Ф.Г. ВАЩУК, О.Г. ЛАВЕР, Н.Я. ШУМИЛО

УжДІІЕП
UZHGOROD STATE INSTITUTE OF INFORMATIC SCIENCES ECONOMICS AND LAW

ТЕОРІЯ РОЗКЛАДІВ

Навчальний посібник

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
УЖГОРОДСЬКИЙ ДЕРЖАВНИЙ ІНСТИТУТ ІНФОРМАТИКИ,
ЕКОНОМІКИ І ПРАВА

Ф.Г.Ващук, О.Г.Лавер, Н.Я. Шумило

ТЕОРІЯ РОЗКЛАДІВ

Навчальний посібник

Ужгород – 2003

ББК 22.18
УДК 519.87
В 23

Навчальний посібник з дисципліни "Теорія розкладів" для викладачів та студентів факультету інформатики Ужгородського державного інституту інформатики, економіки і права, а також математичних та інженерно-технічних спеціальностей вищих навчальних закладів.

Рецензенти:

Павлов Олександр Анатолійович – доктор технічних наук, професор, завідувач кафедри автоматизованих систем обробки інформації та управління, декан факультету інформатики та обчислювальної техніки Національного технічного університету "Київська політехніка"

Кривий Сергій Лук'янович – доктор фізико-математичних наук, професор Українського державного університету харчових технологій

Автори:

Ващук Федір Григорович – доктор технічних наук, професор, завідувач кафедри загальної інформатики та математичного моделювання УжДІЕП, ректор УжДІЕП

Лавер Олександр Григорович – кандидат фізико-математичних наук, доцент, завідувач кафедри фізико-математичних дисциплін

Шумило Наталія Ярославівна – асистент кафедри фізико-математичних дисциплін

Рекомендовано до друку Вченою радою УжДІЕП, протокол № 6 від 06 березня 2003 р.

ISBN 966-7781-13-5

Передмова

Помітне в останній час зростання інтересу до питань побудови оптимальних розкладів для різних обслуговуючих систем обумовлено суттєвим підвищенням рівня автоматизації всіх видів людської діяльності, в тому числі й управління цією діяльністю. Якість функціонування сучасного виробництва багато в чому визначається рішеннями, які приймаються на етапах календарного планування й оперативного управління. Поряд із покращенням якості планових рішень все більш жорсткими стають вимоги до скорочення строків їх виконання, підвищення оперативності та гнучкості керування.

Не дивлячись на те, що методи і результати теорії розкладів представляють значний інтерес як для практиків, так і для теоретиків, вона не користується поки тою увагою, яку безсумнівно заслуговує, а розрізненні публікації з теорії розкладів з'являються в журналах багатьох різних напрямків.

Теорію розкладів можна порівняти з теорією запасів, хоча остання отримала набагато більший та більш систематичний розвиток. Це, ймовірно, пояснюється тим, що математичні моделі, які тут виникають, менш привабливі у порівнянні з моделями, до яких приводить теорія запасів, хоча слід відмітити, що практичний інтерес задач впорядкування не менший. Більш того, теорія розкладів у теперішньому, досить примітивному стані містить не менш важливі прикладні результати, ніж розроблена в більшій степені теорія запасів. Звичайно, тільки деякі із задач як в теорії розкладів, так і в теорії запасів розв'язані точно (система з одною машиною є не більш загальною, ніж задача з одним типом виробів в теорії запасів), однак обидві теорії дозволяють отримати цілий ряд цікавих висновків. Зокрема, зрозуміло, що врахування тривалості обслуговування, так само, як і врахування планових строків закінчення обслуговування складають важливі питання при впорядкуванні робіт.

Даний посібник написаний для студентів факультету інформатики спеціальності "Інформаційні управляючі системи та технології" з метою ознайомлення їх з колом задач та методів теорії розкладів для систем з детермінованими характеристиками. Наводиться багато числових прикладів та ілюстрацій.

Мета та задачі дисципліни

Курс теорії розкладів має своєю метою вивчення основних класів задач теорії розкладів, сучасних методів оптимізації в цій області дослідження операцій, а також здобуття навичків практичного застосування методів оптимізації в календарному плануванні при розв'язанні задач оптимального керування економічними об'єктами.

В результаті навчання *студент повинен знати*:
— існуючі підходи до розв'язання задач календарного планування;
— класи задач впорядкування, для яких існують ефективні методи розв'язання;
— методологію розв'язання NP-важких задач впорядкування;

повинен вміти:
— здійснити формалізацію постановки задачі й отримати математичну модель;
— вибрати метод оптимізації для отриманої математичної моделі;
— провести аналіз та дослідження отриманих результатів розв'язання задач.

При вивченні даної дисципліни використовуються знання студентів із математичного аналізу, дискретної математики (теорія множин, теорія бінарних відображень, теорія графів), математичних методів дослідження операцій.

Знання, отримані студентами при вивченні теорії розкладів, використовуються в курсах математична економіка і моделі функціонування автоматизованих систем управління, інтегровані автоматизовані системи управління, при виконанні курсових і дипломних робіт.

Вступ

1. **Предмет і задачі дисципліни. Основні поняття та визначення.**
2. **Математична модель.**
3. **Класифікація задач впорядкування.**
4. **Вхідні та шукані величини задачі.**
5. **Критерії оцінки систем.**
6. **Розклад і вартість.**
7. **Деякі області застосування результатів теорії розкладів у інформатиці та обчислювальній техніці.**
8. **Методи розв'язання задач теорії розкладів:**
   - **математичне програмування та теорія розкладів;**
   - **комбінаторний підхід;**
   - **евристичні та ймовірносні методи.**

Предмет і задачі дисципліни. Основні поняття та визначення

З поняттям розкладу (календарного плану, розпорядку дня, плану-графіку, режиму роботи і т.п.) кожна людина знайомиться, як тільки з'являється на світ. Дитину годують, укладають спати, гуляють з нею за годинами. Як вона підросте, то освоює режим роботи дошкільних закладів. Далі, школа з її розкладом занять... Розклад руху літаків, поїздів, автобусів... Розпорядок роботи магазинів, підприємств, закладів, організацій. Програми радіо і телепередач, розклад роботи кінотеатрів... Виробничі плани на зміну, добу, тиждень, квартал, рік... Кожна людина, по суті, кожний день складає розклад своєї діяльності, погоджуючи його з розкладом діяльності других людей. Для деяких людей це стає професійним обов'язком. Їм доводиться розробляти календарні плани роботи підприємств і закладів, складати розклад руху транспорту, організувати навчальний процес і т.д. Наша мета – вивчити те спільне, що характеризує такі задачі незалежно від їх конкретного змісту

Практично більшість задач впорядкування так чи інакше розв'язується, оскільки літаки приземляються, банківські клієнти влаштовують свої справи, ми вчимося за розкладом і основна маса населення непогано проводить свої вихідні дні. Однак більшість цих рішень приймаються інтуїтивно.

Інколи впорядкування здійснюється випадково; більш часто роботи виконуються в тому порядку, в якому вони виникають. Ми не задумуємося про розумність дисципліни перший прийшов – перший обслужений,

оскільки вона відповідає притаманному нам почуттю звичайного порядку. Однак така дисципліна, мабуть, і прийнята у магазинних чергах, але зовсім не обов'язкова для виконання робіт на підприємстві.

Формальна схема процесу впорядкування приблизно така:

Нехай: 1) α — це сукупність наслідків при виконанні спочатку задачі A, а потім B, 2) β — це сукупність наслідків при виконанні спочатку задачі B, а потім A. Тоді, якщо α краще, ніж β, то вибирається послідовність: "A, потім B."

Але подібна формалізація не універсальна.

Теорія розкладів являє собою єдину наукову дисципліну, яка вивчає розподільні задачі, в яких обмеженим ресурсом є час. Пропоновані нею методи знаходять застосування не тільки в управляючих системах, зв'язаних з матеріальним виробництвом, але і в системах, які координують роботу обчислювальних центрів, науково-дослідних інститутів, конструкторських бюро.

Виникнення і подальший розвиток теорії розкладів характеризувались спробами вивчити широке коло задач, починаючи з простішої задачі вибору послідовності виконання N робіт одним виконавцем і закінчуючи так званою загальною задачею, пов'язаною з аналізом багатоетапних технологічних процесів. Не дивлячись на простоту постановок, лише деякі задачі розв'язані точно, і одна з причин цього полягає в складності дослідження питань збіжності методів і алгоритмів, які пропонуються в тих чи інших випадках.

Побудова прийнятного календарного плану (розкладу) рівносильна узгодженню в часі багатьох виробничих операцій.

Розкладом можна назвати документ, який містить відомості: про кількість і номенклатуру виконуваних робіт, включаючи їх етапи; про моменти початку і закінчення кожної роботи; про місце і технічні засоби виконання кожної роботи; про витрати часу (і матеріальних ресурсів) на всі виконувані роботи. Цих відомостей досить для формального представлення розкладів, хоча на практиці вони можуть доповнюватися і уточнюватися в інтересах більш повного врахування тої реальності, яка відображена в моделі.

Математична модель

При операційному підході до пошуку найбільш ефективних шляхів досягнення мети здійснюється побудова *математичної моделі,* яка містить опис цієї моделі та відображає умови поведінки операції. На цьому етапі аналізується зміст операції, визначаються необхідні дії, виявляються умови їх виконання, оцінюються різні обмеження і т.п.

Оцінка і порівняння ефективності можливих способів дій при досягненні поставленої мети проводяться на основі побудованої моделі. Ця ж модель дозволяє прийняти й найкраще в розглядуваній ситуації рішення.

У середині 50-их років почалися інтенсивні дослідження з побудови й аналізу *моделей календарного планування* та розробки методів прийняття планових рішень з використанням цих моделей.

Часова прив'язка всієї множини дій, пов'язаних із досягненням заданої мети, уже сама по собі досить складна задача. Якщо ж мова йде про побудову найкращого календарного плану, та ще й у найскоріший строк, то складність цієї задачі дуже зростає. Традиційні методи календарного планування все менше відповідали запитам практики. За цих умов застосування математичних методів і обчислювальної техніки відкривало певні перспективи та дозволяло надіятися на успішне подолання труднощів, що виникають.

Коло питань, пов'язаних з побудовою найкращих календарних планів (розкладів), особливо з розробкою математичних методів отримання рішень з використанням відповідних моделей, вивчається в рамках *теорії розкладів*.

Ця теорія використовує характерний для дослідження операцій модельний підхід до аналізу реальних процесів. Моделі, які вивчаються в теорії розкладів, відображають специфічні ситуації, що виникають при календарному плануванні різних видів цілеспрямованої людської діяльності.

Слід відмітити, що вже на початковій стадії розвитку теорії розкладів стало ясно, що задачі оптимального календарного планування досить трудомісткі.

При побудові математичних моделей календарного планування серйозні труднощі викликає недостатня поінформованість дослідника про аналітичний процес.

Основним поняттям теорії розкладів є поняття операції. *Операцію* можна розглядати як елементарну задачу, яка підлягає виконанню.

Кожна операція характеризується:

1) індексом належності до певної роботи (перший індекс);

2) індексом належності до певної машини (другий індекс);

3) числом, яке являє собою тривалість операції.

По першому індексу вся множина операцій розбивається на повну систему підмножин, що не перетинаються. Ці підмножини називаються *роботами*. Розбиття заданої множини по другому індексу призводить до отримання підмножин (що не перетинаються) операцій, які відносяться до певних *машин*.

Для кожної роботи задається послідовність операцій (яка визначається технологічним процесом). Таке часткове впорядкування операцій здійснюється заданням *відношення порядку*. Якщо x і y — дві

операції одної й тої ж роботи, і якщо по деяким міркуванням x повинна виконуватися раніше, ніж y, то кажуть, що x *передує* y. Це записується у вигляді: $x \prec y$. Відношення $x \prec y$ можна записати також у вигляді $y \succ x$ і сказати, що операція y *слідує за* x.

Відношення порядку транзитивне:

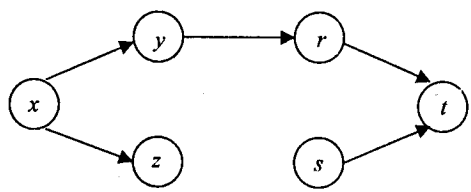
якщо $x \prec y$, $y \prec z$, то $x \prec z$.

Будемо казати, що операція x *безпосередньо передує* операції y (або операція y *безпосередньо слідує за* операцією x), і записувати

$$x \prec\prec y \text{ або } y \succ\succ x,$$

якщо $x \prec y$ і нема операції z, такої, що $x \prec z \prec y$.

Часто буває зручно представляти згадані відношення у вигляді наступного орієнтованого графа. Вершини (вузли) графа відображають операції, а дуги – відношення безпосереднього слідування. Дві вершини зв'язані відношенням порядку, якщо існує шлях між ними.



Машиною будемо називати прилад, здатний виконувати все, що пов'язано з деякою операцією; *системою обслуговування* – множину всіх машин, які використовуються для виконання деякої підмножини операцій. Сукупність машин, робіт (операцій) і дисциплін призначення операцій відповідним машинам назвемо *процесом обслуговування*. Поняття "машина" будемо вживати в узагальненому розумінні, яке включає в себе сукупність людей, машин і допоміжного обладнання, необхідних для виконання операції.

Складання розкладу для процесу обслуговування означає, що для кожної операції на часовій осі задається ділянка, коли ця операція повинна виконуватися відповідною машиною, тобто, що для кожної операції задається один або більше інтервалів $(b_1, c_1), (b_2, c_2), \ldots$ таких, що

1) $(c_1 - b_1) + (c_2 - b_2) + \ldots \geq$ тривалості операції;

2) число b_{1x} — значення b_1, що відноситься до x, не менше, ніж b_{1y} для двох операцій x і y, що відносяться до одної роботи і таких, що $y \prec x$;

3) кожний із інтервалів (b_i, c_i) розміщений цілком в середині одного з відрізків часу доступності відповідної машини.

З другого боку, розклад може розглядатися як задача *впорядкування операцій, виконуваних кожною машиною.* Це впорядкування в деякому розумінні аналогічно тому, яке задане для операцій кожної роботи. Можна вказати багато дисциплін, які формують ту чи іншу послідовність (черговість). Майже вся теорія, розроблена на даний час, відноситься до досить обмеженого числа моделей *простого процесу обслуговування.* Під останнім розуміють процес, для якого суттєві наступні обмеження:

1) Кожна машина може бути назначена в любий момент часу. Машини не можуть виходити з ладу і, отже, бути недоступними із-за поломок чи ремонту. Вони не можуть бути недоступними також із-за перезмінок, характерних для виробництва.

/ Кожна машина формально являє собою інтервал $(0,T)$, *де* T — *це довільне велике число.*

2) Роботи являють собою строго впорядковані послідовності операцій. Із окремих операцій цих послідовностей не може формуватися ніяка нова послідовність.

/ Для заданої операції x *існує не більше одної операції* y *такої, що* $y \prec\prec x$, *і одної операції* z *такої, що* $x \prec\prec z$.

3) Кожна операція виконується тільки одною машиною.

4) Існує тільки по одній машині кожного типу.

/ Між номерами машин і другими індексами операцій, що вказують номер машини, що їх виконує, існує взаємно однозначна відповідність.

5) Відсутні переривання операцій. Це означає, що якщо деяка машина почала виконувати операцію, то ця операція повинна бути завершена перш, ніж на цій машині почнеться виконання якої-небудь другої операції.

/ Для кожної операції заданий єдиний інтервал $(в,c)$, *причому довжина операції рівна* $(b-c)$.

6) Інтервали виконання послідовних операцій одної й тої ж роботи не перетинаються. Одночасно не може виконуватися дві операції одної й тої ж роботи.

/ Величина b_x, *що відповідає призначенню операції* x, *не може бути меншою* c_y *для довільної операції* y *такої, що* $y \prec x$.

7) В кожний момент часу машина може виконувати не більше одної операції.

/ Розглянемо інтервал (b_x,c_x), *який назначений для виконання операції* x *деякою машиною. Тоді для всякої другої операції, що здійснюється цією машиною в інтервалі* (b_y,c_y), *буде*

$$\text{або } b_x \geq c_y, \text{ або } c_x \leq b_y.$$

Задача теорії розкладів вважається заданою, якщо визначені:

1) роботи і операції, що підлягають виконанню;

2) кількість і типи машин, які виконують операції;
3) порядок проходження машин;
4) критерії оцінки розкладу.

Класифікація задач впорядкування

Найпростішим і найбільш вивченим класом задач теорії розкладів є задачі *впорядкування.* В цих задачах розподіл робіт по блокам і тривалостям їх виконання припускається заданим. Необхідно вказати найбільш ефективну стратегію управління чергами вимог на виконання робіт кожним блоком.

В задачах *узгодженості* основна увага приділяється вибору тривалості робіт при заданому їх розподілі по блокам.

В задачах розподілу припускається, що один і той же блок може виконувати різні роботи або набори робіт. Необхідно вказати в деякому розумінні найкращий розподіл робіт по блокам.

У всіх перерахованих класах задач теорії розкладів сукупність робіт і структура системи, що їх реалізовує, припускаються заданими і незмінними.

I. Найбільш відомою в теорії розкладів задачею *впорядкування* є задача побудови оптимального (в тому чи іншому розумінні) розкладу процесу обробки "деталей" деякою сукупністю "машин".

Процес обробки кожної деталі включає послідовне виконання деякої множини операцій. Кожна операція виконується деякою машиною. Тривалість виконання кожної операції припускається відомою. Вона може бути як детермінованою, так і випадковою величиною.

Кожна машина одночасно вміє виконувати не більше одної операції.

При побудові найкращого, скажімо по швидкодії, розкладу процесу обробки всіх деталей необхідно вирішити, в якій саме послідовності повинні виконуватися операції кожною машиною.

Якщо одна і та ж операція може бути виконана не на одній машині, а на довільній із деякої сукупності машин, то поряд із впорядкуванням операцій необхідно провести і розподіл їх по машинам.

II. Класичною задачею узгодженості є задача визначення тривалостей виконання робіт заданої сукупності з метою мінімізації сумарної вартості робіт при заданому загальному часі виконання всіх робіт сукупності.

Кожна робота характеризується тривалістю і вартістю її виконання. Припускається, що тривалість виконання робіт може змінюватися в заданих межах. Вартість виконання роботи залежить від тривалості.

Роботи розглядуваної сукупності не є незалежними. Для кожної роботи відома множина робіт, після виконання яких може бути початe виконання даної роботи.

Необхідно вибрати тривалість кожної роботи (в заданих межах) таким чином, щоб загальний час виконання всіх робіт не перевищував заданої величини, а сумарна вартість їх виконання була найменшою. Нас може цікавити також зменшення загального часу виконання всіх робіт при заданій сумарній вартості їх виконання.

III. Найпростішою розподільною задачею є задача балансування "складальної лінії".

У цій задачі необхідно розподілити деяку сукупність робіт по робочим місцям, розміщеним у лінію. Для кожної роботи відома тривалість її виконання і множина робіт, після завершення яких може бути початe виконання даної роботи.

Із врахуванням заданих відношень порядку роботи розподіляються таким чином, щоб сумарний час виконання робіт на кожному робочому місці не перевищував задану величину, і число робочих місць було мінімальним.

Задачі теорії розкладів відрізняються числом виконуваних в системі робіт, характером поступлення їх в систему і порядком участі окремих машин у виконанні конкретної роботи.

В залежності від характеру поступлення робіт розрізняють два види задач: *статичні* і *динамічні.*

В *статичних* задачах, якщо система вільна, в неї одночасно поступає певне число робіт. Після цього нові роботи не поступають і розклад складається для повністю визначеного і відомого наперед числа робіт.

В *динамічних* задачах виконання робіт відбувається неперервно. Роботи поступають в систему в деякі моменти часу, які можна передбачити тільки в статистичному значенні. Тому моменти наступних поступлень не визначені.

Отже, впорядкування в динамічних і статичних задачах потребує зовсім різних методів розв'язання.

Для класифікації задач теорії розкладів у подальшому використовується запис **A|B|C|D**, де

A характеризує процес поступлення робіт. Для динамічних задач **A** являє собою функцію розподілу часу між поступленнями. Для статичних задач **A** відповідає числу одночасно поступивших робіт, якщо з цього приводу нічого спеціально не оговорено. Якщо на місці **A** стоїть n, то це означає довільне, але скінчене число робіт в статичному випадку.

B характеризує число машин в системі. Якщо на місці **B** стоїть m, то це означає довільну кількість машин.

C характеризує порядок (дисципліну) виконання робіт машинами. Якщо на місці **C** стоїть **F**, то це відповідає конвеєрній системі; якщо **R** – то

випадковій; якщо **G** – довільній. Для системи, що складається з одної машини, вказані дисципліни гублять зміст, і тому для такої системи третій параметр опускається.

D характеризує критерій оцінки розкладу.

Загальна задача теорії розкладів, для якої розв'язок ще не отримано, запишеться так:

$$\mathbf{n|m|G|F_{max}} —$$

впорядкувати n робіт в довільній системі із m машин так, щоб мінімізувати максимальну тривалість проходження роботи.

Вхідні та шукані величини задачі

Різноманітність критеріїв, що використовуються для оцінки розкладів, частково обумовлена різними ситуаціями при їх складанні. Щоб врахувати особливості, характерні кожній із таких ситуацій, вводяться додаткові величини, які відображають їх специфіку. Часто вибір того чи іншого критерію диктується можливостями розв'язання задачі, і ці критерії виявляються відмінними від найбільш природних.

У цьому пункті вводяться величини, які використовуються далі як критерії оцінки розкладів, та вводяться деякі відношення між цими величинами.

Необхідно чітко уявляти собі різницю між *вхідними* і *шуканими величинами* задачі. Вхідні визначаються специфікою розв'язуваної задачі. Шукані є результатом складання розкладу. Вхідні дані будемо позначати маленькими латинськими буквами x, y, z, h а шукані – великими латинськими буквами X, Y, Z, H.

Вхідні величини.

Будемо говорити про систему із m машин і вважати, що кожна машина ототожнюється з відповідним індексом $1,2,\ldots,m$. Пронумеруємо роботи числами від 1 до n, і нехай

r_i – *момент готовності*, або *момент появи*, або *момент поступлення*. Ця величина являє собою момент поступлення i-ї роботи в систему із деякого зовнішнього джерела. Суттєво, що r_i є мінімальний можливий час початку першої із операцій роботи i.

d_i – *плановий строк*. Ця величина являє собою момент, до якого i-а робота повинна бути виконана.

Допустима тривалість проходження роботи в системі дорівнює

$$a_i = d_i - r_i.$$

Робота i складається з g_i операцій. Для кожної операції задається набір наступних величин:

$$\begin{array}{ll} m_{i1}, & p_{i1}, \\ m_{i2}, & p_{i2}, \\ \vdots & \vdots \\ m_{ig_i} & p_{ig_i}, \end{array}$$

де m_{ij} – номер машини, на якій виконується операція j; $1 \le m_{ij} < m$; p_{ij} – тривалість виконання операції, тобто довжина інтервалу часу, необхідного машині m_{ij} для виконання операції j.

Нехай $p_i = \sum_{j=1}^{g_i} p_{ij}$ — загальна тривалість всіх операцій роботи i.

Нерідко робота складається із окремих частин, кожна з яких вимагає аналогічного виконання. В цих випадках вся робота визначається як сума її складових частин, а тривалість кожної її операції виражається добутком тривалості відповідної операції одної її частини на загальне число частин. Далі будемо вважати, що число частин, із яких складається робота, нам відоме до складання розкладу, і воно не змінюється після його складення.

Нарешті, припустимо, що p_{ij} включає в себе всі настройки машини, що передують виконанню операції, і всі перенастройки її після виконання операції. Це означає, що час, необхідний для підготовки машини до виконання деякої операції, не залежить від того, яку операцію останньою виконувала машина. У більшості випадків таке припущення є хорошим наближенням до дійсності, і значно спрощує математичну модель.

Шукані величини задачі.

В задачах "ідеального" впорядкування встановлюється тільки послідовність виконання операцій кожної роботи. Другими словами, це означає, що знаходиться тривалість очікування початку кожної операції кожної роботи.

Позначимо через W_{ij} інтервал часу між закінченням $(j-1)$-ої та початком j-ої операції i-ої роботи. Тоді загальна *тривалість очікування* для роботи i рівна сумі тривалостей очікування всіх її операцій:

$$W_i = \sum_{j=1}^{g_i} W_{ij}.$$

Результатом складання розкладу завжди є *задання множини чисел* W_{ij}.

Найбільш важливими величинами, що є функціями W_{ij}, будуть:

1) момент закінчення роботи;
2) тривалість проходження роботи в системі;
3) різниця між плановим строком і моментом фактичного закінчення роботи.

Ці величини позначаються наступним чином:

C_i – *момент закінчення роботи* i, тобто момент закінчення її останньої операції:

$$C_i = r_i + W_{i1} + p_{i1} + W_{i2} + p_{i2} + \ldots + W_{ig_i} + p_{ig_i} = r_i + \sum_{j=1}^{g_i} p_{ij} + \sum_{j=1}^{g_i} W_{ij} = r_i + p_i + W_i$$

;

F_i – *тривалість проходження роботи* i в системі, тобто:

$$F_i = W_{i1} + p_{i1} + W_{i2} + p_{i2} + \ldots + W_{ig_i} + p_{ig_i} = \sum_{j=1}^{g_i} p_{ij} + \sum_{j=1}^{g_i} W_{ij} = p_i + W_i = C_i - r_i.$$

Тривалість проходження називають ще *циклом обробки* або *виробничим циклом.*

Часове зміщення L_i роботи i дорівнює

$$L_i = C_i - d_i = F_i - a_i.$$

Визначимо *запізнення* T_i та *випередження* E_i роботи i як

$$T_i = \max(0, L_i), \qquad E_i = \max(0, -L_i)$$

Часове зміщення, запізнення та випередження оцінюють фактичний час закінчення роботи в порівнянні з її плановим строком. Часове зміщення кожної роботи може мати любий знак. Якщо воно додатне, тобто робота завершується після планового строку, то воно дає значення запізнення, а якщо від'ємне, тобто робота завершується до планового строку, то модуль зміщення дає значення випередження.

Розклад повністю описується множиною величин W_{ij}. Два розклади для одної й тої ж задачі тотожні, якщо їх множини W_{ij} ідентичні. Але можна, наприклад, вважати еквівалентними всі розклади, в яких співпадають моменти закінчення робіт (це не строга тотожність).

Взагалі, встановлення *критерію оцінки* в теорії розкладів означає задання множини еквівалентних класів і відношення переваги серед них. Наприклад, прийняття середнього значення часу закінчення роботи в якості критерію оцінки розкладу означає:

1) всі розклади, в яких однаковий середній час закінчення роботи $\overline{C}'$, еквівалентні, так що все рівно, який із них вибрати;

2) розклад із середнім $\overline{C}'$ має перевагу над розкладом із середнім $\overline{C}''$, тоді і тільки тоді, коли $\overline{C}' < \overline{C}''$.

Розклад S є *оптимальним* відносно деякого критерію, якщо він належить еквівалентному класу $\{S'\}$ такому, що не існує якого-небудь (не пустого) класу, який має перевагу над $\{S'\}$.

До критеріїв оцінки відносять середнє і максимальне значення моментів часу закінчення робіт і тривалість проходження, а також середні і максимальні значення часового зміщення і запізнення. Усі ці критерії належать до одного класу *регулярних критеріїв*. Регулярний критерій M є зростаючою функцією моментів закінчення кожної із n робіт

$$M = f(C_1, C_2, \ldots, C_n).$$

Таким чином, якщо $M' = f(C_1', C_2', \ldots, C_n')$, то $M' > M$, якщо $C_i' \geq C_i$, $i = \overline{1,n}$, хоча б для одного i.

Середні значення приведених вище величин тісно зв'язані один з одним. Для довільної роботи мають місце наступні співвідношення:

$$L_i = F_i - a_i = C_i - r_i - a_i = C_i - d_i.$$

У випадку n робіт складається n відповідних співвідношень

$$\sum_{i=1}^{n} L_i = \sum_{i=1}^{n} F_i - \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} C_i - \sum_{i=1}^{n} r_i - \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} C_i - \sum_{i=1}^{n} d_i.$$

Поділивши кожний член написаної рівності на n, отримаємо

$$\overline{L} = \overline{F} - \overline{a} = \overline{C} - \overline{r} - \overline{a} = \overline{C} - \overline{d}.$$

У кожному конкретному випадку, $\overline{a}$, $\overline{r}$ і $\overline{d}$ задані й не залежать від ввідного впорядкування. Тому для всякого розкладу $\overline{L}$ відрізняється від $\overline{F}$ точно на $\overline{a}$, від $\overline{C}$ – точно на $\overline{d}$ і т.д. Звідси слідує, що розклад, оптимальний відносно $\overline{F}$, оптимальний відносно $\overline{C}$ і $\overline{L}$.

Так як кожна із наведених величин визначається тривалістю очікування, то середні тривалість очікування також є регулярним критерієм

$$C_i = r_i + W_i + p_i, \qquad \overline{C} = \overline{r} + \frac{\sum_{i=1}^{n} W_i}{n} + \frac{\sum_{i=1}^{n} p_i}{n}.$$

Оскільки у виразі для $\overline{C}$ перший і останній члени сталі, то розклад, що мінімізує середній час очікування робіт, мінімізує також середню тривалість очікування.

Середнє запізнення і середнє випередження зв'язані із середнім часовим зміщенням співвідношеннями

$$T_i - E_i = L_i, \qquad \overline{T} - \overline{E} = \overline{L}.$$

Відмітимо, що звідси не слідує, що розклад, який мінімізує середнє запізнення, мінімізує також середнє зміщення.

Критерії оцінки системи

Іноді буває зручно оцінювати розклад величинами, які відносяться не до робіт, а до системи обслуговування. Із цих величин найбільш важливими є *коефіцієнт використання* та *об'єм роботи в системі*. Коефіцієнт використання характеризує виробничу потужність машини й представляє відношення тривалості зайнятості її виконанням роботи до загального часу, коли вона доступна і може бути використана.

Припускається, що всі машини доступні на протязі виконання всіх n робіт, тобто від $r_{\min}$ до $C_{\max}$ і всі роботи з'являються в системі одночасно. При цих припущеннях коефіцієнт використання обернено пропорційний максимальній тривалості проходження:

$$\overline{U} = \frac{\sum_{i=1}^{n} p_i}{mF_{\max}}.$$

Існує два принципово різних способи вимірювання об'єму роботи, що міститься в системі: можна рахувати число робіт, що знаходяться в системі, або їх сумарну тривалість.

Для вираження роботи, що міститься в системі, використовують наступні величини:

$N(t)$ – число робіт у системі в момент часу t;

$\overline{N}(t_1,t_2)$ — середнє число робіт в системі на протязі інтервалу часу (t_1,t_2):

$$\overline{N}(t_1,t_2) = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} N(t)dt;$$

$P(t)$ – сумарна тривалість виконання всіх операцій для всіх робіт, що знаходяться в системі в момент часу t. Ця величина називається *об'ємом роботи*;

$\overline{P}(t_1,t_2)$ – середній об'єм роботи за інтервал часу (t_1,t_2):

$$\overline{P}(t_1,t_2) = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} P(t)dt.$$

Розклад і вартість

Практично, питання про те, коли і в якому порядку виконуються роботи, як правило, впливає на величину витрат, пов'язаних із їх виконанням, але в ідеалізованих задачах чистого впорядкування цей вплив не враховується.

Припущення, що множина робіт визначена наперед і не залежить від впорядкування, означає, що загальний прибуток системи фіксований або ніяк не зв'язаний із розкладом.

Припущення, що ефективність використання обладнання також не залежить від розкладу, означає, що нами ігноруються всі вартості, що зазвичай називаються прямими цінами. В дійсності ціни, які можуть бути поставлені у відповідність розв'язкам чистого впорядкування, являють собою в більшій степені ціни виробництва, ніж ціни товарів.

Існує три види вартості, якими визначається складений розклад. Це вартість експлуатації машини, вартість зберігання робіт і вартість їх затримок.

Витрати на зберігання зазвичай оцінюються в 2-3% від вартості запасів у місяць. Вони залежать від фізичного об'єму запасів. У залежності від обставин, витрати можуть визначатися числом робіт, об'ємом незавершеної чи завершеної роботи. Завжди існують економічні причини, що змушують прагнути до зменшення середніх розмірів запасів. Цим пояснюється прагнення зменшити середню тривалість проходження робіт. Це прагнення також пояснюється конкуренцією, так як зменшення тривалості проходження дозволяє виробити більше товарів і, відповідно, знизити ціни.

Коефіцієнт використання є дуже важливим економічним фактором, що характеризує складений розклад. Якщо розклад такий, що простої приладів у ньому мінімальні або мінімальна середня тривалість проходження робіт, то це означає, що ті ж самі прилади за один і той же час виконують більшу роботу, ніж при іншому розкладі. З другої сторони, правильно складений розклад дозволяє виконати ту саму роботу меншою кількістю приладів.

У ряді випадків більш зручною економічною оцінкою є вартість затримки.

Деякі області застосування результатів теорії розкладів у інформатиці та обчислювальній техніці

1. Організація великих інформаційних систем – створення апаратно-програмних комплексів паралельної обробки даних (масово-паралельних систем МРР).
2. Вдосконалення мікроархітектури процесорів (одна із особливостей процесора Р6 – наявність блоку диспетчера/обробки, який планує розклад обробки суб'єктів мікрооперацій з урахуванням їх взаємозалежності за даними та наявним ресурсом).
3. Розробка програмного забезпечення планування колективної роботи (управління проектами).

Методи розв'язання задач теорії розкладів

"Часовий" характер задач теорії розкладів виділяє їх в особливий клас, що суттєво відрізняється від "об'ємних" економічних задач. Якщо в останніх вимагається відповісти на питання що і скільки виготовляти, то в задачах теорії розкладів необхідно визначити коли, в якій послідовності виконувати роботи. Ця відмінність в сутності задач визначає відмінності в

методах і можливостях їх розв'язання. Для задач об'ємного характеру розвинутий досить сильний апарат, головним чином математичного програмування, який дозволяє з успіхом добиватися їх розв'язків. Для задач теорії розкладів такий апарат розвинутий ще досить слабо.

Пошук оптимального чи близько до оптимального розкладу здійснюється за допомогою одного з чотирьох підходів:

1) математичного програмування;
2) комбінаторного;
3) евристичного;
4) статистичного (ймовірнісного).

1. Математичне програмування та теорія розкладів

Основи теорії розкладів розвивались в той час, коли математичні моделі почали застосовуватися для розв'язання економічних задач. Робилися змоги побудувати математичні моделі й для задач теорії розкладів. При цьому зіткнулися з такою проблемою. У математичній моделі система обмежень відображає те положення речей, що деяка сукупність умов повинна виконуватись сумісно. В задачах теорії розкладів ряд умов повинні виконуватись альтернативно: або i-а робота запускається раніше j-ої, або навпаки.

Формулювання загальної задачі складання розкладу в термінах лінійного цілочисельного програмування

Нехай маємо систему із n робіт та m машин. Кожна робота складається із g_i операцій. Кожній операції приписується три індекси:

i – номер роботи, що містить цю операцію;

j – номер операції в середині роботи, $j = 1,..., g_i$;

k – номер машини, на якій операція повинна виконуватися.

Обмеження на час і порядок виконання операцій машинами такі:

1) кожна машина виконує одночасно не більше одної операції;
2) операції виконуються у вказаній послідовності;
3) ніякі дві операції, що відносяться до одної роботи, не виконуються одночасно.

Для спрощення викладу і позначень приймемо, що кожна робота вимагає в точності одного виконання на кожній із машин $(g_i = m, i = \overline{1,n})$.

Нехай t_{ik} – тривалість виконання роботи i машиною k;

$$r_{ijk} = \begin{cases} 1, \text{ якщо операція } j \text{ роботи } i \text{ виконується машиною } k, \\ 0, \text{в противному випадку;} \end{cases}$$

$\underline{t}_{ik}$ – момент початку виконання роботи i машиною k (дорівнює початку виконання відповідної операції роботи i машиною k).

Із того, що кожна машина в один момент часу може виконувати не більше одної роботи, випливає, що для кожної пари робіт I та J виконується лишень одна із нерівностей:

$$\begin{cases} \underline{t}_{Ik} - \underline{t}_{Jk} \geq \underline{t}_{Jk} - \text{виконанню роботи } I \text{ передує виконання роботи } J \\ \text{або} \\ \underline{t}_{Jk} - \underline{t}_{Ik} \geq \underline{t}_{Ik} - \text{виконанню роботи } J \text{ передує виконання роботи } I. \end{cases} \quad (1)$$

Таке обмеження типу "або-або" не можна описати в рамках звичайного лінійного програмування і вимагає введення цілочисельних змінних.

Нехай

$$Y_{IJk} = \begin{cases} 1, \text{ якщо робота } I \text{ передує роботі } J \text{ на машині } k \text{ (не обов'язково безпосередньо)}, \\ 0, \text{ в противному випадку.} \end{cases}$$

Зрозуміло, що із двох величин Y_{IJk} і Y_{JIk} досить задати одну.

