, 2000, pp. 477-482.
7. *Mourelle M.* Swarm intelligent systems. Berlin: Heidelberg: Springer Verlag, 2006, 217 p.
8. *Lebedev B.K., Lebedev V.B.* Optimizatsiya metodom kristallizatsii rossypi al'ternativ [Optimization by the method of crystallization of alternatives field], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2013, No. 7 (144), pp. 11-17.
9. *Lebedev B.K., Lebedev V.B.* Metod kristallizatsii rossypi al'ternativ [The method of crystallization of a placer of alternatives], *Sb. nauchnykh trudov VII Mezhdunarodnoy nauchno-prakticheskoy konferentsii "Integrirovannye modeli i myagkie vychisleniya v iskusstvennom intellekte"* [Collection of scientific papers of the VII International scientific-practical conference "Integrated models and soft computing in artificial intelligence"]. Vol. 2. M.: Izd-vo Fizmatlit, 2013, pp. 903-912.
10. *Ayupov A.B., Marchenko A.M.* Metod optimizatsii trassiruemosti v analiticheskom algoritme razmeshcheniya [Traceability optimization method in the analytical placement algorithm], *Informatsionnye tekhnologii* [Information technologies], 2008, No. 2, pp. 12-17.
11. *Norenkov I.P.* Osnovy avtomatizirovannogo proektirovaniya: uchebnik [Basics of computer-aided design: textbook]. Mjscow: Izd-vo MGTU im. N.E. Baumana, 2006, 336 p.
12. *Sherwani N.A.* Algorithms for VLSI Physical Design Automation, *Third Edition, Kluwer Academic Publisher*. USA: 2013, 572 p.

13. *Alpert C.J.* Handbook of Algorithms for Physical design Automation, *Auerbach Publications Taylor & Francis Group*. USA, 2009, 349 p.
14. *Lebedev B.K., Lebedev V.B., Lebedev O.B.* Metody, modeli i algoritmy razmeshcheniya: monografiya [Placement methods, models and algorithms: monograph]. Rostov-on-Don: Izd-vo YuFU, 2015, 150 p.
15. *Cong J., Romesis M., Xie M.* UCLA Optimality Study Project. Available at: http://cadlab.cs.ucla.edu/~pubbench. 2004.
16. *Cong J., Romesis M., Xie M.* Optimality, Scalability and Stability Study of Partitioning and Placement Algorithms, *Proc. of the International Symposium on Physical Design*. Monterey, CA, 2003, pp. 88-94.
17. *Zaporozhets D.Y., Zaruba D.V., Kureichik V.V.* Hierarchical approach for VLSI components placement, *Advances in Intelligent Systems and Computing*, 2015, pp. 79-87.
18. *Kureichik Vl., Kureichik V., Leschanov D., Zaruba D.* Hybrid approach for VLSI fragments placement, *Advances in Intelligent Systems and Computing*, 2018, pp. 349-358.
19. IBM-PLACE 2.0 benchmark suits. Available at: http://er.cs.ucla.edu/benchmarks/-ibm-place2/bookshelf/ibm-place2-all-bookshelf-nopad.tar.gz.
20. *Adya S.N.* ISPD02 IBM-MS Mixed-size Placement Benchmarks. Available at: http://vlsicad.eecs.umich.edu/BK/ISPD02bench/.
21. *Roy J.A., Papa D.A., Markov I.L.* Capo: Congestion-Driven Placement for Standard-cell and RTL Netlists with Incremental Capability, *In: Modern Circuit Placement. Series on Integrated Circuits and Systems.* Springer, Boston, MA Springer Science + Business Media, LLC, 2007, pp. 123-146.
22. *Adya S.N.* Consistent placement of macro-blocks using floor planning and standard-cell placement, *In Proc. Intl. Symp. on Physical Design*, 2002, pp. 12-17.
23. *Wang M., Yang X., Sarrafzadeh M.* Dragon 2000: Standard-cell Placement Tool for Large Industry Circuits. ICCAD, 2000, pp. 260-263.

Статью рекомендовал 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; ттел.: 89085135512; кафедра систем автоматизированного проектирования; доцент.

Лебедев Владимир Борисович – Московский государственный технический университет имени Н.Э. Баумана; e-mail: lebedev.vlad.bor@mail.ru; 105005, г. Москва, ул. Бауманская 2-я, д. 5, стр. 1; тел.: 89287775005; Научно-производственное объединение «Новые технологии»; с.н.с.

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.

Lebedev Vladimir Borisovich – Moscow State Technical University named after N.E. Bauman; e-mail: lebedev.vlad.bor@mail.ru; 105005, Moscow, st. Baumanskaya 2-nd, 5, build. 1; phone: +79287775005; Research and Production Association "New Technologies"; senior researcher.