УДК 519.8 doi: 10.15421/321812

И.В. Козин, С.Е.Батовский
Запорожский национальный университет

ГЕНЕРАЦИЯ СЛУЧАЙНЫХ КАРТ ПРЯМОУГОЛЬНОГО РАСКРОЯ

Исследуется проблема генерации безотходных карт прямоугольного раскроя. Для различных видов карт исследуются соответствующие модели и алгоритмы генерации. Предложены генераторы для безотходных карт раскроя на прямоугольники ограниченных размеров, генераторы карт гильотинного раскроя.

Ключевые слова: карта раскроя, безотходный прямоугольный раскрой, алгоритмы генерации прямоугольного раскроя, гильотинный раскрой.

Досліджується проблема генерації безвідходних карт прямокутного розкрою. Для різних видів карт досліджуються відповідні моделі і алгоритми генерації. Запропоновано генератори для безвідходних карт розкрою на прямокутники обмежених розмірів, генератори карт гільйотинного розкрою. Відомо, що велика кількість прикладних оптимізаційних задач на сьогоднішній день не можуть бути точно розв'язані, оскільки їх обчислювальна складність відноситься до класу NP-важких. У багатьох випадках для пошуку наближених рішень використовуються метаевристики різних типів, але при виборі тієї чи іншої метаевристики залишається відкритим питання про якість обраного методу. Пропонується кілька можливих варіантів розв'язання проблеми, одним з яких є перевірка метаевристичних алгоритмів на прикладах з відомих тестових бібліотек з відомими рекордами. Іншим підходом до вирішення проблеми оцінки якості алгоритмів, що розглядається, є порівняння «нового» алгоритму з іншими алгоритмами, робота яких вже детально вивчена. Побудова генератора випадкових завдань з відомим оптимальним рішенням, може вирішити проблему отримання «середніх» оцінок точності використовуваного алгоритму в порівнянні з іншими методами. У роботі розглядається побудова генераторів випадкових повних карт прямокутного розкрою з урахуванням певних обмежень на розміри складових їх елементарних прямокутників. Існування наборів таких карт формує базу тестових завдань для перевірки якості наближених алгоритмів пошуку оптимального розкрою. Прямокутний розкрій, який розглядається в роботі, також є основою для побудови розкроїв з використанням більш складних фігур. В якості найпростішого методу генерації випадкових прямокутних карт розкрою наводиться метод, який використовує гільйотинний розкрій. Також, наводиться більш складний алгоритм генерації випадкового прямокутного розкрою, робота якого полягає в генерації випадкової точкової сітки і видалення деяких випадкових точок з цієї сітки. Велика увага приділяється саме реалізаціям наведених методів, оскільки основною метою статті є застосування генераторів на практиці. Всі наведені алгоритми вже використовуються в програмній системі тестування еволюційно-фрагментарних алгоритмів для різних класів оптимізаційних задач на графах.

Ключові слова: карта розкрою, безвідходний прямокутний розкрій, алгоритми генерації прямокутного розкрою, гільйотинний розкрій.

It is known that a large number of applied optimization problems can't be exactly solved nowadays, because their computational complexity is related to the NP-hard class. In many cases metaheuristics of various types are used to search for approximate solutions, but the choice of the concrete metaheuristic has open question of the quality of the chosen method.

There are several possible solutions to this problem, one of which is the verification of metaheuristic algorithms using examples from known test libraries with known records. Another approach to solving the problem of evaluating the quality of algorithms is to compare the "new" algorithm with other algorithms, the work of which has already been investigated. The construction a generator of random problems with a known optimal solution can solve the problem of obtaining "average" estimates of the accuracy for used algorithm in comparison with other methods.

The article considers the construction of generators of random non-waste maps of rectangular cutting with restrictions on the rectangles of limited sizes. The existence of sets of such cards forms the basis of test problems for checking the quality of approximate algorithms for searching for optimal solution. Rectangular cutting, which is considered in the article, is also the basis for building cuts using more complex shapes. As the simplest method of generating random rectangular non-waste maps, considered a method that uses guillotine cutting. Also, a more complex algorithm for generating a random rectangular cut is given, whose job is to generate a random dot grid and remove some random points from this grid. Much attention is paid to the implementation of the above methods, since the main purpose of the article is to simplify using of generators in practice.

