УДК 519.87

И.В. Козин, Е.В. Кривцун

Моделирование однослойных и двухслойных трассировок

Описана математическая модель многослойной трассировки на основе представления ее в виде допустимого слова в некотором конечном алфавите. Показана фрагментарная структура задачи поиска трассировки минимальной плотности. Наличие фрагментарной структуры позволяет для поиска приближенных решений использовать стандартную эволюционную модель на перестановках.

Ключевые слова: задача трассировки, плотность трассировки, комбинаторная оптимизация, фрагментарная структура, эволюционная модель.

Запропоновано математичну модель багатошарового трасування на основі представлення трасування у вигляді допустимого слова в деякому скінченному алфавіті. Показано фрагментарну структуру задачі пошуку трасування мінімальної щільності. Наявність фрагментарної структури дозволяє використовувати стандартну еволюційну модель на перестановках для пошуку наближених рішень.

Ключові слова: задача трасування, щільність трасування, комбінаторна оптимізація, фрагментарна структура, еволюційна модель.

Введение. *BGA*-корпуса (*BallGridArray*) – сегодня стандарт для размещения высокотехнологичных и сложных полупроводниковых приборов, таких как программируемая пользователем вентильная матрица (ППВМ), *FPGA* и микропроцессоры. Трассировка печатных плат в области высокой плотности размещения контактов (*BGA*-компонента) – важная и сложная техническая задача [1]. При использовании стандартных методов САПР часто возникают проблемы блокировки контактных площадок и кратных пересечений проводников. Основные конструктивно-технологические ограничении – плотность распределения проводников, количество слоев, количество межслойных переходов и др. Предлагается метод глобальной трассировки, позволяющий устранить указанные проблемы.

Постановка задачи

Рассмотрим задачу построения однослойных и двухслойных трассировок на плате с матрично расположенными контактами с конструктивным ограничением на плотность распределения проводников. Как показано в [2], эта задача может быть сведена к задаче трассировки на квадрате, который представляет собой верхнюю левую четверть платы.

Пусть на плоскости задана квадратная целочисленная решетка размером $n \times n$, где n – количество *узлов* (контактов) на стороне квадрата. На рис. 1 приведен пример такой решетки с $n = 5$. Назовем единичные отрезки, соединяющие узлы решетки *звеньями*, а единичные квадраты решетки, – *клетками*. *Граница решетки* – это узлы и звенья левой и верхней сторон квадрата. Граница решетки размером $n \times n$ образует n-й *уровень*. Остальные уровни решетки определим рекурсивно по следующему правилу. Удаляя границу решетки размером $(k+1)\times(k+1)$, звенья, инцидентные узлам границы и содержание прилегающих к границе клеток, получаем решетку размером $k \times k$, граница которой образует k-й уровень (рис. 1).

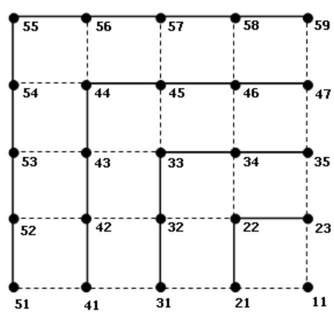

Рис. 1. Решетка размером 5×5

Клеткам решетки, находящимся между узлами уровня k и $k+1$ припишем уровень k. Таким образом, существует ровно $2k-1$ клеток k-го уровня, $k = \overline{1, n-1}$.

***Определение* 1**. *Трассировка* – набор линий, соединяющих некоторые узлы решетки (*нача-*

ла) с точками на границе решетки и не выходящих за пределы решетки. Каждой такой линии соответствует начало и последовательность звеньев решетки, которые она пересекает. Узлы n-го уровня также отнесем к линиям трассировки. Линии, у которых начала и последовательности прохождения звеньев решетки одинаковы, будем считать *эквивалентными*. Трассировки, состоящие из попарно эквивалентных линий, назовем *эквивалентными*.

***Определение* 2**. *Допустимой* трассировкой на решетке размером $n \times n$ назовем трассировку, при которой:

- при движении вдоль линии от начала к границе решетки, номер уровня проходимых клеток не понижается;
- линия трассировки пересекает каждое звено решетки не более одного раза.

