Оценка алгоритмов для решения задачи одномерного раскроя

Miro Gradišar
Факультет Организационных Наук, Университет Марибора, Словения

Gortan Resinovič
Факультет Экономики, Университет Любляны, Словения

Резюме

Данная работа посвящена исследованию проблемы оценки и сравнения различных алгоритмов решения задачи одномерного раскроя с минимальными потерями материала. Выделено три возможных типа задач. Рассмотренный метод оценки алгоритмов позволяет произвести сравнение решений выделенных типов задач. Приведен практический пример использования данного метода для сравнения двух алгоритмов.

Ключевые слова: Раскрой; Эвристика; Оптимизация; Оценка

1 Введение

Задача одномерного раскроя встречается во многих производственных процессах [3,7,8,11,12]. Это и привлекло в течение прошлых нескольких лет все увеличивающееся внимание исследователей со всех континентов [1,13]. Большинство стандартных задач одномерного раскроя, как известно, является NP-трудными. Однако во многих случаях задачи раскроя могут быть представлены моделью математического программирования, и решение может быть найдено с помощью приблизительных методов и эвристик.

Задачи раскроя могут классифицироваться согласно Dyckhoff [2]. Схема классификации проблем раскроя была развита им во многих различных публикациях. Он классифицировал данную проблему, используя четыре характеристики: *размерность*, *вид назначения*, *ассортимент раскраиваемых полос* и *ассортимент заказанных длин*.

Dyckhoff разделяет решение проблем раскроя материалов на две группы: раскрой, *ориентированный на раскраиваемую полосу*, и раскрой, *ориентированный на заказанную длину*.

Подход, ориентированный на полосу, характеризуется индивидуальной обработкой каждой раскраиваемой полосы.

При *подходе, ориентированном на заказанную длину*, сначала заказанные длины заготовок объединяются в группы, для которых в дальнейшем определяется интенсивность – количество, необходимое для удовлетворения требований заказчика. В этом случае главным образом используется классический метод Gilmore, основанный на линейном программировании (LP) и метод "последующей генерации столбцов" Gomory [5,6] или любой другой метод на основе LP (LPM) [14,15,16]. Применение LPM возможно только в случае, когда все раскраиваемые полосы материала имеют одинаковую длину [5] или имеется несколько групп полос стандартных длин. Это означает, что все раскраиваемые листы в пределах одной группы идентичны [6]. В случаях, когда различные длины полос встречаются с различными частотами, такой подход не применим, и поэтому решение может быть найдено только при ориентации на полосу [8,9]. Таким образом, в случаях, когда полосы раскраиваемого материала имеют различные длины, возможны две различные ситуации и, следовательно, два различных подхода решения: один из них применяется, если есть несколько групп полос стандартных длин, другой – когда все полосы различны по длине. Это важное разделение не учтено в схеме классификации Dyckhoff, так как там рассматриваются только три возможных варианта для ассортимента раскраиваемых полос:

(O) Одна раскраиваемая полоса.

(I) Большое количество идентичных полос.

(V) Полосы различных длин.

Поэтому мы предлагаем дополнительный вариант:

(G) Несколько групп идентичных раскраиваемых полос.

Расширенная схема классификации Dyckhoff имеет следующий вид:

1. Размерность
   (N) Число измерений
2. Вид назначения
   (B) Все длинные полосы раскраиваются на выборочные заказанные длины.
   (V) Выборочные полосы раскраиваются на все заказанные длины.
3. Ассортимент раскраиваемых полос
   (O) Одна раскраиваемая полоса.
   (I) Большое количество идентичных полос.
   (G) Несколько групп идентичных полос.
   (V) Различные по длине полосы.
4. Ассортимент заказанных длин
   (F) Несколько различных заказанных длин.
   (M) Большое количество различных заказанных длин.
   (R) Большое количество заказов на несколько групп длин.
   (C) Большое количество идентичных заказанных длин.

