УДК 681.3

А.В. Панішев, д.т.н., проф.
Житомирський державний технологічний університет
О.М. Подоляка, ст. викл.
Національний аерокосмічний університет «ХАІ»
О.О. Подоляка, к.т.н., доц.
Харківський національний автомобільно-дорожній університет

АЛГОРИТМ ПОШУКУ МНОЖИНИ ВСІХ РОЗВ'ЯЗКІВ ЗАДАЧІ ПРО ПРИЗНАЧЕННЯ

Пропонується ефективний поліноміальний алгоритм пошуку множини розв'язків задачі про призначення, ідея якого полягає у відшуканні множини елементів матриці призначень, які формують оптимальні розв'язки.

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими та практичними завданнями. Необхідно знайти всі довершені паросполучення (перестановки) $\pi^* \in P$ з мінімальною вагою для заданої матриці β^{MM}, де M – порядок матриці:

$$\rho \ (\pi^*) = \min_{\pi \in \rho} \ \rho \ (\pi), \qquad (1)$$

де: $\rho(\pi) = \sum_{i,j \in \pi} \beta_{ij}$ – вага паросполучення; β_{ij} – елемент матриці, що належить допустимому паросполученню.

Слід зазначити, що знаходження всіх розв'язків задачі у більшості випадків має тільки науковий інтерес. Проте при вирішенні багатьох практичних задач має сенс з множини всіх оптимальних розв'язків сформувати підмножину розв'язків, які задовольняють другому тощо за значущістю критерію, або довести, що такої підмножини не існує.

Одним з ключових положень даної роботи є наступне твердження: пошук елементів матриці, які належать множині оптимальних розв'язків, може бути виконаний за поліноміальний час. Ці елементи утворюють *оптимальну множину*. Зазначимо, що процес побудови розв'язків на основі оптимальної множини у загальному випадку не є поліноміальним.

Основна увага у роботі приділяється знаходженню множини всіх оптимальних розв'язків задачі про призначення. В основу алгоритму розв'язання даної задачі покладено вперше сформульовану і доведену теорему заборон, що показує, які елементи матриці не входять до оптимальних розв'язків і отже можуть бути заборонені. Обчислювальна складність алгоритму складає $n^3 \mathrm{Log}(n)$.

У роботі показано, що значення заборонених елементів матриці можуть бути зменшені. Зменшення заборонених елементів матриці названо *нормалізацією*. Також у роботі сформульовано і доведено теорему нормалізації, яка визначає мінімальну величину забороненого елемента, при якій оптимальна множина залишається незмінною. У цьому випадку заборонений елемент може бути представлений у вигляді *близького* числа, яке утворюють сума основної (*pri*) і додаткової частини (*ext*), де додаткова частина є добутком нескінченного малого числа ε на числовий коефіцієнт. Звичайне число можна розглядати як близьке, у якого коефіцієнт при ε дорівнює нулю.

Основна частина.

Очевидні твердження

Оптимальний розв'язок неможливо поліпшити. Тобто будь-яка допустима заміна (перестановка) елементів оптимального розв'язку не призводить до його поліпшення, у кращому випадку заміна може дати еквівалентний розв'язок.

Твердження 1. Видалення рядка і стовпця елемента оптимального розв'язку

Якщо викреслити рядок і стовпець якого-небудь елемента оптимального розв'язку, то елементи розв'язку, що залишилися, формують оптимальний розв'язок в отриманій підматриці.

Твердження 2. Заміна пар елементів

Розглянемо рисунок, на якому схематично представлена матриця β.

Рядок/ Стовпець	b		d
a	$\beta_{a,b}$	.	$\beta_{a,d}$
.	.		.

c	$\beta_{c,b}$	.	$\beta_{c,d}$

Якщо в оптимальному розв'язку матриці β замінити будь-які два елементи за правилом $\beta_{a,b} \wedge \beta_{c,d} \Rightarrow \beta_{a,d} \wedge \beta_{c,b}$, то отриманий розв'язок може бути:

1. еквівалентним оптимальним розв'язком, що відрізняється від початкового двома елементами;
2. допустимим (не оптимальним) розв'язком.