Линии трассировки могут пересекаться, а линия трассировки соединяет узел решетки с одной из точек границы, которую назовем *точкой выхода* соответствующего узла.

***Определение* 3**. Допустимые трассировки назовем *слабо эквивалентными*, если порядок следования точек выхода узлов на границе решетки у этих трассировок одинаков.

***Определение* 4**. Трассировку назовем *однослойной*, если существует слабо эквивалентная ей трассировка, линии которой не пересекаются.

***Определение* 5**. Трассировку назовем *m-слойной*, если ее можно представить в виде объединения m однослойных трассировок.

На рис. 2 приведена реализация двухслойной трассировки.

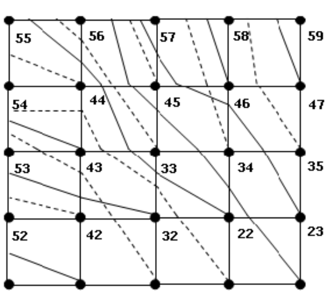

Рис. 2. Двухслойная трассировка

Кодирование трассировок с помощью слов

Рассмотрим алфавит $\Gamma_n = \{a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots, a_{n,1}, \ldots, a_{n,2n-1}\}$, $n = 1, 2, \ldots$. Буквы, первый индекс которых начинается с числа k, назовем буквами k-го уровня. Таким образом, существует одна буква первого уровня – $a_{1,1}$, три буквы второго уровня и т.д. Количество букв k-го уровня равно $2k-1$.

***Определение* 6**. *Допустимым* словом в алфавите $\Gamma_n = \{a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots, a_{n,1}, ..., a_{n,2n-1}\}$ будем называть любой упорядоченный набор букв этого алфавита, в котором ни одна из букв не встречается более одного раза и имеет место канонический порядок букв n-го уровня, т.е. буквы этого уровня встречаются в слове в порядке $a_{n,1}, a_{n,2}, ..., a_{n,2n-1}$. Кроме того, каждое допустимое слово начинается буквой $a_{n,1}$ и заканчивается буквой $a_{n,2n-1}$. Очевидно, каждому допустимому слову соответствует некоторая трассировка. В дальнейшем рассматриваются только допустимые слова.

Длиной слова назовем количество входящих в него букв.

***Определение* 7**. Слово A будем называть *полным*, если оно содержит все буквы алфавита, причем каждая из букв встречается только один раз.

Таким образом, длина любого полного слова равна $1 + 3 + ... + (2n-1) = n^2$.

Спектром слова A назовем последовательность слов

$$A_n = A, \quad A_{n-1} = A_n \setminus \{a_{n,j}\}_{j=1}^{2n-1},$$

$$A_{n-2} = A_{n-1} \setminus \{a_{n-1,j}\}_{j=1}^{2n-3}, \ldots, A_0 = A_1 \setminus \{a_{1,1}\} = \varnothing.$$

***Определение* 8**. Полное слово A будем называть 1-*правильным*, если для любого $k \in \{1, 2, \ldots, n\}$ слово A_k из его спектра допустимо в алфавите Γ_k.

Очевидно, что если слово A есть 1-правильным, то каждое слово A_k $(1 \le k \le n)$) из его спектра также 1-правильное в алфавите Γ_k.

***Определение* 9**. *Подсловом* слова A будем называть слово, получаемое из A путем удаления некоторых букв. При этом порядок следования оставшихся букв не меняется. Например, слово $a_{2,1}, a_{2,3}, a_{3,5}$ есть подсловом слова $a_{1,1}$, $a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, a_{3,4}, a_{3,5}$.

***Определение* 10**. Пусть B – подслово слова A. *Дополнением* B в A будем называть слово, получаемое из A путем удаления всех букв, содержащихся в слове B. Дополнение B в A будем обозначать B-A.

***Определение* 11**. Слово A назовем *монотонным*, если каждая буква алфавита входит в это слово не более одного раза и последовательность букв каждого уровня встречается в слове в порядке возрастания второго индекса.

Очевидно, что каждое 1-правильное слово и любое его подслово монотонны.

***Определение* 12**. Слово B будем называть *полуправильным*, если существует такое 1-правильное слово A, для которого слово B – это подслово.

***Определение* 13**. Полное слово A назовем 2-*правильным*, если существует его полуправильное подслово B, дополнение которого A-B также – полуправильно.

