УДК 519.71

ПОРІВНЯЛЬНИЙ АНАЛІЗ АЛГОРИТМІВ РОЗВ'ЯЗАННЯ ЗАДАЧІ ПРЯМОКУТНОГО РОЗКРОЮ НАПІВОБМЕЖЕНОЇ СМУГИ

Ткаченко М.В.
Дніпропетровський національний університет імені Олеся Гончара
Лозовська Л.І.
Національна металургійна академія України

Розглянуто постановки задач прямокутного розкрою. Досліджено найбільш поширені алгоритми розв'язання задач прямокутного розкрою. Проведено аналіз ефективності роботи наведених алгоритмів. Проведено порівняльний аналіз ефективності роботи розглянутих алгоритмів.
Ключові слова: задача розкрою, задача упаковки, задача прямокутного розкрою, NP-важка задача, напівобмежена смуга.

Постановка проблеми. На сьогоднішній день і в найближчому майбутньому в різних сферах виробництва виникають і будуть виникати проблеми ресурсо- та енергозбереження, пов'язані із задачами розкрою і упаковки. Такі задачі відносяться до проблеми оптимізаційного геометричного моделювання, що полягає в оптимізації розміщення даного виду об'єктів в заданих областях.

Складність розв'язку цих задач полягає в тому, що вони відносяться до класу NP-важких проблем оптимізації, тобто для яких поки що не існує методів і алгоритмів, що знаходять точний розв'язок за поліноміальний час.

Для таких задач доцільно провести аналіз розв'язання і навести рекомендації та порівняльну характеристику по вибору алгоритмів розв'язання.

Аналіз останніх досліджень і публікацій. Аналіз вітчизняної та зарубіжної літератури, інформаційних інтернет-джерел дозволяє зробити висновок, що дослідженням і розробкою методів розв'язку даного класу задач займаються: Харківська школа Р-У Академіка Ю.Г. Стояна [2]; Інститут алгоритмів і наукових обчислень Німеччини (Т. Ленгауер); В. Міленковік, К. Даніельс (США); К. Доусланд, В. Доусланд (Великобританія); ряд російських вчених, серед яких Е.А. Мухачева, М.А. Верхотуров, В.В. Мартинов, А.А. Петунін, В.Д. Фроловський.

Виділення не вирішених раніше частин загальної проблеми. Оскільки задачі оптимального розкрою відносяться до класу NP-важких проблем оптимізації, до сьогоднішнього часу не розроблено універсального алгоритму для їх розв'язання, і, як наслідок, доводиться при розв'язанні практичних задач обирати алгоритм, який забезпечить прийнятну якість її розв'язання. А це означає необхідність проведення порівняльного аналізу вже існуючих алгоритмів і за необхідності їх удосконалення.

Мета статті. Головною метою цієї роботи є огляд найпоширеніших алгоритмів розв'язання задачі прямокутного розкрою та проведення порівняльного аналізу наведених алгоритмів.

Виклад основного матеріалу. Розглянемо декілька типових задач розкрою та упаковки.

Задача лінійного розкрою. Нехай дано стержень довжиною L та n видів деталей. Деталь виду j має довжину $l_j > 0$ та вартість $c_j > 0$, j = 1,..., n. Стержень необхідно розкроїти на деталі таким чином, щоб їх загальна вартість була максимальною.

Побудуємо математичну модель. Для цього введемо невідомі x_j – кількість деталей вигляду j, j = 1, ..., n. Тоді загальна довжина отриманих деталей обмежена розміром стержня.

$$\sum_{j=1}^{n} l_j x_j \le L \qquad (1)$$

Обмеження на змінні:

$$x_j \ge 0, \text{ цілі, } j = 1, \ldots, n. \qquad (2)$$

Загальна вартість отриманих деталей дорівнює:

$$\sum_{j=1}^{n} c_y x_y \to \max \qquad (3)$$

У даному випадку (1), (2) – умови, що визначають множину можливих значень змінних x_j, j = 1, ..., n. Серед усіх можливих розв'язків системи (1), (2) потрібно знайти те, на якому досягається максимум цільової функції (3).

Задача прямокутного розкрою. Є напівнескінченна смуга S фіксованої ширини W і множина прямокутників R = $\{R_i = \overline{1,m}\}$ з розмірами (l_i, w_i), де l_i – довжина, w_i – ширина і-го прямокутника, m – кількість прямокутників. Позначимо вихідні дані як (W, m, R). Потрібно розмістити прямокутники на смузі так, щоб довжина зайнятої частини смуги досягла мінімуму [1, с. 105].

Допустимою упаковкою прямокутників R на напівнескінченній смузі S називається розміщення прямокутників на смузі S так, щоб виконувались наступні умови:

– сторони прямокутників із R паралельні сторонам смуги S;

– прямокутники із R між собою не перетинаються (не мають спільних внутрішніх точок).

