

Дніпровський національний університет імені Олеся Гончара
Міністерство освіти і науки України
Дніпровський національний університет імені Олеся Гончара
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

КОВАЛЕНКО ЄВГЕН ОЛЕКСАНДРОВИЧ

УДК 519.854.2:519.176

ДИСЕРТАЦІЯ

**РОЗРОБКА АЛГОРИТМІВ ВРАХУВАННЯ ВПЛИВУ ПЕРЕРИВАНЬ НА
ОПТИМАЛЬНІСТЬ РОЗВ'ЯЗКІВ У ЗАДАЧАХ УПОРЯДКУВАННЯ**

11 – Математика та статистика

113 – Прикладна математика

Подається на здобуття ступеня доктора філософії.

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Є.О. Коваленко

Науковий керівник:

Турчина Валентина Андріївна,

кандидат фізико-математичних наук,

доцент

Дніпро – 2025

АНОТАЦІЯ

Коваленко Є.О. Розробка алгоритмів врахування впливу переривань на оптимальність розв'язків у задачах упорядкування. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 113 «Прикладна математика» (11 – Математика та статистика). – Дніпровський національний університет імені Олеся Гончара, Дніпро, 2025.

Дисертаційна робота присвячена розробці та обґрунтуванню алгоритмів врахування впливу переривань на значення цільової функції в підкласі задач теорії розкладів, що має назву задачі упорядкування.

Математичні основи теорії розкладів знаходять широке застосування при розв'язанні теоретичних та прикладних задач, пов'язаних з плануванням виробничих процесів, будівництва, підтримки комунікаційних мереж, у сферах обслуговування, логістики, при проєктуванні обчислювальних систем тощо. Для виконання деякої скінченної множини робіт задано певну кількість ресурсів (станків, працівників, процесорів тощо). Необхідно розподілити роботи між наявними ресурсами таким чином, щоб оптимізувати заздалегідь визначений для конкретної прикладної задачі критерій. Результатом такого розподілу є розклад виконання робіт. При цьому можуть бути задані додаткові умови: час готовності робіт до виконання; директивні терміни, що визначають бажані моменти завершення всіх робіт; обмеження на порядок виконання робіт; пріоритетність робіт; інші. Найчастіше застосовуються критерії оптимальності, що відповідають мінімізації: суми моментів завершення робіт, запізнення чи відхилення від директивних термінів, часу простоїв, або загальної тривалості виконання.

Ряд задач теорії розкладів передбачає, що задані ресурси представлені одним виконавцем. У тих випадках, коли їх кількість більша, допускається паралельне виконання робіт за умов, що в будь-який момент часу конкретна

робота може виконуватися не більше, ніж одним виконавцем, і відповідно кожен виконавець може виконувати не більше однієї роботи одночасно. Задачі теорії розкладів, в яких допускається наявність несуперечливих технологічних обмежень на порядок виконання робіт, класифікуються як задачі паралельного упорядкування. Вони відносяться до оптимізаційних задач на орієнтованих графах, де множині робіт відповідають вершини, а дуги задають технологічні обмеження. Дві класичні постановки таких задач полягають в побудові паралельних упорядкувань вершин орграфів мінімальної довжини при заданій ширині, або мінімальної ширини при заданій довжині. Відомо, що вони є *NP*-важкими.

У даній роботі основну увагу було приділено задачам у першій постановці. Для них відомі точні алгоритми експоненціальної складності, основані на методах повного чи спрямованого перебору, зведення до сіткових мереж та задач цілочисельного лінійного програмування. Для деяких підкласів задач упорядкування існують точні алгоритми поліноміальної складності, які для загальних постановок можуть давати наближені розв'язки. Однак оцінка якості таких розв'язків показує, що отримане значення цільової функції може майже вдвічі перевищувати оптимальне. Зокрема це стосується алгоритмів, заснованих на використанні рівневого принципу та лексикографічної розмітки вершин.

В класичних задачах упорядкування не розглядаються постановки, що враховують можливі переривання при виконанні робіт. Проте в багатьох прикладних задачах є можливість допускати деякі переривання при виконанні. Тоді виникає питання, а чи не може їх дозвіл покращити значення цільової функції? Дослідження цього питання розпочалися у середині минулого століття. Перший алгоритм розв'язання задачі у постановці, де допускалися переривання, був представлений у роботі Р. Макнотона у 1959 р. Він має поліноміальну складність і є точним для випадків, коли задана скінченна множина ідентичних за ефективністю виконавців, між якими розподіляється скінченна кількість робіт, що можуть виконуватися у довільному порядку, при цьому необхідно мінімізувати час завершення усіх робіт. Відтоді ця тема неодноразово

привертала увагу науковців. Алгоритми розв'язання для інших підкласів задач, де переривання робіт дозволені, були опубліковані у працях Е. Л. Лоулера, Т. Гонзалеса, Д. С. Джонсона, С. Сані, П. Брукера, Е. Г. Коффмана. Для ряду постановок в результаті досліджень Дж. К. Ленстри, Дж. Д. Ульмана, Дж. Ду, Р. А. Сіттерса та інших було доведено, що вони в загальному випадку є *NP*-повними, навіть з урахуванням допустимості переривань. Слід зауважити, що в перелічених роботах основна увага була приділена задачам з різними особливостями виконавців, критеріями оптимальності, відносними тривалостями виконання робіт, які здебільшого вважалися незалежними. В той же час специфіка технологічних обмежень на порядок виконання робіт залишалася малодослідженою. Найчастіше якщо такі випадки і розглядалися, то стосувалися постановок, у яких відповідний граф задачі відноситься до класу ланцюгів, чи вхідних або вихідних дерев.

У даній роботі досліджено задачі паралельного упорядкування вершин оргграфів, що відносяться до різних підкласів. Для них був проведений порівняльний аналіз розв'язків за умов, коли переривання при виконанні робіт відповідної прикладної задачі дозволені, чи заборонені, і отримано якісні оцінки можливого виграшу від такого дозволу. Це дало змогу виявити такі підкласи, для яких застосування переривань може суттєво впливати на оптимальність одержаного розв'язку. Прикладами таких підкласів є: повні дводольні, паралельно-последовні, дерева спеціальної структури, а також графи, що складаються з ізолюваних вершин. Для останніх було показано, що якщо час виконання усіх робіт однаковий, то дозвіл переривань дає змогу зменшити значення цільової функції майже у два рази, порівняно з випадком, коли такого дозволу немає. Отримана відповідна оцінка і показано, що вона є досяжною.

Для випадків, у яких технологічні обмеження моделюються повним дводольним графом було встановлено, що вплив переривань на зменшення значення цільової функції можна розглядати окремо для підмножини вершин кожної долі, окрім цього досліджено теоретичні аспекти залежності виграшу від початкових даних.

У постановках, де обмеження на порядок утворюють орієнтовані дерева спеціальних підкласів, в ході досліджень було встановлено, що виграш від переривань може суттєво відрізнятись в залежності від тривалості виконання робіт.

Розглянуто одне узагальнення задачі, пов'язане з обмеженнями на кількість ресурсів, які можна використовувати у фіксовані моменти часу. Для її розв'язання було розроблено алгоритми, які є точними за умов, що відповідний оргграф задачі складається з множини ізольованих вершин, або є регулярним і відноситься до підкласу повних дводольних.

Був проведений аналіз можливості покращення наближених розв'язків, одержаних за допомогою двох відомих алгоритмів поліноміальної складності. Показано, що дозвіл на переривання робіт, які відповідають вершинам графа, розміщеним на нещільно заповнених місцях упорядкування, може скоротити його довжину.

Встановлено зв'язок між постановками задачі упорядкування, в якій допустимі переривання, та одновимірної задачі пакування. Сформульовані умови, за яких можливе зведення задач в одній з постановок до іншої.

Наукова новизна результатів, одержаних у ході дисертаційного дослідження, полягає в наступному:

- виявлено нові підкласи графів для яких дозвіл переривань покращує розв'язки задач упорядкування;
- удосконалено математичні моделі задач упорядкування, які дозволяють враховувати дозвіл на переривання робіт у відповідних прикладних задачах;
- *вперше* введені оцінки для апріорного визначення виграшу від дозволу переривань для ряду підкласів графів;
- *вперше* визначено, які початкові дані задачі при дозволених перериваннях мають вплив на оптимальність розв'язку для одного підкласу графів;

- *вперше* проаналізовано вплив значень вагових коефіцієнтів, що відповідають часу виконання робіт, на ефективність переривань для одного підкласу дерев;
- *дістало подальшого розвитку* дослідження зв'язку задач пакування та упорядкування;
- *дістав подальшого розвитку* аналіз впливу дозволу переривань на оптимальність наближених розв'язків;
- *вперше* розроблено точні алгоритми розв'язання для узагальнених задач з перериваннями для двох підкласів графів;
- розроблено програмний продукт, який реалізує запропоновані алгоритми;
- проведено ряд обчислювальних експериментів з метою верифікації отриманих оцінок та запропонованих алгоритмів.

Практичне значення одержаних результатів полягає у наступному: отримані теоретичні результати дозволяють прогнозувати доцільність переривань в тих прикладних задачах, де технологічні процеси моделюються:

- графами, що складаються з ізольованих вершин;
- паралельно-послідовними графами;
- повними дводольними графами;
- спеціальними класами дерев.

Такі процеси зокрема виникають у сфері обслуговування, в управлінні проектами при розподілі робіт між працівниками, у промисловості при виготовленні чи утилізації виробів, у плануванні будівництва, проведенні паралельних обчислень тощо.

Розроблені алгоритми можуть застосовуватися до ряду прикладних задач, в яких кількість доступних ресурсів не є сталою величиною.

Отримані результати можуть бути рекомендовані до використання в навчальному процесі при підготовці студентів спеціальностей «Системний аналіз» та «Прикладна математика».

Основні результати дисертаційної роботи опубліковано в 14 наукових працях: 5 статей у наукових фахових виданнях України категорії Б з фізико-

математичних наук, 9 тез доповідей у збірниках матеріалів регіональних і міжнародних науково-практичних конференцій та семінарів.

Ключові слова: задачі комбінаторної оптимізації, теорія розкладів, дискретна оптимізація, задачі паралельного упорядкування, граф, регулярні графи, орієнтовані графи, повні дводольні графи, граціозна розмітка, дерева-зірки, гусениці, комбінаторна конфігурація, оптимальне розбиття, переривання, наближені розв'язки.

ABSTRACT

Kovalenko Y. Development of algorithms taking into account the influence of interruptions on the sequencing problems solution optimality. – Qualifying scientific work on manuscript rights.

Dissertation for obtaining the degree of Doctor of Philosophy in the specialty 113 «Applied Mathematics» (11 – Mathematics and Statistics). – Oles Honchar Dnipro National University, Dnipro, 2024.

The dissertation is devoted to the development and justification of algorithms that account for the influence of interruptions on the value of the objective function in a subclass of scheduling theory problems called sequencing problems.

The mathematical foundations of scheduling theory are widely used in solving theoretical and applied problems related to the planning of production processes, construction, communication networks support, in the areas of service, logistics, in the design of computing systems, etc. To perform a finite set of jobs, a certain number of resources (machine tools, workers, processors, etc.) is given. It is necessary to distribute the jobs between the available resources in such a way as to optimize the criterion previously determined for a specific applied problem. The result of such distribution is a schedule for the execution of jobs. In this case, additional conditions may be set: the time of readiness for each job's execution; directive terms that determine the desired moments of completion of all jobs; restrictions on the order of jobs' execution; priority of jobs; others. The most commonly used optimality criteria are the minimization of: the sum of the times of job completion, the delay or deviation from the target terms, the downtime, or the total execution duration.

A number of scheduling theory problems assume that the given resources are represented by a single executor. In cases where their number is greater, parallel jobs' execution is allowed under the condition that at any given time, a specific work can be performed by no more than one executor, and accordingly, each executor can perform no more than one job simultaneously. Scheduling theory problems in which the

presence of consistent technological constraints on the order of jobs' execution is allowed are classified as parallel ordering problems. They belong to optimization problems on directed graphs, where the set of jobs corresponds to vertices, and the arcs specify technological constraints. Two classical statements of such problems consist in constructing parallel sequences of digraphs' vertices of minimal length at a given width or of minimal width at a given length. It is known that they are *NP*-hard.

In this work, the main attention was paid to the problems in the first formulation. For them, exact algorithms of exponential complexity are known based on the methods of complete or directed search, reduction to mesh networks, and integer linear programming problems. For some subclasses of sequencing problems, there are exact algorithms of polynomial complexity, which, for general formulations, can give approximate solutions. However, an assessment of the quality of such solutions shows that the obtained value of the objective function can be almost twice as high as the optimal one. In particular, this applies to algorithms based on the use of the level principle and lexicographic marking of vertices.

In classical sequencing problems, formulations that take into account possible interruptions in the execution of jobs are not considered. However, in many applied problems, it is possible to allow some interruptions in the execution. Hence, the question arises whether their allowance can improve the objective function value? Research on this issue began in the middle of the last century. The first algorithm for solving a problem in a formulation where interruptions were allowed was presented in the work of R. McNaughton in 1959. It has polynomial complexity and is exact for cases where a finite set of identically efficient executors is given, among which a finite number of jobs are distributed, which can be performed in an arbitrary order, while it is necessary to minimize all of the jobs' completion time. Since then, this topic has repeatedly attracted the attention of scientists. Algorithms for solving other subclasses of problems where interruptions of jobs are allowed were published in the works of E. L. Lawler, T. Gonzalez, D. S. Johnson, S. Sani, P. Brooker, and E. G. Coffman. For a number of formulations, as a result of research by J. K. Lenstra, J. D. Ullman, J. Du, R. A. Sitters, and others, it was proved that they are *NP*-complete in the general case,

even taking into account the admissibility of interruptions. It should be noted that in the listed works, the main attention was paid to problems with different characteristics of executors, optimality criteria, and relative durations of work execution, which were mostly considered independent. At the same time, the specifics of technological restrictions on the order of job execution remained poorly studied. Most often, if such cases were considered, they concerned statements in which the corresponding graph of the problem belongs to the class of chains, intrees, or outtrees.

In this paper, we investigate the problems of the parallel sequencing of vertices of digraphs belonging to different subclasses. For them, a comparative solutions analysis was carried out under conditions when interruptions during the execution of the work of the corresponding applied problem are allowed or prohibited, and qualitative estimates of the possible gain from such permission were obtained. This made it possible to identify such subclasses for which the use of interruptions can significantly affect the obtained solution's optimality. Examples of such subclasses are: complete bipartite, parallel-sequential, trees of a special structure, as well as graphs consisting of isolated vertices. For the latter, it was shown that if the execution time of all jobs is the same, then allowing interruptions reduces the value of the objective function by almost two times compared to the case when there is no such permission. The corresponding estimate was obtained and shown that it is achievable.

For cases where technological constraints are modeled by a complete bipartite graph, it was found that the effect of interruptions on the reduction of the objective function value can be considered separately for a subset of vertices of each partition, in addition, theoretical aspects of the dependence of the gain on the initial data were investigated.

In statements where constraints on the order form oriented trees of special subclasses, the research found that the gain from interruptions can differ significantly depending on the jobs' duration.

One generalization of the problem was considered, related to constraints on the number of resources that can be used at fixed points in time. Algorithms were developed to solve it, which are exact under the condition that the corresponding

digraph of the problem consists of a set of isolated vertices or is regular and belongs to the subclass of complete bipartite.

An analysis was conducted of improving approximate solutions possibility obtained using two well-known algorithms of polynomial complexity. It is shown that allowing interruptions of jobs corresponding to graph vertices located at sparsely populated sequencing locations can reduce its length.

The relationship between the statements of the sequencing problem, in which interruptions are allowed, and the one-dimensional packing problem is established. The conditions under which it is possible to reduce the problems in one of the statements to another are formulated.

The scientific novelty of the results obtained in the course of the dissertation research are the following:

- new subclasses of graphs were *identified* for which the permission of interruptions improves the solutions of sequencing problems;
- Mathematical models of sequencing problems were *improved*, which allow taking into account the permission of interruptions in the corresponding applied problems;
- for the *first time*, estimates were introduced for a priori determination of the gain from allowing interruptions for a number of graph subclasses;
- for the *first time*, it was determined which initial data of the problem with allowed interruptions have an impact on the optimality of the solution for one subclass of graphs;
- for the *first time*, the impact of the values of weight coefficients corresponding to the time of work execution on the efficiency of interruptions for one subclass of trees was analyzed;
- the study of the connection between packing and ordering problems was *further developed*;
- the analysis of the influence of allowing interruptions on the optimality of approximate solutions was *further developed*;

- for the *first time*, exact algorithms for generalized problems with interruptions were developed for two subclasses of graphs;
- a software product was developed that implements the proposed algorithms;
- a number of computational experiments were conducted to verify the obtained estimates and the proposed algorithms.

The practical significance of the results obtained is as follows: the obtained theoretical results allow us to predict the feasibility of interruptions in those applied problems where technological processes are modeled by:

- graphs, formed of isolated vertices;
- parallel-sequential graphs;
- complete bipartite graphs;
- special classes of trees.

Such processes, in particular, arise in the service sector, in project management when distributing work between employees, in industry when manufacturing or disposing of products, in construction planning, conducting parallel calculations, etc.

The developed algorithms can be applied to a number of applied problems in which the number of available resources is not a constant value.

The obtained results can be recommended for use in the educational process when training students in the specialties "Systems Analysis" and "Applied Mathematics".

The main results of the dissertation work have been published in 14 scientific works: 5 articles in scientific professional publications of Ukraine of category «B» in physical and mathematical sciences, 9 abstracts of reports in collections of materials of regional and international scientific and practical conferences and seminars.

Keywords: combinatorial optimization problems, scheduling theory, discrete optimization, parallel sequencing problems, graph, regular graphs, directed graphs, complete bipartite graphs, graceful marking, star trees, caterpillars, combinatorial configuration, optimal partitioning, interruption, approximate solutions.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. **Коваленко Є.О., Турчина В.А.** Про покращення наближених розв’язків задачі паралельного упорядкування та аналіз моделі одного її узагальнення. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2025. Т. 2, Вип. 157. С. 35-47. doi: <https://doi.org/10.34185/1562-9945-2-157-2025-04>. Режим доступу до ресурсу: <https://journals.nmetau.edu.ua/index.php/st/article/view/1963/1233> (фахове видання, категорія Б).
2. Турчина В.А., **Коваленко Є.О.** Умови зменшення довжини паралельних упорядкувань вершин спеціальних орграфів при наявності переривань. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2024. Т. 6, Вип. 155. С. 196-207. doi: <https://doi.org/10.34185/1562-9945-6-155-2024-19>. Режим доступу до ресурсу: <https://journals.nmetau.edu.ua/index.php/st/article/view/1927> (фахове видання, категорія Б).
3. Турчина В.А., **Коваленко Є.О.** Дослідження задачі упорядкування з перериваннями для одного підкласу дерев. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2023. Вип. 23. С. 118-125. doi: <https://doi.org/10.15421/322313>. Режим доступу до ресурсу: <https://pm-mm.dp.ua/index.php/pmmm/article/view/382> (фахове видання, категорія Б).
4. Турчина В.А., **Коваленко Є.О.** Вплив початкових даних задачі паралельного упорядкування з перериваннями на оптимальність розв’язку. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2022. Вип. 22. С. 158-167. doi: <https://doi.org/10.15421/322217>. Режим доступу до ресурсу: <https://pm-mm.dp.ua/index.php/pmmm/article/view/382>

mm.dp.ua/index.php/pmmm/article/view/351 (фахове видання, категорія Б).

5. Коваленко Є.О., Турчина В.А. Аналіз впливу структури графів на оптимальність розв'язку задач паралельного упорядкування з перериваннями. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2021. Вип. 21. С. 130-137. doi: <https://doi.org/10.15421/322113>. Режим доступу до ресурсу: <https://pm-mm.dp.ua/index.php/pmmm/article/view/316> (фахове видання, категорія Б).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Турчина В.А., Коваленко Є.О. Доцільність дослідження дозволу переривань в одній задачі теорії розкладів. *Автоматика 2024: Тези XXVII Міжнародної конференції з автоматичного керування, Дніпро, 20-22 листопада 2024 р., м. Дніпро: ДНУ, 2024. С. 200-201. Режим доступу до ресурсу: [http://automatika2024.dp.ua/files/Автоматика-2024%20\(тези%20доповідей\).pdf#page=200](http://automatika2024.dp.ua/files/Автоматика-2024%20(тези%20доповідей).pdf#page=200).*
2. Малієнко О.О., Коваленко Є.О. Дослідження впливу переривань на виникнення аномалій у задачах паралельного упорядкування. *Комбінаторні конфігурації та їхні застосування: Матеріали XXVI Міжнародного науково-практичного семінару, (Кропивницький – Запоріжжя – Київ, 13-15 червня 2024 року) / за ред. Л.Ф. Гуляницького, м. Кропивницький – Запоріжжя – Київ, 2024. С. 103-107. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2024/conf/6.3/CCTA-2024-proc.pdf#page=103.*
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. doi: <https://doi.org/10.36074/logos->

[02.02.2024.044](https://archive.logos-science.com/index.php/conference-proceedings/article/view/1548). Режим доступу до ресурсу: <https://archive.logos-science.com/index.php/conference-proceedings/article/view/1548>.

4. Турчина В.А., **Коваленко Є.О.** Априорна оцінка довжини упорядкувань для спеціальних графів. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2023): Матеріали XXI міжнародної науково-практичної конференції, 22-24 листопада 2023 р., м. Дніпро, 2023.* С. 293-294. Режим доступу до ресурсу: <http://mpzis.dnu.dp.ua/wp-content/uploads/2023/11/mpzis-2023.pdf#page=293>.
5. Турчина В.А., **Коваленко Є.О.** Переривання в задачах упорядкування вершин граціозних дерев. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Запоріжжя – Кропивницький, 14-16 червня 2023 року) / за ред. Г.П. Донця, м. Запоріжжя - Кропивницький, 2023.* С. 214-219. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2023/conf/1.4/Petrenyuk_ISPS-25-proc.pdf#page=214.
6. Турчина В.А., **Коваленко Є.О.** Порівняльний аналіз задач упорядкування та пакування. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2022): Матеріали XX ювілейної міжнародної науково-практичної конференції, 23-25 листопада 2022. м. Дніпро, 2022.* С. 208-209. Режим доступу до ресурсу: <http://mpzis.dnu.dp.ua/wp-content/uploads/2022/12/MPZIS-2022-1.pdf#page=209>.
7. Турчина В.А., **Коваленко Є.О.** Паралельні упорядкування для повних дводольних графів. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Кропивницький – Запоріжжя, 13-14 травня 2022 року). / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2022.* С. 82-86. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2023/conf/1.4/Petrenyuk_ISPS-25-proc.pdf#page=75.

8. **Коваленко Є.О., Турчина В.А.** Аналіз структури графів в задачах паралельного упорядкування з перериваннями. *Комбінаторні конфігурації та їхні застосування: Матеріали XXIII Міжнародного науково-практичного семінару імені А.Я. Петренюка, присвяченого 70-річчю Льотної академії Національного авіаційного університету (Запоріжжя – Кропивницький, 13-15 травня 2021 року)* / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2021. С. 86-90. Режим доступу до ресурсу:
https://www.glau.kr.ua/images/docs/sbornik/materiali_23_mnp_seminaru.pdf#page=86.

9. **Y.Kovalenko, V.Turchina, O.Hurko.** On special classes of scheduling theory problems. *Сучасні науково-технічні дослідження у контексті мовного простору (англійською мовою): Матеріали X Регіональної науково-практичної конференції молодих науковців та студентів, 13 травня 2021 р. м. Дніпро, 2021. С. 28-30.* Режим доступу до ресурсу:
https://www.dnu.dp.ua/docs/ndc/2021/19_Сучасні%20науково-технічні%20дослідження%20у%20контексті%20мовного%20простору.pdf#page=28.

ЗМІСТ

ВСТУП.....	19
Розділ 1. ЗАДАЧІ УПОРЯДКУВАННЯ ЯК СКЛАДОВІ ТЕОРІЇ РОЗКЛАДІВ.....	25
1.1 Застосування апарату теорії розкладів для моделювання прикладних задач	25
1.2 Задачі теорії розкладів, які враховують порядок.....	26
1.3 Роль переривань у задачах теорії розкладів	31
1.4 Постановки задач упорядкування і їх місце в теорії розкладів.....	37
1.5 Ключові питання, що потребують досліджень	43
1.6 Висновки до розділу	45
Розділ 2. ДОСЛІДЖЕННЯ ВПЛИВУ ПЕРЕРИВАНЬ ДЛЯ ПЕВНИХ ПІДКЛАСІВ ГРАФІВ	47
2.1 Упорядкування без наявності часткового порядку	47
2.2 Вплив переривань для повних дводольних графів	51
2.2.1 Переривання у випадках різних кількостей вершин у долях	56
2.2.2 Теоретичні аспекти залежності виграшу від початкових даних задачі	63
2.3 Специфіка паралельно-послідовних графів та її вплив на переривання....	66
2.4 Паралельні упорядкування вершин граціозних дерев	71
2.4.1 Оцінка впливу переривань для дерев-зірок	72
2.4.2 Класи дерев, що зводяться до дерев-зірок	78
2.4.3 Апріорна оцінка довжини упорядкувань	85
2.4.4 Інші підкласи дерев.....	86
2.5 Висновки до розділу	98
Розділ 3. ДЕЯКІ УЗАГАЛЬНЕННЯ ПІДХОДІВ ДО АНАЛІЗУ ЗАДАЧ З ПЕРЕРИВАННЯМИ	101
3.1 Вплив переривань на оптимальність в одній узагальненій задачі упорядкування	101
3.1.1 Узагальнення в задачі без наявності часткового порядку	102
3.1.2 Аналіз впливу переривань для дводольних графів	109
3.2 Наближені алгоритми для узагальнених задач з перериваннями	116
3.3 Зв'язок задач пакування та упорядкування з перериваннями	122

3.4 Висновки до розділу 125

Розділ 4. ПРОГРАМНА РЕАЛІЗАЦІЯ..... 127

4.1 Опис основних програмних модулів..... 127

4.2 Опис інтерфейсу та інструкція користувача 130

ВИСНОВКИ..... 134

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 136

ВСТУП

Актуальність теми. В теорії розкладів за більш ніж 70 років досліджень був розроблений потужний теоретичний апарат, який активно застосовується при розв'язанні прикладних задач, пов'язаних з плануванням у різноманітних галузях промисловості, при будівництві, у сфері послуг, при обробці запитів комунікаційних мереж, у логістиці, у системах, що здійснюють паралельні обчислення, тощо. Деякі з них потребують певного розподілу між скінченною кількістю виконавців скінченної множини робіт при наявних обмеженнях, які стосуються специфіки їх виконання, з метою оптимізації відповідних критеріїв. Задачі теорії розкладів, в яких допускається наявність несуперечливих технологічних обмежень на порядок виконання робіт, класифікуються як задачі паралельного упорядкування. Вони відносяться до оптимізаційних задач на орієнтованих графах, де множині робіт відповідають вершини, а дуги задають технологічні обмеження. Вважається, що в будь-який момент часу конкретна робота може виконуватися не більше, ніж одним виконавцем, і відповідно кожен виконавець може виконувати не більше однієї роботи одночасно.

Дві класичні постановки таких задач полягають в побудові паралельних упорядкувань вершин орграфів при умові, що деякі значення параметрів відомі, а деякі потрібно оптимізувати. Більшість постановок таких задач відносяться до класу *NP*-важких. При їх розв'язанні застосовуються точні алгоритми експоненціальної складності, до яких відносяться підходи основані на методах повного чи спрямованого перебору, зведення до сіткових мереж та задач цілочисельного лінійного програмування. Для деяких підкласів задач упорядкування існують точні алгоритми поліноміальної складності, які для загальних постановок можуть давати наближені розв'язки. Однак оцінка якості таких розв'язків показує, що отримане значення цільової функції може майже вдвічі перевищувати оптимальне. Зокрема це стосується алгоритмів, заснованих на використанні рівневого принципу та лексикографічної розмітки вершин.

У класичних постановках передбачається неперервне виконання усіх робіт, але на практиці виявилось, що дозвіл переривань при виконанні деяких з них в окремих випадках може значно покращити значення відповідного критерію оптимальності. Тому дослідження питань доцільності переривань виконання робіт є актуальним, оскільки відповідні результати дозволяють ефективно організовувати виробничі процеси.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційне дослідження проводилося в рамках тем науково-дослідних робіт Міністерства освіти і науки України «Розробка та реалізація методів оптимального функціонування складних систем» (№ держреєстрації 0122U001466, 2022–2024 рр.) при кафедрі обчислювальної математики та математичної кібернетики Дніпровського національного університету імені Олеся Гончара.

Мета і завдання дослідження. *Мета дисертаційної роботи* полягає у отриманні нових теоретичних результатів, що стосуються моделей та методів розв'язання задач паралельного упорядкування, які враховують допустимість переривань.

Завданнями дослідження були:

- одержання нових результатів для задач оптимального упорядкування вершин, що стосуються аналізу впливу переривань на оптимальність;
- дослідження можливого покращення значення цільової функції залежно від структури графа;
- визначення оцінок покращення для тих підкласів, для яких воно є суттєвим;
- розробка нових точних алгоритмів розв'язання задач упорядкування, в яких виконання робіт дозволено переривати;
- розробка програмного продукту, що реалізує відомі та нові запропоновані алгоритми;

- перевірка отриманих теоретичних результатів та розроблених алгоритмів шляхом проведення обчислювальних експериментів.

Об'єктом дослідження виступають математичні моделі прикладних задач, які формуються у вигляді задач паралельного упорядкування вершин ографу.

Предметом дослідження є алгоритми та методи аналізу впливу переривань на оптимальність розв'язків у задачах паралельного упорядкування.

При дослідженні застосовувалися *методи* теорії розкладів, теорії оптимальних упорядкувань, дискретної та комбінаторної оптимізації.

Наукова новизна одержаних результатів полягає в наступному:

- виявлено нові підкласи графів, для яких дозвіл переривань покращує розв'язки задач упорядкування;
- узагальнено основні поняття для побудови математичних моделей, які дозволяють враховувати дозвіл на переривання робіт у відповідних прикладних задачах;
- вперше введені оцінки для апріорного визначення виграшу від дозволу переривань для ряду підкласів графів;
- вперше визначено, які початкові дані задачі при дозволених перериваннях мають вплив на оптимальність розв'язку для одного підкласу графів;
- вперше проаналізовано вплив значень вагових коефіцієнтів, що відповідають часу виконання робіт, на ефективність переривань для одного підкласу дерев;
- дістало подальшого розвитку дослідження зв'язку задач пакування та упорядкування;
- дістав подальшого розвитку аналіз впливу дозволу переривань на оптимальність наближених розв'язків;
- вперше розроблено точні алгоритми розв'язання для узагальнених задач з перериваннями для двох підкласів графів;
- розроблено програмний продукт, який реалізує запропоновані алгоритми;
- проведено ряд обчислювальних експериментів з метою верифікації отриманих оцінок та запропонованих алгоритмів.

Практичне значення одержаних результатів. Проведені дослідження за темою дисертаційної роботи стосувалися аналізу можливого впливу дозволу переривань на оптимальність розв'язків задач паралельного упорядкування. Отримані теоретичні результати дозволяють прогнозувати доцільність переривань в тих прикладних задачах, де технологічні процеси моделюються графами таких підкласів: графами, які складаються з ізольованих вершин; паралельно-послідовними; повними дводольними; спеціальними підкласами дерев.

Такі процеси зокрема виникають у сфері обслуговування, в управлінні проєктами при розподілі робіт між працівниками, у промисловості при виготовленні чи утилізації виробів, у плануванні будівництва, проведенні паралельних обчислень тощо.

Розроблено алгоритми для задач, в яких кількість доступних ресурсів не є сталою величиною та обґрунтована доцільність їх застосування до ряду прикладних сфер.

Отримані результати можуть бути рекомендовані до використання в навчальному процесі при підготовці студентів спеціальності «Системний аналіз» та «Прикладна математика».

Особистий внесок здобувача. Основні результати, отримані в ході дисертаційного дослідження, представлені у 14 наукових працях [1-14]. Усі результати, що виносяться на захист, отримані автором особисто. У статті [1] здобувачеві належить аналіз можливості покращення наближених розв'язків, алгоритм розв'язання задачі, доведення тверджень; співавторці Турчиній В.А. належить формулювання умов для застосування розроблених алгоритмів, та тверджень що стосуються точності цих алгоритмів. У статті [2] здобувачеві належить аналіз впливу переривань для підкласу одностантних дерев, перерахування способів побудови графа задачі, доведення теореми; співавторці Турчиній В.А. належить визначення можливої структури орієнтованих гусениць та формулювання теореми. У статті [3] здобувачеві належить оцінка можливого зменшення значення цільової функції за рахунок переривань, проведення

обчислювальних експериментів; співавторці Турчиній В.А. належить порівняльний аналіз обчислювальних експериментів із теоретичними результатами. У статті [4] здобувачеві належить аналіз впливу дозволу переривань на оптимальність розв'язків для повних дводольних графів, доведення твердження, оцінки можливого покращення значення цільової функції; співавторці Турчиній В.А. належить формулювання твердження та визначення випадків, за яких переривання впливають на оптимальність розв'язків. У статті [5] здобувачеві належить визначення оцінок можливого виграшу від переривань та їх граничних значень; співавторці Турчиній В.А. належить визначення умов, за яких дозвіл переривань зменшує значення цільової функції. У публікації [6] здобувачеві належить аналіз прикладних задач, що моделюються графами розглянутих підкласів; співавторці Турчиній В.А. належить визначення підкласів графів, для яких дозвіл переривань є доцільним. У публікації [7] здобувачеві належить формулювання умов, за яких можливо уникнути виникнення аномалій, у вигляді твердження; співавторці Малієнко О.О. належить доведення твердження. У публікації [8] здобувачеві належать сформульовані означення, розроблені алгоритми, доведення тверджень; співавторці Турчиній В.А. належить визначення умов в означенні узагальненого упорядкування та формулювання тверджень. У публікації [9] здобувачеві належить доведення сформульованих тверджень; співавторці Турчиній В.А. належить формулювання тверджень. У публікації [10] здобувачеві належить узагальнення залежності можливого виграшу від початкових даних задачі для випадку комбінованої зірки та визначення граничних значень оцінок; співавторці Турчиній В.А. належить аналіз залежності виграшу для вхідних та вихідних зірок. У публікації [11] здобувачеві належить формулювання умов, за яких можливе зведення задачі упорядкування до задачі пакування; співавторці Турчиній В.А. належить формулювання обмежень задач пакування, до яких можуть бути зведені задачі упорядкування у випадках дозволених та заборонених переривань. У публікації [12] здобувачеві належить аналіз залежності виграшу від початкових даних задачі, визначення граничних значень

оцінок; співавторці Турчиній В.А. належить формулювання оцінки виграшу для досліджуваного підкласу графів. У публікації [13] здобувачеві належить аналіз можливості зменшення значення цільової функції, гранична оцінка для випадку незалежних робіт; співавторці Турчиній В.А. належить визначення графа, для якого верхня гранична оцінка виграшу є досяжною. У публікації [14] здобувачеві належить формулювання задачі упорядкування, яка враховує можливість переривань при виконанні робіт; співавторці Турчиній В.А. належить класифікація задач упорядкування; співавторці Гурко О.В. належить визначення прикладних сфер застосування задач упорядкування.

Апробація результатів дисертації. Основні положення та результати дисертаційної роботи доповідлися та обговорювалися на семінарі при науковій раді НАН України з проблеми «Кібернетика», який функціонує при Дніпровському національному університеті імені Олеся Гончара; на міжнародних наукових конференціях «Математичне та програмне забезпечення інтелектуальних систем» (м. Дніпро, 2022, 2023 рр.), «Комбінаторні конфігурації та їхні застосування» (м. Київ – Запоріжжя – Кропивницький, 2021, 2022, 2023, 2024 рр.); «Автоматика» (Дніпро, 2024 р.); «Theoretical and empirical scientific research: concept and trends» (Oxford, UK, 2024 p.).

Публікації. Основні результати дисертаційної роботи опубліковано в 14 наукових працях: 5 статей ([1-5]) у наукових фахових виданнях України категорії «Б» з фізико-математичних наук, 9 тез доповідей у збірниках матеріалів міжнародних конференцій та семінарів, регіональній науково-практичній конференції ([6-14]).

Структура та обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, переліку використаних джерел, що містить 113 найменувань на 8 сторінках та одного додатку на 4 сторінках. Загальний обсяг дисертації — 147 сторінок, обсяг основного тексту — 135 сторінок. Робота містить 68 рисунків та 1 таблицю.

Розділ 1. ЗАДАЧІ УПОРЯДКУВАННЯ ЯК СКЛАДОВІ ТЕОРІЇ РОЗКЛАДІВ

1.1 Застосування апарату теорії розкладів для моделювання прикладних задач

Теорія розкладів як наука виникла в середині ХХ ст. в результаті стрімкого технологічного розвитку та ускладнення виробничих процесів і, як наслідок, загостренню потреби у раціональному використанні часу. Вона активно розвивалася впродовж подальших десятиліть.

В теорії розкладів вивчаються такі задачі, де для заданої скінченної множини робіт, скінченної кількості виконавців та певних умов щодо специфіки виконання робіт необхідно знайти такий розподіл робіт між виконавцями, щоб цей розподіл, який називається розкладом, був найкращим за визначеним умовами задачі критерієм, чи критеріями. Природа самих робіт та виконавців визначається конкретними прикладними задачами. Так, наприклад, роботами в задачах можуть виступати окремі операції, деталі, програмні продукти та ін. В той час як виконавцями можуть бути люди, станки, бригади, процесори тощо.

Перші прикладні задачі теорії розкладів виникли у промисловості при плануванні виробництва. Застосуванню теоретичного апарату теорії розкладів для розв'язання практичних задач цього напрямку присвячено роботи [15-22]. Із часом сфера застосувань теорії розкладів розширилася на інші види людської діяльності. Так із побудовою розкладів пов'язані задачі розробки медичних препаратів [23] та обробки запитів клієнтів фармацевтичними компаніями [24]; складання графіків робочих годин [25-27] та розподілення робіт між робітниками при управлінні проєктами [28-30]; обслуговування клієнтів [31] та інші.

У подальшому розвиток апаратного забезпечення сприяв появі обчислювальних систем із багатопроцесорною та багатоядерною архітектурою, в яких дедалі більше операцій могли виконуватися паралельно. Це дало поштовх дослідженню питання оптимального використання процесорних ресурсів, яке із часом стає дедалі більш актуальним [32-38]. При цьому розвиток підходів до

організації паралельних обчислень дозволяє розв'язувати задачі все більших порядків складності, витрачаючи прийнятний для практичного застосування час [39-43].

Розвиток засобів комунікації та обміну даними сприяв дослідженням прикладних задач при проектуванні та підтримці бездротових Bluetooth [44,45] а згодом і Wi-Fi мереж [46,47].

Деякі задачі виробництва та сфери послуг потребують комплексного підходу, який передбачає врахування додаткових факторів пов'язаних із транспортуванням або ресурсними обмеженнями. Математичні моделі таких прикладних задач поєднують у собі обмеження як задач теорії розкладів, так і інших відомих задач дискретної оптимізації, таких як задачі транспортування [48,49] та пакування [50,51].