All the above algorithms are already used in the software system for testing evolutionary-fragmentary algorithms for various classes of optimization problems on the graphs.

Keywords: cutting map, wasteless rectangular cutting, rectangular cutting generation algorithms, guillotine cutting.

Введение. Большое количество прикладных оптимизационных задач на сегодняшний день не могут быть точно решены, поскольку их вычислительная сложность относится к классу NP-трудных [1].

Однако потребность в решении таких задач возрастает и, соответственно, увеличивается поток работ, посвященных поискам приближенных решений трудных задач. Во многих случаях для поиска приближенных решений используются метаэвристики различных типов. При использовании метаэвристик открытым остается вопрос о качестве предлагаемого метода. Поскольку эвристические алгоритмы не имеют априорных оценок сходимости, то неизвестно насколько полученное приближённое решение близко к оптимальному. Как следствие, очень трудно оценить целесообразность применения алгоритма в различных оптимизационных задачах.

Одним из возможных решений данной проблемы может служить проверка метаэвристических алгоритмов на примерах из известных тестовых библиотек с известными рекордами, и попытка улучшить эти рекорды (ORLIB [2]).

Другим подходом к решению проблемы оценки качества алгоритмов является сравнение «нового» алгоритма с другими алгоритмами, работа которых уже детально изучена. Такое сравнение должно проходить на большой базе тестовых задач со случайным набором начальных данных. Это

позволит вычислить «средние» сравнительные характеристики эффективности тестируемого алгоритма. Если удается построить генератор случайных задач с известным оптимальным решением, то можно получить «средние» оценки точности используемого алгоритма в сравнении с другими методами. Таким образом, задача генерации тестовых индивидуальных задач для массовых NP-трудных задач безусловно является актуальной.

Обзор имеющихся результатов. Задача раскроя впервые сформулирована Канторовичем в 1939 году [3]. В 1951 году, ещё до того, как компьютеры стали широко доступны, Л.В. Канторович и В.А. Залгаллер предложили [4] способ решения задачи экономного использования материала при раскрое с помощью линейного программирования.

Большинство существующих точных методов решения задачи прямоугольного негильотинного раскроя сводятся к перебору всего множества допустимых решений. Методы улучшенного перебора объединены для этих задач под названием «метода ветвей и границ». В 1977 г. И.В. Романовским представлена общая идея переборного метода для решения общей экспоненциальной задачи, и предложена его конкретизация в виде метода «ветвей и границ» для решения задач упаковки [5]. В дальнейшем метод получает развитие за счет введения процедур сокращения перебора в работах В.М. Картака и Э.А. Мухачевой [6, 7]. Независимо за рубежом выходит серия статей S. Martello и P. Toth, посвященная разработке улучшенных версий метода ветвей и границ [8].

Большинство существующих комбинаторных методов решения задач прямоугольного негильотинного раскроя и размещения объектов сложных форм сводятся к перебору всего множества допустимых решений. Методы улучшенного перебора объединены и для этих задач под названием «метода ветвей и границ». В 1986 г. Ю.Г. Стоян и С.В. Яковлев описали общую схему метода и привели основные алгоритмы [9]. Другой подход к переборным методам решения разработан в середине 80-х г. А.И. Липовецким [10]. На основе понятия «зоны» доказывается, что для любой упаковки прямоугольников можно указать такой их порядок, при котором каждый следующий прямоугольник не пересекается ни с одной из зон предыдущих (топологическая сортировка). Метод зон реализован в 1988 г., усовершенствован и исследован в 2001 г. В.В. Бухваловой [11].

Тестовые примеры для задачи прямоугольного раскроя можно найти в [12]. Ряд генераторов тестовых примеров приведены в [13].

Формулировка целей. Целью настоящей работы является построение генераторов случайных полных карт прямоугольного раскроя с учетом определенных ограничений на размеры составляющих их элементарных прямоугольников. Наборы таких карт могут в последствии выступать в качестве базы тестовых задач для проверки качества приближенных алгоритмов поиска оптимального раскроя. Прямоугольный раскрой также является основой для построения раскроев с использованием более сложных фигур. В этом случае «сложные» фигуры вписываются в некоторые

прямоугольники, из которых генерируется начальный прямоугольный раскрой.