Реализации слова

Каждый узел решетки размером $n \times n$ обозначим соответствующей буквой из набора $a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots, a_{n,1}, \ldots, a_{n,2n-1}$, где первый индекс означает номер уровня, а второй порядковый номер узла на соответствующем уровне. Тогда с точностью до слабой эквивалентности каждая допустимая трассировка может быть задана допустимым словом в алфавите $\Gamma_n = \{a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots, a_{n,1}, \ldots, a_{n,2n-1}\}$, Наоборот, каждое допустимое слово задает трассировку на решетке $n \times n$ единственную с точностью до слабой эквивалентности. Другими словами, допустимое слово определяет класс слабо эквивалентных между собой трассировок. Каждую трассировку – элемент этого класса, будем называть *реализацией* соответствующего слова.

***Определение* 14**. Трассировку назовем *полной*, если слово, представляющее ее, полное, т.е. в полной трассировке на границу выводятся все узлы решетки.

Пусть задана допустимая трассировка на целочисленной решетке размером $n \times n$. Тогда очевидно, что эта трассировка порождает допустимые трассировки и на любой решетке размером $k \times k$, состоящей из клеток уровня от единицы до $k-1$, $k = \overline{1, n-1}$. Трассировку, порождаемую на уровне с номером k, назовем *подтрассировкой уровня* k. Очевидно, что подтрассировка уровня n совпадает с исходной.

Теорема 1. Каждая полная однослойная трассировка определяется 1-правильным словом с точностью до эквивалентности, и наоборот, каждое 1-правильное слово задает полную однослойную трассировку [3].

На решетке размером $n \times n$ рассмотрим некоторую реализацию T слова A. Обозначим через $B_k(T), k = \overline{1, n}$, слово, порожденное k-уровневой подтрассировкой реализации T. Тогда всегда выполняются равенства $B_n(T) = A_n \equiv A$, $B_1(T) = A_1 = a_{11}$. Следовательно, слова спектра A_k будут не равны словам $B_k(T)$. Можно показать, что среди множества реализаций слова A существует подмножество таких трассировок, для которых

$$B_k \setminus \{a_{k,1}, a_{k,2}, \ldots, a_{k,2k-1}\} = A_{k-1}, \quad \text{для всех } k = \overline{1, n-1}, \tag{1}$$

т.е. слово B_k, соответствующее подтрассировке уровня k, будет отличаться от элемента спектра A_k лишь местами расположения букв k-го уровня.

На рис. 3 изображены две слабоэквивалентные трассировки, являющиеся реализациями слова $A = a_{kj} a_{ki}$. Для реализации T' выполняется условие (1), а для реализации T не выполняется.

Любая трассировка на решетке размером $n \times n$ – это реализация некоторого слова в алфавите Γ_n. Рассмотрим реализацию допустимого слова A, в которой две заданные линии пересекаются. Пересечение двух линий назовем *устранимым*, если существует такая реализация слова A, в которой эти линии не пересекаются. Соответственно, пересечение линий будем называть *неустранимым*, если не существует реализации допустимого слова A, в которой эти линии не пересекаются. На рис. 3 приведен пример неустранимого пересечения, а на рис. 4 –

пример устранимого пересечения. Все приведенные трассировки – допустимы.

