Г.З. Халімов

КОНСПЕКТ ЛЕКЦІЙ
з дисципліни
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНІ СИСТЕМИ
(частина ІІ)



Харків 2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

КОНСПЕКТ ЛЕКЦІЙ
з дисципліни
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНІ СИСТЕМИ
(частина ІІ)

для студентів усіх форм навчання спеціальності 125 «Кібербезпека»
за освітньою програмою «Безпека інформаційних комунікаційних систем»

Електронне видання

ЗАТВЕРДЖЕНО
кафедрою БІТ.
Протокол № 12 від 04.04.2019

Харків 2019

Конспект лекцій з дисципліни «Інформаційно-комунікаційні системи», частина ІІ, для студентів усіх форм навчання спеціальності 125 «Кібербезпека» за освітньою програмою «Безпека інформаційних комунікаційних систем» [Електронний ресурс] / Упоряд. Г.З. Халімов. – Електронне видання. – Харків: ХНУРЕ, 2019. – 207 с. – pdf 6,19 Mb.

Упорядник Г.З. Халімов

Рецензент О.В. Сєвєрінов, канд. техн. наук, доцент, каф. БІТ

ЗМІСТ

Передмова 5

Лекція 1. Методи забезпечення якості обслуговування в мережах з комутацією пакетів 6

1.1 Загальний аналіз забезпечення якості обслуговування в мережах з комутацією пакетів 6

1.2 Класифікації трафіка додатків і якість обслуговування 9

1.3 Аналіз черг на основі моделі масового обслуговування 13

1.4 Контрольні запитання та завдання 21

Лекція 2. Аналіз мереж на основі моделей масового обслуговування 22

2.1 Теорема Літтла 22

2.2 Аналіз черг обслуговування на основі моделі ланцюга Маркова 26

2.3 Контрольні запитання та завдання 32

Лекція 3. Методи забезпечення якості обслуговування на основі керування чергами та кондиціювання трафіка 33

3.1 Методи керування чергами 33

3.2 Механізми кондиціювання трафика 41

3.3 Контрольні запитання та завдання 45

Лекція 4. Методи керування трафіком на основі зворотного зв'язку та резервування ресурсів 46

4.1 Механізми використання зворотного зв'язку для керування трафіком 46

4.2 Резервування ресурсів та контроль доступу 50

4.3 Контрольні запитання та завдання 68

Лекція 5. Алгоритми маршрутизації 69

5.1 Алгоритми побудови остовного дерева мінімальної ваги 69

5.2 Мережні алгоритми знаходження найкоротшого шляху. Алгоритм Беллмана–Форда 75

5.3 Контрольні запитання та завданя 78

Лекція 6. Маршрутизація при деградації та відновленні мережі 79

6.1 Проблема коливання трафіка під час адаптивної маршрутизації 79

6.2 Маршрутизація при деградації та відновленні мережі. Лавинний алгоритм оновлення топології мережі 83

6.3 Задача побудови топології мережі 85

6.4 Контрольні запитання та завдання 87

Лекція 7. Завдання побудови топології мережі 88

7.1 Розв'язання задачі вибору пропускної здатності ліній мережі зв'язку 88

7.2 Евристичні методи вибору пропускної здатності ліній мережі зв'язку 89

7.3 Контрольні запитання та завдання 93

Лекція 8. Методи оптимальної маршрутизації 94
8.1 Задача надійності мережі зв'язку, k -зв'язані мережі. Алгоритм Клейтмана 94
8.2 Побудова топології остовного дерева. Алгоритм Ессау–Вільямса 96
8.3 Задача оптимальної маршрутизації 98
8.4 Розв'язання задач оптимальної маршрутизації 102
8.5 Контрольні запитання та завдання 104
Лекція 9. Методи оптимальної маршрутизації Франка–Волфа 105
9.1 Сутність методу Франка–Волфа 105
9.2 Приклад розв'язання задачі оптимальної маршрутизації методом Франка–Волфа 106
9.3 Контрольні запитання та завдання 109
Лекція 10. Проекційні методи оптимальної маршрутизації 110
10.1 Нелінійна оптимізація без обмеження 110
10.2 Приклад розв'язання задачі оптимальної маршрутизації без обмеження 116
Лекція 11. Протокол міжмережної взаємодії 120
11.1 Формат IP-пакета 120
11.2 Схема IP-маршрутизації 124
11.3 Спрощена таблиця маршрутизації 126
11.4 Таблиці маршрутизації кінцевих вузлів 129
Лекція 12. Маршрутизація з використанням маски 131
12.1 Приклад IP-маршрутизації без маски 131
12.2 Визначення маршрутизації з використанням маски 139
12.3 Структуризація мережі масками однакової довжини 139
Лекція 13. IP-маршрутизації з фрагментацією пакетів 144
13.1 Перегляд таблиць маршрутизації з масками 144
13.2 Фрагментація IP-пакетів 153
13.3 Механізм фрагментації 155
Теоретичний матеріал до практичного заняття 160
1 Основні функції протоколу IP 160
2 Структура IP-пакета 161
3 Таблиці маршрутизації в IP-мережах 165
4 Маршрутизація без використання масок 171
5. Маршрутизація з використанням масок 175
6 Фрагментація IP-пакетів 189
7 Протокол надійної доставки TCP-повідомлень 194
Запитання та завдання 203
Рекомендована література 206

ПЕРЕДМОВА

Електронний конспект лекцій з дисципліни «Інформаційно-комунікаційні системи» містить основні відомості щодо архітектури комп'ютерних мереж, технологій побудови локальних мереж, мережних протоколів, математичних моделей мережного обслуговування, методів забезпечення якості обслуговування, маршрутизації, побудови мережних топологій.

Курс лекцій розділений на дві частини за семестрами навчання. В цілому обмежений вибір тем лекцій визначається навчальною програмою підготовки. Наприкінці викладення лекцій надано ряд запитань для перевірки знань студентів. Практичні вправи винесені на практичні та лабораторні заняття.

Зміст електронного конспекту на матеріалі двох фундаментальних підручників: «Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов» (В.Г. Олифер, Н.А. Олифер) та «Сети передачи данных» (Д. Бертсекас, Р. Галагер).

Основні концепції та принципи розглянуто за допомогою простих математичних засобів, моделей і аналізу. Для ілюстрації ідей широко використовуються рисунки, приклади, що пояснюють більш тонкі деталі. Включення математичного аналізу забезпечує глибше розуміння матеріалу і розвиток здібностей до досліджень у цій галузі.

Частина курсу лекцій присвячена викладенню питань забезпечення якості обслуговування в мережах з комутацією пакетів, маршрутизації, побудови мережних топологій із заданими властивостями. Математичний аналіз побудований на основі методів множинного доступу, моделей графів і масового обслуговування. Вивчення алгоритмів маршрутизації ґрунтується на оптимізаційних методах пошуку екстремумів, синтезі топологій на базі лінійного програмування.

Попри обмеженість лекційних тем, їх вибір достатній для формування базових знань у цій галузі та є демонстрацією класичної роботи математичних методів під час пошуку мережних рішень.

Лекція 1
МЕТОДИ ЗАБЕЗПЕЧЕННЯ ЯКОСТІ ОБСЛУГОВУВАННЯ В МЕРЕЖАХ З КОМУТАЦІЄЮ ПАКЕТІВ

1.1 Загальний аналіз забезпечення якості обслуговування в мережах з комутацією пакетів

Черги є невід'ємним атрибутом мереж з комутацією пакетів. Принцип роботи таких мереж враховує наявність буфера у кожного вхідного й вихідного інтерфейсу комутатора пакетів. Буферизація пакетів під час перевантажень є основним механізмом підтримки пульсуючого трафіка, що забезпечує високу продуктивність мереж цього типу. Як ви знаєте, у мережах з іншим типом комутації, а саме в мережах з комутацією каналів, проміжна буферизація даних не підтримується. Водночас черги означають невизначену затримку під час передачі пакетів через мережу, а в деяких випадках і втрати пакетів через переповнення буфера комутатора або маршрутизатора, відведеного під чергу. Затримки та втрати пакетів – це головне джерело проблем для чутливого до затримок трафіка. Оскільки сьогодні оператори пакетних мереж дуже зацікавлені в передачі пульсуючого трафіка, їм необхідні засоби досягнення компромісу між вимогами граничного завантаження своєї мережі і якістю обслуговування водночас усіх типів трафіка.

Існує два підходи до визначення того, які характеристики продуктивності й надійності слід віднести щодо характеристик якості обслуговування, тобто до тих характеристик, які можуть бути покращені за допомогою методів QoS.

У першому випадку, під характеристиками QoS розуміють тільки три характеристики:

- однобічна затримка пакетів;
- варіація затримок пакетів;
- втрати пакетів.

Інший підхід полягає в розширеному тлумаченні характеристик QoS, коли характеристики швидкості потоку, такі, як середня швидкість, пікова швидкість і пульсація, також відносять до характеристик QoS.

У методах забезпечення якості обслуговування використовуються різні механізми, спрямовані на зниження негативних наслідків перебування пакетів у чергах, зі збереженням, водночас, позитивної ролі черг. Набір механізмів доволі широкий, і в цій лекції вони розглядаються досить докладно.

Більшість з них враховує й використовує у своїй роботі факт існування у мережі трафіка різного типу в тому сенсі, що кожний тип трафіка висуває різні вимоги до характеристик продуктивності й надійності мережі. Наприклад, трафік перегляду веб-сторінок малочутливий до затримок пакетів і не вимагає гарантованої пропускної здатності мережі, але чутливий до втрат пакетів; водночас голосовий трафік дуже чутливий до затримок пакетів, вимагає гарантованої пропускної здатності мережі, але може «терпіти» втрату невеликого відсотка пакетів без значного збитку для якості (втім, остання властивість багато в чому залежить від використовуваного методу кодування голосового сигналу).

Домогтися одночасного дотримання усіх характеристик QoS для усіх видів трафіка досить складно. Одним з найзначущих факторів, що впливають на характеристики якості обслуговування, є рівень завантаження мережі трафіком, тобто рівень використання пропускної здатності ліній зв'язку мережі.

Якщо цей рівень постійно досить низький, то трафік усіх додатків обслуговується з високою якістю більшу частину часу (хоча короткочасні перевантаження мережі, що призводять до затримок і втрат пакетів, усі рівноможливі, але вони відбуваються дуже рідко). Такий стан мережі називається «недовантаженим» або використовується термін «мережа з надлишковою пропускною здатністю» (англ. – overprovisioning). Постійно підтримувати всі частини мережі в недовантаженому стані досить дорого й складно, але для найбільш відповідальної частини мережі, такої як магістраль, цей підхід застосовується, і пов'язаний він з постійним спостереженням за рівнем завантаження каналів магістралі й періодичним збільшенням їхньої пропускної здатності з наближенням завантаження до критичного рівня.

Методи QoS засновані на іншому підході, а саме: на тонкому перерозподілі наявної пропускної здатності між трафіком різного типу відповідно до вимог додатків. Зрозуміло, що ці методи ускладнюють мережне устаткування, оскільки означають необхідність знання вимог усіх класів трафіка, уміння їх класифікувати й розподіляти пропускну здатність мережі між ними. Остання властивість зазвичай досягається за рахунок використання декількох черг пакетів для кожного вихідного інтерфейсу комунікаційного устаткування замість однієї черги; при цьому в чергах застосовують різні алгоритми обслуговування пакетів, чим і досягається диференційоване обслуговування трафіка різних класів. Тому, методи QoS часто асоціюються з технікою керування чергами.

Крім власне техніки організації черг, до методів QoS належать методи контролю параметрів потоку трафіка, оскільки для гарантовано якісного обслуговування слід бути впевненими, що потоки, які обслуговуються, відповідають певному профілю. Ця група методів QoS отримала назву методів кондиціювання трафіка.

Особливе місце займають методи зворотного зв'язку, які призначені для повідомлення джерела трафіка про перевантаження мережі. Ці методи розраховані на те, що при отриманні повідомлення джерело знизить швидкість видачі пакетів у мережу й тим самим ліквідує причину перевантаження.

Механізми QoS можна застосовувати по-різному. У тому випадку, коли вони застосовуються до окремих вузлів без урахування реальних маршрутів проходження потоків трафіка через мережу 1, умови обслуговування трафіка цими вузлами покращуються, але гарантій того, що потік буде обслужений із заданим рівнем якості, такий підхід не дає. Гарантії можна забезпечити, якщо застосовувати методи QoS системно, резервуючи ресурси мережі для потоку протягом всього його маршруту, інакше кажучи, «від краю до краю».

До методів QoS тісно примикають методи інжинірингу трафіка. Згідно з методами інжинірингу трафіка маршрути передачі даних керуються так, щоб забезпечити збалансоване завантаження усіх ресурсів мережі й виключити за рахунок цього перевантаження комунікаційного устаткування і утворення довгих черг. На відміну від методів QoS у методах інжинірингу трафіка не вдаються до організації черг з різними алгоритмами обслуговування на мережному устаткуванні. Водночас, у методах QoS у їхньому традиційному розумінні не використовують такий потужний важіль впливу на раціональний розподіл пропускної здатності, як зміна маршрутів трафіка залежно від фактичного завантаження ліній зв'язку, що дозволяє легко відокремити методи QoS від методів інжинірингу трафіка.

У наступній групі методів боротьба з перевантаженнями ведеться за рахунок зниження постійного навантаження на мережу. Тобто в цих методах проблема розглядається з іншого боку: якщо пропускної здатності мережі недостатньо для якісної передачі трафіка додатків, то чи не можна зменшити обсяг самого трафіка? Найочевиднішим способом зниження обсягу трафіка є його компресія. Існують й інші способи, які призводять до того самого результату. Наприклад, розміщення джерела даних ближче до його споживача (кешування даних).

1.2 Класифікації трафіка додатків і якість обслуговування

На сьогодні проведено велику роботу з класифікації трафіка додатків. Як основні критерії класифікації були прийняті три характеристики трафіка:

- відносна передбачуваність швидкості передачі даних;
- чутливість трафіка до затримок пакетів;
- чутливість трафіка до втрат і викривлення пакетів.

Передбачуваність швидкості передачі даних

Відносно передбачуваності швидкості передачі даних додатки поділяються на два великі класи: *з потоковим* і *з пульсуючим* трафіком.

Додатки з потоковим трафіком (stream) породжують рівномірний потік даних, який надходить у мережу з постійною бітовою швидкістю (Constant Bit Rate, CBR). У випадку комутації пакетів трафік таких додатків є послідовністю пакетів однакового розміру (рівного B біт), що прямують один за одним через однаковий інтервал часу T (рис. 1.1).

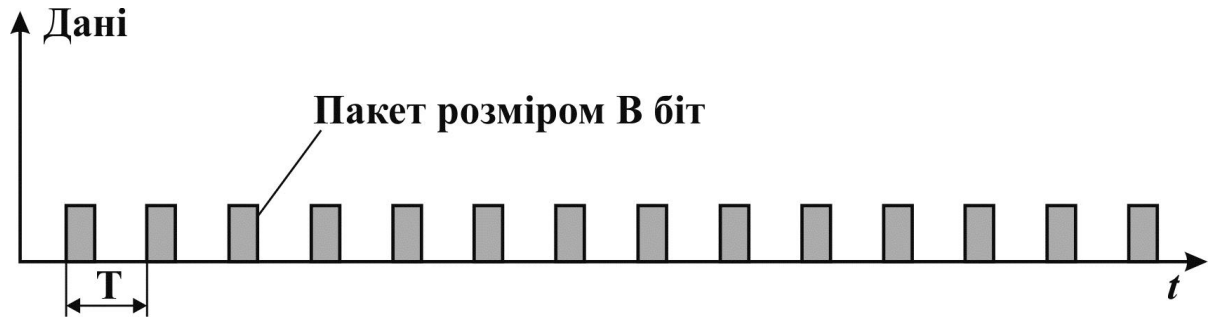


Рисунок 1.1 – Потоковий трафік

Постійна бітова швидкість потокового трафіка (CBR) може бути обчислена шляхом усереднення на одному періоді:

$$\text{CBR} = \frac{B}{T}\ біт/с\,.$$

У загальному випадку, постійна бітова швидкість потокового трафіка менше номінальної максимальної бітової швидкості протокола, за допомогою якого передаються дані, оскільки між пакетами існують паузи. Як буде показано далі, максимальна швидкість передачі даних за допомогою протоколу Ethernet становить 9,76 Мбіт/с (для кадрів максимальної довжини), що менше номінальної швидкості цього протоколу, яка дорівнює 10 Мбіт/с.

Додатки з пульсуючим трафіком (burst) відрізняються високим ступенем непередбачуваності. У цих додатках періоди мовчання змінюються пульсацією,

протягом якої пакети «щільно» прямують один за одним. У результаті трафік характеризується змінною бітовою швидкістю (Variable Bit Rate, VBR), що ілюструє рис. 1.2.

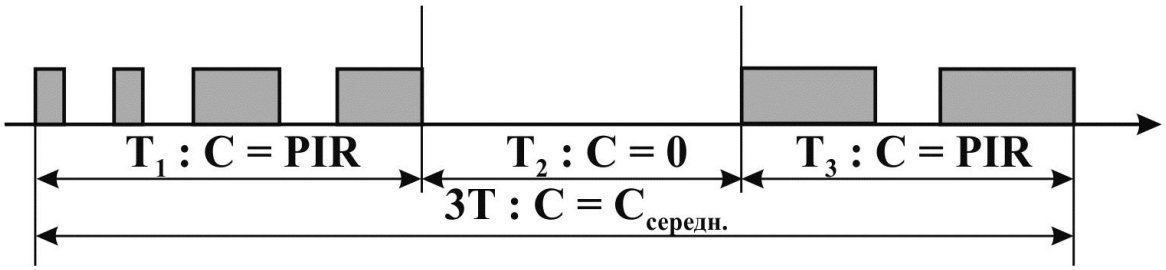


Рисунок 1.2 – Пульсуючий трафік

Так, під час роботи додатків файлового сервісу інтенсивність трафіка, що генерується додатком, може падати до нуля, коли файли не передаються, і підвищуватися до максимально доступної, обмеженої тільки можливостями мережі, коли файловий сервер передає файл.

На рис. 1.2 показано три періоди вимірювань T_1, T_2 і T_3. Для спрощення розрахунків прийнято, що пікові швидкості на першому й третьому періодах рівні між собою й дорівнюють PIR, а всі три періоди мають однакову тривалість T. Враховуючи це, можна обчислити величину пульсації B, яка дорівнює кількості бітів, переданих на періоді пульсації:

$$B = PIR \times T .$$

Отже, величина пульсації для періодів T_1 і T_3 дорівнює B, а на періоді T_2 – нулю.

Для наведеного прикладу можна підрахувати коефіцієнт пульсації. (Нагадаємо, що він дорівнює відношенню пікової швидкості на будь-якому невеликому періоді часу до середньої швидкості трафіка, обмірюваної на тривалому періоді часу.) Оскільки пікова швидкість на періоді T_1 (або T_3) дорівнює B/T, а середня швидкість на сумарному періоді $T_1 + T_2 + T_3$ становить $2B/3T$, коефіцієнт пульсації дорівнює $3/2$.

Практично будь-який трафік, навіть трафік потокових додатків, має ненульовий коефіцієнт пульсації. Просто значення коефіцієнтів пульсації потокового та пульсуючого трафіків значно різняться. У додатків з пульсуючим трафіком він зазвичай перебуває у межах від 2 до 100, а в потокових додатків він близький до 1.

У локальних мережах коефіцієнт пульсації зазвичай вищий, ніж у глобальних, оскільки на магістралях глобальних мереж трафік є сумою трафіків

багатьох джерел, що за законом великих чисел призводить до згладжування результуючого трафіка.

Чутливість трафіка до затримок пакетів

Ще один критерій класифікації додатків за типом трафіка – їхня чутливість до затримок пакетів і їхніх варіацій. Далі перелічені основні типи додатків у порядку підвищення чутливості до затримок пакетів.

• ***Асинхронні додатки***. Практично не мають обмежень на час затримки (еластичний трафік). Приклад такого додатка – електронна пошта.

• ***Інтерактивні додатки***. Затримки можуть бути позначені користувачами, але вони не впливають негативно на функціональність додатків. Приклад – текстовий редактор, що працює з віддаленим файлом.

• ***Ізохронні додатки***. Мають поріг чутливості до варіацій затримок, при перевищенні якого різко знижується функціональність додатка. Приклад – передача голосу, коли при перевищенні порога варіації затримок в 100–150 мс різко знижується якість відтвореного голосу.

• ***Надчутливі до затримок додатки.*** Затримка доставки даних зводить функціональність додатка до нуля. Приклад – додатки, які керують технічним об'єктом у реальному часі. При запізнюванні керуючого сигналу на об'єкті може відбутися аварія.

Врешті-решт, інтерактивність додатка завжди підвищує його чутливість до затримок. Наприклад, широкомовне розсилання аудіоінформації може витримувати значні затримки передачі пакетів (залишаючись чутливим до варіацій затримок), а інтерактивна телефонна або телевізійна розмова їх не терпить, що добре помітно під час трансляції розмови через супутник. Тривалі паузи в розмові вводять співрозмовників в оману, часто вони втрачають терпіння й починають чергову фразу одночасно.

Поряд з наведеною класифікацією, яка тонко диференціює чутливість додатків до затримок і їхніх варіацій, існує й більш грубий розподіл додатків за цією самою ознакою на два класи: *асинхронні* й *синхронні*.

До асинхронних належать ті додатки, які нечутливі до затримок передачі даних у дуже широкому діапазоні, аж до декількох секунд, а всі інші додатки, на функціональність яких затримки впливають суттєво, належать до синхронних.

Інтерактивні додатки можуть належати як до асинхронних (текстовий редактор), так і до синхронних (відеоконференція).

Чутливість трафіка до втрат і викривлення пакетів

Зрештою, останнім критерієм класифікації додатків є їхня чутливість до втрат пакетів. Тут зазвичай поділяють додатки на дві групи:

• ***Додатки, чутливі до втрати даних***. Практично всі додатки, що передають алфавітно-цифрові дані (до яких належать текстові документи, коди програм, числові масиви тощо), мають високу чутливість до втрати окремих, навіть невеликих, фрагментів даних. Такі втрати часто призводять до повного знецінювання іншої успішно прийнятої інформації. Наприклад, відсутність хоча б одного байта в коді програми робить її зовсім непрацездатною. Усі традиційні мережні додатки (файловий сервіс, сервіс баз даних, електронна пошта тощо) належать до цього типу додатків.

• ***Додатки, стійкі до втрати даних***. До цього типу належить багато додатків, які передають трафік з інформацією про інерційні фізичні процеси. Стійкість до втрат пояснюється тим, що невелику кількість відсутніх даних можна визначити на основі прийнятих. Так, при втраті одного пакета, що несе кілька послідовних вимірів голосу, відсутні виміри в ході відтворення голосу можуть бути замінені апроксимацією на основі сусідніх значень.

До такого типу належить більша частина додатків, які працюють з мультимедійним трафіком (аудіо- і відеододатки). Однак стійкість до втрат має свої межі, тому відсоток загублених пакетів не може бути великим (наприклад, не більше 1%). Можна зазначити також, що не будь-який мультимедійний трафік настільки стійкий до втрат даних. Так, компресований голос і відеозображення дуже чутливі до втрат, тому належать до першого типу додатків.

Класи додатків

Загалом, між значеннями трьох характеристик якості обслуговування (відносна передбачуваність швидкості передачі даних; чутливість трафіка до затримок пакетів; чутливість трафіка до втрат і викривлення пакетів) немає суворого взаємозв’язку. Тобто додаток з рівномірним потоком може бути як асинхронним, так і синхронним, а, наприклад, синхронний додаток може бути як чутливим, так і нечутливим до втрат пакетів. Однак практика показує, що з усього різноманіття можливих комбінацій значень цих трьох характеристик є небагато таких, що охоплюють велику частину додатків, які використовуються сьогодні.

Наприклад, наступна комбінація характеристик додатка «породжуваний трафік – рівномірний потік, додаток ізохронний, стійкий до втрат», відповідає таким популярним додаткам, як IP-телефонія, підтримка відеоконференцій,

аудіомовлення через Інтернет. Стійких комбінацій характеристик, які описують певний клас додатків, існує не так уже й багато. Так, під час стандартизації технології ATM, яка від самого початку розроблялася для підтримки різних типів трафіка, було визначено 4 класи трафіка (і відповідних додатків): A, B, C і D. Для кожного класу рекомендується використовувати власний набір характеристик QoS. Крім того, для всіх додатків, що не входять ні в один із цих класів, був визначений клас X, у якому комбінація характеристик додатка може бути довільною.

Класифікація ATM, що є на сьогодні найбільш детальною та узагальненою, не вимагає для свого розуміння знання технології ATM, тому вже може бути розглянута (табл. 1.1).

Таблиця 1.1 – Класи трафіка

Клас трафіка	**Характеристики**
A	Постійна бітова швидкість, чутливість до затримок, передача зі встановленням з'єднання (голосовий трафік, трафік телевізійного зображення). Параметра QoS: пікова швидкість передачі даних, затримка, джиттер
B	Змінна бітова швидкість, чутливість до затримок, передача зі встановленням з'єднання (компресований голос, компресоване відеозображення). Параметра QoS: пікова швидкість передачі даних, пульсація, середня швидкість передачі даних, затримка, джиттер
C	Змінна бітова швидкість, еластичність, передача зі встановленням з'єднання (трафік комп'ютерних мереж, у яких кінцеві вузли працюють за протоколами зі встановленням з'єднань – frame relay, X.25, TCP). Параметра QoS: пікова швидкість передачі даних, пульсація, середня швидкість передачі даних
D	Змінна бітова швидкість, еластичність, передача без встановлення з'єднання (наприклад, трафік комп'ютерних мереж, у яких кінцеві вузли працюють за протоколами без встановлення з'єднань – IP/UDP, Ethernet). Параметра QoS: не визначені
X	Тип трафіка і його параметри визначаються користувачем

1.3 Аналіз черг на основі моделі масового обслуговування

Визначити основні характеристики QoS і сформулювати вимоги до них – означає наполовину розв'язати задачу. Користувач формулює свої вимоги до якості обслуговування у вигляді деяких граничних значень

характеристик QoS, які не можуть бути перевищені. Наприклад, він може вказати, що граничне значення варіації затримки пакетів не має перевищувати 50 мс з імовірністю 0,99.

Але як змусити мережу впоратися з поставленим завданням? Які заходи слід вжити, щоб варіації затримок дійсно не перевищили цю величину? І як гарантувати користувачу, що середня швидкість передачі його потоку через мережу відповідатиме середній швидкості вхідного потоку мережі?

Для розуміння механізмів підтримки QoS корисно досліджувати процес утворювання черг у мережних пристроях і зрозуміти найістотніші фактори, що впливають на довжину черги.

Модель M/M/1

Існує галузь прикладної математики, предметом якої є процеси утворення черг. Ця дисципліна так і називається – теорія черг. Ми не занурюватимемося в математичні основи цієї теорії, наведемо лише деякі її висновки, істотні для розглянутої нами проблеми QoS.

На рис. 1.3 показана найпростіша модель черги, відома за назвою M/M/1.

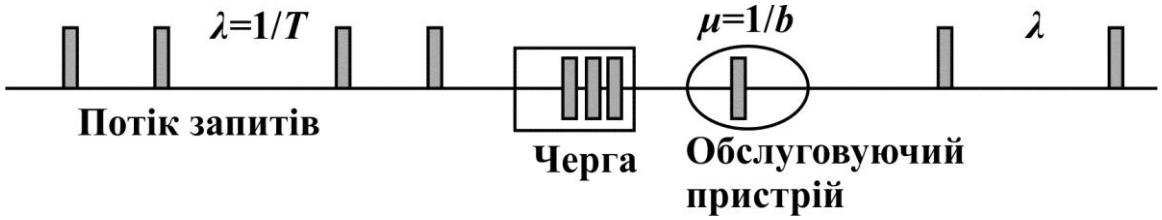


Рисунок 1.3 – Модель M/M/1

Основними елементами моделі є:

• вхідний потік абстрактних заявок на обслуговування;

• буфер;

• обслуговуючий пристрій;

• вихідний потік обслужених заявок.

Заявки надходять на вхід буфера у випадкові моменти часу. Якщо у момент надходження заявки буфер порожній і обслуговуючий пристрій вільний, то заявка відразу ж передається у цей пристрій для обслуговування. Обслуговування також триває невизначений час.

Якщо у момент надходження заявки буфер порожній, але обслуговуючий пристрій зайнятий обслуговуванням заявки, що прийшла раніше, то заявка очікує його завершення в буфері.

Як тільки обслуговуючий пристрій завершує обслуговування чергової заявки, вона передається на вихід, а пристрій вибирає з буфера наступну заявку

(якщо буфер не порожній). Заявки, що виходять з обслуговуючого пристрою, утворюють вихідний потік. Буфер вважається нескінченним, тобто заявки ніколи не губляться через те, що вичерпаний обсяг буфера.

Якщо заявка, яка прибула, застає буфер непорожнім, то вона стає в чергу й очікує обслуговування. Заявки вибираються з черги в порядку надходження, тобто дотримується дисципліна обслуговування «першим прийшов – першим обслужений» (First-In, First-Out, FIFO).

Теорія черг дозволяє оцінити для цієї моделі середню довжину черги та середній час очікування заявки в черзі залежно від характеристик вхідного потоку й часу обслуговування.

Вважатимемо, що середній час між надходженнями заявок відомий і дорівнює T. Це означає, що інтенсивність надходження заявок, яка традиційно позначається в теорії черг символом X, дорівнює

$$X = \frac{1}{T} \ \textit{заявок за секунду}.$$

Випадковий процес надходження заявок описується у цій моделі функцією розподілу інтервалів між надходженнями заявок. Щоб спростити отримання компактних аналітичних результатів, зазвичай вважають, що ці інтервали описуються так званим марківським розподілом (інша назва – пуассонівський розподіл), щільність якого показана на рис. 1.4. З рисунка видно, що вхідний потік є суттєво пульсуючим, оскільки є ненульова імовірність того, що інтервал між заявками буде дуже невеликим, близьким до нуля, а також того, що він буде дуже великим. Середнє відхилення інтервалів також дорівнює T.

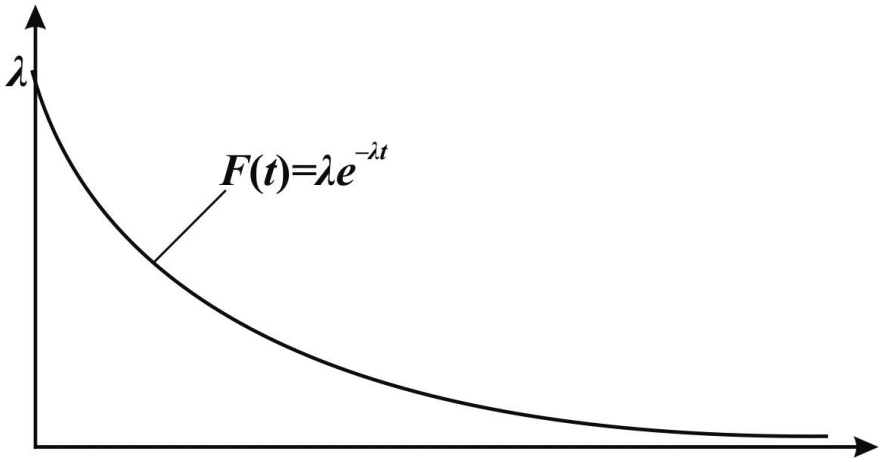


Рисунок 1.4 – Щільність розподілу вхідного потоку

Також вважатимемо, що середній час обслуговування заявки дорівнює b. Це означає, що обслуговуючий пристрій здатний просувати заявки на вихід

з інтенсивністю $1/b = p$. Знову ж для отримання аналітичного результату вважають, що час обслуговування – це випадкова величина з пуассонівською щільністю розподілу.

Прийняття таких припущень дає простий результат для середнього часу очікування заявки в черзі, який ми позначимо через w:

$$w = p\frac{b}{1-p} \ .$$

Тут через p позначено відношення λ/μ.

Параметр p називають коефіцієнтом використання (utilization) обслуговуючого пристрою. Для будь-якого періоду часу цей показник дорівнює відношенню часу зайнятості обслуговуючого пристрою до величини цього періоду.

Залежність середнього часу очікування заявки w від p ілюструє рис. 1.5.

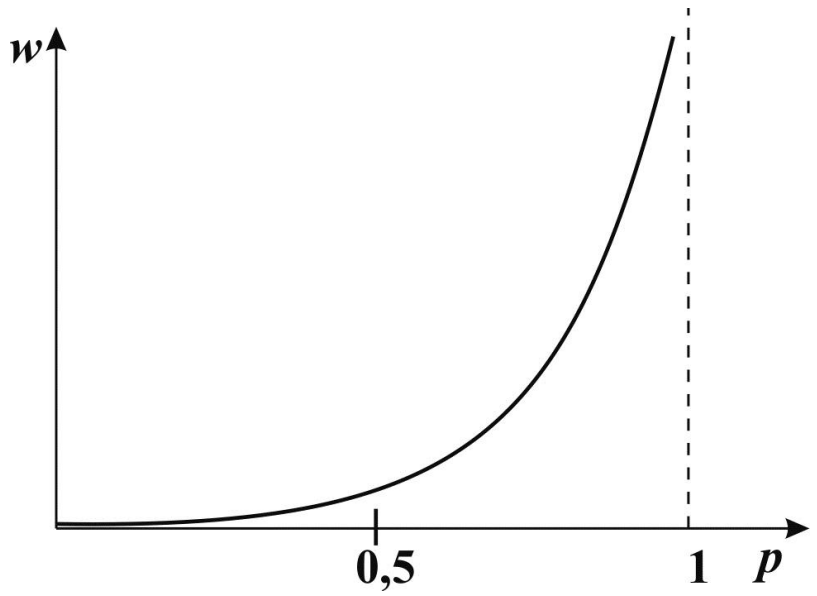


Рисунок 1.5 – Залежність середнього часу очікування заявки від коефіцієнта використання ресурсу

Як видно з поведінки кривої, параметр p відіграє ключову роль у створенні черги. Якщо значення p близьке до нуля, то середній час очікування також дуже близький до нуля. А це означає, що заявки майже ніколи не очікують обслуговування в буфері (у момент їх надходження він виявляється порожнім), а відразу попадають в обслуговуючий пристрій. І навпаки, якщо p наближається до 1, то час очікування зростає дуже швидко й нелінійно (і в межі дорівнює нескінченності). Така поведінка черги інтуїтивно зрозуміла, адже p – це відношення середньої інтенсивності вхідного потоку до середньої

інтенсивності його обслуговування. Чим ближче середні значення інтервалів між пакетами до середнього часу обслуговування, тим складніше обслуговуючому приладу справлятися з навантаженням.

За допомогою моделі $M/M/1$ можна приблизно моделювати мережу з комутацією пакетів (рис. 1.6).

Так, вхідний потік пакетів, що надходять на вхід інтерфейса комутатора (тут використовуватимемо цей термін як узагальнену назву пристрою комутації пакетів), зображений у моделі потоком заявок, а буфер моделі відповідає буферу інтерфейса комутатора. Середній час обслуговування заявки відповідає середньому часу просування пакета процесором комутатора із вхідного буфера у вихідний канал.

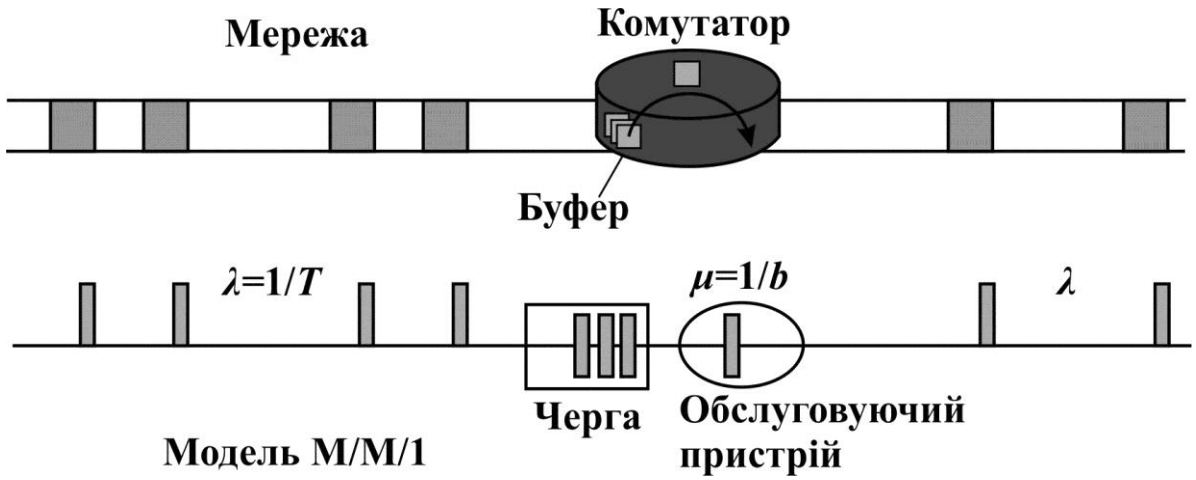


Рисунок 1.6 – Вихідний інтерфейс як поділюваний ресурс комутатора

Зрозуміло, що наведена модель дуже спрощено описує процеси, які відбуваються в комутаторі. Проте, вона дуже корисна для розуміння основних факторів, що впливають на величину черги.

Мережні інженери добре знайомі з графіком, зображеним на рис. 1.5. Вони інтерпретують цей графік як залежність затримок у мережі від її завантаження. Параметр p моделі відповідає коефіцієнту використання мережного ресурсу, який бере участь у передачі трафіка, тобто вихідного інтерфейса комутатора.

У наведеному графіку є й дещо несподіване. Важко уявити, що обслуговуючий пристрій (мережний ресурс) практично перестає справлятися зі своїми обов’язками, коли його коефіцієнт використання наближається до 1. Адже в цьому випадку навантаження не перевищує його можливостей, а лише наближається до цієї межі. Інтуїтивно не дуже зрозуміла також причина

існування черг при значеннях p навколо 0,5. Інтенсивність обробки трафіка вдвічі перевищує інтенсивність навантаження, а черги існують!

Такі парадоксальні, на перший погляд, результати характерні для систем, у яких протікають випадкові процеси. Оскільки λ і μ – це середні значення інтенсивностей потоків на великих проміжках часу, то на невеликих проміжках часу вони можуть суттєво відхилятися від цих значень. Черга створюється на тих проміжках, де інтенсивність надходження пакетів значно перевершує інтенсивність обслуговування.

Перевантаження ресурсів може призвести до повної деградації мережі, коли, незважаючи на те, що мережа передає пакети, корисна швидкість передачі даних виявляється рівною нулю. Це відбувається у тому випадку, коли затримки доставки всіх пакетів перевершують деякий поріг, і пакети по тайм-ауту відкидаються вузлом призначення як застарілі. Якщо ж протоколи, що працюють у мережі, використовують надійні процедури передачі даних на основі квитування й повторної передачі загублених пакетів, то процес перевантаження зростатиме лавиноподібно.

Існує ще один важливий параметр, який має безпосередній вплив на створення черг у мережах з комутацією пакетів. Цим параметром є варіація інтервалів вхідного потоку пакетів, тобто пульсація вхідного трафіка. Ми аналізували поведінку моделі теорії черг у припущенні, що вхідний потік описується пуассонівським розподілом, який має досить велике стандартне відхилення варіації (нагадаємо, що середня варіація його дорівнює T при середньому значенню інтервалу T, а коефіцієнт варіації дорівнює 1). А що буде, якщо варіація інтервалів вхідного потоку буде менше? Чи вхідний потік виявиться надпульсуючим?

На жаль, моделі теорії черг не дають для цих випадків простих аналітичних залежностей. Тому для отримання результатів доводиться застосовувати методи імітаційного моделювання мереж або проводити вимірювання в реальній мережі.

На рис. 1.7 показано сімейство залежностей w від p, отриманих для різних значень коефіцієнта варіації CV вхідного потоку.

Імітаційна модель враховує фіксовану затримку в мережі. Одна з кривих, у якої $CV = 1$, відповідає пуассонівському вхідному потоку. З рисунка видно, що чим менше пульсує вхідний потік (CV наближається до нуля), тим менше проявляється ефект лавиноподібного утворювання черги при наближенні коефіцієнта завантаження ресурсу до 1. І навпаки, чим більше CV, тим раніше (за менших значень p) починає цей ефект проявлятися.

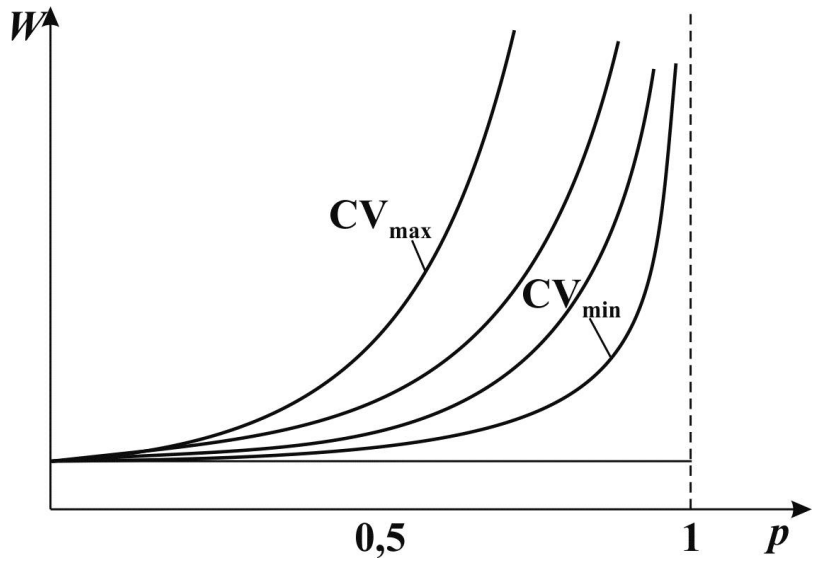


Рисунок 1.7 – Вплив ступеня пульсації потоку на затримки

З поведінки графіків на рисунку можна зробити два висновки:

– по-перше, для оцінки значень затримок у чергах на комутаторах мережі недостатньо інформації про коефіцієнт завантаження p, необхідно також знати параметри пульсації трафіка;

– по-друге, для зниження рівня затримок потрібно знижувати значення p й зменшувати пульсацію трафіка.

Черги та різні класи трафіка

Подивимося, як можна застосувати наші знання про залежність поведінки черг від коефіцієнта завантаження для реалізації основної ідеї методів QoS, а саме диференційованого обслуговування класів трафіка з різними вимогами до характеристик продуктивності та надійності мережі. Щоб простіше було в цьому розібратися, розділимо всі потоки на два класи – чутливий до затримок (трафік реального часу, наприклад, голосовий) і еластичний, який допускає більші затримки, але чутливий до втрат даних.

Ми знаємо, – якщо забезпечити для чутливого до затримок трафіка коефіцієнт завантаження кожного ресурсу не більше 0,2, то, мабуть, затримки у кожній черзі будуть невеликими й, швидше за все, прийнятними для багатьох типів додатків цього класу.

Для еластичного трафіка, що слабко чутливий до затримок, можна допустити більш високий коефіцієнт завантаження, але не більший 0,9. Для того щоб пакети цього класу не губилися, слід передбачити для них буферну пам’ять, достатню для зберігання всіх пакетів періоду пульсації. Ефект від такого розподілу завантаження ресурсу ілюструє рис. 1.8.

Затримки чутливого до затримок трафіка дорівнюють w_s, а затримки еластичного трафіка – w_e.

Щоб досягти різних коефіцієнтів використання ресурсів для різних класів трафіка, потрібно в кожному комутаторі для кожного ресурсу підтримувати дві різні черги. Алгоритм вибірки пакетів з черг має віддавати перевагу черзі чутливих до затримок пакетів. Якби всі пакети першої черги обслуговувалися пріоритетно, а пакети другої черги – тільки тоді, коли перша черга порожня, то для трафіка першої черги трафік другої черги фактично перестав би існувати. Тому, якщо відношення середньої інтенсивності пріоритетного трафіка λ_1 до продуктивності ресурсу μ дорівнює 0,2, то й коефіцієнт завантаження для нього дорівнює 0,2. А от для еластичного трафіка, пакети якого завжди очікують обслуговування пріоритетних пакетів, коефіцієнт завантаження підраховується інакше. Якщо середня інтенсивність еластичного трафіка дорівнює λ_2, то для нього ресурс буде завантажений на $(\lambda_1 + \lambda_2)/\mu$. Тому, якщо ми бажаємо, щоб для еластичного трафіка коефіцієнт завантаження становив 0,9, то його інтенсивність має випливати зі співвідношення $\lambda_2/\mu \sim 0,7$.

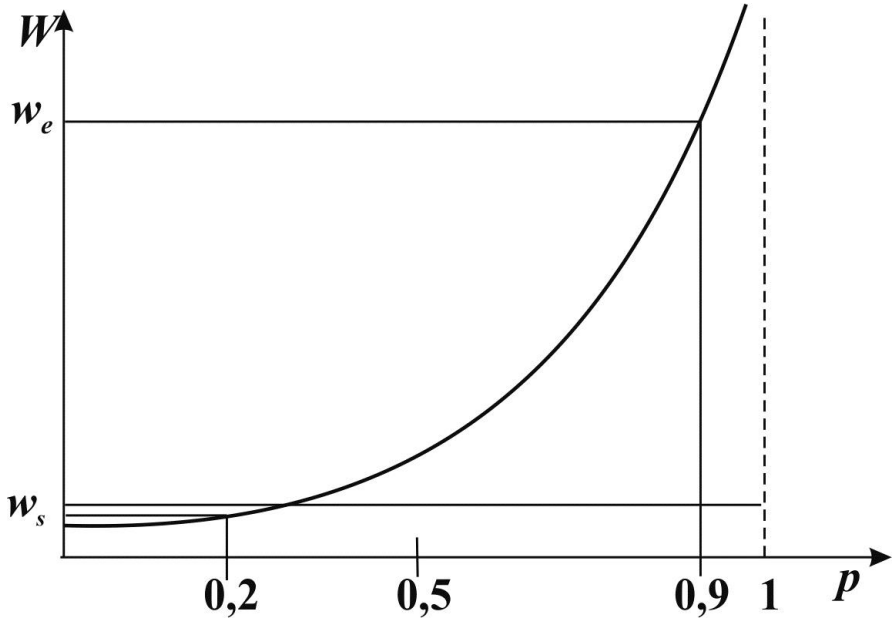


Рисунок 1.8 – Обслуговування еластичного й чутливого до затримок трафіка

Основна ідея, яка лежить в основі всіх методів підтримки характеристик QoS, полягає у тому, що: загальна продуктивність кожного ресурсу має бути розділена між різними класами трафіка нерівномірно.

Можна ввести більш ніж два класи обслуговування й намагатися, щоб кожний клас працював на своїй частині кривої затримок. Якщо таке завдання

вирішено, то можна забезпечити покращення характеристик QoS за рахунок інших методів, наприклад, знижуючи пульсацію трафіка.

Залишилося з'ясувати, як можна забезпечити такі умови для різних класів трафіка у кожному вузлі мережі.

1.4 Контрольні запитання та завдання

1. Назвіть основні характеристики якості обслуговування в мережах з передачею пакетів.
2. Назвіть основні механізми забезпечення якості обслуговування в мережах з передачею пакетів.
3. Дайте класифікацію трафіків додатків.
4. Що таке коефіцієнт пульсацій?
5. Дайте класифікацію додатків за чутливістю до затримок пакетів.
6. Назвіть класи додатків.
7. Дайте характеристику моделі $M/M/1$.
8. Які параметри передачі впливають на формування черг у мережах з комутацією пакетів.

Лекція 2
АНАЛІЗ МЕРЕЖ НА ОСНОВІ МОДЕЛЕЙ МАСОВОГО ОБСЛУГОВУВАННЯ

2.1 Теорема Літтла

Однією з найважливіших характеристик мережі передачі даних є середня затримка, необхідна для доставки пакета від джерела до місця призначення. Більш того, результати дослідження затримки дуже впливають на вибір і роботу мережних алгоритмів, таких, як алгоритми маршрутизації і алгоритм керування потоком. З цієї причини важливо зрозуміти природу і механізм затримки, а також те, як затримка залежить від пропускної здатності мережі.

Головною методологічною основою для аналізу затримки мережі є теорія масового обслуговування. Однак її використання часто вимагає спрощення припущень, оскільки, на жаль, більш реалістичні припущення надзвичайно ускладнюють змістовний аналіз. З цієї причини в деяких випадках неможливо провести точні кількісні розрахунки затримки на основі моделей теорії масового обслуговування. Проте, ці моделі часто є основою для розумних апроксимацій затримки, а також дозволяють отримати якісні результати.

Надалі ми зосередимо увагу на затримці пакета всередині підмережі зв'язку (тобто на мережному рівні). Ця затримка є сумою затримок на кожній лінії підмережі, через яку проходить пакет. Кожна затримка на лінії у свою чергу складається з чотирьох компонент.

Затримка на обробку – затримка між моментом, коли пакет був правильно прийнятий на початковому вузлі лінії, і моментом, коли пакет був поставлений у чергу на передачу по лінії. (У деяких системах слід додати до цієї затримки деякий додатковий час, що витрачається на обробку на рівнях УЛПД і фізичному).

Затримка в черзі – затримка між моментом, коли пакет був поставлений у чергу на передачу, і моментом, коли він починає передаватися. Протягом цього часу пакет очікує, поки будуть передані інші пакети з черги.

Затримка передачі – затримка між моментами, коли передадуться перший і останній біти пакета.

Час поширення – проміжок часу від моменту, коли останній біт був переданий на початковому вузлі лінії, до моменту, коли він буде прийнятий у остаточному вузлі цієї лінії. Ця затримка пропорційна фізичній відстані

між передавачем та приймачем, і зазвичай мала, за винятком випадку супутникової лінії зв’язку.

Під час такого розгляду не враховується можливість того, що може знадобитися повторна передача по лінії через похибки передачі або з будь-яких інших причин. Для більшості реальних ліній зв’язку, за винятком ліній множинного доступу, повторні передачі бувають рідко, і вони не розглядатимуться. Час поширення залежить від фізичних характеристик лінії зв’язку і не залежить від потоку, що передається по лінії. Затримка обробки також не залежить від величини потоку, який оброблюється у відповідному вузлі, якщо обчислювальна потужність вузла не обмежена. Ми приймемо це припущення для зручності. В іншому випадку окрема черга на обробку має бути введена перед чергами на передачу. Значна частина подальшого аналізу зосереджена нами на затримках у черзі і затримках передачі. Спочатку ми розглянемо єдину лінію зв’язку і проаналізуємо деякі класичні моделі теорії масового обслуговування. Потім розглянемо мережу і обговоримо ті наближення, які допускаються, коли обираються аналітичні моделі для розрахунку затримки.

Хоча головний акцент робиться на моделі мережі з комутацією пакетів, деякі розроблені моделі придатні також для мережі з комутацією каналів. Теорія масового обслуговування широко розвивалася виходячи з потреб створення математичних моделей для телефонії.

Розглянемо системи масового обслуговування, в яких вимоги на обслуговування з’являються у випадкові моменти часу. Розподіл імовірностей часу між двома моментами появи вимог, що прямують один за одним, і розподіл ймовірностей часу обслуговування вимоги задані.

Стосовно мереж передачі даних вимоги є пакетами, які призначені для передачі по лінії зв’язку. Час обслуговування відповідає часу передачі пакета й дорівнює L/C, де L – довжина пакета в бітах, а C – пропускна здатність лінії в біт/с. У цій лекції зручно не враховувати розходження між пакетами й кадрами, що є на рівні 2; тому включимо в довжини пакетів заголовки й трейлери кадрів. У деяких інших застосуваннях (на яких ми не зупинятимемося докладно) вимоги є активним обміном інформацією (або віртуальні ланцюги) між точками в мережі, а час обслуговування відповідає тривалості обміну.

Зазвичай ми цікавитимемося оцінкою таких величин:

1. Середня кількість вимог у системі (тобто типова кількість вимог, які або очікують у черзі, або обслуговуються).

2. Середня затримка вимоги (тобто «типовий» час, що вимога проводить, очікуючи в черзі, плюс час обслуговування).

Спочатку необхідно пояснити наведені терміни. Введемо позначення $p_n(t)$ – імовірність того, що n вимог перебуває в черзі або на обслуговуванні в момент t. Типовою є ситуація, коли дані початкові ймовірності $p_n(0)$ в момент 0 і є досить статистичною інформацією для визначення, принаймні в принципі, імовірностей $p_n(t)$ для всіх моментів t. Увівши позначення $\overline{N}(t)$ – середню кількість вимог у системі в момент t, маємо

$$\overline{N}(t) = \sum_{n=0}^{\infty} n p_n(t).$$

Зазначимо, що як $\overline{N}(t)$, так і $p_n(t)$ залежать від t і початкового розподілу ймовірностей $\{p_0(0), p_1(0), \ldots\}$. Однак системи масового обслуговування, які розглядатимуться, зазвичай досягають рівноваги в тому розумінні, що для деяких p_n і N (незалежно від початкового розподілу)

$$\lim_{t\to\infty} p_n(t) = p_n, \quad n = 0, 1, \ldots \tag{2.1}$$

і

$$N = \sum_{n=0}^{\infty} n p_n(t) = \lim_{t\to\infty} \overline{N}(t).$$

Насамперед нас цікавитимуть рівноважні ймовірності й середня кількість вимог у системі. Зазначимо, що, можливо, $N = \infty$. Це справедливо, коли швидкість надходження вимог перевищує можливості обслуговування в системі. Функції реалізації кількості вимог у системі позначатимуться через $N(t)$. Функція реалізації, усереднена за часом на відрізку $[0,\ t]$, позначається як

$$N_t = \frac{1}{t}\int_0^t N(\tau)\, d\tau.$$

Майже будь-яка система, що цікавить нас, є ергодичною у тому розумінні, що

$$\lim_{t\to\infty} N_t = \lim_{t\to\infty} \overline{N}(t) = N$$

з імовірністю 1. Рівність середнього за великим інтервалом часу й середнім за реалізаціями різних випадкових процесів прийматиметься в цій лекції на інтуїтивному рівні, оскільки під час чіткого математичного доведення використовуються викладення, які лежать за межами даного матеріалу.

Що стосується середньої затримки вимоги, то достатньо статистичної інформації, щоб визначити в принципі розподіл імовірностей затримки кожної

конкретної вимоги (тобто першої, другої й т.д.). Отже, можна визначити середню затримку кожної вимоги. Середня затримка k-ї вимоги, яка позначається через $\overline{T}_k$, зазвичай сходиться при $k \to \infty$ до стаціонарного значення

$$T = \lim_{k\to\infty} \overline{T}_k .$$

Зазначену межу називатимемо середньою затримкою вимоги. (Знову можливо, що $T = \infty$.) Для систем, які цікавлять нас, стаціонарна середня затримка T також дорівнює (з імовірністю одиниця) затримці вимоги, усередненої за великим інтервалом часу, тобто

$$T = \lim_{k\to\infty} \overline{T}_k = \lim_{k\to\infty} \frac{1}{k} \sum_{i=1}^{k} T_i ,$$

де T_i – затримка i-ї вимоги.

Середня кількість вимог у системі N і середня затримка T пов'язані простою формулою, що дає можливість виражати одне середнє через інше. Цей результат, відомий як теорема Літтла, має такий вигляд

$$N = \lambda T ,$$

де λ – середня швидкість надходження вимог визначається як математичне очікування кількості вимог, що надійшли в інтервалі

$$\lambda = \lim_{t\to\infty} \frac{\begin{array}{c}\textit{математичне очікування кількості вимог,}\\ \textit{що надійшли в інтервалі } [0,\ t]\end{array}}{t} .$$

(Припустимо, що зазначена межа існує.) Приклади, які відображають теорему Літтла, зустрічаються у повсякденному житті. Наприклад, у дощовий день потік машин у годину пік рухається повільніше, ніж у середньому (велике T), тому на вулицях більше машин (велике N). Аналогічно ресторан швидкого обслуговування (невелике T) має потребу в меншому приміщенні (мале N), ніж звичайний ресторан, за однаковою швидкістю надходження вимог.

Значущість теореми Літтла полягає в її великій спільності. Вона справедлива майже для всіх систем масового обслуговування, які в межі $\left(\lim_{t\to\infty}\right)$ досягають статистичної рівноваги. Система необов'язково повинна мати тільки одну чергу. Теорема справедлива для багатьох складних систем, до яких щось надходить і щось виходить, за відповідної інтерпретації величин N, λ і T. Наведені далі приклади ілюструють широкий діапазон її застосувань.

Приклад

Якщо λ – швидкість надходження вимог до лінії передачі, N_Q – середня кількість пакетів, що очікують у черзі (але не передаються), W – середній час, упродовж якого пакет очікує в черзі (не враховуючи час передачі), то, відповідно до теореми Літтла, маємо

$$N_Q = \lambda W.$$

Більш того, якщо $\overline{X}$ – середній час передачі, то, відповідно до теореми Літтла, отримуємо, що середня кількість пакетів, які знаходяться в процесі передачі, дорівнюють

$$p = \lambda \overline{X}.$$

Оскільки передаватися може лише один пакет, отримуємо, що p також дорівнює коефіцієнту використання лінії, тобто частці часу, протягом якої лінія передає пакети.

2.2 Аналіз черг обслуговування на основі моделі ланцюга Маркова

Система масового обслуговування $M/M/1$ – це система з однією чергою й одним обслуговуючим пристроєм (у контексті систем зв'язку – з однією лінією зв'язку). Моменти надходження вимог утворюють пуассонівський потік з інтенсивністю λ, а розподіл імовірностей часу обслуговування є експоненційним з середнім значенням, що дорівнює $1/\lambda$ секунд. Стисло пояснимо зміст прийнятих позначень. Позначення $M/M/1$ є стандартним для теорії масового обслуговування.

1. Перший символ вказує на природу процесу надходження вимог (наприклад, M позначає процес без запам'ятовування, який тут називається пуассонівським процесом (у ньому інтервали між надходженнями вимог розподілені експоненціально), G позначає довільний розподіл інтервалів між вимогами, D – детерміновані інтервали).

2. Другий символ вказує на природу розподілу ймовірностей тривалості обслуговування (наприклад, M, G, D – позначення для експоненціального, довільного і детермінованого розподілів відповідно). В усіх випадках послідовні інтервали між надходженнями вимог і тривалості обслуговування вважаються статистично незалежними.

3. Останнє число вказує на кількість обслуговуючих приладів.

Ми вже встановили з теореми Літтла співвідношення між основними величинами:

$$N = \lambda T, \quad N_Q = \lambda W ,$$

N – середня кількість вимог у системі,

T – середній час перебування вимоги в системі,

N_Q – середня кількість вимог, що очікують у черзі,

W – середній час, протягом якого вимога очікує в черзі.

Однак величини N, T, N_Q і W не можуть бути знайдені, доки ми не дізнаємося дещо більше про статистичні характеристики системи. Використовуючи ці характеристики, можна визначити стаціонарну вірогідність p_n – ймовірність перебування вимог у системі, $n = 0, 1, \ldots$

За допомогою цих імовірностей можна отримати

$$N = \sum_{n=0}^{\infty} n p_n ,$$

а застосування теореми Літтла дає

$$T = N/\lambda .$$

Є аналогічні формули для N_Q і W, які розглядаються далі.

Випадковий процес $\{A(t) | t \geq 0\}$, який бере невід'ємні цілі значення, називається пуассонівським процесом з інтенсивністю λ, якщо:

1) $A(t)$ – процес, який дає сумарну кількість моментів надходжень за годину від 0 до t, тобто $A(0) = 0$ і при $s < t$ величина $A(t) - A(s)$ дорівнює кількості моментів надходжень на інтервалі $(s, t]$;

2) кількість моментів надходжень на непересічних часових інтервалах є незалежною;

3) кількість моментів надходжень у будь-якому інтервалі тривалості τ має пуассонівський розподіл з параметром $\lambda\tau$, тобто для будь-яких $t, \tau > 0$

$$P\{A(t+\tau) - A(t) = n\} = e^{-\lambda\tau} \frac{(\lambda\tau)^n}{n!}, \quad n = 0, 1, \ldots \qquad (2.2)$$

Перелічимо деякі цікаві для нас властивості пуассонівського процесу:

I. Інтервали між моментами надходжень незалежні і розподілені експоненційно з параметром λ, тобто, якщо позначити через момент надходження i-ї вимоги, то інтервали часу $\tau_n = t_{n+1} - t_n$ мають функцію розподілу ймовірностей.

Перелічимо деякі властивості пуассонівського процесу, які нас цікавлять, і взаємно незалежні

$$P\{\tau_n \le s\} = 1 - e^{-\lambda s}, \quad s \ge 0 \ . \tag{2.3}$$

(Відповідна щільність ймовірності $p(\tau_n) = \lambda e^{-\lambda \tau_n}$. Середнє значення і дисперсія τ_n величини дорівнюють $1/\lambda$ і $1/\lambda^2$ відповідно.)

II. Для будь-яких $t \ge 0$ та $\delta \ge 0$

$$P\{A(t+\delta) - A(t) = 0\} = 1 - \lambda\delta + o(\delta), \tag{2.4}$$

$$P\{A(t+\delta) - A(t) = 0\} = \lambda\delta + o(\delta), \tag{2.5}$$

$$P\{A(t+\delta) - A(t) \ge 2\} = o(\delta), \tag{2.6}$$

де використовується загальне позначення $o(\delta)$ для будь-якої функції від δ, такої, що

$$\lim_{\delta \to 0} \frac{o(\delta)}{\delta} = 0 .$$

Наше припущення щодо процесу обслуговування полягає в тому, що тривалості обслуговування вимог мають експоненційний розподіл з параметром μ, тобто, якщо s_n – час обслуговування n-ї вимоги, то

$$P\{s_n \le s\} = 1 - e^{-\mu s}, \quad s \ge 0 .$$

(Щільність розподілу s_n дорівнює $p(s_n) = \mu e^{-\mu s_n}$, а середнє значення і дисперсія дорівнюють $1/\mu$ і $1/\mu^2$ відповідно.) Крім того, тривалості обслуговування s_n взаємно незалежні, а також не залежать від усіх інтервалів між моментами надходження вимог. Параметр μ називається швидкістю обслуговування й показує, з якою швидкістю працює обслуговуючий пристрій (тобто кількість вимог, що обслуговуються в одиницю часу), якщо він зайнятий.

Ми могли б розглядати процес $N(t)$ у термінах теорії ланцюгів Маркова з неперервним часом (як у більшій частині літератури з систем масового обслуговування). Але для наших цілей тут достатньо використовувати простішу теорію ланцюгів Маркова з дискретним часом. Звернемо увагу на такі моменти

$$0, \delta, 2\delta, \ldots, k\delta, \ldots,$$

де δ – мале позитивне число.

Введемо позначення:

N_k – кількість вимог у системі в момент $k\delta$.

Оскільки $N_k = N(k\delta)$, а як уже було сказано, $N(t)$ – ланцюг Маркова з неперервним часом, отримуємо, що $\{N_k | k = 0, 1, \ldots\}$ – ланцюг Маркова з дискретним часом.

Позначимо через P_{ij} відповідні перехідні ймовірності

$$P_{ij} = P\{N_{k+1} = j | N_k = i\}.$$

Зауважимо, що, P_{ij} залежить від δ, але, для того щоб зберегти прості позначення, ми не враховуватимемо цю залежність. Використовуючи формули (2.4) – (2.6), отримуємо

$$P_{00} = 1 - \lambda\delta + o(\delta), \tag{2.7}$$

$$P_{ii} = 1 - \lambda\delta - \mu\delta + o(\delta), \quad i \geq 1, \tag{2.8}$$

$$P_{i,i+1} = \lambda\delta - \mu\delta + o(\delta), \quad i \geq 0, \tag{2.9}$$

$$P_{i,i-1} = \mu\delta + o(\delta), \quad i \geq 1, \tag{2.10}$$

$$P_{ij} = o(\delta), \quad i \text{ та } j \neq i, i+1, i-1.$$

Для того щоб зрозуміти, як знаходяться ці рівняння, зауважимо, що в стані $i \geq 1$ ймовірність того, що в інтервалі $I = I_k = (k\delta, (k+1)\delta)$ не надійде жодна вимога і жодна вимога не піде з системи, дорівнює $(e^{-\lambda\delta})(e^{-\mu\delta})$. Це випливає з того, що кількість надходжень і кількість виходів мають пуассонівский розподіл і не залежать одне від одного. Розкладаючи цю ймовірність у ряд за δ, отримуємо

$$P\{\text{в інтервалі } I_k \text{ 0 вимог надійшло та 0 відійшло}\} = 1 - \lambda\delta - \mu\delta + o(\delta). \tag{2.11}$$

Аналогічно маємо, що

$$P\{\text{в інтервалі } I_k \text{ 0 вимог надійшло та 1 відійшла}\} = \mu\delta + o(\delta),$$

$$P\{\text{в інтервалі } I_k \text{ 1 вимога надійшла та 0 відійшло}\} = \lambda\delta + o(\delta).$$

Сума цих імовірностей дорівнює одиниці плюс $o(\delta)$. Отже ймовірність більш ніж одного надходження або виходу надзвичайно мала при малих δ. Це означає, що p_{ii} – ймовірність однакової кількості $i \geq 1$ надходжень і виходів в інтервалі I_k з точністю до $o(\delta)$ збігається з імовірністю (2.11); це показує справедливість рівності (2.8). Співвідношення (2.7), (2.9) і (2.10) встановлюються аналогічно.

Діаграма переходів зі стану в стан ланцюга Маркова $\{N_k\}$ показана на рис. 2.1, тут усі $o(\delta)$ опущені.

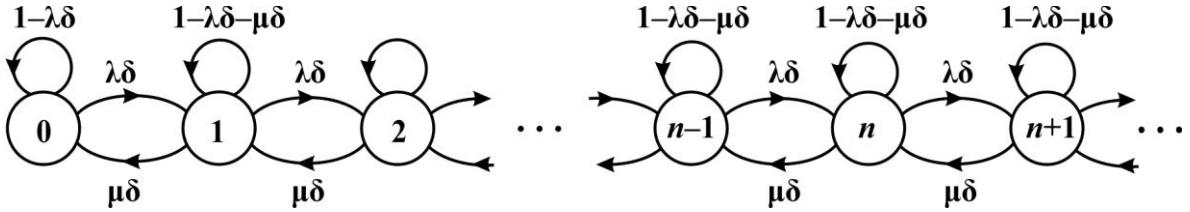


Рисунок 2.1 – Ланцюг Маркова з дискретним часом для системи $M/M/1$. Стан відповідає наявності вимог у системі. Перехідні ймовірності показані з точністю до $o(\delta)$

Розглянемо тепер стаціонарні ймовірності

$$p_n = \lim_{k\to\infty} P\{N_k = n\} = \lim_{t\to\infty} P\{N(t) = n\}.$$

Зауважимо, що за будь-яких $k \geq 1$, $n \geq 0$ протягом часу від δ до $k\delta$ загальна кількість переходів зі стану n у $n+1$ має відрізнятися від загальної кількості переходів з $n+1$ у n не більш, ніж на 1. Отже, у стаціонарному режимі ймовірність того, що система знаходиться в стані n і при подальшому переході потрапляє в стан $n+1$, дорівнює ймовірності того, що система знаходиться в стані $n+1$ і здійснює перехід у стан n, тобто

$$p_n\lambda\delta + o(\delta) = p_{n+1}\mu\delta + o(\delta). \qquad (2.12)$$

Ці рівняння називаються рівняннями глобального балансу, що відповідають множині станів $\{0, 1, \ldots, n\}$ і $\{n+1, n+2, \ldots\}$. Оскільки p_n не залежить від δ, спрямовуючи $\delta \to 0$ в (3.20), отримуємо

$$p_{n+1} = pp_n, \quad n = 0, 1, \ldots, \quad \text{де } p = \lambda/\mu.$$

Звідси випливає, що

$$p_{n+1} = p^{n+1}p_0, \quad n = 0, 1, \ldots . \qquad (2.13)$$

Якщо $p < 1$ (швидкість обслуговування перевищує швидкість надходжень), усі ймовірності p_n є позитивними, а їхня сума дорівнює одиниці. Отже,

$$1 = \sum_{n=0}^{\infty} p_n = \sum_{n=0}^{\infty} p^n p_0 = \frac{p_0}{1-p}. \qquad (2.14)$$

Це співвідношення разом з формулою (2.13) остаточно дає

$$p_n = p^n(1-p), \quad n = 0, 1, \ldots . \qquad (2.15)$$

Тепер можна обчислити середню кількість вимог у системі в стаціонарному режимі

$$N = \lim_{t\to\infty} E\{N(t)\} = \sum_{n=0}^{\infty} np_n = \sum_{n=0}^{\infty} np^n(1-p) = p(1-p)\sum_{n=0}^{\infty} np^{n-1} =$$

$$= p(1-p)\frac{\partial}{\partial p}\left(\sum_{n=0}^{\infty} p^n\right) = p(1-p)\frac{\partial}{\partial p}\left(\frac{1}{1-p}\right) = p(1-p)\frac{1}{p}\frac{1}{(1-p)^2}$$

і, нарешті, використовуючи рівність $p = \lambda/\mu$, отримуємо

$$N = \frac{p}{1-p} = \frac{\lambda}{\mu - \lambda}. \qquad (2.16)$$

Цю рівність показано графічно на рис. 2.2. При зростанні p значення N також зростає, і при $p \to 1$ маємо $N \to \infty$. Графік справедливий для $p < 1$. Якщо $p > 1$, обслуговуючий пристрій не може впоратися з потоком надходжень вимог і черга необмежено зросте. У термінах системи пакетної передачі $p > 1$ означає, що $\lambda L > C$, де λ – швидкість надходження (у кількості пакетів за секунду), L – середня довжина пакета в бітах, C – пропускна здатність каналу у бітах за секунду.

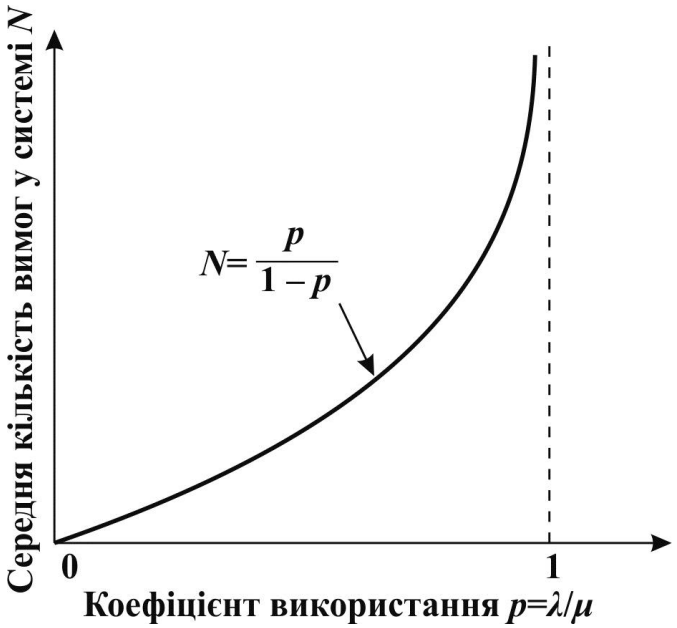


Рисунок 2.2 – Середня кількість вимог у системі $M/M/1$ залежно від коефіцієнта використання. При $p \to 1$ маємо $N \to \infty$

Середня затримка вимоги (час очікування в черзі плюс час обслуговування), відповідно до теореми Літтла, дорівнює

$$T = \frac{N}{\lambda} = \frac{p}{\lambda(1-p)}. \qquad (2.17)$$

Підставляючи $p = \lambda/\mu$, отримуємо

$$T = 1/(\mu - \lambda). \qquad (2.18)$$

Середній час очікування в черзі W дорівнює середній затримці пакета T мінус середній час обслуговування $1/\mu$, тому

$$W = \frac{1}{\mu - \lambda} - \frac{1}{\mu} = \frac{p}{\mu - \lambda}.$$

З теореми Літтла маємо, що середня кількість вимог у черзі

$$N_Q = \lambda W = \frac{p^2}{1-p}.$$

Дуже корисно розглядати величину p як коефіцієнт використання системи масового обслуговування, тобто як стаціонарну частку часу, протягом якого зайнятий обслуговуючий пристрій. Ми цей коефіцієнт використовували раніше в більш загальних умовах під час застосування теореми Літтла. Маємо $p = 1 - p_0$, де p_0 – ймовірність того, що в системі немає вимог; виходить інше доведення формули для (2.14).

