Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи балансування інформаційними потоками комп’ютерних мереж
(тема)

Виконав:

студент II курсу, групи КСМм-23-1

Соколовський С.О.
(прізвище, ініціали)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Янковський О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Соколовському Сергію Олеговичу
(прізвище, ім’я, по батькові)

1. Тема роботи Методи балансування інформаційними потоками комп’ютерних мереж

затверджена наказом по університету від “ 22 ” листопада 2024 р. № 1237 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 20 січня 2025 р.

3. Вхідні дані до роботи

1) Моделі та методи для управління мережевими інформаційними потоками

2 Методи балансування інформаційними потоками

3) Перелік використаних програмних та апаратних засобів: OpNet 14, NS-3

4. Перелік питань, що потрібно опрацювати у роботі

1) Аналіз сучасного стану проблеми

2) Огляд технологій управління перевантаженням та чергами маршрутизаторів

3) Моделі управління мережним трафіком

4) Вибір програмних та апаратних засобів моделювання

5) Проведення експериментальних досліджень

6) Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 17 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз стану проблеми та сучасних методів її	26.11.24–28.11.24	
	вирішення		
2	Огляд технологій управління перевантаженням	29.11.24–04.12.24	
3	Розробка моделі управління мережним	05.12.24 –09.12.24	
	трафіком		
4	Вибір програмних та апаратних засобів	10.12.24 –18.12.24	
	моделювання		
5	Тестування запропонованого метода	19.12.24–24.12.24	
6	Оформлення пояснювальної записки	25.12.24–-03.01.25	

Дата видачі завдання 25 листопада 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Янковський О.А.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 80 с., 22 рис., 1 табл., 1 дод., 32 джерел.

БАЛАНСУВАННЯ НАВАНТАЖЕННЯ, БАГАТОШЛЯХОВА МАРШРУТИЗАЦІЯ, БАГАТОШЛЯХОВА ПЕРЕДАЧА, ОБ'ЄДНАННЯ РЕСУРСІВ, ОЦІНКА ПРОДУКТИВНОСТІ, ПІДТРИМКА TCP, ПЕРЕУПОРЯДКУВАННЯ ПАКЕТІВ, ПРОГРАМНО ВИЗНАЧЕНА МЕРЕЖА.

Метою кваліфікаційної роботи є поліпшення продуктивності комп'ютерних мереж завдяки балансуванню навантаженням на мережевих каналах зв'язку.

У невеликих мережах зазвичай використовуються пристрої Ethernet, які застосовують такі рішення, як протокол Spanning Tree Protocol (STP), щоб пересилати пакети одним шляхом без петель. Однак це запобігає використанню неактивних каналів, які можуть зменшити перевантаження та збільшити загальну пропускну здатність мережі.

В кваліфікаційній роботі пропонується метод балансування навантаженням між шляхами за допомогою програмно визначених мереж (SDN). Запропонований механізм обчислює кілька шляхів із непересічними зв'язками, які мають найменшу кількість переходів між джерелом і одержувачем. Крім того, запропонований алгоритм має функцію «контролю перемикання», яка перевіряє, чи перевищує поточне завантаження шляху відсоток його пропускної здатності, і чи потенційний новий шлях, має заповненість принаймні на відсоткове значення, менше, ніж значення поточного шляху.

ABSTRACT

Master’s thesis: 80 pages, 22 figures, 1 tables, 1 appendices, 32 sources.

LOAD BALANCING, MULTIPATH ROUTING, MULTIPATH TRANSMISSION, PACKET REORDERING, PERFORMANCE EVALUATION, RESOURCE POOLING, SOFTWARE DEFINED NETWORK, TCP-FRIENDLY.

The goal of the qualification work is to improve the performance of computer networks by balancing the load on network communication channels.

Small networks typically use Ethernet devices that implement solutions such as the Spanning Tree Protocol (STP) to forward packets over a single path without loops. However, this prevents the use of inactive channels, which can reduce congestion and increase the overall network throughput.

The qualification work proposes a mechanism for load balancing between paths using software-defined networking (SDN). The proposed mechanism calculates multiple paths with disjoint connections that have the fewest hops between the source and the destination. In addition, MLB has a “switch control” function that checks whether the current load on a path exceeds a percentage of its bandwidth, and whether a potential new path has a load at least a percentage lower than the current path.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ACK – підтвердження нового пакету TCP (англ., New TCP packet acknowledgment)

BCMS – багатоадресне планування обмеженого перевантаження (англ., Bounded Congestion Multicast Scheduling)

DSLB – динамічне балансування навантаження на сервер (англ., Dynamic Server Load Balancing)

ECMP – багатошляховий шлях рівної вартості (англ., Equal-cost Multipath)

FACK – направлене підтвердження (англ., Forward Acknowledgements)

IP – Інтернет-протокол (англ., Internet protocol)

MIF – багатоінтерфейсний (англ., Multiple Interfaces)

MPTCP – багатошляховий TCP (англ., Multipath TCP)

NAMA – адаптивна багатошляхова агрегація мережевого рівня (англ. Adaptive Multipath Aggregation)

IP – Інтернет-протокол (англ., Internet protocol)

ISDN – цифрова мережа інтегрованих послуг (англ., Integrated Services Digital Network)

RTT – час подвійного оберту (англ., Round trip time)

SDN – програмно-конфігуровані мережі (англ., Software-Defined Networking)

TCP – протокол керування передачею (англ., Transmission Control Protocol)

UDP – протокол дейтаграм користувача (англ., User Datagram Protocol)

ВСТУП

Інтернет спочатку був розроблений як «мережа з двома з’єднаннями», щоб гарантувати, що жоден збій не спричинить втрату зв’язку будь-якої частини мережі, яка не має збою. По суті, будь-яка пара джерело-приймач повинна підтримувати більше одного шляху, щоб забезпечити надійність і відмовостійкість мережі. Різноманітні послуги, які пропонуються в Інтернеті через комерційні хмари або в корпоративних мережах, все більше споживаються користувачами.

Таке зростання, як правило, сприяє збільшенню основної мережевої інфраструктури з впровадженням нових хостів і, як наслідок, нових комутаторів та маршрутизаторів, необхідних для розширення мережі. Загалом із розширенням мережі створюються нові зв’язки, які створюють надлишковість шляхів. Таким чином, маючи надлишкові канали зв’язку, їх можна використовувати не лише в ситуаціях недоступності, але й у повсякденному балансуванні навантаження між ними. Такий тип використання надлишковості дає змогу збільшити пропускну здатність трафіку даних шляхом поєднання пропускної здатності двох або більше мережевих шляхів.

1 МЕТА КВАЛІФІКАЦІЙНОЇ РОБОТИ

Незважаючи на те, що найчастіше в комп’ютерній мережі існує багато ресурсів, вони не використовуються повністю. Причина полягає в тому, що за замовчуванням звичайний стек TCP/IP використовує лише один «найкращий» шлях відповідно до певних показників маршрутизації, інші доступні шляхи залишаються в режимі очікування лише для цілей резервного копіювання та відновлення.

У невеликих мережах зазвичай використовуються пристрої Ethernet, які застосовують такі рішення, як протокол Spanning Tree Protocol (STP), щоб пересилати пакети за одним шляхом без петель. Однак це запобігає використанню неактивних каналів, які можуть зменшити перевантаження та збільшити загальну пропускну здатність мережі.

Таким чином, метою магістерської кваліфікаційної роботи є поліпшення продуктивності комп’ютерних мереж завдяки застосуванню різноманітних методів балансування навантаженням на мережевих каналах.

2 ПЕРЕДУМОВИ ПОЯВЛЕННЯ БАГАТО ШЛЯХОВОЇ МАРШРУТИЗАЦІЇ

2.1 Що таке Multipath?

