«Системні технології» 2 (151) 2024 «System technologies»
DOI 10.34185/1562-9945-2-151-2024-07
УДК 519.8

К.Д. Караваєв

ПРО НЕОБХІДНІ УМОВИ ІСНУВАННЯ ЩІЛЬНИХ УПОРЯДКУВАНЬ В КЛАСИЧНІЙ ЗАДАЧІ ПАРАЛЕЛЬНОГО УПОРЯДКУВАННЯ

Анотація. Побудова схем паралелізації обчислень на ЕОМ при обробці великих масивів даних нерозривно пов'язана з задачами паралельного упорядкування. Розглядається класична задача мінімізації довжини упорядкування при заданій ширині, в якій шукане упорядкування є щільним. Досліджуються необхідні умови існування щільного упорядкування при застосуванні методу гілок та меж, пов'язані з обмеженістю потужності місць та можливістю їх заповнення. Отримані умови були зведені до однієї, запропоновано ефективні алгоритми її перевірки в загальному випадку та для графів, всі вершини яких знаходяться на критичних шляхах. В результаті дослідження також були отримані нові уточнені оцінки знизу довжини упорядкування та запропоновані узагальнення спеціальних упорядкувань, які враховують ширину упорядкування.
Ключові слова: графи, комбінаторна оптимізація, теорія розкладів, метод гілок та меж, розмітка графів, щільні упорядкування, оптимальний розподіл вершин.

Постановка проблеми. Бурхливий розвиток теорії розкладів у середині минулого століття був пов'язаний з різноманітністю важливих практичних застосувань задач, які в ній розглядаються. До них належать планування у багатьох сферах виробництва, таких як будівництво, хімічна та автомобільна промисловість, сільське господарство тощо; питання побудови розкладів транспорту і оптимізації логістики, побудова схем розпаралелювання обчислень на ЕОМ та багато іншого.

Окрема увага дослідників була приділена задачам, де на порядок виконання робіт накладені деякі технологічні обмеження. Однією з розповсюджених математичних моделей цих задач є задачі паралельного упорядкування. Вони формулюються як комбінаторні оптимізаційні задачі на орієнтованих ациклічних графах, вершини яких відповідають роботам, а дуги задають частковий порядок їх виконання.

Класичною вважається наступна постановка [1]. Нехай задано ациклічний орграф $G(V,U)$ та параметр-ширина h, який відповідає наявній кількості ви-

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

конавців. Необхідно побудувати таке упорядкування S його вершин, тобто розподілити їх між місцями, розташованими в лінію, в якому:

- дотримано виконання технологічних обмежень, тобто, якщо між деякими вершинами v_i та v_j існує шлях у G, то v_i знаходиться в S лівіше за v_j;
- на кожному місці знаходиться не більше h вершин;
- кількість непорожніх місць у ньому l (його довжина) є мінімальною.

Довжина упорядкування є формалізацією загального часу виконання усіх завдань. Вважається, що виконавці є універсальними, а час виконання всіх завдань є однаковим та дорівнює 1. Таку задачу прийнято позначати $S(G, h, l)$. Отримане упорядкування називають щільним, якщо на кожному місці розташовано рівно h вершин (таке упорядкування завжди є оптимальним).

Окрім задачі $S(G, h, l)$ також розглядається задача $S(G, l, h)$, в якій за заданою довжиною упорядкування потрібно побудувати упорядкування з мінімальним значенням ширини.

В загальному випадку ці задачі виявилися NP-важкими, тому для знаходження точних розв'язків застосовуються методи направленого перебору, зокрема метод гілок та меж. Точні поліноміальні алгоритми відомі лише для деяких спеціальних випадків.

Аналіз останніх досліджень і публікацій. Оскільки поліноміальна розв'язність задачі $S(G, h, l)$ при фіксованих $h > 2$ є невідомою, то основними напрямками досліджень цієї задачі є пошук класів графів, для яких існують точні поліноміальні алгоритми, та побудова наближених алгоритмів.

Значне просування протягом останніх років було здійснено в напрямку побудови $(1+\varepsilon)$-наближених алгоритмів, що мають квазіполіноміальну складність [2-5]. Найкращою відомою до того відносною похибкою наближених алгоритмів для $h = 3$ було значення $4/3$ [6].

Також ведеться активна розробка алгоритмів, які засновані на метаевристиках, таких як Баєсова оптимізація та генетичні алгоритми [7,8]. Окрім цього проводяться дослідження в напрямку узагальнення точних відомих поліноміальних алгоритмів для розширення класу задач, для яких вони можуть знаходити точні розв'язки [9,10].

Окрім класичної задачі науковцями також розглядаються її узагальнення, які мають ускладнені структури завдань та виконавців, додаткові умови на виконання робіт, інші цільові функції, тощо [11]. В зв'язку з розвитком туманних

обчислень протягом останніх років, багато робіт присвячені дослідженню таких задач в рамках саме цієї сфери застосування [12].

В рамках дослідження методу гілок та меж ведеться робота щодо побудови схем перебору для узагальнених задач і скорочення перебору шляхом уточнення оцінок знизу довжини та виключення гілок, що відповідають ізоморфним підграфам [13,14].

Мета дослідження. Дослідити обмеження, що накладає на проміжні графи у методі гілок та меж умова щільності шуканого упорядкування. Отримати необхідні умови існування щільного упорядкування та запропонувати ефективні методи їх перевірки.