1.2 Задачі теорії розкладів, які враховують порядок

Множини робіт та виконавців у задачах теорії розкладів складають так звану обслуговуючу систему, яка в залежності від конкретних постановок може бути або одностадійною або багатостадійною. Одностадійними є системи, де кожна з робіт може бути повністю виконана будь-яким з виконавців, в той час як багатостадійні системи включають роботи, які поділяються на скінченні множини стадій, кожній з яких ставиться у відповідність така підмножина виконавців, що будь-який з них (лише один одночасно) може виконати цю стадію роботи в повному обсязі [52].

В задачах теорії розкладів для одностадійних систем задається час виконання кожної роботи та можуть вводитися додаткові умови:

- директивні терміни, коли необхідно або бажано, щоб роботу було завершено;
- пріоритетність робіт;
- моменти часу, коли робота готова до виконання;
- інші умови.

Якщо на порядок виконання робіт накладаються несуперечливі технологічні обмеження щодо порядку їх виконання, то такі задачі можуть бути сформульовані як оптимізаційні задачі на графах, які відносяться до класу задач паралельного упорядкування [53]. Найбільш дослідженими з таких задач є дві:

- 1) Задача зі скінченною кількістю робіт та виконавців, технологічні обмеження утворюють граф, який є орієнтованим вхідним деревом або лісом, необхідно мінімізувати час завершення виконання робіт.
- 2) Задача з довільною кількістю робіт та двома виконавцями, технологічні обмеження утворюють граф довільного вигляду, необхідно мінімізувати час виконання робіт.

Для розв'язання задачі в цих постановках існують точні алгоритми поліноміальної складності: алгоритм, заснований на рівневому принципі, для першої [53], та алгоритми, засновані на побудові максимального паросполучення [54] і лексикографічної розмітки (для графів без транзитивних дуг) [55] для другої.

Якщо в задачах теорії розкладів природа робіт та виконавців для пошуку розв'язків задачі значення не має, то їх можна пронумерувати натуральними числами.

Задається деяка скінченна множина робіт:

$$A = \{1, 2, \dots, n\} \quad (1.1)$$

та скінченна множина виконавців:

$$B = \{1, 2, \dots, m\}. \quad (1.2)$$

Також в загальному випадку заданий час

$$t_{ij} > 0, i \in A, 1 \leq j \leq m \quad (1.3)$$

на виконання роботи i виконавцем j . Цей час може залежати від ефективності кожного фіксованого виконавця j , тоді виконується:

$$t_{ij} = \frac{t_i}{\tau_j}, i = \overline{1, n}, j = \overline{1, m} \quad (1.4)$$

де t_i — час виконання роботи i виконавцем з ефективністю 1, а τ_j — ефективність j -ого виконавця. Якщо $\tau_j = \tau$ для $j = \overline{1, m}$, то виконавці вважаються ідентичними.

Якщо за умовою задачі робота i' може виконуватися тільки після завершення виконання роботи i , то на множині робіт A задається відношення строгого порядку, що позначається \rightarrow , виходячи з якого виконання роботи i повинно завершитися до початку виконання роботи i' , якщо $i \rightarrow i'$.

Означення розкладу введемо наступним чином. Процес виконання робіт може бути описаний за допомогою сукупності $s = \{s_1(t), s_2(t), \dots, s_m(t)\}$ кусково-постійних неперервних зліва функцій $s_j = s_j(t)$ ($j = \overline{1, m}$), які задані на $0 \leq t < \infty$ і приймають значення $0, 1, \dots, n$. Якщо $s_j(t') = i \neq 0$, то в момент часу t' виконавець j виконує роботу i , якщо $s_j(t') = 0$, то в момент часу t' виконавець j неактивний. Окрім того очевидно, що мають виконуватись умови, за якими робота i не може одночасно виконуватись більш ніж одним виконавцем. Розкладом називається така сукупність функцій s , яка задовольняє переліченим умовам [56].

Оскільки задачі теорії розкладів, як правило, відносяться до оптимізаційних, то їх характеризує цільова функція, що потребує мінімізації (рідше максимізації) на множині допустимих розкладів [57]. Допустимим називається розклад, який не порушує відношення часткового порядку, задане на множині A .

Для класифікації задач теорії розкладів застосовують трикомпонентний запис виду $\alpha|\beta|\gamma$, запропонований у [58] де α — характеристики виконавців (машин, процесорів), β — характеристики робіт, γ — критерій оптимальності задачі. В задачах, де дозволяється паралельне виконання робіт виконавцями, при характеристиці виконавців використовують позначки P , Q та R :

- P відповідає випадку ідентичних виконавців, тобто тих, які з однаковою ефективністю виконують будь-яку роботу;
- Q використовують якщо виконавці є уніфікованими, тобто не всі з них мають однакоvu ефективність;

- R застосовується у випадку, коли ефективність залежить від конкретної роботи, що виконується.

У випадках, коли потрібно вказати і кількість виконавців, що можуть працювати паралельно, вказується також натуральне число, що відповідає даному значенню.

Компонента β є послідовністю максимум 6 значень відповідно $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ та β_6 . Зупинимося на тих, які використовуються в даній роботі.

- β_1 визначає, чи допускаються в задачі переривання при виконанні робіт, відповідно має позначення *pmtn* якщо це так, якщо ні — то не вказується.
- β_2 задається позначенням *pres* у разі, коли на множині робіт задано відношення часткового порядку. Тоді задачу теорії розкладів можна сформулювати як оптимізаційну задачу на орграфі. Якщо структура орграфа визначена, то вона ідентифікується на місці β_2 . Так, наприклад, вхідному (вихідному) дереву відповідають *intree* (*outtree*), а довільному — *tree*. Аналогічно позначення *chain* відповідає графу, що має структуру ланцюга, та *sp-graph* — випадку паралельно-послідовного графа.
- $\beta_4 = t_{ij}$ визначає час виконання робіт i виконавцем j .

Якщо цільова функція враховує штрафи різного роду (штрафні функції), то прикладами таких функцій можуть бути:

- C_i — момент завершення роботи, момент часу, коли була виконана робота під номером $i, i \in A$;
- L_i — зміщення в часі, яке визначається різницею $C_i - d_i, i \in A$;
- T_i — запізнення, що визначається як $\max\{0, C_i - d_i\}, i \in A$;
- E_i — випередження, що визначається як $\max\{0, d_i - C_i\}, i \in A$;

де d_i — директивний термін завершення роботи під номером i , тобто такий момент часу, до якого цю роботу бажано завершити.

Деякі критерії оптимальності в задачах теорії розкладів:

1) мінімаксний критерій — при якому необхідно мінімізувати цільову функцію, яка являє собою максимум значень штрафів робіт. Наприклад:

- $C_{\max} \rightarrow \min$ — задача швидкодії, для розв'язку якої проводиться мінімізація часу, коли будуть виконані усі роботи;
- $L_{\max} \rightarrow \min$ — критерій мінімізації максимального зміщення часу

$$L_{\max} = \max_{i \in A} L_i ;$$

2) сумарний критерій — де цільова функція має вигляд суми штрафів робіт.

Приклади таких критеріїв:

- $\sum_{i \in A} C_i \rightarrow \min$ — критерій мінімізації сумарного часу виконання усіх робіт;
- $\sum_{i \in A} T_i \rightarrow \min$ — критерій мінімізації сумарного часу запізнення виконання робіт;

3) багатокритеріальні задачі, в яких цільова функція поєднує у собі кілька критеріїв.

Для деяких постановок задач теорії розкладів дослідниками було встановлено їх клас складності та розроблено відповідні алгоритми знаходження розв'язків. Відомості про частину таких задач наведено в табл. 1 [59].

Таблиця 1.1. Деякі відомості про окремі класи задач

Позначення	Клас складності задачі (P/NP)	Складність алгоритму	Джерело
$P t_i = t; tree C_{\max}$	P	$O(n)$	[53]
$P2 t_i = t; prec C_{\max}$	P	$O(n^{\log^7})$	[60]
$Q2 t_i = t; chains C_{\max}$	P	$O(l_{ch})$	[61]
$P t_i = t; intree L_{\max}$	P	$O(n)$	[62,63]
$P t_i = t; outtree \sum C_i$	P	$O(n \log n)$	[53]
$Pm t_i = t; tree \sum C_i$	P	$O(n^m)$	[64]

Таблиця 1.1. (продовження)

$P2/t_i = t; prec/\sum C_i$	P	$O(n^{\log^7})$	[55]
$R/\sum C_i$	P	$O(mn^3)$	[65]
$P2/\ C_{max}$	NP	—	[66]
$P/\ C_{max}$	NP	—	[67]
$P/t_i = 1; prec/C_{max}$	NP	—	[68]
$P2/chains/C_{max}$	NP	—	[69]
$Q/t_i = 1; chains/C_{max}$	NP	—	[70]
$P/t_i = 1; outtree/L_{max}$	NP	—	[62]
$P/t_i = 1; prec/\sum C_i$	NP	—	[71]
$P2/chains/\sum C_i$	NP	—	[69]
$P/pmtn/C_{max}$	P	$O(n)$	[72]
$P/outtree; pmtn/C_{max}$	P	$O(n^2)$	[73]
$P/tree; pmtn/C_{max}$	P	$O(n \log m)$	[74]
$Q/chains; pmtn/C_{max}$	P	$O(n + m \log n)$	[75]
$P/intree; pmtn/L_{max}$	P	$O(n^2)$	[76]
$Q/pmtn/L_{max}$	P	$O(n \log n + mn)$	[77]
$P/t_i = t; outtree; pmtn/\sum C_i$	P	$O(n^2)$	[78]
$P2/t_i = t; prec; pmtn/\sum C_i$	P	$O(n^2)$	[79]
$Q/pmtn/\sum C_i$	P	$O(n \log n + mn)$	[77]
$P/t_i = 1; prec; pmtn/C_{max}$	NP	—	[56]
$P2/chains; pmtn/\sum C_i$	NP	—	[69]
$R/pmtn/\sum C_i$	NP	—	[80]

1.3 Роль переривань у задачах теорії розкладів

У теорії розкладів виділяють окремі класи задач, в яких допускаються переривання, або забороняються. Перериваннями будемо називати ті випадки, коли виконання роботи призупиняється до її завершення. Після цього виконання

продовжується цим чи іншим виконавцем або до наступного переривання, або до повного завершення роботи. В даній роботі увага приділяється саме задачам, в яких переривання дозволені. Вперше таку задачу для ідентичних виконавців було розглянуто у [72]. Результатам досліджень можливого впливу переривань на розв'язок було присвячено багато робіт.

В [56] пропонується алгоритм знаходження оптимального розкладу для систем із незалежними завданнями та довільною тривалістю. В основі цього алгоритму лежить ідея побудови розкладу з мінімальними простоями у системі, тобто такого, яке міститиме якнайменше відрізків часу, коли один чи більше виконавців не зайняті жодною роботою. Довжина розкладу визначається як:

$$l = \max \left\{ \frac{\sum_{i=1}^n t_i}{m}, \max_i \{t_i\} \right\}. \quad (1.5)$$

Тобто довжина дорівнює сумарній тривалості усіх робіт, поділеній на кількість виконавців, або тривалості найдовшої роботи, якщо вона перевищує це значення. Алгоритм передбачає послідовне призначення робіт кожному виконавцеві доти, доки не буде досягнута величина l , значення якої обчислено за (1.5). Якщо додавання чергової роботи перевищує це значення, то процес її виконання розділяється на дві частини. Одна з них розміщується так, щоб завершуватися при досягненні величини l , інша частина призначається наступному виконавцеві і починається з моменту часу 0. Проілюструємо роботу алгоритму на прикладі.

Приклад 1.1. Нехай дано множину незалежних робіт $A = \{1, 2, 3, 4, 5\}$ з відповідними тривалостями $t_1 = 5, t_2 = 3, t_3 = 3, t_4 = 3, t_5 = 4$. Множина виконавців $B = \{1, 2, 3\}$. Розподілення робіт наведено на рис. 1.1.

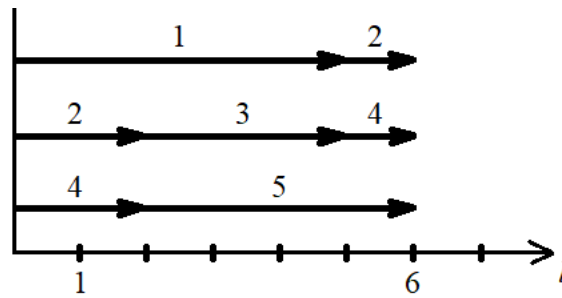


Рис. 1.1. Розподіл незалежних робіт за алгоритмом

Окрім цього автор наводить доведення, що побудований за наведеним алгоритмом розклад є оптимальним.

Також у [56] увага приділяється оцінці кількості переривань. Оскільки кожне переривання потребує додаткових витрат пам'яті, то корисно з'ясувати максимальну можливу кількість переривань при побудові оптимального розкладу. Верхня оцінка кількості можливих переривань визначається наступною теоремою.

Теорема 1.1. Нехай s^* — оптимальний розклад, складений за запропонованим алгоритмом для системи незалежних робіт, що виконуються на m ідентичних процесорах. Тоді s^* має не більше $m-1$ переривань. Більше того, оцінка $m-1$ для числа переривань не може бути покращеною в тому сенсі, що існує така система робіт, для якої будь-який розклад мінімальної довжини містить не менше $m-1$ переривань.

Так наприклад, якщо маємо систему із $m+1$ робіт, кожна з яких триває m одиниць часу, то кожна робота буде виконуватись не більш ніж на двох проміжках часу, тобто перериватиметься не більше одного разу. Зовсім перериватися не будуть дві роботи: ті, які призначаються першою та останньою. Таким чином оптимальний розклад буде дійсно мати не більше (а в даному випадку рівно) $m-1$ переривань, як показано на рис. 1.2.

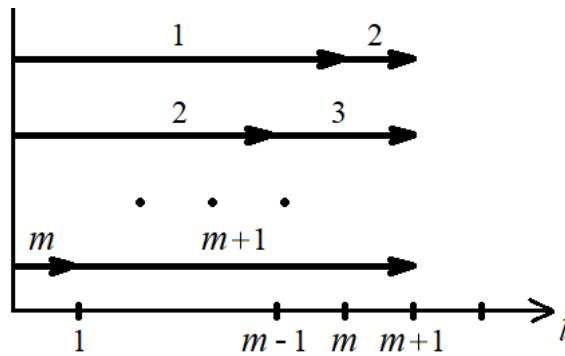


Рис. 1.2. Розподіл $m+1$ незалежних робіт із $m-1$ перериваннями

Автори роботи [79] наводять алгоритм поліноміальної складності для розв'язання задачі відшукування оптимального розкладу на двох ідентичних паралельних процесорах, при побудові якого дозволені переривання. Мінімізуються наступні параметри розкладу:

- 1) загальний час виконання всіх робіт;
- 2) сума моментів завершення виконання кожної з робіт, тобто усіх моментів часу, коли виконання роботи припинено, але не перервано.

Описується система, що містить множину з n робіт, причому тривалість виконання кожної роботи рівна деякій фіксованій одиниці часу (англ. unit), звідси походить назва *UET*-система (англ. unit-execution-time system) — тобто така, що складається лише з робіт певної тривалості. Окрім того на цій множині задано відношення часткового порядку. Дозволяється переривання виконання роботи із можливістю продовжити її пізніше на тому ж, або іншому процесорі. За основу береться відомий алгоритм поліноміальної складності для двох процесорів, що базується на лексикографічній розмітці вершин відповідного графа системи, де роботам відповідають вершини орієнтованого графа [55]. В процесі присвоєння міток на кожній ітерації формується по одній підмножині вершин, яким послідовно присвоюються мітки. Таким чином множина всіх робіт розбивається на $k \leq n$ неперетинних підмножин A_k, A_{k-1}, \dots, A_0 таких, що кожна робота з A_i обов'язково має бути виконана раніше за будь-яку з робіт підмножини A_{i-1} , $i = 1, 2, \dots, k$. Це так звані послідовні множини (англ. follow-sets). Спочатку

будується оптимальне упорядкування без переривань. Після цього для кожної послідовної множини розглядаються 3 випадки:

- 1) послідовна множина містить лише одну роботу;
- 2) послідовна множина містить дві роботи;
- 3) послідовна множина містить більше двох робіт.

У першому випадку робота з множини виконується на першому процесорі до завершення, після чого розглядається наступна послідовна множина. У другому випадку дві роботи виконуються паралельно на обох процесорах. У третьому випадку аналізується, чи парна кількість робіт. Якщо кількість парна, то на процесорах роботи виконуються попарно аналогічно другому випадку. У разі непарної кількості для останніх трьох робіт проводиться процедура скорочення (англ. contraction), принцип якої проілюстровано на рис. 1.3. Під позначенням $S[\overline{i, j}]$ мається на увазі множина всіх елементів упорядкування, що знаходяться на місцях з i по j , $i < j \leq n$. Таким чином сукупність робіт $S[\overline{i, j}]$, $j - i = 3$ тривалістю 2 одиниці часу зводиться до $3/2$ одиниць. При цьому сума моментів завершення робіт не збільшується. Для випадку без переривань маємо

$$1+1+2=4 \text{ одиниці часу,}$$

як і для випадку з перериваннями

$$1+3/2+3/2=4 \text{ одиниці часу.}$$

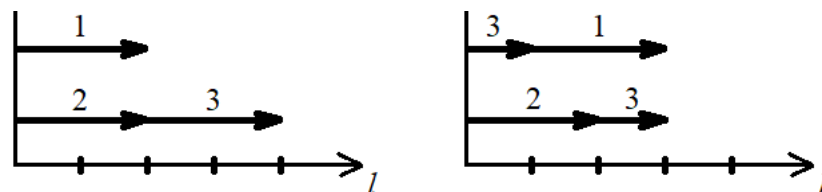


Рис. 1.3. Процедура скорочення

В статті доводиться, що побудований таким чином розклад є, як назвав його автор, ідеальним, тобто алгоритм мінімізує перший критерій та не погіршує другий, при цьому складність алгоритму оцінюється як $O(n^2)$.

У статті [81] основна увага приділяється побудові розкладів у випадках непов'язаних процесорів, зокрема розв'язанню задач виду $R3|pmtn|C_{max}$ та $R|pmtn|C_{max}$. Додатковим обмеженням, введеним для цих задач є відсутність однопроцесорних переривань, або розривів (англ. splits). Цей термін був запропонований у [82] для позначення ситуацій, коли після переривання виконання роботи через певний проміжок часу продовжується на тому ж процесорі. Автори зауважують, що наявність такого обмеження обумовлена зокрема тим, що такі розриви зазвичай у прикладних задачах призводять до необґрунтованих часових або матеріальних витрат.

Для задачі $R3|pmtn|C_{max}$ авторами [81] було доведено, що вона є NP -важкою навіть за досить специфічних початкових умов, коли більша частина робіт такі, які потребують значного часу для виконання на усіх процесорах, крім одного. Конкретно розглядалася наступна постановка. Дано k робіт таких, що тривалість виконання j -ої становить $\frac{a_j}{b}$ на першому процесорі, причому $\sum_{j=1}^k a_j = 2b$. Передбачається, що існує підмножина $S \subset \{1, \dots, k\}$ така, що $\sum_{j \in S} a_j = b$. Також вводяться три додаткові роботи:

- A потребує по 3 одиниці часу на будь-якому процесорі;
- B потребує 2 одиниці часу на другому процесорі;
- C потребує 2 одиниці часу на третьому процесорі.

Було доведено, що одним з оптимальних розв'язків даної задачі є розклад довжини 3, де робота A займає на кожному процесорі по одиниці місця в порядку 2, 1, 3, робота B розміщена на проміжку $[1,3)$ другого процесора, а робота C — на проміжку $[0,2)$ третього. Роботи множини $\{1, \dots, k\}$, які належать підмножині S , займають на першому процесорі проміжок $[0,1)$, тоді як решта — проміжок $[2,3)$. Альтернативним розв'язком є розклад, в якому порядок виконання робіт на другому та третьому процесорах зворотній.

Побудова оптимального розкладу для задачі $R|pmtn|C_{max}$ базується на подібному підході та розглядається як узагальнення задачі $R3|pmtn|C_{max}$.

Серед питань, які потребують подальшого розгляду автори зазначають з'ясування, чи є задача із фіксованою кількістю непов'язаних процесорів NP -важкою у сильному сенсі, або розв'язною за псевдополіноміальний час.

1.4 Постановки задач упорядкування і їх місце в теорії розкладів

В залежності від типу шуканого розв'язку, способу закріплення робіт за виконавцями, узгодженістю часу та порядку їх виконання виділяються такі основні класи задач теорії розкладів [83]:

- 1) задачі упорядкування, де необхідно визначити оптимальний спосіб призначення послідовностей робіт виконавцям;
- 2) задачі узгодження, сфокусовані на виборі часу робіт при заданому розподілі;
- 3) задачі розподілу, де необхідно вказати найкраще розподілення робіт між виконавцями.

Багато досліджень теорії розкладів присвячені задачам упорядкування, які на практиці часто мають обмеження на порядок виконання робіт через технологічні умови, що впливають із природи конкретної прикладної задачі. Ці обмеження задаються відношеннями часткового порядку, що відображають умови передування при виконанні робіт. Відношення порядку позначається \rightarrow та вводиться аналогічно тому, як було введено в пункті 1.1.

Класичний варіант постановки задачі паралельного упорядкування полягає у тому, що для заданих скінченних множин робіт та виконавців, залежно від виду задачі, необхідно задовольнити одну з наступних вимог:

- визначити мінімальний проміжок часу, за який усі роботи може бути виконано без порушень у технологічному процесі;

- визначити мінімальну кількість виконавців, які не порушуючи технологічних вимог на порядок, можуть виконати усі необхідні роботи у заданий термін.

При цьому тривалість кожної роботи однакова, а на порядок їх виконання може бути накладено певні технологічні обмеження. Окрім того, усі виконавці є ідентичними у тому сенсі, як було визначено в пункті 1.1.

Дані задачі, як і більшість задач теорії розкладів є *NP*-важкими, а це означає, що в загальному випадку алгоритми пошуку оптимальних розв'язків будуть мати експоненціальну складність. Один із підходів при формалізації цієї задачі полягає в представленні її як оптимізаційної задачі на графах [56].

Нехай існує певна скінченна множина $V: |V| = n$, на якій задано відношення строгого порядку, і де n — кількість робіт.

Означення 1.1. Паралельним упорядкуванням S елементів множини V будемо називати таке їх розміщення на не більш, ніж n місцях, при якому кожен елемент може бути розташований лише на одному з них. Окрім цього, для будь-якої пари елементів $a \rightarrow b$, $a, b \in V$ порядковий номер місця, на якому знаходиться елемент a менший за відповідний номер місця розташування елемента b .

Означення 1.2. Кількість місць, на яких розміщені елементи називають довжиною упорядкування, яку позначають $l(S)$.

Позначимо через n_k кількість елементів, що розташовані в упорядкуванні на k -ому місці.

Означення 1.3. Шириною паралельного упорядкування $h(S)$ називається величина $\max_k n_k, k = \overline{1, l}$.

Враховуючи сформульовану постановку задачі упорядкування, доцільним буде подання робіт у вигляді множини вершин V орієнтованого графа $G = (V, U)$, а відповідно технологічних обмежень у вигляді множини дуг U . Іншими словами, кожній роботі ставиться у відповідність одна з вершин графа, а дуга

$(i, j) \in U$ тільки у тому випадку, якщо за технологічними обмеженнями робота i безпосередньо передуює роботі j , де $i, j \in V$.

В результаті наведених міркувань та означень 1.1-1.3 маємо дві постановки задач паралельного упорядкування:

- за заданими орграфом G та шириною упорядкування $h(S)$ мінімізувати довжину $l(S) \rightarrow \min_s$;
- за заданими орграфом G та довжиною упорядкування $l(S)$ мінімізувати ширину $h(S) \rightarrow \min_s$.

Варто також зауважити, що в силу природи технологічних обмежень наведений граф G не містить петель, циклів та кратних дуг. Окрім того очевидно, що в поставлених задачах застосовується мінімакський критерій оптимальності, про що свідчать означення 1.2 та 1.3. За загальноприйнятою термінологією задача у першій постановці дістала назву прямої, а в другій — відповідно оберненої. В даній роботі основна увага приділяється прямим задачам упорядкування.

Для скорочення запису задач упорядкування будемо використовувати запис виду $S(A, B, C)$, де A — граф, який відповідає умові задачі, B — задані параметри, C — цільова функція. Можливі варіанти:

$$A = \begin{cases} G \\ T \\ K_{i,j} \\ G_{\text{пп}} \end{cases}, \quad B = \begin{cases} h(S) \\ l(S) \\ h_i(S) \end{cases}, \quad C = \begin{cases} h(S) \\ l(S) \end{cases},$$

де G — довільний ациклічний орієнтований граф без петель, T — кореневе дерево або ліс, $K_{i,j}$ — дводольний граф із кількістю вершин i та j в долях, $G_{\text{пп}}$ — паралельно-послідовний граф.

Тоді перша задача матиме запис $S(G, h(S), l(S))$, а друга відповідно — $S(G, l(S), h(S))$. Також можна використовувати простіший запис: $S(G, h, l)$ та $S(G, l, h)$ відповідно.

Якщо розглянути в якості прикладу пряму задачу паралельного упорядкування із графом G , заданим на рис. 1.4, де $h(S) = n$, то легко бачити, що довжина мінімального паралельного упорядкування $l(S)$ в такому випадку буде рівна максимальному шляху в G , за умови, що рахування йде по вершинах. Стає очевидно, що будь-яке переміщення вершин, які лежать на шляхах найбільшої довжини, призводить лише до зростання $l(S)$. Такі шляхи прийнято називати критичними. Якщо граф не містить дуг, а лише ізольовані вершини, що відповідають роботам однакової тривалості, то вважається, що критичний шлях має одиничну довжину.

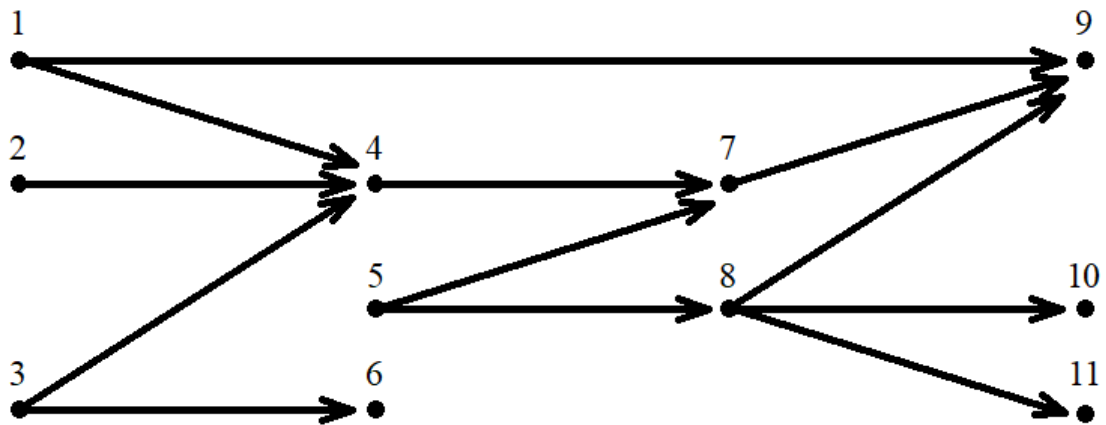


Рис. 1.4. Орієнтований граф із критичним шляхом довжини 4

Ось деякі можливі паралельні упорядкування вершин графа:

$$1) \ S_1 = \begin{pmatrix} 1 & 4 & 9 \\ 2 & 5 & 7 & 10 \\ 3 & 6 & 8 & 11 \end{pmatrix}, \text{ для якого } h(S_1) = 3, l(S_1) = 4;$$

$$2) \ S_2 = \begin{pmatrix} 1 & 9 \\ 2 & 4 & 7 & 10 \\ 3 & 8 & 11 \\ 5 & 6 \end{pmatrix}, \text{ для якого } h(S_2) = 4, l(S_2) = 4;$$

$$3) \ S_3 = \begin{pmatrix} 2 & 1 & 4 & 7 & 9 \\ 3 & 6 & 5 & 8 & 11 & 10 \end{pmatrix}, \text{ для якого } h(S_3) = 2, l(S_3) = 6.$$

Залежність між відповідними довжинами критичного шляху та упорядкування, як правило, корисно враховувати при відшуванні оптимального розв'язку, через це важливим є питання визначення множини критичних шляхів графа. Для цього корисно використовувати два упорядкування спеціального вигляду, які позначаються \bar{S} та \underline{S} відповідно.

Означення 1.4. Паралельне упорядкування, за якого вершини графа займають крайнє праве допустиме місце позначається \bar{S} .

Означення 1.5. Паралельне упорядкування, за якого вершини графа займають крайнє ліве місце позначається \underline{S} .

Ті вершини, які в обох наведених упорядкуваннях будуть займати одне і те саме місце, називаються критичними. Зрозуміло, що для них виконується наступна умова:

$$i \in (\bar{S}[k] \cap \underline{S}[k]), 1 < k < l, i \in V. \quad (1.6)$$

Для побудови порядкування \underline{S} існує наступний алгоритм [84,85].

Алгоритм 1.1.

Крок 1. Вважається, що усі місця в \underline{S} є пустими.

Крок 2. Приймається $k = 1$.

Крок 3. У графі G відшукується множина вершин без вхідних дуг і заноситься на k -е місце в \underline{S} .

Крок 4. Вершини, які були занесені на k -е місце упорядкування \underline{S} видаляються з графа разом із дугами, що з них виходять (якщо такі вершини та дуги є).

Крок 5. Перепозначається граф G , якщо це необхідно.

Крок 6. Якщо $|V| = 0$, то кінець алгоритму, якщо ні, то $k = k + 1$ та перехід на крок 3.

Існує аналогічний алгоритм побудови паралельного упорядкування \bar{S} [84,85].

Алгоритм 1.2.

Крок 1. Вважається, що усі місця в \bar{S} є пустими.

Крок 2. Приймається $k = n$.

Крок 3. У графі G шукається множина вершин без вихідних дуг і заноситься на k -е місце в \bar{S} .

Крок 4. Вершини, які були занесені на k -е місце упорядкування \bar{S} видаляються з графа разом із дугами, що до них входять (якщо такі вершини та дуги є).

Крок 5. Перепозначається граф G , якщо це необхідно.

Крок 6. Якщо $|V| = 0$, то кінець алгоритму, якщо ні, то $k = k - 1$ та перехід на крок 3.

Для випадків, коли у задачах упорядкування дозволені переривання, більша увага приділяється вагам вершин, які відповідають тривалості виконання робіт. Очевидно, що використання таких паралельних упорядкувань, які подаються в означенні 1.1, стає неможливим. Постає питання відшукування модифікованої форми запису, яка б враховувала окрім розміщення вершини на позиції також її вагу. Для цього введемо поняття узагальненого паралельного упорядкування наступним чином.

Поставимо у відповідність кожній вершині множини V упорядковану пару (i, w_i) , де i — порядковий номер вершини ($i \in N$), w_i — вага цієї вершини ($w_i \in R$, $w_i > 0$).

Означення 1.6. Розбиттям упорядкованої пари (i, w_i) будемо називати будь-яку скінченну множину упорядкованих пар виду $(i, c_1), (i, c_2), \dots, (i, c_r)$ де

$$\sum_{k=1}^r c_k = w_i, 0 < c_k \leq 1, r \in N.$$

Означення 1.7. Узагальненим паралельним упорядкуванням вершин графа G (або елементів множини V) називається лінійне упорядкування підмножин, що складаються з елементів розбиттів цих вершин, для яких виконуються наступні умови:

1) якщо вершина i передуює вершині j , то всі елементи виду (i, c) знаходяться в S на позиціях не правіше за позицію будь-якого елемента виду

(j, c') , де $0 < c \leq 1$, $0 < c' \leq 1$, $i, j = \overline{1, n}$, $i \neq j$, причому знаходиться на одній позиції вони можуть лише за умови, що $c + c' \leq 1$.

2) у кожній з підмножин, яка включає елементи $(i_1, c_1), (i_2, c_2), \dots, (i_m, c_m)$,

$$\sum_{k=1}^m c_k \leq 1, \quad 0 < c_k \leq 1, \quad \{i_1, i_2, \dots, i_m\} \subseteq V;$$

3) для будь-якої пари елементів (i, c_p) та (i, c_q) серед усіх підмножин, що стоять на одному місці в упорядкуванні, $p \neq q$, $p, q < r$;

4) множина всіх пар виду $(i, c_1), (i, c_2), \dots, (i, c_r)$, які включені в упорядкування, є розбиттям пари (i, w_i) для кожного фіксованого значення i .

Для узагальненої постановки задачі уточнимо поняття ширини та довжини упорядкування. На відміну від класичної постановки, де $S[i]$ — множина вершин, які стоять на i -ому місці упорядкування, тут ширина — це максимум серед кількості підмножин, елементами яких виступають упорядковані пари виду (i, c_k) , де i — порядковий номер вершини, c_k — частка вагового коефіцієнта цієї вершини. Довжиною узагальненого упорядкування називається величина, яка має наступний вигляд:

$$l = \sum_{i=1}^n \left(\max_{S[i]} \left(\sum_{k=1}^m c_k \right) \right).$$

1.5 Ключові питання, що потребують досліджень

При вивченні питань, що виникають у теорії розкладів активно застосовуються моделі і методи розв'язання задач дискретної та комбінаторної оптимізації, теорії графів. Дослідженню таких задач присвячено роботи Сергієнка І.В. [86,87], Гуляницького Л.Ф. [88,89], Козіна І.В. [90,91], Яковлева С.В., Новожилової М.В., Стецюка П.І., неперервних задач оптимального розбиття множин — Кісельової О.М. та її учнів [92-95], розробці методів та алгоритмів складання оптимальних розкладів — Ємця О.О. [96,97]. Потужним інструментом при дослідженні ряду прикладних задач є застосування апарату теорії графів. Так, зокрема, в роботах Гук Н.А. вебграфи

використовуються при аналізі структури сайтів [98-100]. Окремі дослідження Донця Г.П. та Семенюти М.Ф. орієнтовані на спеціальні класи графів [101,102]. Задачі паралельного упорядкування вершин орграфів активно вивчалися Бурдюком В.Я., Турчиною В.А. та їх учнями. Деякі постановки задач теорії розкладів є повністю дослідженими, тобто для них розроблені ефективні алгоритми поліноміальної складності, що знаходять точні розв'язки. Це стосується перш за все тих задач, для яких вдалося довести їх належність до класу поліноміально розв'язних. Але деякі їх інколи незначні узагальнення, що виникають у зв'язку з реальними практичними задачами можуть призводити до того, що задачі стають *NP*-важкими. Тоді постає питання про пошук таких підкласів узагальнених задач, які можна точно розв'язати за поліноміальний час.

В постановках деяких задач чітко йде мова про необхідність переривання при виконанні всіх або певних робіт. Для таких задач розробляються відповідні алгоритми, які враховують ці вимоги.

У випадку, коли переривання не є обов'язковими, але вони не виключаються, виникає питання про доцільність їх введення з метою мінімізації відповідних цільових функцій.

У зв'язку з цим потребують подальшого дослідження наступні питання:

- оцінка ефективності застосування переривань та їх впливу на значення цільової функції в порівнянні з випадками, коли вони заборонені;
- виділення структури графів, для яких переривання впливають на оптимальність, та розробка для них відповідних методів і алгоритмів;
- встановлення граничних значень виграшу від застосування переривань для тих підкласів графів, де переривання зменшують значення цільової функції;
- формулювання необхідних умов, за яких оцінки є досяжними;
- визначення або уточнення грубих апріорних оцінок довжини упорядкування для випадків заборонених та дозволених переривань;
- дослідження узагальнень при постановці задач упорядкування;
- розробка алгоритмів, які враховують дані узагальнення;

- аналіз можливих зв'язків між задачами паралельного упорядкування та іншими задачами дискретної оптимізації;
- розробка відповідного програмного забезпечення, яке реалізує наявні алгоритми та оцінює виграш від використання переривань для задач упорядкування в різних постановках.

Ці питання детально досліджувалися в даній роботі.

1.6 Висновки до розділу

Задачі теорії розкладів — оптимізаційні задачі, що, як правило, моделюють технологічні процеси, в яких планується використання деякого обсягу заданих ресурсів для виконання певного об'єму робіт. Оптимальним розкладом в них називається такий спосіб розподілу робіт між ресурсами, що мінімізує (рідше максимізує) визначений заздалегідь критерій. Серед критеріїв оптимальності найчастіше використовуються мінімаксні (мінімізація часу завершення або зміщення виконання робіт), сумарні (мінімізація сумарних моментів завершення робіт, відхилень або запізнень) або об'єднання кількох з них.

В загальних постановках ряд задач теорії розкладів відносяться до класу *NP*-повних, тому їх дослідження з метою як виділення спеціальних підкласів цих задач, для яких вдається отримати точні алгоритми поліноміальної складності, так і розробка ефективних наближених алгоритмів є актуальними. Для спрощення запису при класифікації задач застосовують скорочений запис виду $\alpha|\beta|\gamma$ за яким постановка задачі враховує характеристики виконавців (машин, процесорів), характеристики робіт, та критерій оптимальності задачі.

Однією з характеристик, що привертає увагу дослідників, є дозвіл на переривання при виконанні робіт. Вплив такого дозволу на оптимальність цільової функції в задачах з кількома виконавцями вперше було розглянуто в [72], що заклало основу подальших досліджень у даному напрямку [69,73-82, 103,104].

Одним з підкласів задачі теорії розкладів є задачі паралельного упорядкування. В них оптимізується розподіл скінченної множини робіт, на порядок виконання яких накладаються технологічні обмеження, між скінченною множиною ідентичних виконавців. Залежно від критерію оптимальності задачі мінімізується або час завершення виконання усіх робіт при заданій кількості виконавців, або кількість виконавців, яка забезпечить завершення виконання в заданий термін. В цих класичних постановках не передбачається застосування переривань при виконанні робіт. Малодослідженими залишаються випадки, коли такі переривання дозволені. Тобто з'ясування того, чи можна покращити значення цільової функції, дозволивши переривання. Якщо це так, то для яких саме підкласів графів таке покращення можливе, а для яких його недоцільно використовувати. Чи суттєвою є розмірність задач при оцінці якості впливу переривань.

Саме з розв'язання таких питань є основним предметом досліджень у даній роботі. Для цього було узагальнено поняття паралельного упорядкування та його характеристик [8].