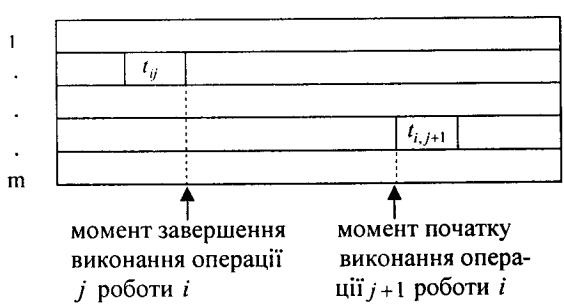
Тепер сформульовані вище обмеження типу "або-або" можна записати у вигляді двох умов, кожна із яких повинна бути виконана:

$$(M + t_{Jk})Y_{IJk} + (\underline{t}_{Ik} - \underline{t}_{Jk}) \geq t_{Jk}, \quad (2)$$

$$(M + t_{Ik})(1 - Y_{IJk}) + (\underline{t}_{Jk} - \underline{t}_{Ik}) \geq t_{Ik}, \quad (3)$$

де $M = \sum_i \sum_k t_{ik}$ – досить велика константа, вибрана так, щоб виконувалась тільки одна із двох умов: $Y_{IJk} = 0$ або $Y_{IJk} = 1$.

Нехай I передує J, тобто $\underline{t}_{Ik} < \underline{t}_{Jk}$ і $Y_{IJk} = 1$. Тоді (3) повністю співпадає з другою умовою "або-або" в (1), а (2) завдяки великому параметру M перетворюється в зайве обмеження, яке не суперечить цілій системі в цілому.



$\sum_k r_{ijk}\underline{t}_{ik}$ – момент початку виконання операції j роботи i. Тоді для всіх операцій, крім останньої операції кожної роботи повинна мати місце нерівність

$$\sum_k r_{ijk}(\underline{t}_{ik} + t_{ik}) \le \sum_k r_{i,(j+1),k}\underline{t}_{ik} \qquad (4)$$

Отже, для задачі з m машинами та n роботами множина змінних і обмежень виглядає наступним чином:

змінні	*їх кількість*
$\underline{t}_{ik} \ge 0$	$m \times n$
$Y_{IJk} = 0$ або 1	$m \times n \times (n-1)/2$
обмеження	*їх кількість*
(2)	$m \times n \times (n-1)/2$
(3)	$m \times n \times (n-1)/2$
(4)	$n \times (m-1)$

Навіть для невеликих задач виходить дуже велика система нерівностей: якщо $m = 4$, $n = 10$, то число змінних буде 220, а обмежень – 390.

Цільові функції можуть бути різними. Так, мінімізація сумарного часу завершення робіт рівносильна мінімізації суми моментів початку виконання останніх операцій всіх робіт:

$$\min \sum_i \sum_k r_{imk}\underline{t}_{ik} .$$

При мінімізації максимального часу завершення робіт $T_{\max}$ додається обмеження виду

$$\sum_k r_{imk}(\underline{t}_{ik} + t_{ik}) \le T_{\max}, \quad i = \overline{1,n},$$

де $T_{\max}$ – змінна, яку потрібно мінімізувати.

При застосуванні методів математичного програмування для розв'язання задач теорії розкладів характерна експоненціальність часу розв'язання задачі. До прийомів, які найбільш широко використовуються для скорочення перебору відносяться прийоми, що базуються на методі гілок і границь або на методі наявного перебору. Ці прийоми полягають у побудові "часткових розв'язків", представлених у вигляді дерева пошуку та застосуванні методів побудови оцінок, що дозволяють відсікати безперспективні частинні розв'язки. Але навіть досконалі прийоми скорочення перебору не дозволяють уникнути експоненціальної трудоємності.

2. Комбінаторний підхід

Зводиться до цілеспрямованої перестановки пар робіт у деякій вихідній послідовності, поки не буде отриманий оптимальний (близький до оптимального) розв'язок.

Згадаємо спочатку визначення таких понять, як задачі класу P, ефективні алгоритми та NP-повні задачі.

Під ефективним алгоритмом розуміють алгоритм, для якого число необхідних кроків росте як поліном від розміру вхідної задачі. Задачі, що мають ефективні (поліноміальні) алгоритми розв'язання, належать класу P-задач.

Клас NP-задач володіє наступними властивостями:

1) ніяку NP-повну задачу не можна розв'язати ніякими відомими поліноміальними алгоритмами;
2) якщо існує поліноміальний алгоритм для деякої NP-повної задачі, то існує поліноміальний алгоритм для всіх NP-повних задач.

Практичне значення NP-повноти полягає в наступному: такі задачі по суті важкорозв'язні з обчислювальної точки зору, вони не піддаються ефективному алгоритмічному розв'язанню і для алгоритму, що коректно розв'язуватиме NP-повну задачу, знадобиться в гіршому випадку експоненціальна кількість часу і, отже, він не буде застосованим на практиці ні до яких, за винятком дуже малих, задач.

Приклад. *Задача Джонсона (поліноміально розв'язна задача).*

Маємо конвеєр із двох машин: $m = 2$. Кожна робота складається із двох операцій з тривалостями a_i та b_i. Мінімізується загальний час обслуговування робіт.

Послідовність, мінімізуюча загальний час роботи, така: спочатку запускаються роботи, для яких $a_i \le b_i$ у порядку неспадання a_i, далі всі роботи, для яких $a_i > b_i$ в порядку незростання b_i (тим самим мінімізуються простої другої машини із-за того, що перша ще не встигла обробити деяку роботу). Доведена оптимальність такої послідовності. Однак на випадок $n > 2$ результати не поширюються. NP-повнота задачі є вагомим доводом при обґрунтуванні необхідності побудови наближених чи евристичних алгоритмів її розв'язання, застосування схем направленого перебору варіантів (таких, як метод послідовного конструювання, аналізу та відсіювання варіантів (узагальнення методу гілок і границь)), а також при обґрунтуванні необхідності, дослідження частинних випадків задачі.

Павловим О.А. та ін. запропонований такий шлях розв'язання NP-повних задач:

1) досліджується важкорозв'язна задача і знаходяться теоретичні властивості, яким задовольняється їх оптимальний розв'язок;

2) на основі цих властивостей розробляється поліноміальний алгоритм розв'язання, при цьому вводиться поняття поліноміальної розв'язності задачі із класу NP, під яким розуміється існування поліноміального алгоритму, що задовольняє наступним умовам:
   а) якщо при розв'язанні довільної індивідуальної задачі виконуються певні аналітичні умови, то ця індивідуальна задача розв'язується строго (тобто отримано строго оптимальний розв'язок);
   б) у результаті роботи поліноміального алгоритму завжди відомо розв'язана чи ні дана індивідуальна задача точно;
   с) поліноміальний алгоритм є ефективним і статистично значимим, тобто при моделюванні довільних індивідуальних задач у більшості випадків отримані розв'язки є точними.

3. Евристичні та ймовірносні методи

Незадовільний стан розвитку точних методів розв'язання задач теорії розкладів обумовив розробку наближених методів, що дозволяють отримати прийнятні розв'язки при порівняно невеликих витратах часу та засобів. Умовно наближені методи поділяються на *евристичні* та *ймовірносні*.

Евристичні алгоритми засновані на прийомі, який називається *прийомом зниження вимог*. Він полягає у відмові від пошуку оптимального розв'язку за прийнятний час. Евристичні алгоритми використовують різні розумні висновки без строгих доведень.

Широко застосовується так званий метод локального пошуку. При цьому наперед вибрана множина перестановок використовується для послідовного покращення початкового розв'язку до тих пір, поки таке покращення можливе. У противному випадку виявляється досягнутим локальний оптимум.

Ще один із напрямків евристичних методів розв'язання задач теорії розкладів полягає у формуванні правил або функцій надання переваг (пріоритетів). Для кожної i-ої роботи із множини очікуючих виконання робіт, обчислюється значення функції f_i пріоритету і вибирається та робота, для якої f_i досягає максимуму чи мінімуму.

Приклади правил пріоритетів:

1. Правило SPT (shortest processing time). Перевага надається тій роботі (операції) з множини готових до обробки на звільненій машині, в якої час виконання на цій машині мінімальний.

2. Правило LRT (longest remaining time). Вимагає вибору напруженої роботи, тобто тої, в якої сума часу виконання операцій, що залишилися, максимальна.
3. Правило LPT (longest processing time). Перевага надається тій роботі (операції) із множини готових до обробки на звільненій машині, в якої час виконання на цій машині максимальний.

Перевагою евристичних методів є зручність реалізації їх на ЕОМ навіть при розв'язанні громіздких задач.

Недоліки евристичних методів полягають у складності оцінки близькості отриманих розкладів до оптимального. Крім того, для кожної функції пріоритету існують задачі, для яких застосування даної функції приводить до поганих результатів. Один із шляхів удосконалення метода функцій пріоритетів полягає в їх прив'язці до класів задач.

Ймовірносні методи пов'язані з k-кратним моделюванням розкладів. Вибір робіт із множини очікуючих виконання здійснюється випадковим чином. Після k-кратного застосування вибирається найкращий розклад, який приймається за розв'язок задачі. При цьому розрізняють:

а) ненаправлений випадковий пошук;
б) направлений випадковий пошук без самонавчання;
в) направлений випадковий пошук із самонавчанням.

Елементи комбінаторного аналізу

1. **Множини, відношення, відображення, графи.**
2. **Впорядкованість.**
3. **Перестановки. Задачі впорядкування. Перестановочний прийом.**

У більшості випадків побудова оптимальних розкладів пов'язана з розв'язанням деяких комбінаторних задач, сформульованих відносно заданих невпорядкованих або частково впорядкованих множин. Ці задачі, як правило, полягають у відшуканні найкращого "довпорядкування" заданих множин або найкращим розбиттям їх на підмножини.

У цьому розділі у досить стислій формі наведені деякі, необхідні далі, відомості із теорії графів, теорії впорядкування множин, теорії оптимізації функції на множині перестановок.

Множини, відношення, відображення, графи

Поняття множини широко використовується не тільки в математиці, але і в других науках. Інтуїтивно під *множиною* прийнято розуміти сукупність, об'єднання деяких елементів, предметів, об'єктів.

Поняття *відношення* є таким же первинним, як і поняття множини. Кажучи, що елементи або множини знаходяться в деякому відношенні, розуміють встановлення між ними деякого певного зв'язку, залежності.

Основним відношенням, що вивчається в математиці, є відображення. *Відображення* φ множини X в множину Y – це таке відношення, коли кожному елементу $x \in X$ ставиться у відповідність множина елементів $\varphi x \subseteq Y$. У цих позначеннях X – область визначення відображення, Y – область його значень, φx – образ елемента x. Відображення φ^{-1}, обернене відображенню φ, визначається як таке відношення, коли кожному $y \in Y$ ставиться у відповідність множина $\varphi^{-1} y$ тих $x \in X$, для яких $y \in \varphi x$, $\varphi^{-1} y$ – прообраз y.

Якщо φx для кожного x складається з єдиного елементу $y \in Y$, то відображення зазвичай називають оператором. Операцією називається відображення підмножини множини A в елементи множини A. Зручною і наглядною формою представлення множин з визначеними на них відношеннями є графи.

Графом прийнято називати наступну конструкцію. Задано n точок, довільно розміщених на площині. Від деяких точок (не обов'язково від

кожної) до деяких других точок проведені стрілки. Можна вважати, що стрілки визначають деяке відображення точок у другі точки з числа заданих.

У загальному випадку вважається, що задано граф, якщо задані непуста множина X і відображення Γ множини X в X.

Графи будемо позначати через $G=(X,\Gamma)$.

Частинним (виродженим) випадком графа являється нуль-граф: нуль-граф складається із ізольованих елементів, іншими словами, для довільного x в нуль-графі $\Gamma x=\varnothing$. На відміну від цього у повному графі для $\forall x$ множина $\Gamma x=X$ (повним називається також граф, для якого $\Gamma x=X\setminus\{x\}$).

У відповідності з графічним представленням прийнято називати елементи множини X точками або вершинами (вузлами) графа, а пари $u=(x,y)$ такі, що $x\in X$, $y\in\Gamma x$ – дугами графа. Множину всіх дуг у графі будемо позначати через $\vec{U}$.

У дузі $u=(x,y)$ вершини x і y – граничні вершини. При цьому x – початок дуги, y – кінець дуги; кажуть також, що дуга виходить із x і заходить в y. Дуга (x,y) – стрілка – має орієнтацію, вона направлена від x до y. Дуга (x,x) називається петлею.

Часто в задачах у відношенні між образом і прообразом орієнтація нас не цікавить, важливо лише те, що вершини x і y з'єднані дугою. Будемо казати в цьому випадку, що вершини x і y з'єднані ребром $[x,y]$ (графічно це відповідає тому, що точки з'єднуються лініями, а не стрілками). Множину ребер позначимо через U.

Графи з дугами називаються *орієнтованими*, графи з ребрами – *неорієнтованими*. Будемо позначати через $G=(X,\vec{U})$ орієнтовані графи, через $G=(X,U)$ неорієнтовані графи.

Граф $G^1=(X^1,\Gamma^1)$ будемо називати *частинним графом* графа $G^2=(X^2,\Gamma^2)$, якщо $X^1=X^2=X$ і для кожного $x\in X$ множина $\Gamma^1x\subseteq\Gamma^2x$ (множина дуг (або ребер) G^2 містить множину дуг (або ребер) G^1).

Граф $G^1=(X^1,\Gamma^1)$ називається *підграфом* графа $G^2=(X^2,\Gamma^2)$, якщо $X^1\subset X^2$ і $\Gamma^1x=\Gamma^2x\cap X^1$ (люба дуга (ребро) (x,y) із G^2 міститься в графі G^1, якщо $x\in X^1$ і $y\in X^1$).

Поняття шляху, контуру, ланцюга, циклу, зв'язності є поняттями деяких відношень, породжених в X відношенням φ.

Дуги (ребра) u і v суміжні, якщо вони різні та мають одну спільну граничну вершину. Вершини x і y суміжні, якщо вони різні й з'єднані ребром або дугою.

Шляхом в графі $G=(X,\vec{U})$ називається послідовність дуг $u_1,u_2,...,u_m$, коли кінець кожної попередньої дуги співпадає з початком наступної, тобто, коли послідовність дуг має вигляд

$$(x_1,x_2),\ (x_2,x_3),\ ...,\ (x_{n-1},x_n).$$

Тут x_1 – початок шляху, x_n – кінець шляху.

Контур – шлях, у якого $x_1=x_n$; елементарний контур не містить інших співпадаючих вершин, крім x_1 і x_n.

Поняттям шляху і контуру в неорієнтованому графі відповідають поняття ланцюга і циклу.

Граф $G=(X,U)$ називається *зв'язним*, якщо довільні дві його вершини можна з'єднати ланцюгом.

Компонентою зв'язності графу G називається довільний його підграф, який поряд з деякою вершиною x містить і всі вершини, з якими x в G з'єднана ланцюгом.

Локальною степеню вершини графа називається число ребер (дуг), інцидентних цій вершині. Якщо граф орієнтований, то число дуг, що виходять (входять) із деякої вершини, називається півстепеню виходу (входу) цієї вершини.

Деревом називається зв'язний неорієнтований граф, що має не менше двох вершин і не містить циклів. У дереві довільні дві вершини зв'язані єдиним ланцюгом.

Деревами часто називаються і орієнтовані графи, якщо вони задовольняють перерахованим умовам без урахування орієнтації їх дуг.

Зв'язний орієнтований граф без контурів називається *прадеревом з коренем* x (x – вершина графа), якщо у вершину x не входить жодна дуга, а в кожну з інших вершин входить рівно по одній дузі.

Якщо орієнтований граф не містить петель і контурів, то всі його вершини можна розподілити по *рангам* (шарам). При цьому до першого рангу відносять всі ті вершини вихідного графа, півстепінь входу яких дорівнює нулю. Видаляючи ці вершини разом з інцидент ними їм дугами, отримаємо деякий підграф. Якщо цей підграф не пустий, то всі його вершини, півстепінь входу яких дорівнює нулю, відносять до другого рангу, видаляючи їх разом з інцидентними їм дугами. Процедура повторюється до тих пір, поки всі вершини вихідного графу не будуть розподілені по рангам.

Впорядкованість

Наведемо необхідні відомості про властивості впорядкованих множин. Всі розглядувані множини припускаються скінченими. Будемо розглядати множини натуральних чисел.

Нехай $N=\{1,2,...,n\}$. Бінарне відношення $\rightarrow$ на множині N називається *відношенням строгого порядку* (або *строгим порядком*), якщо воно антирефлексивне, тобто ні для якого $i\in N$ не виконується $i\rightarrow i$, й транзитивне, тобто якщо $i\rightarrow j$ та $j\rightarrow k$, то $i\rightarrow k$. Звідси випливає, що це відношення антисиметричне, тобто якщо виконується $i\rightarrow j$, то неможливо $j\rightarrow i$.

Якщо $i\rightarrow j$, то кажуть, що елемент i передує елементу j; якщо крім того, не існує k такого, що $i\rightarrow k$ та $k\rightarrow j$, то елемент i безпосередньо передує елементу j. Елементи i та j називаються *непорівнюваними*, якщо не має місце жодне із співвідношень $i\rightarrow j$ та $j\rightarrow i$.

Відношення строгого порядку $\rightarrow$ називається повністю строгим порядком, якщо для довільних різних i та j має місце $i\rightarrow j$ або $j\rightarrow i$.

Множина N із заданим на ній відношенням строгого порядку $\rightarrow$ називається *впорядкованою множиною*.

Елемент $i\in N$ називається мінімальним (максимальним), якщо не існує $j\in N$ такого, що $j\rightarrow i$ (відповідно $i\rightarrow j$). Інші елементи називаються *проміжковими*.

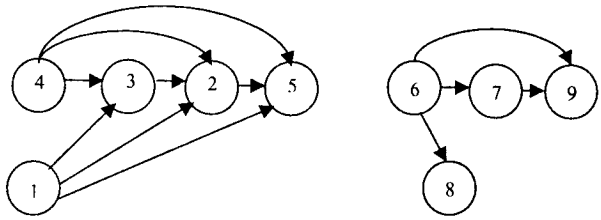
Орієнтований граф $(N,\vec{U})$ називається *графом відношення строгого порядку* $\rightarrow$, заданого на множині N, якщо $(i,j)\in\vec{U}$ в тому і тільки в тому випадку, коли $i\rightarrow j$. Граф, очевидно, не містить петель і контурів та є транзитивно замкнутим. Якщо відношення $\rightarrow$ – повністю строгий порядок, то довільна пара вершин графа з'єднана дугою.

Більш поширеним є представлення впорядкованої множини у вигляді графа $\left(N,\vec{U}_p\right)$ відношення, що називається редукцією відношення строгого порядку. Граф $(N,\vec{U}_p)$ відрізняється від $(N,\vec{U})$ тим, що він не містить дуги (i,j) всякий раз, коли існує шлях із i від j, відмінний від дуги (i,j). Іншими словами, дуга $(i,j)\in\vec{U}_p$ тоді і тільки тоді, коли i безпосередньо передує j.

У ряді випадків впорядкована множина представляється неорієнтованим графом (N,U) при наступній додатковій умові: якщо i

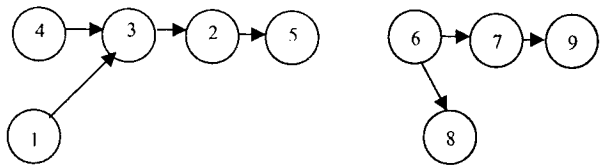
безпосередньо передує j, то вершина j на рисунку зображається вище вершини i й ці вершини з'єднуються ребром.

На рисунку 2.1а) зображено граф $(N,\vec{U})$ строгого порядку $\rightarrow$, заданого на множині $N=\{1,2,3,4,5,6,7,8,9\}$ умовами $4\rightarrow 3$, $7\rightarrow 9$, $3\rightarrow 2$, $1\rightarrow 3$, $2\rightarrow 5$, $6\rightarrow 7$, $6\rightarrow 8$. На рисунку 2.1б) і рисунку 2.1в) зображено графи $(N,\vec{U}_p)$ та (N,U).



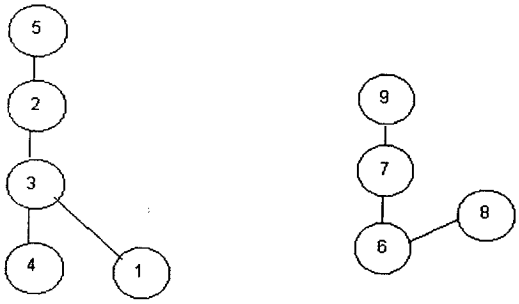
Граф $(N,\vec{U})$ строгого порядку $\rightarrow$, заданий на $N=\{1,2,3,4,5,6,7,8,9\}$

Рис.2.1 а)



Граф $(N,\vec{U}_p)$, $N=\{1,2,3,4,5,6,7,8,9\}$

Рис.2.1 б)



Граф (N,U), $N=\{1,2,3,4,5,6,7,8,9\}$

Рис.2.1 в)

Перестановки. Задачі впорядкування. Перестановочний прийом

Поняття *перестановки* добре відоме – розміщення елементів деякої множини в довільному порядку (без повторень) являє собою перестановку елементів цієї множини.

Перестановка звичайно представляється послідовністю елементів (чисел) із $N=\{1,2,...,n\}$. Символічним записом цієї конструкції є $\pi=(i_1,i_2,...,i_n)$ або $\pi=([1],[2],...,[n])$, де i_k або $[k]$ – елемент із N, що займає в послідовності π k-те зліва місце.

Перестановка може бути представлена таблицею виду $\{(i,j)\}$, яка відображає той факт, що в перестановці елемент $i\in N$ займає j-те місце.

Певне поширення отримало представлення перестановки у вигляді матриці $\|x_{ij}\|$ розмірності $n\times n$, елементами якої є числа 0 або 1. Значення $x_{ij}=1$ при цьому відповідає запису (i,j). У цій матриці сума елементів, що стоять в кожному рядочку і в кожному стовпчику, дорівнює 1.

Під задачею впорядкування розуміють, як правило, задачу побудови одної або декількох перестановок, що задовольняють певним обмеженням і забезпечують екстремум деякої функції чи функцій, визначених на розглядуваній множині перестановок.

Тут ми розглянемо порівняно прості задачі впорядкування, розв'язок яких може бут отримано з використанням так званого *перестановочного прийому*. Суть цього прийому полягає в попередньому дослідженні впливу взаємного розміщення сусідніх елементів перестановки на значення оптимізуючої функції. У ряді випадків це дозволяє різко звузити множину можливих розв'язків.

Останнє характерно для ситуації, в якій оптимізуючій функції $f(\pi)$, заданій на множині всіх перестановок елементів множини N, виявляється можливим зіставити деяку функцію $\omega(i)$, $i\in N$, що володіє наступною властивістю. Якщо $\pi=(i_1,i_2,...,i_k,i_{k+1},...,i_n)$ і $\omega(i_k)\geq\omega(i_{k+1})$, то $f(\pi)\leq f(\pi')$, де π' відрізняється від π тільки транспозицією елементів i_k та i_{k+1}.

Для побудови перестановки π^*, якій відповідає найменше значення $f(\pi)$, у цьому випадку достатньо впорядкувати елементи N у порядку не зростання значень $\omega(i)$, $i\in N$.

Дійсно, в противному випадку здійснюючи транспозицію елементів i_k та i_{k+1}, всякий раз, коли $\omega(i_{k+1})>\omega(i_k)$, приходимо до перестановки π^*, і $f(\pi^*)$ не перевищує значення, що відповідає вихідній перестановці.

1) Нехай дано два n-вимірних вектори $\alpha=(\alpha_1,\alpha_2,...,\alpha_n)$ та $\beta=(\beta_1,\beta_2,...,\beta_n)$, компоненти яких -- дійсні числа. Позначимо через $\pi=(i_1,i_2,...,i_n)$ деяку перестановку елементів множини $N=\{1,2,...,n\}$. Визначимо

$$f_1(\pi)=\sum_{k=1}^{n}\alpha_k\beta_k .$$

Потрібно побудувати перестановку π_1^* елементів множини N, якій відповідає найменше значення $f_1(\pi)$.

Не порушуючи загальності, будемо припускати, що компоненти векторів α і β пронумеровані так, що $\alpha_1\le\alpha_2\le...\le\alpha_n$.

Розглянемо перестановку π', яка відрізняється від π транспозицією елементів i_k та i_{k+1}. Маємо

$$f_1(\pi')-f_1(\pi)=(\alpha_k-\alpha_{k+1})(\beta_{i_{k+1}}-\beta_{i_k}).$$

Якщо $\beta_{i_k}\ge\beta_{i_{k+1}}$, то ця різниця невід'ємна.

Таким чином, впорядковуючи числа β_i у порядку їх не зростання, отримаємо шукану перестановку $\pi_1^*=(i_1^*,i_2^*,...,i_n^*)$. Тут $\beta_{i_k^*}\ge\beta_{i_{k+1}^*}$ для всіх $k=\overline{1,n-1}$.

Теорема. Функція $f_1(\pi)$ досягає найменшого значення тоді і тільки тоді, коли меншим α_k відповідають більші β_{i_k}.

2) Розглянемо задачу побудови послідовності π_2^*, якій відповідає найменше значення функції

$$f_2(\pi)=\max_{1\le u\le n}\left(\sum_{k=1}^{u}\alpha_{i_k}+\beta_{i_u}\right).$$

Перш за все знайдемо достатні умови, при яких $f_2(\pi)\le f_2(\pi')$, де π' відрізняється від π транспозицією елементів, які займають k-ту та $(k+1)$-у позиції, тобто, якщо $\pi=(i_1,i_2,...,i_{k-1},i_k,i_{k+1},i_{k+2},...,i_n)$, то $\pi'=(i_1,i_2,...,i_{k-1},i_{k+1},i_k,i_{k+2},...,i_n)$. Маємо

$$f_2(\pi)=\max\left[\max_{\substack{1\le u\le k-1\\ k+2\le u\le n}}\left(\sum_{j=1}^{u}\alpha_{i_j}+\beta_{i_u}\right),\ \sum_{j=1}^{k}\alpha_{i_j}+\beta_{i_k},\ \sum_{j=1}^{k+1}\alpha_{i_j}+\beta_{i_{k+1}}\right].$$

Аналогічно,

$$f_2(\pi')=\max\left[\max_{\substack{1\le u\le k-1\\ k+2\le u\le n}}\left(\sum_{j=1}^{u}\alpha_{i_j}+\beta_{i_u}\right),\ \sum_{j=1}^{k}\alpha_{i_j}+\alpha_{i_{k+1}}+\beta_{i_{k+1}},\ \sum_{j=1}^{k+1}\alpha_{i_j}+\beta_{i_k}\right].$$

Отже, для того, щоб $f_2(\pi) \le f_2(\pi')$ досить, щоб

$$\min(\alpha_{i_k} + \beta_{i_k}, \beta_{i_{k+1}}) \le \min(\alpha_{i_{k+1}} + \beta_{i_{k+1}}, \beta_{i_k}). \qquad (1)$$

В свою чергу, для того, щоб мало місце останнє співвідношення, досить, щоб

$$sign(\alpha_{i_k})\left[M - \min(\alpha_{i_k} + \beta_{i_k}, \beta_{i_k})\right] \le sign(\alpha_{i_{k+1}})\left[M - \min(\alpha_{i_{k+1}} + \beta_{i_{k+1}}, \beta_{i_{k+1}})\right] \qquad (2)$$

де M – досить велике число (значення $M > \max\limits_{1 \le l \le n} \min(\alpha_l + \beta_l, \beta_l)$).

Дійсно, можливі наступні випадки:

1) $\alpha_{i_k} \le 0$, $\alpha_{i_{k+1}} \ge 0$.

В цьому випадку $\alpha_{i_k} + \beta_{i_k} \le \beta_{i_k}$ і $\beta_{i_{k+1}} \le \alpha_{i_{k+1}} + \beta_{i_{k+1}}$;

2) $\alpha_{i_k} < 0$, $\alpha_{i_{k+1}} < 0$.

За умовою

$$\min(\alpha_{i_k} + \beta_{i_k}, \beta_{i_k}) \le \min(\alpha_{i_{k+1}} + \beta_{i_{k+1}}, \beta_{i_{k+1}}),$$

звідки

$$\alpha_{i_{k+1}} + \beta_{i_{k+1}} \le \min\left(\beta_{i_{k+1}}, \beta_{i_{k+1}}, \alpha_{i_{k+1}} + \beta_{i_{k+1}}\right);$$

3) $\alpha_{i_k} > 0$, $\alpha_{i_{k+1}} > 0$.

За умовою

$$\min(\alpha_{i_k} + \beta_{i_k}, \beta_{i_k}) \ge \min(\alpha_{i_{k+1}} + \beta_{i_{k+1}}, \beta_{i_{k+1}}),$$

звідки

$$\beta_{i_{k+1}} \le \min\left(\alpha_{i_k} + \beta_{i_k}, \beta_{i_k}, \alpha_{i_{k+1}} + \beta_{i_{k+1}}\right).$$

Отже, в любому випадку має місце співвідношення (1). Таким чином, справедлива наступна теорема.

Теорема. Для того, щоб перестановці $\pi = (i_1, i_2, ..., i_n)$ відповідало найменше значення функції $f_2(\pi)$, досить, щоб співвідношення (2) виконувалось для всіх $k = \overline{1, n-1}$.

Для знаходження оптимальної перестановки π_2^* чисел множини N кожному із $i \in N$ слід приписати „вагу" (пріоритет)

$$\omega(i) = sign(\alpha_i)\left[M - \min(\alpha_i + \beta_i, \beta_i)\right]$$

і впорядкувати ці числа в порядку не спадання „ваги".

Клас оптимальних перестановок, взагалі кажучи, можна розширити, якщо безпосередньо скористатися співвідношенням (1).

Детерміновані задачі впорядкування

3.1. Системи обслуговування з одним приладом

У даному розділі припускається, що кожна робота складається тільки з одної операції. В цьому випадку множину робіт можна розбити на групи в залежності від виду операції й кожна машина, виконуюча певну операцію, не залежить від других. Отже, можна обмежитися складанням розкладу тільки для одної машини та виконуючою нею підмножиною робіт, які виконуються цією машиною.

Будемо вважати, що число робіт n скінчене і відоме наперед, і що всі вони повинні бути виконані. Крім того, вважаємо, що машини використовуються тільки для виконання розглядуваних робіт, що вони завжди доступні й не виходять з ладу. Далі приймемо, що всі n робіт поступають в систему одночасно, так що при складанні розкладу процес обслуговування може початися з довільної з них, і що виконання кожної з робіт відбувається або без настройки машини, або настройка не залежить від попередньої роботи. В останньому випадку тривалість настройки перед якою-небудь роботою залежить тільки від самої роботи, і цю тривалість можна приєднати до тривалості роботи, що виконується.

Будемо покладати, що:

$m = 1$, тобто є тільки одна машина;

$g_i = 1$, тобто кожна робота складається з одної операції;

$r_i = 0$, тобто всі роботи поступають одночасно;

$p_{i1} = p_i$, тобто тривалість роботи є довільною величиною, що задається наперед і не залежить від розкладу;

$a_i = d_i$, тобто допустима тривалість проходження співпадає з плановим строком, так як $d_i = r_i + a_i = 0 + a_i$;

$W_{i1} = W_i$, тобто тривалість очікування роботою i початку її виконання не залежить від другого індексу, так як робота складається з однієї операції;

$C_i = F_i = p_i + W_i$, тобто момент закінчення роботи рівний тривалості проходження роботи.

Перестановочні розклади

У складних випадках можуть бути корисними розклади, які допускають *переривання* (коли виконання роботи переривається до її завершення і робота видаляється з машини) та *штучно вводимі простої* (коли машина простоює при наявності очікування виконання роботи).

Теорема. Для системи $\mathbf{n|1}$ (n робіт на 1 машині) розклад, оптимальний відносно регулярного критерію, належність класу, який виключає переривання або штучні простої.

Доведення.

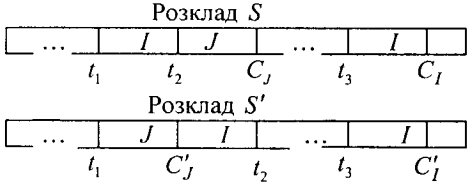
1) *Випадок штучних простоїв.*

Розглянемо розклад S, який містить простої на відрізку від t_1 до t_2, де $0 \le t_1 < t_2 \le C_{\max}$.

Існує розклад S', який відрізняється від розкладу S тільки тим, що машина не простоює на відрізку $[t_1, t_2]$ і всі події, що відбуваються після t_2, випереджають відповідні події S на $t_2 - t_1$. При цьому, очевидно, не змінюються моменти закінчення робіт, що відбуваються до t_1, а для робіт, що закінчуються після t_2, моменти закінчення для S зсуваються вліво на $t_2 - t_1$. Зрозуміло, що $C_i' \le C_i$, $i = \overline{1,n}$, так що значення регулярного критерію для S' не перевищує його значення для S. Тому оптимальний розклад належить S', тобто класу без простоїв.

2) *Випадок переривань.*

Розглянемо розклад S, при якому робота I починає виконуватися в момент t_1 і в момент t_2, що передує її закінченню, переривається роботою J. (Нехай J не переривається). В момент часу t_3 робота I знову відновлюється і продовжується до її повного завершення.



Тепер розглянемо розклад S', який відрізняється від S тим, що першою виконується робота J. В результаті отримуємо, що в S' закінчиться раніше тільки робота J, а моменти закінчення інших робіт не зміняться. Тому значення любого регулярного критерію для S' не перевищує його значення для S. Якщо в розкладі S' виконання роботи I переривається другими роботами, то можна повторити описану процедуру перестановки перериваючих робіт перед I до тих пір, поки робота I не буде виконуватись без переривань. В результаті всіх цих перестановок регулярний критерій не збільшиться. Із сказаного вище випливає, що оптимальний розклад належить класу S', що не містить переривань.

Теорему доведено.

Основним результатом доведеної теореми є те, що пошук оптимального розкладу повинен проводитися в класі *перестановочних* розкладів.

Найбільш поширений спосіб оцінки якості розкладу для детермінованої системи обслуговування полягає в наступному. Кожному розкладу S відповідає вектор $\bar{t}(S)=(\overline{t_1}(S),...,\overline{t_n}(S))$ моментів закінчення обслуговування вимог при цьому розкладі. Задається дійсна неспадна функція n змінних $F(x)=F(x_1,...,x_n)$. Якість розкладу S характеризується значенням цієї функції при $x=\bar{t}(S)$. З двох розкладів кращим вважається той, якому відповідає менше значення $F(x)$. Розклад якому відповідає найменше значення $F(x)$, називається *оптимальним*.

При заданні функції $F(x)$ кожній роботі i ставлять у відповідність деяку неспадну функцію $\varphi_i(t)$, яка в кількісному відношенні виражає "штраф", який потрібно "заплатити", якщо обслуговування роботи i закінчиться в момент часу t. Якість розкладу S характеризується сумарним або максимальним штрафом, який необхідно заплатити при обслуговуванні роботи за цим розкладом, тобто

$$F_{\sum}(S)=\sum_{i=1}^{n}\varphi_i(\overline{t}_i(S)) \text{ або } F_{\max}(S)=\max_{1\le i\le n}\{\varphi_i(\overline{t}_i(S))\}.$$

Якщо всі вимоги поступають на обслуговування одночасно ($r_i=0$, $i=\overline{1,n}$), то при пошуку оптимального розкладу можна обмежитися

розглядом класу S розкладів без переривань і простоїв обслуговуючого приладу. В цьому випадку кожний розклад $s \in S$ однозначно визначається послідовністю обслуговування вимог – перестановкою $\pi = (i_1, i_2, ..., i_n)$ елементів множини N. При такому розкладі робота i_k починає виконуватися в момент часу $t^0_{i_k} = \sum_{j=1}^{k-1} t_{ij}$ і завершується в момент часу $\bar{t}_{i_k} = t^0_{i_k} + t_{i_k}$, $k = \overline{1,n}$, $t^0_{i_1} = 0$. Таким чином, якщо $r_i = 0$, $i = \overline{1,n}$, то задача пошуку оптимального розкладу зводиться до пошуку найкращої перестановки серед $n!$ перестановок множини N.

Постановка задачі.

Всі вимоги поступають в чергу одночасно ($r_i = 0$, $i = \overline{1,n}$), кожній вимозі i поставлена у відповідність неспадна функція штрафу $\varphi_i(t)$. Необхідно побудувати розклад S^*, якому відповідає найменше значення $F_{\max}(S)$, тобто необхідно знайти таку перестановку π^*, якій відповідає найменше значення $F_{\max}(\pi) = \max\limits_{1 \le k \le n} \left\{ \varphi_{i_k} \left(\sum\limits_{j=1}^{k} t_{i_j} \right) \right\}$.

Алгоритм побудови оптимальної перестановки.

1. Покладемо $T = \sum_{i=1}^{n} t_i$.
2. Обчислимо значення $\varphi_i(T)$, $i = \overline{1,n}$ і виберемо найменше з них (або одне з найменших, якщо їх декілька). Нехай цим значенням буде $\varphi_{i'}(T)$. Тоді $i^*_n = i'$.
3. Покладемо $T_1 = T - t_{i'}$.
4. Обчислимо значення $\varphi_i(T_1)$, $i \in N_1 = N \setminus i^*_n$ та виберемо найменше з них – $\varphi_{i^*_{n-1}}$.
5. Продовжуючи цей процес отримаємо перестановку $\pi^* = (i_1^*, i_2^*, ..., i_n^*)$.