2.3 Контрольні запитання та завдання

1. Сформулюйте теорему Літтла.
2. Поясніть значущість теореми Літтла.
3. Дайте характеристику моделі $M/M/1$.
4. Дайте визначення пуассонівського розподілу потоків.
5. Дайте визначення ланцюгів Маркова для системи $M/M/1$.
6. Наведіть формулу обчислення середньої кількості вимог у системі в стаціонарному режимі.
7. Поясніть, як збільшується кількість вимог у системі від коефіцієнта використання.

Лекція 3
МЕТОДИ ЗАБЕЗПЕЧЕННЯ ЯКОСТІ ОБСЛУГОВУВАННЯ НА ОСНОВІ КЕРУВАННЯ ЧЕРГАМИ ТА КОНДИЦІЮВАННЯ ТРАФІКА

3.1 Методи керування чергами

Черги й різні класи трафіка

Поглянемо, як можна застосувати наші знання про залежність поведінки черг від коефіцієнта завантаження для реалізації основної ідеї методів QoS, а саме диференційованого обслуговування класів трафіка з різними вимогами до характеристик продуктивності й надійності мережі. Щоб простіше було в цьому розібратися, ділитимемо всі потоки на два класи – чутливий до затримок (трафік реального часу, наприклад, голосовий) і еластичний, що дозволяє великі затримки, але чутливий до втрат даних.

Ми знаємо, що якщо забезпечити для чутливого до затримок трафіка коефіцієнт завантаження кожного ресурса не більше 0,2, то, мабуть, затримки в кожній черзі будуть невеликими й, швидше за все, прийнятними для багатьох типів додатків цього класу.

Для еластичного трафіка, слабко чутливого до затримок, можна допустити більш високий коефіцієнт завантаження, але не більше 0,9. Для того щоб пакети цього класу не губилися, слід передбачити для них буферну пам’ять, достатню для зберігання всіх пакетів періода пульсації. Ефект від такого розподілу завантаження ресурса ілюструє рис. 3.1.

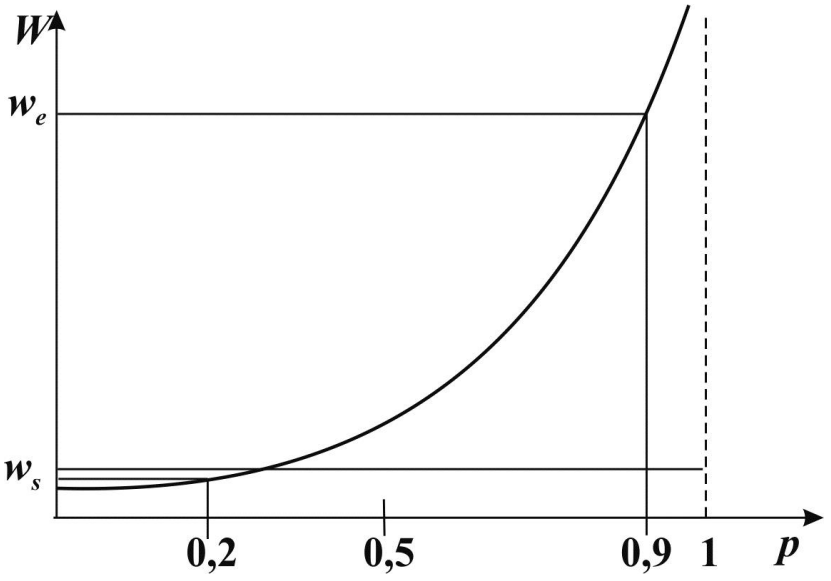


Рисунок 3.1 – Обслуговування еластичного й чутливого до затримок трафіка

Затримки чутливого до затримок трафіка дорівнюють w_s, а затримки еластичного трафіка – w_e.

Щоб досягти різних коефіцієнтів використання ресурсів для різних класів трафіка, потрібно в кожному комутаторі для кожного ресурса підтримувати дві різні черги. Алгоритм вибірки пакетів з черг має віддавати перевагу черзі чутливих до затримок пакетів. Якби всі пакети першої черги обслуговувалися пріоритетно, а пакети другої – лише тоді, коли перша черга порожня, то для трафіка першої черги трафік другої черги фактично перестав би існувати. Тому, якщо відношення середньої інтенсивності пріоритетного трафіка λ_1 до продуктивності ресурсу μ дорівнює 0,2, то й коефіцієнт завантаження для нього дорівнює 0,2. А от для еластичного трафіка, пакети якого завжди очікують обслуговування пріоритетних пакетів, коефіцієнт завантаження підраховується інакше. Якщо середня інтенсивність еластичного трафіка дорівнює λ_2, то для нього ресурс буде завантажений на $(\lambda_1 + \lambda_2)/\mu$. Так що, якщо ми хочемо, щоб для еластичного трафіка коефіцієнт завантаження становив 0,9, то його інтенсивність має знаходитися співвідношенням $\lambda_2/\mu = 0,7$.

Головна ідея, яка полягає в основі всіх методів підтримки характеристик QoS, така: загальна продуктивність кожного ресурсу має бути поділена між різними класами трафіка нерівномірно.

Можна ввести більше двох класів обслуговування й намагатися, щоб кожен клас працював на своїй частині кривої затримок. Якщо таке завдання вирішено, то можна забезпечити покращення характеристик QoS за рахунок інших методів, наприклад, знижуючи пульсацію трафіка.

Залишилося з’ясувати, як можна забезпечити такі умови для різних класів трафіка в кожному вузлі мережі.

Техніка керування чергами

Техніка керування чергами потрібна для роботи в періоди часових перевантажень, коли мережний пристрій не справляється з передачею пакетів на вихідний інтерфейс у тому темпі, у якому вони надходять. Якщо причиною перевантаження є недостатня продуктивність процесорного блока мережного пристрою, то необроблені пакети тимчасово накопичуються у вхідній черзі відповідного вхідного інтерфейса.

Черг до вхідного інтерфейса може бути декілька, якщо ми диференціюємо запити на обслуговування по декількох класах. У тому ж

випадку, коли причиною перевантаження є обмежена пропускна здатність вихідного інтерфейса, пакети тимчасово зберігаються у вихідний черзі (або чергах) цього інтерфейсу.

Черга FIFO

У черзі FIFO у разі перевантаження всі пакети поміщуються в одну загальну чергу й вибираються з неї у тому порядку, у якому надійшли. В усіх пристроях з комутацією пакетів алгоритм FIFO використовується за замовчуванням, тому така черга також зазвичай називається чергою «за замовчуванням». Перевагами цього підходу є простота реалізації й відсутність потреби в конфігурації. Однак йому притаманний і корінний недолік – неможливість диференційованої обробки пакетів різних потоків. Усі пакети знаходяться у загальній черзі на рівних підставах. Разом знаходяться і пакети чутливого до затримок голосового трафіка, і пакети нечутливого до затримок, але дуже інтенсивного трафіка резервного копіювання, тривалі пульсації якого можуть надовго затримати голосовий пакет.

Пріоритетне обслуговування

Черги з пріоритетним обслуговуванням дуже популярні в багатьох галузях обчислювальної техніки, зокрема, в операційних системах, коли одним додаткам слід віддати перевагу перед іншими під час обробки їх у мультипрограмній суміші.

Застосовуються ці черги й для переважної, порівняно з іншими, обробки одного класу трафіка.

Механізм пріоритетного обслуговування заснований на поділі всього мережного трафіка на невелику кількість класів і подальшого призначення кожному класу деякої числової ознаки – пріоритету.

Класифікація трафіка є окремим завданням. Пакети можуть розбиватися на пріоритетні класи на підставі різних ознак: адреси призначення, адреси джерела, ідентифікатора додатка, що генерує цей трафік, будь-яких інших комбінацій ознак, які містяться в заголовках пакетів. Правила класифікації пакетів є частиною політики адміністрування мережі.

Точка класифікації трафіка може розміщуватися в кожному комунікаційному пристрої. Більш масштабно – розміщення функцій класифікації трафіка в одному або декількох пристроях, розташованих на межі мережі (наприклад, у комутаторах корпоративної мережі, до яких підключаються комп’ютери користувачів, або у вхідних маршрутизаторах

мережі постачальника послуг). У цьому випадку необхідне спеціальне поле в пакеті, у якому можна запам’ятати призначене значення пріоритету, щоб ним могли скористатися інші мережні пристрої, які обробляють трафік після пристрою, що класифікує. Таке поле є в заголовку багатьох протоколів.

У тих же випадках, коли спеціального поля пріоритету в заголовку немає, розробляється додатковий протокол, який вводить новий заголовок з таким полем (наприклад, з протоколом Ethernet).

Пріоритети можуть призначатися не тільки комутатором або маршрутизатором, а й додатком на вузлі-відправнику. Необхідно також враховувати, що якщо в мережі відсутня централізована політика призначення пріоритетів, кожен мережний пристрій може не погодитися з пріоритетом, призначеним даному пакету в іншій точці мережі. У цьому випадку він перепише значення пріоритету згідно з локальною політикою, прийнятою безпосередньо на цьому пристрої.

У мережному пристрої, що підтримує пріоритетне обслуговування, є кілька черг (буферів) – по одній для кожного пріоритетного класу. Пакет, що надійшов у період перевантажень, поміщується в чергу, відповідну його пріоритетному класу.

(Іноді кілька черг зображують у вигляді однієї черги, у якій знаходяться заявки різних класів. Якщо заявки вибираються з черги відповідно до їх пріоритетів, то це просто інше подання одного й того самого механізму.)

На рис. 3.2 наведений приклад використання чотирьох пріоритетних черг з високим, середнім, нормальним і низьким пріоритетами.

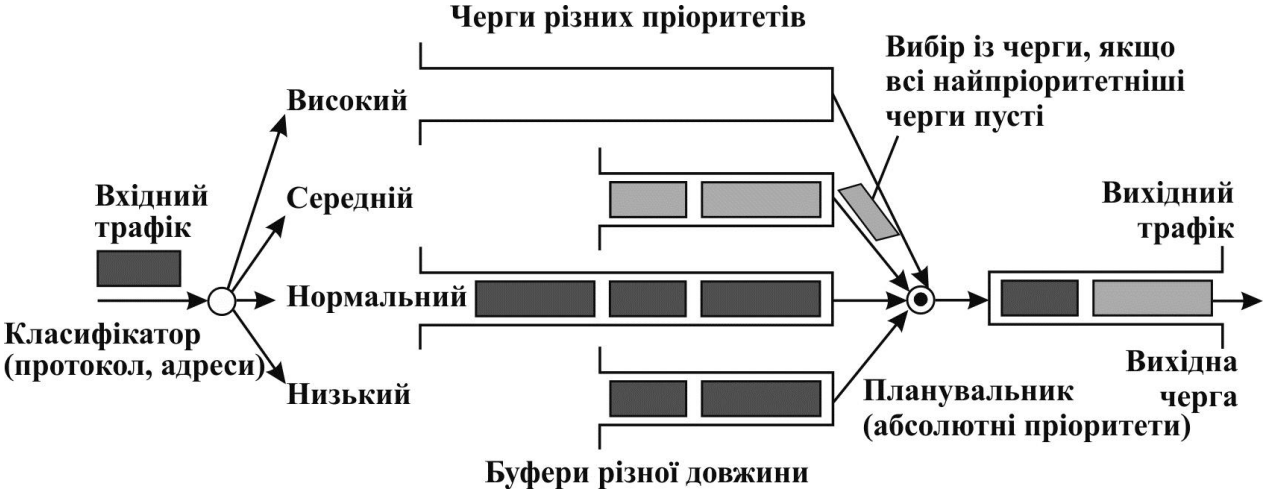


Рисунок 3.2 – Пріоритетні черги

Доти, доки з більш пріоритетної черги не будуть обрані всі наявні в ній пакети, пристрій не перейде до обробки наступної, менш пріоритетної, черги.

Тому пакети з низьким пріоритетом оброблюються тільки тоді, коли стають порожніми усі вищі черги: з високим, середнім і нормальним пріоритетами.

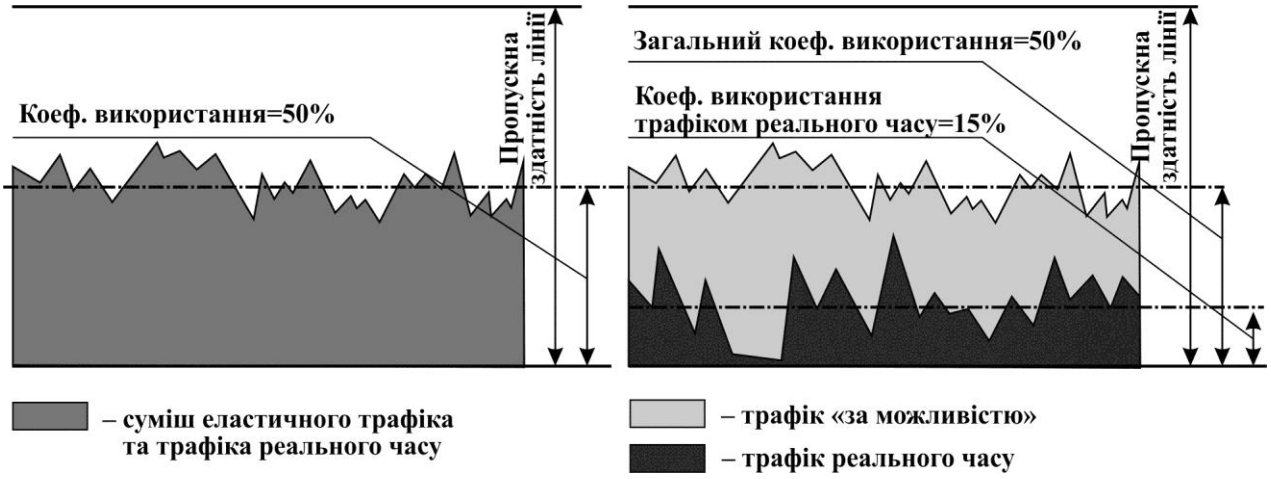
Розмір буфера мережного пристрою визначає максимальну довжину черги пакетів, які очікують обслуговування; якщо пакет надходить при заповненому буфері, то він просто відкидається. Зазвичай за замовчуванням усім пріоритетним чергам відводяться однакові буфери, але багато пристроїв дозволяють адміністратору призначати кожній черзі буфер індивідуального розміру. Розмір буфера визначається, в ідеальному випадку, так, щоб його було достатньо з невеликим запасом для зберігання черги середньостатистичної довжини. Однак встановити це значення досить складно, оскільки воно змінюється залежно від навантаження мережі, тому потрібне постійне й тривале спостереження за роботою мережі. У загальному випадку, чим вище значущість трафіка для підприємства, чим більше його інтенсивність і пульсації, тим більший розмір буфера потрібно цьому трафіку. У прикладі, наведеному на рис. 3.2, для трафіка вищого й нормального пріоритетів обрані великі розміри буферів, а для інших двох класів – менші.

Мотиви прийнятого рішення для вищого пріоритету очевидні, а трафік нормального пріоритету має, очевидно, високу інтенсивність і значний коефіцієнт пульсацій.

Пріоритетне обслуговування черг забезпечує високу якість обслуговування для пакетів з найбільш пріоритетною чергою. Якщо середня інтенсивність їх надходження в пристрій не перевищує пропускної здатності вихідного інтерфейсу (і продуктивності внутрішніх просувних блоків самого пристрою), то пакети вищого пріоритету завжди отримують ту пропускну здатність, яка їм потрібна. Рівень затримок високопріоритетних пакетів також мінімальний. Однак він не нульовий і залежить, в основному, від характеристик потоку цих пакетів – чим вище пульсації потоку і його інтенсивність, тим імовірніше виникнення черги, утвореної пакетами даного високопріоритетного потока. Трафік всіх інших пріоритетних класів майже прозорий для пакетів вищого пріоритету. Слово «майже» стосується ситуації, коли високопріоритетний пакет змушений чекати завершення обслуговування низькопріоритетного пакета, якщо його прихід збігається за часом з початком просування низькопріоритетного пакета на вихідний інтерфейс. Цей ефект ілюструє рис. 3.3, де показано, що після поділу всього трафіка на пріоритетний і звичайний (тобто тут є дві черги), коефіцієнт використання для пріоритетного трафіка знизився з 50 до 15%.

Що ж стосується інших пріоритетних класів, то якість їх обслуговування буде нижче, ніж у пакетів найвищого пріоритету, причому рівень зниження

може бути дуже суттєвим. Якщо коефіцієнт навантаження вихідного інтерфейса, який визначається тільки трафіком вищого пріоритетного класу, наближається в якийсь період часу до одиниці, то трафік інших класів на цей час просто заморожується. Тому пріоритетне обслуговування зазвичай застосовується для чутливого до затримок класу трафіка, що має невелику інтенсивність. За таких умов обслуговування цього класу не занадто обмежує обслуговування решти трафіка. Наприклад, голосовий трафік чутливий до затримок, але його інтенсивність зазвичай не перевищує 8–16 Кбіт/с, тому при призначенні йому вищого пріоритету збиток іншим класам трафіка виявиться не значним.



а) б)

Рисунок 3.3 – Зниження коефіцієнта використання лінії для пріоритетного трафіка: а) весь трафік обслуговується однією чергою; б) трафік реального часу обслуговується пріоритетною чергою, а інший – чергою за замовчуванням

Зважені черги

Механізм зважених черг розроблений для того, щоб можна було надати всім класам трафіка певний мінімум пропускної здатності. Під вагою даного класу розуміють відсоток від повної пропускної здатності вихідного інтерфейса, який надається класу трафіка.

При зваженому обслуговуванні, так само, як і при пріоритетному, трафік поділяється на кілька класів, і для кожного класу ведеться окрема черга пакетів. Але з кожною чергою пов’язується не пріоритет, а відсоток пропускної здатності ресурсу, який гарантується даному класу трафіка під час перевантаження цього ресурсу. Для вхідного потоку таким ресурсом є процесор, а для вихідного (після виконання комутації) – вихідний інтерфейс.

Приклад

Показаний на рис. 3.4 пристрій для п’яти класів трафіка підтримує п’ять черг до вихідного інтерфейса комутатора. Цим чергам при перевантаженнях виділяється відповідно 10, 10, 30, 20 і 30% пропускної здатності вихідного інтерфейса.

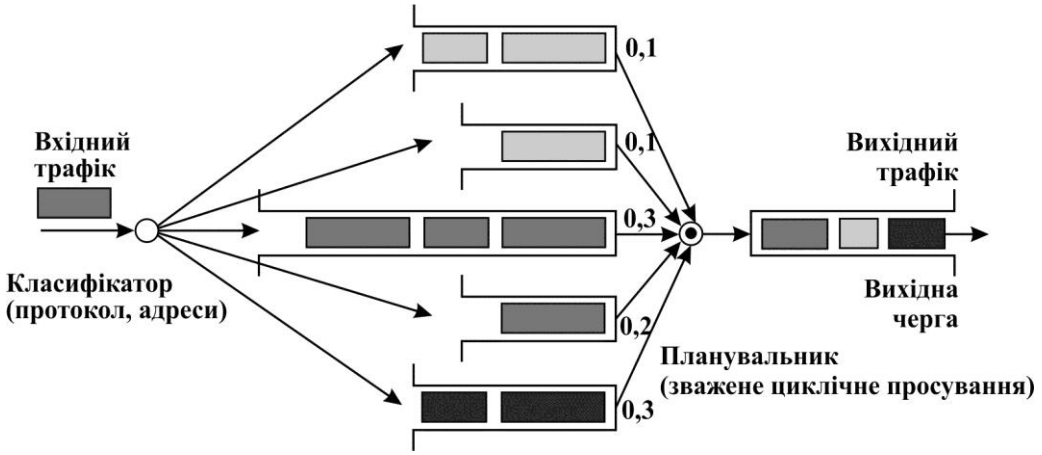


Рисунок 3.4 – Зважені черги

Досягається поставлена мета за рахунок того, що черги обслуговуються послідовно і циклічно, і в кожному циклі обслуговування з кожної черги вибирається така кількість байтів, яка відповідає базі даної черги. Так, якщо цикл перегляду черг в розглянутому прикладі дорівнює одній секунді, а швидкість вихідного інтерфейса становить 100 Мбіт/с, то при перевантаженнях в кожному циклі першої черги приділяється 10% часу, тобто 100 мс і вибирається 10 Мбіт даних, з другої – також 10 Мбіт, з третьої – 30 Мбіт, з четвертої – 20 Мбіт, з п’ятої – 30 Мбіт.

У результаті кожному класу трафіка дістається гарантований мінімум пропускної здатності, що в багатьох випадках є більш бажаним результатом, ніж пригнічення фонових класів високопріоритетним.

Оскільки дані вибираються з черги пакетами, а не бітами, то реальний розподіл пропускної здатності між класами трафіка завжди дещо відрізняється від запланованого. Так, у попередньому прикладі замість 10% перший клас трафіка міг би отримувати при перевантаженнях 9 або 12%. Чим більше час циклу, тим точніше дотримуються необхідні пропорції між класами трафіка, оскільки з кожної черги вибирається більша кількість пакетів, і вплив розміру кожного пакета усереднюється.

Але тривалий цикл призводить до великих затримок передачі пакетів. Так, при обраному нами для прикладу циклі в одну секунду затримка може скласти одну секунду і більше – адже арбітр повертається до кожної черги не частіше, ніж один раз на секунду, крім того, в черзі може знаходитися більше одного пакета. Тому при виборі часу цикла слід забезпечити баланс між точністю дотримання пропорцій пропускної здатності і прагненням до зниження затримки.

Для нашого прикладу час циклу в 1000 мкс є прикладом такого балансу. З одного боку, він забезпечує обслуговування черги кожного класа кожні 1000 мкс, а, значить, більш низький рівень затримок. З іншого боку, цього часу достатньо, щоб вибрати з кожної черги в середньому по кілька пакетів (першій черзі в нашому прикладі відводитиметься 100 мкс, що достатньо для передачі у вихідний канал одного пакета Fast Ethernet або десяти пакетів Gigabit Ethernet).

На рівень затримок і варіації затримок пакетів для деякого класу трафіка при зваженому обслуговуванні значною мірою впливає відносний коефіцієнт використання. У цьому випадку коефіцієнт підраховується як відношення інтенсивності вхідного трафіка класу до пропускної здатності, виділеної цьому класу відповідно до його ваги. Наприклад, якщо ми виділили першій черзі 10% від загальної пропускної здатності вихідного інтерфейса, тобто 10 Мбіт/с, а середня інтенсивність потоку, який потрапляє в цю чергу, дорівнює 3 Мбіт/с, то коефіцієнт використання для цього потоку складе $3/10 = 0{,}3$. Залежність на рис. 1.5 (лекція 1) показує, що затримки при такому значенні коефіцієнта використання будуть незначними. Якби інтенсивність вхідного потоку цієї черги була 9 Мбіт/с, то черги були б значними, а при перевищенні межі 10 Мбіт/с частина пакетів потоку постійно відкидалася б через переповнення черги.

Якісна поведінка черги і, відповідно, затримок, тут виглядає приблизно так само, як і в разі черги FIFO – чим менше коефіцієнт завантаження, тим менше середня довжина черги і тим менші затримки.

Як і для пріоритетного обслуговування, при зваженому обслуговуванні адміністратор може призначати різним класам черг буфери різних розмірів. Зменшення розмірів буферів для черг призводить до зростання кількості втрат пакетів при перевантаженнях, але знижує час очікування для тих пакетів, які не були відкинуті і потрапили в чергу.

Ще одним варіантом зваженого обслуговування є зважене справедливе обслуговування (Weighted Fair Queuing, WFQ). У разі подібного обслуговування

пропускна здатність ресурсу розділяється між всіма потоками порівну, тобто «справедливо».

Зважене обслуговування забезпечує необхідні співвідношення між інтенсивностями трафіка різних черг тільки в періоди перевантажень, коли кожна черга постійно заповнена. Якщо ж котрась із черг порожня (тобто для трафіка даного класу поточний період не є періодом перевантаження), то при перегляді черг вона ігнорується, і її час обслуговування розподіляється між іншими чергами відповідно до їх ваги. Тому в окремі періоди трафік певного класу може мати більшу інтенсивність, ніж відповідний відсоток від пропускної здатності вихідного інтерфейса.

Комбіновані алгоритми обслуговування черг

Кожний з описаних підходів має свої переваги та недоліки. Пріоритетне обслуговування, забезпечуючи мінімальний рівень затримок для черги найвищого пріоритету, не дає ніяких гарантій відносно середньої пропускної здатності для трафіка черг нижчих пріоритетів. Зважене обслуговування забезпечує заданий розподіл середньої пропускної здатності, але не враховує вимог до затримок.

Існують комбіновані алгоритми обслуговування черг. У найбільш популярному алгоритмі подібного роду підтримується одна пріоритетна черга, а інші черги обслуговуються відповідно зі зваженим алгоритмом. Зазвичай пріоритетна черга використовується для чутливого до затримок трафіка, а інші – для еластичного трафіка декількох класів. Кожний клас еластичного трафіка отримує деякий мінімум пропускної здатності при перевантаженнях. Цей мінімум обчислюється як відсоток від пропускної здатності, що залишився від пріоритетного трафіка.

Вочевидь, що слід якось обмежити пріоритетний трафік, щоб він не поглинав усю пропускну здатність ресурсу. Зазвичай для цього застосовуються механізми кондиціювання трафіка, які розглядаються далі.

3.2 Механізми кондиціювання трафіка

Механізми кондиціювання трафіка контролюють поточні параметри потоків трафіка, такі як його середня швидкість і пульсація. Як ми пам’ятаємо, основною ідеєю методів QoS є виділення певної частки пропускної здатності певним потокам трафіка, при цьому величина цієї частки має бути достатньою для того, щоб коефіцієнт використання ресурсу для потоку був досить низьким, і, відповідно, якість обслуговування потоку була задовільною. Черги з різними алгоритмами обслуговування дозволяють реалізувати тільки одну частину цієї

ідеї – вони виділяють певну частку пропускної здатності деякому потоку пакетів. Однак залишається друга частина завдання – утримання швидкості потоку в певних межах з метою забезпечення бажаного коефіцієнта використання пропускної здатності, яка виділена потоку за допомогою певної черги. Якщо ж швидкість потока не відповідатиме очікуваній, то вся робота з виділення потоку пропускної здатності не призведе до бажаного результату, оскільки коефіцієнт використання цієї пропускної здатності відрізнятиметься від очікуваного, і потрібна якість обслуговування не буде досягнута.

Механізми кондиціювання трафіка є свого роду контрольно-пропускними пунктами, які перевіряють трафік на вході в комутатор (або формують трафік на виході з нього – для чого це потрібно, ми розглянемо далі). Існує кілька механізмів кондиціювання трафіка.

Класифікація трафіка

Класифікація трафіка є елементом QoS, що дозволяє визначити, які пакети потрібно відправити в ту чи іншу чергу. Класифікація зазвичай виконується засобами фільтрації трафіка, наявними в комутаторах і маршрутизаторах пакетних мереж; для цих засобів використовується також така назва, як списки контролю доступу (Access Control List, ACL; їх не можна плутати із засобами контролю допуску (admission control) трафіка, які також використовуються в системах забезпечення якості обслуговування, але мають інше призначення).

Для класифікації використовуються різні ознаки пакетів, наприклад, адреси призначення й джерела, тип протоколу транспортного або прикладного рівня. Ми вже згадували класифікацію трафіка під час опису пріоритетних і зважених черг, оскільки без цього механізму кондиціювання трафіка неможливо задіяти різні черги до того самого вихідного інтерфейсу.

Профілювання

Профілювання є заходом примусового впливу на трафік, який служить для обмеження швидкості потоку пакетів. Профілювання забезпечує відповідність потоку пакетів заданому швидкісному профілю; як параметри профіло зазвичай вибирається середня швидкість потоку пакета, що вимірюється на певному інтервалі часу (застосовуються й складніші варіанти профілювання, наприклад, ті, що враховують середню й пікові швидкості). Пакети, які не вкладаються у заданий профіль, або відкидаються, або декваліфікуються, тобто містяться в класі обслуговування з нижчими

пріоритетами, наприклад, переводяться із пріоритетного класу в стандартний клас, що обслуговується «за можливістю». Англомовна назва операції профілювання – policing (дослівно – «підтримувати порядок поліцейськими засобами») – здається жорсткішою і, можливо, дає більш адекватне уявлення про характер дій.

Профілювання зазвичай застосовують для обмеження трафіка, що надходить у пріоритетну чергу, оскільки цей механізм є єдино можливим засобом запобігання ситуації витиснення всього іншого трафіка пріоритетним трафіком.

Рисунок 3.5 ілюструє дію механізму профілювання, показуючи значення швидкості трафіка, що вимірюються на досить малих інтервалах часу до та після профілювання.

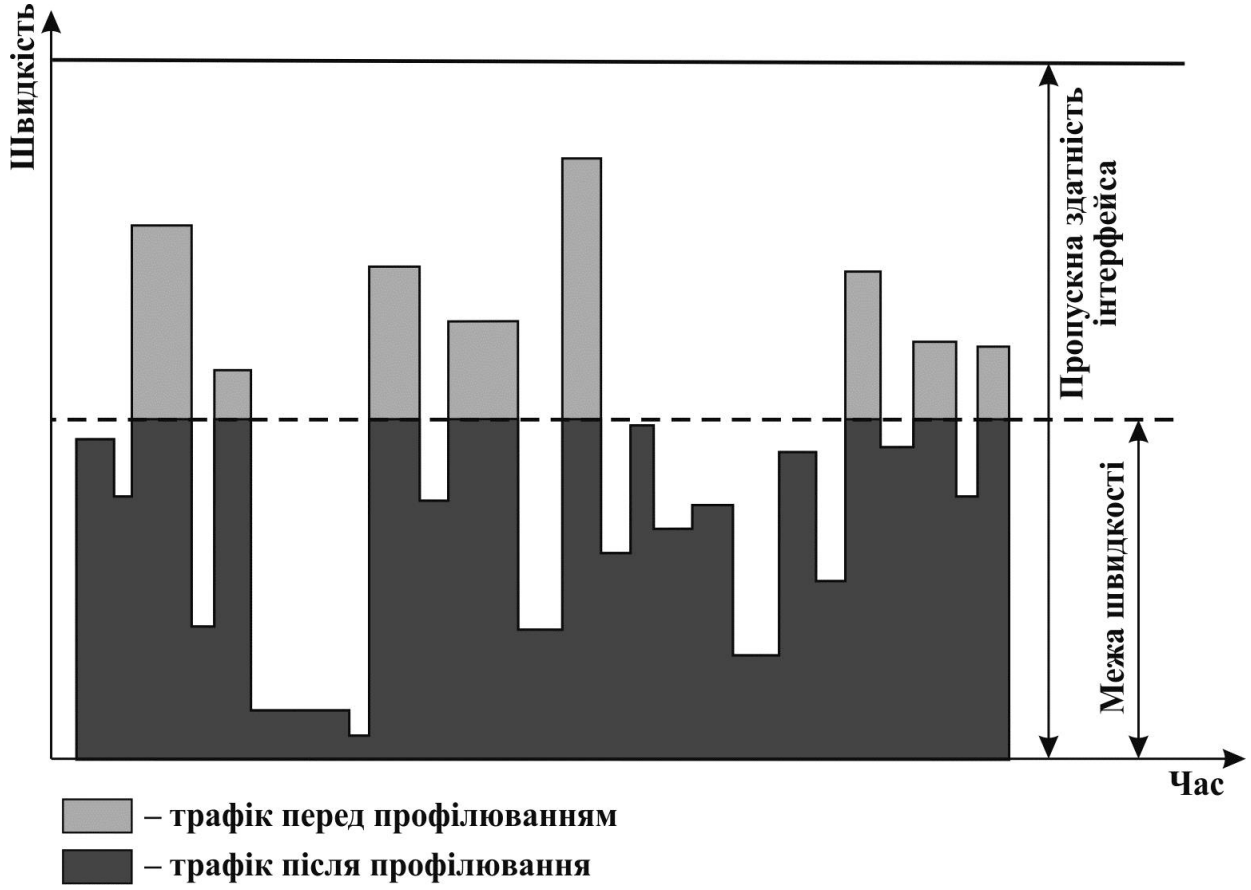


Рисунок 3.5 – Ефект профілювання – відкидання надлишкового трафіка

Як видно з рисунка, відкидання пакетів під час профілювання призводить до стримання швидкості потоку на заданому рівні в ті інтервали часу, коли швидкість вхідного потоку перевершує цю межу, і до збереження вихідної швидкості в інші періоди.

Формування трафіка

Формування трафіка – це процес навмисної затримки деяких пакетів із загального потоку з метою утримання середньої швидкості трафіка в деяких заданих межах.

Формування трафіка нагадує профілювання, оскільки має схожу мету – обмеження швидкості трафіка, але досягається ця мета іншим способом. Замість того, щоб відкидати надлишкові пакети (тобто ті, передача яких могла б призвести до перевищення ліміту швидкості), механізм формування трафіка затримують пакети-порушники так, що результуюча швидкість виявляється в заданих межах. Ефект формування трафіка ілюструє рис. 3.6 (на задньому плані рисунка показана швидкість результуючого пакета, а на передньому – швидкість вихідного потоку (напівпрозорим заповненням)).

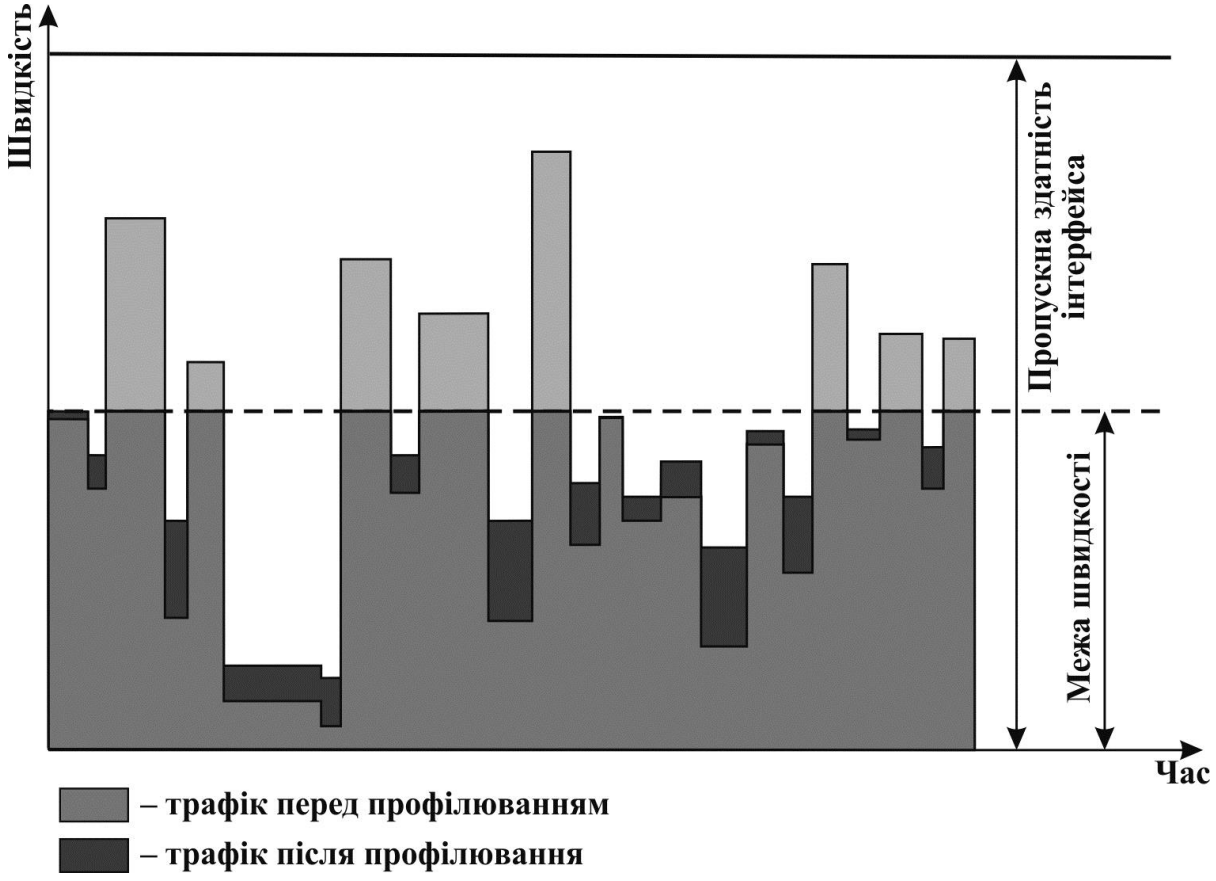


Рисунок 3.6 – Ефект формування трафіка – згладжування

З рисунка видно, що швидкість трафіка згладжується, оскільки надлишкові пакети не відкидаються, а передаються із затримкою в інші інтервали часу. Тим самим швидкість вихідного потоку знижується протягом періодів часу з надлишковим трафіком і зростає в тих наступних інтервалах, де вона виявляється менше встановленої межі.

Зазвичай шляхом формування обробляють трафік, що виходить із комутатора або маршрутизатора. Це робиться в тих випадках, коли відомо, що деякий комунікаційний пристрій далі маршрутом проходження потоку пакетів застосовує профілювання. Профіль формування трафіка обирається таким, що дорівнює профілю трафіка, який профілюється. Це гарантує відсутність втрат трафіка через відкидання надлишкових пакетів.

Механізми кондиціювання трафіка можуть підтримуватися кожним вузлом мережі або реалізовуватися тільки в приграничних пристроях. Останній варіант часто використовують постачальники послуг, кондиціонуючи трафік своїх клієнтів.

3.3 Контрольні запитання та завдання

1. Поясніть особливості обслуговування еластичного і чутливого до затримок трафіка.
2. Поясніть основну ідею, що лежить в основі всіх методів підтримки характеристик QoS.
3. Поясніть призначення керування чергами.
4. Поясніть сутність методу FIFO.
5. Поясніть сутність пріоритетного обслуговування черг.

Лекція 4
МЕТОДИ КЕРУВАННЯ ТРАФІКОМ НА ОСНОВІ ЗВОРОТНОГО ЗВ'ЯЗКУ ТА РЕЗЕРВУВАННЯ РЕСУРСІВ

4.1 Механізми використання зворотного зв'язку для керування трафіком

Зворотний зв'язок. Призначення

Алгоритми керування чергами й кондиціювання трафіка не запобігають перевантаженням, а лише деяким «справедливим» способом в умовах дефіциту перерозподіляють ресурси між різними потоками або класами трафіка. Алгоритми керування чергами належать до механізмів керування перевантаженнями (congestion management), які починають працювати тоді, коли мережа вже перевантажена.

Існує інший клас засобів, які називаються механізмом запобігання перевантаженням (congestion avoidance). Цей механізм заснований на використанні зворотного зв'язку, за допомогою якого перевантажений вузол мережі, реагуючи на перевантаження, просить попередні вузли, розташовані вздовж маршруту проходження потоку (або потоків, що належать до одного класу), тимчасово знизити швидкість трафіка. Після того як перевантаження у даному вузлі зникне, він посилає інше повідомлення, що дозволяє підвищити швидкість передачі даних.

Отже, при виникненні перевантаження механізм запобігання перевантажень за рахунок зворотного зв'язку тимчасово знижує навантаження. Існує й інша назва цього механізму – активне керування чергами.

Учасники зворотного зв'язку

Існує кілька механізмів зворотного зв'язку. Вони відрізняються інформацією, що передається зворотним зв'язком, а також типом вузла, що генерує цю інформацію й тим, хто реагує неї, – кінцевий вузол (комп'ютер) або проміжний (комутатор чи маршрутизатор).

На рис. 4.1 показані різні варіанти організації зворотного зв'язку.

Зворотний зв'язок 1 організований між двома кінцевими вузлами мережі. Цей варіант забезпечує найбільш радикальне зниження навантаження на мережу, оскільки лише кінцевий вузол може знизити швидкість надходження інформації в мережу. Однак цей вид зворотного зв'язку не належить до методів

керування перевантаженням, оскільки його призначення – боротьба з перевантаженнями вузла призначення, а не з перевантаженнями мережних пристроїв. Принципово це одна й та сама проблема, бо вона є наслідком тимчасового перевищення швидкості надходження пакетів у ресурс над швидкістю обробки цих пакетів. Але ресурсом у цьому випадку виступає не комутатор мережі, а кінцевий вузол. Традиційно за цим видом зворотного зв'язку закріпилася власна назва – контроль потоку. Пристрої мережі не приймають участь у роботі цього виду механізму зворотного зв'язку, вони лише передають відповідні повідомлення між кінцевими вузлами. Незважаючи на різні назви, у методах керування перевантаженням і контролю потоку використовуються спільні механізми.

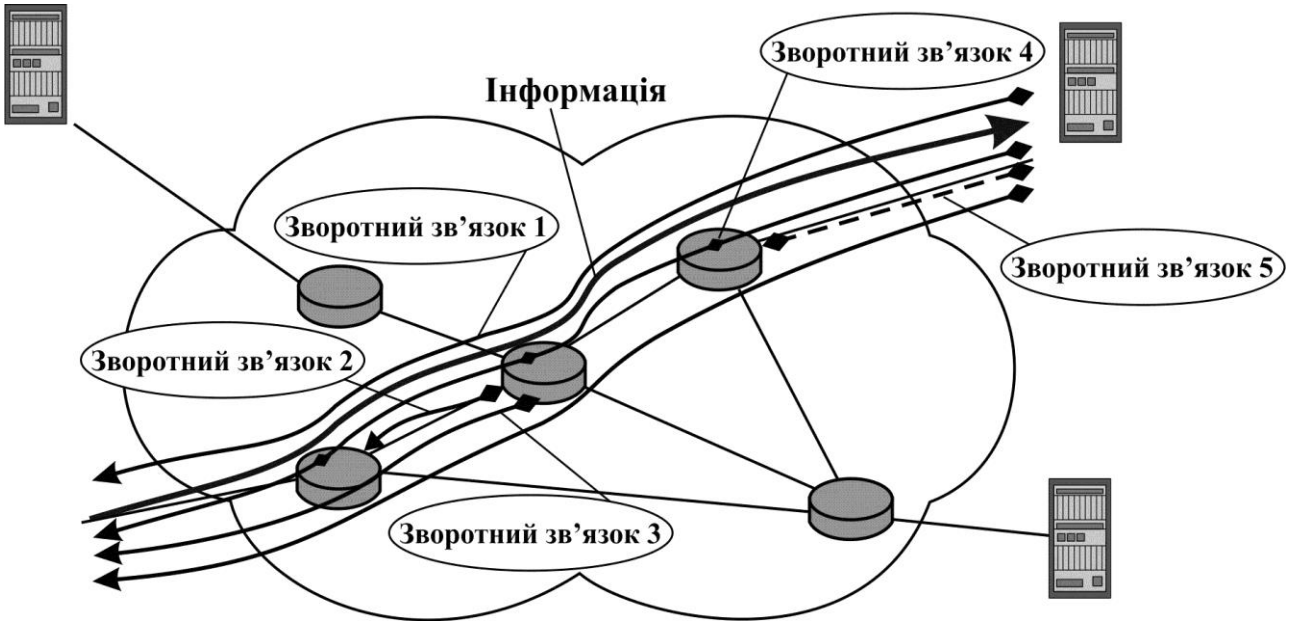


Рисунок 4.1 – Учасники зворотного зв'язку

В ході організації зворотного зв'язку важливо враховувати час передачі інформації з мережі. У високошвидкісних глобальних мережах за час передачі повідомлення про перевантаження вузла призначення вузол-джерело може встигнути передати через мережу тисячі пакетів, так що перевантаження не буде ліквідоване вчасно. З теорії автоматичного керування відомо, що затримки в контурі зворотного зв'язка можуть призводити до багатьох небажаних ефектів, протилежних первинним цілям. Наприклад, у системі можуть початися коливальні процеси, і вона ніколи не зможе прийти в рівноважний стан. Подібні явища спостерігалися на ранній стадії розвитку Інтернету, коли через недосконалість алгоритмів зворотного зв'язку й маршрутизації в ньому виникали ділянки перевантажень, які періодично переміщалися по мережі. Причина такої проблеми зрозуміла – затримка у контурі зворотного зв'язку

призводить до того, що регулюючий елемент отримує застарілу інформацію про стан регульованого елемента. У цьому випадку вузол-джерело отримує інформацію про стан черги вузла-отримувача із затримкою. Тому можливі ситуації, коли вузол-джерело починає знижувати швидкість передачі інформації, хоча насправді черги у вузлі-отримувачі вже немає, і, навпаки, підвищувати швидкість передачі інформації у той момент, коли вузол-отримувач почав зазнавати перевантаження. Для боротьби з такими явищами в контур зворотного зв'язку зазвичай вводиться інтегруючий елемент, який на кожному кроці обробляє не тільки поточне повідомлення зворотного зв'язку, але й кілька попередніх повідомлень, що дозволяє врахувати динаміку зміни ситуації й реагувати адекватно.

Зворотний зв'язок 2 організований між двома сусідніми комутаторами. Комутатор повідомляє сусідові, який знаходиться рівнем вище, що він зазнає перевантаження і його буфер заповнився до критичної величини. Отримавши таке повідомлення, сусід, розташований вище, має знизити на деякий час швидкість передачі даних у напрямку перевантаженого комутатора й тим самим вирішити проблему перевантаження. Для мережі в цілому це менш ефективне рішення, оскільки потік продовжуватиме рух від вузла-джерела з тією самою швидкістю, що й раніше. Але для комутатора, що зазнає перевантаження, це є гарним виходом, оскільки він отримує час для того, щоб розвантажити чергу, що переповнилася. Правда, проблема переноситься в комутатор, розташований вище, у якому тепер може виникнути перевантаження, оскільки він починає передавати дані зі свого буфера з меншою швидкістю. Перевагою описаного методу є низька затримка зворотного зв'язку, оскільки вузли є сусідами (якщо вони, звичайно, не з'єднані супутниковим каналом).

Зворотний зв'язок 3 організований між проміжним комутатором і вузлом-джерелом. Повідомлення зворотного зв'язку хоча й передаються декількома комутаторами мережі в напрямі вузла-джерела, але вони на нього не реагують.

У *зворотному зв'язку 4*, як і в зворотному зв'язку 1, повідомлення про перевантаження породжується вузлом-отримувачем і передається вузлу-джерелу. Однак є й важлива відмінність: у цьому випадку кожний проміжний комутатор реагує на це повідомлення. По-перше, він знижує швидкість передачі даних у напрямі вузла призначення, по-друге, він може змінити зміст повідомлення. Наприклад, якщо вузол призначення просить знизити швидкість до 30 Мбіт/с, то проміжний комутатор може знизити цю величину до 20 Мбіт/с, оцінивши стан свого буфера. Крім того, організувати повідомлення зворотного зв'язку може будь-який комутатор мережі, а не тільки вузол призначення.

В ході описання різних варіантів організації зворотного зв’язку малося на увазі, що повідомлення про перевантаження йде в напрямі, зворотному напряму передачі користувальницької інформації (власне, тому цей механізм так і називається). Але деякі комунікаційні протоколи не передбачають можливості генерації подібних повідомлень проміжними вузлами. У таких умовах часто застосовують штучний прийом – передача повідомлення про перевантаження вузлу призначення, що перетворить його в повідомлення зворотного зв’язку й відправляє в потрібному напрямі, тобто в напрямі джерела. Цей варіант показаний на рисунку як *зворотний зв’язок 5*.

Інформація зворотного зв’язку

У методах зворотного зв’язку, які застосовуються сьогодні, використовуються такі основні типи повідомлень:

• ознака перевантаження;

• максимальна швидкість передачі;

• максимальний обсяг даних (кредит);

• непрямі ознаки.

Ознака перевантаження не говорить про ступінь перевантаженості мережі або вузла, вона лише фіксує факт наявності перевантаження. Реакція вузла, який отримав таке повідомлення, може бути різною. У деяких протоколах вузол зобов’язаний припинити передачу інформації в певному напрямку доти, доки не отримає інше повідомлення зворотного зв’язку, яке дозволяє продовження передачі. В інших протоколах вузол поводиться адаптивно, він знижує швидкість на деяку величину й очікує реакції мережі. Якщо повідомлення з ознакою перевантаження надходять і далі, то він продовжує зниження швидкості.

У другому типі повідомлень вказується максимальна швидкість передачі, тобто поріг швидкості, якого має дотримуватися джерело або проміжний вузол, розташований за потоком вище. У цьому випадку слід обов’язково враховувати час передачі повідомлення по мережі, аби виключити коливальні процеси в мережі й забезпечити потрібну швидкість реакції на перевантаження. Тому, в територіальних мережах такий спосіб зазвичай реалізується силами всіх комутаторів мережі (зворотний зв’язок 4 у нашому прикладі).

Повідомлення про максимальний обсяг даних використовується в алгоритмі ковзного вікна, що широко застосовується в пакетних мережах.

Цей алгоритм дозволяє не тільки забезпечувати надійну передачу даних, але й реалізувати зворотний зв’язок для контролю потоку між кінцевими вузлами. Параметром, що несе інформацію зворотного зв’язку, є «вікно» –

число, тісно пов’язане з поточним розміром вільного простору в буфері приймаючого вузла. Вікно також називають кредитом, що приймаючий дає передавальному вузлу. Передавальний вузол може з будь-якою швидкістю передати обсяг інформації (або певну кількість пакетів, якщо вікно вимірюється у пакетах), що відповідає кредиту. Але, якщо кредит вичерпаний, то передавальний вузол не має права передавати інформацію, поки не отримає наступний кредит. Під час перевантажень приймаючий вузол зменшує розмір вікна, тим самим знижуючи навантаження. Якщо ефект перевантаження зникає, то приймаючий вузол збільшує розмір вікна. Недоліком цього алгоритму є те, що він працює тільки в протоколах із встановленням з’єднання.

І, нарешті, у деяких випадках передавальний вузол визначає, що приймаючий вузол (або вузли) зазнає перевантаження, за деякими непрямими ознаками, без отримання повідомлення зворотного зв’язку. Такими непрямими ознаками можуть бути факти втрати пакетів. Для того, щоб протокол міг виявляти факти втрат пакетів, це має бути протокол із установленням з’єднання. Тоді закінчення тайм-ауту або прихід дубліката позитивної квитанції побічно свідчить про те, що пакет загублений. Однак втрата пакета не завжди свідчить про перевантаження мережі. Перевантаження мережі – це тільки одна з можливих причин втрати пакета, іншою причиною може бути ненадійна робота комунікаційних пристроїв (відмови устаткування, перекручування даних через завади). Але, оскільки реакція на перевантаження та ненадійну роботу мережі має бути однаковою й полягати в зниженні швидкості передачі, то неоднозначність причини втрати пакета не є проблемою.

Прикладом протоколу, що використовує неявну інформацію про перевантаження, є протокол TCP. Цей протокол за допомогою явної інформації зворотного зв’язку (про розмір вікна) здійснює контроль потоку, а за допомогою неявної (втрати пакетів, дублікати квитанцій) – керує перевантаженням.

4.2 Резервування ресурсів та контроль доступу

Розглянуті методи підтримки якості обслуговування орієнтовані в основному на боротьбу з перевантаженнями або запобігання їх у межах окремого вузла мережі. Разом з тим зрозуміло, що для забезпечення гарантованого рівня якості обслуговування деякого потоку пакетів необхідно скоординоване застосування цих методів на всьому шляху проходження потоку через мережу. Такою координуючою процедурою є процедура резервування ресурсів мережі для певного потоку. Ця процедура дозволяє налаштувати всі

механізми підтримки якості обслуговування уздовж проходження потоку так, щоб потік з деякими заданими характеристиками швидкості був обслужений із заданими характеристиками QoS.

Основна ідея процедури в тому, що кожному вузлу мережі уздовж маршруту проходження потоку задається питання, чи може цей вузол обслужити деякий новий потік із заданими характеристиками QoS, якщо відомі граничні характеристики швидкості потока, такі як середня й пікова швидкості? Кожний вузол при відповіді на це запитання має оцінити свої можливості, тобто перевірити, чи достатньо у нього вільних ресурсів, щоб прийняти на обслуговування новий потік і обслуговувати його якісно. При позитивній відповіді вузол має певним чином зарезервувати частину своїх ресурсів для даного потоку, щоб при надходженні пакетів потока на вхідні інтерфейси використовувати ці ресурси для їхнього обслуговування з гарантованим рівнем якості.

У загальному випадку кожний вузол самостійно вирішує, які ресурси він має резервувати для обслуговування деякого потоку із заданою якістю. Як показує практика, основним ресурсом, необхідним для якісного обслуговування пакетів, є пропускна здатність інтерфейсу, через який пакети потоку залишають вузол. Тому надалі ми, трохи спрощуючи ситуацію, застосовуватимемо формулювання «резервування пропускної здатності» замість «резервування ресурсів».

Сутність резервування полягає в тому, щоб обмежити рівень перевантажень певного потоку або декількох потоків деякою прийнятною величиною. Ця величина має бути такою, щоб механізми QoS (керування чергами, кондиціювання трафіка та зворотного зв’язку), що застосовуються у вузлах мережі, справлялися з короткочасними невеликими перевантаженнями й забезпечували необхідні значення характеристик QoS.

Однак, що ж означає резервування пропускної здатності в мережах з комутацією пакетів? Раніше ми не зустрічалися з таким механізмом, усі попередні пояснення роботи мережних пристроїв обходилися без нього. Річ у тім, що цей механізм не є обов’язковим, а використовується лише тоді, коли потрібне гарантоване виконання вимог якості обслуговування пакетів.

Резервування пропускної здатності в мережах з комутацією пакетів схоже на аналогічну процедуру в мережах з комутацією каналів тим, що певному потоку даних призначається певна частина пропускної здатності лінії зв’язку. Але тут це призначення набагато гнучкіше, – якщо відведена пропускна здатність у деякий період часу недовикористовується потоком, то вона може бути передана іншим потокам.

Ця обставина дозволяє ефективніше витрачати пропускну здатність ліній зв'язку, але призводить до ефекту поступової деградації якості транспортного сервісу через перевантаження, й черги замість простої відмови в обслуговуванні, що має місце в мережі з комутацією каналів, коли пропускної здатності виявляється недостатньо для обслуговування деякого потоку. Мета гнучкого резервування – забезпечити потік зарезервованою пропускною здатністю в ті періоди, коли вона йому потрібна вся, тобто в періоди перевантажень. Іншою відмінністю резервування в пакетних мережах є та обставина, що воно може виконуватися не лише «від краю до краю», але й для якихось окремих вузлів на маршруті потоку, але цей випадок не може гарантувати необхідний рівень характеристик QoS, оскільки перевантаження навіть в одному вузлі можуть призвести до затримок і втрат пакетів.

Резервування пропускної здатності в пакетній мережі «від краю до краю» починається з операції, яка називається контролем допуску в мережу потоку, що просить зарезервувати для свого обслуговування деяку пропускну здатність мережі між її двома кінцевими вузлами. Ця операція полягає в перевірці наявності доступної (тобто незарезервованої для інших потоків) пропускної здатності у кожному з вузлів мережі протягом усього маршруту проходження потоку (тут ми не зупиняємося на проблемі пошуку маршруту потоку, вона докладно розглядається у розділі «Інжиніринг трафіка»). Зрозуміло, що максимальна середня швидкість потоку має бути меншою, ніж запитувана пропускна здатність, інакше потік буде обслужений з надто поганою якістю навіть незважаючи на те, що йому була зарезервована певна пропускна здатність.

Якщо результат контролю допуску позитивний у кожному вузлі (випадок, показаний на рис. 4.2), то мережні пристрої запам'ятовують факт резервування, щоб з появою пакетів даного потоку розпізнати їх і виділити їм зарезервовану пропускну здатність. Крім того (у перспективі), при успішному резервуванні доступна для резервування пропускна здатність зменшується на величину, зарезервовану за даним потоком. Як видно з опису процедури, для її реалізації необхідно знати маршрут проходження потоку, для якого виконується резервування. У мережах з розподіленим принципом побудови таблиць маршрутизації, коли кожний мережний пристрій самостійно визначає наступний вузол на маршруті, з'ясування маршруту може бути досить складним завданням. Але ми залишимо дослідження цієї проблеми до розгляду конкретних технологій у подальших лекціях, а поки вважатимемо, що маршрут нам відомий.

Слід підкреслити, що резервування – це процедура, яка виконується перед тим, як реальний трафік буде спрямований у мережу.

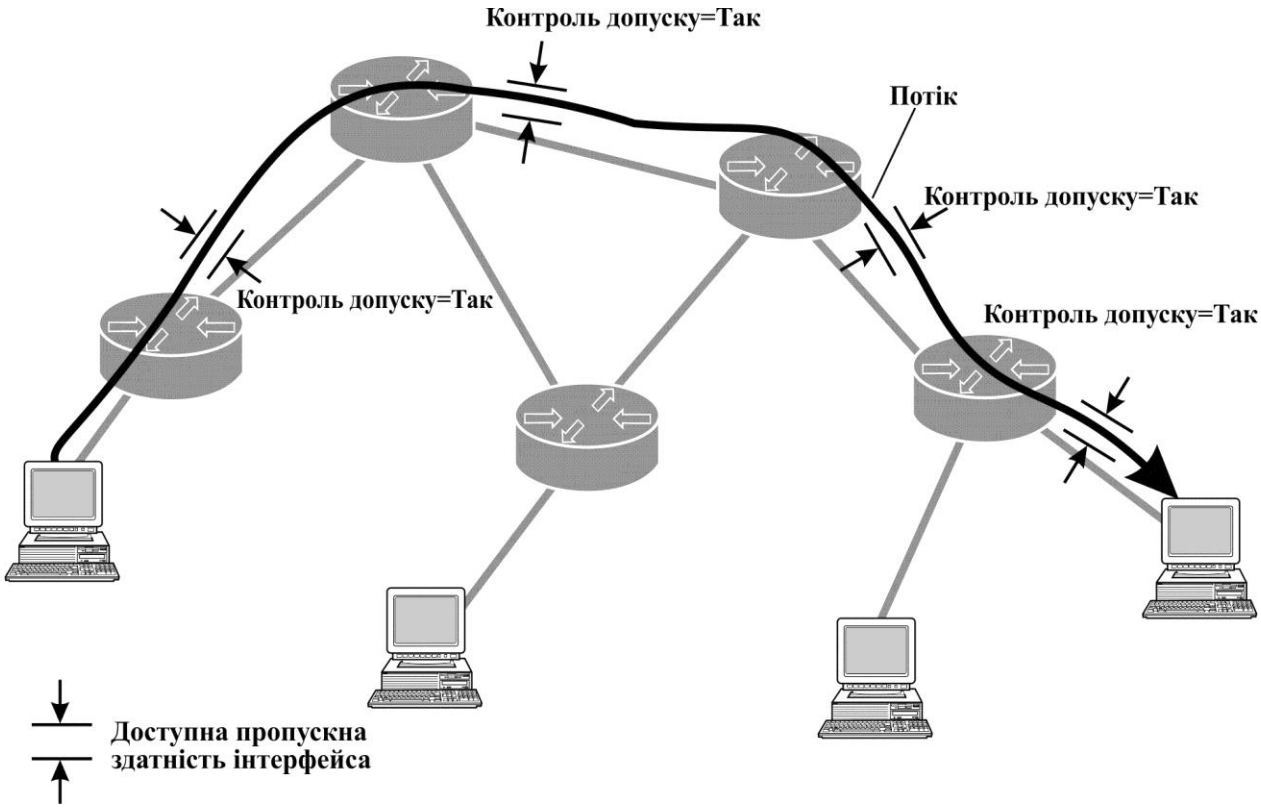


Рисунок 4.2 – Контроль допуску потоку

Давайте тепер подивимося, як саме виконується власне виділення пропускної здатності потоку в моменти часу, коли його пакети надходять на вхід комунікаційного пристрою S2, який запам’ятав факт резервування пропускної здатності для потоку F1 на вихідному інтерфейсі P2 (рис. 4.3).

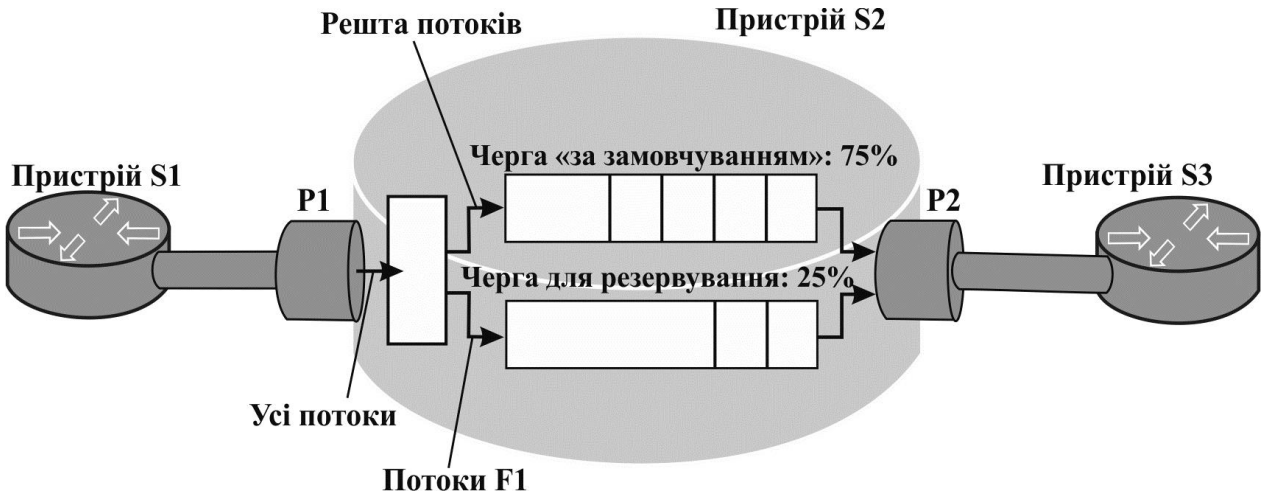


Рисунок 4.3 – Виділення зарезервованої пропускної здатності

Таке виділення можна забезпечити різними способами. У нашому прикладі це буде зроблено з використанням зважених черг. Нехай потоку F1 під час резервування було виділено 25% пропускної здатності інтерфейса P2

(звичайне резервування можна виконувати як в абсолютних величинах, наприклад, у мегабітах за секунду, так і у відсотках; це, власне, деталі реалізації механізмів QoS у конкретних пристроях). Також для простоти припустимо, що резервування було виконано лише для потоку F1, але для всіх інших потоків, які проходять через вихідний інтерфейс P2, резервування не проводилося.

Щоб досягти бажаного результату, достатньо організувати для вихідного інтерфейса дві зважені черги – чергу для потоку F1 з вагою 25% і чергу «за замовчуванням» для всіх інших потоків. Крім того, необхідно активізувати класифікатор, який перевірятиме пакети на всіх вхідних інтерфейсах пристрою S2 (на рис. 4.3 показаний лише один вхідний інтерфейс P1), відбирати пакети потоку F1 за заданими під час резервування ознаками і направляти їх у чергу для потоку F1. У ті періоди, коли швидкість потоку F1 виявиться меншою зарезервованої пропускної здатності в 25%, невикористана її частина використовуватиметься потоками з черги «за замовчуванням» – у силу алгоритму роботи зважених черг. Зате в періоди, коли швидкість потоку F1 досягне заявленого максимуму середньої швидкості в 25%, вся зарезервована пропускна здатність вихідного інтерфейса виділятиметься потоку F1, а всі інші потоки задовольнятимуться 75%, які залишилися. Значення в 75% може виявитися недостатньо для якісного обслуговування цих потоків, і тоді їхні пакети затримуватимуться або, навіть, губитимуться під час переповнення черги «за замовчуванням». Може статися й так, що значення в 75% виявиться занадто великим для інших потоків, і вони обслуговуватимуться з високою якістю. Яка з двох ситуацій спостерігатиметься частіше, ми не знаємо, оскільки у нас немає ніякої попередньої інформації про «інші» потоки. Цей приклад добре ілюструє особливість методів забезпечення параметрів QoS, – вони вимагають контролю над потоками, тобто знання їхніх маршрутів і середніх швидкостей. У протилежному випадку гарантій параметрів QoS досягти важко, можна говорити лише про обслуговування «за можливістю».

В описаному прикладі не використаний механізм профілювання трафіка. За наявності окремої зваженої черги для потоку, який зарезервував пропускну здатність, цей механізм не є обов’язковим, оскільки сам механізм зважених черг обмежить пропускну здатність потоку в потрібних межах у періоди перевантажень, коли всі зважені черги заповнюються повністю. Але кількість зважених черг у мережних пристроях зазвичай обмежена не занадто великою величиною, наприклад, їх може бути не більше 16 або 32. Однак, кількість потоків, для яких бажано зарезервувати пропускну здатність, може бути значно

більшою. У такій ситуації можна організувати одну зважену чергу для усіх резервованих потоків з пропускною здатністю, що дорівнює або більша суми резервованих пропускних здатностей потоків. А для того, щоб необхідні частки пропускної здатності виділялися кожному потоку, необхідно після класифікації виконати профілювання кожного потоку на рівні запитаної ним швидкості. Правда, в цьому випадку в періоди неактивності інших потоків ми втрачаємо можливості надавати окремим потокам більше пропускної здатності, ніж вони запросили, але це плата за масштабоване рішення, що базується на одній зваженій резервованій черзі.

Використання зважених черг – не єдиний варіант резервування пропускної здатності в пакетних мережах. З цією метою можна задіяти пріоритетні черги. Застосування пріоритетної черги може бути не тільки можливим, але й необхідним, якщо потоку крім певного рівня пропускної здатності потрібно забезпечити мінімально можливий рівень затримок пакетів.

Під час використання пріоритетної черги профілювання необхідне завжди, оскільки пріоритетний механізм не забезпечує обмеження швидкості потоку, як це робить механізм зваженого обслуговування.

Слід підкреслити, що резервування призводить до очікуваних результатів лише в тих випадках, коли реальна швидкість потоків, для яких було виконане резервування, виявляється не вищою, ніж пропускна здатність, запитана під час резервування й реалізована в ході конфігурування мережних пристроїв. У протилежному випадку результати можуть виявитися навіть гірші, ніж за наявності єдиної черги «за замовчуванням» і обслуговуванні «за можливістю». Так, якщо швидкість потоку виявиться вищою, ніж межа, що враховується механізмом профілювання, то частина пакетів буде відкинута навіть у тому випадку, якщо пристрій не перевантажений і може відмінно впоратися із запропонованим трафіком без застосування механізмів QoS.

Що ж змінюється в мережі після резервування? Під час надходження на вхідний інтерфейс комутатора пакетів потоку, для яких було виконане резервування, механізм класифікації розпізнає пакети, які належать до цього потоку, і направляє їх у потрібну чергу. При цьому пакети можуть проходити через механізм профілювання, покликаний запобігти ситуації обслуговування потоку, швидкість якого перевищує зазначену при резервуванні.

У результаті резервування мережа виявляється завантаженою раціонально. В ній немає ресурсів, які працюють зі значним перевантаженням. Механізми організації черг, як і раніше, забезпечують тимчасову буферизацію пакетів у періоди пульсацій. Оскільки ми планували завантаження ресурсів з розрахунку

середніх швидкостей передачі даних, то на періодах пульсацій протягом деякого обмеженого часу швидкості потоків можуть перевищувати середні швидкості, тому механізми боротьби з перевантаженнями, як і раніше, потрібні. Для забезпечення необхідних середніх швидкостей потоків на періодах перевантажень відповідні потоки можуть обслуговуватися за допомогою зважених черг. Зберігається також головна перевага методу комутації пакетів: якщо деякий потік не витрачає відведеної йому пропускної здатності, то вона може виділятися для обслуговування іншого потоку. Нормальною практикою є резервування пропускної здатності тільки для частини потоків, у той час як інші потоки обслуговуються без резервування, отримуючи обслуговування за можливістю (з максимальними зусиллями). Тимчасово вільна пропускна здатність може для таких потоків виділятися динамічно, без порушення взятих зобов'язань з обслуговування потоків, для яких ресурси зарезервовані.

Забезпечення заданого рівня затримок

Під час опису процедури резервування пропускної здатності ми сфокусувалися на механізмах виділення пропускної здатності деякому потоку й залишили без уваги одну важливу деталь: яку пропускну здатність має запитувати потік для того, щоб затримки його пакетів не перевищували певної величини? Єдина думка, що була висловлена з цього приводу: запитувана пропускна здатність має бути вищою, ніж максимальна швидкість потоку, інакше деяка частина пакетів просто може постійно відкидатися мережею, так що якість обслуговування виявиться гарантовано низькою.

Однак ця «деталь» насправді обертається складною проблемою, оскільки ми не можемо, наприклад, сконфігурувати чергу пріоритетного або зваженого обслуговування так, щоб вона чітко забезпечила будь-який заздалегідь заданий поріг затримок і його варіації. Те, що пакети направлені у пріоритетну чергу лише дозволяє гарантувати, що затримки будуть досить низькими – значно нижче, ніж у пакетів, які обробляються в черзі за замовчуванням. Ми також знаємо, що за наявності зважених черг затримки знижуватимуться зі зниженням відносного коефіцієнта використання пропускної здатності, відведеної черги. Але це все якісні показники, а от кількісно оцінити значення затримок дуже складно.

Як саме постачальник послуг може виконати свої зобов'язання перед клієнтами? Дуже «просто» – він має постійно вимірювати фактичні значення характеристик трафіка в мережі й гарантувати користувачам мережі величини затримок відповідно до спостережуваних результатів.

Приклад

Нехай мережа пропонує три рівні якості обслуговування трафіка: золотий – для дуже чутливого до затримок трафіка, срібний – для трафіка, що чутливий до затримок і потребує гарантованої пропускної здатності, та бронзовий – для трафіка, що обслуговується за можливістю. Оператор мережі може різними способами досягти того, щоб на золотому рівні обслуговування дійсно гарантувалися дуже низькі величини затримок, варіацій затримок і втрат пакетів для трафіка, на срібному – досить низькі значення цих характеристик, але вище, ніж у золотого, а на бронзовому гарантувалися лише певні величини втрат пакетів і зовсім не гарантувалися значення затримок. Для реалізації такої стратегії обслуговування оператор може, наприклад, організувати на всіх комутаторах мережі пріоритетну чергу для обслуговування золотого трафіка й відвести їй 25% пропускної здатності на кожному вихідному інтерфейсі; зважену чергу з 50% пропускної здатності для срібного трафіка та зважену чергу з тими 25%, що залишилися, – для бронзового трафіка. А далі він повинен приймати на обслуговування потоки користувачів у кожний клас і виконувати постійний моніторинг характеристик трафіка кожного класу. І, якщо, наприклад, моніторинг показує, що затримки в 95% пакетів золотого трафіка не перевищують 15 мс, то оператор може гарантувати цю величину користувачам золотого рівня обслуговування. Та оскільки оператору треба бути готовим до прийому на обслуговування нових користувачів, то природно було б залишати деякий запас і гарантувати, скажімо, затримку в 20 мс замість фактичного значення в 15 мс. Аналогічно слід вчинити зі срібним трафіком, а для бронзового достатньо вимірювати лише частку втрат пакетів. У тому випадку, коли моніторинг показує наближення фактичних параметрів трафіка певного рівня обслуговування до гарантованих, можна або додати пропускну здатність для черг цього класу, або припинити прийом нових користувачів до цього класу.

