DOI 10.34185/1562-9945-2-157-2025-04
УДК 519.854.2

Є.О. Коваленко, В.А. Турчина

ПРО ПОКРАЩЕННЯ НАБЛИЖЕНИХ РОЗВ'ЯЗКІВ ЗАДАЧІ ПАРАЛЕЛЬНОГО УПОРЯДКУВАННЯ ТА АНАЛІЗ МОДЕЛІ ОДНОГО ЇЇ УЗАГАЛЬНЕННЯ

Анотація. Один із актуальних напрямків досліджень в теорії розкладів стосується тих задач, в яких при виконанні робіт дозволяються їх переривання. В даній роботі розглядається одна з постановок таких задач, до якої зводяться прикладні задачі пов'язані з оптимізацією певних виробничих процесів. Це стосується випадків, коли задану скінченну множину робіт, на порядок виконання яких накладаються технологічні обмеження, необхідно виконати використовуючи наявну кількість ресурсів в найкоротші терміни. В математичній постановці таку прикладну задачу можна звести до задачі паралельного упорядкування вершин орграфа. Досліджено можливість покращення наближених розв'язків цих задач за рахунок дозволу переривання робіт. Запропоновано модифікацію алгоритму розв'язання узагальненої задачі упорядкування для випадків, коли відповідний граф є регулярним і відноситься до підкласу повних дводольних.
Ключові слова: теорія розкладів, дискретна оптимізація, граф, регулярні графи, переривання, оптимальне розбиття, паралельні упорядкування, наближені розв'язки, узагальнення задачі.

Постановка проблеми. Деякі прикладні задачі організації процесу виготовлення продукції на виробництві, моделювання складних обчислювальних систем, ефективного управління ресурсами на проєктах та інші, пов'язані з плануванням на підприємствах, можуть зводитися до задач дискретної оптимізації, зокрема теорії розкладів. Для їх розв'язання розроблено методи та алгоритми, що дозволяють оптимізувати відповідні процеси залежно від конкретних потреб, як то скорочення загального часу завершення виконання робіт, мінімізація залучених ресурсів, виконання всього обсягу роботи в зазначені терміни, або зменшення відставання від них.

Одним з підкласів задач теорії розкладів є задачі паралельного упорядкування. За їх допомогою зручно моделювати такі прикладні задачі, де заздалегідь визначені скінченні множини робіт та ресурсів, за допомогою яких ці роботи мають бути виконані (станки, машини, процесори, люди, тощо). Вважається, що кожен виконавець може виконувати будь-яку роботу з однаковою ефективністю, причому щонайбільше одну в будь-який момент часу. На множині робіт задано частковий порядок, який відповідає технологічним обмеженням задачі. Залежно від постановки конкретної задачі необхід-

но мінімізувати або час завершення виконання усіх робіт, або максимум одночасно задіяних ресурсів без порушення технологічних обмежень.

Задачі упорядкування у класичній постановці передбачають, що множина робіт, час виконання кожної з яких рівний, задана за допомогою множини V вершин графа, тоді як множина дуг U відповідає технологічним обмеженням. Таким чином початкові умови задаються орієнтованим незваженим графом $G(V,U)$. Необхідно відшукати таке паралельне упорядкування S вершин заданого графа, яке або при заданій ширині $h(S)$ має мінімальну довжину $l(S)$, або при заданій довжині має мінімальну ширину [1].

В загальному випадку задачі упорядкування є *NP*-важкими, але були виділені їх підкласи, для розв'язання яких було розроблено точні алгоритми поліноміальної складності [2]. Один з таких алгоритмів заснований на рівневому принципі. Він дає точний розв'язок задачі упорядкування за умов, що граф задачі має структуру кореневого дерева або лісу, який складається з таких дерев. Інший алгоритм базується на лексикографічній розмітці вершин графа. Він є точним для графів без транзитивних дуг за умови, що $h = 2$. Застосування цих алгоритмів при розв'язанні задач упорядкування для довільних графів з рівними ваговими коефіцієнтами вершин в загальному випадку призводить до наближених розв'язків, при яких подекуди можливе збільшення значення цільової функції майже в два рази, порівняно з оптимальним.