Ця заміна називається *перехресною* або заміною по два. Вона давно відома і зазвичай використовується при розв'язанні задачі комівояжера і задачі про призначення.

Висновок твердження 2

Якщо $\beta_{a,d} \wedge \beta_{c,b}$ – елементи оптимального розв'язку, а $\beta_{a,b} \wedge \beta_{c,d}$ – замінювані елементи, то $\beta_{a,b} \wedge \beta_{c,d} \le \beta_{a,d} \wedge \beta_{c,b}$; $\beta_{a,b} + \beta_{c,d} \le \beta_{a,d} + \beta_{c,b}$; або

$$\left(\beta_{a,b} - \beta_{a,d}\right) \le \left(\beta_{c,d} - \beta_{c,b}\right). \tag{2}$$

Необхідно зазначити, що в початковій матриці можуть існувати підматриці порядку $K \in [3, M]$, в яких можуть існувати заміни елементів по K [1, 2, 3].

Лінійні різниці

Лінійною різницею i-го та k-го рядків матриці називатимемо елемент, який обчислюється за формулою $j \in [0, M-1]$. Зменшуване значення називатимемо клієнтським значенням лінійної різниці, а значення, що віднімається, – серверним значенням лінійної різниці. Мультимножина лінійних різниць (МмЛР) – це упорядкована послідовність лінійних різниць. Як основний критерій упорядкування виступає величина лінійної різниці. Всю мультимножину упорядковуватимемо за зростанням значень елементів.

Лінійну різницю називатимемо *найбільшою*, якщо їй відповідає найбільший елемент МмЛР. Клієнтське значення називатимемо *найбільшим*, якщо воно формує найбільшу лінійну різницю МмЛР.

Лінійну різницю називатимемо *максимальною*, якщо їй відповідає один з рівних максимальних елементів, і при цьому в МмЛР існує відмінний від максимального елемент. Клієнтське значення називатимемо *максимальним*, якщо йому відповідає максимальний елемент МмЛР.

Визначення найменшого і мінімального значення вводиться так само.

Теорема 1. Заборона

Якщо в МмЛР рядків r та i є найбільший елемент $D_j = \beta_{rj} - \beta_{ij}$, то елемент β_{rj} не може формувати оптимальний розв'язок задачі й отже може бути заборонений.

Теорема 2. Заборона

Якщо клієнтське значення β_{ij} є максимальним в МмЛР, то елемент β_{rj} не може формувати оптимальний розв'язок задачі й отже може бути заборонений.

Зауваження. Дану теорему має значення застосовувати, якщо є багато МмЛР, в яких є множина однакових максимальних елементів. Цьому випадку відповідає матриця з множиною елементів, що повторюються, які знаходяться в прямокутних підматрицях.

Ролі пар елементів, які створюють лінійні різниці

Зрозуміло, що після застосування теореми 1 деякі елементи матриці будуть заборонені. І в МмЛР з'являться лінійні різниці, які матимуть заборонені елементи. Класифікуємо ці різниці й визначимо критерії їх упорядкування в МмЛР.

Нехай кожний елемент матриці містить:

1. Значення;
2. Дозвіл;
3. Індекс стовпця елемента в матриці (щоб після упорядкування елементів МмЛР визначити місцеположення елемента у початковій матриці).

Якщо дозвіл елемента a відповідає false (або 0), то елемент матриці є забороненим і позначається $\overline{a}$, отже він не може утворювати оптимальний розв'язок.

Нехай a і c – пара елементів, що знаходяться в різних рядках і в одному стовпці матриці. Тоді можливі такі комбінації дозволів:

№	1 (агент)	2 (клієнт)	3 (сервер)	4 (спонжер)
Дозвіл a	1	1	0	0
Дозвіл c	1	0	1	0

Ролі пар елементів визначимо стосовно рядка, в якому знаходиться елемент, використовуючи відому архітектуру клієнт-сервер.

Тоді вважатимемо, що елементи $a_i = 1 \wedge c_i = 0$ виступають у ролі клієнтів, оскільки для того, щоб елемент a_i належав розв'язку, необхідний відповідний елемент $c_j = 1, j \neq i$. Можна сказати, що клієнт a запитує послугу (допустимий шлях) у серверу c.