Виклад основного матеріалу дослідження. У роботі [9] було показано, що будь-яка класична задача паралельного упорядкування може буде зведена до задачі, в якій шукане упорядкування є щільним. Очевидною перевагою такої задачі є те, що довжина оптимального упорядкування відома і потрібно лише його побудувати.

Щільність упорядкування накладає ряд додаткових обмежень на проміжні упорядкування, що будуються при використанні методу гілок та меж [13], тому їх врахування може дозволити значно скоротити кількість гілок, що підлягають розгляду.

Відзначимо спершу, що через те, що довжина упорядкування відома, можемо визначити місця, які в упорядкуванні може займати кожна з вершин. Розглянемо узагальнену схему визначення крайнього лівого та правого місця, які в упорядкуванні може займати вершина.

Для визначення цих місць в задачі $S(G,l,h)$ використовуються спеціальні упорядкування $\underline{S}$ та $\overline{S}$ [1]. Для кожної вершини v_i визначаються місця $\underline{\mu_i}$ та $\overline{\mu_i}$, які вона займає в цих упорядкуваннях відповідно. Тоді проміжок допустимих місць, що може займати ця вершина, визначається наступним чином: $\left[\underline{\mu_i}; \overline{\mu_i} + l - \underline{l}\right]$, де $\underline{l}$ – довжина критичного шляху у графі G.

Відмітимо, що цей алгоритм неможливо застосувати до задачі $S(G,h,l)$ в загальному випадку, оскільки довжина упорядкування є невідомою величиною. Помітимо також, що в наведених міркуваннях не враховується інформація про допустиму ширину упорядкування h, бо для $S(G,l,h)$ вона є шуканою величиною. Проте для щільного упорядкування в задачі $S(G,h,l)$ обидві величини є

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

відомими, тому зможемо більш точно визначити межі можливого розташування вершин.

Значення $\underline{\mu_i}$ та $\underline{l} - \overline{\mu_i} + 1$ можна розглядати як елементарні оцінки знизу довжини упорядкування для підграфів $\underline{G_i}$ та $\overline{G_i}$, можливо порожніх, вершини яких мають шляхи до вершини v_i та з цієї вершини відповідно. Дійсно, $\underline{\mu_i}$ та $\underline{l} - \overline{\mu_i} + 1$ на одиницю більші за довжини критичних шляхів у цих графах (необхідність додавати одиницю пов'язана з відсутністю вершини v_i у графах). Враховуючи, що всі вершини графів $\underline{G_i}$ та $\overline{G_i}$ мають бути розташованими в упорядкуванні ліворуч та праворуч від v_i відповідно, можемо замість довжини критичного шляху використати будь-яку іншу відому оцінку знизу для довжини упорядкування, оскільки це є теоретично мінімально можливою кількістю місць, на яких можуть бути розташовані всі вершини графу. Таким чином отримаємо деякі уточнені значення $\underline{\mu_i^h}$ та $\overline{\mu_i^h}$, які можемо використовувати аналогічним чином.

Відмітимо також, що такі ж міркування можна застосовувати й при обчислені оцінок, які використовують спеціальні упорядкування $\underline{S}$ та $\overline{S}$. Таким чином зможемо побудувати упорядкування $\underline{S^h}$ та $\overline{S^h}$, в яких кожна вершина v_i графу займає крайнє ліве та крайнє праве місце у відповідності з деякою оцінкою довжини для графів $\underline{G_i}$ та $\overline{G_i}$. Ті оцінки, що використовують упорядкування $\underline{S^h}$ та $\overline{S^h}$ замість $\underline{S}$ та $\overline{S}$ будуть уточненими оцінками довжини знизу.

Зауважимо, що упорядкування $\underline{S^h}$ та $\overline{S^h}$ мають дві ключові відмінності від упорядкувань $\underline{S}$ та $\overline{S}$: по-перше, вони можуть мати порожні місця, по-друге, їх довжини можуть бути різними. Так, наприклад, для графу $K_{1,4}$, $h = 3$ та базової оцінки довжини упорядкування $\overline{S^h}$ має порожнє друге місце, його довжина дорівнює 3, в той же час довжина упорядкування $\underline{S^h}$ дорівнює 2. З цієї причини при визначені крайнього правого допустимого місця в упорядкуванні замість довжини критичного шляху у графі потрібно використовувати довжину упорядкування $\overline{S^h}$.

Усі наведені міркування підсумовано у наступному алгоритмі.

Алгоритм визначення поміток $\underline{\mu_i^h}$

1. Вершин без вхідних дуг отримують помітки $\underline{\mu_i^h} = 1$.

2. Розглянемо множину вершин $\underline{V}$, які ще не мають поміток, а усі їх попередники вже помічені.

3. Для кожної вершини $v_i \in \underline{V}$ побудуємо спеціальне упорядкування $\underline{S_i^h}$, в якому попередники v_j вершини v_i будуть розташовані відповідно на місцях $\underline{\mu_j^h}$.

4. Вершина v_i отримує помітку $\underline{\mu_i^h} = 1 + f\left(\underline{S_i^h}\right)$, де $f(\cdot)$ – деяка відома оцінка знизу довжини упорядкування.

5. Якщо у графі залишилися вершини без поміток повертаємося на 2, інакше кінець.

Аналогічні помітки $\overline{\mu_i^h}$ можна отримати, якщо рухатись від вершин, що не мають вихідних дуг.