Результати, отримані для класичних постановок задач упорядкування, хоч і моделюють деякі реальні процеси, більшу цінність представляють в теоретичному значенні. В прикладних задачах часто потрібно враховувати додаткові умови, що не зменшують складність задачі. Так наприклад, тривалість виконання робіт може бути різною, дозволені переривання при виконанні робіт, не всі виконавці можуть виконувати роботу в будь-який момент часу та ін. У [3] наведені деякі узагальнені формулювання таких задач, серед яких одним з випадків відповідає моделі, коли обмеження на кількість вершин, які можна розташовувати на фіксованих місцях, задається відповідним вектором.

Задачі побудови відповідних паралельних упорядкувань позначимо в загальному вигляді $S(A, B, C)$, де A — відповідає графу, який задає обмеження, B — заданому параметру задачі, C — цільовій функції.

В даній роботі основна увага приділяється задачам $S(G, h, l)$ та $S(G, h_i, l)$. В прикладних задачах дане узагальнення відображає ситуації, коли кількість доступних робочих ресурсів може змінюватися протягом того часу, який виділено на виконання робіт. Наприклад, якщо мова йде про людські ресурси, то їх зайнятість може змінюватися в результаті планування відпусток, скорочення або розширення штату. Кількість залучених станків або процесорів може залежати від масштабування виробничого процесу, або початку чи закінчення періоду обслуговування.

Аналіз останніх досліджень і публікацій. Прикладні задачі, що моделюються у вигляді задач комбінаторної оптимізації на графах привертають до себе увагу науковців. У роботі [4] розглядається оптимізаційна задача розміщення виробництва, в якій початкові умови задаються у вигляді графа. Розглянута авторами задача є *NP*-важкою, через що метою дослідження була розробка ефективного алгоритму поліноміальної

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

складності для пошуку наближених розв'язків. У вигляді орієнтованого графа спеціального виду (вебграфа) моделюються зв'язки між вебсторінками, що дозволяє формулювати та ефективно розв'язувати прикладні задачі оптимізації в цьому напрямку досліджень [5]. Цікавість дослідників також викликають задачі на графах певних структур. Так у [6] розглядається задача розмітки графів, що відносяться до підкласу регулярних. Такі графи знаходять практичне застосування у ряді прикладних задач. Випадок, коли задача управління проєктом моделюється у вигляді задачі теорії розкладів розглядається у [7]. Окрема увага приділяється ситуації, коли в процесі виконання робіт, передбачених проєктом, до нього залучаються додаткові ресурси та досліджується вплив дозволу переривань на загальну тривалість їх виконання.

Мета дослідження. З'ясувати, чи можна покращити якість наближених розв'язків для задач паралельного упорядкування, якщо дозволити переривання при виконанні робіт. Сформулювати алгоритм розв'язання узагальненої задачі для одного підкласу графів. Визначити умови, за яких цей алгоритм є точним.

Виклад основного матеріалу дослідження. Розглянемо можливість зменшення значення цільової функції за умови дозволу переривань при застосуванні алгоритму, що заснований на рівневому принципі.

При довільній будові графа відома оцінка точності розв'язку [2]:

$$\frac{l_A}{l^*} \le \frac{2h}{h+1}, \qquad (1)$$

де l_A — довжина упорядкування, побудованого за алгоритмом, l^* — довжина оптимального упорядкування.

Також відомі умови, за яких оцінка (1) досяжна, і відповідно непокращувана. Розглянемо граф G_1 наступного вигляду.

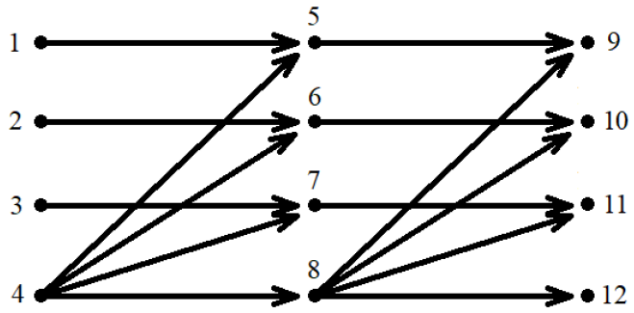

Рисунок 1 — Граф G_1

При заданій $h = 3$ застосування алгоритму дасть упорядкування:

$$\begin{pmatrix} 1 & 4 & 5 & 8 & 9 & 12 \\ 2 & & 6 & & 10 & \\ 3 & & 7 & & 11 & \end{pmatrix}.$$

В той час як оптимальним буде упорядкування:

$$\begin{pmatrix} 4 & 3 & 6 & 9 \\ 1 & 8 & 7 & 10 \\ 2 & 5 & 12 & 11 \end{pmatrix}.$$

