ТЕОРІЯ ОПТИМАЛЬНИХ РІШЕНЬ

УДК 519.854.33

С.В. ЧУПОВ

НАБЛИЖЕНИЙ АЛГОРИТМ ПАРАЛЕЛЬНОГО ЛЕКСИКОГРАФІЧНОГО ПОШУКУ ДЛЯ БАГАТОВИМІРНОЇ БУЛЕВОЇ ЗАДАЧІ ПРО РАНЕЦЬ ПРИ ФІКСОВАНОМУ ВПОРЯДКУВАННІ ЗМІННИХ

На основі аналізу структурних особливостей багатовимірної булевої задачі про ранець, представлено наближений алгоритм лексикографічного пошуку розв'язків високої якості, у процесі роботи якого визначення лексикографічних максимумів окремих множин здійснюється паралельно. Обгрунтовується правило вибору множин, які аналізуються алгоритмом, так щоб вони утворювали розбиття множини допустимих розв'язків задачі. Проведені експериментальні дослідження з використанням відомого тестового набору задач. Результати експериментів свідчать про високу якість, отриманих за прийнятний час, розв'язків.

Вступ. Враховуючи те, що багатовимірна булева задача про ранець (МВКР) – *NP*- складна, точні алгоритми пошуку оптимального розв'язку можуть бути застосовані для неї тільки при не великій кількості змінних. Сучасні практичні задачі містять сотні і навіть тисячі змінних. Для задач великої розмірності придатні лише наближені методи, які дозволяють знаходити «майже» оптимальні розв'язки за прийнятний час. Серед них варто виділити алгоритми, засновані на методах табу, генетичні алгоритми, гібридні алгоритми які об'єднують у собі ідеї інших методів [1 – 5]. Окремо слід зазначити алгоритм глобального рівноважного пошуку, що відноситься до класу адаптивних методів відпалу [6, 7]. Якщо булеві вектори фіксованої розмірності впорядкувати лексикографічно, тоді усі допустимі розв'язки задачі належатимуть лексикографічному проміжку $I: x^{l\min} \le^L \le^L x \le^L x^{l\max}$ [8], де $x^{l\min}$ – лексикографічний мінімум, $x^{l\max}$ – лексикографічний максимум множини допустимих розв'язків МВКР. Точний алгоритм лексикографічного пошуку здійснює напрямлений перегляд усіх придатних розв'язків [8] на інтервалі I, починаючи з $x^{l\max}$. З метою підвищення ефективності процесу пошуку якісних розв'язків, у роботі пропонується розбивати проміжок I на послідовність лексикографічних інтервалів I^k, $k = 1, ..., n$, так, щоб вона утворювала розбиття цього проміжку. Маючи розбиття

$\left\{I^1, I^2, \ldots, I^{n_x}\right\}$ лексикографічного інтервалу I, пошук якісних розв'язків на кожному інтервалі I^k, k=1,..., n_x може здійснюватись паралельно. Обираючи різні алгоритми пошуку якісних розв'язків на лексикографічних проміжках розбиття, отримуються різні схеми паралельного лексикографічного пошуку. Зокрема, у роботі описується та досліджується наближений алгоритм пошуку в якому стохастичний алгоритм лексикографічного пошуку розв'язку МВКР [9] працює як окрема гілка паралельної схеми для аналізу та виявлення розв'язків високої якості на окремих лексикографічних інтервалах з розбиття I. Останній пункт роботи представляє результати числових експериментів, які показують ефективність роботи запропонованого алгоритму.

Постановка задачі. Лексикографічно напрямлений аналіз розв'язків. Математична модель багатовимірної булевої задачі про ранець записується так:

максимізувати $$x_0 = f_0(x) \equiv \sum_{j=1}^{n} c_j x_j, \tag{1}$$

за умов $$x \in X^D, \tag{2}$$

де $X^D = \left\{ x \in B^n \middle| \sum_{j=1}^{n} a_{ij} x_j \le b_i, i = \overline{1,m} \right\}$, $a_{ij} \ge 0$, $b_i > 0$, $c_j \ge 0$, $i = 1,\ldots,m$, $j = 1,\ldots,n$, $B^n = \underbrace{\{0,1\} \times \ldots \times \{0,1\}}_{n}$ – множина усіх булевих векторів розмірності n.