Розділ 2. ДОСЛІДЖЕННЯ ВПЛИВУ ПЕРЕРИВАНЬ ДЛЯ ПЕВНИХ ПІДКЛАСІВ ГРАФІВ

2.1 Упорядкування без наявності часткового порядку

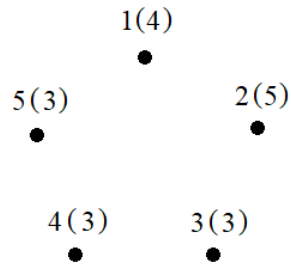
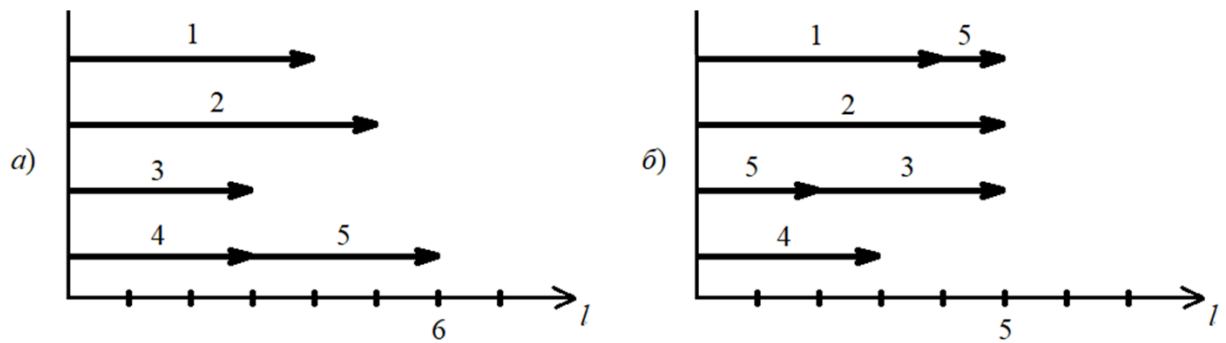
Розглянемо задачу $S(G, h, l)$ для випадку графів, які містять лише ізольовані вершини, тобто $U = \emptyset$. Оскільки час виконання робіт може бути різним, то при математичному моделюванні мова ведеться про зважені графи, у зв'язку з чим введемо позначення для вершин у вигляді упорядкованих пар виду (v_i, w_i) , де w_i — вага вершини $v_i \in V, i = \overline{1, n}$. У наступних прикладах якщо ваги вершин в задачі упорядкування будуть різними, зокрема не рівними 1, то на рисунках вони позначатимуться у дужках біля номера відповідної вершини. З'ясуємо, чи матиме вплив дозвіл переривань на довжину оптимального упорядкування для різних початкових умов задачі, і в разі, якщо це дійсно так, визначимо виграш, який матимемо при цьому. Будемо оцінювати його наступним чином:

$$W = \left(1 - \frac{l_{\Pi}^*}{l^*} \right) \cdot 100\%, \quad (2.1)$$

де l^* — довжина оптимального упорядкування без переривань, l_{Π}^* — довжина оптимального упорядкування з перериваннями. Введена оцінка дозволить визначати можливий виграш від переривань і заздалегідь приймати рішення щодо доцільності їх використання при розв'язанні конкретних прикладних задач.

Проаналізуємо питання впливу переривань на оптимальність розв'язків на деяких прикладах.

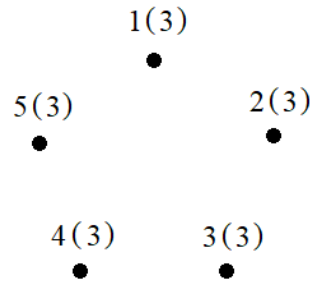
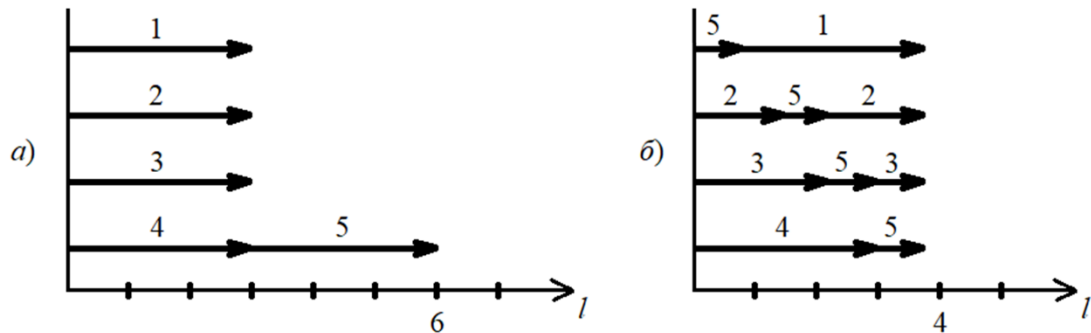
Приклад 2.1. Розв'язати задачу $S(G_1, 4, l)$ для випадків заборонених та дозволених переривань.

Рис. 2.1. Граф G_1 Рис. 2.2. Схеми розподілу вершин графа G_1 без переривань а) та з перериваннями б) при $h = 4$

З рисунку 2.2 видно, що за рахунок дозволу переривань було зменшено довжину упорядкування шляхом заповнення простоїв у системі, тобто порожніх місць. Довжини оптимальних упорядкувань при заборонених та дозволених перериваннях дорівнюють відповідно 6 та 5. Оптимальний розв'язок із використанням переривань дає вигреш $W = 16,6\%$ відносно оптимального розв'язку без переривань.

На наступному прикладі покажемо, як зміниться вигреш при схожих початкових даних, але якщо ваги вершин будуть однаковими.

Приклад 2.2. Розв'язати задачу $S(G_2, 4, l)$ для випадків заборонених та дозволених переривань.

Рис. 2.3. Граф G_2 Рис. 2.4. Схеми розподілу вершин графа G_2 без переривань а) та з перериваннями б) при $h = 4$

Довжина оптимального упорядкування з перериваннями становить $3\frac{3}{4}$, тобто виграш від переривань $W = 37,5\%$. Отже, якщо порівняти результат із попереднім прикладом, то можна дійти висновку, що при однакових вагах вершин виграш від переривань буде більшим, ніж для аналогічної задачі, в якій ваги вершин графа різні. Також варто зауважити, що при однакових вагах

вершин довжина упорядкування з перериваннями завжди буде рівною $\frac{\sum_{i=1}^n w_i}{h}$. В

іншому разі був би можливий випадок, за якого $l_{II}^* = \max_{1 \leq i \leq n} (w_i) \geq \frac{\sum_{i=1}^n w_i}{h}$.

Наступним розглянемо випадок впливу переривань аналогічної задачі, при іншій ширині h .

Приклад 2.3. Розв'язати задачу $S(G_2, 3, l)$ для випадків заборонених та дозволених переривань.

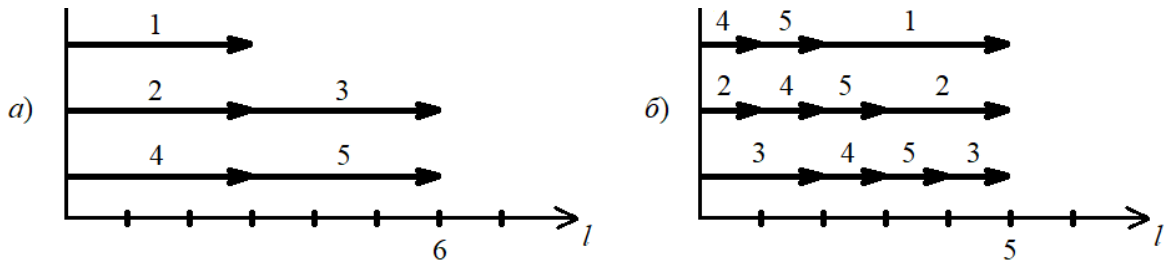


Рис. 2.5. Схеми розподілу вершин графа G_2 без переривань а) та з перериваннями б) при $h = 3$

У цій задачі виграш від використання переривань становить $W = 16,6\%$. При порівнянні з попередніми випадками звернемо увагу на те, що хоч дозвіл на використання переривань і усунув порожні місця в упорядкуванні, однак виграш все ще є незначним. Із цього можна зробити висновок, що виграш від переривань залежить і від ширини упорядкування.

Наведемо узагальнюючий приклад, на якому спробуємо відшукати максимально можливий виграш від переривань для графів, що складаються з ізольованих вершин.

Для формалізації задач про побудову розкладу виконання скінченної множини

Приклад 2.4. Розв'язати задачу $S(G_3, h, l)$ для випадків заборонених та дозволених переривань при $h = p$.

$$\begin{array}{ccccccc} 1(p) & 2(p) & 3(p) & \dots & p+1(p) \\ \bullet & \bullet & \bullet & \dots & \bullet \end{array}$$

Рис. 2.6. Граф G_3

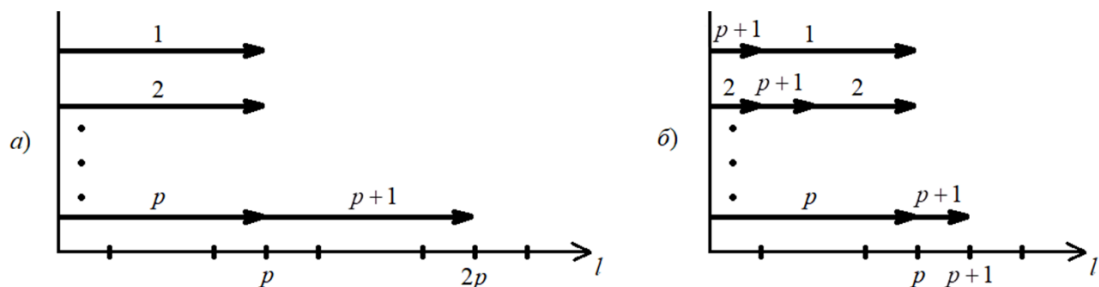


Рис. 2.7. Схеми розподілу вершин графа G_3 без переривань а) та з перериваннями б) при $h = p$

Довжина оптимального упорядкування без переривань для цієї задачі $l^* = 2p$, тоді як із перериваннями маємо $l_{\Pi}^* = p + 1$. Розглянемо їх відношення, узяті з (2.1), і припустимо, що $p \gg 1$:

$$\frac{l_{\Pi}^*}{l^*} = \frac{p+1}{2p} \rightarrow 0,5.$$

Тоді верхнє граничне значення оцінки:

$$\lim_{p \rightarrow \infty} W = \lim_{p \rightarrow \infty} \left(1 - \frac{p+1}{2p} \right) \cdot 100\% = 50\%.$$

Це свідчить про те, що у випадку, коли $U = \emptyset$, дозвіл переривань може суттєво покращити значення цільової функції, у найліпшому випадку майже вдвічі. Варто зауважити, що в наведеному прикладі вагові коефіцієнти вершин рівні, а ширина упорядкування $h = |V| - 1$. При дослідженні впливу переривань на оптимальність розв'язків у задачах, де відповідні графи належать до інших підкласів, аналізу випадків подібних початкових умов буде приділено особливу увагу.

2.2 Вплив переривань для повних дводольних графів

Як виявилось, результати дослідження графів $G = (V, U)$, що містять множину ізольованих вершин, можна використати і при аналізі ефективності переривань для регулярних графів одного підкласу, а саме дводольних. Розглянемо дводольний граф $G = (V, U) = K_{k,r}$ з $|V| = n$ де $V_1 \cup V_2 = V, V_1 \cap V_2 = \emptyset, |V_1| = k, |V_2| = r, n = k + r$. Якщо розглянути випадок, коли $K_{k,r}$ є повним, тобто кількість дуг $|U| = kr$, то легко бачити, що жодна з робіт, якій відповідає вершина з підмножини V_2 , не може бути розпочатою до того, як всі роботи, що відповідають вершинам з підмножини V_1 будуть повністю виконані. Із цього випливає, що можна розглядати оптимальний розклад для графа $G = (V, U)$ як сукупність оптимальних розкладів графів $G_1 = (V_1, U_1)$ та

$G_2 = (V_2, U_2)$ з ізольованими вершинами, тобто $U_1 = \emptyset, U_2 = \emptyset$. Припустимо, що ваги всіх вершин однакові і дорівнюють 1.

Почнемо аналіз із графів, в яких $k = r$. Знайдемо l^* та l_{Π}^* для графа із рисунку 2.8 при $h = 3$.

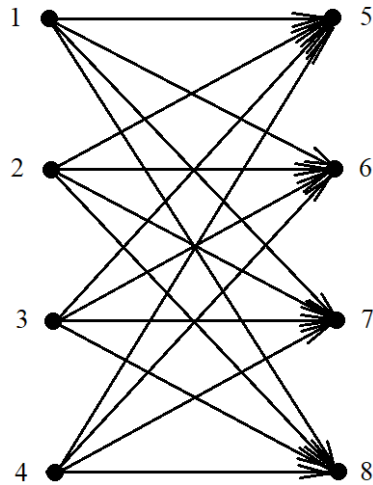


Рис. 2.8. Граф $K_{4,4}$

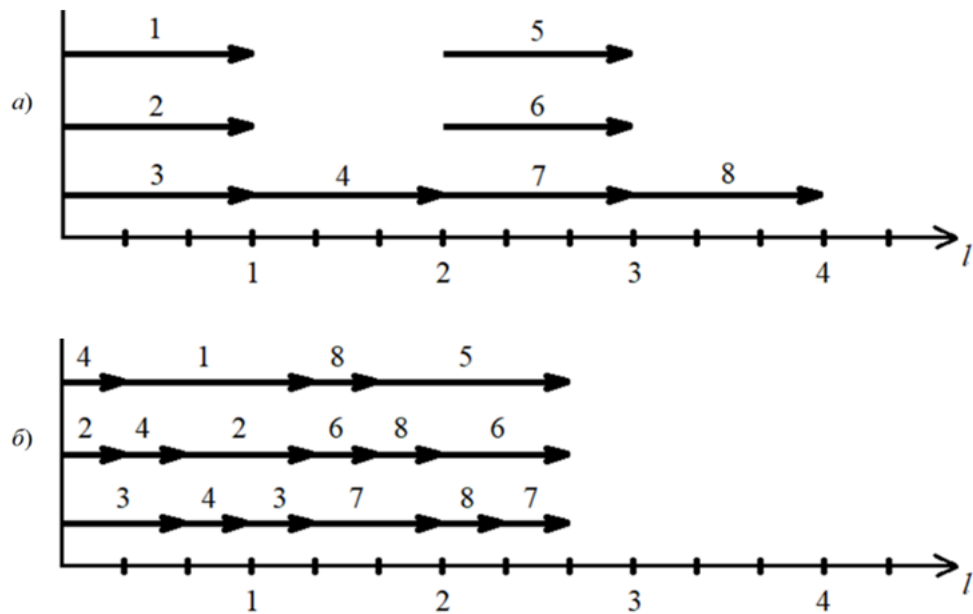


Рис. 2.9. Схеми розподілу вершин повного графа $K_{4,4}$ без переривань а) та з перериваннями б) при $h = 3$

Як можна побачити з рисунку 2.9, довжина оптимального упорядкування з перериваннями є меншою через розподілення вершин під номерами 4 та 8 між трьома місцями упорядкування. Виграш від використання переривань становить

$$W = \left(1 - \frac{2}{3}\right) \cdot 100\% \approx 33\% .$$

Покажемо, що вилучення хоча б однієї дуги з повного дводольного графа суттєво погіршує вплив дозволу переривань. Нехай для визначеності з графа було вилучено дугу (1,5), як показано на рисунку 2.10. Схеми розподілу вершин для цього випадку наведені на рисунку 2.11.

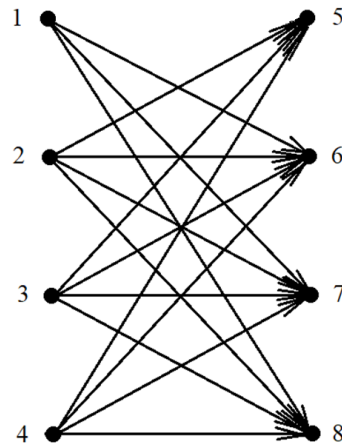


Рис. 2.10. Неповний граф $K_{4,4}$

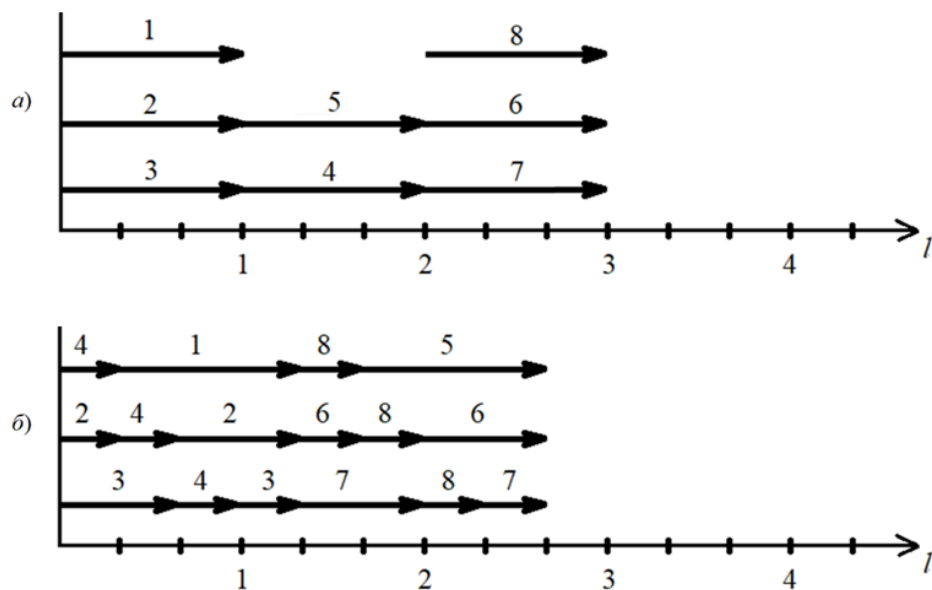


Рис. 2.11. Схеми розподілу вершин неповного графа $K_{4,4}$ без переривань

а) та з перериваннями б) при $h = 3$

Розрахований за формулою виграш від переривань тут становить $W = \left(1 - \frac{8}{9}\right) \cdot 100\% \approx 11\%$, що свідчить про його значне погіршення порівняно з випадком повного графа.

Тепер розглянемо граф $K_{5,5}$ (рис. 2.12) та наведемо упорядкування його вершин при $h = 4$ (рис. 2.13).

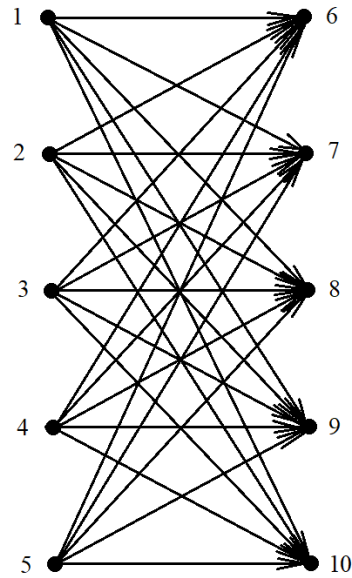


Рис. 2.12. Граф $K_{5,5}$

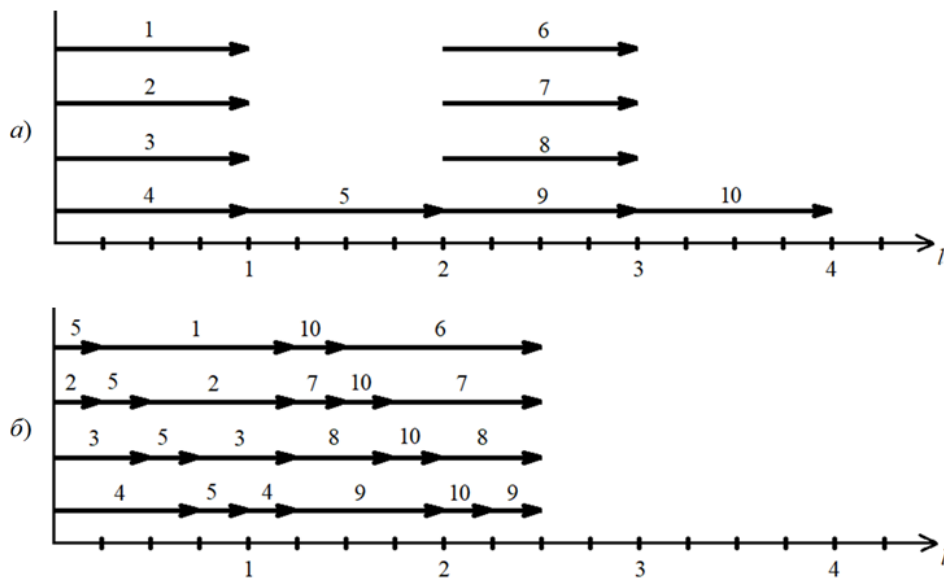


Рис. 2.13. Схема розподілу вершин графа $K_{5,5}$ без переривань а) та з перериваннями б) при $h = 4$

Аналогічно до графа $K_{4,4}$, розглянутого раніше, оптимальне упорядкування з перериваннями має меншу довжину через розподілення вершин 5 та 10 між усіма місцями в упорядкуванні. Виграш від використання переривань становить $W = \left(1 - \frac{5}{8}\right) \cdot 100\% \approx 38\%$.

Для довільного числа $k > 2$ граф $K_{k,k}$ буде мати вигляд, наведений на рисунку 2.14.

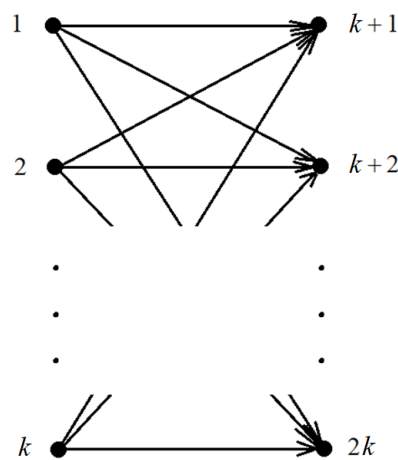


Рис. 2.14. Граф $K_{k,k}$

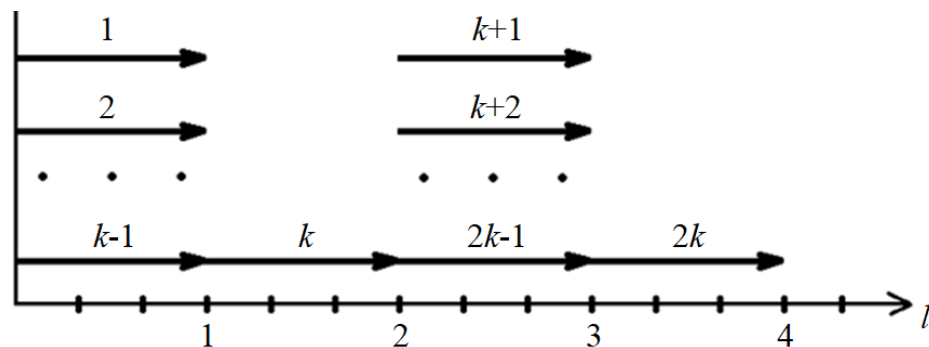


Рис. 2.15. Схема розподілу вершин графа $K_{k,k}$ без переривань

При $h = k - 1$ оптимальне упорядкування вершин графа $K_{k,k}$ з перериваннями має меншу довжину за упорядкування без переривань через те, що вершини під номерами k та $2k$ розподіляються між $(k - 1)$ місцями упорядкування. Виграш від використання переривань становить:

$$W = \left(1 - \frac{\left(2 + \frac{2}{k-1} \right)}{4} \right) \cdot 100\% = \left(1 - \frac{\left(1 + \frac{1}{k-1} \right)}{2} \right) \cdot 100\% .$$

Проаналізувавши цей вираз, легко бачити, що при дедалі більших значеннях k буде відбуватися стабілізація величини виграшу, при якій він прямуватиме до значення, близького до 50%.

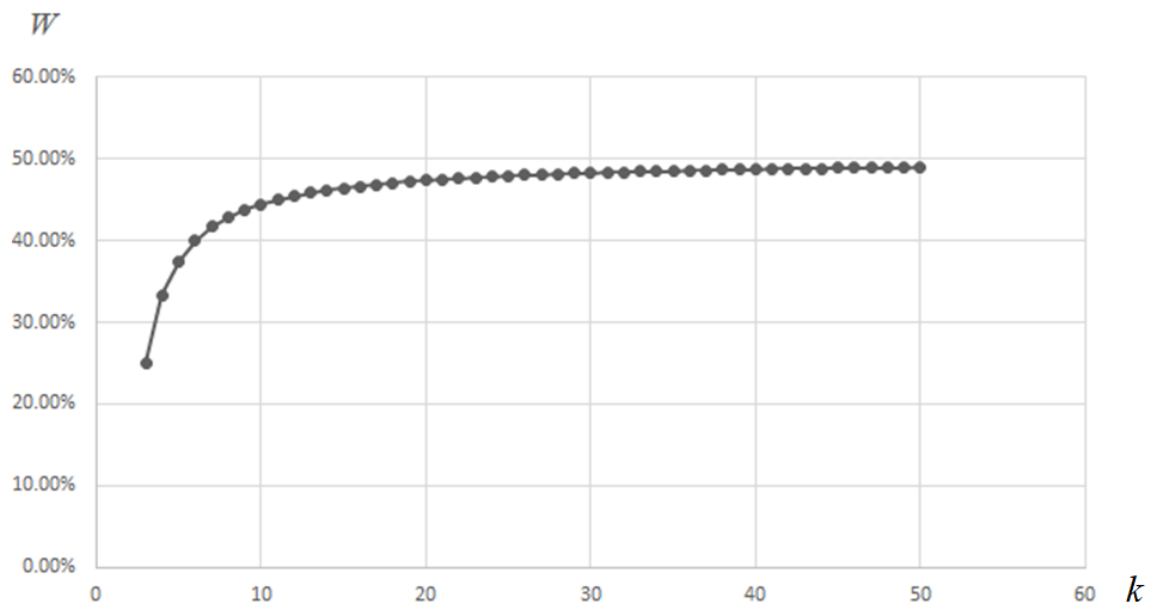
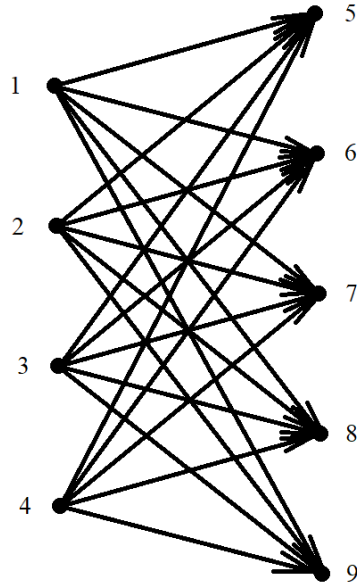
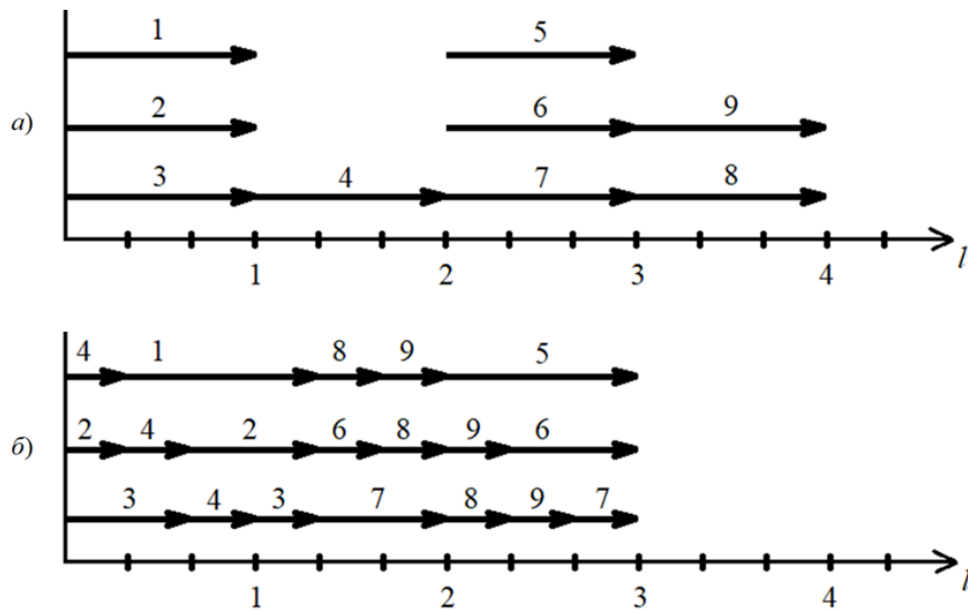


Рис. 2.16. Залежність виграшу від кількості вершин при $h = k - 1$

2.2.1 Переривання у випадках різних кількостей вершин у долях

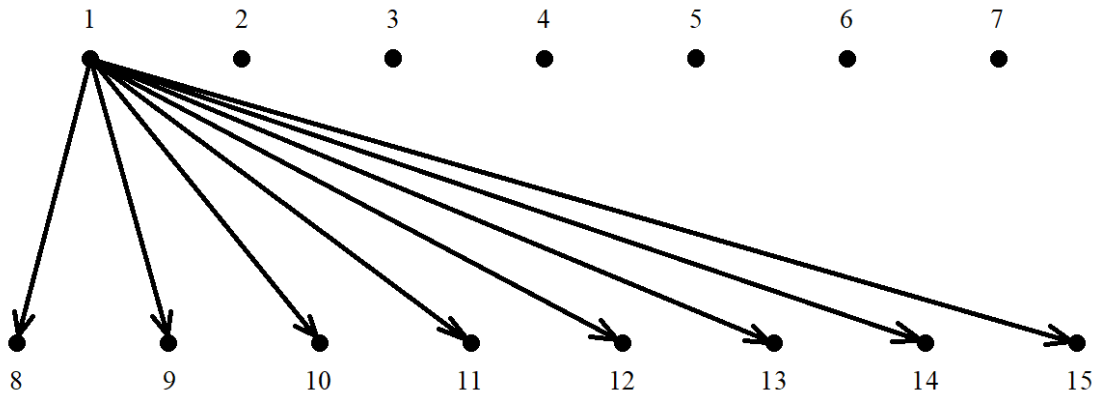
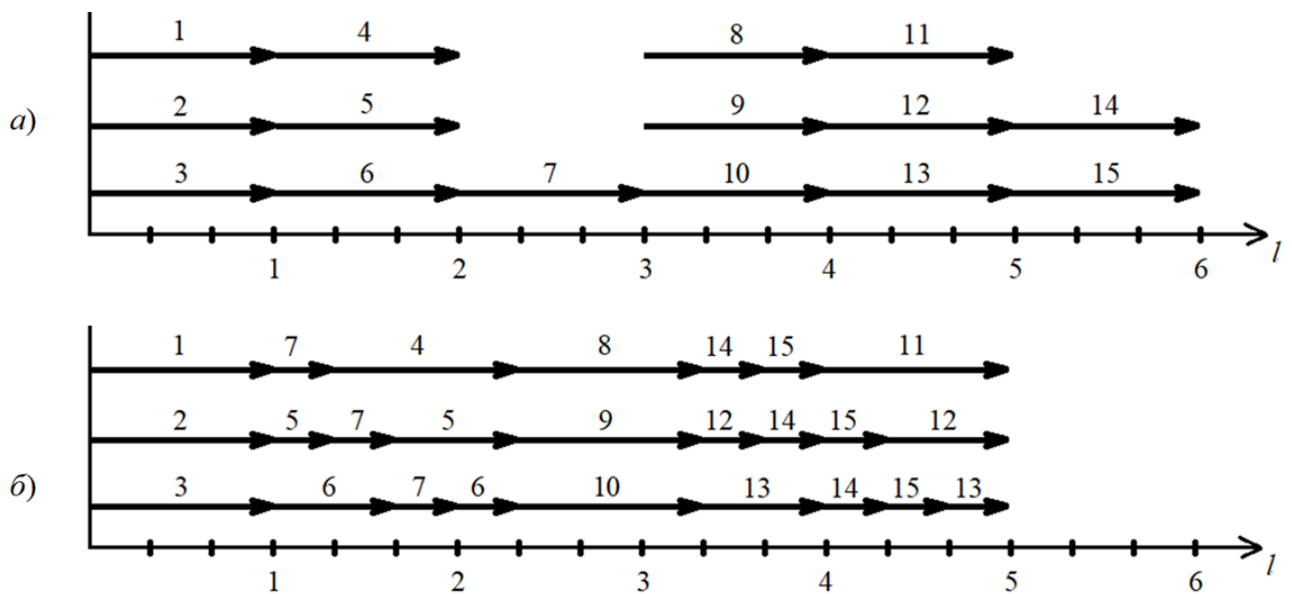
Наступними розглянемо випадки, де $k \neq r$. Почнемо з тих, де $k < r$. На рисунку 2.17 наведено повний граф $K_{4,5}$, а на рисунку 2.18 відповідно схеми розподілу його вершин для випадків без переривань та з перериваннями.

Рис. 2.17. Граф $K_{4,5}$ Рис. 2.18. Схеми розподілу вершин графа $K_{4,5}$ без переривань а) та з перериваннями б) при $h = 3$

Виграш від використання переривань становить $W = \left(1 - \frac{9}{12}\right) \cdot 100\% = 25\%$.

Так само, як і в випадках $k = r$, довжина оптимального упорядкування з перериваннями є меншою за довжину без переривань за рахунок того, що вершини під номерами 4, 8 та 9 розподіляються між місцями упорядкування.

Тепер при $h = 3$ як і для попереднього прикладу розглянемо граф із більшою кількістю вершин у долях — $K_{7,8}$ та наведемо схеми упорядкування вершин [4]. Для спрощення рисунку відмітимо, що із кожної вершини 2–7 першої долі йдуть дуги в кожну вершину 8–15 другої долі аналогічно до тих дуг, які виходять із вершини 1.

Рис. 2.19. Граф $K_{7,8}$ Рис. 2.20. Схеми розподілу вершин графа $K_{7,8}$ без переривань а) та з перериваннями б) при $h = 3$

Виграш від використання переривань становить $W = \left(1 - \frac{15}{18}\right) \cdot 100\% \approx 17\%$. В оптимальному упорядкуванні вершин графа $K_{7,8}$ з перериваннями вершини під номерами 7, 14 та 15 розподіляються між місцями в упорядкуванні, через що

довжина є меншою за довжину оптимального упорядкування без переривань. Далі побудуємо упорядкування цього графу при $h = 4$.

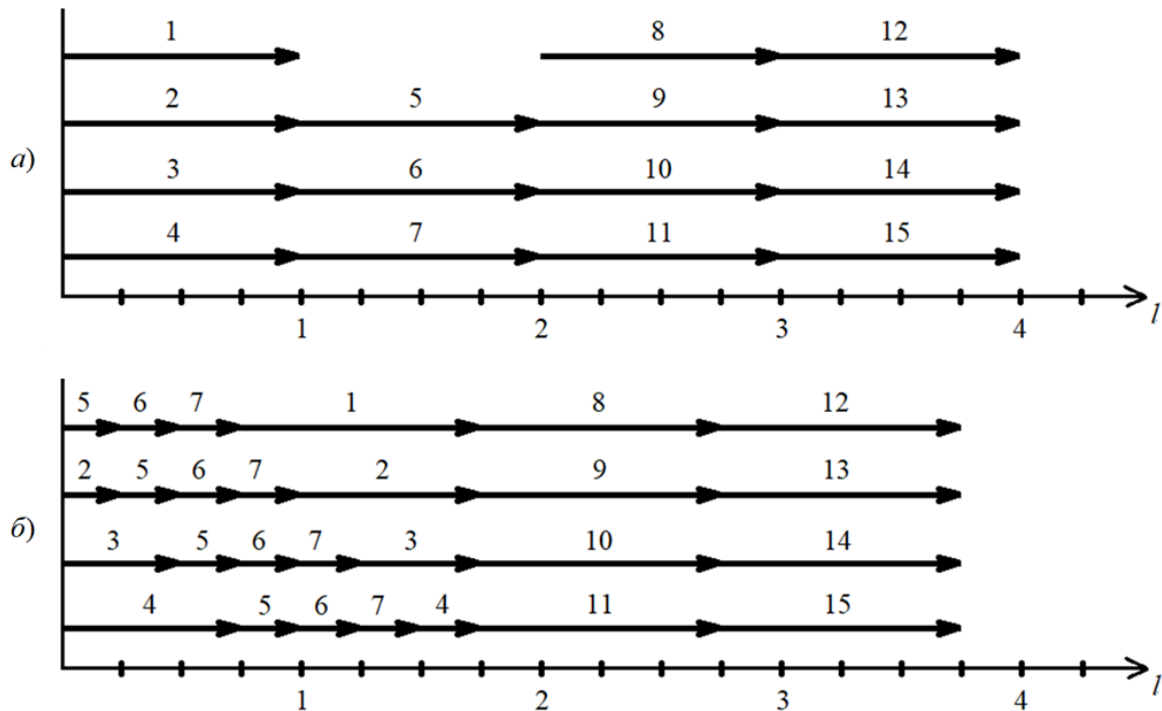


Рис. 2.21. Схеми розподілу вершин графа $K_{7,8}$ без переривань а) та з перериваннями б) при $h = 4$

Тепер виграш від використання переривань становить $W = \left(1 - \frac{17}{18}\right) \cdot 100\% \approx 6\%$ тобто він зменшився порівняно з випадком коли $h = 3$.

Тут так само перериваються і розподіляються між місцями упорядкування три вершини, але всі вони тепер належать першій долі. Для робіт, що відповідають вершинам другої долі переривання в цьому випадку недоцільні.

Проаналізуємо, чи завжди виграш від переривань зменшується при збільшенні ширини упорядкування на прикладі випадку $h = 5$ для того ж графа $K_{7,8}$.

