УДК 519.8 doi: 10.15421/322110

К.Д. Караваєв, В.А. Турчина
Дніпровський національний університет імені Олеся Гончара

АНАЛІЗ ВПЛИВУ АВТОМОРФІЗМУ ГРАФУ НА СХЕМИ НАПРАВЛЕНОГО ПЕРЕБОРУ

У статі досліджується залежність між наявністю автоморфізму у графі та ізоморфізмом його підграфів, отриманих шляхом видаленням з нього відкритих вершин. Запропоновано алгоритм для скорочення кількості розгалужень у методі гілок та меж для задачі паралельного упорядкування вершин орграфів. Показано, що результуюча множина містить всі неізоморфні підграфи, проте також може містити й ізоморфні.

Ключові слова: дискретна оптимізація, теорія розкладів, оптимальні упорядкування, метод гілок та меж, перерахування без ізоморфізму.

K.D. Karavaiev, V.A. Turchyna
Oles Honchar Dnipro National University

ANALYSIS OF THE EFFECT OF GRAPH AUTOMORPHISM ON STATE SEARCH SCHEMES

Scheduling theory problems have found wide practical application in various spheres of life. One of the most common tasks in manufacturing is to determine some order of jobs that maximizes or minimizes some required quality criterion. In cases when some technological constraints on the order of the job execution are given, discrete optimization problems on graphs can be chosen as a mathematical model of the problem.

This article is devoted to the problem of finding a parallel sequencing of vertices of a given directed graph of minimum length, in which at each place there is no more than a given fixed number of vertices.

Since the problem belongs to the class of NP-complete, the construction of efficient state search schemes is of great importance. A variant of branch-and-bound method in which possible combinations of open vertices are enumerated is commonly applied for solving such problems.

Since the number of combinations grows rapidly as the number of vertices in a graph increases, the search for a solution in some cases can take significant amount of time. Previous research on increasing the efficiency of the method mainly focused on improving the accuracy of the lower-bound estimate of time expenditures. The authors investigated an approach to speed up search when using the branch-and-bound method by removing states corresponding to isomorphic graphs.

The relationship between the presence of an automorphism in a graph and the isomorphism of its subgraphs obtained by removing open vertices from it is considered. As a result, an algorithm to reduce the number of added possible states in the branch-and-bound method is proposed. It is shown that the set of subgraphs generated by the algorithm contains all non-isomorphic subgraphs, but may also contain some isomorphic ones.

Keywords: discrete optimization, scheduling theory, optimal sequencing, branch-and-bound algorithm, isomorph-free enumeration.

К.Д. Караваев, В.А. Турчина
Днипровский национальный университет имени Олеся Гончара

АНАЛИЗ ВЛИЯНИЯ АВТОМОРФИЗМА ГРАФА НА СХЕМЫ НАПРАВЛЕННОГО ПЕРЕБОРА

В статье исследуется зависимость между наличием автоморфизма в графе и изоморфизмом его подграфов, полученных удалением открытых вершин. Предложен алгоритм для сокращения разветвлений в методе ветвей и границ для задачи параллельного упорядочения вершин орграфов. Показано, что результирующее множество содержит все неизоморфные подграфы, но может также содержать изоморфные.

Ключевые слова: дискретная оптимизация, теория расписаний, оптимальные упорядочения, метод ветвей и границ, перечисление без изоморфизма.

Одними з важливих задач теорії дискретної оптимізації є задачі упорядкування вершин орієнтовних графів. Такі задачі найчастіше виникають при необхідності побудови розкладів для розподілу задач між виконавцями з метою досягнення екстремуму деякого критерію якості.

Дослідження задач з цього класу розпочалося у середині минулого століття для оптимальної організації робіт на конвеєрі на автомобільній фабриці. Розв'язок задачі був отриманий швидко і вважалося, що буде знайдено метод розв'язання і в загальному випадку.

Розглянемо класичну постановку задачі оптимального упорядкування.

Нехай маємо n завдань, для яких наявні деякі виробничі зв'язки, що регламентують порядок їх виконання. Технологічні обмеження задачі природно подавати у вигляді орієнтовного графу $G(V,U)$, у якому вершини відповідають завданням, а дуги – виробничим зв'язкам. В класичній постановці вважається, що витрати часу на виконання кожного завдання однакові.

Означення 1. Паралельним упорядкуванням вершин орієнтовного графу $G=(V,U)$ називається таке упорядкування його вершин по місцях, розташованих у лінію, при якому з того, що пара $(i,j)\in U$ випливає, що вершина i розташовується в упорядкуванні S лівіше вершини j, тобто з того, що $(i,j)\in U \wedge (i\in S[p], j\in S[q])$ випливає, що $p<q$.

Означення 2. Довжиною l упорядкування S називається число непорожніх місць в ньому: $l(S)=\sum_{i=1}^{n} sign|S[i]|$, де $S[i]$ – множини елементів, що знаходяться в упорядкуванні S на місці i.

