УДК 004.023:004.896

Гулаєва Н. М., Щур О. П.

АНАЛІЗ ПАРАМЕТРІВ ГЕНЕТИЧНОГО АЛГОРИТМУ РОЗВ'ЯЗКУ ЗАДАЧІ ОРТОГОНАЛЬНОЇ УПАКОВКИ

Досліджено задачу двовимірної ортогональної упаковки прямокутних об'єктів у напівнескінченну смугу фіксованої ширини. Розглянуто способи розв'язку задачі за допомогою генетичних алгоритмів, проаналізовано вплив генетичних операторів на ефективність роботи генетичного алгоритму, запропоновано проблемно-орієнтовані генетичні оператори. Здійснено обчислювальні експерименти на відомих тестових прикладах та дано рекомендації щодо вибору параметрів для різних класів задач.

Ключові слова: задача двовимірної ортогональної упаковки, IBL-декодер, генетичний алгоритм, генетичні оператори.

Вступ

Оптимізаційна задача упаковки-розкрою формулюється як задача оптимального розташування множини предметів меншого розміру (деталей) на об'єктах більшого розміру (заготовках); цільовою функцією може бути, наприклад, мінімізація втрат заготовок під час розташування на них усіх деталей або максимізація кількості деталей, які можна розташувати на всіх заготовках. Це завдання постає в багатьох галузях промисловості: розкрій матеріалів (папір, деревина, скло, метал, шкіра, текстиль) у процесі виробництва готових виробів, розташування елементів електронних схем на платах, верстання матеріалів у поліграфічній промисловості, упаковка вантажів у контейнери, низка практичних завдань планування зайнятості та розподілу виробничих ресурсів тощо.

Задача про раціональний розкрій матеріалів сформульована видатним радянським ученим Л. В. Канторовичем у 1939 р. [5]. Суттєвий внесок у розробку теорії та методів розв'язку задач раціонального розкрою зробили американські вчені П. Гілмор і Р. Гоморі в першій половині 1960-х рр.. [17–18]. В оглядових працях 1970-х рр. було виявлено зв'язок між задачами розкрою та упаковки, обґрунтовано доцільність спільного розгляду цих задач. Питанням систематизації задач розкрою й упаковки присвячено роботи [15; 28].

Тут розглянуто задачу двовимірної ортогональної упаковки прямокутних деталей у напівнескінченну смугу з забороною повороту деталей. Така задача виникає у меблевій, швейній, скляній промисловості під час розкрою такого декорованого або структурованого матеріалу, як деревина або гофрована тканина, що надходить у рулонах, довжину яких порівняно з розмірами прямокутних деталей можна розглядати як нескінченну. Задача полягає в розташуванні прямокутних деталей на смузі мінімальної довжини за виконання таких умов:

- сторони прямокутників є паралельними сторонам смуги;
- прямокутники не перетинаються один з одним;
- прямокутники не перетинаються зі сторонами смуги;
- повороти прямокутників на число, що є кратним 90°, заборонено.

Задача двовимірної ортогональної упаковки належить до класу NP-складних задач комбінаторної оптимізації [4; 16]. В зв'язку з цим наявні точні методи її розв'язку основано на схемі повного перебору, отже, є мало прийнятними для розв'язку завдань, що трапляються на практиці. Тому з метою скорочення обсягів обчислень поряд з точними активно розробляються різноманітні наближені методи розв'язку цієї задачі. Загальноприйнятою класифікацією методів розв'язку задачі розкрою-упаковки є класифікація на алгоритмічні (точні), евристичні та метаевристичні [2; 3; 19; 27]. Серед останніх виділяють генетичні алгоритми, які набувають все більшого поширення.

Нагадаємо, що в основу генетичних алгоритмів покладено ідею використання принципів біологічної еволюції для розв'язання оптимізаційних задач. Пошук розв'язку задачі в генетичному алгоритмі здійснюється не з єдиної точки, як у більшості традиційних методів оптимізації, а з цілої множини точок – популяції особин. Кожна особина популяції (хромосома) представляє закодований розв'язок заданої проблеми та характеризується певним рівнем пристосованості, який обраховується за допомогою функції пристосованості (функції здоров'я) і фактично він є

© *Гулаєва Н. М., Щур О. П.*, 2012

оцінкою якості закодованого розв'язку. Задача оптимізації розглядається як задача пошуку особини з найкращим значенням функції пристосованості. Цей пошук здійснюється на основі механізмів спадковості, мінливості та відбору, які реалізуються за допомогою різноманітних генетичних операторів (кросинговеру, мутації, інверсії тощо).

Можна будувати широкий спектр генетичних алгоритмів, які відрізнятимуться своїми параметрами: способом кодування розв'язків задачі; методами відбору особин, що допускаються до породження нащадків; типами і коефіцієнтами застосування генетичних операторів; розміром популяції; умовою закінчення роботи алгоритму. Добре відомо, що від вдалого вибору параметрів генетичного алгоритму залежить якість отриманого розв'язку. З цієї точки зору здається дивним, що досі не було здійснено ґрунтовного аналізу впливу генетичних операторів на ефективність роботи генетичних алгоритмів розв'язку задачі упаковки: у більшості робіт, присвячених генетичним алгоритмам розв'язку цієї задачі, якість алгоритмів поліпшувалася або завдяки вибору більш вдалого способу кодування та алгоритму декодування, або за рахунок гібридизації генетичного алгоритму з іншими евристичними та метаевристичними алгоритмами.

Тут проаналізовано вплив генетичних операторів кросинговеру, мутації та методів відбору в батьківський пул на роботу генетичного алгоритму розв'язку задачі двовимірної ортогональної упаковки прямокутних об'єктів у напівнескінченну смугу фіксованої ширини. Дано рекомендації щодо добору параметрів генетичного алгоритму й здійснено порівняння ефективності роботи запропонованого алгоритму з алгоритмами, запропонованими іншими авторами.