Як зазначалося раніше, початковий Інтернет був розроблений як «мережа з двома зв’язками» з урахуванням різноманітності шляхів. Тим не менш, комп’ютери з декількома мережевими інтерфейсами не були безпосереднім пріоритетом розробки на ранній стадії. Тільки маршрутизатори були оснащені кількома фізичними мережевими інтерфейсами. Проте з того часу Інтернет значно розвинувся. Наприклад, сьогодні більшість серверів оснащено декількома мережевими інтерфейсами.

Велика кількість мережевих ресурсів із домену сервера підштовхнула впровадження багатошляхової передачі в мережах центрів обробки даних. У сфері споживчої електроніки поширення мобільних пристроїв, оснащених інтерфейсами стільникового зв’язку (наприклад, 4G і LTE) і WiFi, представлених смартфонами, призводить до зростання кількості багатодомових хостів в Інтернеті. Таким чином, існує невідповідність між одношляховим транспортуванням і безліччю доступних мережевих шляхів. Цим багатоінтерфейсним пристроям потрібна багатошляхова здатність для підвищення продуктивності та стійкості наскрізного зв’язку.

Тим часом технологічний прогрес зробив багатошляхову передачу можливою в теорії. Це надає деякі основні переваги:

- надійність;
- агрегація смуги пропускання;
- справедливість і оптимальність за Парето;
- безпека.

Багатошляхова передача може підвищити надійність передачі даних, оскільки додаткові шляхи можуть підтримувати з’єднання у разі відмови каналу або менш продуктивного шляху.

У безпровідному середовищі надійність можна додатково підвищити, оскільки перешкоди сигналу зведені до мінімуму завдяки використанню гетерогенних методів безпровідного доступу.

Очікується, що агрегація смуги пропускання може потенційно помножити пропускну здатність на кількість доступних шляхів. Якщо таким чином можна досягти ефективного агрегування пропускної здатності, багатоінтерфейсний пристрій може отримати набагато кращу продуктивність.

Справедливість TCP вимагає, щоб протокол багатошляхової передачі отримував не більшу пропускну здатність спільного каналу вузького місця, ніж конкуруючий поток TCP. Це важливо, оскільки TCP є домінуючим транспортним протоколом в Інтернеті. Якщо нові протоколи отримують несправедливу пропускну здатність, вони, як правило, спричиняють такі проблеми, як коллапс перевантаження.

RP – це концепція, яка змінила уявлення про справедливість таким чином, що зробила багатошляховий зв’язок широко прийнятним на практиці. Замість незалежної обробки кожного ресурсу шляху, принцип RP підтримує покращене використання ресурсів кількох шляхів, дозволяючи окремим шляхам працювати так, ніби вони є одним великим ресурсом. Цей принцип є значним кроком до практичної кінцевої системи з підтримкою багатьох шляхів.

Оптимальність за Парето – це стан розподілу ресурсів, при якому немає альтернативного стану, який би покращив становище деяких людей, не погіршивши становище інших.

У випадку багатошляхової передачі це означає, що оновлення деяких звичайних одношляхових користувачів до багатошляхових не може зменшити пропускну здатність інших користувачів з будь-якою вигодою для оновлених користувачів.

Оскільки дані можуть розподілятися незалежними шляхами, зловмисникам буде складніше захопити весь вміст.

Багатошляхова передача також має свої проблеми, з якими можна зіткнутися з точки зору практичного розгортання:

- перевпорядкування пакетів;
- справедливість;
- сумісність;
- оптимальність за Парето;
- різноманітність шляхів;
- безпека.

Важко запланувати передачу пакетів даних за різнорідними шляхами, не викликаючи перевпорядкування та погіршення продуктивності. Надійне рішення для багатошляхової передачі має бути в змозі впоратися з будь-якою неоднорідністю шляху, коливаннями пропускної здатності або тремтінням. Воно також повинно бути в змозі мати справу з постійним перевпорядкуванням пакетів даних. Інакше у користувачів буде менше стимулів для оновлення, якщо рішення не працюватиме належним чином у певних мережевих середовищах.

Традиційно справедливість була однією з перешкод для багатошляхової передачі. Раніше це було на основі «для кожного інтерфейсу», що було несправедливо, якщо пізніше виникне загальне вузьке місце.

Наприклад, просте використання кількох потоків призведе до несправедливої частки смуги пропускання у вузькому місці; наприклад, n TCP-потоків отримують приблизно в n разів більшу пропускну здатність, ніж один конкуруючий TCP-потік.

Важко реалізувати загальне багатошляхове рішення без модифікації стандартизованих протоколів або зміни мережевого обладнання сторонніх виробників. Наприклад, на канальному рівні потрібні спеціальні налаштування (навіть обладнання) з обох сторін. На інших рівнях вище рівня зв’язку або хости, або мережі (іноді обидва) потребують оновлення, щоб підтримувати багатошляхову передачу.

MPTCP є першою пропозицією щодо багатошляхової передачі, яка вимагає оптимальності за Парето [1], але мало досвіду, наскільки добре/часто він дотримується цієї вимоги. Існують випадки, коли MPTCP може працювати гірше, ніж звичайний TCP, через неоднорідність шляху.

Різноманітність шляхів описує можливість мати кілька непересічних шляхів для досягнення пункту призначення. Користувачі очікують отримати високу пропускну здатність від використання кількох шляхів. Але якщо шляхи (або їх частина) проходять через спільне вузьке місце зв'язку, множинні потоки можуть отримати лише таку пропускну здатність, як порівнянний потік TCP завдяки гарантії справедливості.

Наразі надійне виявлення вузьких місць на практиці справді складно. Жодна окрема схема вибору шляху не може відповідати всім мережевим середовищам.

Багатошляхова передача порушила моделі довіри організацій, розміщених в одному мережевому провайдері. Наприклад, хоча поділ трафіку ускладнює пошук, брандмауери або шлюзи можуть пропустити частину даних, що передаються більш ніж одним мережевим провайдером, і, таким чином, не зможуть проаналізувати потік. Це призведе до поломки рішень безпеки, включаючи виявлення вторгнень і запобігання витоку даних.

2.2 Еволюція багатошляхової передачі

Перша стаття про TCP була опублікована в 1974 році. У наступному році було запропоновано дисперсійну маршрутизацію [2]. З цього моменту були запропоновані різні форми багатошляхової передачі. Наприклад, ідея створення можливості багатошляхового доступу до TCP була вперше запропонована в [3] як проект Інтернету в IETF у 1995 році. У 2002 році перша мережа 3G, яка стала комерційно активною, була запущена в Південній Кореї, яка сприяла поширенню мобільних пристроїв, оснащених кількома безпровідними інтерфейсами.

У 2006 році Кі та ін. [4] використовували моделювання потоку рідини, щоб продемонструвати, що багатошляховий транспорт може забезпечити не тільки надійність, але й збалансоване перевантаження стабільним чином.

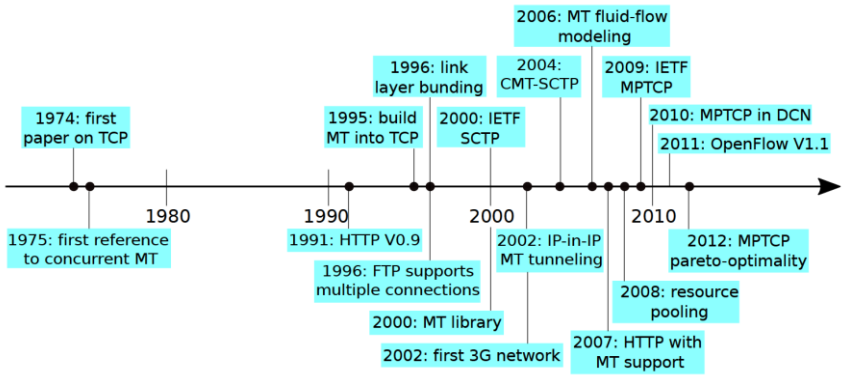

Рисунок 2.1 – Віхи в еволюції багатошляхової передачі

Кі та ін. [4] використовували моделювання потоку рідини, щоб продемонструвати, що багато шляхова передача може забезпечити не тільки надійність, але й збалансоване перевантаження стабільним чином. У тому ж році Shakkottai et al. [5] використовували некооперативну цінову гру, щоб показати, що мультидомінг перевершує уніхомінг з точки зору пропускної здатності та прибутку для постачальників послуг Інтернету (ISP).