Означення 3. Шириною h упорядкування S називається величина, що дорівнює найбільшій кількості елементів, що розташовані на одному місці: $h(S)=\max_i |S[i]|$, де $i=1,\ldots,n$.

Класична задача. По заданим графу G і значенню ширини h побудувати паралельне упорядкування мінімальної довжини.

Виявилось, що у загальному випадку ця задача є NP-важкою. Точні алгоритми поліноміальної складності відомі лише для деяких спеціальних видів графу G та $h=2$.

Одним з класичних і найбільш поширених точних методів розв'язання задач дискретної оптимізації є метод гілок та меж. Це метод спрямованого перебору множини допустимих розв'язків. Схему методу зручно зображати у вигляді так званого дерева варіантів. При розробці методу для конкретної задачі необхідно визначитися з двома питаннями:

1) як саме розбивати допустиму множину на підмножини;

2) як саме оцінювати значення цільової функції при вибраному розбитті.

У класичному варіанті застосування методу гілок та меж до задачі [1-3] розгалуження дерева варіантів ведеться шляхом обрання деякої комбінації відкритих вершин (вершин без вхідних дуг) у поточному графі. Ці вершини розташовуються у шукане упорядкування на місце з номером рівним відстані від поточного листа до кореня дерева варіантів. Після чого визначається оцінка знизу довжини паралельного упорядкування для підграфу, отриманого видаленням з поточного графу обраних вершин, до якої додається відстань до кореня дерева.

Для подальшого розгалуження обирається лист дерева варіантів, що має найменшу оцінку, і у якості поточного графу розглядається її відповідний підграф. Розгалуження відбувається доти, доки не отримаємо лист з найменшою у дереві оцінкою довжини упорядкування, для якого відповідний граф буде порожнім.

Зрозуміло, що швидкість роботи методу безпосередньо залежить від кількості вершин у дереві варіантів, які підлягають розгляду. Тому важливим напрямком дослідження методів гілок та меж є шляхи скорочення перебору.

Одним з таких шляхів є уточнення оцінок знизу для довжини упорядкування [3,4].

Справді, чим швидше мінімальна оцінка знизу довжини у дереві варіантів співпаде з оптимальним значенням, тим швидше буде знайдений розв'язок. Відмітимо, що при використанні лише цього підходу неминуче зіштовхнемося із випадком, коли повторно аналізуються однакові графи. Розглянемо цей випадок детальніше на прикладі.

Приклад. Розглянемо граф з рис. 1. Усі його вершини знаходяться на критичних шляхах. Розглянемо застосування до нього методу гілок та меж при $h = 3$ та при використанні базової оцінки знизу довжини упорядкування:

$$l \geq \max(\underline{l}, \lceil n/h \rceil),$$

де $\lceil \cdot \rceil$ – округлення до більшого цілого, n – кількість вершин у графі, $\underline{l}$ – довжина критичного шляху, h – ширина упорядкування.

Отримане для цієї оцінки дерево розгалужень зображено на рис. 2.

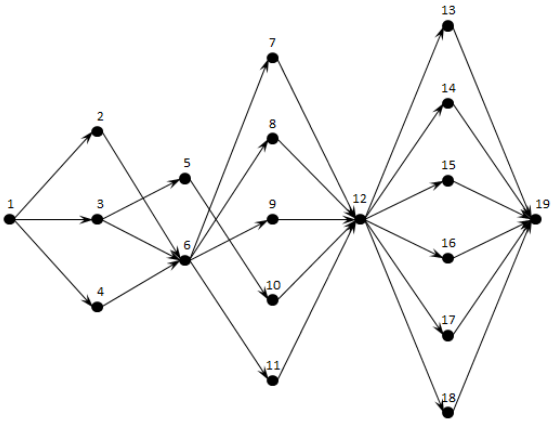

Рис. 1. Граф з прикладу 1.

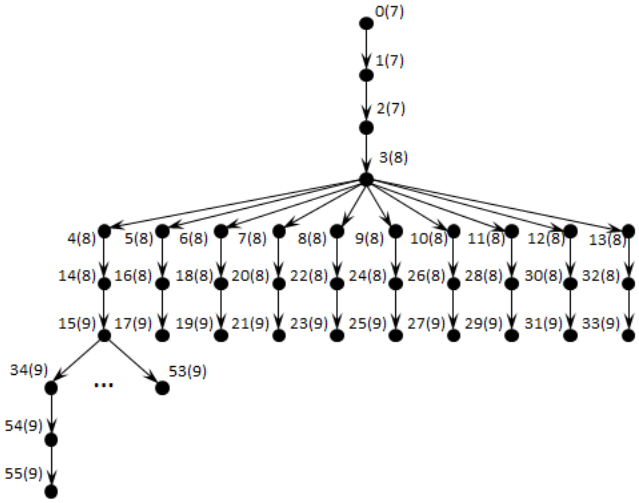

Рис. 2. Дерево варіантів з прикладу 1.