Перейдемо тепер до розгляду обмежень, що накладає вимога щільності на проміжні упорядкування. Тож нехай на деякому етапі побудови дерева розгалужень вже заповненими в упорядкуванні є перші k місць та нерозташованими залишаються вершини підграфу $\tilde{G}$. Будемо розглядати обмеження в порядку складності їх перевірки.

Очевидно, якщо у графі $\tilde{G}$ менше ніж h відкритих вершин, то побудувати щільне упорядкування вже не вдасться, тому поточну гілку можна відсікти.

Помітимо, що в процесі розташування вершин підграфи $\overline{G_i}$ залишаються незмінними, а отже значення поміток $\overline{\mu_i^h}$ можна порахувати один раз для початкового графу G, враховуючи у подальшому, що їх значення потрібно розглядати відносно вихідного графу, а не графу $\tilde{G}\left(\tilde{V}, \tilde{U}\right)$ (вони будуть відрізнятися на значення різниці між довжинами упорядкувань $\overline{S^h}$ для цих двох графів). Для зручності подальшого викладення будемо використовувати зведені помітки крайнього правого місця, яке вершина може займати в упорядкуванні: $\overline{\lambda_i^h} = \overline{\mu_i^h} + l - \overline{l^h}$, де $\overline{l^h}$ – довжина упорядкування $\overline{S^h}$ для початкового графу G.

Розглянемо тепер множини вершин $\overline{V_j^h} = \left\{v_i \in \tilde{V} : \overline{\lambda_i^h} \le j\right\}$, $j = k+1, \ldots, l$, тобто тих вершин, які мають бути розташовані у шуканому упорядкуванні до місця j включно. З іншого боку, кількість вершин, що можна розташувати на місця з $k+1$ по j дорівнює $h * (j - k)$. Зрозуміло, що у випадку, коли потуж-

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

ність відповідної множини перевищує цю кількість, тобто $\left|\overline{V_j^h}\right| > h * (j - k)$, побудувати щільне упорядкування також вже не вдасться і подальші розгалуження можна не проводити. Відзначимо, що додатково можна було б перевіряти, що всі множини $\overline{V_j^h}$, $j = 1,\dots,k$ є порожніми, проте достатньо перевірити лише множину $\overline{V_k^h}$, за побудовою. Ця перевірка вписується у загальну схему, оскільки маємо $h * (k - k) = 0$. Маємо для цього випадку $\overline{\lambda_i^h} \le k \Rightarrow l - k \le \overline{l^h} - \overline{\mu_i^h} < \overline{l^h} - \overline{\mu_i^h} + 1$, тобто довжина критичного шляху у $\tilde{G}$ буде перевищувати кількість місць доступних для розташування.

Зауважимо, що тут і далі, довжина критичного шляху розуміється у дещо узагальненому сенсі як мінімальна кількість місць, необхідна для розташування всіх вершин графу відповідно до деякого критерію.

Для перевірки наступних обмежень потрібно обчислити помітки $\underline{\mu_i^h}$ для вершин графу $\tilde{G}$, які на відміну від $\overline{\mu_i^h}$ постійно змінюються внаслідок розташування вершин. Також будемо використовувати зведені помітки крайнього лівого допустимого місця: $\underline{\lambda_i^h} = \underline{\mu_i^h} + k$.

Порівняємо тепер для кожної вершини помітки $\underline{\lambda_i^h}$ та $\overline{\lambda_i^h}$. Якщо знайдеться така вершина v_i, для якої $\underline{\lambda_i^h} > \overline{\lambda_i^h}$, то щільне упорядкування побудувати також не вдасться, оскільки для розміщення всіх нащадків та попередників цієї вершини знадобиться більше ніж $l - k$ місць: $\underline{\lambda_i^h} > \overline{\lambda_i^h} \Rightarrow \underline{\mu_i^h} + \left(\overline{l^h} - \overline{\mu_i^h}\right) > l - k$. Перейдемо тепер до вершин, для яких ці помітки співпадають: $V_j^h = \left\{v_i \in \tilde{V} : \underline{\lambda_i^h} = \overline{\lambda_i^h} = j\right\}$, $j = k + 1,\dots,l$. Всі вершини множин V_j^h можуть бути розташовані лише на місці j, тоді, якщо хоча б одна з них містить більше ніж h елементів, то також не отримаємо щільне упорядкування. Отже, отримали необхідну умову: $\left|V_j^h\right| \le h$.

Залишилося розглянути вершини, для яких $\underline{\lambda_i^h} < \overline{\lambda_i^h}$. Побудуємо множини $V_{j_1,j_2}^h = \left\{v_i \in \tilde{V} : j_1 \le \underline{\lambda_i^h}, \overline{\lambda_i^h} \le j_2\right\}$, $k + 1 \le j_1 \le j_2 \le l$, всі вершини цих множин можуть бути розміщенні лише на місцях з відрізку $[j_1; j_2]$. З іншого боку цей діапазон вміщує $h * (j_2 - j_1 + 1)$ вершин, а отже у випадку, коли потужність якоїсь з

цих множин перевищує наведене значення, також не вдасться побудувати щільне упорядкування, тобто має виконуватись: $\left|V^{h}_{j_1, j_2}\right| \le h * (j_2 - j_1 + 1)$.

Підсумовуючи розглянуті випадки, бачимо, що неможливо отримати щільне упорядкування через те, що для розміщення вершин графу $\tilde{G}$ необхідно більше ніж $l - k$ місць. А отже, якби вдалося побудувати оцінку знизу довжини, яка б враховувала всі зазначені характеристики у структурі графу, то не потрібно було б перевіряти кожен пункт окремо.