На основе обоих подходов (ориентированном на заказанную длину и ориентированном на полосу) разработано множество алгоритмов и методов одномерного раскроя, учитывающих различные факторы и критерии. Поэтому провести их общее сравнение очень трудно, а иногда и невозможно. Но если положить, что *потери материала* – самый важный общий критерий, то можно решить эту проблему, ограничивая сравнение только критерием минимизацией потерь. Мы решили разработать метод оценки алгоритмов одномерного раскроя материалов именно относительно потерь материала. Описание такого метода пока не было найдено в литературе. В [4] описан метод, который позволяет сравнивать только ориентированные заказанную длину алгоритмы.

В статье приведено определение потерь материала при различных подходах и для различных типов раскроя. Определены параметры задачи. Разработан метод оценки, учитывающий данные параметры. В конце статьи представлено практическое применение метода.

2 Определение потерь материала

В различных задачах раскроя материалов и различных подходах к их решению по-разному определяются потери материала. Если мы хотим сравнить различные методы решения относительно потерь, то мы должны сначала ввести их общее определение для всех типов задач. Такое общее определение потерь материала возможно при исследовании *общей задачи одномерного раскроя* (General One-Dimensional Cutting Stock Problem , G1D-CSP). G1D-CSP определяется следующим образом:

Для выполнения каждого заказа имеется достаточно широкий ассортимент раскраиваемых полос материала (O или I, или G, или V). Необходимо раскроить определенное число полос материала на необходимое число заказанных длин. При нехватке материала, всегда имеется возможность автоматически или вручную относить необходимое количество некоторых заказанных длин к доступному количеству материала [9]. Мы рассматриваем длины как целые числа. Если они первоначально не являются целыми числами, мы предполагаем, что всегда возможно преобразовать их в целые числа путем умножения на некоторые коэффициенты. Введем обозначения:

li - заказанные длины заготовок; i=1,...,n.
di - требуемое количество заготовок заказанной длины li.
Uj - длины раскраиваемых полос; j=1,...,m.
pij - число заготовок длины li, раскроенных из полосы j-й длины.

Задача может быть представлена следующей моделью целочисленного программирования [9]:

(1) $\min \sum_{j=1}^{m} t_j$ (минимум остатков материала, который меньше либо равен max li).

(2) $U_j - \sum_{i=1}^{n} l_i \cdot p_{ij} = \delta_j \quad \forall j$ (ограничение "ранца")

(3) $\sum_{j=1}^{m} p_{ij} = d_i \quad \forall\ i$ (ограничение требования - должно быть получено точно заказанное количество заготовок каждой длины)

(4) $\sum_{j=1}^{m} w_j \le 1$ (максимальное количество остаточных длин, больших max li)

$$p_{ij} \ge 0, \text{ integer} \qquad \forall\ i, j$$
$$t_j \ge 0 \qquad \forall\ j$$
$$\delta_j \ge 0, \text{ integer} \qquad \forall\ j,$$

здесь используется следующие функции:

$$z_j = \begin{cases} 0, \text{ если } p_{ij} = 0 & \forall\ i \\ 1 \text{ в противном случае} \end{cases}$$

для указания использования полосы j -й длины в раскройном плане;

$$w_j = \begin{cases} 1, \text{ если } z_j = 1 \wedge \delta_j > \max l_i \\ 0 \text{ в противном случае} \end{cases}$$

для указания, больше ли остаток материала от полосы j -й длины максимальной длины заказанной заготовки;

$$t_j = \begin{cases} \delta_{j,} \text{ если } z_j = 1 \wedge \delta_j \le \max l_i \\ 0 \text{ в противном случае.} \end{cases}$$

tj указывает размер остатка от полосы j-й длины.