Наведемо проміжні упорядкування:

$$S_1=\{1,\ldots\};\ S_2=\left\{\begin{matrix}2\\1,3,\\4\end{matrix},\ldots\right\};\ S_3=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix},\ldots\right\};\ S_4=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix},\ldots\right\};\ S_5=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8\\10\end{matrix},\ldots\right\};\ S_6=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\11\end{matrix},\ldots\right\};$$

$$S_7=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9\\10\end{matrix},\ldots\right\};\ S_8=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9,\\11\end{matrix},\ldots\right\};\ S_9=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\10,\\11\end{matrix},\ldots\right\};\ S_{10}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9,\\10\end{matrix},\ldots\right\};\ S_{11}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9,\\11\end{matrix},\ldots\right\};$$

$$S_{12}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\10,\\11\end{matrix},\ldots\right\};\ S_{13}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}9\\10,\\11\end{matrix},\ldots\right\};\ S_{14}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix},\ldots\right\};\ S_{15}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}12,\ldots\right\};$$

$$S_{16}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8\\10\end{matrix}\begin{matrix}9\\11,\\\end{matrix},\ldots\right\};\ S_{17}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8\\10\end{matrix}\begin{matrix}9\\11,\\\end{matrix}12,\ldots\right\};\ S_{18}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\11\end{matrix}\begin{matrix}9\\10,\\\end{matrix},\ldots\right\};\ S_{19}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\11\end{matrix}\begin{matrix}9\\10,\\\end{matrix}12,\ldots\right\};$$

$$S_{20}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9\\10\end{matrix}\begin{matrix}8\\11,\\\end{matrix},\ldots\right\};\ S_{21}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9\\10\end{matrix}\begin{matrix}8\\11,\\\end{matrix}12,\ldots\right\};\ S_{22}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9,\\11\end{matrix}\begin{matrix}8\\10,\\\end{matrix},\ldots\right\};\ S_{23}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\9,\\11\end{matrix}\begin{matrix}8\\10,\\\end{matrix}12,\ldots\right\};$$

$$S_{24}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\10,\\11\end{matrix}\begin{matrix}8\\9,\\\end{matrix},\ldots\right\};\ S_{25}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\10,\\11\end{matrix}\begin{matrix}8\\9,\\\end{matrix}12,\ldots\right\};\ S_{26}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9\\10\end{matrix}\begin{matrix}7\\11,\\\end{matrix},\ldots\right\};\ S_{27}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9\\10\end{matrix}\begin{matrix}7\\11,\\\end{matrix}12,\ldots\right\};$$

$$S_{28}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9\\11\end{matrix}\begin{matrix}7\\10,\\\end{matrix},\ldots\right\};\ S_{29}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\9\\11\end{matrix}\begin{matrix}7\\10,\\\end{matrix}12,\ldots\right\};\ S_{30}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\10,\\11\end{matrix}\begin{matrix}7\\9,\\\end{matrix},\ldots\right\};\ S_{31}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}8\\10,\\11\end{matrix}\begin{matrix}7\\9,\\\end{matrix}12,\ldots\right\};$$

$$S_{32}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}9\\10,\\11\end{matrix}\begin{matrix}7\\8,\\\end{matrix},\ldots\right\};\ S_{33}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}9\\10,\\11\end{matrix}\begin{matrix}7\\8,\\\end{matrix}12,\ldots\right\};\ S_{34}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,14,\\15\end{matrix},\ldots\right\};$$

$$S_{35}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,14,\\16\end{matrix},\ldots\right\};\ S_{36}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,14,\\17\end{matrix},\ldots\right\};\ S_{37}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,14,\\18\end{matrix},\ldots\right\};$$

$$S_{38}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,15,\\16\end{matrix},\ldots\right\};\ S_{39}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,15,\\17\end{matrix},\ldots\right\};\ S_{40}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,15,\\18\end{matrix},\ldots\right\};$$

$$S_{41}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,16,\\17\end{matrix},\ldots\right\};\ S_{42}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,16,\\18\end{matrix},\ldots\right\};\ S_{43}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}13\\12,17,\\18\end{matrix},\ldots\right\};$$

$$S_{44}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}14\\12,15,\\16\end{matrix},\ldots\right\};\ S_{45}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}14\\12,15,\\17\end{matrix},\ldots\right\};\ S_{46}=\left\{\begin{matrix}2\\1,3,\\4\end{matrix}\begin{matrix}5\\6,\\\end{matrix}\begin{matrix}7\\8,\\9\end{matrix}\begin{matrix}10\\11,\\\end{matrix}\begin{matrix}14\\12,15,\\18\end{matrix},\ldots\right\};$$