На основі поняття лексикографічного максимуму множини процес точного лексикографічного пошуку оптимального розв'язку задачі (1), (2) зводиться до відшукання лексикографічних максимумів послідовності множин $X^0, X^1, \ldots, X^k, \ldots$, де $X^0 = X^D$, $X^k = \left\{ x \in X^D \middle| x \le^L x^{k-1},\ f_0(x) > x_0^{k-1} \right\}$, $k = 1,2,\ldots$, $x^k = \max^L X^k$, $x_0^k = f_0(x^k)$, $k = 0,1,\ldots$ [8]. Якщо $X^0 \ne \varnothing$, то у результаті застосування детермінованого алгоритму лексикографічного пошуку [8] будується лексикографічно спадна послідовність допустимих розв'язків $x^0 >^L x^1 >^L \ldots$ $\ldots >^L x^k >^L \ldots$, якій відповідає зростаюча послідовність $x_0^0 > x_0^1 > \ldots > x_0^k > \ldots$ значень цільової функції (1). Процес розв'язання задачі завершується як тільки на деякому кроці $k\,(k > 0)$, виявиться, що $X^k = \varnothing$. У цьому разі оптимальним розв'язком задачі (1), (2) буде вектор x^{k-1}, отриманий на попередньому кроці. Враховуючи особливості структури множини допустимих розв'язків МВКР, є можливість будувати модифіковані схеми пошуку лексикографічних максимумів множин які у декілька разів ефективніше ніж стандартний алгоритм. Тим не менш, на окремих кроках детермінованого алгоритму пошук лексикографічного максимуму деяких множин з послідовності $X^0, X^1, \ldots, X^k, \ldots$ може займати досить тривалий час, що значно впливає на загальну ефективність алгоритму лексикографічного пошуку. В роботі [9] представлено стохастичний алгоритм лексикографічного пошуку *APrLexMax*(*X*), де *X* – підмножина множини допустимих

розв'язків задачі X^D, визначена певним чином. У залежності від обраних параметрів алгоритму в процесі пошуку лексикографічних максимумів множин аналізуються не всі придатні розв'язки [8], як це робиться у детермінованому алгоритмі, а лише їх частина. Звісно, що при такому перегляді потрібний розв'язок може бути пропущений, але, як свідчать результати числових експериментів, при вдалому виборі параметрів алгоритму *APrLexMax*(*X*) у переважній більшості випадків завжди отримується розв'язок високої якості і при цьому за доволі короткий час. Усе це справедливо, якщо кількість допустимих розв'язків задачі відносно мала. При збільшенні розмірності задачі, кількість допустимих розв'язків значно збільшується. У такому випадку робота алгоритму *APrLexMax*(*X*) на лексикографічному проміжку I – мало ефективна.

Наближений алгоритм паралельного лексикографічного пошуку в фіксованому порядку. Структурні властивості множини допустимих розв'язків задачі булевого програмування, детально розглянуті в [10], до якої відноситься і задача (1), (2), дозволяють будувати наближені алгоритми лексикографічного пошуку розв'язків високої якості МВКР, у яких пошук лексикографічних максимумів окремих множин згідно детермінованого або стохастичного алгоритмів лексикографічного пошуку [8, 9] може здійснюватись паралельно. У даному пункті пропонується наближений алгоритм паралельного лексикографічного пошуку розв'язку високої якості, що працює в одному фіксованому впорядкуванні змінних задачі.

Припустимо, що для певного вектора $x \in B^n$ його *k*-а координата рівна 1, $x_k = 1$. Визначимо розв'язки $\overline{z}^k$ та $\tilde{z}^k$ у такий спосіб:

$$\overline{z}^k = (x_1,\ldots,x_{k-1},\underbrace{0,1,\ldots,1}_{n-k}),\ \tilde{z}^k = (x_1,\ldots,x_{k-1},0,\underbrace{0,\ldots,0}_{n-k}).$$

Не важко бачити, що $\tilde{z}^k <^L \overline{z}^k$. Нехай $x_{k-1} = 1$, тоді розв'язок $\tilde{z}^k$ можна представити як:

$$\tilde{z}^k = (x_1,\ldots,x_{k-1},0,\underbrace{0,\ldots,0}_{n-k}) = (x_1,\ldots,x_{k-2},1,0,\underbrace{0,\ldots,0}_{n-k}). \quad (3)$$