Відмітимо, що з виконання останньої необхідної умови існування щільного упорядкування випливає виконання майже всіх попередніх більш простіших умов. По-перше, легко побачити, що $V^{h}_{j} = V^{h}_{j, j}$, $j = k + 1, \dots, l$ та $h * (j - j + 1) = h$, а отже умова для вершин з рівними зведеними помітками виконується. По-друге, якщо $\left|\overline{V^{h}_{k}}\right| = 0$, тоді $\overline{V^{h}_{j}} = V^{h}_{k+1, j}$, $j = k + 1, \dots, l$ і $h * (j - (k + 1) + 1) = h * (j - k)$, тобто виконується умова для вершин, права зведена помітка яких не перевищує j. По-третє, розглянемо множину $V^{h}_{k+2, l}$: вона не містить відкритих вершин, оскільки всі відкриті вершини мають помітки $\underline{\lambda^{h}_{i}} = k + 1$. Тоді, якщо їх кількість менша за h, то матимемо $\left|V^{h}_{k+2, l}\right| > h * (l - k) - h = h * (l - (k + 2) + 1)$, а отже відповідна умова буде порушена. Останні дві умови, пов'язані з $\overline{V^{h}_{k}}$ та вершинами, для яких $\underline{\lambda^{h}_{i}} > \overline{\lambda^{h}_{i}}$, можуть бути замінені перевіркою довжини критичного шляху у $\tilde{G}$, як показано вище.

Відзначимо також, що майже в усіх попередніх необхідних умовах перевірялося обмеження потужності місць, а не можливість їх заповнення. Так, наприклад, можна розглядати множини вершин, які можуть бути розташовані на заданому проміжку: $\dot{V}^{h}_{j_1, j_2} = \left\{v_i \in \tilde{V} : [j_1; j_2] \cap \left[\underline{\lambda^{h}_{i}}; \overline{\lambda^{h}_{i}}\right] \neq \varnothing\right\}$, $k + 1 \le j_1 \le j_2 \le l$. Очевидно, що для щільного заповнення проміжку місць необхідно, щоб потужності всіх цих множин були не меншими за $h * (j_2 - j_1 + 1)$. Розглянемо тепер множини $V^{h}_{k+1, j_1 - 1}$ та $V^{h}_{j_2 + 1, l}$. За побудовою, ці множини, якщо вони непорожні, містять вершини, які можна розташувати виключно на проміжках $[k + 1; j_1 - 1]$ та $[j_2 + 1; l]$ відповідно, а отже вони не містять спільних елементів. Більш того, перетин $\dot{V}^{h}_{j_1, j_2}$ з цими множинами також є порожнім, оскільки проміжки не перетинаються, і також можна побачити, що кожна вершина належить принаймні

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

одній з цих множин. Тоді у випадку недостатньої кількості вершин для розміщення $\left|V^{h}_{j_1, j_2}\right| < h * (j_2 - j_1 + 1)$ отримаємо, що

$$\left|V^{h}_{k+1, j_1\text{-}1} \cup V^{h}_{j_2+1, l}\right| = \left|V^{h}_{k+1, j_1\text{-}1}\right| + \left|V^{h}_{j_2+1, l}\right| > h * (l - k) - h * (j_2 - j_1 + 1) = h * (l - k - j_2 + j_1 - 1) =$$
$$= h * (l - (j_2 + 1) + 1) + h * ((j_1 - 1) - (k + 1) + 1),$$

а тоді, за принципом Діріхле, принаймні одна з необхідних умов для множин $V^{h}_{k+1, j_1 - 1}$ та $V^{h}_{j_2+1, l}$ буде порушена. А отже, цю необхідну умову також можна не перевіряти окремо.

Насправді, обидві зазначені необхідні умови можна об'єднати у одну. Для цього проведемо спочатку перетворення над умовою для діапазонів місць, аби звести її до більш зручного вигляду. Розділимо обидві частини формули на h та скористаємося тим, що для $\forall a, c \in \mathrm{Z}, b \in \mathrm{N}$ виконується $a \le b * c \Leftrightarrow \frac{a}{b} \le c \Leftrightarrow \left\lceil \frac{a}{b} \right\rceil \le \lceil c \rceil = c$, оскільки $\lceil \cdot \rceil$ є неспадною функцією, тоді отримаємо $\left\lceil \frac{\left|V^{h}_{j_1, j_2}\right|}{h} \right\rceil \le j_2 - j_1 + 1$ для $k + 1 \le j_1 \le j_2 \le l$. Перенесемо тепер ліворуч праву частину нерівності та додамо l до обох частин, тоді для тих саме значень j_1, j_2 матимемо $(j_1 - 1) + \left\lceil \frac{\left|V^{h}_{j_1, j_2}\right|}{h} \right\rceil + (l - j_2) \le l$. Оскільки остання нерівність має виконуватись для всіх значень $k + 1 \le j_1 \le j_2 \le l$, то вона є еквівалентною наступній:

$$\max_{k+1 \le j_1 \le j_2 \le l} \left((j_1 - 1) + \left\lceil \frac{\left|V^{h}_{j_1, j_2}\right|}{h} \right\rceil + (l - j_2) \right) \le l.$$

Розглянемо детальніше доданки, що утворюють цей вираз. Другий доданок відповідає мінімальній кількості місць, необхідних для розташування вершин, що можуть розташовуватись виключно на місцях з діапазону $[j_1; j_2]$. Перший та третій доданок – мінімальній довжині критичних шляхів зліва та справа для цих вершин відповідно.