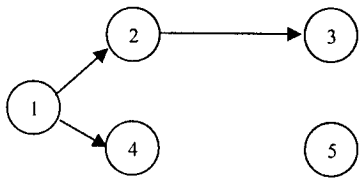
Цей алгоритм може бути використаний і у випадку, коли на множині N задано відношення строгого порядку. В цьому випадку вибір вимог i^*_n здійснюється серед тих вимог множини N, які можуть обслуговуватися останніми. Вибір вимог i^*_{n-1} здійснюється серед тих вимог множини N_1, які можуть обслуговуватися останніми в цій множині і т.д.

Приклад

На ЕОМ необхідно розв'язати 6 задач, тривалість розв'язання і функції штрафу яких наведено в таблиці.

I	1	2	3	4	5
t_i	2	1	1	2	2
$\varphi_i(t)$	$2t$	t^3-1	t^2	$t-3$	2^t

Задачі взаємопов'язані. При їх розв'язанні потрібно дотримуватись порядку, який задається такою схемою:



Розв'язання.

Маємо $T=8$. Оскільки останніми можуть розв'язуватися задачі 3, 4, 5, то обчислюємо $\varphi_3(8)=64$; $\varphi_4(8)=5$ і $\varphi_5(8)=256$. В якості i_5^* вибираємо задачу 4.

Множина $N_1=\{1,2,3,5\}$, значення $T_1=6$. У множині N_1 останніми можуть розв'язуватися задачі 3 і 5. Маємо $\varphi_3(6)=36$; $\varphi_5(6)=64$. Отже, $i_4^*=3$. Множина $N_2=\{1,2,5\}$; значення $T_2=5$. У множині N_2 останніми можуть розв'язуватися задачі 2 і 5. Маємо $\varphi_2(5)=124$, $\varphi_5(5)=32$. Отже, $i_3^*=5$.

Множина $N_3=\{1,2\}$ є лінійно впорядкованою. Отже, $i_2^*=2$; $i_1^*=1$.

Перестановка $\pi^*=(1,2,5,3,4)$ є шуканою оптимальною перестановкою. Значення $\bar{t}_1=2$, $\bar{t}_2=3$, $\bar{t}_3=6$, $\bar{t}_4=8$; $\varphi_1(2)=4$, $\varphi_2(3)=26$, $\varphi_3(6)=36$, $\varphi_4(8)=5$; значення $F_{\max}(\pi^*)=36$.

Розглянемо задачу мінімізації сумарного штрафу $F_\Sigma(S)$ при обслуговуванні вимог одним приладом в припущенні, що $r_i=0$, $\varphi_i(t)=a_it+b_i$, $a_i\geq 0$, $i=\overline{1,n}$.

В цьому випадку при пошуку оптимального розкладу також можна обмежитися розглядом класу розкладів без переривань. Якщо вимоги обслуговуються в послідовності $\pi=(i_1,i_2,\ldots,i_n)$, то сумарний штраф

$$F_\Sigma(\pi)=\sum_{k=1}^{n}\varphi_{i_k}\left(\sum_{j=1}^{k}t_{i_j}\right).$$

Нехай $\pi' = (1, 2, ..., n)$ і π'' відрізняється від π' тільки транспозицією елементів l і $l+1$, тобто $\pi'' = (1, \ldots, l-1, l+1, l, l+2, \ldots, n)$. Легко бачити, що $F_\Sigma(\pi') - F_\Sigma(\pi'') = a_{l+1}t_l - a_l t_{l+1}$.

Для того, щоб $F_\Sigma(\pi') \le F_\Sigma(\pi'')$ необхідно й досить, щоб $\frac{a_{l+1}}{t_{l+1}} \le \frac{a_l}{t_l}$. Це співвідношення володіє властивістю транзитивності. Якщо воно виконується для всіх $l = \overline{1, n}$, то π' – оптимальна перестановка.

Таким чином, для побудови оптимальної перестановки кожній операції i достатньо приписати вагу $\omega(i) = \frac{a_i}{t_i}$ і впорядкувати вимоги у порядку незростання ваг. Для цього необхідно виконати не більше $O(n \log_2 n)$ операцій.

Інтервали черговості

Розглянемо прості методи розв'язання задачі оптимального впорядкування вимог для деяких частинних випадків функції штрафу. Припустимо, що множина N вимог не впорядкована, всі вимоги поступають в чергу на обслуговування одночасно, в момент часу $r_i = 0$, критерій оптимальності розкладу – сумарний штраф.

Нехай вимоги i та j обслуговуються безпосередньо одна за другою і їх обслуговування починається в момент часу $t \ge 0$. Якщо при цьому першою обслуговується вимога i, то сумарний штраф за обслуговування цих вимог

$$F_\Sigma(t, i, j) = \varphi_i(t + t_i) + \varphi_j(t + t_i + t_j).$$

Якщо першою обслуговується вимога j, то сумарний штраф за обслуговування цих двох вимог

$$F_\Sigma(t, j, i) = \varphi_j(t + t_j) + \varphi_i(t + t_i + t_j).$$

Визначимо $R_{ij}(t) = F_\Sigma(t, i, j) - F_\Sigma(t, j, i)$.

Величина $R_{ij}(t)$ показує, наскільки зміниться сумарний штраф при переході від послідовності $i \prec j$ до послідовності $j \prec i$.

Якщо $R_{ij}(t) < 0$, то перевага надається послідовності $i \prec j$. Аналогічно, якщо $R_{ij}(t) > 0$, то першою в момент часу t слід обслужити вимогу j. Якщо ж $R_{ij}(t) = 0$, то порядок обслуговування цих вимог є довільним.

Оскільки функції $\varphi_i(x)$ та $\varphi_j(x)$ припускаються кусково-перервними на інтервалі $\left(0; T = \sum_{k=1}^{n} t_k\right)$, то цей інтервал може бути розбитий на скінчене число інтервалів, у кожному з яких величина $R_{ij}(t)$ додатна, від'ємна чи рівна нулю.

Інтервал $(\theta_1;\theta_2)$ називається *інтервалом черговості* типу $i \prec j$, якщо $R_{ij}(t) < 0$ для всіх $t \in (\theta_1;\theta_2)$. Якщо $R_{ij}(t) > 0$ для всіх $t \in (\theta_1;\theta_2)$, то інтервал $(\theta_1;\theta_2)$ називається інтервалом черговості типу $j \prec i$. При $R_{ij}(t) = 0$ для всіх $t \in (\theta_1;\theta_2)$ порядок обслуговування вимог довільний.

Величина $R_{ij}(t)$ за визначенням залежить від величин t, t_i, t_j та функцій штрафу $\varphi_i(x)$ і $\varphi_j(x)$. У кожному конкретному випадку в результаті простих перетворень можуть бути виділені інтервали черговості.

Нехай, наприклад, $\varphi_i(x) = a_i x^2$, $\varphi_j(x) = a_j x^2$. Знайдемо інтервали черговості при різних сполученнях a_i, a_j, t_i, t_j.

Маємо

$$R_{ij}(t) = a_i(t+t_i)^2 + a_j(t+t_i+t_j)^2 - a_j(t-t_j)^2 - a_i(t+t_i+t_j)^2 =$$
$$= 2(a_j t_i - a_i t_j)t + \left[2t_i t_j(a_j - a_i) + a_j t_i^2 - a_i t_j^2\right].$$

Нехай $\frac{a_i}{t_i} = \frac{a_j}{t_j}$. У цьому випадку $R_{ij}(t)$ не залежить від t і визначається величиною

$$Q(a_i, a_j, t_i, t_j) = 2t_i t_j(a_j - a_i) + a_j t_i^2 - a_i t_j^2.$$

Якщо $Q(a_i, a_j, t_i, t_j) < 0$, то при послідовному обслуговуванні вимог i та j вимога i завжди обслуговується першою. В цьому випадку інтервал $(-\infty;\infty)$ (а, отже, й інтервал $(0;T)$) є інтервалом черговості типу $i \prec j$. Аналогічно, якщо $Q(a_i, a_j, t_i, t_j) > 0$, то $(-\infty;\infty)$ – інтервал черговості типу $j \prec i$. Нарешті, якщо $Q(a_i, a_j, t_i, t_j) = 0$, то порядок обслуговування вимог довільний, якщо ці вимоги обслуговуються безпосередньо одна за другою.

Нехай $\frac{a_i}{t_i} \neq \frac{a_j}{t_j}$. Величина $R_{ij}(t) = 0$ при $t_0 = \frac{2t_i t_j(a_j - a_i) + a_j t_i^2 - a_i t_j^2}{2(a_i t_j - a_j t_i)}$. Інтервал $(-\infty;+\infty)$ розбивається на три інтервали черговості $(-\infty;t_0)$, (t_0), $(t_0;+\infty)$. Тип інтервалів $(-\infty;t_0)$ і $(t_0;+\infty)$ визначається знаком $R_{ij}(t)$ на цих інтервалах. При $t = t_0$ порядок обслуговування вимог i та j довільний.

За припущенням обслуговування всіх вимог здійснюється на часовому інтервалі $(0;T)$ або, у всякому випадку, на інтервалі $(0;+\infty)$. Отже, в

практичному відношенні інтерес представляє виділення інтервалів черговості в інтервалі планування $(0;T)$ або $(0;+\infty)$. Якщо $t_0 < 0$ або $t_0 > T - t_i - t_j$, то $(0;T)$, очевидно, є інтервалом черговості, тип якого визначається знаком $R_{ij}(t)$. Якщо $0 < t_0 < T - t_i - t_j$, то інтервал $(0;T)$ розбивається на три інтервали черговості. Якщо $t_0 = 0$ або $t_0 = T - t_i - t_j$, то маємо два інтервали черговості, один із яких (t_0).

Це твердження, взагалі кажучи, несправедливе, якщо після завершення процесу обслуговування одної з вимог i чи j до початку обслуговування другої проходить деякий проміжок часу, зайнятий, наприклад, обслуговуванням деякої третьої вимоги.

Виділення інтервалів черговості може бути проведено для довільних пар функцій штрафу $\varphi_i(x)$, $\varphi_j(x)$ і довільних t_i, t_j. Для цього достатньо розв'язати (відносно t) відповідне рівняння $R_{ij}(t) = 0$.

Знання інтервалів черговості є досить корисним при пошуку оптимальних розкладів методом послідовного конструювання варіантів.

Знак величини $R_{ij}(t)$, а отже, і тип інтервалу черговості визначається в загальному випадку значеннями t, t_i, t_j й конкретним видом функції штрафу $\varphi_i(x)$ та $\varphi_j(x)$. При деяких сполученнях $t_i, t_j, \varphi_i(x), \varphi_j(x)$ знак $R_{ij}(t)$ може не залежати від t, як це було показано вище.

Покажемо такі пари функцій $\varphi_i(x)$ та $\varphi_j(x)$, для яких знак $R_{ij}(t)$ не залежить від t при будь-яких комбінаціях t_i та t_j:

1) $\varphi_i(x) = a_i x + b_i$, $\varphi_j(x) = a_j x + b_j$;
$R_{ij}(t) = a_j t_i - a_i t_j$;

2) $\varphi_i(x) = a_i e^{\alpha x} + b_i$, $\varphi_j(x) = a_j e^{\alpha x} + b_j$;
$R_{ij}(t) = e^{\alpha t}\left[a_i e^{\alpha t_i} - a_j e^{\alpha t_j} + (a_j - a_i) e^{\alpha(t_i + t_j)}\right]$.

Тут $R_{ij}(t)$ залежить від t, однак знак цієї величини визначається тільки значеннями α, a_i, a_j, t_i, t_j, а при фіксованих α, a_i, a_j тільки значеннями t_i, t_j.

3) $\varphi_i(x) = f(x) + b_i$, $\varphi_j(x) = f(x) + b_j$,

де $f(x)$ – зростаюча на $(-\infty;+\infty)$ функція.

Оскільки $R_{ij}(t) = f(t + t_i) - f(t + t_j)$, то знак $R_{ij}(t)$ не залежить від t при будь-яких конкретних значеннях t_i та t_j.

Таким чином, вказані три пари функцій володіють властивістю незалежності знаку величини $R_{ij}(t)$ від значення $t \in (-\infty;+\infty)$. Інтервал $(-\infty;+\infty)$, а отже і $(0;+\infty)$, і $(0,T)$ є інтервалами черговості для цих пар

функцій. Тип цих інтервалів визначається конкретними значеннями параметрів функцій і значеннями t_i та t_j.

Теорема. Нехай $\varphi_i(x)$, $\varphi_j(x)$ – строго зростаючі достатньо гладкі (існує третя похідна) на $(-\infty;+\infty)$ функції. Для того, щоб знак величини $R_{ij}(t)$ не залежав від t, $t \in (-\infty;+\infty)$ при довільних t_i, t_j, необхідно і досить, щоб:

1) $\varphi_k(x) = a_k x + b_k$, $k = i, j$, або
2) $\varphi_k(x) = a_k e^{\alpha x} + b_k$, $k = i, j$, або
3) $\varphi_k(x) = f(x) + b_k$, $k = i, j$.

Можна показати, що при достатньо загальних припущеннях відносно функції штрафу $\varphi_i(x)$, $\varphi_j(x)$ відповідним вибором значень t_i, t_j будь-який заданий інтервал можна перетворити в інтервал черговості того чи іншого типу.

Особливий інтерес представляє виявлення умов, при яких заданий інтервал буде інтервалом черговості незалежно від конкретних значень t_i, t_j, але із врахуванням порівняльних співвідношень між цими значеннями.

Теорема. Якщо функції штрафу $\varphi_i(x)$, $\varphi_j(x)$ скільки завгодно раз диференційовані, а $\varphi_i^{(n)}(x) \geq \varphi_j^{(n)}(x)$ для $\forall n = 1, 2, \ldots$ на часовому інтервалі $(\theta_1 - \varepsilon; \theta_2 + \varepsilon)$ де $\varepsilon > 0$, то при $t_i < t_j$ інтервал $(\theta_1; \theta_2)$ є інтервалом черговості типу $i \prec j$.

Знання інтервалів черговості в значній мірі прискорює процес пошуку оптимальної послідовності обслуговування вимог і в ряді випадків приводить до дуже простих способів її побудови.

Теорема. Нехай $r_k = 0$, $k = \overline{1,n}$, всі неспадні функції штрафу $\varphi_k(x)$ належать одному і тільки одному із наступних трьох класів:

a) $\varphi_k(x) = a_k x + b_k$,
b) $\varphi_k(x) = a_k e^{\alpha x} + b_k$,
c) $\varphi_k(x) = f(x) + b_k$,

і значення $\omega(k)$ дорівнюють відповідно

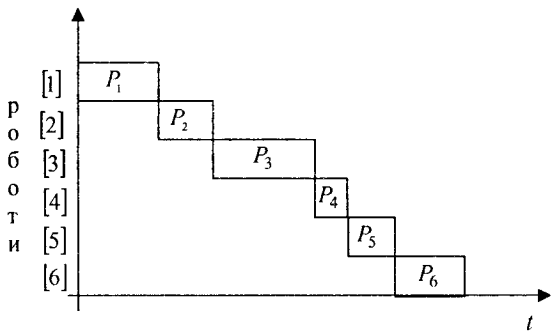
a) $\omega(k) = a_k / t_k$,
b) $\omega(k) = (1 - e^{\alpha t_k}) / (a_k e^{\alpha t_k})$,

c) $\omega(k) = -t_k$.

Для того, щоб послідовності $\pi = (i_1,...,i_n)$ відповідав найменший сумарний штраф, достатньо, щоб $\omega(i_\upsilon) \geq \omega(i_\mu)$ для всіх $1 \leq \upsilon < \mu \leq n$.

Впорядкування за мінімумом тривалості робіт

Впорядкування для системи $\mathbf{n|1}$ схематично можна представити так:



Загальна площа фігури, обмеженої осями координат і зовнішньою ступінчатою лінією, дорівнює сумі тривалостей проходження робіт. Відмітимо, що площа всіх прямокутників, що містяться між зовнішньою і внутрішньою ступінчастими лініями, не залежить від впорядкування. Разом з тим, площа між осями координат і ступінчатою ламаною залежить від перестановки прямокутників і, отже, від розкладу.

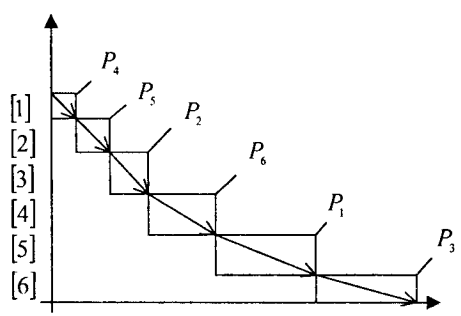


Рис. 3.1.

Як видно з рисунка 3.1, кожний прямокутник можна замінити вектором з кутовим коефіцієнтом $-\frac{1}{p_i}$ і рівним по довжині діагоналі прямокутника. Тоді довільний розклад представляється у вигляді послідовності векторів, розміщених так, що кінець одного співпадає з початком другого. Інтуїтивно здається, що площа під такою векторною кривою буде мінімальною, якщо крива випукла вниз. На рисунку 3.1 показана така крива. Вона починається з вектора, який має максимальний модуль кутового коефіцієнта. Всі наступні вектори розміщені в порядку спадання модуля їх кутового коефіцієнта. Представлений на рисунку 3.1 графік відповідає такому порядку виконання робіт, що

$$p_{[1]} \le p_{[2]} \le \ldots \le p_{[n]}.$$

($p_{[i]}$ – означає тривалість роботи, що виконується в i чергу).

В подальшому такий алгоритм впорядкування будемо називати *впорядкуванням за мінімумом тривалості робіт* (Shortest – processing – time sequensing) або скорочено SPT.

Оптимальність впорядкування SPT встановлюється наступною теоремою.

Теорема. Середній час перебування робіт в системі $\mathbf{n|1|\overline{F}}$ мінімальний, якщо після впорядкування тривалості робіт не спадають:

$$p_{[1]} \le p_{[2]} \le \ldots \le p_{[n]},$$

і максимальний, якщо після впорядкування тривалості робіт не зростають:

$$p_{[1]} \ge p_{[2]} \ge \ldots \ge p_{[n]}.$$

Доведення.

Оскільки розглядаються тільки перестановочні розклади, то тривалість проходження для роботи, що виконується на k-ій позиції деякого розкладу, дорівнює

$$F_{[k]} = \sum_{i=1}^{k} p_{[i]}.$$

Середня тривалість проходження для n робіт визначається як

$$\overline{F} = \frac{\sum_{k=1}^{n} F_{[k]}}{n} = \frac{\sum_{k=1}^{n}\sum_{i=1}^{k} p_{[i]}}{n} = \frac{1}{n}\left(p_{[1]} + p_{[1]} + p_{[2]} + p_{[1]} + p_{[2]} + p_{[3]} + \ldots + p_{[1]} + \ldots p_{[n]}\right)$$

$$= \frac{\sum_{i=1}^{n}(n-i+1)p_{[i]}}{n}.$$

Відомо, що сума попарних добутків членів двох числових послідовностей має мінімальне значення, якщо одна із цих послідовностей зростає, а друга спадає. Так як коефіцієнти $(n-i+1)$ зменшуються із збільшенням i, то мінімум середньої тривалості проходження досягається тільки в тому випадку, коли $p_{[i]}$ з ростом i зростають або, по крайній мірі, не спадають. Сума попарних добутків досягає максимального значення, якщо обидві послідовності впорядковані однаково, тобто роботи виконуються в порядку зменшення або, по крайній мірі, незростання їх тривалості.

Теорему доведено.

Таким чином:

1. Якщо розклад не належить класу SPT-розкладів, то існує k таке, що $p_{[k]} > p_{[k+1]}$.
2. Розклад можна покращити перестановкою порядку виконання робіт k і $k+1$.
3. Довільний із $n!$ можливих розкладів для n робіт зводиться до розкладу SPT послідовними перестановками так, що розклад кожного наступного кроку буде кращим розкладу попереднього.

Дослідження розкладу методом попарних перестановок сусідніх за черговістю робіт досить зручне, але не завжди приводить до цілі. Цей метод непридатний і в ряді випадків для системи $\mathbf{n|1}$.

Наприклад, розглянемо систему $\mathbf{3|1|\overline{T}}$ з наступними вихідними даними:

Роботи	Тривалість	Плановий строк
a	1	4
b	2	2
c	3	3

Критерієм оцінки розкладу служить

$$\overline{T} = \frac{1}{3}[\max(C_a - d_a, 0) + \max(C_b - d_b, 0) + \max(C_c - d_c, 0)],$$

причому критерій $\overline{T}$ являє собою регулярний критерій. Всі 6 можливих послідовностей виконання робіт цієї задачі зображені на рисунку 3.2.

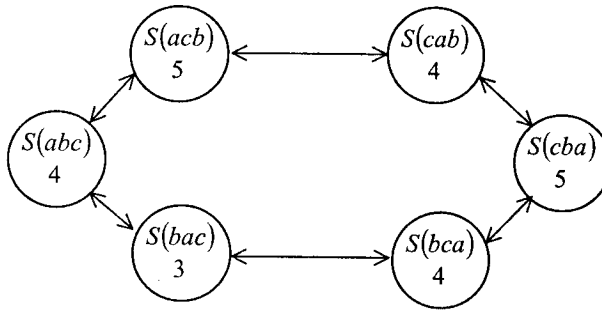


Рис. 3.2.

Стрілки на цьому рисунку з'єднують послідовності, які відрізняються одна від одної попарною перестановкою сусідніх по черзі робіт. Кожне впорядкування має по два таких зв'язки. Розклад оцінюється сумарною затримкою всіх робіт.

Якщо прийняти за вихідний розклад $S(cab)$, то не існує попарної перестановки роботи, що покращує розклад, і можливості методу попарних перестановок виявляються вичерпаними раніше, ніж досягається оптимум $S(bac)$. Тут справа в тому, що використаний критерій не транзитивний. Дійсно, з порівняння $S(acb)$ і $S(cab)$ слідує, що c повинно передувати a. Із порівняння $S(cab)$ і $S(cba)$ слідує, що a повинно передувати b. Однак звідси ще не випливає, що c повинно передувати b. Взагалі, взаємне розміщення двох робіт залежить від порядку виконання третьої роботи. Наприклад, якщо b виконується третьою, то c повинно передувати a (порівняємо $S(acb)$ і $S(cab)$). Якщо b виконується першою або другою, то a повинно передувати c (порівняємо $S(bac)$ і $S(bca)$, а також $S(abc)$ і $S(cba)$).

За допомогою аналогічного доведення можна показати, що в розглядуваному випадку оптимальним буде розклад SPT, якщо за критерій оптимальності прийняти мінімум одного із середніх: часу закінчення роботи, тривалості очікування або часового зміщення. Очевидно, що SPT-розклад мінімізує і ряд других характеристик: мінімальну тривалість проходження, мінімальний час закінчення робіт, максимальну тривалість очікування і мінімальну ненульову тривалість очікування.

Якщо *впорядкувати за максимумом тривалості роботи* (longest – processing – time sequencing або скорочено LPT), тобто виконувати роботи в порядку зменшення їх тривалості, то величини, що досягають мінімуму при SPT, будуть максимальними при LPT.

Впорядкування у відповідності з директивними строками

Із можливих критеріїв оцінки розкладів найбільш важливим, звичайно, є критерій своєчасного виконання робіт.

Оскільки оцінкою розкладу служить порушення планового строку, то зрозуміло, що впорядкування повинно здійснюватись із врахуванням інформації, що міститься в наборі величин $\{d_i\}$. Найбільш простим і очевидним використанням планових строків являється таке впорядкування, що

$$d_{[1]} \le d_{[2]} \le \ldots \le d_{[n]}.$$

Однак не зовсім зрозуміло, що досягається таким впорядкуванням. Відповідь на це запитання дає наступна теорема.

Теорема. Розклад, мінімізуючий максимум часового зміщення і максимум запізнення робіт в системі $\mathbf{n \mid 1 \mid L_{max}}$, такий, що роботи виконуються в порядку неспадання планових строків.

Доведення.

Розглянемо розклад S, що не належить класу розкладів з неспадаючими плановими строками. Тоді в S існує таке k, що $d_{[k]} > d_{[k+1]}$. Нехай K і K' – номери робіт відповідно в k-ій та $(k+1)$-ій позиціях розкладу S.

Тепер розглянемо розклад S', який відрізняється від S тільки перестановкою черговості робіт K і K', тобто в S' робота K' виконується k-ою, а робота K виконується $(k+1)$-ою. В обох розкладах порядок виконання $(k-1)$ перших і $(n-k-1)$ останніх робіт той самий. Крім того, очевидно, що в обох розкладах початок і закінчення виконання цих $(n-2)$ робіт повністю однакові. Тому часове зміщення для них однакове. Нехай максимум цього зміщення буде L. Обидва розклади відрізняються лише часовим зміщенням робіт K і K'. Нам потрібно показати, що $\max(L, L_K, L_{K'})$ в S' менший або рівний максимуму тих же величин в S. Якщо максимум $(L, L_K, L_{K'})$ дорівнює L, то обидва розклади будуть еквівалентними. Тому припустимо, що максимум дорівнює L_K або $L_{K'}$, і доведемо, що теорема вірна в цьому випадку. Якщо $t = \sum_{i=1}^{k-1} p_{[i]}$ (ця величина однакова і для S, і для S'), то відповідні часові зміщення мають наступний вигляд:

для S:

$$L_K(S) = L_{[k]} = t + p_K - d_K,$$

$$L_{K'}(S) = L_{[k+1]} = t + p_K + p_{K'} - d_{K'};$$

для S':

$$L_K(S') = L_{[k+1]} = t + p_{K'} + p_K - d_K,$$
$$L_{K'}(S') = L_{[k]} = t + p_{K'} - d_{K'}.$$

Звідси слідує, що

$$L_{K'}(S) > L_K(S'), \text{ так як } d_K > d_{K'},$$
$$L_{K'}(S) > L_{K'}(S'), \text{ так як } p_K > 0,$$
$$L_{K'}(S) > \max\{L_K(S'), L_{K'}(S')\},$$
$$\max\{L, L_K(S), L_{K'}(S)\} \geq \max\{L, L_K(S'), L_{K'}(S')\}.$$

Таким чином, для довільного розкладу S існує можливість його покращення шляхом попарної перестановки сусідніх робіт так, щоб роботи виконувались у порядку, зворотному їх плановим строкам.

Теорему доведено.

Зменшення максимального запізнення для S' в порівнянні з S випливає із результатів, отриманих для часового зміщення:

$$T_{\max}(S) = \max\{0, L_{\max}(S)\} \geq \max\{0, L_{\max}(S')\} = T_{\max}(S').$$

Впорядкування у відповідності з резервом часу

Інформацію, що міститься в наборі величин $\{d_i\}$, можна використовувати інакше, а саме проводити впорядкування у відповідності з резервом часу кожної роботи. *Резерв часу* роботи i в момент t дорівнює $d_i - p_i - t$ і являє собою максимально допустиму тривалість очікування, при якій не буде затримки. Робота з мінімальним резервом часу має, очевидно, більше шансів бути затриманою, тому при встановленні послідовності вона повинна мати переваги. Оскільки час t є спільним для всіх робіт, то впорядкування повинно бути наступним:

$$d_{[1]} - p_{[1]} \leq d_{[2]} - p_{[2]} \leq \ldots \leq d_{[n]} - p_{[n]}.$$

Інтуїтивно здається, що впорядкування такого виду є деяким покращенням розкладу, отриманого у відповідності з плановими строками. Однак має місце дещо несподіваний результат, який міститься в наступній теоремі.

Теорема. Розклад, максимізуючий мінімальне часове зміщення і мінімальне запізнення робіт в системі $\mathbf{n|1}$, такий, що роботи виконуються в порядку неспадання резерву часу.

Доведення.

Доведення проводимо так само, як і для попередньої теореми. Використовуючи ті ж позначення, отримаємо:

$$L_K(S) < L_{K'}(S'), \text{ так як } (d_K - p_K) > (d_{K'} - p_{K'}),$$

$$L_K(S) < L_K(S'), \text{ так як } p_{K'} > 0,$$

$$L_K(S) < \min\{L_{K'}(S'), L_K(S)\}.$$

Нехай L – мінімальна затримка $(n-2)$-ох робіт. Тоді

$$\min\{L, L_K(S), L_{K'}(S)\} \le \min\{L, L_{K'}(S'), L_K(S')\},$$

тобто, *теорему доведено.*

Якщо поряд з мінімізацією максимального часового зміщення намагаються мінімізувати і середню тривалість проходження, то впорядкування повинно виконуватися у відповідності із наступною теоремою.

Теорема. Якщо для системи $\mathbf{n|1}$ існує таке впорядкування, що максимальне запізнення робіт дорівнює 0, то існує й впорядкування з роботою K в останній позиції, яке зберігає нульовим максимальне запізнення і одночасно мінімізує середню тривалість проходження, тоді і тільки тоді, коли

a) $d_K \ge \sum_{i=1}^{n} p_i$;

b) $p_K \ge p_i$ при таких i, що $d_i \ge \sum_{j=1}^{n} p_j$.

Ця теорема стверджує, що робота виконується останньою, тільки якщо це не призводить до її запізнення і якщо її тривалість максимальна серед всіх робіт, виконання яких в останню чергу не призводить до запізнення.

Для побудови розкладу з найбільшою кількістю робіт, виконаних у задані директивні строки, потрібно вибрати в перестановці $\pi = (1,\ldots,n)$ серед перших k вимог вимогу з найбільшим значенням t_i і видалити її. Для отриманої перестановки повторюємо ті ж дії до тих пір, поки не буде знайдена перестановка, в якій всі вимоги, що залишилися, обслуговуються в задані директивні строки. Після того, як ці вимоги будуть обслужені, всі інші вимоги можуть обслуговуватися в довільній послідовності.

Приклад

Завод повинен виконати 5 замовлень. Тривалості t_i виконання замовлень і директивні строки d_i наведені в таблиці.

i	1	2	3	4	5
t_i	4	1	2	2	3
d_i	4	5	6	7	7

Замовлення пронумеровані так, що $d_i \le d_{i+1}$, $i = \overline{1,4}$. В якій послідовності слід виконувати замовлення, щоб найбільше число їх було виконано в задані директивні строки?

Розв'язання.

Виконуючи замовлення в послідовності $\pi = (1, 2, 3, 4, 5)$, маємо $\overline{t_1} = 4 \le d_1 = 4$; $\overline{t_2} = 5 \le d_2 = 5$; $\overline{t_3} = 7 \le d_3 = 6$. Видаляємо замовлення 1, оскільки серед замовлень 1, 2 і 3 йому відповідає найбільше значення $t_i = 4$.

Виконуючи замовлення, що залишилися у порядку $\pi = (2, 3, 4, 5)$, маємо

$$\overline{t_2} = 1 \le d_2 = 5;\ \overline{t_3} = 3 \le d_3 = 6;\ \overline{t_4} = 5 \le d_4 = 7;\ \overline{t_5} = 8 \ge d_5 = 7.$$

Видаляємо замовлення 5, оскільки серед замовлень 2, 3, 4 і 5 йому відповідає найбільше значення $t_i = 3$.

Виконуючи замовлення, що залишилися в послідовності $\pi = (2, 3, 4)$, приходимо до висновку, що всі вони виконуються в задані директивні строки. Після того, як ці замовлення будуть виконані, можна виконувати замовлення 1 і 5 в довільній послідовності, тобто $\pi^* = (2, 3, 4, 1, 5)$ або $\pi^* = (2, 3, 4, 5, 1)$. Число замовлень, що запізнюються дорівнює 2, і цими замовленнями є 1-е і 5-е.

3.2. Системи обслуговування з декількома приладами

1. **Ідентичні прилади. Загальний час обслуговування. Переривання.**
2. **Ідентичні прилади. Загальний час обслуговування. Однакові тривалості.**
3. **Ідентичні прилади. Директивні строки. Однакові тривалості.**
4. **Ідентичні прилади. Мінімізація максимального часового зміщення.**
5. **Різні прилади. Мінімізація сумарного і максимального штрафів.**
6. **Нефіксовані маршрути. Загальний час обслуговування. Два прилади.**
7. **Нефіксовані маршрути. Загальний час обслуговування. Три і більше приладів.**
8. **Нефіксовані маршрути. Загальний час обслуговування. Переривання.**

Ідентичні прилади. Загальний час обслуговування. Переривання

Розглядається задача побудови оптимального за швидкодією розкладу обслуговування n вимог m паралельними ідентичними приладами при можливості переривань обслуговування.

Оскільки в кожний момент часу кожна вимога може обслуговуватися тільки одним приладом, то при довільному (допустимому) розкладі S величина

$$t_{\max}(S) \geq T_1 = \max_{1 \leq i \leq n} \{t_i\}.$$

Аналогічно, оскільки в кожний момент часу кожен прилад може обслуговувати тільки одну вимогу, то

$$\bar{t}_{\max}(S) \geq T_2 = \sum_{i=1}^{n} \frac{t_i}{m}.$$

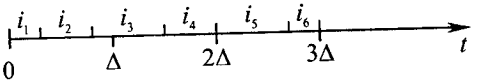
Тим самим

$$\bar{t}_{\max}(S) \geq \Delta = \max\{T_1, T_2\}.$$

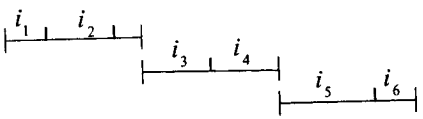
$m-1$ (максимальне число переривань для розкладу, отриманого у відповідності з алгоритмом упаковки) являється досяжною нижньою границею числа переривань в оптимальних розкладах. Іншими словами, існує така множина вимог із заданими тривалостями обслуговування, для якої оптимальний розклад містить не менше $m-1$ переривань.

Нехай множина N *невпорядкована*. Опишемо алгоритм побудови розкладу S^*, при якому $\bar{t}_{\max}(S^*) = \Delta$. Розклад S^*, очевидно, і буде шуканим оптимальним (за швидкодією) розкладом.

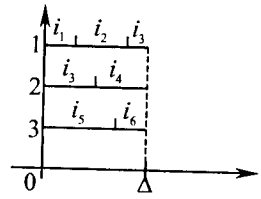
Виберемо довільну перестановку $\pi = (i_1, i_2, \ldots, i_n)$ елементів множини N і побудуємо відповідний розклад у припущенні, що всі вимоги обслуговуються одним приладом (нехай $n = 6$).



"Розріжемо" цей розклад на m частин довжиною Δ (нехай $m = 3$):



Сумістимо отримані частини в часі:



Позначимо побудований розклад через S^*. Цей розклад є допустимим: кожний прилад одночасно обслуговує не більше однієї вимоги і кожна вимога одночасно обслуговується не більш, ніж одним приладом. Значення $\bar{t}_{\max}(S^*) = \Delta$.

Вибираючи різні перестановки π, будемо отримувати різні розклади S^*. Будь-який із цих розкладів містить не більше $m-1$-ого переривання. Наведемо приклад, коли не існує оптимального (за швидкодією) розкладу з числом переривань, меншим $m-1$.

Нехай $N = \{1, 2, \ldots, m+1\}$ і $t_i = m \geq 4$, $i = \overline{1,n}$ (випадок $m \leq 3$ очевидний). Використовуючи описаний алгоритм, можна побудувати розклад S^* з $\bar{t}_{\max}(S^*) = m+1$ і числом переривань, рівним $m-1$. При цьому розкладі в часовому інтервалі $[0; m+1]$ немає простоїв обслуговуючих приладів. Очевидно, що будь-який оптимальний розклад не може допускати простоїв приладів у цьому інтервалі. Якщо припустити, що існує оптимальний розклад з числом переривань, меншим $m-1$, то при цьому розкладі, як мінімум, три вимоги повинні обслуговуватися без переривань трьома різними приладами. Повинен також існувати деякий підінтервал інтервалу $[0; m+1]$, в якому ці три прилади обслуговують, у крайньому випадку, дві нові вимоги. Інші $m-3$ прилади у вказаному підінтервалі можуть обслуговувати не більше, ніж $m-4$ вимоги, тобто хоча б один із цих приладів у вказаному підінтервалі простоює.

Ідентичні прилади. Загальний час обслуговування. Однакові тривалості

Розглянемо задачу побудови оптимального за швидкодією розкладу обслуговування частково впорядкованої множини вимог, що мають однакові тривалості обслуговування, паралельними ідентичними приладами в припущенні, що а) граф редукції відношення строгого порядку є деревовидним або б) число простоїв дорівнює двом.

Вимоги множини $N = \{1, 2, \ldots, n\}$ обслуговуються m паралельними ідентичними приладами. Всі вимоги поступають в чергу на

обслуговування одночасно (в момент часу $r=0$) і мають однакові тривалості обслуговування. Можна вважати $t_i=1\ \left(i=\overline{1,n}\right)$, де t_i – тривалість обслуговування вимоги i. Заборонені переривання в обслуговуванні вимог. На множині N задано відношення строгого порядку $\rightarrow$, яке визначає можливу послідовність обслуговування вимог. Граф редукції цього відношення позначимо через G. Нехай $\overline{t_i}(S)$ – момент завершення обслуговування вимоги i при розкладі S. Необхідно побудувати допустимий відносно $\rightarrow$ розклад S^* обслуговування вимог множини N, при якому загальний час обслуговування всіх вимог

$$T(S)=\max_{i\in N}\overline{t_i}(S)$$

є найменшими. Значення $T(S)$ будемо називати *довжиною* розкладу S, а розклад S^* – *оптимальним* (за швидкодією) розкладом.