Якщо $x_{k-1} = 1$, тоді аналогічним чином визначаються розв'язки $\overline{z}^{k-1}$ та $\tilde{z}^{k-1}$:

$$\overline{z}^{k-1} = (x_1,\ldots,x_{k-2},\underbrace{0,1,\ldots,1}_{n-k+1}),\ \tilde{z}^{k-1} = (x_1,\ldots,x_{k-2},0,\underbrace{0,\ldots,0}_{n-k+1}).$$

Враховуючи (3) маємо строгу лексикографічну нерівність:

$$\overline{z}^{k-1} <^L \tilde{z}^k. \quad (4)$$

Легко зрозуміти, що множина булевих векторів, які належать лексикографічному проміжку $\overline{z}^{k-1} <^L x <^L \tilde{z}^k$ є порожньою, $\left\{x \in B^n \middle| \overline{z}^{k-1} <^L x <^L \tilde{z}^k\right\} = \varnothing$.

Множину допустимих розв'язків задачі (1), (2), які належать лексикографічному інтервалу $\tilde{z}^k \le^L x \le^L \overline{z}^k$ позначимо Z^k, $Z^k = \left\{x \in X^D \middle| \tilde{z}^k \le^L x \le^L \overline{z}^k\right\}$. Враховуючи лексикографічну нерівність (4), перетин множин Z^{k-1} та Z^k є порожнім, $Z^k \bigcap Z^{k-1} = \varnothing$.

Для довільного булевого вектора $x \in B^n$ визначимо множину індексів J^x не нульових координат цього розв'язку:

$$J^x = \left\{ j \in \{1,2,\ldots,n\} \middle| x_j = 1 \right\} = \left\{ j_1^x, j_2^x, \ldots, j_{n_x}^x \right\},$$

де $n_x = \sum_{j=1}^{n} x_j$ – кількість відмінних від нуля координат вектора *x*.

Теорема 1. Нехай $x \in B^n$ – довільний не нульовий булевий вектор та J^x – множина індексів не нульових координат цього вектора. Тоді множини Z^k, $k \in J^x$ утворюють розбиття множини допустимих розв'язків задачі X^D.

Доведення. Нехай для деякого значення *p*, такого що $1 < p \le n_x$, маємо індекси $k = j_p^x$ та $s = j_{p-1}^x$. Тоді $\tilde{z}^k = (x_1,\ldots,x_{k-1},0,\underbrace{0,\ldots,0}_{n-k}) = (x_1,\ldots,x_{s-1},\underset{(s)}{1},0,\ldots,0,\underset{(k)}{0},\underbrace{0,\ldots,0}_{n-k})$

та $\overline{z}^s = (x_1,\ldots,x_{s-1},\underset{(s)}{0},1,\ldots,1,\underset{(k)}{1},1,\ldots,1)$.

На підставі вищевикладених міркувань також маємо, що множина булевих розв'язків, які належать лексикографічному проміжку $\overline{z}^s <^L x <^L \tilde{z}^k$ – порожня, $\left\{ x \in B^n \middle| \overline{z}^s <^L x <^L \tilde{z}^k \right\} = \varnothing$, а отже $Z^k \bigcap Z^s = \varnothing$. Враховуючи те, що $\tilde{z}^{j_1^x} <^L \overline{z}^{j_1^x} <^L \tilde{z}^{j_2^x} <^L \ldots <^L \overline{z}^{j_{n_x}^x}$, для довільних $k \in J^x, s \in J^x, k \ne s$ має місце $Z^k \bigcap Z^s = \varnothing$. З того, що $\left\{ x \in B^n \middle| \overline{z}^{j_p^x} <^L x <^L \tilde{z}^{j_{p+1}^x} \right\} = \varnothing$, $p = 1,2,\ldots,n_x - 1$ та $Z^{j_p^x} \subset \left\{ x \in B^n \middle| \tilde{z}^{j_p^x} \le^L x \le^L \overline{z}^{j_p^x} \right\}$, $p = 1,2,\ldots,n_x$ випливає $Z^{j_1^x} \bigcup Z^{j_1^x} \bigcup \ldots$ $\ldots \bigcup Z^{j_{n_x}^x} = X^D$. Таким чином, послідовність множин Z^k, $k \in J^x$ представляє собою розбиття множини допустимих розв'язків X^D задачі (1), (2).