Постановка задачі

Задачу двовимірної ортогональної упаковки прямокутних об'єктів у напівнескінченну смугу фіксованої ширини (*Two Dimensional Orthogonal Rectangular Strip Packing Problem*) формулюємо так: вихідну інформацію задано набором даних (*W, m, w, l*), де $W \in Z^+$ – ширина напівнескінченної смуги; $m \in Z^+$ – кількість прямокутників; $w = (w_1, w_2, \ldots, w_i, \ldots, w_m)$, $w_i \in Z^+$ – ширина *i*-го прямокутника; $l = (l_1, l_2, \ldots l_i, \ldots, l_m)$, $l_i \in Z^+$ – довжина *i*-го прямокутника. Вводиться прямокутна система координат *XOY*, причому осі *OX* та *OY* збігаються відповідно з нижньою необмеженою та боковою сторонами смуги. Позиція *i*-го прямокутника називається *горизонтальною*, якщо його сторона довжини l_i є паралельною необмеженій стороні смуги, а друга сторона є перпендикулярною їй. Задамо горизонтальну позицію *i*-го прямокутника парою координат його нижнього лівого кута (x_i, y_i). Набір векторів (x_i, y_i), $i = 1, 2, 3 \ldots m$, називається *допустимою прямокутною упаковкою*, якщо виконуються умови:

- сторони прямокутників паралельні сторонам смуги (умова ортогональності);
- прямокутники не перетинаються один з одним:

$$\forall\, i, j = 1, \ldots, m, i \neq j : ((x_i \geq x_j + l_j) \vee (x_j \geq x_i + l_i)) \wedge \wedge ((y_i \geq y_j + w_j) \vee (y_j \geq y_i + w_i)).$$

- прямокутники не виходять за межі смуги:

$$\forall\, i = 1, \ldots, m : (x_i \geq 0) \wedge (y_i \geq 0) \wedge ((y_i + w_i) \leq W);$$

- будь-які повороти прямокутників заборонені.

Прямокутник розташований *щільно*, якщо він дотикається хоча б нижньою та лівою сторонами до інших прямокутників упаковки або до краю смуги. Упаковку називають *щільною*, якщо всі її прямокутники розташовані щільно.

Допустима прямокутна упаковка називається *оптимальною*, якщо довжина зайнятої упакованими прямокутниками частини смуги сягає мінімуму (при цьому досягається мінімум втрат невикористаних частин смуги). Як бачимо, оптимальна упаковка є щільною.

Розв'язком задачі двовимірної ортогональної упаковки прямокутних деталей у напівнескінченну смугу є оптимальна упаковка.

Опис і реалізація генетичного алгоритму

Наведемо загальну схему генетичного алгоритму розв'язку поставленої задачі та опишемо його параметри.

Схема генетичного алгоритму

1. Ініціалізація початкової популяції.
2. Обчислення пристосованості хромосом у популяції.
3. Перевірка умови завершення роботи. Якщо так, то відбувається перехід на п.6, якщо ні – то на п.4.
4. Генерація нової популяції:
   4.1. Відбір хромосом у батьківський пул.
   4.2. Породження нащадків завдяки застосуванню генетичних операторів.
5. Перехід на п.2.
6. Зупинка алгоритму. Результат – найкраща хромосома в популяції.

Кодування розв'язків

Генетичні алгоритми оперують хромосомами – закодованими розв'язками поставленої задачі. Для застосування генетичних алгоритмів до задачі двовимірної ортогональної упаковки сьогодні розроблено досить багато різних способів кодування: прямий код, кодування переста-

новками прямокутників, кодування парою послідовностей, кодування блок-структурами, кодування орієнтованими бінарними деревами, кодування номерами евристик тощо, див.: [1; 7; 8; 12; 21; 23; 26]. Найпоширенішим серед цих способів є *кодування перестановками*. За такого кодування хромосома містить послідовність номерів прямокутників, яка представляє порядок їхньої укладки; правила укладки прямокутників задаються спеціальним алгоритмом – декодером. Вибір декодера має принципове значення, оскільки одну і ту саму перестановку прямокутників можна розшифрувати різними способами, а відтак одержати різні упаковки, зокрема упаковки різної довжини.

У генетичних алгоритмах, як правило, використовують однопрохідні он-лайн декодери: «наступний придатний» (*Next-fit – NF*), «перший придатний» (*First fit – FF*), «нижній лівий» (*Bottom-Left – BL*), «удосконалений нижній лівий» (*Improved Bottom Left – IBL*), «нижній лівий із заповненням» (*Bottom-Left-Fill – BLF*), «мінімізація площі навколишнього прямокутника» (*Minimization of Enclosing Rectangle Area – MERA*), «блочний декодер» (*Block Decoder – BD*) та ін., див. : [6; 9; 14; 22; 25]. Такі декодери розглядають прямокутники у заданому хромосомою порядку та упаковують кожен із них у вільну ділянку смуги, що обирається за певною стратегією відбору. Особливо популярними є декодери, що генерують *BL*-компактні упаковки. Упаковка називається *BL*-компактною, якщо вона задовольняє *BL*-умову: жоден розташований на смузі прямокутник не може бути зсунутий далі вниз або вліво. Опишемо деякі з цих декодерів.