Дійсно, для даного випадку виконується рівність

$$\frac{l_A}{l^*} = \frac{2h}{h+1} = \frac{6}{4}.$$

З'ясуємо, чи можна зменшити довжину упорядкування, побудованого за даним алгоритмом, якщо дозволити переривання. В першу чергу звертатимемо увагу на нещільно заповнені місця в упорядкуванні. Нехай дозволяється переривання роботи, що відповідає вершині 4. Тоді упорядкування матиме такий вигляд:

$$\begin{pmatrix} \{(4;0,5),(1;0,5)\} & (1;0,5) & (5,1) & (8,1) & (9,1) & (12,1) \\ (2,1) & (4;0,5) & (6,1) & & (10,1) & \\ (3,1) & & (7,1) & & (11,1) & \end{pmatrix}.$$

Довжина такого упорядкування $l_A = 5,5$. Якщо дозволити також переривання робіт, що відповідають вершинам 8 та 12, то отримане упорядкування

$$\begin{pmatrix} \{(4;0,5),(1;0,5)\} & (1;0,5) & \{(8;0,5),(5;0,5)\} & (5;0,5) & \{(12;0,5),(9;0,5)\} & (9;0,5) \\ (2,1) & (4;0,5) & (6,1) & (8;0,5) & (10,1) & (12;0,5) \\ (3,1) & & (7,1) & & (11,1) & \end{pmatrix}$$

матиме довжину $l_A = 4,5$.

Розглянемо також алгоритм заснований на застосуванні лексикографічної розмітки графа. При $h > 2$ в загальному випадку він дає наближений розв'язок. В [2] було оцінено точність алгоритму для випадків $h \geq 3$, яка описується співвідношенням:

$$\frac{l_A}{l^*} \leq 2 - \frac{2}{h}. \qquad (2)$$

В якості одного з прикладів автори наводять розв'язок задачі для графа G_2, зображеного на рис. 2 при $h = 3$.

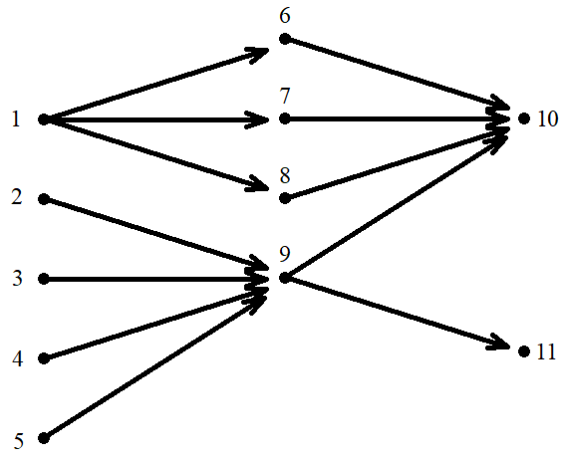

Рисунок 2 — Граф G_2

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Розв’язок, отриманий за допомогою алгоритму

$$\begin{pmatrix} 3 & 1 & 7 & 6 & 10 \\ 4 & 2 & 8 & 11 & \\ 5 & & 9 & & \end{pmatrix}$$

має довжину $l_A = 5$, тоді як довжина оптимального упорядкування

$$\begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & \end{pmatrix}$$

складає $l^* = 4$. Дійсно, повертаючись до відношення (2), маємо:

$$\frac{l_A}{l^*} \le 2 - \frac{2}{3} \Rightarrow \frac{5}{4} < \frac{4}{3}.$$

Нехай граф G_2' отримано з графа G_2 шляхом вилучення з нього вершин 10 і 11 та інцидентних до них дуг, після чого спрямування решти дуг було змінене на протилежне (рис. 3).

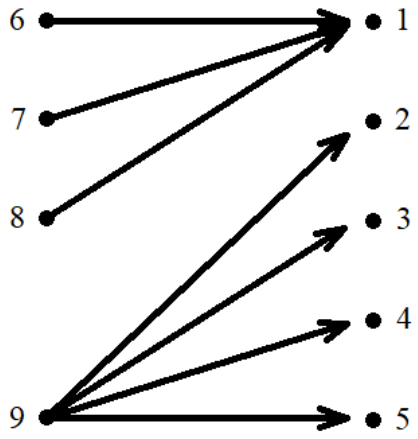

Рисунок 3 — Граф G_2'