Присвоїмо часовим інтервалам одиничної довжини, починаючи з моменту часу $t=0$, номери 1,2,... Інтервал із номером θ має вигляд $(\theta-1,\theta]$. Далі не будемо робити різницю між вимогою і відповідною їй вершиною графа G. Через N^- та N^+ будемо позначати множини всіх мінімальних і максимальних (відносно $\rightarrow$) елементів множини N; $A^\circ(i)$ – множину прямих нащадків, $B^\circ(i)$ – множину безпосередніх попередників вимоги i; $h(i)$ – висота вершини i в графі G.

Нехай кожна компонента зв'язності графа G редукції відношення строгого порядку $\rightarrow$ являється *вхідним деревом*.

Опишемо алгоритм побудови допустимого відносно $\rightarrow$ розкладу, який будемо називати *h-алгоритмом*, а побудований у відповідності з цим алгоритмом розклад будемо називати *h-розкладом*.

Кількість кроків h-алгоритму дорівнює довжині h-розкладу. Крок θ складається із виконання не більше $m+1$ ітерації, на кожній із яких (крім останньої) в одиничному часовому інтервалі θ призначається на обслуговування одна із вимог. На останній ітерації здійснюється перехід до наступного кроку. Загальне число ітерацій дорівнює $T(S)+n-1$, де $T(S)$ – довжина h-розкладу.

Попередньо покладемо $S_L(t)=0$ для $L=\overline{1,m}$, $t\geq 0$ і вважаємо всі вимоги множини N не поміченими. Покладаємо $\theta=1$.

На кожному кроці θ виконуються наступні ітерації. Знаходимо прилад H, для якого виконується $S_H(\theta)=0$. Серед непомічених вимог множини N^+ виберемо вимогу j з найбільшою висотою $h(j)$. Покладаємо $S_H(t)=j$ на інтервалі θ, помічаємо вимогу j і переходимо до наступної ітерації. Якщо прилад H знайти не вдається або всі вимоги множини N^+ помічені, то виключаємо із N помічені вимоги. Отримуючи

$N \neq \varnothing$, збільшуємо θ на 1 і переходимо до наступного кроку. Якщо $N = \varnothing$, h-розклад $S(t) = \{s_1(t), s_2(t), ..., s_m(t)\}$ побудовано.

Нехай кожна компонента зв'язності графа редукції відношення строгого порядку, заданого на N, є вхідним деревом:

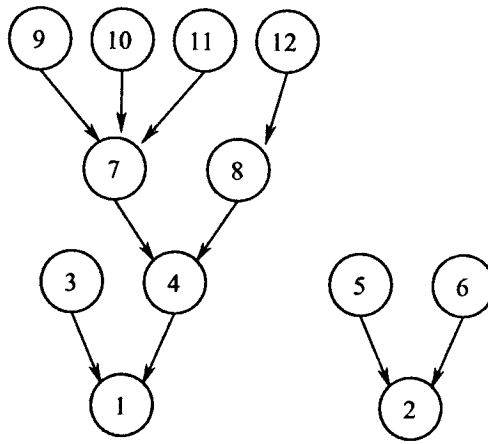


Рис. 3.3.

Вимоги (вершини графа) пронумеровані "по шарам" знизу вверх, зліва направо. Вимога 7 може обслуговуватися тільки після завершення обслуговування вимог 9, 10 і 11, вимога 8 – після 12, 4 – після 7, 8 і т. д. Вимоги 2, 5 і 6 можуть обслуговуватися незалежно від інших вимог (при виконанні умови, що вимога 2 може обслуговуватися тільки після завершення обслуговування вимог 5 і 6).

Вимога $i \in N$ називається максимальною відносно $\rightarrow$, якщо не існує такої вимоги j, що $j \rightarrow i$. На графі такій вимозі відповідає деяка початкова вершина, в яку не заходить жодна дуга. Множину всіх максимальних елементів позначимо через N^+. У розглядуваному прикладі $N^+ = \{3, 5, 6, 9, 10, 11, 12\}$.

Нехай λ – список вимог множини N, впорядкованих за спаданням номерів. На кожній ітерації h-алгоритму в якості вимоги j будемо брати перший непомічений елемент списку λ, що належить множині N^+. Отриманий в результаті розклад назвемо *λ-розкладом* (розклади такого типу називають *списочними*). Очевидно, λ-розклад, побудований у відповідності зі списком $\lambda = (n, n-1, ..., 2, 1)$, являється і h-розкладом.

Приклад

Нехай $m=3$, $N=\{1,2,...,12\}$, $t_i=1$, $r_i=0$, $i=\overline{1,12}$ і граф редукції відношення строгого порядку, заданого на N, зображений на рисунку 3.1. Побудувати λ-розклад.

В умовах розглядуваного прикладу на першому кроці множина $N^+=\{3,5,6,9,10,11,12\}$. Нехай $m=3$. Тоді $|N^+|=7>3$ і $N_1=\{10,11,12\}$. Вихідна для 2-го кроку множина N представляється на рисунку 3.4.

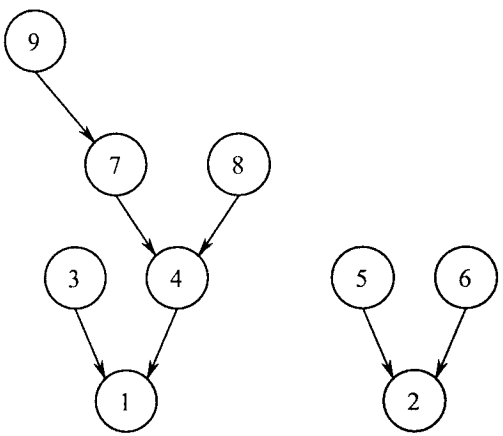


Рис. 3.4.

і множина $N^+=\{3,5,6,8,9\}$. Оскільки $|N^+|=5>3$, то $N_2=\{6,8,9\}$. Процес продовжується до тих пір, поки множина N не стане пустою.

Один із оптимальних за швидкодією розкладів представлений на рисунку 3.5.

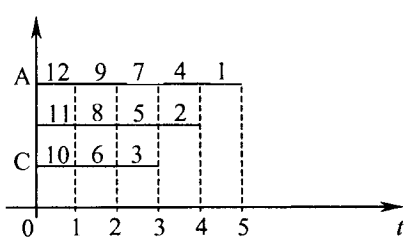


Рис. 3.5.

Якщо кожна компонента зв'язності графа редукції відношення строгого порядку $\rightarrow$, заданого на N, є вихідним деревом, то можна діяти наступним чином. Замінити орієнтацію кожної дуги на протилежну. Тим

самим на множині N буде задано відношення порядку, зворотне вихідному. Використовуючи описаний алгоритм, будуємо оптимальний за швидкодією розклад (рисунок 3.5). "Перевертаючи" його, отримуємо шуканий розклад.

Теорема. Якщо граф G редукції відношення строгого порядку $\to$, заданого на N, являється вхідним деревом, то h-розклад є оптимальним (за швидкодією) розкладом обслуговування вимог множини N.

Перейдемо до розгляду випадку, коли граф G редукції відношення $\to$, заданого на множині N, являється довільним орієнтованим безконтурним графом, але число простоїв $m = 2$.

Нехай $\nu = (\nu_1, \nu_2, ..., \nu_k)$ і $\mu = (\mu_1, \mu_2, ..., \mu_l)$ – послідовності цілих чисел, $k, l \geq 0$. Якщо $k = 0$, послідовність ν пуста. Будемо казати, що послідовність ν *лексикографічно менша* послідовності μ, якщо

1) існує таке i $(1 \leq i \leq k)$, що для всіх j $(1 \leq j < i)$ виконується $\nu_j = \mu_j$ та $\nu_i < \mu_i$, або

2) $\nu_j = \mu_j$ $(j = \overline{1,k})$ і $k \leq l$.

Пронумеруємо вершини графа G наступним чином. Присвоїмо номер 1 одній із кінцевих вершин. Нехай присвоєно номери 1, 2,..., $j-1$ й Q – множина таких непронумерованих вершин, у яких немає непронумерованих нащадків. Для кожної вершини $i \in Q$ побудуємо послідовність $a(i)$ всіх прямих нащадків (тобто вимог множини $A^{\circ}(i)$), в якій елементи розміщені за спаданням номерів. Присвоїмо номер j одній із вимог $i \in Q$ з лексикографічно найменшою послідовністю $a(i)$.

Для такої перенумерації вершин графа G потрібно виконати не більше $O(n^2)$ операцій.

Опишемо алгоритм побудови допустимого відносно $\to$ розкладу, який будемо називати (за аналогією з описаним вище) λ-розкладом.

Нехай вершини графа G пронумеровані числами 1, 2,..., n так як це вказано раніше, і список $\lambda = (n, n-1, ..., 2, 1)$.

Число кроків алгоритму побудови λ-розкладу S дорівнює довжині розкладу $T(S)$. На кроці θ $(\theta = \overline{1, T(S)})$ здійснюється побудова розкладу на інтервалі з номером θ, і вимоги, призначені на обслуговування в цьому інтервалі, виключаються з множини N. Нехай на кроці θ i та j – вимоги з найбільшими номерами із множин N^+ і $N^+ \setminus i$ відповідно. Покладаємо на інтервалі θ значення $s_1(t) = i$, і якщо $|N^+| > 1$, то $s_2(t) = j$, а в

протилежному випадку $s_2(t)=0$. Виключаємо i, а якщо $|N^+|>1$, то і j із N. Якщо $N\neq\varnothing$, збільшуємо θ на 1 і переходимо до наступного кроку. Якщо $N=\varnothing$, покладаємо $s_1(t)=s_2(t)=0$ при $t><0$. В результаті отримаємо λ-розклад $S(t)=\{s_1(t),s_2(t)\}$.

Для побудови множини N^+ і виключення елементів i та j із N на кожному кроці алгоритму достатньо виконати не більше $O(n)$ операцій. Отже, загальне число операцій алгоритму не перевищує $O(n^2)$.

Теорема. λ-розклад є оптимальним (за швидкодією) розкладом обслуговування вимог множини N двома приладами.

Зауваження. λ-розклад, взагалі кажучи, не оптимальний, якщо $m\neq 2$.

Ідентичні прилади. Директивні строки. Однакові тривалості

1. Розглянемо задачу побудови розкладу обслуговування частково впорядкованої множини вимог, що мають однакові довжини тривалості обслуговування, паралельними ідентичними приладами, при яких не порушуються директивні строки. Припускається, що граф редукції відношення строгого порядку являється вхідним деревом або число приладів рівне двом.

Вимоги множини $N=\{1,2,...,n\}$ обслуговуються m паралельними ідентичними приладами. Вимоги поступають в чергу на обслуговування одночасно в момент часу $r=0$ Довжини t_i обслуговування вимог однакові. Без втрати загальності можна покласти $t_i=1$ $(i=\overline{1,n})$. Для кожної вимоги i задано директивний строк d_i, до якого необхідно завершити його обслуговування.

Заборонені переривання в обслуговуванні вимог. На множині N задано відношення строгого порядку $\rightarrow$, яке визначає можливу послідовність обслуговування вимог. Граф редукції цього відношення позначимо через G.

Через N^- та N^+ будемо позначувати множину усіх мінімальних та максимальних (відносно $\rightarrow$) елементів множини N, через $B(i)$ та $A(i)$ – множини всіх вимог j, для яких виконується $j\rightarrow i$ та $i\rightarrow j$ відповідно; позначення $i\mapsto k$ означає, що вимога k є прямим нащадком вимоги i.

Необхідно побудувати допустимий відносно $\rightarrow$ розклад S обслуговування вимог множини N, при якому $\bar{t}_i(S)\le d_i$ $(i=\overline{1,n})$. Тут

$\overline{t_i}(S)$ – момент завершення обслуговування вимоги i при розкладі S. Такий розклад називається *допустимим* відносно заданих директивних строків.

Введемо корисне в подальшому поняття λ-розкладу. Присвоїмо часовим інтервалам одиничної довжини, починаючи з моменту часу $t=0$, номери 1, 2, ... Інтервал з номером θ має вигляд $(\theta-1,\theta]$. Розглянемо список $\lambda=(i_1,i_2,...,i_n)$ вимог (що являють собою деяку перестановку вимог) та визначимо розклад, побудований у відповідності зі списком λ, наступним чином.

Попередньо вважаємо $\theta=1$, $S_L(t)=0$ при $L=\overline{1,m}$, $t\geq 0$, та вважаємо всі елементи списку λ непоміченими. На кожному кроці виконуємо наступні дії.

Знаходимо прилад H з найменшим номером, для якого $S_H(\theta)=0$. Знаходимо першу непомічену вимогу j списку λ, яка належить множині N^+. Помітимо вимогу j, вважаємо $S_H(t)=j$ на інтервалі θ і переходимо до наступного кроку. Якщо прилад H або вимогу j знайти не вдається, то вилучаємо помічені вимоги із списку λ і множини N; переходимо до наступного кроку, збільшуючи θ на 1. Побудова розкладу закінчується тоді, коли список λ (а відповідно, і множина N) стає пустим.

Побудований у відповідності з цим алгоритмом розклад будемо називати *λ-розкладом.*

Приклад 1.

Нехай $N=\{1,2,...,11\}$, $m=2$, $r_i=1$, $d_i=0$, $i=\overline{1,11}$; граф G наведений на рис.3.6,а), $\lambda=(1,2,...,11)$. Відповідний λ-розклад наведено на рис.3.6,б).

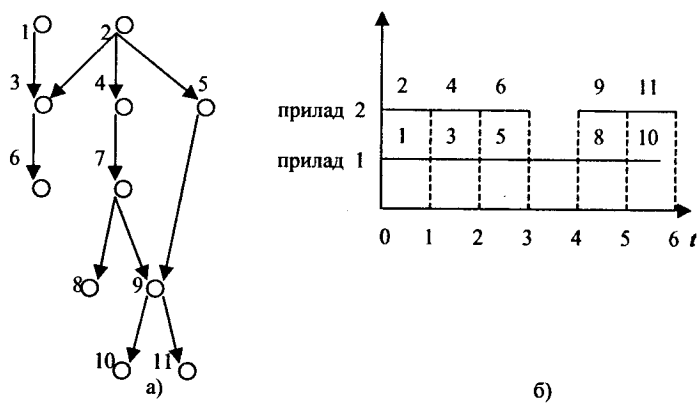
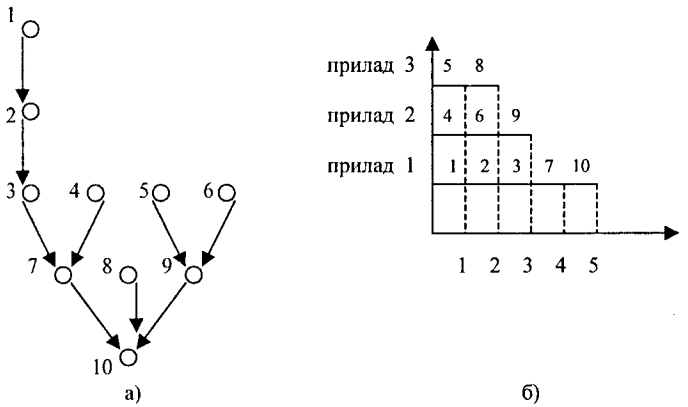


Рис.3.6

Приклад 2.

Нехай $N=\{1,2,...,10\}$, $m=3$, $t_i=1$, $r_i=0$, $i=\overline{1,10}$; граф G наведений на рис.3.7,а), $\lambda=(1,2,...,10)$. Відповідний λ-розклад наведено на рис.3.7,б).



а) б)

Рис. 3.7.

2. Розглянемо задачу побудови допустимого відносно заданих директивних строків розкладу в припущенні, що граф G являється вхідним деревом. Нагадаємо, що $r_i=0$ та $t_i=1$ $(i=\overline{1,n})$.

Якщо $i\to j$, то для того, щоб вимоги i та j обслуговувались без порушення директивних строків, необхідно обслуговування вимоги i завершити не пізніше моменту часу d_j-1. Тому директивний строк вимоги i можна покласти рівним $\min\{d_i,d_j-1\}$. Використовуючи цей факт, можна запропонувати наступний алгоритм модифікації директивних строків.

Алгоритм складається із n кроків. На першому кроці покладаємо $d'_k=d_k$ для кореня k дерева. На кожному наступному кроці вибираємо таку вимогу i, для якої не знайдено значення d'_i, але для його прямого нащадка j значення d'_j знайдено. Покладаємо $d'_i=\{d_i,d'_j-1\}$.

Неважко бачити, що *розклад допустимий відносно модифікованих директивних строків d'_i тоді і тільки тоді, коли він допустимий відносно початкових директивних строків d_i.*

Таким чином, в подальшому можна говорити про розклад, допустимий відносно директивних строків, не уточнюючи, чи являються ці строки початковими чи модифікованими.

Теорема. Для того щоб існував допустимий (відносно директивних строків) розклад, необхідно і досить, щоб був допустимим λ-розклад, що відповідає списку $\lambda = (i_1, i_2, ..., i_n)$, де $d'_{i_j} \le d'_{i_{j+1}} \quad (j = \overline{1, n-1})$.

Таким чином, якщо G – вхідне дерево, $r_i = 0$ та $t_i = 1 \quad (i = \overline{1,n})$, то для побудови допустимого відносно директивних строків розкладу (якщо він існує) достатньо: а) обчислити модифіковані директивні строки d'_i; б) отримати список λ вимог, впорядкованих по неспаданню значень d'_i; в) побудувати λ-розклад. В результаті виконання не більше $O(n \log n)$ операцій буде побудовано допустимий розклад або зроблено висновок про те, що допустимого розкладу не існує.

Приклад

Нехай в умовах прикладу 2 значення директивних строків d'_i вимог задано таблицею. Отримані модифіковані директивні строки d'_i наведені в цій же таблиці.

i	1	2	3	4	5	6	7	8	9	10
d_i	6	5	6	4	9	8	7	3	2	8
d'_i	4	5	6	4	1	1	7	3	2	8

Список λ вимог, впорядкованих по неспаданню значень d'_i, має вигляд: $\lambda = (5, 6, 9, 8, 1, 4, 2, 3, 7, 10)$.

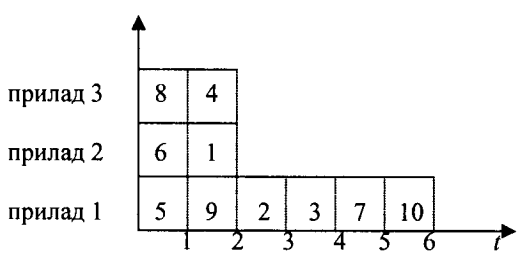


Рис. 3.8.

Відповідний цьому списку λ-розклад наведено на рисунку 3.8. Цей розклад являється допустимим відносно заданих директивних строків.

На завершення цього питання відмітимо, що якщо всі директивні строки *однакові*, тобто $d_i = d \quad (i = \overline{1,n})$, то із процедури знаходження

модифікованих директивних строків отримуємо $d_i' = d - (h_i - 1)$, де h_i – висота вершини i графа G. Список λ вимог, впорядкованих за неспаданням модифікованих директивних строків, в цьому випадку співпадає зі списком вимог, впорядкованих по незростанню висот, і не залежить від величини d. Якщо в якості d взяти найменший директивний строк, при якому існує допустимий відносно цього строку розклад S, то цей розклад являється, очевидно, оптимальним по швидкодії. Отже, в розглядуваному випадку (G – вхідне дерево, $r_i = 0$, $t_i = 1$, $i = \overline{1,n}$) оптимальним за швидкодією розкладом являється λ-розклад, що відповідає списку λ, в якому вимоги впорядковані по незростанню висот.

3. Перейдемо до розглядуваного випадку, коли граф G редукції відношення строгого порядку, заданого на множині N, являється довільним орієнтованим безконтурним графом, але число приладів $m = 2$. Також припускається, що $r_i = 0$ та $t_i = 1$ $(i = \overline{1,n})$.

Відмітимо, що якщо k вимог множини $A(i)$ мають директивні строки, що не перевищують d, то обслуговування вимоги i при довільному допустимому відносно директивних строків розкладі повинно бути завершено не пізніше моменту часу $d - \lfloor k/2 \rfloor$, де $[x]$ – найменше ціле, більше або рівне x. Тому директивний строк вимоги i можна покласти рівним $\min\{d_i, d - \lfloor k/2 \rfloor\}$. Використовуючи цей факт, можна запропонувати наступний *алгоритм* модифікації директивних строків.

Вважаємо спочатку модифікований директивний строк $d_j' = d_j$ для всіх $j \in N^-$. На кожному наступному кроці вибираємо вимогу $i \in N$, для якої не обчислений модифікований директивний строк, але для всіх вимог множини $A(i)$ модифіковані директивні строки обчислені. Нехай $d^{(1)}, d^{(2)}, \ldots, d^{(l)}$ – множина всіх різних модифікованих директивних строків, що відповідають вимогам із $A(i)$, а $g\left(i, d^{(k)}\right)$ – число елементів множини $A(i)$, які мають модифікований директивний строк, що не перевищує $d^{(k)}$ $(k = \overline{1,l})$. Покладаємо

$$d_i' = \min\left\{d_i \min_{1 \le k \le l}\left\{d^{(k)} - \left\lceil \frac{1}{2} g\left(i, d^{(k)}\right)\right\rceil\right\}\right\}.$$

Розклад допустимий відносно модифікованих директивних строків тоді і тільки тоді, коли він допустимий відносно вихідних директивних строків.

Теорема. Для того щоб існував допустимий (відносно директивних строків) розклад, необхідно і достатньо, щоб був допустимим λ-розклад, що відповідає списку $\lambda = (i_1, i_2, ..., i_n)$, де $d'_{i_j} \le d'_{i_{j+1}}$ $(j = \overline{1, n-1})$.

Таким чином, якщо $m = 2$, $r_i = 0$, $t_i = 1$, $i = \overline{1,n}$, і відношення строгого порядку $\rightarrow$ задане в транзитивно замкненій формі, то для *побудови допустимого* відносно директивних строків розкладу (якщо він існує) достатньо:

а) обчислити модифіковані директивні строки d'_i (не більше $O(n^2)$ операцій);

б) отримати список λ вимог, впорядкованих по неспаданню значень d'_i (не більше $O(n \log n)$ операцій);

в) побудувати λ-розклад (не більше $O(n^2)$ операцій).

В результаті виконання не більше $O(n^2)$ операцій буде побудовано допустимий розклад, або зроблено висновок про те, що допустимого розкладу не існує.

Неважко бачити, що якщо всі вихідні директивні строки *однакові* $(d_i = d, i = \overline{1,n})$, то список λ вимог, впорядкованих по неспаданню модифікованих директивних строків, не залежить від величини d, а визначається тільки видом графа G.

Отже, *оптимальний за швидкодією розклад* обслуговування частково впорядкованої множини вимог однакової тривалості двома приладами можна побудувати, поклавши всі вихідні директивні строки рівними одній і тій же величині (наприклад $d_i = d$, $i = \overline{1,n}$) і застосувавши наведений вище алгоритм побудови допустимого відносно директивних строків розкладу.

Зауваження 1. Алгоритм побудови допустимого розкладу, описаний в п. 3, можна застосовувати й у випадку, коли *моменти* r_i *поступлення вимог відрізняються, а всі директивні строки однакові* $(d_i = d,\ i = \overline{1,n})$.

Зауваження 2. Наведені в пунктах 2 і 3 алгоритми можна використати для побудови розкладу, допустимого відносно директивних строків з довжиною, що не перевищує заданого числа d. Для цього достатньо покласти вихідні директивні строки, що більші від d, рівними d.

Ідентичні прилади. Мінімізація максимального часового зміщення

Розглянемо задачу побудови розкладу обслуговування вимог паралельними ідентичними приладами, якому відповідає найменше значення максимального часового зміщення. У випадку частково впорядкованої множини вимог припускається, що тривалості обслуговування вимог однакові, вимоги поступають в чергу на обслуговування одночасно і заборонені переривання; граф редукції відношення строгого порядку є вхідним деревом або число приладів рівне двом. У випадку невпорядкованої множини вимог моменти їх поступлення можуть бути різними і дозволені переривання.

1. Вимоги множини $N = \{1, 2, ..., n\}$ обслуговуються m паралельними ідентичними приладами. Вимога $i \in N$ поступає в чергу на обслуговування в момент часу $r_i \geq 0$, тривалість її обслуговування рівна $t_i > 0$, а бажаний (директивний) строк закінчення обслуговування рівний $d_i \geq 0$. На множині N задано відношення строгого порядку $\rightarrow$, що визначає можливу послідовність обслуговування вимог. Граф редукції цього відношення позначимо через $G = (N, U)$. Розклад S допустимий відносно $\rightarrow$, якщо для будь-яких $i, j \in N$ таких, що $i \rightarrow j$, із $S_H(t') = i$ $(1 \leq H \leq m)$ слідує $S_L(t) \neq j$ $(L = \overline{1,m})$ для всіх $t \leq t'$.

Оптимальним розкладом будемо називати допустимий відносно $\rightarrow$ розклад S^*, якому відповідає найменше значення функціоналу

$$L_{\max}(S) = \max_{i \in N} L_i(S),$$

де $L_i(S) = \overline{t_i}(S) - d_i$ – часове зміщення в обслуговуванні вимоги i;

$\overline{t_i}(S)$ — момент закінчення обслуговування вимоги i при розкладі S.

Значення $L^* = L_{\max}(S^*)$ будемо називати оптимальним значенням максимального часового зміщення.

Перш ніж перейти до безпосереднього опису алгоритмів побудови оптимальних розкладів, зробимо одне зауваження.

Для будь-якого допустимого відносно $\rightarrow$ розкладу S справедливі нерівності $\overline{t_i}(S) \leq d_i + L_{\max}(S)$ і $L^* \leq L_{\max}(S)$. Отже, не існує допустимого відносно $\rightarrow$ розкладу S такого, що $\overline{t_i}(S) \leq d_i + \tau$ при $\tau < L^*$ $(i = \overline{1,n})$. Тим самим задача побудови оптимального розкладу зводиться до знаходження найменшого значення τ, при якому існує допустимий як відносно $\rightarrow$, так і відносно модифікованих директивних строків $d' = d_i + \tau$ розклад. Цей розклад є шуканим оптимальним розкладом S^*, а знайдене значення $\tau = L^*$.

Нижче розглядаються наступні частинні випадки задачі побудови оптимального розкладу:

а) $r_i = 0$, $t_i = 1$, $i = \overline{1,n}$, заборонені переривання процесу обслуговування і граф G є вхідним деревом;

б) $r_i = 0$, $t_i = 1$, $i = \overline{1,n}$, заборонені переривання і $m = 2$;

в) r_i, t_i, d_i – цілі числа, $i = \overline{1,n}$, $G = (N, \varnothing)$ і дозволені переривання.

2. Розглянемо випадки а) і б). Якщо відоме значення L^*, то для побудови оптимального розкладу можна, очевидно, скористатися алгоритмами побудови розкладів, допустимих як відносно $\rightarrow$, так і відносно директивних строків, рівних $d_i + L^*$ (див. пп. 2, 3 попереднього параграфу). Кожний із цих алгоритмів містить ту характерну особливість, що при будь-якому $\delta \geq 0$ і директивних строках, рівних $d_i + L^* + \delta$, будується один і той же розклад.

Таким чином, для побудови оптимального розкладу у випадках а) і б) достатньо в якості директивних строків вибрати величини $d_i + W$, де W – достатньо велике число, і скористатися відповідними алгоритмами попереднього параграфа.

3. Розглянемо випадок в). Розв'язок задачі побудови оптимального розкладу будемо шукати, вибираючи пробні значення τ і перевіряючи, чи існує розклад S, допустимий відносно директивних строків, рівних $d_i + \tau$. Якщо такий розклад існує, то розглядуване значення τ будемо називати *допустимим*. Починаючи з деякого недопустимого значення τ, будемо збільшувати τ до тих пір, поки не отримаємо найменше допустиме значення τ. В силу зробленого в п. 1 зауваження це значення τ рівне L^*.

Побудову розкладу обслуговування вимог без порушення директивних строків у розглядуваному випадку можна виконати з використанням потокової моделі. Для кожного пробного значення τ потокова модель будується наступним чином. Нехай $\{e_1, e_2, \ldots\ldots, e_{2n}\}$, де $e_1 \leq e_2 \leq \ldots \leq e_{2n}$ – множина значень r_i та $d_i + \tau$ $(i = \overline{1,n})$ і $E_k = (e_k, e_{k+1}]$ $(k = \overline{1,2n-1})$. Якщо $r_i = d_j + \tau$ для деяких i та j, то e з меншим індексом відповідає значенню r_i. Мережа Г містить джерело x_0, з'єднане дугами пропускної здатності $M(e_{k+1} - e_k)$ з вершинами x_k $(k = \overline{1,2n-1})$ (які відповідають інтервалам E_k), і стік z, в який входять дуги пропускної здатності t_i із вершин y_i $(i = \overline{1,n})$ (що відповідають вимогам i). Дуга (x_k, y_i) пропускної здатності $e_{k+1} - e_k$ проводиться тоді і тільки тоді, коли $r_i \leq e_k$ і $e_{k+1} \leq d_i + \tau$. Пробне значення τ допустиме тоді і тільки тоді,

коли величина максимального потоку в мережі рівна $\theta = \sum_{i=1}^{n} t_i$ (тобто насичуються вихідні дуги мережі).

Структура мережі Г, очевидно, залежить від величини τ. Значення τ будемо називати *критичним*, якщо існують такі i та j із N, що $d_i + \tau = r_j$. Структура мережі залишається незмінною для всіх значень τ, що містяться між двома послідовними критичними значеннями.

Значення $\tau = L^*$ будемо шукати в два етапи.

На першому етапі знаходимо τ_0 – найбільше недопустиме критичне значення τ.

На другому етапі виконуємо наступну процедуру для значень $\tau = \tau_\nu$, починаючи з $\nu = 0$. Знаходимо максимальну величину потоку і значення пропускної здатності мінімального розрізу мережі, що відповідає значенню τ_ν. Нехай R_ν – розріз мережі, якому відповідає мінімальна пропускна здатність θ_ν. Якщо $\theta_\nu < \theta$, то збільшуємо значення τ_ν так, щоб пропускна здатність розрізу R_ν стала рівною θ. Отримане значення τ позначимо через $\tau_{\nu+1}$ і повторюємо процедуру. В результаті отримаємо зростаючу послідовність значень τ_ν. Процес закінчується на кроці k, на якому $\theta_k = \theta$ і, відповідно, τ_k – найменше допустиме значення τ, тобто $\tau_k = L^*$.

Доведено, що побудова *оптимального за швидкодією* розкладу при довільних r_i зводиться до побудови розкладу з найменшим значенням максимального часового зміщення при $r_i = 0$ $(i = \overline{1,n})$.

Відмітимо також, що розкладу, якому відповідає найменше значення $L_{\max}(S)$, відповідає і найменше максимальне запізнення.

Різні прилади. Мінімізація сумарного і максимального штрафів

Розглянемо ряд задач мінімізації сумарного і максимального штрафу за обслуговування вимог паралельними неідентичними приладами (і, в тому числі, задачі побудови оптимальних за швидкодією розкладів), для яких відомі поліноміально обмежені алгоритми розв'язання.

1. Вимоги множини $N = \{1, 2, ..., n\}$ обслуговуються m паралельними приладами. Всі вимоги поступають в чергу на обслуговування одночасно в момент часу $r = 0$. Тривалість обслуговування вимоги i приладом L рівна $t_{iL} \geq 0$. Кожній вимозі i поставлена у відповідність неспадна функція штрафу $\varphi_i(t)$.

Задача мінімізації сумарного штрафу полягає в побудові розкладу S^* обслуговування вимог множини N, при якому функціонал

$$F_\Sigma(S)=\sum_{i=1}^{n}\varphi_i(\bar{t}_i(S)) \qquad (1)$$

приймає найменше значення.

Задача мінімізації максимального штрафу полягає в побудові розкладу S^* обслуговування вимог множини N, при якому функціонал

$$F_{\max}(S)=\max_{i\in N}\{\varphi_i(\bar{t}_i(S))\} \qquad (2)$$

приймає найменше значення.

Тут $\bar{t}_i(S)$ – момент закінчення обслуговування вимог при розкладі S. Розклад S^* в кожній із задач будемо називати *оптимальним*.

Якщо переривання обслуговування кожної вимоги заборонені, то розклад однозначно визначається розбиттям множини N на підмножини $N_1, N_2, \ldots, N_m$ (деякі з них, можливо, пусті) і заданням послідовності обслуговування вимог множини N_L приладом L $\left(L=\overline{1,m}\right)$.

Нижче розглядаються задачі мінімізації сумарного штрафу при забороні переривань у випадках, коли:

а) $\varphi_i(t)=t,\ i=\overline{1,n}$;

б) $\varphi_i(t)=t,\ t_{iL}=a_L t_i,\ t_i>0,\ a_L>0,\ i=\overline{1,n},\ L=\overline{1,m}$;

в) $\varphi_i(t)$ – неспадна функція, $t_{iL}=a_L,\ a_L>0,\ i=\overline{1,n},\ L=\overline{1,m}$;

г) $\varphi_i(t)=\alpha_i u_i(t),\ t_{iL}=1,\ i=\overline{1,n},\ L=\overline{1,m}$ (тут $u_i(t)=0$, якщо $t\le d_i$, $u_i(t)=1$, якщо $t>d_i$), $\alpha_i>0,\ d_i\ge 0$ – директивний строк завершення обслуговування вимоги i; d_i $\left(i=\overline{1,n}\right)$ – цілі числа;

і задачі мінімізації максимального штрафу у випадках, коли:

а) $\varphi_i(t)$ – неспадна функція, $t_{iL}=a_L,\ a_L>0,\ i=\overline{1,n},\ L=\overline{1,m}$, і переривання заборонені;

б) $\varphi_i(t)=t,\ i=\overline{1,n}$;

в) $\varphi_i(t)=t-d_i,\ i=\overline{1,n}$.

2. Перейдемо до розгляду першої із вказаних задач. Необхідно побудувати розклад S^*, якому відповідає найменше значення $\sum_{i=1}^{n}\bar{t}_i(S^*)$.

Нехай S – деякий розклад, вимога i обслуговується при цьому розкладі приладом L і після вимоги i приладом L обслуговується $k-1$ вимоги $(1\le k\le n)$. Тоді тривалість обслуговування t_{iL} входить в якості доданку у вираз для обчислення значення $\bar{t}_i$ і обчислення значень $\bar{t}_j$ для

$k-1$ вимоги j, які обслуговуються приладом L після вимоги i. Суму $\sum_{i=1}^{n} \bar{t}_i(S)$ можна представити у вигляді: kt_{iL} плюс ті доданки, які не залежать від t_{iL}. Якщо вимога i обслуговується приладом L останньою, то коефіцієнт при t_{iL} буде рівним 1, якщо передостанньою – то він дорівнює 2 і т.д.

Введемо змінну x_{iLk}, яка приймає значення 1, якщо вимога i обслуговується приладом L і цей прилад обслуговує $k-1$ вимогу після вимоги i. Інакше $x_{iLk}=0$. Тоді розглядувана задача може бути сформульована у вигляді наступної задачі транспортного типу.

Необхідно мінімізувати

$$\sum_{i=1}^{n}\sum_{L=1}^{m}\sum_{k=1}^{n} kt_{iL}x_{iLk} \qquad (3)$$

при умовах

$$\sum_{L=1}^{m}\sum_{k=1}^{n} x_{iLk}=1, \quad i=\overline{1,n}, \qquad (4)$$

$$\sum_{i=1}^{n} x_{iLk}\le 1, \quad L=\overline{1,m},\ k=\overline{1,n}, \qquad (5)$$

$$x_{iLk}\ge 0, \quad i=\overline{1,n},\ L=\overline{1,m},\ k=\overline{1,n}. \qquad (6)$$

Умова (4) означає, що кожна вимога i повинна обслуговуватись одним із приладів і займати певну позицію в послідовності обслуговування вимог, які відповідають цьому приладу. Умова (5) означає, що в послідовності обслуговування вимог, які відповідають будь-якому приладу, кожна з позицій зайнята не більше, ніж однією вимогою. Розв'язок оптимальної задачі може бути отриманий за $O(n^2)$ операцій).

3. Нехай тривалість обслуговування вимоги i приладом L рівна $t_{iL}=a_L t_i$, де $a_L>0$, $t_i>0$, $i=\overline{1,n}$, $L=\overline{1,m}$, тобто для кожного приладу L задана "продуктивність" $1/a_L$ обслуговування вимог. І надалі необхідно побудувати розклад, якому відповідає найменше значення $\sum_{i=1}^{n} \bar{t}_i(S)$.

Розглянемо допоміжну задачу. Дано два n-вимірні вектора $\alpha=(\alpha_1,\alpha_2,\ldots,\alpha_n)$ і $\beta=(\beta_1,\beta_2,\ldots,\beta_n)$, компоненти яких – дійсні числа. Позначимо через $\pi=(i_1,i_2,\ldots,i_n)$ деяку перестановку елементів множини $\{1,2,\ldots,n\}$. Визначимо

$$f(\pi)=\sum_{k=1}^{n}\alpha_k\beta_{i_k}.$$

Потрібно побудувати перестановку π^* елементів множини $\{1, 2, \ldots, n\}$, якій відповідає найменше значення $f(\pi)$. Не порушуючи загальності, будемо вважати, що компоненти векторів α і β пронумеровані так, що $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n$.