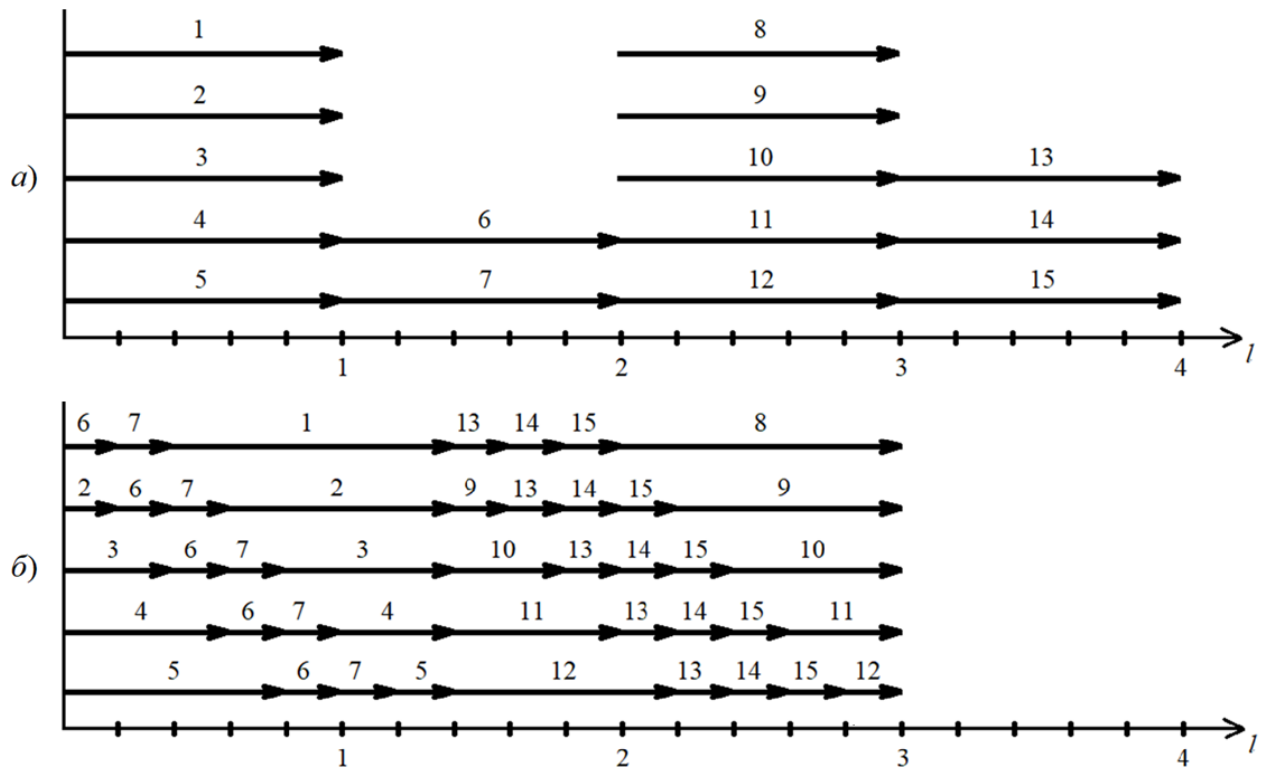
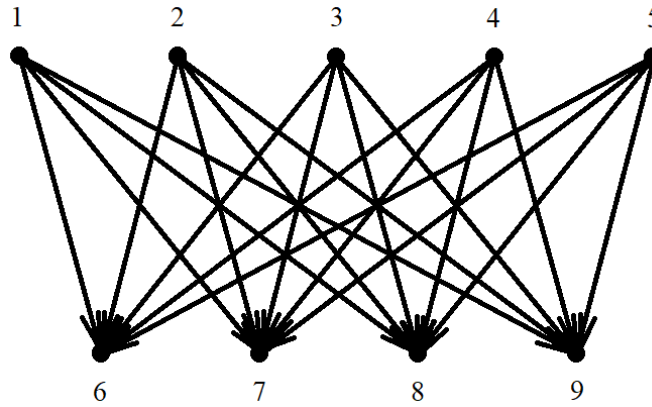
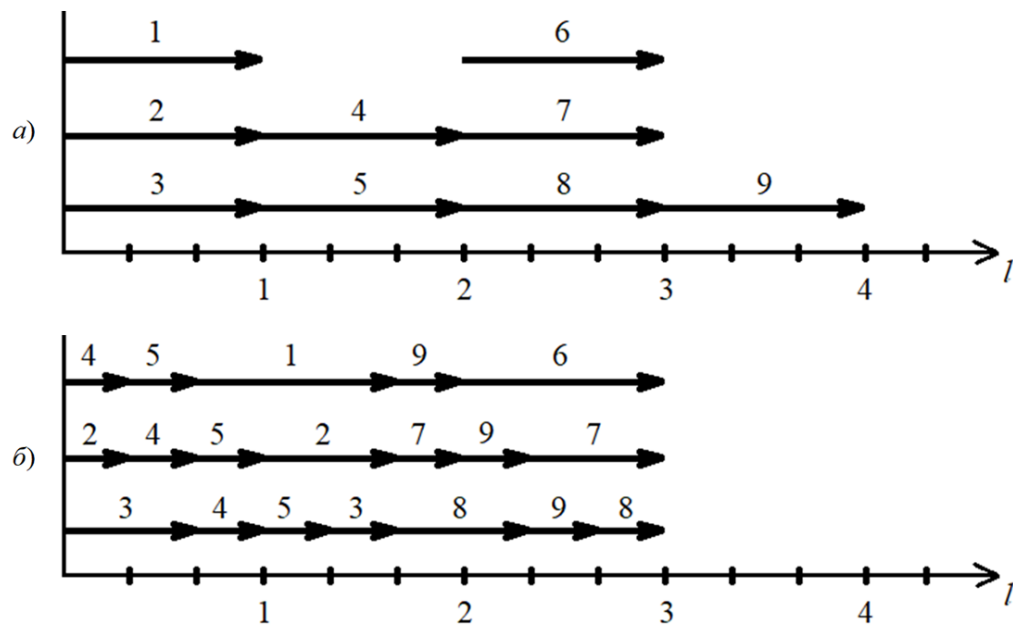


Рис. 2.22. Схема розподілу вершин графа $K_{7,8}$ без переривань а) та з перериваннями б) при $h = 5$

Як видно з упорядкувань при $h = 5$ виграш становить $W = \left(1 - \frac{3}{4}\right) \cdot 100\% = 25\%$, тобто він значно зріс порівняно з випадком $h = 4$. Із наведених прикладів випливає, що при збільшенні ширини h виграш від переривань може як збільшуватися, так і зменшуватися.

Розглянемо випадки, коли $k > r$, та проведемо порівняльний аналіз із результатами, отриманими для $k < r$. Почнемо з розгляду графа $K_{5,4}$, для якого визначимо схеми розподілу вершин для випадків, коли переривання дозволені та заборонені, при $h = 3$.

Рис. 2.23. Граф $K_{5,4}$ Рис. 2.24. Схеми розподілу вершин графа $K_{5,4}$ без переривань а) та з перериваннями б) при $h = 3$

При порівнянні упорядкувань вершин графів $K_{4,5}$ та $K_{5,4}$, легко бачити, що відповідні довжини цих упорядкувань як у випадку із застосуванням переривань, так і без них є однаковими. Внаслідок чого і вигрaш від використання переривань так само дорівнює $W = \left(1 - \frac{9}{12}\right) \cdot 100\% = 25\%$.

Наступним розглянемо граф $K_{8,7}$ і порівняємо відповідні довжини його оптимальних упорядкувань з урахуванням допустимості переривань, та графа $K_{7,8}$. Для цього графа упорядкування без переривань та з перериваннями при $h = 3$ мають наступний розподіл вершин.

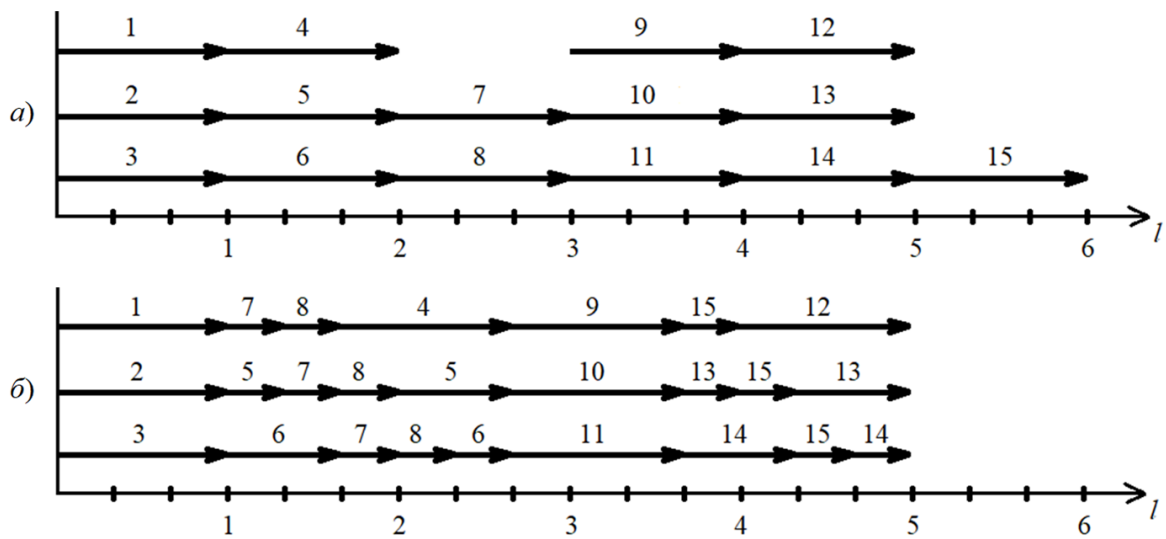


Рис. 2.25. Схема розподілу вершин графа $K_{8,7}$ без переривань а) та з перериваннями б) при $h = 3$

Так само, як і для графів $K_{4,5}$ і $K_{5,4}$, для графів $K_{7,8}$ та $K_{8,7}$ довжини оптимального упорядкування без переривань та з перериваннями є рівними.

Виграш від використання переривань становить $W = \left(1 - \frac{5}{6}\right) \cdot 100\% \approx 17\%$.

У наведених прикладах зменшення довжини упорядкування очевидно відбувається за рахунок застосування переривань при розподілі вершин тієї долі, до якої вони належать. Враховуючи, що розглядаються повні дводольні графи, це підтверджує початкову думку, що упорядкування вершин кожної долі можна розглядати окремо. Також звернемо увагу на те, що при значеннях чи то k , чи то r , кратних ширині h (або обох одночасно), розв'язок задачі упорядкування для відповідної долі буде тривіальним і на значення оптимуму цільової функції дозвіл переривань ніяк не вплине.

На основі наведених прикладів сформулюємо наступне твердження.

Твердження 2.1. Для довільних k, r : $k > 1$, $r > 1$, $k, r \in \mathbb{N}$ довжини оптимальних упорядкування вершин орграфів $K_{k,r}$ та $K_{r,k}$ є рівними і якщо переривання дозволені, і якщо заборонені $\forall h > 1, h \in \mathbb{N}$.

Доведення. Очевидно, що твердження є вірним для випадку, коли $k = r$ оскільки ми маємо граф $K_{k,k}$.

Тепер доведемо справедливість твердження при $k \neq r$. Як було зазначено, упорядкування вершин кожної долі графа можна розглядати окремо. Тоді позначимо довжину оптимального упорядкування без переривань та з перериваннями вершин першої долі відповідно l_k^* та $l_{\Pi k}^*$, а другої — відповідно l_r^* та $l_{\Pi r}^*$. Також уведемо позначення довжини оптимального упорядкування всіх вершин графа $K_{k,r}$ без переривань:

$$l_{k,r}^* = l_k^* + l_r^* \quad (2.2)$$

та з перериваннями:

$$l_{\Pi k,r}^* = l_{\Pi k}^* + l_{\Pi r}^* . \quad (2.3)$$

Аналогічно для графа $K_{r,k}$:

$$l_{r,k}^* = l_r^* + l_k^* ; \quad (2.4)$$

$$l_{\Pi r,k}^* = l_{\Pi r}^* + l_{\Pi k}^* . \quad (2.5)$$

Оскільки у загальному випадку $l_{k,r}^*, l_{r,k}^* \in N$, $l_{\Pi k,r}^*, l_{\Pi r,k}^* \in Q$ то операція додавання є комутативною, тобто:

$$l_k^* + l_r^* = l_r^* + l_k^* ; \quad (2.6)$$

$$l_k^* + l_r^* = l_r^* + l_k^* . \quad (2.7)$$

Підставивши (2.2)-(2.5) у (2.6) та (2.7), доводимо, що $l_{k,r}^* = l_{r,k}^*$ та $l_{\Pi k,r}^* = l_{\Pi r,k}^*$.

2.2.2 Теоретичні аспекти залежності виграшу від початкових даних задачі

Аналіз показує, що виграш при дозволенних перериваннях передусім залежить від порівняння оптимальної довжини упорядкувань для випадків дозволенних переривань для заданого графа, та заборонених, тобто від значення дробу:

$$L = \frac{l_{\Pi}^*}{l^*}. \quad (2.8)$$

Оскільки упорядкування вершин кожної з долей можна розглядати окремо, то достатньо буде розглянути розподіл k вершин при ширині h . Проаналізуємо можливі випадки. Позначимо довжину тієї частини упорядкування, яку займають вершини цієї долі l_k для випадків заборонених переривань, і відповідно $l_{k\Pi}$ — для дозволених. Тоді виграш при розподілі вершин цієї долі W_k буде залежати від відношення цих величин, обчисленого за формулою (2.1).

I. При значеннях k кратних h очевидно виграш буде нульовим, адже в такому випадку всі $k = ph$ вершин ($p \in N$) розміщуються на p позиціях і переривання є недоцільними.

II. Якщо $k < h$, то в упорядкуванні без переривань вершини будуть займати першу позицію, тобто $l_k = 1$. Переривання тут неможливі через те, що одна і та сама вершина не може бути розподілена одночасно на кілька місць в одній позиції.

III. Випадки, коли $h < k < 2h$. Нехай $c \in N$ — остача від ділення k на h , $0 < c < h$. Тоді, враховуючи відсутність взаємних залежностей при розподіленні вершин однієї долі, $l_k = \left\lceil \frac{k}{h} \right\rceil$, або

$$l_k = \left\lceil \frac{h+c}{h} \right\rceil. \quad (2.9)$$

Тут округлення вгору необхідне через те, що $l_k \in N$. Враховуючи величину константи c , чисельник дробу (2.9) завжди буде меншим за $2h$, тобто незалежно від значень h та c за рахунок округлення будемо мати $l_k = 2$.

На відміну від l_k , у загальному випадку $l_{k\Pi} \in Q$. Це означає, що вираз для розрахунку довжини упорядкування з перериваннями є аналогічним до (2.9) за винятком округлення, тобто

$$l_{k\Pi} = \frac{h+c}{h} = 1 + \frac{c}{h}. \quad (2.10)$$

Підставивши (2.9) та (2.10) у (2.8), маємо

$$\frac{\left(1 + \frac{c}{h}\right)}{2} = \frac{1}{2} + \frac{c}{2h}. \quad (2.11)$$

Нарешті, підставивши (2.11) у (2.1) одержуємо

$$W_k = \left(1 - \left(\frac{1}{2} + \frac{c}{2h}\right)\right) \cdot 100\%,$$

а спростивши цей вираз, отримуємо

$$W_k = \left(\frac{1}{2} - \frac{c}{2h}\right) \cdot 100\%. \quad (2.12)$$

IV. Узагальнення $(p-1)h < k < ph$. Маємо у знаменнику (2.8) вираз

$$l_k = \left\lceil \frac{(p-1)h + c}{h} \right\rceil = p, \quad (2.13)$$

а в чисельнику відповідно

$$l_{kII} = \frac{(p-1)h + c}{h} = p - 1 + \frac{c}{h}. \quad (2.14)$$

Відношення l_{kII} / l_k матиме вигляд:

$$\frac{\left(p - 1 + \frac{c}{h}\right)}{p} = 1 - \frac{1}{p} + \frac{c}{ph}. \quad (2.15)$$

Тоді оцінка виграшу:

$$W_k = \left(1 - \left(1 - \frac{1}{p} + \frac{c}{ph}\right)\right) \cdot 100\%,$$

або при спрощенні

$$W_k = \left(\frac{1}{p} - \frac{c}{ph}\right) \cdot 100\%. \quad (2.16)$$

Із (2.16) легко бачити, що виграш дійсно не залежить безпосередньо від кількості вершин k , натомість залежить від частки при діленні $\frac{k}{h}$ націло та числа c , яке є остачею від цього ділення. У тому, що (2.16) є узагальненням легко переконатися, адже при підставленні $p = 2$ одержимо випадок III.

2.3 Специфіка паралельно-послідовних графів та її вплив на переривання

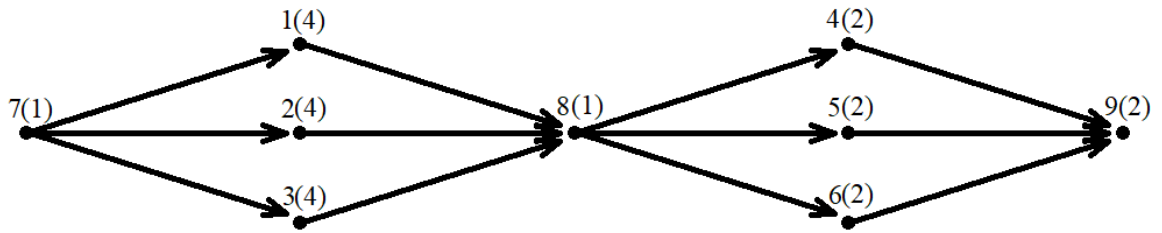
Розглянемо приклади розв'язання задач упорядкування для випадків заборонених та дозволених переривань, де графи $G = (V, U)$ мають структуру спеціального виду і задовольняють умови:

- 1) $|V| = n, |U| = 2(n - 2);$
- 2) $\forall v_i : i = \overline{1, (n-2)} \exists (v_{n-1}, v_i), (v_i, v_n) \in U .$

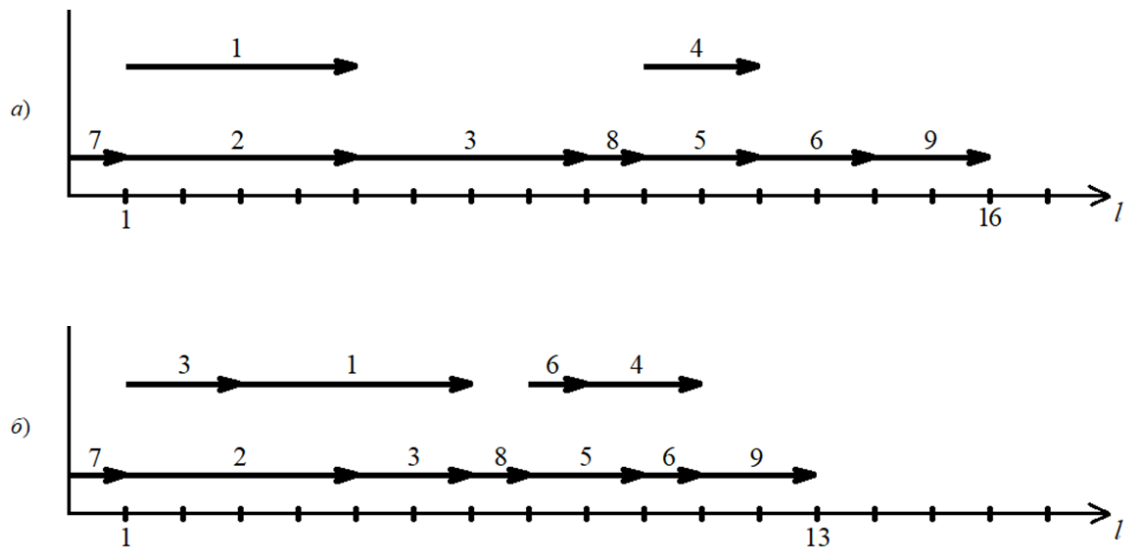
За своєю будовою графи такого вигляду відносяться до підкласу паралельно-послідовних [105], тоді вершина v_{n-1} називатиметься джерелом, а v_n — стоком. У сукупності ці вершини утворюють так звану термінальну пару [106]. Також в якості узагальнення розглянемо випадки послідовної композиції графів подібної структури. Послідовною композицією паралельно-послідовних орграфів виду $G' = (V', U')$ та $G'' = (V'', U'')$ називатимемо граф $G = (V, U)$, в якому вершина-стік v'_n графа G' об'єднується шляхом злиття з вершиною-джерелом v''_{n-1} графа G'' , відповідно усі дуги виду $(v''_{n-1}, v''_i), v''_i \in V''$ перетворюються на (v'_n, v''_i) , після чого вершини перенумеровуються. Слід зауважити, що паралельна композиція таких графів, яка передбачає злиття відповідних вершин-джерел та вершин-стоків призводить до утворення графа, що матиме таку ж структуру, як і ті, з яких він утворений.

Проаналізуємо як дозвіл переривань впливає на оптимальність розв'язків для таких графів. Розглянемо задачу упорядкування вершин паралельно-послідовного графа, утвореного послідовною композицією двох підграфів.

Приклад 2.5. Задано зважений граф G_4 та ширина упорядкування $h = 2$. Побудувати паралельне упорядкування мінімальної довжини.

Рис. 2.26. Паралельно-послідовний граф G_4

Знайдемо оптимальні розв'язки без переривань і з перериваннями для заданої ширини та порівняємо оптимальні значення цільових функцій.

Рис. 2.27. Схема розподілу вершин графа G_4 без переривань а) та з перериваннями б) при $h = 2$

Для G_4 упорядкування з перериваннями дає вигрaш $W = 18,75\%$. З огляду на рисунок 2.27 стає очевидно, що вигрaш при дозволі переривань виникає за рахунок кращого розподілу робіт між виконавцями, що зменшує сумарні простої в обслуговуючій системі. Ці простої можуть виникати на тих місцях упорядкування, що розташовані між позиціями термінальних вершин підграфів.

Переконаємося, що переривання матимуть вплив на довжину оптимального упорядкування навіть для графа, який не утворений послідовною композицією.

Приклад 2.6. Розглянемо граф G_5 та для ширини $h = 4$ знайдемо паралельне упорядкування мінімальної довжини.

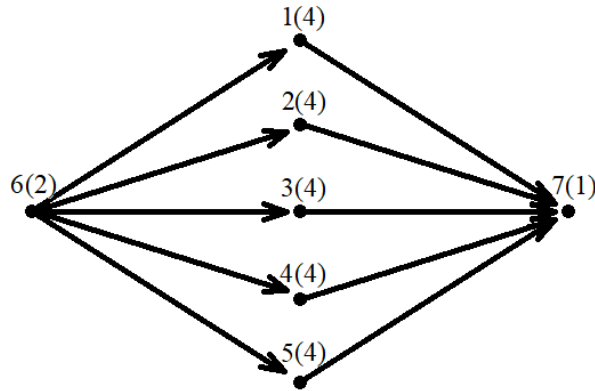


Рис. 2.28. Паралельно-послідовний граф G_5

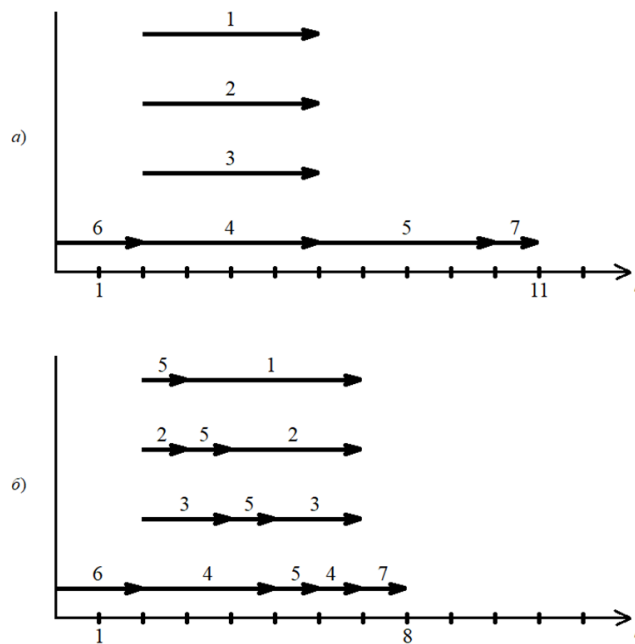


Рис. 2.29. Схема розподілу вершин графа G_5 без переривань а) та з перериваннями б) при $h = 4$

Із наведеного прикладу видно, що навіть для такого графа дозвіл переривань здатен значно покращити розв'язок, у даному випадку $W = 27, (27)\%$. Розглянемо узагальнення такого випадку для паралельно-послідовного графа, що утворений без послідовної композиції кількох підграфів.

Приклад 2.7. Знайти паралельне упорядкування мінімальної довжини графа G_6 при ширині упорядкування $h = p$.

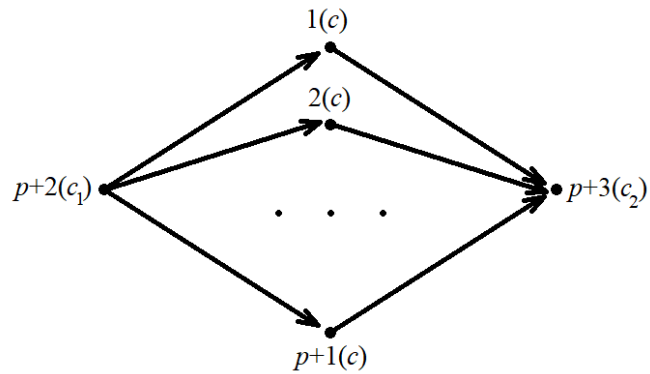


Рис. 2.30. Паралельно-послідовний граф G_6

При $c = p$ розподіл вершин в упорядкуванні для випадків заборонених та дозволених переривань показано на рисунку 2.31.

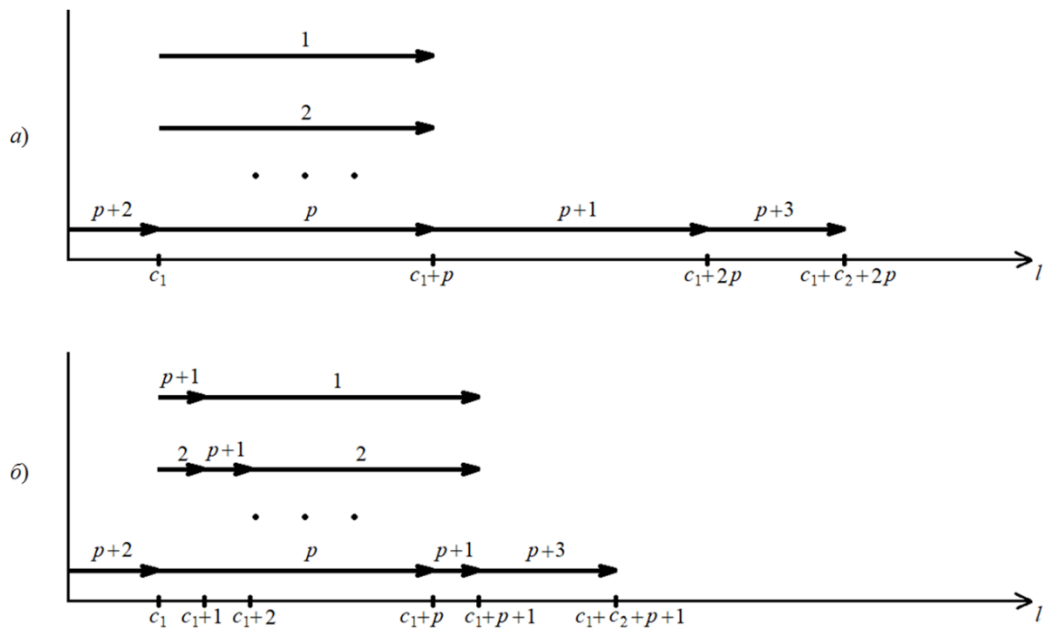


Рис. 2.31. Схема розподілу вершин графа G_6 без переривань а) та з перериваннями б) при $h = p$

Для зручності позначимо $S = c_1 + c_2$ та матимемо вигляд наступного вигляду:

$$W = \left(1 - \frac{C + p + 1}{C + 2p} \right) \cdot 100\% .$$

При оцінці виграшу розглянемо два випадки:

- 1) $C \gg p$, тоді виграш $W \rightarrow 0\%$;
- 2) $p \gg C$, тоді виграш $W \rightarrow 50\%$.

На наступному етапі узагальнення розглянемо граф із довільною кількістю підграфів виду G_6 , об'єднаних послідовною композицією.

Приклад 2.8. Для заданого графа G_7 та ширини упорядкування $h = p$ знайти паралельне упорядкування із мінімальною довжиною.

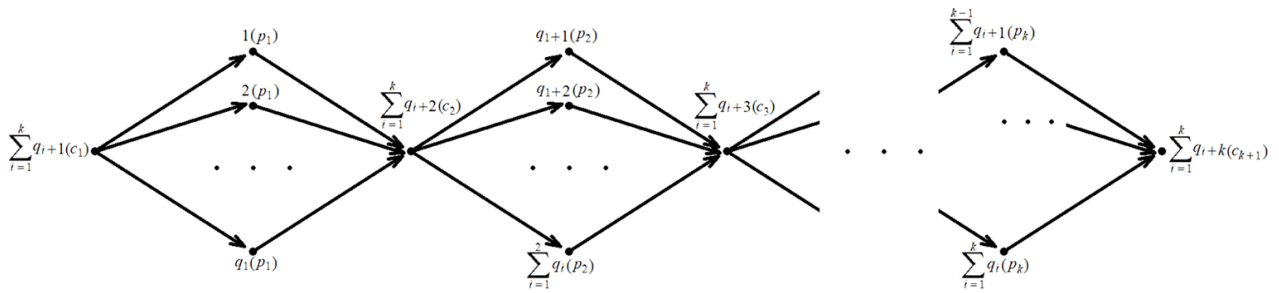


Рис. 2.32. Паралельно-послідовний граф G_7

У випадку, коли всі $q_i = p+1$, $i = \overline{1, k}$ матимемо упорядкування без переривань та з перериваннями за схемами, які показано на рисунку 2.33. Виграш від переривань будемо оцінювати аналогічно тому, як це робили для прикладу 2.7. Для зручності позначимо

$$C = \sum_{i=1}^{k+1} c_i .$$

Тоді виграш матиме вигляд:

$$W = \left(1 - \frac{\sum_{i=1}^k (p_i + 1) + C}{\sum_{i=1}^k 2p_i + C} \right) \cdot 100\% .$$

Як і в попередньому прикладі розглянемо 2 випадки:

1) $C \gg \sum_{i=1}^k p_i$, тоді виграш $W \rightarrow 0\%$.

2) $\sum_{i=1}^k p_i \gg C$, тоді виграш $W \rightarrow 50\%$.

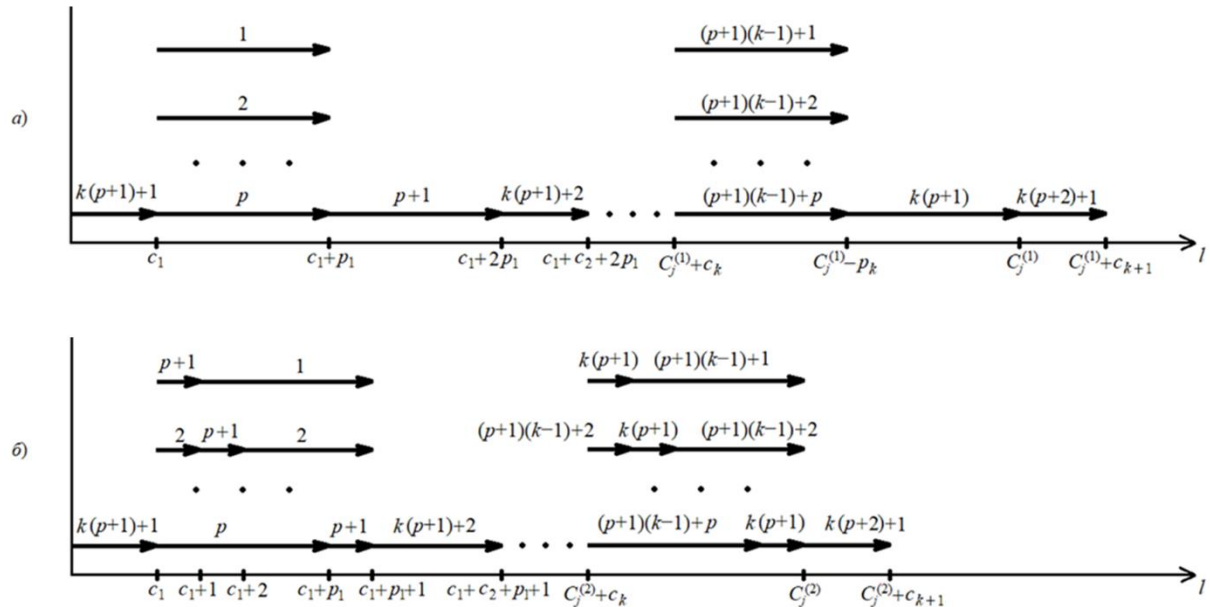


Рис. 2.33. Схема розподілу вершин графа G_6 без переривань а) та з перериваннями б) при $h = p$

На графіках рис. 2.33 для спрощення були введені позначення виду:

$$C_j^{(1)} = \sum_{i=1}^j (c_i + 2p_i) \text{ та } C_j^{(2)} = \sum_{i=1}^j (c_i + p_i + 1).$$

2.4 Паралельні упорядкування вершин граціозних дерев

Одним із спеціальних класів дерев є такий, що містить граціозні дерева. Дерево називають граціозним, якщо воно допускає граціозну розмітку. Означення такої розмітки, наведене в [107].

Означення 2.1. Граціозною називають таку розмітку вершин неорієнтованого дерева T із n ребрами, при якій кожній вершині ставиться у

взаємно однозначну відповідність мітка із множини $\{0, 1, \dots, n\}$, причому всі індуковані мітки ребер є різними.

Означення 2.2. Індукованою міткою ребра є абсолютна величина різниці міток вершин, інцидентних до цього ребра.

У 1967 р. було сформульовано гіпотезу, згідно з якою всі дерева є граціозними [108]. На даний момент вона залишається гіпотезою, проте граціозність деяких класів дерев була обґрунтована. Розглянемо задачі паралельного упорядкування для відповідних дерев таких класів, для яких доведено можливість граціозної розмітки. Оскільки поняття граціозності відноситься до неорієнтованих дерев, а в подальшому будуть розглядатися орієнтовані, то можливі різні варіанти заміни ребер на дуги. Окрім того будемо вважати, що графи зважені та їх вершини мають одиничну вагу, якщо не вказано інакше.

2.4.1 Оцінка впливу переривань для дерев-зірок

Одним з найперших класів дерев, для яких було доведено можливість граціозної розмітки, є зірки. Наведемо відоме означення [107].

Означення 2.3. Зіркою називається дерево, що має одну вершину степеня, більшого за 1, а решта вершин мають степінь 1.

При детальнішому розгляді орієнтованих зірок можна виділити такі три категорії на основі того, яким саме чином ребра замінюються дугами.

1. Всі дуги виходять із центральної вершини зірки, тоді маємо вихідне кореневе дерево.

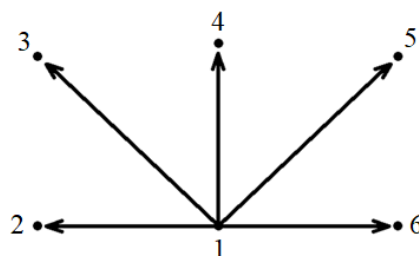


Рис. 2.34. Вихідна зірка

2. Всі дуги входять в центральну вершину зірки, тоді маємо вхідне дерево.

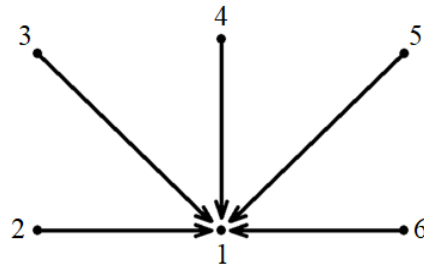


Рис. 2.35. Вхідна зірка

3. Частина дуг входять в центр зірки, решта — виходять.

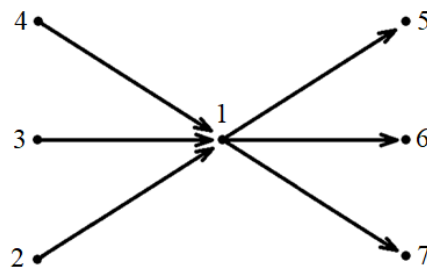


Рис. 2.36. Зірка із вхідними та вихідними дугами

При розв'язанні задач паралельного упорядкування, коли обмеження на порядок відповідають графу, що має структуру вихідної зірки, її центральна вершина ставиться на перше місце упорядкування. Решта вершин вносяться в довільному порядку на наступні місця, адже вони не пов'язані між собою дугами. Для вхідної зірки аналогічним чином спершу всі вершини, крім центральної, розміщуються в упорядкуванні у довільному порядку, після чого заноситься центральна. У випадку комбінованої зірки частина вершин, із яких дуги прямують до центральної, вносяться до упорядкування першими, потім центральна вершина зірки, після чого — решта вершин.

Нехай скінченна множина робіт, на порядок виконання яких накладаються умови слідування, моделюється деревом, що є орієнтованою зіркою. З'ясуємо вплив переривань при виконанні робіт на оптимальність розв'язку в відповідних задачах упорядкування для цих випадків.

Легко бачити, що оптимальне упорядкування для вхідних та вихідних зірок із кількістю вершин $(p+1)$ відрізняється від випадку графа з p ізольованими вершинами лише тим, що містить ще одну вершину на першій (для вихідних зірок) чи останній (для вхідних) позиції. Це також означає, що довжина оптимального упорядкування буде на 1 більшою, ніж для відповідного графа з ізольованими вершинами незалежно від ширини упорядкування h .

Тоді виграш становить

$$W = \left(1 - \frac{\frac{p}{h} + 1}{\left\lceil \frac{p}{h} \right\rceil + 1} \right) \cdot 100\% .$$

Для $h = p - 1$ спростимо відношення довжин

$$\frac{\frac{p}{p-1} + 1}{\left\lceil \frac{p}{p-1} \right\rceil + 1} = \frac{\frac{p-1+1}{p-1} + 1}{\left\lceil \frac{p-1+1}{p-1} \right\rceil + 1} = \frac{\frac{p-1}{p-1} + \frac{1}{p-1} + 1}{\left\lceil \frac{p-1}{p-1} + \frac{1}{p-1} \right\rceil + 1} = \frac{1 + \frac{1}{p-1} + 1}{\left\lceil 1 + \frac{1}{p-1} \right\rceil + 1} = \frac{\frac{1}{p-1} + 2}{\left\lceil \frac{1}{p-1} \right\rceil + 2} .$$

В результаті округлення дробу $\frac{1}{p-1}$ до найменшого цілого, що перевищує дане значення (тобто округлення вверх), завжди маємо 1 для будь-яких значень $p \geq 2$, тобто остаточно маємо

$$W = \left(1 - \frac{\frac{1}{p-1} + 2}{3} \right) \cdot 100\% .$$

При $p \gg 1$

$$\frac{1}{p-1} \rightarrow 0 \Rightarrow W \rightarrow 33, (3)\% .$$

Отже, для достатньо великої кількості вершин у вхідній чи вихідній зірці дозвіл переривань даватиме істотний виграш при $h = p - 1$. Тепер узагальнимо міркування, припускаючи, що $2 \leq h \leq p - 1$ і покажемо залежність $W(h)$ для деяких конкретних значень p .