Верхний предел (Upper bound, UB) для остаточной длины может быть установлен между 0 и максимальной длиной заготовки li (условие(4)). max li выбран как UB, чтобы гарантировать, что все остаточные длины материала в результате условия(3) будут учтены при оптимизации. Остаточная длина полосы материала, которая является больше max li, может использоваться в дальнейшем и не рассматривается как потеря материала. Если получаются остаточные длины большие, чем max li, то всегда существует возможность повторно комбинировать заготовки заказанных длин таким способом, что останется только одна остаточная длина полосы материала, большая чем max li [Gradisar 1,2]. Чтобы гарантировать это, необходимо выполнение условия (4). Без его выполнения, в случае достаточно длинной раскраиваемой полосы, раскройный план будет безостаточным только при некотором увеличении полосы [8].

Для решения G1D-CSP необходим подход, ориентированный на полосу, потому что все полосы могут иметь различные длины. Существует два варианта: применение точных методов (метод «ветвей и границ», динамическое программирование) или алгоритмы приближения в форме *последовательной эвристической процедуры* (*Sequential Heuristic Procedure*, SHP). SHP имеет лучшую надежность в широком диапазоне случаев.

Если полосы материала имеют одинаковую длину или делятся на несколько групп стандартных длин, мы получаем *стандартную одномерную задачу раскроя* (*Standard One-Dimensional Cutting Stock Problem* , S1D-CSP), где возможен подход, ориентируемый на заказанную длину [4,10]. Задача типа S1D-CSP может быть представлена следующим образом:

li – заказанные длины; i=1,..., n.
di – требуемое количество заготовок заказанной длины li.
Lk – длины раскраиваемых полос; k = 1,..., p,
где p - количество групп стандартных длин полос материала.

Раскройный план содержит шаблоны раскроя и их интенсивность (частоты), обеспечивающие выполнение заказа. Шаблон получения заказанной длины ***c*** из полосы ***k***-й длины может быть описан вектором

$$(a_{1ck}\ ,\ a_{2ck}\ ,\ a_{3ck}\ ,...,\ a_{nck}\ ) \qquad (5)$$

что соответствует

$$\sum_{i=1}^{n} l_i \cdot a_{ick} \le L_k\ , \qquad (6)$$

$a_{ick} \geq 0$, целое, (7)
где $\boldsymbol{a_{ick}}$ определяет число экземпляров i-й заказанной длины в данном шаблоне.

Обозначим:

$\boldsymbol{X_{ck}}$ - интенсивность (частота) заказанной длины c в раскрое полосы k-й длины,
$\boldsymbol{t_k}$ - общее количество заготовок (5), получаемых при раскрое полосы k-й длины согласно (6) и (7).

Получим следующую формулировку задачи целочисленного программирования:

$$\min \sum_{k=1}^{p} \sum_{c=1}^{t_k} x_{ck} \cdot L_k \quad \text{(минимум расхода материала)} \qquad (8)$$

$$\sum_{k=1}^{p} \sum_{c=1}^{t_k} a_{ick} \cdot x_{ck} \geq d_i \quad i = 1,..., n$$ (количество заготовок может превышать требуемое) (9)

$$x_{ck} \geq 0, \text{ целое, } \ c = 1,..., t_k; \ k = 1,..., p.$$ (10)

Задача типа S1D-CSP может быть решена с помощью классического алгоритма Gilmore на основе LP и метода "последующей генерации столбцов" Gomory или любым другим методом линейного программирования (LPM). При подходе, ориентированном на заказанную длину объектом оптимизации является не сумма потерь, а сумма длин полос материала, участвующих в раскрое (8). Это означает, что существует два различия по критерию потерь материала между G1D-CSP и S1D-CSP:

1. В G1D-CSP заказанные длины раскроены точно в требуемом количестве. В S1D-CSP их число может быть больше требуемого (условие 9).

2. Последняя полоса материала в G1D-CSP вообще раскраивается не полностью. В результате получается остаточная длина, которая может использоваться позже. В S1D-CSP все используемые полосы раскраиваются полностью.