Розглянемо перестановку π', яка відрізняється від π транспозицією елементів i_k та i_{k+1}. Маємо:

$$f(\pi') - f(\pi) = (\alpha_k - \alpha_{k+1})(\beta_{i_{k+1}} - \beta_{i_k}).$$

Якщо $\beta_{i_k} \leq \beta_{i_{k+1}}$, то ця різниця невід'ємна. Таким чином, впорядковуючи числа β_l в порядку їх не спадання, дістаємо шукану перестановку $\pi^* = (i_1^*, i_2^*, \ldots, i_n^*)$. Тут $\beta_{i_k^*} \leq \beta_{i_{k+1}^*}$ для всіх $k = \overline{1, n-1}$.

Тим самим функціонал $f(\pi)$ досягає найменшого значення тоді і тільки тоді, коли великим α_k відповідають менші β_{i_k}.

Перейдемо до розгляду задачі побудови розкладу S^* з найменшим значенням $\sum_{i=1}^{n} \bar{t}_i(S)$. Не втрачаючи загальності, будемо вважати, що вимоги $i \in N$ пронумеровані так, що $t_1 \geq t_2 \geq \ldots \geq t_n$. Побудуємо $m \times n$-матрицю $A = \|\alpha_{Lk}\|$, вважаючи $\alpha_{Lk} = ka_L$, тобто

$$A = \begin{Vmatrix} a_1 & 2a_1 & \ldots & na_1 \\ a_2 & 2a_2 & \ldots & na_2 \\ \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots \\ a_m & 2a_m & \ldots & na_m \end{Vmatrix}.$$

Впорядкуємо елементи матриці A в порядку неспадання і позначимо j-й елемент отриманої послідовності через β_j, так що $\beta_1 \leq \beta_2 \leq \ldots \leq \beta_{mn}$.

Покладемо у відповідність елементу $i \in N$ елемент β_i матриці A. Якщо $\beta_i = \alpha_{Lk}$, то вимогу i будемо обслуговувати приладом L, причому ця вимога буде займати k-е з кінця місце в послідовності обслуговування вимог цим приладом. Зауважимо, що якщо деякій вимозі i покладено у відповідність елемент α_{Lk} $(k > 1)$, то для будь-якого $\alpha_{Lk'}$, $k' < k$, знайдеться відповідний йому елемент $j \in N$ (так як $\alpha_{Lk'} < \alpha_{Lk}$). В результаті отримуємо послідовності вимог, які обслуговуються кожним із приладів. Тим самим визначено деякий розклад S без переривань процесу обслуговування. Цей розклад оптимальний в силу наведених вище (при мінімізації функціонала $f(\pi)$) міркувань.

Якщо прилади ідентичні (випадок, коли всі a_L однакові), то розклад S^*, побудований описаним вище способом, являється, по суті, розкладом,

побудованим за відомим правилом "найкоротшої операції": в момент звільнення приладу серед вимог, не назначених на обслуговування, вибирається вимога з найменшою тривалістю обслуговування і призначається на обслуговування даним приладом.

4. Нехай $t_{iL} = a_L$, $i = \overline{1,n}$, $L = \overline{1,m}$, і функції штрафу $\varphi_i(t)$ – неспадаючі. Тоді задача побудови розкладу, якому відповідає найменше значення функціоналу (1), зводиться до наступної задачі транспортного типу.

Введемо змінну x_{iLk}, яка приймає значення 1, якщо вимога i обслуговується приладом L, займаючи k-е місце в послідовності обслуговування вимог цим приладом. В противному випадку $x_{iLk} = 0$. Нехай $c_{iLk} = \varphi_i(ka_L)$, $i = \overline{1,n}$, $L = \overline{1,m}$, $k = \overline{1,n}$.

Необхідно мінімізувати

$$\sum_{i=1}^{n}\sum_{L=1}^{m}\sum_{k=1}^{n} c_{iLk}x_{iLk} \qquad (7)$$

при умові

$$\sum_{L=1}^{m}\sum_{k=1}^{n} x_{iLk} = 1, \quad i = \overline{1,n}, \qquad (8)$$

$$\sum_{i=1}^{n} x_{iLk} \le 1, \qquad L = \overline{1,m},\ k = \overline{1,n}, \qquad (9)$$

$$x_{iLk} \ge 0,\ i = \overline{1,n},\ L = \overline{1,m},\ k = \overline{1,n}. \qquad (10)$$

Умова (8) означає, що кожна вимога повинна обслуговуватись одним із приладів, займаючи визначену позицію в послідовності обслуговування вимог цим приладом. Умова (9) означає, що в послідовності обслуговування вимог, яка відповідає будь-якому приладу, кожна із позицій зайнята не більше, ніж однією вимогою.

Задача (7)–(10) аналогічна задачі (3)–(6), і її розв'язок може бути отриманий в результаті виконання не більше $O(n^3)$ операцій. Аналогічно розв'язується також задача мінімізації максимального штрафу при $t_{iL} = a_L$, $i = \overline{1,n}$, $L = \overline{1,m}$, тільки функціонал (7) замінюється на $\max\limits_{i,L,k} c_{iLk}x_{iLk}$.

5. Нехай $t_{iL} = 1$, $i = \overline{1,n}$, $L = \overline{1,m}$, і функції штрафу мають вигляд $\varphi_i(t) = \alpha_i u_i(t)$, де $u_i(t) = 0$, якщо $t \le d_i$, $u_i(t) = 1$, якщо $t > d_i$; $\alpha_i > 0$, $i = \overline{1,n}$. Тут d_i – ціле число, директивний строк завершення обслуговування вимоги i. Необхідно побудувати розклад, при якому функціонал (1) приймає найменше значення (задача I).

Поряд з даною задачею розглянемо задачу II, яка відрізняється від попередньої тим, що тривалості обслуговування вимог дорівнюють $1/m$ і вимоги обслуговуються одним приладом.

Будемо говорити, що при розкладі S обслуговування вимог m паралельними приладами немає невиправданих простоїв приладів, якщо або $S_L(t)=0$ на проміжку $(0,\infty)$, або $S_L(t)\neq 0$ на проміжку $(0,t']$ і $S_L(t)=0$ при $t>t'$ ($L=\overline{1,m}$). Аналогічно, при розкладі S' обслуговування вимог одним приладом немає невиправданих простоїв приладу, якщо $S'(t)\neq 0$ на проміжку $\left(0,\sum_{i=1}^{n}t_i\right]$ і $S'(t)=0$ за межами цього інтервалу.

Нехай $\{S\}$ і $\{S'\}$ – множини всіх розкладів без переривань і без невиправданих простоїв приладів для задач I і II відповідно.

Очевидно, існують оптимальні розклади для задач I і II, які містяться в множинах $\{S\}$ і $\{S'\}$.

Розклади S і S' для задач I і II назвемо спряженими, якщо для кожного одиничного інтервалу $(\theta-1,\theta]$, $\theta=1,2,\ldots$, виконується $S'(t)=S_L(\theta)$ для всіх $t\in(\theta-1+(L-1)/m,\theta-1+L/m)$ (і навпаки, $S_L(t)=S'(\theta-1+L/m)$ для всіх $t\in(\theta-1,\theta]$, $L=\overline{1,m}$).

Неважко бачити, що а) якщо розклад S' належить $\{S'\}$, то спряжений йому розклад S належить $\{S\}$ (обернене твердження, взагалі кажучи, не вірне), б) значення функціоналів (1) для спряжених розкладів в задачах I і II співпадають.

Звідси випливає, що *для того, щоб отримати оптимальний розклад для задачі* I, *досить знайти оптимальний розклад* $S'^{*}\in\{S'\}$ *для задачі* II *і побудувати спряжений йому розклад.*

Приклад

В якості задачі I розглянемо задачу, в якій $m=3$, $n=10$, значення d_i та α_i, $i=\overline{1,10}$, наведені в таблиці. Оптимальний розклад S'^{*} для відповідної їй задачі II представлений на рис. 3.9а), а спряжений йому розклад S^{*}, який є оптимальним для задачі I , представлений на рис. 3.9б). При цьому $F_{\Sigma}(S^{*})=F_{\Sigma}(S'^{*})=2$.

i	1	2	3	4	5	6	7	8	9	10
d_i	1	1	1	2	2	2	2	3	3	4
α_i	5	4	6	2	3	4	5	1	3	2

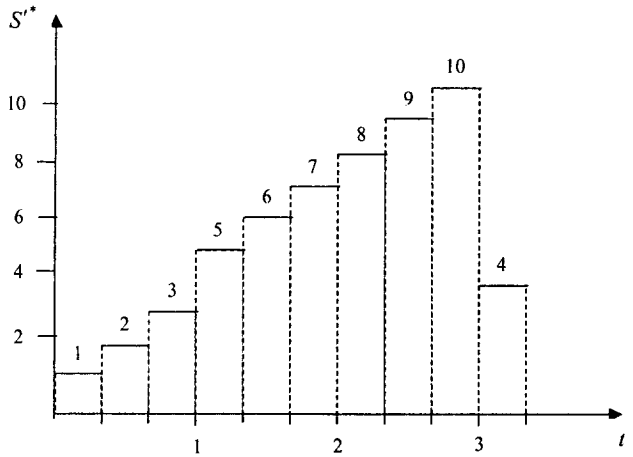


Рис. 3.9 а)

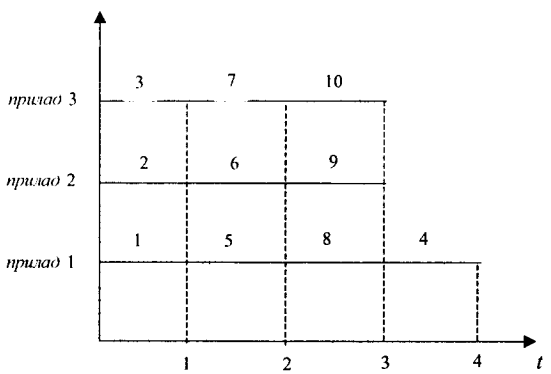


Рис. 3.9 б)

6. Перейдемо до розгляду задач мінімізації максимального штрафу. Нехай $\varphi_i(t) = t \ \ (i = \overline{1,n})$ і дозволені переривання в обслуговуванні кожної вимоги.

Нехай S – деякий розклад обслуговування вимог. Позначимо через τ_{iL} сумарну довжину часових інтервалів, під час яких прилад L

обслуговує вимогу i. Має місце співвідношення $\sum_{L=1}^{m}(\tau_{iL}/t_{iL})=1 \ \ (i=\overline{1,n})$. Позначимо $T=\max_{i\in N}\bar{t}_i(S)$. Очевидно, значення T і τ_{iL} є допустимим планом наступної задачі лінійного програмування:

$$T \to \min, \tag{11}$$

$$\sum_{L=1}^{m}\frac{\tau_{iL}}{t_{iL}}=1, \ \ i=\overline{1,n}, \tag{12}$$

$$\sum_{L=1}^{m}\tau_{iL}\le T, \quad i=\overline{1,n}, \tag{13}$$

$$\sum_{i=1}^{n}\tau_{iL}\le T, \ L=\overline{1,m}, \tag{14}$$

$$\tau_{iL}\ge 0, \ i=\overline{1,n}, \ L=\overline{1,m}. \tag{15}$$

З іншого боку, якщо T і τ_{iL}, де $i=\overline{1,n}$, $L=\overline{1,m}$ – розв'язки задачі (11)–(15) й існує розклад S, при якому сумарна довжина інтервалів обслуговування вимоги i приладом L дорівнює τ_{iL}, а $\max_{i\in N}\bar{t}_i(S)=T$, то розклад S є шуканим оптимальним розкладом. Покажемо, що розклад S існує та вкажемо спосіб його побудови.

Нехай T і $\tau=\|\tau_{iL}\|$ – розв'язок задачі (11)–(15). Покладемо $\tilde{T}=T$. Очевидно маємо

$$\tilde{T}=\max\left\{\max_{1\le i\le n}\sum_{L=1}^{m}\tau_{iL}, \ \max_{1\le L\le m}\sum_{i=1}^{n}\tau_{iL}\right\}. \tag{16}$$

Назвемо рядок i (стовпець L) матриці τ щільним, якщо $\sum_{L=1}^{m}\tau_{iL}=\tilde{T}$ (відповідно $\sum_{i=1}^{n}\tau_{iL}=\tilde{T}$). Нехай $V(\tau)$ – підмножина додатних елементів матриці τ, яка містить по одному елементу з кожного щільного рядка і щільного стовпця та не більше одного елемента із кожних, що залишилися рядків і стовпців.

Нехай δ – найбільше число, яке задовольняє наступні умови:

а) $\delta\le\tau_{iL}$, якщо $\tau_{iL}\in V(\tau)$ і τ_{iL} – елемент щільного рядка або стовпця;

б) $\delta\le\tau_{iL}+\tilde{T}-\sum_{L=1}^{m}\tau_{iL}$, якщо $\tau_{iL}\in V(\tau)$, але рядок i не є щільним;

в) $\delta\le\tau_{iL}+\tilde{T}-\sum_{i=1}^{n}\tau_{iL}$, якщо $\tau_{iL}\in V(\tau)$, але стовпець L не є щільним;

г) $\delta \le \widetilde{T} - \sum_{L=1}^{m} \tau_{iL}$, якщо $V(\tau)$ не містить елементів рядка i;

д) $\delta \le \widetilde{T} - \sum_{i=1}^{n} \tau_{iL}$, якщо $V(\tau)$ не містить елементів стовпця L.

Наприклад, нехай $n=4$, $m=3$, $T=11$ і матриця τ має вигляд:

$$\tau = \begin{Vmatrix} 3 & \mathbf{4} & 4 & 11 \\ \mathbf{4} & 0 & 0 & 4 \\ 0 & 6 & 0 & 6 \\ 4 & 0 & \mathbf{6} & 10 \\ 11 & 10 & 10 & \end{Vmatrix}$$

Тут напроти кожного рядка (стовпця) записана сума елементів даного рядка (стовпця). Перший рядок і перший стовпець є щільними. Виберемо в якості $V(\tau)$ множину елементів матриці τ (виділених жирним шрифтом). Отримаємо:

$$\delta \le \tau_{12} = 4, \quad \delta \le \tau_{21} = 4,$$

$$\delta \le \tau_{21} + \widetilde{T} - \sum_{L=1}^{3} \tau_{2L} = 4 + 11 - 4 = 11,$$

$$\delta \le \tau_{34} + \widetilde{T} - \sum_{L=1}^{3} \tau_{3L} = 6 + 11 - 10 = 7,$$

$$\delta \le \tau_{12} + \widetilde{T} - \sum_{L=1}^{4} \tau_{i2} = 4 + 11 - 10 = 5,$$

$$\delta \le \tau_{34} + \widetilde{T} - \sum_{i=1}^{4} \tau_{i4} = 6 + 11 - 10 = 7,$$

$$\delta \le \widetilde{T} - \sum_{L=1}^{3} \tau_{3L} = 11 - 6 = 5.$$

Таким чином, у розглядуваному прикладі $\delta = 4$.

Перейдемо безпосередньо до побудови шуканого розкладу S. Розклад S на інтервалі $(\eta, \eta+\delta]$, де $\eta = 0$, будується таким чином. Покладемо $S_L(t) = \tau_{iL}$, для кожного елемента $\tau_{iL} \in V(\tau)$ на інтервалі $(\eta, \eta + \min\{\tau_{iL}, \delta\}]$, і якщо $\tau_{iL} < \delta$, покладемо $S_L(t) = 0$ на інтервалі $(\eta + \tau_{iL}, \eta + \delta)$. Якщо в множині $V(\tau)$ немає елементів стовпця L, то покладемо $S_L(t) = 0$ на всьому інтервалі $(\eta, \eta + \delta]$.

Покладемо $\tau'_{iL} = \max\{0, \tau_{iL} - \delta\}$, якщо $\tau_{iL} \in V(\tau)$, і $\tau'_{iL} = \tau_{iL}$ в протилежному випадку. Нехай $T' = \widetilde{T} - \delta$. В результаті отримаємо матрицю $\tau' = \|\tau'_{iL}\|$ і значення T', для яких виконується співвідношення

$$T' = \max\left\{\max_{1\le i\le n}\sum_{L=1}^{m}\tau'_{iL},\ \max_{1\le L\le m}\sum_{i=1}^{n}\tau'_{iL}\right\}.$$

Зазначимо, що в матриці τ' хоча б на один додатний елемент менше, ніж в матриці τ, або ж на однин щільний рядок або стовпець (по відношенню до T') більше, ніж в τ.

Позначимо τ' через τ і T' через $\tilde{T}$. Покладемо η рівне $\eta+\delta$. Знайдемо нові $V(\tau)$ і δ. Описаним вище способом будуємо розклад S на інтервалі $(\eta, \eta+\delta]$ і т.д. Загальне число кроків не перевищує величини $k+m+n$, де k – число додатних елементів у початковій матриці τ.

Для розглядуваного прикладу отримуємо послідовно

$$\tau = \begin{array}{|ccc||c|}\hline \mathbf{3} & 0 & 4 & 7\\ 0 & 0 & 0 & 0\\ 0 & \mathbf{6} & 0 & 6\\ 4 & 0 & \mathbf{2} & 6\\ \hline\hline 7 & 6 & 6 & \end{array} \qquad \tilde{T}=7,\ \delta=3,$$

$$\tau = \begin{array}{|ccc||c|}\hline 0 & 0 & \mathbf{4} & 4\\ 0 & 0 & 0 & 0\\ 0 & \mathbf{3} & 0 & 3\\ \mathbf{4} & 0 & 0 & 4\\ \hline\hline 4 & 3 & 4 & \end{array} \qquad \tilde{T}=4,\ \delta=4.$$

Отриманий в результаті розклад S представлено на рисунку 3.10.

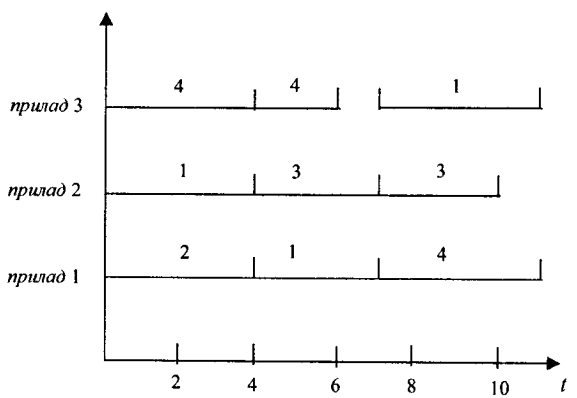


Рис. 3.10.

В описаному процесі побудови розкладу S на кожному кроці знаходиться множина $V(\tau)$.

Для побудови множини $V(\tau)$ досить побудувати матрицю $u = \|u_{ij}\|$

$$u = \begin{Vmatrix} \tau & \beta \\ \gamma & \tau^t \end{Vmatrix},$$

порядку $n+m$, де t – знак транспонування, $\beta = \|\beta_{ij}\|$ і $\gamma = \|\gamma_{ij}\|$ – квадратні матриці з невід'ємними елементами порядку n і m відповідно. Матриці β і γ визначені таким чином, щоб сума елементів кожного рядка і кожного стовпця матриці u дорівнювала $\tilde{T}$.

Далі потрібно розв'язати задачу про призначення

$$\sum_{i=1}^{n+m}\sum_{j=1}^{n+m} a_{ij}x_{ij} \to \min,$$

$$\sum_{j=1}^{n+m} x_{ij} = 1, \quad i = \overline{1, n+m},$$

$$\sum_{i=1}^{n+m} x_{ij} = 1, \quad j = \overline{1, n+m},$$

де $a_{ij} = W$, $j = \overline{m+1, n+m}$, якщо $\sum_{j=1}^{n}\beta_{ij} = 0$, $j = \overline{1,n}$; $a_{ij} = W$, $i = \overline{n+1, n+m}$, якщо $\sum_{i=1}^{m}\gamma_{ij} = 0$, $j = \overline{1,m}$; $a_{ij} = u_{ij}$ в інших випадках; W – досить велике число.

7. Нехай $\varphi_i(t) = t - d_i$, $i = \overline{1,n}$ і дозволені переривання в обслуговуванні вимог.

В цьому випадку задача побудови розкладу S^*, якому відповідає найменше значення функціоналу (2), являє собою задачу побудови розкладу з найменшим максимальним часовим зміщенням $L_{\max}(S) = \max_{i \in N} L_i(S)$, де $L_i(S) = \bar{t}_i(S) - d_i$ – часове зміщення в обслуговуванні вимоги i, d_i – заданий директивний строк, $i = \overline{1,n}$.

Пронумеруємо вимоги по неспаданню директивних строків: $d_1 \le d_2 \le ... \le d_n$. Нехай S – деякий розклад обслуговування вимог.

Позначимо через $\tau_{iL}^{(k)}$ сумарну довжину часових інтервалів, під час яких прилад L обслуговує вимогу i в інтервалі $(d_{k-1} + L_{\max}(S),\ d_k + L_{\max}(S)]$ при розкладі S. Тут $k = 1, 2, ..., n$, $d_0 = -L_{\max}(S)$.

Неважко бачити, що значення $L_{\max}(S)$ і $\tau_{iL}^{(k)}$ $(k=\overline{1,n},\ i=\overline{1,n},\ L=\overline{1,m})$ є допустимим планом наступної задачі лінійного програмування:

$$L_{\max}(S)\to\min, \tag{17}$$

$$\sum_{L=1}^{m}\sum_{k=1}^{i}\frac{\tau_{iL}^{(k)}}{t_{iL}}=1,\quad i=\overline{1,n}, \tag{18}$$

$$\sum_{L=1}^{m}\tau_{iL}^{(1)}\le d_1+L_{\max}(S),\quad i=\overline{1,n}, \tag{19}$$

$$\sum_{L=1}^{m}\tau_{iL}^{(k)}\le d_k-d_{k-1},\quad i=\overline{k,n},\quad k=\overline{2,n}, \tag{20}$$

$$\sum_{i=1}^{n}\tau_{iL}^{(1)}\le d_1+L_{\max}(S),\quad L=\overline{1,m}, \tag{21}$$

$$\sum_{i=1}^{n}\tau_{iL}^{(k)}\le d_k-d_{k-1},\quad L=\overline{1,m},\quad k=\overline{2,n}, \tag{22}$$

$$\tau_{iL}^{(k)}\ge 0,\ (k=\overline{1,n},\quad i=\overline{1,n},\quad L=\overline{1,m}). \tag{23}$$

З іншого боку, якщо ми маємо розв'язок задачі (17)–(23), то можна побудувати розклад, якому відповідає найменше значення максимального часового зміщення. Для цього досить скористатися процедурами, аналогічними описаним в попередньому пункті.

Нефіксовані маршрути. Загальний час обслуговування. Два прилади

Розглянемо задачу побудови оптимального за швидкодією розкладу обслуговування n вимог m приладами у припущенні, що всі вимоги поступають в чергу на обслуговування в момент часу $r=0$, кожна вимога може обслуговуватися приладами в довільному порядку, переривання процесу обслуговування кожної вимоги кожним приладом заборонені.

Нехай система містить два прилади A та B і відомі тривалості a_i та b_i обслуговування вимог $i\in N=\{1,\dots,n\}$ приладами A та B відповідно.

Тоді із (24)

$$\bar{t}_{\max}(S)\ge\max\left\{\max_{i\in N}t_i(M),\ \max_{L\in M}t_L(N)\right\} \tag{24}$$

випливає, що для довільного розкладу справедливе співвідношення (25):

$$\bar{t}_{\max}(S)\ge\max\left\{a(N),\ b(N),\ \max_{i\in N}\{a_i+b_i\}\right\}, \tag{25}$$

де $a(N)=\sum_{i\in N}a_i$, $b(N)=\sum_{i\in N}b_i$.

Існує такий розклад S^*, що у співвідношенні (25) при $S = S^*$ має місце рівність. Опишемо спосіб побудови такого розкладу. Розклад S^* будемо шукати в класі таких розкладів, що для кожного приладу існує не більше двох часових інтервалів, в кожному з яких цей прилад функціонує без простоїв, обслуговуючи вимоги неперервно одну за одною. Розклад такого класу однозначно визначають заданням двох наборів виду $(L; \pi_1; \pi_2; t_0; w_L)$, де L – назва приладу, t_0 – момент початку неперервного обслуговування вимог множини $\{\pi_1\}$ в послідовності π_1, w_L – тривалість простою приладу L після завершення обслуговування вимог множини $\{\pi_1\}$ і до початку неперервного обслуговування вимог множини $\{\pi_2\}$ в послідовності π_2.

Множину N вимог розіб'ємо на дві підмножини $N = \{i \in N \mid a_i \le b_i\}$ та $N_2 = N \setminus N_1$. Припустимо, що $N_1 \neq \varnothing$ і $N_2 \neq \varnothing$. Серед вимог множини N_1 вибираємо таку вимогу k, що $b_k \ge \max\{a_i \mid i \in N_1\}$, а серед вимог множини N_2 вибираємо вимогу r, для якої $a_r \ge \max\{b_i \mid i \in N_2\}$.

Розглянемо пару наборів

$$(A; \{k, \pi_{N_1 \setminus k}\}; \{\pi_{N_2 \setminus r}; r\}; 0; w_A),$$
$$(B; \{k, \pi_{N_1 \setminus k}\}; \{\pi_{N_2 \setminus r}; r\}; a_k; w_B),$$

де в якості w_A, w_B вибрано довільні невід'ємні числа, що задовольняють умові

$$a(N) - a_k + w_A = b(N) - b_r + w_B. \qquad (26)$$

Ця пара наборів визначає розклад, оскільки вимоги обслуговуються обома приладами в одній і тій же послідовності, причому до моменту початку обслуговування кожної вимоги приладом B вона вже обслужена приладом A.

Опишемо процес перетворення цього розкладу в шуканий оптимальний розклад.

1) Припустимо, що $a(N) - a_k \ge b(N) - b_r$. Тоді із (26) слідує, що $w_A \le w_B$. Введемо в розгляд пару наборів

$$(A; \{k, \pi_{N_1 \setminus k}\}; \{\pi_{N_2 \setminus r}, r\}; 0; 0),$$
$$(B; \{r, k, \pi_{N_1 \setminus k}\}; \pi_{N_2 \setminus r}; \max\{a(N) - b(N), 0\}; 0). \qquad (27)$$

(Тут позначення r не співпадає із позначенням $r = 0$ для початку виконання вимог).

а) Якщо має місце нерівність

$$\max\{a(N) - b(N), 0\} + b_r \le a(N) - a_r,$$

то (27) визначає розклад із загальним часом обслуговування, рівним $\max\{a(N), b(N)\}$.

б) Нехай виконується нерівність

$$\max\{a(N) - b(N), 0\} + b_r > a(N) - a_r. \qquad (28)$$

При цьому, очевидно, в деякий момент часу вимога r обслуговується обома приладами одночасно, тобто (27) не визначає розклад.

Замінимо (27) парою наборів

$$\begin{aligned}&\left(A;\left(k,\pi_{N_1\setminus k}\right);\left(\pi_{N_2\setminus r},r\right);a_r+b_r-a(N);0\right),\\&\left(B;\left(r,k,\pi_{N_1\setminus k}\right);\pi_{N_2\setminus r};0;0\right).\end{aligned}\qquad(29)$$

Якщо $a(N)<b(N)$, то нерівність (28) рівносильна нерівності $a_r+b_r>a(N)$ і (9) визначає розклад із загальним часом обслуговування, рівним $\max\{a_r+b_r,b(N)\}$.

Нехай $a(N)\geq b(N)$. Тоді нерівність (28) рівносильна нерівності $a_r+b_r>b(N)$.

Якщо $a_r+b_r>a(N)$, то (29) визначає розклад із загальним часом обслуговування, рівним a_r+b_r.

Якщо $a_r+b_r\leq a(N)$, то неважко переконатися, що пара наборів

$$\begin{aligned}&\left(A;\left(k,\pi_{N_1\setminus k}\right);\left(\pi_{N_2\setminus r},r\right);0;0\right),\\&\left(B;\left(r,k,\pi_{N_1\setminus k}\right);\pi_{N_2\setminus r};a(N)-a_r-b_r;0\right)\end{aligned}$$

визначає розклад із загальним часом обслуговування, рівним $a(N)$.

2) Якщо $a(N)-a_k<b(N)-b_r$, то $w_A>w_B$. Введемо в розгляд пару наборів

$$\begin{aligned}&\left(A;\pi_{N_1\setminus k};\left(\pi_{N_2\setminus r},r,k\right);0;0\right),\\&\left(B;\left(k,\pi_{N_1\setminus k}\right);\left(\pi_{N_2\setminus r},r\right);0;0\right).\end{aligned}\qquad(30)$$

Якщо $a_k+b_k\leq a(N)$, то (30) визначає розклад із загальним часом обслуговування, рівним $\max\{a(N),b(N)\}$.

Якщо $a_k+b_k>a(N)$, то пара наборів (30) не визначає розклад, оскільки в деякий момент часу вимога k одночасно обслуговується двома приладами.

Замінимо (30) парою наборів

$$\begin{aligned}&\left(A;\pi_{N_1\setminus k};\left(\pi_{N_2\setminus r},r,k\right);a_k+b_k-a(N);0\right),\\&\left(B;\left(k,\pi_{N_1\setminus k}\right);\left(\pi_{N_2\setminus r},r\right);0;0\right).\end{aligned}$$

Неважко переконатися, що ця пара наборів визначає розклад із загальним часом обслуговування, рівним $\max\{a_k+b_k,b(N)\}$.

Таким чином, у всіх розглянутих випадках можна побудувати розклад S^* такий, що у співвідношенні (25) при $S=S^*$ має місце знак рівності.

Можна показати, що й у випадках, коли $N_1=\varnothing$ або $N_2=\varnothing$, також можна побудувати розклад S^*, що володітиме вказаною властивістю.

Якщо $N_1=\varnothing$, то шуканий оптимальний розклад визначається парою наборів виду

$$\left(A; \pi_{N_2 \setminus r}; r; \max\{a_r + b_r - a(N), 0\}; 0\right),$$
$$\left(B; r; \pi_{N_2 \setminus r}; \max\{\min\{a(N) - b(N), a(N) - a_r - b_r\}, 0\}; 0\right).$$

Якщо $N_2 = \varnothing$, – то парою наборів виду

$$\left(A; \pi_{N_1 \setminus k}; k; \max\{a_k + b_k - a(N), 0\}; 0\right),$$
$$\left(B; k; \pi_{N_1 \setminus k}; 0; 0\right).$$

Якщо $N_1 = \varnothing$ і $N_2 = \varnothing$, то із наведеного вище слідує, що шуканий оптимальний розклад визначається наборами

$$\left(A; \{k; \pi_{N_1 \setminus k}\}, \{\pi_{N_2 \setminus r}, r\}; \max\{a_r + b_r - a(N), 0\}; 0\right),$$
$$\left(B; \{r, k, \pi_{N_1 \setminus k}\}, \pi_{N_2 \setminus r}; \max\{\min\{a(N) - b(N), a(N) - a_r - b_r\}, 0\}; 0\right),$$

якщо $a(N) - a_k \geq b(N) - b_r$, і

$$\left(A; \pi_{N_1 \setminus k}; \{\pi_{N_2 \setminus r}, r, k\}; \max\{a_k + b_k - a(N), 0\}; 0\right),$$
$$\left(B; \{k, \pi_{N_1 \setminus k}\}, \{\pi_{N_2 \setminus r}, r\}; 0; 0\right),$$

якщо $a(N) - a_k < b(N) - b_r$.

При $m = 2$ оптимальний по швидкодії розклад може бути побудований в результаті виконання не більше, ніж $O(n)$ операцій. В силу симетрії задачі оптимальний алгоритм може бути використаний і у випадку, коли $m > 2$, але $n = 2$.

Приклад

Є дві різні книжки A і B та 6 читачів, які бажають прочитати ці книжки. Тривалість a_i та b_i (кількість днів, які необхідні кожному читачеві для прочитання кожної із книжок A та B) наведені в таблиці:

	1	2	3	4	5	6
a_i	2	6	2	3	3	6
b_i	4	3	1	5	4	4

Припускається, що кожний читач може одночасно читати тільки одну книжку, і почавши її читати, читає неперервно до тих пір, поки не прочитає всю книжку. Кожна книжка в даний момент часу може знаходитися не більше, ніж у одного читача. Необхідно, щоб всі читачі прочитали всі книжки в найкоротший термін.

Розв'язання.

У розглядуваному прикладі $n = \{1, 2, 3, 4, 5, 6\}$, $N_1 = \{1, 4, 5\}$, $N_2 = \{2, 3, 6\}$, $k \in \{1, 4, 5\}$, $r \in \{2; 6\}$. Для визначеності будемо покладати $k = 4$, $r = 2$. Маємо $a(N) = 22$, $b(N) = 21$, $a(N) - a_k = 19 > b(N) - b_r = 18$.

Відповідна пара наборів, що визначає один із шуканих оптимальних розкладів S^*, має вигляд

$$(A; (4, 1, 5); (3, 6, 2); 0; 0),$$
$$(B; (2, 4, 1, 5); (3, 6); 1; 0).$$

Розклад S^* представлено на рисунку 3.11.

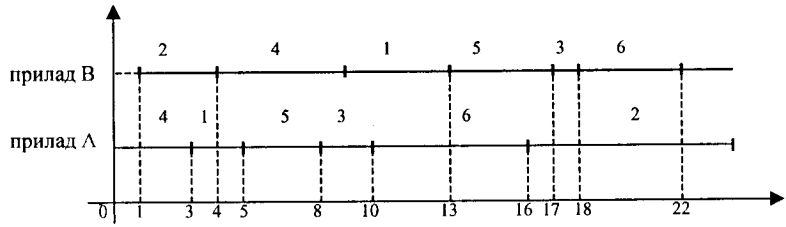


Рис. 3.11.

Опишемо ще два алгоритми побудови оптимальних перестановок.

1. Розбиваємо множину N на дві (не обов'язково непусті) підмножини N_1 та N_2, де N_1 – множина всіх вимог $i \in N$, таких, що $a_i \le b_i$, N_2 – множина всіх інших вимог. Спочатку слід обслужити вимоги множини N_1 в порядку неспадання значень a_i, а потім вимоги множини N_2 в порядку незростання значень b_i.
2. Для побудови оптимальної перестановки можна поставити у відповідність кожній вимозі i вагу
$$\omega(i) = sign(b_i - a_i)(W - \min\{a_i - b_i\})$$
і, впорядкувати вимоги в порядку незростання ваг. Тут W досить велике число (наприклад, $W \ge \max\limits_{1 \le i \le n}\{a_i + b_i\}$).

Приклад

У двох номерах A і B літературного журналу опублікована повість, яку хотіли би прочитати шестеро читачів (початок повісті в номері A, закінчення – в номері B). Кожний із читачів вказав кількість днів, на протязі яких він хотів би мати в своєму розпорядженні кожний з цих номерів.

i	1	2	3	4	5	6
a_i	1	3	4	2	2	1
b_i	3	1	3	2	3	2

Як бібліотекарю обслужити читачів, щоб у найкоротший термін виконати побажання читачів?

Розв'язання.

Згідно з першим алгоритмом маємо $N_1 = \{1, 4, 5, 6\}$, $N_2 = \{2, 3\}$. Оптимальні перестановки: $\pi_1^* = (1, 6, 4, 5, 3, 2)$, $\pi_2^* = (1, 6, 5, 4, 3, 2)$, $\pi_3^* = (6, 1, 4, 5, 3, 2)$, $\pi_4^* = (6, 1, 5, 4, 3, 2)$.

Використовуючи другий алгоритм (покладаючи для визначеності $W = 10$), отримаємо $\omega(1) = 9$; $\omega(2) = -9$; $\omega(3) = -7$; $\omega(4) = 0$; $\omega(5) = 8$; $\omega(6) = 9$. Отже, оптимальними перестановками будуть π_2^*, π_4^*.

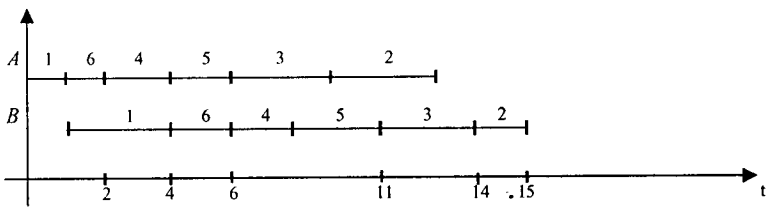


Рис. 3.12

Один із оптимальних розкладів S^*, що відповідає перестановці π_1^*, представлений на рисунку 3.12.

Загальний час $\bar{t}_{\max}(S^*) = 15$.

Описаний вище підхід легко узагальнити на випадок, коли множина вимог N розбита на (не обов'язково непусті) підмножини N_{AB}, N_{BA}, N_A і N_B. Кожна вимога множини N_{AB} повинна бути обслужена спочатку приладом A, потім приладом B. Кожна вимога множини N_{BA} повинна бути обслужена спочатку приладом B, потім приладом A. Нарешті, кожна із вимог множини N_A і N_B повинна бути обслужена тільки приладом A або B відповідно.

В цьому випадку загальний час обслуговування вимог досягає найменшого значення, якщо прилад A обслуговує вимоги у послідовності N_{AB}, N_A і N_{BA}, а прилад B – в послідовності N_{BA}, N_B, N_{AB}. Вимоги множини N_{AB} слід обслуговувати у вказаному вище порядку, вимоги N_{BA} – у порядку, зворотному вказаному (що відповідає зміні ролей обслуговуючих приладів). Вимоги множин N_A і N_B можуть обслуговуватись в довільному порядку. При цьому кожний прилад може обслуговувати вимоги неперервно одну за одною.