Тоді пара, що відповідає випадку 1, виступає у ролі клієнта і сервера. Таку пару називатимемо *агентом*. Відповідно пару 2 – *клієнтом*, пару 3 – *сервером*. Пара 4 не надає елементів розв'язку і тому є паразитичною. Таку пару називатимемо *спонжером*. Спонжери бажано зберігати в окремій мультимножині, оскільки теорему 1 (заборона) не має сенсу застосовування для спонжерів.

Упорядкування елементів МмЛР

Побудова МмЛР відбувається на основі двокритеріального упорядкування. За основний критерій виступає значення лінійної різниці. Другий критерій використовується, якщо значення лінійних різниць рівні, але вони виступають в різній ролі. Цей критерій будується з урахуванням того, щоб елементи лінійних різниць, що створюють дозволені й заборонені шляхи, формували кластери.

Тоді, якщо лінійні різниці рівні, то: *спонжер>сервер > агент > клієнт*.

Алгоритм побудови мультимножини лінійних різниць

Нехай M – кількість рядків, а N – кількість стовпців матриці.

Щоб побудувати всю МмЛР, необхідно для сполучень C_M^2 пар індексів рядків $(s_1, s_2) \in C_M^2$ сформувати мультимножину лінійних різниць.

Код алгоритму має вигляд:

```
for (s1 = 0; s1 < M-1; ++s1)
    for (s2 = s1+1; s2 < M; ++s2)
        test_stirng_s(s1,s2)
```

Зауваження: Метод test_stirng_s(s1,s2) можна реалізувати так, що він будуватиме одну мультимножину для обох пар (s_1, s_2) і (s_2, s_1). У цьому випадку, алгоритм пошуку необхідних елементів у МмЛР ускладнюється, оскільки в упорядкуванні елементів враховуються їх ролі.

Обчислювальна складність алгоритму побудови мультимножини лінійних різниць

Обчислювальна складність побудови сполучень $C_M^2 = \frac{1}{2} M(M-1) = O(M^2)$ [4, 6].

Обчислювальна складність побудови окремого МмЛР становить $N\log(N)$.

Якщо матриця квадратна (M=N), то обчислювальна складність алгоритму дорівнює $M^3 \log(M)$.

Теорема 3. Нормалізації забороненого елемента матриці

Нормалізація – процес зменшення значень елементів матриці, при якому оптимальна множина не змінюється. Нормалізація допустима тільки для заборонених елементів матриці. Тому теореми заборони і нормалізації тісно пов'язані. По суті нормалізація має на увазі дві дії: заборона елемента і зменшення його значення.

Нехай:

β – початкова матриця і $\overline{\beta_{rc}}$ – її заборонений елемент, який знаходиться в рядку r і в стовпці c;

β_{rj} – дозволені елементи рядка r, $j \in [0, N-1]$;

β_{ic} – дозволені елементи стовпця c, $i \in [0, M-1]$;

ε – нескінченно мала величина.

Тоді

$$\overline{\beta_{rc}} = \beta_{rc}^{Norm} = \max_{\substack{i,j \\ \beta_{rj} + \beta_{ic} < \beta_{rc} + \beta_{ij}}} \left[\beta_{rj} + \beta_{ic} - \beta_{ij} \right] + \varepsilon = \beta'_{rj} + \beta'_{ic} - \beta'_{ij} + \varepsilon . \tag{3}$$

Елемент β'_{ij} назвемо *нормалізатором*, оскільки за індексами елемента (r, c) і його нормализатора (i , j) можна визначити нормалізоване значення β_{rc}^{Norm}. Елемент $\overline{\beta_{rc}}$ може мати кілька нормалізаторів, оскільки у матриці може бути декілька рівних максимальних значень, що задовольняють формулу (3).

Слід зазначити, що як ознака наявності нескінченно малої складової у елемента матриці може виступати логічне значення, яке використовується для позначення дозволу цього елемента.

Твердження 3. Про допустимий діапазон значень забороненого елемента

Нехай елемент $\overline{\beta_{rc}} = a$ є забороненим. Якщо значення $\overline{\beta_{rc}}$ належатиме інтервалу $[a, \infty]$, то оптимальна множина не зміниться. Інакше кажучи, значення забороненого елемента можна збільшити у разі необхідності.