У 2008 році Wischik та ін. [6] досліджував принцип RP, який змушує набір ресурсів поводитися як єдиний об’єднаний ресурс. Цей принцип є значним кроком до практичної кінцевої системи з підтримкою багатьох шляхів. З 2009 року IETF почала визначати та стандартизувати MPTCP, який використовує зв’язаний алгоритм контролю перевантаження для досягнення принципу RP.

3 ТИПИ БАГАТОШЛЯХОВОЇ ПЕРЕДАЧІ

3.1 З’єднання рівня каналу

Високошвидкісні робочі станції та центри обробки даних можуть легко наситити існуючі локальні мережі (LAN). На канальному рівні багатошляхову передачу зазвичай називають зв’язуванням або агрегацією каналів, оскільки кілька фізичних каналів об’єднуються (або агрегуються) в один логічний канал.

Основна мета зв’язування рівня каналу полягає в тому, щоб координувати кілька незалежних каналів між фіксованою парою систем, забезпечуючи віртуальне з’єднання з більшою пропускною здатністю, ніж те, що може підтримувати одне з’єднання. На рисунку 3.1 показаний спрощений приклад агрегації каналів між двома комутаторами Ethernet (SW1 і SW6).

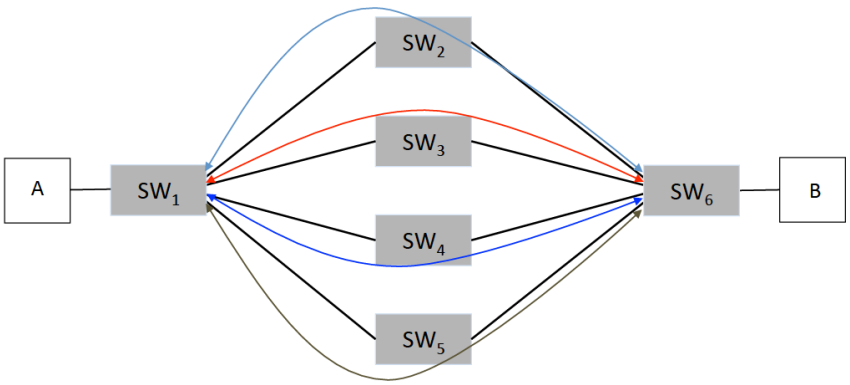

Рисунок 3.1 – Приклад агрегації каналів між двома комутаторами Ethernet

Ці комутатори можуть отримати підвищену пропускну здатність шляхом розподілу даних між кількома інтерфейсами.

У наступному обговоренні ми використовуємо Таблицю III і Таблицю IV, щоб узагальнити основні алгоритми та підходи відповідно.

Таблиця 3.1 – Ключові алгоритми канального рівня

Алгоритм	Проблеми для вирішення	Опис
WRR (Зважений Round Robin)	Розподіл навантаження	Призначений для кращого розподілу даних за шляхами з різними можливостями.
FLSA (алгоритм справедливого розподілу навантаження), FQA (алгоритм справедливого черги)	Розподіл навантаження	FLSA отримується шляхом перетворення операцій алгоритму чесної черги (FQA) у зворотньому порядку.
PCA (Розподіл на з’єднання)	Розподіл навантаження, перевпорядкування пакетів	Розподіляє кадри для кожного з’єднання.
PFA (розподіл на потоки)	Розподіл навантаження, перевпорядкування пакетів	Розподіляє трафік за принципом «потік за потоком». Наприклад, трафік, що належить одному TCP-потоку
ECT (дерево рівних витрат)	Розподіл навантаження	Забезпечує переадресацію за найкоротшим шляхом у сітчастій мережі Ethernet.
ECMP (рівноцінний багатошляховий доступ)	Розподіл навантаження	Пересилання пакетів до одного пункту призначення може бути різними кількома «шляхами».
RR (Round Robin)	Розподіл навантаження	Алгоритм впорядковує шляхи та надсилає кожен фрагмент даних на доступний шлях у циклічному порядку.

Слід зазначити, що алгоритми в таблиці 3.1 можуть бути не тільки прийнятними підходами на канальному рівні, але також можуть використовуватися на інших рівнях.

Багатоканальний PPP (MP) [7], розроблений для інтегрованих послуг цифрової мережі (ISDN), об’єднує декілька каналів за допомогою протоколу PPP [8]. Щоб виявити втрату та розлад фрагментів, MP використовує 4-байтовий заголовок повторного упорядкування (RSH) для синхронізації та виявлення втрачених фрагментів на стороні адресата. Таким чином, на приймальній стороні потрібен буфер переупорядкування для розміщення непорядкових фрагментів, викликаних агрегацією каналів.

MP пропонує схему планування Weighted Round Robin (WRR), щоб дані могли розподілятися пропорційно швидкості передачі каналів. Для досягнення цієї мети було запропоновано два методи фрагментації. Перший ділить пакети на сегменти з розмірами, пропорційними швидкості передачі різних шляхів. Інший метод ділить пакети на безліч невеликих фрагментів однакового розміру та розподіляє кількість фрагментів пропорційно швидкості передачі різних шляхів.

Adiseshu та ін. [9] додали рівень «strIPe», віртуальний IP-інтерфейс під рівнем IP і над рівнем каналу даних, для агрегування кількох каналів даних. Показано, як алгоритм FQA можна перетворити на алгоритм справедливого розподілу навантаження (FLSA), а потім запропонновано, щоб FQA мав працювати у зворотному порядку алгоритму розподілу навантаження, щоб вирішити проблему розподілу навантаження зі змінним розміром пакета. Це означає, що ідентичне обладнання (або того самого постачальника) потрібне з обох сторін агрегації. Також розглянуто проблему доставки FIFO для двох окремих випадків. Наприклад, якщо RSH можна додати до кожного пакету, проблему можна вирішити за допомогою додаткового номера переупорядкування; якщо неможливо додати заголовок, запропонувано спосіб синхронізації у випадку втрати кадру для забезпечення доставки квазі-FIFO.

FatVAP [10] є ще однією роботою в безпровідному середовищі для з'єднання каналів. FatVAP агрегує пропускну здатність, доступну для кількох точок безпровідного доступу (WAP), до яких варто підключитися, і балансує їх навантаження, плануючи трафік до різних точок доступу відповідно до їх доступної пропускної здатності. Щоб продовжувати забезпечувати сумарну пропускну здатність, доступну для всіх точок доступу, FatVAP використовує постійну оцінку як наскрізної, так і безпровідної смуги пропускання для реагування до змін протягом кількох секунд. FatVAP використовує стратегію Per-Flow Allocation (PFA) для розподілу трафіку між точками доступу.

Наприклад, коли надходить новий потік, FatVAP визначає, якій AP призначити цей потік, і записує відображення в хеш-таблицю. Наступні пакети в потоці просто надсилаються через точку доступу, записану в хеш-таблиці.

У специфікаціях IEEE протокол керування агрегацією каналів (LACP) дозволяє об'єднувати декілька каналів Ethernet для формування групи агрегування каналів (LAG).

Таким чином, клієнт керування доступом до медіа (MAC) може розглядати LAG як єдиний канал. LACP дозволяє мережевому пристрою погоджувати автоматичне групування каналів, надсилаючи пакети LACP одноранговому пристрою.