Введемо прямокутну систему координат, в якій осі O_x, O_y збігаються з нижньою і лівою сторонами смуги, і позначимо допустиму упаковку через P = $\{P_1, P_2, \ldots, P_m\}$, де $P_i = (x_i, y_i)$ – координати лівого нижнього кута і-го прямокутника. Тоді координати прямокутників для допустимої упаковки повинні відповідати наступним умовам:

$$x_i \ge 0,\ y_i \ge 0,\ y_i + \omega_i \le W,\ i = \overline{1,m}. \qquad (4)$$

$$(x_i + l_i \le x_j) \vee (x_j + l_j \le x_i) \vee (y_i + \omega_i \le y_j) \vee (y_j + \omega_j \le y_i), \qquad (5)$$

$$i, j = \overline{1,m},\ i \ne j.$$

Допустима упаковка P_0, для якої довжина зайнятої частини смуги $L = \max_{i=\overline{1,m}}(x_i + l_i)$ досягає мінімуму, називається оптимальною.

Порівняльний аналіз алгоритмів розв'язку задачі прямокутного розкрою напівобмеженої смуги.

Маємо набір з n прямокутників і напівобмежену смугу із фіксованою шириною W і нескінченною висотою. Кожен прямокутник по ширині не перевищує W. Задача – укласти прямокутники на смугу без перетинів так, щоб смуга стала якомога меншою довжиною. Домовимося, що всі прямокутники повинні бути упаковані ортогонально, їх не можна повертати.

Алгоритм Next Fit Decreasing High. Прямокутники упорядковано відповідно до незростання висоти (decreasing high), найвищий розташовується в лівому нижньому кутку смуги, тим самим утворивши перший рівень, по висоті рівний йому. Решта прямокутників розташовуються зліва направо, поки є місце на поточному рівні. Прямокутник, який не вміщується на рівні, розміщується вище, утворюючи наступний рівень, і так далі (рис. 1).

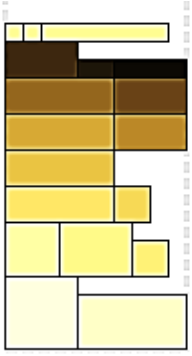

Рис. 1. Приклад роботи алгоритму Next Fit Decreasing High

Алгоритм First Fit Decreasing High. Схожий на попередній алгоритм, але для кожного наступного прямокутника знаходиться місце не тільки на останньому рівні, а починаючи з самого нижнього. Звідси і «first fit» – прямокутник розміщується на перший відповідний рівень знизу. Інтуїтивно зрозуміло, що упаковка буде якіснішою. Ще одна суттєва відмінність – в продуктивності, так як в гіршому випадку доведеться розглядати на кожному кроці всі рівні знизу догори.

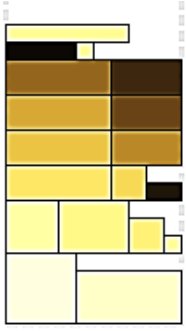

Рис. 2. Приклад роботи алгоритму First Fit Decreasing High

Алгоритм Best Fit Decreasing High. Модифікація попереднього алгоритму. Суть її в тому, що з рівнів, що підходять для упаковки чергового прямокутника, вибирається не перший, а кращий. Кращий рівень – це такий, на якому залишиться мінімум місця після упаковки поточного прямокутника. Інакше кажучи, вибирається мінімальний простір, що підходить, і це сприяє кращому заповненню рівнів.

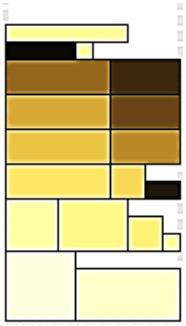

Рис. 3. Приклад роботи алгоритму Best Fit Decreasing High

Алгоритм Split Fit. Спочатку відбираються прямокутники, які ширше ніж половина смуги. Вони будуть упаковані в першу чергу. З них вибираються ще більш широкі – ширше, ніж 2/3 смуги; вони упаковуються алгоритмом First Fit Decreasing High. Над ними, починаючи з наступного рівня (рівень L), упаковуються ті, що залишилися (ті, що все ще ширше 1/2, але вже 2/3). Вони теж упаковуються за допомогою First Fit Decreasing High алгоритму. Такий розподіл створює пустий регіон шириною 1/3 праворуч від щойно упакованих, що починається на рівні L і закінчується поточною верхньою межею упаковки (тобто він не розміщується вище прямокутників шириною більше 1/2 але менше 2/3 ширини смуги). Всі прямокутники, що залишилися, які вужче, ніж 1/2 смуги, упаковуються за допомогою того ж First Fit Decreasing High алгоритму в першу чергу в утворений регіон, а якщо не вміщаються-то зверху.