Упорядкування побудоване за алгоритмом

$$\begin{pmatrix} 6 & 1 & 2 & 5 \\ 7 & 9 & 3 & \\ 8 & & 4 & \end{pmatrix}$$

має довжину $l_A' = 4$, а оптимальне упорядкування — довжину $l^* = 3$. Якщо дозволити переривання роботи 1, яка відповідає вершині розміщеній на частково заповненому місці, то це не вплине на оптимальність розв’язку. Натомість дозвіл на переривання робіт 5 та 9 призведе до того, що побудоване упорядкування матиме довжину $l_A' = 3$, рівну оптимальній:

$$\begin{pmatrix} \{(9;0,5),(6;0,5)\} & \{(6;0,5),(5;0,5)\} & \{(3;0,5),(5;0,5)\} \\ (7,1) & \{(9;0,5),(3;0,5)\} & (2,1) \\ (8,1) & (1,1) & (4,1) \end{pmatrix}.$$

Отже, в загальному випадку дозвіл на переривання довільної вершини, розміщеної на частково заповненому місці, не гарантує покращення наближеного розв'язку.

Розглянемо випадки, коли в задачі задано $G(V,U)$, де $|V| = n$ та замість ширини h множина значень $h_i \le n$, де $i = \overline{1,n}$. Для графів G, які містять множину ізольованих вершин (тобто $U = \varnothing$), було запропоновано точні алгоритми розв'язання відповідних узагальнених задач для випадків, коли переривання заборонені (алгоритм 1), та коли дозволені для усіх робіт (алгоритм 2) [3].

Розглянемо можливість модифікації цього алгоритму для випадків, коли граф задачі відноситься до підкласу регулярних, а саме має структуру повного дводольного, тобто $G = K_{m,r}$ де $m + r = n$. Як було зазначено в [8] в цьому випадку вершини першої долі графа можуть бути занесені до упорядкування таким же чином, як і у випадку, коли $U = \varnothing$. Зрозуміло, що після цього до упорядкування вершини другої долі можуть заноситися в довільному порядку.

При занесенні вершин першої долі можуть виникнути два випадки залежно від того, чи виконується умова:

$$\exists k : \sum_{i=1}^{k} h_i = m. \qquad (3)$$

Якщо (3) виконується, то усі m вершин першої долі можуть бути занесені до упорядкування таким чином, що місця з першого по k-те будуть щільно заповнені, і дозвіл переривань для цих вершин не покращить значення цільової функції. Для побудови цієї частини упорядкування можна скористатися алгоритмом 1, вважаючи, що замість n маємо m вершин. Тоді залишиться внести до упорядкування r вершин другої долі. Для них перевіряється виконання аналогічної умови, але відлік починається уже з місця (k+1). Умова матиме наступний вигляд:

$$\exists p : \sum_{i=k+1}^{k+p} h_i = r. \qquad (4)$$

Якщо (4) виконується, то усі вершини другої долі займатимуть в упорядкуванні щільно місця з (k+1)-го по p, а переривання будуть недоцільними і для робіт, що відповідають вершинам другої долі. В разі невиконання умови (4) для внесення до упорядкування вершин другої долі можна застосовувати алгоритм 2, вважаючи місце під номером (k+1) першим, а кількість вершин дорівнює r замість n.

У випадку, коли умова (3) не виконується, вершини першої долі графа можуть бути занесеними до упорядкування за алгоритмом 2, приймаючи n рівним m. Тоді місця упорядкування з першого по k-те будуть заповнені щільно, а місце (k+1) — майже щільно, при цьому воно буде заповнене на величину

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

$$c = \frac{m - \sum_{i=1}^{k} h_i}{h_{k+1}}, \qquad (5)$$

де $0 < c < 1$. При цьому очевидно при виконанні робіт, що відповідають вершинам цієї долі, переривання будуть необхідними (не обов'язково для всіх вершин). Для розподілу решти вершин перевірка умови (4) не потрібна. Для розміщення в упорядкуванні вершин другої долі пропонується використання алгоритму 2 з наступними модифікаціями.

Алгоритм.

Крок 0. Вважається, що вершини першої долі графа занесені в упорядкування S' за алгоритмом 2 і значення *c* обчислене за (5).

Крок 1. Усі місця в *S* вважаються порожніми, причому перше місце може бути заповнене лише на величину $(1-c)$, де $0 < c < 1$; $h_1 = h'_{k+1}$, де h'_{k+1} отримано з алгоритму 2.

Крок 2. $p = 1$.