Эти два отличия делают невозможным прямое сравнение G1D-CSP и S1D-CSP по критерию минимальных потерь материала. Для такого сравнения, мы должны изменить или G1D-CSP, или S1D-CSP. Различия проявляются также в ограничениях LPM, который является наиболее часто используемым методом решения для S1D-CSP. При изменении G1D-CSP LPM в общем случае все еще не может использоваться, потому что длины полос материала могут быть различными. Поэтому мы решили изменить S1D-CSP. Если мы введем описанные различия как дополнительные ограничения в S1D-CSP, то получим новый тип задачи, которая не может быть решена лишь с помощью метода, ориентируемого на заказанную длину. Однако она решается при некоторой комбинации этого метода с методом, ориентированным на полосу. Поэтому эту новую задачу можно назвать г*ибридной задачей 1-мерного раскроя* (*Hybrid 1-Dimensional Cutting Stock Problem*, H1D-CSP). H1D-CSP вводится не только для определения потерь материала, что позволяет производить сравнение с G1D-CSP, но и из-за новых свойств полученного решения:

1. Способность раскраивать полосы точно на число заготовок заказанных длин, необходимое для выполнения заказа.

2. Способность последовательно накапливать остаточные длины материала в одной части, которая может использоваться в дальнейшем.

3. Способность использовать полосы материала нестандартных длин.

Мы полагаем, что эти дополнительные особенности могут быть полезными во многих практических ситуациях. Пример метода, основанного на гибридном подходе для решения H1D-CSP, представлен в [10].

3 Метод

В литературе нами не был найден метод оценки G1D-CSP или H1D-CSP. Gau и Waesher [4] описали постановку задачи для S1D-CSP с I-типом ассортимента раскраиваемых полос. Они выделяют 5 параметров задачи:

m – размерность задачи,
L – стандартная длина полосы,
v1,v2 – нижняя и верхняя границы заказанных длин,
đ – среднее требуемое значение заказанных длин.

Наша цель состоит в том, чтобы обобщить их метод для любого типа задач относительно ассортимента раскраиваемых полос. Поэтому нам необходимо ввести еще пять параметров задачи, которые позволят определить также различные стандартные и нестандартные длины полос материала. Они представлены в таблице 1:

n	количество различных длин заготовок
v1, v2	нижняя и верхняя границы длин заготовок, т.е. $v1 \le li \le v2$ ($i = 1,...,n$)
đ	среднее требуемое значение длины заготовки
p	количество стандартных длин раскраиваемых полос
s1, s2	нижняя и верхняя границы стандартных длин полос, т.е. $s1 \le Lk \le s2$ ($k=1,...,p$)
m	количество нестандартных длин раскраиваемых полос
u1, u2	нижняя и верхняя границы нестандартных длин полос, т.е. $u1 \le Uj \le u2$ ($j = 1,...,m$)

Таблица 1: Параметры задачи для G1D-CSP.

Метод состоит из следующих основных шагов:

1. Инициализация значений параметров задачи:
    - инициализация значений заказанных длин заготовок и требований
    - определение стандартных длин полос раскраиваемого материала
    - определение нестандартных длин полос раскраиваемого материала

2. Если алгоритм является решением задачи типа S1D-CSP, то задача должна быть преобразована в H1D-CSP [10]

    - для решения S1D-CSP применяется LPM
    - частоты (интенсивности) шаблонов заказанных длин, которые превышают требования, уменьшаются на 1
    - недостающее количество заготовок заказанных длин выкраивается из остаточной длины с помощью SHP

    иначе

    - решение задачи

3. Анализ результатов.