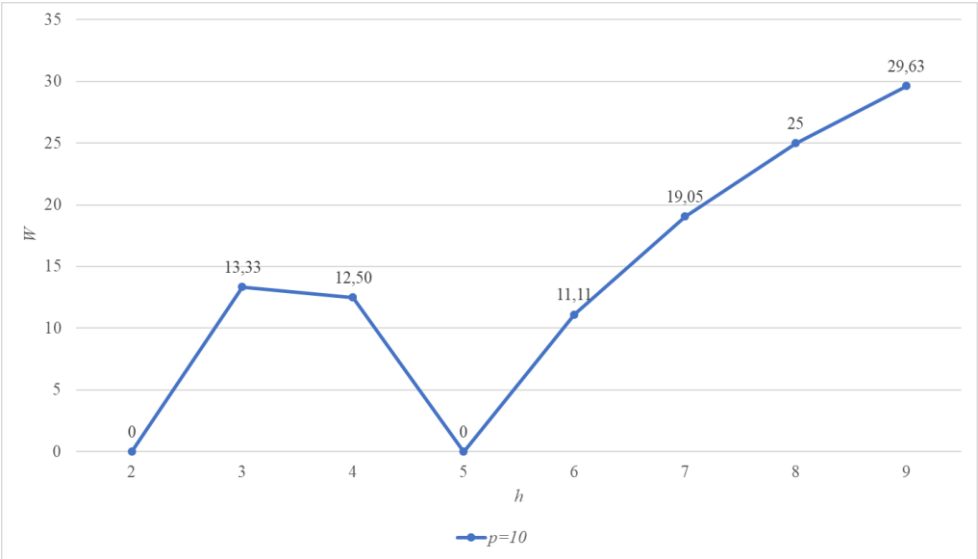


Рис. 2.37. Залежність виграшу від ширини упорядкування при $p = 10$

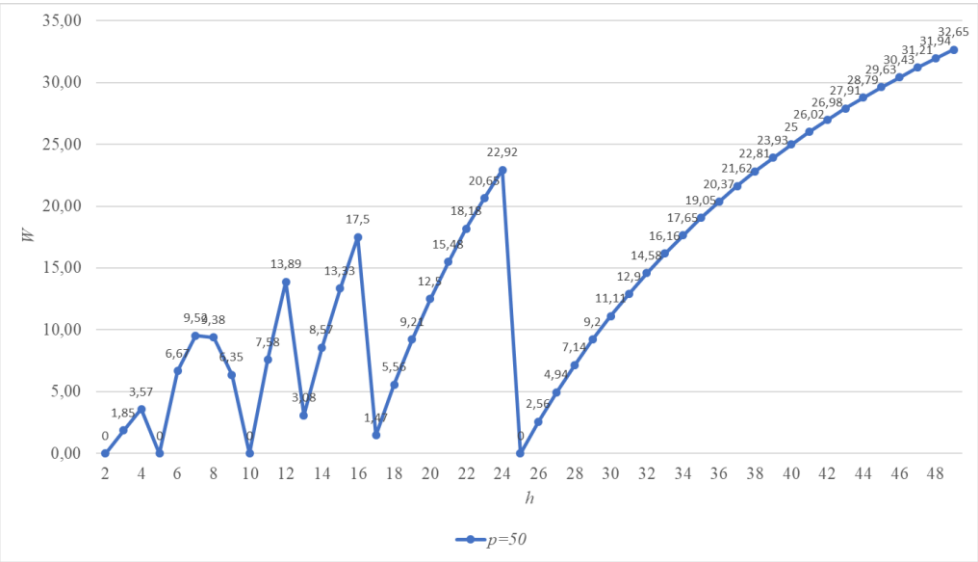


Рис. 2.38. Залежність виграшу від ширини упорядкування при $p = 50$

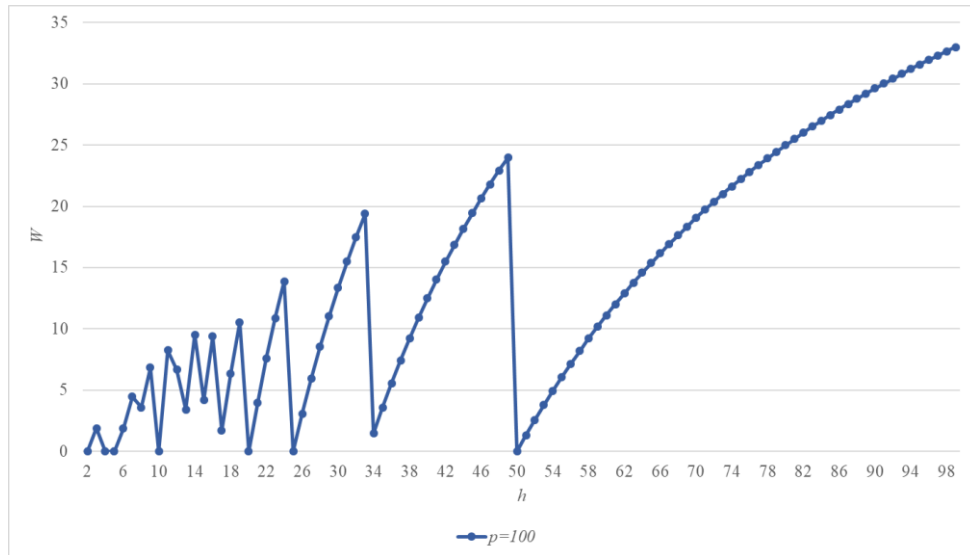


Рис. 2.39. Залежність виграшу від ширини упорядкування при $p = 100$

Для комбінованого випадку, коли в зірці є p_1 дуг, що входять до центральної вершини, та p_2 таких, які виходять з неї (сумарно $p_1 + p_2 = p$), виграш розраховується наступним чином:

$$W = \left(1 - \frac{\frac{p_1 + p_2}{h} + 1}{\left\lceil \frac{p_1}{h} \right\rceil + \left\lceil \frac{p_2}{h} \right\rceil + 1} \right) \cdot 100\% \text{ або } W = \left(1 - \frac{p_1 + p_2 + h}{h \left(\left\lceil \frac{p_1}{h} \right\rceil + \left\lceil \frac{p_2}{h} \right\rceil + 1 \right)} \right) \cdot 100\%,$$

де $h \geq 2$, а відповідно $p_1, p_2 \geq 3$.

Розглядалися різні комбінації p_1 та p_2 , але закономірностей між виграшами при порівнянні випадків для вхідних та вихідних зірок із кількістю дуг p_1 та p_2 виявлено не було.

Встановимо залежності виграшу від різних значень параметрів p_1 , p_2 та h . Для визначеності припустимо, що $p_1 \geq p_2$. Як видно із рис. 2.37-2.39, максимальні значення виграшу досягаються при значеннях h близьких до кількості дуг. Через це вважатимемо, що $h = p_2 - 1$. Виграш при різних комбінаціях значень p_1 , p_2 та h для $p = 50$ наведено на рисунку 2.40, а для $p = 100$ — на рисунку 2.41.

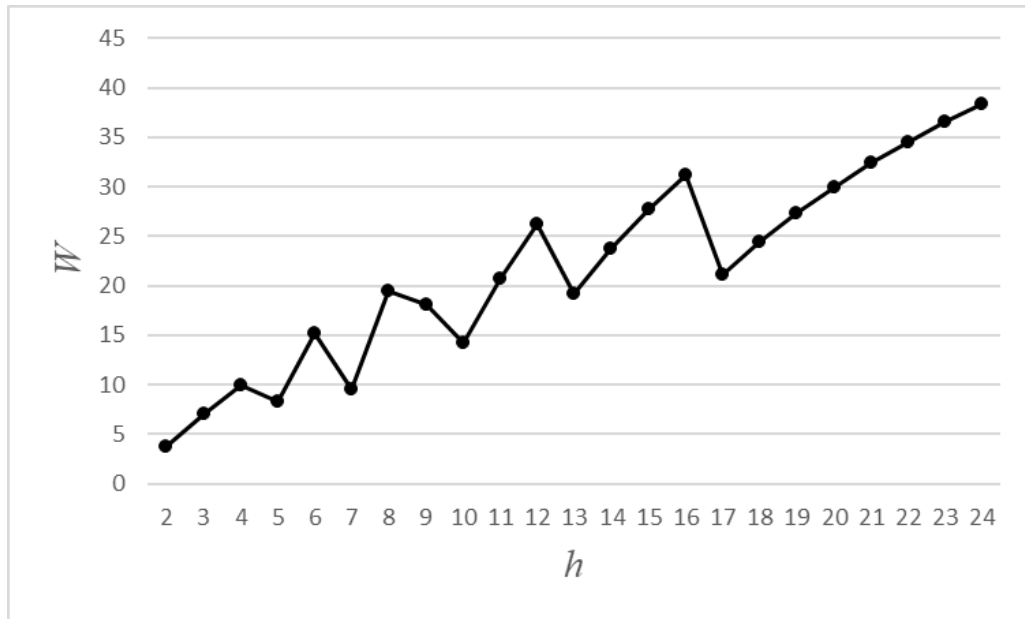


Рис. 2.40. Залежність виграшу від параметрів для $p = 50$

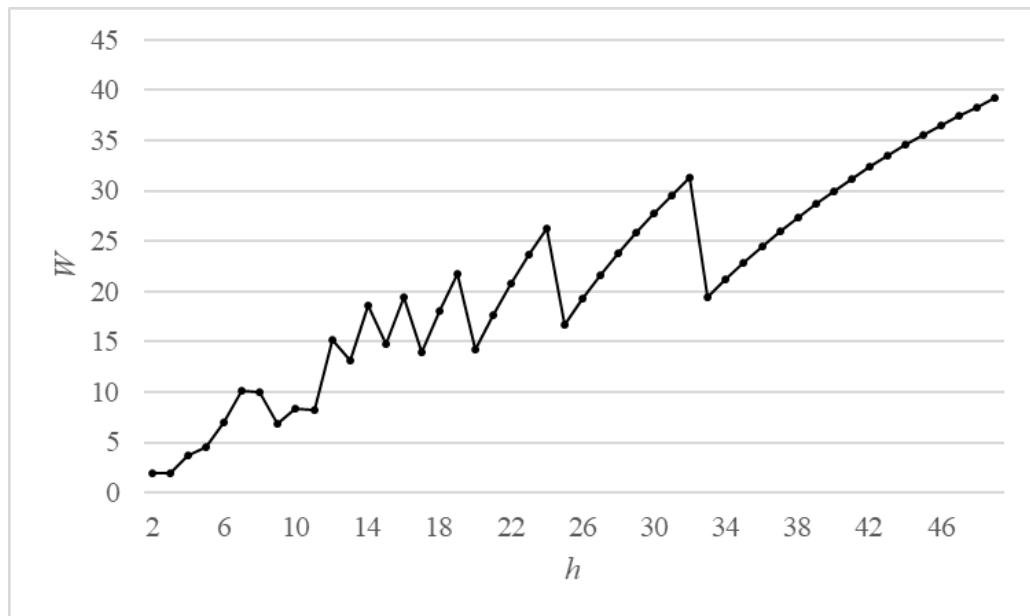


Рис. 2.41. Залежність виграшу від параметрів при $p = 100$

Як видно з графіків, максимальні значення виграшу досягаються при $p_1 = p_2$. З'ясуємо, до якого значення сходиться виграш. Введемо позначення $p' = p_1 = p_2$ і відповідно $p' = h + 1$. Після підстановки отримуємо:

$$W = \left(1 - \frac{3h+2}{h \left(\left\lceil \frac{h+1}{h} \right\rceil + \left\lceil \frac{h+1}{h} \right\rceil + 1 \right)} \right) \cdot 100\% ,$$

або при спрощенні

$$W = \left(1 - \frac{3h+2}{5h}\right) \cdot 100\% .$$

Для значень $h \gg 2$ (а отже і $p' \gg 2$) виграш $W \rightarrow 40\%$.

2.4.2 Класи дерев, що зводяться до дерев-зірок

Звернемося до іншого класу дерев, для яких було доведено можливість граціозної розмітки — оливкові дерева.

Означення 2.4. Оливковим називається дерево, яке складається з p ланцюгів, що з'єднані в одній вершині, при цьому довжина i -ого ланцюга ($i = \overline{1, p}$) дорівнює i [107].

Перейдемо від оливкових дерев до зірок. Для цього зробимо перетворення оливкового дерева у зважену зірку наступним чином. Кожен i -ий ланцюг, довжина якого більша за одиницю, згорнемо до одного ребра, отримавши загалом p ребер виду $\{i, (p+1)\}$, в той же час перенумерувавши вершини. Тоді $(p+1)$ — центральна вершина зірки, що має ваговий коефіцієнт 1, а кожна інша вершина i має вагу i . При заміні ребер дугами аналогічно до зірок отримаємо 3 випадки:

- 1) всі дуги виходять із центральної;
- 2) всі дуги входять в центральну;
- 3) певна частина дуг входять, решта — виходять із центральної.

З'ясуємо, як саме дозвіл переривань у випадку, коли всі вершини мають різну вагу, впливає на значення цільової функції.

Процес побудови упорядкування вершин такої зірки відбувається аналогічно тому, як це робилося у випадку одиничних ваг нецентральных вершин. На останнє місце (для вхідної) або на перше (для вихідної) ставиться центральна вершина. Решта вершин, оскільки вони не зв'язані між собою дугами, можуть розміщуватися в упорядкуванні по аналогії з випадком незалежних вершин. Будемо вимагати, щоб для підмножини нецентральных вершин

виконувалась умова $\max_{1 \leq i \leq p} (w_i) < \frac{\sum_{i=1}^p w_i}{h}$. Враховуючи специфіку таких зірок, оцінка виграшу при кількості вершин $(p+1)$ має вигляд:

$$W = \left(1 - \frac{\frac{\sum_{i=1}^p i}{h} + 1}{l^* + 1} \right) \cdot 100\% .$$

Оскільки апріорне визначення довжини оптимального упорядкування без переривань в даному випадку є нетривіальною задачею, то оцінимо це значення знизу наступним чином

$$l^* \geq \left\lceil \frac{\sum_{i=1}^p i}{h} \right\rceil .$$

Тоді оцінка

$$W = \left(1 - \frac{\frac{\sum_{i=1}^p i}{h} + 1}{\left\lceil \frac{\sum_{i=1}^p i}{h} \right\rceil + 1} \right) \cdot 100\%$$

відповідає мінімальному гарантованому виграшу.

Слід відмітити, що виграшу від переривань не буде у випадку, коли

$p \geq \frac{\sum_{i=1}^p i}{h}$ оскільки довжини упорядкувань з перериваннями та без них будуть однаковими. З'ясуємо, для яких значень h ця нерівність не виконується.

Враховуючи, що ваги вершин утворюють арифметичну прогресію, замінімо $\sum_{i=1}^p i$ на формулу суми перших p її членів:

$$\frac{p(p+1)}{2h} > p \Rightarrow \frac{p(p+1)}{2p} > h \Rightarrow h < \frac{p+1}{2}.$$

У подальшому будемо розглядати лише ті випадки, для яких h задовольняє цій умові, при цьому зауважимо, що вона не може бути виконана при $p = 3$ через те, що в такому випадку це б означало, що $h < 2$. Проаналізуємо, яким буде мінімальний гарантований виграш для графів із кількістю вершин $(p+1)$. Ширину упорядкування візьмемо рівною найбільшому натуральному числу, що не перевищує $\left(\frac{p+1}{2} - 1\right)$. Відповідний графік залежності виграшу від параметра p зображено на рис. 2.42.

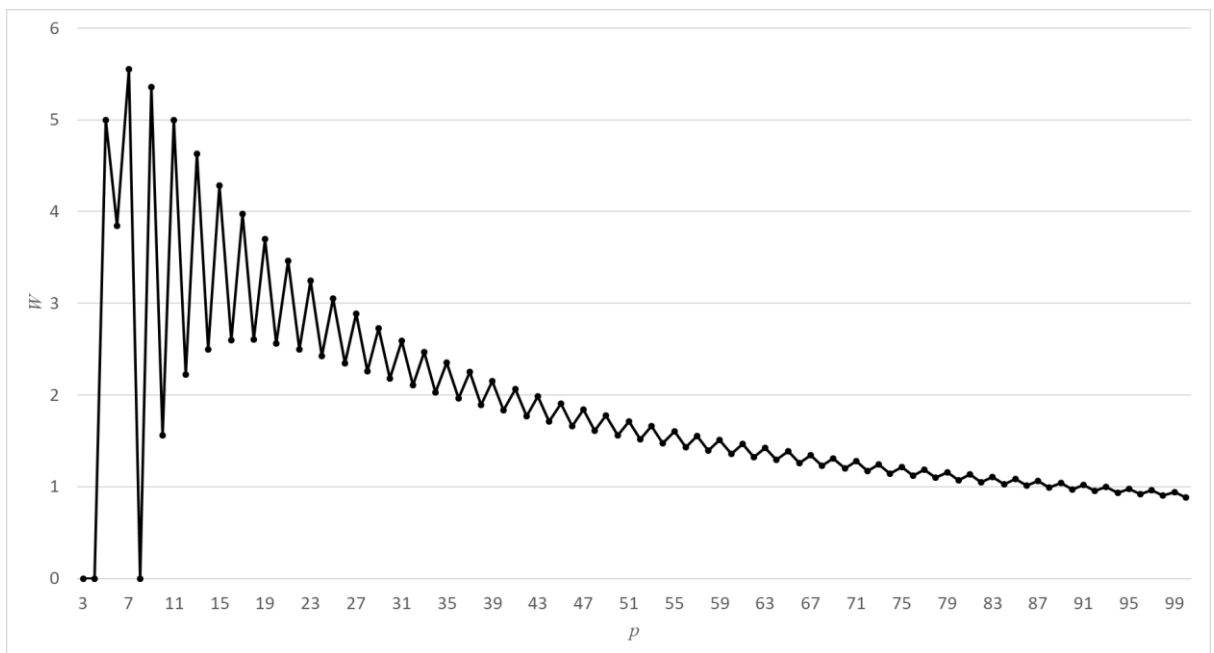


Рис. 2.42. Залежність гарантованого виграшу від p для вхідної або вихідної зірки

З графіка, зображеного на рисунку, видно, що при збільшенні кількості вершин графа з урахуванням того, що відповідно значення h теж зростає, гарантований виграш буде зменшуватися.

Проаналізуємо, до якого значення сходиться гарантований виграш в такому разі.

$$W = \left(1 - \frac{\frac{1}{h} \cdot \frac{p(p+1)}{2} + 1}{\left\lceil \frac{1}{h} \cdot \frac{p(p+1)}{2} \right\rceil + 1} \right) \cdot 100\% .$$

Тепер, враховуючи, що $h = \frac{p+1}{2} - 1 = \frac{p-1}{2}$, маємо:

$$W = \left(1 - \frac{\frac{2}{p-1} \cdot \frac{p(p+1)}{2} + 1}{\left\lceil \frac{2}{p-1} \cdot \frac{p(p+1)}{2} \right\rceil + 1} \right) \cdot 100\% .$$

Спростивши, маємо

$$W = \left(1 - \frac{\left(\frac{p(p+1)}{p-1} \right) + 1}{\left\lceil \frac{p(p+1)}{p-1} \right\rceil + 1} \right) \cdot 100\% .$$

Оскільки $p \geq 3$, то значення $\frac{p(p+1)}{p-1}$ збільшується зі зростанням p . Окрім того легко бачити, що воно буде відрізнятися від $\left\lceil \frac{p(p+1)}{p-1} \right\rceil$ менше, ніж на 1. Тоді для великих значень p

$$\frac{\frac{p(p+1)}{p-1} + 1}{\left\lceil \frac{p(p+1)}{p-1} \right\rceil + 1} \rightarrow 1 \Rightarrow W \rightarrow 0 .$$

Для комбінованого випадку, коли в зірці є не лише дуги, що входять до центральної вершини, а й такі, що виходять із неї, проаналізуємо залежність можливого гарантованого виграшу для графа, що містить p_1 вхідних та p_2 вихідних дуг, враховуючи що $p_1 + p_2 = p$. Вважатимемо, що $p_1 > 2, p_2 > 2$ та $2 \leq h < \max(p_1, p_2)$. Також припустимо, що будь-яка вершина v_j , в яку входить

дуга із центральної, має більшу вагу за будь-яку вершину v_i , із якої входить дуга в центральну, тобто

$$\forall v_i \in V : (v_i, v_{p+1}) \in U, \forall v_j \in V : (v_{p+1}, v_j) \in U \Rightarrow w_i < w_j.$$

Подібно до перших двох випадків оцінимо значення виграшу

$$W = \left(1 - \frac{\max\left(\frac{p_1(p_1+1)}{2h}, p_1\right) + \max\left(\frac{p_2(p_1+1+p)}{2h}, p\right) + 1}{l^*} \right) \cdot 100\%.$$

Оцінка довжини оптимального упорядкування без переривань

$$l^* \geq \max\left(\left\lceil \frac{p_1(p_1+1)}{2h} \right\rceil, p_1\right) + \max\left(\left\lceil \frac{p_2(p_1+1+p)}{2h} \right\rceil, p\right) + 1.$$

Таким чином мінімальний гарантований виграш матиме вигляд

$$W = \left(1 - \frac{\max\left(\frac{p_1(p_1+1)}{2h}, p_1\right) + \max\left(\frac{p_2(p_1+1+p)}{2h}, p\right) + 1}{\max\left(\left\lceil \frac{p_1(p_1+1)}{2h} \right\rceil, p_1\right) + \max\left(\left\lceil \frac{p_2(p_1+1+p)}{2h} \right\rceil, p\right) + 1} \right) \cdot 100\%.$$

Для кожного фіксованого значення p були визначені можливі гарантовані виграші для усіх допустимих пар (p_1, p_2) таких, що $p_1 + p_2 = p$ та $p_1 > 2, p_2 > 2$. На рисунку 2.43 зображено графік залежності виграшу від параметра p , що є максимальним серед усіх значень, які обчислені для відповідних комбінацій p_1 та p_2 при фіксованих $p \leq 100$ (де $p \in N$).

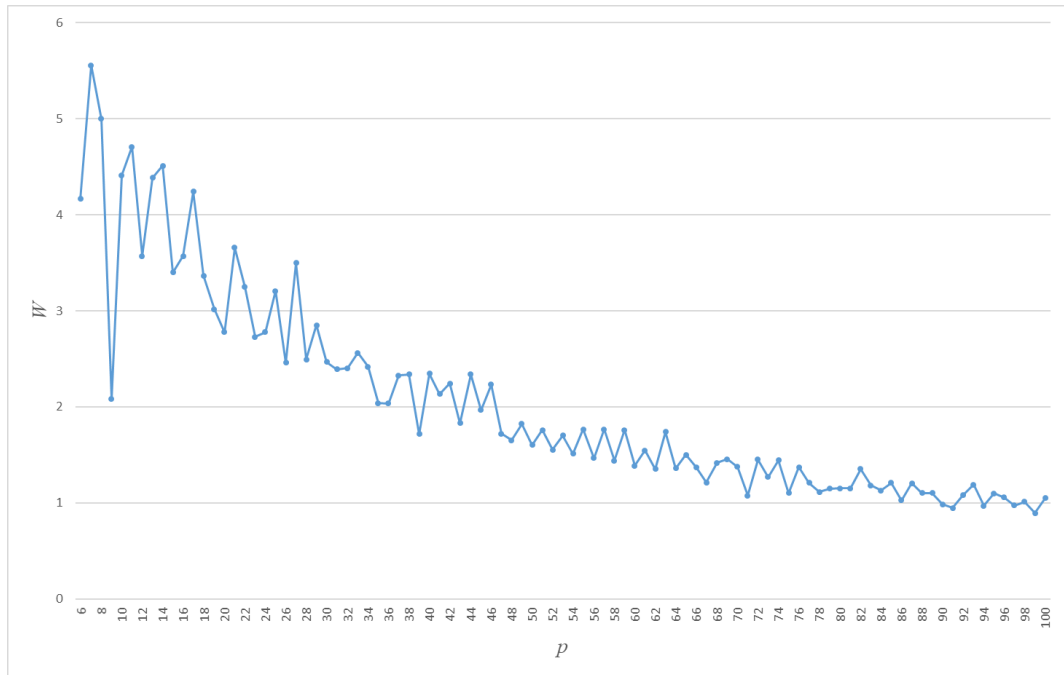


Рис. 2.43. Залежність гарантованого виграшу від p у комбінованому випадку

Проаналізуємо поведінку виграшу при різних значеннях p , p_1 та p_2 . Легко бачити, що значення виразів $\max\left(\frac{p_1(p_1+1)}{2h}, p_1\right)$ та $\max\left(\left\lceil \frac{p_1(p_1+1)}{2h} \right\rceil, p_1\right)$ залежать від p_1 так само, як у вхідній зірці з (p_1+1) вершинами. Вирази $\max\left(\frac{p_2(p_1+1+p)}{2h}, p\right)$ та $\max\left(\left\lceil \frac{p_2(p_1+1+p)}{2h} \right\rceil, p\right)$ як і попередня пара відрізняються один від одного не більше, ніж на 1. Враховуючи, що $p_2 = p - p_1$, розглянемо дріб

$$\frac{p_2(p_1+1+p)}{2h} = \frac{(p-p_1)(p_1+1+p)}{2h} = \frac{p^2 + p - p_1^2 - p_1}{2h}.$$

Значення цього дробу буде збільшуватися при збільшенні p за фіксованого h , але зменшуватиметься при зростанні p_1 . Однак враховуючи, що $p > p_1$ а також $p_1 \geq 3$ максимальне значення $p_1 = p - 3$ маємо

$$\frac{p^2 + p - p_1^2 - p_1}{2h} = \frac{p^2 + p - (p-3)^2 - (p-3)}{2h} = \frac{6p-6}{2h} = \frac{3p-3}{h}.$$

Отже, при збільшенні p значення цих дробів буде зростати, а відповідно

$$\frac{\max\left(\frac{p_1(p_1+1)}{2h}, p_1\right) + \max\left(\frac{p_2(p_1+1+p)}{2h}, p\right) + 1}{\max\left(\left\lceil \frac{p_1(p_1+1)}{2h} \right\rceil, p_1\right) + \max\left(\left\lceil \frac{p_2(p_1+1+p)}{2h} \right\rceil, p\right) + 1} \rightarrow 1 \Rightarrow W \rightarrow 0.$$

Було проведено ряд обчислювальних експериментів, у ході яких встановлені значення найбільших для кожного фіксованого $p \leq 100$ вигравів, обраних серед усіх можливих комбінацій p_1 та p_2 . Графік цієї залежності наведено на рисунку 2.44, де для порівняння продубльовано графік з рисунку 2.43. Наявна розбіжність в одержаних значеннях вигравів може пояснюватися тим, що оцінка довжини оптимального упорядкування без переривань, зокрема доданку $\max\left(\left\lceil \frac{p_2(p_1+1+p)}{2h} \right\rceil, p\right)$ є достатньо грубою.

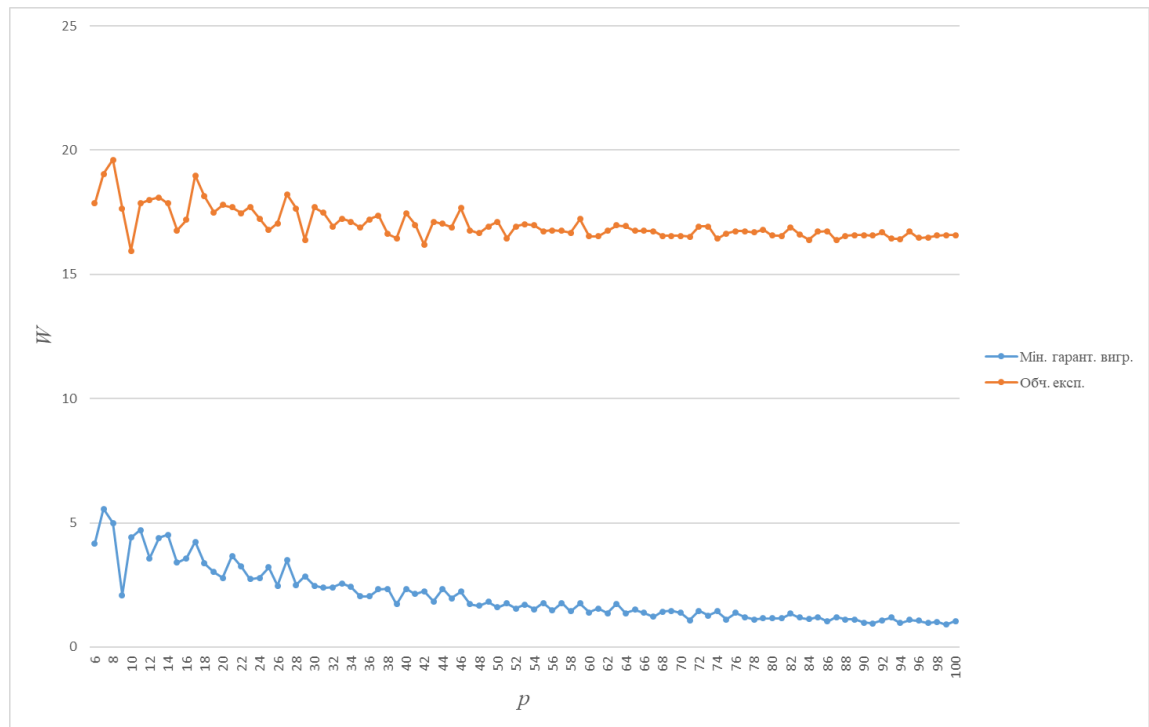


Рис. 2.44. Порівняння оцінок виграшу з результатами обчислювального експерименту

Попри існування даних розбіжностей між значеннями верхнього граничного та мінімального гарантованого вигравів, стає очевидним, що при збільшенні значення параметра p вони зменшуються. Порівняння одержаних

результатів із тими, що були отримані в п. 2.4.1 свідчить, що для випадків рівних та різних вагових коефіцієнтів вершин при сталій структурі графів даного підкласу ефективність застосування переривань може різко відрізнятись.

2.4.3 Априорна оцінка довжини упорядкувань

При дослідженні впливу переривань для зірок із різними ваговими коефіцієнтами вершин була застосована груба оцінка довжини упорядкування для випадків, коли переривання не передбачені. Це спонукає проаналізувати можливість більш точно априорно оцінити значення цільової функції. Нехай кількість вершин вхідної чи вихідної зірки дорівнює $(p+1)$, причому вершина з номером $(p+1)$ є центральною і має довільний ваговий коефіцієнт $c \in N$. Детальніше розглянемо випадок, коли вагові коефіцієнти решти вершин відрізняються на 1 і для зручності вважатимемо, що для i -ої вершини ($1 \leq i \leq p$) він дорівнює i . Позначимо як $l_{(p)}^*$ мінімальну кількість місць, на які можна розмістити нецентральні вершини $1, 2, \dots, p$ такої зірки.

Нагадаємо, що для графа, який має структуру вхідної чи вихідної зірки центральна вершина буде займати відповідно останні чи перші c місць упорядкування, причому жодна інша вершина не може бути розміщена на тих же місцях. Проаналізуємо, якою є мінімальна кількість місць, на які можна розмістити решту p вершин. Нехай $p = kh + r, 0 \leq r < h$, де h — задана ширина упорядкування.

Твердження 2.2. Для $k > 3$ довжину оптимального упорядкування вершин вхідної чи вихідної зірки l^* можна представити у вигляді $l^* = l_{(p-2h)}^* + l_k + c$, де $l_k = \text{const}$.

Доведення. Розглянемо $2h$ вершин із найбільшими ваговими коефіцієнтами $(kh + r), (kh + r - 1), \dots, ((k - 2)h + r + 1)$. Згрупуємо їх попарно таким чином, щоб сума вагових коефіцієнтів в кожній з пар була однаковою. Тоді маємо h пар

вершин, кожна з яких займатиме при внесенні в упорядкування місця з першого по $kh + r + (k - 2)h + r + 1 = (2k - 2)h + 2r + 1$.

Таким чином, позначивши $l_k = (2k - 2)h + 2r + 1$, одержуємо $l^* = l_{(p-2h)}^* + l_k + c$.

Твердження 2.3. Для $k > 3$ довжину l^* оптимального упорядкування вершин вхідної чи вихідної зірки можна представити у вигляді $l^* = l_{(2h+r)}^* + L_k + c$ якщо k — парне, і $l^* = l_{(3h+r)}^* + L_k + c$, якщо непарне, а $L_k = \text{const}$.

Доведення. Враховуючи попереднє твердження, довжину оптимального упорядкування можна представити у вигляді

$$l^* = l_{(p-2h)}^* + l_k + c, \quad (2.17)$$

де $l_k = (2k - 2)h + 2r + 1$. Оскільки $p - 2h = (k - 2)h + r$ бачимо що твердження є істинним у випадках, коли $k - 2 \leq 3$, $L_k = l_k$. В іншому разі $l_{((k-2)h+r)}^* = l_{((k-4)h+r)}^* + l_{k-2}$. Підставивши цей вираз у (2.17) маємо $l^* = l_{((k-4)h+r)}^* + l_k + l_{k-2} + c$. Якщо $k - 4 \leq 3$, то позначивши $L_k = l_k + l_{k-2}$ одержуємо або $l^* = l_{(2h+r)}^* + L_k + c$, або $l^* = l_{(3h+r)}^* + L_k + c$.

Процес продовжується ітераційно, на кожному кроці до упорядкування заносяться $2h$ вершин так, як описано при доведенні твердження 2.2, та виокремлюються константні доданки $l_k, l_{k-2}, \dots, l_{k-2q}$ в (2.17), де $q = \frac{(k-4)}{2}$ якщо k — парне, і $q = \frac{(k-5)}{2}$ якщо непарне. Тоді $L_k = \sum_{j=0}^q l_{k-2j}$, а $l^* = l_{(2h+r)}^* + L_k + c$ або $l^* = l_{(3h+r)}^* + L_k + c$ відповідно для парних та непарних значень k .

Таким чином показано, що значення довжини оптимального упорядкування вершин зірки можна априорно уточнити.

2.4.4 Інші підкласи дерев

Серед інших підкласів дерев, що допускають граціозну розмітку варто також виділити ланцюги та одностантні дерева (або гусениці).

Означення 2.5. Дерево, в якому дві вершини мають степені рівні 1, а решта вершин — степені рівні 2, називається ланцюгом [107].

Орієнтований ланцюг складається з $|V| = k$ вершин, послідовно з'єднаних дугами множини $U = \{(v_1, v_2), (v_2, v_3), \dots, (v_{k-1}, v_k)\}$. Легко бачити, що подібна структура графа не дозволяє розміщувати жодну пару вершин на одне й те саме місце в упорядкуванні незалежно від кількості вершин графа, чи їх вагових коефіцієнтів, тобто такий випадок не потребує розв'язання задачі упорядкування.

З іншого боку, гусениці — дерева, похідні від ланцюгів, мають структуру, яка допускає паралельне упорядкування вершин.

Означення 2.6. Гусеницею називається дерево, яке після вилучення з нього всіх вершин степені 1 перетворюється на ланцюг [107].

Слід зауважити, що збереження довжини ланцюга в результаті вилучення таких вершин не обов'язкове.

Проаналізуємо можливі варіанти структури гусениць залежно від способу, за яким ребра замінюються дугами [2]. Нехай гусениця утворена з орієнтованого ланцюга довжини k шляхом додавання до нього висячих вершин степенів 1, які з'єднуються вхідними дугами з вершинами ланцюга, утворюючи з ними кореневі піддерева (рис. 2.45).

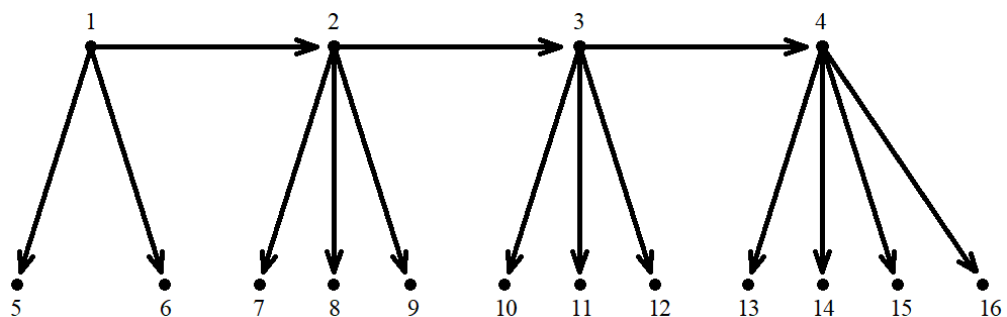


Рис. 2.45. Орієнтована гусениця із вихідними дугами

Для визначеності припустимо, що всі вершини мають одиничні вагові коефіцієнти. При побудові паралельного упорядкування на перші k місць розміщуються вершини заданого ланцюга. Решта вершин гусениці заносяться з урахуванням часткового порядку починаючи з тих, які приєднані до початку

ланцюга. У випадках, коли задана ширина $h \geq 2$ за умов, що в побудованому оптимальному упорядкуванні без переривань на останньому місці стоїть менше ніж h вершин, причому на двох останніх місцях стоять лише висячі вершини, значення цільової функції може бути покращено.

При упорядкуванні вершин графа з рис. 2.45 переривання зменшують довжину при $h = 2$.

$$h = 2: \begin{pmatrix} 1 & 2 & 6 & 7 & 8 & 10 & 12 & 14 & 16 \\ 5 & 3 & 4 & 9 & 11 & 13 & 15 \end{pmatrix}, l^* = 9;$$

$$h = 2: \begin{pmatrix} 1 & 2 & 6 & 7 & 8 & 10 & 12 & \{(14;0,5), (16;0,5)\} & (14;0,5) \\ 5 & 3 & 4 & 9 & 11 & 13 & 15 & (16;0,5) \end{pmatrix}, l_{II}^* = 8,5.$$

Наступним розглянемо схожий варіант структури орієнтованої гусениці, що відрізняється лише спрямованістю дуг, якими до орієнтованого ланцюга довжини k приєднані висячі вершини степенів 1.

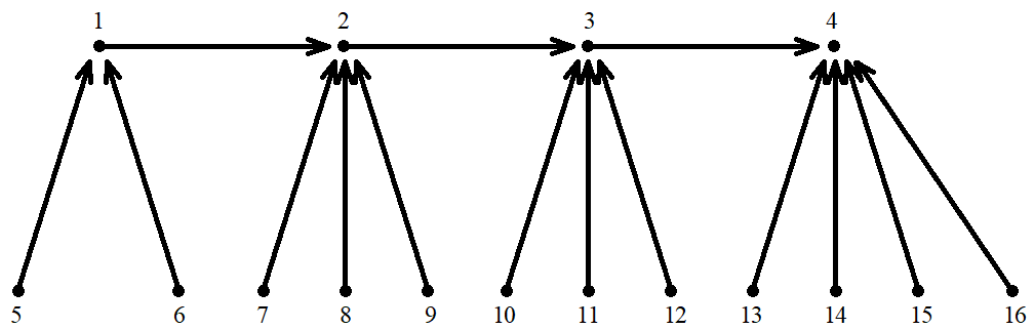


Рис. 2.46. Орієнтована гусениця із вхідними дугами

Для такої гусениці принцип, за яким буде утворюватися упорядкування, буде також подібним до попереднього випадку, однак розміщення вершин починається з кінця. Рухаючись з кінця до початку упорядкування, першими на k місць ставляться вершини ланцюга. Далі в тому ж напрямі з урахуванням часткового порядку заносяться допустимі висячі вершини, за можливості дозаповнюючи місця. Зменшення довжини упорядкування в такому випадку можливе, якщо з урахуванням заданої ширини $h \geq 2$ перші два місця упорядкування містять лише

вершини, що не мають вхідних дуг, окрім цього перше місце заповнене частково, тобто $|S[1]| < h$. Побудоване таким способом упорядкування вершин графа з рис. 2.46 при $h = 2$ має вигляд:

$$\begin{pmatrix} (5;0,5) & \{(5;0,5),(6;0,5)\} & 8 & 10 & 12 & 1 & 2 & 3 & 4 \\ (6;0,5) & 7 & 9 & 11 & 13 & 14 & 15 & 16 \end{pmatrix}.$$

Інший принцип побудови упорядкування для вершин такого графа полягає в занесенні на перші місця упорядкування вершин без вхідних дуг починаючи з тих, які приєднані до першої вершини ланцюга. Самі вершини ланцюга вносяться до упорядкування на наступні місця після того, як усі відповідні приєднані до них уже розподілені.

$$h=2: \begin{pmatrix} 5 & 1 & 8 & 2 & 11 & 3 & 14 & 16 & 4 \\ 6 & 7 & 9 & 10 & 12 & 13 & 15 \end{pmatrix}; h=3: \begin{pmatrix} 5 & 1 & 2 & 12 & 3 & 4 \\ 6 & 8 & 10 & 13 & 15 \\ 7 & 9 & 11 & 14 & 16 \end{pmatrix}.$$

Також розглянемо графи які утворені з орієнтованих зірок шляхом покрокового послідовного злиття нецентральної вершин, так що кожна наступна приєднана зірка має лише одну спільну вершину з попередньою. Приклад такого графа, що складається з поєднання вихідних зірок, зображено на рис. 2.47.