Як видно навіть із цього стислого опису, гарантування рівня затримок у мережі є досить складною справою.

Цим пояснюється той факт, що часто оператори віддають перевагу якісному опису різних класів послуг, говорячи, наприклад, про мінімальні затримки найвищого класу обслуговування, але не даючи кількісних гарантій.

Інжиніринг трафіка

Під час розгляду системи забезпечення якості обслуговування, що заснована на резервуванні, ми не стали торкатися питання маршрутів проходження потоків через мережу. Ми прийняли, що маршрути вже обрані, причому цей вибір зроблено без урахування вимог QoS. І в умовах заданості маршрутів ми намагалися забезпечити проходження по цих маршрутах такого набору потоків, для якого можна гарантувати дотримання вимог QoS.

Очевидно, що завдання забезпечення вимог QoS можна вирішити більш ефективно, якщо прийняти, що маршрути проходження трафіка не фіксовані, а також підлягають вибору. Це дозволило б мережі обслуговувати більше потоків з гарантіями QoS за ти самих характеристиках самої мережі, тобто пропускної здатності каналів і продуктивності комутаторів і маршрутизаторів.

Завдання вибору маршрутів для потоків (або класів) трафіка з урахуванням дотримання вимог QoS вирішують методи інжинірингу трафіка (Traffic Engineering, TE). За допомогою цих методів прагнуть домогтися ще однієї мети – за можливістю максимально та збалансовано завантажити всі ресурси мережі, щоб мережа при заданому рівні якості обслуговування мала як можна вищу сумарну продуктивність.

Методи TE, як і інші розглянуті раніше методи, засновані на резервуванні ресурсів. Тобто вони не тільки дозволяють знайти раціональний маршрут для потоку, але й резервують для нього пропускну здатність ресурсів мережі, які розташовані уздовж цього маршруту.

Методи інжинірингу трафіка порівняно нові для мереж з комутацією пакетів. Багато в чому це пояснюється тим, що передача еластичного трафіка не висувала жорстких вимог до параметрів QoS. Крім того, Інтернет тривалий час не був комерційною мережею, тому завдання максимального використання ресурсів не вважалося першочерговим для IP-технологій, що лежать в основі Інтернету.

Сьогодні ситуація змінилася. Мережі з комутацією пакетів мають передавати різні види трафіка із заданою якістю обслуговування, максимально використовуючи можливості своїх ресурсів. Але для цього їм слід змінити деякі, що вже стали традиційними, підходи до вибору маршрутів.

Недоліки традиційних методів маршрутизації

Основним принципом роботи протоколів маршрутизації в мережах з комутацією пакетів уже тривалий час є вибір маршруту на основі топології мережі без урахування інформації про її поточне завантаження.

Для кожної пари «адреса джерела – адреса призначення» такі протоколи обирають єдиний маршрут, не зважаючи інформаційні потоки, які проходять через мережу. Внаслідок цього всі потоки між парами кінцевих вузлів мережі йдуть найкоротшим (відповідно до деякої метрики) маршрутом. Обраний маршрут може бути більш раціональним, наприклад, якщо взяти до уваги номінальну пропускну здатність каналів зв'язку або внесені ними затримки, чи менш раціональним, якщо враховується лише кількість проміжних маршрутизаторів між вихідним і кінцевим вузлами.

***УВАГА**. У традиційних методах маршрутизації найкращий обраний маршрут розглядається як єдино можливий, навіть якщо існують інші, хоча й дещо гірші маршрути.*

Класичним прикладом неефективності такого підходу є так звана «риба» – мережа з топологією, наведеною на рис. 4.4. Незважаючи на те, що між комутаторами A і E існує два шляхи (верхній – через комутатор B і нижній – через комутатори C і D), весь трафік від комутатора A до комутатора E відповідно до традиційних принципів маршрутизації подається верхнім шляхом. Лише тому, що нижній шлях трохи довший (на одну ретрансляційну ділянку) від верхнього, він ігнорується, хоча міг би працювати «паралельно» з ним.

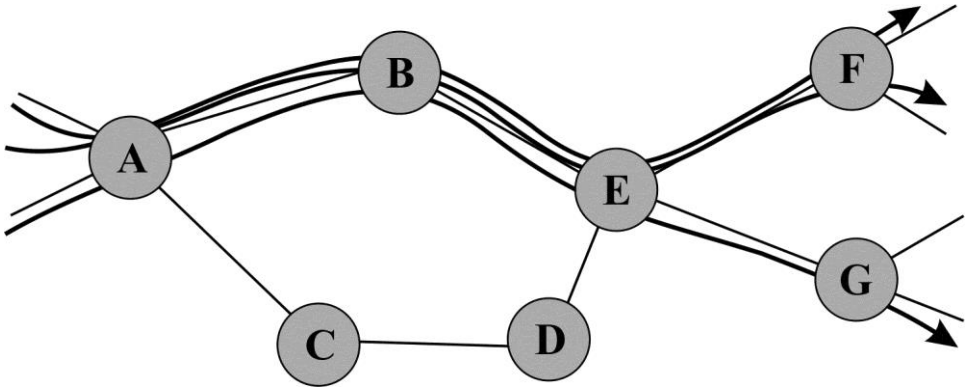


Рисунок 4.4 – Неефективність найкоротших шляхів

Такий підхід призводить до того, що навіть якщо найкоротший шлях перевантажений, пакети однаково посилають цим шляхом. Так, у мережі, що подана на рис. 4.4, верхній шлях продовжуватиме використовуватися навіть тоді, коли його ресурсів не вистачатиме для обслуговування трафіка від комутатора A до комутатора E, а нижній шлях простоюватиме, хоча, можливо, ресурсів комутаторів вистачило б для якісної передачі цього трафіка.

Тут наявна неповноцінність методів розподілу ресурсів мережі – одні ресурси працюють з перевантаженням, а інші не використовуються зовсім. Традиційні методи боротьби з перевантаженнями цю проблему вирішити не в змозі, тому потрібні принципово інші механізми.

Методи інжинірингу трафіка

Вихідними даними для методів інжинірингу трафіка є:

• характеристики передавальної мережі – її топологія і продуктивність складових її комутаторів та ліній зв'язку (рис. 4.5);

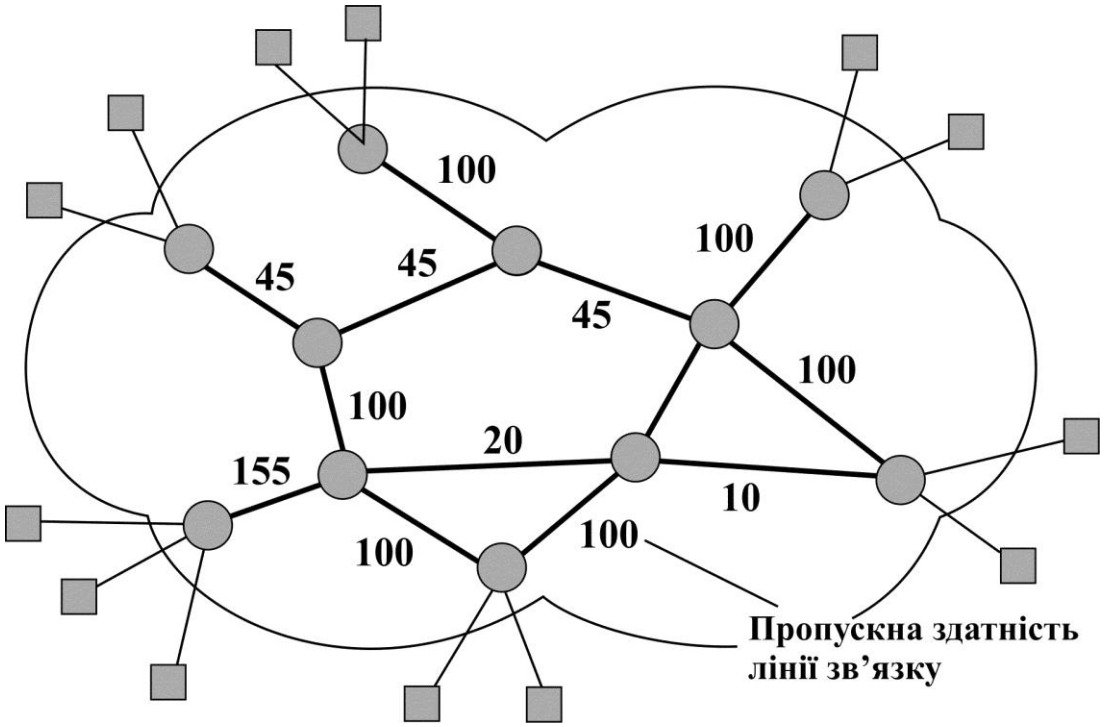


Рисунок 4.5 – Топологія мережі й продуктивність її ресурсів

• відомості про запропоноване навантаження мережі, тобто про потоки трафіка, які мережа має передавати між своїми примежовими комутаторами (рис. 4.6).

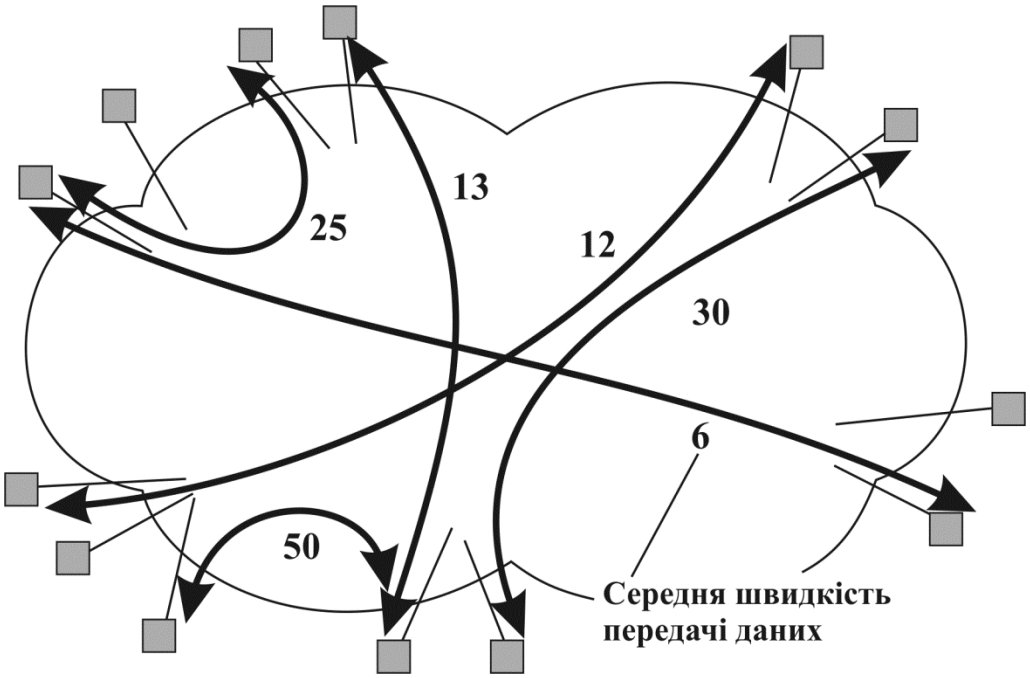


Рисунок 4.6 – Запропоноване навантаження

Нехай продуктивність процесора кожного комутатора достатня для обслуговування трафіка всіх його вхідних інтерфейсів, навіть якщо трафік надходить на інтерфейс із максимально можливою швидкістю, яка дорівнює пропускній здатності інтерфейса. Тому, під час резервування ресурсів, вважатимемо ресурсами пропускну здатність ліній зв'язку між комутаторами, що визначає також пропускну здатність двох інтерфейсів, зв'язаних цією лінією.

Вважатимемо, що розв'язком завдання ТЕ є такий набір маршрутів для заданої множини потоків трафіка, для якого всі значення коефіцієнтів використання ресурсів уздовж маршруту проходження кожного потоку не перевищують деякого заданого порога $K_{\max}$.

На рис. 4.7 показаний один з можливих розв'язків завдання, яке ілюструють рис. 4.5 і 4.6. Знайдені маршрути гарантують, що максимальний коефіцієнт використання будь-якого ресурсу для будь-якого потоку не перевищує 0,6.

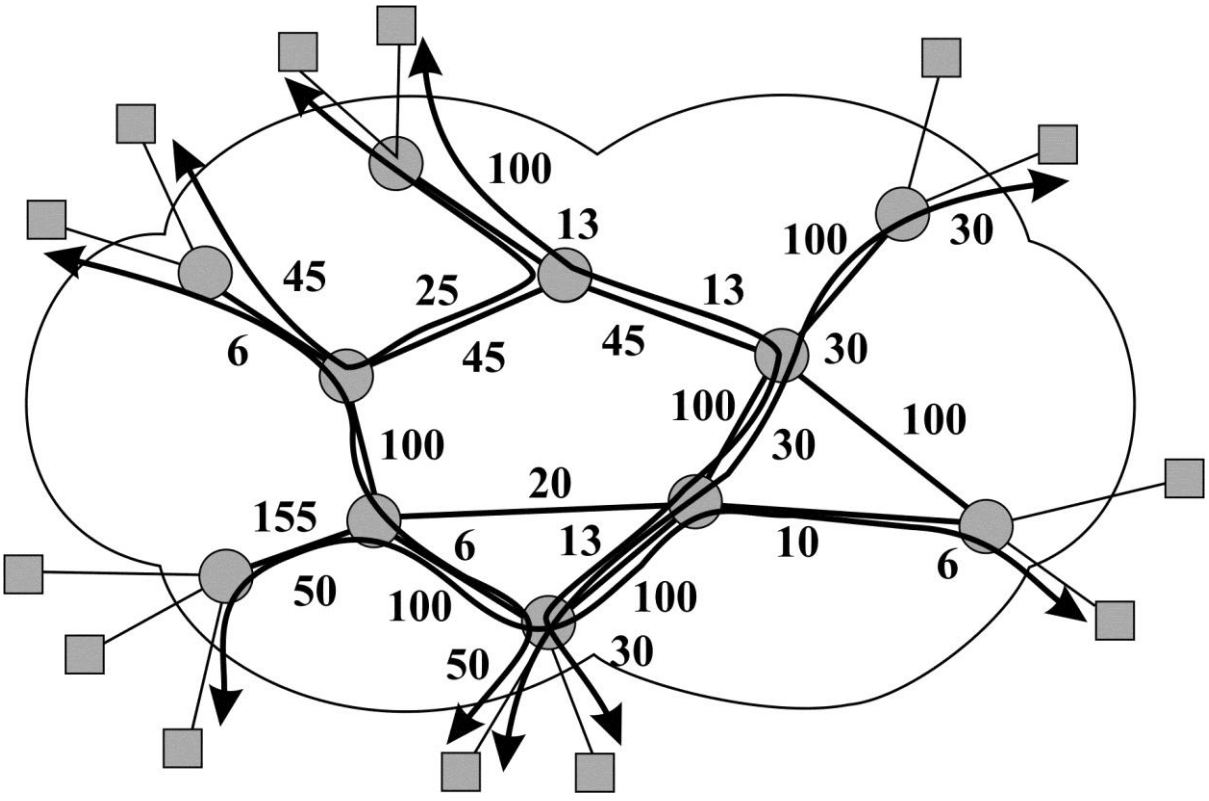


Рисунок 4.7 – Розподіл навантаження по мережі – вибір шляхів передачі трафіка

Розв'язок завдання ТЕ можна шукати по-різному.

По-перше, можна шукати його завчасно, у фоновому режимі. Для цього потрібно знати вихідні дані: топологію й продуктивність мережі, вхідні й вихідні точки потоків трафіка, середню швидкість передачі даних у них. Після цього завдання раціонального розподілу шляхів проходження трафіка

при фіксованих точках входу й виходу, а також заданому рівні максимального значення коефіцієнта використання ресурсу можна передати деякій програмі, яка, наприклад, за допомогою спрямованого перебору варіантів знайде розв'язок. Результатом роботи програми будуть точні маршрути для кожного потоку із зазначенням усіх проміжних комутаторів.

По-друге, завдання ТЕ можна вирішувати в оперативному режимі, доручивши її комутаторам мережі. Для цього використовуються модифікації стандартних протоколів маршрутизації.

Модифікація протоколів маршрутизації полягає в тому, що вони повідомляють один одному не тільки топологічну інформацію, але й поточне значення вільної пропускної здатності для кожного ресурсу.

Після того, як розв'язок знайдено, його потрібно реалізувати, тобто втілити в таблицях маршрутизації. На цьому етапі може виникнути проблема – у тому випадку, якщо ми хочемо прокласти ці маршрути в дейтаграмній мережі. Річ у тім, що таблиці маршрутизації в них враховують лише адреси призначення пакетів. Комутатори й маршрутизатори таких мереж (наприклад, IP-мереж) не працюють з потоками, для них потоків у явному вигляді не існує, кожен пакет у ході його просування є незалежною одиницею комутації. Можна сказати, що таблиці просування цих мереж відбивають лише топологію мережі (напрямок просування до певних адрес призначення).

Тому привнесення методів резервування в дейтаграмні мережі відбувається з великими труднощами. У протоколах резервування, щоб визначити потік для дейтаграмного маршрутизатора, крім адреси призначення використовується деякий додатковий набір ознак. При цьому поняття потоку потрібно лише на етапі резервування, а під час просування пакетів, як і раніше, працює традиційна для цього типу мереж схема, що враховує лише адресу призначення.

Тепер спробуємо уявити ситуацію, коли в нас є кілька потоків між двома кінцевими вузлами, і ми хочемо направити їх різними маршрутами. Ми прийняли таке рішення, виходячи з балансу завантаження мережі, коли вирішували завдання інжинірингу трафіка. Дейтаграмний комутатор або маршрутизатор не має можливості реалізувати наше рішення, оскільки для всіх цих потоків у нього в таблиці просування є лише один запис, який відповідає загальній адресі призначення пакетів цих потоків. Змінювати логіку роботи комутаторів і маршрутизаторів дейтаграмних мереж недоцільно, оскільки це занадто принципова модернізація.

У результаті методи інжинірингу трафіка сьогодні використовуються лише в мережах з віртуальними каналами, для яких не становить труднощів

реалізувати знайдене рішення для групи потоків. Кожному потоку (або групі потоків з однаковими маршрутами) виділяється віртуальний канал, який прокладається відповідно до обраного маршруту. Методи інжинірингу трафіка успішно застосовуються в мережах ATM і Frame Relay, що працюють на основі техніки віртуальних каналів. IP-мережі також опираються на методи TE, коли ті використовуються в мережах ATM або Frame Relay, що працюють у складеній мережі, побудованій на основі протоколу IP. Існує також порівняно нова технологія MPLS, що розроблена спеціально як засіб привнесення техніки віртуальних каналів у IP-мережі. На основі технології MPLS у IP-мережах можна також вирішувати завдання TE.

Інжиніринг трафіка різних класів

Під час розв'язання завдання інжинірингу трафіка ми вважали, що всі потоки трафіка висували однакові вимоги до якості обслуговування. Тобто користувачів мережі задовольняло, що всі потоки обслуговуються із заданою середньою швидкістю (вона, природно, у кожного потоку своя і відрізняється від інших).

Більш реальною є ситуація, коли у кожного користувача мережі є кілька класів трафіка, і ці класи відрізняються різними вимогами до якості обслуговування. Ця проблема вже обговорювалася під час розгляду питань резервування ресурсів.

У методах TE, що враховують наявність у мережі трафіка з різними вимогами QoS, проблема вирішується так само, як і в методах резервування ресурсів окремих вузлів. Якщо в нас є, наприклад, два класи трафіка, то ми задаємося двома рівнями максимального використання ресурсів.

Для досягнення такого результату з кожним ресурсом мають бути пов'язані два лічильники вільної пропускної здатності – один для пріоритетного, другий – для еластичного трафіка. Під час визначення можливості проходження маршруту через конкретний ресурс для пріоритетного трафіка середня інтенсивність нового потоку має дорівнювати вільній пропускній здатності для пріоритетного трафіка.

Якщо вільної пропускної здатності достатньо і новий потік піде через даний інтерфейс, то значення середньої швидкості передачі даних для нового потоку необхідно відняти як з лічильника завантаження пріоритетного трафіка, так і з лічильника завантаження еластичного трафіка, оскільки пріоритетний трафік завжди обслуговуватиметься перед еластичним і створить для еластичного трафіка додаткове навантаження. Якщо ж завдання TE

вирішується для еластичного трафіка, то його середня швидкість передачі даних дорівнює вільній пропускній здатності лічильника еластичного трафіка й у випадку позитивного рішення значення цієї швидкості віднімається лише з лічильника еластичного трафіка, оскільки для пріоритетного трафіка еластичний трафік прозорий.

Модифіковані протоколи маршрутизації мають поширювати по мережі інформацію про два параметри вільної пропускної здатності – для кожного класу трафіка окремо. Якщо ж завдання узагальнюється для випадку передачі через мережу трафіка декількох класів, то, відповідно, з кожним ресурсом має бути зв'язано стільки лічильників, скільки класів трафіка існує в мережі, а протоколи маршрутизації мають поширювати вектор вільних пропускних здатностей відповідної розмірності.

Робота в недовантаженому режимі

Як ми вже зазначали, найпростішим способом забезпечення вимог QoS для всіх потоків є робота мережі в недовантаженому режимі, або ж з надлишковою пропускною здатністю.

Говорять, що мережа має надлишкову пропускну здатність, коли всі частини мережі в будь-який момент часу мають таку пропускну здатність, якої достатньо, щоб обслужити всі потоки трафіка, що протікає в цей час через мережу, із задовільними характеристиками продуктивності й надійності. Інакше кажучи, жодне з мережних пристроїв такої мережі ніколи не піддається перевантаженням, які могли б призвести до значних затримок або втрат пакетів через переповнення черг пакетів (зазвичай, це не виключає випадків втрат мережею пакетів з інших причин, не пов'язаних з перевантаженням мережі, наприклад, через перекручування сигналів у лініях зв'язку або відмов мережних вузлів або ліній зв'язку).

Простота цього підходу є його головною перевагою, оскільки він вимагає лише збільшення пропускної здатності ліній зв'язку й, відповідно, продуктивності комунікаційних пристроїв мережі. Ніяких додаткових зусиль з дослідження характеристик потоків мережі й конфігурування додаткових черг і механізмів кондиціювання трафіка, як у випадку застосування методів QoS, тут не потрібно.

Зазначимо, що визначення мережі з надлишковою пропускною здатністю було навмисно спрощене, щоб передати суть ідеї. Більш точне визначення має враховувати випадковий характер процесів, що протікають у мережі, і оперувати статистичними визначеннями подій, тобто говорити, що такі події, як тривалі

затримки або втрати пакетів через переповнення черг у мережі з надлишковою пропускною здатністю, відбуваються так рідко, що ними можна знехтувати. У результаті трафік усіх додатків у подібній мережі переноситься з високою якістю.

Однак довести, що мережа дійсно є мережею з надлишковою пропускною здатністю, на практиці досить важко. Тільки постійне вимірювання часу доставки пакетів усім кінцевим вузлам мережі може показати, що мережа задовольняє даний опис, – ми вже стикалися з цією ситуацією, коли розглядали механізм гарантування певного рівня затримок пакетів під час застосування методів QoS. Але моніторинг затримок і їхніх варіацій є тонкою й трудомісткою роботою.

Зазвичай оператори, які хочуть підтримувати свою мережу в недовантаженому стані й за рахунок цього забезпечувати високу якість обслуговування, роблять простіше, – вони здійснюють моніторинг рівня трафіка в лініях зв'язку мережі, тобто вимірюють коефіцієнт використання пропускної здатності ліній зв'язку. При цьому лінія зв'язку вважається недовантаженою, якщо її коефіцієнт використання постійно не перевершує деякий досить низький рівень, наприклад 10%. Маючи такі значення вимірів, можна вважати, що лінія в середньому не відчуває перевантажень, а значить, затримки пакетів будуть низькими, – ми знаємо про таку залежність між коефіцієнтом завантаження ресурсу й затримками з теорії масового обслуговування, розглянутої на прикладі найпростішої моделі $M/M/1$.

Однак навіть настільки низькі значення завантаження не виключають появи на лінії короткочасних пульсацій трафіка, здатних призводити до підвищення пікової швидкості трафіка до величини пропускної здатності лінії та, отже, до значних затримок або втрат невеликої кількості пакетів. Для деяких типів додатків такі втрати можуть бути досить чутливими.

Багато засобів моніторингу швидкості трафіка, особливо вбудовані в комутатори й маршрутизатори, вимірюють швидкість трафіка, усереднюючи її на занадто довгих інтервалах. У результаті такі засоби моніторингу просто не здатні зареєструвати короткочасні пульсації трафіка й часто дають занадто оптимістичну оцінку завантаженості мережі.

Цю проблему ілюструє рис. 4.8. На ньому показано результати вимірювання швидкості трафіка на інтерфейсі з пропускною здатністю в 2 Мбіт/с.

На рисунку подані три криві, отримані для одного й того самого трафіка при різних інтервалах усереднення даних. Сірою суцільною лінією показані результати, отримані для інтервалу усереднення даних в 1 мс; пунктирна чорна

лінія демонструє результати для інтервалу усереднення в 2 мс, а штрих-пунктирна чорна лінія відповідає інтервалу в 25 мс.

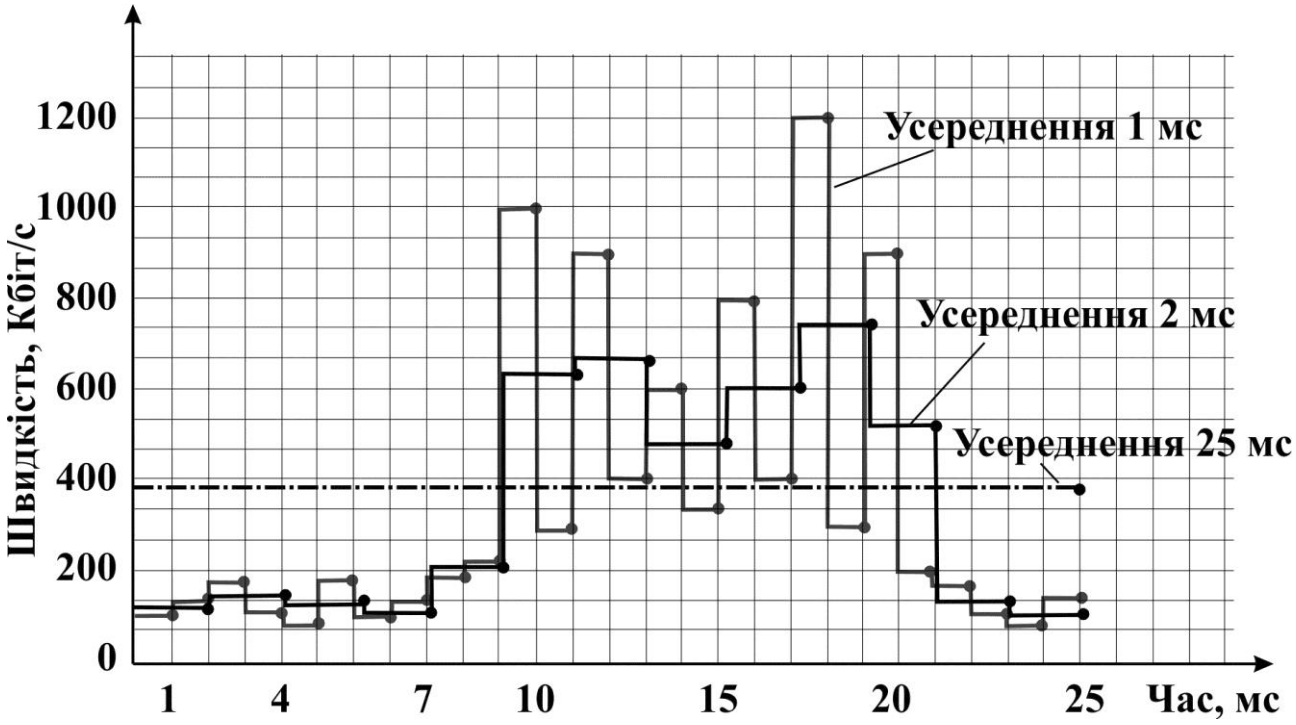


Рисунок 4.8 – Залежність результатів вимірювань швидкості трафіка від часу усереднення

Звичайна практика для оцінки стану недовантаження інтерфейсу полягає у використанні межі в 25% від його пропускної здатності як індикатора недовантаження. Для нашого прикладу це відповідає швидкості трафіка 500 Кбіт/с.

Тоді, використовуючи результати моніторингу інтерфейсу з інтервалом усереднення в 25 мс, ми вважаємо, що інтерфейс недовантажений і нам не слід турбуватися про можливі затримки та втрати пакетів через перевантаження інтерфейсу. Однак дивлячись на сіру криву (усереднення 1 мс), ми бачимо, що в шести інтервалах швидкість набагато перевищувала 500 Кбіт/с, а значить, на цих інтервалах тривалі затримки й втрати пакетів цілком могли статися. Нарешті, дані, отримані при усередненні в 2 мс, показують, що інтерфейс перебуває поблизу межі недовантаження.

Дані, використані для побудови кривих на рис. 4.8, були штучно підібрані так, щоб показати крайні ситуації. Але ці криві дійсно відбивають тонкий і важливий ефект вимірювань, який необхідно враховувати під час моніторингу завантаження ліній зв’язку мережі: занадто тривалі інтервали усереднення в ході вимірюваня швидкості можуть істотно викривити картину й призвести до втрати важливої інформації, а в остаточному підсумку –

до переоцінки можливостей мережі якісно передавати трафік. Часто на практиці виконують моніторинг завантаження ліній зв'язку з п'ятисекундним інтервалом усереднення, що явно недостатньо для оцінки стану мережі.

Для більш достовірної оцінки стану мережі слід доповнювати моніторинг завантаження ліній зв'язку мережі хоча б вибірковим моніторингом характеристик QoS, таких як затримки, варіації затримок і втрати пакетів. У цьому випадку можна з великою впевненістю говорити про те, що мережа дійсно є мережею з надлишковою пропускною здатністю, що гарантує всім типам трафіка якісне обслуговування.

Крім того, вибірковий моніторинг характеристик QoS може допомогти у визначенні межі завантаження ліній, що слугує для оцінки їхнього недовантаження. У нашому прикладі як межу ми використовували значення 25%, але цілком можливо, що це емпіричне значення для деякої конкретної мережі потрібно уточнювати.

Висновки

Якість обслуговування в його вузькому сенсі фокусує увагу на характеристиках і методах передачі трафіка через черги комунікаційних пристроїв. Методи забезпечення якості обслуговування займають сьогодні важливе місце в сімействі технологій мереж з комутацією пакетів, оскільки без їхнього застосування складно забезпечити якісну роботу сучасних мультимедійних додатків, таких як IP-телефонія, відео- і радіомовлення, інтерактивне дистанційне навчання тощо.

Характеристики QoS відбивають негативні наслідки перебування пакетів у чергах, які проявляються в зниженні швидкості передачі, затримках пакетів і їхніх втрат.

Існують різні типи трафіка, які відрізняються чутливістю до затримок і втрат пакетів. Найбільш груба класифікація трафіка поділяє його на два класи: трафік реального часу (чутливий до затримок) і еластичний трафік (нечутливий до затримок у широких межах).

Методи QoS засновані на перерозподілі наявної пропускної здатності ліній зв'язку між трафіком різного типу відповідно до вимог додатків.

Пріоритетні й зважені черги є основним інструментом виділення пропускної здатності певним потокам пакетів.

Механізм профілювання дозволяє контролювати швидкість потоку пакетів і обмежувати її відповідно із заздалегідь заданим рівнем.

Зворотний зв’язок є одним з механізмів QoS; він дозволяє тимчасово знизити швидкість надходження пакетів у мережу для ліквідації перевантаження у вузлі мережі.

Резервування пропускної здатності «від краю до краю» дозволяє домогтися гарантованої якості обслуговування для потоку пакетів. Резервування засноване на процедурі контролю допуску потоку в мережу, у ході якої перевіряється наявність доступної пропускної здатності для обслуговування потоку уздовж маршруту його проходження.

Методи інжинірингу трафіка полягають у виборі раціональних маршрутів проходження потоків через мережу. Вибір маршрутів забезпечує максимізацію завантаження ресурсів мережі при одночасному дотриманні необхідних гарантій відносно параметрів якості обслуговування трафіка.

Недовантажена мережа (вона ж мережа з надлишковою пропускною здатністю) може забезпечити якісне обслуговування трафіка всіх типів без застосування методів QoS; але для того щоб переконатися, що мережа дійсно недовантажена, потрібно постійно проводити моніторинг рівнів завантаження ліній зв’язку мережі, виконуючи вимірювання з досить високою частотою.

4.3 Контрольні запитання та завдання

1. Поясніть особливості керування чергами на основі зворотного зв’язку.
2. Перелічіть варіанти організації зворотного зв’язку.
3. Що належить до інформації зворотного зв’язку?
4. Поясніть сутність інжинірингу трафіка різних класів трафіка.
5. Поясніть сутність резервування пропускної здатності мережі.

Лекція 5
АЛГОРИТМИ МАРШРУТИЗАЦІЇ

5.1 Алгоритми побудови остовного дерева мінімальної ваги

Двома головними функціями, які виконує алгоритм маршрутизації, є вибір маршрутів для різних пар відправник–адресат і забезпечення правильної доставки повідомлень їхнім адресатам після того, як маршрути обрані. Друга функція забезпечується шляхом використання різних протоколів і структур даних (маршрутними таблицями). Основна увага приділятиметься першій функції (вибору маршрутів) і тому, як це впливає на характеристики мережі.

Існують дві основні характеристики, на які істотно впливає алгоритм маршрутизації – пропускна здатність (кількість обслуговувань) і середня затримка пакета (якість обслуговування). Маршрутизація взаємодіє з керуванням потоками у визначенні характеристик за допомогою механізму зворотного зв'язку, показаного на рис. 5.1. Коли трафік, що надходить у підмережу від зовнішніх джерел, відносно малий, він повністю приймається мережею й тоді

Пропускна здатність = Навантаження, що надходить.

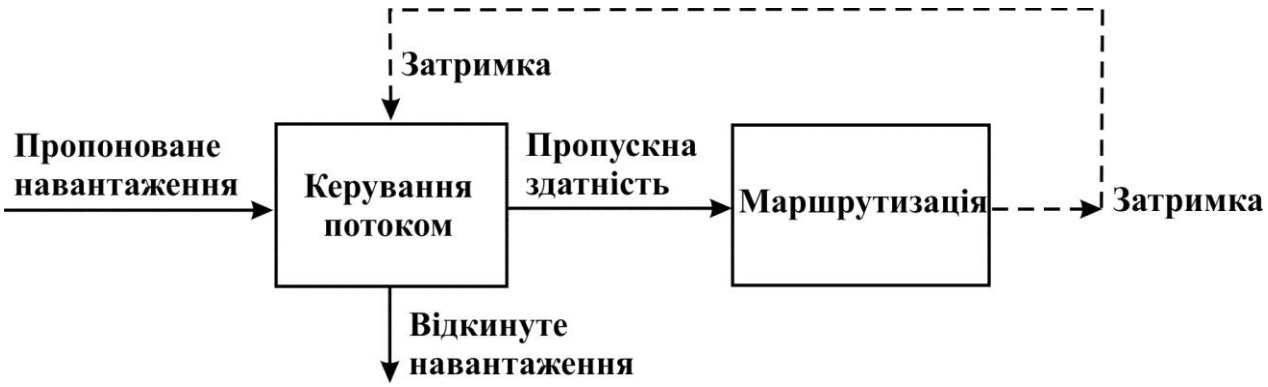


Рисунок 5.1 – Взаємозв'язок між маршрутизацією й керуванням потоком. Коли гарна маршрутизація зберігає затримку малою, керування потоком дозволяє приймати в мережу більше трафіка

Коли навантаження, що надходить, надмірно велике, частина цього навантаження буде відкидатися алгоритмом керування потоками й тоді

Пропускна здатність =
= Навантаження, що надходить – Відкинуте навантаження

Трафік, прийнятий у мережу, матиме середню затримку пакетів, що залежить від того, які маршрути були обрані алгоритмом маршрутизації.

Але на пропускну здатність істотно впливає також алгоритм маршрутизації (хоча, можливо, і не безпосередньо), оскільки алгоритм керування потоками зазвичай діє на основі підтримки балансу між пропускною здатністю та середньою затримкою, наприклад, навантаження, що надходить, не приймається, як тільки затримка стає занадто великою. Тому, якщо алгоритму маршрутизації вдається успішно підтримувати невелику затримку, то алгоритм керування потоками дозволяє приймати в мережу більше трафіка. Хоча точний баланс між затримкою та пропускною здатністю встановлюється алгоритмом керування потоками, гарна маршрутизація в умовах великого пропонованого трафіка дає більш бажану криву «затримка – пропускна здатність», згідно з якою діє алгоритм керування потоками (рис. 5.2).



Рисунок 5.2 – Криві залежності затримки від пропускної здатності для гарної й поганої маршрутизації

Існує кілька способів класифікації алгоритмів маршрутизації.

Один з них полягає в *поділі всіх алгоритмів на централізовані й розподілені*. У централізованих алгоритмах вибір усіх маршрутів здійснюється у центральному вузлі, а в розподілених – у вузлах мережі. При цьому вузли обмінюються інформацією, якщо це необхідно. Але зазначимо, що часто така класифікація належить до реалізації, а не до математичного опису алгоритму. Цілком можливо, що на деякому рівні математичної абстракції розподілений і централізований алгоритми стають еквівалентними.

Інша класифікація алгоритмів маршрутизації заснована на *залежності (чи незалежності) зміни маршрутів від інтенсивностей вхідних потоків*. У статичних алгоритмах маршрутизації шлях, що використовується кожною парою відправник – адресат, фіксований і не залежить від коливань трафіка. Він може мінятися тільки у випадку виходу з ладу будь-якого вузла або лінії.

Під час використання алгоритмів цього типу не може досягатися велика пропускна здатність за значної варіації вхідного трафіка. Такий спосіб маршрутизації рекомендується застосовувати або для дуже простих мереж, або тоді, коли ефективність роботи мережі несуттєва. Для більшості основних мереж з комутацією пакетів використовуються різновиди адаптивної маршрутизації, під час якої шляхи від відправників до адресатів для нового трафіка змінюються у відповідь на перевантаження. Річ у тім, що в результаті зміни статистики вхідного трафіка на окремих ділянках мережі можуть виникати перевантаження. У цьому випадку алгоритм маршрутизації повинен намагатися змінити маршрути й направити потоки в обхід місць скупчення пакетів.

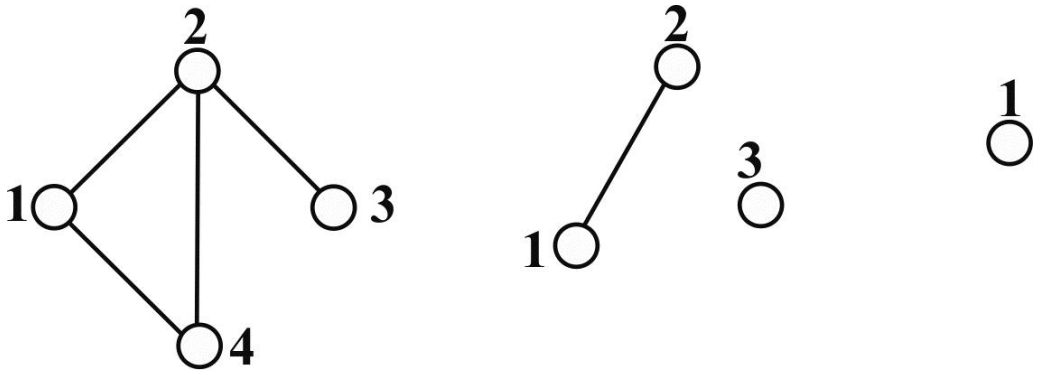
Методи маршрутизації використовують ряд простих графових алгоритмів. Розглянемо, наприклад, знаходження найкоротшого шляху, коли відомі довжини всіх ліній мережі, й потрібно знайти шлях, що з'єднує два даних вузли і має мінімальну сумарну довжину. Якщо довжина лінії деяким чином відбиває ступінь її навантаження, то пошук найкоротшого шляху еквівалентний пошуку найменш навантаженого шляху між двома вузлами, а це має відношення до завдання маршрутизації. Тут ми докладно розглянемо, як завдання вибору найкоротшого шляху і його роль у маршрутизації, так і інші завдання, що представляють самостійний інтерес. Почнемо ми з того, що введемо деякі основні графові позначення.

Неорієнтовані графи

Ми визначаємо граф $G=(N, A)$ як кінцеву непусту множину вузлів N і деякий набір A пар різних вузлів з N. Кожна пара вузлів у A називається дугою. Кілька прикладів графів показані на рис. 5.3. Вузли часто називаються вершинами, а дуги – ребрами, галузями, ланками або лініями. Головна мета формального визначення $G=(N, A)$ полягає в означенні того, що місце розташування вузлів і конкретна форма дуг при зображенні графа на папері зовсім несуттєві. Якщо n_1 і n_2 є вузлами графа, а (n_1, n_2) – дугою, то говорять, що ця дуга інцидентна n_1 і n_2. Деякі автори визначають графи так, що дугам дозволяється мати обидва кінці інцидентними одному й тому самому вузлу, але ми навмисне не допускаємо такі петлі. Ми також не допускаємо кратні дуги між однією й тією самою парою вузлів.

Переходом по графу G називається послідовність вузлів $(n_1, n_2, \ldots, n_l)$, таких, що усі пари (n_1, n_2), (n_2, n_3), …, (n_{l-1}, n_l) є дугами графа G. Перехід, у якому немає повторюваних вузлів, називається ***шляхом***. Перехід $(n_1, \ldots, n_l)$,

у якому $n_1 = n_l$, $l > 3$ і немає повторюваних вузлів, крім $n_1 = n_l$, називається ***циклом***. Ці визначення проілюстровані на рис. 5.4.



а) б) в)

$N = \{1, 2, 3, 4\}$ $N = \{1, 2, 3\}$ $N = \{1\}$

$A = \{(1, 2), (2, 3), (4,1), (2, 4)\}$ $A = (\{1, 2\})$ $A = \{\ \}$

Рисунок 5.3 – Три приклади графів із множиною вузлів N і множиною дуг A

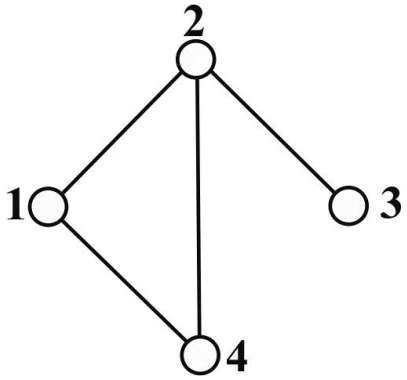


Рисунок 5.4 – Ілюстрація переходів, шляхів і циклів.
Послідовності (1, 4, 2, 3), (1, 4, 2, 1), (1, 4, 2, 1, 4, 1), (2, 3, 2) і (2) є переходами.
Кожна з послідовностей (1, 4, 2, 3) і (2) є шляхом, а (1, 4, 2, 1) – циклом.
Зазначимо, що (2) і (2, 3, 2) не розглядаються як цикли

Вважатимемо, що граф зв'язний, якщо для кожного вузла i існує шлях $(i = n_1, n_2, \ldots, n_l = j)$ до будь-якого іншого вузла j. Зазначимо, що на рис. 5.3 графи (а) і (в) зв'язані, а граф (б) незв'язаний. Ми виявляємо відсутність зв'язності графа, якщо бачимо, що дві підмножини вузлів не з'єднані між собою дугами. Це спостереження можна довести.

Деревом називається зв'язний граф, що не має циклів. ***Остовним деревом*** графа G називається підграф графа G, що є деревом і містить усі вузли графа G. Наприклад, на рис. 5.5 підграфи (б) і (в) є остовними деревами

графа (а). Далі показаний простий алгоритм, що будує остовне дерево довільного графа $G=(N,A)$:

1. Нехай n – довільний вузол з N. Покладемо $N'=\{n\}$, $A'=\{\ \}$.

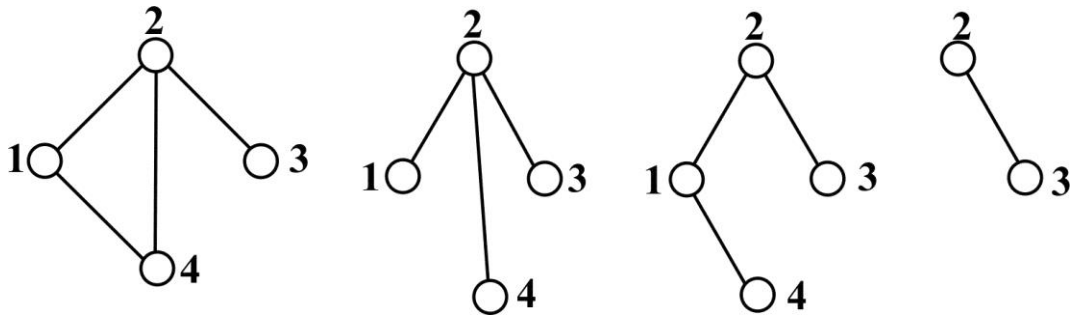
2. Якщо $N'=N$, то стоп ($G'=(N',A')$ – остовне дерево); у протилежному випадку перехід до кроку 3.

3. Нехай дуга $(i,j)\in A$, така, що $i\in N'$, $j\in N-N'$. Оновлюємо A' й N':

$N':=N'\cup\{j\}$

$A':=A'\cup\{(i,j)\}$.

Перехід до кроку 2.



(а) (б) (в) (г)

Рисунок 5.5 – Граф (а) і три підграфи (б), (в) і (г).
Підграфи (б) і (в) є остовними деревами

Розглянемо завдання, що являє інтерес для мереж передачі даних; це завдання приводить до розподіленого алгоритмічного розв'язання. Нехай заданий зв'язний граф $G=(N,A)$ з N вузлами та A дугами й вагами w_{ij} для кожної дуги $(i,j)\in A$. Завдання полягає в побудові остовного дерева з мінімальною сумою ваг дуг. Таке дерево називається остовним деревом мінімальної ваги (МОД для стислості). МОД може виявитися корисним під час поширення повідомлень, адресованих усім вузлам мережі, наприклад, керуючих повідомлень від вузла-супервізора. У такому контексті вага дуги означає вартість передачі повідомлення по цій дузі в будь-якому напрямку, а вага всього остовного дерева – вартість передачі повідомлення всім вузлам остовного дерева.

Алгоритм Прим–Дійкстра починає з одного довільного вузла, обраного як фрагмент, і потім збільшує цей фрагмент шляхом послідовного додавання дуг мінімальної ваги, які полишають граф. **Алгоритм Крускаля** починає з усіх вузлів, які розглядаються як одновузлові фрагменти, водночас, і потім послідовно з'єднує будь-які два фрагменти дугою, що має мінімальну вагу

серед дуг, додавання яких не створює циклу (рис. 5.6). Обидва алгоритми закінчують роботу після $N-1$ ітерації.

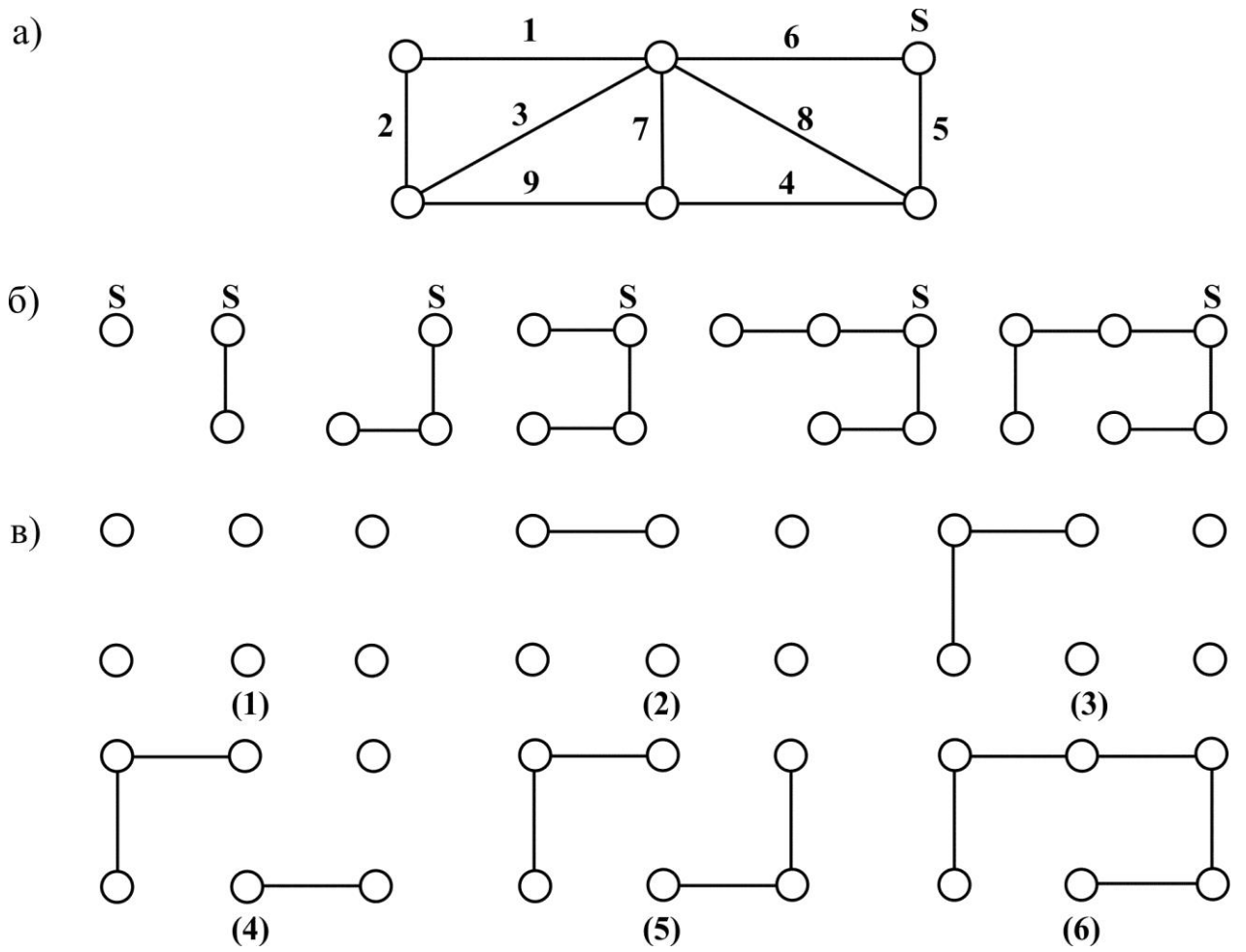


Рисунок 5.6 – Побудова остовного дерева мінімальної ваги:
а – граф із зазначеними вагами дуг; б – послідовні ітерації алгоритму Прим–Дійкстра; в – послідовні ітерації алгоритму Крускаля

Алгоритм Крускаля водночас створює кілька фрагментів, які в підсумку поєднуються в МОД, однак на кожному кроці до наявного набору фрагментів додається тільки одна дуга. Є можливість одночасного додавання до кожного фрагмента дуг мінімальної ваги, що полишають граф, методом розподіленого алгоритму. Це можна робити, якщо МОД єдине.

Розподілений алгоритм побудови МОД у графі з єдиним МОД працює так. Він починає з деякого набору фрагментів (ними можуть бути, наприклад, вузли). Кожний фрагмент відшукує дугу мінімальної ваги, що йде від нього, додає її до себе та інформує про це фрагмент, що лежить на іншому кінці цієї дуги. Можна довести, що якщо дуга, яка з'єднує на черговому кроці два фрагменти, дійсно є дугою, що полишає граф, і має мінімальну вагу для одного із фрагментів, то після кожного кроку алгоритму результатом буде набір дійсно

фрагментів МОД і цикли ніколи не виникнуть. Більш того, процес приєднання нових дуг триватиме доти, доки не залишиться жодної дуги, що полишає граф, і буде лише один фрагмент, за необхідністю МОД. Тому алгоритм не може зупинитися, не знайшовши МОД. Загалом, для того, щоб алгоритм працював правильно, не обов'язково, щоб процедура приєднання дуг була синхронною для всіх фрагментів. По суті, потрібна лише схема, відповідно до якої вузли й дуги фрагмента визначають дугу мінімальної ваги, що полишає граф. Це можна зробити різними способами, але бажано створити таку схему, яка б виконувала це з мінімальними ресурсами зв'язку.

5.2 Мережні алгоритми знаходження найкоротшого шляху. Алгоритм Беллмана–Форда

Припустимо, що вузол 1 є вузлом-джерелом і потрібно знайти довжини найкоротших шляхів від вузла 1 до кожного вузла графа. Для цього алгоритму дугові відстані можуть бути як позитивними так і негативними, але ми припустимо, що немає циклів негативної довжини (це припущення обговорюватиметься пізніше докладніше). Для спрощення позначень покладемо $\alpha_{ij} = \infty$, якщо в графі відсутня дуга (i, j). Основна ідея алгоритму Беллмана–Форда в тому, щоб спочатку знайти довжини найкоротших шляхів, за умови, що шляхи містять не більше однієї дуги, потім довжини найкоротших шляхів за умови, що шляхи містять не більше двох дуг і т.д. Найкоротший шлях за умови, що шлях містить не більше h дуг, надалі називатиметься найкоротшим $(\le h)$ шляхом.

Нехай $D_i^{(h)}$ – довжина найкоротшого $(\le h)$ шляху від вузла 1 до вузла i. Вважатимемо, що $D_i^{(h)} = 0$ для всіх h. Суть алгоритм Беллмана–Форда така.

Спочатку

$$D_i^{(0)} = \infty \quad \text{для всіх} \quad i \neq 1. \tag{5.1}$$

При кожному наступному $h \ge 0$

$$D_i^{(h+1)} = \min_j \left[D_j^{(h)} + d_{ji} \right] \quad \text{для всіх} \quad i \neq 1 \tag{5.2}$$

Рисунок 5.7 ілюструє роботу алгоритму. Для доведення того, що цей алгоритм призводить до правильного розв'язку, спочатку зазначимо, що (5.1) і (5.2) дають $D_i^{(1)} = d_{1i}$ для всіх $i \neq 1$ і вони дійсно є довжинами найкоротших (≤ 1) шляхів. Далі доведемо індукцією по h, припускаючи для даного h, що $D_i^{(h)}$ є довжинами найкоротших $(\le h)$ шляхів для всіх $i \neq 1$, і доводячи,

що рівність (5.2) дає довжину найкоротшого $(\le h+1)$ шляху від вузла 1 до кожного вузла $i \neq 1$. Спочатку покажемо, що ліва частина (5.2) більше або дорівнює правій частині, а потім доведемо протилежну нерівність. Припустимо, що $(1,\ldots,m,k,i)$ – найкоротший $(\le h+1)$ шлях від 1 до i. Тоді його довжина дорівнює довжині шляху $(1,\ldots,m,k)$ плюс d_{ki}. Оскільки $(1,\ldots,m,k)$ містять не більше h дуг, то

$$D_i^{(h+1)} \ge D_k^{(h)} + d_{ki} \ge \min_j \left[D_j^{(h)} + d_{ji} \right].$$

Для доведення зворотної нерівності припустимо, що мінімум у правій частині (5.2) досягається при $j=k$ і що $(1,\ldots,m,k)$ є найкоротшим $(\le h)$ шляхом, довжина якого, за припущенням, дорівнює $D_k^{(h)}$. Тоді довжина шляху $(1,\ldots,m,k,i)$ дорівнює правій частині (5.2). Оскільки $(1,\ldots,m,k,i)$ – деякий шлях, то

$$D_i^{(h+1)} \le D_k^{(h)} + d_{ki} = \min_j \left[D_j^{(h)} + d_{ji} \right].$$

І, нарешті, якщо $(1,\ldots,m,k,i)$ не є шляхом (тобто якщо він містить цикл), то i має лежати на шляху $(1,\ldots,m,k)$, а оскільки всі цикли мають не негативну довжину за припущенням, то, отже, попередня нерівність зберігається, що й доводить остаточно рівність (5.2).

Шлях може містити не більше $N-1$ дуг (де N – кількість вузлів) і тому, якщо немає циклів негативної довжини, то $D_i^{(N-1)}$ буде довжиною найкоротшого шляху від 1 до i. Також легко зазначити, що якщо $D_i^{(h+1)} = D_i^{(h)}$ для всіх i і певного h, то наступні ітерації з великими h не змінять довжин найкоротших шляхів і $D_i^{(h)}$ буде довжиною найкоротшого шляху для кожного i.

Кількість ітерацій алгоритма в найгіршому випадку дорівнює $N-1$, кожна ітерація має бути проведена для $N-1$ вузла, а для кожного вузла мінімізації здійснюється якнайбільше по $N-1$ змінній. Отже, у найгіршому випадку обсяг обчислень зростає як N^3, що записується у вигляді $O\left(N^3\right)$. Позначення $O\left(p(N)\right)$, де $p(N)$ – многочлен від N, використовується для залежного від N числа, що менше $cp(N)$ для всіх N, де c – деяка константа, яка не залежить від N. Дійсно, ретельніший підрахунок показує, що обсяг обчислень дорівнює $O(mA)$, де A – кількість дуг, а m – кількість ітерацій

(m також дорівнює максимальній кількості дуг, що утримуються в найкоротшому шляху).

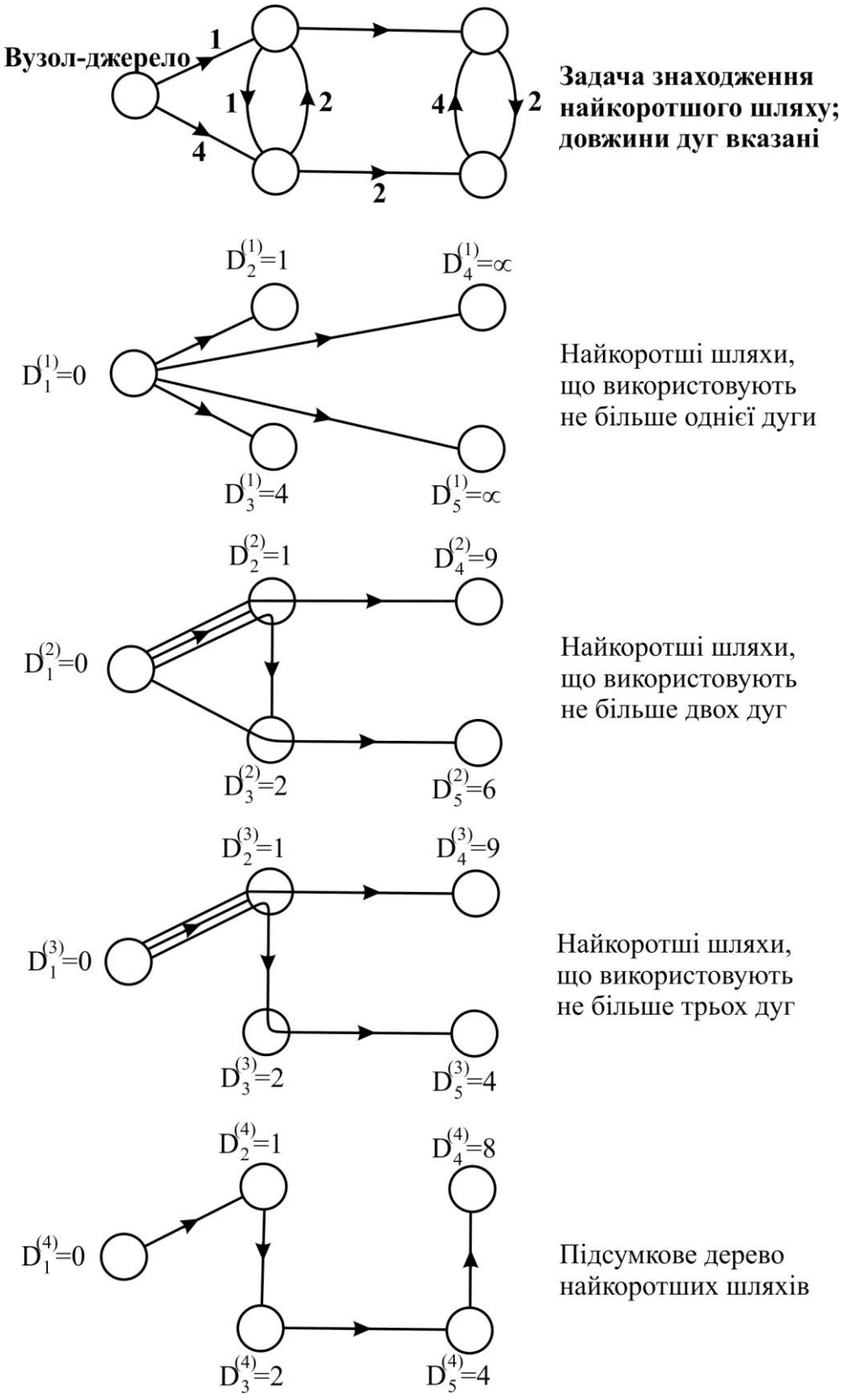


Рисунок 5.7 – Послідовні ітерації в методі Беллмана – Форда

5.3 Контрольні запитання та завдання

1. Дайте класифікацію алгоритмів маршрутизації.
2. На які основні характеристики якості обслуговування впливає алгоритм маршрутизації?
3. Дайте визначення остовного дерева мінімальної ваги.
4. У чому сутність алгоритму Прим–Дійкстра?
5. Поясніть сутність алгоритму Крускаля.
6. Поясніть сутність алгоритму Беллмана–Форда.

Лекція 6
МАРШРУТИЗАЦІЯ
ПРИ ДЕГРАДАЦІЇ ТА ВІДНОВЛЕННІ МЕРЕЖІ

6.1 Проблема коливання трафіка під час адаптивної маршрутизації

Алгоритм Дійкстра

Цей алгоритм вимагає, щоб довжини всіх дуг були позитивні (на щастя, це виконується в мережах передачі даних). Обсяг обчислень у найгіршому разі для цього алгоритму значно менший, ніж в алгоритмі Беллмана–Форда. Основна ідея алгоритма полягає у пошуку найкоротших шляхів у порядку зростання довжини шляху. Найкоротшим шляхом серед усіх найкоротших від вузла 1 є шлях, що складається з однієї дуги, яка з'єднує вузол 1 з найближчим сусіднім вузлом, оскільки будь-який шлях, що складається з декількох дуг, буде завжди довшим за довжину першої дуги внаслідок припущення про позитивність усіх дугових довжин. Наступним найкоротшим має бути або шлях з однієї дуги до наступного найближчого сусіда вузла 1, або найкоротший шлях із двох дуг, що проходить через вузол, обраний на першому кроці, і т.д. Щоб формально описати цю процедуру у вигляді алгоритму, вважатимемо, що кожен вузол i має позначку D_i, яка означає оцінку довжини найкоротшого шляху від вузла 1. Коли оцінка стає незмінною, вважатимемо, що вузол остаточно позначений, і множину остаточно позначених вузлів зазначимо як P. Вузол, який буде додано на черговому кроці до P, є найближчим до вузла 1 серед усіх вузлів, які ще не увійшли в P.

Рисунок 6.1 ілюструє головну ідею. На k-му кроці є множина P із k найближчих вузлів до вузла 1, D_i – найкоротша відстань від вузла 1 до кожного вузла i з P. Серед усіх шляхів, що з'єднують вузол 1 з будь-яким вузлом не з P, найкоротший шлях має пройти по вузлах з P (оскільки $d_{ij} > 0$).

Тому $(k+1)$-й найближчий вузол і відповідна найкоротша відстань є мінімізацією по $j \notin P$ величини $\min_{i \notin P} \{D_i + d_{ij}\}$. Ці обчислення можна провести ефективно, у результаті чого їхня складність буде порядку $O(N^2)$.

Формально алгоритм працює так.

Спочатку $P = \{1\}$, $D_1 = 0$ й $D_j = d_{1j}$ для $j \neq 1$.

Крок 1 (пошук наступного найближчого вузла). Знайти $i \notin P$, такий, що

$$D_i = \min_{j \notin P} D_j \ .$$

Покласти $P := P \cup \{i\}$. Якщо P містить усі вузли, то на цьому робота алгоритму закінчується.

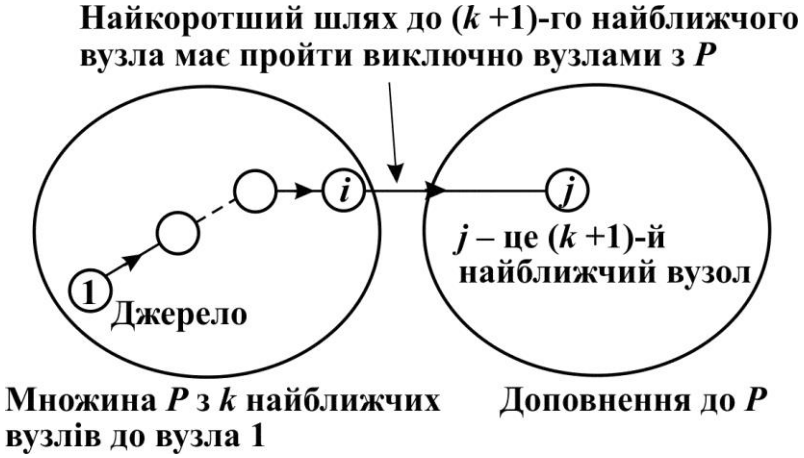


Рисунок 6.1 – Основна ідея алгоритму Дійкстра

Крок 2 (відновлення позначок.) Для всіх $j \notin P$ покласти

$$D_{j:} = \min\left[D_j, D_i + d_{ij} \right].$$

Перейти до кроку 1.

На рис. 6.2 подано приклади виконання основних кроків для обчислення маршрутів з використанням алгоритмів Беллмана–Форда і Дійкстра.

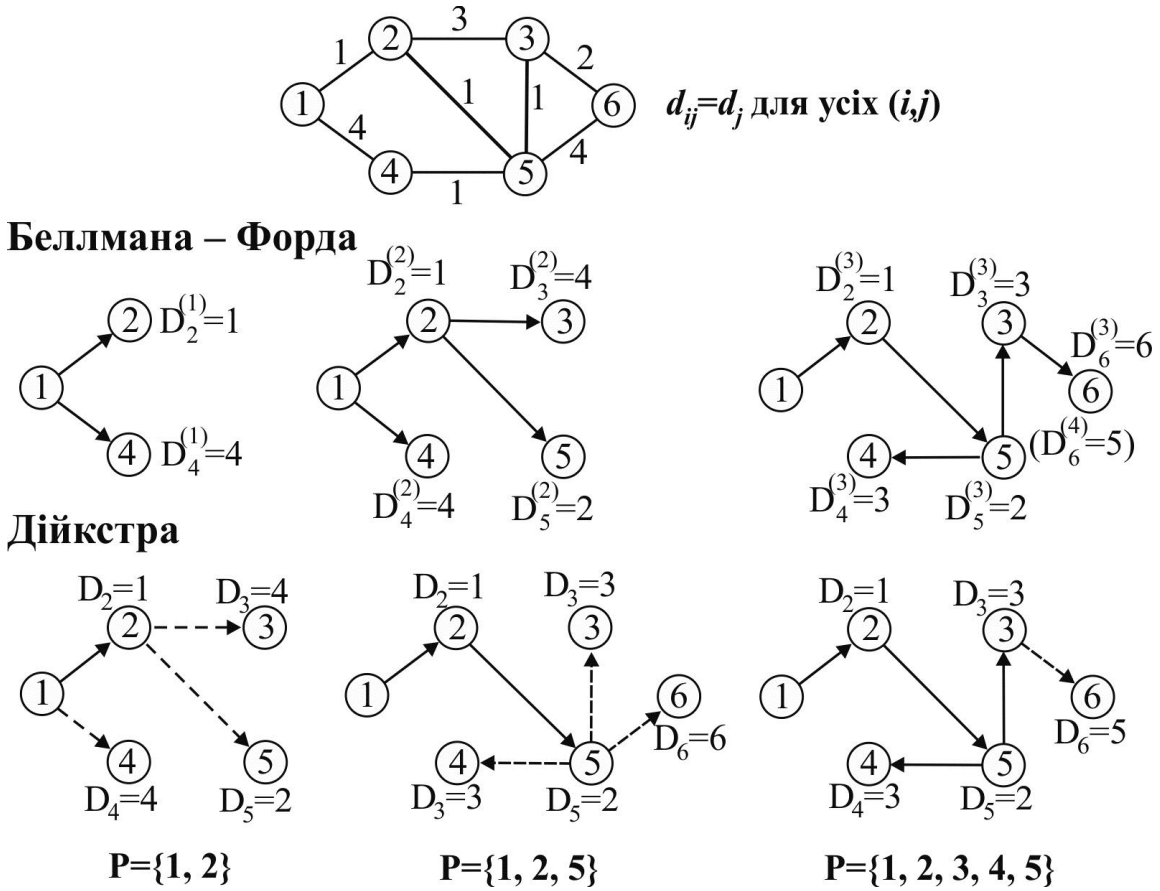


Рисунок 6.2 – Приклад використання алгоритмів Беллмана–Форда і Дійкстра

Питання стійкості в дейтаграмних мережах

Для того щоб навести простий приклад коливань, розглянемо дейтаграмну мережу та припустимо, що є два шляхи, по яких вузол-відправник може направити трафік вузлу-отримувачу. Вибір даного шляху під час маршрутизації призведе до збільшення довжини цього шляху й у наступний період часу інший шлях стане кращим в ході вибору маршруту, у результаті чого виникнуть коливання.

Виявляється, що тип коливань може бути набагато гіршим, як показує наведений приклад.

Приклад

Розглянемо мережу з 16 вузлами, показану на рис. 6.3, у якій вузол 16 є єдиним вузлом-адресатом.