$$S_{47}=\begin{Bmatrix}2&5&7&10&14\\1,3,&6,&8,&11,&12,16,...\\4&9&&&17\end{Bmatrix};\ S_{48}=\begin{Bmatrix}2&5&7&10&14\\1,3,&6,&8,&11,&12,16,...\\4&9&&&18\end{Bmatrix};\ S_{49}=\begin{Bmatrix}2&5&7&10&14\\1,3,&6,&8,&11,&12,17,...\\4&9&&&18\end{Bmatrix};$$

$$S_{50}=\begin{Bmatrix}2&5&7&10&15\\1,3,&6,&8,&11,&12,16,...\\4&9&&&17\end{Bmatrix};\ S_{51}=\begin{Bmatrix}2&5&7&10&15\\1,3,&6,&8,&11,&12,16,...\\4&9&&&18\end{Bmatrix};\ S_{52}=\begin{Bmatrix}2&5&7&10&15\\1,3,&6,&8,&11,&12,17,...\\4&9&&&18\end{Bmatrix};$$

$$S_{53}=\begin{Bmatrix}2&5&7&10&16\\1,3,&6,&8,&11,&12,17,...\\4&9&&&18\end{Bmatrix};\ S_{54}=\begin{Bmatrix}2&5&7&10&13\ 16\\1,3,&6,&8,&11,&12,14,17,...\\4&9&&&15\ 18\end{Bmatrix};\ S_{55}=\begin{Bmatrix}2&5&7&10&13\ 16\\1,3,&6,&8,&11,&12,14,17,19\\4&9&&&15\ 18\end{Bmatrix}.$$

Отже, при використанні цієї оцінки в методі гілок та меж знадобилося 55 розгалужень, причому розгалуження 1-15 є необхідними, тобто вони завжди будуть наявні у дереві не залежно від оцінки, оскільки отримані у випадках, коли усі вершини одного фіксованого рівня у дереві мали однакові оцінки. Аналогічно, розгалуження 34-55 також будуть у кожному дереві. Усі інші розгалуження, пов'язані з вибором вершини, що розглядається.

Розглянемо детальніше розгалуження 4-13. Легко побачити, що всім їм відповідають ізоморфні підграфи, тоді для кожного з них довжини оптимальних упорядкувань є рівними, а отже всі гілки, що йдуть з вершин 4-13 будуть однаковими і має сенс розглядати лише одну з них. Аналогічна ситуація відбувається з розгалуженнями 34-53. Отже, при додатковій перевірці на ізоморфність, вдалося б знайти оптимальне упорядкування за 10 розгалужень.

Варто відмітити також, що розгалужень 16-33 вдалося б також уникнути при застосуванні більш точнішої оцінки, тобто знадобилося б 37 розгалужень замість 55. Навіть маючи абсолютно точну оцінку, яка завжди дорівнює оптимальній довжині, цю кількість не вдалося б зменшити. А оскільки задача належить до класу NP-важких, то очікується, що таку оцінку побудувати не можна, а отже для будь-якої оцінки у загальному випадку у дереві розгалужень будуть виникати розгалуження, які відповідають ізоморфним графам, та для яких оцінка буде давати однакові значення. Так, якби оцінка рівна 9 вперше була отримана у розгалуженні 54, то у дереві було б додаткових 379 розгалужень. Важливо відмітити, що кількість можливих розгалужень швидко зростає з кількістю відкритих вершин.

З цього випливає важливість дослідження гілок дерев на ізоморфність відповідних підграфів, що дасть змогу не лише значно зменшити кількість вершин, що підлягають аналізу, а отже значно його пришвидшити, а й скоротити витрати ресурсів пам'яті.

Перед тим як перейти до розгляду вказаної проблеми, наведемо відомі поняття «ізоморфізму», «автоморфізму» та «інваріанту» орієнтовних графів [5].

Два орієнтовні графи G та H є ізоморфними, якщо між множинами їх вершин існує взаємно однозначна відповідність, що зберігає суміжність вершин та орієнтацію дуг. Позначається $G\cong H$ або $G=H$.

Автоморфізмом орієнтовного графу G називається ізоморфізм G на себе, або автоморфізм є підстановкою множини вершин V, що зберігає суміжність та орієнтацію дуг. Дві вершини i та j, що переходять одна в одну під дією

автоморфізму $f:V\to V$ ($i\in V, j\in V$, $f(i)=j$) будемо називати взаємозамінними та позначати $i\sim j$.

Інваріантом графу G називається деяка характеристика графу, найчастіше числова, пов'язана із графом G, яка зберігається для будь-якого графу, що є ізоморфним G. Прикладами інваріантів є кількість вершин чи дуг у графі.

Відзначимо, що на разі невідомо чи належать задачі перевірки графів на ізоморфність та визначення взаємозамінних вершин до класу P або NP-повних. Варто зазначити також, що для практично важливих класів графів існують алгоритми, що швидко визначають ізоморфізм та знаходять взаємозамінні вершини для графів великої розмірності [6-9].