1. «*Нижній лівий*» (*BL*). Перший прямокутник розташовується в нижньому лівому куті смуги. Кожен наступний прямокутник спочатку розташовується у верхньому правому куті упакованої площі смуги, звідки він послідовно зсувається вліво настільки, наскільки це можливо, потім униз, наскільки можливо, знову вліво, вниз і так доти, доки не задовольнятиметься *BL*-умова. Складність *BL*-алгоритму – $O(m^2)$.
2. «*Удосконалений нижній лівий*» (*IBL*). Декодер є модифікацією декодера «нижній лівий», але відрізняється від останнього стратегією розташування чергового прямокутника: під час розташування чергового прямокутника зсув уліво має перевагу. Це означає, що прямокутник зсувається вниз лише тоді, коли рух вліво неможливий. Зсув прямокутників відбуватиметься доти, доки не задовольнятиметься *BL*-умова. Цей декодер є ефективнішим за *BL*-декодер ([25]). Складність *IBL*-алгоритму – $O(m^2)$.
3. «*Нижній лівий із заповненням*» (*BLF*). На відміну від двох попередніх, декодер «нижній лівий із заповненням» здатний заповнювати «дірки», утворені внаслідок упаковки прямокутників. Для цього *BLF*-декодер постійно тримає в пам'яті упорядкований знизу-догори та зліва-направо список точок, які можуть бути координатами лівого нижнього кута чергового прямокутника; послідовно переглядаючи цей список, алгоритм намагається розташувати черговий прямокутник та у випадку успіху оновлює список точок. Порівняно з двома попередніми декодерами *BLF*-декодер може породжувати упаковки меншої довжини. Експериментально було показано, що *BLF*-декодер є на 25 % ефективніший від *BL*-декодера ([20]). Складність *BLF*-алгоритму – $O(m^3)$.

Звернемо увагу на те, що для вибору декодера для генетичного алгоритму важливим критерієм є не лише якість породжуваних ним упаковок, а й часова складність його роботи, адже декодування хромосоми відбувається щоразу при переобчисленні функції пристосованості. У цій роботі для кодування розв'язків використовують перестановки прямокутників, а як декодер – IBL-декодер: при часовій складності – $O(m^2)$ – він генерує достатньо компактну упаковку.

Тип репродукції

Генераційний: на кожній ітерації алгоритму відбувається повна заміна особин популяції їхніми нащадками.

Функція пристосованості

Найчастіше функцією пристосованості стає або довжина упакованої частини смуги ($L \to min$, де L – довжина використаної частини смуги), або коефіцієнт розкрою – відношення загальної площі упакованих прямокутників до площі зайнятої ними частини смуги, тобто відношення корисної площі до витраченої ($\frac{S_shapes}{S} \to 1$, де *S_shapes* – загальна площа прямокутників, S – площа використаної частини смуги).

Зазначимо, що при використанні *IBL*-декодера не виключено можливість утворення дірок між прямокутниками в упаковці. Розробимо функцію пристосованості, яка явно враховувала б коефіцієнт розкрою, роблячи водночас розкрій із дірками менш привабливим.

Розрідженістю гена хромосоми назвемо величину $sparseness = \frac{free_space}{block_width}$, де *free_space* – частина лівої сторони прямокутника, що не дотична до жодного іншого прямокутника або до краю смуги, *block_width* – ширина прямокутника.

Назвемо ген *слабким*, якщо *sparseness*>0, тобто під час упакування відповідного прямокутника утворюється дірка, та *міцним*, інакше.

Відносною розрідженістю хромосоми назвемо величину $sparseness(chr) = \sum_{i=1}^{N} sparseness(i)$, де N – кількість генів у хромосомі, *sparseness(i)* – розрідженість i-го гену.

Тоді функцію пристосованості, яка враховує коефіцієнт розкрою та накладає штраф за наявність дірок, можна задати наступним чином:

$$Fitness(chr) = \frac{S - shapes}{S} - sparseness(chr) \to \max$$

Відбір

Для аналізу були обрані так оператори відбору:

1. *Відбір відтинанням* (*Truncation Selection*), або *стратегія елітаризму (Elite Selection*). У цьому випадку відбувається автоматичне перенесення певної кількості найкращих хромосом популяції – еліти – в батьківський пул, «дарування» життя найкращим.
2. *Відбір за методом рулетки* (*Roulette-Wheel Selection*). Згідно з цим методом ймовірність особини бути обраною в батьківський пул є прямо пропорційною значенню її пристосованості: $P_s(X) = \frac{f(X)}{\sum_{i=1}^{N} f(X_i)}$, де $f(X)$ – оцінка пристосованості хромосоми X, N – кількість хромосом в популяції.

Перерахуємо обрані для аналізу генетичні оператори, які прийнятні для використання в задачах із кодуванням розв'язків перестановками.

Оператор кросинговеру

1. *PMX-кросинговер* (*Partially-Mapped Crossover*). Роботу оператору опишемо двома кроками. На першому кроці випадково обираються дві точки схрещування та відбувається обмін генетичним матеріалом між обраними точками. У більшості випадків такий обмін частинами породжує недопустимі хромосоми: різні гени хромосоми можуть набути однакових значень. Тому другий крок полягає в усуненні дублів. Для цього визначають відображення на основі значень генів, якими обмінялися хромосоми. У прикладі нижче відображення визначають так: 1→4, 8→5 для протонащадка_1; 4→1, 5→8 для протонащадка_2:

Батько_1: (1 2 3|4 5 6 7| 8 9) Протонащадок_1: (**1** 2 3|1 8 7 6| **8** 9)

Батько_2: (4 5 2|1 8 7 6| 9 3) Протонащадок_2: (**4 5** 2|4 5 6 7| 9 3)

Нащадок_1: (4 2 3|1 8 7 6| 5 9)

Нащадок_2: (1 8 2|4 5 6 7| 9 3)

2. *OX-кросинговер* (*Order Crossover*) *з одною точкою схрещування*. Випадково обирають точку схрещування. Значення генів першого батька до цієї точки копіюються в першого нащадка без змін. Значення генів цього нащадка після точки схрещування заповнюються тими значеннями генів другого батька, які ще не потрапили до його генотипу, причому в тому порядку, в якому ці значення зустрічаються в другого батька. Аналогічно формується другий нащадок:

Батько_1: (1 2 3 4| 5 6 7 8 9) Нащадок_1: (1 2 3 4| 5 8 7 6 9)

Батько_2: (4 5 2 1| 8 7 6 9 3) Нащадок_2: (4 5 2 1| 3 8 7 6 9)

Оператор мутації

Для аналізу було обрано *мутацію вставкою* (*Insertion Mutation*). Випадково обирається позиція в хромосомі, і значення гену вилучається з цієї позиції та вставляється у випадково обрану нову позицію:

Батько: (1 2 3 4 5 6 7 8 9) Нащадок: (1 2 6 3 4 5 7 8 9)

Наведені оператори можна використовувати в будь-яких задачах, розв'язком яких є перестановки елементів. Недоліком цих операторів при використанні в задачі упаковки-розкрою є те, що випадковий вибір позиції може зруйнувати хромосому в її міцному гені, тобто зруйнувати знайдену щільну частину упаковки – частину без дірок. Зважаючи на специфіку задачі упаковки-розкрою, ми запропонували проблемно-орієнтовані оператори.

1. *WP-OX-кросинговер* (*Weak Point Order Crossover*) працює так само, як *OX*-кросинговер з одною точкою схрещування, але відрізняється від останнього тим, що точка схрещування хромосоми обирається не випадково за рівномірним розподілом, а згідно з детермінованою процедурою. В множину точок підозрілих на точки схрещування потрапляють усі слабкі гени хромосоми. Серед цих точок обирається найслабший ген – ген із найбільшим значенням розрідженості; цей ген і стає точкою схрещування. Перевагою описаного кросинговеру є те, що надається можливість передати нащадкам найщільнішу частину хромосоми, не руйнуючи вже знайдені вдалі частини розв'язку.
2. *WP-мутація* (*Weak Point Mutation*). Найслабший ген хромосоми та ген, номер позиції якого в хромосомі є більшим за номер позиції найслабшого гена, обмінюються своїми значеннями. Таким чином, *WP*-мутація дає змогу покращити щільність упаковки в її критичному місці.

3. *Оператор упаковки* (*Blocking*) об'єднує вже упаковані прямокутники в один *метапрямокутник*; для об'єднання обираються такі прямокутники (два або більше), що в описуванні навколо них іншого прямокутника мінімальних розмірів не залишиться незайнятих ділянок (відходів) або ділянок, зайнятих іншими прямокутниками упаковки. Надалі метапрямокутник розглядається як звичайний прямокутник: його можна зсувати чи групувати з іншими прямокутниками та метапрямокутниками. Водночас відносне розташування прямокутників всередині метапрямокутника залишається незмінним. Можна сказати, що застосування оператора упаковки запобігає руйнуванню знайдених оптимальних розв'язків підзадач, оскільки наявні метапрямокутники (які представляють оптимальні розв'язки підзадач), не будуть зруйновані внаслідок застосування генетичних операторів. Оператор упаковки має застосовуватися першим у послідовності генетичних операторів. Як показали експерименти, застосування оператора упаковки прискорює роботу алгоритму та за певних умов покращує якість знайденого алгоритмом розв'язку. Зауважимо, що вперше концепцію метапрямокутників було запропоновано та впроваджено в генетичний алгоритм у праці [24] для розв'язку задачі прямокутного гільйотинного розкрою: у метапрямокутниках допускали відходи, а розв'язки задачі кодували деревами розрізів.

Умова закінчення роботи алгоритму

Алгоритм закінчує роботу після наперед заданої кількості ітерацій.

Результати експериментів

Задля виявлення оптимальних значень параметрів генетичного алгоритму розв'язку поставленої задачі проведено серію обчислювальних експериментів. Як тестові дані використано відомі тести для двовимірної задачі упаковки, а також спеціально згенеровані тестові приклади. Усі тестові приклади розбито на два класи:

- Однорідні об'єкти: усі прямокутники є схожими за формою та розміром. За запропонованим у [20] методом були згенеровані набори даних, де прямокутники відрізняються один від одного за довжиною/шириною не більше, ніж на 75 %.
- Неоднорідні об'єкти: прямокутники можуть суттєво відрізнятись один від одного за формою та розмірами. Для експериментів використано набори тестових даних Є. Гоппер [20], скласифіковані за сімома категоріями залежно від розмірів. У цих наборах співвідношення довжини/ширини прямокутників варіюється в межах від 1 до 3, а співвідношення сторін різних прямокутників – від 1 до 7.

Для всіх тестових прикладів відомий оптимальний розв'язок. Характеристики використаних однорідних (H) та неоднорідних (C) тестових наборів резюмовано у табл. 1 і табл. 2.

Таблиця 1. **Класи однорідних об'єктів**

Приклад	Кількість об'єктів	Ширина смуги	Оптимальна довжина
H1	16	27	21
H2	26	24	34
H3	43	24	48
H4	67	38	44

Таблиця 2. **Класи неоднорідних об'єктів (тести Є. Гоппер)**

Категорія / Приклади	Кількість об'єктів	Ширина смуги	Оптимальна довжина
C1/P1,P2	16, 17	20	20
C3/P1	28	60	30
C4/P1	49	60	60
C6/P1,P2	97, 97	80	120

Критерієм оцінки розв'язку, знайденого генетичним алгоритмом, слугує значення відхилення від нижньої границі довжини упаковки – відхилення (у відсотках) довжини знайденої алгоритмом упаковки від оптимальної довжини смуги: $gap = \frac{L - L_{opt}}{L_{opt}} \cdot 100\%$, де L – найкращий результат алгоритму (мінімальна знайдена довжина смуги), L_{opt} – оптимальна довжина смуги.