Постановка задачи прямоугольного раскроя. Рассмотрим набор прямоугольников, заданных своими размерами (h_i, w_i), $i = 1, 2, ..., n$. Допустимым решением считается такое размещение всех этих прямоугольников (или некоторых из них) на прямоугольном листе размером $H \times W$ при котором внутренности прямоугольников попарно не пересекаются. Каждое допустимое решение задается набором индексов $I \in \{1, 2, ..., n\}$ и набором пар координат $\{(x_i, y_i)\}_{i \in I}$. Каждая пара определяет положение левого верхнего угла размещаемого прямоугольника. Допустимое решение задачи будем называть картой раскроя.

Пусть I – множество индексов, соответствующих допустимому решению задачи. Тогда задача поиска оптимального решения состоит в поиске такого подмножества индексов I, для которого будет минимальной разность $HW - \sum_{i \in I} h_i w_i \to \min$. Установить оптимальность той или иной карты раскроя достаточно сложно. Однако существует случай, когда проблема оптимальности карты имеет очевидное решение. Если в карте раскроя множество индексов задает все прямоугольники $I = \{1, 2, ..., n\}$ и при этом выполняется условие $HW - \sum_{i \in I} h_i w_i = 0$, то карту раскроя будем называть полной картой. Очевидно, что полная карта раскроя соответствует оптимальному решению задачи.

Генерация случайных полных карт прямоугольного раскроя. Рассмотрим некоторые из методов генерации случайных прямоугольных раскроев, которые могут быть взяты в основу генератора.

Наиболее простой случайный раскрой может быть получен с помощью алгоритма генерации гильотинного раскроя [11,14]. Суть такого раскроя заключается в том, что каждый разрез выполняется строго от одного края листа до другого, параллельно одной из его сторон.

Алгоритм работает рекурсивно. Глубина рекурсии n, а также размеры $W \times H$ исходного листа f_0 определяются заранее. На очередном шаге для фрагмента f_i^k случайным образом выбирается одно из направлений разреза: вертикальное или горизонтальное. Затем генерируется случайное число α из интервала $(0, 1)$. При этом, вертикальный разрез выполняется на расстоянии $\alpha \times W$ от левого края листа, а горизонтальный – на расстоянии $\alpha \times H$ от верхнего. Каждый из полученных фрагментов f_{i+1}^1 и f_{i+1}^2, так же рекурсивно разрезается, если для него выполняется условие: $i + 1 < n$.

Пример работы данного алгоритма для $n = 4$ представлен ниже (рис. 1).

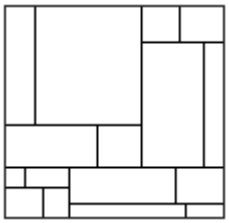

Рис. 1. Пример гильотинного раскроя.

Одна из вариаций данного алгоритма упраздняет выбор направления разреза, генерируя сразу два числа α и β, и рассекая очередной фрагмент сразу на четыре части. Структура раскроя при этом становится более однообразной (рис. 2).

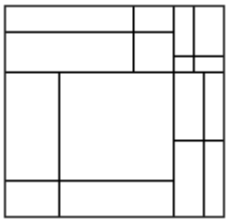

Рис. 2. Шестнадцать фрагментов, полученные другой вариацией гильотинного раскроя.

Существует более сложный алгоритм генерации случайного прямоугольного раскроя, работа которого состоит из нескольких этапов:

1. генерации случайной сетки из точек, которая будет покрывать исходный лист;
2. удаления некоторых точек сетки случайным образом;
3. преобразования полученной сетки в соответствующий прямоугольный раскрой.

Перед началом работы алгоритма определяются размеры $W \times H$ исходного листа, а также, генерируются параметры m и n случайной точечной сетки, на которую условно разбивается исходный лист (рис. 3).

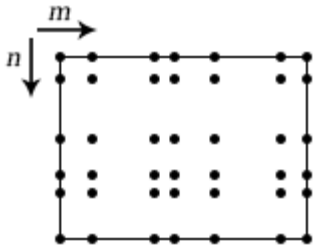

Рис. 3. Пример случайной сетки из точек, которой покрывается исходный лист.

Параметр $m\left(m \geq 4\right)$, при этом, задаёт количество точек сетки вдоль одной стороны листа, а параметр $n\left(n \geq 4\right)$ – вдоль другой. Крайние точки сетки всегда располагаются по краю листа.