Наведена проблема є тематично близькою до задачі генерації деякого класу графів без ізоморфізму [10-14], що тісно пов'язана з визначенням взаємозамінних вершин. Далі дослідимо твердження, що можуть скоротити кількість розгалужень, завдяки врахуванню взаємозамінності відкритих вершин.

Відмітимо також, що можливий підхід, який заснований на перевірці ізоморфізму за допомогою інваріантів. Насправді, відомо, що для деяких інваріантів ймовірність, що два випадкові графи будуть мати однакові значення цих інваріантів та не будуть при цьому ізоморфними, майже нульова [15]. Проте у випадку, що розглядається, при розгалужені підграфи утворюються з графу шляхом видалення деяких відкритих вершин, а тому більша частина структури підграфів буде залишатися сталою, а отже можна очікувати, що й більшість інваріантів для підграфів будуть збігатися.

У подальших дослідженнях вважається, що якимось чином вдалося визначити, які вершини у графі є взаємозамінними. Перевага у роботі віддається тому, як можна скористатися цією інформацією, а не як її отримати.

Введемо допоміжне означення.

Означення 4. Два набори вершин графу α та β називатимемо еквівалентними, якщо існує бієктивне відображення $f:\alpha\to\beta$ таке, що проектує вершини α у відповідні їм взаємозамінні вершини з β, тобто $\forall i\in\alpha: f(i)\in\beta\wedge i\sim f(i)$.

При розгляді цієї проблеми природно виникають два питання: чи можна при видаленні *еквівалентних* наборів отримати не *ізоморфні* підграфи; та чи можна при видаленні *неекві*валентних наборів отримати *ізоморфні підграфи*. Відповіді на обидва питання є позитивними, проілюструємо це на прикладах.

Приклад 2. Розглянемо граф G з рис. 3. Взаємозамінними відкритими вершинами у ньому є вершини 1 та 4, а також вершини 2 та 3. Нехай $h=2$, отже маємо обрати 2 вершини. З вершин 1-4 можна утворити 4 попарно еквівалентних набори: $(1,2),(1,3),(2,4),(3,4)$. Граф G_1 утворений з графу G видаленням набору $(3,4)$, а граф G_2 – набору $(2,4)$.

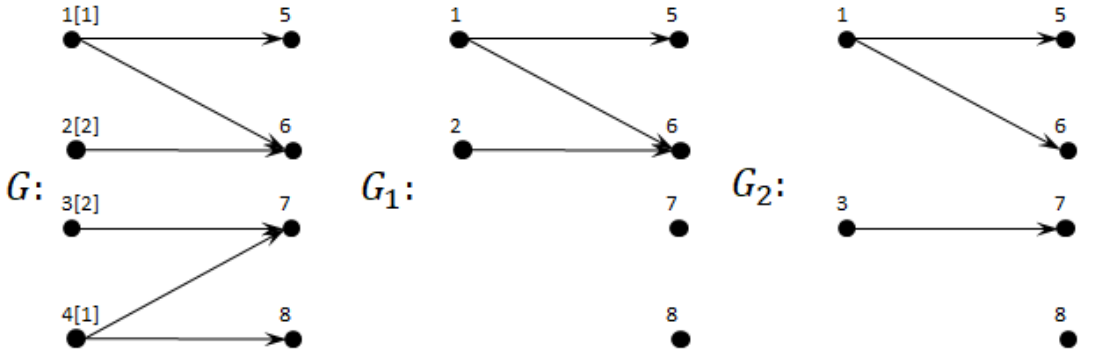

Рис. 3. Граф та його підграфи з прикладу 2.

Легко побачити, що графи G_1 та G_2 не ізоморфні: вони мають різну кількість ізольованих вершин. Також помітимо, що граф утворений видаленням набору $(1,2)$ ізоморфний графу G_1, а видаленням набору $(1,3)$ – графу G_2.

З іншого боку цікавим є факт, що після видалення вершини 4, що присутня в обох наборах, вершини 2 та 3 перестають бути взаємозамінними, тобто отримали неізоморфні графи через те, що видалили не взаємозамінні вершини. В той же час у випадку наборів $(1,2)$ та $(3,4)$ після видалення, наприклад, вершин 1 та 4, вершини 2 та 3 залишаються взаємозамінними та навпаки.

Приклад 3. Розглянемо граф G з рис. 4. Взаємозамінними відкритими вершинами у ньому є вершини 1 та 2. Нехай $h=2$, маємо обрати для видалення 2 вершини. Розглянемо два не еквівалентних набори $(1,2)$ та $(2,3)$. Граф G_1 утворений з графу G видаленням набору $(2,3)$, а граф G_2 – набору $(1,2)$.