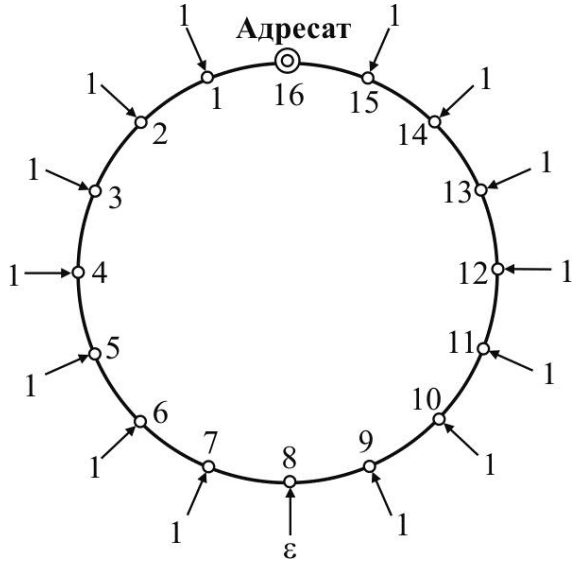


Рисунок 6.3 – Кільцева мережа із шістнадцятьома вузлами з прикладу 1. Вузол 16 є єдиним адресатом

Нехай інтенсивність вхідного трафіка (одиниці даних/сек) у кожному з вузлів $i = 1, \ldots, 7, 9, \ldots, 15$ дорівнює одиниці, а інтенсивність вхідного трафіка у вузлі 8 дорівнює $\varepsilon > 0$, де ε дуже мале. Припустимо, що довжина лінії (i, j) дорівнює

$$d_{ij} = F_{ij},$$

де F_{ij} – інтенсивність трафіка, що надходить в лінію, включно з вхідним і транзитним трафіками. Припустимо, що всі вузли обчислюють свої найкоротші шляхи до вузла-адресата кожні T секунд, використовуючи як довжини ліній інтенсивності F_{ij}, обмірювані протягом попередніх T секунд, а потім направляють

свій трафік найкоротшим шляхом протягом наступних T секунд. Нехай спочатку вузли 1–7 обрали маршрути за годинниковою стрілкою, а вузли 8–15 – проти годинникової стрілки. Це досить гарна маршрутизація, що збалансовано розподіляє вхідний трафік за двома напрямками. На рис. 6.4 показано інтенсивності трафіків, які надходять у лінію для початкової та подальших маршрутизацій, заснованих на пошуку найкоротших шляхів. Після трьох відновлень найкоротших шляхів алгоритм призводить до стійкого коливального режиму, у якому весь трафік по черзі направляється то за годинниковою стрілці, то проти неї; зміна напрямку відбувається під час кожного чергового відновлення. Зрозуміло, що така маршрутизація – найгірша серед усіх можливих.

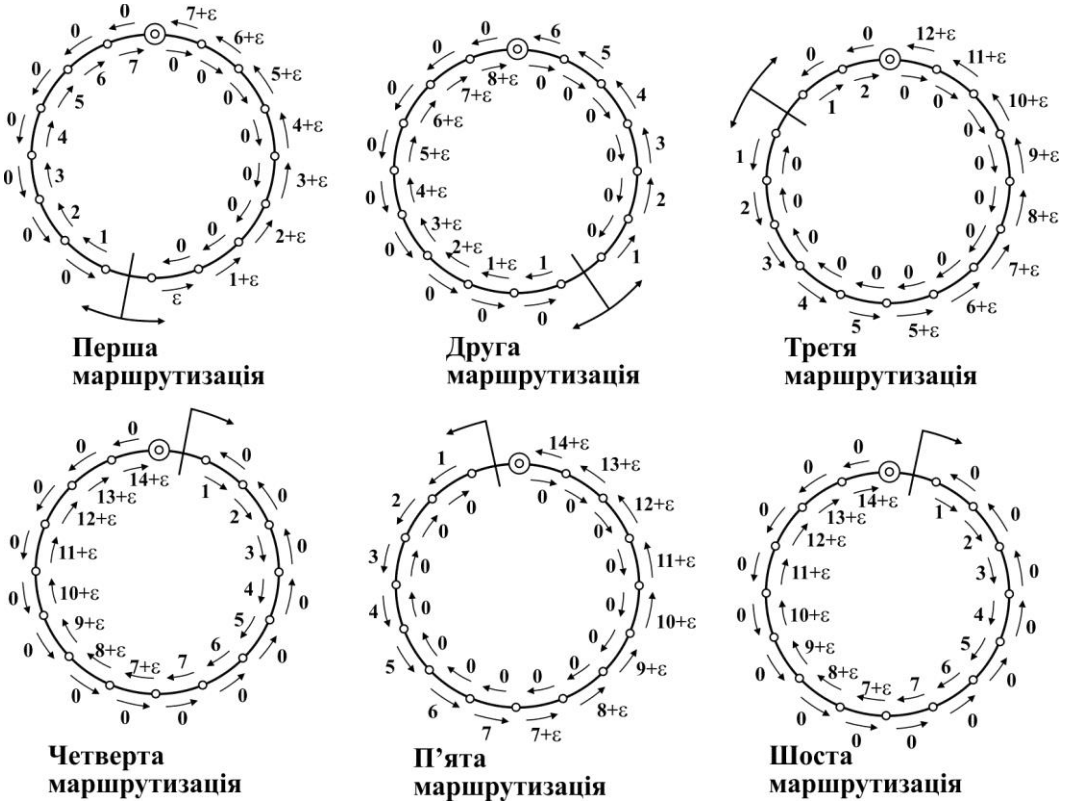


Рисунок 6.4 – Коливання у кільцевій мережі у разі, коли довжини ліній d_{ij} дорівнюють інтенсивностям трафіка F_{ij}, що надходить в лінії. Кожен вузол посилає одну одиницю вхідного трафіка вузлу-адресату, за винятком середнього вузла 8, який посилає $\varepsilon > 0$, де ε дуже мале. Числа поруч з лініями означають інтенсивність потоку, який проходить по лінії в кожному з двох напрямків. Як приклад обчислень найкоротших шляхів розглянемо першу ітерацію. Середній вузол 8 обчислює довжину шляху за годинниковою стрілкою $28(=0+1+2+\ldots+7)$ і довжину шляху

проти годинникової стрілки $28+8\varepsilon\left(=\varepsilon+1+\varepsilon+2+\varepsilon+\ldots+7+\varepsilon\right)$ і направляє свій трафік на найкоротший (за годинниковою стрілкою) шлях під час другої маршрутизації. Відповідними числами для вузла 9 будуть 28 і $\left(28+7\varepsilon\right)$, і тому вузол 9 також направляє свій трафік за годинниковою стрілкою. Всі інші вузли виявляють, що шлях, використаний ними при першій маршрутизації, є найкоротшим і тому вони не будуть направляти свій трафік на інший шлях

Причиною поганої маршрутизації в попередньому прикладі є той факт, що інтенсивності навантажень, які надходять у лінії, залежать від обраної маршрутизації, що у свою чергу залежить від інтенсивностей, які проходять по лініях трафіків, у результаті чого виникає ефект зворотного зв'язку. Це схоже на систему зі зворотним зв'язком з теорії керування, й тому тут можна використовувати відповідні методи.

Можна показати, що розглянутий вище тип нестійкості з'являється в основному тоді, коли довжина d_{ij} лінії (i, j) зростає безупинно й монотонно зі зростанням навантаження F_{ij} й $d_{ij}=0$, яке проходить по лінії, якщо $F_{ij}=0$. Ці коливання можна погасити шляхом додавання позитивної константи до довжини лінії так, щоб $d_{ij}=\alpha>0$, коли $F_{ij}=0$. Скалярна величина α (довжина лінії при нульовому навантаженні) називається коефіцієнтом зсуву. Коливання в первинному алгоритмі ARPANET гасилися за допомогою досить великого коефіцієнта зсуву. Зазначимо, що якщо α – досить велике число, то відповідні найкоротші шляхи будуть шляхами з найменшою кількістю ліній до вузла-адресата. У результаті маршрутизація стане статичною, такою. яка не може мати жодних коливань зазначеного вище типу, але вона також буде повністю нечутлива до перевантажень, що виникають.

6.2 Маршрутизація при деградації та відновленні мережі. Лавинний алгоритм оновлення топології мережі

Завдання, з яким часто доводиться зіштовхуватися під час маршрутизації, складається з передачі керуючої інформації від місць у мережі, де вона збирається, туди, де вона необхідна. Це завдання ускладнюється, якщо лінії можуть виходити з ладу.

Надійне отримання маршрутної інформації в місцях, де вона необхідна, в умовах, коли лінії можуть виходити з ладу, пов'язане з рядом тонкощів, які ще до кінця не оцінені.

Ось деякі з них.

1. Інформація про топологічні зміни має передаватися по лініях, які можуть виходити з ладу. У деяких мережах необхідно вміти підтримувати їхнє функціонування, навіть коли деякі частини мережі стають незв'язаними одна з одною.

2. Доводиться мати справу з багаторазовими топологічними змінами (такими як зміна статусу лінії з діючої на таку, що вийшла з ладу, й назад протягом короткого часу), і тому виникає проблема розрізнення старої й нової інформації про зміни. Як приклад розглянемо метод поширення серед усіх вузлів інформації про топологічні зміни, які називаються лавинним алгоритмом. Суть цього методу полягає в тому, що кожен вузол стежить за статусом усіх ліній, що йдуть від нього, і після виявлення зміни статусу будь-якої лінії посилає пакет усім своїм сусідам, повідомляючи їм про цю зміну. Сусіди пересилають цей пакет своїм сусідам і т.д. Необхідно здійснити деякі заходи, для того щоб цей оновлюючий пакет не циркулював мережею нескінченно довго; це питання буде розглянуто пізніше. Лавинний алгоритм працює, коли є тільки одна топологічна зміна, і може виявитися непридатним, коли можливі багаторазові зміни, як показано на рис. 6.5.

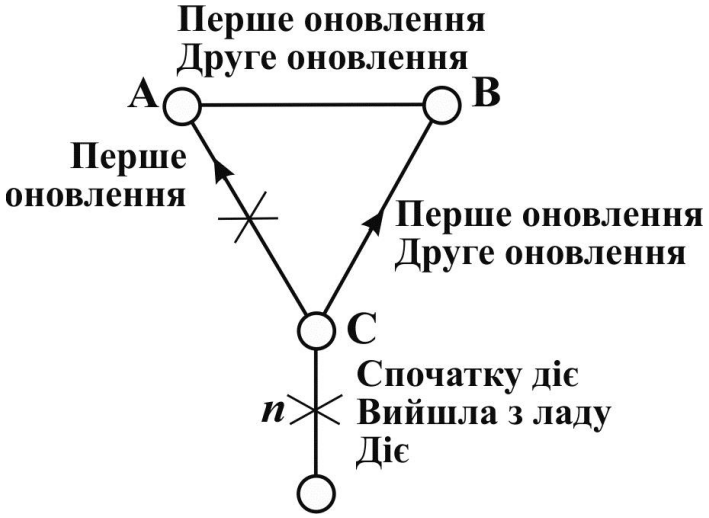


Рисунок 6.5 – Приклад, коли лавинний алгоритм не працює

На рисунку – спочатку лінія n діюча, потім виходить з ладу. Припустимо, що два оновлювальні повідомлення проходять шляхом CBA швидше, ніж перше оновлююче повідомлення проходить по лінії CA. Припустимо також, що лінія CA виходить з ладу після першого оновлюючого повідомлення, але до того, як по ньому пройде друге оновлююче повідомлення. Тоді останнє отримане A повідомлення містить інформацію, що лінія n вийшла з ладу,

у той час як насправді лінія діє. Проблемою є те, що застаріла інформація помилково вважається новою.

Способи подолання цих труднощів розглядатимуться в даному розділі; тут буде показано, що іноді прості очевидні рішення можуть містити замасковані помилки.

3. Оновлююча інформація про топологічні зміни використовується деяким алгоритмом, який обчислює нові маршрутні шляхи (наприклад, алгоритмом пошуку найкоротшого шляху). Цілком можливо, що оновлююча інформація про топологічні зміни надійде під час роботи алгоритму. Тоді алгоритм має в ході своєї роботи враховувати ці зміни, або зупинитися й заново почати діяти після отримання нових даних. Виконання того чи іншого може виявитися нетривіальним, особливо, якщо алгоритм є розподіленим.

4. У результаті відновлення будь-якої лінії дві незв’язні частини мережі можуть знову виявитися зв’язними. Кожна із частин, можливо, має застарілу інформацію про топологію іншої частини. Алгоритм має гарантувати, що в остаточному підсумку обидві частини узгоджуються, і їм стане відома правильна топологія мережі.

6.3 Задача побудови топології мережі

У цьому підрозділі ми обговоримо алгоритми побудови топології мережі передачі даних. Суть задачі у тому, що при заданих потоках потрібно створити таку мережу, що змогла б їх обслуговувати за мінімальну вартість і при дотриманні деяких вимог до характеристик. Зазвичай загальна постановка задачі полягає в такому.

Нехай задано:

1) Географічне місцезнаходження множини пристроїв, які потрібно з’єднати один з одним. Для простоти ці пристрої називатимемо терміналами.

2) Матриця трафіка, що вказує інтенсивності вхідних трафіків від кожного термінала до кожного іншого термінала.

Потрібно побудувати (рис. 6.6):

1. Топологію підмережі зв’язку для обслуговування трафіка, що виникає у терміналів. Це включає вибір місця розташування вузлів, ліній і пропускної здатності кожної лінії.

2. Локальну мережу доступу, тобто набір ліній зв’язку, що з’єднують термінали із вхідними вузлами підмережі.

Під час побудови мережі слід прагнути до того, щоб:

1) середня затримка пакета або повідомлення була меншою деякого заданого рівня (при заданому номінальному трафіку й обраному типі алгоритму маршрутизації);

2) мережа відповідала деяким вимогам з надійності, гарантуючи цілісність мережного обслуговування, незважаючи на ряд виходів з ладу ліній і вузлів;

3) мінімізувати деяку функцію від капітальних та експлуатаційних витрат під час виконання пунктів 1 і 2.

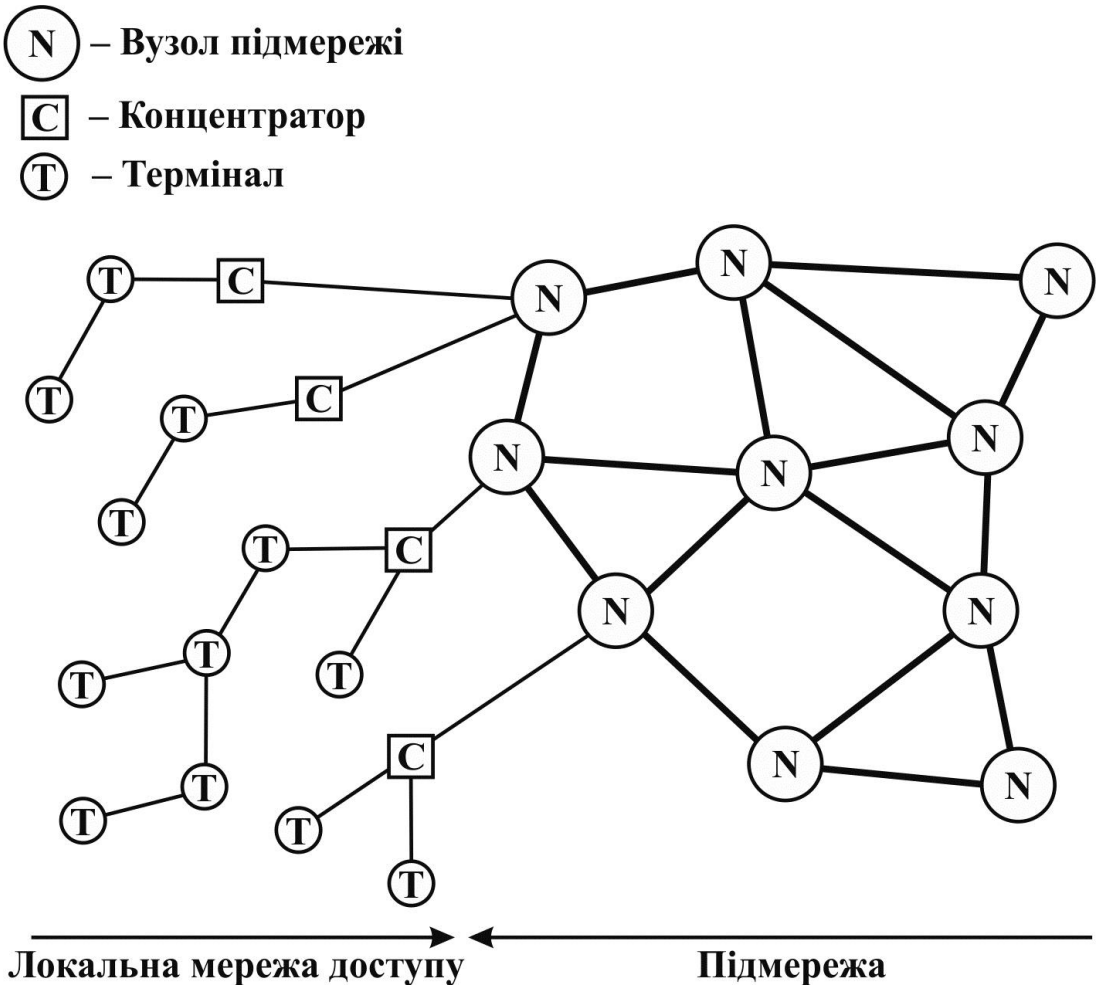


Рисунок 6.6 – Основними питаннями задачі побудови топології є побудова локальної мережі доступу й підмережі зв’язку

Загалом така постановка задачі є занадто широкою, її важко сформулювати точно, не говорячи вже про її розв’язання. Але у багатьох випадках ситуація дещо спрощується. Наприклад, можливо, що частина цього завдання вже розв’язана, оскільки деякі ділянки локальної мережі доступу й (або) підмережі зв’язку вже створені.

Крім того, може цілком природним виглядати розбиття цієї задачі на окремі частини, наприклад, можливо, що задача створення підмережі зв’язку вирішуватиметься окремо від завдання створення локальної мережі доступу. Насправді це припускатиметься далі. Зрештою, не часто доводиться мати справу з чітко поставленими завданнями, які можна розв’язати точно й у розумний термін. Зазвичай доводиться задовольнятися наближеним розв’язком, отриманим евристичним методом, що поєднує теорію, метод спроб і помилок та здоровий глузд.

6.4 Контрольні запитання та завдання

1. Поясніть проблему коливання трафіка в ході адаптивної маршрутизації.
2. Як можна зменшити коливання трафіка в дейтаграмній маршрутизації?
3. Поясніть проблеми маршрутизації в ході деградації та відновлення мережі.
4. Сформулюйте завдання побудови топології мережі.

Лекція 7
ЗАВДАННЯ ПОБУДОВИ ТОПОЛОГІЇ МЕРЕЖІ

7.1 Розв'язання задачі вибору пропускної здатності ліній мережі зв'язку

Для того щоб оцінити характеристики алгоритму маршрутизації, необхідно кількісно визначити поняття перевантаження в мережі. У цьому підрозділі формулюються деякі моделі (які далі називатимуться потоковими моделями), засновані на інтенсивностях трафіка, що надходить у лінії мережі. Ці моделі використовуватимуться в ході постановки завдання оптимальної маршрутизації. Ми також використовуватимемо потокові моделі під час розгляду побудови різних частин мережної топології.

Перевантаження в мережах передачі даних кількісно можна визначити з використанням статистик вхідних процесів, що описують черги в мережі. Ці статистики визначають розподіл довжини черги й час очікування пакета в кожній лінії. Очевидно, що при гарній маршрутизації мають бути малі середні значення й дисперсії затримки пакета в кожній черзі. На жаль, зазвичай буває важко уявити кількісний параметр у вигляді єдиного функціонала, що підходить для оптимізації. Головною причиною цього є те, що, як правило, немає явного аналітичного виразу для середніх значень або дисперсій довжин черг у мережах передачі даних.

Зручною, хоча й не зовсім досконалою альтернативою є вимірювання навантажень у лініях з використанням середнього трафіка, який проходить по лінії. Точніше, ми припускаємо, що статистика потока, що надходить у будь-яку лінію (i, j), змінюється лише через відновлення маршрутів, і під навантаженням у лінії (i, j) ми розумітимемо інтенсивність трафіка F_{ij}, що надходить. Називатимемо F_{ij} потоком, що проходить по лінії (i, j), і вимірюватимемо його в одиницях даних у секунду, де одиницями даних можуть бути біти, пакети, повідомлення тощо. Іноді має сенс вимірювати потік в одиницях, які, як припускається, прямо пропорційні одиницям даних у секунду, наприклад, у кількості віртуальних ланцюгів, що проходять по даній лінії.

У потокових моделях робиться неявне припущення, що статистика трафіка, який надходить у мережу, не змінюється в часі. Таке припущення є розумним, коли ця статистика змінюється дуже повільно порівняно з середнім часом, необхідним для зменшення черг у мережі, і коли потоки в лініях

вимірюються шляхом часового усереднення. Типовим прикладом мережі, що задовольняє такі умови, є мережа, у якій кожна пара відправник – адресат призначена для великої кількості користувачів, а інтенсивність трафіка у кожного з цих користувачів мала порівняно з сумарною інтенсивністю. Вираз виду

$$\sum_{(i,j)} D_{ij}\left(F_{ij}\right), \tag{7.1}$$

де кожна функція D_{ij} є такою, що монотонно зростає, часто обирають як вартісну функцію при оптимізації. Часто використовується така формула:

$$D_{ij}\left(F_{ij}\right) = \frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij}F_{ij}\ , \tag{7.2}$$

де C_{ij} – пропускна здатність лінії (i, j), що вимірюється в тих самих одиницях, що й F_{ij}, а d_{ij} – затримка через обробку й поширення. При цьому вираз (7.1) означає середню кількість пакетів у системі в припущенні, що кожна черга описується моделлю $M/M/1$; це відповідає клейнроківській апроксимації незалежністю та теоремі Джексона. Хоча це припущення зазвичай не виконується на практиці, вирази (7.1) і (7.2) дуже корисні, оскільки вони якісно відбивають факт виникнення перевантаження, коли потік F_{ij} наближається за своїм значенням до відповідної пропускної здатності лінії C_{ij}. Іншою вартісною функцією з аналогічними якісними властивостями є

$$\max_{(i,j)}\left\{F_{ij}/C_{ij}\right\}, \tag{7.3}$$

тобто максимум коефіцієнта використання лінії. Числові результати показують, що зазвичай немає великого розходження між випадками використання вартісних функцій (7.1)–(7.2) або (7.3) під час оптимізації маршрутизації. Це показує, що слід обирати ту вартісну функцію, яку легше оптимізувати. Надалі ми зосередимо увагу на вартісній функції вигляду (7.1).

7.2 Евристичні методи вибору пропускної здатності ліній мережі зв'язку

Зазвичай під час використання цих методів починають з якоїсь конкретної топології мережі, а потім послідовно підправляють цю топологію шляхом зміни пропускної здатності якої-небудь однієї або одночасно декількох ліній. Отже, поблизу існуючої топології відшукується інша топологія, що задовольняє заданим обмеженням і має меншу вартість. Зазначимо, що термін «топологія» означає в цьому випадку перелік пропускних здатностей усіх

можливих ліній мережі. Зокрема, якщо пропускна здатність лінії дорівнює нулю, це вказує на те, що фактично лінії немає. Зазвичай припускається таке:

1. Відомі вузли мережі й інтенсивності вхідних потоків для кожної пари вузлів.

2. Обрано модель маршрутизації, яка дозволяє знаходити потоки F_{ij} для усіх ліній (i, j) при заданих пропускних здатностях C_{ij}. Найчастіше потоки по лініях визначаються, виходячи з моделі оптимальної маршрутизації. Наприклад, F_{ij} можна визначити, мінімізуючи середню затримку пакета

$$D = \frac{1}{\gamma} \sum_{(i,j)} \left(\frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij} F_{ij} \right), \tag{7.4}$$

обчислюється на основі апроксимаційної моделі $M/M/1$ (7.2), де γ – інтенсивність сумарного вхідного потоку, а C_{ij} й d_{ij} – відповідно пропускна здатність і затримка через обробку й поширення сигналу на лінії (i, j).

3. Є обмеження на максимальну затримку. Зазвичай потрібно, щоб обрані пропускні здатності C_{ij} і потоки F_{ij} (обумовлені алгоритмом маршрутизації, згідно з п. 2) були такими, щоб затримка, яка обчислюється за формулою (7.4) (або за будь-якою іншою аналогічною формулою), була менше деякого, заздалегідь обраного, порога.

4. Існує вимога за надійністю. Наприклад, часто потрібно, щоб мережа була двозв'язаною, тобто щоб після виходу з ладу одного будь-якого вузла всі інші вузли залишалися зв'язковими. І взагалі може знадобитися, щоб після виходу з ладу $(k-1)$ вузла всі інші вузли залишалися зв'язковими, – таку мережу називають k-зв'язковою. Далі більш докладно будуть розглянуті питання, пов'язані з оцінкою надійності мережі.

5. Існує критерій вартості, відповідно до якого порівнюються різні топології. Мета полягає в пошуку топології, що задовольняла б вимогам з затримки й надійності відповідно до пп. 3 і 4, і мала б якнайменшу вартість відповідно до п. 5. Опишемо ітеративний евристичний метод розв'язання цього завдання. До початку чергової ітерації в розпорядженні є найкраща поточна топологія і топологія, що перевіряється. Перша топологія задовольняє вимогам по затримці і надійності і є найкращою серед усіх розглянутих досі топологій, а друга топологія – це та, яка розглядатиметься в даній ітерації. Припускається, що спочатку ці топології обиралися спеціально, наприклад, кількість ліній в мережі бралася досить великим для того, щоб вимоги по затримці і надійності свідомо задовольнялися.

Кроки ітерації ідуть у такій послідовності:

Крок 1 (обчислення потоків). Обчислити потоки F_{ij} для топології, що перевіряються, використовуючи для цього деякий алгоритм маршрутизації відповідно до п. 2.

Крок 2 (перевірка обмеження за затримкою). Обчислити середню затримку пакета D за наявною формулою (наприклад, по (7.4)) для топології, що перевіряється. Якщо

$$D \le T,$$

де T – заданий поріг, то перейти до кроку 3, у протилежному випадку перейти до кроку 5.

Крок 3 (перевірка вимоги за надійністю). Перевірити, чи задовольняє топологія, що перевіряється, вимоги з надійності. (Методи виконання цього кроку будуть незабаром обговорюватися.) Якщо ця вимога не задовольняється, то перейти до кроку 5, у протилежному випадку перейти до кроку 4.

Крок 4 (перевірка зниження вартості). Якщо вартість топології, що перевіряється, менше вартості кращої поточної топології, то замінити кращу поточну топологію на ту, що перевіряється.

Крок 5 (створення нової топології, що перевіряється). Використовуючи будь-який евристичний алгоритм, змінити одну або кілька пропускних здатностей кращої поточної топології так, щоб у результаті вийшла топологія, що перевіряється, що раніше не розглядалася. Перейти до кроку 1.

Зазначимо, що топологія, яка перевіряється, обирається як краща поточна на кроці 4 лише тоді, коли вона задовольняє вимоги із затримки й надійності на кроках 2 і 3, і коли вона зменшує вартість (крок 4). Алгоритм зупиняється, коли не вдається створити нову топологію, яка, перевіряється, або коли подальшого істотного покращення очікувати не доводиться. Природно, що немає ніякої гарантії того, що остаточне розв’язання виявиться оптимальним. Можна спробувати домогтися кращого результату, побудувавши алгоритм з іншою початковою топологією. Інша можливість полягає у модифікації крока 4 так, щоб топологія, яка перевіряється, іноді випадково обиралася як краща поточна навіть тоді, коли її вартість виявлялася більшою, ніж знайдена дотепер. Схеми такого типу, відомі як алгоритми відпускання, стали популярними останнім часом. Можна показати, що за досить широких припущень глобального оптимального розв’язку можна досягти, використовуючи для цього алгоритм відпускання. Однак час обчислень у даному випадку може виявитися дуже великим. Такий підхід ще не застосовувався в контексті створення топології мережі.

Існує ряд евристичних правил, запропонованих у літературі, для створення нових топологій, які перевіряються (крок 5). Одним з таких правил є просто зменшення пропускної здатності тієї лінії, що здається недовантаженою (малі значення F_{ij}/C_{ij} й F_{ji}/C_{ji}), або взагалі усунення цієї лінії. Іншим правилом є збільшення пропускної здатності якоїсь перевантаженої лінії у випадку, коли не виконується обмеження на затримку. Комбінацією цих правил є евристичний метод заміни галузей, в ході якого одна лінія віддаляється й замість неї додається інша лінія. Корисним під час вибору ліній, які слід видаляти й додавати, є так званий метод насиченого розрізу. Ідея методу полягає у пошуку такої розбивки (або розрізу) вузлів на дві підмножини N_1 й N_2, для яких лінії, що з'єднують N_1 і N_2, найбільш навантажені. Здається природним, що додавання будь-якої лінії, що з'єднує вузол N_1 з вузлом N_2, допоможе понизити рівень навантаження цього розрізу.

Метод працює так (рис. 7.1):

Число поруч із кожною лінією означає ступінь навантаження даної лінії. Одна за одною найбільш навантажені лінії тимчасово віддаляються доти, доки мережа не розпадеться на дві частини N_1 і N_2. Потім додається лінія, що з'єднує будь-який вузол з N_1 із будь-яким вузлом з N_2, будь-яка слабко навантажена лінія віддаляється.

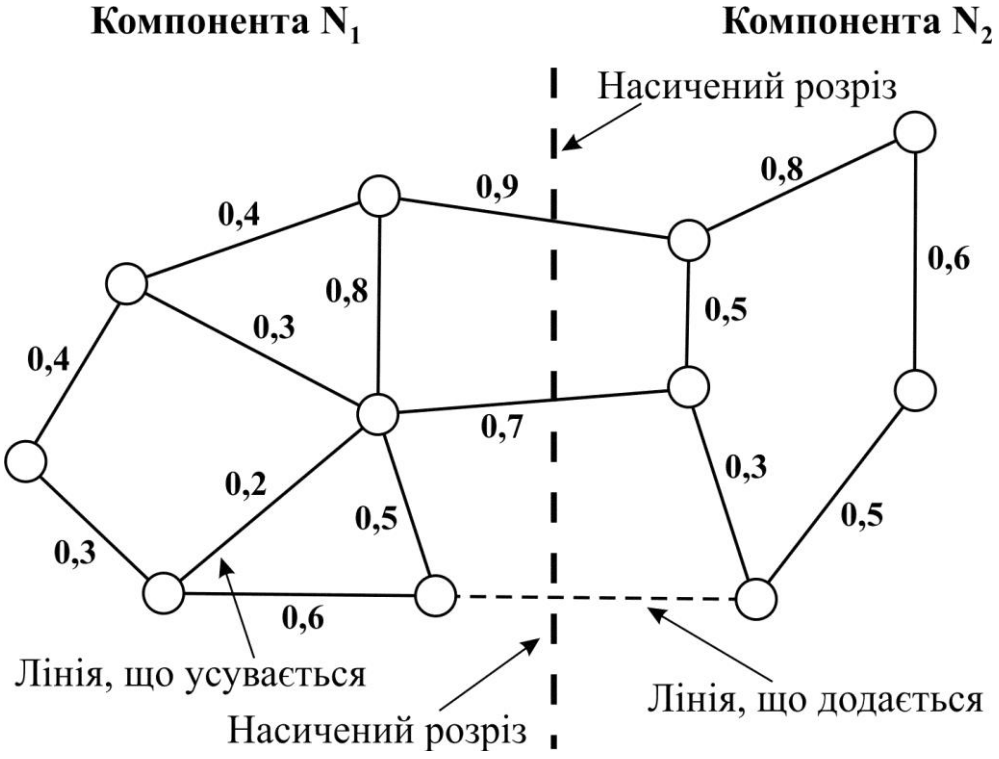


Рисунок 7.1 – Ілюстрація методу насиченого розрізу, який можна використовувати в ході евристичної заміни галузей

1. Підготувати список усіх ненаправлених ліній (i, j), розсортованих у порядку зменшення ступеня навантаження, кількісно виражається як $\max\{F_{ij}/C_{ij}, F_{ji}/C_{ji}\}$.

2. Знайти лінію k, таку, що: а) коли видалити всі лінії, розташовані у списку вище за k, то мережа залишиться зв'язновою і б) з видаленням k разом з усіма розташованими вище за неї лініями у списку, мережа розпадається на дві незв'язні частини N_1 і N_2.

3. Видалити з мережі найменш навантажену лінію й замість неї додати нову лінію, що з'єднує вузол із N_1 із вузлом з N_2.

7.3 Контрольні запитання та завдання

1. Наведіть функціонали для вимірювання навантажень у лініях з використанням середнього трафіка, що проходити по лінії.

2. У чому сутність евристичних методів вибору пропускної здатності ліній мережі зв'язку?

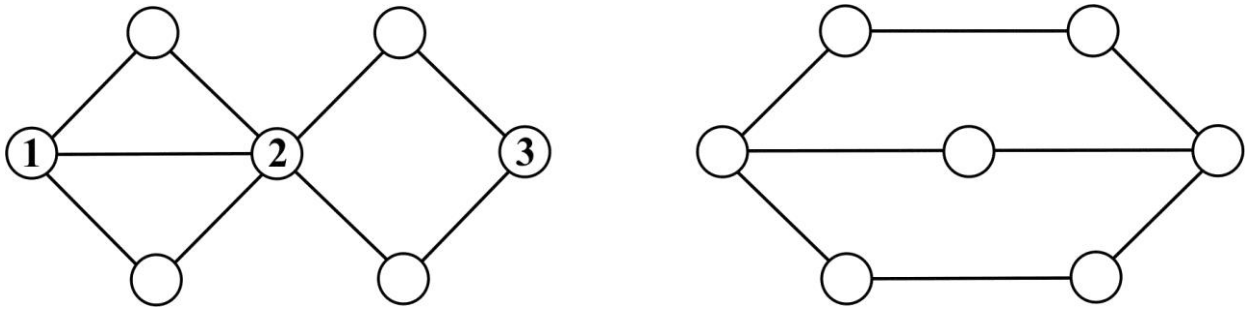
3. Поясніть сутність алгоритму насиченого розрізу для побудови топології мережі.

Лекція 8
МЕТОДИ ОПТИМАЛЬНОЇ МАРШРУТИЗАЦІЇ

8.1 Задача надійності мережі зв'язку, k-зв'язані мережі. Алгоритм Клейтмана

Найпоширенішою вимогою з надійності мережі є така, щоб вона була k-зв'язаною. Визначимо тепер k-зв'язаність формально.

Називатимемо два вузли i та j у неорієнтованому графі k-зв'язними, якщо є шлях, який з'єднує i та j у кожному підграфі, отриманому видаленням із графа $(k-1)$ вузлів, відмінних від i та j, разом з їхніми суміжними дугами. Легко помітити, що це еквівалентно вимозі, щоб i та j були з'єднані дугою, або існувало не менше k з'єднуючих i та j непересічних у вузлах шляхів, тобто таких, що будь-які два із цих шляхів не мали загальних вузлів, крім i та j. Називатимемо граф k-зв'язним, якщо кожна пара вузлів у графі є k-зв'язною. Ці визначення проілюстровані на рис. 8.1.



(а) (б)

Рисунок 8.1 – Ілюстрація визначення k-зв'язності.
У графі (а) вузли 1 і 2 є 6-зв'язними, вузли 2 і 3 – двозв'язні,
а вузли 1 і 3 – однозв'язні. Граф (а) є однозв'язним, а граф (б) – двозв'язним

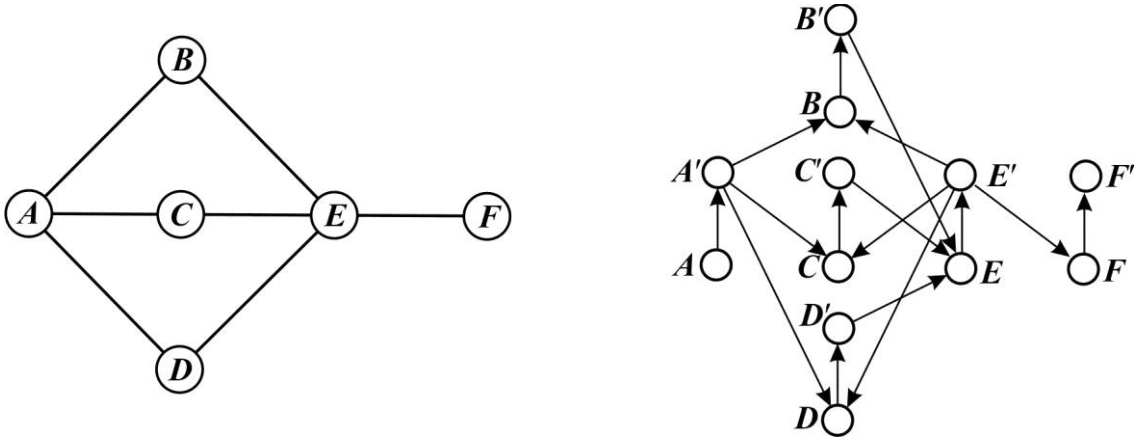
Становить інтерес також поняття дугової зв'язності, обумовлене як мінімальна кількість дуг, які треба видалити для того, щоб граф виявився незв'язним. Нижня границя k для дугової зв'язності може бути встановлена шляхом перевірки k-(вузлової) зв'язності для розширеного графа, у якому кожна дуга (i, j) замінена на дві дуги (i,n) й (n, j), де n – новий вузол. Це стає очевидним, якщо помітити, що вихід з ладу вузла n у розширеному графі відповідає виходу з ладу дуги (i, j) у початковому графі.

Можна обчислити у вузлах кількість шляхів, що не перетинаються, які з'єднують два вузли, шляхом розв'язання відповідної задачі про максимальний потік. Це класична комбінаторна задача детально досліджена в багатьох роботах. Тут вона не обговорюватиметься, оскільки ця тема дещо виходить за межі даної лекції. Хто вже знайомий з цією задачею, може побачити на рис. 8.2 її зв'язок із завданням k-зв'язності двох вузлів.

Для того, щоб знайти кількість шляхів, які не мають загальних вузлів, від вузла A до вузла F у графі (а) потрібно:

1) Замінити кожен вузол i двома вузлами i та i' й орієнтованою дугою (i,i').

2) Замінити кожну дугу (i, j) графа (а) двома орієнтованими дугами (i', j) та (j',i). 3) Вилучити всі дуги, що приходять до A і відходять від F', внаслідок чого отримаємо граф (б). 4) Приписати нескінченну пропускну здатність дугам (A,A') і (F,F') й одиничну пропускну здатність всім іншим дугам. Тоді максимальний потік, який можна направити від вузла A до вузла F в графі (б), дорівнює кількості шляхів, що не мають загальних вузлів, які з'єднують вузли A та F у графі (а)



(а) (б)

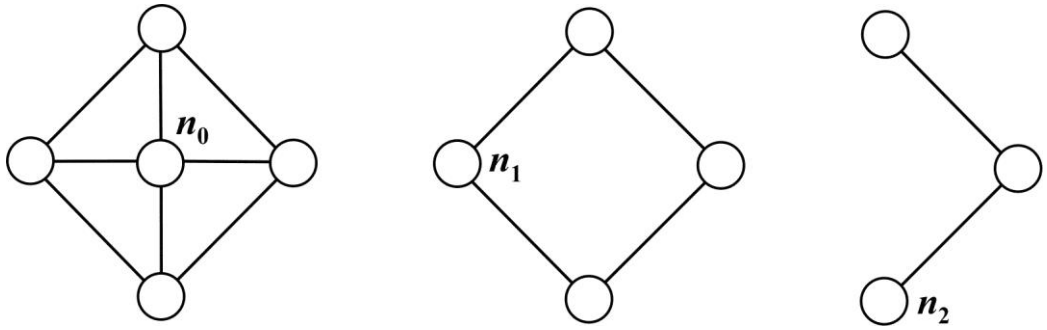
Рисунок 8.2 – Ілюстрація тесту k-зв'язності для тих, хто знайомий із завданням про максимальний потік

Одним зі способів перевірки k-зв'язності графа є перевірка k-зв'язності кожної пари вузлів. Але існують більш ефективні методи. Зокрема, метод Клейтмана, що полягає в такому.

Обирається довільний вузол n_0 і перевіряється k-зв'язність цього вузла з кожним іншим вузлом. Потім із графа видаляється вузол n_0 разом з усіма суміжними дугами, обирається будь-який інший вузол n_1 і перевіряється $(k-1)$-зв'язність цього вузла з кожним іншим вузлом. Так триває або до того

моменту, як буде перевірено, що вузол n_{k-1} однозв'язаний з кожним з вузлів, що залишилися, або виявиться, що деякий n_i вузол $i=0,1,\ldots,k-1,$ не є $(k-i)$-зв'язним з одним з вузлів, що залишилися. В останньому випадку очевидно, що граф не є k-зв'язним, оскільки, якщо вузол n_i не є $(k-i)$-зв'язним з деяким іншим з вузлів, що залишилися, то, можна знайти $(k-i-1)$ вузлів, таких, що їхнє вилучення разом з видаленням $n_0,\ldots,n_{i-1}$ вузлів робить граф незв'язним. Цей процес показаний на рис. 8.3:

Обирається довільний вузол n_0 і перевіряється 3-зв'язність між n_0 і кожним іншим вузлом. Потім видаляється вузол n_0 і суміжні з ним дуги, у результаті чого виходить граф (б). В (б) обирається довільний вузол n_1 і перевіряється двозв'язність між n_1 і кожним іншим вузлом у (б). Потім видаляється вузол n_1 і суміжні з ним дуги, у результаті чого виходить граф (в). Оскільки граф (в) зв'язний, то звідси випливає, що початковий граф (а) є тризв'язним.



(а) (б) (в)

Рисунок 8.3 – Перевірка 3-зв'язності графа (а) алгоритмом Клейтмана

Сумарна кількість перевірок пар вузлів на зв'язність в алгоритмі Клейтмана дорівнює $\sum_{i-1}^{k}(N-1)=kN-k(k+1)/2$, де N – кількість вузлів. Існує інший алгоритм, розроблений Івеном, що потребує близько N перевірок на зв'язність.

8.2 Побудова топології остовного дерева. Алгоритм Ессау–Вільямса

Для деяких мереж, для яких питання надійності менш істотні, може цілком підійти топологія остовного дерева. Використання такої топології узгоджується з ідеєю зосередження всієї пропускної здатності лише на невеликій кількості ліній для того, щоб зменшити середню затримку.

Можна побудувати топологію остовного дерева, приписавши ваги w_{ij} кожній потенційно можливій лінії (i, j) та використовуючи алгоритм побудови остовного дерева мінімальної ваги (МОД). Деякі складнощі можуть з'явитися, якщо існує обмеження на максимальний трафік, припустимий для передачі по одній лінії. У цьому випадку виникає задача побудови МОД з обмеженням, у якій є матриця вхідного трафіка від кожного вузла до кожного іншого вузла й потрібно побудувати МОД, у якому потік по кожній лінії не перевищує деяку, заздалегідь установлену межа. Внаслідок цього обмеження задача набуває комбінаторного характеру і для її розв'язання зазвичай застосовуються евристичні методи.

Для розв'язання окремого випадку задачі побудови МОД з обмеженням може застосовуватися алгоритм Ессау–Вільямса. У цьому випадку є центральний вузол 0 і N інших вузлів. Весь трафік має проходити через центральний вузол, і тому можна вважати, що весь вхідний трафік надходить від нецентральних вузлів до центрального й навпаки. Це завдання виникає також у контексті задачі побудови локальної мережі доступу; у цьому випадку роль центрального вузла відіграє концентратор трафіка.

Алгоритм Ессау–Вільямса є, по суті, методом евристичного обміну галузей. Він починається з остовного дерева, у якому центральний вузол безпосередньо пов'язаний з кожним з N інших вузлів (передбачається, що такий розв'язок є припустимим). У ході кожної чергової ітерації з наявного остовного дерева видаляється лінія $(i,0)$, що з'єднує деякий вузол i з центральним вузлом 0, а замість неї додається деяка лінія (i, j). Ці лінії обираються так, щоб:

1) не утворився цикл;

2) було задоволене обмеження на пропускні здатності для всіх ліній нового остовного дерева;

3) економія $w_{i0} - w_{ij}$ у вазі лінії, що виходить при заміні $(i,0)$ на (i, j), була б позитивною й максимальною серед усіх вузлів i та j, що задовольняють вимоги 1 і 2.

Алгоритм зупиняється, коли не вдається знайти такі вузли i та j, які б задовольняли вимогаи 1–3 при заміні $(i,0)$ на (i, j). Робота цього алгоритму проілюстрована на рис. 8.4.

Умова задачі подана в (а). Вузол 0 є центральним вузлом. Ваги ліній зазначені поруч з лініями. Вхідні потоки від вузлів 1, 2, 3 і 4 до центрального вузла зазначені поруч зі стрілками. Обмеження полягає у тому, щоб потік

по кожній лінії був не більше 8. Алгоритм закінчується після двох ітерацій, і в результаті отримуємо дерево (д), що має сумарну вагу 13. Закінчення відбувається внаслідок того, що, коли лінія $(1,0)$ або $(4,0)$ видаляється й додається будь-яка лінія, не суміжна з вузлом 0, порушується обмеження на пропускну здатність будь-якої лінії (потік по лінії ≤ 8). Оптимальне дерево, показане в (б), має сумарну вагу 12.

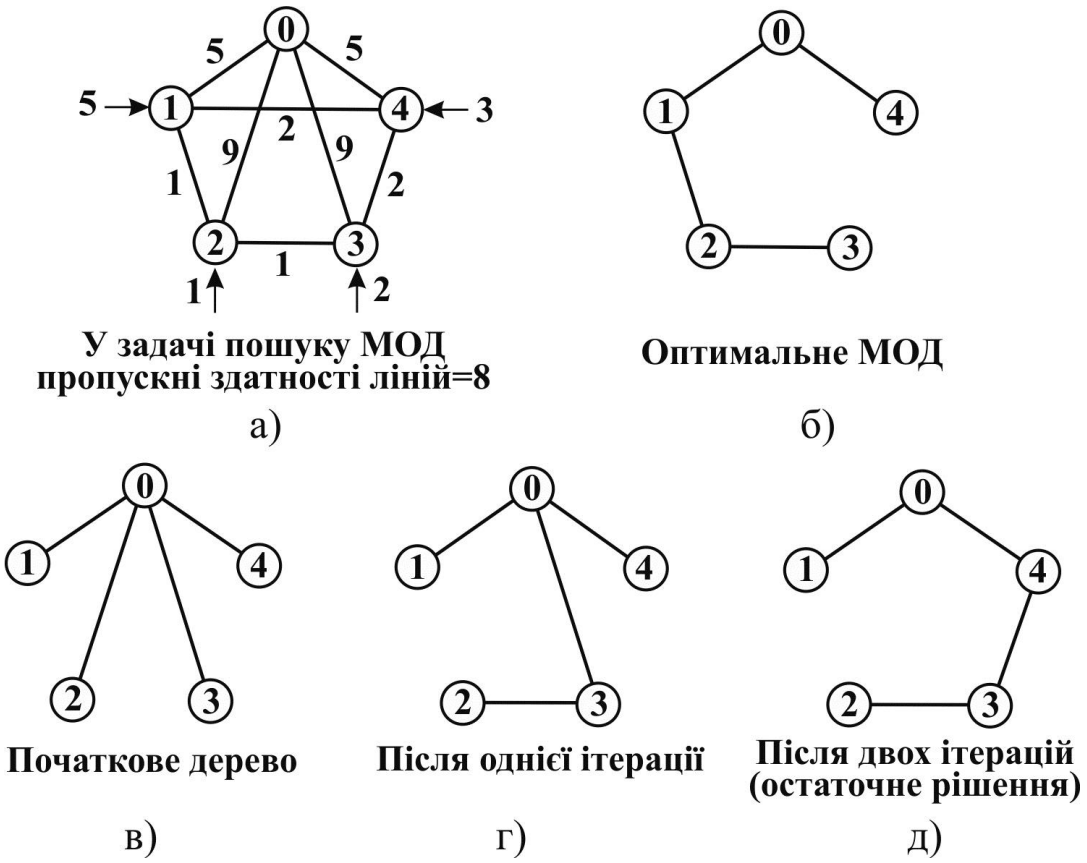


Рисунок 8.4 – Ілюстрація евристичного алгоритму Ессау–Вільямса для побудови МОД з обмеженням

8.3 Задача оптимальної маршрутизації

Згадаємо вид вартісної функції

$$\sum_{(i,j)} D_{ij}\left(F_{ij}\right). \tag{8.1}$$

Тут F_{ij} – сумарний потік (в одиницях даних у секунду), що проходить по лінії (i,j):

$$F_{ij} = \sum_{\substack{\text{за всіма шляхами } p,\\ \text{що містять } (i,j)}} x_p \ , \tag{8.2}$$

де x_p – потік (в одиницях даних у секунду), що проходить по шляху p.

Для кожної OA-пари w існують обмеження

$$\sum_{p \in P_w} x_p = r_w \ , \tag{8.3}$$

$$x_p \geq 0 \quad \text{для всіх} \quad p \in P_w, \tag{8.4}$$

де r_w – відома інтенсивність вхідного трафіка OA-пари w (в одиницях даних у секунду), а P_w – множина орієнтованих шляхів для w. Завдання, що формулюється з використанням невідомого вектора шляхових потоків $x = \left\{ x_p \middle| p \in P_w, w \in W \right\}$, записується у вигляді

$$\begin{aligned} &\text{мінімізувати} \quad \sum_{(i,j)} D_{ij} \left[\sum_{\substack{\text{за всіма шляхами } p, \\ \text{що містять } (i,j)}} x_p \right], \\ &\text{за обмежень} \quad \sum_{p \in P_w} x_p = r_w \quad \text{для усіх} \quad w \in W, \\ &\qquad\qquad x_p \geq 0 \quad \text{для усіх} \quad p \in P_w, w \in W. \end{aligned} \tag{8.5}$$

Далі описуватимемо оптимальну маршрутизацію з використанням похідних функцій D_{ij}. Передбачається, що кожна D_{ij} є двічі диференційована функція від F_{ij} і визначена в напівінтервалі $\left[0, C_{ij} \right)$, де значення C_{ij} дорівнює або деякій позитивній константі (у типових випадках – це пропускна здатність лінії), або нескінченності. Передбачається, що перша й друга похідні D_{ij}, які позначаються відповідно через D'_{ij} і D''_{ij}, суворо позитивні для всіх F_{ij} з $\left[0, C_{ij} \right)$. Звідси, зокрема, випливає, що D_{ij} є опуклою функцією від F_{ij}, яка монотонно зростає на всьому напівінтервалі $\left[0, C_{ij} \right)$ (рис. 8.5). Крім того передбачається, що $D_{ij}\left(F_{ij} \right) \to \infty$ при $F_{ij} \to C_{ij}$. Оскільки потоки в лініях є лінійними функціями шляхових потоків, то можна безпосередньо перевірити, що задача (8.5), що розглядається як задача зі змінними $\left\{ x_p \right\}$, які належать до шляхових потоків, має опуклу двічі диференційовану вартісну функцію й опуклу обмежувальну множину.

Опис розв’язків задачі оптимальної маршрутизації буде отримано за такої умови оптимальності, що є необхідною й достатньою.

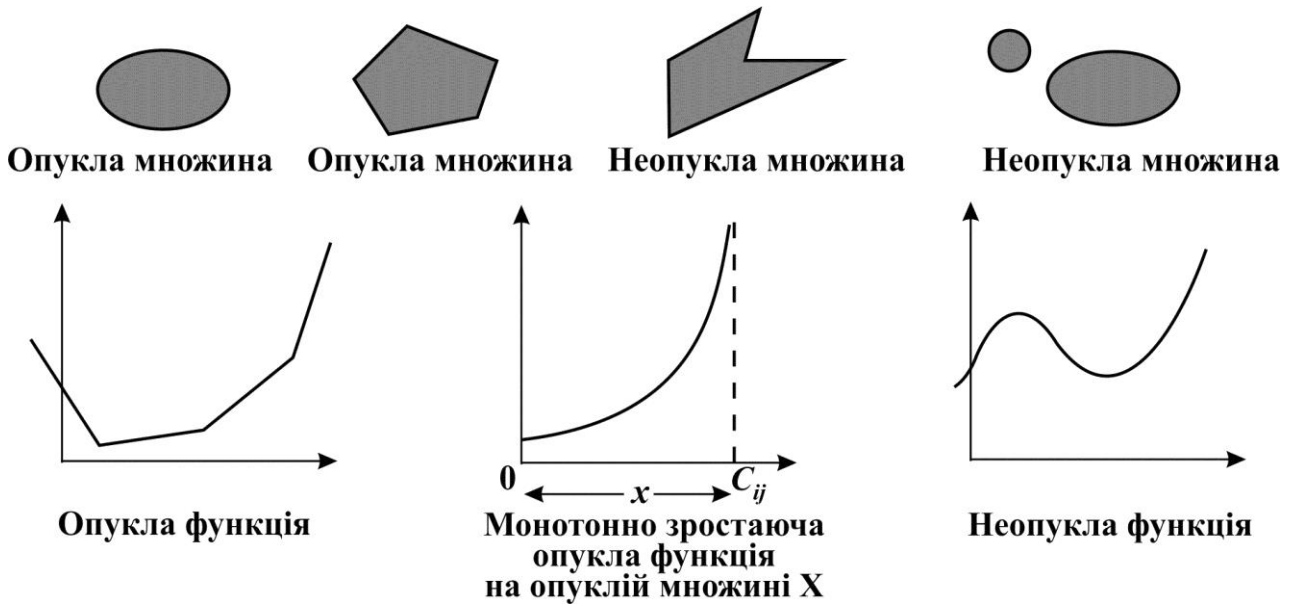


Рисунок 8.5 – Опуклі та неопуклі множини й функції

Лема. Нехай f – деяка диференційована опукла функція n-мірного вектора $x=(x_1,\ldots,x_n)$ й X – деяка опукла множина векторів.

Тоді $x^*\in X$ буде оптимальним розв'язком задачі

$$\text{мінімізувати } f(x)$$
$$\text{за обмеженя } x\in X \tag{8.6}$$

тоді й тільки тоді, коли

$$\sum_{i=1}^{n}\frac{\partial f(x^*)}{\partial x_i}\left(x_i-x_i^*\right)\geq 0 \quad \text{для усіх} \quad x\in X, \tag{8.7}$$

де $\partial f(x^*)/\partial x_i$ – значення першої похідної f по i-й координаті x_i в точці x^*.

Оскільки і вартісна функція й усі обмежувальні множини задачі мінімізації (8.5) є опуклими, то можна застосувати лему. Нехай x буде вектором шляхових потоків x_p. Для конкретності припустимо, що шляхи p послідовно перенумеровані. Тоді, якщо $W=\{w_1,w_2,\ldots,w_m\}$ – множина OA-пар, то відповідними множинами шляхів є

$$P_{w_1}=\{1,2,\ldots,n_1\},$$
$$P_{w_2}=\{(n_1+1),\ldots,n_2\},$$
$$\ldots$$
$$P_{w_m}=\{(n_{m-1}+1),\ldots,n_m\},$$

де $n_1, n,\ldots,n_m$ – деякі цілі числа, такі, що $n_1<n_2<\ldots<n_m$.

Координатами вектора шляхових потоків x будуть шляхові потоки $x_1, x_2, \ldots, x_{n_m}$, тобто

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_m} \end{bmatrix}.$$

Позначимо через $D(x)$ вартісну функцію задачі (8.5)

$$D(x) = \sum_{(i,j)} D_{ij} \left[\sum_{\substack{\text{за всіма шляхами } p, \\ \text{що містять } (i,j)}} x_p \right],$$

а через $\partial D(x)/\partial x_p$ – частинну похідну D по x_p. Тоді

$$\frac{\partial D(x)}{\partial x_p} = \sum_{\substack{\text{за всіма шляхами } p, \\ \text{що містять } (i,j)}} D'_{ij}, \qquad (8.8)$$

де перші похідні D'_{ij} беруться при сумарних потоках, що відповідають x.

З рівності (8.8) видно, що $\dfrac{\partial D(x)}{\partial x_p}$ є довжиною шляху p, якщо довжину кожної лінії (i, j) покласти такою, що дорівнює першій похідній D'_{ij}, узятої в x. Тому далі $\dfrac{\partial D(x)}{\partial x_p}$ називається першопохідною довжиною шляху p.

Відповідно до леми $x^* = \{x_p^*\}$ є вектором оптимальних шляхових потоків, якщо він задовольняє обмеження задачі маршрутизації й умову (8.7). Дану умову можна переписати у вигляді

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\partial D(x^*)}{\partial x_p} \left(x_p - x_p^* \right) \geq 0 \qquad (8.9)$$

для всіх x_p, задовольняючим обмеженням

$$\sum_{p \in P_w} x_p = r_w, \quad x_p \geq 0 \quad \text{для усіх} \quad p \in P_w, \quad w \in W. \qquad (8.10)$$

Умови (8.9) і (8.10) можна об’єднати стосовно OA-пари і для усіх $w \in W$ записати у вигляді

$$\sum_{p \in P_w} \frac{\partial D(x^*)}{\partial x_p} \left(x_p - x_p^* \right) \geq 0 \quad \text{для всіх} \quad x_p \geq 0, \qquad (8.11)$$

і $p \in P_w$, таких, що $\sum_{p \in P_w} x_p = r_w$.

(Щоб перевірити це, потрібно фіксувати будь-яку OA-пару w й покласти $x_p = x_p^*$ в (8.9) для усіх $w' \neq w$ і $p \in P_{w'}$.) Умова (8.11) еквівалентна вимозі, щоб для всіх $w \in W$ $x_p^* > 0$ тільки тоді, коли

$$\frac{\partial D\left(x^*\right)}{\partial x_{p'}} \geq \frac{\partial D\left(x^*\right)}{\partial x_p} \qquad \text{для всіх} \qquad p' \in P_w . \tag{8.12}$$

Щоб це побачити, розглянемо довільний шлях p, для якого $x_p^* > 0$, і деякий інший шлях p', і, взявши досить мале $\varepsilon > 0$, у нерівності (8.11) покладемо $x_p = x_p^* - \varepsilon$, $x_{p'} = x_{p'}^* + \varepsilon$ і $x_{p''} = x_{p''}^*$ для всіх інших $p'' \neq p, p'$. Сенс умови (8.12) в тому, що набір шляхових потоків оптимальний тоді й тільки тоді, коли шляховий потік позитивний лише для тих шляхів, які мають мінімальну першопохідну довжину. З умови (8.12) також випливає, що в точці оптимуму шляхи, по яких проходить ненульова частина вхідного потока r_w OA-пари w, повинні мати однакову довжину (меншу або рівну довжину, ніж всі інші шляхи для w).

8.4 Розв'язання задач оптимальної маршрутизації

Розглянемо мережу, що складається із двох ліній (рис. 8.6), у якій вузол 1 – це єдиний відправник, а вузол 2 – єдиний отримувач. Відомий вхідний потік r потрібно розділити на два шляхових потоки r_1 й r_2 так, щоб мінімізувати вартісну функцію, засновану на апроксимації моделлю $M/M/1$

$$D(x) = D_1(x_1) + D_2(x_2),$$

де $D_i(x_i) = x_i/(C_i - x_i)$ для $i = 1, 2$ й C_i – пропускна здатність лінії i. Для того, щоб це завдання мало сенс, ми повинні припустити, що r менше, ніж максимальна швидкість передачі $C_1 + C_2$, яку може мати мережа.

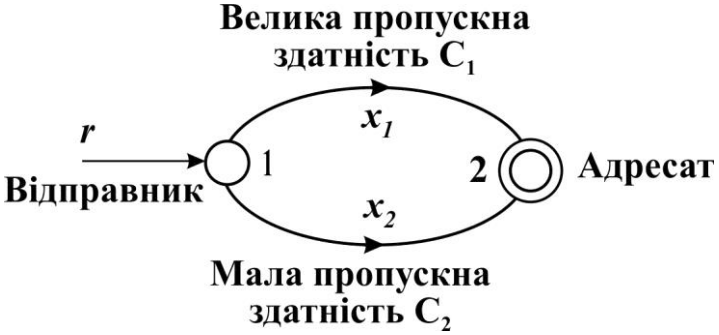


Рисунок 8.6 – Приклад задачі маршрутизації,
що включає одну OA-пару та два шляхи

У точці оптимуму має виконуватися умова найкоротшого шляху (8.12) так само, як і такі обмеження:

$$x_1^* + x_2^* = r, \quad x_1^* \geq 0, \quad x_2^* \geq 0.$$

Припустимо, що $C_1 \geq C_2$. Тоді з міркувань здорового глузду оптимальний потік x_1 не може бути менше ніж x_2 (не має сенсу посилати більше трафіка по лінії з меншою пропускною здатністю). Єдиними можливостями є такі.

1. $x_1^* = r$ і $x_2^* = 0$. Відповідно до (8.12) це може відбутися тільки тоді, коли

$$\frac{dD_1(r)}{dx_1} \leq \frac{dD_2(0)}{dx_2}.$$

Звідси (оскільки похідна від $\frac{x}{C-x}$ дорівнює $\frac{C}{(C-x)^2}$)

$$\frac{C_1}{(C_1 - r)^2} \leq \frac{1}{C_2}$$

або

$$r \leq C_1 - \sqrt{C_1 C_2}. \tag{8.13}$$

2. $x_1^* > 0$ і $x_2^* > 0$. У цьому випадку з (8.12) випливає, що довжини шляхів 1 і 2 однакові, тобто

$$\frac{dD_1(x_1^*)}{dx_1} \leq \frac{dD_2(x_2^*)}{dx_2}$$

або, що еквівалентно,

$$\frac{C_1}{(C_1 - x_1^*)^2} = \frac{C_2}{(C_2 - x_2^*)^2}. \tag{8.14}$$

З цієї рівності й обмеження $x_1^* + x_2^* = r$ можна знайти значення x_1^* і x_2^*. Безпосередні обчислення показують, що зразковим є

$$x_1^* = \frac{\sqrt{C_1}\left[r - \left(C_2 - \sqrt{C_1 C_2}\right)\right]}{\sqrt{C_1} + \sqrt{C_2}},$$

$$x_2^* = \frac{\sqrt{C_2}\left[r - \left(C_1 - \sqrt{C_1 C_2}\right)\right]}{\sqrt{C_1} + \sqrt{C_2}}.$$

Оптимальний розв'язок показаний на рис. 8.7 для r, що приймає значення в діапазоні $[0, C_1 + C_2)$ можливих значень. Можна помітити, що для досить малих r, коли має місце рівність (8.13), використовується лише найвисокошвидкісна лінія 1. Коли r перевищує граничне значення

$$C_1 - \sqrt{C_1 C_2},$$

при якому першопохідні довжини двох ліній стають рівними й нерівність (8.13) переходить у рівність, лінія 2 з меншою пропускною здатністю також використовується. У міру зростання r потоки на обох лініях збільшуються так, щоб першопохідні довжини залишалися рівними відповідно до (8.14). Така поведінка є типовою для оптимальної маршрутизації, якщо вартісна функція заснована на апроксимації моделлю $M/M/1$.

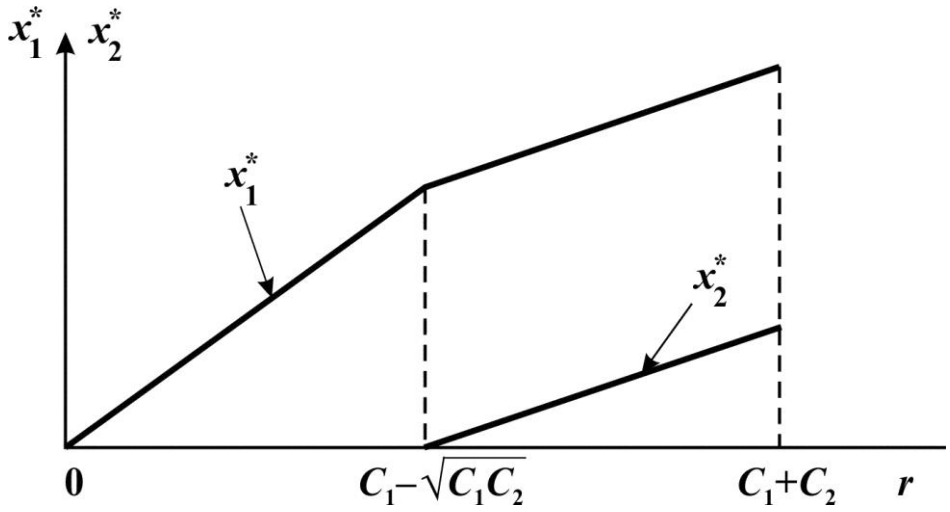


Рисунок 8.7 – Оптимальні шляхові потоки для розглянутого прикладу маршрутизації. При невеликому вхідному трафіку використовується лише одна лінія з більшою пропускною здатністю. Коли вхідний трафік перевищує поріг $C_1 - \sqrt{C_1 C_2}$, частина трафіка направляється по лінії з меншою пропускною здатністю

8.5 Контрольні запитання та завдання

1. Дайте визначення k-зв'язаної мережі.

2. У чому полягає суть методу Клейтмана оцінки зв'язності мережі. Дайте оцінку складності.

3. Поясніть сутність алгоритма Ессау–Вільямса для побудови топології мережі.

4. Наведіть опис задачі оптимальної маршрутизації.

5. Наведіть опис розв'язку задачі оптимальної маршрутизації.

Лекція 9
МЕТОДИ ОПТИМАЛЬНОЇ МАРШРУТИЗАЦІЇ ФРАНКА – ВОЛФА

9.1 Сутність методу Франка–Волфа

У цьому підрозділі подано один зі способів реалізації ідеї малих змін уздовж припустимих напрямків спуску.

Для даного вектора припустимих шляхових потоків $x = \{x_p\}$ знайти шлях з мінімальною першопохідною довжиною (МППД-шлях) для кожної OA-пари. (Значення перших похідних D'_{ij} беруться у точці x.) Нехай $\bar{x} = \{\bar{x}_p\}$ – вектор шляхових потоків, який утворюється, якщо вхідний потік r_w кожної OA-пари $w \in W$ спрямований по відповідному МППД-шляху.

Нехай α^* – такий розмір кроку, що мінімізує $D\left[x + \alpha(\bar{x} - x)\right]$ по всіх $\alpha \in [0,1]$, тобто

$$D\left[x + \alpha^*(\bar{x} - x)\right] = \min_{\alpha \in [0,1]} D\left[x + \alpha(\bar{x} - x)\right] \qquad (9.1)$$

Тоді значення шляхових потоків на такому кроці дорівнюють

$$x_p := x_p + \alpha^*(\bar{x} - x) \quad \text{для усіх} \quad p \in P_w, \ w \in W, \qquad (9.2)$$

і процес повторюється знову.

Описаний алгоритм є окремим випадком так званого методу Франка–Волфа розв'язання більш загальних задач нелінійного програмування з опуклими обмежувальними множинами і називається також методом девіації потоку. Можна показати, що в межі він зменшує значення вартісної функції до мінімуму, хоча в міру наближення до оптимуму швидкість збіжності стає дуже малою. Приблизний хід ітеративного методу спуску, заснованого на допустимих напрямках спуску, поданий на рис. 9.1. На кожній ітерації отримуємо допустиму точку з меншою вартістю.

Зауважимо, що припустимий напрям, який використовується в ітерації (9.2), має таке пояснення: частка α^* кожного потоку, що проходить ненайкоротшим шляхом (шлях, для якого $\bar{x}_p = 0$), перекидається на найкоротший шлях (для якого $\bar{x}_p = r_w$) для кожної OA-пари w. Характерною особливістю тут є те, що потік перекидається з ненайкоротших шляхів у рівних

пропорціях. Це відрізняє метод Франка–Волфа від градієнтних проекційних методів. Ці методи також перекидають потоки з ненайкоротших шляхів на найкоротші шляхи, але вони роблять це, як правило, у неоднакових пропорціях.

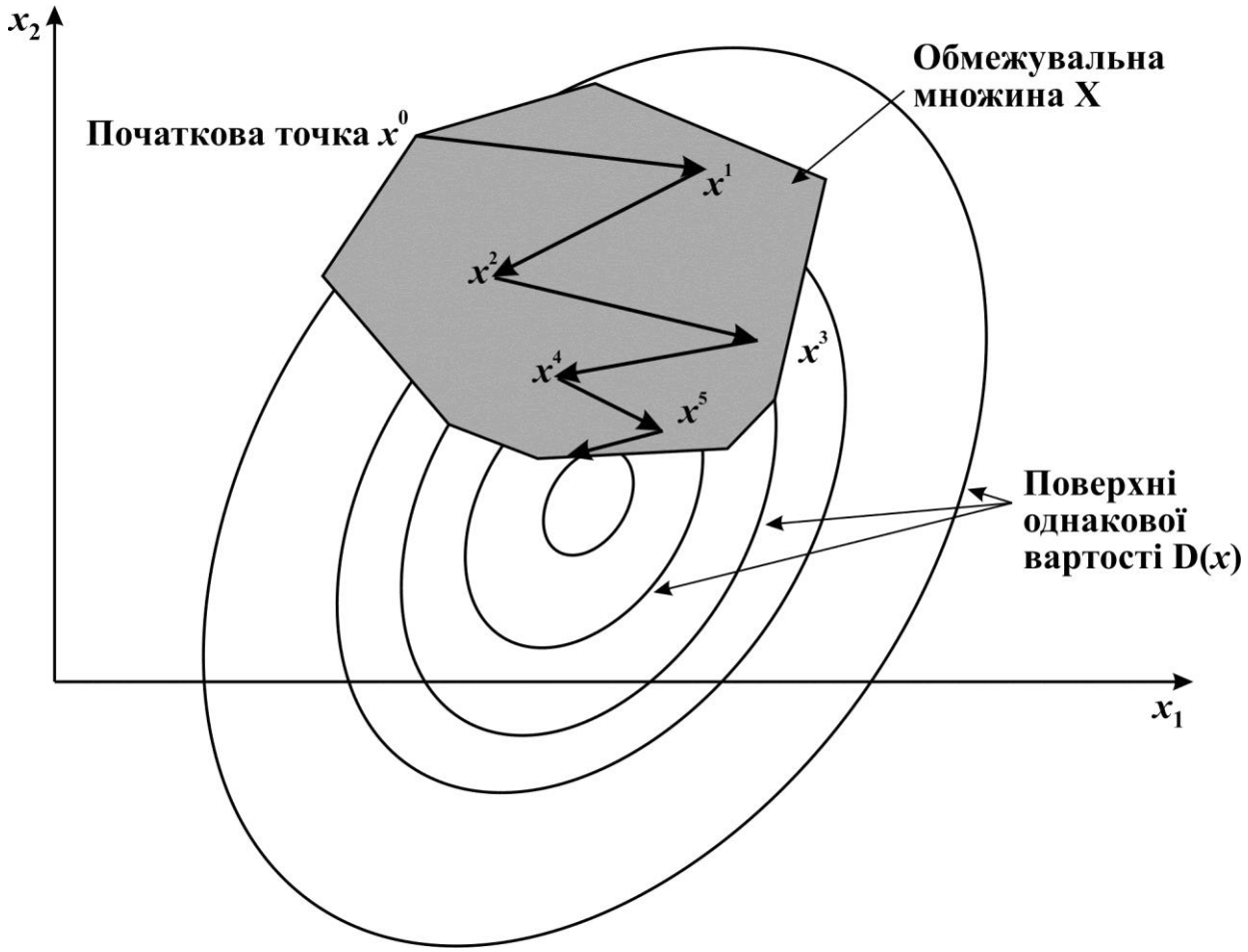


Рисунок 9.1 – Ітеративний метод спуску,
заснованого на допустимих напрямках спуску

9.2 Приклад розв'язання задачі оптимальної маршрутизації методом Франка – Волфа

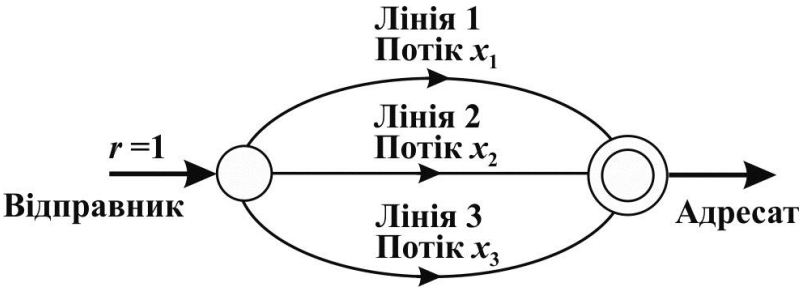
Розглянемо мережу, яка складається з трьох ліній, одного вузла-відправника й одного вузла-адресата, показану на рис. 9.2. Є три шляхи з відповідними потоками x_1, x_2 і x_3, які задовольняють обмеження

$$x_1 + x_2 + x_3 = 1, \quad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

Вартісною функцією є

$$D(x) = \frac{1}{2}\left(x_1^2 + x_2^2 + x_3^2\right) + 0{,}55x_3.$$

Лінійний член $0{,}55x_3$ можна приписати великому часу обробки й великому часу поширення сигналу по лінії 3.