У табл. 3–4 наведено результати експериментів для описаних класів задач під час використання різних генетичних операторів. Інші параметри генетичного алгоритму задавались так: максимальна кількість ітерацій – 100; кількість особин у популяції – 100; ймовірність застосування оператора кросинговеру – 1; ймовірність застосування оператора мутації – 0,05. При застосуванні елітарного відбору в батьківський пул відбирається 20 % найкращих особин популяції, решта 80 % відбираються за методом рулетки (за експериментами таке співвідношення дає найкращі результати). Як у більшості робіт із генетичних алгоритмів, для аналізу не тільки якості знайдених розв'язків, а й стабільності роботи побудованих алгоритмів, виконували по 10 прогонів генетичного алгоритму з однаковими параметрами для кожної задачі; найкращий (*best*) та середній (*average*) результати по всіх прогонах зберігались.

Таблиця 3. **Результати експериментів (*gap*, %) на класах однорідних об'єктів**

Параметри								Класи задач								average
відбір		кросинговер			мутація		упаковка	H1		H2		H3		H4		
елітарний	рулетки	OX	PMX	WP-OX	вставкою	WP		best	average	best	average	best	average	best	average	
*		*			*			0,00	8,57	5,88	7,06	4,17	7,50	9,09	9,09	8,06
*		*				*		0,00	**0,00**	5,88	9,41	8,33	8,33	9,09	9,09	6,71
*		*						0,00	**0,00**	5,88	5,88	4,17	5,00	9,09	9,09	4,99
*			*		*			0,00	8,57	5,88	9,41	8,33	8,33	9,09	9,09	8,85
*			*			*		0,00	2,86	5,88	9,41	8,33	9,17	9,09	9,09	7,63
*			*					0,00	2,86	5,88	9,41	8,33	9,17	4,55	8,18	7,40
*				*	*			0,00	2,86	5,88	8,24	4,17	7,50	9,09	9,09	6,92
*				*		*		0,00	2,86	5,88	9,41	8,33	8,33	4,55	8,18	7,20
*				*				0,00	2,86	5,88	7,06	8,33	8,33	4,55	8,18	6,61
	*	*			*			0,00	**0,00**	0,00	**4,71**	4,17	**4,17**	4,55	7,27	4,04
	*	*				*		0,00	**0,00**	0,00	**4,71**	4,17	**4,17**	4,55	**6,36**	**3,81**
	*	*						0,00	**0,00**	0,00	**4,71**	4,17	5,00	4,55	**6,36**	4,02
	*		*		*			0,00	**0,00**	5,88	7,06	4,17	7,50	6,82	8,18	5,69
	*		*			*		0,00	**0,00**	5,88	5,88	4,17	6,67	4,55	8,18	5,18
	*		*					0,00	**0,00**	5,88	5,88	4,17	5,83	4,55	8,18	4,97
	*			*	*			0,00	**0,00**	5,88	5,88	4,17	5,00	4,55	7,27	4,54
	*			*		*		0,00	**0,00**	5,88	5,88	4,17	7,50	6,82	7,27	5,16
	*			*				0,00	**0,00**	5,88	5,88	4,17	5,83	4,55	7,27	4,75
*					*		*	0,00	**0,00**	0,00	5,88	8,33	9,17	9,09	11,36	6,60
*						*	*	0,00	**0,00**	5,88	10,00	8,33	11,67	9,09	13,18	8,71
	*				*		*	0,00	**0,00**	0,00	6,47	4,17	7,08	9,09	10,00	5,89
	*					*	*	0,00	**0,00**	5,88	8,82	4,17	12,08	13,64	13,64	8,64

Таблиця 4. **Результати експериментів (*gap*, %) на класах неоднорідних об'єктів**

Параметри								Класи задач												average
відбір		кросинговер			мутація			C1/P1		C1/P2		C3/P1		C4/P1		C6/P1		C6/P2		
елітарний	рулетки	OX	PMX	WP-OX	вставкою	WP	упаковка	best	average	best	average	best	average	best	average	best	average	best	average	
*		*			*			10,00	13,00	10,00	15,00	10,00	14,00	18,33	20,33	20,83	24,50	23,33	26,33	18,86
*		*				*		10,00	12,00	10,00	16,00	10,00	14,00	18,33	21,67	22,50	25,33	23,33	26,00	19,17
*		*						5,00	5,00	10,00	10,00	10,00	12,67	16,67	21,33	23,33	24,33	19,17	24,50	16,31
*			*		*			10,00	9,00	10,00	14,00	10,00	14,00	16,67	20,33	19,17	23,50	24,17	27,67	18,08
*			*			*		0,00	15,00	15,00	15,00	10,00	13,33	16,67	22,33	22,50	24,33	25,00	27,33	19,56
*			*					10,00	10,00	10,00	13,00	13,33	16,00	20,00	21,67	23,33	25,67	20,83	24,33	18,44
*				*	*			5,00	13,00	10,00	12,00	10,00	14,00	16,67	19,33	20,83	24,17	24,17	26,17	18,11
*				*		*		5,00	6,00	10,00	12,00	10,00	11,33	15,00	18,33	17,50	20,50	18,33	20,00	14,69
*				*				5,00	13,00	15,00	15,00	10,00	12,00	18,33	19,67	19,17	21,67	20,83	23,33	17,44
	*	*			*			0,00	5,00	5,00	**8,00**	6,67	8,67	15,00	16,33	12,50	**16,17**	14,17	16,83	11,83
	*	*				*		5,00	9,00	5,00	9,00	6,67	8,67	13,33	17,00	19,17	19,33	20,00	21,00	14,00
	*	*						0,00	4,00	10,00	10,00	10,00	10,67	15,00	18,67	20,83	23,50	23,33	24,33	15,19
	*		*		*			5,00	6,00	10,00	10,00	10,00	12,67	13,33	17,33	16,67	18,67	16,67	18,00	13,78
	*		*			*		5,00	6,00	10,00	10,00	10,00	12,00	15,00	18,33	17,50	20,50	17,50	23,17	15,00
	*		*					10,00	10,00	11,00	11,00	10,00	12,00	18,33	19,33	17,50	22,50	24,17	26,00	16,81
	*			*	*			0,00	**0,00**	10,00	10,00	6,67	**8,00**	13,33	**15,00**	15,00	16,50	11,67	**15,33**	**10,81**
	*			*		*		5,00	7,00	10,00	11,00	10,00	14,00	16,67	19,67	18,33	20,50	14,17	23,17	15,89
	*			*				10,00	10,00	10,00	13,00	10,00	12,00	18,33	21,00	18,33	22,33	25,00	26,33	17,44
*					*		*	0,00	**0,00**	5,00	14,50	10,00	13,67	18,33	21,67	21,67	24,17	22,50	26,50	16,75
*						*	*	10,00	10,00	10,00	**12,50**	13,33	15,00	15,00	18,67	17,50	21,00	12,50	21,08	16,38
	*				*		*	0,00	**0,00**	*5,00*	13,50	*6,67*	**7,67**	*8,33*	**9,50**	*10,83*	**15,17**	*10,00*	**14,33**	**10,03**
	*					*	*	10,00	10,00	10,00	13,00	10,00	13,67	16,67	20,00	16,67	22,67	17,50	23,08	17,07