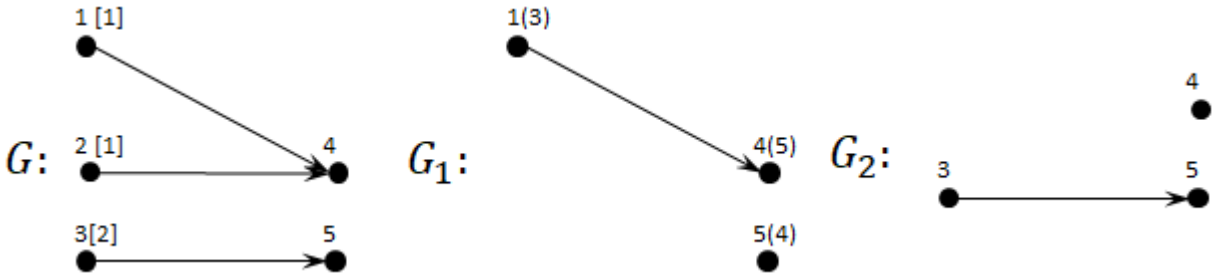

Рис. 4. Граф та його підграфи з прикладу 3.

Графи G_1 та G_2 є ізоморфними (ізоморфізм вказаний на зображенні графу G_1 у дужках). Також будуть ізоморфними графи, утворені видаленням еквівалентних наборів $(1,3)$ та $(2,3)$.

Відмітимо також, що аналогічно до прикладу 2, після видалення вершини 2, що є спільною для наборів, вершин 1 та 3 стають взаємозамінними, тобто отримали ізоморфні графи через те, що видаляємо взаємозамінні вершини.

З розглянутих прикладів бачимо, що визначення взаємозамінних вершин лише у початковому графі та побудови еквівалентних наборів вершин недостатньо для визначення ізоморфності підграфів, що утворюються при розга-

луженні. З іншого боку побачили, що важливим є визначення взаємозамінних вершин після видалення кожної вершини з графу. На основі цього можна запропонувати наступний алгоритм генерації неізоморфних підграфів.

Алгоритм

Вхідні дані: граф G, ширина упорядкування h.

Вихідні дані: множина наборів вершин N.

procedure generate$(G, h_{\max}, V_{out}, P, N)$
  if length(P) + *length*$(V_{out}) < h_{\max}$ *then*
    exit;
  end if

  $S \leftarrow$ {попарно не взаємозамінні вершини у G з множини V_{out}};
  $V'_{out} \leftarrow V_{out}$;
  for v *in* S
    if length(P) + 1 = $h_{\max}$ *then*
      $N \leftarrow N \cup (P \cup v)$;
    else
      generate$(G \setminus v, h_{\max}, V'_{out}, P \cup v, N)$;
      $V'_{out} \leftarrow V'_{out} \setminus$ {вершини з V_{out} взаємозамінні з v};
    end if
  end for
end procedure

$N \leftarrow \varnothing; P \leftarrow \varnothing; V_{out} \leftarrow$ {відкриті вершини з G};
generate$(G, h_{\max} = \min(h, length(V_{out})), V_{out}, P, N)$;

Для обґрунтування алгоритму потрібно довести два факти: що множина N містить набори, що дають всі можливі неізоморфні підграфи, і що множина N не містить наборів, що дають ізоморфні підграфи.

Перший факт напряму випливає з наступної теореми.

Теорема 1. Якщо у графі G вершини i та j взаємозамінні, то графи G_1 та G_2, утворені видаленням вершин i та j з G відповідно, є ізоморфними.

Доведення. Розглянемо автоморфізм f, який переводить вершину i у вершину j. Побудуємо дві матриці суміжності: в першій рядки і стовпці відповідають вершинам у порядку $1,2,\ldots,n$ (n – кількість вершин у графі), в другій – у порядку $f(1), f(2),\ldots, f(n)$.

За означенням автоморфізму ці матриці суміжності будуть збігатися, більш того, оскільки $f(i) = j$, i-тий рядок і стовпець у другій матриці відповідає вершини j, а отже матриці, що утворені з них видаленням цих рядків і стовпчиків, співпадають. З іншого боку, отримані матриці є матрицями суміжності для графів G_1 та G_2, з чого випливає, що вони ізоморфні за означенням. ■

Справді, з теореми випливає, що отримати неізоморфні графи, видаляючи взаємозамінні вершини, неможливо. А отже, коли у відповідності до алгоритму беремо лише одну вершину з кожної множини попарно взаємозамінних вершин, то не втрачаємо жодного з наборів відкритих вершин, що дають різні неізоморфні графи.

Перед дослідженням другого факту для обґрунтування алгоритму відмітимо, що важливою є умова, що видаляються саме відкриті вершини.

Приклад 4. Розглянемо граф G з рис. 5. Граф G_1 утворений з графу G видаленням вершини 3, граф G_2 – вершини 2, граф G_3 – вершини 1.