Якщо виконується умова

$$\left(\min\left(h_1, r\right)\right)(1-c) + \sum_{i=2}^{p+1} h_i \geq r \qquad (6)$$

то $l_{П}^{*} = \max\left(1 - c + \frac{r - \left(\min\left(h_1, r\right)\right)(1-c)}{h_{p+1}}, 1\right)$ та перехід на крок 5, інакше перехід на крок 4.

Крок 3. Якщо виконується умова (6), то

$$l_{П}^{*} = p - c + \frac{r - \left(\left(\min\left(h_1, r\right)\right)(1-c) + \sum_{i=2}^{p} h_i\right)}{h_{p+1}}$$

і перехід на крок 5.

Крок 4. $p = p + 1$ та перехід на крок 3.

Крок 5. $j = 1$, $c_j = 0$.

Крок 6. Визначається підмножина $H_{(j)} \subset \left\{h_1, h_2, ..., h_{p+1}\right\}$ таким чином, що $h_i \geq j \Rightarrow h_i \in H_{(j)}$.

Крок 7. $\forall h_i \in H_{(j)}$ на місця *i* в упорядкування заносяться вершини в наступному порядку, за необхідності застосовуючи розбиття:

1) якщо $j > 1$ і виконується умова $H_{(j)} \neq \left\{h_{p+1}\right\} \cap H_{(j)} \neq \left\{h_1\right\}$, то та вершина, для якої $c_j > 0$;

2) $\left|H_{(j)}\right|$ вершин якщо $c_j + \left|H_{(j)}\right| < l_{П}^{*}$, інакше $\left|H_{(j)}\right| - \left\lfloor \left(l_{П}^{*} - c_j\right) \right\rfloor$ вершин;

3) якщо $c_j + \left|H_{(j)}\right| > l^*_{П}$, то обрана до занесення в упорядкування вершина q розбивається на часові інтервали t та (1 - t) так, що $c_j + \left(\left|H_{(j)}\right| - 1\right) - \left(1 - t\right) = l^*_{П}$, при цьому до ($q$,$t$) теж може бути застосоване розбиття.

Крок 8. Якщо $c_j > 0$ та виконується умова $H_{(j-1)} = \{h_i\} \cup H_{(j)} = \{h_i\}$, то замість внесення частини розбиття вершини (v,c_j) на позицію i обирається одна вершина v' серед внесених до упорядкування, для якої виконуються умови:

1) $v' \notin S[i]$;

2) $(v', t') \in S$, де $c_j \le t' \le 1$.

В упорядкуванні (v', t') замінюється на $\{(v, c_j), (v', t' - c_j)\}$ або $\{(v, c_j)\}$ якщо $t' = c_j$, а на позицію i ставиться (v', c_j).

Крок 9. Якщо $j < \max\limits_i h_i$ то $j = j + 1$, $c_j = t$ та перехід на крок 6, інакше формуємо упорядкування S^*, де

$$S^*[i] = \begin{cases} S'[i], \text{ якщо } i \le k \\ S'[i] \cup S[i], \text{ якщо } i = k + 1 \\ S'[i - k], \text{ якщо } i > k + 1 \end{cases}.$$

Крок 10. Кінець алгоритму.

Проілюструємо роботу наведеного алгоритму на наступному прикладі.

Приклад 1. Для повного дводольного графа $K_{8,6}$, та заданої послідовності h_i (4, 6, 5, 3, 3, 5, 2, 4, 3, 4, 5, 2, 3, 4) знайти упорядкування вершин, яке має мінімальну довжину.

Для визначеності вважатимемо, що вершини першої долі мають номери від 1 до 8, відповідно вершини другої — від 9 до 14. Упорядкування у випадку заборонених переривань матиме наступний вигляд:

$$\begin{pmatrix} 1 & 5 & 9 & 14 \\ 2 & 6 & 10 & \\ 3 & 7 & 11 & \\ 4 & 8 & 12 & \\ & & 13 & \end{pmatrix}.$$

Його довжина l^* = 4. Коли переривання дозволені, за алгоритмом 2 вносимо вершини першої долі до упорядкування:

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

$$\begin{pmatrix} \left\{\left(1,\frac{2}{3}\right),\left(7,\frac{1}{3}\right)\right\} & \left(2,\frac{2}{3}\right) & \\ \left\{\left(2,\frac{1}{3}\right),\left(3,\frac{2}{3}\right)\right\} & \left\{\left(3,\frac{1}{3}\right),\left(4,\frac{1}{3}\right)\right\} & \\ \left\{\left(4,\frac{2}{3}\right),\left(5,\frac{1}{3}\right)\right\} & \left(5,\frac{2}{3}\right) & \\ \left\{\left(6,\frac{1}{3}\right),\left(8,\frac{2}{3}\right)\right\} & \left(7,\frac{2}{3}\right) & \\ & \left\{\left(1,\frac{1}{3}\right),\left(8,\frac{1}{3}\right)\right\} & \\ & \left(6,\frac{2}{3}\right) & \end{pmatrix}.$$