У LACP збирач кадрів (FC) на приймачі відповідає за підтримку будь-якого обмеження впорядкування кадрів. Щоб уникнути перевпорядкування кадрів, розподільник кадрів (FD) у відправника передає всі кадри, які складають дане з'єднання, лише на один канал, що є стратегією розподілу за з'єднаннями (PCA) (дуже схожою на стратегію PFA).

Тому в FC не потрібна схема переупорядкування кадрів або буфер переупорядкування. На додаток до стандартів агрегації каналів IEEE існує ряд власних схем агрегації, включаючи EtherChannel від Cisco, Aggregated Ethernet від Juniper, Multi-Link Trunking від AVAYA. Ці запатентовані схеми агрегації та стандарти IEEE 802.3ad дуже схожі та досягають однієї мети.

Починаючи з версії 1.1 [11], OpenFlow підтримує багатоканальне агрегування на рівні 2. У специфікації комутатора OpenFlow було введено Link Aggregation (LA), щоб отримати можливість для одного порту вказувати на групу інших портів. Використовуючи LACP для обміну динамічною інформацією між пристроями, що підтримують LA, контролер OpenFlow має повний контроль над перемикачами розподілу та збирання кадрів на кількох каналах.

Було виявлено, що LACP на комутаторах OpenFlow забезпечує дещо нижчу пропускну здатність, ніж на звичайних комутаторах. Таким чином, комутатори OpenFlow потребують додаткової оптимізації для досягнення еквівалентної продуктивності.

В [12] представлено адаптивну багатошляхову архітектуру пересилання в мережі центру обробки даних OpenFlow рівня 2. В архітектурі між крайовими вузлами завчасно встановлюються шляхи пересилання від усіх до всіх.

Сукупна пропускна здатність досягається за рахунок одночасного використання всіх доступних шляхів.

Щоб уникнути проблеми з доставкою поза чергою через використання всіх доступних шляхів, використовується алгоритм планування в стилі PCA. Зокрема, алгоритм виключає шляхи, довжина шляху яких значно перевищує довжину найкоротшого шляху.

За останні кілька років у стандартах IEEE та IETF з’явилися значні нові протоколи, розроблені для підтримки багатошляхової переадресації на канальному рівні, наприклад, Shortest Path Bridging (SPB) [13] та IETF Transparent Interconnection of a Lot of Links (TRILL) [14].

SPB і TRILL є потенційними наступниками протоколу Spanning Tree Protocol (STP). SPB тепер підтримує багатошляхову переадресацію за допомогою стратегій маршрутизації Equal Cost Multipath (ECMP) і Equal Cost Tree (ECT). Наразі TRILL підтримує багатошляхову переадресацію лише за допомогою стратегії маршрутизації ECMP.

Як у стратегіях ECMP, так і в стратегіях ECT, якщо до пункту призначення існує декілька шляхів з однаковою вартістю, мережевий трафік розподіляється між цими кількома шляхами.

Щоб уникнути проблем із перевпорядкуванням і виявленням MTU шляху, подібно до FatVAP, і TRILL, і SPB використовують багатошляхове пересилання для кожного потоку.

Зокрема, кадри, що належать одному потоку даних, проходять один і той же шлях, а кадри, що належать іншим потокам, можуть проходити іншими шляхами.

З таблиці 3.1 можна побачити, що основними проблемами, які намагаються вирішити алгоритми, є розподіл навантаження та переупорядкування пакетів.

Серед усіх алгоритмів найпростішим є Round Robin (RR), коли відправник розподіляє фрагменти між усіма доступними каналами рівними частинами та впорядкованим чином. У довгостроковій перспективі планування RR забезпечує справедливу частку фрагментів, якщо ці фрагменти мають однаковий розмір. Тим не менш, базова стратегія планування RR рідко використовується на практиці, оскільки вона не передбачає розподілу навантаження з фрагментами змінного розміру або різними пропускними можливостями каналів. Щоб вирішити ці проблеми, варіації зваженого RR є більш широко використовуваними стратегіями планування.

Основна перевага зв’язування канального рівня полягає в тому, що швидкість передачі сигналу каналу відносно стабільна і може бути використана для пом’якшення перевпорядкування.

Однак підходи канального рівня працюють лише на з’єднанні точка-точка і навіть вимагають встановлення спеціальних карт Ethernet з обох сторін. Таким чином, вони не застосовуються в загальних сценаріях наскрізного зв’язку, коли різні залучені домени контролюються різними постачальниками.

3.2 Агрегація пропускної здатності рівня IP

Рівень IP, спочатку запропонований для обробки глобальної адресації та маршрутизації, є природним кандидатом на розміщення можливості багатошляхового зв’язку для покращення наскрізного зв’язку. Перевага мережевого підходу полягає в тому, що він прозорий для транспортних протоколів і додатків, що значно полегшує широке розгортання. Теоретично кожен пакет TCP-потоку можна надіслати іншим шляхом, а протокол IP гарантує, що всі пакети досягнуть місця призначення.

Наприклад, Sun та ін. [15] досліджували використання багатошляхової маршрутизації для зменшення затримки передачі файлів у безпровідній мережі.

Зокрема, запропоновано скористатися перевагами коду стирання на рівні пакетів (наприклад, коду цифрового фонтану) для передачі файлу даних із надлишковістю по набору шляхів. Було отримано інтуїтивне розуміння компромісу між швидкістю кодування та зменшенням затримки.

Дослідження було проведено для спеціального мережевого середовища, де пара джерела та одержувача має багатий набір ідентичних і непересічних шляхів, отже, проблема перевпорядкування пакетів не виникає. Тим не менш, у більшості практичних мережевих середовищ, коли пакети всередині одного з’єднання проходять більше ніж один шлях, вони можуть мати різну затримку розповсюдження та надходити не в порядку. Одержувач TCP надсилає повторювані підтвердження (ACK) відправнику, через що відправник TCP помилково інтерпретує переупорядкування пакетів як втрату пакета. Таким чином, TCP зазнає значного зниження продуктивності через часте перевпорядкування пакетів і використання кількох шляхів із різними характеристиками погіршує проблему.

Далі сучасний стан поділено на три категорії: інкапсуляція IP-in-IP, проходження трансляції мережевих адрес (NAT) і розділення ідентифікатора/локатора.

3.2.1 Інкапсуляція IP-in-IP

Підхід для агрегування пропускної здатності кількох IP-шляхів полягає у використанні механізмів тунелювання, які прозоро перенаправляють пакети між двома хостами на рівні маршрутизації. Наприклад, було запропоновано інкапсуляцію IP-in-IP для розділення потоку даних між кількома мережевими інтерфейсами. Як показано на рисунку 3.2, на джерелі (A) транспортний рівень збирає всі пакети так, ніби вони проходять через A1 і адресуються B1.