В процессе генерации случайной точечной сетки заполняются два массива $a\left[i\right]$, $i = \overline{1,m}$ и $b\left[j\right]$, $j = \overline{1,n}$ случайными числами в возрастающем порядке. При этом, произвольная точка сетки будет задана парой чисел $\left(x, y\right)$, где

$x \in a[i],\ y \in b[j]$. Ниже приведена реализация данного процесса для массива $a[i]$ (лист. 1).

```
var a[m];

a[0] = 0;
for (int i = 1; i < m; i++)
{
   a[i] = a[i - 1] + random(0, 1);
}

for (int i = 1; i < m; i++)
{
   // Примечание. Для массива b используется listHeight.
   a[i] = listWidth * (a[i] / a[m - 1]);
}
```

Лист. 1. Процесс генерации массива интервалов сетки.

Применяя данный подход, вероятность получить ту или иную точечную сетку одинакова среди всех возможных точечных сеток с такими же значениями параметров m и n.

Каким бы странным это не показалось, но удаление случайных точек из полученной сетки можно выполнять ещё до её генерации, поскольку процесс зависит только от входных параметров m и n.

В следующем процессе используется матрица $s[i, j]$ специального вида размером $m \times n$, элементы которой могут принимать только три значения: 0, 1 и 2. Каждый элемент матрицы соответствует единственной точке на полученной сетке. Если элемент матрицы принимает значение 0, то соответствующая ему точка должна быть удалена из сетки. Значение элемента равное 2 запрещает удаление соответствующей точки, а значение равное 1 является значением по умолчанию, и означает, что манипуляции с точкой ещё не были проведены. Крайние точки сетки всегда изначально зафиксированы и не могут быть удалены. Ниже представлена реализация процесса инициализации данной матрицы (лист 2).

После инициализации алгоритм случайным образом перебирает все элементы матрицы. Если очередной элемент матрицы принимает значение 1, то соответствующая ему точка помечается как удалённая и значение элемента сбрасывается в 0, при этом соседние по диагонали элементы матрицы получают значение 2, а соответствующие им точки защищаются от удаления (лист. 3).

Последовательность перебора элементов матрицы $s[i, j]$ наиболее эффективно может быть представлена случайной комбинаторной перестановкой из $m \times n$ элементов, которая генерируется уже известным тасованием Фишера-Йетса [15]. Равновероятная случайная перестановка обеспечивает равную вероятность получения случайной матрицы после выполнения процесса.

Каждый элемент перестановки однозначно раскладывается на два индекса соответствующего элемента матрицы с помощью операций целочисленного деления и получения остатка от деления.

```
var s[m, n];

for (int i = 0; i < m; i++)
{
   for (int j = 0; j < n; j++)
   {
      if ((i == 0) or (i == m - 1) or (j == 0) or (j == n - 1))
      {
         s[i, j] = 2;
      }
      else
      {
         s[i, j] = 1;
      }
   }
}
```

Лист. 2. Процесс инициализации матрицы.

```
for (int i = 0; i < (m * n); i++)
{
   int x = permutation[i] % m; // a % b - остаток от деления a на b
   int y = permutation[i] / n; // a / b - целочисленное деление a на b

   if (s[x, y] == 1)
   {
      s[x, y] = 0;
      s[x - 1, y - 1] = 2;
      s[x - 1, y + 1] = 2;
      s[x + 1, y - 1] = 2;
      s[x + 1, y + 1] = 2;
   }
}
```

Лист. 3. Процесс удаления точек сетки.

На последнем этапе алгоритма сведения, которые были получены на первых двух этапах, объединяются и преобразовываются в удобный формат. Они могут быть использованы для быстрого визуального отображения раскроя, преобразованы в множество узловых точек, преобразованы в множество прямоугольных фрагментов и т.д., например, визуальный раскрой может быть построен, если каждую пару соседних зафиксированных точек сетки соединить прямой линией.

Несколько примеров работы алгоритма представлены ниже. Слева изображён результат, полученный для малых значений m и n (рис. 3а), а справа – наоборот, для больших (рис. 3б).

Для некоторых практических задач недостатком алгоритма может считаться наличие вытянутых прямоугольников в большинстве раскроев. Для сглаживания таких результатов существует небольшая модификация

алгоритма, суть которой состоит в том, чтобы не выполнять процесс удаления случайных точек сетки до конца. В этом случае доля перебираемых элементов матрицы $s[i, j]$ задаётся некоторым параметром $\alpha \in [0,1]$, тем самым используя только первые $\alpha(m \times n)$ элементов перестановки, задающей порядок перебора элементов.