3.1 Определение значений параметров задачи:

Исследование алгоритма требует входных данных, которые могут быть сгенерированы согласно параметрам задачи как случайная выборка одной или более тестовых задач. В нашем случае единственная тестовая задача может быть описана, как 2n+p+m–мерный вектор случайных переменных:

(l1,…, ln, d1,…, dn, L1,…, Lp, U1,…, Um). (11)

Случайные переменные генерируются с помощью универсального генератора случайных чисел. С разрешения авторов мы использовали генератор псевдо-случайных чисел, описанный в [4], т.к. он обладает такими свойствами, как простота, хаотичность, мобильность и воспроизводимость. Ядро генератора не фиксировано заранее. Поэтому, при тестировании алгоритмов результаты должны всегда содержать информацию о ядре, которое использовалось в каждом тестовом эксперименте, что позволит другим воспроизвести тестовые данные.

Все длины (заказанных заготовок, стандартные и нестандартные), которые являются целыми числами, определены через следующие две процедуры:

- генерация реализации yi случайной переменной Y , равномерно распределенной на интервале (b1, b2+1), где b1 представляет нижнюю, а b2 - верхнюю границу:

  yi = b1 + (b2 - b1) * randi (0, 1) + randi (0, 1),
  (12)

- ее округление вниз до ближайшего целого.

Если сгенерированы две идентичные длины заказанной заготовки, то вторая длина удаляется после того, как ее заказ будет прибавлен к заказу на первую длину.

Требование di для различных заказанных длин заготовок li (i=1,…,n) определяются в случае, когда общее требование (заказ) D = n*đ равномерно распределен между заказанными длинами заготовок, согласно следующим правилам [4]:

di = max {1, [đ'i + 0.5] },
(13)

$$d_i = \max \{1, \lfloor đ'_i + 0.5 \rfloor \}, \tag{13}$$

$$đ'_i = \frac{\text{rand}_i (0, 1)}{\sum_{k=1}^{n} \text{rand}_k (0, 1)} \cdot D \qquad i = 1,..., n - 1, \tag{14}$$

$$d_n = \max \{1, D - \sum_{k=1}^{n-1} d_i \}. \tag{15}$$

Требования đ'i , кроме требования đ'n , корректируются прибывлением 0.5 для предотвращения систематической недооценки требований di , кроме dn. Иначе остаточное требование dn было бы систематически завышено.

4. Практический пример

Предложенным методом решен практический пример со сравнением двух алгоритмов . Данный подход позволяет произвести испытание и получить экспериментальные данные для всех трех типов задач одномерного раскроя. Как ма могли удебиться ранее, сравнение между различными методами относительно остатков материала возможно, только если задача определена, как G1D-CSP или H1D-CSP. Если задача определена как S1D-CSP, то она должна быть преобразована к H1D-CSP. Поэтому мы решили сравнить два ориентируемых на полосу решения G1D-CSP. Первая зазача называется COLA и предназначета для производства одежды [8]. Вторая, CUT [9] обобщенная и улучшенная версия COLA.

Тестирование этих задач были проведены со следующими значениями параметров:

- определение заказанных длин заготовок и требований на них:
  Параметрам задачи были присвоены различные значения n (n = 5, 10, 15), v1 и v2 (v1 = 100 и v2 = 200, v1 = 200 и v2 = 600, v1 = 300 и v2 = 900) и đ (đ = 10, 20, 30). Путем различных комбинаций этих параметров были получены 27 задач.
- определение стандартных длин полос материала:

В нашем случае стандартные длины полос не рассматривались, и все три параметра: p, s1 и s2 - были установлены в 0.

- определение нестандартных длин полос материала:

  Количество нестандартных длин полос m может варьироваться в пределах от 10 до 90, нижняя граница длины,s1 - от 500 до 1500, верхняя граница, s2 - от 1500 до 4500.

Описание инициализации параметров задачи и определение ядра последовательности исследования задач, представлены в виде программной схемы процедуры PROGEN.