Розглянемо тепер випадок, коли довжина критичного шляху перевищує $l - k$. У цьому випадку знайдуться такі значення $j_1 > j_2$, для яких множина $V^{h}_{j_1, j_2}$ не буде порожньою. Тоді для цих вершин маємо $\left\lceil \frac{\left|V^{h}_{j_1, j_2}\right|}{h} \right\rceil \ge 1$, і тоді

$(j_1 - 1) + \left\lceil \frac{\left| V^{h}_{j_1, j_2} \right|}{h} \right\rceil + (l - j_2) \geq l + (j_1 - j_2) > l$, що вписується у загальну перевірку. А отже зможемо об'єднати цю умову з попередньою, замінивши допустиму область на множину пар (j_1, j_2), для яких множина $V^{h}_{j_1, j_2}$ не є порожньою. Це не зменшує загальності, оскільки у випадку $j_1 \leq j_2$ та $V^{h}_{j_1, j_2} = \varnothing$ матимемо $(j_1 - 1) + \left\lceil \frac{\left| V^{h}_{j_1, j_2} \right|}{h} \right\rceil + (l - j_2) = l - 1 - (j_2 - j_1) < l$, тобто умова буде виконана.

Відмітимо окремо, що для двох пар значень (j_1, j_2) та (j_1', j_2') таких, що $j_1' \leq j_1, j_2 \leq j_2', V^{h}_{j_1, j_2} = V^{h}_{j_1', j_2'}$, матимемо наступне співвідношення:

$$(j_1' - 1) + \left\lceil \frac{\left| V^{h}_{j_1', j_2'} \right|}{h} \right\rceil + (l - j_2') = (j_1 - 1) + \left\lceil \frac{\left| V^{h}_{j_1, j_2} \right|}{h} \right\rceil + (l - j_2) - (j_1 - j_1') - (j_2' - j_2) \leq$$

$$\leq (j_1 - 1) + \left\lceil \frac{\left| V^{h}_{j_1, j_2} \right|}{h} \right\rceil + (l - j_2),$$ а отже будь-яке розширення діапазонів не призводить до порушення умови і розглядати має сенс лише найкоротші з них.

З іншого боку, у випадку $j_1 = k + 1, j_2 = l$ множина $V^{h}_{j_1, j_2}$ міститиме всі вершини, за побудовою позначок $\underline{\lambda}^{h}_{i}$ та $\overline{\lambda^{h}_{i}}$, тобто $\left| V^{h}_{j_1, j_2} \right| = h * (l - k)$, оскільки шукане упорядкування є щільним. Тоді отримаємо $(j_1 - 1) + \left\lceil \frac{\left| V^{h}_{j_1, j_2} \right|}{h} \right\rceil + (l - j_2) = (k + 1 - 1) + \left\lceil \frac{h * (l - k)}{h} \right\rceil + (l - l) = l$, а отже шуканий максимум обмежений зверху та знизу значенням довжини упорядкування, і необхідна умова існування щільного упорядкування приймає остаточний вигляд:

$$l = \max_{(a,b): V^{h}_{a,b} \neq \varnothing} \left((a - 1) + \left\lceil \frac{\left| V^{h}_{a,b} \right|}{h} \right\rceil + (l - b) \right).$$

Розглянемо тепер як зсуви значень $\underline{\lambda}^{h}_{i}, \overline{\lambda^{h}_{i}}$ на константи $\underline{\Delta}$ та $\overline{\Delta}$ відповідно впливають на отриманий вираз. Зафіксуємо деяку довільну пару значень $(a, b): V^{h}_{a,b} \neq \varnothing$. Розглянемо тепер пару $\left(a + \underline{\Delta}, b + \overline{\Delta}\right)$, зрозуміло, що в рамках зсунутих позначень множина вершин $V^{h}_{a + \underline{\Delta}, b + \overline{\Delta}}$ співпаде з множиною $V^{h}_{a,b}$ у вихідних позначеннях, а отже другий доданок виразу не зміниться. Для третього

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

доданку маємо $\left(l+\overline{\Delta}\right)-\left(b+\overline{\Delta}\right)=l-b$, а отже зсув також не впливає і на його значення. Для першого доданку – $(a+\underline{\Delta}-1)=(a-1)+\underline{\Delta}$, тобто підсумкове значення правої частини зміниться на $\underline{\Delta}$.

Відзначені спостереження дозволяють перейти від розгляду поміток $\underline{\lambda_i^h}, \overline{\lambda_i^h}$ до $\underline{\mu_i^h}, \overline{\mu_i^h}$. При цьому допустима множина прийме вигляд $X=\left\{(a,b): 1\le a\le \underline{l^h}, 1\le b\le \overline{l^h}, V_{a,b}^h\ne\varnothing\right\}$, де $\underline{l^h}$, $\overline{l^h}$ – довжини упорядкувань $\underline{S^h}$ та $\overline{S^h}$ для графу $\tilde{G}$, тут і далі $V_{a,b}^h$ будуються, використовуючи значення $\underline{\mu_i^h}, \overline{\mu_i^h}$ замість $\underline{\lambda_i^h}, \overline{\lambda_i^h}$, без зміни позначень. Тоді умова прийме вигляд:

$$l=k+\max_{(a,b)\in X}\left((a-1)+\left\lceil\frac{\left|V_{a,b}^h\right|}{h}\right\rceil+\left(\overline{l^h}-b\right)\right).$$

Така форма є більш звичною для методу гілок та меж, оскільки тепер всі елементи, які приймають участь в пошуку максимуму пов'язані лише з проміжним графом $\tilde{G}$. Введена допустима множина також є більш зручною для опрацювання випадку $\underline{l^h}\ne\overline{l^h}$.