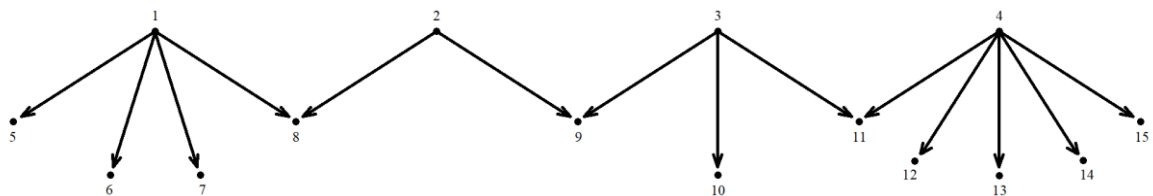


Рис. 2.47. Граф як об'єднання вихідних зірок

У випадку, коли граф, що має таку структуру, неорієнтований, він буде гусеницею.

При такій структурі графа побудова паралельного упорядкування її вершин відбувається наступним чином. На перші місця ставляться центральні вершини зірок, з яких вона складається, починаючи з тих, які мають більший степінь. Після цього вносяться решта вершин. Якщо при заданому $h \geq 2$ в побудованому таким способом упорядкуванні без переривань останнє місце заповнене частково, до того ж на останніх двох місцях відсутні центральні вершини зірок, то значення цільової функції можна зменшити за рахунок дозволу переривань.

$$h = 2: \begin{pmatrix} 4 & 3 & 5 & 7 & 9 & 11 & 13 & 15 \\ 1 & 2 & 6 & 8 & 10 & 12 & 14 \end{pmatrix}, l^* = 8;$$

$$h = 2: \begin{pmatrix} 4 & 3 & 5 & 7 & 9 & 11 & \{(13;0,5), (15;0,5)\} & (13;0,5) \\ 1 & 2 & 6 & 8 & 10 & 12 & 14 & (15;0,5) \end{pmatrix}, l_{\Pi}^* = 7,5.$$

Проаналізуємо, як зміниться довжина оптимальних упорядкувань вершин розглянутих графів, а також як вплине умова різних вагових коефіцієнтів на ефективність переривань. Припустимо, що вершини орієнтованого ланцюга довжини k мають довільні вагові коефіцієнти $p_1, p_2 \dots p_k$ такі, що $p_i \gg 1$ ($i = \overline{1, k}$). При приєднанні до довільних вершин v_i орієнтованого ланцюга додаткових вершин одиничної ваги v_r ($r > k$) шляхом введення відповідних дуг (v_i, v_r) одержуємо структуру графа з однією вершиною, що не має вхідних дуг. Приклад такого графа наведено на рис. 2.48.

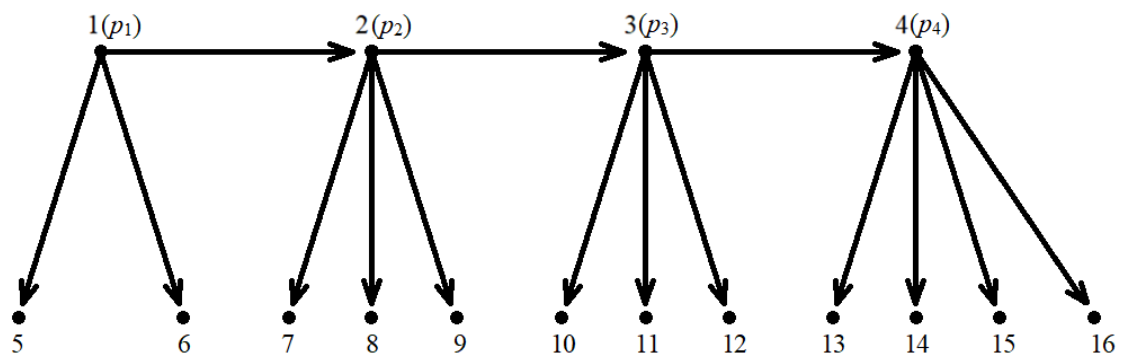


Рис. 2.48. Орієнтований ланцюг із вихідними дугами та неодиначними вагами вершин

При заборонених перериваннях оптимальне упорядкування його вершин ширини $h = 3$ має довжину $l^* = \sum_{i=1}^4 p_i + 2$ і будується за схемою з рис. 2.49.

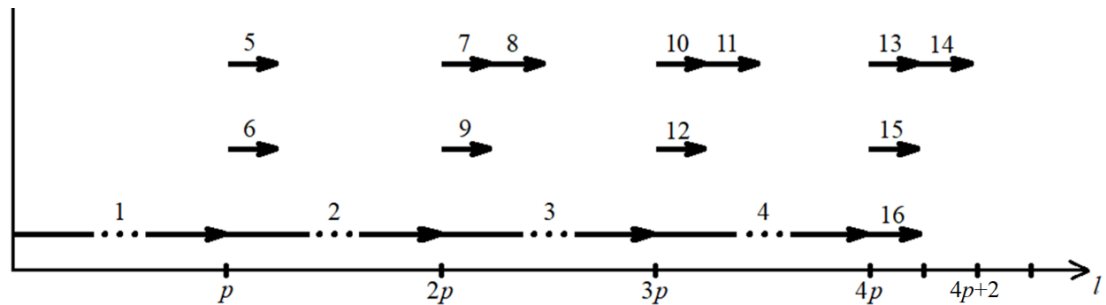


Рис. 2.49. Схема розподілу вершин гусениці з рис. 2.48 без переривань

Зменшення значення цільової функції за рахунок дозволу переривань у даному випадку є незначним, адже побудоване з їх урахуванням упорядкування має довжину $l_{\Pi}^* = \sum_{i=1}^4 p_i + 1\frac{1}{3}$, яка відрізняється від l^* менш, ніж на одиницю.

Враховуючи початкову умову $p_i \gg 1$, дійдемо висновку, що покращення є несуттєвим. В результаті аналогічних міркувань для випадку, коли спрямованість дуг, що з'єднують вершини v_r ($r > k$) із ланцюгом є зворотною, тобто маємо граф з однією вершиною, що не має вихідних дуг, оптимальне упорядкування з перериваннями так само несуттєво змінить значення цільової функції.

Наступним проаналізуємо випадок, коли не всі вершини мають однакові вагові коефіцієнти, однак на відміну від попередніх графів, вершини ланцюга мають ваги рівні 1, а приєднані до них — відповідно рівні p . На рис. 2.50 зображено структуру такого графа, причому вважається, що вершини 5-16 мають вагу p .

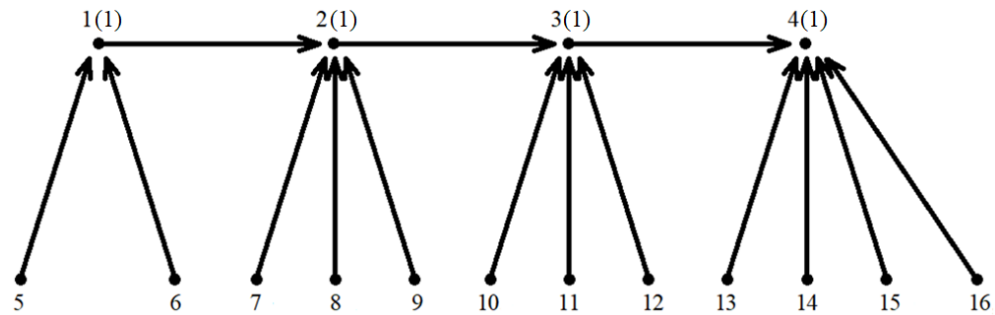


Рис. 2.50. Зважена вхідна гусениця

Побудоване упорядкування вершин такого графа при $h = 3$ та заборонених перериваннях має довжину $l^* = 4p + 2$. Зі схеми розподілу вершин стає очевидно, що за рахунок дозволу переривань домогтися скорочення довжини в даному разі неможливо, адже частково заповненим є лише останнє місце упорядкування, що містить останню вершину ланцюга.

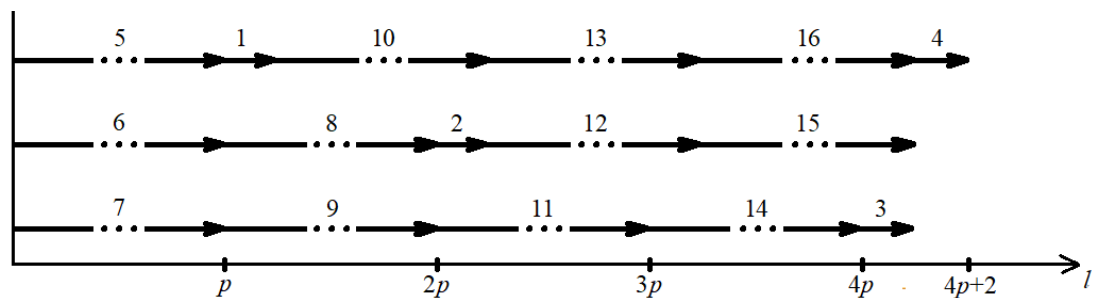


Рис. 2.51. Схема розподілу вершин гусениці з рис. 2.50 без переривань

Щоб виявити, чи є випадки, коли дозвіл переривань дасть суттєвий виграш значення цільової функції для гусениці, що має подібну структуру, розглянемо граф утворений з наведеного на рис. 2.50 шляхом приєднання чергової вершини v_r (з подальшою перенумерацією). Для визначеності, вважатимемо, що вона приєднана до вершини v_2 , що проілюстровано на рис. 2.52.

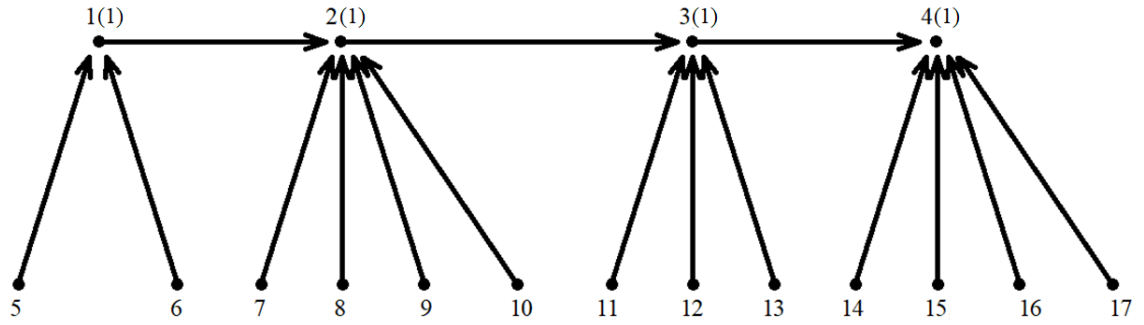


Рис. 2.52. Зважена гусениця із приєднаною вершиною

Упорядкування вершин такого графа без переривань за ширини $h = 3$ має наступну структуру, з якої видно, що кількість частково заповнених місць складає $(p+1)$:

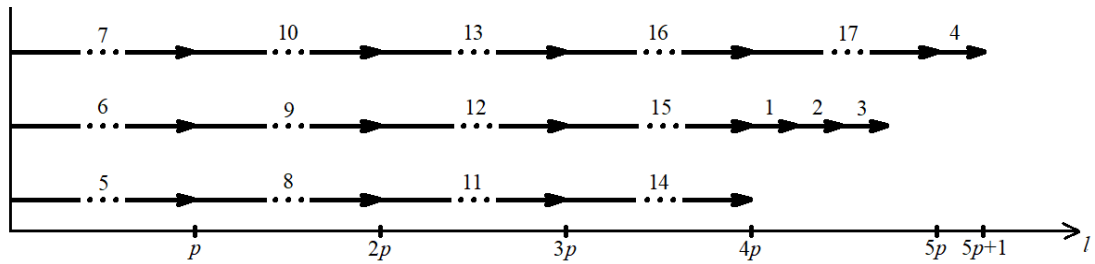


Рис. 2.53. Схема розподілу вершин гусениці з рис. 2.52 без переривань

Дозвіл переривань скорочує довжину упорядкування з $l^* = 5p + 1$ до $l_{\pi}^* = 4\frac{1}{3}p + 2$. Різниця між відповідними довжинами складає $\left(\frac{2}{3}p - 1\right)$, що свідчить про більш суттєвий вииграш у порівнянні з попереднім випадком. Структура оптимального упорядкування з перериваннями наведена на рис. 2.54.

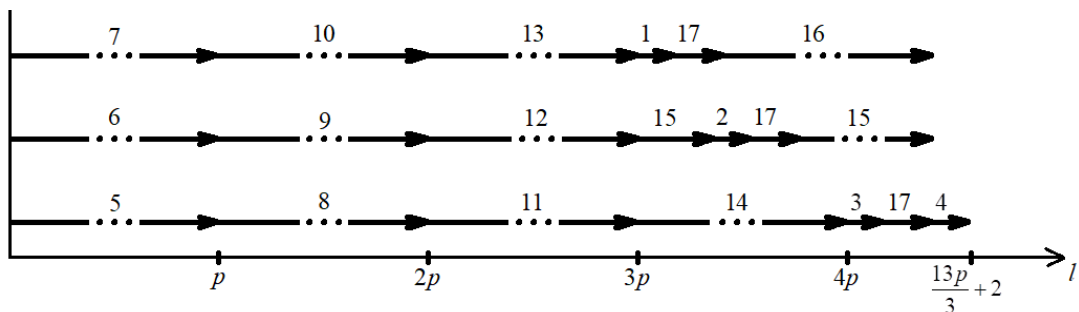


Рис. 2.54. Схема розподілу вершин гусениці з рис. 2.50 з перериваннями

При внесенні тих вершин, які приєднані до орієнтованого ланцюга, пріоритетність при виборі кожної наступної вершини для розміщення визначається довжиною найдовшого шляху, який починається в цій вершині.

Далі з'ясуємо, чи впливає на виграш від переривань спосіб, за яким до ланцюга приєднуються вершини v_r . Нехай більше половини таких вершин, приєднуються до останньої вершини ланцюга. Для визначеності розглянемо приклад, в якому вершини 1-3 утворюють орієнтований ланцюг, а більше половини вершин ваги p (наприклад 8-17), приєднані до однієї з них, тобто нерівномірно (рис. 2.55).

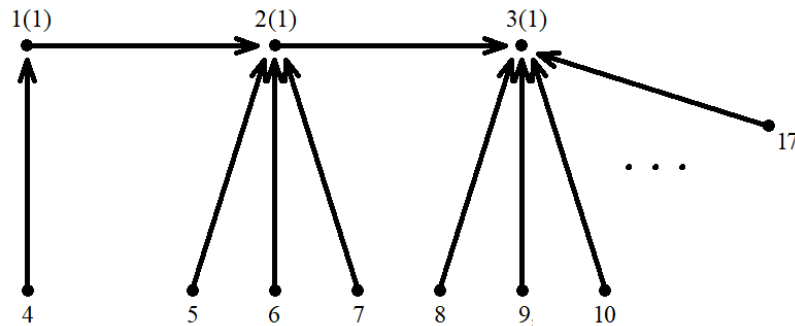


Рис. 2.55. Гусениця з перерозподіленими вершинами, приєднаними до ланцюга

Схематично оптимальне упорядкування вершин даного графа без переривань при ширині рівній 4 має довжину $l^* = 4p + 1$ і виглядає наступним чином (рис. 2.56).

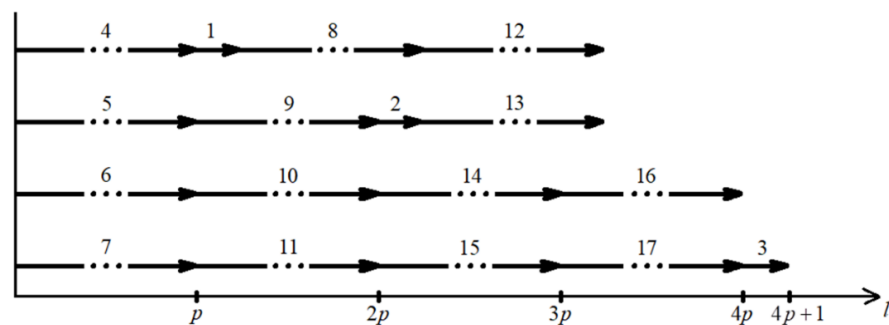


Рис. 2.56. Схема розподілу вершин гусениці з рис. 2.55 без переривань

Дозволивши переривання, ця довжина може бути скорочена до величини рівної $l_{\Pi}^* = 3\frac{1}{2}p + 1\frac{1}{2}$, тобто на значення $\left(\frac{1}{2}p - \frac{1}{2}\right)$.

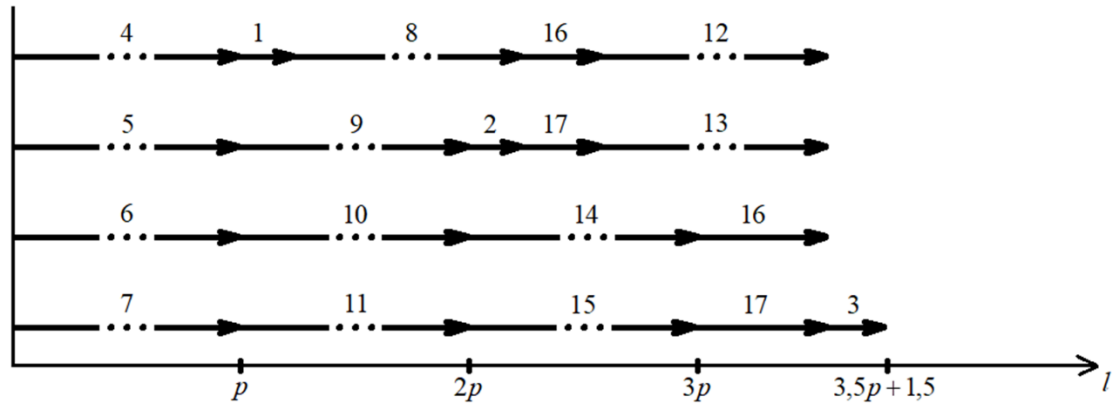


Рис. 2.57. Схема розподілу вершин гусениці з рис. 2.55 з перериваннями

Розглянуті приклади свідчать, що в загальному випадку множину вершин, які мають вагові коефіцієнти p можна розглядати незалежно від того, до яких саме вершин ланцюга вони приєднані. Це також означає, що розв'язавши одну задачу упорядкування для заданих підмножин вершин ланцюга з одиничними ваговими коефіцієнтами, та приєднаних вершин ваги p , одержані довжини упорядкувань будуть фіксованими для будь-якого способу приєднання. Розглянемо кількість можливих способів це зробити.

Якщо при приєднанні вершин ваги p суттєвим є врахування нумерації конкретних вершин, то кількість можливих варіантів приєднання до ланцюга вершин буде дорівнювати кількості відповідних сполучень. Представимо побудову графа у вигляді ітеративного процесу, при якому на кожному кроці до однієї з вершин ланцюга довжини k послідовно приєднується деяка кількість вершин ваги p . Нехай їх кількість для кожної i -ої вершини ланцюга складає m_i ($\forall i = \overline{1, k}; m_i \geq 1$), відповідно загальна їх кількість $\sum_{i=1}^k m_i = m$. Для простоти запису почнемо приєднання починаючи з k -ої вершини у зворотному порядку. На

першій ітерації серед m вершин обирається m_k , кількість варіантів цього вибору

складає $\frac{\left(\sum_{i=1}^k m_i\right)!}{m_k! \left(\sum_{i=1}^{k-1} m_i\right)!}$. На наступній ітерації серед вершин, які залишилися

обирається m_{k-1} , тобто кількість варіантів дорівнює $\frac{\left(\sum_{i=1}^{k-1} m_i\right)!}{m_{k-1}! \left(\sum_{i=1}^{k-2} m_i\right)!}$. Аналогічно для

усіх наступних ітерацій, аж до приєднання m_2 . На тому етапі, коли необхідно приєднати m_1 вершин, просто приєднується решта. Таким чином кількість способів приєднання вершин до орієнтованого ланцюга дорівнює

$$\sum_{j=0}^{k-2} \left(\frac{\left(\sum_{i=1}^{k-j} m_i\right)!}{m_{k-j}! \left(\sum_{i=1}^{k-j-1} m_i\right)!} \right).$$

Якщо при приєднанні вершин до ланцюга їх нумерація є несуттєвою, то потрібно враховувати лише можливу кількість вершин, які з'єднуються з відповідними вершинами ланцюга.

Для графа, який має структуру гусениці, сформулюємо умови, при яких переривання дозволяють зменшити значення цільової функції у вигляді наступної теореми.

Теорема 2.1. Дозвіл переривань при побудові оптимального упорядкування може зменшити значення цільової функції якщо:

1) множина вершин V ($|V| = n$) розбивається на дві неперетинні підмножини V_1 та V_2 ($|V_1| = n_1$, $|V_2| = n_2$, де $n_1 + n_2 = n$), $w_i \in \{c, p\}$ — вагові коефіцієнти вершин (де $i = \overline{1, n}$ та $p \gg c$), причому

$$\begin{cases} w_i = c, \text{ якщо } v_i \in V_1 \\ w_i = p, \text{ якщо } v_i \in V_2 \end{cases};$$

2) вершини підмножини V_1 утворюють орієнтований ланцюг;

3) $\forall v_j \in V_2 \exists! (v_j, v_k) \in U$, де $v_k \in V_1$;

$$4) \quad |V_2| > h \text{ та } \frac{|V_2|}{h} \notin N.$$

При цьому $l^* - l_{II}^* > \frac{(h-q)}{h} p$, де q — остача від ділення $|V_2|$ на h .

Доведення. Оцінимо значення цільової функції в разі, якщо переривання заборонені. Неважко переконатися в тому, що $\left\lceil \frac{|V_2|}{h} \right\rceil p + c \leq l^* \leq \left\lceil \frac{|V_2|}{h} \right\rceil p + |V_1|c$.

Дійсно, незалежно від того, до яких вершин орієнтованого ланцюга прямують дуги з вершин, що належать підмножині V_2 , завжди знайдеться спосіб їх розміщення в упорядкуванні, при якому повністю будуть заповнені $\left\lceil \frac{|V_2|}{h} \right\rceil p$ місць, а частково заповненими — відповідно ще p місць (за рахунок умови 4). З іншого боку, якщо наприклад всі вершини підмножини V_2 приєднані до першої вершини ланцюга, то вся підмножина вершин V_1 може бути розміщена в упорядкуванні лише послідовно після них, займаючи $|V_1|c$ місць. У разі, якщо до останньої вершини ланцюга приєднано не менше q вершин і $(|V_1|-1)c \leq p$, то на p частково заповнених місць можна розмістити перші $(|V_1|-1)$ вершини ланцюга, а відповідно останню — на місця безпосередньо після внесення решти вершин графа.

У випадку дозволених переривань оцінка довжини має наступний вигляд $\frac{|V_2|}{h} p + c \leq l^* \leq \frac{|V_2|}{h} p + |V_1|c$ з міркувань, аналогічних до випадку заборонених переривань. Єдиною відмінністю є те, що нижня оцінка досяжна за умови, коли до останньої вершини ланцюга приєднано не менше $(h+q)$ вершин. Знайдемо різницю між відповідними значеннями довжини упорядкувань. Для оцінки знизу:

$$l^* - l_{II}^* = \left\lceil \frac{|V_2|}{h} \right\rceil p + c - \left(\frac{|V_2|}{h} p + c \right) = \left\lceil \frac{|V_2|}{h} \right\rceil p - \frac{|V_2|}{h} p = \left(\left\lceil \frac{|V_2|}{h} \right\rceil - \frac{|V_2|}{h} \right) p.$$

Для оцінки зверху:

$$l^* - l_{\pi}^* = \left\lceil \frac{|V_2|}{h} \right\rceil p + |V_1|c - \left(\left\lceil \frac{|V_2|}{h} \right\rceil p + |V_1|c \right) = \left\lceil \frac{|V_2|}{h} \right\rceil p - \frac{|V_2|}{h} p = \left(\left\lceil \frac{|V_2|}{h} \right\rceil - \frac{|V_2|}{h} \right) p.$$

Тобто різниця буде однаковою в обох випадках. Враховуючи умову 4 представимо $|V_2|$ у вигляді $(mh + q)$, де $m \in N$. Тоді

$$\left(\left\lceil \frac{|V_2|}{h} \right\rceil - \frac{|V_2|}{h} \right) p = \left(\left\lceil \frac{mh + q}{h} \right\rceil - \frac{mh + q}{h} \right) p = \left(m + \left\lceil \frac{q}{h} \right\rceil - m - \frac{q}{h} \right) p = \left(\left\lceil \frac{q}{h} \right\rceil - \frac{q}{h} \right) p.$$

Оскільки q — остача від ділення на h , то очевидно, що $\frac{q}{h} < 1$ і відповідно

$\left\lceil \frac{q}{h} \right\rceil = 1$. З цього випливає, що

$$l^* - l_{\pi}^* = \left(1 - \frac{q}{h} \right) p = \left(\frac{h - q}{h} \right) p.$$

Зрештою показано, що в разі виконання наведених умов дозвіл переривань може зменшити значення цільової функції, більше того, на величину, яку можна визначити апіорно, адже вона залежить тільки від початкових даних.

2.5 Висновки до розділу

Таким чином було досліджено вплив дозволу переривань на оптимальність цільової функції в задачі упорядкування при дозволених перериваннях, проаналізовано залежність цього впливу від структури відповідного орграфа та початкових даних.

Введено оцінку, за якою обчислювався виграш значення цільової функції, який підтверджує доцільність переривань, причому ця оцінка отримується апіорно.

Для випадку, коли граф складається лише з ізолюваних вершин ($U = \emptyset$), виявлено, що при різних вагових коефіцієнтах вершин, застосування переривань дає менший виграш порівняно з аналогічним графом, в якому всі ваги вершин рівні. Також було показано, що верхнє граничне значення можливого виграшу

складає 50% за умови, коли задана ширина упорядкування є на 1 меншою за кількість вершин.

Для повних дводольних графів було проаналізовано випадки різних співвідношень потужностей вершин відповідних долей. У випадку, коли потужності однакові, виявлено, що переривання можуть давати виграш близький до 50% за умови, коли кількість вершин в кожній долі на 1 більша заданої ширини упорядкування. Також показано, що в результаті вилучення лише однієї випадкової дуги з такого графа виграш від переривань знижується. Для випадку різних потужностей сформульовано та доведено твердження, що значення виграшу буде однаковим для випадків $K_{k,r}$ та $K_{r,k}$, k та r — кількості вершин у відповідних долях. Проаналізовано вплив початкових даних на виграш і з'ясовано, що на ефективність переривань при розподілі вершин однієї долі впливає не безпосередня кількість цих вершин, а частка та остача від ділення націло цього числа на значення заданої ширини упорядкування.

Дослідження впливу застосування переривань у випадках, коли граф задачі є паралельно-послідовним показало, що виграш безпосередньо залежить від співвідношення вагових коефіцієнтів вершин, з'єднаних послідовно, та паралельно. Виграш в найгіршому випадку відсутній, тоді як в найкращому — сягає 50%.

Було розглянуто випадки, коли граф задачі має спеціальну структуру, для якої доведено, що вона допускає граціозну розмітку. Першими з таких графів досліджувалися дерева спеціального вигляду — зірки.

Для зірок з одиничними вагами вершин аналіз впливу від дозволу переривань на значення цільової функції показав, що залежно від спрямованості дуг виграш може сягати верхнього граничного значення 33,(3)% у випадках вхідної чи вихідної зірки, та 40% у комбінованому випадку.

Іншим підкласом дерев, для яких доведена можливість граціозної розмітки, є оливкові. При аналізі графів, що належать цьому підкласу, був застосований перехід до випадку зірок з різними вагами нецентральных вершин. Для них показано, що апріорне визначення точного значення виграшу від

переривань значно ускладнене в порівнянні з випадком рівних вагових коефіцієнтів, в результаті чого розглядався мінімальний гарантований виграш від дозволу переривань. В загальному випадку він не дає суттєвого покращення значення цільової функції незалежно від того, чи є відповідна зірка вхідною, вихідною або комбінованою. Порівняння мінімального гарантованого виграшу із тим, який було одержано в результаті обчислювального експерименту, виявило доцільність уточнення його оцінки. Запропоновано та сформульовано у вигляді двох тверджень способів, за яким довжина паралельного упорядкування без переривань (а отже й оцінка виграшу) може бути апіорно уточненою.

Аналіз впливу переривань був проведений для випадків, коли відповідний граф задачі має структуру одностантного дерева (гусениці). Досліджено доцільність застосування переривань в залежності від наступних характеристик такого графу: будови; спрямованості дуг; значення вагових коефіцієнтів вершин; способу розподілу вершин, що приєднані до ланцюга, який утворює гусеницю.

З використанням комбінаторних конфігурацій визначено кількість можливих способів приєднання висячих вершин до ланцюга. Сформульовано та доведено теорему, яка визначає умови, за яких дозвіл переривань може зменшити значення цільової функції для графів, що мають структуру орієнтованої гусениці.

Основні результати розділу опубліковано в роботах [2-5,9,10,12,13].

Розділ 3. ДЕЯКІ УЗАГАЛЬНЕННЯ ПІДХОДІВ ДО АНАЛІЗУ ЗАДАЧ З ПЕРЕРИВАННЯМИ

3.1 Вплив переривань на оптимальність в одній узагальненій задачі упорядкування

Одним з важливих питань при дослідженні задач паралельного упорядкування є аналіз випадків виникнення аномалій — ситуацій, за яких послаблення початкових умов призводить до неочікуваного погіршення значення цільової функції [109]. Як правило, аномалії можуть виникати у випадках, коли окрім обмежень на порядок виконання робіт задача має деякі додаткові обмеження. Такі обмеження зокрема накладаються за допомогою списку пріоритетів.

Означення 3.1. Списком пріоритетів L називається послідовність (v_1, \dots, v_n) вершин графа $G = (V, U)$, $|V| = n$, згідно з якою, якщо вершина v_j стоїть в L правіше за v_k , то в упорядкуванні v_j розміщується на місцях, що не передують розміщенню вершини i_k ($j, k = \overline{1, n}$).

Одне з узагальнень при постановці задач упорядкування, за якого також можуть виникати аномальні випадки, передбачає, що замість ширини упорядкування h в початкових умовах задана множина значень $h_i \in N$ таких, що $2 \leq h_i \leq n$, $|S[i]| \leq h_i$, $1 \leq i \leq n$.

Опишемо умови, за яких в задачі упорядкування можна уникнути виникнення аномалій, пов'язаних з наявністю наведених специфічних обмежень, за рахунок дозволу переривань:

- список пріоритетів має бути побудований згідно з алгоритмом побудови оптимального списку пріоритетів [110,111]; роботи з високим пріоритетом можуть переривати виконання робіт з нижчим пріоритетом, що гарантує виконання більш важливих із них першочергово;

- виконання перерваної роботи продовжується пізніше, але узгоджується зі списком пріоритетів, що дозволяє розміщувати більш пріоритетні роботи на крайніх лівих допустимих місцях упорядкування [112];

- у будь-який момент виконання роботи можливе її переривання та перерозподіл, що дозволяє враховувати зміни кількості доступних виконавців та прогрес виконання робіт в реальному часі.

У разі виконання цих умов використання алгоритму пріоритетного динамічного перерозподілу дозволяє ефективно призначати виконання робіт виконавцям, враховуючи пріоритети та можливість переривання робіт. Це мінімізує загальний час виконання і запобігає виникненню аномалій, які можуть з'являтися внаслідок зміщення виконання більш пріоритетних робіт менш пріоритетними.

Для задач без наявного списку пріоритетів проаналізуємо, який вплив матиме дозвіл переривань на значення цільової функції при описаному узагальненні, якщо технологічні обмеження задані оргграфом однієї зі структур, розглянутої в попередньому розділі.

3.1.1 Узагальнення в задачі без наявності часткового порядку

Нехай граф задачі задано множиною ізольованих вершин, тобто $G = (V, U)$, де $|V| = n$, $U = \emptyset$. Спершу розглянемо випадок, в якому вагові коефіцієнти всіх вершин одиничні.

Якщо переривання заборонені, то вершини графа розміщуються в упорядкуванні за наступним алгоритмом.

Алгоритм 3.1.

Крок 1. Всі місця в упорядкуванні S вважаються пустими.

Крок 2. Приймається $k = 1$. На перше місце упорядкування розміщуються h_k вершин. Якщо $h_k = n$, то кінець алгоритму, $l^* = 1$.

Крок 3. Приймається $k = k + 1$.

Крок 4. Якщо $n - \sum_{i=1}^{k-1} |S[i]| \geq h_k$, то в множину $S[k]$ заносяться довільні h_k

вершин, інакше заносяться решта $n - \sum_{i=1}^{k-1} |S[i]|$ вершин.

Крок 5. Якщо $\sum_{i=1}^k |S[i]| = n$, то кінець алгоритму, $l^* = k$. Інакше перехід на крок 3.

У випадку, коли переривання дозволені, слід розглянути два випадки. Якщо $\exists k' \in N : \sum_{i=1}^{k'} h_i = n$, $1 \leq k' \leq \left\lceil \frac{n}{2} \right\rceil$, то переривання будуть недоцільними, для побудови упорядкування пропонується застосовувати алгоритм 3.1. Інакше застосовуємо наступний алгоритм побудови упорядкування.

Алгоритм 3.2.

Крок 1. Всі місця в упорядкуванні S вважаються пустими.

Крок 2. Приймається $k = 1$.

Крок 3. Якщо $\sum_{i=1}^{k+1} h_i > n$, то $l_{II}^* = k + \frac{\left(n - \sum_{i=1}^k h_i\right)}{h_{k+1}}$ та перехід на крок 5.

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

Крок 5. Приймається $j = 1$, $c_j = 0$.

Крок 6. Визначається підмножина $H_{(j)} \subseteq \{h_1, h_2, \dots, h_{k+1}\}$ 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 упорядкуванні заносяться вершини в такому порядку, за необхідності застосовуючи розбиття:

- 1) та вершина, для якої $c_j > 0$, якщо $H_{(j)} \neq \{h_{k+1}\}$;
- 2) $|H_{(j)}|$ вершин якщо $c_j + |H_{(j)}| < l_{II}^*$, інакше $|H_{(j)}| - 1$ вершину;

3) якщо $c_j + |H_{(j)}| > l_{\Pi}^*$, то r -а вершина розбивається на (r, t) та $(r, 1-t)$ так, що $c_j + |H_{(j)}| - (r, 1-t) = l_{\Pi}^*$, при цьому до (r, t) теж може бути застосоване розбиття.

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

- 1) $v' \notin S[k+1]$;
- 2) $(v', t') \in S[i]$, де $i < k+1$, $c_j \leq t' \leq 1$.

В упорядкуванні (v', t') замінюється на $\{(v, c_j), (v', t' - c_j)\}$, а на позицію $(k+1)$ ставиться (v', c_j) .

Крок 9. Якщо $j < \max_i h_i$ то $j = j + 1$, $c_j = 1 - t$ та перехід на крок 6, інакше кінець алгоритму.

Застосуємо алгоритм на прикладі.

Приклад 3.1. Для графа G_8 (рис. 3.1), $n = 12$, та заданої послідовності h_i (3, 5, 3, 4, 3) знайти упорядкування мінімальної довжини.

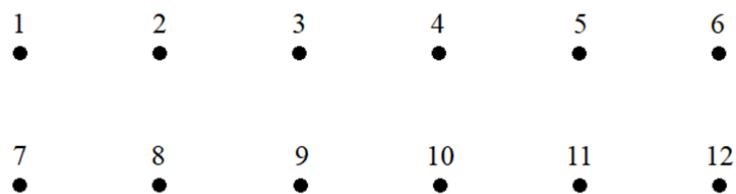


Рис. 3.1. Граф G_8

Побудуємо спочатку упорядкування для випадку, коли переривання заборонені, отримаємо упорядкування

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

Довжина цього упорядкування $l^* = 4$.

Тепер побудуємо упорядкування за умови, що переривання дозволені.

При k рівних 1 та 2 умова кроку 3 алгоритму 3.2 не виконується, але починає виконуватися при $k = 3$: $\sum_{i=1}^{k+1} h_i = 3 + 5 + 3 + 4 = 15 > 12$, отримуємо значення

$$l_{\pi}^* = 3 + \frac{12-11}{4} = 3\frac{1}{4}.$$

Приймаємо $j = 1$, $c_1 = 0$. Визначаємо $H_{(1)} = \{h_1, h_2, h_3, h_4\} = \{3, 5, 3, 4\}$ та вносимо 4 вершини в упорядкування:

$$\left((1;1) (2;1) (3;1) \left(4; \frac{1}{4} \right) \right).$$

Тут виконання роботи, що відповідає вершині 4 розбивається на дві складові $\left(4; \frac{1}{4} \right)$ та $\left(4; \frac{3}{4} \right)$, які в упорядкуванні стоять на різних місцях.