Тепер можна внести до упорядкування вершини другої долі за модифікованим алгоритмом. Отримуємо наступне упорядкування:

$$\begin{pmatrix} \left\{\left(1,\frac{2}{3}\right),\left(7,\frac{1}{3}\right)\right\} & \left\{\left(2,\frac{2}{3}\right),\left(9,\frac{1}{3}\right)\right\} & \left\{\left(9,\frac{10}{15}\right),\left(10,\frac{2}{15}\right)\right\} \\ \left\{\left(2,\frac{1}{3}\right),\left(3,\frac{2}{3}\right)\right\} & \left\{\left(3,\frac{1}{3}\right),\left(4,\frac{1}{3}\right),\left(10,\frac{1}{3}\right)\right\} & \left\{\left(10,\frac{8}{15}\right),\left(11,\frac{4}{15}\right)\right\} \\ \left\{\left(4,\frac{2}{3}\right),\left(5,\frac{1}{3}\right)\right\} & \left\{\left(5,\frac{2}{3}\right),\left(11,\frac{1}{3}\right)\right\} & \left\{\left(11,\frac{6}{15}\right),\left(12,\frac{6}{15}\right)\right\} \\ \left\{\left(6,\frac{1}{3}\right),\left(8,\frac{2}{3}\right)\right\} & \left\{\left(7,\frac{2}{3}\right),\left(12,\frac{1}{3}\right)\right\} & \left\{\left(12,\frac{4}{15}\right),\left(13,\frac{8}{15}\right)\right\} \\ & \left\{\left(1,\frac{1}{3}\right),\left(8,\frac{1}{3}\right),\left(13,\frac{1}{3}\right)\right\} & \left\{\left(13,\frac{2}{15}\right),\left(14,\frac{10}{15}\right)\right\} \\ & \left\{\left(6,\frac{2}{3}\right),\left(14,\frac{1}{3}\right)\right\} & \end{pmatrix}$$

Його довжина $l_{\Pi}^{*}=2\frac{4}{5}$.

Твердження 1. Модифікований алгоритм для задачі $S(K_{m,r},h_i,l)$ у випадку, коли не виконується умова (3) та дозволені переривання, є точним.

Доведення. Очевидно, що при застосуванні немодифікованого алгоритму в побудованій частині упорядкування, де розміщуються вершини першої долі, лише останнє місце буде заповнено нещільно. Отже, якщо при застосуванні модифікованого алгоритму щільно або майже щільно будуть розміщені вершини другої долі, то побудоване упорядкування буде оптимальним.

Доведемо твердження окремо для випадків, коли умова (6):

$$\left(\min\left(h_1,r\right)\right)\left(1-c\right)+\sum_{i=2}^{p+1}h_i\geq r$$

виконується при $p = 1$ та при $p\geq 2$.

При $p = 1$ якщо $r < h_1$, то можна вважати, що $h_1 = r$, адже на це місце можна розмістити лише r розбиттів вершин. Тоді ця умова фактично перетворюється на $h_1(1-c)+h_{p+1} > r$. При оцінці довжини, яку займуть вершини в цьому випадку, розглядається максимум $l_{\Pi}^{*} = \max\left(1-c+\frac{r-h_1(1-c)}{h_{p+1}},1\right)$, тобто навіть якщо $1-c+\frac{r-h_1(1-c)}{h_{p+1}} < 1$, упорядкування можна дозаповнити не менше, ніж на величину однієї вершини, тобто 1. В іншому разі дозаповнюється (k+1)-е місце упорядкування (або перше місце за модифікованим алгоритмом) на величину $(1-c)$. Таким чином це місце буде заповнене щільно. На друге місце упорядкування розподіляються $r-h_1(1-c)$ вершин. Повертаючись до умови (6), легко бачити, що $h_{p+1} > r-h_1(1-c)$, отже ці вершини можуть бути розміщені на даному місці так, що його заповненість складатиме $\frac{r-h_1(1-c)}{h_{p+1}}$.