Обираючи різні допустимі розв'язки задачі (1), (2), отримуємо різні розбиття множини X^D.

Нехай $\hat{x}$ – найкращий, за значенням цільової функції, знайдений допустимий розв'язок задачі (1), (2) та $\hat{C}$ – значення цільової функції у цьому розв'язку. Визначимо множину $J^{\hat{x}}$ та множини Z^k, $k \in J^{\hat{x}}$ щодо розв'язку $\hat{x}$. $\hat{Z}^k$, $k \in J^{\hat{x}}$ позначимо множини розв'язків з Z^k, $k \in J^{\hat{x}}$, значення цільової функції у яких більше за $\hat{C}$:

$$\hat{Z}^k = \left\{ x \in Z^k \middle| f_0(x) > \hat{C} \right\},\ k \in J^{\hat{x}}. \tag{5}$$

Якщо деяка множина $\hat{Z}^k$ – порожня, то це означає, що у цій множині не існує допустимих розв'язків задачі з більшим за $\hat{C}$ значенням цільової функції. Якщо множина $\hat{Z}^k$ не порожня, тоді вона містить допустимий розв'язок задачі з кращим значенням цільової функції ніж у $\hat{x}$. Визначення того чи порожня множина $\hat{Z}^k$ чи ні можливо здійснювати шляхом відшукання лексикографічного

максимуму цієї множини. Враховуючи той факт, що послідовність множин Z^k, $k \in J^{\hat{x}}$ представляє собою розбиття множини X^D, є можливість організувати паралельний пошук лексикографічних максимумів множин з послідовності $\hat{Z}^k \subset Z^k$, $k \in J^{\hat{x}}$ так, що розв'язки які аналізуються у процесі такого пошуку будуть переглядатися не більше одного разу. Якщо алгоритм пошуку лексикографічних максимумів множин $\hat{Z}^k$, $k \in J^{\hat{x}}$ – детермінований [8], тоді після паралельного виконання такого пошуку для усіх множин з послідовності (5) серед знайдених допустимих розв'язків оберемо найкращий за значенням цільової функції який і буде оптимальним розв'язком задачі (1), (2). Якщо усі множини з послідовності (5) виявляться порожніми, тоді за оптимальний розв'язок обираємо $\hat{x}$.

На практиці розмірність задачі, що розв'язується, може бути досить великою. Для підвищення ефективності процесу пошуку лексикографічних максимумів множин з послідовності (5) варто використовувати наближені або стохастичні алгоритми. Але при їх використанні певна частина допустимих розв'язків може бути пропущена, у тому числі й оптимальний. Це означає, що для досягнення розв'язку високої якості, одноразового паралельного пошуку лексикографічних максимумів множин послідовності (5) (як у випадку використання детермінованого алгоритму) може виявитись не достатньо. Крім того, теоретично потрібно $\left|J^{\hat{x}}\right|$ процесорів для реалізації такого паралельного пошуку. Але у реальності можливості обчислювальної техніки є обмеженими, так як одним із критичних ресурсів обчислювальної системи є кількість незалежних фізичних процесорів. На практиці, виходячи з реальної кількості фізичних процесорів, потрібно розбити процес паралельного пошуку лексикографічних максимумів множин з послідовності (5) на послідовні серії паралельних пошуків, у кожній з яких буде використовуватись не більше заданої наперед кількості фізичних процесорів.

У формальній схемі алгоритму використовуються наступні позначення: $p_{\max}$ – максимальна кількість задач, що розв'язуються паралельно; $k_{\max}$ – максимальна кількість кроків алгоритму; $\hat{x}$ – найкращий за значенням цільової функції розв'язок задачі; $\hat{C}$ – значення цільової функції у $\hat{x}$; x – найкращий за значенням цільової функції розв'язок задачі після виконання чергового кроку алгоритму; C^x – значення цільової функції у x;. y – найкращий за значенням цільової функції розв'язок задачі після виконання чергового етапу алгоритму; C^y – значення цільової функції розв'язку y.

Алгоритм ***AParallelLexKnapsack1***($p_{\max}$, $k_{\max}$).

Крок 0.