Як бачимо з табл. 3, у випадку однорідних об'єктів найкращі результати (середнє відхилення 3,81 %) отримують при відборі рулеткою, *OX*-кросинговері та проблемно-орієнтованій *WP*-мутації; близькі результати (4,04 % і 4,02 %) отримують за тих самих параметрів відбору та кросинговеру, але з використанням мутації вставкою або взагалі без використання оператора мутації. Можна твердити, що вплив методу мутації не має чітких ознак і для прискорення роботи алгоритму мутацію можна не застосовувати.

У випадку неоднорідних об'єктів, якщо немає оператора упаковки, найкращі результати (середнє відхилення 10,81 %) отримано при відборі рулеткою, *WP*-*OX*-кросинговері та мутації вставкою. Так, з аналізу табл. 4 випливає, що для чотирьох із шести тестових вибірок найкращі результати отримано саме в указаних значеннях параметрів. Також непогані результати (середнє відхилення 11,83 %) дають застосування відбору рулеткою, *OX*-кросинговера та мутації вставкою.

Можна зробити висновок, що застосування розроблених проблемно-орієнтованих операторів кросинговеру та мутації дає незначні (близько 1 %) покращення порівняно з застосуванням класичних генетичних операторів. Тому, розробляючи програмне забезпечення з використанням стандартних пакетів роботи з генетичними алгоритмами, можна знехтувати поліпшеннями від проблемно-орієнтованих операторів.

З табл. 4 бачимо, що застосування розробленого оператора упаковки надає певної нестабільності в роботі алгоритму: найкраще та середнє значення можуть суттєво різнитися. Наприклад, за відбору рулеткою і мутації вставкою для задачі C1/P2 відмінність найкращого та середнього значень становить 8,5 %, для задачі C6/P2 при елітарному відборі та мутації вставкою – 9 %. Водночас оператор упаковки в поєднанні з мутацією вставкою дає покращення результатів у середньому на 0,78 %, а для класів задач із кількістю об'єктів більше 40 (класи C4, C6) – від 1 % до 5 %. Таке покращення є суттєвим, оскільки в усіх випадках, якщо зростає кількість об'єктів в упаковці, збільшується коефіцієнт *gap*. Тому для складних задач рекомендовано застосовувати оператор упаковки, збільшуючи кількість прогонів генетичного алгоритму.

Таким чином, для розв'язку поставленої задачі за допомогою генетичного алгоритму з *IBL*-декодером можна рекомендувати наступні набори параметрів:

- для однорідних об'єктів: пропорційний відбір, *OX*-кросинговер, *WP*-мутація;
- для неоднорідних об'єктів, кількість об'єктів менше 40: пропорційний відбір, *WP*-*OX*-кросинговер, мутація вставкою;
- для неоднорідних об'єктів, кількість об'єктів більше 40: пропорційний відбір, мутація вставкою, оператор упаковки.

На рис. 1 подано результат роботи генетичного алгоритму для прикладу C6/P1 під час застосування рекомендованих значень параметрів (пропорційний відбір, мутація вставкою, оператор упаковки). Довжина результуючої розкладки – 132. Пунктирними лініями позначено межі прямокутників, суцільними – утворених за роботи алгоритму метапрямокутників.

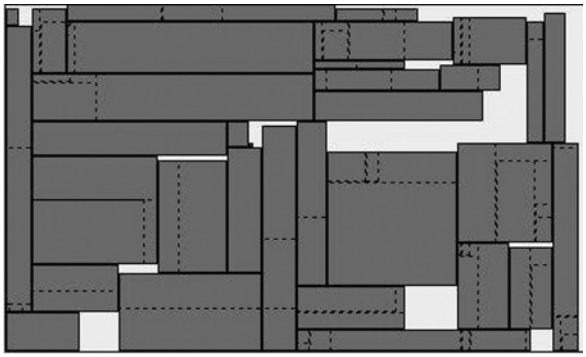

Рис. 1. Оптимальна упаковка 97 прямокутників (тест C6/P2)