Значення забороненого елемента матриці має сенс збільшувати, якщо необхідно зменшити значення лінійної різниці МмЛР, в якій він є серверним значенням.

Твердження 4. Про дозвіл забороненого елемента

Якщо дозволити елемент $\overline{\beta_{rc}} = a + \varepsilon$ і зменшити на ε його значення, то довжина оптимального розв'язку задачі не зміниться.

Теорема 4. Нормалізація найбільшого елемента МмЛР

Нехай:

β – початкова матриця;

Ms – мультимножина рядків r та i;

$Ms_{max^1} = \beta_{rk} - \beta_{ik}$ – найбільший елемент Ms (найбільша лінійна різниця), k – індекс стовпця елемента Ms_{max^1};

Ms_{max^2} – наступний за найбільшим елемент Ms;

Тоді

$$\overline{\beta_{rk}} = \beta_{ik} + Ms_{\max^2} + \varepsilon . \qquad (4)$$

Після застосування теореми 4 значення найбільшого елемента буде зменшене до значення наступного елемента мультимножини. Тому в кожному МмЛР будуть, як мінімум, дві лінійні різниці, що відрізняються на нескінченно малу величину ε.

Алгоритм нормалізації найбільшого елемента МмЛР має постійну обчислювальну складність.

Теорема 5. Нормалізація максимальних спонжерів

Нехай:

β – початкова матриця;

$SpMs$ – мультимножина спонжерів рядків r та i матриці β;

$SpMs_k = \overline{\beta_{rk}} - \overline{\beta_{ik}}$ – k-тий елемент $SpMs$;

Ms – мультимножина серверів, агентів і клієнтів рядків r та i;

Ms_{max} – максимальний елемент Ms;

Якщо $SpMs_k \geq Ms_{max}$, то

$$\overline{\beta_{rk}} = \overline{\beta_{ik}} + Ms_{max} + \varepsilon . \qquad (5)$$

Дана теорема може бути сформульована наступним чином: якщо на початку МмЛР є кілька спонжерів, наступним за якими йде менший за значенням елемент, то клієнтські значення цих спонжерів можна зменшити за формулою (5).

Обчислювальна складність алгоритму нормалізації спонжерів на підставі теореми 4, в загальному випадку, є лінійною і отже вона не перевершує обчислювальної складності упорядкування елементів МмЛР. Тому обчислювальна складність загального алгоритму побудови МмЛР та їх нормалізації складає $M^3 \log(M)$. Цей загальний алгоритм назвемо нормалізаційним.

Приклад розв'язання задачі

Розглянемо матрицю β:

3	2	1	3	4	1	7	1
2	3	2	3	1	3	2	6
7	7	2	5	4	5	7	2
3	4	3	2	4	5	7	3
1	7	1	6	3	2	7	1
2	1	3	7	1	6	5	5
5	2	4	1	4	1	7	6
3	2	6	2	2	7	3	3

Виконаємо побудову МмЛР нульового і першого рядка. Лінійні різниці цих рядків матимуть такий вигляд:

стовпець	0	1	2	3	4	5	6	7
знач.	1	1	-1	0	3	-2	5	-5

Тепер відсортуємо за незростанням отримані лінійні різниці. МмЛР матиме вигляд:

стовпець	6	4	0	1	3	2	5	7
знач.	5	3	1	1	0	-1	-2	-5

У мультимножині є один найбільший елемент в стовпці 6 і наступний за значенням елемент у стовпці 4.

За теоремою 1 найбільший елемент слід заборонити, а його значення можна зменшити за теоремою 4. Звідси $\overline{\beta_{06}} = \beta_{16} + (\beta_{04} - \beta_{14}) = 2 + 3 = 5$.

Операції побудови МмЛР, заборони і нормалізації найбільших елементів слід виконати для всіх сполучень рядків і стовпців матриці (див. алгоритм побудови мультимножини лінійних різниць). Якщо у побудованій мультимножині з'являться максимальні спонжери, їх слід нормалізувати за теоремою 5. Виконання перелічених операцій являє собою нормалізаційний алгоритм розв'язання задачі.