Задаємо впорядкування змінних задачі та відшукуємо лексикографічний максимум $\hat{x}$ множини допустимих розв'язків задачі X^D у обраному порядку, $\hat{x} = \max^L X^D$. Зауважимо, що на ефективність роботи алгоритмів лексикографічного пошуку значно впливає вдалий вибір порядку в якому буде відбуватися лексикографічний спуск починаючи з лексикографічного максимуму множини

X^D. Фіксуємо значення цільової функції у $\hat{x}, \hat{C} = f_0(\hat{x})$ та $x = \hat{x}$, $C^x = \hat{C}$. Покладаємо $k = 1$ і переходимо до першого кроку.

Крок k (*k*>0).

Якщо $k > k_{\max}$, тоді припиняємо обчислення і як наближений оптимальний розв'язок обираємо останній найкращий за значенням цільової функції допустимий розв'язок $\hat{x}$.

Якщо $k \le k_{\max}$, тоді будуємо множину індексів J^x щодо розв'язку *x*, покладаємо $r = \left|J^x\right|$, $s = 1$, $y = (\underbrace{0,\ldots,0}_{n})$, $C^y = 0$. Переходимо до виконання 1-го етапу.

Eman s (*s* > 0).

Будуємо $p = \min(r, p_{\max})$ множин $\hat{Z}^{j_q^x}$, $q = r, r-1, \ldots, r-p-1$. Організовуємо паралельний пошук лексикографічних максимумів цих множин за стохастичним алгоритмом лексикографічного пошуку $A\Pr LexMax(\hat{Z}^{j_q^x})$, $q = r, r-1, \ldots, r-p-1$ [9]. Нехай $x^q = \max^L \hat{Z}^{j_q^x}$, $q = r, r-1, \ldots, r-p-1$ – розв'язки, знайдені у результаті такого пошуку та z^s – найкраща за значенням цільової функції з них, C^z – значення цільової функції розв'язку z^s. Слід зазначити, що у процесі пошуку лексикографічних максимумів множин $\hat{Z}^{j_q^x}$, $q = r, r-1, \ldots, r-p-1$ деякі з них або усі можуть виявитись порожніми. Якщо після виконання етапів чергового кроку алгоритму виявиться, що усі $\left|J^{\hat{x}}\right|$ множин $\hat{Z}^{j_q^x}$ – порожні, тоді це означає, що або розв'язок задачі потрібної якості вже знайдено або, при використанні стохастичного алгоритму лексикографічного пошуку з даними параметрами, шуканий розв'язок було пропущено. У цьому випадку або уточнюються параметри алгоритму стохастичного лексикографічного пошуку, або обирається інший придатний розв'язок *x* для початку роботи наступного кроку алгоритму.

Коректуємо значення $r = r - p$.

Якщо $C^z > \hat{C}$, тоді отримали кращий розв'язок. Покладаємо $\hat{x} = z^s$, $\hat{C} = C^z$, $x = z^s$, $C^x = C^z$, $k = k + 1$ і переходимо до наступного кроку.

Якщо $C^z > C^y$, тоді покладаємо $y = z^s$, $C^y = C^z$.

Якщо $r \ge 1$, тоді покладаємо $s = s + 1$ і переходимо до наступного етапу.

Якщо r<1, тоді покладаємо $x = y$, $C^x = C^y$, $k = k + 1$ і переходимо до наступного кроку.

На рисунку показана структурна схема запропонованого алгоритму. Функція *BuildJx*(*x*) визначає множину J^x та будує множини $\hat{Z}^{j_q^x}$, $q = 1, \ldots, n_x$.

Процедура $Parallel\begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ g_m \end{pmatrix}$ паралельно виконує дії $g_1, g_2, \ldots g_m$.

$\hat{x} = max^L X^D;\ \hat{C} = f_0(\hat{x}); x = \hat{x};\ C^x = \hat{C}; k = 1;$
$while(k \le k_{max})\{$
$J^x = BuildJx(x); r = |J^x|;\ s = 1;\ y = \bar{0};\ C^y = 0;$
$while(r \ge 1)\{$
$p = Min(r, p_{\max});$
$Parallel\begin{pmatrix} x^r & = max^L \hat{Z}^{j_r^x} \\ x^{r-1} & = max^L \hat{Z}^{j_{r-1}^x} \\ \vdots & \\ x^{r-p-1} & = max^L \hat{Z}^{j_{r-p-1}^x} \end{pmatrix};$
$r = r - p;$
$C^z = max\{f_0(x^q), q = r, \dots, r-p-1\};$
$z^s = max^L\{x \in \{x^q\}_{q=r}^{r-p-1} | f_0(x^q) = C^z\};$
$if(C^z > C^y)\{$
$y = z^s;\ C^y = C^z;$
$if(C^z > \hat{C})\{$
$\hat{x} = z^s;\ \hat{C} = C^z;$
$break;$
$\}$
$\}$
$s = s + 1;$
$\}$
$x = y; C^x = C^y;$
$k = k + 1;$
$\}$