У загальному випадку, отриманий максимум буде оцінкою знизу довжини упорядкування для проміжного графу $\tilde{G}$, що випливає з наведеного вище аналізу доданків, з яких він складається. Можна також переконатися, що він є узагальненням вже відомих оцінок [14]. Це дозволяє очікувати, що використання такої оцінки знизу також дозволить скоротити кількість розгалужень у методі гілок та меж у загальному випадку.

Перейдемо тепер до питання, як можна ефективно обчислювати значення цього максимуму за відомих значень $\left(\underline{\mu_i^h}, \overline{\mu_i^h}\right), i=\overline{1,n}$.

Скористаємося тим, що маємо перевірити всі проміжки $[a;b]$ по черзі. Зафіксуємо деякі значення a та b. Легко побачити, що $V_{a,b+1}^h=V_{a,b}^h\cup\left\{v_i\in\tilde{V}:\overline{\mu_i^h}=b+1, \underline{\mu_i^h}\ge a\right\}$ та $V_{a+1,b}^h=V_{a,b}^h/\left\{v_i\in\tilde{V}:\overline{\mu_i^h}\le b, \underline{\mu_i^h}=a\right\}$.

Обчислимо спочатку числовий масив V_h, де на кожному місці $k=1,\dots,\overline{l^h}$ буде знаходитись кількість вершин, для яких $\overline{\mu_i^h}=k$, тоді $\left|V_{1,b}^h\right|=\sum_{k=1}^{b}V_h[k]$. Це дає змогу обчислити значення виразу для всіх значень b за лінійний час.

Для переходу до $a = 2$ маємо виключити з розгляду всі вершини, для яких $\underline{\mu_i^h} = 1$. Для цього достатньо у масиві V_h зменшити відповідні значення на кількість вершин $v_i : \overline{\mu_i^h} = k, \underline{\mu_i^h} = 1$. Після цього матимемо $\left|V_{2,b}^h\right| = \sum_{k=1}^{b} V_h[k]$. Продовжуючи таким чином знайдемо шуканий максимум.

Помітимо також, що якщо попередньо відсортуємо масив позначок за зростанням $\underline{\mu_i^h}$, то кожне значення масиву потрібно буде використати лише 1 раз і складність алгоритму складатиме $O\left(\underline{l^h} * \overline{l^h} + n\right)$.

Псевдокод запропонованого алгоритму наведено на рис. 1.

Вхідні дані: ширина упорядкування h,

лексикографічно відсортований масив пар $L = \left\{\left(\underline{\mu_i^h}, \overline{\mu_i^h}\right), i = \overline{1,n}\right\}$.

Вихідні дані: значення оцінки знизу $\tilde{l}$.

```
procedure enh_lower_estimate(L, h)
    μ̄_min := min_i(μ̄_i^h); μ̄_max := max_i(μ̄_i^h);
    for k := μ̄_min to μ̄_max
        V_h[k] := 0;
    end for
    for i := 1 to n
        V_h[μ̄_i^h] := V_h[μ̄_i^h] + 1;
    end for

    l̃ := 0; j := 1;
    while j ≤ n
        cnt := 0;
        for k := μ̄_min to μ̄_max
            cnt := cnt + V_h[k];
            if cnt > 0 then
                l̃ := max(l̃, μ_j^h − 1 + ⌈cnt/h⌉ + μ̄_max − k);
            end if
        end for
        while j ≤ n and (j = 1 or μ_{j−1}^h = μ_j^h)
            V_h[μ̄_j^h] := V_h[μ̄_j^h] − 1;
            j := j + 1;
        end while
    end while

    return l̃;
end procedure
```

Рисунок 1 – Псевдокод обчислення оцінки для довільного графу

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

У випадку, коли упорядкування $\underline{S}^h$ та $\overline{S^h}$ співпадають (всі вершини знаходяться на критичних шляхах), обчислення може бути пришвидшене до лінійної складності $O\left(\underline{l}^h\right)$, оскільки $\underline{\mu}_i^h = \overline{\mu_i^h}, i = \overline{1,n}$ і кількість вершин у $V_{a,b}^h$ можна обчислити, підсумувавши кількості вершин на місцях з a по b в упорядкуванні $\underline{S}^h$ ($\overline{S^h}$).

Цільова функція приймає вигляд $\max\limits_{1\le a\le b\le \underline{l}^h}\left((a-1)+\left\lceil\frac{1}{h}\sum\limits_{i=a}^{b}\left|\underline{S}^h[i]\right|\right\rceil+\left(\overline{l^h}-b\right)\right)$. Помітимо, що елемент $\overline{l^h}$ є сталим, а $l=(b-a+1)$ дорівнює кількості місць, що приймає участь у підсумуванні, тоді пошук максимуму зводиться до наступної задачі:

$$\max_{1\le a\le b\le \underline{l}^h}\left(\left\lceil\frac{1}{h}\sum_{i=a}^{b}\left|\underline{S}^h[i]\right|\right\rceil-l\right)=\max_{1\le a\le b\le \underline{l}^h}\left\lceil\frac{1}{h}\sum_{i=a}^{b}\left|\underline{S}^h[i]\right|-l\right\rceil=\max_{1\le a\le b\le \underline{l}^h}\left\lceil\frac{1}{h}\left(\sum_{i=a}^{b}\left|\underline{S}^h[i]\right|-l*h\right)\right\rceil=$$

$$=\max_{1\le a\le b\le \underline{l}^h}\left\lceil\frac{1}{h}\sum_{i=a}^{b}\left(\left|\underline{S}^h[i]\right|-h\right)\right\rceil=\left\lceil\frac{1}{h}\max_{1\le a\le b\le \underline{l}^h}\sum_{i=a}^{b}\left(\left|\underline{S}^h[i]\right|-h\right)\right\rceil.$$