Ітерація (номер) k	0	10	20	40	80	160	320
x_1	0,4	0,4593	0,4702	0,4795	0,4866	0,4917	0,4959
x_2	0,3	0,4345	0,4562	0,4717	0,4823	0,4893	0,4938
x_3	0,3	0,1061	00735,	0,0490	0,0310	0,0189	0,0110
Вартість	0,2945	0,2588	0,2553	0,2532	0,2518	0,2510	0,2506
$\frac{D\left(x^{k+1}\right)-D\left(x^*\right)}{D\left(x^k\right)-D\left(x^*\right)}$	0,7164	0,9231	0,9576	0,9774	0,9882	0,9939	0,9969

Рисунок 9.2 – Задача з прикладу і послідовні ітерації методу Франка–Волфа.

Вартісна функція має вигляд $D(x)=\frac{1}{2}\left(x_1^2+x_2^2+x_3^2\right)+0,55x_3$

Це проста задача, яку можна розв'язати аналітично. Можна довести, що для оптимального розв'язку $x=\left(x_1^*, x_2^*, x_3^*\right)$ виконується симетрія $x_1^*=x_2^*$ й тому є дві можливості: (а) $x_3^*=0$ і $x_1^*=x_2^*=1/2$, (б) $x_3^*=\beta>0$ і $x_1^*=x_2^*=(1-\beta)/2$. Випадок (б) неможливий, оскільки, відповідно до умови оптимальності, якщо $x_3^*>0$, то довжина шляху $3\left(=\frac{\partial D(x^*)}{\partial x_3}=0,1\beta+0,55\right)$ має бути меншою або дорівнювати довжинам шляхів 1 і 2 $\left(=\frac{\partial D(x^*)}{\partial x_1}=\frac{1-\beta}{2}\right)$, що, очевидно, неможливе. Тому оптимальним розв'язком є $x^*=(1/2,1/2,0)$.

Розглянемо тепер застосування ітерації Франка–Волфа (9.2) до вектора припустимих шляхових потоків $x=(x_1, x_2, x_3)$. Першопохідні довжини усіх трьох шляхів дорівнюють

$$\frac{\partial D(x)}{\partial x_1}=x_1, \quad \frac{\partial D(x)}{\partial x_2}=x_2, \quad \frac{\partial D(x)}{\partial x_3}=0,1x_3+0,55.$$

Використовуючи, по суті, ті самі доводи, що й раніше, зауважуємо, що найкоротшим шляхом буде або 1, або 2, залежно від того, $x_1 \le x_2$ чи $x_2 < x_1$, і відповідним потоком по найкоротшому шляху буде або $\bar{x}=(1,0,0)$, або

$\bar{x} = (0, 1, 0)$ відповідно. (У випадку $x_1 = x_2$ перевага надається шляху 1.) Тому ітерації Франка–Волфа мають вигляд

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} := \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \alpha^* \begin{bmatrix} 1 - x_1 \\ -x_2 \\ -x_3 \end{bmatrix} = \begin{bmatrix} x_1 + \alpha^*(x_2 + x_3) \\ (1 - \alpha^*) x_2 \\ (1 - \alpha^*) x_3 \end{bmatrix}, \quad \text{якщо} \quad x_1 \le x_2;$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} := \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \alpha^* \begin{bmatrix} -x_1 \\ 1 - x_2 \\ -x_3 \end{bmatrix} = \begin{bmatrix} (1 - \alpha^*) x_1 \\ x_2 + \alpha^*(x_1 + x_3) \\ (1 - \alpha^*) x_3 \end{bmatrix}, \quad \text{якщо} \quad x_2 < x_1.$$

Отже, на кожній ітерації частка α^* потоку по ненайкоротшому шляху перекидається на найкоротший шлях.

Розмір кроку α^* обчислюється шляхом лінійної мінімізації на $[0, 1]$. Оскільки $D(x)$ є квадратичною функцією, то мінімізацію можна виконати аналітично. Маємо

$$D\left[x + \alpha(\bar{x} - x)\right] =$$

$$= \frac{1}{2}\left(\left[x_1 + \alpha(\bar{x}_1 - x_1)\right]^2 + \left[x_2 + \alpha(\bar{x}_2 - x_2)\right]^2 + 0{,}1\left[x_3 + \alpha(\bar{x}_3 - x_3)\right]^2\right) +$$

$$+ 0{,}55\left[x_3 + \alpha(\bar{x}_3 - x_3)\right].$$

Диференціюючи цей вираз по α й прирівнюючи похідну до нуля, отримуємо значення $\bar{\alpha}$, за якого досягається мінімум функції $D\left[x + \alpha(\bar{x} - x)\right]$ по α без урахування обмежень. Воно дорівнює

$$\bar{\alpha} = \frac{x_1(\bar{x}_1 - x_1) + x_2(\bar{x}_2 - x_2) + (0{,}1x_3 + 0{,}55)(\bar{x}_3 - x_3)}{(\bar{x}_1 - x_1)^2 + (\bar{x}_2 - x_2)^2 - 0{,}1(\bar{x}_3 - x_3)^2},$$

де $\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3)$ – поточне значення вектора потоку по найкоротшому шляху $(1, 0, 0)$ або $(0, 1, 0)$ залежно від того, $x_1 \le x_2$ чи $x_2 < x_1$. Оскільки $(\bar{x} - x)$ є напрямом спуску, то $\bar{\alpha} \ge 0$. Оптимальне значення розміру кроку дає мінімум з урахуванням обмеження, що α належить $[0, 1]$, тому

$$\alpha^* = \min[1, \bar{\alpha}].$$

Можна перевірити, що у цьому прикладі $\bar{\alpha} < 1$ а, отже, $\alpha^* = \bar{\alpha}$.

На рис. 9.2 і 9.3 показані послідовні ітерації цього методу. Можна зазначити, що швидкість збіжності вповільнюється поблизу оптимального розв'язання. Причина в тому, що при наближенні до x^* напрямки пошуку

стають ортогональними напряму, що веде від поточного значення x^k до $x*$. Як показано в таблиці на рис. 8.2, відношення послідовних помилок у вартості

$$\frac{D\left(x^{k+1}\right)-D\left(x*\right)}{D\left(x^{k}\right)-D\left(x*\right)},$$

хоча завжди й менше одиниці, але фактично наближається до одиниці при $k \to \infty$. Ця величина відома як сублінійна швидкість збіжності; вона типова для методу Франка–Волфа.

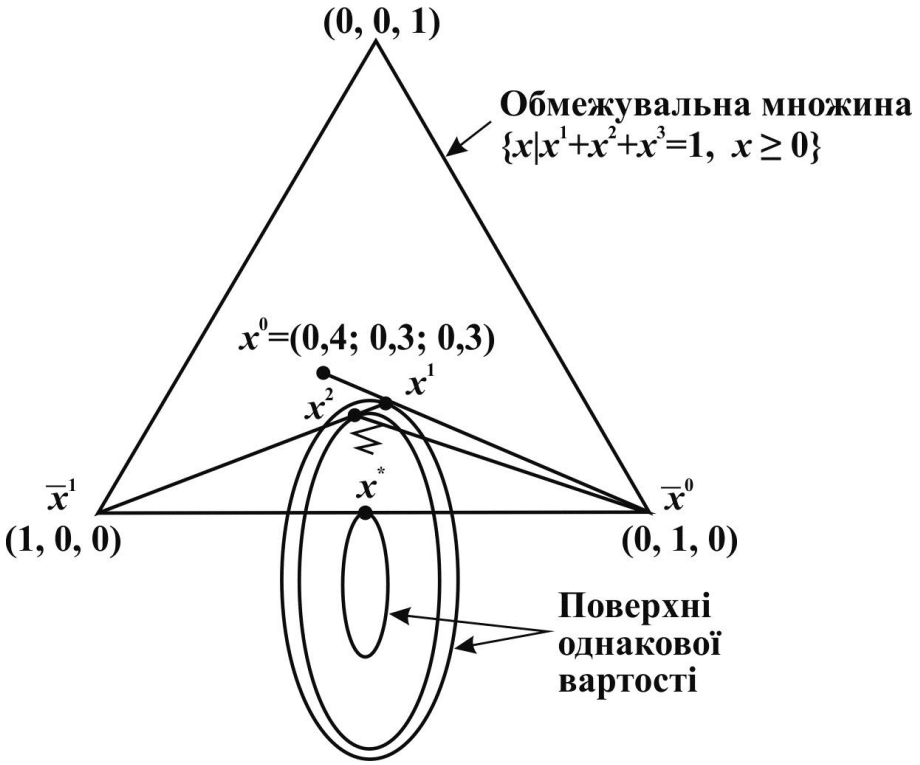


Рисунок 9.3 – Послідовні ітерації методу Франка–Волфа

9.3 Контрольні запитання та завдання

1. Дайте визначення першопохідної потока.
2. У чому полягає сутність методу Франка–Волфа?

Лекція 10
ПРОЕКЦІЙНІ МЕТОДИ ОПТИМАЛЬНОЇ МАРШРУТИЗАЦІЇ

Розглянемо тепер деякий клас алгоритмів припустимого напряму для розв'язання задачі оптимальної маршрутизації. Ці алгоритми працюють швидше метода Франка–Волфа й більше пристосовані для розподілених реалізацій. Вони також засновані на найкоротших шляхах і знаходять шлях з мінімальною першопохідною довжиною для кожної OA-пари на кожній ітерації. Збільшення потоку обчислюється для кожного шляху на основі відносних величин довжин шляхів й іноді на основі других похідних вартісної функції. Якщо це збільшення виявляється настільки великим, що шляховий потік стає негативним, то він покладається рівним нулю, тобто «проектується» назад у позитивний октант. Існує кілька методів такого типу, які становлять інтерес у зв'язку з задачею маршрутизації. Всі вони можуть розглядатися як версії з обмеженням більш загальних методів оптимізації без обмежень, таких, як метод найшвидшого спуску й метод Ньютона, які докладніше описані в літературі з нелінійного програмування. Спочатку ми стисло опишемо ці методи в загальній постановці нелінійної оптимізації, а потім розглянемо застосування цих методів до задачі маршрутизації.

10.1 Нелінійна оптимізація без обмеження

Нехай f – двічі диференційована функція n-мірного вектора $x=(x_1,\ldots,x_n)$; градієнтна матриця й матриця Гесса в будь-якій точці x, позначені через $\nabla f(x)$ і $\nabla^2 f(x)$ відповідно, і мають вигляд

$$\nabla f(x)=\begin{bmatrix}\dfrac{\partial f(x)}{\partial x_1}\\ \cdots \\ \dfrac{\partial f(x)}{\partial x_n}\end{bmatrix}, \qquad \nabla^2 f(x)=\begin{bmatrix}\dfrac{\partial^2 f(x)}{(\partial x_1)^2} & \cdots & \dfrac{\partial^2 f(x)}{\partial x_1 \partial x_n}\\ & \cdots & \\ \dfrac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \dfrac{\partial^2 f(x)}{(\partial x_n)^2}\end{bmatrix}.$$

Припустимо, що для всіх x матриця $\nabla^2 f(x)$ є ненегативно визначеною, безперервною по x (симетрична $n\times n$ матриця A з елементами A_{ij} називається позитивно напіввизначеною, якщо квадратична форма $\sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij} z_i z_j$ ненегативна

для усіх векторів $z=(z_1,\ldots,z_n)$. Матриця A позитивно визначена, якщо ця квадратична форма позитивна для усіх векторів $z\neq 0$). Можна показати, що звідси витікає опуклість функції f.

Метод найшвидшого спуску для знаходження мінімуму функції f за відсутності обмежень починається з деякої довільної початкової точки x^0 і діє відповідно до ітерації

$$x^{k+1}=x^k-\alpha^k\nabla f\left(x^k\right),\quad k=0,1,\ldots, \tag{10.1}$$

де розмір кроку α^k є позитивним числом, обумовленим відповідно до деякого правила. Ідея тут схожа на ту, що обговорювалася у попередній лекції, а саме змінювати x^k в напрямі спуску. В (10.1) зміна здійснюється в напрямі негативного градієнта, що є напрямом спуску, тому що це робить негативним скалярний добуток з вектором градієнта (крім випадку, коли $\nabla f\left(x^k\right)=0$, при цьому вектор x^k є оптимальним). Величина α^k або обирається шляхом мінімізації виразу

$$f\left[x^k-\alpha^k\nabla\left(x^k\right)\right]=\min_{\alpha>0} f\left[x^k-\alpha\nabla\left(x^k\right)\right], \tag{10.2}$$

або покладається такою, що дорівнює постійній позитивній величині $\bar{\alpha}$

$$\alpha^k=\bar{\alpha}\quad \text{для усіх}\quad k. \tag{10.3}$$

Існує багато результатів, що належать до збіжності ітерації (10.1) з розміром кроку (10.2) або (10.3). Наприклад, якщо в f тільки одна точка мінімуму за відсутності обмежень, то можна показати, що послідовність $\left\{x^k\right\}$, породжена (10.1) і (10.2), сходиться до точки мінімуму за будь-якого початкового x^0. Так само за будь-якого початкового вектора x^0 послідовність, породжена (10.1) і (10.3), сходиться до точки мінімуму, якщо $\bar{\alpha}$ досить мале. Але, на жаль, швидкість збіжності $\left\{x^k\right\}$ може бути досить малою. Можна показати, що у випадку правила лінійної мінімізації (10.2), якщо f є позитивно визначеною квадратичною функцією, то має місце нерівність

$$\frac{f\left(x^{k+1}\right)-f^*}{f\left(x^k\right)-f^*}\leq\left(\frac{M-m}{M+m}\right)^2, \tag{10.4}$$

де $f^*=\min_x f(x)$, а M і m відповідно максимальне й мінімальне власні значення матриці $\nabla^2 f(x)$. Крім того, існує така початкова точка x^0, для якої нерівність (10.4) перетворюється в рівність для кожного k. Отже, якщо співвідношення M/m велике (у цьому випадку поверхні однакових значень функції f є дуже витягнутими еліпсоїдами), то швидкість збіжності мала.

Аналогічні результати можна довести для методу, обумовленому (10.1) і (10.3), причому ці результати якісно залишаються тими самими для функції f скрізь безперервною та позитивно визначеною матрицею Гесса.

Швидкість збіжності методу найшвидшого спуску можна покращити, якщо попередньо помножити вектор градієнта на відповідно підібрану позитивно визначену масштабну матрицю B^k, у результаті чого отримаємо ітерацію

$$x^{k+1} = x^k - \alpha^k B^k \nabla f\left(x^k\right), \quad k = 0, 1, \ldots. \tag{10.5}$$

Цей метод заснований на ідеї зміни x^k уздовж напряму спуску. (Скалярний добуток напряму зміни $-B^k \nabla f\left(x^k\right)$ і градієнта $\nabla f\left(x^k\right)$ негативний, оскільки B^k є позитивно визначеною матрицею.) З погляду швидкості збіжності найкращий метод маємо, якщо

$$B^k = \left[\nabla^2 f\left(x^k\right)\right]^{-1} \tag{10.6}$$

(припустивши, що матриця $\nabla^2 f\left(x^k\right)$ не вироджена). *Це метод Ньютона*, який, як можна показати, має дуже велику (квадратичну) швидкість збіжності поблизу точки мінімуму, якщо розмір кроку α^k обраний таким, що дорівнює одиниці. На жаль, ця швидкість збіжності досягається за рахунок того, що в ході обчислення зворотної матриці в (10.6) може знадобитися дуже великий обсяг обчислень. Часто виявляються корисними інші способи вибору β^k, за яких апроксимується оптимальний вибір $\left[\nabla^2 f\left(x^k\right)\right]^{-1}$, але не потрібно такого великого обсягу обчислень. Один із простих способів, який часто дає непогані результати, полягає в тому, що за B^k приймається діагональна апроксимація зворотної матриці Гесса, тобто

$$B^k = \begin{bmatrix} \left[\dfrac{\partial^2 f\left(x^k\right)}{\left(\partial x_1\right)^2}\right]^{-1} & & & \\ & \left[\dfrac{\partial^2 f\left(x^k\right)}{\left(\partial x_2\right)^2}\right]^{-1} & & 0 \\ & & \cdots & \\ 0 & & & \left[\dfrac{\partial^2 f\left(x^k\right)}{\left(\partial x_n\right)^2}\right]^{-1} \end{bmatrix}. \tag{10.7}$$

За такого вибору ітерацію (10.5) можна переписати у більш простому вигляді

$$x_i^{k+1} = x_i^k - \alpha^k \left[\frac{\partial^2 f\left(x^k\right)}{\left(\partial x_i\right)^2} \right]^{-1} \frac{\partial f\left(x^k\right)}{\partial x_i}, \quad i = 1, \ldots, n. \tag{10.8}$$

Нелінійна оптимізація в позитивному октанті

Далі розглянемо задачу мінімізації функції f за обмеження $x_i \geq 0$ для $i = 1, \ldots, n$, тобто задачу

$$\text{мінімізувати } f(x)$$
$$\text{за обмежень } x \geq 0. \tag{10.9}$$

Прямим аналогом методу найшвидшого спуску буде

$$x^{k+1} = \left[x^k - \alpha^k \nabla f\left(x^k\right) \right]^+, \quad k = 0, 1, \ldots, \tag{10.10}$$

де для будь-якого вектора z ми позначаємо через $[z]^+$ проекцію z на позитивний октант

$$[z]^+ = \begin{bmatrix} \max\{0, z_1\} \\ \max\{0, z_2\} \\ \vdots \\ \max\{0, z_n\} \end{bmatrix}. \tag{10.11}$$

Даний метод проілюстрований на рис. 10.1.

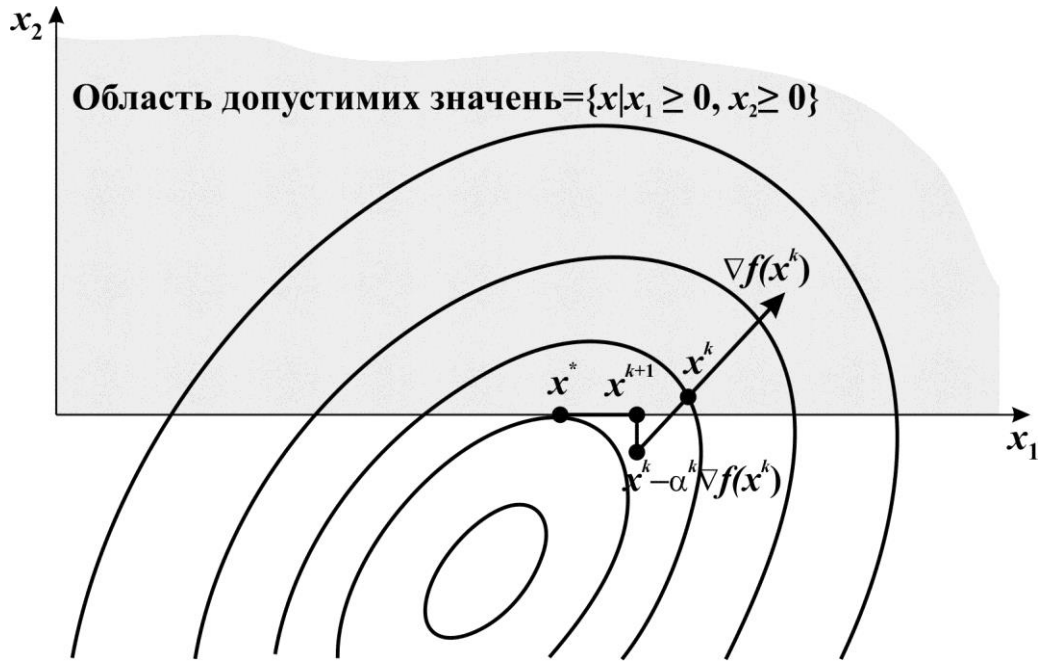


Рисунок 10.1 – Робота проекційного градієнтного алгоритму

Можна показати, що результати про збіжність, згадані раніше у зв'язку з (10.1) – методом найшвидшого спуску без обмежень, також правильні для аналога з обмеженням (10.10). Те саме правильно й для методу

$$x^{k+1} = \left[x^k - \alpha^k B^k \nabla f\left(x^k\right) \right]^+ ,$$

де B^k – діагональна, позитивно визначена масштабна матриця. Якщо B^k обрано у вигляді (10.7), то звідси випливає така ітерація (відповідно до (10.8) і (10.11)):

$$x_i^{k+1} = \max\left\{ 0,\ x_i^k - \alpha^k \left[\frac{\partial^2 f\left(x^k\right)}{\left(\partial x_i\right)^2} \right]^{-1} \frac{\partial f\left(x^k\right)}{\partial x_i} \right\}, \quad i = 1, \ldots, n. \qquad (10.12)$$

Тепер отримано вирази як для перших, так і для других похідних скороченої вартісної функції $\tilde{D}\left(\tilde{x}\right)$, й тому можна застосовувати метод проекцій (10.12). Ітерація має вигляд

$$x_p^{k+1} = \max\left\{ 0,\ x_p^k - \alpha^k H_p^{-1}\left(d_p - d_{\bar{p}_w}\right) \right\} \text{ для усіх } \ w \in W, \quad p \in P_w, \quad p \neq \bar{p}_w , \quad (10.13)$$

де d_p і $d_{\bar{p}_w}$ є першопохідними довжинами шляхів p і $\bar{p}_w$; вони дорівнюють

$$d_p = \sum_{\substack{\text{по всем линиям } (i,j) \\ \text{на пути } p}} D'_{ij}\left(F_{ij}^k\right), \qquad d_{\bar{p}_w} = \sum_{\substack{\text{по всем линиям } (i,j) \\ \text{на пути } p}} D'_{ij}\left(F_{ij}^k\right), \qquad (10.14)$$

а H_p – другопохідна довжина

$$H_p = \sum_{(i,j) \in L_p} D''_{ij}\left(F_{ij}^k\right). \qquad (10.15)$$

Розмір кроку α^k є деяким позитивним числом, яке можна обирати різними способами. Наприклад, α^k може бути константою або його можна обирати шляхом деякої лінійної мінімізації. Вибір розміру кроку обговорюватиметься пізніше.

Можна зробити так висновки щодо ітерації (10.13):

1. Оскільки для кожної OA-пари $w \in W$ ми маємо $d_p \geq d_{\bar{p}_w}$ для всіх $p \neq \bar{p}_w$, то це означає, що всі потоки по ненайкоротших шляхах $x_p\left(p \neq \bar{p}_w\right)$, які більше нуля, стануть меншими на величину, що буде перекинута на МППД-шлях $\bar{p}_w$. Якщо розмір кроку α^k досить великий, то весь потік з ненайкоротших шляхів буде перекинутий на найкоротший шлях. Тому ітерацію (10.13) можна також розглядати як узагальнення метода адаптивної маршрутизації, заснованого на найкоротших шляхах і такого, що має параметри α^k, H_p і $\left(d_p - d_{\bar{p}_w}\right)$, які визначають величину потоку, що перекидається на найкоротший шлях.

2. Потоки по ненайкоротших шляхах x_p, $p \neq \bar{p}_w$, які дорівнюють нулю, залишаться рівними нулю. Тому обчислення, зазначені в (10.13), слід проводити лише для тих шляхів, по яких проходить позитивний потік.

3. Лише на тих шляхах, по яких проходив позитивний потік при початковій картині потоків або тими, які були МППД-шляхами на деякій попередній ітерації, можливий позитивний потік на початку ітерації. Це має велике значення, оскільки звідси випливає, що алгоритм прагне зберегти кількість шляхів, які несуть потік, невеликим, що позитивно позначається на кількості обчислень і обсязі пам'яті, необхідної для виконання кожної ітерації.

Що стосується вибору розміру кроку α^k, то існує кілька можливостей. Можна покласти α^k таким, що дорівнює константі ($\alpha^k \equiv \alpha$ для усіх k). Для такого вибору можна показати, що для будь-якої початкової множини шляхових потоків існує $\bar{\alpha} > 0$, таке, що, коли $\bar{\alpha} \in (0, \bar{\alpha}]$, то послідовність, породжена ітерацією (10.13)–(10.15), збігається до оптимальної вартості задачі. Вибір величини постійного розміру кроку має велике значення. З досвіду нелінійного програмування й аналізу відомо, що розмір кроку, який дорівнює одиниці, звичайно добре підходить для методу Ньютона, а також діагональної апроксимації методу Ньютона (10.7), де використовується масштабування, засноване на других похідних. Досвід показав, що вибір α^k в (10.13) близьким до одиниці зазвичай виявляється досить гарним незалежно від значень вхідних потоків r_w. Виходить навіть краще, якщо ітерацію (10.13) виконувати за один раз тільки для однієї OA-пари (або для одного відправника), тобто спочатку виконати (10.13) з $\alpha^k = 1$ для єдиної OA-пари (або відправника), потім перерахувати значення відповідних сумарних потоків у лініях, з огляду на зміну шляхових потоків цієї OA-пари, і продовжити обчислення для наступної OA-пари, циклічно обходячи усі OA-пари. Розумне пояснення цього засновано на тому, що, відкидаючи недіагональні члени в матриці Гесса ((10.5) і (10.7)), ми тим самим, по суті, нехтуємо взаємодією між потоками різних OA-пар. Інакше кажучи, ітерація (10.13) певною мірою заснована на передумові, що кожна OA-пара перебудовуватиме власні шляхові потоки, у той час як інші OA-пари зберігають свої потоки незмінними. Виконання ітерації (10.13) за один раз тільки для однієї OA-пари зменшує потенційний небажаний ефект, пов'язаний з відкиданням недіагональних членів у матриці Гесса, і збільшує впевненість у тому, що одиничний розмір кроку є підходящим і ефективним. За цих обставин досвід показує, що ітерація (10.13) зазвичай працює добре з одиничним розміром кроку.

Використання постійного розміру кроку добре підходить для розподілених додатків. Для централізованих обчислень можна вибирати α^k шляхом якого-небудь простого лінійного пошуку. Наприклад, можна почати з одиничного розміру кроку, обчислити відповідну вартість, і якщо ніякого зменшення в порівнянні з $D\left(x^k\right)$ не вийде, то поступово зменшувати розмір кроку доти, поки зменшення $D\left(x^{k+1}\right) < D\left(x^k\right)$ не буде отримано. Не можна показати, що така схема збігається з оптимальним рішенням. Однак на практиці вона зазвичай працює добре.

Проекційний алгоритм зазвичай швидко приводить в околицю оптимального рішення. Підійшовши близько до розв'язку (наскільки «близько», залежить від конкретного завдання), алгоритм сповільнює роботу. Його швидкість біля розв'язку часто є задовільною й зазвичай набагато більшою, ніж у методі Франка–Волфа.

Для того щоб проекційний алгоритм сходився швидше поблизу оптимального розв'язку, необхідно враховувати недіагональні члени в матриці Гесса.

10.2 Приклад розв'язання задачі оптимальної маршрутизації без обмеження

Розглянемо мережу, показану на рис. 10.2.

Є лише дві OA-пари (1.5) і (2.5) з відповідними вхідними потоками $r_1 = 4$ та $r_2 = 8$. Розглянемо такі два шляхи для кожної OA-пари.

Шляхи для OA-пари (1, 5)

$$p_1(1) = \{(1,4),(4,5)\},$$

$$p_2(1) = \{(1,3),(3,4),(4,5)\}.$$

Шляхи для OA-пари (2, 5)

$$p_1(2) = \{(2,4),(4,5)\},$$

$$p_2(2) = \{(2,3),(3,4),(4,5)\}.$$

Розглянемо приклад задачі маршрутизації, у якій вартісні функції для всіх ліній однакові й мають вигляд

$$D_{ij}\left(F_{ij}\right) = \frac{1}{2}\left(F_{ij}\right)^2 \quad \text{для всіх} \quad (i,\ j).$$

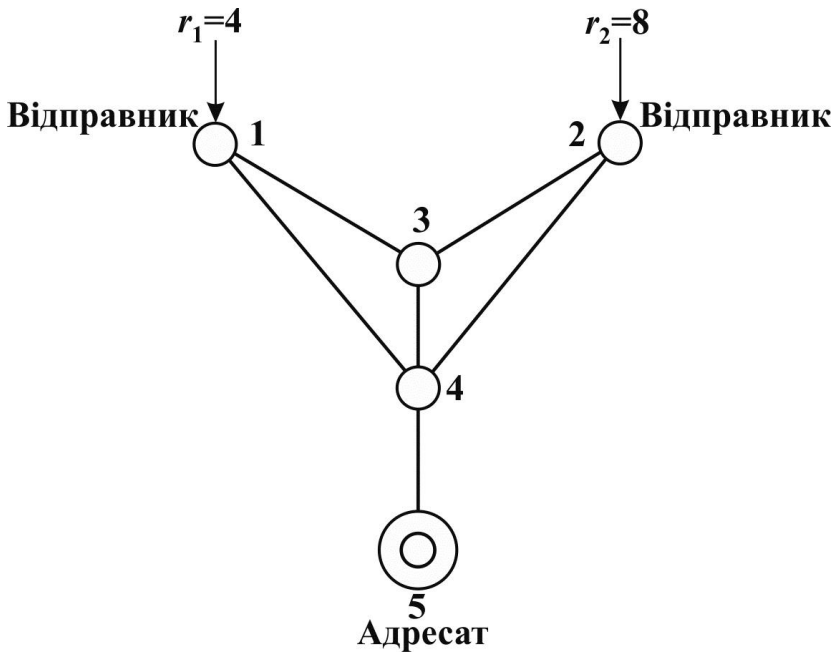


Рисунок 10.2 – Приклад мережі, у якій вузол 5 є єдиним адресатом та є два відправника: вузли 1 і 2

Розглянемо картину початкових шляхових потоків, у якій весь потік кожної *OA*-пари проходить по середній лінії (3, 4). Отримані в результаті потоки, подані в табл. 10.1 і 10.2.

Таблиця 10.1 – Початкові шляхові потоки для прикладу маршрутизації

OA-пара	Шлях	Шляховий потік
(1, 5)	$p_1(1)$	0
	$p_2(1)$	4
(2, 5)	$p_1(2)$	0
	$p_2(2)$	8

Таблиця 10.2 – Початкові сумарні потоки по лініях для прикладу маршрутизації

Лінія	Сумарний потік у лінії
(1, 3)	4
(1, 4)	0
(2, 3)	8
(2, 4)	0
(3, 4)	12
(4, 5)	12
Інші	0

Першопохідна довжина кожної лінії дорівнює

$$D'_{ij}\left(F_{ij}\right)=F_{ij}\,,$$

тому сумарні потоки в лініях, подані в табл. 10.2, є також довжинами ліній для першої ітерації. Відповідні першопохідні довжини шляхів подані в табл. 10.3.

Таблиця 10.3 – Початкові першопохідні довжини для прикладу маршрутизації

OA-пара	Шлях	Першопохідна шляху
(1, 5)	$p_1(1)$	12
	$p_2(1)$	28
(2, 5)	$p_1(2)$	12
	$p_2(2)$	32

Найкоротшими шляхами для першої ітерації будуть $p_1(1)$ і $p_1(2)$ для *OA*-пар (1,5) і (2, 5) відповідно.

Конкретний вид ітерації (10.13)–(10.15) покажемо для першої *OA*-пари. У цьому випадку для ненайкоротшого шляху $p=p_2(1)$ і найкоротшого шляху $\overline{p}=p_1(1)$ маємо $d_p=28$ й $d_{\overline{p}}=12$. Також $H_p=3$ (кожна лінія має другопохідну довжину $D''_{ij}=1$ й існують три лінії, які належать або $p_1(1)$, або $p_2(1)$, але не обом водночас. Тому ітерація (10.13) має вигляд

$$x^p:=\max\left\{0,4-\alpha^k\frac{1}{3}\left(28-12\right)\right\}=\max\left\{0,4-\frac{16\alpha^k}{3}\right\}=4-\min\left\{\frac{4,16\alpha^k}{3}\right\}$$

і

$$x_{\overline{p}}:=r_1-x_p\,.$$

У більш загальній постановці нехай $x_1(1)$, $x_2(1)$, $x_1(2)$ і $x_2(2)$ означають потоки по шляхах $p_1(1)$, $p_2(2)$, і $p_1(2)$, $p_2(2)$ відповідно перед початком наступної ітерації. Відповідні довжини шляхів дорівнюють

$$d_{p_1(1)}=x_1(1)+r_1+r_2,$$

$$d_{p_2(1)}=2x_2(1)+x_2(2)+r_1+r_2,$$

$$d_{p_1(2)}=x_1(2)+r_1+r_2,$$

$$d_{p_2(2)}=2x_2(2)+x_2(1)+r_1+r_2.$$

Другопохідна довжина H_p з (10.15) дорівнює 3. Алгоритм (10.13)–(10.15) має такий вигляд.

Для $i = 1, 2$

$$x_1(i) := \begin{cases} x_1(i) - \min\left[x_1(i),\ \dfrac{\alpha^k}{3}\left[d_{p_1(i)} - d_{p_2(i)} \right] \right], & \text{якщо} \quad d_{p_1(i)} > d_{p_2(i)}, \\ x_1(i) + \min\left[x_2(i),\ \dfrac{\alpha^k}{3}\left[d_{p_2(i)} - d_{p_1(i)} \right] \right] \end{cases}$$

у протилежному випадку, $x_2(i) := r_i - x_1(i)$.

Зауважимо, що наявність лінії (4, 5) ніяк не позначається на вигляді ітерації, і так це й має бути, оскільки сумарний потік по лінії (4, 5) завжди дорівнює $r_1 + r_2$ незалежно від маршрутизації.

З таблиці 10.4 видно, що при одиничному розмірі кроку збіжність до околиці розв'язку дуже швидка для обох варіантів «все водночас» і «одна за один раз». Зі зростанням розміру кроку з'являється небезпека втрати властивості збіжності й, зазвичай, насамперед перестає сходитися алгоритм у варіанті «все водночас».

Таблиця 10.4 – Послідовності вартостей, які генеруються проекційним градієнтним методом, для прикладу маршрутизації, що розглядається, при різних розмірах кроку

Ітерація	Варіант «усі водночас»			Варіант «одна за один раз»		
k	$\alpha^k \equiv 0{,}5$	$\alpha^k \equiv 1{,}0$	$\alpha^k \equiv 1{,}8$	$\alpha^k \equiv 0{,}5$	$\alpha^k \equiv 1{,}0$	$\alpha^k \equiv 1{,}8$
0	184,00	184,00	184,00	184,00	184,00	184,00
1	110,88	104,00	112,00	114,44	101,33	112,00
2	102,39	101,33	118,72	103,54	101,00	109,15
3	101,28	101,03	112,00	101,63		101,79
4	101,09	101,00	118,72	101,29		101,56
5	101,03		112,00	101,08		101,33
6	101,01		118,72	101,03		101,18
7	101,00		112,00	101,01		101,12
8			118,72	101,00		101,09
9			112,00			101,06
10			118,72			101,03

Лекція 11
ПРОТОКОЛ МІЖМЕРЕЖНОЇ ВЗАЄМОДІЇ

11.1 Формат IP-пакета

Протокол міжмережної взаємодії

Протокол IP (Intranet Protocol – міжмережний протокол) описаний у документі RFC 751. У кожній наступній мережі, що лежить на шляху переміщення пакета, протокол IP звертається до засобів транспортування цієї мережі, щоб з їхньою допомогою передати пакет на маршрутизатор, який веде до наступної мережі, або безпосередньо на вузол-отримувач. Отже, однією з найважливіших функцій IP є підтримка інтерфейсу з технологіями, що розташовані нижче за мережі, які утворюють складену мережу. Крім того, до функції протоколу IP входить підтримка інтерфейсу з протоколами транспортного рівня, що лежить вище, зокрема, з протоколом TCP, який вирішує усі питання забезпечення надійної доставки даних по складеній мережі у стеку TCP/IP.

Протокол IP належить до протоколів без встановлення з'єднань, він підтримує обробку кожного IP-пакета як незалежної одиниці обміну, не пов'язаної з іншими пакетами.

У протоколі IP немає механізмів, зазвичай застосовуваних для забезпечення вірогідності кінцевих даних.

Якщо під час просування пакета відбувається будь-яка помилка, то протокол IP за своєю ініціативи нічого не здійснює для виправлення цієї помилки. Наприклад, якщо на проміжному маршрутизаторі пакет був відкинутий через помилку за контрольною сумою, то модуль IP не намагається заново послати загублений пакет. Інакше кажучи, протокол IP реалізує політику доставки «за можливістю» (з максимальними зусиллями).

Розглянемо основну функцію протоколу IP – маршрутизацію – докладно. Ґрунтовно вивчимо структуру таблиць маршрутизації як без використання, так і з використанням масок. Наведемо приклади застосування масок однакової та змінної довжини, адресних просторів, що перекриваються, поділу на підмережі й об'єднання підмереж. Також дослідимо можливості протоколу IP, пов'язані з фрагментацією пакетів.

Формат IP-пакета

Є прямий зв'язок між кількістю полів заголовка пакета й функціональною складністю протоколу, який працює з цим заголовком. Чим простіше заголовок – тим простіше відповідний протокол. Більша частина дій протоколу пов'язана з обробкою тієї службової інформації, яка переноситься в полях заголовка пакета. Вивчаючи призначення кожного поля заголовка IP-пакета, ми отримуємо не лише формальні знання про структуру пакета, але й ознайомлюємося з основними функціями протоколу IP. IP-пакет складається з полів заголовка й даних. Далі перераховані поля заголовка показано на рис. 11.1.

<table>
<tr><td>4 біта
Номер
версії</td><td>4 біта
Довжина
заголовка</td><td colspan="5">8 біт
Тип сервісу
PR | D | T | R</td><td colspan="3">16 біт
Загальна довжина</td></tr>
<tr><td colspan="6">16 біт
Ідентифікатор пакета</td><td colspan="2">3 біта
Прапорці
D | M</td><td colspan="2">13 біт
Зсув фрагмента</td></tr>
<tr><td colspan="2">8 біт
Час життя</td><td colspan="4">8 біт
Протокол
верхнього рівня</td><td colspan="4">16 біт
Контрольна сума</td></tr>
<tr><td colspan="8">32 біта
IP-адреса джерела</td></tr>
<tr><td colspan="8">32 біта
IP-адреса призначення</td></tr>
<tr><td colspan="8">Параметри та вирівнювання</td></tr>
</table>

Рисунок 11.1 – Структура заголовка IP-пакета

Поле номера версії займає 4 біти та ідентифікує версію протоколу IP. Сьогодні скрізь використовується версія 4 (IPv4), хоча все частіше зустрічається й нова версія (IPv6).

Значення довжини заголовка IP-пакета також займає 4 біти й вимірюється в 32-бітних словах. Завичай заголовок має довжину в 20 байт (п'ять 32-бітних слів), але при додаванні деякої службової інформації це значення може бути збільшене за рахунок додаткових байтів у полі параметрів. Найбільша довжина заголовка становить 60 байт.

Поле типу сервісу (Type of Service, ToS) має й іншу, сучаснішу назву – байт диференційованого обслуговування, або DS-байт. Цим двом назвам відповідають два варіанти інтерпретації цього поля. В обох випадках дане поле слугує одній меті – зберіганню ознак, які відбивають вимоги до якості обслуговування пакета. У попередньому варіанті перші три біти

містять значення пріоритету пакета: від найнижчого – 0 до найвищого – 7. Маршрутизатори й комп'ютери можуть брати до уваги пріоритет пакета й обробляти більш важливі пакети в першу чергу. Наступні три біти поля ToS визначають критерій вибору маршруту. Якщо біт D (Delay – затримка) встановлений в 1, то маршрут має обиратися для мінімізації затримки доставки даного пакета; встановлений біт T (Throughput – пропускна здатність) – для максимізації пропускної здатності; а біт R (Reliability – надійність) – для максимізації надійності доставки. Два біти, що залишилися мають нульове значення.

Стандарти диференційованого обслуговування, прийняті наприкінці 1990-х років, дали нову назву цьому полю й перевизначили призначення його бітів. В DS-байті також використовуються тільки старші 6 бітів, а два молодші біти залишаються як резерв.

Поле загальної довжини займає 2 байти та характеризує загальну довжину пакета з урахуванням заголовка й поля даних. Максимальна довжина пакета обмежена розрядністю поля, яке визначає цю величину, і становить 65535 байт, але у більшості комп'ютерів і мереж такі великі пакети не використовуються. Під час передачі по мережах різного типу довжина пакета обирається з урахуванням максимальної довжини пакета протоколу нижнього рівня, який несе IP-пакети. Якщо це кадри Ethernet, то обираються пакети з максимальною довжиною 1500 байт, які вміщаються в поле даних кадра Ethernet. У стандартах TCP/IP передбачається, що усі хости мають бути готові приймати пакети довжиною до 576 байт (незалежно від того, приходять вони цілком чи фрагментами).

Ідентифікатор пакета займає 2 байти й використовується для розпізнавання пакетів, що утворювалися шляхом поділу на частини (фрагментації) вихідного пакета. Усі частини (фрагменти) одного пакета повинні мати однакове значення цього поля.

Прапорці займають 3 біти й містять ознаки, пов'язані із фрагментацією. Встановлений в 1 біт DF (Do not Fragment – не фрагментувати) забороняє маршрутизатору фрагментувати даний пакет, а встановлений в 1 біт MF (More Fragments – більше фрагментів) говорить про те, що даний пакет є проміжним (не останнім) фрагментом. Біт, що залишився, зарезервований.

Поле зсуву фрагмента займає 13 біт і задає зсув у байтах поля даних цього фрагмента відносно початку поля даних вихідного (не фрагментованого) пакета. Використовується під час складання/розбирання фрагментів пакетів. Зсув має бути кратним 8 байтам.

Поле часу життя (Time To Live, TTL) займає один байт і використовується для завдання граничного терміну, протягом якого пакет може переміщуватися по мережі. Час життя пакета вимірюється в секундах і задається джерелом. Після закінчення кожної секунди перебування на кожному з маршрутизаторів, через які проходить пакет під час своєї «подорожі» по мережі, з його поточного часу життя віднімається одиниця; одиниця віднімається й у тому випадку, якщо час перебування був менше секунди. Оскільки сучасні маршрутизатори рідко обробляють пакет довше, ніж одну секунду, то час життя можна інтерпретувати як максимальну кількість транзитних вузлів, які дозволено пройти пакету.

Якщо значення поля часу життя стає нульовим до того, як пакет досягає отримувача, пакет знищується. Отже, час життя є свого роду годинниковим механізмом самознищення пакета.

Поле протоколу верхнього рівня займає один байт і містить ідентифікатор, що вказує, якому протоколу верхнього рівня належить інформація, розміщена в полі даних пакета. Значення ідентифікаторів для різних протоколів приводяться в документі RFC 1700, доступному за адресою http://www.iana.org.

Наприклад, 6 означає, що в пакеті знаходиться повідомлення протоколу TCP, 17 – протоколу UDP, 1 – протоколу ICMP.

Контрольна сума заголовка займає 2 байти (16 біт) і обчислюється лише за заголовком. Оскільки деякі поля заголовка змінюють своє значення у процесі передачі пакета по мережі (наприклад, поле часу життя), контрольна сума перевіряється й повторно обчислюється на кожному маршрутизаторі й кінцевому вузлі як доповнення до суми всіх 16-бітних слів заголовка.

Поля IP-адрес джерела й приймача мають однакову довжину – 32 біти.

Поле параметрів є необов’язковим і використовується зазвичай лише в ході налагодження мережі. Це поле складається з декількох підполів одного з восьми визначених типів. У цих підполях можна вказувати точний маршрут, реєструвати маршрутизатори, які проходить пакет, вміщувати дані системи безпеки або тимчасові оцінки. Оскільки кількість підполів у полі параметрів може бути довільною, то наприкінці заголовка має бути додано декілька нульових байтів для вирівнювання заголовка пакета по 32-бітній межі.

Далі наведена роздруківка значень полів заголовка одного з реальних IP-пакетів, захоплених у мережі Ethernet засобами аналізатора протоколів мережного монітора (Network Monitor, NM) компанії Microsoft. У NM в дужках дає шістнадцяткові значення полів, крім того, програма іноді подає числові

коди полів у вигляді, більш зручному для читання. Наприклад, NM інтерпретує код 6 у поле протоколу, поміщаючи туди назву відповідного протоколу – TCP

IP: Version = 4 (0x4)
IP: Header Length = 20 (0x14)
IP: Service Type = 0 (0x0)
IP: Precedence = Routine
IP: ...0…. = Normal Delay
IP: ….0… = Normal Throughput
IP: …..0.. = Normal Reliability
IP: Total Length = 54 (0x36)
IP: Identification = 31746 (0x7C02)
IP: Flags Summary = 2 (0x2)
IP: …….0 = Last fragment in datagram
IP: ……1. = Cannot fragment datagram
IP: Fragment Offset = 0 (0x0) bytes
IP: Time to Live = 128 (0x80)
IP: Protocol = TCP – Transmission Control
IP: Checksum = 0xeb86
IP: Source Address = 194.85.135.75
IP: Destination Address = 194.85.135.66
IP: Data: Number of data bytes remaining = 34 (0x0022)

11.2 Схема IP-маршрутизації

Розглянемо механізм IP-маршрутизації на прикладі складеної мережі, що подана на рис. 11.2. У цій мережі 20 маршрутизаторів (зображених у вигляді пронумерованих квадрантних блоків) поєднують 18 мереж у загальну мережу; N1, N2, ..., N18 – номера мереж.

На кожному маршрутизаторі й кінцевих вузлах A і B функціонують протоколи IP.

До декількох інтерфейсів (портів) маршрутизаторів приєднуються мережі. Кожний інтерфейс маршрутизатора можна розглядати як окремий вузол мережі: він має мережну адресу й локальну адресу в тій підмережі, яка до нього підключена. Наприклад, маршрутизатор під номером 1 має три інтерфейси, до яких підключені мережі N1, N2, N3. На рисунку мережні адреси цих портів позначені IP_{11}, IP_{12} і IP_{13}.

Інтерфейс IP_{11} є вузлом мережі N1 і, отже, у поле номера мережі порту IP_{11} міститься номер N1. Аналогічно інтерфейс IP_{12} – це вузол у мережі N2, а порт IP_{13} – вузол у мережі N3.

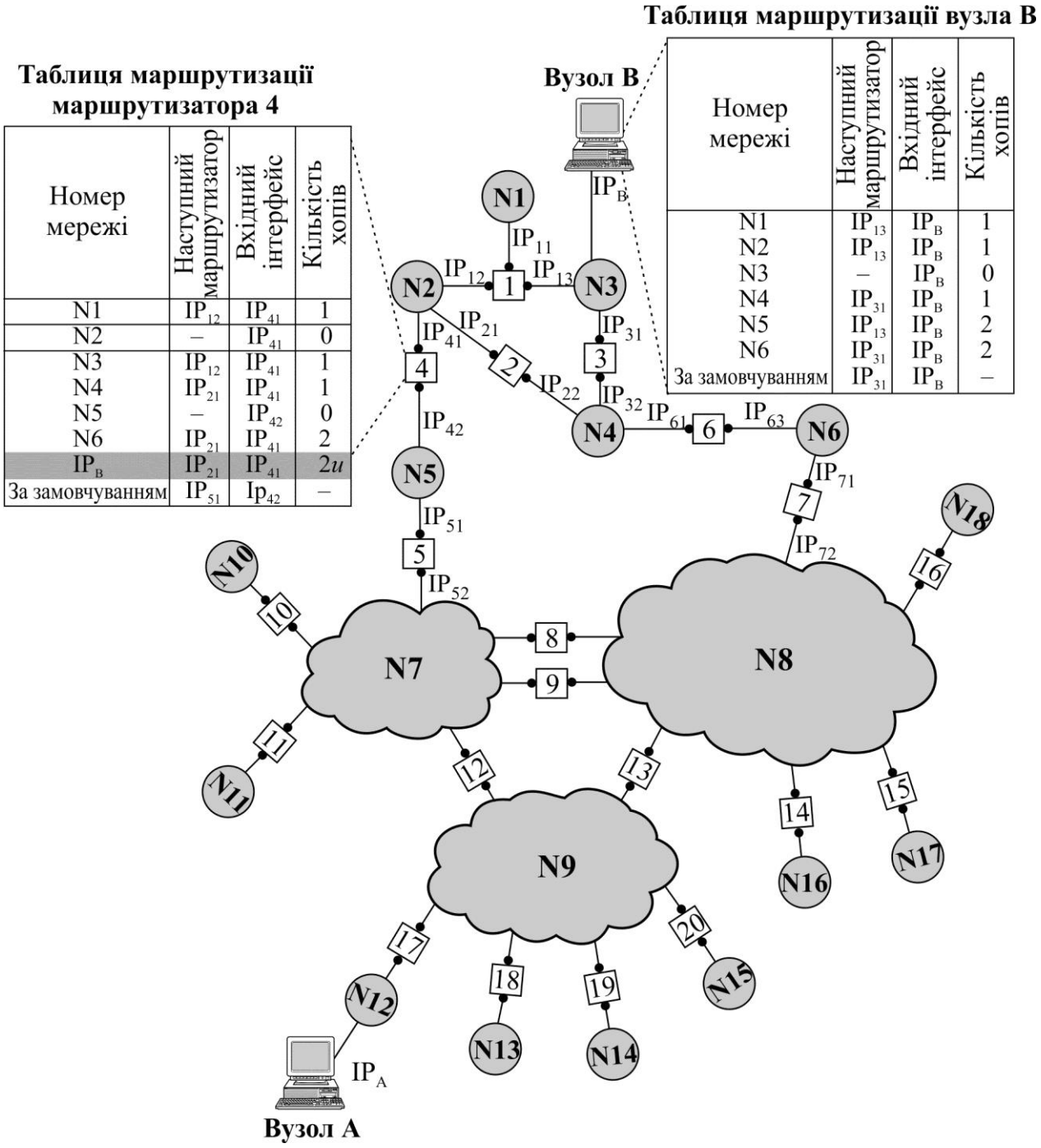


Рисунок 11.2 – Принципи маршрутизації у складеній мережі

Отже, маршрутизатор можна розглядати як сукупність декількох вузлів, кожний з яких входить у свою мережу. Як єдиний пристрій маршрутизатор не має виділеної адреси, ні мережної, ні локальної.

У складних складених мережах майже завжди існують декілька альтернативних маршрутів для передачі пакетів між двома кінцевими вузлами.

Так, пакет, відправлений з вузла А у вузол В, може пройти через маршрутизатори 17, 12, 5, 4 та 1 або маршрутизатори 17, 13, 7, 6 і 3. Неважко знайти ще кілька маршрутів між А и В.

ПРИМІТКА. *За наявності у маршрутизатора блоку керування (наприклад, згідно з протоколом SNMP) цей блок має власні локальну та мережну адреси, за якими до нього звертається центральна станція керування. Ці адреси обираються з того ж пулу, що й адреси фізичних інтерфейсів маршрутизатора. У технічній документації такого роду адреси називаються адресами зворотної петлі (looopback address), або адресами віртуальних інтерфейсів (virtual interface address). На відміну від адрес* 127.x.x.x*, зарезервованих для передачі даних між програмними компонентами, що перебувають у межах одного комп'ютера, адреси віртуальних інтерфейсів припускають звертання до них ззовні.*

Завдання вибору маршруту з декількох можливих вирішують маршрутизатори, а також кінцеві вузли. Маршрут обирається на підставі наявної у цьому устаткуванні інформації про поточну конфігурацію мережі, а також на підставі критерію вибору маршруту. Критерієм часто виступає затримка проходження маршруту окремим пакетом, середня пропускна здатність маршруту для послідовності пакетів або найпростіший критерій, який враховує лише кількість пройдених на маршруті проміжних маршрутизаторів (ретрансляційних ділянок).

Отримана в результаті аналізу інформація про маршрути подальшого проходження пакетів вміщуються в таблицю маршрутизації.

11.3 Спрощена таблиця маршрутизації

Використовуючи умовні позначки для мережних адрес маршрутизаторів і номерів мереж, показані на рис. 11.2, подивимося, як могла б виглядати таблиця маршрутизації, наприклад, у маршрутизаторі 4 (табл. 11.1).

Перший стовпчик таблиці містить адреси призначення пакетів.

У кожному рядку таблиці за адресою призначення вказується мережна адреса наступного маршрутизатора (точніше, мережна адреса інтерфейсу наступного маршрутизатора), на яку потрібно направити пакет, щоб той пересувався в напрямку до заданої адреси раціональним маршрутом.

Перед тим як передати пакет наступному маршрутизатору, поточний маршрутизатор має визначити, на який з декількох власних портів (IP_{41} або IP_{42}) він має помістити даний пакет. Для цього слугує третій стовпчик таблиці маршрутизації, який містить мережні адреси вихідних інтерфейсів.

Таблиця 11.1 – Таблиця маршрутизації маршрутизатора 4

Адреса призначення	Мережна адреса наступного маршрутизатора	Мережна адреса вихідного порту	Відстань до мережі призначення
N1	IP_{12}(R1)	IP_{41}	1
N2	–	IP_{41}	0 (приєднана)
N3	IP_{12}(R1)	IP_{41}	1
N4	IP_{21}(R2)	IP_{41}	1
N5	–	IP_{42}	0 (приєднана)
N6	IP_{21}(R2)	IP_{21}	2
IPB	IP_{21}(R2)	IP_{41}	2
Маршрут за замовчуванням	IP_{51}(R5)	IP_{42}	–

***ПРИМІТКА**. Таблиця 11.1 значно спрощена порівняно з реальними таблицями, наприклад, тут відсутні стовпці з масками, ознаками стану маршруту, часом, протягом якого дійсні записи даної таблиці (їх застосування розглядатиметься пізніше). Замість номера мережі призначення може бути зазначена повна мережна адреса окремого вузла призначення. Крім того, як вже зазначалося, тут вказані адреси мереж умовного формату, які не відповідають якомусь певному мережному протоколу. Проте ця таблиця містить основні поля, наявні в реальних таблицях.*

Деякі реалізації мережних протоколів допускають наявність у таблиці маршрутизації відразу декількох рядків, які відповідають одній й тій самій адресі призначення. У цьому випадку під час вибору маршруту береться до уваги стовпчик, що становить відстань до мережі призначення. У цьому разі відстань вимірюється в будь-якій метриці, що використовується відповідно до заданого в мережному пакеті критерію. Відстань може вимірюватися часом проходження пакета по лініях зв’язку, різними характеристиками надійності ліній зв’язку на даному маршруті, пропускною здатністю або іншою величиною, що відбиває якість даного маршруту стосовно заданого критерію.

У табл. 11.1 відстань між мережами вимірюється хопами. Відстань для мереж, безпосередньо підключених до портів маршрутизатора, тут приймається такою, що дорівнює 0, але у деяких реалізаціях відлік відстаней починається з 1.

Коли пакет надходить на маршрутизатор, модуль IP вилучає з його заголовка номер мережі призначення й послідовно порівнює його з номерами мереж з кожного рядка таблиці. Рядок з номером мережі, що збігається, показує найближчий маршрутизатор, на який слід направити пакет. Наприклад, якщо на будь-який порт маршрутизатора 4 надходить пакет, адресований у мережу N6, то з таблиці маршрутизації випливає, що адреса наступного маршрутизатора – IP_{21}, тобто черговим етапом руху даного пакета буде рух до порту 1 маршрутизатора 2.

Найчастіше як адреса призначення у таблиці вказується не вся IP-адреса, а лише номер мережі призначення. Отже, для всіх пакетів, що направляються в одну й ту саму мережу, протокол IP пропонуватиме той самий маршрут (ми поки не беремо до уваги можливі зміни стану мережі, такі як відмови маршрутизаторів або обриви кабелів). Але у деяких випадках виникає необхідність для одного з вузлів мережі визначити специфічний маршрут, який відрізняється від маршруту, заданого для усіх інших вузлів мережі. Для цього в таблицю маршрутизації поміщають для даного вузла окремий рядок, що містить його повну IP-адресу й відповідну маршрутну інформацію. Такого роду запис є в табл. 11.1 для вузла *B*. Нехай, наприклад, адміністратор маршрутизатора 4, керуючись міркуваннями безпеки, вирішив, що пакети, які прямують у вузол *B* (повна адреса IP_B), мають іти через маршрутизатор 2 (інтерфейс IP_{21}), а не маршрутизатор 1 (інтерфейс IP_{12}), через який передаються пакети всім іншим вузлам мережі N3. Якщо в таблиці є записи про маршрути як до мережі в цілому, так і до її окремого вузла, то з надходженням пакета, адресованого даному вузлу, маршрутизатор надасть перевагу специфічному маршруту. Оскільки пакет може бути адресований у будь-яку мережу складеної мережі, може здатися, що кожна таблиця маршрутизації повинна мати записи про всі мережі, що входять у складену мережу. Однак при такому підході у випадку великої мережі обсяг таблиць маршрутизації може виявитися дуже великим, що вплине на час її перегляду, потребуватиме багато місця для зберігання тощо. Тому на практиці широко відомий прийом зменшення кількості записів у таблиці маршрутизації, заснований на введенні маршруту за замовчуванням (default route), який враховує особливості топології мережі. Розглянемо, наприклад, маршрутизатори, що перебувають на периферії складеної мережі. У їхніх таблицях достатньо записати номери тільки тих

мереж, які безпосередньо приєднані до даного маршрутизатора або розташовані поблизу на тупикових маршрутах. Про всі інші мережі можна зробити у таблиці єдиний запис, що вказує на маршрутизатор, через який пролягає шлях до всіх цих мереж. Такий маршрутизатор називається маршрутизатором за замовчуванням (default router). У нашому прикладі на маршрутизаторі 4 є специфічні маршрути лише для пакетів, що прямують у мережі N1–N6. Для усіх інших пакетів, адресованих у мережі N7–N18, маршрутизатор пропонує продовжити шлях через той самий порт IP_{51} маршрутизатора 5, який у цьому випадку і є маршрутизатором за замовчуванням.

11.4 Таблиці маршрутизації кінцевих вузлів

Завдання маршрутизації вирішують не лише проміжні вузли (маршрутизатори), але й кінцеві вузли – комп'ютери. Розв'язок цієї задачі починається з того, що засобами протоколу IP на кінцевому вузлі визначається, направлений пакет в іншу мережу чи адресований якомусь вузлу даної мережі. Якщо номер мережі призначення збігається з номером даної мережі, це означає, що пакет маршрутизувати не потрібно. А якщо ні, то маршрутизація потрібна.

Структури таблиць маршрутизації кінцевих вузлів і транзитних маршрутизаторів аналогічні. Звернемося знову до мережі, зображеної на рис. 11.2. Таблиця маршрутизації кінцевого вузла *B*, що належить мережі N3, могла б виглядати як табл. 11.2. Тут IP_B – мережна адреса інтерфейсу комп'ютера *B*. На підставі цієї таблиці кінцевий вузол *B* обирає, на який з двох наявних у локальній мережі N3 маршрутизаторів (R1 або R3) слід посилати той чи інший пакет.

Таблиця 11.2 – Таблиця маршрутизації кінцевого вузла *B*

Номер мережі призначення	Мережна адреса наступного маршрутизатора	Мережна адреса вихідного порту	Відстань до мережі призначення
N1	IP_{13}(R1)	IP_B	1
N2	IP_{13}(R1)	IP_B	1
N3	–	IP_B	0
N4	IP_{31}(R3)	IP_B	1
N5	IP_{13}(R1)	IP_B	2
N6	IP_{31}(R3)	IP_B	2
Маршрут за замовчуванням	IP_{31}(R3)	IP_B	–

Кінцеві вузли у ще більше, ніж маршрутизатори, користуються прийомом маршрутизації за замовчуванням. Хоча вони також у загальному випадку мають у своєму розпорядженні таблицю маршрутизації, її обсяг зазвичай незначний, що пояснюється периферійним розташуванням усіх кінцевих вузлів. Кінцевий вузол часто взагалі працює без таблиці маршрутизації, маючи лише відомості про адресу маршрутизатора за замовчуванням. За наявності одного маршрутизатора у локальній мережі цей варіант – єдиний можливий для усіх кінцевих вузлів. Але навіть за наявності декількох маршрутизаторів у локальній мережі, коли перед кінцевим вузлом стоїть проблема їх вибору, часто у комп'ютерах для підвищення продуктивності звертаються до завдання маршрута за замовчуванням.

Розглянемо таблицю маршрутизації іншого кінцевого вузла складеної мережі – вузла *A* (табл. 11.3). Компактний вигляд таблиці маршрутизації вузла *A* відбиває той факт, що усі пакети, які направляються з вузла *A*, або не виходять за межі мережі N12, або неодмінно проходять через порт 1 маршрутизатора 17. Цей маршрутизатор і визначений у таблиці маршрутизації як маршрутизатор за замовчуванням.

Таблиця 11.3 – Таблиця маршрутизації кінцевого вузла *A*

Номер мережі призначення	Мережна адреса наступного маршрутизатора	Мережна адреса вихідного порту	Відстань до мережі призначення
N12	–	IP_A	0
Маршрут за замовчуванням	$IP_{17,1}$(R17)	IP_A	–

Ще однією відмінністю роботи маршрутизатора й кінцевого вузла є спосіб побудови таблиці маршрутизації. Якщо маршрутизатори, як правило, автоматично створюють таблиці маршрутизації, обмінюючись службовою інформацією, то для кінцевих вузлів таблиці маршрутизації часто створюються вручну адміністраторами й зберігаються у вигляді постійних файлів на дисках.

Лекція 12
МАРШРУТИЗАЦІЯ З ВИКОРИСТАННЯМ МАСКИ

12.1 Приклад IP-маршрутизації без маски

Перегляд таблиць маршрутизації без масок

Розглянемо алгоритм перегляду таблиці маршрутизації, реалізований на маршрутизаторі протоколом IP. В процесі його опису ми використовуватимемо табл. 11.1 і рис. 11.2.

Таблиця 11.1 – Таблиця маршрутизації маршрутизатора 4

Адреса призначення	Мережна адреса наступного маршрутизатора	Мережна адреса вихідного порту	Відстань до мережі призначення
N1	IP_{12}(R1)	IP_{41}	1
N2	–	IP_{41}	0 (приєднана)
N3	IP_{12}(R1)	IP_{41}	1
N4	IP_{21}(R2)	IP_{41}	1
N5	–	IP_{42}	0 (приєднана)
N6	IP_{21}(R2)	IP_{21}	2
IPB	IP_{21}(R2)	IP_{41}	2
Маршрут за замовчуванням	IP_{51}(R5)	IP_{42}	–

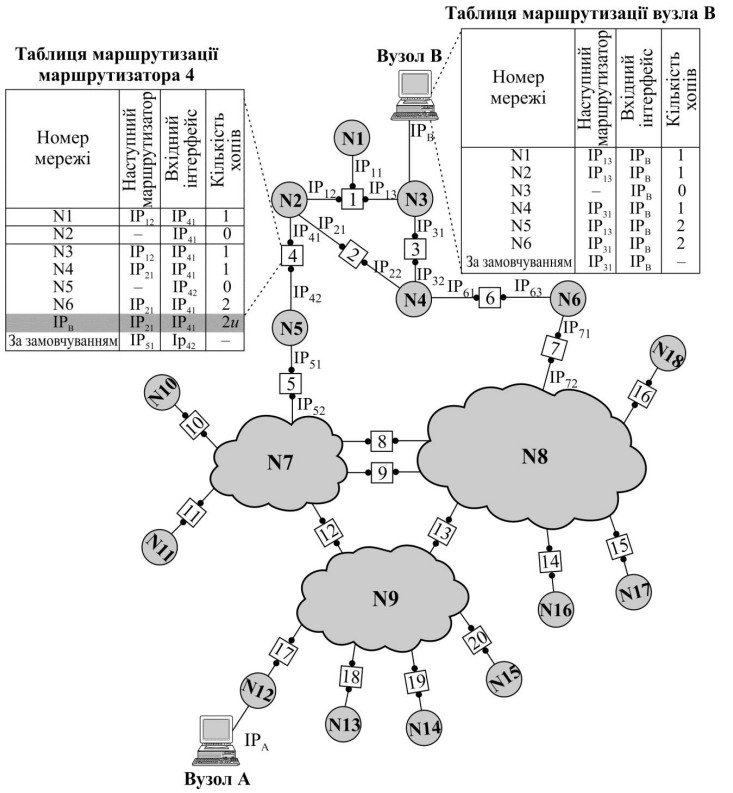


Рисунок 11.2 – Принципи маршрутизації у складеній мережі

1. Нехай на один з інтерфейсів маршрутизатора надходить пакет. Протокол IP виликає з пакета IP-адресу призначення (припустимо, адресу призначення IP_B).

2. Виконується перша фаза перегляду таблиці – пошук конкретного маршруту до вузла. IP-адреса (повністю) послідовно рядок за рядком порівнюється із вмістом поля адреси призначення таблиці маршрутизації. Якщо відбувся збіг (як у табл. 11.1), то з відповідного рядка вилучаються адреса наступного маршрутизатора (IP_{21}) та ідентифікатор вихідного інтерфейса (IP_{41}). На цьому перегляд таблиці закінчується.

3. Припустимо тепер, що в таблиці немає рядка з адресою призначення IP_B, – тобто збігу не відбулося. У цьому випадку протокол IP переходить до другої фази перегляду – пошуку маршруту до мережі призначення. З IP-адреси виділяється номер мережі (у нашому прикладі з адреси IP_B виділяється номер мережі N3), й таблиця знову проглядається на предмет збігу номера мережі в якомусь рядку з номером мережі з пакета. У разі збігу (у нашому прикладі він відбувся) з відповідного рядка таблиці вилучаються адреса наступного маршрутизатора (IP_{12}) та ідентифікатор вихідного інтерфейса (IP_{41}). Перегляд таблиці на цьому завершується.

4. Нарешті, припустимо, що адреса призначення в пакеті була така, що збігу не відбулося ні в першій, ні у другій фазах перегляду. У такому випадку засобами протоколу IP або обирається маршрут за замовчуванням (і пакет прямує за адресою IP_{51}), або, якщо маршрут за замовчуванням відсутній, пакет відкидається. Перегляд таблиці на цьому закінчується.

> ***УВАГА***. *Послідовність фаз у даному алгоритмі чітко визначена, у той час як послідовність перегляду або, що те саме, порядок розташування рядків у таблиці, включаючи запис про маршрут за замовчуванням, ніяк не позначається на результаті.*

Джерела й типи записів у таблиці маршрутизації

Практично для усіх маршрутизаторів існують три основні джерела записів у таблиці.

Одним із джерел записів у таблиці маршрутизації є програмне забезпечення стека TCP/IP, яке під час ініціалізації маршрутизатора автоматично заносить до таблиці кілька записів, у результаті чого створюється так звана мінімальна таблиця маршрутизації. Програмне забезпечення формує записи про безпосередньо підключені мережі й маршрути за замовчуванням, інформація про які з’являється в стеку під час ручного конфігурування

інтерфейсів комп'ютера або маршрутизатора. До таких записів у наведених прикладах належать записи про мережі 213.34.12.0 та 198.21.17.0, а також запис про маршрут за замовчуванням у маршрутизаторі Unix і запис 0.0.0.0 у маршрутизаторі ОС Windows XP. Крім того, програмне забезпечення автоматично заносить до таблиці маршрутизації записи про адреси особливого призначення. У наведених прикладах таблиця маршрутизатора ОС Windows 2000 містить найповніший набір записів такого роду. Декілька записів у цій таблиці пов'язані з особливою адресою 127.0.0.0. Записи з адресою 224.0.0.0 потрібні для опрацювання групових адрес. Крім того, до таблиці можуть бути занесені адреси, призначені для опрацювання широкомовних розсилань (наприклад, записи 8 і 11 містять адреси відправлення широкомовного повідомлення у відповідних підмережах, а останній запис у таблиці – адреса обмеженого широкомовного розсилання). Зазначимо, що у деяких таблицях записи про особливі адреси взагалі відсутні.

Ще одним джерелом записів у таблиці є адміністратор, який безпосередньо формує записи за допомогою деякої системної утиліти, наприклад, програми route, що наявна в операційних системах Unix і Windows XP. В апаратних маршрутизаторах також завжди є команда для ручного задання записів таблиці маршрутизації. Задані вручну записи завжди є статичними, тобто вони не мають терміну життя. Ці записи можуть бути як постійними, тобто такими, що зберігаються у процесі перезавантаження маршрутизатора, так і часовими, що зберігаються у таблиці лише до вимикання обладнання. Часто адміністратор вручну затягає запис про маршрут за замовчуванням. У такий же спосіб у таблицю маршрутизації може бути внесений запис про специфічний для вузла маршрут.

І, нарешті, третім джерелом записів можуть бути протоколи маршрутизації, такі як RIP або OSPF. Ці записи завжди є динамічними, тобто мають обмежений термін життя.

Програмні маршрутизатори Windows XP та Unix не показують джерело появи того чи іншого запису в таблиці, а апаратний маршрутизатор використовує для цієї мети поле джерела.

Приклад IP-маршрутизації без масок

Розглянемо процес просування пакета у складеній мережі на прикладі IP-мережі, показаної на рис. 12.1. При цьому вважатимемо, що всі вузли мережі, розглянутої у прикладі, мають адреси, засновані на класах. Особлива увага буде приділена взаємодії протоколу IP із протоколами дозволу адрес ARP і DNS.