Процедура PROGEN:

```
for i = 1 to 3 do
for j = 1 to 3 do
        for k = 1 to 3 do
                n ← i·5
                v1 ← j·100
                v2 ← j·300
                p ← 0
                s1 ← 0
                s2 ← 0
                đ ← k·10
                m ← i·j·10
                u1 ← j·500
                u2 ← j·1500
                seed ← n·100000000
                        seed ← seed + v1 · 100000
                        seed ← seed + v2 · 1000
                        seed ← seed + đ · 100
                        seed ← seed + m
                        r ← 1
                call PGEN (n, v1, v2, p, s1, s2, đ, m, u1, u2, seed, r)
        endfor
endfor
endfor
```

Параметр r в процедуре PGEN определяет, сколько раз должно быть проведено тестирование. В нашем случае r был установлен в 1, потому что мы хотели сравнить два отдельных решения каждой задачи.

Данные первой и последней из 27 вариантов задач, сгенерированные с помощью процедуры PGEN, вызываемой из PROGEN, представлены в таблице2.

TEST 1

Заказанные длины

№	Длина	Количество
1	261	13
2	211	2
3	169	9
4	128	14
5	115	12

Нестандартные полосы

№	Длина
1	1420
2	1330
3	1156
4	1056
5	980
6	896
7	859
8	816
9	777
10	698

TEST 27

Заказанные длины

№	Длина	Количество	№	Длина	Количество	№	Длина	Количество
1	898	36	6	658	16	11	492	39
2	861	45	7	641	23	12	471	21
3	801	3	8	570	23	13	414	47
4	748	24	9	548	39	14	327	40
5	733	46	10	498	16	15	303	32

Нестандартные полосы

№	Длина	№	Длина	№	Длина	№	Длина
1	4477	24	4066	46	3166	69	2348
2	4445	25	4056	47	3161	70	2320
3	4441	26	4028	48	3141	71	2309
4	4398	27	3959	49	3118	72	2278
5	4396	28	3917	50	3008	73	2275
6	4365	29	3880	51	2982	74	2254
7	4347	30	3835	52	2963	75	2237
8	4335	31	3783	53	2948	76	2231
9	4332	32	3721	54	2926	77	2226
10	4330	33	3643	55	2898	78	2210
11	4280	34	3601	56	2867	79	2175
12	4265	35	3600	57	2811	80	2115
13	4259	36	3578	58	2807	81	2059
14	4245	37	3439	59	2797	82	1939
15	4232	38	3379	60	2785	83	1909
16	4231	39	3370	61	2777	84	1882
17	4227	40	3314	62	2767	85	1675
18	4212	41	3300	63	2717	86	1602
19	4207	42	3299	64	2468	87	1578
20	4171	43	3212	65	2447	88	1552
21	4166	44	3204	66	2402	89	1537
22	4153	45	3175	67	2364	90	1520
23	4144			68	2348		

Таблица 2 : Две экспериментальные задачи, сгенерированные с помощью PGEN
Результаты сравнения двух аглоритмов представлены в таблице 3.