Пошук розв'язання в кластерних матрицях

Після розв'язання задачі нормалізаційним алгоритмом ми одержуємо наступну матрицю Mn :

1(0)	2(1)	1(1)	0(0)	2(0)	1(1)	3(0)	1(1)
2(1)	1(0)	2(1)	1(0)	1(1)	1(0)	2(1)	2(0)
2(0)	3(0)	2(1)	1(0)	3(0)	2(0)	4(0)	2(1)
3(1)	4(0)	3(1)	2(1)	4(0)	3(0)	5(0)	3(1)
1(1)	2(0)	1(1)	0(0)	2(0)	1(0)	3(0)	1(1)
2(1)	1(1)	2(0)	1(0)	1(1)	1(0)	2(0)	2(0)
2(0)	2(1)	2(0)	1(1)	2(0)	1(1)	3(0)	2(0)
3(1)	2(1)	3(0)	2(1)	2(1)	2(0)	3(1)	3(1)

де в кожній комірці вказано значення елемента, а у круглих дужках – його дозвіл (0 – елемент заборонений, 1 – дозволений).

Подальша нормалізація неможлива, оскільки у кожному МмЛР знаходяться рівні за значенням максимальні агенти, які приховують розв'язок задачі. Щоб вирішити цю проблему необхідно дослідити побудовану матрицю. Для цього виконаємо процедуру віднімання мінімальних елементів з рядків і стовпців матриці за допомогою процедури find_min_matrix. Нагадаємо, що коли з рядка або стовпця матриці відняти константу, то елементи розв'язку не зміняться, а довжина розв'язку зміниться на цю константу (див. Угорський метод).

Мінімальні елементи стовпців матриці Mn наступні:

1	1	1	0	1	1	2	1

Мінімальні елементи рядків після віднімання мінімальних елементів стовпців будуть такими:

0	0	1	2	0	0	0	1

Після виконання процедури отримаємо мінімальну еквівалентну матрицю, в якій однакові значення знаходяться в квадратних підматрицях.

0	1	0	0	1	0	1	0	d
1	0	1	1	0	0	0	1	e
0	1	0	0	1	0	1	0	d
0	1	0	0	1	0	1	0	d
0	1	0	0	1	0	1	0	d
1	0	1	1	0	0	0	1	e
1	1	1	1	1	0	1	1	f
1	0	1	1	0	0	0	1	e
a	b	a	a	b	c	b	a	

Введемо визначення. Два рядки (стовпця) матриці лінійно рівні, якщо в МмЛР рівні всі лінійні різниці. Нехай процедура пошуку індексів лінійно рівних рядків і стовпців іменується find_equal_lines. Лінійно рівні рядки і стовпці позначені однаковими буквами.

Далі виконаємо процедуру find_cluster_matrix побудови кластерної підматриці. Вона спочатку знаходить множину лінійно нерівних рядків (a, b, c) і стовпців (d, e, f), на підставі яких будується кластерна матриця. Елементи цієї матриці виділені кольором. Ці елементи називатимемо кластерними елементами, або просто кластерами. Слід зазначити, що у деяких випадках кластерна підматриця може бути прямокутною. Проте матрицю можна зробити квадратною за допомогою групування відповідних рядків.

У нашому випадку ми одержуємо квадратну матрицю.

0	1	0
1	0	0
1	1	0

Оскільки ми виключили значення, що повторюються, то в МмЛР може бути або мало максимальних агентів, або їх не буде взагалі. Отже шукаємо розв'язок у кластерній підматриці за допомогою процедури нормалізаційного алгоритму normalize:

0	1	0	d4
1	0	0	e3
1	1	0	f1
a4	b3	c1	

Введені позначення, наприклад d4 означають, що кількість лінійно рівних рядків d дорівнює 4.

Елементи розв'язку (кластери) виділені сірим кольором. Якщо всім знайденим елементам відповідають квадратні кластери (підматриці), це означає, що знайдено оптимальний розв'язок основної задачі. В іншому випадку (коли кластери не квадратні), для заборони елементів матриці можна скористатися теоремою 2. У даному прикладі всі кластери квадратні, оскільки розв'язок записується у вигляді {(d4,a4), (e3,b3) (f1,c1)}. Попередньо можна сказати, що розв'язок формують два «незалежні кластери» (d4,a4) і (e3,b3) та один загальний для всіх розв'язків елемент (f1,c1), який, по суті, також є кластером. Всі перестановки елементів кластера еквівалентні.

