УЗАГАЛЬНЕНА ЗАДАЧА ПАРАЛЕЛЬНОГО УПОРЯДКУВАННЯ

Задача 1а. По заданим графу G і вектору значень ширини $h_i, i = \overline{1,n}$ побудувати паралельне упорядкування мінімальної довжини, в якому на i-ому місці буде стояти не більше h_i вершин.

Твердження. Якщо існує точний алгоритм A поліноміальної складності для деякої фіксованої ширини упорядкування $\tilde{h}$, тоді існує поліноміальний алгоритм і для *задачі 1а*, в якій всі $h_i < \tilde{h}, i = \overline{1,n}$.

Аналіз випадків порушення оптимальності алгоритму, заснованого на рівневому принципі

При використанні алгоритму, заснованого на рівневому принципі, для розв'язання *задачі 1а*, він буде наближеним навіть для дерев, більш того він залишається наближеним і для бінарних дерев.

Отриманий розв'язок може бути неоптимальним з двох причин:

1) через неоднозначність вибору вершин з однаковими мітками;

2) через те, що у оптимальному упорядкуванні вершини, які мають менші мітки, стоять раніше аніж вершини з більшими.

Розглянемо приклади порушення оптимальності алгоритму, заснованого на рівневому принципі, для бінарних дерев з обох причин.

Приклад 1. Застосуємо рівневий принцип для побудови $P_S(G, h_i, l)$ для графу з рис. 1 та $h_i = 2,3,1,1,1, \dots$.

1 3 5 6 7

2 4

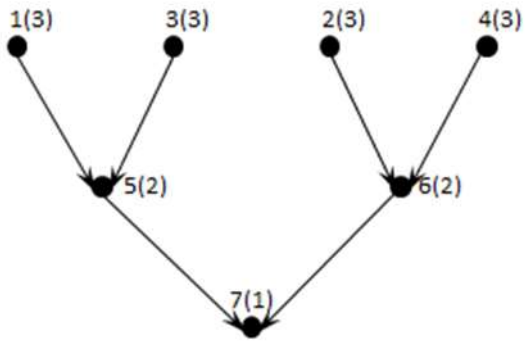

Рис. 1. Зображення графу з прикладу 1

Маємо 4 вершини, які мають мітку рівну 3, це вершини 1, 2, 3 та 4. На перше місце можна поставити $h_1 = 2$ вершин, які, згідно з алгоритмом, ми обираємо довільно. Виберемо, наприклад, вершини 1 та 2, при такому виборі не відкривається жодна вершина. Тоді на друге місце можемо поставити лише 2 вершини з $h_2 = 3$ можливих, а саме 3 і 4. Це відкриє вершини 5 та 6. Оскільки, $h_3 = h_4 = h_5 = 1$, то решту вершин можемо послідовно поставити лише по одній на кожне місце. Отже, отримаємо наступне упорядкування

$$S = \left\{\begin{matrix}1 & 3\\ 2, & 4,\end{matrix} 5,6,7\right\}.$$

Отримане упорядкування має довжину $l = 5$. Проте, якщо замість вершин 1 та 2 на першому кроці обрати вершини 1 та 3, то це відкриє вершину 5, і тоді можна буде поставити на 2 місце вже 3, а не 2 вершини. Матимемо наступне:

$$S^* = \left\{\begin{matrix}1 & 2\\ 3, & 4, 6,7\\ & 5\end{matrix}\right\}.$$

Таке упорядкування вже буде оптимальним, і його довжина $l^* = 4$ (меншу довжину отримати неможливо, оскільки перші три місця вміщують не більше 6 вершин, а у графі їх 7). Бачимо, що отримали неоптимальний розв'язок через довільність вибору вершин з однаковими мітками.

Приклад 2. Застосуємо рівневий принцип для побудови $P_S(G, h_i, l)$ для графу з рис. 2 та $h_i = 2,2,4,2,1,1,1,1, \dots$.

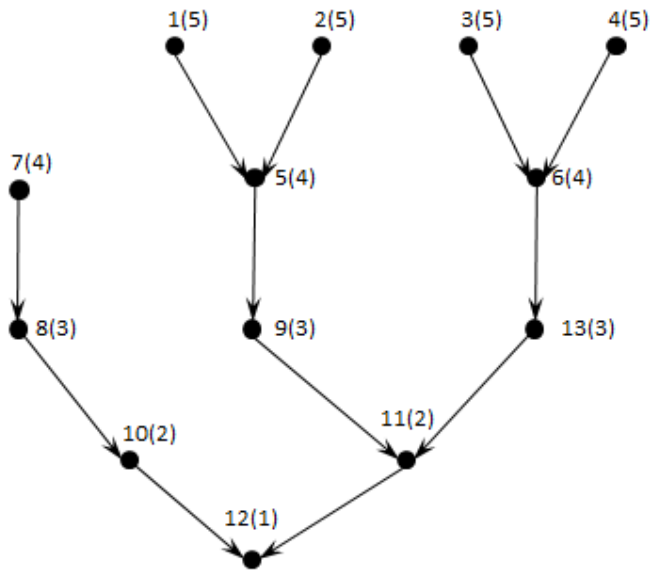

Рис. 2. Зображення графу з прикладу 2

У цьому випадку, згідно з рівневим принципом, перші два місця потрібно заповнити вершинами 1-4 (по дві на кожне), що відкриє вершини 5 та 6. Потім на 3 місце зможемо поставити лише 3 вершини з $h_3 = 4$: 5, 6 та 7 вершини. Після чого на 4 та 5 місцях розташовуємо вершини 8, 9 та 13. Вершини, що залишилися (10, 11 та 12) розташовуємо послідовно на місцях з 6 по 8. Врешті решт, отримаємо:

$$S = \begin{Bmatrix} 1 & 3 & 5 & 8 \\ 2, & 4, & 6, & 9, \\ & & 7 & \end{Bmatrix} 13,10,11,12\Big\}.$$

Довжина такого упорядкування $l = 8$. Оптимальне ж упорядкування отримаємо у випадку, якщо на перше місце поставимо 1 та 7 вершину, а на друге – 2 і 8, тоді третє місце зможемо заповнити повністю вершинами 3, 4, 5 та 10, потім відкритими залишаться вершини 6 і 9, які й поставимо на 4 місце, а вершини 13, 11 та 12 займуть місця з 5 по 7. Отже, маємо наступне:

$$S^* = \begin{Bmatrix} & & 3 & \\ 1 & 2 & 4 & 6, \\ 7, & 8, & 5, & 9, \\ & & 10 & \end{Bmatrix} 13,11,12\Big\}.$$

Довжина такого упорядкування $l^* = 7$ буде оптимальною. У цьому випадку причиною неоптимальності було те, що вершини 7 та 8, які мають мітки 4 стоять у оптимальному упорядкуванні раніше, аніж вершини 3 та 4, які мають мітки рівні 5.

З прикладів 1 та 2 бачимо, що алгоритм, заснований на рівневому принципі, перестає бути точним навіть для бінарних дерев. І якщо довільність вибору можна було б скорегувати шляхом переходу до якогось ізоморфного графу, для якого отримане упорядкування було б оптимальним, то друга причина вказує на неспроможність методу як такого, що може потенційно точно розв'язувати *задачу 1а*. Отже, розв'язання цієї задачі, хоча б для бінарних дерев, потребує розробки нових підходів та алгоритмів, які будуть більш детально враховувати структуру графу.