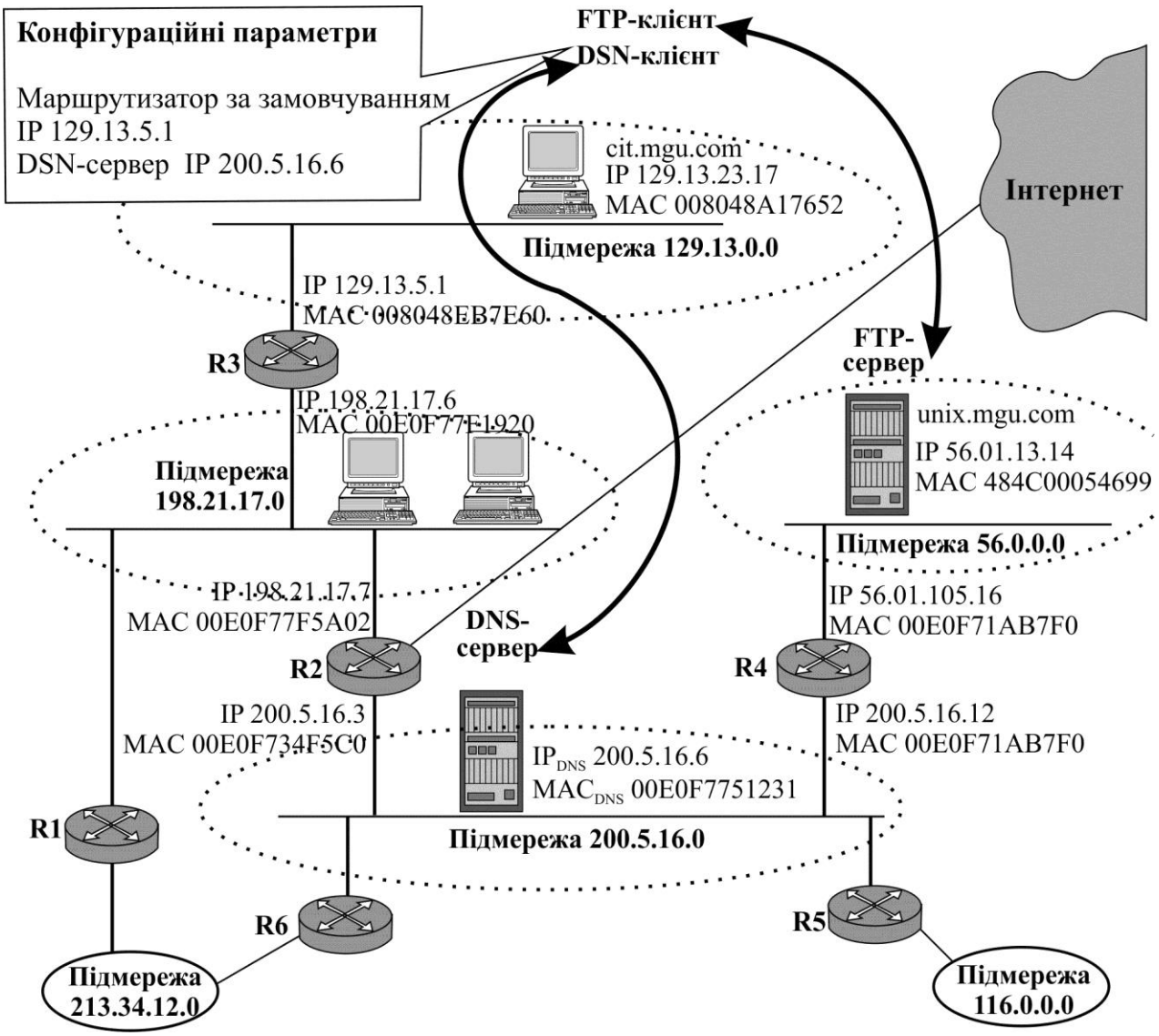


Рисунок 12.1 – Приклад IP-маршрутизації

Отже, нехай користувачу комп’ютера cit.mgu.com, що знаходиться у мережі 129.13.0.0, необхідно встановити зв’язок з FTP-сервером. Користувачу відоме символьне ім’я сервера unix.mgu.com, тому він набирає на клавіатурі команду звернення до FTP-сервера по імені:

> ftp unix.mgu.com

Виконання цієї команди ініціює три послідовні операції:

1. DNS-клієнт (що працюює на комп’ютері cit.mgu.com) передає DNS-серверу повідомлення, у якому міститься запит про IP-адресу сервера unix.mgu.com, з яким він хоче зв’язатися за протоколом FTP.

2. DNS-сервер, виконавши пошук, передає відповідь DNS-клієнту про найдену IP-адресу сервера unix.mgu.com.

3. FTP-клієнт (що працюює на тому самому комп’ютері cit.mgu.com), використовуючи знайдену IP-адресу сервера unix.mgu.com, передає повідомлення FTP-сервера, що працює на ньому.

Давайте послідовно, крок за кроком, розглянемо, як у процесі розв'язання цих задач взаємодіють між собою протоколи DNS, IP, ARP і Ethernet, і що відбувається при цьому з кадрами й пакетами.

1. Формування IP-пакета з інкапсульованим у нього DNS-запитом. Програмний модуль FTP-клієнта, отримавши команду > ftp unix.mgu.com, передає запит до працюючої на цьому самому комп'ютері клієнтської частини протоколу DNS, яка, у свою чергу, формує до DNS-сервера запит, який інтерпретується, приблизно так: «Яка IP-адреса відповідає символьному імені unix.mgu.com?» Запит упаковується в UDP-дейтаграму, потім в IP-пакет. У заголовку пакета як адреса призначення вказується IP-адреса 200.5.16.6 DNS-сервера. Ця адреса відома програмному забезпеченню клієнтського комп'ютера, оскільки він входить до множини його конфігураційних параметрів. Сформований IP-пакет переміщуватиметься по мережі в незмінному вигляді (як показано на рис. 12.2), доки не дійде до адресата – DNS-сервера.

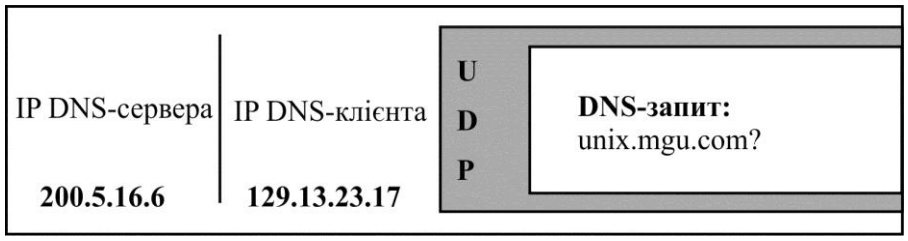


Рисунок 12.2 – IP-пакет з інкапсульованим у нього DNS-запитом

2. Передача кадра Ethernet з IP-пакетом маршрутизатору R3. Для передачі даного IP-пакета необхідно його упакувати у кадр Ethernet, вказавши у заголовку MAC-адресу отримувача. Технологія Ethernet здатна доставляти кадри лише тим адресатам, які знаходяться з відправником у межах однієї підмережі. Якщо адресат розташований поза цією підмережею, то кадр потрібно передати найближчому маршрутизатору, щоб той взяв на себе турботу про подальше переміщення пакета. Для цього модуль IP, порівнявши номери мереж в адресах відправника та отримувача, тобто 129.13.23.17 й 200.5.16.6, з'ясовує, що пакет направляється в іншу мережу, отже, його необхідно передати маршрутизатору, у даному випадку маршрутизатору за замовчуванням. IP-адреса маршрутизатора за замовчуванням також відома клієнтському вузлу, оскільки він належить до множини конфігураційних параметрів. Але для кадру Ethernet необхідно вказати не IP-адресу, а MAC-адресу отримувача. Дана проблема розв'язується за допомогою протоколу ARP, який для відповіді на запитання: «Яка MAC-адреса відповідає IP-адресі 194.87.23.1?» – робить пошук у своїй ARP-таблиці. Оскільки звернення до маршрутизатора відбуваються часто, вважатимемо, що потрібна

MAC-адреса знаходиться у таблиці й має значення 008048EB7E60. Після отримання цієї інформації клієнтський комп'ютер cit.mgu.com відправляє маршрутизатору R3 пакет, упакований у кадр Ethernet (рис. 12.3).

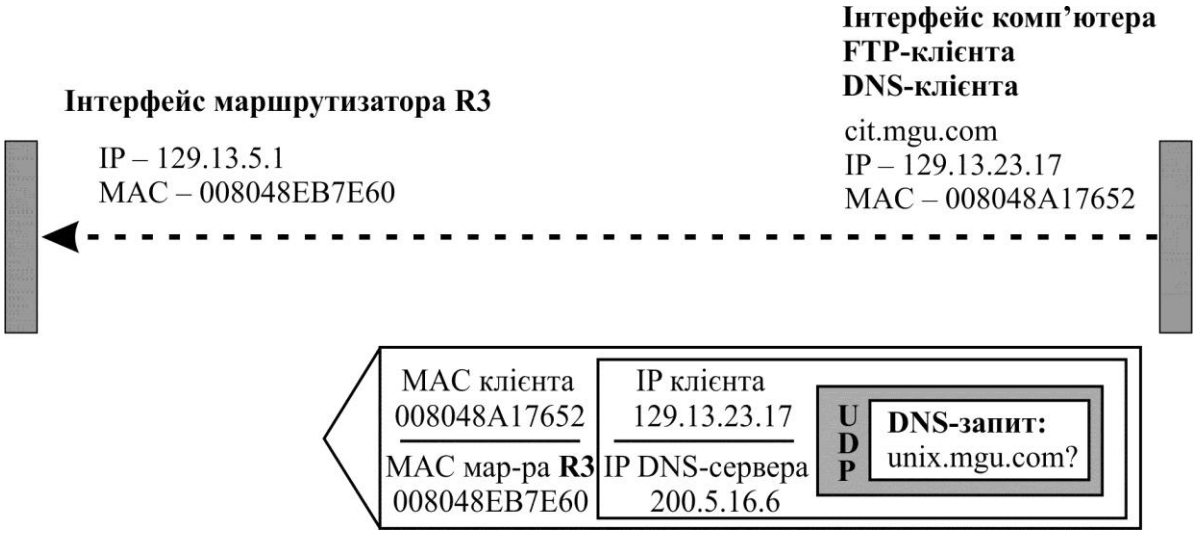


Рисунок 12.3 – Кадр Ethernet з інкапсульованим IP-пакетом, відправлений з клієнтського комп'ютера

3. Визначення IP-адреси і MAC-адреси наступного маршрутизатора R2. Кадр приймається інтерфейсом 129.13.5.1 маршрутизатора R3. Протокол Ethernet, який працює на даному інтерфейсі, вилучає із цього кадра IP-пакет і передає його протоколу IP. Протокол IP виявляє у заголовку пакета адресу призначення 200.5.16.6 і переглядає записи своєї таблиці маршрутизації. Припустимо, маршрутизатор R3 не виявляє специфічного маршруту для адреси призначення 200.5.16.6, але знаходить у своїй таблиці такий запис:

200.5.16.0 198.21.17.7 198.21.17.6

Цей запис говорить про те, що пакети для мережі 200.5.16.0 маршрутизатор R3 має передавати на свій вихідний інтерфейс 198.21.17.6, з якого вони надійдуть на інтерфейс наступного маршрутизатора R2, що має IP-адресу 198.21.17.7. Але знання IP-адреси недостатньо, щоб передати пакет по мережі Ethernet. Необхідно визначити MAC-адресу маршрутизатора R3. Як відомо, таку роботу виконує протокол ARP. Припустимо, цього разу в ARP-таблиці немає запису про адресу маршрутизатора R3. Тоді в мережу відправляється широкомовний ARP-запит, який надходить на всі інтерфейси мережі 198.21.17.0. Відповідь приходить тільки від інтерфейса маршрутизатора R3: «Я маю IP-адресу 198.21.17.7 і моя MAC-адреса 00E0F77F5A02» (рис. 12.4).

Тепер, знаючи MAC-адресу маршрутизатора R2 (00E0F77F5A02), маршрутизатор R3 відсилає йому IP-пакет з DNS-запитом (рис. 12.5).

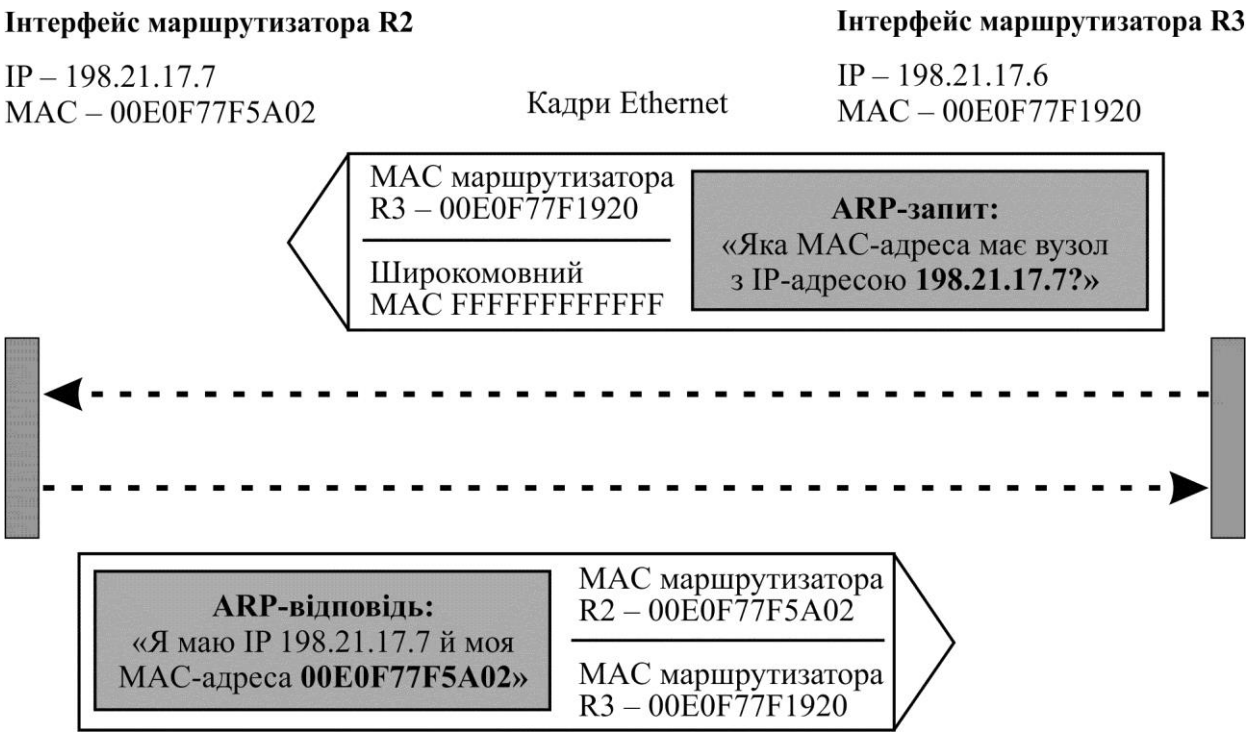


Рисунок 12.4 – Кадри Ethernet з інкапсульованими ARP-запитом і ARP-відповіддю



Рисунок 12.5 – Кадр Ethernet з DNS-запитом, відправлений з маршрутизатора R3 маршрутизатору R2

4. Маршрутизатор R2 доставляє пакет DNS-сервера. Модуль IP на маршрутизаторі R2 діє відповідно до вже не раз описаної нами процедури: відкинувши заголовок кадра Ethernet, він вилучає з пакета IP-адресу призначення та переглядає свою таблицю маршрутизації. Там він виявляє, що мережа призначення 200.5.16.0 є безпосередньо приєднаною до його другого інтерфейсу. Отже, пакет не треба маршрутизувати, але потрібно визначити MAC-адресу вузла призначення. Протокол ARP «на прохання» протоколу IP знаходить (або з ARP-таблиці, або за запитом) необхідну MAC-адресу 00E0F7751231 DNS-сервера. Отримавши відповідь про MAC-адресу,

маршрутизатор R2 відправляє в мережу призначення кадр Ethernet з DNS-запитом (рис. 12.6).

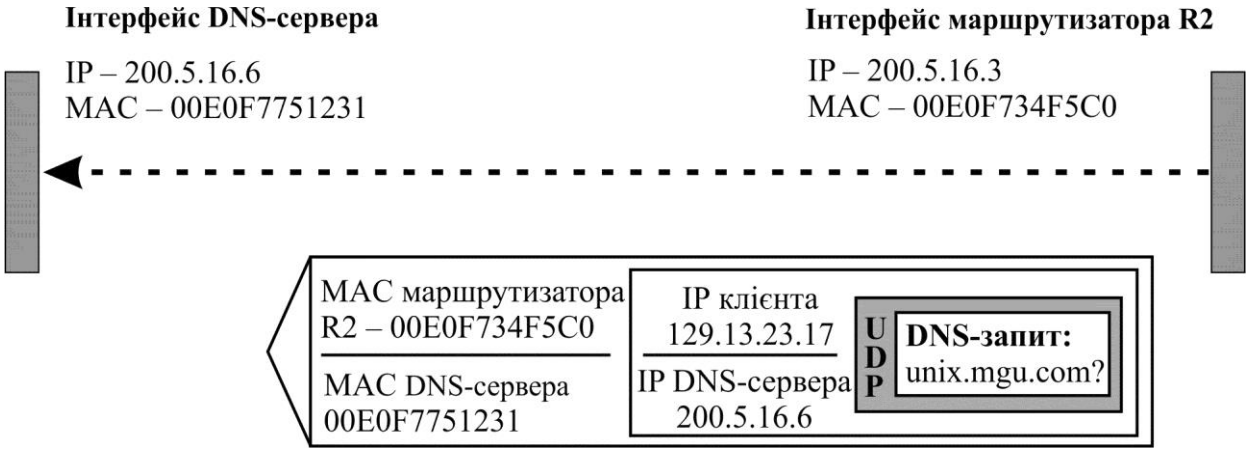


Рисунок 12.6 – Кадр Ethernet з DNS-запитом,
відправлений з маршрутизатора R2

5. Мережний адаптер DNS-сервера захоплює кадр Ethernet, виявляє збіг MAC-адреси призначення, який міститься у заголовку, зі своєю власною адресою й надсилає його модулю IP. Після аналізу полів заголовка IP з пакета вилучаються дані протоколів, розташованих вище. DNS-запит передається програмному модулю DNS-сервера. DNS-сервер переглядає свої таблиці, можливо, звертається до інших DNS-серверів і в результаті формує відповідь, з таким змістом: «Символьному імені unix.mgu.com відповідає IP-адреса 56.01.13.14»

Процес доставки DNS-відповіді клієнту cit.mgu.com цілковито аналогічний процесу передачі DNS-запиту, який ми щойно так докладно описали. Працюючи в тісному взаємозв’язку, протоколи IP, ARP і Ethernet передають клієнту DNS-відповідь через усю складену мережу (рис. 12.7).

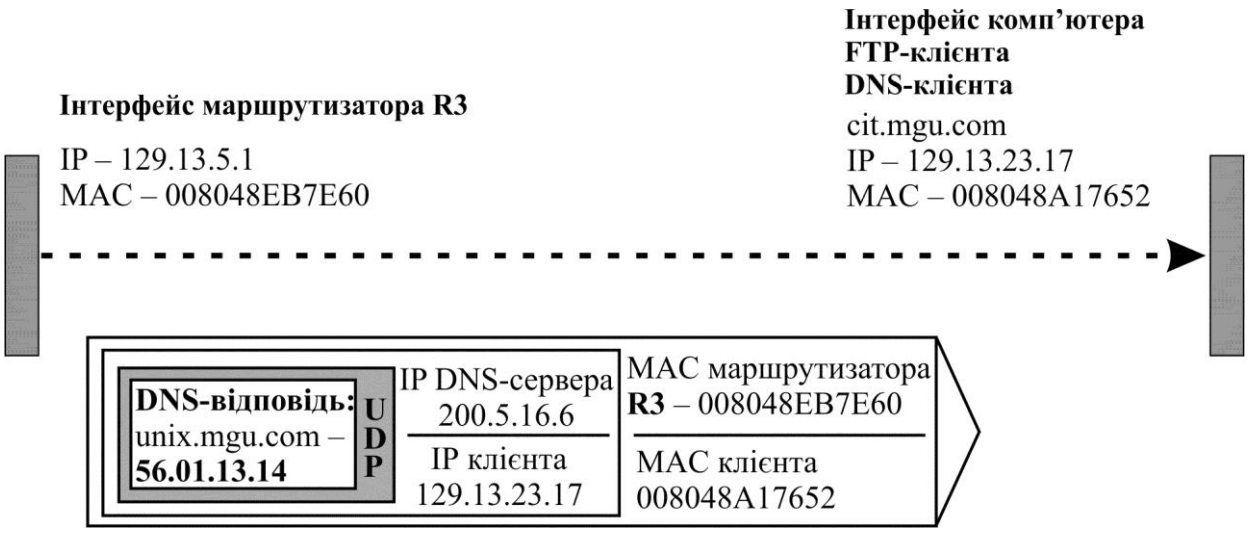


Рисунок 12.7 – Кадр Ethernet з DNS-відповіддю,
відправлений з маршрутизатора R3 комп’ютеру-клієнту

FTP-клієнт, отримавши IP-адресу FTP-сервера, посилає йому своє повідомлення, використовуючи ті самі, описані раніше, механізми доставки даних через складену мережу. Але буде досить корисно подумки відтворити цей процес, звертаючи особливу увагу на значення адресних полів заголовків кадрів і заголовка вкладеного IP-пакета.

> ***ПРИМІТКА.*** *Зазначимо, що під час усієї подорожі пакета складеною мережею від клієнтського комп'ютера до DNS-сервера IP-адреси отримувача та відправника у полях заголовка IP-пакета не змінюються. Проте у заголовку кожного нового кадра, що переносив пакет від одного маршрутизатора до іншого, MAC-адреси відправника та отримувача змінюються на кожному відрізку шляху.*

12.2 Визначення маршрутизації з використанням маски

Алгоритм маршрутизації ускладнюється, коли в систему адресації вузлів вносяться додаткові елементи – маски. У чому причини відмови від методу адресації, заснованого на класах, який добре себе зарекомендував протягом багатьох років? Основна з них – потреба в структуризації мереж в умовах дефіциту нерозподілених номерів мереж.

Часто адміністратори мереж зазнають незручностей, оскільки кількість централізовано виділених їм номерів мереж недостатня для того, щоб структурувати мережу належним чином, наприклад, розвести всі слабко взаємодіючі комп'ютери по різних мережах.

У такій ситуації можливі два шляхи. *Перший* з них пов'язаний з отриманням від будь-якого центрального органа додаткових номерів мереж. *Другий* спосіб, що застосовується частіше, пов'язаний з використанням технології масок, яка дозволяє розділити одну мережу на декілька.

12.3 Структуризація мережі масками однакової довжини

Припустимо, адміністратор отримав у своє розпорядження мережу класу *B*: 129.44.0.0.

Він може організувати мережу з більшою кількістю вузлів, номери яких доступні йому з діапазону 0.0.0.1–0.0.255.254. Усього в його розпорядженні є $\left(2^{16}-2\right)$ адреси. Віднімання двійки пов'язано з урахуванням того, що адреси з одних нулів і одних одиниць мають спеціальне призначення й не підходять для

адресації вузлів. Але йому не потрібна одна велика неструктурована мережа. Виробнича необхідність вимагає у адміністратора іншого рішення, відповідно до якого мережа має бути розділена на три окремі підмережі, при цьому трафік у кожній підмережі має бути надійно локалізований. Це дозволить легше діагностувати мережу й проводити у кожній з підмереж особливу політику безпеки. (Зазначимо, що поділ великої мережі за допомогою масок має ще одну перевагу, – він дозволяє приховати внутрішню структуру мережі підприємства від зовнішнього спостереження й тим самим підвищити її безпеку.)

На рис. 12.8 показано поділ усього отриманого адміністратором адресного діапазону на чотири рівні частини – кожна по 2^{14} адреси. При цьому кількість розрядів, доступна для нумерації вузлів, зменшилася на два біти, а префікс (номер) кожної з чотирьох мереж став довшим на два біти.

Отже, кожний із чотирьох діапазонів можна записати у вигляді IP-адреси з маскою, яка складається з 18 одиниць, або в десятковій нотації – 255.255.192.0.

129.44.0.0/18 (10000001 00101100 **00**000000 00000000)
129.44.64.0/18 (10000001 00101100 **01**000000 00000000)
129.44.128.0/18 (10000001 00101100 **10**000000 00000000)
129.44.192.0/18 (10000001 00101100 **11**000000 00000000)

З наведених записів видно, що адміністратор отримує можливість використовувати для нумерації підмереж два додаткових біти (виділених жирним шрифтом). Саме це дозволяє йому зробити з однієї централізовано виділеної мережі чотири, у даному прикладі це

129.44.0.0/18, 129.44.64.0/18, 129.44.128.0/18, 129.44.192.0/18.

ПРИМІТКА. *Деякі програмні й апаратні маршрутизатори, дотримуючись застарілих рекомендацій RFC 950,* ***не підтримують номери підмереж, які складаються або лише з одних нулів, або лише з одних одиниць***. *Наприклад, для такого типу устаткування номер мережі 129.44.0.0 з маскою 255.255.192.0, що використовується у нашому прикладі, виявиться неприпустимим, оскільки у даному випадку розряди в полі номера підмережі мають значення 00. З аналогічних міркувань неприпустимим може виявитися номер мережі 129.44.192.0 з тим же значенням маски. Тут номер підмережі складається лише з одиниць. Але сучасні маршрутизатори, що підтримують рекомендації RFC 1878, вільні від цих обмежень.*



Рисунок 12.8 – Поділ адресного простору 129.44.0.0 мережі класу *B* на чотири рівні частини

Приклад мережі, побудованої шляхом поділу на чотири мережі однакового розміру, показаний на рис. 12.9. Увесь трафік у внутрішню мережу 129.44.0.0, що направляється із зовнішньої мережі, надходить через маршрутизатор R1. З метою структуризації інформаційних потоків у внутрішній мережі встановлений додатковий маршрутизатор R2. Кожна із новостворених мереж 129.44.0.0/18, 129.44.64.0/18, 129.44.128.0/18 та 129.44.192.0/18 підключена до відповідно сконфігурованих портів внутрішнього маршрутизатора R2.

ПРИМІТКА. *В одній із цих мереж (129.44.192.0/18), що виділена для організації з'єднання між зовнішнім і внутрішнім маршрутизаторами, для адресації вузлів задіяні всього дві адреси – 129.44.192.1 (порт маршрутизатора R2) і 129.44.192.2 (порт маршрутизатора R1). Величезна кількість вузлів у даній підмережі не використовується. Такий приклад обрано винятково з навчальною метою, щоб показати неефективність мереж однакового розміру.*

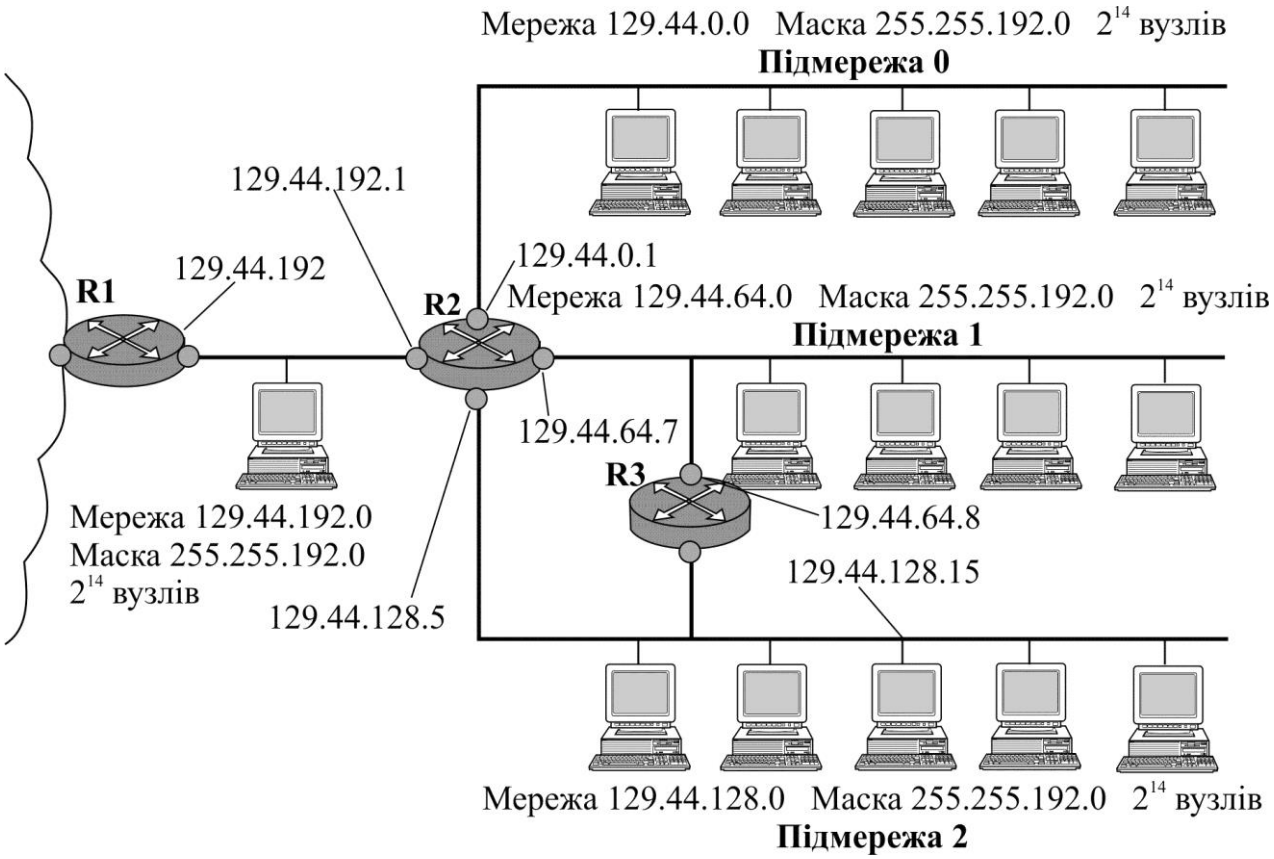


Рисунок 12.9 – Маршрутизація з використанням масок однакової довжини

Ззовні мережа як і раніше виглядає як єдина мережа класу *B*. Але загальний трафік, що надходить у мережу, поділяється локальним маршрутизатором R2 між чотирма мережами. В умовах, коли механізм класів не діє, маршрутизатор повинен мати інший засіб, що дозволив би йому визначати, яка частина 32-розрядного числа, поміщеного в поле адреси призначення, є номером мережі. Саме цій меті слугує додаткове поле маски, включене в таблицю маршрутизації (табл. 12.1).

Таблиця 12.1 – Таблиця маршрутизатора R2 у мережі з масками однакової довжини

Адреса призначення	Маска	Адреса наступного маршрутизатора	Адреса порту	Відстань
129.44.0.0	255.255.192.0	129.44.0.1	129.44.0.1	Підключена
129.44.64.0	255.255.192.0	129.44.64.7	129.44.64.7	Підключена
129.44.128.0	255.255.192.0	129.44.128.5	129.44.128.5	Підключена
129.44.192.0	255.255.192.0	129.44.192.1	129.44.192.1	Підключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–
129.44.128.15	255.255.255.255	129.44.64.8	129.44.64.7	–

Перші чотири записи у таблиці відповідають внутрішнім підмережам, безпосередньо підключеним до портів маршрутизатора R2.

Запис 0.0.0.0 з маскою 0.0.0.0 відповідає маршруту за замовчуванням.

Останній запис визначає специфічний маршрут до вузла 129.44.128.15. У тих рядках таблиці, де як адреси призначення зазначено повну IP-адресу вузла, маска має значення 255.255.255.255. На відміну від усіх інших вузлів мережі 129.44.128.0, до яких пакети надходять з інтерфейсу 129.44.128.5 маршрутизатора R2, до даного вузла вони мають приходити через маршрутизатор R3.

Лекція 13
IP-МАРШРУТИЗАЦІЇ З ФРАГМЕНТАЦІЄЮ ПАКЕТІВ

13.1 Перегляд таблиць маршрутизації з масками

Алгоритм перегляду таблиць маршрутизації, що містять маски, має багато загального з описаним алгоритмом перегляду таблиць, що не містять маски. Але у ньому є й істотні відмінності.

1. Пошук наступного маршрутизатора для IP-пакета, що надійшов, протокол починає з того, що вилучає з пакета адресу призначення (позначимо її IP_D). Потім протокол IP розпочинає процедуру перегляду таблиці маршрутизації, що також складається з двох фаз, як і процедура перегляду таблиці, у якій стовпчик маски відсутній.

2. Перша фаза полягає в пошуку специфічного маршруту для адреси IP_D. З цією метою з кожного запису таблиці, у якому маска має значення 255.255.255.255, вилучається адреса призначення й порівнюється з адресою з пакета IP_D. Якщо в будь-якому рядку збіг відбувся, то адреса наступного маршрутизатора для даного пакета береться з даного рядка.

3. Друга фаза виконується лише у тому випадку, якщо під час першої фази не відбувся збіг адрес. Вона полягає в пошуку неспецифічного маршруту, загального для групи вузлів, до якої належить і пакет з адресою IP_D. Для цього засобами IP заново проглядається таблиця маршрутизації, причому з кожним записом проводяться такі дії:

1) маска (позначимо її M), що міститься у даному записі, «накладається» на IP-адресу вузла призначення IP_D, вилучену з пакета: IPD AND M;

2) отримане в результаті число порівнюється зі значенням, яке поміщене в поле адреси призначення того самого запису таблиці маршрутизації;

3) якщо відбувається збіг, протокол IP відповідно позначає цей рядок;

4) якщо переглянуто не всі рядки, то протокол IP аналогічно переглядає наступний рядок, якщо всі (зокрема рядок про маршрут за замовчуванням), то перегляд записів закінчується і відбувається перехід до наступного кроку.

4. Після перегляду всієї таблиці маршрутизатор виконує одну з трьох дій:

1) якщо не відбулося жодного збігу й маршрут за замовчуванням відсутній, то пакет відкидається;

2) якщо відбувся один збіг, то пакет відправляється по маршруту, зазначеному в рядку з адресою, що збіглася;

3) якщо відбулося кілька збігів, то всі позначені рядки порівнюються й обирається маршрут з того рядка, у якому кількість двійкових розрядів, що збіглися, найбільша (інакше кажучи, у ситуації, коли адреса призначення пакета належить одночасно декільком підмережам, маршрутизатор використовує найбільш специфічний маршрут).

***ПРИМІТКА**. У багатьох таблицях маршрутизації запис з адресою 0.0.0.0 і маскою 0.0.0.0 відповідає маршруту за замовчуванням. Дійсно, будь-яка адреса в пакеті, що надійшов, після накладення на нього маски 0.0.0.0 дасть адресу мережі 0.0.0.0, що збігається з адресою, зазначеною у записі. Оскільки маска 0.0.0.0 має нульову довжину, то цей маршрут вважається найбільш неспецифічним і використовується лише за відсутності збігів з іншими записами з таблиці маршрутизації.*

Проілюструємо, як маршрутизатор R2 (рис. 13.1) використовує описаний алгоритм для роботи зі своєю таблицею маршрутизації (табл. 13.1).

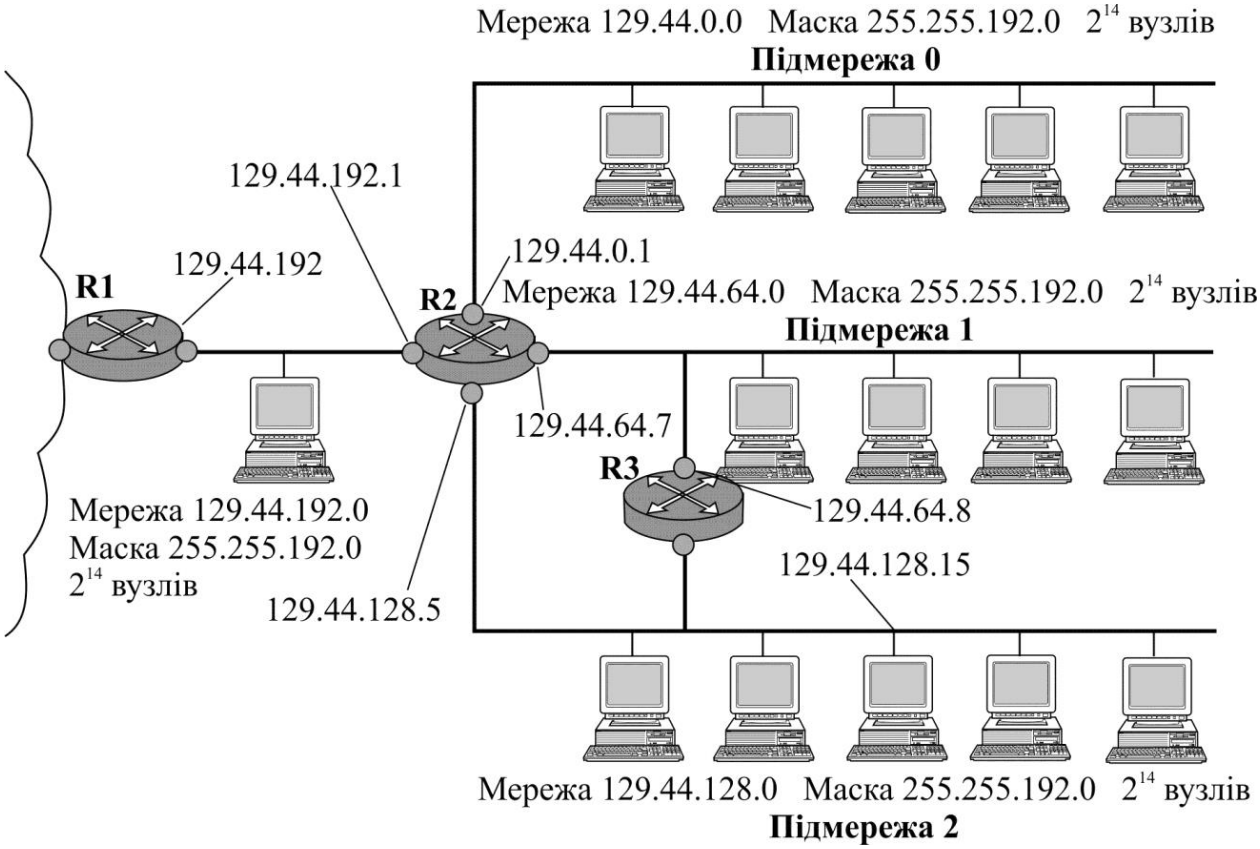


Рисунок 13.1 – Маршрутизація з використанням масок однакової довжини

Нехай на маршрутизатор R2 надходить пакет з адресою призначення 129.44.78.200. Модуль IP, установлений на даному маршрутизаторі, насамперед, порівняє цю адресу з адресою 129.44.128.15, для якої визначено специфічний

маршрут. Збігу немає, тому модуль IP починає послідовно опрацьовувати всі рядки таблиці, накладаючи маски й порівнюючи результати доти, доки не знайде збіги номера мережі в адресі призначення та у рядку таблиці.

У результаті визначається маршрут для пакета 129.44.78.200 – він має бути відправлений на вихідний порт маршрутизатора 129.44.64.7 у мережу 129.44.64.0, безпосередньо підключену до даного маршрутизатора.

Таблиця 13.1 – Таблиця маршрутизатора R2 у мережі з масками однакової довжини

Адреса призначення	Маска	Адреса наступного маршрутизатора	Адреса порту	Відстань
129.44.0.0	255.255.192.0	129.44.0.1	129.44.0.1	Підключена
129.44.64.0	255.255.192.0	129.44.64.7	129.44.64.7	Підключена
129.44.128.0	255.255.192.0	129.44.128.5	129.44.128.5	Підключена
129.44.192.0	255.255.192.0	129.44.192.1	129.44.192.1	Підключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–
129.44.128.15	255.255.255.255	129.44.64.8	129.44.64.7	–

Перекриття адресних просторів

Зі складностями використання масок адміністратор вперше стикається не тоді, коли починають конфігурувати мережні інтерфейси й створювати таблиці маршрутизації, а набагато раніше – на етапі планування мережі.

Планування містить:

– визначення кількості мереж, з яких складатиметься корпоративна мережа;

– оцінку необхідної кількості адрес для кожної мережі, отримання пулу адрес від постачальника послуг;

– розподіл адресного простору між мережами. Останнє завдання часто виявляється нетривіальним, особливо коли вирішується в умовах дефіциту адрес.

Розглянемо приклад використання масок для організації адресних просторів, що перекриваються.

Нехай на деякому підприємстві було ухвалене рішення звернутися до постачальника послуг для отримання пулу адрес, достатнього для створення мережі, структура, якої показана на рис. 13.2.

Мережа клієнта містить три підмережі.

Дві з них – це надійно захищені від зовнішніх атак внутрішні мережі відділів: мережа Ethernet на 600 користувачів і мережа Token Ring на 200 користувачів.

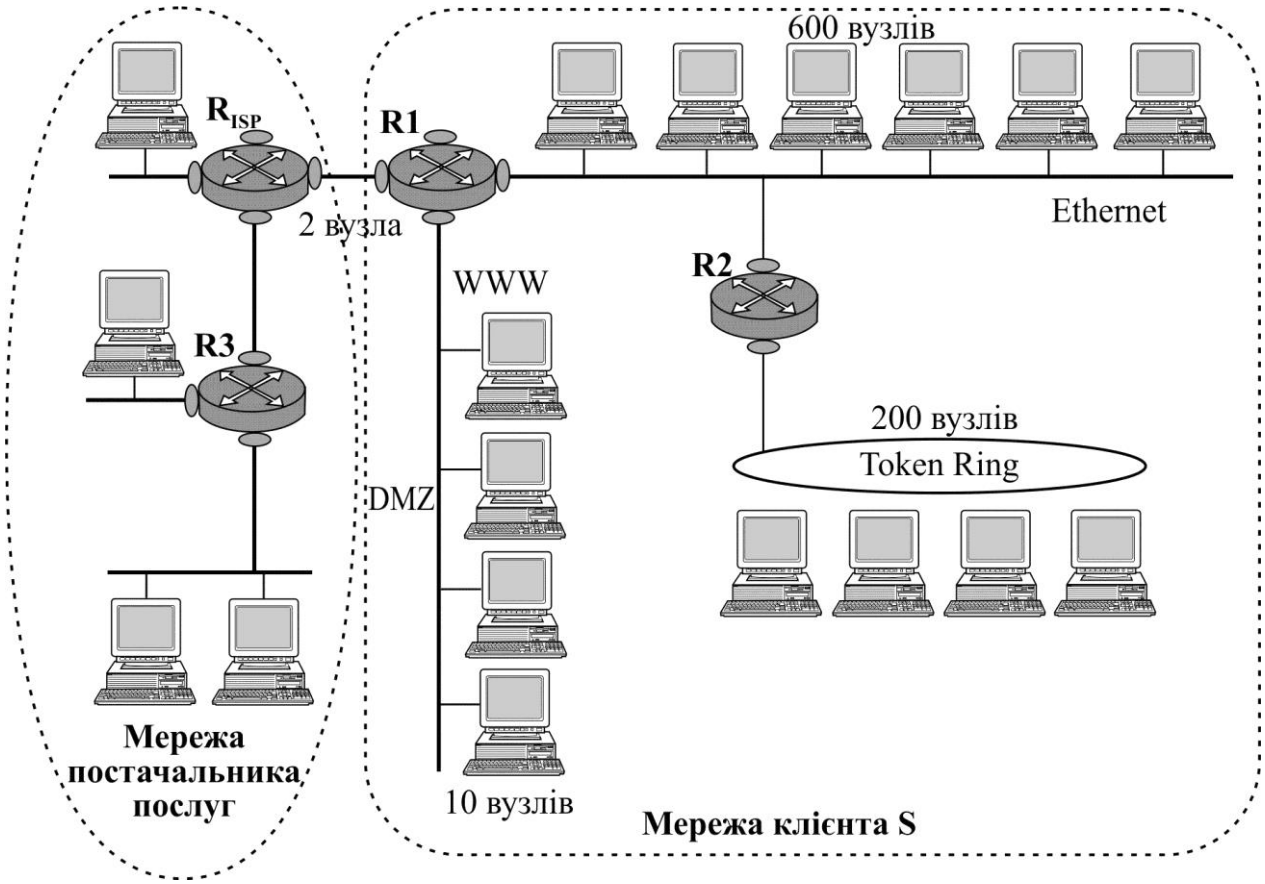


Рисунок 13.2 – Мережі постачальника послуг і клієнта

Підприємство також передбачає окрему, відкриту для доступу ззовні мережу на 10 вузлів, головне призначення якої – надання інформації у режимі відкритого доступу для потенційних клієнтів. Такого роду ділянки корпоративної мережі, у яких розташовуються веб-сервери, FTP-сервери та інші джерела публічної інформації, називають демілітаризованою зоною (Demilitarized Zone, DMZ).

Ще одна мережа на два вузли буде потрібна для зв’язку з постачальником послуг, тобто загальна кількість адрес, необхідних для адресації мережних інтерфейсів, становить 812.

Крім того, необхідно, щоб пул доступних адрес містив для кожної з мереж широкомовні адреси, які складаються лише з одиниць, а також адреси, що складаються тільки з нулів.

Розв’язання. Враховуючи також, що в будь-якій мережі адреси всіх вузлів повинні мати однакові префікси, стає очевидним, що мінімальна кількість адрес, необхідна клієнту для побудови задуманої мережі, може істотно відрізнятися від значення 812, отриманого простим підсумовуванням.

У даному прикладі постачальник послуг вирішує виділити клієнту безперервний пул з 1024 адрес. Значення 1024 обране якнайближче

до необхідної кількості адрес, що дорівнює степеню двійки $\left(2^{10}=1024\right)$. Постачальник послуг виконує пошук області такого розміру в наявному у нього адресному просторі – 131.57.0.0/16, частина якого, як показано на рис. 13.3, вже розподілена. Позначимо розподілені ділянки й клієнтів, що володіють ними, через S1, S2 і S3. Постачальник послуг знаходить серед нерозподілених ще адрес безперервну ділянку розміром 1024 адреси, початкова адреса якого кратна розміру даної ділянки. Отже, наш клієнт отримує пул адрес 131.57.8.0/22, позначений на рисунку через S.

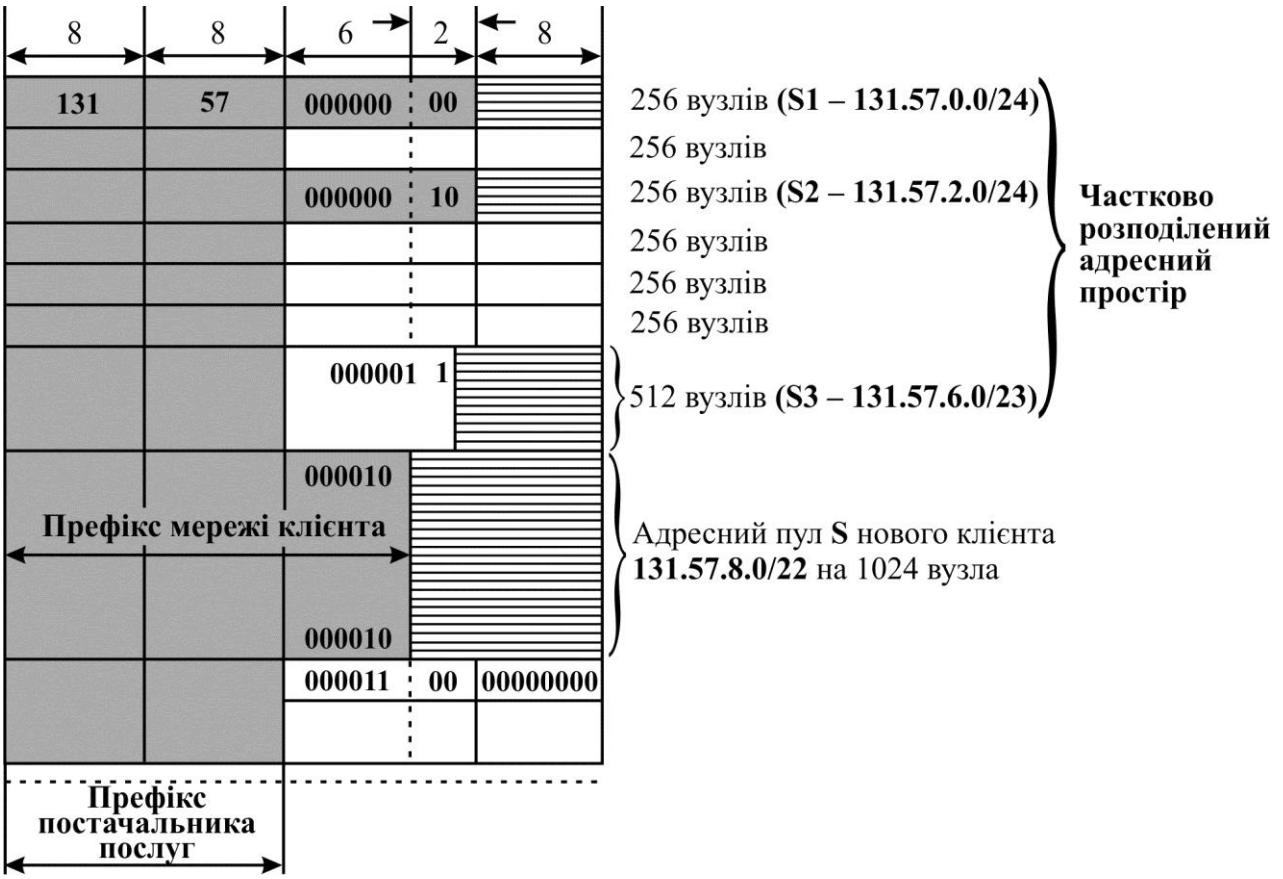


Рисунок 13.3 – Адресний простір постачальника послуг

Далі починається найскладніший етап – розподіл отриманого від постачальника послуг адресного пулу S між чотирма мережами клієнта. Насамперед адміністратор вирішив призначити для найбільшої мережі (Ethernet на 600 вузлів) увесь пул адрес 131.57.8.0/22, отриманий від постачальника послуг (рис. 13.4). Номер, призначений для цієї мережі, збігається з номером мережі, отриманим від постачальника послуг. А як бути з трьома мережами, що залишилися? Адміністратор зважив, що для мережі Ethernet потрібно тільки 600 адрес, а з 624, що залишилися, «викроїв» мережу Token Ring 131.57.9.0/24 на 250 адрес. Скориставшись тим, що для Token Ring потрібно лише 200 адрес,

він «вирізав» з неї дві ділянки: для мережі DMZ 131.57.9.16/28 на 16 адрес і для єднальної мережі 131.57.9.32/30 на 4 адреси. Внаслідок цього всі мережі клієнта отримали достатню (а іноді й з надлишком) кількість адрес.

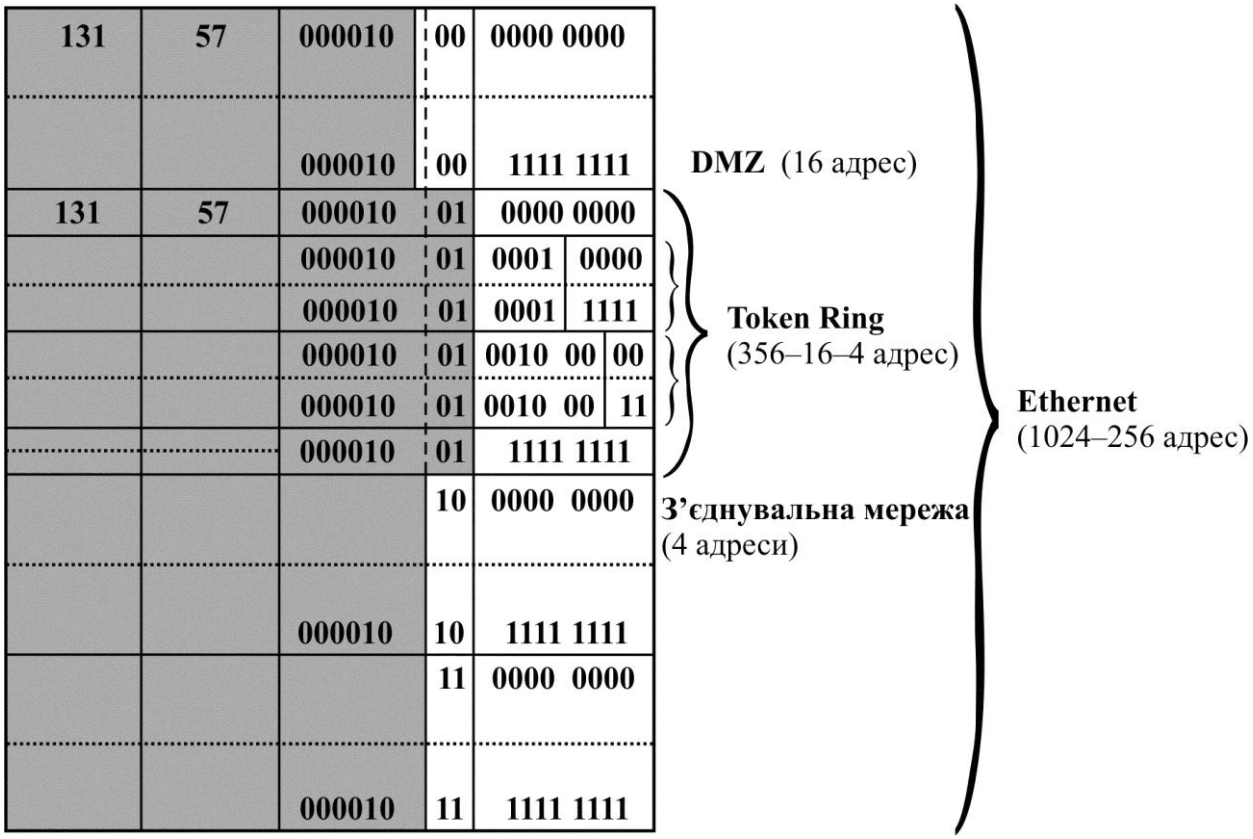


Рисунок 13.4 – Планування адресного простору для мереж клієнта

Наступний етап – це конфігурування мережних інтерфейсів кінцевих вузлів і маршрутизаторів. Кожному інтерфейсу повідомляється його IP-адреса й відповідна маска. На рис. 13.5. показана сконфігурована мережа клієнта.

Після конфігурування мережних інтерфейсів мають бути створені таблиці маршрутизації маршрутизаторів R1 і R2 клієнта. Вони можуть бути згенеровані автоматично або за участі адміністратора. Таблиця маршрутизації маршрутизатора R2 відповідає табл. 13.2.

У даній таблиці немає маршруту за замовчуванням, тобто усі пакети, адресовані мережам, адреси яких явно не зазначені в таблиці, відкидатимуться маршрутизатором. Нехай, наприклад, на маршрутизатор R2 надходить пакет з адресою призначення 131.57.9.29. Внаслідок перегляду таблиці отримуємо такі результати для кожного рядка:

(131.57.9.29) AND (255.255.252.0) = 131.57.8.0 – збіг;

(131.57.9.29) AND (255.255.255.0) = 131.57.9.0 – збіг;

(131.57.9.29) AND (255.255.255.240) = 131.57.9.16 – збіг;

(131.57.9.29) AND (255.255.255.252) = 131.57.9.28 – немає збігу.

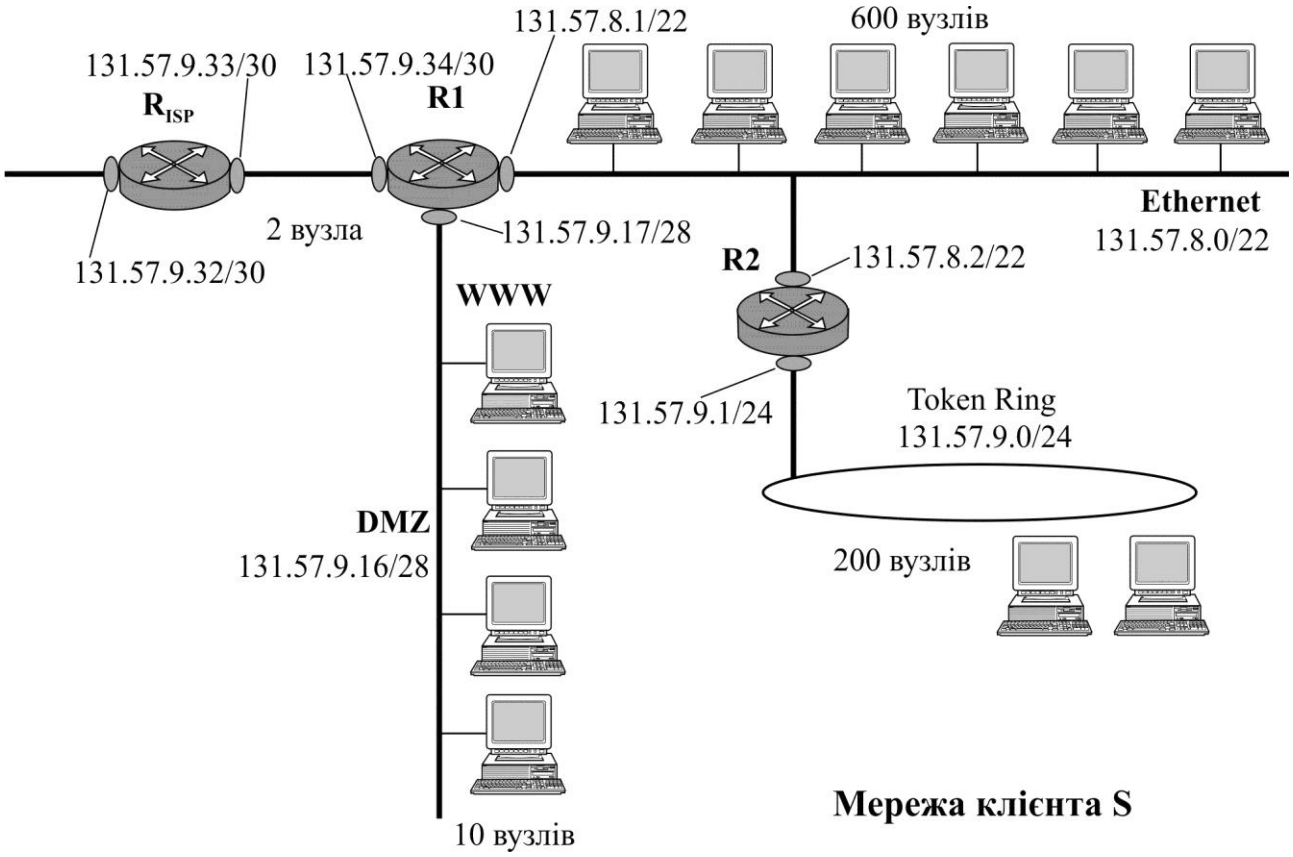


Рисунок 13.5 – Зконфігурована мережа клієнта

Таблиця 13.2 – Таблиця маршрутизації маршрутизатора R2

Адреса призначення	Маска	Адреса наступного маршрутизатора	Адреса порту	Відстань
131.57.8.0	255.255.252.0	131.57.8.2	131.57.8.2	Підключена
131.57.9.0	255.255.255.0	131.57.9.1	131.57.9.1	Підключена
131.57.9.16	255.255.255.240	131.57.8.1	131.57.8.2	1
131.57.9.32	255.255.255.252	131.57.8.1	131.57.8.2	1

Оскільки за наявності декількох збігів обирається маршрут з того рядка, у якому збіг адреси призначення з адресою з пакета має найбільшу довжину, визначено, що пакет з адресою 131.57.9.29 направляється в мережу DMZ.

CIDR

За останні кілька років в Інтернеті багато чого змінилося: різко зросла кількість вузлів і мереж, підвищилася інтенсивність трафіка, змінився характер даних, що передаються. Через недосконалість протоколів маршрутизації обмін повідомленнями про відновлення таблиць став призводити до збоїв

магістральних маршрутизаторів, що відбуваються через перевантаження в процесі обробки великого обсягу службової інформації. Так, сьогодні таблиці магістральних маршрутизаторів в Інтернеті можуть містити до декількох сотень і навіть тисяч маршрутів.

На вирішення цієї проблеми спрямована, зокрема, і технологія безкласової міждоменної маршрутизації (Classless Inter-Domain Routing, CIDR).

Сутність технології CIDR полягає в такому. Кожному постачальнику послуг Інтернету призначається безперервний діапазон IP-адрес. За такого підходу всі адреси кожного постачальника послуг мають загальну старшу частину – префікс, тому маршрутизація на магістралях Інтернету може здійснюватися на основі префіксів, а не повних адрес мереж. А це означає, що замість безлічі записів за кількістю мереж буде достатньо помістити один запис для всіх мереж, що мають загальний префікс. Таке агрегування адрес дозволить зменшити обсяг таблиць у маршрутизаторах усіх рівнів, а, отже, прискорити роботу маршрутизаторів і підвищити пропускну здатність Інтернету.

Раніше ми розглядали приклади, де адміністратори корпоративних мереж за допомогою масок ділили на кілька частин безперервний пул адрес, отриманий від постачальника послуг, щоб використовувати ці частини для структуризації своєї мережі. Такий варіант застосування масок називається поділом на підмережі.

Разом з тим у процесі поділу на підмережі за допомогою масок проявлявся й зворотний ефект їхнього застосування. Тобто, для того щоб направити увесь сумарний трафік, адресований із зовнішнього оточення у корпоративну мережу, розділену на підмережі, достатньо, щоб в усіх зовнішніх маршрутизаторах був у наявності один рядок.

У цьому рядку на місці адреси призначення має бути зазначений загальний префікс для усіх цих мереж. Тут ми маємо справу з операцією, зворотною поділу на підмережі, – операцією агрегування кількох мереж в одну велику. Повернемося до рис. 13.3, на якому показаний адресний простір постачальника послуг з ділянками S1, S2, S3 і S, переданими в користування чотирьом клієнтам. Цей приклад також ілюструє рис. 13.6. Внаслідок агрегування мереж клієнтів у табл. 13.3 маршрутизатора R_{ISP} постачальника послуг для кожного клієнта буде виділено по одному рядку незалежно від кількості підмереж, організованих ними у своїх мережах. Так, замість чотирьох маршрутів до чотирьох мереж клієнта S, у таблиці заданий тільки один загальний для всіх маршрут (виділений жирним шрифтом).

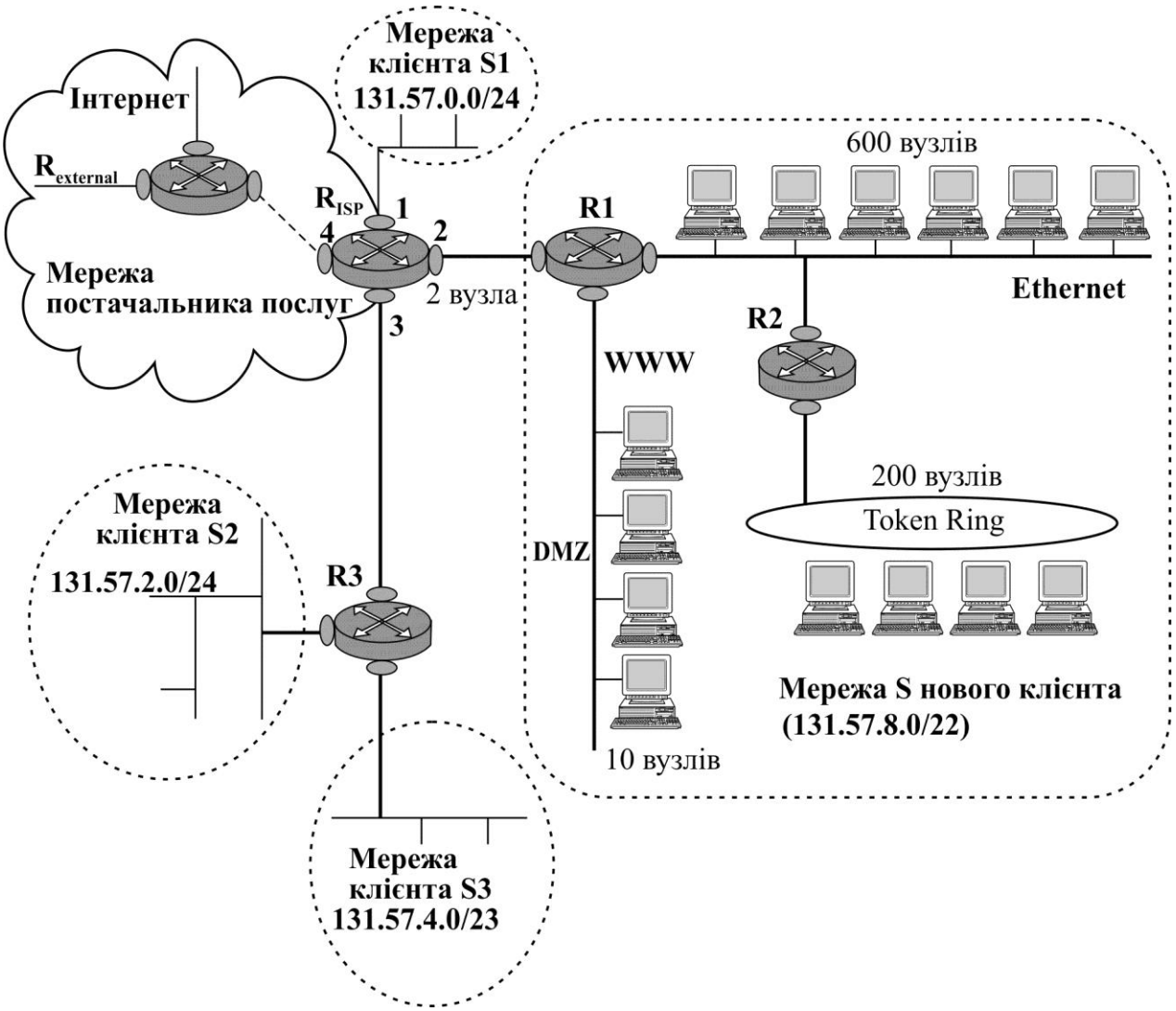


Рисунок 13.6 – Об’єднання підмереж

Таблиця 13.3 – Таблиця маршрутизатора R_{ISP} постачальника послуг

Адреса призначення	Маска	Наступний маршрутизатор	Номер вихідного інтерфейсу	Відстань
131.57.0.0 (S1)	255.255.255.0	R3	1	Підключена
131.57.2.0 (S2)	255.255.255.0	R3	3	1
131.57.4.0 (S3)	255.255.254.0	R1	3	1
131.57.8.0 (S)	255.255.252.0	R1	2	Підключена
Маршрут за замовчуванням	0.0.0.0	Rexternal	4	–

Отже, впровадження технології CIDR дозволяє розв’язати дві основні задачі:

1. Більш економне витрачання адресного простору. Завдяки технології CIDR постачальники послуг отримують можливість «нарізати» блоки з виділеного їм адресного простору в чіткій відповідності з вимогами кожного

клієнта, при цьому у клієнта залишається простір для маневру на випадок майбутнього зростання.

2. Зменшення кількості записів у таблицях маршрутизації за рахунок об'єднання маршрутів – один запис у таблиці маршрутизації може представляти велику кількість мереж. Якщо всі постачальники послуг Інтернету почнуть дотримуватися стратегії CIDR, то особливо помітний виграш досягатиметься в магістральних маршрутизаторах.

Необхідною умовою ефективного використання технології CIDR є локалізація адрес, тобто призначення адрес, які мають префікси, що збігаються, мережам, територіально розташованим по сусідству. Тільки в такому випадку трафік може бути агрегованим.

На жаль, сьогодні розподіл адрес носить багато в чому випадковий характер. Кардинальний шлях вирішення проблеми – перенумерування мереж. Але ця процедура пов'язана з певними часовими та матеріальними витратами, і для її проведення користувачів потрібно якось стимулювати. Як стимул розглядається, наприклад, введення оплати за рядок у таблиці маршрутизації або ж за кількість вузлів у мережі. Перша вимога підводить споживача до думки отримати у постачальника послуг таку адресу, щоб маршрутизація трафіка в його мережу йшла на підставі префікса, і номер його мережі не фігурував більше в магістральних маршрутизаторах.

Вимога оплати кожної адреси вузла також може підштовхнути користувача зважитися на перенумерування для того, щоб отримати стільки адрес, скільки йому потрібно.

Технологія CIDR уже успішно використовується у поточній версії протокола IP (Ipv4) і підтримується такими протоколами маршрутизації, як OSPF, RIP-2, BGP4 (як правило, на магістральних маршрутизаторах Інтернету). Особливості застосування технології CIDR у новій версії протоколу IP (Ipv6) будуть розглянуті пізніше.

13.2 Фрагментація IP-пакетів

Важливою особливістю протоколу IP, що відрізняє його від інших мережних протоколів (наприклад, від мережного протоколу IPX, що у минулому деякий час конкурував з IP), є його здатність виконувати динамічну фрагментацію пакетів в процесі передачі їх між мережами з різними максимально припустимими значеннями довжини поля даних кадрів (Maximum Transmission Unit, MTU). Значення MTU залежать як від протоколу, так і від настроювання мережних інтерфейсів.

Насамперед зазначимо різницю між фрагментацією повідомлень у вузлі-відправнику й динамічною фрагментацією повідомлень у транзитних вузлах мережі – маршрутизаторах. У першому випадку розподіл повідомлення на декілька менших частин (фрагментація) відбувається в ході передачі даних між протоколами одного й того самого стека усередині комп'ютера. Протоколи, що виконують фрагментацію в межах вузла, аналізують тип технології нижнього рівня, визначають її MTU і розділяють повідомлення на такі частини, які вміщуються в кадри канального рівня того самого стека протоколів.

У стеку TCP/IP це завдання вирішує протокол TCP, який розбиває потік байтів, переданий йому із прикладного рівня, на сегменти потрібного розміру, наприклад, по 1460 байт, якщо на нижньому рівні даної мережі працює протокол Ethernet. Протокол IP у вузлі-відправнику, як правило, не використовує свої можливості з фрагментації пакетів.

А от на транзитному вузлі – маршрутизаторі, коли пакет необхідно передати з мережі з більшим значенням MTU у мережу з меншим значенням MTU, здатності протокола IP виконувати фрагментацію стають затребуваними. Пакети-фрагменти, подорожуючи мережею, можуть удруге зазнати фрагментації на будь-якому з проміжних маршрутизаторів.

Параметри фрагментації

Кожен з фрагментів має бути забезпечений повноцінним заголовком IP. Деякі з полів заголовка (ідентифікатор, TTL, прапорці DF і MF, зсув) безпосередньо призначені для подальшого складання фрагментів у вихідне повідомлення.

Ідентифікатор пакета використовується для розпізнавання пакетів, що утворювалися шляхом розподілу на частини (фрагментації) вихідного пакета. Усі частини (фрагменти) одного пакета повинні мати однакове значення цього поля. Модуль IP, який відправляє пакет, встановлює в поле ідентифікатора значення, яке має бути унікальним для даної пари відправника та отримувача протягом усього часу, поки даний пакет (або будь-який його фрагмент) може існувати в складовій IP-мережі.

Поле часу життя (Time Te Live, TTL) займає один байт і визначає граничний термін, протягом якого пакет може переміщатися мережею. Час життя пакета вимірюється в секундах і задається джерелом (відправником). Як вже зазначалося, по закінченні кожної секунди перебування на кожному з маршрутизаторів, через які проходить пакет під час своєї «подорожі» мережею, з його поточного часу життя віднімається одиниця; одиниця віднімається й у тому випадку, якщо час перебування був менше секунди. Оскільки сучасні

маршрутизатори рідко опрацьовують пакет довше, ніж за одну секунду, то час життя можна інтерпретувати як максимальну кількість транзитних вузлів, які дозволено пройти пакету. Якщо значення поля часу життя стає нульовим до того, як пакет досягає отримувача, пакет знищується. Під час складання фрагментів хост-отримувач використовує значення TTL як крайній термін очікування відсутніх фрагментів.

Поле зсуву фрагмента надає отримувачу інформацію про становище фрагмента щодо початку поля даних вихідного нефрагментованого пакета. Так, наприклад, перший фрагмент матиме в полі зміщення нульове значення. У пакеті, не розбитому на фрагменти, поле зміщення також має нульове значення. Зсув задається в байтах і має бути кратним 8 байт.

Встановлений в одиницю однобітний прапорець MF (More Fragments – більше фрагментів) говорить про те, що даний пакет є проміжним (не останнім) фрагментом. Модуль IP, що відправляє нефрагментований пакет, встановлює біт MF у нуль.

Прапорець DF (Do not Fragment – не фрагментувати), встановлений в одиницю, забороняє маршрутизатору фрагментувати даний пакет. Якщо позначений у такий спосіб пакет не може досягти отримувача без фрагментації, то модуль IP його знищує, а вузлу-відправнику надсилає діагностичне повідомлення.

> ***ПРИМІТКА***. *Можливість заборонити фрагментацію дозволяє в деяких випадках прискорити роботу додатків. Для цього спочатку необхідно дослідити мережу, визначити максимальний розмір пакета, який зможе пройти весь шлях без фрагментації, а потім використовувати пакети такого або меншого розміру для обміну даними. Дана можливість дозволяє також запобігти фрагментації в тих випадках, коли хост-отримувач не має достатніх ресурсів для складання фрагментів.*

13.3 Механізм фрагментації

Розглянемо механізм фрагментації на прикладі складеної мережі, показаної на рис. 13.7.