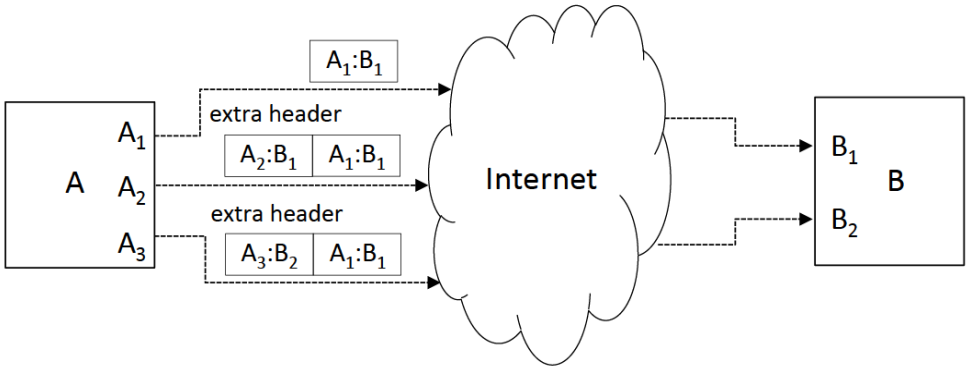

Рисунок 3.2 – Тунелювання IP-в-IP між двома хостами з декількома інтерфейсами

Пакети, що виходять на інтерфейс A2, інкапсулюються в нові IP-пакети, кожен з яких має додатковий заголовок, що має адресат B1 і джерело A2. Подібним чином, кожен пакет, що виходить на інтерфейс A3, може бути інкапсульований у новий IP-пакет із адресатом B2 та джерелом A3. Потім адресат (B) може розпізнавати пакети IP-in-IP і видаляти зовнішній заголовок. Це залишає оригінальні пакети з джерелом A1 і пунктом призначення B1, які будуть доставлені в мережевий стек до TCP у прозорий спосіб. Така ж схема інкапсуляції використовується для тунелювання в стандарті мобільного IP. Щоб уникнути швидкої повторної передачі, використовується планувальник WRR, який розподіляє пакети пропорційно ефективним швидкостям шляхів.

Також була представлена архітектуру мережевого рівня для агрегування пропускної здатності на кількох шляхах для програм реального часу [27]. Зроблено припущення, що інфраструктурний проксі (як Home-Agent у Mobile IP) знає про декілька інтерфейсів клієнта та тунелює захоплені пакети до клієнта за допомогою інкапсуляції IP-in-IP. Перевага проксі-рішення полягає в тому, що воно повністю кероване та дозволяє серверам залишатися незмінними та приховуватися за допомогою кількох IP-адрес від TCP. Було запропонувано алгоритм планування, Earliest Delivery Path First (EDPF), щоб гарантувати, що пакети відповідають своїм кінцевим термінам відтворення, плануючи пакети на основі розрахункового часу доставки пакетів. Щоб підвищити загальну продуктивність агрегації смуги пропускання на основі тунелювання IP-в-IP за допомогою мінімізації переупорядкування пакетів, запропоновано двосторонній підхід.

По-перше, політика планування EDPF на основі пари пакетів для додатків TCP (PET) була використана для розподілу трафіку на різні шляхи. Конструкція PET має ту саму концепцію EDPF, але з ідеалізованими значеннями затримки та пропускної здатності, які замінюють оцінки. По-друге, працюючи разом із політикою планування, політика керування буфером на стороні приймача (BMP) використовується для затримки пересилання пакетів, що не відповідають порядку, до TCP і виявлення втрат, щоб можна було приховати різноманітні несприятливі ефекти.

Було представлено PRISM [16], ще один підхід на основі проксі, який дозволяє TCP ефективно використовувати з’єднання WWAN. PRISM використовує міжрівневий підхід, який передбачає підтримку транспортного і мережевого рівня. Класифікується PRISM як підхід мережевого рівня, оскільки проксі-сервер PRISM, який є основним об’єктом для підтримки багатошляхового доступу, розташований на мережевому рівні.

PRISM використовує алгоритм планування пакетів, тобто адаптивний планувальник (ADAS), щоб підтримувати актуальний стан шляху. Використовуючи актуальну інформацію про стан, ADAS надсилає пакети

відповідно до очікуваного часу надходження (варіант алгоритму EDPF), щоб зменшити перевпорядкування пакетів. ADAS також використовує стан шляху для коригування ваги шляху за допомогою стратегії адитивного збільшення та мультиплікативного зменшення (AIMD) від TCP, щоб ADAS міг динамічно реагувати на перевантаження часткових шляхів і контролювати обсяг трафіку, який розподіляється на цих шляхах.

Крім того, PRISM маскує наслідки невпорядкованої доставки, ідентифікуючи фальшиві дублікати ACK і змінюючи їх послідовність, щоб відправник TCP отримував правильно послідовні ACK.

Також був розроблений інший багатошляховий мережевий протокол на основі проксі під назвою Enhancements for TCP On a Multi-homed mobile router (ETOM), який прозоро працює як для клієнтів, так і для серверів. ETOM включає два проксі-компоненти замість одного: MR (Mobile Router) і HA (Home Agent). Клієнт і MR (а також сервер і HA) використовують звичайне одношляхове з’єднання, тоді як усі пакети, що переміщуються між MR і HA, є інкапсульованими IP-в-IP.

ETOM використовує буфер переупорядкування, щоб усунути перевпорядкування пакетів. Наприклад, пакети, що не відповідають порядку, буферизуються в HA, доки не будуть отримані відсутні пакети, а потім пакети надсилаються за призначенням. ETOM також використовує варіант алгоритму EDPF для подальшого зменшення перевпорядкування пакетів.

На відміну від інших підходів рівня IP, ETOM використовує порядковий номер підпотоку у внутрішньому заголовку IP для виявлення втрати пакетів між MR і HA.

3.2.2 Використання NAT

На відміну від попереднього підходу, який спирається на інкапсуляцію IP, існує кілька підходів, які використовують переваги NAT замість тунелювання.

В [17] було представлено систему MAR (рисунок 3.3), NAT-маршрутизатор для приміського мобільного доступу, який забезпечує набір локальних інтерфейсів і ряд глобальних безпровідних інтерфейсів. Перший забезпечує доступ до локальних мобільних пристроїв, а другий підтримує різноманітні безпровідні технології глобальної зони. Маршрутизатор MAR діє як блок NAT, тобто розташований посередині і транслює IP-адреси та порти пакетів у двох напрямках. Маршрутизатор MAR може працювати окремо або співпрацювати з проксі-сервером MAR.

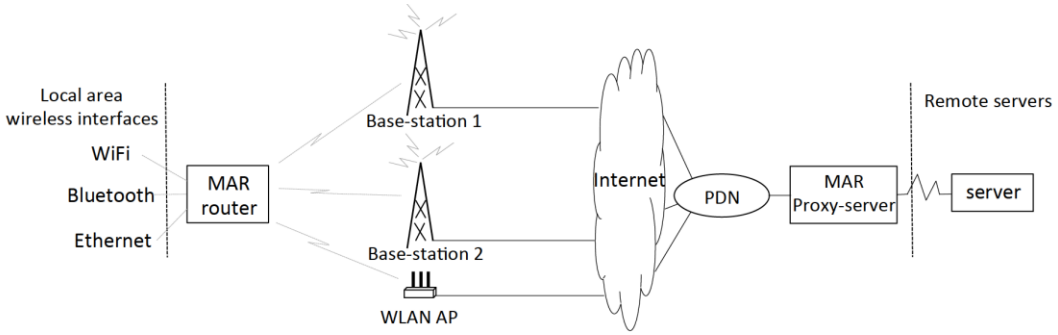

Рисунок 3.3 – Архітектура MAR системи

З таким проксі-сервером використовується пакетно-орієнтований режим планування (POSM), де пакети одного потоку TCP можуть бути доставлені кількома шляхами, а маршрутизатор MAR може реалізувати інтелектуальну оптимізацію, включаючи уникнення тристороннього рукостискання TCP, повільний старт, помилкові тайм-аути тощо.

Коли проксі-сервер відсутній, використовується режим планування, орієнтованого на потоки, де стратегія розподілу для кожного потоку планує всі пакети, що належать до того самого потоку TCP, на той самий шлях. MAR надає API, який може використовувати будь-який спеціальний протокол планування. Але сам протокол планування не є частиною архітектури MAR. MAR також призначений для визначення ваги, яку слід призначити кожному інтерфейсу для правильного балансування навантаження (наприклад, динамічного переміщення навантаження з каналів

низької якості на канали кращої якості). Маршрутизатор MAR передбачалося розмістити в транспортних засобах, що рухаються, де користувачі зможуть використовувати свої пристрої для перегляду веб-сторінок і потокового аудіо/відео. Таким чином, балансування трафіку відбувається лише в одному напрямку (тобто від віддалених серверів до локальних пристроїв).