								Overall trim loss				
								COLA		CUT		
	n	v_1	v_2	*đ*	m	u_1	u_2	*seed*	(cm)	(%)	(cm)	(%)
No.	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
1	5	100	300	10	10	500	1500	510301010	36	0.4158004	36	0.3875135
2	5	100	300	20	10	500	1500	510302010	34	0.3382075	9	0.0895255
3	5	100	300	30	10	500	1500	510303010	61	0.7232630	61	0.7232630
4	5	200	600	10	20	1000	3000	520601020	161	0.9202104	161	0.9202104
5	5	200	600	20	20	1000	3000	520602020	302	0.8522166	322	0.8981869
6	5	200	600	30	20	1000	3000	520603020	353	0.8407364	244	0.5811322
7	5	300	900	10	30	1500	4500	530901030	7	0.0323593	3	0.0136668
8	5	300	900	20	30	1500	4500	530902030	277	0.5168200	277	0.5168200
9	5	300	900	30	30	1500	4500	530903030	739	1.0310860	408	0.6125298
10	10	100	300	10	20	500	1500	1010301020	9	0.0491051	10	0.0543242
11	10	100	300	20	20	500	1500	1010302020	26	0.1305876	18	0.0904068
12	10	100	300	30	20	500	1500	1010303020	22	0.1051123	20	0.0955566
13	10	200	600	10	40	1000	3000	1020601040	8	0.0245126	14	0.0417374
14	10	200	600	20	40	1000	3000	1020602040	771	1.1070910	847	1.2452770
15	10	200	600	30	40	1000	3000	1020603040	120	0.1547389	128	0.1650548
16	10	300	900	10	60	1500	4500	1030901060	6	0.0115826	7	0.0128414
17	10	300	900	20	60	1500	4500	1030902060	1292	0.9982693	422	0.3266734
18	10	300	900	30	60	1500	4500	1030903060	1881	1.1487160	1516	0.9231125
19	15	100	300	10	30	500	1500	1510301030	43	0.1671136	75	0.2955083
20	15	100	300	20	30	500	1500	1510302030	10	0.0337860	12	0.0405432
21	15	100	300	30	30	500	1500	1510303030	9	0.0284126	8	0.0252557
22	15	200	600	10	60	1000	3000	1520601060	35	0.0545239	12	0.0190669
23	15	200	600	20	60	1000	3000	1520602060	320	0.2910784	271	0.243766
24	15	200	600	30	60	1000	3000	1520603060	110	0.0888565	203	0.1639808
25	15	300	900	10	90	1500	4500	1530901090	2	0.0027623	14	0.0194820
26	15	300	900	20	90	1500	4500	1530902090	847	0.4018217	433	0.2066826
27	5	300	900	30	90	1500	4500	1530903090	219	0.0876708	176	0.0705244
Sum							7700			5707		

n, v1 , v2 , đ, m, u1 , u2 и seed распределены по столбцам (1)-(8). Seed генерируется процедурой PROGEN. Все длины приведены в сантиметрах. Столбцы (9), (10) полный планируемый остаток материала, полученный в результате алгоритма COLA, а столбцы (11), (12) - алгоритма CUT. В столбцах (9), (11) отходы материала выражены в сантиметрах, а в столбцах (10), (12) - в процентах. Несмотря на одинаковое значение остатков, выраженных в сантиметрах, которые мы получили в результате алгоритмов COLA и CUT в первом случае (36 см), значения, выраженные в процентах, различаются. Это объясняется различными длинами раскраиваемых полос, используемых в обоих случаях . В обеих алгоритмах в 19 из 27 случаев полная потеря материала (остатки) была меньше самой короткой заказанной длины заготовки. В случае 24 длина самой короткой заготовки была 221см. Это означает, что мы, конечно, нашли оптимальное решение.

Но другие характеристики двух программ различны. Наихудший результат из полученных с помощью алгоритма COLA составляет 1881 см (1.1487160 %), а для алгоритма CUT эта величина равна 1516 см (0.9231125 %). Среднее значение потерь материала для COLA составило 285.19 см, а для CUT - 211.37 см. Сумма потерь материала во всех 27 случаях для алгоритма CUT была на 1993 см меньше, чем для алгоритма COLA. По полученным результатам исследования алгоритм CUT может бытьоценен выше, чем COLA. Но по быстродействию алгоритм COLA превосходит CUT на 30 %. Программа CUT получает решение всех 27 задач на персональном компьютере(Pentium II, 233 MHz) за 15 с, а программаCOLA - только 10 с.

5. Заключение

В статье приведено исследование алгоритмов одномерного раскроя. Существует множество алгоритмов и методов решения задач одномерного раскроя, учитывающих различные факторы и критерии. Однако, самым важным фактором является потеря материала. Мы решили разработать метод сравнения различных алгоритмов решения задач одномерного раскроя по критерию потери материала. Для различных типов задач потеря материалов определяется различными способами. Из задач Ассортимента раскраиваемых полос мы выделили три типа. Кроме S1D-CSP и H1D-CSP, уже описанных в литературе, представлен также тип задач G1D-CSP. Предложенный метод оценки позволяет сравнивать решения всех трех типов задач. Разработана постановка задачи типа G1D-CSP. Представлено исследование сравнения двух ориентированных на заготовку алгоритмов COLA и CUT.