Оскільки у кожному кластері можуть бути заборонені елементи, то необхідно заборонити дозволені елементи, які не утворюють допустимих перестановок. Тобто кожний кластер необхідно перевірити за допомогою нормалізаційного алгоритму. Ці операції виконує процедура clear_cluster. Роботу процедури розглянемо на прикладі кластера (d4, a4).

1(0)	1(1)	0(0)	1(1)
2(0)	2(1)	1(0)	2(1)
3(1)	3(1)	2(1)	3(1)
1(1)	1(1)	0(0)	1(1)

Виділені елементи даної матриці слід заборонити. Для цього можна побудувати МмЛР по стовпцях (нормалізаційний алгоритм), або звернути увагу, що елемент (2,2) матриці є єдиним дозволеним у другому стовпці, а отже решту дозволених елементів другого рядка можна заборонити. Тому можна стверджувати, що елементи (2,2) і (3,0) є загальними для всіх розв'язків задачі, а отже ранг початкового кластера стане рівний двом.

У кінці необхідно відобразити знайдений розв'язок на початкову матрицю за допомогою процедури represent_cluster_solution і заборонити відповідні елементи. У результаті одержуємо наступну нормалізовану матрицю, у якій різні кластери і загальні елементи всіх розв'язків виділені кольором:

1(0)	2(0)	1(1)	0(0)	2(0)	1(0)	3(0)	1(1)	d
2(0)	1(0)	2(0)	1(0)	1(1)	1(0)	2(1)	2(0)	e
2(0)	3(0)	2(1)	1(0)	3(0)	2(0)	4(0)	2(1)	d
3(0)	4(0)	3(0)	2(1)	4(0)	3(0)	5(0)	3(0)	d
1(1)	2(0)	1(0)	0(0)	2(0)	1(0)	3(0)	1(0)	d
2(0)	1(1)	2(0)	1(0)	1(1)	1(0)	2(0)	2(0)	e
2(0)	2(0)	2(0)	1(0)	2(0)	1(1)	3(0)	2(0)	f
3(0)	2(1)	3(0)	2(0)	2(1)	2(0)	3(1)	3(0)	e
a	b	a	a	b	c	b	a	

Відображення розв'язків на початкову матрицю має вигдяд:

3	2	1	3	4	1	7	1	d
2	3	2	3	1	3	2	6	e
7	7	2	5	4	5	7	2	d
3	4	3	2	4	5	7	3	d
1	7	1	6	3	2	7	1	d
2	1	3	7	1	6	5	5	e
5	2	4	1	4	1	7	6	f
3	2	6	2	2	7	3	3	e
a	b	a	a	b	c	b	a	

Елементи розв'язків (оптимальної множини) виділені кольором, невиділені елементи є забороненими.

Тепер можна вивести всі розв'язки задачі. Ці дії виконує рекурсивна процедура print_solution. Алгоритм процедури побудови паросполучення можна знайти у [3]. Оскільки число розв'язків у загальному випадку не поліноміальне, обчислювальна складність процедури буде також неполіноміальною. Але у переважній більшості практичних задач число оптимальних розв'язків не перевершує M^2 (де M – порядок матриці), тому середню обчислювальну складність процедури print_solution можна вважати поліноміальною.

Результати розв'язання задачі.

Оптимальна множина, яка містить елементи оптимальних розв'язків:

{(0,2), (0,7), (1,4), (1,6), (2,2), (2,7), (3,3), (4,0), (5,1), (5,4), (6,5), (7,1 , (7,4), (7,6)}.

Довжина оптимального розв'язку S = 12.

Список всіх оптимальних розв'язків, побудованих на основі оптимальної множини:

1. { (0,7), (1,6), (2,2), (3,3), (4,0), (5,4), (6,5) (7,1) };
2. { (0,2), (1,6), (2,7), (3,3), (4,0), (5,4), (6,5) (7,1) };
3. { (0,7), (1,6), (2,2), (3,3), (4,0), (5,1), (6,5) (7,4) };

4. { (0,7), (1,4), (2,2), (3,3), (4,0), (5,1), (6,5) (7,6) };
5. { (0,2), (1,6), (2,7), (3,3), (4,0), (5,1), (6,5) (7,4) };
6. { (0,2), (1,4), (2,7), (3,3), (4,0), (5,1), (6,5) (7,6) }.