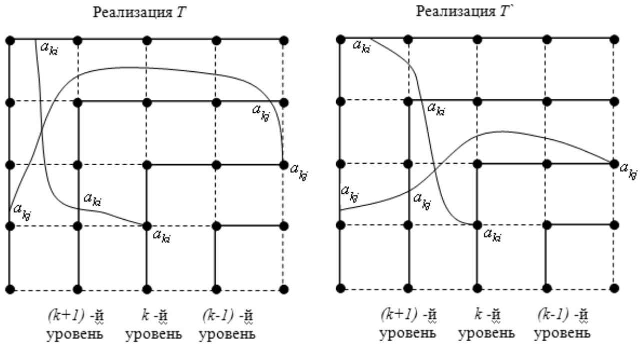

Рис. 3. Смещение пересечения линий k-го уровня в клетки k-го уровня

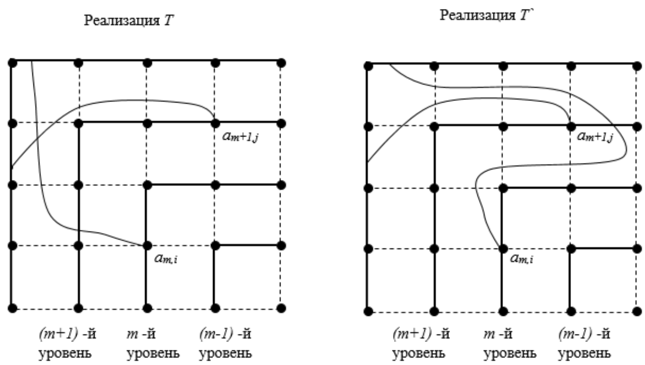

Рис. 4. Устранимое пересечение (слева) линий разных уровней на двух реализациях слова $A = a_{m+1,j}\, a_{m,i}$

На рис. 3 видно, что в реализации допустимого слова A любые пересечения линий одного уровня неустранимы, поскольку в допустимой трассировке номер уровня клеток при движении вдоль линии (т.е. от начала к точке выхода) не может понижаться. При этом пересечению двух линий соответствует инверсия вторых индексов соответствующих букв в слове A. В то же время пересечение любых двух линий разных уровней, выходящих из внутренних узлов, устранимо (рис. 4), так как всегда можно построить такую реализацию слова A, в которой эти линии не пересекаются (через одно звено может проходить сколько угодно линий), хотя при этом могут появиться новые пересечения этих линий с другими линиями реализации.

Пусть A – допустимое слово. Разложим его на подслова $\tilde{N}_k$, $k = 1, 2, ..., n$, содержащие только буквы с первым индексом, равным k. Рассмотрим слова с $k = 2, ..., n-1$, заменим в них каждую букву на ее второй индекс. Получим числовые последовательности N_k, $k = 2, ..., n-1$.

Теорема 2. Если в любой последовательности N_k, $k \in \{2, ..., n-1\}$ можно выделить возрастающую подпоследовательность такую, что ее дополнение тоже возрастает, то слово A 2-правильно.

Доказательство. Неустранимым пересечениям линий трассировки соответствуют буквы одного уровня слова A, у которых вторые индексы представляют собой инверсии. Отсюда следует, что минимальное количество слоев, необходимых для разнесения пересекающихся линий трассировки, равно максимальному числу максимальных возрастающих подпоследовательностей, найденному на множестве последовательностей N_k, $k \in \{2, ..., n-1\}$. Соответственно, если каждая из последовательностей $N_k, k \in \{2, ..., n-1\}$ делится не больше, чем на две максимальные возрастающие подпоследовательности, то необходимо два слоя, в каждом из которых получаем однослойную трассировку.

Теорема 3. Каждому 2-правильному слову соответствует множество двухслойных трассировок (реализаций данного слова).

Доказательство.

Шаг 1. Разложим 2-правильное слово на два полуправильных подслова. Дополним каждое подслово до 1-правильного слова произвольным образом.

Шаг 2. По двум полученным 1-правильным словам однозначно строим две однослойные трассировки. Это возможно сделать согласно теореме 1.

Шаг 3. В каждой трассировке оставляем только те линии $l_{i,j}$, для которых в соответствующем подслове есть буквы $a_{i,j}$. Таким образом, будут получены два слоя двухслойной трассировки.

Поскольку на первом шаге два 1-правильных слова определялись неоднозначно, то двухслойная трассировка также неоднозначно определяется по 2-правильному слову. Описанная процедура лежит в основе приведенного ниже фрагментарного алгоритма построения правильных слов и их реализаций.

Из теоремы 3 следует *основной результат раздела*: каждая двухслойная трассировка есть реализация 2-правильного слова.

Плотность реализации

Рассмотрим некоторую двухслойную трассировку *T* на решетке размером $n \times n$, которая есть реализацией 2-правильного слова *A*. Назовем клетки решетки по правому нижнему узлу. Обозначим через P_{ij} количество линий обоих уровней, проходящих через клетку $a_{ij}, i=\overline{1,n-1}$, $j=\overline{1,2i-1}$. Это число назовем *плотностью клетки* в реализации *T*.