Також в [18] було запропонувано архітектуру OSCAR, яка працює в розподіленому середовищі. Вузол із підтримкою OSCAR може спільно використовувати пропускну здатність, доступну від сусідів із підтримкою OSCAR, для з'єднання як зі старими, так і з підтримкою OSCAR. OSCAR має модуль NAT з обох сторін з'єднання. У відправника модуль NAT замінює IP-адреси джерела та призначення на IP-адреси, що використовуються для передачі. Отримавши пакет, модуль NAT на одержувачі змінює IP-адреси джерела та призначення, замінюючи їх узгодженими, перш ніж доставити пакет до TCP. Коли з'єднання проходить через спільного сусіда зі старим сервером, сусід також повинен мати модуль NAT, який виконує операцію трансляції адреси. OSCAR використовує модуль переупорядкування пакетів для вирішення проблем із переупорядкуванням пакетів. Зокрема, він затримує пакети та їхні ACK з обох сторін відповідно перед тим, як пересилати їх на верхній рівень.

Деякі підходи на основі інкапсуляції IP-в-IP і NAT, припускають наявність у мережі проксі-інфраструктури. Тим не менш, такі підходи працюють лише для зв'язку TCP у відкритому вигляді та не працюють за наявності шифрування IPsec або механізмів автентифікації.

3.2.3 Протокол ідентифікації хоста

(HIP) [19] було запропоновано та реалізовано для забезпечення підтримки багатоадресного доступу для відновлення після відмови з можливістю балансування навантаження на основі потоку. Вводиться додатковий рівень адресації, щоб дозволити змінювати IP-адреси на

мережевих інтерфейсах, зберігаючи при цьому постійні ідентифікатори транспортного рівня. Цей протокол дозволяє потокам IP-пакетів динамічно змінювати шляхи за наявності збою каналу. Він природним чином захищає наявність кількох шляхів від транспортного та прикладного рівнів, представляючи лише глобальну ідентифікацію однорангового хоста.

SIMA є розширенням HIP для використання мультиінтерфейсності для призначення окремих з'єднань TCP незалежно для різних шляхів. Подібно до FatVAP, TRILL, SPB, MAR і OSCAR, SIMA також використовує стратегію багатошляхового пересилання, де створюються правила зв'язування потоків, щоб визначити використання локальних інтерфейсів. SIMA не визначає жодних додаткових політик надсилання чи отримання для пом'якшення проблеми з переупорядкуванням, замість цього він використовує модуль обробки пакетів IPsec Encapsulating Security Payload (ESP), вбудований у HIP для обробки кожного пакета даних, розробив і реалізував Multipath HIP (mHIP), багатошляховий планувальник, заснований на HIP, для розподілу трафіку по декількох доступних шляхах.

Використовуючи алгоритм планування EDPF, розділяються пакети в TCP-з'єднанні на кілька шляхів, щоб пом'якшити перевпорядкування пакетів. Тим не менш, алгоритм EDPF ефективний лише проти перевпорядкування пакетів зі стабільними шляхами з точки зору пропускної здатності та затримки. Щоб реагувати на динамічні характеристики шляху, використовується техніка маркування як частина схеми уникнення багатошляхового перевантаження, щоб зміни характеристик шляху можна було виявити за один час проходження туди й назад (RTT).

3.3 Багатошляхова передача на транспортному рівні

У порівнянні з підходами на основі IP-рівня, підходи на транспортному рівні мають певні невід'ємні переваги, оскільки контроль перевантаження можна використовувати як механізм для розподілу ресурсів у мережі.

На цьому рівні кінцеві системи можуть легко отримати інформацію про кожен шлях: затримку, рівень втрат і стан перевантаження. Потім ця інформація може бути використана для реагування на перевантаження в мережі шляхом переміщення трафіку подалі від перевантажених шляхів.

Сучасні транспортні протоколи, орієнтовані на з'єднання, наприклад TCP, протокол передачі керування потоком (SCTP) і протокол керування перевантаженням дейтаграм (DCCP), передають дані лише по одному шляху між джерелом і пунктом призначення в будь-який момент часу. Були зроблені численні спроби налаштувати ці існуючі транспортні протоколи для забезпечення багатошляхового доступу.

Подібно до агрегації пропускної здатності на мережевому рівні, одночасна багатошляхова передача на транспортному рівні підвищує частоту перевпорядкування пакетів через різні характеристики шляху, включаючи пропускну здатність під час виконання, RTT, втрати та помилки. Зокрема, якщо з'єднання розділено на кілька мережевих шляхів, загальна пропускна здатність потенційно може бути навіть гіршою, ніж пропускна здатність, доступна на будь-якому з шляхів. Є дві основні причини цього.

Перша походить від впливу неоднорідного RTT. Наприклад, TCP очікує доставку пакетів через мережу за принципом «перший прийшов - перший вийшов». Перевпорядкування пакетів на одержувачі призводить до отримання дублікатів ACK у відправника. Відправник швидко повторно передає «відсутній» пакет, який ще може бути в транзиті на шляху з великим RTT. Другою причиною є блокування буфера прийому через неоднорідність шляху або збій шляху.

3.3.1 SCTP на основі багатошляхової передачі

SCTP – це одноадресний транспортний протокол загального призначення, орієнтований на підключення. Підключення SCTP означає асоціацію.

Дані користувача сегментуються на блоки так званих блоків даних, які ідентифікуються унікальними порядковими номерами передачі (TSN). Механізм вибіркового підтвердження SACK використовується за замовчуванням для підтвердження отриманих фрагментів даних і звітування про прогалини (тобто відсутні фрагменти даних, зазначені їхніми TSN) відправнику.

На відміну від TCP, SCTP було розроблено з урахуванням того, що асоціація SCTP дозволяє використовувати кінцеві точки джерела та призначення з кількома адресами. Незважаючи на це, SCTP використовує лише один основний шлях і перемикається на інший шлях для повторної передачі втрачених пакетів або як резервний у разі збою основного шляху.

SCTP використовує єдину структуру буфера на обох кінцевих точках, але підтримує кілька станів для кожного пункту призначення: окреме вікно перевантаження (cwnd), поріг повільного старту (ssthresh), таймер повторної передачі та оцінку RTT.

Було запропонувано BA-SCTP, протокол агрегації пропускної здатності на основі SCTP. BA-SCTP реалізує механізм для виявлення вузьких місць, які є спільними для потоків з одного агрегатного з’єднання. На основі цього механізму BA-SCTP виконує уніфікований алгоритм контролю перевантажень для потоків, які мають однакове вузьке місце, замість того, щоб застосовувати контроль перевантаження для кожного потоку окремо.

Такий підхід гарантує, що потоки BA-SCTP є справедливими з іншими потоками TCP, які мають теж саме вузьке місце. BA-SCTP використовує стратегію планування в стилі WRR, стратегію розподілу даних на основі вікна перевантаження, де кожен підпоток отримує дані зі спільного буфера надсилання щоразу, коли у підпотоку є вікно перевантаження для надсилання даних.

Також було запропоновано W-SCTP, SCTP зі елементами Westwood [20] для використання агрегації пропускної здатності. Контроль перевантажень у стилі TCP Westwood може повністю використати переваги

оцінки пропускної здатності, яку можна використовувати для розподілу трафіку між декількома потоками. W-SCTP використовує планувальник у стилі EDPF, який вибирає шлях для наступного пакета, передбачаючи, чи зможе він якнайшвидше доставити пакет до місця призначення.

CMT-SCTP використовує функцію мультиінтерфейсу від SCTP для правильної передачі даних між кінцевими хостами з декількома адресами. Було визначено три негативні побічні ефекти CMT і запропонувано алгоритми для їх усунення відповідно.