РИСУНОК. Структурна схема алгоритму *AParallelLexKnapsack*1

Результати числових експериментів. Для аналізу ефективності роботи алгоритму паралельного лексикографічного пошуку розв'язку задачі (1), (2) *AParallelLexKnapsack*1($p_{\max}$, $k_{\max}$) використовувався набір тестових задач Chu-Beasley про багатовимірний булевий ранець з відомої бібліотеки OR–Library (*http://people.brunel.ac.uk/~mastjjb/jeb/orlib/mknapinfo.html*). У цих задачах цікавим є те, що вони розбиті на групи відповідно до щільності ненульових розв'язків у рекордному. Під щільністю значень розв'язку будемо розуміти відношення кількості ненульових значень до загальної кількості координат розв'язку, що виражене у відсотках [8]. Робота методу аналізувалася на тестових задачах у яких кількість змінних становила 100, а кількість обмежень – 5, 10 та 30. Задачі однієї розмірності, умовно, розбиті на три групи зі щільністю ненульових значень 25 % (x.100-00 – x.100-09), 50 % (x.100-10 – x.100-19) та 75 % (x.100-20 – x.100-29) у кожній групі відповідно. В таблиці наведено результати роботи наближеного алгоритму паралельного лексикографічного пошуку, f_0^{bks} – значення цільової функції задачі, отримане за генетичним алгоритмом [1], f_0^* – значення цільової функції задачі, отримане даним алгоритмом лексикографічного пошуку, *Time*(*хв.*:*сек.*) – середній час роботи наближеного алгоритму паралельного лексикографічного пошуку після 10 спроб.

ТАБЛИЦЯ. Результати розв'язання тестових задач для $n = 100$

Назва	f_0^{bks}	f_0^*	*Time*	Назва	f_0^{bks}	f_0^*	*Time*
5.100-00	24381	24381	0:22	10.100-00	23064	23064	0:30
5.100-01	24274	24274	0:58	10.100-01	22801	22801	0:13
5.100-02	23551	23551	0:16	10.100-02	22131	22131	0:49
5.100-03	23534	23534	0:04	10.100-03	22772	22772	0:17
5.100-04	23991	23991	0:05	10.100-04	22751	22751	0:10
5.100-05	24613	24613	0:03	10.100-05	22777	22777	0:16
5.100-06	25591	25591	0:00	10.100-06	21875	21875	0:31
5.100-07	23410	23410	0:24	10.100-07	22635	22635	0:35
5.100-08	24216	24216	0:09	10.100-08	22511	22511	0:12
5.100-09	24411	24411	0:01	10.100-09	22702	22702	0:07
5.100-10	42757	42757	1:12	10.100-10	41395	41395	0:12
5.100-11	42545	42545	1:14	10.100-11	42344	42344	0:01
5.100-12	41968	41968	0:51	10.100-12	42401	42401	0:05
5.100-13	45090	45090	0:09	10.100-13	45624	45624	0:15
5.100-14	42218	42218	0:19	10.100-14	41884	41884	1:28
5.100-15	42927	42927	1:08	10.100-15	42995	42995	0:00
5.100-16	42009	42009	0:28	10.100-16	43559	**43574**	0:58
5.100-17	45020	45020	0:38	10.100-17	42970	42970	1:13
5.100-18	43441	43441	0:58	10.100-18	42212	42212	0:11
5.100-19	44554	44554	0:39	10.100-19	41207	41207	1:52
5.100-20	59822	59822	0:15	10.100-20	57375	57375	0:02
5.100-21	62081	62081	1:19	10.100-21	58978	58978	0:12
5.100-22	59802	59802	1:37	10.100-22	58391	58391	0:18
5.100-23	60479	60479	1:46	10.100-23	61966	61966	0:23
5.100-24	61091	61091	0:36	10.100-24	60803	60803	0:02
5.100-25	58959	58959	0:31	10.100-25	61437	61437	0:57
5.100-26	61538	61538	1:46	10.100-26	56377	56377	0:13
5.100-27	61520	61520	1:17	10.100-27	59391	59391	0:43
5.100-28	59453	59453	0:21	10.100-28	60205	60205	0:31
5.100-29	59965	59965	1:30	10.100-29	60633	60633	0:35
30.100-00	21946	21946	0:00	30.100-15	41058	41058	0:13
30.100-01	21716	21716	0:04	30.100-16	41062	41062	8:43
30.100-02	20754	20754	0:01	30.100-17	42719	42719	0:04
30.100-03	21464	21464	0:02	30.100-18	42230	42230	0:00
30.100-04	21814	21814	0:05	30.100-19	41700	41700	0:00
30.100-05	22176	22176	0:11	30.100-20	57494	57494	0:05
30.100-06	21799	21799	0:00	30.100-21	60027	60027	0:41
30.100-07	21397	21397	2:02	30.100-22	58025	58025	3:14
30.100-08	22493	22493	7:01	30.100-23	60776	60776	0:23
30.100-09	20983	20983	0:04	30.100-24	58884	58884	0:13
30.100-10	40767	40767	0:14	30.100-25	60011	60011	0:00
30.100-11	41304	41304	0:51	30.100-26	58132	58132	0:00
30.100-12	41560	**41630**	0:53	30.100-27	59064	59064	0:00
30.100-13	41041	41041	2:09	30.100-28	58975	58975	0:05
30.100-14	40872	**40889**	1:16	30.100-29	60603	60603	0:01