Тобто отримали задачу пошуку безперервної послідовності з елементами $\left|\underline{S}^h[i]\right|-h$, яка має максимальну суму. Така задача ефективно розв'язується за допомогою алгоритму Кадана [15], складність якого і складає $O\left(\underline{l}^h\right)$.

Псевдокод цього алгоритму наведено на рис. 2.

Вхідні дані: ширина упорядкування h,
масив потужностей місць упорядкування $\underline{S_i^h}$: $C=\left\{\left|\underline{S_i^h}\right|, i=1,\dots,\underline{l^h}\right\}$.
Вихідні дані: значення оцінки знизу $\tilde{l}$.

```
procedure enh_lower_estimate_cp(C, h)
    cnt := 0; l̃ := l̲ʰ; l := 0;
    for i := 1 to l̲ʰ
        cnt := cnt + C[i];
        l := l + 1;
        if cnt ≥ l * h then
            l̃ := max(l̃, ⌈cnt/h⌉ + l̲ʰ − l);
        else
            cnt := 0; l := 0;
        end if
    end for

    return l̃;
end procedure
```

Рисунок 2 – Псевдокод обчислення оцінки для графу, всі вершини якого знаходяться на критичних шляхах

Висновки. Було розглянуто і досліджено низку необхідних умов існування щільного упорядкування, пов'язаних з обмеженістю потужності місць та можливістю їх заповнення. Ці умови вдалося об'єднати в одну та запропонувати ефективний алгоритм її перевірки.

В результаті також були отримані нові уточнені оцінки знизу довжини упорядкування та запропоновані узагальнення спеціальних упорядкувань $\underline{S^h}$ та $\overline{S^h}$, які враховують значення ширини h.

Подальшого дослідження потребує експериментальна оцінка прискорення методу гілок та меж при використанні запропонованих оцінок довжини упорядкування, як для графів з щільним упорядкуванням, так і в загальному випадку. Цікавим також є пошук класів графів, для яких отримані оцінки дають точне значення довжини, та вивчення алгоритмів, заснованих на методі гілок та меж з обмеженою глибиною пошуку.

ЛІТЕРАТУРА

1. Бурдюк В. Я., Турчина В. А. Алгоритмы параллельного упорядочения: учебное пособие. Днепропетровск: ДГУ, 1985. 83 с.

2. Levey E., Rothvoss T. A (1+epsilon)-Approximation for Makespan Scheduling with Precedence Constraints Using LP Hierarchies. *SIAM Journal on Computing*. 2019. P. 201–217.

3. Garg Sh. Quasi-PTAS for scheduling with precedences using LP hierarchies. *In 45th International Colloquium on Automata, Languages, and Programming, ICALP*. 2018.

4. Li S. Towards PTAS for Precedence Constrained Scheduling via Combinatorial Algorithms. *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2021. P. 2991–3010.

5. Nederlof J., Swennenhuis C. M. F., Węgrzycki K. A Subexponential Time Algorithm for Makespan Scheduling of Unit Jobs with Precedence Constraints. 2023. 25 p. (Preprint. arXiv; arXiv:2312.03495).

6. Gangal D., Ranade A. Precedence constrained scheduling in (2–7/3p+1)optimal. *Journal of Computer and System Sciences*. 2008. Vol. 74, no. 7. P. 1139–1146.

7. Muhuri P. K., Biswas S. K. Bayesian optimization algorithm for multi-objective scheduling of time and precedence constrained tasks in heterogeneous multiprocessor systems. *Applied Soft Computing*. 2020. Vol. 92. P. 106–274.

8. Yoo M. Real-time task scheduling by multiobjective genetic algorithm. *Journal of Systems and Software*. 2009. Vol. 82, no. 4. P. 619–628.

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

9. Turchyna V., Karavaiev K. Analysis of algorithms for constructing dense sequencing of digraphs vertices. *Proceedings of The Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020)*, Zaporizhzhia, 2020. P. 690–703.
10. Chardon M., Moukrim A. The Coffman-Graham Algorithm Optimally Solves UET Task Systems with Overinterval Orders. *SIAM Journal on Discrete Mathematics*. 2005. Vol. 19, no. 1. P. 109–121.
11. Vásquez Ó. C. The Scheduling Zoo. URL: http://schedulingzoo.lip6.fr/ (date of access: 01.03.2024).
12. Li K. Scheduling Precedence Constrained Tasks for Mobile Applications in Fog Computing. *IEEE Transactions on Services Computing*. 2022. P. 1–14.
13. Караваєв К.Д., Турчина В.А. Аналіз впливу автоморфізму графу на схеми направленого перебору. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*, м. Дніпро, 2021. Вип. 21. С. 94–104.
14. Турчина В.А., Караваєв К.Д. Дослідження оцінок довжини паралельного упорядкування вершин графу. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*, м. Дніпро, 2018. Вип. 18. С. 186–195.
15. Bentley J. Programming pearls. *Communications of the ACM*. 1984. Vol. 27, no. 9. P. 865–873.