По-перше, було запропонувано алгоритм SFR-CACC для усунення непотрібних швидких повторних передач шляхом використання іншої інтерпретації інформації SACK. По-друге, було використано алгоритм зростання вікна перевантаження (cwnd), щоб відстежувати найраніший незавершений TSN для кожного пункту призначення та оновлювати cwnd, навіть за відсутності нових ACK. По-третє, було запропоновано новий алгоритм із затримкою ACK для CMT-SCTP, а саме із затримкою ACK для CMT (DAC). Алгоритм дозволяє одержувачу затримувати надсилання ACK сегмента, що не відповідає порядку.

Була проведена значна робота над ядром CMT-SCTP, щоб подолати його недолік і підвищити продуктивність. Було виявлено, що всі п’ять політик повторної передачі можуть спричинити зниження пропускної здатності через блокування буфера прийому. Цю проблему блокування також називають головним блокуванням (HLB). Рисунок 3.4 ілюструє його спрощений приклад.

Як показано на рисунку, приймач CMT-SCTP підтримує єдиний буфер прийому, який спільно використовується двома потоками субасоціації в асоціації. C1 (фрагмент 1) передається через шлях 2 і втрачається через затор або збій шляху. Протягом періоду часу повторної передачі C1 буфер прийому не може вмістити будь-які інші пакети через контроль потоку, тому загальна пропускна здатність знижується. Щоб вирішити проблему, було запропоновано складену політику повторної передачі параметрів під назвою

RTX-LCS. Вона обмежує вибір шляху повторної передачі, враховуючи три загальні умови: cwnd, ssthresh і рівень втрат. Розділення буфера використовується, щоб уникнути того, щоб один шлях займав надто багато буферного простору, що перешкоджає іншим шляхам надсилати нові пакети.

Інтелектуальна швидка ретрансляція справляється з помилковими спалахами швидкої ретрансляції. Наприклад, не враховуються фрагменти, переміщені з іншого шляху, у рішенні про швидкі повторні передачі по новому шляху.

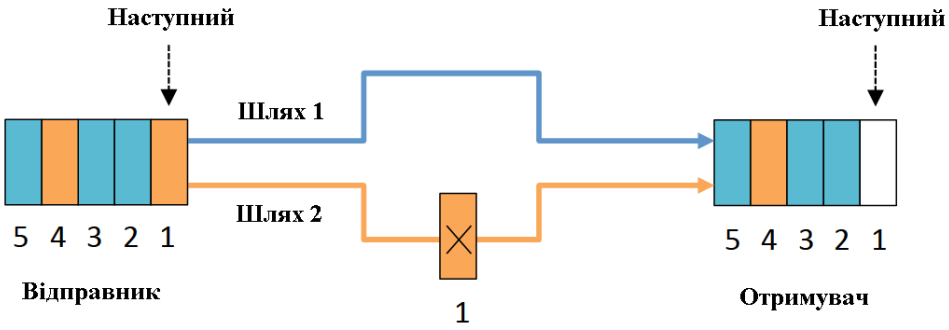

Рисунок 3.4 – HLB: буфер прийому більше не може вмістити інші фрагменти до прибуття головного блоку

Було представлено оптимізовану методику обробки буфера для подальшого підвищення продуктивності. Зокрема, було запропоновано динамічно використовувати спільний буферний простір, щоб швидший шлях міг мати можливість надсилати більше даних, надаючи йому більше буферного простору.

Крім того, було запропоновано використовувати функцію Multi-streaming SCTP для пом’якшення проблеми HLB. Зокрема, кожному повідомленню присвоюється ідентифікатор, який вказує на потік.

За допомогою цього ідентифікатора протоколу потрібно лише відновити послідовність повідомлень. Отже, після втрати пакета лише повідомлення уражених потоків повинні бути відкладені, щоб відновити послідовність.

В [21] було запропоновано Concurrent Multipath TCP (cmpTCP) – розширення SCTP. cmpTCP розділяє пакети одночасно по всіх доступних шляхах із спільного буфера надсилання. cmpTCP підтримує віртуальну чергу повторної передачі (RTxQ) на кожному шляху, щоб контролювати кількість незавершених байтів на шляху. Одержувач надсилає ACK тим самим шляхом, за яким отримано пакети. Ці дві конструкції можуть допомогти ігнорувати помилкові звіти про розриви та усунути непотрібні повторні передачі пакетів. Також розроблена марковська модель cmpTCP для оцінки швидкості транспортування даних на кожному шляху, коли передача досягає стабільного стану. cmpTCP використовує планувальник у стилі WRR, враховуючи кількість незавершених байтів і розмір вікна перевантаження.

Розглянута раніше робота щодо CMT-SCTP виконує незалежний контроль перевантажень на кожному шляху та мало враховує справедливість щодо інших одношляхових потоків. Наприклад, у випадку n шляхів CMT-SCTP асоціація отримає в n разів більшу частку пропускної здатності конкуруючого потоку SCTP або TCP без CMT через те саме вузьке місце.

Концепція RP [6] є важливою віхою для контролю агресивності багатошляхової передачі. У контексті багатошляхової передачі це дозволяє набору ресурсів поводитися як єдиний ресурс, балансуючи трафік між кількома шляхами. Як показано на рисунку 3.5, коли кілька потоків TCP конкурують за один шлях, TCP може справедливо розділити пропускну здатність шляху між ними. Коли потоки проходять більш ніж одним шляхом, шляхи розглядаються як єдиний об’єднаний ресурс. За умови відповідної координації передача трафіку може переміщатися від більш завантажених шляхів до менш завантажених, і можна впоратися з більшими сплесками.

Було запропоновано контроль перевантаження RP для CMT-SCTP, позначений як CMT/RP-SCTP, шляхом поєднання CMT-SCTP з концепцією RP. Метою CMT/RP-SCTP є покращення пропускної здатності даних, залишаючись при цьому справедливим до одночасних одноканальних потоків у спільному вузькому місці.

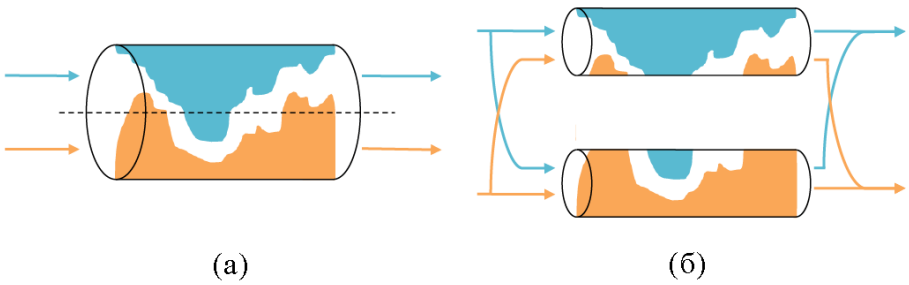

Рисунок 3.5 – Один шлях справедливо розподіляє свій ресурс між конкуруючими потоками (а), кілька шляхів розглядаються як єдиний об’єднаний ресурс (б)

Наприклад, коли два шляхи використовуються одночасно для передачі даних і вони мають спільну вузьку лінію, загальна пропускна здатність, отримана асоціацією CMT/RP-SCTP, має бути подібною до пропускної здатності стандартного SCTP. Однак CMT/RP-SCTP передбачає подібні шляхи, тобто шляхи, що мають дуже схожі характеристики з точки зору пропускної здатності, затримки та рівня втрат.

Була досліджена поведінка CMT/RPv2-SCTP на різних шляхах і виявлено, що CMT/RPv2-SCTP досягає цілей RP. Крім того, порівняно з MPTCP, CMT/RPv2-SCTP рівномірно розподіляє пропускну здатність між потоками, коли це можливо, незалежно від кількості шляхів, що використовуються для транспортування.