Определение 15. *Плотностью реализации T на решетке размером $n \times n$ слова A будем называть число $P_n(A,T)$, равное максимальной из плотностей клеток решетки в реализации T*:

$$P_n(A,T) = \max_{i,j}\{P_{ij}\}, \quad i=\overline{1,n-1},\ j=\overline{1,2i-1}.$$

Тогда требование равномерности сводится к задаче

$$P_n(A,T) \to \min$$

на множестве реализаций 2-правильных слов в алфавите Γ_n.

Фрагментарная модель и генерация правильных слов

Рассмотрим вопрос об описании фрагментарной модели трассировки. Сначала рассмотрим модель для однослойной трассировки. Такая трассировка определяется 1-правильным словом в алфавите $\Gamma_n = \{a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots$ $\ldots, a_{n,1}, \ldots, a_{n,2n-1}\}$.

Используем буквы алфавита как элементарные фрагменты фрагментарной структуры [4]. Пусть задана некоторая перестановка букв s_{n^2}. Задача – построить алгоритм формирования 1-правильного слова из этой перестановки.

Пусть на начальном шаге формируемое слово *A* пусто. На очередном шаге алгоритма будем просматривать слово, определяемое перестановкой s_{n^2}, слева направо и искать первую букву a_{ij}, удовлетворяющую одному из следующих *условий присоединения*:

- $i=n, j=1$;
- $i<n, j=1$ и в слове *A* уже присутствуют буквы $a_{n,1}, a_{n-1,1}, \ldots, a_{i+1,1}$;
- $i>1, j=2i-1$ и в слове *A* уже присутствуют буквы $a_{i,2j-2}$ и $a_{1,1}, a_{2,3}, \ldots, a_{i-1,2i-3}$;
- $j>1, j<2i-1$ и в слове *A* уже присутствует буква $a_{i,j-1}$.

Хотя бы одно из перечисленных условий всегда выполняется. Найденная буква a_{ij} добавляется в конец слова *A* и удаляется из слова s_{n^2}. Затем шаг алгоритма повторяется до тех пор, пока в слове s_{n^2} не останется букв. Таким образом, построение слова *A* заканчивается через n^2 шагов.

Генерация 1-правильных слов осуществляется по следующей схеме. Сначала генерируется случайная перестановка из n^2 букв алфавита. Затем применяется рассмотренный фрагментарный алгоритм.

Рассмотрим теперь фрагментарную модель и алгоритм генерации 2-правильных слов. В качестве элементарных фрагментов выберем буквы из алфавита $\Gamma_n = \{a_{1,1}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, \ldots$ $\ldots, a_{n,1}, \ldots, a_{n,2n-1}\}$, а также из дублей этого алфавита, $\Gamma'_n = \{a'_{1,1}, a'_{2,1}, a'_{2,2}, a'_{2,3}, a'_{3,1}, \ldots, a'_{n,1}, \ldots, a'_{n,2n-1}\}$ и $\Gamma''_n = \{a''_{1,1}, a''_{2,1}, a''_{2,2}, a''_{2,3}, a''_{3,1}, \ldots, a''_{n,1}, \ldots, a''_{n,2n-1}\}$. Обозначим знаком ω_{ij} одну (любую) из двух букв a_{ij} или a'_{ij}. Общее количество букв этого *утроенного* алфавита равно $3n^2$. Допустимыми фрагментами есть все слова, которые можно дополнить до 2-правильного слова дописыванием букв.

Пусть задана некоторая перестановка букв всех трех алфавитов s_{3n^2}. Задача – построить алгоритм формирования 2-правильного слова из этой перестановки.

Пусть на начальном шаге слова *A* и *B* пусты. На каждом очередном шаге алгоритма будем просматривать слово, определяемое перестановкой s_{3n^2}, слева направо и искать первую букву $\omega_{i,j}$, удовлетворяющую одному из следующих условий присоединения:

- $i = n, j = 1$;
- $i < n, j = 1$ и в слове A уже присутствуют буквы $\omega_{n,1}, \omega_{n-1,1}, ..., \omega_{i+1,1}$;
- $i > 1, j = 2i - 1$ и в слове A уже присутствуют буквы $\omega_{i,2j-2}$ и $\omega_{1,1}, \omega_{2,3}, ..., \omega_{i-1,2i-3}$;
- $j > 1, j < 2i - 1$ и в слове A уже присутствует буква $\omega_{i,j-1}$.