У випадку, коли умова (6) виконується при $p \geq 2$, так само до щільності дозаповнюється перше місце за модифікованим алгоритмом, тобто розподіляються розбиття вершин із сумарною величиною $h_1(1-c)$. Також щільно заповнюються місця з другого по p-те загалом на $\sum_{i=2}^{p} h_i$. Таким чином, розміщено $r-\left(h_1(1-c)+\sum_{i=2}^{p} h_i\right)$ вершин на місцях з першого по p-те. За умовою (6) маємо $h_{p+1} > r-\left(h_1(1-c)+\sum_{i=2}^{p} h_i\right)$, тобто решта вершин, які розбиваються, можуть бути розміщені на (p+1)-ому місці, дозаповненням його на частку $\frac{r-\left(h_1(1-c)+\sum_{i=2}^{p} h_i\right)}{h_{p+1}}$.

Таким чином, розподіл вершин другої долі за модифікованим алгоритмом є оптимальним.

Твердження 2. Умови $h_i \leq m$ при $1 \leq i \leq k+1$ та $h_i \leq r$ при $k+2 \leq i \leq k+p+1$ є достатніми для того, щоб модифікований алгоритм був точним для k та p взятих з нерівностей:

$$\begin{cases} \sum_{i=1}^{k} h_i < m < \sum_{i=1}^{k+1} h_i \\ \sum_{i=k+1}^{k+p} h_i < r < \sum_{i=k+1}^{k+p+1} h_i \end{cases}.$$

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Доведення. Розглянемо окремо умови твердження. Достатність першої з них ($h_i \le m$ при $1 \le i \le k+1$) доводиться за аналогією до того, як це було зроблено для твердження 2, наведеного у [3].

Доведемо достатність другої умови. За модифікованим алгоритмом (k+1)-е місце упорядкування дозаповнюється розбиттями вершин другої долі, причому якщо $m \le r$, то до щільного заповнення. В іншому разі щільне заповнення буде неможливим через те, що згідно з алгоритмом вершини, які залишилися, не можна буде розбити відповідним чином. Якщо $p > 1$, то місця з (k+2)-го по (k+p)-е будуть щільно заповнені. Місце (k+p+1) буде заповнене на сталу частку тільки якщо $h_{k+p+1} \le r$. Інакше дозаповнити це місце на відповідну величину неможливо.

Висновки. Проаналізовано ефективність двох відомих алгоритмів поліноміальної складності, які є точними для двох спеціальних класів графів, та наближеними для довільних графів. Досліджено можливість покращення наближених розв'язків за рахунок дозволу переривань при розміщенні вершин.

Для задачі упорядкування в узагальненій постановці розглянуто алгоритм її розв'язання, який враховує дозвіл на переривання робіт, і є точним для випадків, коли відповідний граф складається лише з ізольованих вершин. Запропоновано модифікацію цього алгоритму, яка дозволяє отримувати точні розв'язки, коли граф задачі має структуру повного дводольного.

ЛІТЕРАТУРА

1. Burdyuk V.Ya., Turchyna V.A. Algoritmi parallelnogo uporyadocheniya: uchebnoe posobie. Dnepropetrovsk: DGU, 1985. 83 p. (in Russian)

2. Coffman E. G., Bruno J. L. Computer and job-shop scheduling theory. New York : Wiley, 1976. 299 p.

3. Коваленко Є.О., Турчина В.А. Про один частковий випадок задачі паралельного упорядкування. *Theoretical and empirical scientific research: concept and trends*. 2024. С. 222-226. Режим доступу: https://doi.org/10.36074/logos-02.02.2024.044.

4. Kozin I. V., Narzullaev U. H., Allomov Z. K. The shuffle frog leaping algorithm for the production location problem. *Computer science and applied mathematics*. 2023. No. 1. P. 11–18. Mode of access: https://doi.org/10.26661/2786-6254-2023-1-02.

5. Долотов І. О., Гук Н. А. Кластеризація зваженого вебграфу із використанням модулярності. *Питання прикладної математики і математичного моделювання*. 2023. Вип. 23. С. 45-52. Режим доступу: https://doi.org/10.15421/322305.

6. Semeniuta M. F. Combinatorial configurations in the definition of antimagic labelings of graphs. *Cybernetics and systems analysis*. 2021. Vol. 57, no. 2. P. 196–204. Mode of access: https://doi.org/10.1007/s10559-021-00344-y.