Приклад

Десять деталей повинні бути оброблені на двох станках A і B. Послідовності й тривалості обробки кожної деталі наведені в таблиці. Переривання в обробці деталей не допускаються.

Необхідно обробити всі деталі в найкоротші строки.

	N_{AB}				N_{BA}			N_A	N_B	
i	1	2	3	4	5	6	7	8	9	10
a_i	1	4	2	2	1	3	2	1	-	-
b_i	3	2	1	3	2	4	1	-	2	3

Розв'язання.

Впорядковуючи деталі множини N_{AB} і N_{BA} відповідно описаних вище алгоритмів, маємо $\pi_{AB} = (1, 4, 2, 3)$ і $\pi'_{AB} = (5, 6, 7)$. Покладаємо $\pi_{BA} = (7, 6, 5)$, $\pi_A = (8)$, $\pi_B = (9, 10)$.

На станку A деталі слід обробляти в послідовності $(\pi_{AB}, \pi_A, \pi_{BA}) = (1, 4, 2, 3, 8, 7, 6, 5)$, а на станку B — в послідовності $(\pi_{BA}, \pi_B, \pi_{AB}) = (7, 6, 5, 9, 10, 1, 4, 2, 3)$. У розглядуваному випадку обидва станки можуть обробляти деталі неперервно одну за одною, починаючи з моменту часу $r = 0$.

Нефіксовані маршрути. Загальний час обслуговування. Три і більше приладів

Будемо розглядати задачу побудови оптимального за швидкодією розкладу S^* обслуговування множини $N = \{1, 2, ..., n\}$ вимог у системі з нефіксованими маршрутами, яка складається з множини $M = \{1, 2, ..., m\}$ приладів. Тривалості $t_{iL} \geq 0$ обслуговування кожної вимоги $i \in N$ кожним приладом $L \in M$ припускаються заданими. Всі вимоги поступають в систему в момент часу $r = 0$. Переривання процесу обслуговування кожної вимоги кожним приладом заборонені.

Нагадаємо, що при довільному розкладі S значення

$$\bar{t}_{\max}(S) \geq \max\{\max_{i \in N} t_i(M), \max_{L \in M} t_L(N)\}. \qquad (31)$$

Розглянемо систему, що містить $m \geq 3$ приладів, серед яких є один домінуючий прилад.

Будемо казати, що прилад Q домінує прилад H (Q, $H \in M$, $Q \neq H$), якщо $t_{iQ} \geq t_{jH}$ для всіх $i, j \in N$, $j \neq i$.

Аналогічно вимога i домінує вимогу j, $i, j \in N$, $i \neq j$, якщо $t_{iQ} \geq t_{jH}$ для всіх $Q, H \in M$, $Q \neq H$.

Прилад (вимогу) назвемо домінуючим (домінуючою), якщо він (вона) домінує всі інші прилади (вимоги).

Будемо покладати, що домінуючим є прилад 1.

Введемо в розгляд матрицю $\|\lambda_{Li}\|$ розмірності $m \times n$, кожний рядок якої представляє собою деяку перестановку елементів множини N. Якщо $\lambda_{Li} = k$, то це означає, що прилад L обслуговує вимогу k i-ою по порядку.

Задамо матрицю $\|\lambda_{Li}\|$, покладаючи

$$\lambda_{1i} = i,\ i = 1,...,n;$$

$$\lambda_{L1} = \lambda_{L-1,n},\ \lambda_{Li} = \lambda_{L-1,i-1},\ i = 2,...,n,\ L = 2,...,m;$$

$$\bar{t}_{11} = t_{11},\ \bar{t}_{i1} = \bar{t}_{i-1,1} + t_{i1},\ i = \overline{2,n};$$

$$\bar{t}_{\lambda_{iL}L} = \bar{t}_{i1},\ i = \overline{1,n},\ L = \overline{2,m}.$$

Побудована матриця $\|\bar{t}_{iL}\|$ визначає розклад із загальним часом обслуговування, рівним

$$t_1(N) = \max\{\max_{L \in M} t_L(N), \max_{i \in N} t_i(M)\}.$$

Із (31) випливає, що цей розклад є оптимальним.

Нехай система, що містить m приладів, обслуговує n вимог ($n \geq m$) і серед приладів є два прилади, домінуючі всі інші прилади, але не один одного.

Будемо вважати, що вказаними приладами є прилади із номерами 1 і 2. Прилади, відмінні від приладів 1 і 2, назвемо недомінуючими.

За допомогою алгоритму складання оптимального розкладу для системи із двох приладів, описаного нами вище, знайдемо оптимальний за швидкодією розклад обслуговування всіх вимог приладами 1 і 2. Для зручності подальших міркувань будемо покладати, що в побудованому розкладі послідовності обслуговування вимог приладами 1 і 2 визначаються співвідношеннями

$$\lambda_{11} = 1,\ \lambda_{1i} = i,\ \lambda_{21} = n,\ \lambda_{2i} = \lambda_{1,i-1},\ i = \overline{2,n} \qquad (32)$$

(в протилежному випадку вимоги можна відповідним чином перенумерувати).

Як раніше було сказано, загальний час обслуговування вимог приладами 1 і 2, рівний $\max\{t_1(N), t_2(N), t_{n1} + t_{n2}\}$.

1) Припустимо, що

$$t_{n1} + t_{n2} \geq \max\{t_1(N), t_2(N)\}. \qquad (33)$$

Тоді загальний час обслуговування у вихідній задачі буде не меншим $t_n(M)$.

Нехай перші два рядки матриці $\|\lambda_{Li}\|$ заповнені у відповідності з (32). Покладемо

$$\lambda_{L1} = \lambda_{L-1,n-1},\ \lambda_{Li} = \lambda_{L-1,i-1},\ L = \overline{2,m},\ i = \overline{2,n-1}.$$

Із алгоритму, описаного нами раніше, слідує, що

$$\bar{t}_{11} = t_{n1} + t_{n2} - t_1(N) + t_{11};$$
$$\bar{t}_{i1} = \bar{t}_{i-1,1} + t_{i1},\ i = \overline{2,n};$$
$$\bar{t}_{n2} = t_{n2},\ \bar{t}_{12} = \bar{t}_{n2} + t_{12};$$
$$\bar{t}_{i2} = \bar{t}_{i-1,2} + t_{i2},\ i = \overline{2,n}.$$

Покладемо, що

$$\bar{t}_{\lambda_{ii}L} = \bar{t}_{i1},\ L = \overline{2,m},\ i = \overline{1,n-1}.$$

Проведена побудова означає, зо за час обслуговування вимоги n приладом 2 всі інші вимоги будуть завідомо обслужені всіма іншими приладами. Далі, під час обслуговування вимоги n приладом 1, всі інші вимоги обслуговуються приладом 2, а недомінуючі прилади простоюють.

Покладемо

$$\lambda_{Ln} = n,\ L = \overline{3,m};$$
$$\bar{t}_{n3} = \bar{t}_{n1} + t_{n3};$$
$$\bar{t}_{nL} = \bar{t}_{n,L-1} + t_{nL},\ L = \overline{4,m}.$$

В результаті отримаємо розклад із загальним часом обслуговування, рівним $t_n(M)$.

2) Нехай $t_{n1} + t_{n2} < \max\{t_1(N), t_2(N)\}$. Як і вище, покладаємо, що перші два рядки матриці $\|\lambda_{Li}\|$ заповнені згідно (32). Інші рядки цієї матриці заповнимо таким чином, щоб рядок L мав вигляд $(n-L+2, n-L+3, ..., n, 1, 2, ..., n-L+1)$, $L = \overline{3,m}$.

Моменти $\bar{t}_{i1}$, $\bar{t}_{i2}$, $i = \overline{1,n}$ завершення обслуговування вимог приладами 1 і 2 можуть бути знайдені із розкладу, побудованого за алгоритмом для двох приладів. Покладемо

$$\bar{t}_{\lambda_{ii}L} = \bar{t}_{i1},\ L = \overline{3,m},\ i = \overline{1,L-2},$$
$$\bar{t}_{\lambda_{ii}L} = \bar{t}_{i-1,2},\ L = \overline{3,m},\ i = \overline{L,n}.$$

Перше із цих співвідношень означає, що кожна вимога з номером j, $j = \overline{2,n-1}$, перед її обслуговуванням приладом 1 буде обслужена недомінуючим приладом у порядку зростання номерів від $L_j = n - j + 2$ $(L_j \le m)$ до m. Із другого співвідношення слідує, що кожна вимога з номером k, $k = \overline{1,n-2}$, після завершення її обслуговування приладом 2 буде обслужена недомінуючим приладом в порядку зростання номерів від

3 до $L_k = n-k+1$ $(L_k \le m)$. Таким чином, побудовано розклад обслуговування всіх вимог множини $N \setminus \{n\}$.

Для вимоги n покладемо $\bar{t}_{n3} = \bar{t}_{n2} + t_{n3}$ і $\bar{t}_{nL} = \bar{t}_{n,L-1} + t_{n,L-1}$, $i = \overline{4,m}$.

Якщо $t_1(N) \ge t_n(M)$, то побудована матриця $\|\bar{t}_{iL}\|$ визначає розклад із загальним часом обслуговування, рівним $\max\{t_1(N), t_2(N)\}$. В протилежному випадку покладемо $\bar{t}_{n1} = \bar{t}_{nm} + t_{n1}$ і отримаємо розклад із загальним часом обслуговування, рівним $\max\{t_2(N), t_n(N)\}$.

Таким чином, в любому випадку побудовано оптимальний розклад S^*. Трудоємність побудови кожної із матриць $\|\lambda_{Li}\|$ та $\|\bar{t}_{iL}\|$ не перевищує $O(nm)$ операцій.

Приклад

Побудувати оптимальний за швидкодією розклад обслуговування п'яти вимог чотирма приладами. Тривалості t_{iL} наведені в таблиці:

i \ L	1	2	3	4
1	7	12	1	3
2	7	8	5	2
3	10	6	4	7
4	8	7	3	3
5	13	30	5	2

Розв'язання.

Неважко переконатися, що прилади 1 і 2 домінують інші прилади, але не один одного.

Оскільки нерівність (33) не виконується ($t_{51} + t_{52} = 43$, $t_1(N) = 45$, $t_2(N) = 63$), то матриці $\|\lambda_{Li}\|$ і $\|\bar{t}_{iL}\|$ будуємо так, як це описано в 2).

Побудуємо оптимальний розклад обслуговування всіх вимог двома приладами 1 і 2.

i \ L	1	2
1	7	42
2	14	50
3	24	56
4	32	63
5	45	30

В розглядуваному випадку вихідна нумерація вимог така, що виконується (32) і матриця $\|\lambda_{Li}\|$ має вигляд

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 2 & 3 & 4 \\ 4 & 5 & 1 & 2 & 3 \\ 3 & 4 & 5 & 1 & 2 \end{pmatrix}$$

Враховуючи, що $t_1(N) = 45 < t_5(M) = 50$, отримуємо матрицю

$$\|\bar{t}_{iL}\| = \begin{pmatrix} 7 & 42 & 50 & 56 \\ 14 & 50 & 56 & 63 \\ 24 & 56 & 63 & 7 \\ 32 & 63 & 7 & 14 \\ 50 & 30 & 35 & 37 \end{pmatrix},$$

яка визначає оптимальний розклад S^*. Загальний час обслуговування дорівнює $\bar{t}_{\max}(S^*) = t_2(N) = 63$.

Нефіксовані маршрути. Загальний час обслуговування. Переривання

Розглянемо задачу побудови оптимального за швидкодією розкладу функціонування системи з нефіксованими маршрутами при умові, що в процесі обслуговування кожної вимоги кожним приладом допускаються переривання.

У систему, що містить m послідовних приладів, в момент часу $r = 0$ поступає множина $N = \{1, 2, ..., n\}$ вимог. Відомо тривалості $t_{iL} \geq 0$ обслуговування кожної вимоги $i \in N$ кожним приладом $L \in M = \{1, 2, ..., m\}$. Порядок проходження приладів кожною вимогою може бути довільним і не обов'язково однаковим для різних вимог. Припускається, що кожний прилад одночасно обслуговує не більше одної вимоги, а кожна вимога одночасно обслуговується не більше, ніж одним приладом.

У процесі обслуговування вимог допускаються переривання: кожна вимога i може обслуговуватися кожним приладом L за декілька "заходів" при збереженні умови, що сумарна тривалість цього обслуговування дорівнює t_{iL} (якщо $t_{iL} > 0$). Процес обслуговування вимоги i приладом L може бути перерваний в любий момент часу. Після переривання прилад L може обслуговувати будь-яку іншу вимогу, а вимога i – обслуговуватися довільним іншим приладом. Процес обслуговування вимоги i приладом L

може бути відновлений в любий момент часу. Число переривань може бути довільним, але скінченим.

Якщо $\bar{t}_{iL}$ – момент (повного) завершення обслуговування вимоги i приладом L при деякому розкладі S і $\bar{t}_{\max}(S) = \max\{\bar{t}_{iL} \mid i \in N, L \in M\}$, то, як і раніше,

$$\bar{t}_{\max}(S) \geq T,\ T = \max\{\max_{i \in N} t_i(M), \max_{L \in M} t_L(N)\}.$$

1) Нехай всі t_{iL} – цілі числа. Покажемо, що *існує розклад* S^* *такий, що* $\bar{t}_{\max}(S^*) = T$ *і переривання процесу обслуговування мають місце тільки в цілочисельні моменти часу.*

Побудуємо дводольний неорієнтований мультиграф $\Gamma = (N, M; U)$, де $N = \{1, 2, ..., n\}$ і $M = \{1, 2, ..., m\}$ – множини вершин, U – множина ребер, кожні дві вершини $i \in N$ та $L \in M$ з'єднані t_{iL} ребрами (якщо $t_{iL} > 0$).

Під реберним розмалюванням мультиграфа будемо розуміти таке співставлення кольорів (чисел 1, 2, ...) ребрам, при якому ніякі два ребра, інцидентні одній і тій же вершині, не замальовані в однаковий колір (їм не приписано одне й те ж число).

Відомо, що мінімальне число кольорів, необхідних для розмальовки ребер дводольного графа (мультиграфа), дорівнює максимальній степені вершини. Оскільки за побудовою максимальна степінь вершини мультиграфа Γ дорівнює T, то він може бути розмальований в T кольорів (тобто його ребрам можуть бути співставленні числа $1, 2, \ldots, T$).

Інтерпретуючи кольори – числа $1, 2, ..., T$ – як номери часових інтервалів $(0, 1]$, $(1, 2]$, ..., $(T-1, T]$, а замалювання ребра $(i, L) \in U$ в колір τ – як обслуговування вимоги i приладом L на часовому інтервалі $(\tau - 1, \tau]$, приходимо до висновку, що шуканий розклад S^* існує, і для його побудови досить розв'язати задачу розмалювання мультиграфа Γ в T кольорів.

Побудований таким чином розклад S^* може виявитися неприйнятним із-за невиправдано великої кількості переривань. Для побудови оптимального розкладу з меншою кількістю переривань може бути використаний інший підхід.

Введемо поняття *узагальненого реберного розмалювання мультиграфа.* Це розмалювання відрізняється від звичайного тільки тим, що, по-перше, ребра, інцидентні одній і тій же парі вершин, можуть змальовуватися в однаковий колір і, по-друге, якщо в колір τ замальовано k_τ ребер виду (i_1, L_1), то ребра виду (i_2, L_2) $(i_1 \neq i_2, L_1 \neq L_2)$ або не замальовуються в колір τ, або замальовується рівно k_τ з них. Число k_τ

будемо називати кратністю кольору τ. При $k_\tau \le 1$ отримуємо звичайне реберне розмалювання мультиграфа.

Узагальнене реберне розмалювання мультиграфа Γ називається *оптимальним*, якщо сума кратностей k_τ всіх використаних при розмалюванні кольорів τ дорівнює T.

Якщо відоме оптимальне узагальнення розмалювання ребер мультиграфа Γ, то відповідний оптимальний (за швидкодією) розклад S^* може бути побудований наступним чином. Пронумеруємо використані при розмалюванні кольори числами $1, 2, ..., T_1$ (очевидно, $T_1 \le T$). Нехай k_τ, $\tau = 1, 2, ..., T_1$ – відповідні кратності кольорів. Розіб'ємо інтервал планування $(0, T]$ на T підінтервалів $(0, k_1]$, $(k_1, k_1 + k_2]$, ..., $\left(\sum_{\tau=1}^{T_1-1} k_\tau, \sum_{\tau=1}^{T_1} k_\tau\right]$ і пронумеруємо їх числами $1, 2, ..., T_1$. Якщо хоча б одне із ребер виду (i, L) замальоване в колір τ, $1 \le \tau \le T_1$ (а отже, в колір τ розмальовано рівно k_τ ребер виду (i, L)), то на часовому інтервалі з номером τ вимога i неперервно обслуговується приладом L.

2) Нехай t_{iL} – довільні невід'ємні числа. Покажемо, що і в цьому випадку існує розклад S^* такий, що $\bar{t}_{\max}(S^*) = T$, і покажемо спосіб його побудови.

Побудуємо матрицю $A = \|a_{pq}\|$ розмірності $(n+m)\times(n+m)$ виду

$$A = \begin{pmatrix} \|t_{iL}\| & D_n \\ D_m & B \end{pmatrix},$$

де $\|t_{iL}\|$ – матриця розмірності $n \times m$ тривалостей обслуговування вимог, D_n і D_m – діагональні матриці розмірностей $n \times n$ та $m \times m$ відповідно, B – матриця розмірності $m \times n$ такі, що

$$\sum_{q=1}^{n+m} a_{pq} = T,\ p = \overline{1, n+m};\ \sum_{p=1}^{n+m} a_{pq} = T,\ q = \overline{1, n+m}.$$

В якості матриці B може бути вибрана, наприклад, матриця, транспонована до $\|t_{iL}\|$.

Побудована таким чином матриця A *кратна двічі стохастичній матриці* (тобто суми її елементів, розміщених в кожному рядку і кожному стовпцю, рівні) і, отже, допускається розбиття виду

$$A = \sum_{k=1}^{l} \delta_k \Pi_k \qquad (34)$$

де $l \le (n+m-1)^2+1$, $\sum_{k=1}^{l}\delta_k = T$, $\delta_k \ge 0$, $k=\overline{1,l}$, а $\Pi_k = \left\|\theta_{pq}^k\right\|$ – *перестановочні матриці* (матриці розмірності $(n+m)\times(n+m)$ з елементами 0 і 1, в кожному рядку і кожному стовпці яких міститься рівно одна одиниця).

Якщо знайдено розбиття (34), то відповідний оптимальний (за швидкодією) розклад S^* може бути побудований наступним чином.

Розіб'ємо інтервал планування $(0,T]$ на l підінтервалів $(0,\delta_1]$, $(\delta_1,\delta_1+\delta_2]$, ..., $\left(\sum_{k=1}^{l-1}\delta_k, \sum_{k=1}^{l}\delta_k\right]$, пронумерувавши їх числами від 1 до l. Якщо при деякому k, $1\le k\le l$, елемент $\theta_{pq}^k = 1$, $1\le p\le n$, $1\le q\le m$, то будемо вважати, що вимога $i=p$ неперервно обслуговується приладом $L=q$ на часовому інтервалі з номером k. Неважко бачити, що при цьому виконуються всі вимоги, що пред'являються до розкладу, і $\bar{t}_{\max}(S^*) = T$.

Відмітимо, що *якщо всі* t_{iL} *– цілі числа, то в* (34) *всі* δ_k *– цілі числа і переривання в розкладі* S^* *мають місце тільки в цілочисельні моменти часу.*

Для знаходження (34) можна скористатися наступним очевидним прийомом. Виділимо в матриці A множину $R(A)$ її ненульових елементів таким чином, щоб у кожному рядку і в кожному стовпці знаходився рівно один елемент цієї множини. Оскільки матриця A кратна двічі стохастичній матриці, то множина $R(A)$ існує.

Нехай δ – найменший із елементів множини $R(A)$, а $\Pi = \left\|\theta_{pq}\right\|$ – перестановочна матриця, в якої $\theta_{pq} = 1$ тоді і тільки тоді, коли $\theta_{pq} \in R(A)$.

Покладемо $\delta_1 = \delta$ і $\Pi_1 = \Pi$. Позначимо отриману в результаті матрицю через A. Ця матриця теж кратна двічі стохастичній матриці. Проводячи відносно неї ті ж судження, отримаємо другий член $\delta_2\Pi_2$ розбиття (34) і т. д. Оскільки на кожному кроці хоча б один із ненульових елементів матриці A обертається в нуль, то через скінчене число кроків (порядку $O(nm)$) буде отримана нульова матриця і процес знаходження (34) буде завершений.

Існують різні способи виділення множини $R(A)$ за заданою матрицею A. Так, множина $R(A)$ може бути виділена в результаті розв'язання задачі про призначення (на вузькі місця) із вартісною матрицею A, або в результаті розв'язання задачі про максимальні паросполучення у відповідному дводольному графі й т.п. В любому

випадку для виділення множини $R(A)$ досить поліноміально обмеженого (відносно n і m) числа операцій.

3.3. Неодночасне поступлення робіт. Переривання. Тривалість настройки, залежна від впорядкування

1. Неодночасне поступлення робіт. Переривання.
2. Тривалість настройки, залежна від впорядкування.

Неодночасне поступлення робіт. Переривання

Раніше ми припускали що всі n робіт поступають в систему одночасно. У цьому випадку оптимальні (відносно регулярних критеріїв) розклади не містять переривань і штучно вводимих простоїв. Якщо роботи поступають неодночасно і $r_i \neq r_j$ (для деяких i та j), то оптимальні розклади належать класу, що включає переривання і штучні простої.

Кажуть, що відбувається *переривання*, якщо почата робота переривається до її закінчення. Якщо розглядаються переривання, то послідовність n індексів уже не визначає розклад для n робіт, так як довільна робота може повторитися в розкладі більше одного разу і слід задати тривалості кожного із інтервалів її виконання. Розрізняють декілька видів переривань в залежності від того, як продовжується виконання перерваної роботи при її відновленні. В одних випадках робота виконується далі з того місця, де вона була перервана; тоді кажуть, що має місце переривання з дообслуговуванням. При цьому тривалість роботи залишається сталою і не залежить від числа переривань. У других випадках розглядається переривання з обслуговуванням заново. Тут виконання роботи починається заново всякий раз, як вона попадає на обслуговування після переривання, тобто вся виконана до переривання частина роботи виявляється безкорисною. В цих умовах тривалість роботи мінімальна, якщо вона не переривається.

По суті, при послідовному поступленні робіт і перериванні з дообслуговуванням має місце та ж ситуація, що і при одночасному поступленні робіт. Справа в тому, що в моменти поступлення робіт можна переглядати всі прийняті раніше рішення, враховуючи тільки тривалість виконуваних, але ще не закінчених робіт. При всякому новому поступленні здійснюється кращий вибір із всіх незавершених робіт, причому це відбувається без усяких втрат.

Очевидним узагальненням алгоритму впорядкування за мінімумом тривалості робіт є впорядкування за мінімумом тривалостей, що

залишились. При такому впорядкуванні мінімізується середня тривалість проходження і зберігаються всі результати, що відносяться до алгоритму SPT у випадку одночасного поступлення робіт.

Аналіз систем з переривранням і обслуговуванням заново дуже складний, і по цьому питанню поки що не опубліковано цікавих результатів. Відмітимо, що тут не можна застосовувати результати, отримані для системи з одночасним поступленням робіт, які поширюються на випадок переривання з дообслуговуванням.

Стратегія простоїв практично використовується завжди при наявності переривань з відновленням обслуговування, коли є попередня інформація про майбутні роботи. Однак загального оптимального алгоритму для цього випадку поки не знайдено.

Тривалість настройки, залежна від впорядкування

Будемо розглядати випадок, коли тривалість настройки машини перед виконанням деякої задачі не можна вважати незалежною від характеру попередньої роботи. Більше того, розкидання тривалості настройки в цих випадках є основним критерієм оцінки розкладу.

Якщо настройка не залежить від впорядкування, то її тривалість можна включити в тривалість роботи p_i. Ми будемо тривалість настройки задавати окремо у вигляді матриці настройок $S = (s_{ij})$, де s_{ij} задають тривалість настройки при переході від роботи i до роботи j.

Наявність настройки змінює величини, які визначають процес обслуговування. Так, наприклад, у раніше розглянутих випадках сумарний час виконання всіх робіт машиною був сталим і не залежав від порядку їх виконання. Але тепер це не так. Тривалості проходження також визначаються інакше:

$$F_{[1]} = s_{[0],[1]} + p_{[1]},$$
$$F_{[2]} = F_{[1]} + s_{[1],[2]} + p_{[2]},$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$F_{[i]} = F_{[i-1]} + s_{[i-1],[i]} + p_{[i]}.$$

Якщо раніше максимальна тривалість проходження була константою і розклад оптимізувався за таким критерієм, як середня тривалість проходження, то тепер оптимізація і за критерієм максимальної тривалості проходження являє собою важку задачу:

$$F_{\max} = F_{[n]} = \sum_{i=1}^{n} s_{[i-1],[i]} + \sum_{i=1}^{n} p_{[i]}.$$

Сума тривалостей всіх робіт в даному випадку стала і може не враховуватися. Тому максимальна тривалість проходження мінімальна, коли мінімальна сумарна тривалість всіх настройок.

Задача мінімізації сумарної тривалості всіх настройок аналогічна так званій *задачі комівояжера.* В цій задачі комівояжер повинен відвідати кожне із n міст тільки один раз і повернутися в початковий пункт, причому його маршрут повинен бути таким, щоб мінімізувати сумарну тривалість пройденого шляху (або сумарний час, або сумарну вартість і т.д.). У нашому випадку кожна робота відповідає місту, а тривалість настройки – відстані між містами . Потрібно організувати маршрут так, щоб сумарна відстань була найменшою. Введемо

$$x_{ij} = \begin{cases} 1, & \text{якщо їде з міста } i \text{ в } j, ((i,j) \text{ належить контуру}), \\ 0, & \text{в протилежному випадку}, ((i,j) \notin \text{контуру}). \end{cases}$$

Математична модель задачі комівояжера:

$$\min \quad F(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij} c_{ij} \quad (i \neq j)$$

при умовах

$$\sum_{i=1}^{n} x_{ij} = 1, \ j = \overline{1,n} \text{ – з кожного міста можна виїхати лише 1 раз,}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \ i = \overline{1,n} \text{ – в кожне місто можна в'їхати лише 1 раз.}$$

Існує додаткове обмеження, яке полягає в тому, що маршрут комівояжера повинен завершитися в початковій точці після одноразового відвідування всіх n міст. Це несуттєве на перший погляд обмеження приводить до того, що одна із найпростіших задач математичного програмування перетворюється в одну із найбільш складних.

Циклом t називається набір впорядкованих пар міст, які утворюють маршрут, що проходить через кожне місто один раз:

$$t = [(i_1, i_2), (i_2, i_3), \ldots, (i_{n-1}, i_n), (i_n, i_1)].$$

Пара (i, j) – комунікація маршруту;

t – це цикл матриці утримок S.

Тоді загальні утримки $Z(t) = \sum_{(i,j) \in t} c_{ij}$.

У циклі міститься тільки один елемент з кожного рядка і кожного стовпчика.

Якщо від деякого елемента деякого i-го рядка або j-го стовпчика відняти мінімальний з цих елементів, то дістанемо якусь нову матрицю, яку називають зведеною, а сам процес – процесом зведення.

Сума цих мінімальних елементів називається звідними константами h^k, де k – порядковий номер зведення.

Позначимо:

$c_{i,j(i)} = \min\limits_{j} c_{ij}$, тоді $c'_{ij} = c_{ij} - c_{i,j(i)}$;

$c^*_{i(j),j} = \min\limits_{i} c_{ij}$, тоді $c''_{ij} = c'_{ij} - c^*_{i(j),j}$ – кінцевий вигляд зведеної матриці, всі елементи якої невід'ємні.

$$h^1 = \sum_{i=1}^{n} c_{i,j(i)} + \sum_{j=1}^{n} c^*_{i(j),j}.$$

h^1 – нижня границя утримок циклу t при старій матриці.

Далі позначимо:

X – вершина, з якої здійснюється галуження;

Y – вершина, яка найбільше належить до перспективної гілки;

$\overline{Y}$ – вершина із забороненою парою міст.

Поставимо у відповідність вершині X суму її звідних констант h^k, яка є нижньою границею множини X і позначимо цю нижню границю через $\omega(X)$.

Алгоритм розв'язання задачі комівояжера.

1. $k = 1$.
2. Здійснюється зведення матриці c (по рядку і стовпцю). Дістанемо матрицю c^k.
3. Обчислимо суму звідних констант h^k, яка буде нижньою границею $\omega(X)$.
4. Вибираємо претендентів для включення в множину Y, тобто ті (i,j), для яких $c^k_{ij} = 0 \quad (i \neq j,\ j = \overline{1,n},\ i = \overline{1,n})$.
5. Для виділення претендентів на включення в Y, підрахуємо наступну величину:
$$\theta(i,j) = \min_{j' \neq j} c_{ij'} + \min_{i' \neq i} c_{i'j}.$$
6. Виберемо $\theta(k,l) = \max \theta(i,j)$. Пара (k,l) включається в множину Y і є забороненою для $\overline{Y}$.
7. Підраховуємо оцінку для множини $\overline{Y}$
$$\omega(\overline{Y}) = \omega(X) + \theta(k,l).$$
8. Оскільки з кожного k-го міста можна виїхати один раз, то викреслюємо k-ий рядок; оскільки з кожного l-ого міста можна виїхати один раз, то викреслюємо l-ий стовпець і ставимо заборону на клітинку (l,k).
9. В кінці кінців отримаємо матрицю 2 на 2 і переходимо до пункту 11. Якщо ні, то переходимо до пункту 10.
10. Перевіряємо чи матриця зведена. Якщо зведена, то рахуємо оцінку $\omega(Y) = \omega(X)$. Якщо не зведена, то робимо зведення, шукаємо h^{k+1}, $\omega(Y) = \omega(X) + h^{k+1}$, $k = k+1$, і переходимо до пункту 4.

11. Перевірка умови оптимальності. Якщо оцінка замкнутого циклу не більша всіх допустимих для подальшого галуження множин (кінцевих на гілках), тоді цей замкнутий маршрут оптимальний.

Якщо існує хоча б одна множина з меншою оцінкою, то цей маршрут запам'ятовують, а процес галуження продовжують, починаючи з множини з меншою оцінкою.

Приклад

Є 6 пунктів, які має відвідати комівояжер один раз, мінімізуючи свій шлях, і повернутися в початкове місто. Відстань між містами задано матрицею:

$$\begin{pmatrix} \infty & 27 & 43 & 16 & 30 & 26 \\ 7 & \infty & 16 & 1 & 30 & 25 \\ 20 & 13 & \infty & 35 & 5 & 0 \\ 21 & 16 & 25 & \infty & 18 & 18 \\ 12 & 46 & 27 & 48 & \infty & 5 \\ 23 & 5 & 5 & 9 & 5 & \infty \end{pmatrix}$$

(рядки – відстань, стовпчики – утримки).

Розв'язання.

1 крок: $k = 1$.

2 крок:

∞	27	43	16	30	26	16
7	∞	16	1	30	25	1
20	13	∞	35	5	0	0
21	16	25	∞	18	18	16
12	46	27	48	∞	5	5
23	5	5	9	5	∞	5
						$\sum = 43$

∞	11	27	0	14	10
6	∞	15	0	29	24
20	13	∞	35	5	0
5	0	9	∞	2	2
7	41	22	43	∞	0
18	0	0	4	0	∞
5	0	0	0	0	0

$\sum = 5$

$$\begin{pmatrix} \infty & 11 & 27 & 0 & 14 & 10 \\ 1 & \infty & 15 & 0 & 29 & 24 \\ 5 & 13 & \infty & 35 & 5 & 0 \\ 0 & 0 & 9 & \infty & 2 & 2 \\ 2 & 41 & 22 & 43 & \infty & 0 \\ 13 & 0 & 0 & 4 & 0 & \infty \end{pmatrix}$$

3 крок:

Сума всіх звідних констант $h^1 = 43 + 5 = 48 = \omega(X)$.

4 крок:

$$c_{14} = 0, \quad c_{42} = 0, \quad c_{56} = 0,$$
$$c_{24} = 0, \quad c_{62} = 0, \quad c_{36} = 0,$$
$$c_{41} = 0, \quad c_{63} = 0. \quad c_{65} = 0.$$

5 крок:

Шукаємо $\theta(i,j) = \min_{j' \neq j} c_{ij'} + \min_{i' \neq i} c_{i'j}$:

$\theta(1,4) = 0 + 10 = 10$, $\theta(4,1) = 0 + 1 = 1$, $\theta(6,2) = 0 + 0 = 0$,
$\theta(2,4) = 0 + 1 = 1$, $\theta(4,2) = 0 + 0 = 0$, $\theta(6,3) = 0 + 9 = 9$,
$\theta(3,6) = 0 + 5 = 5$, $\theta(5,6) = 0 + 2 = 2$, $\theta(6,5) = 0 + 2 = 2$.

6 крок:

$$\theta(k,l) = \max \theta(i,j) = 10 \quad k = 1, \quad l = 4.$$

7 крок:

Підраховуємо оцінку для множини $\overline{Y}$:

$$\omega(\overline{Y}) = \omega(X) + \theta(k,l) = 48 + 10 = 58.$$

8 крок:

Викреслюємо перший рядок і четвертий стовпчик. На перетині викреслених ∞ замість цифри 0 ставимо ∞. Пара (1, 4) вибирається для галуження.

Все спочатку:

	1	2	3	5	6	
2	1	∞	15	29	24	1
3	15	13	∞	5	0	0
4	∞	0	9	2	2	0
5	2	41	22	∞	0	0
6	13	0	0	0	∞	0
						$\sum = 1$

	1	2	3	5	6
2	0	∞	14	28	23
3	15	13	∞	5	0
4	∞	0	9	2	2
5	2	41	22	∞	0
6	13	0	0	0	∞
	0	0	0	0	0

$h^2 = 1$.

$\omega(\overline{Y}) = \omega(X) + \theta(k,l) = 48 + 10 = 58$.

$c(2,1) = 0$, $c(6,2) = 0$,
$c(3,6) = 0$, $c(6,3) = 0$,
$c(4,2) = 0$, $c(6,5) = 0$.
$c(5,6) = 0$,

$\theta(2,1) = 2 + 14 = 16$, $\theta(6,2) = 0 + 0 = 0$,
$\theta(3,6) = 5 + 0 = 5$, $\theta(6,3) = 9 + 0 = 9$,
$\theta(4,2) = 0 + 2 = 2$, $\theta(6,5) = 0 + 2 = 2$.
$\theta(5,6) = 2 + 0 = 2$,

$\theta(k,l) = \max \theta(i,j) = 16$, $k = 2$, $l = 1$.

	2	3	5	6
3	13	∞	5	0
4	0	9	2	2
5	41	22	∞	0
6	0	0	0	∞

$\omega(\overline{Y}) = \omega(X) + \theta(2,1) = 48 + 16 = 64$.

$c_{36} = 0$, $\theta(3,6) = 5 + 0 = 5$,
$c_{42} = 0$, $\theta(4,2) = 2 + 0 = 2$,
$c_{56} = 0$, $\theta(5,6) = 22 + 0 = 22$,
$c_{62} = 0$, $\theta(6,2) = 0 + 0 = 0$,
$c_{63} = 0$, $\theta(6,3) = 0 + 9 = 9$,
$c_{65} = 0$, $\theta(6,5) = 0 + 2 = 2$.

$\theta_{\max} = \theta_{56} = 22 \quad \Rightarrow (5,6)$

	2	3	5	
3	10	∞	5	5
4	0	9	2	0
6	0	0	∞	0
				$\sum = 5$

$h^3 = 5.$

$\omega(Y) = \omega(X) + h^k = 48 + 5 = 53;$

$\omega(\overline{Y}) = \omega(X) + \theta_{56} = 53 + 22 = 75.$

$c_{35} = 0, \quad \theta(3,5) = 8 + 0 = 8,$

$c_{42} = 0, \quad \theta(4,2) = 2 + 0 = 2,$

$c_{62} = 0, \quad \theta(6,2) = 0 + 0 = 0,$

$c_{63} = 0, \quad \theta(6,3) = 0 + 7 = 7.$

	2	3
4	∞	7
6	0	∞

$\theta_{\max} = \theta_{35} = 8 \Rightarrow (3,5).$

Залишається (6,2), (4,3).

$\omega(\overline{Y}) = 53 + 8 = 61.$

(1,4),(2,1),(5,6),(3,5),(6,2),(4,3);

$1 \to 4 \to 3 \to 5 \to 6 \to 2 \to 1.$

Природнім і зазвичай таким, що використовується практично алгоритмом вибору маршруту в задачі комівояжера являється *переміщення до ближнього, ще не відвіданого міста.* Це означає, що для заданого місцезнаходження у відповідній матриці S вибирається мінімальний елемент, який визначає наступне місто маршруту (при цьому, звичайно, не розглядаються елементи, що відповідають пройденим містам). Така процедура нагадує алгоритм SPT при впорядкуванні робіт.