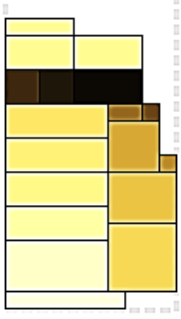

Рис. 4. Приклад роботи алгоритму Split Fit

Алгоритм Floor Ceiling No Rotation (рис. 5). Прямокутники упорядковано відповідно до незростання висоти і застосовується алгоритм Best Fit Decreasing High з деякими модифікаціями. У кожного рівня є «підлога» (floor) і «стеля» (ceiling). Поки є можливість, прямокутники пакуються на «підлогу» зліва направо. Коли місце закінчується, робиться спроба упакувати на «стелю» справа наліво; якщо немає місця на стелі, то тільки тоді починається новий рівень. На кожному кроці перевіряються усі рівні – спочатку «підлога», потім «стеля» – на наявність найбільш прийнятного місця. Метод вдало упаковує по «стелях» найдрібніші прямокутники.

Алгоритм Join (рис. 6). Упорядковані по незростанню висоти прямокутники об'єднуються в

пари так, щоб різниця висоти в парі не перевищувала заданої частки, зазвичай 0-10%. Ще одна умова – щоб їх сумарна ширина могла вміститися в смугу. Отримані прямокутники упаковуються разом з рештою без пари за допомогою алгоритмів Next Fit Decreasing High і First Fit Decreasing High, вибирається кращий розв'язок. Існує варіація цього алгоритму, коли прямокутники упорядковано відповідно по ширині і об'єднуються вертикально, з тією ж умовою максимального відхилу ширини в парі на задану кількість відсотків.

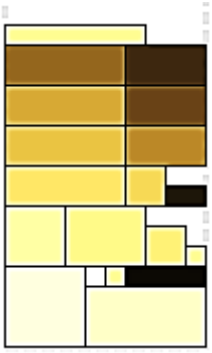

Рис. 5. Приклад роботи алгоритму Floor Ceiling No Rotation

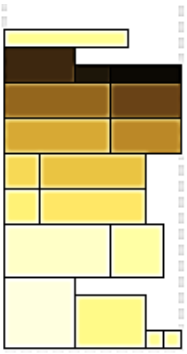

Рис. 6. Приклад роботи алгоритму Join

Алгоритм Sleator (рис. 7) використовує інтуїтивний принцип упаковки ранця: скласти на дно найбільші предмети і а зверху розмістити дрібні. З прямокутників вибираються найширші, ширше половини смуги, і розміщуються один на одного з вирівнюванням по лівому краю. Решту упорядковано відповідно до незростання висоти і починають укладати один за одним зліва направо зверху на вже укладені. Як тільки їх сумарна ширина перевищить половину ширини смуги, що залишилася, вони розміщуються один на одного то ліворуч (починаючи від лівого краю смуги), то праворуч (від середини), за принципом «де на даний момент менше».

Алгоритм Knapsack 0-1 (рис. 8). Це окремий випадок задачі про ранець; примітний тим, що крім відповіді на основне питання (отриманий розмір упаковки) дає ще і розв'язок – список предметів, які потрібно упакувати. Порядок дій такий: прямокутники упорядковуються за незростанням висоти; упаковується перший прямокутник на новий рівень; для цього рівня знаходиться розв'язок задачі Knapsack 0-1, який дає нам список прямокутників, максимізований за площею. Вибрані прямокутники вилучаються зі списку не упакованих, рівень закривається і відкривається новий – утворений першим (найвищим) з решти. Повторюємо, поки є прямокутники [2, с. 11].

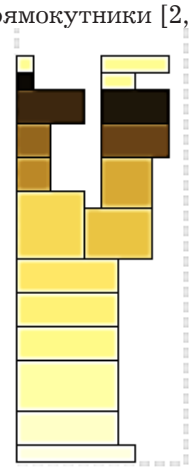

Рис. 7. Приклад роботи алгоритму Sleator

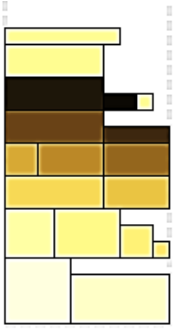

Рис. 8. Приклад роботи алгоритму Knapsack-01

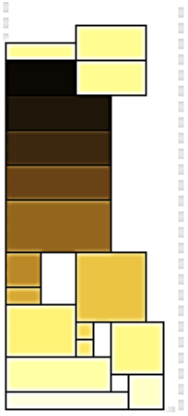

Рис. 9. Приклад роботи алгоритму Burke

Алгоритм Burke (рис. 9). Це безрівневий алгоритм, для якого вводиться додаткова «карта висот» – масив, за яким в процесі заповнення смуги можна відстежувати найменш заповнені області та їх ширину. Спочатку він заповнений нулями. Прямокутники упорядковано відповідно до незростання ширини. Потім на кожному кроці алгоритму:

1. Обчислюється позиція найнижчої області – індекс мінімального значення масиву;

2. Вибирається найбільш придатний прямокутник – який вміщується в цю область, і який максимально її заповнює по ширині;

3. Якщо відповідний прямокутник знайдений, він розміщується в цій області одним із способів:

3.1. По лівому краю області;

3.2. Ближче до вищого сусіда; якщо один із сусідів – край смуги, то ближче до краю;

3.3. Ближче до менш високого сусіда; якщо один із сусідів – край смуги, то далі від краю. До значень масиву, що відповідають ширині прямокутника, додається його висота.

4. Якщо придатного прямокутника немає, область «заповнюється» шляхом вирівнювання її висоти до висоти найближчого краю.

Результат роботи кожного алгоритму – це рівень заповненості смуги, і чим він менший, тим краще. У таблиці 1 наведено результати роботи розглянутих алгоритмів – отримана висота смуги, на якій розміщуються фігури.

Як можна бачити, алгоритм *Floor Ceiling No Rotation* показує найбільш близькі до оптимальних результати.

Висновки і пропозиції. За рахунок своєї простої постановки, задача прямокутного розкрою може бути застосована до величезної кількості практичних задач: безпосередньо до упаковки об'єктів або розкрою матеріалу, розподілу коштів, планування завдань на кластерах та інші.

Розглянуті алгоритми розв'язання задач прямокутного розкрою використовують різні підходи, такі як розбиття смуги на горизонтальні рівні або просто упаковка прямокутників на площині. Емпіричним шляхом було визначено, що алгоритм *Floor Ceiling No Rotation* показує найкращі результати по упаковці, тому його використання для вирішення такого роду задач є найбільш доцільним.

З метою підвищення ефективності з точки зору отримання найменшої висоти упаковки, існує кілька модифікацій на деякі з наведених алгоритмів.

Таблиця 1

Результати роботи наведених алгоритмів

Назва алгоритму	Оптимальний розв'язок	Отриманий розв'язок
Next Fit Decreasing High	150	181
First Fit Decreasing High	150	171
Best Fit Decreasing High	150	171
Split Fit	150	171
Floor Ceiling No Rotation	150	161
Join	150	181
Sleator	150	241
Knapsack 0-1	150	181
Burke	150	220

Список літератури:

1. Картак В.М. Задача упаковки прямоугольников: точный алгоритм на базе матричного представления / В.М. Картак. – Уфа: УГАТУ, 2007.
2. David Pisinger. Algorithms for Knapsack Problems, Ph.D. thesis, February 1995.
3. Могилевский Д.В. Определение оптимальных параметров резки стали 10Х12НВМФА / Д.В. Могилевский, Т.А. Литвинова, Н.Н. Подрезов, Р.В. Пирожков // Инженерный вестник Дона, 2013. – № 2.
4. Стоян Ю.Г. Математические модели и оптимизационные методы геометрического проектирования / Ю.Г. Стоян, С.В. Яковлев. – Киев: Наук. думка, 1986.
5. Nthabiseng Ntene. An Algorithmic Approach to the 2D Oriented Strip Packing Problem, 2007.

Ткаченко М.В.
Днепропетровский национальный университет имени Олеся Гончара
Лозовская Л.И.
Национальная металлургическая академия Украины

СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ ПРЯМОУГОЛЬНОГО РАСКРОЯ ПОЛУОГРАНИЧЕННОЙ ЛЕНТЫ

Аннотация
Рассмотрены постановки задач прямоугольного раскроя. Исследованы наиболее распространенные алгоритмы решения задач прямоугольного раскроя. Проведен анализ эффективности работы приведенных алгоритмов. Проведен сравнительный анализ работы рассмотренных алгоритмов.

Ключевые слова: задача раскроя, задача упаковки, задача прямоугольного раскроя, NP-трудная задача, полуограниченная лента.

Tkachenko M.V.
Oles Honchar Dnipropetrovsk National University
Lozovskaya L.I.
National Metallurgical Academy of Ukraine

COMPARATIVE ANALYSIS OF ALGORITHMS SOLVING TWO-DIMENSIONAL PACKING PROBLEM ONTO SEMI-INFINITE STRIP

Summary
Two-dimensional strip packing problems were considered. Investigated the most common algorithms solving two-dimensional packing problems. Analized the efficiency of these algorithms. Introduces a comparative analysis of the considered algorithms.

Keywords: strip packing problem, bin packing problem, cutting and packing, two-dimensional rectangular strip packing problem, NP-hard problem, semi-infinite strip.