Если хотя бы одно из перечисленных условий выполняется, то буква ω_{ij} добавляется в конец слова A и из слова s_{3n^2} удаляются буквы $a_{i,j}$ и $a'_{i,j}$.

Параллельно с этим по аналогичному алгоритму строится слово B. Ищется первая буква $a''_{i,j}$, удовлетворяющая одному из следующих условий присоединения:

- $i = n, j = 1$;
- $i < n, j = 1$ и в слове B уже присутствуют буквы $a''_{n,1}, a''_{n-1,1}, ..., a''_{i+1,1}$;
- $i > 1, j = 2i - 1$ и в слове B уже присутствуют буквы $a''_{i,2j-2}$ и $a''_{1,1}, a''_{2,3}, ..., a''_{i-1,2i-3}$;
- $j > 1, j < 2i - 1$ и в слове B уже присутствует буква $a''_{i,j-1}$.

Если хотя бы одно из перечисленных условий выполняется, то буква $a''_{i,j}$ добавляется в конец слова B и удаляется из слова s_{3n^2}. Затем шаг алгоритма повторяется. Через n^2 шагов будут построены два 1-правильных слова A и B, описывающие однослойные трассировки на решетках $n \times n$. Время работы данного алгоритма есть $O(n^4)$.

Слово A состоит из букв двух видов, а слово B – указатель, который будет определять порядок следования букв второго вида в слове A. Изменим слово A так, чтобы оно стало 2-правильным словом, описывающим двухслойную трассировку на решетке $n \times n$. Для этого сначала удалим из него все буквы алфавита Γ'_n (за исключением букв n-го уровня). Затем будем добавлять в слово A соответствующие буквы алфавита Γ''_n из слова B, т.е. такие $a''_{i,j}$, индексы которых совпадают с индексами удаленных $\omega_{i,j}$. Относительное расположение добавляемых букв сохраняется таким же, как в слове B. Причем, если добавляемая буква $a''_{i,j}$ находилась в слове B между буквами $a''_{n,k}$ и $a''_{n,k+1}$, то в слово A она будет помещаться между $\omega_{n,k}$ и $\omega_{n,k+1}$. При этом остается неоднозначность в порядке следования новых и старых букв, находящихся между символами $\omega_{n,k}$ и $\omega_{n,k+1}$. Избавиться от этой неоднозначности можно, например, приписав буквам каким-либо образом веса. Предлагается перед добавлением букв из слова B в слово A определить веса букв следующим образом: вес буквы $\omega_{i,j}$ слова A, находящейся между символами $\omega_{n,k}$ и $\omega_{n,k+1}$, равен отношению количества букв от $\omega_{n,k}$ до этой буквы (включая $\omega_{i,j}$) к общему количеству букв между $\omega_{n,k}$ и $\omega_{n,k+1}$ (включая $\omega_{n,k+1}$). Аналогично, по отношению к $\omega_{n,k}$ и $\omega_{n,k+1}$, определяются веса добавляемых букв $a''_{s,k}$. Затем буквы располагаются в слове A между символами $\omega_{n,k}$ и $\omega_{n,k+1}$ в порядке возрастания веса. В случае совпадения весов приоритет имеют буквы $\omega_{i,j}$.

Полученное в результате использования алгоритма слово есть 2-правильным словом и описывает двухслойную трассировку в квадратной целочисленной решетке на плоскости размером $n \times n$.

Таким образом, определено отображение множества перестановок S_{3n^2} в множество 2-правильных слов.

Эволюционно-фрагментарная модель и результаты работы ЭВФ алгоритма

На основе фрагментарной модели построена стандартная эволюционно-фрагментарная модель [3]. Базовое множество эволюционной модели – множество перестановок S_{3n^2}, в котором каждая перестановка соответствует определенному порядку следования фрагментов. В качестве оператора кроссовера выбирается любой бинарный оператор $K : S_{3n^2} \times S_{3n^2} \to S_{3n^2}$, сохраняющий порядок следования элементов в перестановке. Оператор мутации данной модели осуществляет транспозицию двух случайных элементов перестановки.