REFERENCES

1. Burdiuk V. Ya., Turchyna V. A. Parallel sequencing algorithms: training manual. Dnipropetrovsk: DSU, 1985. 83 p.
2. Levey E., Rothvoss T. A (1+epsilon)-Approximation for Makespan Scheduling with Precedence Constraints Using LP Hierarchies. *SIAM Journal on Computing*. 2019. P. 201–217.
3. Garg Sh. Quasi-PTAS for scheduling with precedences using LP hierarchies. *In 45th International Colloquium on Automata, Languages, and Programming, ICALP*. 2018.
4. Li S. Towards PTAS for Precedence Constrained Scheduling via Combinatorial Algorithms. *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2021. P. 2991–3010.
5. Nederlof J., Swennenhuis C. M. F., Węgrzycki K. A Subexponential Time Algorithm for Makespan Scheduling of Unit Jobs with Precedence Constraints. 2023. 25 p. (Preprint. arXiv; arXiv:2312.03495).

6. Gangal D., Ranade A. Precedence constrained scheduling in (2–7/3p+1)optimal. *Journal of Computer and System Sciences*. 2008. Vol. 74, no. 7. P. 1139–1146.
7. Muhuri P. K., Biswas S. K. Bayesian optimization algorithm for multi-objective scheduling of time and precedence constrained tasks in heterogeneous multiprocessor systems. *Applied Soft Computing*. 2020. Vol. 92. P. 106–274.
8. Yoo M. Real-time task scheduling by multiobjective genetic algorithm. *Journal of Systems and Software*. 2009. Vol. 82, no. 4. P. 619–628.
9. Turchyna V., Karavaiev K. Analysis of algorithms for constructing dense sequencing of digraphs vertices. *Proceedings of The Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020)*, Zaporizhzhia, 2020. P. 690–703.
10. Chardon M., Moukrim A. The Coffman-Graham Algorithm Optimally Solves UET Task Systems with Overinterval Orders. *SIAM Journal on Discrete Mathematics*. 2005. Vol. 19, no. 1. P. 109–121.
11. Vásquez Ó. C. The Scheduling Zoo. URL: http://schedulingzoo.lip6.fr/ (date of access: 30.03.2024).
12. Li K. Scheduling Precedence Constrained Tasks for Mobile Applications in Fog Computing. *IEEE Transactions on Services Computing*. 2022. P. 1–14.
13. Karavaiev K. D., Turchuna V.A. Analysis of the effect of graph automorphism on state search schemes. *Problems of applied mathematics and mathematical modelling*, Dnipro, 2021. Vol. 21. P. 94–104.
14. Turchuna V. A., Karavaiev K. D. Investigation of the estimates of the length of parallel alignment of the vertices of the graph. *Problems of applied mathematics and mathematical modelling*, Dnipro, 2018. Vol. 18. P. 186–195.
15. Bentley J. Programming pearls. *Communications of the ACM*. 1984. Vol. 27, no. 9. P. 865–873.

Received 12.03.2024.
Accepted 14.03.2024.

On the necessary conditions for the existence of dense sequencing in the classical parallel sequencing problem

The rapid development of the scheduling theory in the middle of the last century was linked to the variety of important practical applications of the problems it considers. Special attention was paid to problems in which the order of job execution is subject to certain technological constraints. One of the common mathematical models of these problems is the parallel sequencing problem.

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

We consider the classical problem of minimizing the length of a sequencing for a given width, in which the target sequencing is dense. Since the polynomial tractability of these problems for fixed width > 2 is unknown, the main areas of research on this problem include searching for classes of graphs for which exact polynomial algorithms exist, developing approximate algorithms and ways to prune state space search schemes.

Substantial progress has been made in recent years in the development of approximate algorithms with quasi-polynomial complexity and algorithms based on metaheuristics.

In addition to the classical problem, scientists also consider its generalizations, which have more complex structures of jobs and workers, additional constraints on the job execution, other objective functions, etc. Due to the development of fog computing in recent years, many articles have been devoted to the study of such problems within this particular application area.

The aim of this study was to investigate the constraints imposed on intermediate graphs by the condition of density of the target sequencing in the branch-and-bound method, to derive the necessary conditions for the existence of a dense sequencing and to propose methods to test them.

The necessary conditions for the existence of a dense sequencing when using the branch-and-bound method, related to the limited capacity of places and the possibility of filling them, are investigated. The obtained conditions were reduced to a single one, and efficient algorithms to test it in general and for graphs with all vertices on critical paths were proposed. In addition, the study also resulted in new improved lower bound estimates of the sequencing length and generalization of special sequencings in which the vertices occupy the leftmost and rightmost possible places, that take into account the sequencing width.

Keywords: graphs, combinatorial optimization, scheduling theory, branch-and-bound method, graph labeling, dense sequencings, optimal vertex distribution.

Караваєв Костянтин Дмитрович – аспірант кафедри обчислювальної математики та математичної кібернетики Дніпровського національного університету імені Олеся Гончара (м. Дніпро).

Karavaiev Kostiantyn – post-graduate student of the Department of Calculating Mathematics and Mathematical Cybernetics, Oles Honchar Dnipro National University, Dnipro, Ukraine.