Усі експериментальні результати отримані на комп'ютері з процесором *Intel*(*R*) *Core*(*TM*) *i*7-3770*K*, 3.50 *GHz* та 16 *Gb* оперативної пам'яті. Аналізуючи наведені у таблиці результати, можна зробити висновок, що для обраних тестових задач алгоритм *AParallelLexKnapsack*1($p_{\max}$, $k_{\max}$) працює досить ефективно у сенсі часу отримання рекордного значення. З 90 розв'язаних задач у трьох отримані кращі розв'язки, для інших досягнуто відомі рекорди. У процесі досліджень для кожної задачі експериментальним шляхом підбиралися значення $p_{\max}$ та $k_{\max}$. Виявилось, що для переважної більшості задач найкращими були значення $p_{\max} = 4$ та $k_{\max} = 10$. Слід відмітити, що значення $p_{\max} = 4$ збігається з кількістю ядер процесора. Також з'ясувалося, що ефективність роботи алгоритму значно залежить від структури задачі, зокрема від щільності значень розв'язку [8]. Для усіх задач рекордне значення отримане не більше ніж за 10 кроків роботи алгоритму. Так для задач із щільністю 25 % у середньому знадобилося провести не більше 10 етапів кожного кроку для досягнення рекордного значення, для задач із щільністю 50 % – 25 етапів, а для задач із щільністю 75 % – 20 етапів. Але при збільшенні розмірності задачі, зокрема кількості змінних, час отримання оптимального розв'язку значно зростає. Це пов'язано з тим, що при збільшенні номера етапу потужність множин Z^k, $k \in J^x$, що аналізуються на цьому етапі значно зростає. Час роботи алгоритму можна значно зменшити, якщо збільшити значення параметрів *tabuMax* або f_t^{lim} алгоритму стохастичного пошуку $APrLexMax(X)$ [9]. Але при такому збільшенні є велика імовірність того, що якісний розв'язок буде пропущений.

Висновки. Аналіз числових експериментів показав достатню ефективність роботи запропонованого алгоритму як за якістю розв'язків так і за часом їх отримання, особливо при щільності значень розв'язків 25 % та 75 %. Але щоб досягти задовільного часу при потрібній точності роботи для задач більшої розмірності та довільної щільності розв'язків, у схему алгоритму потрібно внести певні зміни. Алгоритм *AParallelLexKnapsack1(* $p_{\max}$, $k_{\max}$ *)* здійснює пошук лише в одному лексикографічному порядку. В деяких задачах оптимальний розв'язок знаходиться на значній лексикографічній відстані від початкового. Тому рухаючись лише в одному напрямку лексикографічного спадання, час досягнення оптимального розв'язку може бути дуже великим. Якщо ж у процесі роботи алгоритму на певних кроках змінювати напрямок лексикографічного руху, можна потрапити на такий лексикографічний порядок у якому оптимальний розв'язок знаходиться лексикографічно близько від початкового. Опинившись у такій ситуації, алгоритм швидко досягне потрібного результату.