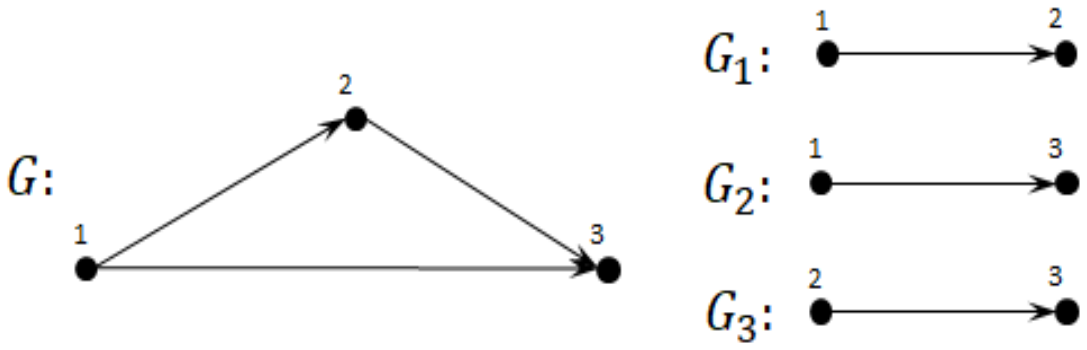

Рис. 5. Граф та його підграфи з прикладу 4.

Бачимо, що жодні дві вершини у графі не є взаємозамінними, проте у всіх випадках отримуємо ізоморфні графи.

Для того, щоб довести, що жодні два набори з N не можуть привести до ізоморфних графів, необхідно перевірити наступне твердження.

Твердження 1. Для будь-яких двох наборів відкритих вершин α та β графу G з того, що графи $G_1 = G \setminus \alpha$ та $G_2 = G \setminus \beta$ є ізоморфними, випливає існування таких послідовностей вершин $\alpha' = (\alpha_1,...,\alpha_m), \alpha_i \in \alpha, \alpha_i \neq \alpha_j$ та $\beta' = (\beta_1,...,\beta_m), \beta_i \in \beta, \beta_i \neq \beta_j$, що α_1 та β_1 є взаємозамінними та для всіх $k = 1..(m-1)$ для графів $G \setminus (\alpha_1,...,\alpha_k)$ та $G \setminus (\beta_1,...,\beta_k)$ існує ізоморфізм, що переводить вершину α_{k+1} у вершину β_{k+1}.

Це твердження є хибним. Для того, щоб це підтвердити розглянемо наступний приклад.

Приклад 5. Розглянемо граф G з рис. 6. Нехай $h = 2$, маємо обрати для видалення 2 вершини. Граф G_1 утворений з графу G видаленням вершин 2 та 3, граф G_2 – вершин 1 та 4.

Графи G_1 та G_2 є ізоморфними (ізоморфізм вказаний на зображенні графу G_1 у дужках), проте серед вершин 1-4 немає взаємозамінних, а отже множина N може містити два набори, видалення яких приводять до ізоморфних графів. Помітимо проте, що при одночасному видаленні з графу вершин 1 та 3, вершини 2 та 4 стають взаємозамінними та навпаки.

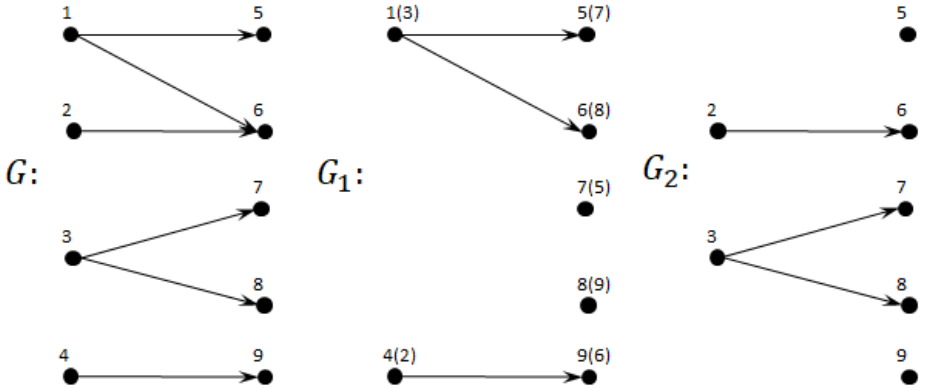

Рис. 6. Граф та його підграфи з прикладу 5.

Також варто перевірити чи є умова теореми 1 не лише необхідною, але й достатньою, тобто слабку версію попереднього твердження.

Твердження 2. Якщо графи G_1 та G_2, утворені видаленням деяких відкритих вершин i та j з G відповідно, є ізоморфними, то вершини i та j є взаємозамінними у графі G.

Це твердження також не є вірним. Проілюструємо це наступним прикладом.

Приклад 6. Розглянемо граф G з рис. 7. Граф G_1 утворений з графу G видаленням вершин 2, граф G_2 – вершин 1.