В одній з підмереж (Frame Relay) значення MTU дорівнює 4080, в іншій (Ethernet) – 1492. Хост, що належить мережі Frame Relay, передає дані хосту в мережі Ethernet. На обох хостах, а також на маршрутизаторі, що зв’язує ці підмережі, встановлено стек протоколів TCP/IP.

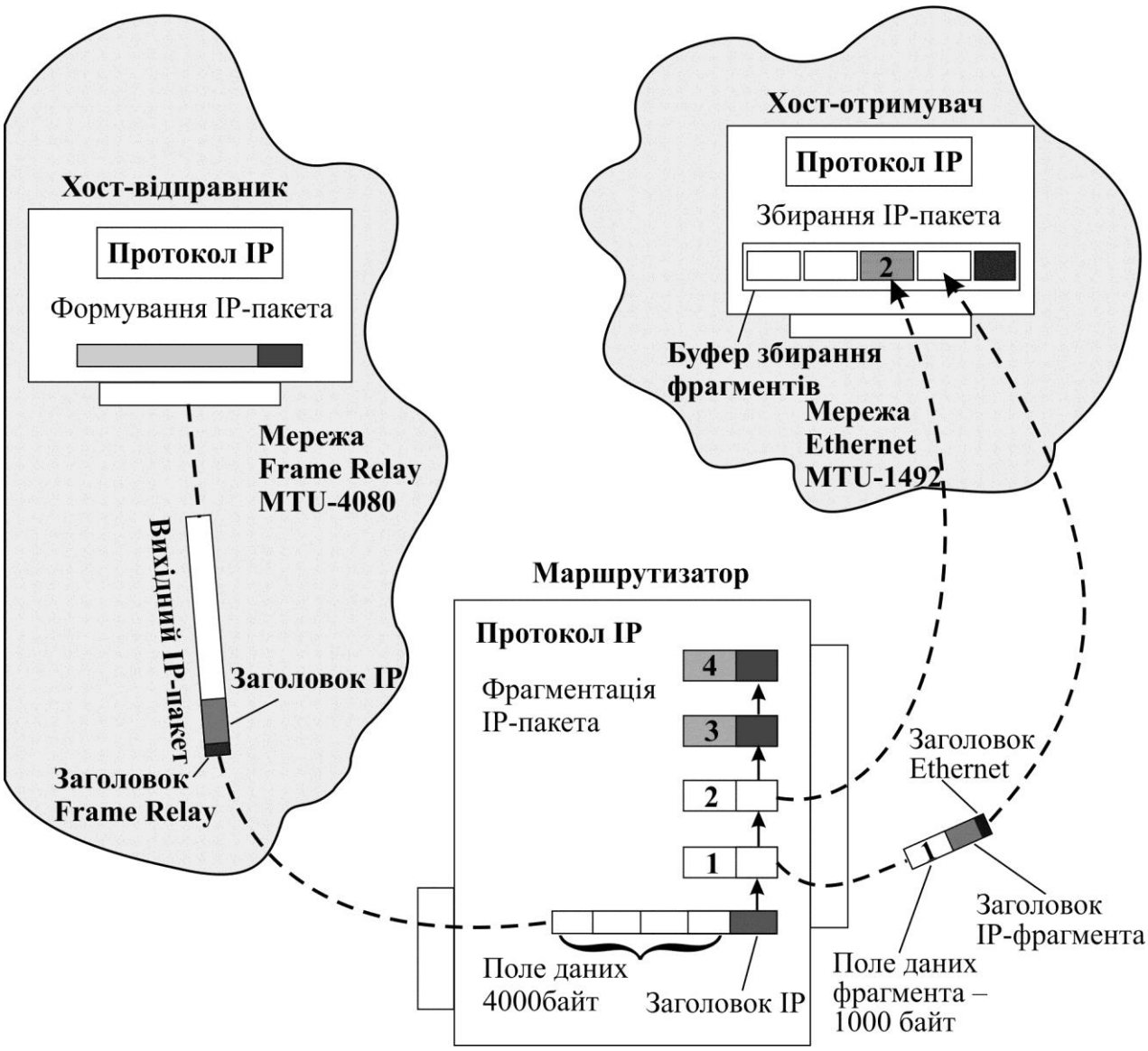


Рисунок 13.7 – Фрагментація

Транспортному рівню хоста-відправника відоме значення MTU технології нижчого рівня (4080). На підставі цього модуль TCP «нарізає» свої сегменти розміром 4000 байт і передає вниз протоколу IP, який поміщує сегменти в поле даних IP-пакетів і генерує для них заголовки. Особливу увагу звернемо на заповнення тих полів заголовка, які прямо пов’язані з фрагментацією:

– пакету присвоюється унікальний ідентифікатор, наприклад, 12456;

– оскільки пакет поки ще не був фрагментированій, у поле зсуву поміщується значення 0;

– ознака MF також обнуляється, це показує, що пакет є своїм і останнім фрагментом водночас;

– ознака DF встановлюється в 1, це означає, що даний пакет можна фрагментувати.

Загальна величина IP-пакета становить 4000 плюс 20 (розмір заголовка IP), тобто 4020 байт, що вміщається в поле даних кадра Frame Relay, яке в даному прикладі дорівнює 4080. Далі модуль IP хоста-відправника передає цей кадр своєму мережному інтерфейсу Frame Relay, який відправляє кадри наступним маршрутизатором.

Модуль IP маршрутизатора за мережною адресою IP-пакета, який надійшов, визначає, що пакет потрібно передати в мережу Ethernet. Але вона має значення MTU, яке дорівнює 1492, що значно менше розміру пакета, який надійшов на вхідний інтерфейс. Отже, IP-пакет необхідно фрагментувати. Модуль IP обирає розмір поля даних фрагмента таким, що дорівнює 1000, так що з одного великого IP-пакета виходить 4 маленьких пакета-фрагмента. Для кожного фрагмента і його заголовка IP в маршрутизаторі створюється окремий буфер (на рисунку фрагменти і відповідні їм буфери пронумеровані від 1 до 4). Протокол IP копіює в ці буфери вміст деяких полів заголовка IP вихідного пакета, створюючи тим самим «заготовки» заголовків IP усіх нових пакетів-фрагментів. Одні параметри заголовка IP копіюються в заголовки всіх фрагментів, інші – лише в заголовок першого фрагмента.

У процесі фрагментації можуть змінитися значення деяких полів заголовків IP у пакетах-фрагментах порівняно із заголовком IP вихідного пакета. Так, кожен фрагмент має власні значення контрольної суми заголовка, зсуву фрагмента й загальної довжини пакета. В усіх пакетах, крім останнього, прапорець MF встановлюється в одиницю, а в останньому фрагменті – у нуль. Отримані пакети-фрагменти мають довжину 1020 байт (з урахуванням заголовка IP), тому вони вільно поміщаються в поле даних кадрів Ethernet.

На рисунку показані різні стадії переміщення фрагментів по мережі. Фрагмент 2 уже досягнув хоста-отримувача й поміщений у прийомний буфер. Фрагмент 1 ще переміщається по мережі Ethernet, інші фрагменти перебувають у буферах маршрутизатора.

А тепер обговоримо, як відбувається складання фрагментованого пакета на хості призначення.

> ***ПРИМІТКА.*** *Зазначимо, що IP-маршрутизатори не збирають фрагменти пакетів у великі пакети, навіть якщо на шляху зустрічається мережа, що допускає таке укрупнення. Це пов'язано з тим, що окремі фрагменти повідомлення можуть переміщуватися складеною мережею різними маршрутами, тому немає гарантії, що всі фрагменти на своєму шляху пройдуть через один визначений маршрутизатор.*

На хості призначення для кожного фрагментованого пакета виділяється окремий буфер. У цей буфер протокол IP, що приймає, поміщає IP-фрагменти, у яких збігаються IP-адреси відправника й отримувача, а також значення в полях ідентифікатора (у нашому прикладі – 12456). Усі ці ознаки показують модулю IP, що дані пакети є фрагментами одного вихідного пакета. Складання полягає в розміщенні даних з кожного фрагмента в позицію, визначену зсувом, зазначеним у заголовку фрагмента.

Коли перший фрагмент вихідного пакета приходить на хост-отримувач, цей хост запускає таймер, який визначає максимальний час очікування прибуття інших фрагментів даного пакета. У різних реалізаціях IP застосовуються різні правила вибору максимального часу очікування. Зокрема, таймер може бути встановлений на фіксований період часу (від 60 до 120 секунд), рекомендований RFC. Як правило, цей інтервал достатній для доставки пакета від відправника отримувачу. В інших реалізаціях максимальний час очікування визначається за допомогою адаптивних алгоритмів виміру й статистичної обробки часових параметрів мережі, що дозволяють оцінювати очікуваний час прибуття фрагментів. Нарешті, тайм-аут може бути обраний на базі значень TTL, фрагментів, що прибувають. Останній підхід заснований на тому, що нема рації очікувати, поки прибудуть інші фрагменти пакета, якщо час життя одного із прибулих фрагментів уже минув.

> ***ПРИМІТКА***. *Якщо хоча б один фрагмент пакета не встигне прийти на хост призначення до моменту закінчення таймера, то жодних дій з дублювання відсутнього фрагмента не відбувається, а усі отримані до цього часу фрагменти пакета відкидаються! Хосту, що надіслав вихідний пакет, направляється ICMP-повідомлення про помилку. Такій поведінці протоколу IP цілком відповідає його кредо «з максимальними зусиллями» – намагатися за можливості, але жодних гарантій не давати.*

Ознакою закінчення складання є відсутність незаповнених проміжків у полі даних і надходження останнього фрагмента (з прапорцем MF рівним нулю) до моменту закінчення тайм-ауту. Після того як дані зібрані, їх можна передати протоколу вищого рівня, наприклад, TCP.

Протокол IP вирішує завдання доставки повідомлень між вузлами складеної мережі. Оскільки він є дейтаграмним, жодних гарантій надійної доставки повідомлень не дається.

Максимальна довжина IP-пакета становить 65535 байт. Заголовок зазвичай має довжину 20 байт і містить інформацію про мережні адреси

відправника й отримувача, параметри фрагментації, час життя пакета, контрольну суму й деякі інші параметри.

Вигляд таблиці IP-маршрутизації залежить від конкретної реалізації маршрутизатора. Незважаючи на значні зовнішні відмінності виведених на екран таблиць, усі вони містять два обов'язкових поля – це поля адреси призначення й наступного маршрутизатора.

Записи у таблицю маршрутизації можуть надходити з різних джерел. По-перше, внаслідок конфігурування програмне забезпечення стека TCP/IP заносить у таблицю записи про безпосередньо підключені мережі й маршрутизатори за замовчуванням, а також записи про особливі адреси. По-друге, адміністратор вручну заносить записи про специфічні маршрути й про маршрут за замовчуванням. По-третє, протоколи маршрутизації автоматично заносять у таблицю динамічні записи про наявні маршрути.

Ефективним засобом структуризації IP-мереж є маски. Маски дозволяють розділити одну мережу на декілька підмереж або об'єднати кілька мереж в одну велику мережу.

Значна роль у майбутніх IP-мережах приділяється технології безкласової міждоменної маршрутизації (CIDR), яка вирішує дві основних задачі. Перша полягає в більш економному витрачанні адресного простору, друга – у зменшенні кількості записів у таблицях.

Важливою особливістю протоколу IP, що відрізняє його від інших мережних протоколів, наприклад, від мережного протоколу IPX, є його здатність виконувати динамічну фрагментацію пакетів при передачі їх між мережами з різними максимально припустимими значеннями довжини поля даних кадрів (MTU).

ТЕОРЕТИЧНИЙ МАТЕРІАЛ
ДО ПРАКТИЧНОГО ЗАНЯТТЯ

1 Основні функції протоколу IP

Основу транспортних засобів стека протоколів TC /IP становить протокол міжмережної взаємодії (Internet Protocol, IP). Він забезпечує передачу дейтаграм від відправника до отримувачів через об'єднану систему комп'ютерних мереж.

Назва даного протоколу – Intrenet Protocol – відображає його суть: він має передавати пакети між мережами. У кожній наступній мережі, що знаходиться на шляху переміщення пакета, протокол IP викликає засоби транспортування, прийняті в даній мережі, щоб з їх допомогою передати пакет на маршрутизатор, що веде до наступної мережі, або безпосередньо на вузол-отримувач.

Протокол IP належить до протоколів без встановлення з'єднань. Перед IP не ставиться завдання надійної доставки повідомлень від відправника до отримувача. Протокол IP обробляє кожен IP-пакет як незалежну одиницю, що не має зв'язку ні з жодними IP-пакетами. У протоколі IP немає механізмів, які зазвичай застосовуються для збільшення вірогідності кінцевих даних: відсутнє квітирування – обмін підтвердженнями між відправником і отримувачем, немає процедури упорядкування, повторних передач або інших подібних функцій. Якщо під час просування пакета відбулася якась помилка, то протокол IP за своєю ініціативою нічого не робить для виправлення цієї помилки. Наприклад, якщо на проміжному маршрутизаторі пакет був відкинутий через закінчення часу життя або через помилку у контрольній сумі, то модуль IP не намагається заново надіслати зіпсований або втрачений пакет. Усі питання забезпечення надійності доставки даних складеною мережею у стеку TCP/IP вирішує протокол TCP, що працює безпосередньо над протоколом IP. Саме TCP організовує повторну передачу пакетів, коли в цьому виникає необхідність.

Важливою особливістю протоколу IP, що відрізняє його від інших мережних протоколів (наприклад, від мережного протоколу IPX), є його здатність виконувати динамічну фрагментацію пакетів у процесі передачі їх між мережами з різними, максимально допустимими, значеннями поля даних кадрів MTU. Здатність до фрагментації сприяла тому, що протокол IP зміг зайняти домінуючі позиції в складних складених мережах.

Є прямий зв’язок між функціональною складністю протоколу і складністю заголовка пакетів, які даний протокол використовують. Це пояснюється тим, що основні службові дані, на підставі яких протокол виконує ту чи іншу дію, переносяться між двома модулями, що реалізують цей протокол на різних машинах, саме в полях заголовків пакетів. Тому дуже корисно вивчити призначення кожного поля заголовка IP-пакета, і це дає не лише формальні знання про структуру пакета, а й пояснює всі основні режими роботи протоколу з обробки та передачі IP-дейтаграм.

2 Структура IP-пакета

IP-пакет складається з заголовка і поля даних. Тема, як правило, має довжину 20 байт і таку структуру (рис. А.1).

<table>
<tr><td colspan="2">4 біта
Номер версії</td><td colspan="2">4 біта
Довжина заголовка</td><td colspan="4">8 біт
Тип сервісу
PR | D | T | R</td><td colspan="4">16 біт
Загальна довжина</td></tr>
<tr><td colspan="6">16 біт
Ідентифікатор пакета</td><td colspan="2">3 біта
Прапори
D | M</td><td colspan="2">13 біт
Зсув фрагмента</td></tr>
<tr><td colspan="3">8 біт
Час життя</td><td colspan="3">8 біт
Протокол верхнього рівня</td><td colspan="4">16 біт
Контрольна сума</td></tr>
<tr><td colspan="10">32 біта
IP-адреса джерела</td></tr>
<tr><td colspan="10">32 біта
IP-адреса призначення</td></tr>
<tr><td colspan="10">Опції та вирівнювання</td></tr>
</table>

Рисунок А.1 – Структура заголовка IP-пакета

Поле ***Номер версії*** (Version), що займає 4 біта, вказує версію протоколу IP. Сьогодні скрізь використовується версія 4 (IPv4), і готується перехід на версію 6 (IPv6).

Поле ***Довжина заголовка*** (IHL) IP-пакета займає 4 біта і вказує значення довжини заголовка, що виміряється в 32-бітових словах. Зазвичай заголовок має довжину в 20 байт (п’ять 32-бітових слів), але зі збільшенням обсягу службової інформації ця довжина може бути збільшена за рахунок використання додаткових байт у полі ***Опції*** (IP Options). Найбільший заголовок займає 60 октетів.

Поле ***Тип сервісу*** (Type of Service) займає один байт і задає пріоритетність пакета і вид критерію вибору маршруту. Перші три біта цього поля утворюють підполе пріоритету пакета (Precedence). Пріоритет може мати значення від найнижчого – 0 (нормальний пакет) до найвищого – 7 (пакет керуючої інформації). Маршрутизатор і комп'ютери можуть брати до уваги пріоритет пакета і обробляти важливіші пакети передусім. Поле ***Тип сервісу*** містить також три біта, які визначають критерій вибору маршруту. Реально вибір здійснюється між трьома альтернативами: малою затримкою, високою вірогідністю і високою пропускною здатністю. Встановлений **біт D (delay)** говорить про те, що маршрут має обиратися для мінімізації затримки доставки даного пакета, **біт T** – для максимізації пропускної здатності, а **біт R** – для максимізації надійності доставки. У багатьох мережах покращення одного з цих параметрів пов'язане з погіршенням іншого, крім того, обробка кожного з них вимагає додаткових обчислювальних витрат. Тому рідко є сенс водночас встановлювати хоча б два з цих трьох критеріїв вибору маршруту. Зарезервовані біти мають нульове значення.

Поле ***Загальна довжина*** (Total Length) займає 2 байта і означає загальну довжину пакета з урахуванням заголовка і поля даних. Максимальна довжина пакета обмежена розрядністю поля, яке визначає цю величину, і становить 65 535 байт, проте в більшості хост-комп'ютерів і мереж такі великі пакети не використовуються. В процесі передачі мережами різного типу довжина пакета обирається з урахуванням максимальної довжини пакета протоколу нижнього рівня, що несе IP-пакети. Якщо це кадри Ethernet, то обираються пакети з максимальною довжиною в 1500 байт, які вміщуються в поле даних кадра Ethernet. У стандарті передбачається, що всі хости мають бути готові приймати пакети довжиною до 576 байт (надходять вони цілком або фрагментами). Хостам рекомендується відправляти пакети розміром більше ніж 576 байт, тільки якщо вони впевнені, що хост, який приймає, або проміжна мережа готові обслуговувати пакети такого розміру.

Поле ***Ідентифікатор пакета*** (Identification) займає 2 байта і використовується для розпізнавання пакетів, які утворилися шляхом фрагментації вихідного пакета. Всі фрагменти повинні мати однакове значення цього поля.

Поле ***Прапорці*** (Flags) займає 3 біта і містить ознаки, пов'язані з фрагментацією. Встановлений **біт DF** (Do not Fragment) забороняє маршрутизатору фрагментувати даний пакет, а встановлений **біт MF** (More Fragments) говорить про те, що даний пакет є проміжним (не останнім) фрагментом. Біт, що залишився, зарезервований.

Поле ***Зсув фрагмента*** (Fragment Offset) займає 13 біт і задає зсув у байтах поля даних цього пакета від початку загального поля даних вихідного пакета, що зазнає фрагментації. Використовується у процесі складання/розбирання фрагментів пакетів під час передачі їх між мережами з різними величинами MTU. Зсув має бути кратним 8 байт.

Поле ***Час життя*** (Time to Live) займає один байт і означає граничний термін, протягом якого пакет може переміщуватися мережею. Час життя даного пакета вимірюється у секундах і задається джерелом передачі. На маршрутизаторах і в інших вузлах мережі після закінчення кожної секунди з поточного часу життя віднімається одиниця; одиниця віднімається і в тому випадку, коли час затримки менше секунди. Оскільки сучасні маршрутизатори рідко обробляють пакет довше, ніж за одну секунду, то час життя можна вважати таким, що дорівнює максимальній кількості вузлів, які дозволено пройти даному пакету до того, як він дістанеться місця призначення. Якщо параметр часу життя стане нульовим до того, як пакет дістанеться отримувача, цей пакет буде знищений. Час життя можна розглядати як годинниковий механізм самознищення. Значення цього поля змінюється під час обробки заголовка IP-пакета.

Ідентифікатор ***Протокол верхнього рівня*** (Protocol) займає один байт і вказує, якому протоколу верхнього рівня належить інформація, розміщена у полі даних пакета (наприклад, це можуть бути сегменти протоколу TCP, дейтаграми UDP, пакети ICMP або OSPF). Значення ідентифікаторів для різних протоколів наводяться у документі RFC «Assigned Numbers».

Контрольна сума (Header Checksum) займає 2 байта і розраховується лише по заголовку. Оскільки деякі поля заголовка змінюють своє значення в процесі передачі пакета мережею (наприклад, час життя), контрольна сума перевіряється і повторно розраховується в ході кожної обробки IP-заголовка. Контрольна сума – 16 біт – підраховується як доповнення до суми всіх 16-бітових слів заголовка. Під час обчислення контрольної суми значення самого поля «контрольна сума» встановлюється в нуль. Якщо контрольна сума невірна, то пакет буде відкинутий, як тільки помилка буде виявлена.

Поля ***IP-адреса джерела*** (Source IP Address) і IP-адреса призначення (Destination IP Address) мають однакову довжину – 32 біта – і однакову структуру.

Поле ***Опції*** (IP Options) є необов'язковим і використовується зазвичай лише під час налагодження мережі. Механізм опцій надає функції управління, які необхідні або просто корисні за певних ситуацій, проте він не потрібен за звичайних комунікацій. Це поле складається з декількох підполів, кожне з

яких може бути одного з восьми визначених типів. У цих підполях можна вказувати точний маршрут проходження маршрутизаторів, реєструвати маршрутизатори, через які проходить пакет, поміщати дані системи безпеки, а також тимчасові позначки. Оскільки кількість підполів може бути довільною, то в кінці поля ***Опції*** має бути додано кілька байт для вирівнювання заголовка пакета за 32-бітною границею.

Поле ***Вирівнювання*** (Padding) використовується для того, щоб переконатися, що IP-заголовок закінчується за 32-бітною границею. Вирівнювання здійснюється нулями.

Нижче наведена роздруківка значень полів заголовка одного з реальних IP-пакетів, захоплених у мережі Ethernet засобами аналізатора протоколів Microsoft Network Monitor.

IP Version = 4 (0x4)
IP Header Length = 20 (0x14)
IP Service Type = 0 (0x0)
IP Precedence = Routine
IP ...0.... = Normal Delay
IP ....0... = Normal Throughput
IP .....0.. = Normal Reliability
IP Total Length = 54 (0x36)
IP Identification = 31746 (0x7C02)
IP Flags Summary ° 2 (0x2)
IP ..... 0 = Last fragment in datagram
IP ..... 1. = Cannot fragment datagram
IP Fragment Offset = 0 (0x0) bytes
IP Time to Live = 128 (0x80)
IP Protocol = TCP – Transmission Control
IP Checksum =0xEB86
IP Source Address = 194.85.135.75
IP Destination Address = 194.85.135.66
IP Data: Number of data bytes remaining = 34 (0x0022)

3 Таблиці маршрутизації в IP-мережах

Програмні модулі протоколу IP встановлюються на усіх кінцевих станціях і маршрутизаторах мережі. Для просування пакетів вони використовують таблиці маршрутизації.

Приклади таблиць різних типів маршрутизаторів

Структура таблиці маршрутизації стека TCP/IP відповідає загальним принципам побудови таблиць маршрутизації, розглянутим вище. Але важливо зазначити, що вигляд таблиці IP-маршрутизації залежить від конкретної реалізації стека TCP/IP. Наведемо приклад трьох варіантів таблиці маршрутизації, з якими міг би працювати маршрутизатор Ml в мережі, поданій на рис. А.2.

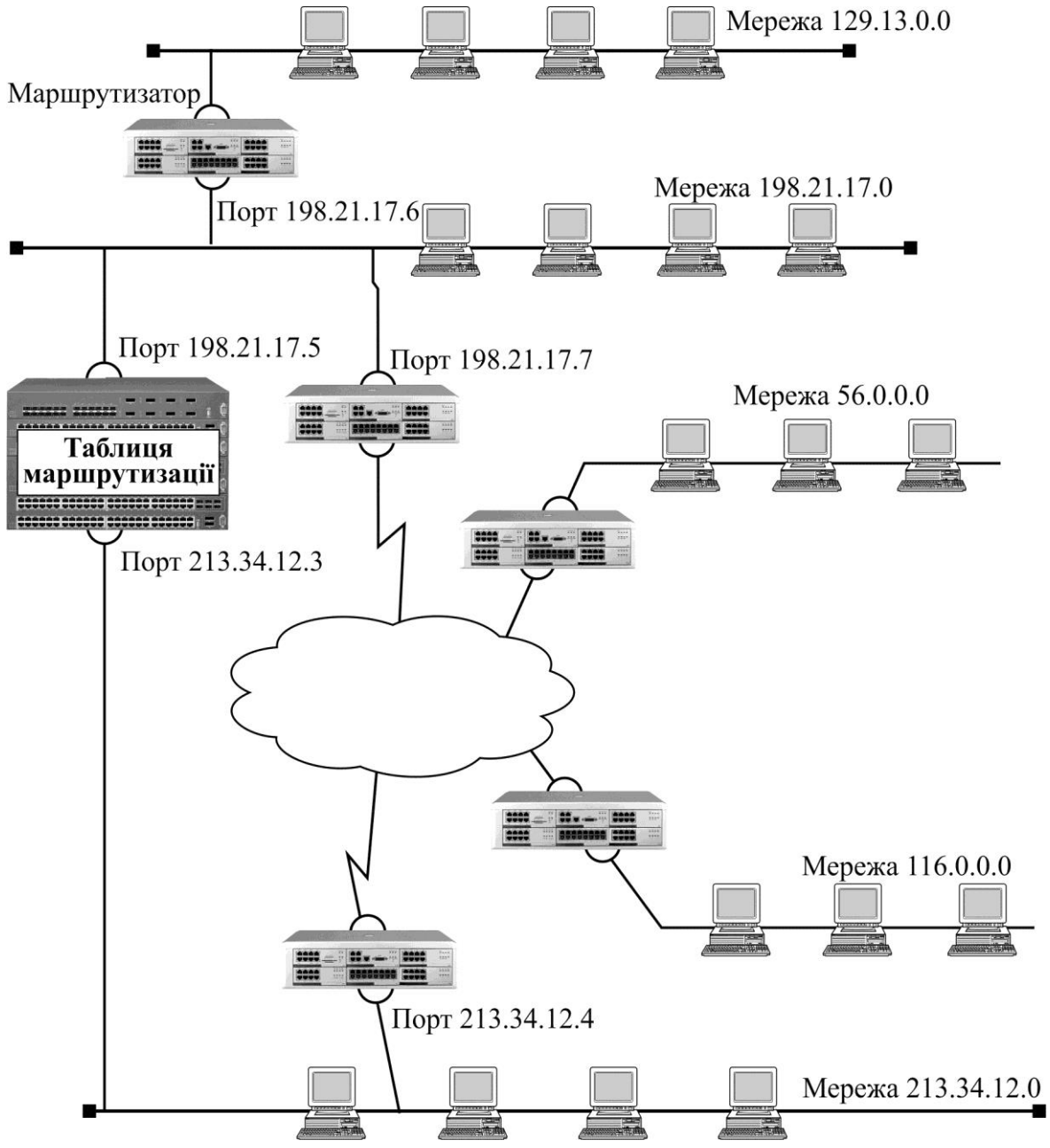


Рисунок А.2 – Приклад мережі, яка маршрутизується

Якщо уявити, що штатний програмний маршрутизатор MPR операційної системи Microsoft Windows NT в даній мережі працює як маршрутизатор Ml, то його таблиця маршрутизації могла б мати такий вигляд (табл. А.1).

Таблиця А.1 – Таблиця програмного маршрутизатора MPR Windows NT

NetworkAddress	Netmask	GatewayAddress	Interface	Metric
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0	1
0.0.0.0	0.0.0.0	198.21.17.7	198.21.17.5	1
56.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	15
116.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	13
129.13.0.0	255.255.0.0	198.21.17.6	198.21.17.5	2
198.21.17.0	255.255.255.0	198.21.17.5	198.21.17.5	1
198.21.17.5	255.255.255.255	127.0.0.1	127.0.0.1	1
198.21.17.255	255.255.255.255	198.21.17.5	198.21.17.5	1
213.34.12.0	255.255.255.0	213.34.12.3	213.34.12.3	1
213.34.12.3	255.255.255.255	127.0.0.1	127.0.0.1	1
213.34.12.255	255.255.255.255	213.34.12.3	213.34.12.3	1
224.0.0.0	254.0.0.0	198.21.17.6	198.21.17.6	1
224.0.0.0	254.0.0.0	213.34.12.3	213.34.12.3	1
255.255.255.255	255.255.255.255	198.21.17.6	198.21.17.6	1

Якщо на місці маршрутизатора М1 встановити апаратний маршрутизатор NetBuilder II компанії 3 Com, то його таблиця маршрутизації для тієї самої мережі може виглядати так, як показано в табл. А.2.

Таблиця А.2 – Таблиця маршрутизації апаратного маршрутизатора NetBuilder II компанії 3 Com

NetBilder#Show – AllRoutes

Total Routes = 5 Total Direkt Network

Destination	Mask	GatewayAddress	Metric	Status	TTL	Source
198.21.17.0	255.255.255.0	198.21.17.5	0	Up	–	Connected
213.34.12.0	255.255.255.0	198.21.17.7	0	Up	–	Connected
56.0.0.0	255.0.0.0	213.34.12.3	14	Up	–	Static
116.0.0.0	255.0.0.0	213.34.12.3	12	Up	–	Static
129.13.0.0	255.255.0.0	198.21.17.6	1	Up	160	RIP

Таблиця А.3 є таблицею маршрутизації для маршрутизатора Ml, реалізованого у вигляді програмного маршрутизатора однієї з версій операційної системи Unix.

Таблиця А.3 – Таблиця маршрутизації Unix-маршрутизатора

Destination	GatewayAddress	Flags	Refcnt	Use	Interface
127.0.0.0	198.21.17.5	UH	1	154	lo0
Default	198.21.17.7	UG	5	43270	le0
198.21.17.0	213.34.12.3	U	35	246876	le0
213.34.12.0	213.34.12.3	U	44	132435	le1
129.13.0.0	198.21.17.6	UG	6	16435	le0
56.0.0.0	213.34.12.4	UG	12	5764	le1
116.0.0.0	213.34.12.4	UG	21	23544	le1

ПРИМІТКА. *Зауважимо, що, оскільки між структурою мережі і таблицею маршрутизації немає однозначної відповідності, то і для кожного з наведених варіантів таблиці можна запропонувати свої «підваріанти», які відрізняються обраним маршрутом до тієї чи іншої мережі. У даному випадку увага концентрується на істотних відмінностях у формі подання маршрутної інформації різними реалізаціями маршрутизаторів.*

Призначення полів таблиці маршрутизації

Незважаючи на досить помітні зовнішні відмінності, в усіх трьох таблицях є всі ті ключові параметри, необхідні для роботи маршрутизатора, які розглядалися раніше у процесі обговорення концепції маршрутизації.

До таких параметрів, безумовно, належать адреса мережі призначення (стовпчики «Destination» в маршрутизаторах NetBuilder і Unix або «Network Address» в маршрутизаторі MPR) і адреса наступного маршрутизатора (стовпчики «Gateway» в маршрутизаторах NetBuilder і Unix або «Gateway Address» в маршрутизаторі MPR).

Третій ключовий параметр – адреса порту, на який необхідно направити пакет, в деяких таблицях вказується прямо (поле «Interface» в таблиці Windows NT), а в деяких – опосередковано. Так, у таблиці Unix-маршрутизатора замість адреси порту задається його умовне найменування – le0 для порту з адресою 198.21.17.5, le1 для порту з адресою 213.34.12.3 та lo0 для внутрішнього порту з адресою 127.0.0.1.

У маршрутизаторі NetBuilder II поле, що позначає вихідний порт у будь-якій формі, взагалі відсутнє. Це пояснюється тим, що адресу вихідного порту завжди можна опосередковано визначити за адресою наступного маршрутизатора. Наприклад, спробуємо визначити за табл. А2 адресу вихідного порту для мережі 56.0.0.0. З таблиці видно, що наступним

маршрутизатором для цієї мережі буде маршрутизатор з адресою 213.34.12.4. Адреса наступного маршрутизатора має належати одній з безпосередньо приєднаних до маршрутизатора мереж, і в даному випадку це мережа 213.34.12.0. Маршрутизатор має порт, приєднаний до цієї мережі, і адресу цього порту 213.34.12.3 ми знаходимо в полі «Gateway» другого рядка таблиці маршрутизації, яка описує безпосередньо приєднану мережу 213.34.12.0. Для безпосередньо приєднаних мереж адресою наступного маршрутизатора завжди є адреса власного порту маршрутизатора. Отже, адреса вихідного порту для мережі 56.0.0 – це адреса 213.34.12.3.

Інші параметри, які можна знайти у поданих версіях таблиці маршрутизації, є необов'язковими для прийняття рішення про шлях проходження пакета.

Наявність або відсутність поля маски в таблиці говорить про те, наскільки сучасний даний маршрутизатор. Стандартним рішенням сьогодні є використання поля маски в кожному записі таблиці, як це зроблено в таблицях маршрутизаторів MPR Windows NT (поле «Netmask») і NetBuilder (поле «Mask»). Опрацювання масок при ухваленні рішення маршрутизаторами розглядатиметься нижче. Відсутність поля маски показує, що або маршрутизатор розрахований на роботу лише з трьома стандартними класами адрес, або він використовує для всіх записів одну і ту саму маску, що знижує гнучкість маршрутизації.

Метрика, як видно з прикладу таблиці Unix-маршрутизатора, є необов'язковим параметром. В інших двох таблицях це поле є, але воно використовується лише як ознака безпосередньо підключеної мережі. Дійсно, якщо в таблиці маршрутизації кожна мережа призначення згадана лише один раз, то поле метрики не братиметься до уваги під час вибору маршруту, оскільки вибір відсутній. А ось ознака безпосередньо підключеної мережі маршрутизатора потрібна, оскільки пакет для цієї мережі обробляється особливим способом, – він не передається наступному маршрутизатору, а відправляється вузлу призначення. Тому метрика 0 для маршрутизатора NetBuilder або 1 для маршрутизатора MPR просто повідомляє маршрутизатору, що ця мережа безпосередньо підключена до його порту, а інше значення метрики відповідає видаленій мережі. Вибір значення метрики для безпосередньо підключеної мережі є досить довільним, головне, щоб метрика віддаленої мережі відраховувалася з огляду на обране початкове значення. В Unix-маршрутизаторі використовується поле ознак, де прапорець G відзначає видалену мережу, а його відсутність – безпосередньо підключену.

Але існують ситуації, коли маршрутизатор має обов'язково зберігати значення метрики для запису про кожну видалену мережу. Ці ситуації виникають, коли записи у таблиці маршрутизації є результатом роботи деяких протоколів маршрутизації, наприклад, протоколу RIP. У таких протоколах нова інформація про будь-яку видалену мережу порівнюється з наявною в таблиці, і якщо метрика нової інформації краще наявної, то новий запис витісняє наявний. У таблиці Unix-маршрутизатора поле метрики відсутнє, і це означає, що він не використовує протокол RIP.

Прапорці записів присутні тільки в таблиці Unix-маршрутизатора. Вони описують характеристики запису.

• U – показує, що маршрут активний і працездатний. Аналогічний сенс має поле «Status» в маршрутизаторі NetBuilder.

• H – ознака специфічного маршруту до певного хосту. Маршрут до всієї мережі, до якої належить даний хост, може відрізнятися від даного маршруту.

• G – означає, що маршрут пакета проходить через проміжний маршрутизатор (gateway). Відсутність цього прапорця показує безпосередньо підключену мережу.

• D – означає, що маршрут отриманий з повідомлення Redirect (перенаправлення) протоколу ICMP. Ця ознака може бути присутньою лише в таблиці маршрутизації кінцевого вузла. Ознака показує, що кінцевий вузол в деякій попередній передачі пакета обрав не найбільш раціональний наступний маршрутизатор на шляху до даної мережі, і цей маршрутизатор за допомогою протоколу ICMP повідомив, що усі наступні пакети до даної мережі слід відправляти через інший маршрутизатор. Протокол ICMP може надсилати повідомлення лише вузлу-відправнику, тому у проміжного маршрутизатора ця ознака зустрітися не може. Ознака ніяк не впливає на процес маршрутизації, вона лише вказує адміністратору джерело появи запису. У таблиці Unix-маршрутизатора використовуються ще два поля, які мають довідкове значення. Поле «Refcnt» показує, скільки разів на даний маршрут посилалися під час просування пакетів. Поле «Use» показує кількість пакетів, переданих даним маршрутом.

У таблиці маршрутизатора NetBuilder також є два довідкових поля. Поле часу життя «TTL» (Time To Live) має сенс для динамічних записів, які мають обмежений термін життя. Поточне значення поля показує остаточний термін життя запису в секундах. Поле «Source» відображає джерело появи запису в таблиці маршрутизації. Хоча це поле є не в усіх маршрутизаторах, але практично для всіх маршрутизаторів існують три основних джерела появи запису в таблиці.

Джерела і типи записів в таблиці маршрутизації

Першим джерелом є програмне забезпечення стека TCP/IP. Під час ініціалізації маршрутизатора це програмне забезпечення автоматично заносить в таблицю кілька записів, внаслідок чого створюється мінімальна таблиця маршрутизації.

Це, по-перше, записи про безпосередньо підключені мережі і маршрутизатори за замовчуванням, інформація про які з'являється у стеку в процесі ручного конфігурування інтерфейсів комп'ютера або маршрутизатора. До таких записів у наведених прикладах належать записи про мережі 213.34.12.0 та 198.21.17.0, а також запис про маршрутизатор за замовчуванням – default в Unix-маршрутизаторі і 0.0.0.0 в маршрутизаторі MPR Windows NT. У наведеному прикладі таблиці для маршрутизатора NetBuilder маршрутизатор за замовчуванням не використовується, отже, під час надходження пакета з адресою призначення, яка відсутня у таблиці маршрутизації, даний пакет буде відкинутий.

По-друге, програмне забезпечення автоматично заносить у таблицю маршрутизації записи про адреси особливого призначення. У наведених прикладах таблиця маршрутизатора MPR Windows NT містить найбільш повний набір подібних записів. Кілька записів у цій таблиці пов'язані з особливою адресою 127.0.0.0 (loopback), яка використовується для локального тестування стека TCP/IP. Пакети, направлені в мережу з номером 127.0.0.0, не передаються протоколом IP на канальний рівень для подальшої передачі в мережу, а повертаються в джерело – локальний модуль IP. Записи з адресою 224.0.0.0 потрібні для опрацювання групових адрес (multicast address). Крім того, у таблицю можуть бути занесені адреси, призначені для опрацювання широкомовних розсилок (наприклад, записи 8 і 11 містять адресу відправлення широкомовного повідомлення у відповідних підмережах, а останній запис у таблиці – адреса обмеженої широкомовної розсилки повідомлення). Зауважимо, що в деяких таблицях записи про особливі адреси взагалі відсутні.

Ще одне джерело появи запису в таблиці є адміністратор, який безпосередньо формує запис за допомогою деякої системної утиліти, наприклад, програми route, наявної в операційних системах Unix і Windows NT. В апаратних маршрутизаторах також завжди є команда для ручного задання записів таблиці маршрутизації. Задані вручну записи завжди є статичними, тобто не мають терміну закінчення життя. Ці записи можуть бути як постійними, тобто зберігаються в ході перезавантаження маршрутизатора, так і тимчасовими, що зберігаються в таблиці лише до вимикання пристрою.

Часто адміністратор вручну заносить запис default про маршрутизатор за замовчуванням. Також у таблицю маршрутизації може бути внесений запис про специфічний для вузла маршрут. Специфічний для вузла маршрут містить замість номера мережі повну IP-адресу, тобто адресу, що має ненульову інформацію не лише в полі номера мережі, а й у полі номера вузла. Припускається, що для такого кінцевого вузла маршрут має обиратися не так, як для всіх інших вузлів мережі, до якої він належить. У разі коли в таблиці є різні записи про просування пакетів для всієї мережі і її окремого вузла, з надходженнямі пакета, адресованого вузлу, маршрутизатор віддасть перевагу запису з повною адресою вузла.

І, нарешті, третім джерелом записів можуть бути протоколи маршрутизації, такі як RIP або OSPF. Такі записи завжди є динамічними, тобто мають обмежений термін життя. Програмні маршрутизатори Windows NT і Unix не показують джерело появи того чи іншого запису в таблиці, а маршрутизатор NetBuilder використовує для цієї мети поле «Source». У наведеному в табл. А.2 прикладі перші два записи створені програмним забезпеченням стека на підставі даних про конфігурацію портів маршрутизатора – це показує ознаку «Connected». Наступні два записи позначені як «Static», що вказує на те, що їх ввів вручну адміністратор. Останній запис є наслідком роботи протоколу RIP, тому в його полі «TTL» є значення 160.

4 Маршрутизація без використання масок

Розглянемо на прикладі IP-мережі (рис. А.3) алгоритм роботи засобів мережного рівня по просуванню пакета у складену мережу. При цьому вважатимемо, що всі вузли мережі, яка розглядається в прикладі, мають адреси, засновані на класах без використання масок. Особливу увагу приділимо взаємодії протоколу IP з протоколами дозволу адрес ARP і DNS.

1. Отже, нехай користувач комп'ютера cit.dol.ru, що знаходиться в мережі Ethernet і має IP-адресу 194.87.23.17 (адреса класу С), звертається за протоколом FTP до комп'ютера sl.msk.su, що належить іншій мережі Ethernet і має IP-адресу 142.06,13.14 (адреса класу В):

> ftp sl.msk.su

Модуль FTP упаковує своє повідомлення в сегмент транспортного протоколу TCP, який в свою чергу поміщає свій сегмент в пакет протоколу IP. У заголовку IP-пакета має бути вказана IP-адреса вузла призначення. Оскільки

користувач комп'ютера cit.dol.ru використовує символьне ім'я комп'ютера sl.msk.su, то стек TCP/IP має визначити IP-адресу вузла призначення самостійно.

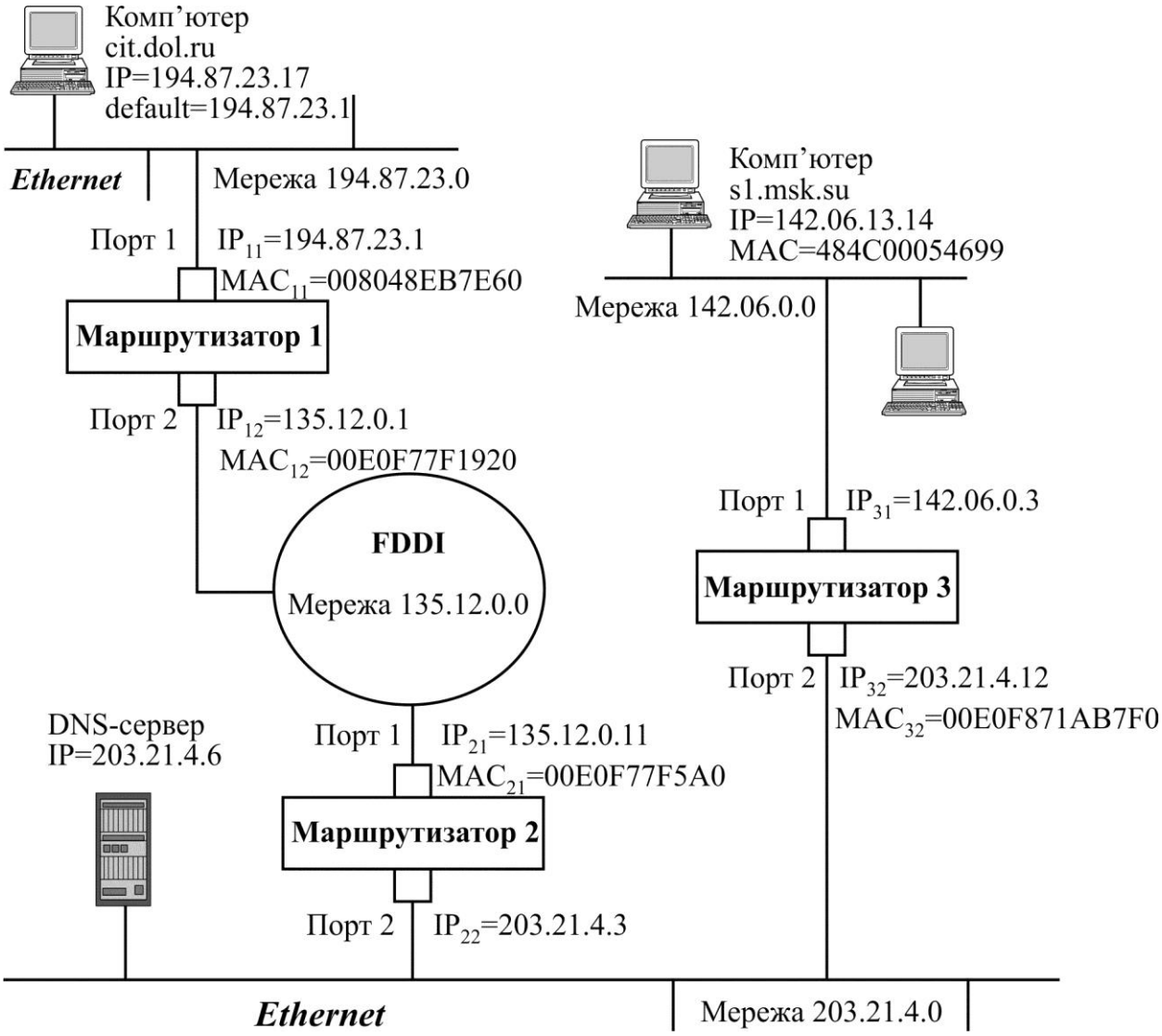


Рисунок А.3 – Приклад взаємодії комп'ютерів через мережу

У процесі конфігурування стека TC/IP у комп'ютері cit.dol.ru було поставлено його власну IP-адресу, IP-адресу маршрутизатора за замовчуванням і IP-адресу DNS-сервера. Модуль IP може зробити запит до сервера DNS, але зазвичай спочатку проглядається локальна таблиця відповідності символьних імен і IP-адрес. Така таблиця зберігається найчастіше у вигляді текстового файла простої структури – кожен його рядок містить запис про одне символьне ім'я та його IP-адресу. В ОС Unix такий файл традиційно носить ім'я hosts і знаходиться у каталозі /etc.

2. Вважатимемо, що комп'ютер dt.dol.ru має файл hosts, а в ньому є рядок

142.06.13.14 sl.msk.su

Отже, дозвіл імені виконується локально, і протокол IP може тепер формувати IP-пакети з адресою призначення 142.06.13.14 для взаємодії з комп'ютером sl.msk.su.

3. Модуль IP комп'ютера cit.dol.ru перевіряє, чи потрібно маршрутизувати пакети з адресою 142.06.13.14. Оскільки адреса мережі призначення (142.06.0.0) не збігається з адресою (194.87.23.0) мережі, якій належить комп'ютер-відправник, то маршрутизація необхідна.

4. Комп'ютер cit.dol.ru починає формувати кадр Ethernet для відправлення IP-пакета маршрутизатору за замовчуванням, IP-адреса якого відома – 194.87.23.1, але невідома MAC-адреса, необхідна для пересування кадра в локальній мережі. Для визначення MAC-адреси маршрутизатора протокол IP звертається до протоколу ARP, який переглядає ARP-таблицю. Якщо останнім часом комп'ютер cit.dol.ru виконував будь-які міжмережні обміни, то швидше за все шуканий запис, що містить відповідність між IP- і MAC-адресами маршрутизатора за замовчуванням, вже знаходиться у кеш-таблиці протоколу ARP. Нехай в даному випадку потрібний запис був знайдений саме у кеш-таблиці:

194.87.23.1 008048EB7E60

Позначимо знайдену MAC-адресу 008048EB7E60 відповідно до номера маршрутизатора та його порта через MAC_{11}.

5. У результаті комп'ютер cit.dol.ru відправляє по локальній мережі пакет, упакований в кадр Ethernet, який має такі поля:



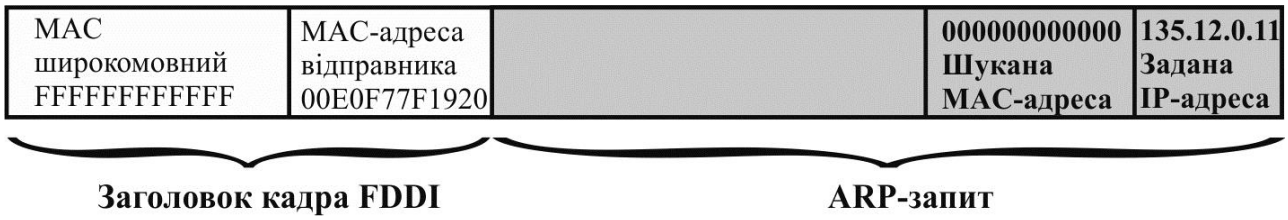
6. Кадр приймається портом 1 маршрутизатора 1 згідно з протоколом Ethernet, оскільки MAC-вузол цього порту розпізнає свою адресу MAC_{11}. Протокол Ethernet вилучає з цього кадра IP-пакет і передає його програмному забезпеченню маршрутизатора, який реалізує протокол IP. Протокол IP вилучає з пакета адресу призначення 142.06.13.14 і переглядає записи своєї таблиці маршрутизації. Нехай маршрутизатор 1 має у своїй таблиці маршрутизації запис

142.06.0.0 135.12.0.11 2,

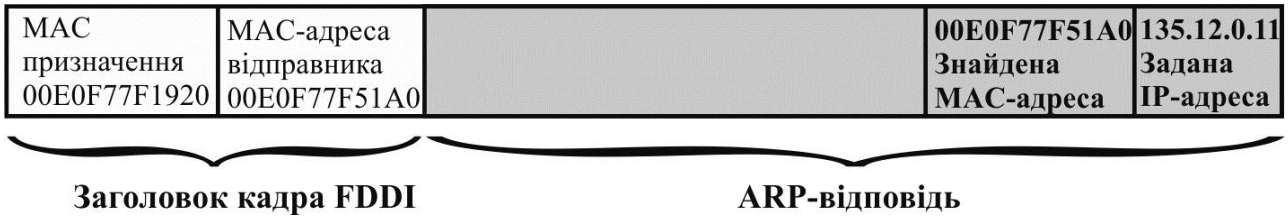
який говорить про те, що пакети для мережі 142.06.0.0 потрібно передавати маршрутизатору 135.12.0.11, який знаходиться в мережі, підключеній до порту 2 маршрутизатора 1.

7. Маршрутизатор 1 переглядає параметри порту 2 і виявляє, що до нього підключена мережа FDDI. Оскільки мережа FDDI має значення MTU більше,

ніж мережа Ethernet, то фрагментація IP-пакета не потрібна. Тому маршрутизатор 1 формує кадр формату FDDI. На цьому етапі модуль IP має визначити MAC-адресу наступного маршрутизатора за відомою IP-адресою 135.12.0.11. Для цього він звертається до протоколу ARP. Припустимо, що потрібного запису в кеш-таблиці не виявилося, тоді в мережу FDDI відправляється широкомовний ARP-запит, який містить поряд з іншою таку інформацію.



Порт 1 маршрутизатора 2 розпізнає свою IP-адресу і надсилає ARP-відповідь за адресою вузла, що запросив:



Тепер, знаючи MAC-адресу наступного маршрутизатора 00E0F77F51A0, маршрутизатор 1 надсилає кадр FDDI у напрямку маршрутизатора 2. Зауважимо, що у полі IP-адреси призначення жодних змін не відбулося.



8. Аналогічно діє модуль IP на маршрутизаторі 2. Отримавши кадр FDDI, він відкидає його заголовок, а із заголовка IP вилучає IP-адресу мережі призначення і переглядає свою таблицю маршрутизації. Там він може знайти запис про конкретну мережу призначення:

142.06.0.0 203.21.4.12 2

або за відсутності такого запису буде використаний запис про маршрутизатор за замовчуванням:

default 203.21.4.12 2.

Визначивши IP-адресу наступного маршрутизатора 203.21.4.12, модуль IP формує кадр Ethernet для передачі пакета маршрутизатору 3 по мережі Ethernet. За допомогою протоколу ARP він знаходить MAC-адресу цього

маршрутизатора і поміщає його в заголовок кадра. IP-адреса вузла призначення, природно, залишається незмінною.



9. Нарешті, після того, як пакет надійшов у маршрутизатор мережі призначення – маршрутизатор 3, – з'являється можливість передачі цього пакета комп'ютеру призначення. Маршрутизатор 3 визначає, що пакет потрібно передати в мережу 142.06.0,0, яка безпосередньо підключена до його першого порту. Тому він посилає ARP-запит по мережі Ethernet з IP-адресою комп'ютера sl.msk.su. ARP-відповідь містить MAC-адресу кінцевого вузла, який передає модуль IP канальному протоколу для формування кадра Ethernet:



10. Мережний адаптер комп'ютера sl.msk.su захоплює кадр Ethernet, виявляє збіг MAC-адреси, що міститься в заголовку, зі своєю власною адресою і направляє його модулю IP. Після аналізу полів IP-заголовка з пакета вилучаються дані, які, в свою чергу, містять повідомлення протоколу, що знаходиться вище. Оскільки у даному прикладі розглядається обмін даними за протоколом FTP, який використовує TCP як транспортний протокол, то в полі даних IP-пакета знаходиться TCP-сегмент. Визначивши з TCP-заголовка номер порту, модуль IP переправляє сегмент у відповідну чергу, з якої даний сегмент надійде до програмного модуля FTP-сервера.

5 Маршрутизація з використанням масок

Використання масок для структуризації мережі

Алгоритм маршрутизації ускладнюється, коли в систему адресації вузлів вносяться додаткові елементи – маски. У чому причина відмови від методу адресації, заснованого на класах, який добре себе зарекомендував протягом багатьох років? Таких причин декілька, і одна з них – потреба в структуризації мереж.

Часто адміністратори мереж відчувають незручності через те, що кількість централізовано виділених їм номерів мереж недостатня для того, щоб структурувати мережу належним чином, наприклад, розмістити усі комп'ютери, що слабко взаємодіють, по різних мережах. У такій ситуації можливі два шляхи. Перший пов'язаний з отриманням від InterNIC або постачальника послуг Internet додаткових номерів мереж. Другий спосіб, що застосовується частіше, пов'язаний з використанням технології масок, яка дозволяє розділяти одну мережу на декілька.

Припустимо, адміністратор отримав у своє розпорядження адресу класу B: 129.44.0.0. Він може організувати мережу з великою кількістю вузлів, номери яких він братиме з діапазону 0.0.0.1–0.0.255.254 (з урахуванням того, що адреси з одних нулів і одних одиниць мають спеціальне призначення і не підходять для адресації вузлів). Але йому не потрібна одна велика неструктурована мережа, виробнича необхідність диктує адміністратору інше рішення, відповідно до якого мережа має бути розділена на три окремих підмережі, при цьому трафік у кожної з них має бути надійно локалізований. Це дозволить легше діагностувати мережу і проводити в кожній з підмереж особливу політику безпеки.

Подивимося, як вирішується ця проблема шляхом використання механізму масок.

Отже, номер мережі, який адміністратор отримав від постачальника послуг, – 129.44.0.0 (10000001 00101100 00000000 00000000). Як маски було обрано значення 255.255.192.0 (11111111111111111111000000 00000000). Після накладення маски на дану адресу кількість розрядів, що інтерпретуються як номер мережі, збільшилася з 16 (стандартна довжина поля номера мережі для класу B) до 18 (кількість одиниць у масці), тобто адміністратор отримав можливість використовувати для нумерації підмереж два додаткових біти. Це дозволяє йому зробити з одного, централізовано заданого йому номера мережі, чотири:

129.44.0.0 (10000001 00101100 00000000 00000000)
129.44.64.0 (10000001 00101100 01000000 00000000)
129.44.128.0 (10000001 00101100 10000000 00000000)
129.44.192.0 (10000001 00101100 11000000 00000000)

Два додаткових останніх біта в номері мережі часто інтерпретуються як номери підмереж (subnet), і тоді чотири перерахованих вище підмережі мають номери 0 (00), 1 (01), 2 (10) і 3 (11) відповідно.

ПРИМІТКА. *Деякі програмні й апаратні маршрутизатори не підтримують номери підмереж, які складаються або тільки з одних нулів, або тільки з одних одиниць. Наприклад, для деяких типів обладнання номер мережі 129.44.0.0 з маскою 255.255.192.0, використаний у нашому прикладі, виявиться неприпустимим, оскільки в цьому випадку розряди в полі номера підмережі мають значення 00. З аналогічних міркувань неприпустимим може виявитися і номер мережі 129.44.192.0 з тим самим значенням маски. Тут номер підмережі складається лише з одиниць. Але сучасні маршрутизатори вільні від цих обмежень. Тому, приймаючи рішення про використання механізму масок, необхідно з'ясувати характеристики того устаткування, яке ви маєте в своєму розпорядженні, щоб відповідно налаштувати маршрутизатори та комп'ютери мережі.*

Внаслідок використання масок була запропонована нижченаведена схема розподілу адресного простору (рис. А.4).

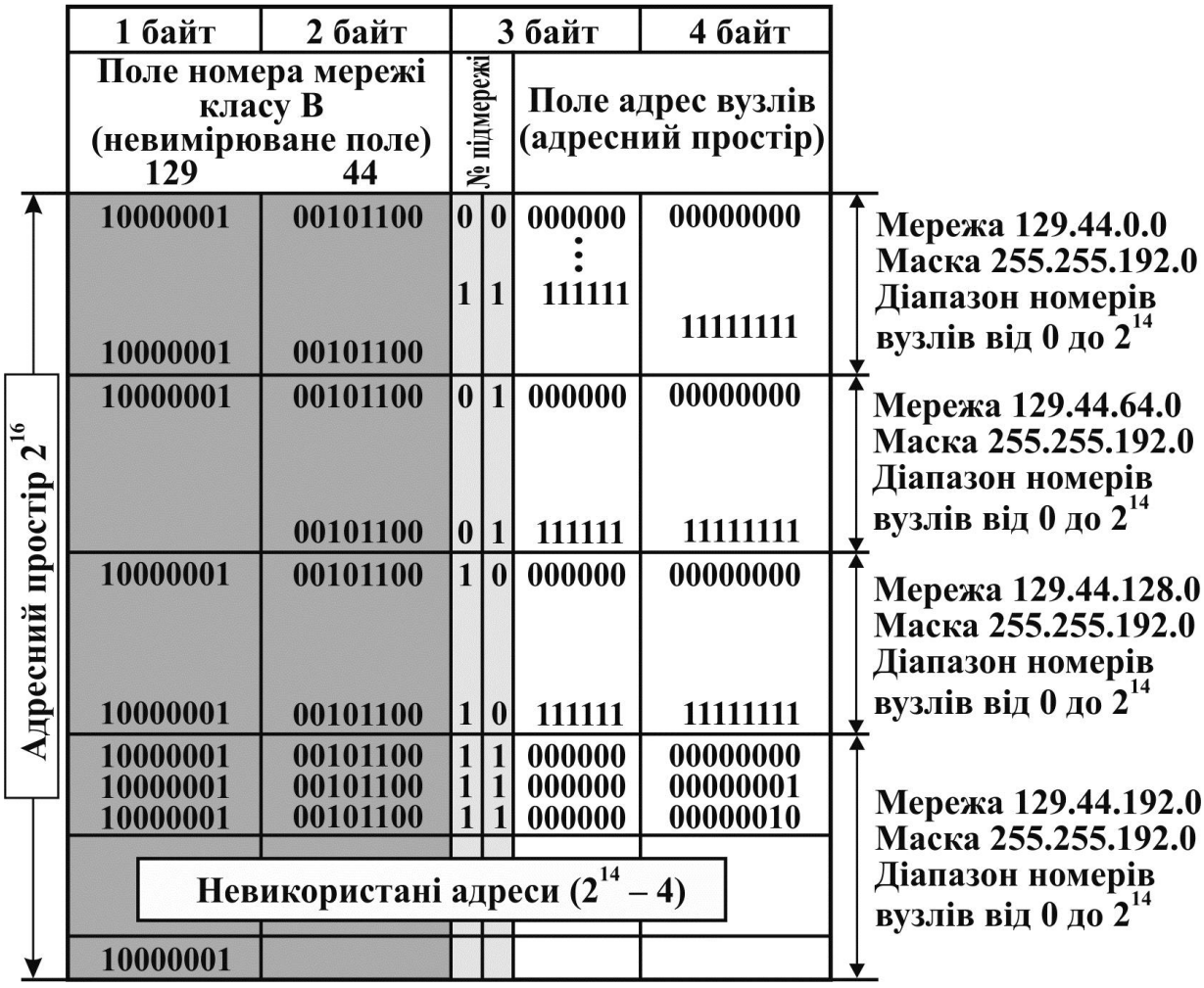


Рисунок А.4 – Поділ адресного простору мережі класу В 129.44.0.0 на чотири рівні частини шляхом використання масок однакової довжини 255.255.192.0

Мережа, що вийшла внаслідок проведеної структуризації, показана на рис. А.5. Весь трафік у внутрішню мережу 129.44.0.0, що направляється із зовнішньої мережі, надходить через маршрутизатор Ml. З метою структуризації інформаційних потоків у внутрішній мережі встановлений додатковий маршрутизатор М2.

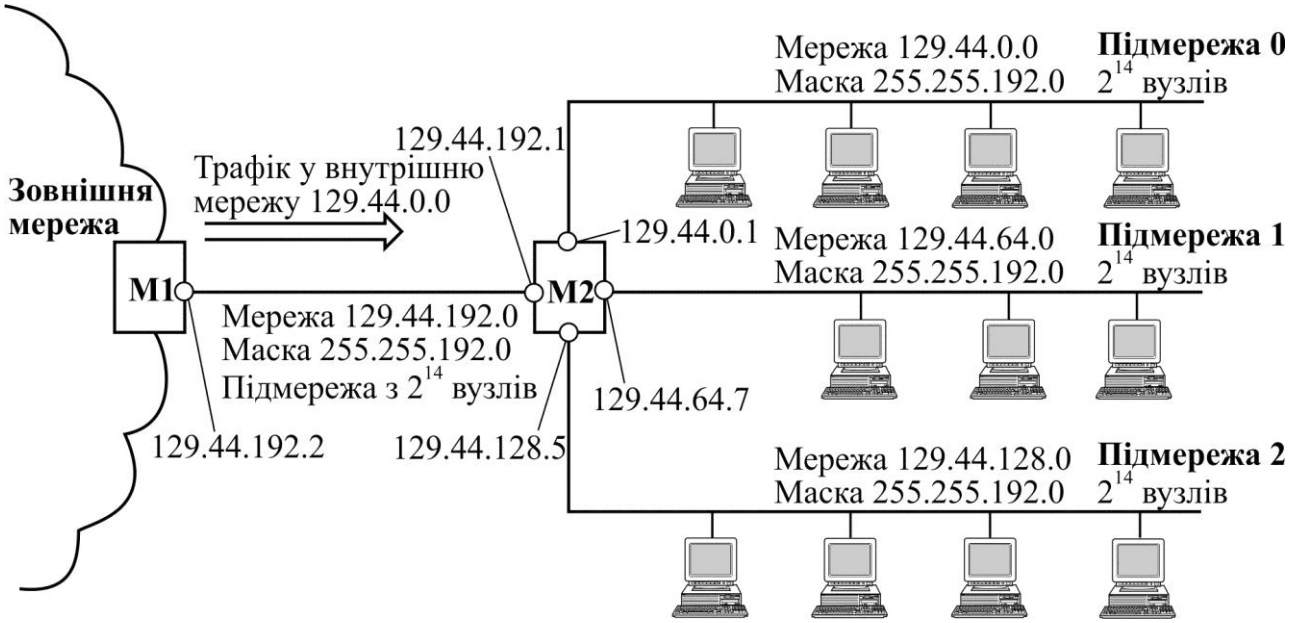


Рисунок А.5 – Маршрутизація з використанням масок однакової довжини

Усі вузли були розподілені за трьома різними мережами, яким були привласнені номери 129.44.0.0, 129.44.64.0 та 129.44.128.0 і маски однакової довжини – 255.255.192.0. Кожна з новостворених мереж була підключена до відповідно сконфігурованих портів внутрішнього маршрутизатора М2. Крім того, ще одна мережа (номер 129.44.192.0, маска 255.255.192.0) була виділена для створення з'єднання між зовнішнім і внутрішнім маршрутизаторами. Особливо відзначимо, що у цій мережі для адресації вузлів були зайняті лише дві адреси 129.44.192.1 (порт маршрутизатора М2) і 129.44.192.2 (порт маршрутизатора Ml), ще дві адреси 129.44.192.0 і 129.44.192.255 є особливими адресами. Отже, величезна кількість вузлів $\left(2^{14}-4\right)$ у цій підмережі ніяк не використовуються.

Ззовні мережа як і раніше виглядає, як єдина мережа класу В, а на місцевому рівні – це повноцінна складена мережа, до якої входять три окремі мережі. Загальний трафік, що надходить, розділяється місцевим маршрутизатором М2 між цими мережами відповідно до таблиці маршрутизації. (Зауважимо, що поділ великої мережі, що має одну адресу старшого класу, наприклад А чи В, за допомогою масок несе в собі ще одну перевагу порівняно

з використанням декількох адрес стандартних класів для мереж меншого розміру, наприклад, С. Це дозволяє приховати внутрішню структуру мережі підприємства від зовнішнього спостереження і тим підвищити її безпеку.)

Розглянемо, як змінюється робота модуля IP, коли стає необхідним враховувати наявність масок.

По-перше, у кожному записі таблиці маршрутизації з'являється нове поле – поле маски.

По-друге, змінюється алгоритм визначення маршруту За таблицею маршрутизації. Після того як IP-адреса вилучається з чергового отриманого IP-пакета, необхідно визначити адресу наступного маршрутизатора, на який слід передати пакет з цією адресою. Модуль IP послідовно переглядає усі записи таблиці маршрутизації. З кожним записом проводяться такі дії.

• Маска М, що міститься в даному записі, накладається на IP-адресу вузла призначення, вилучену з пакета.

• Отримане в результаті число є номером мережі призначення пакета, який опрацьовується. Воно порівнюється з номером мережі, поміщеним у даному записі таблиці маршрутизації.

• Якщо номери мереж збігаються, то пакет передається маршрутизатору, адреса якого поміщена у відповідному полі даного запису.

Тепер розглянемо цей алгоритм на прикладі маршрутизації пакетів у мережі, зображеній на рис. А.5. Усі маршрутизатори зовнішньої мережі, зустрівши пакети з адресами, що починаються з 129.44, інтерпретують їх як адреси класу В і направляють маршрутами, що ведуть до маршрутизатора Ml. Маршрутизатор Ml в свою чергу направляє весь вхідний трафік мережі 129.44.0.0 на маршрутизатор М2, а саме на його порт 129.44.192.1.

Маршрутизатор М2 опрацьовує всі пакети, що надійшли на нього, відповідно до таблиці маршрутизації (табл. А.4).

Перші чотири записи у таблиці відповідають внутрішнім підмережам, безпосередньо підключеним до портів маршрутизатора М2.

Запис 0.0.0.0 з маскою 0.0.0.0 відповідає маршруту за замовчуванням. Дійсно, будь-яку адресу в пакеті, що надійшов, після накладення на нього маски 0.0.0.0 дасть адресу мережі 0.0.0.0, яка збігається з адресою, зазначеною у записі. Маршрутизатор виконує порівняння з адресою 0.0.0.0 в останню чергу, в тому випадку коли адреса, яка надійшла, не дала збігу з жодним записом у таблиці, що відрізняється від 0.0.0.0. Записів з адресою 0.0.0.0 у таблиці маршрутизації може бути кілька. У цьому випадку маршрутизатор передає пакет на усіх вищезазначені маршрути.

Таблиця А.4 – Таблиця маршрутизатора М2 у мережі з масками однакової довжини

Номер мережі	Маска	Адреса наступного маршрутизатора	Адреса порта	Відстань
129.44.0.0	255.255.128.0	129.44.0.1	129.44.0.1	Підключена
129.44.128.0	255.255.192.0	129.44.128.3	129.44.128.3	Підключена
129.44.192.0	255.255.255.248	129.44.192.1	129.44.191.1	Підключена
129.44.224.0	255.255.224.0	129.44.224.5	129.44.224.5	Підключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–

Нехай, наприклад, з маршрутизатора Ml на порт 129.44.192.1 маршрутизатора М2 надходить пакет з адресою призначення 129.44.78.200. Модуль IP починає послідовно переглядати усі рядки таблиці, доти, доки не знайде збіги номера мережі в адресі призначення і в рядку таблиці. Маска з першого рядка 255.255.192.0 накладається на адресу 129.44,78.200, в результаті виходить номер мережі 129.44.64.0.

У двійковому вигляді:

10000001.00101100.01001110.11001000

11111111.11111111.11000000.00000000

- -

10000001.00101100.01000000.00000000

Отриманий номер 129.44.64.0 порівнюється з номером мережі в першому рядку таблиці 129.44.0.0. Оскільки вони не збігаються, то відбувається перехід до наступного рядка. Тепер вилучається маска з другого рядка (в даному випадку вона має таке саме значення, але в загальному випадку це зовсім не обов’язково) і накладається на адресу призначення пакета 129.44.78.200. Зрозуміло, що через збіг довжини масок буде отриманий той самий номер мережі 129.44.64.0. Цей номер збігається з номером мережі у другому рядку таблиці, а значить, знайдено маршрут для даного пакета, – він має бути відправлений на порт маршрутизатора 129.44.64.7 в мережу, безпосередньо підключену до даного маршрутизатора.

Ще один приклад.

IP-адреса 129.44.141.15 (10000001 00101100 10001101 00001111), яка при використанні класів ділиться на номер мережі 129.44.0.0 і номер вузла 0.0.141.15, тепер, у ході використання маски 255.255.192.0, інтерпретуватиметься як пара: 129.44.128.0 – номер мережі, 0.0.13.15 – номер вузла.

Використання масок змінної довжини

У попередньому прикладі використання масок (рис. А.4 і А.5) усі підмережі мають однакову довжину поля номера мережі – 18 двійкових розрядів, і, отже, для нумерації вузлів у кожній з них відводиться по 14 розрядів. Тобто всі мережі дуже великі і мають однаковий розмір. Проте в цьому випадку, як і у багатьох інших, більш ефективним було б розбиття мережі на підмережі різного розміру. Зокрема, велика кількість вузлів, цілком бажана для користувальницької підмережі, явно є надмірною для підмережі, яка зв'язує два маршрутизатори за схемою «точка–точка». У цьому випадку потрібні лише дві адреси для адресації двох портів сусідніх маршрутизаторів. У попередньому ж прикладі для даної допоміжної мережі Ml – М2 був використаний номер, який дозволяє адресувати 2^{14} вузлів, що робить таке рішення неприйнятним надмірністю. Адміністратор може раціональніше розподілити адресний простір, що у в його розпорядженні, за допомогою масок змінної довжини.

На рис. А.6 наведено приклад розподілу адресного простору, за якого надмірність IP-адрес може бути зведена до мінімуму. Половину з наявних адрес $\left(2^{15}\right)$ було виділено для створення мережі з адресою 129.44.0.0 і маскою 255.255.128.0. Наступна порція адрес, що становить чверть усього адресного простору $\left(2^{14}\right)$, була призначена для мережі 129.44.128.0 з маскою 255.255.192.0. Потім у просторі адрес був «вирізаний» невеликий фрагмент для створення мережі, призначеної для зв'язування внутрішнього маршрутизатора М2 із зовнішнім маршрутизатором Ml.

В IP-адресі такої виродженої мережі для поля номера вузла як мінімум мають бути відведені два двійкових розряди. З чотирьох можливих комбінацій номерів вузлів: 00, 01,10 і 11 два номери мають спеціальне призначення і не можуть бути присвоєні вузлам, але ті два 10 і 01, що залишилися, дозволяють адресувати порти маршрутизаторів.