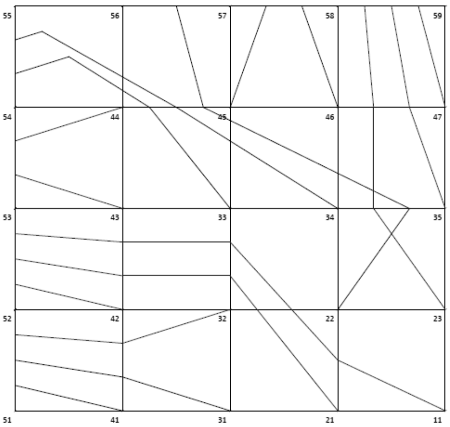

Рис. 5. Решение с $P_n(A,T) = 3$ для двухслойной трассировки на решетке размером 4 × 4.

Заключение. Предлагаемый в статье подход позволяет применить универсальный метод поиска приближенных решений на основе фрагментарно-эволюционной модели к задачам многослойной трассировки. В перспективе эволюционно-фрагментарный алгоритм позволяет значительно ускорить поиск приемлемых реализаций трассировки, а также учесть многочисленные ограничения, возникающие при практических построениях многослойных трассировок.

1. *Лузин С.*Ю., *Полубасов О.Б.* САПР *TopoR*: трассировка печатных плат с *BGA*-компонентами // Современная электроника. – 2008. – № 7 – С. 44–49. – *http*://*www.soel.ru*/ *issues*/ ?*id*=343867
2. *Трассировка* подключающей пластины многозондового устройства контроля *BGA*-компонентов / И.В. Жарикова, С.В. Курапов, И.Ш. Невлюдов и др. – Вісн. Запорізьск. нац. ун-ту. Фізико-математичні науки. – 2013. – № 2. – С. 28–36.
3. *Козин И.В*., *Кривцун Е.В*., *Пинчук В.П.* Эволюционно-фрагментарная модель задачи трассировки // Кибернетика и системный анализ. – 2015. – Т. 51, № 3 – С. 125–131.
4. *Козин И.В.*, *Полюга С.И.* О свойствах фрагментарных структур // Вісник Запорізького нац. ун-ту: Зб. наук. праць. Фізико-математичні науки. – Запоріжжя: ЗНУ, 2012. – № 1. – С. 99–106.

Поступила 22.01.2016

Тел. для справок: +38 061 228-7641, 234-5738, 213-4169 (Запорожье)

E-mail: *ainc*00*@gmail.com, kryvtsun@ukr.net*

UDC 519.87

I.V. Kozin, O.V. Kryvtsun

Modelling of the Single-Layer and Two-Layer Routing Problems

Keywords: routing problem, routing solution density, combinatorial optimization, fragmentary structure, evolutional model.

The printed circuit board (PCB) routing in a contact high-density region is an important and difficult technical task. The main structural and technological constraints are the density distribution of the conductors, the number of layers, the number of vias and others.

The single-layer and two-layer routing problems on the PCB with a matrix arrangement of the contacts are considered. The construction of the feasible routing solution set mapping into the set of code words over a finite alphabet is described. The properties of the set of code words are investigated. The concepts of full word, 1-regular and 2-regular words are introduced. The routing algorithm for code word realization is designed. It is shown that every full single-layer routing solution is determined by 1-regular word up to equivalence, and, conversely, the realization of each 1-regular word defines a full single-layer routing solution. It is established that the two-layer routing realization set corresponds to each 2-regular word. It is well founded that any full two-layer routing solution is a 2-regular word realization.

It is grounded that both 1-regular and 2-regular word sets can be represented as a maximal fragment set of some fragmentary structure. The 1-regular and 2-regular word generation algorithms are introduced.

The concept of routing solution density is presented and the problem of the minimal density routing solution search as the combinatorial optimization problem on the set of feasible words is formulated. Based on the fragmentary representation, the evolutionary-fragmentary model of the routing problem, where search space is a permutation set, is proposed. As the crossover operator in the model, the binary operator, which keeps the sequence of the elements in permutations is selected. The mutation operator in this model implements transposition of two random elements in the permutation. The proposed approach allows to use the universal method of the approximate optimal solutions searching, based on evolutionary-fragmentary model, for the routing problems.

●