7. Nakonechna T. V. On the influence of interruptions in the jobs execution in project management. *Problems of applied mathematics and mathematical modeling*. 2024. Vol. 24. P. 151-158. Mode of access: https://doi.org/10.15421/322416.

8. Турчина В.А., Коваленко Є. О. Вплив початкових даних задачі паралельного упорядкування з перериваннями на оптимальність розв'язку. *Питання прикладної матема-*

тики і математичного моделювання. 2022. Вип. 22. С. 158-167. Режим доступу: https://doi.org/10.15421/322217.

REFERENCES

1. Burdyuk V.Ya., Turchyna V.A. (1985). Parallel sequencing algorithms: textbook. Dnipropetrovsk: DSU,. 83 p.
2. Coffman, E. G., Bruno, J. L. (1976). Computer and job-shop scheduling theory. Wiley.
3. Kovalenko Y. O., Turchyna V. A. (2024) On a partial case of the parallel sequencing problem. *Theoretical and empirical scientific research: concept and trends*, 222-226. Mode of access: https://doi.org/10.36074/logos-02.02.2024.044.
4. Kozin, I. V., Narzullaev, U. H., Allomov, Z. K. (2023). The shuffle frog leaping algorithm for the production location problem. *Computer Science and Applied Mathematics*, 1, 11–18. https://doi.org/10.26661/2786-6254-2023-1-02.
5. Dolotov I. O. Guk N. A. (2023). Clustering of a weighted webgraf with the usage of modularity. *Problems of applied mathematics and mathematic modeling* 23, 45–52. Mode of access: https://doi.org/10.15421/322305.
6. Semeniuta M. F. (2021). Combinatorial Configurations in the Definition of Antimagic Labelings of Graphs. *Cybernetics and systems analysis*, 57(2), 196–204. Mode of access: https://doi.org/10.1007/s10559-021-00344-y.
7. Nakonechna T. V. (2024). On the influence of interruptions in the jobs execution in project management. *Problems of applied mathematics and mathematical modeling*, 24, 151-158. Mode of access: https://doi.org/10.15421/322416.
Turchyna V. A., Kovalenko Y. O. (2022). The influence of the parallel sequencing problem with interruptions initial data on the solution optimality. *Problems of applied mathematics and mathematical modeling*, 22, 158-167. Mode of access: https://doi.org/10.15421/322217.

Received 05.03.2025.
Accepted 07.03.2025.

On improving approximate solutions to parallel sequencing problem and one of its generalizations model analysis

One of the current research directions in scheduling theory concerns those problems in which interruptions are allowed during the job execution. Scheduling theory problems are widely used in modeling work, particularly production, processes related to planning. To solve them, methods and algorithms have been developed that allow optimizing the relevant processes depending on specific needs, such as reducing the total time to complete jobs, minimizing the resources involved, completing the entire amount of jobs within the specified time frame, or reducing the time delay. One of such problems implies a given finite set of jobs, the order of execution of which is subject to technological restrictions. This set of jobs must be performed using the available amount of resources in the shortest time period. In mathematical formulation, such an applied problem can be reduced to the vertices' of a digraph parallel sequencing problem.

The sequencing problems are mostly NP-hard, but their subclasses have been identified, for the solution of which exact algorithms of polynomial complexity have been developed. The use of such known algorithms of polynomial complexity when solving sequencing problems

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

for arbitrary graphs with equal vertex weights generally leads to approximate solutions. Occasionally it is possible for the objective function value to increase almost twice, compared to the optimal one. The possibility of improving approximate solutions to these problems by allowing job interruptions is investigated.

One generalization of the parallel sequencing problem corresponds to a model where the limit on the number of vertices that can be placed at fixed locations is given by the corresponding vector. A modification of the algorithm for solving this generalized problem is proposed for cases when the corresponding graph is regular and belongs to the subclass of complete bipartite graphs.

Коваленко Євген Олександрович – аспірант кафедри обчислювальної математики та математичної кібернетики Дніпровського національного університету імені Олеся Гончара.

Турчина Валентина Андріївна – к.ф.-м.н., доцент, завідувачка кафедри обчислювальної математики та математичної кібернетики Дніпровського національного університету імені Олеся Гончара.

Kovalenko Yevhen – Postgraduate student of the Department of Computational Mathematics and Mathematical Cybernetics of Oles Honchar Dnipro National University.

Turchyna Valentyna – Candidate of Physical and Mathematical Sciences, Associate Professor, Head of the Department of Computational Mathematics and Mathematical Cybernetics of Oles Honchar Dnipro National University.