Надалі на генетичний алгоритм, що розглядається тут, з рекомендованими значеннями параметрів для відповідного класу задач посилатимемось як на алгоритм *GA+IBL*. У табл. 5 подано результати порівняння роботи алгоритму *GA+IBL* на тестах Є. Гоппер із роботою інших алгоритмів, зокрема, з роботою *IBL*-евристики (*IBL-R* – на випадково згенерованій перестановці; *IBL-DW* – на упорядкованій за незростанням ширини прямокутників перестановці; *IBL-DH* – на упорядкованій за незростанням довжини прямокутників перестановці); з роботою генетичного алгоритму з використанням ефективнішого *BLF*-декодера – алгоритму *GA+BLF* ([27]); з роботою найкращих відомих нині алгоритмів розв'язку задачі ортого-

Таблиця 5. **Порівняння роботи алгоритму *GA+IBL* з іншими відомими алгоритмами (*gap*, %)**

Клас	IBL-R	IBL-DW	IBL-DH	GA+BLF	GA+IBL		R-GRASP		H-SP	
				Best	Best	Avg	Best	Avg	Best	Avg
C1	45	40	10	4	5	5	0	0	0	1.33
C3	23	50	13	5	7	8	1.08	1.08	2.22	2.22
C4	50	47	22	3	8	10	1.64	1.64	1.67	2.11
C6	48.5	56	18.5	4	10.5	14.5	0.83	1.56	1.11	1.39
Avg	41.625	48.25	15.875	4	7.625	9.375	0.8875	1.07	1.25	1.7625

нальної двовимірної упаковки із забороною повороту *GRASP* ([10]) та *H-SP* ([11]).

З табл. 5 видно, що проста IBL-евристика дає найкращий результат за упорядкування вхідної послідовності за незростанням довжини прямокутників (IBL-DH). Як і очікувалось, порівняно з *IBL-DH* генетичний алгоритм *GA+IBL* дає поліпшення в середньому на 7 %, а порівняно з *IBL-DW* та *IBL-R* – більше, ніж на 30 %. Генетичний алгоритм *GA+IBL* поступається алгоритму *GA+BLF* на 1 %–6,5 %, що пояснюється застосуванням в останньому більш досконалого декодера: експериментально показано, що ефективність роботи генетичного алгоритму суттєво залежить від використаного декодера [20]. Водночас несуттєве погіршення якості розв'язку порівняно з алгоритмом *GA+BLF* (в середньому на 3,6 %) за умов суттєво меншої складності роботи декодера ($O(m^2)$ порівняно з $O(m^3)$) дає алгоритму *GA+IBL* переваги тоді, коли розв'язок необхідно знайти досить швидко. Алгоритм *GA+IBL* поступається найкращим відомим сьогодні алгоритмам (в середньому результат є гіршим на 8 %). Утім, часова оцінка роботи та простота реалізації дають змогу припускати можливість практичного використання запропонованого підходу.

Висновки

У цьому дослідженні проаналізовано вплив застосування різноманіття спеціалізованих операторів на роботу генетичного алгоритму розв'язку задачі двовимірної ортогональної упаковки прямокутних об'єктів у напівнескінченну смугу фіксованої ширини за кодування розв'язків перестановками прямокутників та використання *IBL*-декодера. Здійснено порівняльний аналіз запропонованого алгоритму.

Алгоритм *GA+IBL* поступається найкращим відомим сьогодні алгоритмам розв'язку поставленої задачі: середні по всіх прогонах результати є гіршими на 5 %–13 %, найкращі – на 5 %–10 %. Однак він має і суттєву перевагу – хорошу ($O(m^2)$) часову складність роботи. Тому вбачаємо перспективними майбутні дослідження описаного підходу. Окреслимо можливі шляхи удосконалення:

- класифікація прямокутників, що мають бути упаковані, на однорідні/неоднорідні, великі/маленькі (можливо, шляхом відповідної модифікації функції оцінювання);
- поліпшення якості початкової популяції, наприклад, додаючи до неї найкращі особини, знайдені на попередніх прогонах алгоритму;
- модифікація декодера (зберігаючи складність $O(m^2)$);
- застосування кількох декодерів та автоматичний вибір декодера для розташування чергового об'єкта; цей підхід особливо перспективний (див. : [1; 11; 13]).