Переходимо до $j = 2$. Значення $c_2 = \frac{3}{4}$, $H_{(2)} = \{h_1, h_2, h_3, h_4\} = \{3, 5, 3, 4\}$, в упорядкування вноситься $\left(4; \frac{3}{4} \right)$ та ще три вершини:

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

Далі $j = 3$, $c_3 = \frac{2}{4}$, $H_{(3)} = \{h_1, h_2, h_3, h_4\} = \{3, 5, 3, 4\}$, вносимо в упорядкування залишок розбиття $\left(7; \frac{2}{4} \right)$ і ще 3 вершини:

$$\left(\begin{array}{ccc} (1;1) & (2;1) & (3;1) \\ \left\{ \left(4; \frac{3}{4} \right), \left(5; \frac{1}{4} \right) \right\} & \left\{ \left(5; \frac{3}{4} \right), \left(6; \frac{1}{4} \right) \right\} & \left\{ \left(6; \frac{3}{4} \right), \left(7; \frac{1}{4} \right) \right\} \\ \left\{ \left(7; \frac{1}{2} \right), \left(8; \frac{1}{2} \right) \right\} & \left\{ \left(8; \frac{1}{2} \right), \left(9; \frac{1}{2} \right) \right\} & \left\{ \left(9; \frac{1}{2} \right), \left(10; \frac{1}{2} \right) \right\} \end{array} \left(4; \frac{1}{4} \right) \right).$$

При $j = 4$ маємо $c_4 = \frac{1}{4}$ та $H_{(4)} = \{h_2, h_4\} = \{5, 4\}$. До упорядкування вносяться $\left(10; \frac{1}{4}\right)$ та ще одна вершина:

$$\left(\begin{array}{ccc} & (2;1) & \\ (1;1) & \left\{ \left(5; \frac{3}{4}\right), \left(6; \frac{1}{4}\right) \right\} & (3;1) \\ \left\{ \left(4; \frac{3}{4}\right), \left(5; \frac{1}{4}\right) \right\} & & \left\{ \left(6; \frac{3}{4}\right), \left(7; \frac{1}{4}\right) \right\} \\ \left\{ \left(7; \frac{1}{2}\right), \left(8; \frac{1}{2}\right) \right\} & \left\{ \left(8; \frac{1}{2}\right), \left(9; \frac{1}{2}\right) \right\} & \left\{ \left(9; \frac{1}{2}\right), \left(10; \frac{1}{2}\right) \right\} \\ & \left\{ \left(10; \frac{1}{4}\right), \left(11; \frac{3}{4}\right) \right\} & \end{array} \right) \begin{pmatrix} \left(4; \frac{1}{4}\right) \\ \left(7; \frac{1}{4}\right) \\ \left(10; \frac{1}{4}\right) \\ \left(11; \frac{1}{4}\right) \end{pmatrix}.$$

Тепер коли $j = 5$ значення $c_5 = 0$, $H_{(5)} = \{h_2\} = \{5\}$, упорядкування добудовується шляхом внесення останньої вершини на місце 2:

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

Отже, для випадків заборонених та дозволених переривань побудовані оптимальні паралельні упорядкування вершин графа G_8 із довжинами відповідно $l^* = 4$ та $l_{\Pi}^* = 3\frac{1}{4}$.

Розглянемо застосування алгоритму для побудови упорядкування вершин того ж графа, але при інших заданих h_i (3, 4, 3, 5, 3). Довжина оптимального упорядкування без переривань буде такою ж, як і в попередньому прикладі, а довжина оптимального упорядкування з перериваннями становитиме $l_{\Pi}^* = 3\frac{2}{5}$.

Детальніше зупинимось на тому етапі алгоритму, де при $j = 4$ занесені вершини у $H_{(4)} = \{h_2, h_4\} = \{4, 5\}$. Частково заповнене упорядкування матиме наступний вигляд:

$$\left(\begin{array}{cccc} & (2;1) & & \left(4; \frac{2}{5}\right) \\ (1;1) & \left\{\left(4; \frac{3}{5}\right), \left(5; \frac{2}{4}\right)\right\} & \left\{\left(5; \frac{3}{5}\right), \left(6; \frac{2}{5}\right)\right\} & \left(7; \frac{2}{5}\right) \\ \left\{\left(4; \frac{3}{5}\right), \left(5; \frac{2}{4}\right)\right\} & \left\{\left(8; \frac{1}{5}\right), \left(9; \frac{4}{5}\right)\right\} & \left\{\left(6; \frac{3}{5}\right), \left(7; \frac{2}{5}\right)\right\} & \left\{\left(10; \frac{1}{5}\right), \left(11; \frac{1}{5}\right)\right\} \\ \left\{\left(7; \frac{1}{5}\right), \left(8; \frac{4}{5}\right)\right\} & \left\{\left(11; \frac{4}{5}\right), \left(12; \frac{1}{5}\right)\right\} & \left\{\left(9; \frac{1}{5}\right), \left(10; \frac{4}{5}\right)\right\} & \left(12; \frac{2}{5}\right) \end{array} \right).$$

При переході до $j = 5$ маємо $c_5 = \frac{2}{5}$, $H_{(5)} = \{h_4\} = \{5\}$, однак при цьому виконуються умови 8 кроку. Через це починаємо переглядати елементи упорядкування починаючи з першого місця, шукаючи такий, який матиме вигляд (v', t') такий, що $v' \notin S[4]$ і $t' \geq c_5$. Таким елементом є $(1; 1)$. Проводячи відповідне розбиття та внесення до упорядкування його елементи, отримуємо:

$$\left(\begin{array}{cccc} & (2;1) & & \left(4; \frac{2}{5}\right) \\ \left\{\left(1; \frac{3}{5}\right), \left(12; \frac{2}{5}\right)\right\} & \left\{\left(5; \frac{3}{5}\right), \left(6; \frac{2}{5}\right)\right\} & (3;1) & \left(7; \frac{2}{5}\right) \\ \left\{\left(4; \frac{3}{5}\right), \left(5; \frac{2}{5}\right)\right\} & \left\{\left(8; \frac{1}{5}\right), \left(9; \frac{4}{5}\right)\right\} & \left\{\left(6; \frac{3}{5}\right), \left(7; \frac{2}{5}\right)\right\} & \left\{\left(10; \frac{1}{5}\right), \left(11; \frac{1}{5}\right)\right\} \\ \left\{\left(7; \frac{1}{5}\right), \left(8; \frac{4}{5}\right)\right\} & \left\{\left(11; \frac{4}{5}\right), \left(12; \frac{1}{5}\right)\right\} & \left\{\left(9; \frac{1}{5}\right), \left(10; \frac{4}{5}\right)\right\} & \left(12; \frac{2}{5}\right) \\ & & & \left(1; \frac{2}{5}\right) \end{array} \right).$$

Означення 3.2. Місце упорядкування $S[i]$ є щільно заповненим, якщо

$$\sum_{(v_k, c_j) \in S[i]} c_j = h_i, \text{ де } i, k \in \{1, 2, \dots, n\}, 0 < c_j \leq 1.$$

Означення 3.3. Упорядкування S будемо називати заповненим щільно, якщо всі його непорожні місця щільно заповнені.

Означення 3.4. Упорядкування S будемо називати заповненим майже щільно, якщо серед його непорожніх місць нещільно заповнене лише останнє.

Твердження 3.1. Алгоритм 3.2 для задачі $S(G, h_i, l)$ де граф $G = (V, U)$ такий, що $|V| = n$, $U = \emptyset$, є точним для випадків, коли переривання дозволені.

Доведення. Щоб довести твердження достатньо показати, що непорожні місця з першого по k -те в упорядкуванні заповнені щільно, а на $(k+1)$ -ому місці розташовано решту вершин, кількість яких менша за h_{k+1} . За алгоритмом 3.2 довжина оптимального паралельного упорядкування при дозволених

перериваннях становить $l_{II}^* = k + \frac{\left(n - \sum_{i=1}^k h_i\right)}{h_{k+1}}$. Тут перший доданок означає, що

місця з 1 по k заповнені щільно, тобто там можуть бути розміщені $\sum_{i=1}^k h_i$ вершин.

За умовою $\sum_{i=1}^k h_i < n < \sum_{i=1}^{k+1} h_i$, тобто залишається ще $n - \sum_{i=1}^k h_i$ вершин, кількість яких менша за h_{k+1} які й будуть поставлені на $(k+1)$ місце. Таким чином показано, що побудоване за алгоритмом 3.2 упорядкування є оптимальним, а сам алгоритм є точним.

Твердження 3.2. Умова $h_i \leq n$, де $1 \leq i \leq n$ є достатньою для того, щоб твердження 3.1 було істинним.

Доведення. Припустимо, що дана умова виконується. Покажемо, що побудоване за алгоритмом 3.2 упорядкування є оптимальним, тобто всі непорожні місця $1 \leq i \leq k$ в ньому щільно заповнені, а місце $(k+1)$ заповнене частково, і його заповненість визначається константою c , де

$$c = \frac{\left(n - \sum_{i=1}^k h_i\right)}{h_{k+1}}.$$

Очевидно, що перше місце завжди буде щільно заповненим, що безпосередньо впливає з факту виконання умови твердження. Місця з другого по k -те також заповнені щільно, бо за умовою алгоритму $\sum_{i=1}^k h_i < n$. Залишилося показати, що місце $(k+1)$ заповнене на константу c . Для $2 \leq h_{k+1} \leq n$ завжди можна провести розбиття деякої підмножини вершин так, щоб це місце було заповнене на вказану величину, причому при $h_{k+1} = n$ на цьому місці будуть розташовані

частини розбиття усіх вершин $\left(1, \frac{1}{n}\right), \left(2, \frac{1}{n}\right) \dots \left(n, \frac{1}{n}\right)$. Якщо виконується нерівність $h_{k+1} > n$, яка не входить в умову твердження, це місце неможливо заповнити на величину c , адже не залишиться вершин, частини розбиття яких можна було б використати. Таким чином, твердження доведено.

3.1.2 Аналіз впливу переривань для дводольних графів

Розглянемо випадки, коли для розглянутого узагальнення задачі граф є регулярним і має структуру повного дводольного, тобто $G = K_{m,r}$ де $m + r = n$. Як було зазначено в 2.2 в цьому випадку вершини першої долі графа можуть бути занесені до упорядкування так, як і ізольовані. Після цього аналогічним чином до упорядкування заносяться вершини другої долі.

Алгоритм, за яким будуть заноситися до упорядкування вершини першої долі графа, залежить від виконання наступної умови

$$\exists k : \sum_{i=1}^k h_i = m. \quad (3.1)$$

Якщо (3.1) виконується для підмножини вершин першої долі, то усі вони можуть бути занесені в довільному порядку до упорядкування, щільно заповнюючи перші k його місць без потреби в застосуванні переривань. Для цього достатньо скористатися алгоритмом 3.1, вважаючи, що замість n маємо m вершин. Після цього залишиться внести до упорядкування решту r вершин, які належать другій долі графа. Перш ніж приступити до внесення, перевіряється виконання аналогічної умови, однак враховується, що перші k місць уже щільно заповнені. Ця умова має наступний вигляд:

$$\exists p : \sum_{i=k+1}^{k+p} h_i = r. \quad (3.2)$$

Якщо (3.2) виконується, то усі вершини другої долі займатимуть в упорядкуванні місця з $(k+1)$ -го по p , які також будуть заповнені щільно, не потребуючи при цьому переривань. В разі, якщо (3.2) не виконується, дозвіл на переривання може зменшити довжину упорядкування. Для занесення вершин

другої долі можна застосувати алгоритм 3.2, якщо вважати місце під номером $(k+1)$ першим, а кількість вершин рівною r .

У разі, якщо для вершин першої долі умова (3.1) не виконується, то дану підмножину можна занести до упорядкування за алгоритмом 3.2, приймаючи n рівним m . Тоді місця упорядкування з першого по k -те будуть заповнені щільно, а місце $(k+1)$ — частково, на величину:

$$c = \frac{m - \sum_{i=1}^k h_i}{h_{k+1}}, \quad (3.3)$$

де $0 < c < 1$. При цьому очевидно при виконанні робіт, що відповідають вершинам цієї долі, переривання будуть доречними (не обов'язково для всіх вершин). Для розподілу решти вершин перевірка умови (3.2) не потрібна.

У випадках, коли для заданого графа $K_{m,r}$ не виконується (3.1) побудова оптимального упорядкування здійснюється за наступною модифікацією алгоритма 3.2.

Алгоритм 3.3.

Крок 0. Вважається, що вершини першої долі графа занесені в упорядкування S' за алгоритмом 3.2 і значення c обчислене за (3.3).

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} визначено на нульовому кроці з упорядкування S' .

Крок 2. $p = 1$.

Якщо виконується умова

$$(\min(h_1, r))(1-c) + \sum_{i=2}^{p+1} h_i \geq r \quad (3.4)$$

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

на крок 4.

Крок 3. Якщо виконується умова (3.4), то

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

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 $H_{(j)} \subset \{h_1, h_2, \dots, h_{p+1}\}$ 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:

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

5) $|H_{(j)}|$ вершин якщо $c_j + |H_{(j)}| < l_{\pi}^*$, інакше $|H_{(j)}| - \lfloor (l_{\pi}^* - c_j) \rfloor$ вершин;

6) якщо $c_j + |H_{(j)}| > l_{\pi}^*$, то обрана до занесення в упорядкування вершина q розбивається на часові інтервали t та $(1 - t)$ так, що $c_j + (|H_{(j)}| - 1) - (1 - t) = l_{\pi}^*$, 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:

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

4) $(v', t') \in S$, де $c_j \leq t' \leq 1$.

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 $j < \max_i h_i$ то $j = j + 1$, $c_j = t$ та перехід на крок 6, інакше формуємо упорядкування S^* , де

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

Кінець алгоритму.

Наведемо приклад розв'язання задачі із застосуванням цієї модифікації алгоритму 3.2.

Приклад 3.2. Для повного дводольного графа $K_{8,6}$, та заданої послідовності h_i (4, 6, 5, 3, 3) знайти упорядкування вершин, яке має мінімальну довжину [1].

Для визначеності вважатимемо, що вершинам присвоєні номери від 1 до 14 починаючи з тих, що належать першій долі. Якщо переривання заборонені, то побудоване оптимальне упорядкування вершин має вигляд:

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

Його довжина дорівнює 4. Коли переривання дозволені, спочатку будемо упорядкування S' , що включає вершини першої долі, як описано у кроці 0 модифікованого алгоритму 3.3:

$$\left((1;1) \left(2; \frac{2}{3} \right) \right) \Rightarrow \left(\begin{matrix} (1;1) & \left(2; \frac{2}{3} \right) \\ \left\{ \left(2; \frac{1}{3} \right), \left(3; \frac{2}{3} \right) \right\} & \left\{ \left(3; \frac{1}{3} \right), \left(4; \frac{1}{3} \right) \right\} \end{matrix} \right) \Rightarrow \left(\begin{matrix} (1;1) & \left(2; \frac{2}{3} \right) \\ \left\{ \left(2; \frac{1}{3} \right), \left(3; \frac{2}{3} \right) \right\} & \left\{ \left(3; \frac{1}{3} \right), \left(4; \frac{1}{3} \right) \right\} \\ \left\{ \left(4; \frac{2}{3} \right), \left(5; \frac{1}{3} \right) \right\} & \left(5; \frac{2}{3} \right) \end{matrix} \right) \Rightarrow$$

$$\begin{aligned}
& \Rightarrow \left(\begin{array}{cc} (1;1) & \left(2; \frac{2}{3}\right) \\ \left\{\left(2; \frac{1}{3}\right), \left(3; \frac{2}{3}\right)\right\} & \left\{\left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right)\right\} \\ \left\{\left(4; \frac{2}{3}\right), \left(5; \frac{1}{3}\right)\right\} & \left(5; \frac{2}{3}\right) \\ (6;1) & \left(7; \frac{2}{3}\right) \end{array} \right) \Rightarrow \left(\begin{array}{cc} & \left(2; \frac{2}{3}\right) \\ \left\{\left(1; \frac{2}{3}\right), \left(7; \frac{1}{3}\right)\right\} & \left\{\left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right)\right\} \\ \left\{\left(2; \frac{1}{3}\right), \left(3; \frac{2}{3}\right)\right\} & \left(5; \frac{2}{3}\right) \\ \left\{\left(4; \frac{2}{3}\right), \left(5; \frac{1}{3}\right)\right\} & \left(7; \frac{2}{3}\right) \\ (6;1) & \left\{\left(1; \frac{1}{3}\right), \left(8; \frac{1}{3}\right)\right\} \end{array} \right) \Rightarrow \\
& \Rightarrow \left(\begin{array}{cc} & \left(2; \frac{2}{3}\right) \\ \left\{\left(1; \frac{2}{3}\right), \left(7; \frac{1}{3}\right)\right\} & \left\{\left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right)\right\} \\ \left\{\left(2; \frac{1}{3}\right), \left(3; \frac{2}{3}\right)\right\} & \left(5; \frac{2}{3}\right) \\ \left\{\left(4; \frac{2}{3}\right), \left(5; \frac{1}{3}\right)\right\} & \left(7; \frac{2}{3}\right) \\ \left\{\left(6; \frac{1}{3}\right), \left(8; \frac{2}{3}\right)\right\} & \left\{\left(1; \frac{1}{3}\right), \left(8; \frac{1}{3}\right)\right\} \\ & \left(6; \frac{2}{3}\right) \end{array} \right).
\end{aligned}$$

Після цього до упорядкування S вносяться вершини другої долі. За алгоритмом починаємо цей процес, враховуючи необхідність дозаповнення 2-ого місця упорядкування до щільності.

$$\begin{aligned}
& \left(\dots \left\{ \left(2; \frac{2}{3}\right), \left(9; \frac{1}{3}\right) \right\}, \left\{ \left(9; \frac{2}{3}\right), \left(10; \frac{2}{15}\right) \right\} \right) \Rightarrow \left(\dots \left\{ \left(2; \frac{2}{3}\right), \left(9; \frac{1}{3}\right) \right\}, \left\{ \left(9; \frac{2}{3}\right), \left(10; \frac{2}{15}\right) \right\} \right. \\
& \quad \left. \left\{ \left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right), \left(10; \frac{1}{3}\right) \right\}, \left\{ \left(10; \frac{8}{15}\right), \left(11; \frac{4}{15}\right) \right\} \right) \Rightarrow \\
& \Rightarrow \left(\begin{array}{cc} \left\{ \left(2; \frac{2}{3}\right), \left(9; \frac{1}{3}\right) \right\} & \left\{ \left(9; \frac{2}{3}\right), \left(10; \frac{2}{15}\right) \right\} \\ \dots \left\{ \left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right), \left(10; \frac{1}{3}\right) \right\} & \left\{ \left(10; \frac{8}{15}\right), \left(11; \frac{4}{15}\right) \right\} \\ \left\{ \left(5; \frac{2}{3}\right), \left(11; \frac{1}{3}\right) \right\} & \left\{ \left(11; \frac{2}{5}\right), \left(12; \frac{2}{5}\right) \right\} \end{array} \right) \Rightarrow \left(\begin{array}{cc} \left\{ \left(2; \frac{2}{3}\right), \left(9; \frac{1}{3}\right) \right\} & \left\{ \left(9; \frac{2}{3}\right), \left(10; \frac{2}{15}\right) \right\} \\ \left\{ \left(3; \frac{1}{3}\right), \left(4; \frac{1}{3}\right), \left(10; \frac{1}{3}\right) \right\} & \left\{ \left(10; \frac{8}{15}\right), \left(11; \frac{4}{15}\right) \right\} \\ \left\{ \left(5; \frac{2}{3}\right), \left(11; \frac{1}{3}\right) \right\} & \left\{ \left(11; \frac{2}{5}\right), \left(12; \frac{2}{5}\right) \right\} \\ \left\{ \left(7; \frac{2}{3}\right), \left(12; \frac{1}{3}\right) \right\} & \left\{ \left(12; \frac{4}{15}\right), \left(13; \frac{8}{15}\right) \right\} \end{array} \right) \Rightarrow
\end{aligned}$$

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

Побудувавши S та S' , за принципом, що описаний на 9 кроці алгоритма 3.3 формуємо наступне упорядкування:

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

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

Твердження 3.3. Модифікований алгоритм 3.2 для задачі $S(K_{m,r}, h_i, l)$ у випадку, коли не виконується умова (3.1) та дозволені переривання, є точним.

Доведення. Очевидно, що при застосуванні алгоритму 3.2 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 (3.4) третього кроку алгоритму:

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

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 (3.3), 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 (3.4) 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 (3.4) 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, заповнюючи 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 3.3 є оптимальним.

Твердження 3.4. 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 3.2 був точним для k та p взятих з нерівностей:

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

Доведення. Розглянемо окремо умови твердження. Достатність першої з них ($h_i \leq m$ при $1 \leq i \leq k+1$) доводиться за аналогією до того, як це було зроблено для твердження 3.2.

Доведемо достатність другої умови. За модифікованим алгоритмом 3.3 $(k+1)$ -е місце упорядкування дозаповнюється розбиттями вершин другої долі, причому якщо $m \leq r$, 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 $h_{k+p+1} \leq r$. Інакше дозаповнити це місце на відповідну величину неможливо.

3.2 Наближені алгоритми для узагальнених задач з перериваннями

Застосування точних алгоритмів поліноміальної складності 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. Виникає питання, а чи не можна покращити якість наближених розв'язків для цих задач, якщо дозволити переривання при виконанні робіт?

Розглянемо таку можливість при застосуванні алгоритму, що заснований на рівневому принципі [53]. Він дає точний розв'язок задачі упорядкування за умов, що граф задачі має структуру кореневого дерева або лісу, який складається з таких дерев.

При застосуванні цього алгоритму у випадку графа довільної будови відома така оцінка точності розв'язку:

$$\frac{l_A}{l^*} \leq \frac{2h}{h+1}, \quad (3.5)$$

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

Відомими є умови, за яких оцінка (3.5) досяжна, і відповідно непокращувана. Розглянемо граф G_9 наступного вигляду [1].

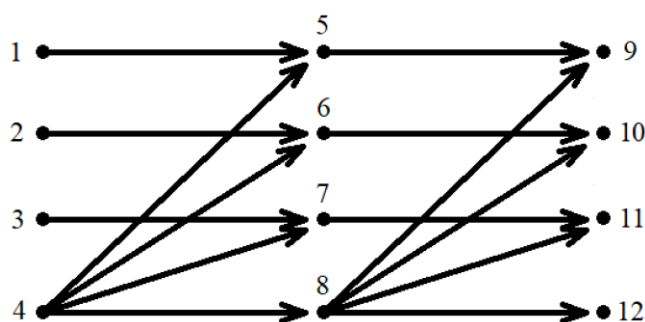


Рис. 3.2. Граф G_9

Застосувавши алгоритм, заснований на рівневому принципі, при $h = 3$ одержуємо упорядкування:

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

При цьому оптимальним упорядкуванням буде наступне:

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

Легко бачити, що в даному випадку виконується рівність оцінки (3.5):

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

Проаналізуємо можливість зменшення довжини наближеного розв'язку, побудованого за даним алгоритмом за рахунок дозволу переривань. Оскільки при цьому питання щільності є ключовим, то першочергово дозволятимемо переривання робіт, що відповідають вершинам, розташованим на нещільно заповнених місцях. Для визначеності, нехай дозволяється переривання роботи, що відповідає вершині 4. Побудоване упорядкування з урахуванням цього дозволу:

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

Його довжина $l'_A = 5,5$. 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 $l = 4,5$.

Інший відомий алгоритм поліноміальної складності заснований на застосуванні лексикографічної розмітки графа. Для графів без транзитивних дуг при ширині упорядкування рівній 2 цей алгоритм є точним, однак при $h > 2$ в загальному випадку дає наближений розв'язок. В [56] наводиться наступна оцінка точності алгоритму для випадків $h \geq 3$:

$$\frac{l_A}{l^*} \leq 2 - \frac{2}{h}. \quad (3.6)$$

Авторами було наведено розв'язання задачі упорядкування вершин графа G_{10} , зображеного на рис. 3.3 при $h = 3$.

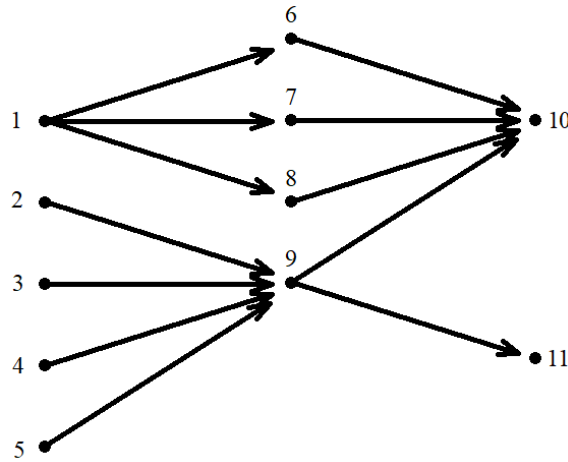


Рис. 3.3. Граф G_{10}

При застосуванні алгоритму одержане упорядкування

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

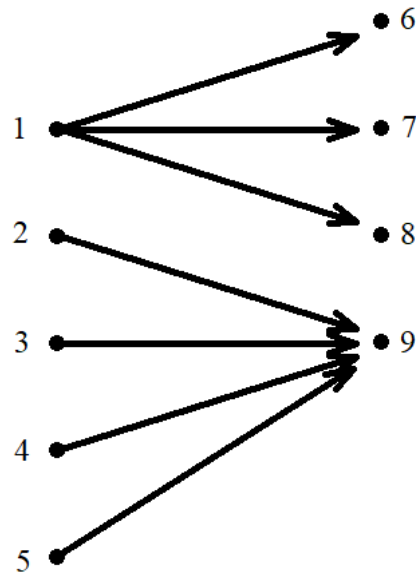
має довжину $l_A = 5$. В той же час довжина оптимального упорядкування, що має вигляд:

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

складає $l^* = 4$. Дійсно, повертаючись до (3.6), маємо:

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

Нехай граф G'_{10} отримано з графа G_{10} шляхом вилучення з нього вершин 10 і 11 та інцидентних до них дуг. Легко бачити, що при розгляді аналогічної задачі для графа G'_{10} оцінка (3.6) є досяжною.

Рис. 3.4. Граф G'_{10}

Побудоване за алгоритмом упорядкування

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

має довжину $l_A = 4$, тоді як довжина оптимального упорядкування

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

складає $l^* = 3$. Таким чином дійсно в даному випадку (3.6) перетворюється на рівність:

$$\frac{l_A}{l^*} = 2 - \frac{2}{h} = \frac{4}{3}.$$

Визначимо, як зміниться довжина упорядкування, якщо дозволити переривання однієї з робіт. Аналогічно до випадку застосування алгоритму, який базується на рівневому принципі, першочергово звертатимемо увагу на нещільно заповнені місця в упорядкуванні. Нехай дозволене переривання роботи, що відповідає вершині 6. Відповідне упорядкування:

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

що має довжину $l'_A = 3,5$. Якщо також переривання дозволене для роботи 1, то упорядкування

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

має довжину рівну оптимальній $l'_A = l^* = 3$. Очевидно, що додатковий дозвіл на переривання роботи під номером 2 не матиме впливу на значення цільової функції, однак якщо при цьому заборонити дозвіл на переривання роботи 1, то отримане упорядкування також матиме довжину $l'_A = l^* = 3$. Таким чином, дозволи на переривання робіт 1 та 2 одночасно не впливають на оптимальність розв'язку, порівняно з випадком, коли переривання дозволені лише для однієї з них.

Нехай граф G''_{10} відрізняється від G'_{10} лише спрямованістю всіх дуг. Аналогічно до попереднього прикладу, 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 розміщених 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:

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

Таким чином було розглянуто можливість покращення за рахунок переривань наближених розв'язків, отриманих при застосуванні двох відомих алгоритмів. Виявлено, що дозвіл на переривання довільної вершини, яка розміщена на частково заповненому місці, не гарантує покращення значення цільової функції.

3.3 Зв'язок задач пакування та упорядкування з перериваннями

Задачі пакування, так само як і задачі упорядкування, мають широке практичне застосування. Встановлення зв'язків між ними дозволить проаналізувати можливості використання методів розв'язання однієї із задач до іншої та надати практичні рекомендації. Окрім того це розширить коло прикладних задач, які можна формулювати у вигляді задач упорядкування.

Спочатку розглянемо можливість зведення задачі $S(G, h, l)$ без наявності часткового порядку, коли граф являє собою n ізольованих вершин, і при заданій ширині необхідно побудувати упорядкування мінімальної довжини. Вагу всіх вершин орграфа вважатимемо однаковою і позначимо як a . Розміщення вершин графа в упорядкуванні будемо зводити до задачі про оптимальне лінійне пакування, яка має наступне формулювання [113].

Задача 3.1. Нехай задана множина одновимірних комірок довжини L і n об'єктів із заданими довжинами $a_1, a_2, a_3, \dots, a_n$. Необхідно знайти мінімальне ціле число комірок B та розбиття множини об'єктів $\bar{N} = \{1, \dots, n\}$ на B підмножин

$N_1, N_2, N_3, \dots, N_B$ таким способом, щоб $\bigcup_{i=1}^B N_i = \bar{N}$, $N_i \cap N_j = \emptyset$, $\sum_{j \in N_i} a_j \leq L$, $i = \overline{1, B}$.

Для формалізації задачі введемо величини x_{ij} та y_i наступним чином:

$$x_{ij} = \begin{cases} 1, \text{ якщо } j\text{-ий об'єкт міститься в } i\text{-тій комірці} \\ 0, \text{ в іншому випадку} \end{cases};$$

$$y_i = \begin{cases} 1, \text{ якщо } i\text{-та комірка непорожня} \\ 0, \text{ в іншому випадку} \end{cases}.$$

Цільова функція задачі пакування має вигляд:

$$B = \sum_{i=1}^n y_i \rightarrow \min. \quad (3.7)$$

Обмеження:

$$\sum_{j=1}^n a_j x_{ij} \leq L y_i, i = \overline{1, B}; \quad (3.8)$$

$$\sum_{i=1}^B x_{ij} = 1, j = \overline{1, n}; \quad (3.9)$$

$$y_i \in \{0; 1\}, i = \overline{1, B}; \quad (3.10)$$

$$x_{ij} \in \{0; 1\}, i = \overline{1, B}, j = \overline{1, n}. \quad (3.11)$$

Для встановлення зв'язку між задачами припустимо, що коміркам відповідають місця в упорядкуванні, а об'єктам довжини $a_1 = a_2 = a_3 = \dots = a_n = a$ відповідають вершини. Тоді довжина комірок $L = \frac{na}{h}$, де h — ширина упорядкування.

У випадку заборонених переривань в задачі упорядкування в кожному комірці розміститься не більше ніж $\left\lfloor \frac{n}{h} \right\rfloor$ об'єктів, при цьому комірки можуть бути заповнені на величину меншу за L . Якщо в початкових умовах задачі кількість об'єктів кратна ширині упорядкування, то в кожній комірці розміститься $\frac{n}{h}$ об'єктів.

Співставимо одержані розв'язки задач лінійного пакування та паралельного упорядкування. Легко бачити, що вершини, які відповідають об'єктам в перших h комірках розподіляються групами по ширині упорядкування, при цьому вершини з підмножин N_i , де $i > h$, розміщуються на наступних позиціях упорядкування.

Розглянемо також випадок, коли при побудові упорядкувань дозволені переривання. Обмеження (3.11) в цьому випадку потребує заміни на інше:

$$0 \leq x_{ij} \leq 1, i, j \in \{1, 2, \dots, n\}. \quad (3.12)$$

При заміні обмеження (3.11) на (3.12) одержуємо уявлення про те, який вигляд будуть мати розбиття вершин та оптимальне значення цільової функції при дозволених перериваннях відповідної задачі упорядкування. Ця заміна дозволяє у тих випадках, коли за умовою задачі n не кратне h , враховувати можливість застосувати розбиття до вершин.

Розглянемо розв'язання задачі упорядкування шляхом зведення її до відповідної задачі пакування на прикладі.

Приклад 3.3. Знайти упорядкування мінімальної довжини для графа $G(V, U)$, $|V| = n$, $U = \emptyset$, де ваги вершин $w_i = 3$ для $i = \overline{1, n}$, $h = 6$, $n = 15$, застосувавши зведення до задачі пакування. Розглянути випадки, коли переривання при виконанні відповідних робіт заборонені та дозволені.

При зведенні до задачі пакування спочатку визначимо довжини комірок:

$$L = \frac{45}{6} = 7 \frac{1}{2}.$$

Кожній вершині v_i ставимо у відповідність об'єкт виду $(i, 3)$.

Для випадку заборонених переривань маємо таке розміщення об'єктів по комірках:

$$\begin{aligned} &\{(1;3), (2;3)\}; \{(3;3), (4;3)\}; \\ &\{(5;3), (6;3)\}; \{(7;3), (8;3)\}; \\ &\{(9;3), (10;3)\}; \{(11;3), (12;3)\}; \\ &\{(13;3), (14;3)\}; \{(15;3)\}. \end{aligned}$$

Одержали 8 непорожніх комірок. У відповідному упорядкуванні місця з першого по шосте будуть заповнені вершинами 1, 2, ..., 12, які знаходяться в перших шести комірках. Місця 7, 8 та 9 відповідно займають вершини 13, 14 та 15. Таким чином довжина оптимального упорядкування $l^* = 9$.

У випадку, коли переривання дозволені, у відповідній задачі пакування комірки заповнюються наступним чином:

$$\left\{ (1;3), (2;3), \left(3;1\frac{1}{2}\right) \right\}; \left\{ \left(3;1\frac{1}{2}\right), (4;3), (5;3) \right\};$$

$$\left\{ (6;3), (7;3), \left(8;1\frac{1}{2}\right) \right\}; \left\{ \left(8;1\frac{1}{2}\right), (9;3), (10;3) \right\};$$

$$\left\{ (11;3), (12;3), \left(13;1\frac{1}{2}\right) \right\}; \left\{ \left(13;1\frac{1}{2}\right), (14;3), (15;3) \right\}.$$

Тоді оптимальне упорядкування матиме вигляд

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

та довжину $l_n^* = 7\frac{1}{2}$. За допомогою описаного підходу можливо здійснити перехід від формулювання задачі упорядкування, за умов, що переривання дозволені чи заборонені, до задачі пакування і навпаки. Це дозволить аналізувати можливість взаємного застосування відомих методів та алгоритмів розв'язання одних задач до інших.

3.4 Висновки до розділу

В даному розділі були проаналізовані деякі узагальнення задачі паралельного упорядкування та досліджено вплив дозволу переривань на оптимальність їх розв'язків.

Розглядалася можливість застосування переривань з метою уникання виникнення аномалій в задачах упорядкування із заданим списком пріоритетів. Умови, за яких вплив аномалій на значення цільової функції може бути скомпенсованим, сформульовано у вигляді твердження.

Для узагальнення задачі упорядкування, при якому замість сталої ширини упорядкування h задається послідовність h_i розроблено два алгоритми побудови оптимального упорядкування для графів без заданого часткового порядку. Алгоритм 3.1 застосовується у випадках, коли переривання заборонені або, з урахуванням початкових даних конкретної задачі, недоцільні. Алгоритм 3.2 дозволяє побудувати оптимальне упорядкування з урахуванням дозволу переривань.

Розроблено модифікацію алгоритму 3.2 у вигляді алгоритму 3.3, застосування якого дозволяє будувати упорядкування в узагальненій задачі, де граф має структуру повного дводольного. Доведено, що упорядкування, побудоване за цим модифікованим алгоритмом, є оптимальним.

Розглянуто два відомі алгоритми поліноміальної складності, один з яких базується на рівневому принципі, а другий — на лексикографічній розмітці графа. Для випадків, коли застосування цих алгоритмів дає наближені розв'язки, досліджено вплив дозволу переривань на оптимальність розв'язків. Показано, що такий дозвіл не гарантує зменшення значення цільової функції.

Проаналізовано можливість зведення задач упорядкування до задач пакування для графів без часткового порядку. Сформульовано обмеження відповідної задачі пакування в залежності від того, чи дозволені переривання при виконанні робіт у початковій задачі.

Основні результати розділу опубліковані у [1,7,8,11].

Розділ 4. ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Опис основних програмних модулів

Розроблений програмний продукт використовує засоби мови програмування C#.

Реалізовано відомі алгоритми розв’язання задач паралельного упорядкування для випадків, коли відповідний оргграф задачі відноситься до підкласу:

- графів, що складаються з множини ізольованих вершин:
 - із однаковими ваговими коефіцієнтами;
 - із різними ваговими коефіцієнтами;
- регулярних, які мають структуру дводольного;
- паралельно-послідовних;
- дерев-зірок;
- оливкових дерев.

Для цих задач враховуються постановки, що допускають дозвіл чи заборону на переривання робіт, знаходиться оптимальне значення цільової функції та в разі наявності оцінюється виграш від переривань.

Також реалізовано розроблені алгоритми 3.1 – 3.3 розв’язання узагальненої задачі упорядкування, де обмеження на ресурси задаються вектором. Перші два з них дають точний розв’язок для графів, що складаються з множини ізольованих вершин одиничної ваги, третій — для повних дводольних графів.

Основними об’єктами програми є класи *Vertex*, *Edge* та *Graph*, з яких перші два є допоміжними.

Клас *Vertex* містить два поля, що зберігають цілочисельні значення, які відповідають номеру вершини оргграфа та його ваговому коефіцієнту, а також два методи, які є конструкторами. Перший являє собою конструктор за замовчуванням, який створює порожній об’єкт класу, другий приймає два

цілочисельні аргументи, значення яких використовує при створенні нового екземпляру класу з відповідним порядковим номером та вагою вершини.

Клас *Edge* складається з двох полів та чотирьох методів. Значення полів відповідають відповідним номерам вершин, інцидентних дузі орграфа. Усі методи класу є конструкторами:

- 1) конструктор за замовчуванням, що створює пустий екземпляр класу;
- 2) приймає два цілочисельні аргументи, значення першого з них відповідає номеру вершини, з якої дана дуга виходить, другого — в яку входить;
- 3) аргументами є два об'єкти класу *Vertex*, з яких отримуються значення номерів вершин;
- 4) приймається один аргумент, що є іншим об'єктом даного класу, і копіюються значення його полів.

У класі *Graph* містяться два поля та два методи. Полями є два масиви об'єктів класів відповідно *Vertex* та *Edge*, які задають множини вершин та дуг графа. Обидва методи даного класу — це конструктори:

- 1) конструктор за замовчуванням, в якому створюється порожній об'єкт класу;
- 2) два аргументи, які передаються у даний метод, відповідають одновимірним масивам об'єктів класів *Vertex* та *Edge*, з яких утворюється новий екземпляр класу.

В силу способу задання початкових даних задачі упорядкування клас *Graph* не містить методів, що відповідають за перевірку умов ациклічності, відсутності петель та кратних дуг. Для кожного підкласу графів, що розглядалися, така перевірка здійснювалася окремо при початковій обробці даних, уведених користувачем.

Окрім перелічених структурних компонентів програма містить метод *DrawCBGraph*, в якому реалізовано можливість візуального відображення повного дводольного графа.

Решта методів, наявних у програмному продукті, реалізують алгоритми розв'язання задач упорядкування у різних постановках та для різних початкових умов.

Метод *Indep_Alg* реалізовує алгоритми розв'язання задачі упорядкування з незалежними роботами, відповідний граф якої не містить дуг, а лише множину ізольованих вершин. Множина аргументів даного методу включає задані цілочисельну ширину упорядкування h , зважений граф G , представлений об'єктом відповідного класу та об'єкт стандартного класу *DataGridView*, в який буде записано побудоване в ході виконання алгоритму паралельне упорядкування з перериваннями. Метод повертає текстовий рядок зі скороченим записом одержаного упорядкування з перериваннями, число, що відповідає його довжині, і значення виграшу від застосування переривань у відсотках.

Методи *CB_Alg*, *SP_Alg*, *Star_Alg* та *Olive_Alg* аналогічно реалізують алгоритми розв'язання задач упорядкування відповідно для повного дводольного, паралельно-послідовного підкласів графів, зірок та оливкових дерев. Аналогічно до *Indep_Alg* в якості аргументів приймається ціле число h та об'єкт класу *DataGridView*. Оскільки структура графів у зазначених методах відома, то для спрощення замість об'єкта класу *Graph* передаються:

- два цілочисельних параметра $n1$ та $n2$, що відповідають кількості вершин у долях для *CB_Alg*;
- вектор n_comp , яким задаються потужності вершин паралельно-послідовних підграфів, поєднаних послідовною композицією для *SP_Alg*;
- параметр n , що визначає кількість нецентральных вершин зірки для *Star_Alg* та *Olive_Alg*.