Неважко навести приклади, які показують, що описаний алгоритм не дає оптимального розв'язку. Крім того, легко підібрати приклади, коли цей алгоритм заводить зовсім в другу сторону та дає дуже погані розв'язки. Однак, за виключенням цих спеціально підібраних випадків, застосування

описаного алгоритму в стандартних ситуаціях представляє інтерес. Якщо існує впевненість, що описаний алгоритм не призведе до дуже поганого розв'язку, то втрата якості компенсується легкістю, з якою він отримується. Цікаво відмітити, що, хоч оптимальний розв'язок отримується циклічною перестановкою і початкова точка маршруту може бути вибрана довільно, тим не менше розв'язки, які отримуються за допомогою алгоритму ближнього міста, залежать від вибору початкової точки маршруту.

Наприклад, матриця

$$S = \begin{pmatrix} - & 1 & 7 & 3 & 14 & 2 \\ 3 & - & 6 & 9 & 1 & 24 \\ 6 & 14 & - & 3 & 7 & 3 \\ 2 & 3 & 5 & - & 9 & 1 \\ 15 & 7 & 11 & 2 & - & 4 \\ 20 & 5 & 13 & 4 & 18 & - \end{pmatrix}$$

є матрицею відстаней для задачі комівояжера. Розв'язки цієї задачі, що отримуються за допомогою алгоритму ближчого міста, наведені в таблиці.

Початковий пункт	Черговість проходження	Довжина маршруту
1	1 – 2 – 5 – 4 – 3 – 6	32
2	2 – 5 – 4 – 1 – 6 – 3	34
3	3 – 4 – 1 – 2 – 5 – 6	24
	3 – 6 – 4 – 1 – 2 – 5	22
4	4 – 1 – 2 – 5 – 6 – 3	24
5	5 – 4 – 1 – 2 – 3 – 6	32
6	6 – 4 – 1 – 2 – 5 – 3	22

Вибір в якості початкової точки третього чи шостого міста приводить до одного й того ж маршруту 3 – 6 – 4 – 1 – 2 – 5 з довжиною 22. Ця величина трохи більша оптимальної довжини, рівної 20 (3 – 6 – 2 – 5 – 4 – 1), не дивлячись на те, що у отриманого маршруту та оптимального співпадають тільки три із шести дуг. У частинних випадках, коли розглядаються всі можливі варіанти вибору початкової точки маршруту або цей вибір здійснюється вдало, отримуємо досить добрий в порівнянні з оптимальним розв'язок. Однак, якщо вибір початкової точки зроблено невдало, то частіше всього розв'язок буде далеким від оптимального. В загальному випадку, якщо проглядаються всі можливі варіанти вибору початкової точки маршруту, то об'єм обчислень (число арифметичних операцій) зросте в n разів, але залишиться незначним у порівнянні з алгоритмами, які гарантують точний розв'язок.

3.4. Впорядкування при наявності обмежень на можливі варіанти розкладів

Раніше ми покладали, що можлива довільна послідовність виконання робіт. Зараз розглянемо випадки, коли певні послідовності виконання робіт заборонені з тих чи інших причин.

I. *Складання розкладу при частковому впорядкуванні.*

Розглянемо випадок, коли складається розклад для частково впорядкованих робіт. Це часткове впорядкування може бути обумовлене різними причинами. Наприклад, воно може бути викликане залежністю тривалості настройки від черговості виконання робіт. Якщо це має місце, то роботи розбиваються на групи з приблизно рівною настройкою, після чого складається для груп розклад, мінімізуючий деякі критерії (наприклад, середню тривалість проходження). Черговість всередині груп встановлюється таким чином, щоб мінімізувати загальну тривалість перенастройки машини. Тепер припустимо, що вихідні n робіт розбиті на k груп з числом робіт $n_1, n_2, \ldots, n_k$ в кожній групі. Припустимо також, що склад груп і черговість робіт в групі фіксовані і що перехід до наступної групи можливий тільки після того, як виконані всі роботи попередньої. Потрібно вибрати один із можливих способів впорядкування груп. Введемо наступні позначення:

p_{ij} – тривалість виконання роботи j в i-ій групі;

F_{ij} – тривалість проходження роботи j в групі i;

$p_i' = \sum_{j=1}^{n_i} p_{ij}$ – загальна тривалість всіх робіт i-ої групи;

$F_i' = F_{i,n_i}$ – тривалість проходження i-ої групи, рівна тривалості проходження останньої роботи в ній.

Якщо розклад повинен мінімізувати середню тривалість проходження групи, тобто величину

$$\overline{F}' = \frac{\sum_{i=1}^{k} F_i'}{k},$$

то, очевидно, потрібно розглядати кожну групу як роботу тривалістю p_i' і застосувати алгоритм впорядкування у відповідності з мінімальною тривалістю. Це означає, що групи впорядковуються так, що

$$p'_{[1]} \le p'_{[2]} \le \ldots \le p'_{[k]}.$$

Якщо потрібно мінімізувати зважену середню тривалість проходження групи

$$\overline{F}'_n = \frac{\sum_{i=1}^{k} n_i F'_i}{n},$$

то впорядкування таке, що

$$\frac{p'_{[1]}}{n_{[1]}} \le \frac{p'_{[2]}}{n_{[2]}} \le \ldots \le \frac{p'_{[k]}}{n_{[k]}}.$$

Наведене впорядкування груп мінімізує також середню тривалість проходження вихідних робіт (цей факт не очевидний):

$$\overline{F} = \frac{\sum_{i=1}^{n} F_i}{n} = \frac{\sum_{i=1}^{k}\sum_{j=1}^{n_i} F_{ij}}{n}.$$

Теорема. Нехай в системі $\mathbf{n|1|\overline{F}}$ роботи розбиті на k груп, що не перетинаються, і нехай виконання робіт всередині групи та перехід від групи до групи здійснюється без настройок, а порядок виконання робіт всередині кожної групи фіксований. Тоді середня тривалість проходження робіт мінімальна, якщо порядок виконання груп робіт такий, що відношення сумарної тривалості всіх робіт групи до числа робіт в ній не спадає, тобто

$$\frac{p'_{[1]}}{n_{[1]}} \le \frac{p'_{[2]}}{n_{[2]}} \le \ldots \le \frac{p'_{[k]}}{n_{[k]}}.$$

II. *Складання розкладу при заданому відношенні передування.*

Часткове впорядкування робіт бажано задавати менш жорстко, ніж це було зроблено у попередньому пункті, наприклад, ввести певні відношення між роботами взамін вимоги обслуговування груп без переривань. Таке відношення можна встановити, використовуючи поняття слідування, безпосереднього слідування, а також домінуючого графа.

Припустимо, що потрібно виконати три роботи: a, b, c. Якщо не накладати ніяких обмежень на порядок їх виконання, то можливі наступні шість варіантів упорядкування:

1) abc; 2) acb; 3) bac; 4) bca; 5) cab; 6) cba.

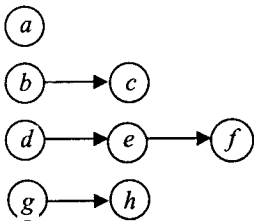
Якщо розглядати b і c як групу в розумінні попереднього пункту, то можливі тільки два варіанти:

1) abc; 2) bca.

Якщо накласти більш слабше обмеження у формі: b передує c, то можливі три варіанти:

1) abc; 2) bac; 3) bca.

Розглянемо тепер більш загальний випадок, коли n робіт розбиті на k ланцюжків. Це означає, що співвідношення слідування визначають для кожної роботи не більше одної попередньої, і не більше одної наступної роботи, наприклад, так:



При наявності таких обмежень впорядкування, яке мінімізує середню тривалість проходження, визначається наступною теоремою:

Теорема. Нехай для системи $\mathbf{n|1|\overline{F}}$ без настройок задані відношення слідування між певними парами робіт так, що для кожної роботи існує не більше одної попередньої й одної наступної; нехай в результаті цього множина робіт розбивається на k ланцюжків, що не перетинаються, всередині кожного з яких задана послідовність виконання робіт, причому допускається переривання ланцюжків. Тоді середня тривалість проходження мінімізується впорядкуванням, що встановлюється таким чином:

1) Для кожної роботи j в i-ому ланцюжку обчислюється

$$x_{ij} = \frac{\sum_{h=1}^{j} p_{ih}}{j}.$$

2) Для кожного ланцюжка i обчислюється

$$y_i(h_i) = \min(x_{i1}, x_{i2}, \ldots, x_{in_i}) = x_{ih_i},$$

тобто h_i – другий індекс мінімальної із величин $x_{i1}, \ldots, x_{in_i}$.

3) Вибирається такий ланцюжок I, що $y_1(h_1) \le y_i(h_i)$ для всіх i та перші h_1 робіт в ланцюжку I формують початок черговості виконання робіт.

4) Обчислюються знову величини y_i без врахування перших h_1 робіт ланцюжка I.

5) Третій і четвертий кроки повторюють до тих пір, поки не будуть впорядковані всі роботи.

3.5. Розклади для систем конвеєрного типу

1. **Перестановочні розклади.**
2. **Мінімізація максимальної тривалості проходження в конвеєрній системі із двох машин.**
3. **Мінімізація середньої тривалості проходження в конвеєрній системі із двох машин.**
4. **Конвеєрна система із трьох машин.**
5. **Впорядкування у великих системах конвеєрного типу.**

Досить часто робота складається з послідовності операцій, кожна із яких виконується відповідною машиною. В цих випадках кажуть, що сукупність машин являється *системою конвеєрного типу*, якщо машини пронумеровані так, що для кожної розглядуваної машини операція K виконується машиною з більшим номером, ніж операція J при $J < K$.

Прикладом такої системи може служити складальна лінія. Взагалі, в якості конвеєрної системи може розглядатися люба сукупність машин, які виконують всі роботи в одному й тому ж порядку. Для такої системи зовсім не обов'язково, щоб кожна робота складалася з операцій, виконуваних на кожній машині, або щоб всі роботи починалися та закінчувалися новими машинами. Суттєво лише те, що всі переміщення робіт, пов'язані із закінченням її на одній машині й початком виконання на другій, повинні відбуватися незмінно в одному напрямку.

Для конвеєрних систем отримано багато результатів, і розклади для них складаються порівняно легше, ніж для систем довільної структури.

Однак простота таких систем відносна і для них існує ще багато нерозв'язаних задач.

Перестановочні розклади

Визначена простота системи з одною машиною була обумовлена тим, що для неї розглядались тільки перестановочні розклади. Такі розклади повністю визначені, якщо задана перестановка індексів робіт.

У загальному випадку складання розкладу наштовхується на значні труднощі, викликані тим, що доводиться розглядати більш широкі класи розкладів, ніж перестановочні. Цим пояснюється і незначна кількість результатів, отриманих у загальному випадку. Конвеєрна система займає проміжкове положення, оскільки в багатьох випадках для неї достатньо розглянути перестановочні розклади, а в більш загальному випадку існує інваріантність черговості по відношенню до першої та останньої машини. Цей факт встановлюється в наступних теоремах 1 і 2.

Теорема 1. Якщо складається розклад для системи $\mathbf{n|m|F}$ і всі роботи доступні одночасно, то оптимальний, з точки зору мінімуму регулярного критерію, розклад належить класу розкладів, у яких однаковий порядок виконання робіт на перших двох машинах.

Доведення.

Якщо в розкладі порядок виконання робіт на перших двох машинах різний, то знайдеться така пара робіт I та J, що на першій машині I безпосередньо передує J, а на другій I слідує за J (причому між ними можливі проміжкові роботи).

Машина 1	…	I	J	…			
Машина 2	…			J	…	I	…

Очевидно, що порядок виконання цих двох робіт першою машиною можна змінити на зворотній, не викликаючи додаткового часового зсуву в початок виконання Будь-якої роботи другою машиною і, отже, всіх наступних. Така перестановка не призводить до зростання часу закінчення любої роботи і не погіршує довільний регулярний критерій. Може статися так, що в результаті такої перестановки початки виконання деяких робіт другою машиною змістяться вліво на часовій осі і, отже, критерій покращиться. Але відсутність погіршення критерію уже достатня для доведення теореми.

Теорему доведено.

Результат теореми не виключає можливість існування задовільних розкладів із різним порядком виконання двох робіт на перших двох машинах, так як для всякого другого розкладу в розглядуваному класі існує еквівалентний йому і, можливо, кращий.

Для конкретних критеріїв можна отримати більш строгий результат.

Теорема 2. Якщо складається розклад для системи $\mathbf{n|m|F|F_{max}}$ і всі роботи доступні одночасно, то оптимум шукається серед розкладів, у яких зберігається порядок виконання робіт на двох перших (1 і 2) та двох останніх ($m-1$ і m) машинах.

Доведення.

Перша частина теореми, що стосується однакового порядку виконання робіт першими машинами, є просто наслідком теореми 1, так як максимальна тривалість проходження являється регулярним критерієм.

Тепер припустимо, що порядок виконання робіт різний для останніх двох машин. Тоді знайдеться така пара робіт I та J, що для машини m I

безпосередньо слідує за J, а для машини $m-1$ робота I передує J (причому між I та J можливі другі роботи).

Машина 1	…		I	…	J	…		
Машина 2	…					J	I	…

Очевидно, що порядок виконання робіт I та J машиною m можна змінити на обернений, не збільшуючи максимальної тривалості проходження цих робіт і не змінюючи тривалості проходження других. Крім того, ймовірно, що зміна порядку виконання робіт I та J машиною m може призвести до деякого зменшення максимальної тривалості проходження, що і доводить теорему.

Теорему доведено.

Наступний простий приклад показує, що теорему 2 не можна узагальнити на випадок довільного регулярного критерію. Припустимо, що складається розклад для двох робіт на трьох машинах, причому складений розклад повинен мінімізувати середню тривалість проходження.

Тривалості робіт задаються в таблиці:

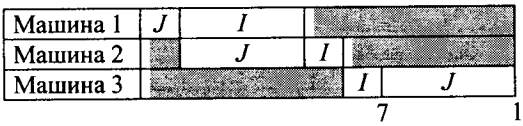
Машини	1	2	3
Робота I	4	1	1
Робота J	1	4	4

Існують два розклади з однаковим порядком робіт на машинах 2 і 3 та однаковими середніми тривалостями проходження, рівними 9,5.

Машина 1	I			J									
Машина 2				I			J						
Машина 3							I			J			
							6						13

Машина 1	J		I										
Машина 2		J				I							
Машина 3							J				I		
											9		10

Поряд з цим існує розклад з різним порядком виконання робіт на машинах 2 і 3 та середньою тривалістю проходження, рівною 9.



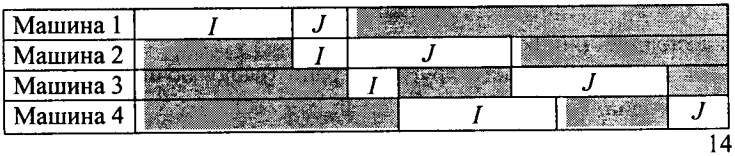
Таким чином отримуємо, що:

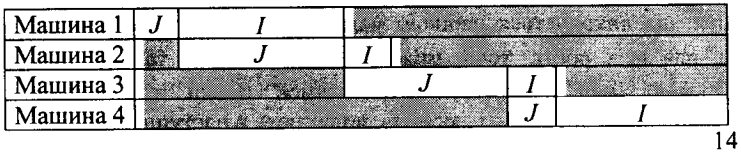
— для довільного регулярного критерію (наприклад, середня тривалість проходження) конвеєрна система з двох машин являється єдиним випадком, коли достатньо розглядати перестановочні розклади;
— для конвеєрної системи із трьох машин достатньо розглядати перестановочні розклади, якщо за критерій прийнята максимальна тривалість проходження;
— для конвеєрної системи із чотирьох і більше машин уже повинні бути розглянуті більш загальні типи розкладів, навіть при цьому критерії.

Наприклад, нехай задані дві роботи, які виконуються на чотирьох машинах. Тривалості їх виконання на кожній із машин задані в таблиці:

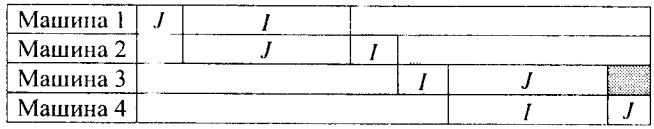
Машини	1	2	3	4
Робота *I*	4	1	1	4
Робота *J*	1	4	4	1

Існують тільки два розклади з однаковим порядком виконання робіт на кожній машині.





Максимальна тривалість проходження в цих випадках дорівнює 14. Тепер розглянемо розклад з однаковим порядком виконання робіт на кожній парі машин 1, 2 і 3, 4, причому порядок виконання робіт на машинах 3, 4 є оберненим по відношенню до порядку на машинах 1, 2.



Максимальна тривалість проходження в цьому випадку дорівнює 12, тобто менша, ніж в розкладах з однаковим порядком виконання робіт для всіх машин.

Мінімізація максимальної тривалості проходження в конвеєрній системі із двох машин $(\mathbf{n \mid 2 \mid F \mid F_{max}})$

Більшість посилань у публікаціях з теорії розкладів відноситься до роботи Джонсона, присвяченій дослідженню конвеєрної системи із двох машин. Для цієї системи Джонсон запропонував впорядкування, мінімізуюче максимальну тривалість проходження.

Алгоритм Джонсона

Для викладення алгоритму Джонсона і доведення його оптимальності зручно тимчасово прийняти його позначення. Тому нехай

$A_i = p_{i1}$ – тривалість першої операції (включаючи настройку, якщо вона необхідна) i-ої роботи;

$B_i = p_{i2}$ – тривалість другої операції (включаючи настройку, якщо вона необхідна) i-ої роботи;

F_i – тривалість проходження i-ої роботи.

Кожна робота характеризується парою (A_i, B_i), де A_i – частина роботи, що виконується першою машиною, B_i – другою. Таке розбиття існує для всіх робіт, хоча, можливо, що деякі A_i та B_i можуть бути рівними 0 (ряд робіт може складатись тільки з однієї операції). Обмеження, що спрощують модель, полягають в наступному:

1) одна машина одночасно не може виконувати більше однієї роботи;

2) одна робота одночасно не може виконуватися декількома машинами, причому A_i для довільного i повинна бути завершена раніше, ніж почнеться B_i.

Задача полягає в наступному. Задані $2n$ величини $A_1,...,A_n,B_1,...,B_n$. Потрібно впорядкувати роботи так, щоб при зроблених припущеннях мінімізувати максимальну із тривалостей проходження F_i.

Із теорем 1 і 2 випливає, що потрібно розглядати тільки розклади з однаковим порядком виконання робіт кожною машиною. Очевидно також, що подібно задачам впорядкування на одній машині тут не потрібно розглядати ні переривання, ні штучні простої. Тому розв'язок представляє собою просто оптимальну перестановку робіт.

Очевидно, що можна обмежитися розглядом розкладів, у яких роботи на першій машині "упаковані щільно", починаючи з першої із них, тобто перша машина працює без простоїв до завершення $A_{[n]}$. Зрозуміло, що до подібного розкладу може бути зведений довільний розклад шляхом відповідних зсувів вліво всіх назначень в ньому, поки не буде досягнута необхідна щільність. Очевидно, що ця процедура не призведе до зростання максимальної тривалості проходження.

Нехай $X_{[i]}$ – час простою другої машини, безпосередньо передуючий $B_{[i]}$. Типовий розклад має такий вигляд:

Машина 1	$A_{[1]}$	$A_{[2]}$		$A_{[3]}$	$A_{[4]}$		$A_{[5]}$	
Машина 2	$X_{[1]}$	$B_{[1]}$	$X_{[2]}$	$B_{[2]}$	$B_{[3]}$	$X_{[4]}$	$B_{[4]}$	$B_{[5]}$

Значення $X_{[i]}$ можуть бути виражені через A_S, B_S наступним чином:

$$X_{[1]} = A_{[1]},$$
$$X_{[2]} = \max\{A_{[1]} + A_{[2]} - B_{[1]} - X_{[1]}, 0\},$$
$$X_{[3]} = \max\{A_{[1]} + A_{[2]} + A_{[3]} - B_{[1]} - B_{[2]} - X_{[1]} - X_{[2]}, 0\}.$$

В загальному випадку можна записати

$$X_{[J]} = \max\left\{\sum_{i=1}^{J} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]} - \sum_{i=1}^{J-1} X_{[i]}, 0\right\}.$$

Звідси отримаємо часткові суми

$$X_{[1]} = A_{[1]},$$
$$X_{[1]} + X_{[2]} = \max\{A_{[1]} + A_{[2]} - B_{[1]}, A_{[1]}\},$$
$$X_{[1]} + X_{[2]} + X_{[3]} = \max\{A_{[1]} + A_{[2]} + A_{[3]} - B_{[1]} - B_{[2]}, X_{[1]} + X_{[2]}\} =$$
$$= \max\left\{\sum_{i=1}^{3} A_{[i]} - \sum_{i=1}^{2} B_{[i]}, \sum_{i=1}^{2} A_{[i]} - B_{[1]}, A_{[1]}\right\}.$$

Взагалі,

$$\sum_{i=1}^{J} X_{[i]} = \max\left\{\sum_{i=1}^{J} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]}, \sum_{i=1}^{J-1} A_{[i]} - \sum_{i=1}^{J-2} B_{[i]}, \ldots, \sum_{i=1}^{2} A_{[i]} - B_{[1]}, A_{[1]}\right\}.$$

Якщо Y_J визначається як

$$Y_J = \sum_{i=1}^{J} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]},$$

то

$$\sum_{i=1}^{J} X_{[i]} = \max\{Y_1, Y_2, ..., Y_J\}.$$

Тепер, якщо $F_{\max}(S)$ позначає максимальну тривалість проходження для конкретного розкладу S, то

$$F_{\max}(S) = \sum_{i=1}^{n} B_{[i]} + \sum_{i=1}^{n} X_{[i]} = \sum_{i=1}^{n} B_{[i]} + \max\{Y_1, Y_2, ..., Y_n\}.$$

Так як $\sum_{i=1}^{n} B_{[i]}$ не залежить від впорядкування, то максимум тривалості проходження залежить тільки від суми інтервалів простоїв другої машини, що еквівалентно, в свою чергу, максимуму $\{Y_i, i = \overline{1,n}\}$. Отже, задача відшукання розкладу S^* такого, що $F_{\max}(S^*) \le F_{\max}(S)$, для довільного S звелася до задачі впорядкування таким чином, щоб мінімізувався максимум n величин Y_i.

***Теорема* 3** (*Джонсона*). В системі $\mathbf{n|2|F|F_{max}}$ при одночасній доступності всіх робіт впорядкування, мінімізуюче максимальну тривалість проходження, таке, що робота j передує роботі $j+1$, якщо

$$\min\{A_j, B_{j+1}\} < \min\{A_{j+1}, B_j\}.$$

Впорядкування робіт у відповідності з теоремою 3 приводить до єдиного оптимального розкладу, якщо для кожної пари робіт має місце строга нерівність. Якщо ж існують такі пари робіт, для яких ця нерівність перетворюється в рівність, то існує безліч оптимальних розкладів.

Доведення теореми складається з двох частин. У першій частині показується, що впорядкування у відповідності із даною нерівністю мінімізує максимум величини Y_i. У другій частині відмічено транзитивність цієї нерівності, в силу чого розклад фактично встановлюється попарним впорядкуванням.

Доведення.

1. *Мінімізація максимуму* Y_i.

Припустимо, що ми заповнюємо у формованій черговості місця J та $J+1$, причому на місце J поміщаємо або роботу j, або $j+1$, а на місце $J+1$ – іншу. У відповідності з нерівністю теореми, робота j поміщається на місце J, якщо

$$\max\{-B_j, -A_{j+1}\} < \max\{-B_{j+1}, -A_j\}. \qquad (35)$$

Тепер розглянемо величину

$$\sum_{i=1}^{J+1} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]} \qquad (36)$$

і відмітимо, що вона не залежить від того, яка з двох робіт займає J-е місце. Додамо (36) до лівої частини (35):

якщо $-B_j > -A_{j+1}$, то $\sum_{i=1}^{J+1} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]} - B_j = Y_{J+1}$,

якщо $-B_j < -A_{j+1}$, то $\sum_{i=1}^{J+1} A_{[i]} - A_{j+1} - \sum_{i=1}^{J-1} B_{[i]} = Y_J$ при розміщенні роботи j на місці J.

Додавши (36) до правої частини (35), отримаємо:

якщо $-B_{j+1} > -A_j$, то $\sum_{i=1}^{J+1} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]} - B_{j+1} = Y'_{J+1}$,

якщо $-B_j < -A_{j+1}$, то $\sum_{i=1}^{J+1} A_{[i]} - A_j - \sum_{i=1}^{J-1} B_{[i]} = Y'_J$ при розміщенні роботи $j+1$ на місці J.

Тепер, якщо розміщення робіт виконано у відповідності з нерівністю теореми, то

$$\max\{Y_J, Y_{J+1}\} < \max\{Y'_J, Y'_{J+1}\},$$

тобто при розміщенні робіт j та $j+1$ на місця J та $J+1$ алгоритм встановлює оптимальну черговість цих робіт. Оскільки жодне з інших значень $\{Y_i\}$ не змінюється в результаті розміщення робіт j та $j+1$ на місця J та $J+1$, то максимум $\{Y_i\}$ або не зміниться, або зменшиться в результаті вказаного розміщення.

2. *Транзитивність відношень.*

Нам потрібно показати, що якщо для робіт i, j, k виконується

$$\min\{A_i, B_j\} \le \min\{A_j, B_i\} \quad \text{та} \quad \min\{A_j, B_k\} \le \min\{A_k, B_j\},$$

то

$$\min\{A_i, B_k\} \le \min\{A_k, B_i\}.$$

тобто в оптимальному розкладі робота i повинна передувати роботі j, а робота j – роботі k.

Якщо роботи i, j, k такі, що нерівності для i, j та j, k перетворюються в рівності, то твердження тривіальне. Тому розглянемо можливі випадки, коли це не так.

Випадок 1. $A_i \le B_j, A_j, B_i$ та $A_j \le B_k, A_k, B_j$.

Тоді $A_i \le A_j \le A_k$, $A_i \le B_i$, так що $A_i \le \min\{A_k, B_i\}$.

Випадок 2. $B_i \le A_i, A_j, B_j$ та $B_k \le A_k, A_j, B_j$.

Тоді $B_k \le B_j \le B_i$, $B_k \le A_k$, так що $B_k \le \min\{A_k, B_i\}$.

Випадок 3. $A_i \le B_j, A_j, B_i$ та $B_k \le A_j, A_k, B_j$.

Тоді $A_i \le B_i$, $B_k \le A_k$, так що $\min\{A_i, B_k\} \le \min\{A_k, B_i\}$.

Випадок 4. $B_j \le A_i, A_j, B_i$ та $A_j \le B_k, A_k, B_j$.

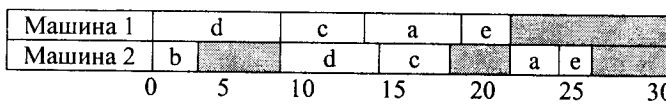
Тоді $A_j = B_j$ і між i, j, k існують відношення рівності.

Теорему доведено.

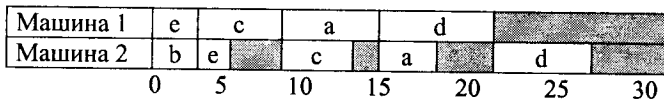
Наступний простий приклад пояснює суть джонсонівського алгоритму.

Позначення робіт	Перша операція A_i	Друга операція B_i
a	6	3
b	0	2
c	5	4
d	8	6
e	2	1

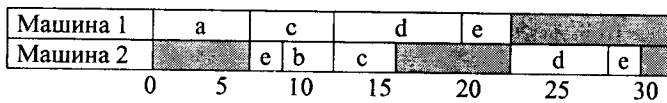
Схема Гранта для впорядкування у відповідності з алгоритмом Джонсона виглядає так:



Максимум тривалості проходження має оптимум, рівний 23, який і досягається при алгоритмі Джонсона. Більшість других впорядкувань (що базуються на інших принципах) призводять до більшої, ніж оптимальне значення, величини максимуму тривалості проходження. Наприклад, LPT-впорядкування, проведене за першою операцією, приводить до максимуму тривалості проходження, рівному 27.



Для порівняння, вхідне впорядкування робіт приводить до максимальної тривалості проходження, рівної 26.



Мінімізація середньої тривалості проходження в конвеєрній системі із двох машин $(n|2|F|\overline{F})$

У конвеєрній системі з двох машин мінімізація середньої тривалості проходження є дуже складною задачею. Для неї не існує конструктивного алгоритму, подібного джонсонівському, хоча теорема 1 справедлива і достатньо розглядати розклади з однаковою черговістю робіт на кожній машині. Якщо ж впорядкувати роботи у відповідності з джонсонівським алгоритмом, то черговість буде не тільки неоптимальною, але навіть не дуже хорошою. Це чітко видно на вище наведених прикладах, де середня тривалість проходження при джонсонівській черговості дорівнює 15, в той час, як для впорядкування SPT вона дорівнює 11,8. Перш, ніж переходити до загального випадку конвеєрної системи, виконуючої n робіт, розглянемо частинний випадок двох робіт. У цьому частинному випадку існує тільки два розклади S і S', наведені нижче:

Розклад S

Машина 1	A_1	A_2		
Машина 2		B_1	B_2	

Розклад S'

Машина 1	A_2	A_1		
Машина 2		B_2		B_1

В розкладі S сума тривалостей проходження двох робіт дорівнює

$$\overline{F} = 2A_1 + B_1 + B_2 + \max\{A_2, B_1\},$$

а в розкладі S' вона дорівнює

$$\overline{F}' = 2A_2 + B_1 + B_2 + \max\{A_1, B_2\}.$$

Ясно, що перша робота повинна передувати другій (тобто повинен бути вибраний розклад S), якщо $\overline{F} < \overline{F}'$, тобто

$$2A_2 - 2A_1 + \max\{A_1, B_2\} - \max\{A_2, B_1\} > 0. \qquad (37)$$

(Відмітимо, що два останніх члена в лівій частині нерівності виражають повну протилежність тому, що стверджується джонсонівським алгоритмом: вибір максимуму A_S і B_S. Якщо цим максимумом є A, то відповідна робота виконується першою, якщо B — то останньою). Звичайно, довільну пару робіт можна впорядкувати безпосередньо на основі (37), але цікаво знайти більш прості умови для тривалостей операцій, із яких буде слідувати оптимальне впорядкування. Для цього можна використати результати для одної машини: впорядкування у відповідності з мінімальною тривалістю операцій мінімізує середню тривалість проходження.

Існує декілька можливостей узагальнення цього результату на системи із двох машин. Перш за все відмітимо, що не має змісту розглядати тільки перші операції робіт для формування кращої черговості, так як навіть при відомому співвідношенні $A_2 > A_1$ твердження (37) може

виконуватися або не виконуватися в залежності від відповідних значень B_1 і B_2. По тим самим причинам недостатньою є інформація про те, що $B_2 > B_1$. Тому формування черговості у відповідності з мінімальною тривалістю тільки одної операції (першої чи другої) неприйнятне.

Звичайно, можна розглядати загальну тривалість роботи (рівну сумі тривалостей всіх операцій) і для кожної пари робіт виконувати першою ту, для якої

$$(A_2 + B_2) > (A_1 + B_1).$$

Однак наступний приклад показує, що із цієї умови ще не випливає виконання (37).

Нехай $B_1 = 0$, $A_2 = B_2$, $A_1 = A_2 + B_2 - \varepsilon$, де $\varepsilon > 0$. В цьому випадку, коли кожна операція першої роботи коротша відповідної операції другої, наведена умова достатня для виконання (37), і перша робота повинна передувати другій. Це дійсно так, оскільки твердження, що

$$A_2 \geq A_1 \text{ і } B_2 \geq B_1$$

означає, що

$$2A_2 - 2A_1 + \max\{A_1, B_2\} - \max\{A_2, B_1\} \geq 0.$$

Справедливість останньої нерівності випливає із розгляду наступних чотирьох випадків:

1. $A_1 > B_2$ і $A_2 > B_1$.

$$2A_2 - 2A_1 + A_1 - A_2 = A_2 - A_1 \geq 0.$$

2. $A_1 > B_2$ і $A_2 < B_1$.

Так як $A_2 > A_1$, то із наведених нерівностей випливає, що $B_1 > B_2$. Це суперечить початковим припущенням, тобто цей випадок неможливий.

3. $B_2 > A_1$ і $A_2 > B_1$.

$$2A_2 - 2A_1 + B_2 - A_2 = (A_2 - A_1) + (B_2 - A_1) \geq 0.$$

4. $B_2 > A_1$ і $B_1 > A_2$.

$$2A_2 - 2A_1 + B_2 - B_1 \geq 0.$$

Очевидно, що *ця умова достатня, але не необхідна.* Існують випадки, коли не всі операції роботи 1 коротші відповідних операцій роботи 2, тим не менше робота 1 повинна передувати роботі 2. Наприклад,

$$A_1 = 1,\ B_1 = 3,\ A_2 4,\ B_2 = 2.$$

Загальний вираз для середньої тривалості проходження n робіт може бути отриманий повторенням суджень Джонсона:

1) достатньо розглядати розклади з однаковим порядком виконанням робіт кожною машиною;
2) достатньо розглядати тільки ті розклади, в яких роботи на першій машині слідують безпосередньо одна за другою;
3) нехай $X_{[i]}$ буде часом простою, передуючим операції $B_{[i]}$ на другій машині;

4) нехай

$$Y_J = \sum_{i=1}^{J} A_{[i]} - \sum_{i=1}^{J-1} B_{[i]}.$$

Тоді тривалості проходження окремих робіт мають наступний вигляд:

$$F_{[1]} = A_{[1]} + B_{[1]} = X_{[1]} + B_{[1]},$$

$$F_{[2]} = X_{[1]} + B_{[1]} + X_{[2]} + B_{[2]},$$

.....

$$F_{[J]} = \sum_{i=1}^{J} B_{[i]} + \sum_{i=1}^{J} X_{[i]} = \sum_{i=1}^{J} B_{[i]} + \max\{Y_1, Y_2, ..., Y_J\}.$$

Середня тривалість проходження визначається як

$$\overline{F} = \frac{1}{n}\left[\sum_{J=1}^{n}\sum_{i=1}^{J} B_{[i]} + \sum_{J=1}^{n} \max\{Y_1, Y_2, ..., Y_J\}\right].$$

Використовуючи цей вираз, можна показати, що для всіх сусідніх у розкладі робіт I та J робота I повинна передувати роботі J, якщо

$$A_J \geq A_I \text{ та } B_J \geq B_I.$$

Однак ці нерівності не дозволяють встановити черговість, оскільки вони визначають відношення порядку тільки між двома роботами і не дозволяють провести повне впорядкування.

Конвеєрна система із трьох машин $(n|3|F|F_{max})$

При складанні розкладу, мінімізуючого максимальну тривалість проходження для конвеєрної системи із трьох машин, достатньо розглядати тільки перестановочні розклади (теорема 2). Тому здається дивним, що для цієї системи в загальному випадку нема конструктивного алгоритму, схожого на розглянутий раніше. Але в ряді частинних випадків такі розв'язки існують. Так, Джонсон розглянув систему із трьох машин, коли

$$\min A_i \geq \max B_i \quad \text{або} \quad \min D_i \geq \max B_i,$$

де D_i – час виконання i-ої роботи на третій машині. Згідно з цими припущеннями всі операції на другій машині за тривалістю менші довільної операції на першій чи на третій. Тому черговість, мінімізуюча максимальну тривалість проходження, формується таким чином, щоб робота I розміщувалася перед роботою J, якщо

$$\min\{A_I + B_I, \quad B_J + D_J\} < \min\{A_J + B_J, \quad D_I + B_I\}.$$

Таким чином, сам по собі алгоритм залишається таким самим, що і для системи із двох машин із тією різницею, що тепер A_i замінюється на $A_i + B_i$, а B_i на $B_i + D_i$.

Одна частинна задача для системи із трьох машин була розглянута Джексоном. В його припущеннях першою машиною виконуються всі перші

операції робіт, третьою – третя, а для кожної другої операції існує окрема машина. Таким чином, маємо $n+2$ машини, n із яких відповідають тільки другій машині (що може одночасно виконувати m робіт) в конвеєрній системі з трьох машин. Джексон показав, що впорядкування, мінімізуюче максимальну тривалість проходження, формується таким чином, що робота I передує J, якщо

$$\min\{A_I + B_I,\quad B_J + D_J\} < \min\{A_J + B_J,\quad D_I + B_I\},$$

де A_i, B_i, D_i мають той самий зміст.

Дане співвідношення є не чим іншим, як джонсонівською умовою оптимальності. Тому розглянутий Джексоном випадок служить ще одним прикладом системи, для якої джонсонівський алгоритм формує оптимальне впорядкування.

В розглянутому випадку B_i можна інтерпретувати як довільну затримку, обов'язкову між першою та другою операціями в конвеєрній системі з двох машин, пов'язану, наприклад, з настройками, охолодженням, сушкою і т. д. Крім того, B_i може приймати від'ємні значення, що відповідає тій ситуації в системі з двох машин, коли інтервали виконання операцій одної й тої ж роботи можуть перетинатися. Іншими словами, це можливо, якщо частина робіт виконується так, що друга операція починається до повного завершення першої. В такій постановці задача була розглянута Міттеном і Джексоном. Відмітимо, що здатність другої машини виконувати одночасно декілька робіт, призводить до того, що оптимальний розклад не обов'язково відноситься до класу розкладів з однаковим порядком виконання робіт на першій і третій машинах.