Список літератури

1. Валиахметова Ю. И. Расширение генетического алгоритма комбинирования эвристик для решения задачи прямоугольной упаковки / Ю. И. Валиахметова, Е. В.Карамова // Вестник ВЭГУ. – 2009. – № 2 (40). – С. 89–94.
2. Глибовець М. М. Застосування інфекційних алгоритмів для побудови стереопар / М. М. Глибовець, С. С. Гороховський, В. О. Михальчук // Наукові праці ЧДУ ім. Петра Могили. Комп'ютерні технології. – 2011. – Т. 160, вип. 148. – С. 99–105.
3. Головистиков А. В. Задачи двумерной прямоугольной упаковки и раскроя : обзор / А. В. Головистиков // Информатика. – 2008. – № 4. – С. 18–33.
4. Гэри М. Вычислительные машины и труднорешаемые задачи / М. Гэри, Д. Джонсон. – М. : Мир, 1982.– 416 с.
5. Канторович Л. В. Математические методы в организации и планировании производства / Л. В. Канторович. – Л. : Изд-во ЛГУ, 1939. – 68 с.
6. Мухачева А. С. Задачи двумерной упаковки : развитие генетических алгоритмов на базе смешанных процедур локального поиска оптимального решения / А. С. Мухачева, А. В. Чиглинцев, М. А. Смагин, Э. А. Мухачева // Информационные технологии. Приложение. – М. : Машиностроение. – 2001. – № 9. – 24 с.
7. Мухачева Э. А. Генетический алгоритм блочной структуры в задачах двумерной упаковки / Э. А. Мухачева, А. С. Мухачева, А. В. Чиглинцев // Информационные технологии. – М. : Машиностроение. – 1999. – № 11. – С. 13–18.
8. Норенков И. П. Эвристики и их комбинации в генетических методах дискретной оптимизации / И. П. Норенков // Информационные технологии. – 1999. – № 1. – С. 2–7.
9. Ahmad A. R. Improved Placement Algorithm for Layout Optimization / Abdul-Rahim Ahmad, Otman A. Basir, Khaled Hassanein, Muhammad Hasan Imam // Proc. of the 2nd Int'l Industrial Engineering Conf. (IEEC'04). – Riyadh, Saudi Arabia. – 2004.
10. Alvarez-Valdes R. Reactive GRASP for the strip-packing problem / R. Alvarez-Valdes, F. Parreño, J. M. Tamarit // Computers & Operations Research. – 2008.– Vol. 35, № 4. – P. 1065–1083.
11. Araya I. An Efficient Hyperheuristic for Strip Packing Problems / I. Araya, B. Neveu, M. C. Riff // Book Adaptive and Multilevel Heuristics, Series Studies in Computational Intelligence. – Springer. – 2008. – P. 61–76.
12. Burke E. K. A Hyper-Heuristic Approach to Strip Packing Problems / Edmund K. Burke, Qiang Guo, Graham Kendall // Proceedings of Problem Parallel Solving from Nature (PPSN XI), Sep 2010, Springer Berlin // Lecture Notes in Computer Science. – 2010. – Vol. 6238. – P. 465–474.
13. Burke E. K. Metaheuristic enhancements of the best-fit heuristic for the orthogonal stock cutting problem computer science / E. K. Burke, G. Kendall, G. Whitwell // Technical Report No NOTTCS-TR-2006-3. – 2006.
14. Chazelle B. The bottom-left bin packing heuristic : An efficient implementation / B. Chazelle // IEEE Trans. Comput. – 1983. – № 32(8). – P. 697–707.
15. Dyckhoff H. A typology of cutting and packing problems / H. Dyckhoff // European Journal of Operational Research. – 1990. – Vol. 44, № 2. – P. 145–159.
16. Fowler R. J. Optimal packing and covering in the plane are NP-complete / R. J. Fowler, M. S. Paterson, S. L. Tatimoto // Information Processing Letters. – 1981. – № 12. – P. 133–137.

17. Gilmore P. C. A linear programming approach to cutting-stock problem / P. C. Gilmore, R. E. Gomory // Operations Research. – 1961. – Vol. 9, № 6. – P. 849–859.
18. Gilmore P. C. A linear programming approach to cutting-stock problem. Part II / P. C. Gilmore, R. E. Gomory // Operations Research. – 1963. – Vol. 11, № 6. – P. 863–888.
19. Hopper E. A Review of the Application of Meta-Heuristic Algorithms to 2D Strip Packing Problems / E. Hopper, B. H. C. Turton // Artificial Intelligence Review. – 2001. – Vol. 16, № 4. – P. 257–300.
20. Hopper E. An Empirical Investigation of Meta-Heuristic and Heuristic Algorithms for a 2D Packing Problem / E. Hopper, B. C. H. Turton // European Journal of Operational Research. – 2000. – Vol. 128, № 1. – P. 34–57.
21. Hopper E. Application of Genetic Algorithms to Packing Problems – A Review / E. Hopper, B. Turton // Proceedings of the 2nd On-line World Conference on Soft Computing in Engineering Design and Manufacturing / eds. P. K. Chawdry, R. Roy, R. K. Kant. – London : Springer Verlag, 1997. – P. 279–288.
22. Jakobs S. On genetic algorithms for the packing of polygons / S. Jakobs // European Journal of Operational Research. – 1996. – Vol. 88. – P. 165–181.
23. Kröger B. Parallel genetic packing of rectangles / B. Kröger, P. Schwenderling, O. Vornberger // Parallel Problem Solving from Nature 1st Workshop. – Springer-Verlag. – 1991. – P. 160–164.
24. B. Kröger Guillotineable bin packing : A genetic approach / B. Kröger // European Journal of Operational Research. – 1995. – Vol. 84. – P. 645–661.
25. Liu D. An improved BL-algorithm for genetic algorithms of the orthogonal packing of rectangles / D. Liu, H. Teng // European Journal of Operational Research. – 1999. – Vol. 112, № 2. – P. 413–420.
26. Murata H. Rectangle-packing-based module placement / Hiroshi Murata, Kunihiro Fujiyoshi, Shigetoshi Nakatake, and Yoji Kakitani // Proc. IEEE/ACM Int. Conf. on Computer-Aided Design. – San Jose, – 1995. – P. 472–479.
27. Riff M. C. A revision of recent approaches for two-dimensional strip-packing problems / M. C. Riff, X. Bonnaire, B. Neveu // Engineering Applications of Artificial Intelligence. – 2009. – Vol. 22, № 4/5. – P. 823–827.
28. Wäscher G. An improved typology of cutting and packing problems / G. Wäscher, H. Haussner, H. Schumann // European Journal of Operational Research. – 2007. – Vol. 183, № 3. – P. 1109–1130.

N. Gulayeva, O. Shchur

THE ANALYSIS OF PARAMETERS OF GENETIC ALGORITHM FOR THE ORTHOGONAL STRIP PACKING PROBLEM

In this paper, approaches to solve the two-dimensional strip-packing problem (2D-SPP) via genetic algorithms are reviewed. Effect of genetic operators on the efficiency of genetic algorithm is analysed and problem-oriented genetic operators are proposed. Extensive computational experiments with well-known instances are performed and genetic algorithm parameters selection recommendations are given.

Keywords: two-dimensional orthogonal strip-packing problem, IBL-decoder, genetic algorithms, genetic operators.

Матеріал надійшов 22.03.2012