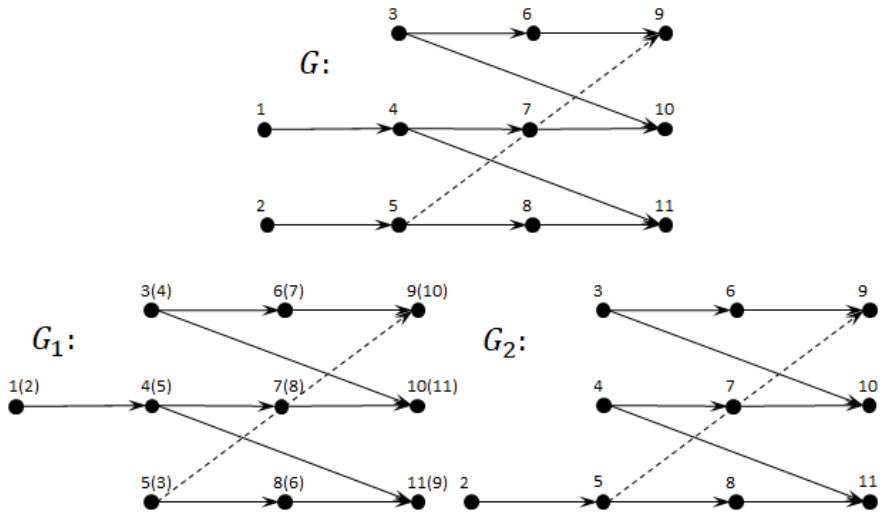

Рис. 7. Граф та його підграфи з прикладу 6.

Графи G_1 та G_2 є ізоморфними (ізоморфізм вказаний на зображенні графу G_1 у дужках), проте вершини 1 і 2 не є взаємозамінними у графі G. Це легко побачити, якщо розглянути відкриту вершину 11: відстань від вершини 1 до 11 дорівнює двом, а від вершини 2 – трьом, з іншого боку, відкритою вершину на відстані два від вершини 2 є вершина 9, а відкритою вершиною на відс-

тані три від 1 – вершина 10, а отже при автоморфізмі вершина 11 мала б відобразитись і у вершину 9, і у вершину 10, що неможливо.

Отже, запропонований алгоритм дозволяє скоротити кількість розгалужень, що підлягають перевірці у методі гілок та меж при його застосуванні до задачі паралельного упорядкування, але не дозволяє повністю позбавитися від ізоморфних підграфів.

Бібліографічні посилання

1. Fernandez, E.B., Buspell B. Bounds on the number of processors and time for multiprocessor optimal schedules. *IEEE Frans. Comput*. 1973. C–22. P. 745–751.
2. Graham R.L. Bounds on multiprocessing timing anomalies. *SIAM J. Appl. Math*. 1964. № 2. P. 416–429.
3. Шахбазян, К.В., Тушкина Т.А. Метод ветвей и границ для задачи параллельного упорядочивания. *Зап. науч. семинаров ЛОМИ АН СССР*. 1973. № 35. С. 146–155.
4. Турчина В.А., Караваєв К.Д. Дослідження оцінок довжини паралельного упорядкування вершин графу. *Питання прикладної математики і математичного моделювання*. Д. 2018. С. 186 – 195.
5. Харари Ф. Теория графов. М., 1973. 300 с.
6. McKay B.D. Practical Graph Isomorphism. *Congressus Numerantium*. 1981. P. 45–87.
7. Junttila T., Kaski P. Engineering an efficient canonical labeling tool for large and sparse graphs. *Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments*. 2007. P. 135–149
8. Darga P., Sakallah K., Markov I.L. Faster Symmetry Discovery using Sparsity of Symmetries. *Proceedings of the 45th Design Automation Conference*. 2008. P. 149–154.
9. Katebi H., Sakallah K., Markov I.L. Symmetry and Satisfiability: An Update. *Proc. Satisfiability Symposium*. 2010. P. 113-127.
10. Matsumoto Y., Moriyama S., Imai H., Bremner D. Matroid enumeration for incidence geometry. *Discrete Comput. Geom*. 2012. P. 17–43.
11. McKay B.D., Royle G.F. Constructing the cubic graphs on up to 20 vertices. *Ars Combinatoria*. 1986. P. 129–140.
12. Mayhew D., Royle G.F. Matroids with nine elements. *J. Comb. Theory*. 2008. P. 415–431.
13. Grüner T., Laue R., Meringer M. Algorithms for Group Actions: Homomorphism Principle and Orderly Generation Applied to Graphs. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. 1997. P. 113-122.
14. McKay, B.D. Isomorph-Free Exhaustive Generation. *Journal of Algorithms*. 1998. P. 306-324.
15. Babai L., Erdős P., Selkow S.M. Random graph isomorphism. *SIAM J. Comput*. 1980. P. 628–635.

Надійшла до редколегії 01.06. 2021.