Для розв'язання задач в узагальненій постановці застосовуються методи *Alg_1* та *Alg_2*. В них реалізовано відповідно алгоритми 3.1 та 3.2 описані у третьому розділі. Множину аргументів складають: одновимірний масив h_arr , що визначає вектор, яким задано обмеження на ресурси; натуральне число n , що відповідає кількості вершин графа G з одиничними ваговими коефіцієнтами; об'єкт стандартного класу *DataGridView* необхідний для запису побудованого

упорядкування. Оскільки згадані алгоритми є точними для випадків, коли відповідні графи складаються з множини ізольованих вершин з вагами рівними 1, то для раціонального застосування процесорних ресурсів у даному разі об'єкт класу *Graph* не використовується, натомість в якості початкових даних метод приймає лише потужність множини вершин графа.

Алгоритм 3.3 реалізується за допомогою методу *Alg_3*. Подібно до *CB_Alg* в якості аргументів передаються два цілочисельних параметра $n1$ та $n2$, що відповідають кількості вершин у долях та об'єкт *DataGridView*, куди буде записане побудоване упорядкування. В якості вектора, що визначає змінну ширину упорядкування передається одновимірний масив натуральних чисел h_arr .

4.2. Опис інтерфейсу та інструкція користувача

При запуску програми перед користувачем відкривається інтерфейс наступного вигляду, що умовно можна розділити на 6 структурних фрагментів (рис. 4.1).

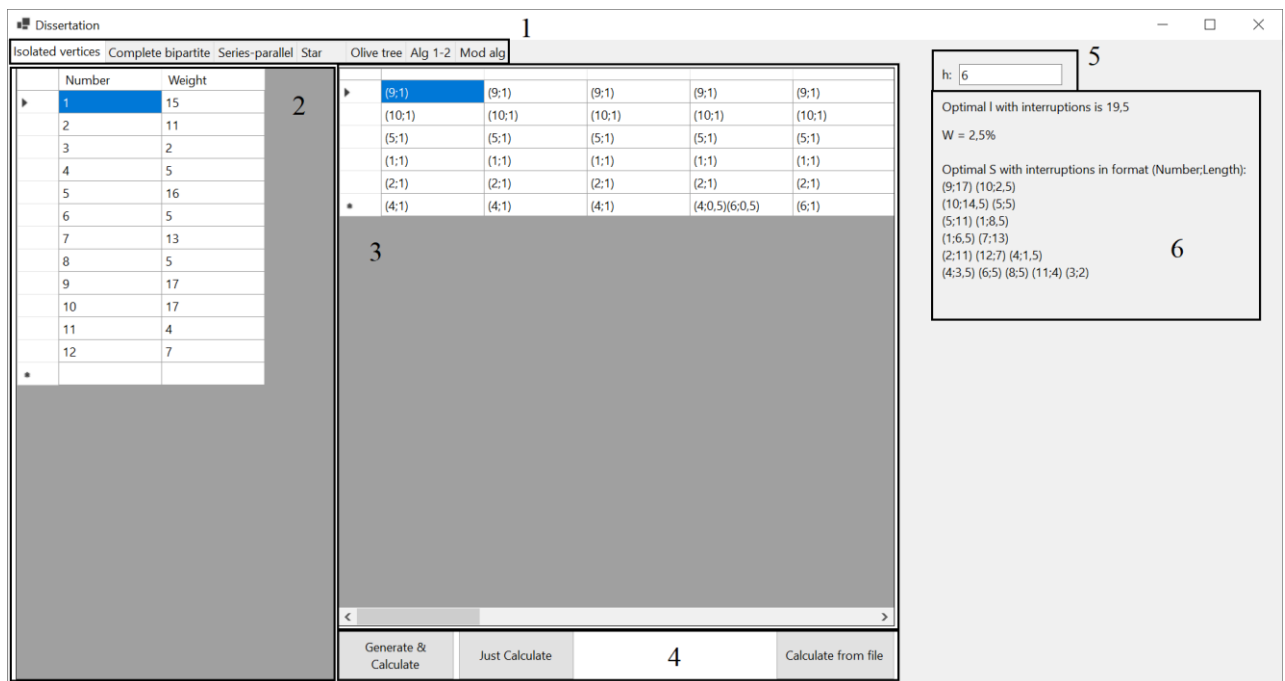


Рис. 4.1. Інтерфейс користувача

На панелі (1) розташовані вкладки, на яких розміщені функціональні елементи: для задання початкових умов задачі (2); виведення побудованого упорядкування (3); керування (4). Залежно від обраної вкладки застосовується відповідний метод програми, що реалізовує алгоритм розв'язання задачі упорядкування:

- «Isolated vertices» — метод *Indep_Alg*;
- «Complete bipartite» — метод *CB_Alg*;
- «Series-parallel» — метод *SP_Alg*;
- «Star» — метод *Star_Alg*;
- «Olive tree» — метод *Olive_Alg*;
- «Alg 1-2» — методи *Alg_1* та *Alg_2*;
- «Mod alg» — метод *Alg_3*.

Початкові умови вводяться на панелі (2) різними способами, залежно від обраної на (1) вкладки:

- у вигляді таблиці зі стовпцями, що відповідають номеру та ваговому коефіцієнту вершин (на «Isolated vertices»);
- двома натуральними числами, що відповідають кількостям вершин в долях (на «Complete bipartite» та «Mod alg»);
- таблицею з одним стовпцем, в якому задаються потужності множин вершин паралельно-послідовних підграфів, що об'єднуються шляхом послідовної композиції, у вигляді натуральних чисел, більших за 2 (на «Series-parallel»);
- одним натуральним числом, що відповідає кількості нецентральних вершин вхідної чи вихідної зірки (на «Star»), ланцюгів, що утворюють оливкове дерево (на «Olive Ttee»), вершин графа, що не містить дуг (на «Alg 1-2»).

Для постановок задачі, де відповідний граф має структуру повного дводольного, при загальній кількості вершин не більший за 50, реалізовано можливість відображення цього графа в інтерфейсі (рис. 4.2) за допомогою методу *DrawCBGraph*.

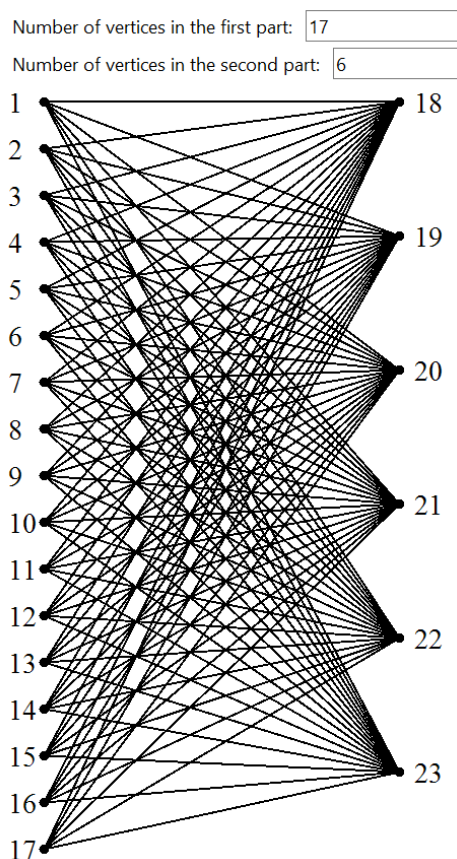


Рис. 4.2. Приклад відображення дводольного графа

На (3) після задання користувачем початкових даних та розв'язання задачі записується побудоване за відповідним алгоритмом паралельне упорядкування. Кожен стовпець таблиці відповідає одному місцю упорядкування, а відповідні елементи розбитів вершин графа задачі відображаються у форматі упорядкованих пар, як описано у розділі 1.

У фрагменті інтерфейсу (4) розташовані кнопки керування. Кнопка «Generate & Calculate» автоматично генерує граф задачі згідно з обраною вкладкою (1) і заповнює відповідні поля та таблиці, а «Just Calculate» натомість лише зчитує введені дані з елементів інтерфейсу. Для вкладок «Isolated vertices» та «Series-parallel» додано можливість введення початкових даних з таблиці, що береться з файлу формату *xlsx*. Приклад початкових даних, взятих з файлу, наведено на рис. 4.3. Кожна з описаних кнопок після обробки початкових даних задачі активує виконання відповідного методу програми, що реалізовує відповідний алгоритм.

	A	B	C
1	1	9	
2	2	4	
3	3	2	
4	4	3	
5	5	7	
6	6	8	
7	7	4	
8	8	5	
9	9	6	
10	10	8	
11	11	4	
12	12	9	
13			

Рис. 4.3. Приклад запису початкових даних у файлі

Текстове поле, розташоване в частині інтерфейсу (5) необхідне для задання ширини упорядкування задачі (значення за замовчуванням — 5). При застосуванні алгоритмів 3.1-3.3 послідовність заданих значень h_i слід записувати через кому, без пробілів. Якщо заданої користувачем послідовності виявиться недостатньо, то додаткові значення послідовності додадуться автоматично шляхом дублювання останнього значення, записаного у текстове поле. У разі застосування інших алгоритмів необхідно ввести натуральне число, більше за 1.

Фрагмент (6) містить текстові об'єкти інтерфейсу, в які після розв'язання задачі упорядкування записуються:

- значення довжини упорядкування при дозволених перериваннях;
- можливий виграш від застосування переривань;
- побудоване упорядкування у скороченій формі для задач малої розмірності (кількість вершин графа до 20).

ВИСНОВКИ

Дисертаційна робота містить нові актуальні результати теоретичних досліджень, що стосуються одного класу задач дискретної оптимізації. Це задачі паралельного упорядкування вершин орграфів, до яких зводяться ряд прикладних задач, які враховують можливість переривання для оптимізації відповідних процесів.

Узагальнюючи проведені дослідження за темою, можна зробити наступні висновки:

1. Проведений огляд та аналіз попередніх досліджень за темою виявив, що потребують подальшого дослідження ряд актуальних питань, пов'язаних з пошуком розв'язків задач паралельного упорядкування вершин орграфів. Ці задачі виникають у ряді прикладних сфер, зокрема пов'язаних з виробництвом, обслуговуванням, управлінням проєктами.

2. Подальшого розвитку дістав математичний апарат теорії паралельних упорядкувань, зокрема було введено нове поняття розбиття для вершин, на основі якого узагальнено означення паралельного упорядкування, а також його основних характеристик. Це узагальнення дозволяє враховувати можливість застосування переривань та задані різні вагові коефіцієнти вершин.

3. Для задач упорядкування отримані нові теоретичні результати, що стосуються можливих переривань при виконанні робіт, а саме: виявлено нові підкласи графів, для яких переривання можуть покращити розв'язки; введені оцінки для апріорного визначення виграшу у випадках, коли граф задачі належить одному з цих підкласів; визначено, від яких початкових даних задачі залежить можливий виграш у випадку, коли граф має структуру повного дводольного; проаналізований вплив заданих вагових коефіцієнтів вершин на ефективність переривань для одного спеціального класу дерев.

4. Показано можливість покращення наближених розв'язків, отриманих за двома відомими алгоритмами поліноміальної складності, за рахунок дозволу переривань окремих робіт. Подальшого розвитку дістало

дослідження зв'язку задач упорядкування із задачами пакування, зокрема сформульовані умови, за яких можливе взаємне зведення однієї з них до іншої.

5. Запропоновано нові ефективні алгоритми розв'язання задач упорядкування при допустимих та заборонених перериваннях в одній узагальненій постановці, яка передбачає задану ширину упорядкування у вигляді вектора, який визначає максимальну допустиму кількість елементів, що можуть бути розміщені на кожному місці упорядкування.

6. Розроблено програмний продукт за допомогою засобів мови програмування *C#* в якому реалізовано відомі та запропоновані алгоритми розв'язання задач упорядкування при дозволених та заборонених перериваннях.

7. Проведено відповідні обчислювальні експерименти для валідації отриманих теоретичних результатів, а саме запропонованих оцінок виграшу від переривань, розроблених алгоритмів розв'язання задачі упорядкування в узагальненій постановці, де відповідний граф належить до підкласу повних дводольних, або складається з множини ізольованих вершин.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Коваленко Є.О., Турчина В.А. про покращення наближених розв'язків задачі паралельного упорядкування та аналіз моделі одного її узагальнення. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2025. Т. 2, Вип. 157. С. 35-47.
2. Турчина В.А., Коваленко Є.О. Умови зменшення довжини паралельних упорядкувань вершин спеціальних орграфів при наявності переривань. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2024. Т. 6, Вип. 155. С. 196-207.
3. Турчина В.А., Коваленко Є.О. Дослідження задачі упорядкування з перериваннями для одного підкласу дерев. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2023. Вип. 23. С. 118-125.
4. Турчина В.А., Коваленко Є.О. Вплив початкових даних задачі паралельного упорядкування з перериваннями на оптимальність розв'язку. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2022. Вип. 22. С. 158-167.
5. Коваленко Є.О., Турчина В.А. Аналіз впливу структури графів на оптимальність розв'язку задач паралельного упорядкування з перериваннями. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2021. Вип. 21. С. 130-137.
6. Турчина В.А., Коваленко Є.О. Доцільність дослідження дозволу переривань в одній задачі теорії розкладів. *Автоматика 2024: Тези XXVII Міжнародної конференції з автоматичного керування, Дніпро, 20-22 листопада 2024 р.*, м. Дніпро: ДНУ, 2024. С. 200-201.
7. Малієнко О.О., Коваленко Є.О. Дослідження впливу переривань на виникнення аномалій у задачах паралельного упорядкування. *Комбінаторні конфігурації та їхні застосування: Матеріали XXVI Міжнародного науково-практичного семінару, (Кропивницький – Запоріжжя – Київ, 13-15 червня 2024 року)* / за ред. Л.Ф. Гуляницького, м. Кропивницький – Запоріжжя – Київ, 2024. С. 103-107.
8. 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.
9. Турчина В.А., Коваленко Є.О. Апріорна оцінка довжини упорядкувань для спеціальних графів. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2023): Матеріали XXI міжнародної науково-практичної конференції, 22-24 листопада 2023 р.*, м. Дніпро, 2023. С. 293-294.

10. Турчина В.А., Коваленко Є.О. Переривання в задачах упорядкування вершин граціозних дерев. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Запоріжжя – Кропивницький, 14-16 червня 2023 року)* / за ред. Г.П. Донця, м. Запоріжжя - Кропивницький, 2023. С. 214-219.
11. Турчина В.А., Коваленко Є.О. Порівняльний аналіз задач упорядкування та пакування. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2022): Матеріали XX ювілейної міжнародної науково-практичної конференції, 23-25 листопада 2022. м. Дніпро, 2022. С. 208-209.*
12. Турчина В.А., Коваленко Є.О. Паралельні упорядкування для повних дводольних графів. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Кропивницький – Запоріжжя, 13-14 травня 2022 року).* / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2022. С. 82-86.
13. Коваленко Є.О., Турчина В.А. Аналіз структури графів в задачах паралельного упорядкування з перериваннями. *Комбінаторні конфігурації та їхні застосування: Матеріали XXIII Міжнародного науково-практичного семінару імені А.Я. Петренюка, присвяченого 70-річчю Льотної академії Національного авіаційного університету (Запоріжжя – Кропивницький, 13-15 травня 2021 року)* / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2021. С. 86-90.
14. Y.Kovalenko, V.Turchina, O.Hurko. On special classes of scheduling theory problems. *Сучасні науково-технічні дослідження у контексті мовного простору (англійською мовою): Матеріали X Регіональної науково-практичної конференції молодих науковців та студентів, 13 травня 2021 р. м. Дніпро, 2021. С. 28-30.*
15. Handbook of production scheduling / ed. by J. W. Herrmann. Boston, MA : Springer US, 2006. 320 p.
16. Mauergauz Y. Advanced planning and scheduling in manufacturing and supply chains. Cham : Springer International Publishing, 2016. 570 p.
17. Ovacik I. M., Uzsoy R. Decomposition methods for complex factory scheduling problems. Boston, MA : Springer US, 2012. 215 p.
18. Marichelvam M. K., Geetha M. Cuckoo search algorithm for solving real industrial multi-objective scheduling problems. *Advances in environmental engineering and green technologies*. 2019. P. 400–414.
19. Ioannou G., Dimitriou S. Lead time estimation in MRP/ERP for make-to-order manufacturing systems. *International journal of production economics*. 2012. Vol. 139, no. 2. P. 551–563.

20. Caridi M., Sianesi A. Multi-agent systems in production planning and control: an application to the scheduling of mixed-model assembly lines. *International journal of production economics*. 2000. Vol. 68, no. 1. P. 29–42.
21. Scheduling computer and manufacturing processes. / A. Orman et al. *The journal of the operational research society*. 1997. Vol. 48, no. 6. P. 659.
22. Choi T.-M., Yeung W.-K., Cheng T. C. E. Scheduling and co-ordination of multi-suppliers single-warehouse-operator single-manufacturer supply chains with variable production rates and storage costs. *International journal of production research*. 2013. Vol. 51, no. 9. P. 2593–2601.
23. Optimize manufacturing of pharmaceutical products with process simulation and production scheduling tools / V. Papavasileiou et al. *Chemical engineering research and design*. 2007. Vol. 85, no. 7. P. 1086–1097.
24. Multi-objective optimization of the order scheduling problem in mail-order pharmacy automation systems / H. Dauod et al. *The international journal of advanced manufacturing technology*. 2016. Vol. 99, no. 1-4. P. 73–83.
25. Cai X., Li K. N. A genetic algorithm for scheduling staff of mixed skills under multi-criteria. *European journal of operational research*. 2000. Vol. 125, no. 2. P. 359–369.
26. Vakharia A., Selim H., Husted R. Efficient scheduling of part-time employees. *Omega*. 1992. Vol. 20, no. 2. P. 201–213.
27. Acar I. Design of an automated staff scheduling system for an independent pharmacy. *Research in Social and Administrative Pharmacy*. 2018. Vol. 14, no. 12. P. 1134–1139.
28. Vanhoucke M. Project management with dynamic scheduling. 2nd ed. Berlin, Heidelberg : Springer Berlin Heidelberg, 2013. 218 p.
29. Herroelen W., Leus R., Demeulemeester E. Critical chain project scheduling: do not oversimplify. *Project management journal*. 2002. Vol. 33, no. 4. P. 48–60.
30. 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.
31. Blocher J. D., Chhajer D. The customer order lead-time problem on parallel machines. *Naval research logistics*. 1996. Vol. 43, no. 5. P. 629–654.
32. Rinnooy Kan A. H. G. Machine scheduling problems: classification, complexity and computations. The Hague : Nijhoff, 1976. 180 p.
33. Handbook of scheduling: algorithms, models, and performance analysis / ed. by J. Y.-T. Leung. Taylor & Francis Group, 2004. 1224 p.
34. Silberschatz A., Gagne G., Galvin P. B. Operating system concepts. 10th ed. Wiley & Sons, Limited, John, 2018. 1006 p.
35. Hachol-Balter M. Performance modeling and design of computer systems: queueing theory in action. Cambridge University Press, 2013. 576 p.

36. Scheduling problems - new applications and trends / ed. by R. da Rosa Righi. IntechOpen, 2020. 154 p.
37. Pinedo M. L. Scheduling: theory, algorithms, and systems. 5th ed. Springer London, Limited, 2016. 670 p.
38. Tan K. C., Dahal K., Cowling P. I. Evolutionary scheduling. Springer, 2010. 640 p.
39. Scheduling theory and its applications / P. Chretienne et al. *Journal of the operational research society*. 1997. Vol. 48, no. 7. P. 764–765.
40. Darte A., Robert Y., Vivien F. Scheduling and automatic parallelization. Boston, MA : Birkhäuser Boston, 2000. 264 p.
41. Drozdowski M. Scheduling for parallel processing. London : Springer London, 2009. 386 p.
42. Rayward-Smith V. J. UET scheduling with unit interprocessor communication delays. *Discrete applied mathematics*. 1987. Vol. 18, no. 1. P. 55–71.
43. Sinnen O. Task Scheduling for Parallel Systems. Hoboken, NJ, USA : John Wiley & Sons, Inc., 2007. 320 p.
44. Misic J., Reddy G.R., Misic V.B. Activity scheduling in bluetooth sensor networks. *Resource, mobility, and security management in wireless networks and mobile communications*. 2006. P. 55–86.
45. Kim Y. M., Lai T. H., Arora A. A qos-aware scheduling algorithm for bluetooth scatternets. High-Performance computing. Hoboken, NJ, USA, 2006. P. 661–681.
46. He Q., Yuan D., Ephremides A. Optimal scheduling for emptying a wireless network: solution characterization, applications, including deadline constraints. *IEEE transactions on information theory*. 2020. Vol. 66, no. 3. P. 1882–1892.
47. Pantelidou A. Scheduling in wireless networks. *Foundations and trends in networking*. 2009. Vol. 4, no. 4. P. 421–511.
48. Averbakh I., Berman O. Routing two-machine flowshop problems on networks with special structure. *Transportation science*. 1996. Vol. 30, no. 4. P. 303–314.
49. Averbakh I., Berman O. A simple heuristic for m-machine flow-shop and its applications in routing-scheduling problems. *Operations research*. 1999. Vol. 47, no. 1. P. 165–170.
50. Chou S.-Y., Lin S.-W. Museum visitor routing problem with the balancing of concurrent visitors. *Complex systems concurrent engineering*. London. P. 345–353.
51. Yu V. F., Lin S.-W., Chou S.-Y. The museum visitor routing problem. *Applied mathematics and computation*. 2010. Vol. 216, no. 3. P. 719–729.
52. Smith W. E. Various optimizers for single-stage production. *Naval research logistics quarterly*. 1956. Vol. 3, no. 1-2. P. 59–66.

53. Hu T. C. Parallel Sequencing and Assembly Line Problems. *Operations Research*. 1961. Vol. 9, no. 6. P. 841–848.
54. Fujii M., Kasami T., Ninomiya K. Optimal sequencing of two equivalent processors. *SIAM journal on applied mathematics*. 1969. Vol. 17, no. 4. P. 784–789.
55. Coffman E. G., Graham R. L. Optimal scheduling for two-processor systems. *Acta informatica*. 1972. Vol. 1, no. 3. P. 200–213.
56. Coffman, E. G., Bruno, J. Computer and job-shop scheduling theory. New York : Wiley, 1976. 299 p.
57. Conway R. W., Maxwell W. L., Miller L. W. Theory of scheduling. Dover Publications, Incorporated, 2003. 294 p.
58. Graham R. L., Lawler E. L., Lenstra J. K., Rinnooy Kan A. H. G. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. *Annals of Discrete Mathematics*. 1979. No. 5. P. 287–326.
59. Brucker P. Scheduling Algorithms. 5th ed. Berlin, Heidelberg : Springer Berlin Heidelberg, 2004. 371 p.
60. Garey M. R., Johnson D. S. Scheduling tasks with nonuniform deadlines on two processors. *Journal of the ACM*. 1976. Vol. 23, no. 3. P. 461–467.
61. Brucker P., Hurink J., Kubiak W. Scheduling identical jobs with chain precedence constraints on two uniform machines. *Mathematical methods of operations research*. 1999. Vol. 49, no. 2. P. 211–219.
62. Brucker P., Garey M. R., Johnson D. S. Scheduling equal-length tasks under treelike precedence constraints to minimize maximum lateness. *Mathematics of operations research*. 1977. Vol. 2, no. 3. P. 275–284.
63. Monma C. L. Linear-Time algorithms for scheduling on parallel processors. *Operations research*. 1982. Vol. 30, no. 1. P. 116–124.
64. Ten notes on equal-processing-time scheduling / P. Baptiste et al. *Quarterly journal of the belgian, french and italian operations research societies*. 2004. Vol. 2, no. 2. P. 111–127.
65. Bruno J., Coffman E. G., Sethi R. Scheduling independent tasks to reduce mean finishing time. *Communications of the ACM*. 1974. Vol. 17, no. 7. P. 382–387.
66. Lenstra J. K., Rinnooy Kan A. H. G., Brucker P. Complexity of machine scheduling problems. *Studies in integer programming*. 1977. P. 343–362.
67. Garey M. R., Johnson D. S. "Strong" NP-Completeness Results. *Journal of the ACM*. 1978. Vol. 25, no. 3. P. 499–508.
68. Ullman J. D. NP-complete scheduling problems. *Journal of computer and system sciences*. 1975. Vol. 10, no. 3. P. 384–393.
69. Du J., Leung J. Y.-T., Young G. H. Scheduling chain-structured tasks to minimize makespan and mean flow time. *Information and computation*. 1991. Vol. 92, no. 2. P. 219–236.

70. Kubiak W. Optimal scheduling of unit-time tasks on two uniform processors under tree-like precedence constraints. *Operations research methods and models of operations research*. 1989. Vol. 33, no. 6. P. 423–437.
71. Lenstra J. K., Rinnooy Kan A. H. G. Complexity of scheduling under precedence constraints. *Operations research*. 1978. Vol. 26, no. 1. P. 22–35.
72. McNaughton R. Scheduling with deadlines and loss functions. *Management science*. 1959. Vol. 6, no. 1. P. 1–12.
73. Lawler E. L. Preemptive Scheduling of Precedence-Constrained Jobs on Parallel Machines. *Deterministic and Stochastic Scheduling*. Dordrecht, 1982. P. 101–123.
74. Gonzalez T. F., Johnson D. B. A new algorithm for preemptive scheduling of trees. *Journal of the ACM*. 1980. Vol. 27, no. 2. P. 287–312.
75. Gonzalez T., Sahni S. Flowshop and jobshop schedules: complexity and approximation. *Operations research*. 1978. Vol. 26, no. 1. P. 36–52.
76. Lawler E. L. Preemptive scheduling of precedence-constrained jobs on parallel machines. *Deterministic and stochastic scheduling*. Dordrecht, 1982. P. 101–123.
77. Preemptive scheduling of uniform machines subject to release dates / J. Labetoulle et al. *Progress in combinatorial optimization*. 1984. P. 245–261.
78. Brucker P., Hurink J., Knust S. A polynomial algorithm for $P \mid p_j = 1, r_j, outtree \mid \sum C_j$. *Mathematical methods of operations research*. 2003. Vol. 56, no. 3. P. 407–412.
79. Coffman E. G., Sethuraman J., Timkovsky V. G. Ideal preemptive schedules on two processors. *Acta informatica*. 2003. Vol. 39, no. 8. P. 597–612.
80. Sitters R. Two np-hardness results for preemptive minsum scheduling of unrelated parallel machines. *Integer programming and combinatorial optimization*. Berlin, Heidelberg, 2001. P. 396–405.
81. Lenstra J. K., Vakhania N. On the complexity of scheduling unrelated parallel machines with limited preemptions. *Operations Research Letters*. 2023. P. 187–189.
82. Shchepin E. V., Vakhania N. On the geometry, preemptions and complexity of multiprocessor and open shop scheduling. *Annals of operations research*. 2007. Vol. 159, no. 1. P. 183–213.
83. Trietsch D., Baker K. R. Principles of Sequencing and Scheduling. Wiley & Sons, Limited, John, 2018. 656 p.
84. Ramamoorthy C. V., Gonzalez M. J. A survey of techniques for recognizing parallel processable streams in computer programs. *1969 Fall Joint Comput. Conf., AFIPS Conf Proc*. 1969. Vol. 35. P. 1–15.
85. Ramamoorthy C. V. A structural theory of machine diagnosis. *1967 Spring Joint Comput. Conf AFIPS Conf. Proc*. 1967. Vol. 30. P. 743–756.

86. Sergienko I. V., Shylo V. P., Roshchyn V. O. Algorithm Unions for Solving Discrete Optimization Problems. *Cybernetics and Systems Analysis*. 2023. Vol. 59, no. 5. P. 753–762.
87. Sergienko I. V., Hulianytskyi L. F., Sirenko S. I. Classification of applied methods of combinatorial optimization. *Cybernetics and Systems Analysis*. 2009. Vol. 45, no. 5. P. 732–741.
88. Hulianytskyi L. Search diversification in ACO algorithms and its application. *Cybernetics and Systems Analysis*. 2025. P. 27–42.
89. Hulianytskyi L., Semeniuta M., Yakymenko S. Methods of decomposition theory and graph labeling in the study of social network structure. *Information Technology and Implementation (IT&I-2024)*, November 20-21, 2024, Kyiv, 2024. P. 490-499.
90. 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.
91. Kozin I. V., Maksyshko N. K., Perepelitsa V. A. A Fragmented Model for the Problem of Land Use on Hypergraphs. *Cybernetics and Systems Analysis*. 2020. Vol. 56, no. 5. P. 753–757.
92. Kiseleva E. M., Prytomanova O. M. Fuzzy problem of the optimal set partition with constraints on the subsets centers location. *System research and information technologies*. 2020. No. 1. P. 78–89.
93. Kiseleva E., Prytomanova O., Hart L. Application of optimal set partitioning theory to solving problems of artificial intelligence and pattern recognition. *System research and information technologies*. 2021. No. 4. P. 91–101.
94. On the implementation of algorithms for solving the simplest dynamic problem of optimal set partitioning / E. M. Kiseleva et al. *Problems of applied mathematics and mathematic modeling*. 2024. P. 66–75.
95. Kiseleva E. M., Prytomanova O. M., Lebediev D. M. Solving continuous-discrete two-stage logistic problem of optimal partitioning-allocation. *Problems of applied mathematics and mathematic modeling*. 2024. P. 99–112.
96. Iemets O. O., Leonova M. V. Polynomial algorithms of solution for some problems of construction of the timetables of a device for demands with waiting. *Reports of the national academy of sciences of ukraine*. 2016. No. 3. P. 26–31.
97. Леонова М. В., Ємець О. О. Задача розкладу для одного приладу з функцією мінімізації часу обслуговування всіх завдань. *Інформатика та системні науки (ICH-2015)*, м. Полтава, 2015. 8 с.
98. Guk N., Verba O., Yevlakov V. Design of a recommendation system based on the transition graph. *Eastern-European journal of enterprise technologies*. 2021. Vol. 3, no. 4 (111). P. 24–31.

99. Huk N. A., Dykhanov S. V., Dolotov I. O. Analysis of the site structure using the concept of modularity. Mathematical and computer modelling. Series: physical and mathematical sciences. 2020. No. 21. P. 99–114.
100. Долотов І. О., Гук Н. А. Кластеризація зваженого вебграфу із використанням модулярності. *Питання прикладної математики і математичного моделювання*. 2023. Вип. 23. С. 45–52.
101. 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.
102. Semeniuta M. F., Donets G. A. Group labeling of some graphs. *Cybernetics and Systems Analysis*. 2020. Vol. 56, no. 5. P. 701–709.
103. Brucker P., Heitmann S., Hurink J. How useful are preemptive schedules? *Operations research letters*. 2003. Vol. 31, no. 2. P. 129–136.
104. Baptiste P., Timkovsky V. G. On preemption redundancy in scheduling unit processing time jobs on two parallel machines. *Operations research letters*. 2001. Vol. 28, no. 5. P. 205–212.
105. Valdes J., Tarjan R. E., Lawler E. L. The recognition of series parallel digraphs. *SIAM J. Comput.* 1982. P. 298–313.
106. Swamy M. N. S., Thulasiraman K. *Graphs, networks, and algorithms*. New York : Wiley, 1981. 592 p.
107. Petreniuk D. A. Graceful trees: the state of arts and the prospects. *Control Systems and Computers*. 2016. No. 1 (261). P. 16–25.
108. Rosa A. On certain valuations of the vertices of a graph. *Theory of Graphs*. 1967. P. 349–355.
109. Graham R. L. Bounds on multiprocessing timing anomalies. *SIAM journal on applied mathematics*. 1969. Vol. 17, no. 2. P. 416–429.
110. Малієнко О.О., Турчина В.А. Порівняльний аналіз аномалій для прямих та зворотних графів. *Питання прикладної математики і математичного моделювання*. 2023. Вип. 23. С. 161–170.
111. Maliienko O.O., Turchyna V.A. Analysis of the impact of task prioritization lists on the potential for avoiding anomalies in task scheduling, *System technologies*. 2024. Vol. 6. P. 167–174.
112. Турчина В.А., Зозуля Ю.С., Підаш А.К. Алгоритми перерахування всіх паралельних упорядкувань фіксованої довжини. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2013. Вип. 13. С. 256–261.
113. Pisinger D., Toth P. *Knapsack problems. Handbook of combinatorial optimization*. Boston, MA, 1998. P. 299–428.

ДОДАТОК А

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. **Коваленко Є.О., Турчина В.А.** Про покращення наближених розв'язків задачі паралельного упорядкування та аналіз моделі одного її узагальнення. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2025. Т. 2, Вип. 157. С. 35-47. doi: <https://doi.org/10.34185/1562-9945-2-157-2025-04>. Режим доступу до ресурсу: <https://journals.nmetau.edu.ua/index.php/st/article/view/1963/1233> (фахове видання, категорія Б).
2. Турчина В.А., **Коваленко Є.О.** Умови зменшення довжини паралельних упорядкувань вершин спеціальних орграфів при наявності переривань. *Збірник наукових праць «Системні технології»*, м. Дніпро, 2024. Т. 6, Вип. 155. С. 196-207. doi: <https://doi.org/10.34185/1562-9945-6-155-2024-19>. Режим доступу до ресурсу: <https://journals.nmetau.edu.ua/index.php/st/article/view/1927> (фахове видання, категорія Б).
3. Турчина В.А., **Коваленко Є.О.** Дослідження задачі упорядкування з перериваннями для одного підкласу дерев. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2023. Вип. 23. С. 118-125. doi: <https://doi.org/10.15421/322313>. Режим доступу до ресурсу: <https://pmm.dp.ua/index.php/pmmm/article/view/382> (фахове видання, категорія Б).
4. Турчина В.А., **Коваленко Є.О.** Вплив початкових даних задачі паралельного упорядкування з перериваннями на оптимальність розв'язку. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2022. Вип. 22. С. 158-167. doi: <https://doi.org/10.15421/322217>. Режим доступу до ресурсу: [https://pmm-](https://pmm.dp.ua/index.php/pmmm/article/view/382)

mm.dp.ua/index.php/pmmm/article/view/351 (фахове видання, категорія Б).

5. Коваленко Є.О., Турчина В.А. Аналіз впливу структури графів на оптимальність розв'язку задач паралельного упорядкування з перериваннями. *Збірник наукових праць «Питання прикладної математики і математичного моделювання»*. Дніпро, 2021. Вип. 21. С. 130-137. doi: <https://doi.org/10.15421/322113>. Режим доступу до ресурсу: <https://pm-mm.dp.ua/index.php/pmmm/article/view/316> (фахове видання, категорія Б).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Турчина В.А., Коваленко Є.О. Доцільність дослідження дозволу переривань в одній задачі теорії розкладів. *Автоматика 2024: Тези XXVII Міжнародної конференції з автоматичного керування, Дніпро, 20-22 листопада 2024 р., м. Дніпро: ДНУ, 2024. С. 200-201. Режим доступу до ресурсу: [http://automatika2024.dp.ua/files/Автоматика-2024%20\(тези%20доповідей\).pdf#page=200](http://automatika2024.dp.ua/files/Автоматика-2024%20(тези%20доповідей).pdf#page=200).*
2. Малієнко О.О., Коваленко Є.О. Дослідження впливу переривань на виникнення аномалій у задачах паралельного упорядкування. *Комбінаторні конфігурації та їхні застосування: Матеріали XXVI Міжнародного науково-практичного семінару, (Кропивницький – Запоріжжя – Київ, 13-15 червня 2024 року) / за ред. Л.Ф. Гуляницького, м. Кропивницький – Запоріжжя – Київ, 2024. С. 103-107. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2024/conf/6.3/CCTA-2024-proc.pdf#page=103.*
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. doi: <https://doi.org/10.36074/logos->

[02.02.2024.044](https://archive.logos-science.com/index.php/conference-proceedings/article/view/1548). Режим доступу до ресурсу: <https://archive.logos-science.com/index.php/conference-proceedings/article/view/1548>.

4. Турчина В.А., **Коваленко Є.О.** Априорна оцінка довжини упорядкувань для спеціальних графів. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2023): Матеріали XXI міжнародної науково-практичної конференції, 22-24 листопада 2023 р., м. Дніпро, 2023.* С. 293-294. Режим доступу до ресурсу: <http://mpzis.dnu.dp.ua/wp-content/uploads/2023/11/mpzis-2023.pdf#page=293>.
5. Турчина В.А., **Коваленко Є.О.** Переривання в задачах упорядкування вершин граціозних дерев. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Запоріжжя – Кропивницький, 14-16 червня 2023 року) / за ред. Г.П. Донця, м. Запоріжжя - Кропивницький, 2023.* С. 214-219. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2023/conf/1.4/Petrenyuk_ISPS-25-proc.pdf#page=214.
6. Турчина В.А., **Коваленко Є.О.** Порівняльний аналіз задач упорядкування та пакування. *Математичне та програмне забезпечення інтелектуальних систем (МПЗІС-2022): Матеріали XX ювілейної міжнародної науково-практичної конференції, 23-25 листопада 2022. м. Дніпро, 2022.* С. 208-209. Режим доступу до ресурсу: <http://mpzis.dnu.dp.ua/wp-content/uploads/2022/12/MPZIS-2022-1.pdf#page=209>.
7. Турчина В.А., **Коваленко Є.О.** Паралельні упорядкування для повних дводольних графів. *Комбінаторні конфігурації та їхні застосування: Матеріали XXV Міжнародного науково-практичного семінару імені А. Я. Петренюка, (Кропивницький – Запоріжжя, 13-14 травня 2022 року). / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2022.* С. 82-86. Режим доступу до ресурсу: https://zp.edu.ua/uploads/dept_s&r/2023/conf/1.4/Petrenyuk_ISPS-25-proc.pdf#page=75.

8. **Коваленко Є.О., Турчина В.А.** Аналіз структури графів в задачах паралельного упорядкування з перериваннями. *Комбінаторні конфігурації та їхні застосування: Матеріали XXIII Міжнародного науково-практичного семінару імені А.Я. Петренюка, присвяченого 70-річчю Льотної академії Національного авіаційного університету (Запоріжжя – Кропивницький, 13-15 травня 2021 року)* / за ред. Г.П. Донця, м. Запоріжжя – Кропивницький, 2021. С. 86-90. Режим доступу до ресурсу:
https://www.glau.kr.ua/images/docs/sbornik/materiali_23_mnp_seminaru.pdf#page=86.
9. **Y.Kovalenko, V.Turchina, O.Hurko.** On special classes of scheduling theory problems. *Сучасні науково-технічні дослідження у контексті мовного простору (англійською мовою): Матеріали X Регіональної науково-практичної конференції молодих науковців та студентів, 13 травня 2021 р. м. Дніпро, 2021. С. 28-30.* Режим доступу до ресурсу:
https://www.dnu.dp.ua/docs/ndc/2021/19_Сучасні%20науково-технічні%20дослідження%20у%20контексті%20мовного%20простору.pdf#page=28.