Впорядкування у великих системах конвеєрного типу

В даному параграфі ми розглянемо досить штучний випадок системи, яка складається із багатьох машин, що виконують дві роботи. Така постановка цікава тим, що допускає досить наглядний графічний метод розв'язання. Цей метод являється частинним випадком більш загального алгоритму, запропонованого Ейкерсом.

Припустимо, що номери машин $1, 2, \ldots, m$ відповідають порядку виконання на них робіт, і нехай тривалості робіт будуть

$p_{11}, p_{12}, \ldots, p_{1m}$ для роботи 1,

$p_{21}, p_{22}, \ldots, p_{2m}$ для роботи 2.

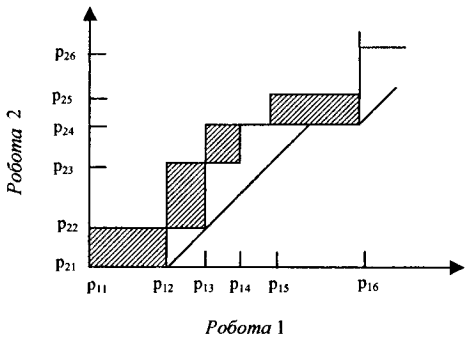


Рис. 3.13.

На рисунку 3.13 ці тривалості відкладені на осях координат. Графічно розклад може бути представлений ламаною лінією, яка

1) проходить від точки (0, 0) до точки $\left(\sum_{j=1}^{m} p_{1j}, \sum_{j=1}^{m} p_{ij}\right)$;
2) складається із відрізків горизонтальних (при виконанні тільки першої роботи), вертикальних (при виконанні тільки другої роботи) і, які проходять під кутом 45° (при одночасному виконанні обох робіт) прямих;
3) лежить цілком за межами заштрихованих на рисунку областей, які відповідають одночасному виконанню робіт обома машинами.

Ламана S на рис. 3.13 є прикладом "лінії розкладу". Ясно, що існує 2^m різних розкладів і стільки ж відповідних їм кривих на графіку, оскільки на кожній із машин роботи можуть виконуватися в довільному порядку. Ці ламані утворюють граф з коренем в точці (0, 0), який називається деревом. Точки галуження графа розміщені в місцях перетину кривої розкладу із заштрихованими областями. При цьому можливі три види галуження, представлені на рисунку 3.14.

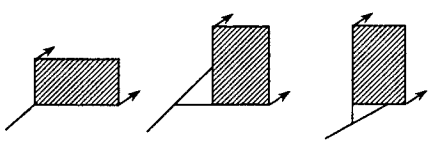


Рис. 3.14.

Максимальна тривалість проходження зв'язана із сумою вертикальних ділянок лінії розкладу. Вона дорівнює цій сумі плюс загальна тривалість всіх робіт на першій машині (або загальна тривалість всіх робіт на другій машині плюс сума горизонтальних ділянок лінії). Оптимальний розклад відповідає тій лінії, у якої мінімальна довжина вертикальних ділянок. Якщо лінія проходить під заштрихованими областями для машини J, то на цій машині робота 1 передує роботі 2.

Таким чином, можна по крайній мірі в принципі провести всі 2^m ламані і вибрати із них ту, в якої мінімальна сума вертикальних ділянок. Однак подібний метод практичну був би мало придатним, якщо фактично таких ліній виявилося б значно менше. Зменшення числа можливих варіантів відбувається тому, що ряд потенціальних гілок виявляється взагалі не можна реалізувати. Це обумовлено тим, що на деяких інтервалах лінія розкладу не перетинається із заштрихованими областями і, отже, на цих інтервалах відсутня можливість будь-якого вибору. З другого боку, існування великих заборонених областей (заштрихованих на рис. 3.13) призводить до того, що ряд ліній співпадає. В результаті, наприклад, отримуємо, що для системи із 12 машин, де число потенціальних розкладів 2^{10}, для знаходження оптимального проглядаються тільки біля 10.

Однак мета даного параграфу полягає не в тому, щоб дати конструктивний алгоритм пошуку розв'язків у достатньо виродженому випадку. На цьому прикладі нам хотілося розвинути в читача інтуїцію, необхідну в подальшому, і графік, представлений на рис. 3.13, скоріше ілюструє характер розкладів і ті властивості, якими повинен володіти кращий із них. Із цього графіка стає очевидним, що розклад повинен будуватися так, щоб ламана складалася із максимального числа ділянок, які проходять під кутом 45°. Крім того, він наглядно показує суттєві труднощі проблеми. Навіть в такому простому випадку неможливо встановити черговість виконання двох робіт на конкретній машині, виходячи із властивостей окремих операцій та інформації, яка міститься в уже проведеному впорядкуванні. Зокрема, не зрозуміло, як обходити в даній точці заборонену область (вверх чи вниз), не дослідивши при цьому подальшого проходження ламаних.

За виключенням наведених тут результатів для системи із m машин та теорем 1 і 2 (відносно черговості виконання робіт на двох перших і двох останніх машинах), в літературі не існує більше аналітичних результатів для системи із декількох машин. Всі публікації обмежуються максимум трьома машинами. В той же час широке розповсюдження отримали чисельні методи аналізу – цілочисельне програмування, метод гілок та границь і т.п.

В задачі впорядкування n робіт відносно m машин існує $(n!)^{m-2}$ можливих розклади. Навіть коли число варіантів біля $2{,}7 \cdot 10^{11}$, і

переглянути всіх їх практично неможливо. Тому, за невеликим виключенням, коли існують обчислювальні алгоритми, для аналізу системи з декількома машинами невідомі ефективні методи, гарантуючі отримання відносно доброго розкладу.

Хеллер опублікував результати дослідження системи, яка складається з 10 машин і для якої вивчається вид розподілу максимальної тривалості проходження різних розкладів (в його термінології час розкладу) та знаходиться метод складання розкладу, що належить лівому кінцю цього розподілу. Модель Хеллера містила 100 робіт із випадковими тривалостями, які рівномірно розподілені між 0 і 9 та приймають тільки цілочисельні значення. Розклади складалися із 10 незалежних перестановок, складених для кожної машини шляхом випадкового набору цифр від 1 до 100 і ототожнених з номерами робіт. Для кожного такого розкладу далі обчислювалась максимальна тривалість проходження. В результаті проведеної роботи він зробив висновок, що "чисельне дослідження показало, що функція розподілу часу розкладу являється нормальною. Теоретично доведено, що розподіл розкладу асимптотично нормальний для систем з великим числом робіт". Хеллер також відмітив, що встановлена нормальність функції розподілу може бути використана для прийняття рішень про завершення формування нових варіантів розкладів, якщо вартість подальшого пошуку перевищує математичне сподівання виграшу від цього пошуку.

Відносно застосування цього результату існує декілька думок, але на шляху його практичної реалізації є по крайній мірі дві суттєві труднощі:

1. Якщо розподіл часу розкладу асимптотично нормальний або апроксимується нормальним розподілом, то відхилення від нормальності в більшій степені будуть відчуватися на кінцях розподілу, тобто в місцях, що представляють найбільший інтерес.
2. Малоймовірно, щоб при формуванні варіантів розкладів базувалися на нормальному розподілі, так як існують більш ефективні та легко реалізовуємі.

Найбільш цікавими результатами дослідження Хеллера виявилось вивчення характеру впорядкування робіт відносно 10 машин. Ним було складено 9037 розкладів для 20 перших робіт при черговості виконання робіт, яка формується випадковим чином для кожної із машин. Крім того, Хеллером для тих самих 20 робіт було складено ще 12000 розкладів, в яких, на відміну від попередніх, черговість виконання робіт на кожній машині була одна й та ж. Хеллер робить висновок, що "якщо шукається розклад, мінімізуючий максимальний час розкладу, то пошук можливих варіантів повинен бути проведений серед класу розкладів з однаковим порядком виконання робіт кожною машиною". Такий висновок є безсумнівно корисним при виборі початкового варіанту для подальшого

пошуку оптимального розкладу. Однак цей клас розкладів не обов'язково є класом належності шуканого оптимуму.

Ньюджент опублікував дослідження такої ж системи із 20 робіт, як і Хеллер. При складанні розкладів він керувався правилами, які назвав "правилами імовірнісних пріоритетів". Не дивлячись на те, що ці правила були розроблені для довільних систем, вони приводять до розкладу з меншими значеннями максимуму тривалості проходження, ніж будь-який із описаних Хеллером. Ньюджент детально дослідив кращі із отриманих ним розкладів і виявив, що вони не належать класу розкладів з однаковим порядком виконання робіт на кожній із 10 машин. Скоріше вони відносяться до класу розкладів із випадковою і локальною перестановкою на послідовних машинах двох сусідніх у черговості робіт. звідси слідує, що знайдений клас розкладів кращий класу розкладів Хеллера (в якості вихідного) для наступного пошуку оптимуму і що в цьому напрямку багато ще потрібно зробити.

Недетерміновані системи впорядкування

1. Випадкове впорядкування.
2. Властивості антитетичних правил.
3. Впорядкування при неповній інформації.

Випадкове впорядкування

Розглянемо випадкове впорядкування (RANDOM) в системі **n | 1** при випадковій тривалості робіт.

Якщо встановлювати черговість виконання робіт виходячи із деяких ознак, що не залежать від тривалостей самих робіт, то тривалості впорядкування робіт будуть випадковими. Наприклад, якщо планові строки робіт не залежать від їх тривалостей, тоді впорядкування у відповідності з плановими строками приводить до випадкового впорядкування тривалостей робіт. Якщо ж впорядкування здійснюється у відповідності з резервами часу, то впорядкована послідовність тривалостей робіт не буде випадковою.

До сих пір ми розглядали тільки детерміновані системи, розклад для яких повністю визначався правилом впорядкування. Якщо ж тривалості робіт є випадковими величинами, то одне і те ж впорядкування кожний раз буде приводити до різних розкладів, тобто тривалість проходження кожної роботи і середня тривалість проходження будуть випадковими величинами, причому функція розподілу середньої тривалості проходження робіт залежить від впорядкування. Для математичного сподівання цього середнього існує наступна теорема:

Теорема. Математичне сподівання середньої тривалості проходження в системі **n | 1** при впорядкуванні, що не враховує тривалість роботи, дорівнює

$$M(\overline{F}) = \frac{n+1}{2}\overline{p},$$

де $\overline{p}$ – середнє арифметичне тривалостей робіт, якщо вони відомі наперед, і

$$M(\overline{F}) = \frac{n+1}{2}\hat{p},$$

де $\hat{p}$ – математичне сподівання тривалостей робіт, якщо вони представляють собою незалежні в сукупності випадкові величини з однаковими функціями розподілу.

Із цієї теореми випливають такі наслідки:

1) математичне сподівання середньої тривалості очікування дорівнює

$$M(\overline{W}) = \frac{n-1}{2}\overline{p};$$

2) математичне сподівання середнього часового зміщення дорівнює

$$M(\overline{L}) = \frac{n+1}{2}\overline{p} - \overline{a};$$

3) математичне сподівання середнього числа робіт в системі дорівнює

$$M\{\overline{N}(0, F_{\max})\} = \frac{n+1}{2}.$$

Властивості антитетичних правил

Два правила складання розкладу для системи $\mathbf{n|1}$ назвемо *антитетичними*, якщо вони приводять до протилежних послідовностей виконання робіт. Так, розклади S та S' антитетичні, якщо робота виконується i-ою в S і $(n-i+1)$-ою в S'. Прикладом антитетичних правил складання розкладу можуть бути впорядкування SPT та LPT. Цікавість до антитетичних правил обумовлена тим, що деякі критерії їх оцінки симетричні відносно відповідних середніх значень, що отримуємо при випадковому впорядкуванні.

Теорема. Якщо S і S' – антитетичні правила для системи $\mathbf{n|1|\overline{F}}$, то середні тривалості проходження при розкладах S та S' пов'язані співвідношенням

$$\overline{F}_S + \overline{F}_{S'} = (n+1)\overline{p}.$$

Доведення.

Нехай $[i]$ та $[i]'$ – позиції робіт, встановлені S і S' відповідно. Тоді

$$\overline{F}S = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)\,p_{[i]} = \frac{1}{n}\sum_{i=1}^{n}(n+1)\,p_{[i]} - \frac{1}{n}\sum_{i=1}^{n} i\,p_{[i]};$$

$$\overline{F}S' = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)\,p_{[i]}' = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)\,p_{[n-i+1]} = \frac{1}{n}\sum_{i=1}^{n} i\,p_{[i]}.$$

$$\overline{F}_S + \overline{F}_{S'} = \frac{1}{n}\sum_{i=1}^{n}(n+1)\,p_{[i]} = (n+1)\overline{p}.$$

Теорему доведено.

Аналогічне доведення дозволить встановити ту ж симетрію для середнього часу закінчення робіт, середньої тривалості очікування, середнього часового зміщення та середнього числа робіт у системі.

Теорема. Якщо в системі $\mathbf{n|1|\overline{F}}$ тривалості робіт незалежні в сукупності й однаково розподілені з функцією розподілу $G(p)$ і математичним сподіванням $\hat{p}$, то математичні сподівання середніх тривалостей проходження при впорядкуванні SPT і LPT мають вигляд

$$M(\overline{F}_{SPT}) = n\hat{p} - (n-1)Q\ , \quad M(\overline{F}_{LPT}) = \hat{p} + (n-1)Q,$$

де $Q = \int_0^\infty pG(p)dG(p)$.

У загальному випадку для Q не можна отримати хорошого вираження через моменти розподілу $G(p)$. Однак у ряді конкретних випадків це вдається зробити. Візьмемо, наприклад, розподіл Ерланга, що часто використовується в якості функції розподілу тривалостей робіт. Густина цього розподілу має вигляд

$$g(p) = \frac{\left(k/\hat{p}\right)^k p^{k-1}}{(k-1)!} e^{-\frac{kp}{\hat{p}}},\ p>0,$$

де $\hat{p}$ – математичне сподівання, $\hat{p}^2/k$ – дисперсія розподілу. При $k=1$ розподіл Ерланга перетворюється в показниковий; якщо k зростає, то дисперсія зменшується, але коефіцієнт асиметрії залишається додатним. При $k\to\infty$ розподіл Ерланга стає виродженим, тобто тривалість роботи дорівнює константі $\hat{p}$.

Впорядкування при неповній інформації

До сих пір ми вважали, що тривалості робіт відомі наперед і що ця інформація використовується при складанні розкладу. Однак у більшості практичних випадків тривалості робіт наперед не відомі, і при складанні розкладу керуються їх апріорною оцінкою.

Припустимо, що всі роботи в системі $\mathbf{n|1}$ мають різні тривалості й нумерація робіт така, що

$$p_1 < p_2 < \ldots < p_n .$$

Припустимо також, що $X_1, X_2, \ldots, X_n$ утворюють набір апріорних оцінок тривалостей робіт, на яких базується впорядкування SPT. Тоді черговість буде такою, що

$$X_{[1]} \le X_{[2]} \le \ldots \le X_{[n]} .$$

Необхідною умовою оптимального впорядкування є рівність $[i] = i$, а не $X_i = p_i$.

Припустимо, що тривалості робіт є незалежними в сукупності випадковими величинами з функцією розподілу $G_i(p)$ і математичним сподіванням $\hat{p}_i$ для роботи i. Припустимо, що $\hat{p}_i$ відомі і що роботи пронумеровані так, що

$$\hat{p}_1 \le \hat{p}_2 \le \ldots \le \hat{p}_n .$$

Припустимо далі, що фактична тривалість P_i не відома і що для впорядкування використовується $\hat{p}_i$.

Нехай роботи перенумеровані у відповідності з розкладом N (наприклад, SPT-розклад, складений на основі $\hat{p}_i$):

$$\overline{F}_N = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)P_i .$$

Далі, нехай S – оптимальний SPT-розклад, встановлений при умові, що істинні значення тривалостей відомі і

$$P_{[1]} \le P_{[2]} \le \ldots \le P_{[n]}, \qquad \overline{F}_S = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)P_{[i]} /$$

Щоб знайти $M(\overline{F}_N - \overline{F_S})$, знайдемо спочатку $M(\overline{F}_N)$:

$$M(\overline{F}_N) = \frac{1}{n}\sum_{i=1}^{n}(n-i+1)\hat{p}_i .$$

Обчислення $M(\overline{F}_S)$ пов'язане з певними труднощами, так як тривалості є випадкові величини і довільна робота в S може виявитися на довільному місці, хоча деякі конфігурації малоймовірні. У цьому питанні нам допоможе наступна теорема.

Теорема. Якщо в системі $\mathbf{n|1|\overline{F}}$ тривалості являють собою випадкові величини P_i з функціями розподілу $G_i(p)$ і впорядкування здійснюється у відповідності зі зростанням тривалостей робіт, то математичне сподівання вкладу роботи I в математичне сподівання середньої тривалості проходження дорівнює

$$Q_I = \hat{p}_I - \frac{1}{n}\int_0^{\infty} p\left[\sum_{\substack{i=1\\ i\neq I}}^{n} G_i(p)\right] dG_I(p),$$

де $\hat{p}_I = M(P_I)$.

Згідно з цією теоремою математичне сподівання середньої тривалості проходження при розкладі S має вигляд:

$$M(\overline{F}_S) = \sum_{i=1}^{n} Q_i = \sum_{i=1}^{n} \hat{p}_i - \frac{1}{n}\int_0^{\infty} \sum_{i=1}^{n} p \sum_{\substack{j=1\\ j\neq i}}^{n} G_j(p) dG_i(p) =$$

$$= \sum_{i=1}^{n} \hat{p}_i + \frac{1}{n}\sum_{i=1}^{n}\int_0^{\infty} pG_i(p)dG_i(p) - \frac{1}{n}\int_0^{\infty} p\left[\sum_{i=1}^{n} G_i(p)\right] d\left[\sum_{i=1}^{n} G_i(p)\right].$$

Впорядкування робіт на основі математичних сподівань їх тривалостей замість фактичних значень призводить до збільшення середньої тривалості проходження в системі. Математичне сподівання цього збільшення, обумовленого неповною інформацією про тривалості робіт, дорівнює

$$M(\overline{F}_N) - M(\overline{F}_S) =$$

$$= \frac{1}{n}\left\{ \sum_{i=1}^{n} (n-i)p_i - \sum_{i=1}^{n}\int_0^{\infty} pG_i(p)dG_i(p) + \int_0^{\infty} p\left[\sum_{i=1}^{n} G_i(p)\right] d\left[\sum_{i=1}^{n} G_i(p)\right] \right\}.$$

Ця величина повністю визначається функціями розподілу тривалостей робіт. Наведений вираз має непростий вигляд, але в кожному конкретному випадку він може бути обчислений.

Завдання для самостійної та індивідуальної роботи студентів

Завдання 1

1. Навести змістовні постановки задач, які зводяться до знаходження оптимального розкладу. При цьому визначити:

- множину впорядкованих робіт (в термінах запропонованої предметної області);
- відношення передування, які накладені на роботи;
- множину машин, на яких ці роботи можуть бути виконані;
- критерій оцінки розкладу.

Дати змістовні постановки задач, в яких:

a) тривалості робіт не залежать від впорядкування робіт;
b) тривалості робіт залежать від впорядкування робіт.

2. Записати математичний вираз для заданого критерію із урахуванням прийнятих позначень та визначити, чи являється цей критерій регулярним, чи ні (з доведенням):

a) максимальне значення запізнення робіт;
b) максимальне значення випередження робіт;
c) максимальне значення часового зміщення робіт;
d) максимальне значення моментів завершення робіт;
e) максимальне значення тривалостей очікування робіт;
f) максимальне значення тривалостей проходження робіт через систему;
g) зважене середнє значення запізнення робіт;
h) зважене середнє значення випередження робіт;
i) зважене середнє значення часового зміщення робіт;
j) зважене середнє значення моментів часу закінчення робіт;
k) зважене середнє значення тривалостей очікування;
l) зважене середнє значення тривалостей проходження робіт через систему;
m) середнє значення запізнення робіт;
n) середнє значення часового зміщення робіт;
o) середнє значення тривалостей очікування робіт;
p) середнє значення тривалостей проходження робіт;
q) середнє значення тривалості проходження робіт.

Варіант	1	2	3	4	5	6	7	8	9	10	11	12
Критерії	q, a	p, b	j, c	n, d	m, e	o, f	a, k	l, b	i, e	g, e	h, f	k, b

3. Пояснити кількість змінних та обмежень, що містяться в формулюванні загальної задачі складання розкладів, як задачі цілосисельного лінійного програмування (розділ 1, п.8).

Завдання 2

Для системи **10|1** скласти розклад виконання робіт, оптимальний за наступними критеріями:

a) мінімізація середнього часу очікування;
b) мінімізація середнього часу проходження;
c) мінімізація середнього часу очікування;
d) мінімізація середнього часу зміщення;
e) мінімізація середнього зваженого часу закінчення;
f) мінімізація середнього зваженого часу проходження;
g) мінімізація середнього зваженого часу очікування;
h) мінімізація середнього зваженого часу зміщення;
i) максимізація середнього часу закінчення;
j) максимізація середнього часу проходження;
k) максимізація середнього часу очікування;
l) максимізація середнього часу зміщення;
m) максимізація середнього зваженого часу закінчення;
n) максимізація середнього зваженого часу проходження;
o) максимізація середнього зваженого часу очікування;
p) максимізація середнього зваженого часу зміщення;
q) мінімізація максимізації зміщення;
r) мінімізація максимізації запізнення;
s) максимізація мінімізації зміщення;
t) максимізація мінімізації запізнення.

Варіант	Критерії	Роботи *i*														
		1			2			3			4			5		
		t_1	u_1	d_1	t_2	u_2	d_2	t_3	u_3	d_3	t_4	u_4	d_4	t_5	u_5	d_5
1	a,d,e,f,h	6	3	8	7	7	10	9	3	10	10	2	10	4	1	6
2	h,c,g,e,b	15	5	30	5	1	15	10	4	20	6	2	20	4	1	20
3	f,e,d,a,h	12	3	20	4	4	10	12	3	15	8	4	10	4	1	6
4	b,c,e,g,h	10	5	15	3	1	14	8	4	10	10	2	10	4	1	6
5	h,e,f,a,d	6	2	18	8	2	18	3	3	10	10	2	10	7	1	10
6	e,h,b,g,c	5	5	15	7	1	20	10	5	10	10	2	10	5	1	10
7	f,e,h,a,d	10	5	30	8	4	20	9	3	12	10	2	20	6	2	10
8	f,h,e,a,d	9	3	12	14	7	20	6	3	12	8	2	10	5	1	6
9	d,a,f,e,h	4	2	8	12	4	20	12	3	30	8	2	10	5	1	10

Варіант	Критерії	Роботи *i*														
		1			2			3			4			5		
		t_1	u_1	d_1	t_2	u_2	d_2	t_3	u_3	d_3	t_4	u_4	d_4	t_5	u_5	d_5
10	h,f,e,d,a	10	5	20	8	4	20	9	3	12	10	2	12	6	2	12
11	b,c,e,g,h	8	4	15	5	1	14	4	4	10	10	2	10	6	1	6
12	h,c,e,g,b	4	4	20	4	1	20	4	2	10	10	5	10	5	1	10

Варіант	Роботи *i*														
	6			7			8			9			10		
	t_6	u_6	d_6	t_7	u_7	d_7	t_8	u_8	d_8	t_9	u_9	d_9	t_{10}	u_{10}	d_{10}
1	2	1	8	9	3	10	5	1	8	4	2	12	12	4	20
2	3	1	10	6	3	10	5	1	8	4	2	12	10	4	15
3	3	1	8	9	3	20	7	1	15	6	1	12	10	5	10
4	2	2	8	9	3	10	5	1	8	4	2	12	12	4	15
5	4	1	10	12	4	20	5	1	10	4	2	12	3	1	12
6	4	2	8	9	3	10	4	1	12	4	2	12	12	3	20
7	4	4	10	8	4	10	5	1	8	4	2	10	5	1	10
8	4	1	8	9	3	9	3	1	8	6	2	12	14	7	30
9	3	1	10	12	3	20	3	1	8	2	2	12	7	1	12
10	4	4	6	8	4	12	5	1	12	4	2	10	5	1	10
11	4	2	8	9	3	10	5	1	8	4	2	12	10	5	15
12	4	2	8	12	3	10	5	1	8	4	2	12	7	7	20

t_i – тривалість i -ої роботи;
d_i – директивний строк i -ої роботи;
u_i – вага i -ої роботи.

Завдання 3

Варіант 1.

Довести теорему.

Розклад в системі $n|1|\overline{T}$ (мінімізація середнього часу закінчення виконання робіт) оптимальний, якщо після впорядкування тривалості робіт не спадають.

Варіант 2.

Довести теорему.

Розклад в системі $n|1|\overline{Z}$ (мінімізація середнього часового зміщення) оптимальний, якщо після впорядкування тривалості робіт не спадають.

Варіант 3.
Довести теорему.

Розклад в системі $n|1|L_{max}$ (мінімізація максимуму запізнення робіт в системі) оптимальний, якщо після впорядкування директивні строки робіт не спадають.

Варіант 4.
Довести теорему.

Розклад в системі $n|1|\overline{F}$ (максимізація середнього часу перебування в системі) оптимальний, якщо після впорядкування тривалості робіт не зростають.

Варіант 5.
Довести теорему.

Розклад в системі $n|1|Z_{min}$ (максимізація мінімального запізнення робіт в системі) оптимальний, якщо після впорядкування резерви робіт не спадають.

Варіант 6.
Довести теорему.

Розклад в системі $n|1|\overline{W}$ (максимізація середнього часу очікування в системі) оптимальний, якщо після впорядкування тривалості робіт не спадають.

Варіант 7.
Довести теорему.

Розклад в системі $n|1|\overline{T}$ (максимізація середнього часу закінчення виконання робіт) оптимальний, якщо після впорядкування тривалості робіт не зростають.

Варіант 8.
Довести теорему.

Розклад в системі $n|1|\overline{L}$ (максимізація середнього часового зміщення) оптимальний, якщо після впорядкування тривалості робіт не зростають.

Варіант 9.
Довести теорему.

Розклад в системі $n|1$ (мінімізація $\overline{T}_u$ середнього зваженого часу закінчення робіт в системі) оптимальний, якщо після впорядкування …

Варіант 10.

Довести теорему.

Розклад в системі $n|1|\overline{F_u}$ (максимізація середнього зваженого часу перебування в системі) оптимальний, якщо після впорядкування …

Варіант 11.

Довести теорему.

Розклад в системі $n|1$ (максимізація $\overline{W_u}$ середнього зваженого запізнення в системі) оптимальний, якщо після впорядкування …

Варіант 12.

Довести теорему.

Розклад в системі $n|1|\overline{W}$ (максимізація середнього часу очікування в системі) оптимальний, якщо після впорядкування тривалості робіт не зростають.

Завдання 4

Для системи $n|1$ скласти розклад із нульовим максимальним запізненням і мінімізуючий середню тривалість проходження робіт.

Варіант	Роботи *i*											
	1		2		3		4		5		6	
	t_1	d_1	t_2	d_2	t_3	d_3	t_4	D_4	t_5	d_5	t_6	D_6
1	3	20	4	36	9	24	5	36	8	30	6	14
2	7	40	8	22	2	39	9	31	4	5	6	37
3	3	40	10	24	7	35	5	6	6	26	3	26
4	12	50	8	25	10	60	2	80	6	27	10	20
5	6	30	4	32	10	45	7	50	4	7	12	27
6	12	20	6	55	7	22	8	50	3	28	10	48
7	15	30	10	55	10	35	8	44	6	60	5	40
8	12	26	10	37	8	40	6	48	6	30	4	50
9	10	20	10	44	8	24	8	45	7	47	5	48
10	18	33	15	58	11	38	10	64	3	40	13	70
11	18	31	10	55	15	50	10	34	5	60	2	60
12	12	30	10	34	10	50	8	55	5	52	2	35

t_i – тривалість i-ої роботи;
d_i – директивний строк i-ої роботи.

Завдання 5

Скласти розклад виконання n робіт одною машиною, мінімізуючий середній зважений час запізнення.

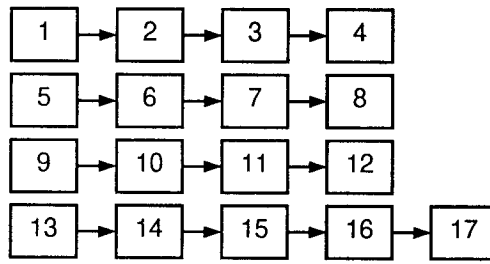
Варіант		1	2	3	4	5	6
	t_i	3	6	4	2	2	-
1	u_i	1	3	2	1	2	-
	a_i	12	6	6	5	10	-
	t_i	2	6	3	4	3	4
2	u_i	1	2	3	1	1	2
	a_i	12	14	2	16	10	6
	t_i	2	3	4	4	6	-
3	u_i	1	3	2	1	2	-
	a_i	2	14	5	9	14	-
	t_i	4	2	3	1	2	-
4	u_i	2	1	1	1	2	-
	a_i	7	4	5	7	2	-
	t_i	2	4	6	3	3	-
5	u_i	1	2	2	1	3	-
	a_i	5	3	10	11	12	-
	t_i	3	6	4	2	6	-
6	u_i	3	3	2	1	3	-
	a_i	2	14	7	12	10	-
	t_i	4	3	8	12	-	-
7	u_i	2	1	2	3	-	-
	a_i	9	4	8	12	-	-
	t_i	12	4	14	16	-	-
8	u_i	6	5	8	4	-	-
	a_i	14	4	26	28	-	-
	t_i	7	8	6	2	-	-
9	u_i	8	4	6	5	-	-
	a_i	13	14	7	2	-	-
	t_i	3	3	2	6	5	-
10	u_i	3	2	1	2	2	-
	a_i	14	5	2	14	9	-
	t_i	3	2	6	4	4	-
11	u_i	1	3	3	2	3	-
	a_i	11	5	10	3	12	-
	t_i	10	5	10	6	7	-
12	u_i	5	5	2	1	1	-
	a_i	10	15	10	10	20	-

Завдання 6

Для одної машини скласти розклад виконання робіт, мінімізуючий середню тривалість проходження робіт, при умові, що:

a) роботи частково впорядковані (ланцюжки не можуть розриватися);

b) для робіт задано відношення передування.



Позначення: [i] – i-а робота з тривалістю t_i.

Варіант	Тривалість роботи i																
	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}	t_{17}
1	2	1	4	3	5	3	8	–	2	6	–	–	7	2	3	6	8
2	3	1	5	–	6	3	–	–	7	–	–	–	8	4	5	2	–
3	8	9	5	2	1	4	2	–	3	–	–	–	7	2	–	–	–
4	5	10	–	–	2	8	2	–	5	3	–	–	4	6	2	8	–
5	1	3	–	–	2	5	3	6	4	10	–	–	6	2	8	–	–
→ **6**	6	2	5	–	2	–	–	–	3	2	5	4	3	1	–	–	–
7	4	5	1	–	3	6	4	2	3	2	–	–	4	8	–	–	–
8	5	6	–	–	1	2	–	–	4	5	6	–	4	6	3	2	–
9	2	4	3	–	5	–	–	–	8	2	6	4	3	4	–	–	–
10	4	3	–	–	5	–	–	–	8	4	6	–	4	5	2	8	–
11	6	–	–	–	8	1	5	–	6	4	8	2	4	3	–	–	–
12	1	4	–	–	2	5	2	7	5	3	4	–	3	–	–	–	–

Завдання 7

Паралельні машини. Роботи виконуються одночасно декількома машинами.

Сконструювати приклад для $n = 8 \div 10$ робіт та $m = 4$ машин. Тривалості робіт і виробнича потужність всіх робіт різні.

Завдання 8

Паралельні машини. Заборонено одночасне виконання робіт декількома машинами. Машини різної виробничої потужності.

Сконструювати приклад для $n = 8 \div 10$ робіт та $m = 3$ машин; коефіцієнти виробничої потужності машин $1 \le k_i \le 2$, $i = \overline{1,3}$.

Завдання 9

Задача мінімізації максимальної тривалості проходження в конвеєрній системі з двох машин ($\mathbf{n|2|F|F_{max}}$). Сконструювати приклад для $n = 8 \div 10$ робіт та $m = 2$ машин.

Контрольні запитання

1. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація середнього числа робіт, які знаходяться в системі $R = \frac{1}{n}\sum_{p=0}^{T_{\max}} k(p)$, де $k(p)$ – кількість робіт, що знаходяться в системі в момент часу p, $p = 1, 2, ..., T_{\max}$? Припускається, що тривалості всіх робіт – цілі числа.
2. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація максимальної тривалості очікування?
3. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація мінімальної тривалості проходження?
4. Який розклад являється оптимальним в системі $\mathbf{n|1|}$максимізація максимального запізнення?
5. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація максимальної тривалості завершення?
6. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація максимальної кількості робіт в системі?
7. Який розклад являється оптимальним в системі $\mathbf{n|1|}$мінімізація середнього числа робіт, які вийшли із системи $R = \frac{1}{n}\sum_{p=0}^{T_{\max}} k(p)$, де $k(p)$ – кількість робіт, що виконуються в системі в момент часу p, $p = 1, 2, ..., T_{\max}$? Припускається, що тривалості всіх робіт – цілі числа.
8. Який розклад являється оптимальним в системі $\mathbf{n|1|}$максимізація мінімальної тривалості проходження?

9. Який розклад являється оптимальним в системі **n|1|**максимізація максимального очікування?
10. Який розклад являється оптимальним в системі **n|1|**мінімізація середнього зваженого числа робіт, які знаходяться в системі $R = \frac{1}{n}\sum_{p=0}^{T_{\max}} k(p)$, де $k(p)$ – кількість робіт, що знаходяться в системі в момент часу p, $p = 1, 2, ..., T_{\max}$? Припускається, що тривалості всіх робіт – цілі числа.
11. Який розклад являється оптимальним в системі **n|1|**мінімізація максимальної тривалості проходження?
12. Який розклад являється оптимальним в системі **n|1|**мінімізація максимальної кількості робіт в системі?
13. Який розклад являється оптимальним в системі **n|1|**мінімізація мінімальної тривалості завершення?
14. Який розклад являється оптимальним в системі **n|1|**мінімізація середнього зваженого числа робіт, які знаходяться в системі $R = \frac{1}{n}\sum_{p=0}^{T_{\max}} k(p)$, де $k(p)$ – кількість робіт, що знаходяться в системі в момент часу p, $p = 1, 2, ..., T_{\max}$? Припускається, що тривалості всіх робіт – цілі числа.
15. Який розклад являється оптимальним в системі **n|1|**мінімізація мінімального очікування?
16. Який розклад являється оптимальним в системі **n|1|**максимізація мінімального очікування?

Варіант	1	2	3	4	5	6	7	8	9	10	11	12
Завдання	a,e, f, b	g,m l, b	j, f, e, b	n,o, c, f	a,p, h, l	g,i, k, d	j,o, h, d	n,p, k, f	a,i, c, l	g,o, k, d	j,p, c, e	n,i, h,m

Література

1. Конвей Р.В., Максвелл В.Л., Миллер Л.В. Теория расписаний. Наука. М.: 1975, - 360 с.
2. Танаев В.С., Гордон В.С., Шафранский И.М. Теория расписаний. Одностадийные системы. Наука. М.: 1984. – 384 с.
3. Танаев В.С., Гордон В.С., Шафранский И.М. Теория расписаний. Многостадийные системы. Наука. М.: 1985. – 384 с.
4. Танаев В.С., Шкурба В.В. Введение в теорию расписаний. Наука. М.: 1975. – 256 с.
5. Основы системного анализа и проектирования АСУ / А.А. Павлов, С.Н. Гриша, В.Н. Томашевский и др. Под общ. ред. А.А. Павлова. – К.: Выща шк. 1991. – 367 с.
6. Жданова О.Г. Методичні вказівки з дисципліни "Математичні методи дослідження операцій". Частина 7. Метод гілок та границь. – К.: НТУУ – "КПІ", 1998. – 42с.

Зміст

Ващук Ф.Г., Лавер О.Г., Шумило Н.Я.
В 23 Теорія розкладів: Навчальний посібник. – Ужгород: Видавництво "Два кольори", 2003. – 136 с.

ISBN 966-7781-13-5

Навчальний посібник з дисципліни "Теорія розкладів" для викладачів та студентів факультету інформатики Ужгородського державного інституту інформатики, економіки і права, а також математичних та інженерно-технічних спеціальностей вищих навчальних закладів.

ББК 22.18
УДК 519.87

Навчальне видання

Ващук Ф.Г., Лавер О.Г., Шумило Н.Я.

ТЕОРІЯ РОЗКЛАДІВ
Навчальний посібник

Науковий редактор О.Г. Лавер
Обкладинка Б.В.Васильєв-Сазанов
Коректор Н.Я. Шумило

Здано до складання 26.08.03. Підп. до друку 26.09.03. Формат 60x84/16.
Папір офс. Друк офс. Гарнітура Times New Roman Cyr. Ум.друк.арк. 7,9.
Ум.фарб.-відб. 8,14. Обл.-вид.арк. 8,25.
Зам. 1305. Тираж 300 пр.

Видавництво "Два кольори"
88000 Ужгород, Гренджі-Донського, 3
Міська друкарня. 88005 Ужгород, Руська, 13.