У нашому прикладі мережа була обрана з запасом на 8 вузлів. Поле номера вузла у такому разі має 3 двійкових розряди, маска в десятковій нотації має вигляд 255.255.255.248, а номер мережі, як видно з рис. А.6, дорівнює у даному конкретному випадку 129.44.192.0.

Якщо ця мережа є локальною, то на ній можуть розташовуватися чотири вузла окрім двох портів маршуртізаторов.

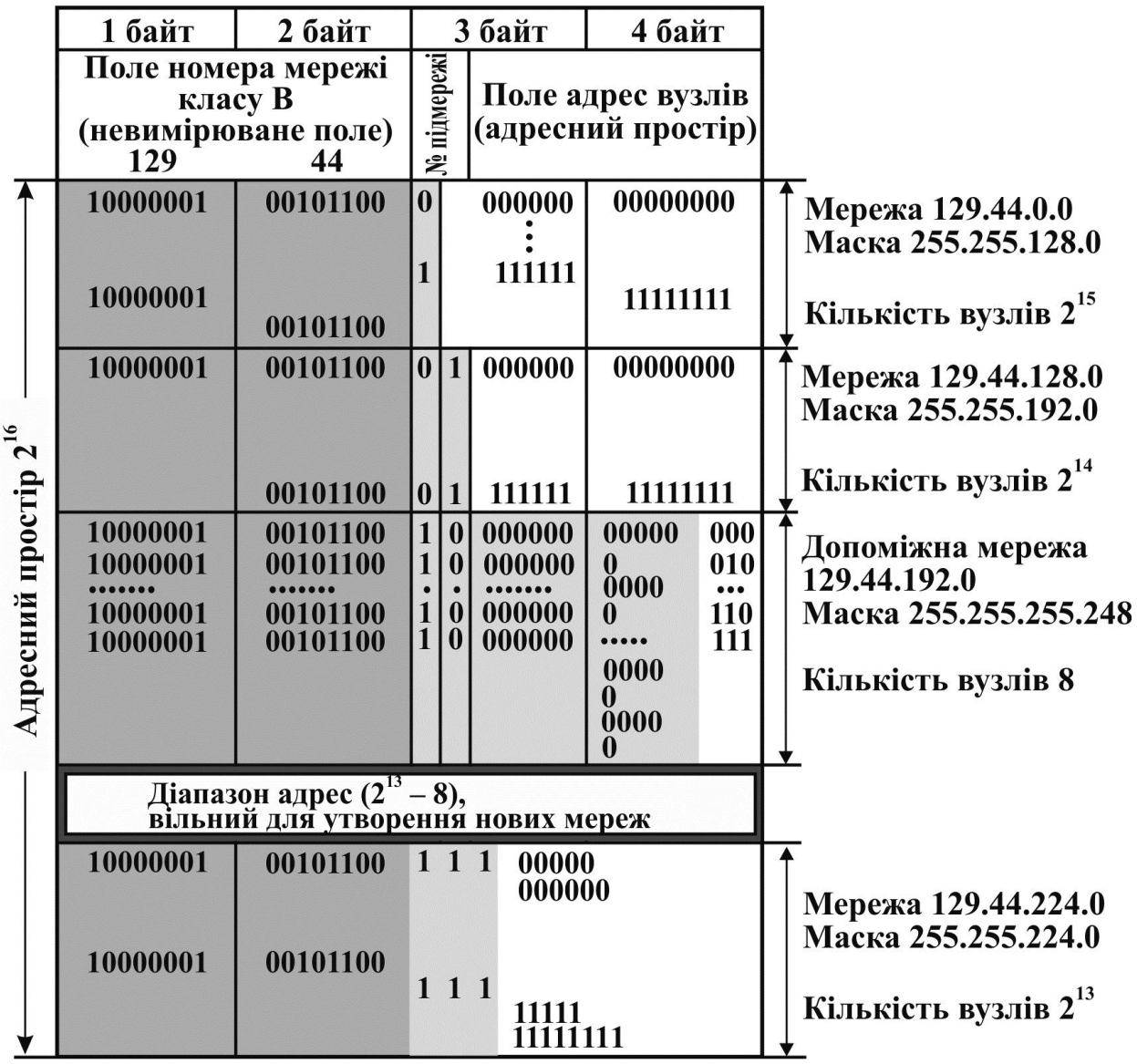


Рисунок А.6 – Розподіл адресного простору мережі класу В 129.44.0.0 на мережі різного розміру шляхом використання масок змінної довжини

ПРИМІТКА *Зауважимо, що глобальним зв'язкам між маршрутизаторами типу «точка–точка» не обов'язково давати IP-адреси, оскільки до такої мережі не можуть підключатися жодні інші вузли, крім двох портів маршрутизаторів. Проте найчастіше таким виродженим мережам усе ж дають IP-адреси. Це робиться, наприклад, для того, щоб приховати внутрішню структуру мережі і звертатися до неї за однією адресою вхідного порта маршрутизатора, у даному прикладі – за адресою 129.44.192.1. Крім того, ця адреса може знадобитися в ходів тунелювання немаршрутизованих протоколів в IP-пакети, що буде розглянуто нижче.*

Адресний простір, що залишився, адміністратор може «нарізати» на різну кількість мереж різного обсягу, залежно від своїх потреб. А із залишків пулу $\left(2^{14}-4\right)$ адрес адміністратор може утворити ще одну досить велику мережу з кількістю вузлів 2^{13}. При цьому вільними залишаться майже стільки ж адрес$\left(2^{13}-4\right)$, які також можуть бути використані для створення нових мереж. Наприклад, з цього «залишку» можна утворити 31 мережу, кожна з яких дорівнює розміру стандартної мережі класу С, і до того ж ще кілька мереж меншого розміру. Зрозуміло, що розбиття може бути іншим, але в будь-якому випадку за допомогою масок змінного розміру адміністратор завжди має можливість набагато раціональніше використовувати всі наявні у нього адреси.

На рис. А.7 показана схема мережі, структурованої за допомогою масок змінної довжини.

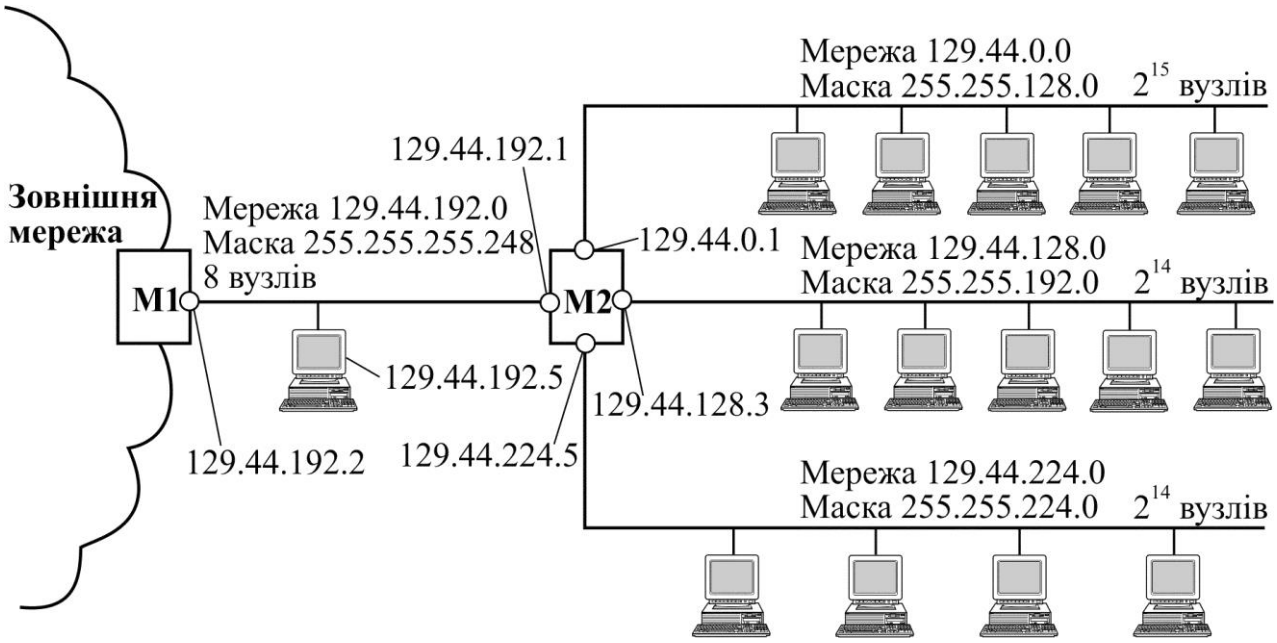


Рисунок А.7 – Мережа, структурована з використанням масок змінної довжини

Таблиця маршрутизації М2, що відповідна структурі мережі на рис. А.7, містить записи про чотири безпосередньо підключені мережі і запис про маршрутизатор за замовчуванням (табл. А.5). Процедура пошуку маршруту з використанням масок змінної довжини нічим не відрізняється від подібної процедури, описаної раніше для масок однакової довжини.

Деякі особливості масок змінної довжини проявляються за наявності так званих «перекриттів». Під перекриттям розуміється наявність декількох маршрутів до однієї й тієї самої мережі або одного й того самого вузла.

У цьому випадку адреса мережі в пакеті, що надійшов, може збігтися з адресами мереж, які містяться водночас у декількох записах таблиці маршрутизації.

Таблиця А.5 – Таблиця маршрутизатора М2 в мережі з масками змінної довжини

Номер мережі	Маска	Адреса наступного маршрутизатора	Адреса порта	Відстань
129.44.0.0	255.255.128.0	129.44.0.1	129.44.0.1	Підключена
129.44.128.0	255.255.192.0	129.44.128.3	129.44.128.3	Підключена
129.44.192.0	255.255.255.248	129.44.192.1	129.44.191.1	Підключена
129.44.224.0	255.255.224.0	129.44.224.5	129.44.224.5	Підключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–

Розглянемо приклад. Нехай пакет, що надійшов із зовнішньої мережі на маршрутизатор Ml, має адресу призначення 129.44.192.5. Нижче наведено фрагмент таблиці маршрутизації маршрутизатора Ml (табл. А.6). Перший з наведених двох записів говорить про те, що усі пакети, адреси яких починаються на 129.44, мають бути передані на маршрутизатор М2. Цей запис виконує агрегування адрес усіх підмереж, утворених на базі однієї мережі 129.44.0.0. Другий рядок говорить про те, що серед усіх можливих підмереж мережі 129.44.0.0 є одна, 129.44.192.0, для якої пакети можна направляти безпосередньо, а не через маршрутизатор М2.

Таблиця А.6 – Фрагмент таблиці маршрутизації маршрутизатора Ml

Номер мережі	Маска	Адреса наступного маршрутизатора	Адреса порта	Відстань
……..	……..	……..	……..	……..
129.44.0.0	255.255.0.0	129.44.192.1	129.44.191.2	2
129.44.192.0	255.255.255.248	129.44.192.2	129.44.192.2	Підключена
……..	……..	……..	……..	……..
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	–

Якщо дотримуватись стандартних алгоритмів пошуку маршруту згідно з таблицею, то спочатку на адресу призначення 129.44.192.5 накладається маска з першого рядка 255.255.0.0 і отримуємо результат 129.44.0.0, який збігається з номером мережі у цьому рядку. Але і з накладенням на адресу 129.44.192.5 маски з другого рядка 255.255.255.248 отриманий результат 129.44.192.0 також збігається з номером мережі у другому рядку. У таких випадках має бути застосоване таке правило: «Якщо адреса належить декільком

підмережам у базі даних маршрутів, то маршрутизатор, який просуває пакет, використовує найбільш специфічний маршрут, тобто обирається адреса підмережі, що дає більший збіг розрядів».

У даному прикладі буде обраний другий маршрут, тобто пакет буде переданий у безпосередньо підключену мережу, а не піде кружним шляхом через маршрутизатор М2.

Механізм вибору найбільш специфічного маршруту є узагальненням поняття «маршрут за замовчуванням». Оскільки у традиційному записі для маршруту за замовчуванням 0.0.0.0 маска 0.0.0.0 має нульову довжину, то цей маршрут вважається найбільш неспецифічним і використовується лише за відсутності збігів з усіма іншими записами з таблиці маршрутизації.

> ***ПРИМІТКА*** *У IP-пакетах з використанням механізму масок як і раніше передається лише IP-адреса призначення, а маска мережі призначення не передається. Тому з IP-адреси пакета, що надійшов, неможливо з'ясувати, яка частина адреси належить номеру мережі, а яка – номеру вузла. Якщо маски в усіх підмережах мають один розмір, то це не створює проблем. Якщо ж для утворення підмереж застосовують маски змінної довжини, то маршрутизатор має якось дізнаватися, яким адресам мереж які маски відповідають. Для цього використовуються протоколи маршрутизації, які переносять між маршрутизаторами не лише службову інформацію про адреси мереж, а й про маски, які відповідають цим номерам. До таких протоколів належать протоколи RIPv2 і OSPF, а, наприклад, протокол RIP маски не поширює і для використання масок змінної довжини не підходить.*

Технологія безкласової міждоменної маршрутизації CIDR

За останні кілька років у мережі Internet багато що змінилося: різко зросла кількість вузлів і мереж, підвищилася інтенсивність трафіка, змінився характер переданих даних. Через недосконаліть протоколів маршрутизації обмін повідомленнями про оновлення таблиць іноді став призводити до збоїв магістральних маршрутизаторів через перевантаження під час обробки великого обсягу службової інформації. Так, у 1994 році таблиці магістральних маршрутизаторів в Internet містили до 70000 маршрутів.

На вирішення цієї проблеми була спрямована, зокрема, і технологія безкласової міждоменної маршрутизації (Classless Inter-Domain Routing, CIDR),

про яку вперше офіційно оголошено у 1993 році, коли були опубліковані RFC 1517, RFC 1518, RFC 1519 та RFC 1520.

Сутність технології CIDR полягає у такому. Кожному постачальнику послуг Internet має призначатися безперервний діапазон у просторі IP-адрес. За такого підходу адреси усіх мереж кожного постачальника послуг мають загальну старшу частину – префікс, тому маршрутизація на магістралях Internet може здійснюватися на основі префіксів, а не повних адрес мереж. Агрегування адрес дозволить зменшити обсяг таблиць в маршрутизаторах усіх рівнів, а отже, прискорити роботу маршрутизаторів і підвищити пропускну здатність Internet.

Розподіл IP-адреси на номер мережі і номер вузла в технології CIDR відбувається не на основі декількох старших біт, що визначають клас мережі (A, B або C), а на основі маски змінної довжини, яка призначається постачальником послуг. На рис. А.8 показаний приклад деякого простору IP-адрес, який є у розпорядженні гіпотетичного постачальника послуг. Усі адреси мають загальну частину в k старших розрядах – префікс. Решта n розрядів використовуються для доповнення незмінного префікса змінною частиною адреси. Діапазон наявних адрес у такому випадку становить $2n$. Коли споживач послуг звертається до постачальника послуг з проханням про виділення йому деякої кількості адрес, то в наявному пулі адрес «вирізається» безперервна область S1, S2, S3 або S4 відповідного розміру. Причому межі цієї області обираються такими, щоб для нумерації необхідної кількості вузлів вистачило деякої кількості молодших розрядів, а значення усіх, що лишились (старших) розрядів було однаковим у всіх адрес даного діапазону. Таким умовам можуть задовольняти лише області, розмір яких кратний степеню двійки, А межі ділянки, яка виділяється, мають бути кратними необхідному розміру.

Розглянемо приклад. Нехай постачальник послуг Internet має пул адрес у діапазоні 193.20.0.0–193.23.255.255 (1100 0001.0001 0100.0000 0000.0000 0000–11000001.0001 0111.11111111.11111111) з загальним префіксом 193.20 (11000001.0001 01) і маскою, що відповідає даному префіксу 255.252.0.0.

Якщо абоненту цього постачальника послуг потрібно зовсім небагато адрес, наприклад 13, то постачальник міг би запропонувати йому різні варіанти: мережа 193.20.30.0, мережа 193.20.30.16 або мережу 193.21.204.48, все з одним і тим самим значенням маски 255.255.255.240. У всіх випадках в розпорядженні абонента для нумерації вузлів є 4 молодших біта.

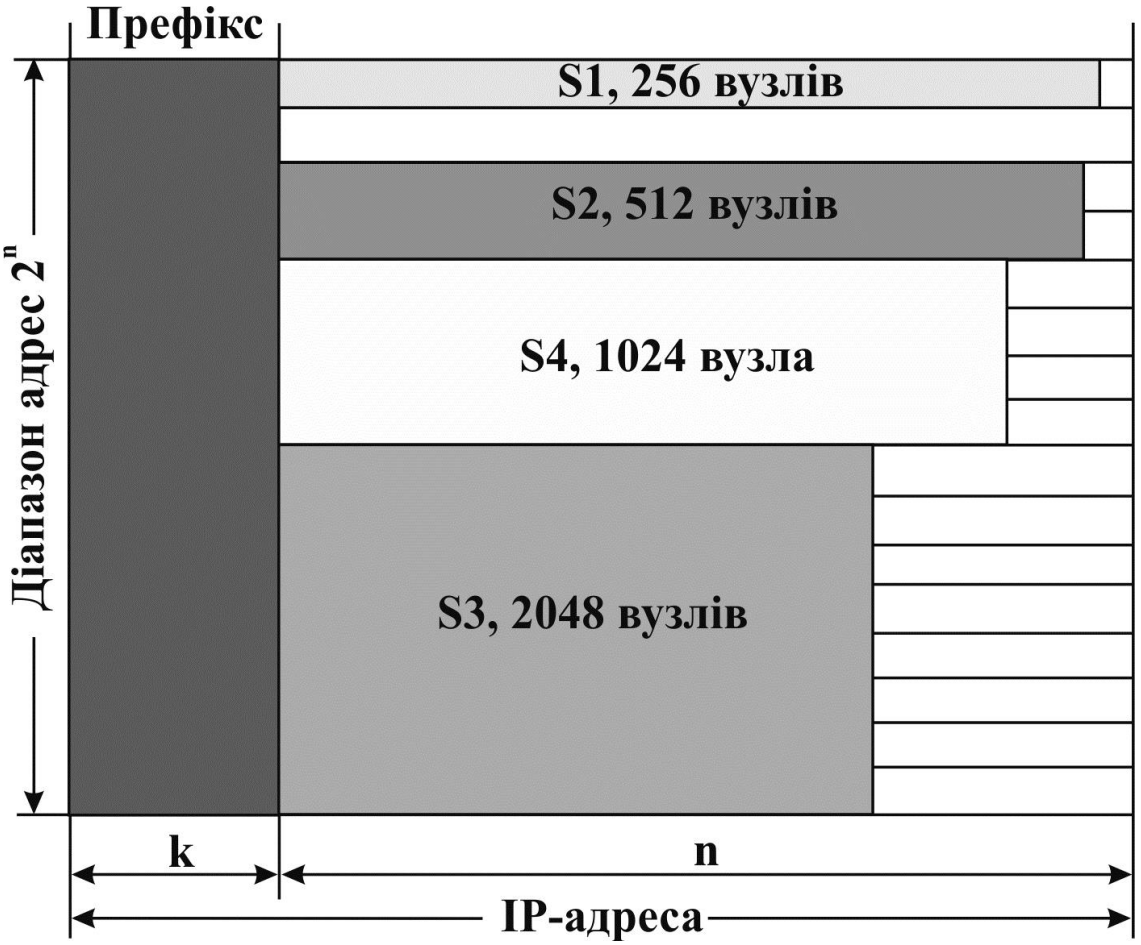


Рисунок А.8 – Технології CIDR

Розглянемо інший варіант, коли до постачальника послуг звернувся великий замовник, який сам, можливо, збирається надавати послуги з доступу в Internet. Йому потрібен блок адрес на 4000 вузлів. У цьому випадку постачальник послуг міг би запропонувати йому, наприклад, діапазон адрес 193,22.160.0–193.22.175.255 з маскою 255.255.240.0. Агрегований номер мережі (префікс) у даному випадку дорівнюватиме 193.22.160.0.

Адміністратор маршрутизатора М2 (рис. А.9) помістить у таблицю маршрутизації лише по одному запису на кожного клієнта, якому було виділено пул адрес, незалежно від кількості підмереж, організованих клієнтом. Якщо клієнт, який отримав мережу 193.22.160.0, через деякий час розділить її адресний простір в 4096 адрес на 8 підмереж, то в маршрутизаторі М2 початкова інформація про виділену йому мережу не зміниться.

Для постачальника послуг верхнього рівня, що підтримує клієнтів через маршрутизатор Ml, зусилля постачальника послуг нижнього рівня з розділення його адресного простору також не помітні. Запис 193.20.0,0 з маскою 255.252.0,0 повністю описує мережі постачальника послуг нижнього рівня в маршрутизаторі Ml.

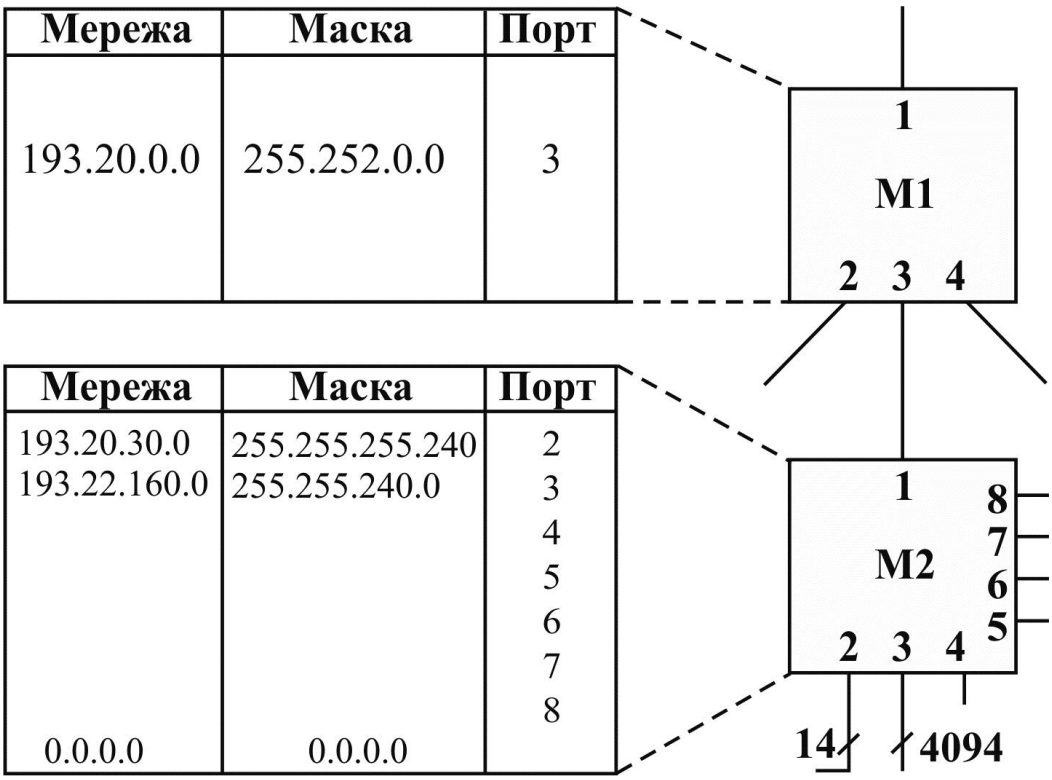


Рисунок А.9 – Виграш в кількості записів у маршрутизаторі з використанням технології CIDR

Отже, впровадження технології CIDR дозволяє вирішити дві основні задачі.

• Більш економне витрачання адресного простору. Дійсно, отримуючи в своє розпорядження адресу мережі, наприклад, класу С, деякі організації не використовують весь можливий діапазон адрес лише тому, що в їхній мережі є набагато менше 255 вузлів. Технологія CIDR відмовляється від традиційної концепції поділу адрес протоколу IP на класи, що дозволяє отримувати в користування стільки адрес, скільки реально необхідно. Завдяки технології CIDR постачальники послуг отримують можливість «нарізати» блоки з виділеного їм адресного простору в точній відповідності з вимогами кожного клієнта, при цьому у клієнта залишається простір для маневру на випадок його майбутнього зростання.

• Зменшення кількості записів у таблицях маршрутизаторів за рахунок об’єднання маршрутів – один запис у таблиці маршрутизації може представляти велику кількість мереж. Дійсно, для всіх мереж, номери яких починаються з однакової послідовності цифр, у таблиці маршрутизації може бути передбачений один запис (рис. А.9). Так, маршрутизатор М2 встановлений в організації, яка використовує техніку CIDR для виділення адрес своїм клієнтам, має підтримувати у своїй таблиці маршрутизації усі 8 записів про мережі клієнтів. А маршрутизатору Ml достатньо мати один запис про

всі ці мережі, на підставі якої він передає пакети з префіксом 193.20 маршрутизатору М2, що їх і розподіляє за потрібними портами.

Якщо усі постачальники послуг Internet дотримуватимуться стратегії CIDR, то особливо помітний виграш досягатиметься в магістральних маршрутизаторах.

Технологія CIDR вже успішно використовується у поточній версії IPv4 і підтримується такими протоколами маршрутизації, як OSPF, RIP-2, BGP4. Передбачається, що ці ж протоколи працюватимуть з новою версією протоколу IPv6. Слід зазначити, що в даний час технологія CIDR підтримується магістральними маршрутизаторами Internet, а не звичайними хостами в локальних мережах.

Використання CIDR в мережах IPv4 у загальному випадку вимагає перенумерації мереж. Оскільки ця процедура пов'язана з певними часовими і матеріальними витратами, для її проведення користувачів потрібно певним чином стимулювати. Як стимули розглядається, наприклад, введення оплати за рядок у таблиці маршрутизації або ж за кількість вузлів у мережі. Під час використання класів мереж абонент найчастіше у повному обсязі займає весь допустимий діапазон адрес вузлів – 254 адреси для мережі класу С або 65534 адреси для мережі класу В. Частина адрес вузлів зазвичай зникає. Вимога оплати кожної адреси вузла допоможе користувачу зважитися на перенумерацию, з тим щоб отримати саме стільки адрес, скільки йому потрібно.

6 Фрагментація IP-пакетів

Протокол IP дозволяє виконувати фрагментацію пакетів, що надходять на вхідні порти маршрутизаторів.

Слід розрізняти фрагментацію повідомлень у вузлі-відправнику й динамічну фрагментацію повідомлень у транзитних вузлах мережі – маршрутизаторах. Практично в усіх стеках протоколів є протоколи, які відповідають за фрагментацію повідомлень прикладного рівня на такі частини, які укладаються в кадри канального рівня. У стеці TCP/IP це завдання вирішує протокол TCP, який розбиває потік байтів, що передається йому з прикладного рівня на повідомлення потрібного розміру (наприклад, на 1460 байт для протоколу Ethernet). Тому протокол IP у вузлі-відправнику не використовує свої можливості з фрагментації пакетів.

А ось за необхідності передати пакет у наступну мережу, для якої розмір пакета є занадто великим, IP-фрагментація стає необхідною. У функції

рівня IP входить розбиття занадто великого для конкретного типу складової мережі повідомлення на більш короткі пакети зі створенням відповідних службових полів, потрібних для подальшого складання фрагментів у вихідне повідомлення.

У більшості типів локальних і глобальних мереж значення MTU, тобто максимальний розмір поля даних, в яке має інкапсулювати свій пакет протокол IP, значно відрізняється. Мережі Ethernet мають значення MTU, що дорівнює 1500 байт, мережі FDDI – 4096 байт, а мережі X.25 найчастіше працюють з MTU у 128 байт.

IP-пакет може бути позначений як і фрагментований. Будь-який пакет, позначений таким, не може бути фрагментований модулем IP за жодних умов. Якщо ж пакет, позначений як і фрагментований, не може дістатися отримувача без фрагментації, то цей пакет просто знищується, а вузлу-відправнику надсилається відповідне ICMP-повідомлення.

Протокол IP допускає можливість використання в межах окремої підмережі її власних коштів фрагментування, невидимих для протоколу IP. Наприклад, технологія ATM ділить IP-пакети, що надходять, на комірки з полем даних у 48 байт за допомогою свого рівня сегментування, а потім збирає комірки у вихідні пакети на виході з мережі. Але такі технології, як ATM, є швидше винятком, ніж правилом.

Процедури фрагментації і складання протоколу IP розраховані на те, щоб пакет міг бути розбитий на практично будь-яку кількість частин, які згодом могли б знову бути складені. Отримувач фрагмента використовує поле ідентифікації для того, щоб не переплутати фрагменти різних пакетів. Модуль IP, що відправляє пакет, встановлює в поле ідентифікації значення, яке має бути унікальним для даної пари відправник–отримувач, а також час, протягом якого пакет може бути активним у мережі.

Поле зсуву фрагмента повідомляє отримувачу положення фрагмента у вихідному пакеті. Зсув фрагмента і довжина визначають частину вихідного пакета, принесену цим фрагментом. Прапорець «more fragments» показує появу останнього фрагмента. Модуль протоколу IP, що відправляє нерозбитий на фрагменти пакет, встановлює в нуль прапорець «more fragments» і зміщення у фрагменті.

Ці поля дають достатню кількість інформації для складання пакета.

Щоб розділити на фрагменти великий пакет, модуль протоколу IP, встановлений, наприклад, на маршрутизаторі, створює кілька нових пакетів і копіює вміст полів IP-заголовка з великого пакета в IP-заголовки усіх нових

пакетів. Дані зі старого пакета діляться на відповідну кількість частин, розмір кожної, окрім останньої, обов'язково має бути кратним 8 байт. Розмір останньої частини даних дорівнює отриманому залишку.

Кожна з отриманих частин даних поміщається у новий пакет. Коли відбувається фрагментація, то деякі параметри IP-заголовка копіюються у заголовки всіх фрагментів, а інші залишаються лише в заголовку першого фрагмента. Процес фрагментації може змінити значення даних, розташованих у полі параметрів, і значення контрольної суми заголовка, змінити значення прапорця «more fragments» і зсув фрагмента, змінити довжину IP-заголовка і загальну довжину пакета. У заголовок кожного пакета заносяться відповідні значення в поле зсуву «fragment offset», а в поле загальної довжини пакета поміщається довжина кожного пакета. Перший фрагмент матиме в полі «fragment offset» нульове значення. У всіх пакетах, окрім останнього, прапорець «more fragments» встановлюється в одиницю, а в останньому фрагменті – в нуль.

Щоб скласти фрагменти пакета, модуль протоколу IP (наприклад, модуль на хост-комп'ютері) об'єднує IP-пакети, що мають однакові значення в полях ідентифікатора, відправника, отримувача і протокола. Отже, відправник має обрати ідентифікатор так, щоб він був унікальний для даної пари відправник-отримувач, для даного протоколу і протягом часу, поки даний пакет (або будь-який його фрагмент) може існувати в складеній IP-мережі.

Очевидно, що модуль протокола IP, який відправляє пакети, повинен мати таблицю ідентифікаторів, де кожен запис співвідноситься з кожним окремим отримувачем, з яким здійснювався зв'язок, і вказує останнє значення максимального часу життя пакета в IP-мережі. Але, оскільки поле ідентифікатора допускає 65536 різних значень, деякі хости можуть використовувати просто унікальні ідентифікатори, які не залежать від адреси отримувача.

У деяких випадках доцільно, щоб ідентифікатори IP-пакетів обиралися протоколами вищого, ніж IP, рівня. Наприклад, у протоколі TCP передбачена повторна передача TCP-сегментів, які з будь-яких причин не дійшли до адресата. Імовірність правильного прийому збільшувалася б, якби за повторної передачі ідентифікатор для IP-пакета був би тим самим, що і в початковому IP-пакеті, оскільки його фрагменти могли б використовуватися для складання правильного TCP-сегмента.

Процедура об'єднання полягає в розміщуванні даних з кожного фрагмента в позицію, зазначену у заголовку пакета в поле «fragment offset».

Кожен модуль IP має бути здатний передати пакет з 68 байт без подальшої фрагментації. Це пов'язано з тим, що IP-заголовок може включати

до 60 байт, а мінімальний фрагмент даних – 8 байт. Кожен отримувач має бути в змозі прийняти пакет з 576 байт як єдиного куска або у вигляді фрагментів, що підлягають складанню.

Якщо біт прапорця заборони фрагментації (Do not Fragment, DF) встановлено, то фрагментація даного пакета заборонена, навіть якщо в цьому випадку він буде втрачений. Даний засіб може використовуватися для запобігання фрагментації в тих випадках, коли хост-отримувач не має достатніх ресурсів для складання фрагментів.

Робота протоколу IP з фрагментації пакетів у хостах і маршрутизаторах проілюстрована на рис. А.10.

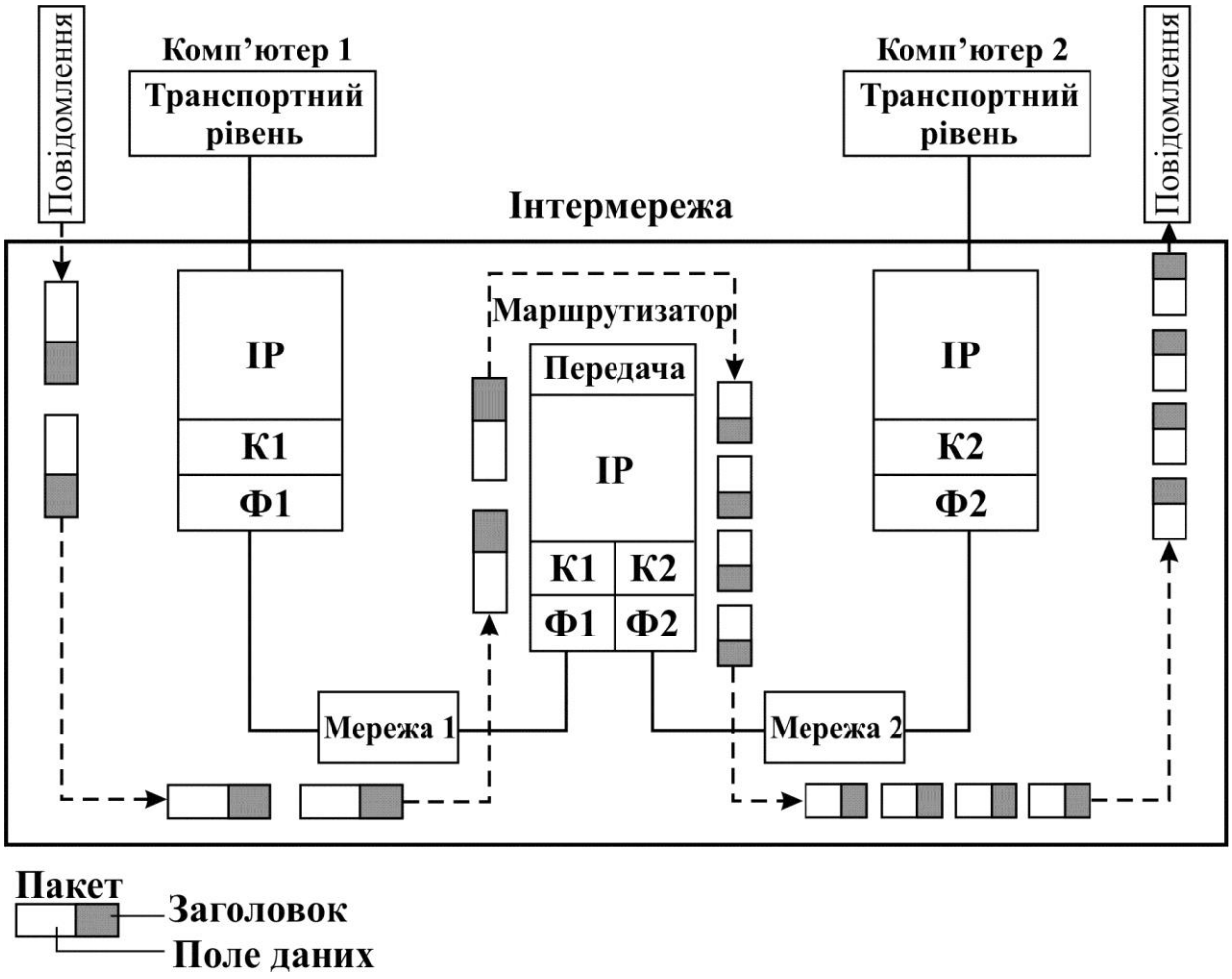


Рисунок А.10 – Фрагментація IP-пакетів у процесі передачі між мережами з різним максимальним розміром пакетів: К1 і 01 – канальний і фізичний рівень мережі 1; К2 і Ф2 – канальний і фізичний рівень мережі 2

Нехай комп'ютер 1 пов'язаний з мережею, що має значення MTU в 4096 байт, наприклад, з мережею FDDI. Під час надходження на IP-рівень комп'ютера 1 повідомлення від транспортного рівня розміром в 5600 байт протокол IP ділить його на два IP-пакета, встановлюючи у першому пакеті

ознаку фрагментації і привласнюючи пакету унікальний ідентифікатор, наприклад, 486. У першому пакеті величина поля зсуву дорівнює 0, а у другому – 2800. Ознака фрагментації в другому пакеті дорівнює нулю, що показує, що це останній фрагмент пакета. Загальна величина IP-пакета становить 2800 плюс 20 (розмір IP-заголовка), то тобто 2820 байт, що вміщується в поле даних кадра FDDI. Далі модуль IP комп'ютера 1 передає ці пакети своєму мережному інтерфейсу (утвореному протоколами канального рівня К1 і фізичного рівня Ф1). Мережний інтерфейс відправляє кадри наступному маршрутизатору.

Після того, як кадри пройдуть рівень мережного інтерфейса маршрутизатора (К1 і Ф1) і звільняться від заголовків FDDI, модуль IP за мережною адресою визначає, що два пакети, які надійшли, потрібно передати в мережу 2, яка є мережею Ethernet і має значення MTU, що дорівнює 1500. Отже, IP-пакети, які надійшли, необхідно фрагментувати. Маршрутизатор вилучає поле даних з кожного пакета і ділить його ще навпіл, щоб кожна частина вмістилася в поле даних кадра Ethernet. Потім він формує нові IP-пакети, кожен з яких має довжину $1400 + 20 = 1420$ байт, що менше 1500 байт, тому вони нормально поміщаються в поле даних кадрів Ethernet.

У результаті в комп'ютер 2 по мережі Ethernet надходять чотири IP-пакети з загальним ідентифікатором 486, що дозволяє протоколу IP, який працює в комп'ютері 2, правильно скласти вихідне повідомлення. Якщо пакети надійшли не в тому порядку, в якому були надіслані, то зсув вкаже правильний порядок їх об'єднання.

Зазначимо, що IP-маршрутизатори не збирають фрагменти пакетів у великі пакети, навіть якщо на шляху зустрічається мережа, що допускає таке укрупнення. Це пов'язано з тим, що окремі фрагменти повідомлення можуть переміщуватися інтермережею різними маршрутами, тому немає гарантії, що всі фрагменти проходять через якийсь проміжний маршрутизатор на їхньому шляху.

У процесі надходження першого фрагмента пакета вузол призначення запускає таймер, який визначає максимально допустимий час очікування надходження інших фрагментів цього пакета. Таймер встановлюється на максимальне з двох значень: початковий установчий час очікування і час життя, вказаний у прийнятому фрагменті. Отже, початкова установка таймера є нижньою межею для часу очікування під час складання. Якщо таймер закінчується раніше прибуття останнього фрагмента, то всі ресурси збирання, пов'язані з даними пакетом, звільняються, всі отримані до цього моменту

фрагменти пакета відкидаються, а до вузла, що надіслав вихідний пакет, направляється повідомлення про помилку за допомогою протоколу ICMP.

7 Протокол надійної доставки TCP-повідомлень

Протокол IP є дейтаграмним протоколом і тому за своєю природою не може гарантувати надійність передачі даних. Дане завдання – забезпечення надійного каналу обміну даними між прикладними процесами в складеній мережі – вирішує протокол TCP (Transmission Control Protocol), що належить до транспортного рівня.

Протокол TCP працює безпосередньо над протоколом IP і використовує для транспортування своїх блоків даних потенційно ненадійний протокол IP. Надійність передачі даних протоколом TCP досягається за рахунок того, що він заснований на встановленні логічних з'єднань між взаємодіючими процесами. Доти, доки програми протоколу TCP продовжують функціонувати коректно, а складена мережа не розпалася на незв'язні частини, помилки в передачі даних на рівні протоколу IP не впливатимуть на правильне отримання даних.

Протокол IP використовується протоколом TCP як транспортний засіб. Перед відправкою своїх блоків даних протокол TCP поміщає їх в оболонку IP-пакета. За необхідності протокол IP здійснює будь-яку фрагментацію та складання блоків даних TCP, необхідні для здійснення передачі та доставки через велику кількість мереж і проміжних шлюзів.

На рис. А.11 показано, як процеси, що виконуються на двох кінцевих вузлах, встановлюють за допомогою протоколу TCP надійний зв'язок через складену мережу, всі вузли якої використовують для передачі повідомлень дейтаграммний протокол IP.

Порти

Протокол TCP взаємодіє через міжрівневі інтерфейси з протоколом IP, що лежить нижче, і протоколами прикладного рівня, що лежать вище, або додатками.

У той час як завданням мережного рівня, до якого належить протокол IP, є передача даних між довільними вузлами мережі, завдання транспортного рівня, яке вирішує протокол TCP, полягає в передачі даних між будь-якими прикладними процесами, що виконуються на будь-яких вузлах мережі. Дійсно, після того як пакет засобами протоколу IP доставлений у комп'ютер-отримувач, дані необхідно направити конкретному процесу-отримувачу.

Кожен комп’ютер може виконувати декілька процесів, більш того, прикладний процес теж може мати кілька точок входу, які виступають як адреси призначення для пакетів даних.

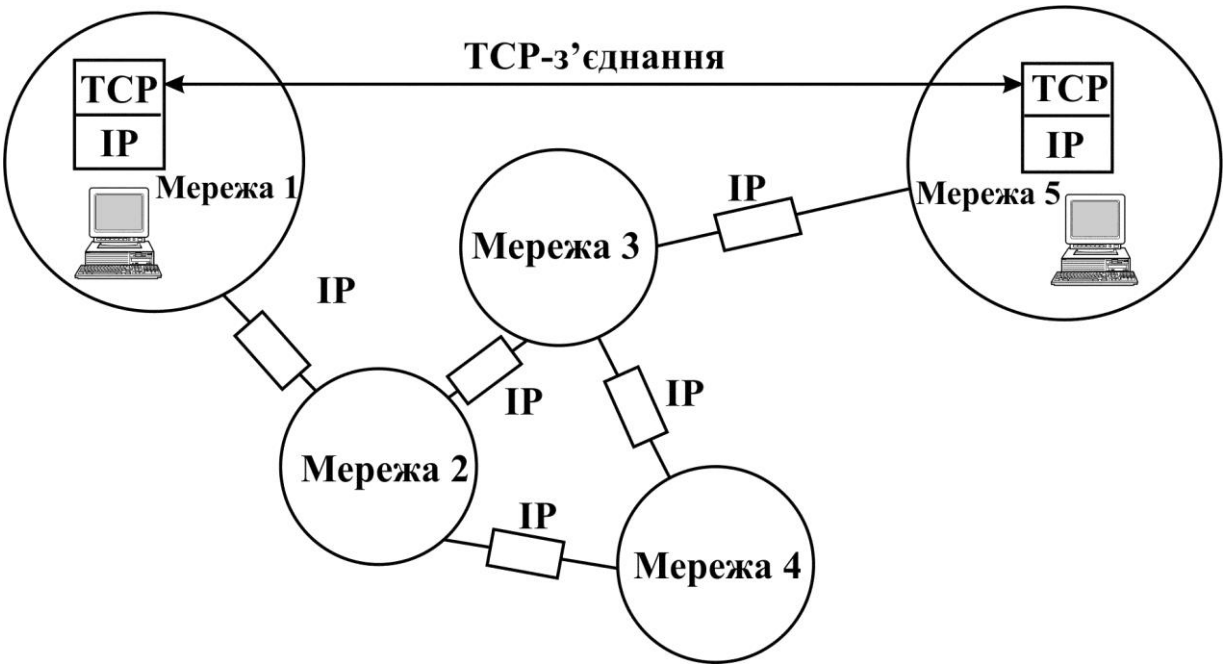


Рисунок А.11 – TCP-з’єднання створює надійний канал зв’язку між кінцевими вузлами

Пакети, що надходять на транспортний рівень, організовуються операційною системою у вигляді множини черг до точок входу різних прикладних процесів. У термінології TCP/IP такі системні черги називаються портами. Отже, адресою призначення, яка використовується протоколом TCP, є ідентифікатор (номер) порта прикладної служби. Номер порта в сукупності з номером мережі і номером кінцевого вузла однозначно визначають прикладний процес у мережі. Цей набір ідентифікуючих параметрів має назву сокет (socket).

Призначення номерів портів прикладним процесам здійснюється або централізовано, якщо ці процеси є популярними загальнодоступними службами (наприклад, номер 21 закріплений за службою віддаленого доступу до файлів FTP, а 23 – за службою віддаленого управління telnet), або локально для тих служб, які ще не стали настільки поширеними, щоб закріплювати за ними стандартні (зарезервовані) номери. Централізоване присвоєння службам портів виконується організацією Internet Assigned Numbers Authority (IANA). Дані номери потім закріплюються і публікуються в стандартах Internet (RFC 1700).

Локальне присвоєння номера порту полягає в тому, що розробник деякого додатка просто пов’язує з ним будь-який доступний, довільно обраний числовий ідентифікатор, звертаючи увагу на те, щоб він не входив до множини

зарезервованих номерів портів. Надалі всі вилучені запити до даного додатка від інших програм слід адресувати із зазначенням призначеного йому номера порту.

Протокол TCP веде для кожного порту дві черги: черга пакетів, що надходять у даний порт з мережі, і черга пакетів, що відправляються даним портом у мережу. Процедура обслуговування протоколом TCP запитів, що надходять від декількох різних прикладних служб, називається мультиплексуванням.

Зворотна процедура розподілу протоколом TCP пакетів, що надходять від мережного рівня між набором високорівневих служб, ідентифікованих номерами портів, називається демультиплексування (рис. А.12).

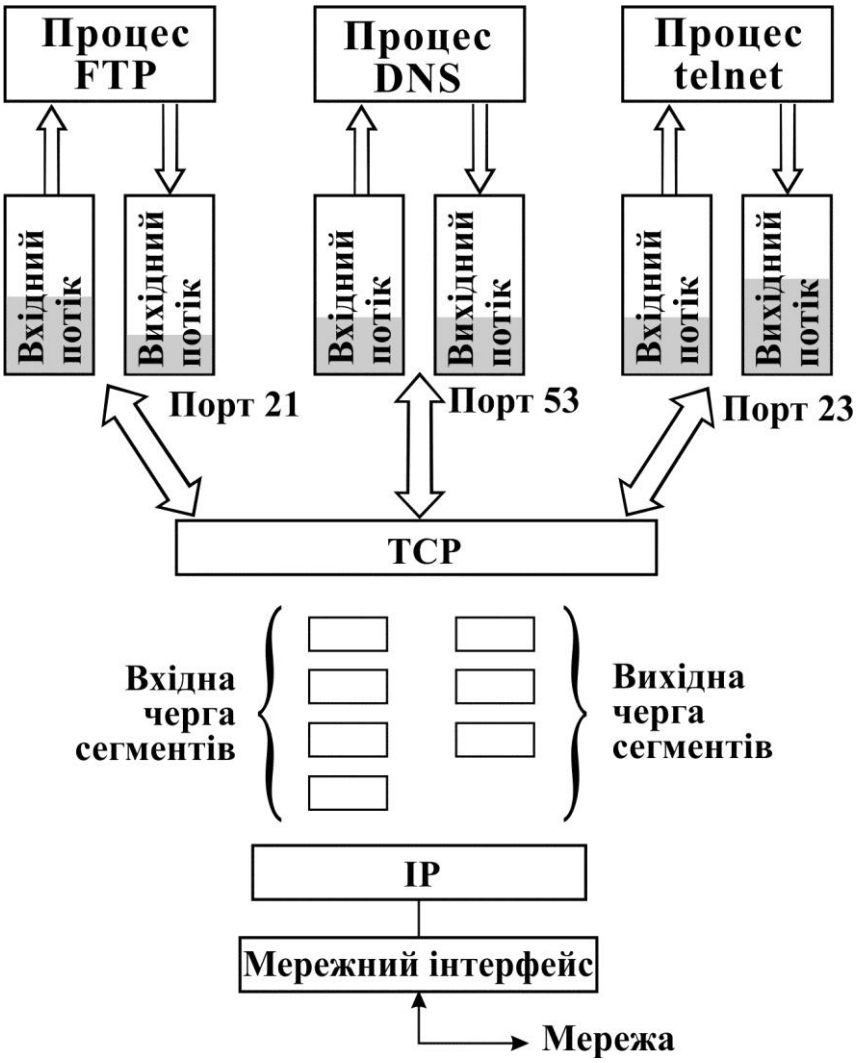


Рисунок А.12 – Функції протоколу TCP
з мультиплексування і демультиплексування потоків

Сегменти і потоки

Одиницею даних протоколу TCP є сегмент. Інформація, що надходить до протоколу TCP у рамках логічного з'єднання від протоколів більш високого

рівня, розглядається протоколом TCP як неструктурований потік байтів. Дані, що надходять, буферизуються засобами TCP.

Для передачі на мережний рівень з буфера «вирізається» деяка безперервна частина даних, яка і називається сегментом (див. рис. А.12). На відміну від багатьох інших протоколів, протокол TCP підтверджує отримання не пакетів, а байтів потоку.

Не всі сегменти, надіслані через з'єднання, будуть одного й того самого розміру, але обидва учасники з'єднання мають домовитися про максимальний розмір сегмента, який вони використовуватимуть. Цей розмір обирається так, щоб в процесі упаковування сегмента в IP-пакет він містився туди цілком, тобто максимальний розмір сегмента не має перевищувати максимального розміру поля даних IP-пакета. Інакше довелося б виконувати фрагментацію, тобто ділити сегмент на кілька частин, щоб розмістити його в IP-пакеті.

З'єднання

Для організації надійної передачі даних передбачається встановлення логічного з'єднання між двома прикладними процесами. Оскільки з'єднання встановлюються через ненадійну комунікаційну систему, засновану на протоколі IP, то для уникнення помилкової ініціалізації з'єднань використовується спеціальна багатокрокова процедура підтвердження зв'язку.

З'єднання у протоколі TCP ідентифікується парою повних адрес обох взаємодіючих процесів – сокетів. Кожен із взаємодіючих процесів може брати участь у декількох з'єднаннях.

Формально з'єднання можна визначити як набір параметрів, який характеризує процедуру обміну даними між двома процесами. Крім повних адрес процесів цей набір містить і параметри, значення яких визначаються в результаті переговорного процесу модулів TCP двох сторін з'єднання. До таких параметрів належать, зокрема, погоджені розміри сегментів, які може надсилати кожна зі сторін, обсяги даних, які дозволено передавати без отримання на них підтвердження, початкові й поточні номери переданих байтів. Деякі з цих параметрів залишаються незмінними протягом усього сеансу зв'язку, а деякі адаптивно змінюються.

В рамках з'єднання здійснюється обов'язкове підтвердження правильності прийому для всіх переданих повідомлень і за необхідності виконується повторна передача. З'єднання в TCP дозволяє вести передачу даних в обидві сторони водночас, тобто повнодуплексну передачу.

Реалізація ковзного вікна в протоколі TCP

У рамках встановленого з'єднання правильність передачі кожного сегмента має підтверджуватися квитанцією отримувача. Квитування – це один з традиційних методів забезпечення надійного зв'язку. У протоколі TCP використовується окремий випадок квитування – алгоритм ковзного вікна.

Особливість використання алгоритму ковзного вікна в протоколі TCP полягає в тому, що, хоча одиницею переданих даних є сегмент, вікно визначене на множині нумерованих байтів неструктурованого потоку даних, що надходять з верхнього рівня, і буферизованих протоколом TCP. Модуль TCP, який отримує, відправляє «вікно» модулю TCP, який відсилає. Дане вікно задає кількість байтів (починаючи з номера байта, про який вже була вислана квитанція), яке приймаючий модуль TCP готовий зараз прийняти.

Квитанція (підтвердження) надсилається лише в разі правильного прийому даних, негативні квитанції не посилаються. Отже, відсутність квитанції означає або прийом перекрученого сегмента, або втрату сегмента, або втрату квитанції. Як квитанцію отримувач сегмента відсилає відповідь-повідомлення (сегмент), в яке поміщає число, що на одиницю перевищує максимальний номер байта в отриманому сегменті. Це число часто називають номером черги.

На рис. А.13 показаний потік байтів, що надходить на вхід протоколу TCP. З потоку байтів модуль TCP нарізає послідовність сегментів. Для визначеності на рисунку прийнято напрямок переміщення даних справа наліво. У цьому потоці можна вказати кілька логічних меж.

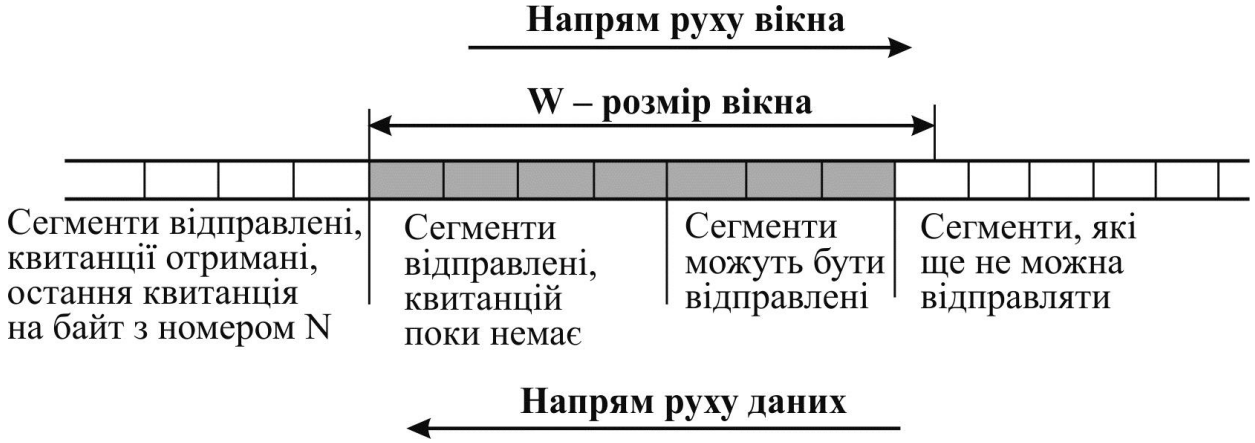


Рисунок А.13 – Особливості реалізації
алгоритму ковзного вікна в протоколі TCP

Перша межа відокремлює сегменти, які вже були відправлені, і на які вже надійшли квитанції. Наступну частину потоку становлять сегменти, які також

вже відправлені, оскільки входять в межі, визначені вікном, але квитанції на них поки не отримані. Третя частина потоку – це сегменти, які поки не відправлені, але можуть бути відправлені, оскільки входять в межі вікна. І нарешті, остання межа вказує на початок послідовності сегментів, жоден з яких не може бути відправлений доти, доки не надійде чергова квитанція і вікно не буде зрушено вправо.

Якщо розмір вікна дорівнює W, а остання за часом квитанція містила значення N, то відправник може надсилати нові сегменти доти, доки в черговий сегмент не потрапить байт з номером $N+W$. Цей сегмент виходить за рамки вікна, і передачу в такому випадку необхідно припинити до надходження наступної квитанції.

Надійність передачі досягається завдяки підтвердженням і номерам черги. Концептуально кожному байту даних присвоюється номер черги. Номер черги для першого байта даних у сегменті передається разом з цим сегментом і називається номером черги для сегмента. Сегменти також несуть номер підтвердження, який є номером для наступного очікуваного байта даних, переданого в зворотному напрямку. Коли протокол TCP передає сегмент з даними, він поміщає його копію в чергу повторної передачі і запускає таймер. Коли надходить підтвердження для цих даних, відповідний сегмент видаляється з черги. Якщо підтвердження не надходить до закінчення терміну, то сегмент посилається повторно.

Вибір часу очікування (тайм-ауту) чергової квитанції є важливим завданням, результат рішення якого впливає на продуктивність протоколу TCP. Тайм-аут не має бути занадто коротким, щоб за можливості виключити надлишкові повторні передачі, які знижують корисну пропускну здатність системи. Але він не має бути і занадто великим, щоб уникнути тривалих простоїв, пов’язаних з очікуванням квитанції, якої не існує або «заблукала».

Під час вибору величини тайм-ауту мають враховуватися швидкість і надійність фізичних ліній зв’язку, їхня протяжність і багато інших подібних факторів. У протоколі TCP тайм-аут визначається за допомогою досить складного адаптивного алгоритму, ідея якого полягає в такому. Під час кожної передачі засікається час від моменту відправлення сегмента до надходження квитанції про його прийом (час обігу). Отримувані значення часу обігу усереднюються з ваговими коефіцієнтами, що зростають від попереднього виміру до наступного. Це робиться для того, щоб посилити вплив останніх вимірів. Як тайм-аут обирається середній час обігу, помножений на деякий коефіцієнт. Практика показує, що значення цього коефіцієнта має

перевищувати 2. У мережах з великим розкидом часу обігу у процесі вибору тайм-ауту враховується і дисперсія цієї величини.

Оскільки кожен байт пронумерований, то кожен з них може бути пізнаний. Прийнятний механізм розпізнавання є накопичувальним, тому упізнання номера X означає, що всі байти з попередніми номерами вже отримані. Цей механізм дозволяє реєструвати появу дублікатів в умовах повторної передачі. Нумерація байтів у межах сегмента здійснюється так, щоб перший байт даних відразу слідом за заголовком мав найменший номер, а наступні за ним байти мали зростаючі номери.

Вікно, що посилається з кожним сегментом, визначає діапазон номерів черги, які відправник вікна (він же отримувач даних) готовий прийняти в даний час. Передбачається, що такий механізм пов'язаний з наявністю в даний момент місця в буфері даних.

Варіюючи величину вікна, можна впливати на завантаження мережі. Чим більше вікно, тим більшу порцію непідтверджених даних можна надіслати в мережу. Але, якщо надійшла більша кількість даних, ніж може бути прийнята програмою TCP, дані будуть відкинуті. Це призведе до зайвих пересилань інформації і непотрібного збільшення навантаження на мережу і програму TCP.

З іншого боку, призначення вікна малого розміру може обмежити передачу даних швидкістю, яка визначається часом подорожі по мережі кожного сегмента, що посилається. Щоб уникнути застосування малих вікон, отримувачу даних пропонується відкладати зміну вікна доти, доки вільне місце не складе 20–40% від максимально можливого обсягу пам'яті для цього з'єднання. Але і відправнику не варто поспішати з надсиланням даних, доки вікно не стане досить великим. З огляду на ці міркування, розробники протоколу TCP запропонували схему, згідно з якою у процесі встановлення з'єднання з'являється велике вікно, але згодом його розмір істотно зменшується.

Якщо мережа не справляється з навантаженням, то виникають черги в проміжних вузлах – маршрутизаторах і в кінцевих вузлах – комп'ютерах.

З переповненням прийомного буфера кінцевого вузла «перевантажений» протокол TCP, відправляючи квитанцію, поміщає в неї новий, зменшений розмір вікна. Якщо він зовсім відмовляється від прийому, то в квитанції вказується вікно нульового розміру. Проте навіть після цього додаток може надіслати повідомлення на порт, який відмовився від прийому. Для цього повідомлення має супроводжуватися позначкою «терміново». У такій ситуації порт зобов'язаний прийняти сегмент, навіть якщо для цього доведеться витіснити з буфера дані, які вже знаходяться там. Після прийому квитанції

з нульовим значенням вікна протокол-відправник час від часу робить контрольні спроби продовжити обмін даними. Якщо протокол-приймач вже готовий приймати інформацію, то у відповідь на контрольний запит він посилає квитанцію із зазначенням ненульового розміру вікна.

Іншим проявом перевантаження мережі є переповнення буферів у маршрутизаторах. У таких випадках вони можуть централізовано змінити розмір вікна, посилаючи керуючі повідомлення деяким кінцевим вузлам, що дозволяє їм диференційовано управляти інтенсивністю потоку даних у різних частинах мережі.

Висновки

• Протокол IP вирішує завдання доставки повідомлень між вузлами складеної мережі. Протокол IP належить до протоколів без встановлення з'єднань, тому він не дає жодних гарантій надійної доставки повідомлень. Всі питання забезпечення надійності доставки даних у складеній мережі в стеку TCP/IP вирішує протокол TCP, заснований на встановленні логічних з'єднань між взаємодіючими процесами.

• IP-пакет складається із заголовка і поля даних. Максимальна довжина пакета 65535 байт, Тема зазвичай має довжину 20 байт і містить інформацію про мережні адреси відправника і отримувача, про параметри фрагментації, про час життя пакета, про контрольну суму і деякі інші. В полі даних IP-пакета знаходяться повідомлення більш високого рівня, наприклад TCP або UDP.

• Вид таблиці IP-маршрутизації залежить від конкретної реалізації маршрутизатора, але, незважаючи на досить сильні зовнішні відмінності, в таблицях усіх типів маршрутизаторів є всі ключові поля, необхідні для виконання маршрутизації.

• Існує кілька джерел, що поставляють записи в таблицю маршрутизації. По-перше, у процесі ініціалізації програмне забезпечення стека TCP/IP заносить до таблиці записи про безпосередньо підключені мережі і маршрутизатори за замовчуванням, а також записи про особливі адреси типу 127.0.0.0. По-друге, адміністратор вручну заносить статичні записи про специфічні маршрути або про маршрутизатор за замовчуванням. По-третє, протоколи маршрутизації автоматично заносять до таблиці динамічні записи про наявні маршрути.

• Ефективним засобом структуризації IP-мереж є маски. Маски дозволяють розділити одну мережу на декілька підмереж. Маски однакової довжини використовуються для розподілу мережі на підмережі рівного розміру,

а маски змінної довжини – для поділу мережі на підмережі різного розміру. Використання масок модифікує алгоритм маршрутизації, тому в цьому випадку висуваються особливі вимоги до протоколів маршрутизації в мережі, до технічних характеристик маршрутизаторів і процедур їх конфігурації.

• Значна роль у майбутньому IP-мереж відводиться технології безкласової міждоменної маршрутизації (CIDR), яка вирішує дві основні задачі. Перша полягає в більш економному витрачанні адресного простору – завдяки CIDR постачальники послуг отримують можливість «нарізати» блоки різних розмірів з виділеного їм адресного простору в точній відповідності з вимогами кожного клієнта. Друга – в зменшенні кількості записів у таблицях маршрутизації за рахунок об’єднання маршрутів – один запис у таблиці маршрутизації може становити велику кількість мереж із загальним префіксом.

• Важливою особливістю протоколу IP, що відрізняє його від інших мережних протоколів, є його здатність виконувати динамічну фрагментацію пакетів у процесі передачі їх між мережами з різними MTU. Це властивість багато в чому сприяла тому, що протокол IP зміг зайняти домінуючі позиції в складних складених мережах.

ЗАПИТАННЯ ТА ЗАВДАННЯ

1. У чому відмінність завдань, які вирішуються протоколами мережного рівня в локальних і глобальних мережах?

2. Порівняйте таблицю моста/комутатора з таблицею маршрутизатора. Як вони формуються? Яку інформацію містять? Від чого залежить їхній обсяг?

3. Таблиця маршрутизації містить записи про мережі призначення. Чи має вона містити записи про всі мережі складеної мережі чи тільки про деякі? Якщо тільки про деякі, то про які саме?

4. Чи може в таблиці маршрутизації бути кілька записів про маршрутизатори за замовчуванням?

5. На рис. Б.1 зображений комп'ютер з двома мережними адаптерами, до яких під'єднані сегменти мережі. Комп'ютер працює під управлінням Windows NT. Чи може комп'ютер А обмінюватися даними з комп'ютером В?

A. Так, завжди.

B. Ні, завжди.

C. Все залежить від того, як налаштована система Windows NT.

Чи може вплинути на відповідь той факт, що в сегментах використовуються різні канальні протоколи, наприклад Ethernet і Token Ring?

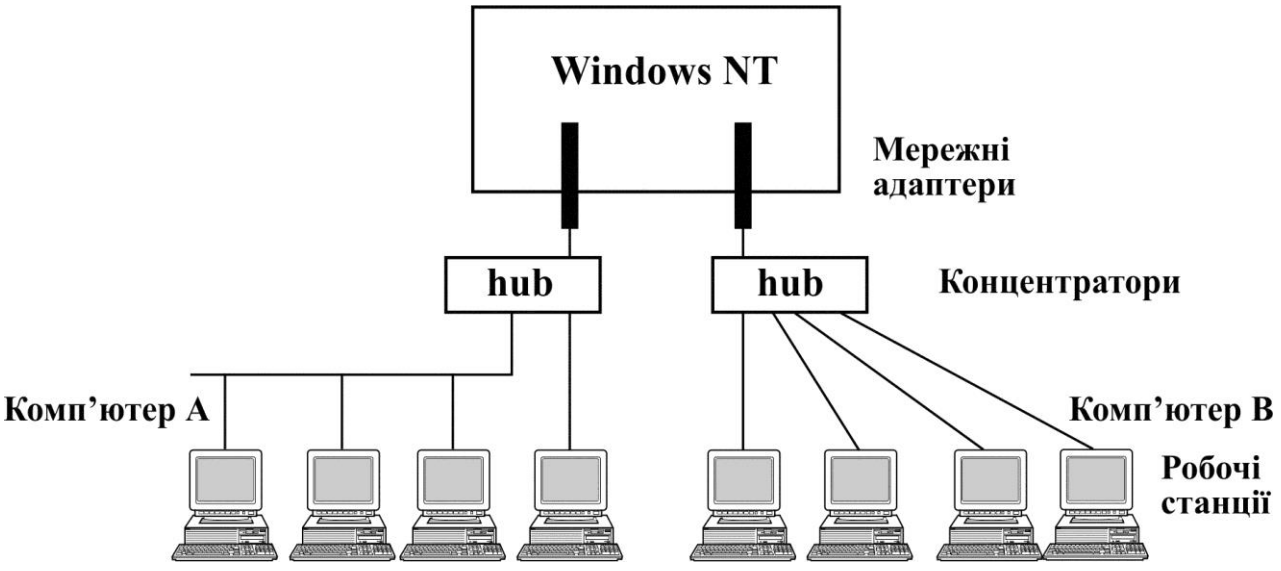


Рисунок Б.1 – Режими роботи комп'ютера з двома мережними картами

6. Скільки рівнів має стек протоколів TCP/IP? Які їхні функції? Які особливості цього стека обумовлюють його лідируюче положення в світі мережних технологій?

7. Які протоколи стека TCP/IP належать до рівня Internet (рівня міжмережної взаємодії)?

8. У чому проявляється ненадійність протоколу IP?

9. Чи можуть бути виявлені помилки на рівні Internet? Чи можуть вони бути виправлені засобами цього рівня?

10. У чому особливості реалізації алгоритму ковзаючого вікна в протоколі TCP?

11. У складених мережах використовуються три види адрес: символьні, мережні і локальні. Які з наведених нижче адрес могли бути локальними, а які – ні у складеній IP-мережі?

A. 6-байтова MAC-адреса (наприклад, 12-B3-3B-51-A2-10);

B. адреса X.25 (наприклад, 25012112654987);

C. 12-байтова IPX-адреса (наприклад, 13.34.B4.0A.C5.10.11.32.54.C5.3B.01);

D. адреса VPI/VCI мережі ATM.

12. Які з нижченаведених тверджень правильні завжди?

A. Кожен порт моста/комутатора має MAC-адресу.

B. Кожен міст/комутатор має мережну адресу.

C. Кожен порт моста/комутатора має мережну адресу.

D. Кожен маршрутизатор має мережну адресу.

E. Кожен порт маршрутизатора має MAC-адресу.

F. Кожен порт маршрутизатора має мережну адресу.

13. Яку частку всієї безлічі IP-адрес складають адреси класу A? Класу B? Класу C?

14. Які з нижченаведених адрес не можуть бути використані як IP-адреса кінцевого вузла мережі, підключеної до Internet? Для синтаксично правильних адрес визначте їхній клас: A, B, C, D або E.127.0.0.1

A. 201.13.123.245
B. 226.4.37.105
C. 103.24.254.0
D. 10.234.17.25
E. 154.12.255.255
F. 13.13.13.13
G. 204.0.3.1
H. 193.256.1.16
I. 194.87.45.0
J. 195.34.116.255
K. 161.23.45.305

15. Нехай IP-адреса деякого вузла підмережі дорівнює 198.65.12.67, а значення маски для цієї підмережі – 255.255.255.240. Визначте номер підмережі. Яке максимальне число вузлів може бути в цій підмережі?

16. Нехай постачальник послуг Internet має в своєму розпорядженні адресу мережі класу B. Для адресації вузлів своєї власної мережі він використовує 254 адреси. Визначте максимально можливу кількість абонентів цього оператора, якщо розміри необхідних для них мереж відповідають класу C? Яка маска має бути встановлена на маршрутизаторі постачальника послуг, що з'єднує його мережу з мережами абонентів?

17. Яку максимальну кількість підмереж теоретично можливо організувати, якщо у вашому розпорядженні є мережа класу C? Яке значення при цьому має маска?

18. Чому навіть в тих випадках, коли використовуються маски, в IP-пакеті маска не передається?

19. Які переваги дає технологія CIDR? Що заважає її широкому впровадженню?

20. Чи є зв'язок між довжиною префікса пулу IP-адрес і числом адрес, що входять в цей пул?

21. Чому в запису про маршрутизатор за замовчуванням як адреса мережі призначення вказується 0.0.0.0 з маскою 0.0.0.0?

22. Чи відрізняється обробка поля MAC-адреси кадру маршрутизатором і комутатором?

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. / В.Г. Олифер, Н.А.Олифер. – СПб.: Питер, 2010. – 944 с.

2. Сети передачи данных. /Д. Бертсекас, Р. Галагер. – М.: Мир, 1989. – 544 с.

3. Комп'ютерні мережі: навч. посібник / [О.Д. Азаров, С.М. Захарченко, О.В. Кадук та ін.] – Вінниця: ВНТУ, 2013. – 371 с.

4. Буров Є.В. Комп'ютерні мережі: підручник /Є.В. Буров. – Львів: «Магнолія 2006», 2010. – 262 с.

5. Кулаков Ю.О., Жуков І.А. Комп'ютерні мережі: навч. посібник / за ред. Ю.О. Кулакова. – К: НАУ, 2009. – 392 с.

6. Комп'ютерні мережі: [навчальний посібник] / А.Г. Микитишин, М.М. Митник, П.Д. Стухляк, В.В. Пасічник. – Львів: «Магнолія 2006», 2013. – 256 с.

Електронне навчальне видання

КОНСПЕКТ ЛЕКЦІЙ
з дисципліни
«ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНІ СИСТЕМИ»
частина ІІ

для студентів усіх форм навчання спеціальності 125 «Кібербезпека»
за освітньою програмою «Безпека інформаційних комунікаційних систем»

Упорядник ХАЛІМОВ Геннадій Зайдулович

Відповідальний випусковий Г.З. Халімов
Редактор Б.П. Косіковська
Комп'ютерна верстка Л.Ю. Свєтайло

План 2019 (друге півріччя), поз. 32

Підп. до використання 27.06.2019 Формат pdf. Обсяг даних 6,19 Mb

ХНУРЕ. Україна. 61166, Харків, просп. Науки, 14, E-mail: info@nure.ua

Підготовлено в редакційно-видавничому відділі ХНУРЕ
Свідоцтво суб'єкта видавничої справи ДК №1409 від 26.06.2003