С.В. Чупов

ПРИБЛИЖЕННЫЙ АЛГОРИТМ ПАРАЛЛЕЛЬНОГО ЛЕКСИКОГРАФИЧЕСКОГО ПОИСКА ДЛЯ МНОГОМЕРНОЙ БУЛЕВОЙ ЗАДАЧИ О РАНЦЕ ПРИ ФИКСИРОВАННОМ УПОРЯДОЧЕНИИ ПЕРЕМЕННЫХ

На основе анализа структурных особенностей задачи о многомерном булевом ранце, представлен алгоритм лексикографического поиска, в процессе работы которого определение лексикографических максимумов отдельных множеств осуществляется параллельно. Обосновывается

правило выбора множеств, которые анализируются алгоритмом, так чтобы они образовывали разбиение множества допустимых решений задачи. Проведены экспериментальные исследования с использованием известного тестового набора задач. Результаты экспериментов свидетельствуют о высоком качестве, полученных за приемлемое время решений.

S.V. Chupov

THE APPROXIMATE ALGORITHM OF PARALLEL LEXICOGRAPHICAL SEARCH FOR THE MULTIDIMENSIONAL BOOLEAN KNAPSACK PROBLEM WITH A FIXED ORDERING OF VARIABLES

On the basis of an analysis of the structural features of the multidimensional boolean knapsack problem it is presented the algorithm of lexicographic search in the course of work of which the determination of lexicographic maxima of separate sets is carried out in parallel. The rule of the selection of the sets, which are analyzed by the algorithm, so that they form a partition of the set of feasible values of the problem, is substantiated. Experimental researches using the known test set of problems have been conducted. The results of the experiments testify to the high quality of solutions obtained within a reasonable time.

1. *Chu P.C., Beasley J.E.* A Genetic Algorithm for the Multidimensional Knapsack Problem. *Journal of Heuristics*. 1998. **4**. P. 63 – 86.
2. *Balev S., Yanev N., Freville A., Andonov R.* A dynamic programming based reduction procedure for the multidimensional 0-1 knapsack problem. Eur. J. Oper. Res.2008. **186**. P. 63–76.
3. *Glover F., Kochenberger G.A.* Critical event tabu search for multidimensional knapsack problems. I.H. Osman, J.P. Kelly, eds., Metaheuristics: Theory and Applications. Kluwer Academic Publishers. 1996. P. 407 – 427.
4. *Vasquez M., Hao J.-K.* A hybrid approach for the 0-1 multidimensional knapsack problem. Proceedings of the Int. Joint Conference on Artificial Intelligence. Seattle, Washington. 2001. P. 328 – 333.
5. *Vasquez M., Vimont Y.* Improved results on the 0-1 multidimensional knapsack problem. Eur. J. Oper. Res. 2005. **1**(165). P. 70 – 81.
6. *Сергиенко И.В., Шило В.П.* Задачи дискретной оптимизации. *Проблемы, методы решения, исследования.* К.: Наук. думка, 2003. 264 с.
7. *Шило В.П.* Решение многомерных задач о ранце методом глобального равновесного поиска. *Теорія оптимальних рішень.* К.: Ін-т кібернетики імені В.М. Глушкова НАН України, 2000. С. 10 – 13.
8. *Чупов С.В.* Новые подходы к решению задач дискретного программирования на основе лексикографического поиска. *Кибернетика и системный анализ*. 2016. № 4. С. 43 – 54.
9. *Чупов С.В.* Алгоритм стохастичного лексикографічного пошуку оптимального розв'язку булевої задачі про ранець. Математика. Інформаційні технології. Освіта: матеріали V Міжнародної науково-практичної конференції (5 – 7 червня 2016 р.). Луцьк: Вежа-Друк, 2016. С. 51 – 52.
10. *Чупов С.В.* Структурні та стохастичні властивості алгоритму лексикографічного пошуку розв'язку задачі дискретної оптимізації. *Компьютерная математика*. К.: Ін-т кібернетики імені В.М. Глушкова НАН України. 2016. № 1. С. 155 – 164.

Одержано 07.03.2017