Ссылки

[1] Bishoff E. E, Waesher G., “Cutting and Packing”/"Раскрой и упаковка", European Journal of Operational Research 84 (1995) 503-505.

[2] Dyckhoff H., “A typology of cutting and packing problems” / "Типология задач раскроя и упаковки", European Journal of Operational Research 44 (1990) 145-159.

[3] Ferreira J. S., Neves M. A. and Fonseca P., “A two-phase roll cutting problem” / "Двухфазовая задача раскроя", European Journal of Operational Research 44 (1990) 185-196.

[4] Gau T., Waesher G., “CUTGEN1: A problem generator for the Standard One-dimensional Cutting Stock Problem”/ "CUTGEN1: постановка задачи стандартного одномерного раскроя", European Journal of Operational Research 84 (1995) 572-579.

[5] Gilmore P. C. and Gomory R. E., “A linear programming approach to the cutting stock problem” / "Применение линейного программирования в задачах раскроя", Operations Research 9 (1961) 849-859.

[6] Gilmore P. C. and Gomory R. E., “A linear programming approach to the cutting stock problem, Part II” / "Применение линейного программирования в задачах раскроя. Часть II", Operations Research 11 (1963) 863-888.

[7] Goulimis C., “ Optimal solutions for the cutting stock problem” / "Оптимальные решения задач раскроя", European Journal of Operational Research 44 (1990) 197-208.

[8] Gradišar M., Jesenko J., Resinovič G., “Optimization of roll cutting in clothing industry” / "Оптиммизация рулонного раскроя в текстильной промышленности", Computers & Operations Research 24 (1997) 945-953.

[9] Gradišar M., Resinovič G., Jesenko J., Kljajić M.: “An algorithm for optimization of one-dimensional cutting” / "Алгоритмы оптимизации одномерного раскроя", European Journal of Operational Research 114 (1999) 557-568.

[10] Gradišar M., Kljajić M., Resinovič G.: “A hybrid approach for optimization of one dimensional cutting”/ "Использование гибридной оптимизации в задачах одномерного раскроя", European Journal of Operational Research 119 (1999) 165-174.

[11] Haessler R. W., Vonderembse M. A., “A procedure for solving the master slab cutting stock problem in the steel industry” / "Процедура решения задачи раскроя слябов в металлургической промышленности", AIIE Transactions 11 (1979) 160-165.

[12] Stadtler H., “A one-dimensional cutting stock problem in the aluminium industry and its solution” / "Задача одномерного раскроя при производстве алюминия и ее решение", European Journal of Operational Research 44 (1990) 209-223.

[13] Sweeney P. E. and Paternoster E. R., “Cutting and packing problems: A Categorised, Application-Orientated Research Bibliography”/ "Задачи раскроя и упаковки: библиографический обзор классификаций и приложений", Journal of the Operational Research Society 43 (1992) 691-706.

[14] Vance P.H., “Branch-and-Price Algorithms for the One-Dimensional Cutting Stock Problem” / "Метод Ветвей-и-Границ для задач одномерного раскроя", Computational Optimization and Applications 9 (1998) 211-228.

[15] Vanderbeck F., “Computational Study of a Column Generation Algorithm for Bin Packing and Cutting Stock Problems”/ "Изучение вычислений методом генерации столбцов решений задач упаковки ранца и задач раскроя", Research Papers in Management Studies, No 14, University of Cambridge, 1996.

[16] Waesher G., Gau T., “Generating Almost Optimal Solutions for the Integer One-dimensional Cutting Stock Problem” / "Получение наиболее оптимальных решений для целочисленных задач одномерного раскроя", Working Paper No. 94/06, Institut fur Wirtschaftswissenshaften, Technische Universität Braunschweig, 1994.