Взагалі, список оптимальних розв'язків задачі можна зменшити, якщо накласти додаткові обмеження на елементи оптимальної множини. У якості такого обмеження може виступати обмеження на час виконання робіт, або призначення певних робіт на деякі машини й ін. Нехай, у нашому випадку, тривалість кожної роботи не повинна перевищувати двох одиниць часу. Після накладення обмеження довжина списку оптимальних розв'язків буде дорівнювати чотирьом, оскільки елемент (7,6) матриці у цьому випадку повинен бути забороненим.

Слід зазначити, що жорсткі обмеження часто не призводять до зменшення числа оптимальних розв'язків. Наприклад якщо обмеження на тривалість робіт встановити рівним одиниці, то після заборони відповідних елементів матриці неможливо буде побудувати жодного довершеного паросполучення. Тобто при даному обмеженні на тривалість виконання робіт не існує оптимальних розв'язків, які задовольняють основному функціоналу (сумарна мінімальна тривалість робіт). Тому в даному випадку діє пріоритет основного функціонала, за яким число розв'язків дорівнює шести.

Таким чином, розглянутий алгоритм доцільно застосовувати для розв'язання задачі про призначення, яка має множину упорядкованих за значимістю критеріїв або функціоналів.

Висновки. Основні характеристики запропонованого алгоритму:

1. Алгоритм гарантує знаходження за поліноміальним часом всіх елементів матриці призначень (оптимальної множини), які формують оптимальні розв'язки. Процедура виводу всіх розв'язків на основі оптимальної множини, у загальному випадку, не є поліноміальною.
2. Пошук одного розв'язку задачі на основі оптимальної множини може бути виконаний за поліноміальний час.
3. Алгоритм можна використовувати для розв'язання задачі про призначення, яка має множину упорядкованих за значимістю критеріїв або функціоналів.
4. У функціоналі задачі може бути використана не тільки операція підсумовування, але і будь-яка інша бінарна операція, результати якої можна порівнювати. Наприклад операція обчислення добутку, якщо роботи позв'язані відношенням набагато більш сильним, ніж просте підсумовування.

ЛІТЕРАТУРА:

1. *Панішев А.В., Подляка О.О.* Одне з узагальнень задачі про призначення з обмеженнями // Вісник ЖІТІ. – Житомир: ЖІТІ. – 1999. – № 11. – С . 139–144.
2. *Костикова М.В., Панишев А.В., Плечистый Д.Д.* Комбинация локального поиска и схемы построения оптимальных подпоследовательностей в задачах выбора и назначения // Искусственный интеллект. – 2003. – № 4. – С. 113–117.
3. *Альфред В. Ахо, Джон Э. Хопкрофт, Джеффри Д. Ульман.* Структуры данных и алгоритмы. – М.: Вильямс, 2000. – 384 с.
4. *Андерсон, Джеймс А.* Дискретная математика и комбинаторика. – М.: Вильямс, 2004. – 960 с.
5. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. – М.: МЦНМО, 2000. – 956 с.
6. Дискретная математика для программистов / Ф.А. Новиков. – С.-Пб.: Питер, 2001. – 304 с.

ПАНІШЕВ Анатолій Васильович – доктор технічних наук, професор, завідувач кафедри інформатики і комп'ютерного моделювання Житомирського державного технологічного університету.

Наукові інтереси:
- математичне моделювання;
- теорія розкладів та її застосування.

ПОДОЛЯКА Олексій Миколайович – асистент кафедри інформатики Національного аерокосмічного університету ім. М.Є. Жуковського.

Наукові інтереси:
- дискретна математика і математичне моделювання;
- теорія програмування та алгоритмів.

ПОДОЛЯКА Оксана Олександрівна – кандидат технічних наук, старший викладач кафедри інформатики Харківського Національного автомобільно-дорожнього університету.

Наукові інтереси:
- математичне моделювання;
- теорія розкладів та її застосування.

Подано 02.06.2007