а) **б)**

Рис. 3. Примеры работы алгоритма.

Сравнить результат работы модифицированного алгоритма (рис. 4б) с работой исходного алгоритма (рис. 4а), можно задав одинаковый порядок перебора элементов матрицы.

а) **б)**

Рис. 4. Сравнение результатов исходного и модифицированного алгоритмов.

На рисунке (рис. 4) можно видеть, что многие большие вытянутые фрагменты ещё не успели образоваться в силу неполного перебора элементов. Однако, значение параметра α близкое к нулю даёт случайный раскрой очень близкий к начальной сетке. В связи с этим, на практике более оправдано использование значения $\alpha \in [0.5, 1]$.

Выводы. В данной статье были рассмотрены некоторые алгоритмы генерации случайных графов, а также, наиболее простые методы генерации случайных прямоугольных раскроев. Большое внимание было уделено именно реализациям данных методов, поскольку идея статьи основана на практическом применении алгоритмов. Большинство из них уже используются в программной системе тестирования эволюционно-фрагментарных алгоритмов для различных классов оптимизационных задач на графах.

Библиографические ссылки

1. **Гэри М.** Вычислительные машины и труднорешаемые задачи [Текст] / М. Гэри, Д. Джонсон; пер. А. Фридман. – М.: Мир. – 1982. – 416 с.
2. Библиотека тестовых задач OR-Library [Электронный ресурс]. – Режим доступа: http://people.brunel.ac.uk/~mastjjb/jeb/info.html. Заголовок с экрана.
3. **Канторович Л.В.** Mathematical methods of organizing and planning production [Текст] / Л.В. Канторович // Management Science. – issue 6 (4). – 1960. – P. 366-422.
4. **Канторович Л.В.** Рациональный раскрой промышленных материалов [Текст] / Л.В. Канторович, В.А. Залгаллер. – Новосибирск: Наука. – 1971.
5. **Романовский И.В.** Алгоритмы решения экстремальных задач [Текст] / И.В. Романовский. – М.: Наука. – 1977.
6. **Мухачева Э.А.** Рациональный раскрой прямоугольных листов на прямоугольные заготовки [Текст]: сб. науч. тр. / Э.А. Мухачева // Оптимальное планирование. – вып. 6. – 1966. – С. 43-115.
7. **Мухачева Э.А.** Задача прямоугольной упаковки: методы локального поиска оптимума на базе блочных структур [Текст] / Э.А. Мухачева, А.С. Мухачева // Автомат. и телемех. – 2004. – № 2. – С. 101-112.
8. **Martello S.** Knapsack Problems: Algorithms and Computer Implementations [Текст] / S. Martello, P. Toth. – Wiley: Chichester. – 1990.
9. **Стоян Ю.Г.** Математические модели и оптимизационные методы геометрического проектирования [Текст] / Ю.Г. Стоян, С.В. Яковлев. – Киев: Наук. думка. – 1986. – 268 с.
10. **Липовецкий А.И.** К оптимизации свободного размещения прямоугольников [Текст] / А.И. Липовецкий // Автоматизация проектирования в машиностроении. – Минск: ИТК АН БССР. – 1985. – С. 80-87.
11. **Бухвалова В.В.** Задача прямоугольного раскроя: метод зон и другие алгоритмы [Текст] / В.В. Бухвалова. – СПб: СПбГУ. – 2001. – 96 с.
12. Тестовые примеры. Упаковка кругов и прямоугольников [Электронный ресурс]. – Режим доступа: http://www.math.nsc.ru/AP/benchmarks/Packing/pack_test1.html. Заголовок с экрана.
13. ESICUP [Электронный ресурс]. – Режим доступа: https://www.fe.up.pt/esicup. Заголовок с экрана.
14. **Morabito M.** Staged and constrained two-dimensional guillotine cutting problems: an and/or-graph approach [Текст] / M. Morabito, M. Arenales // European Journal of Operational Research. – 1996. – P. 548-560.
15. **Кнут Д.** Искусство программирования, том 2. Получисленные методы / Д. Кнут. – М.: «Вильямс». – 3-е изд. – 2007. – 832 с.

Надійшла до редколегії 31.05.2018.