3.3.2 Багатошляхова передача на основі TCP

На відміну від SCTP, який був розроблений із підтримкою мультиінтерфейсних пристроїв, TCP не знає кількох інтерфейсів і дозволяє лише одну IP-адресу для кінцевої точки. Незважаючи на це, TCP домінував в Інтернет-трафіку та викликав багато інтересів у тому, щоб TCP підтримував одночасну багатошляхову передачу. SCTP і MPTCP мають багато схожих проблем і певних алгоритмів.

Magalhaes [22] запропонував надійний транспортний протокол мультиплексування (R-MTP), який є надійним на основі швидкості мультиплексування даних транспортного протоколу через кілька мережевих інтерфейсів (тобто планувальник у стилі WRR).

Він ґрунтується на явному дослідженні пропускної здатності за допомогою методу пар пакетів для оцінки пропускної здатності з метою відповідного налаштування швидкості на доступних шляхах. Наприклад, він вимірює час між надходженнями пакетів і тремтіння, щоб визначити дефіцит пропускної здатності. Період зондування має відбуватися в точному часовому масштабі, щоб відобразити коливання доступної смуги пропускання.

Паралельний TCP (pTCP) функціонує як оболонка модифікованої версії TCP. Він відкриває кілька потоків TCP, по одному для кожного використовуваного інтерфейсу. pTCP виконує розділення даних між кількома мікропотоками (потоками TCP), враховуючи різницю в їх пропускній здатності. Зокрема, pTCP використовує співвідношення cwnd/RTT, планувальник WRR-PULL, для розподілу трафіку пропорційно пропускній здатності шляху.

Крім того, pTCP має кілька інших стратегій вирішення конкретних проблем. Наприклад, вікно перевантаження може бути завищеним, особливо безпосередньо перед виникненням заторів. Це може призвести до небажаної затримки даних у підпотоках. Замість того, щоб пізніше перепризначати дані іншим підпотокам, pTCP використовує стратегію відкладеного прив’язування, щоб адаптуватися до миттєвих змін пропускної здатності шляху. Зокрема, він отримує дані зі спільного буфера надсилання лише тоді, коли заплановано негайне надсилання даних через підпоток.

Щоб уникнути переповнення приймального буфера, pTCP використовує стратегію Packet Re-striping для повторної передачі пакета через інший підпоток замість того підпотоку, який передав цей пакет раніше, і використовує Redundant Striping для надсилання дубльованого пакета.

Крім того, pTCP використовує механізм зворотного зв'язку SACK для відновлення втраченого пакета за набагато коротший період часу.

Протокол керування прийомом (RCP) [23] – це транспортний протокол, орієнтований на отримувача, з мінімізованою конструкцією відправника. Приймач контролює всі ключові функції в RCP. Для підтримки CMT було запропоновано розширення RCP із кількома станами, тобто радіальний RCP (R2CP). R2CP підтримує один RCP-канал (те саме, що й TCP-потік) на наскрізний шлях, а керування перевантаженням здійснюється окремими RCP-каналами. Трафік планується до кожного каналу RCP на основі (розрахункового) часу, коли запитуваний сегмент прибуде через відповідну канал (варіант EDPF).

Кожен канал RCP підтримує внутрішній простір локального порядкового номера, щоб полегшити виявлення втрат і відновлення. Локальний порядковий номер можна перетворити на глобальний порядковий номер і навпаки.

Багатошляховий протокол керування передачею M/TCP використовує One-Way-Trip Time (OWTT), аналогічний метод FPS, у відправника для оцінки часу затримки прямого шляху, щоб обчислити таймер RTO для кожного шляху. Крім того, M/TCP використовує два механізми для боротьби з втратою пакетів. У разі швидкої повторної передачі M/TCP використовує політику повторної передачі, тобто дублює відсутній сегмент і надсилає кожну копію всіма шляхами, щоб отримати бажану швидку та надійну повторну передачу; у разі тайм-ауту повторної передачі, відсутній сегмент у потоці надсилається через інші потоки.

Крім того, M/TCP використовує два алгоритми для передачі ACK у випадку CMT. А саме, використовуючи дубльований алгоритм ACK, приймач M/TCP надсилає ACK відразу після отримання сегмента даних по більш ніж одному шляху; використовуючи дубльований і затриманий ACK, приймач передає ACK для всіх інших сегментів даних більш ніж одним шляхом.

Приблизно в 2009 році було запропоновано MPTCP [24] з урахуванням властивості справедливості, а також функції RP. Зокрема, MPTCP, який обговорюється IETF, має наступний набір цілей:

- покращення пропускної здатності – MPTCP має працювати принаймні як один потік TCP, що працює на найкращому шляху;
- підпотоки MPTCP не повинні займати більше пропускної здатності, ніж окремий потік TCP у спільному вузькому місці;
- баланс перевантаження – MPTCP має найбільше використовувати найменш перевантажений шлях.

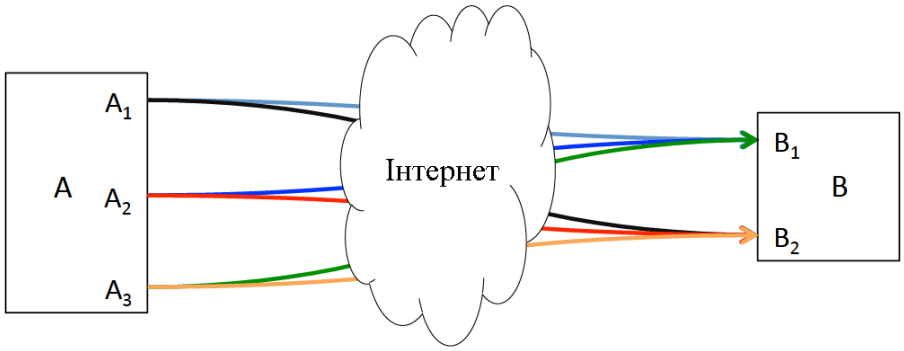

Рисунок 3.6 – Можливі мережеві шляхи між двома хостами з підтримкою MPTCP

MPTCP представляє стандартний API сокетів TCP для верхнього рівня, щоб застарілі програми могли працювати на MPTCP прозоро. Алгоритм пов’язаного контролю перевантаження, Linked Increase Algorithm (LIA) використовується для гарантування справедливого розподілу ресурсів на кількох шляхах і надання функції RP між ними. Його функція RP може перенаправляти трафік із більш завантажених шляхів на менш завантажені. На додаток до спільного алгоритму контролю перевантажень, MPTCP також має кілька інших конструктивних особливостей. Наприклад, MPTCP додає порядкові номери на рівні підключення, щоб повторно зібрати потік даних у порядку з кількох підпотоків.

На ранній стадії MPTCP використовується стратегія планування WRR у стилі PUSH, коли планувальник намагається заповнити всі підпотоки, коли з програми надходять дані.

Стратегії планування PUSH і WRR-PULL є варіантами WRR. Їх відмінність полягає в тому, що стратегія WRR-PULL використовує менше часу очікування в черзі підпотоку перед його фактичною передачею.

Однією з проблем розгортання MPTCP у центрах обробки даних є ефект Incast, поведінка MPTCP, а також TCP, що призводить до грубого недовикористання пропускної здатності зв’язку в певних моделях трафіку.

На продуктивність MPTCP впливає не лише буфер прийому, а й буфер відправлення. Виявлено, що в MPTCP-з’єднанні з декількома підпотоками з високим BDP може статися блокування буфера надсилання

3.4 Багатошляховий доступ на прикладному рівні

Надання можливості багатошляхового доступу на прикладному рівні привертає багато уваги, оскільки такий підхід майже не залежить від базових технологій доступу та маршрутизації мережевого рівня. Зазвичай програма встановлює кілька транспортних з’єднань, прив’язує їх до різних IP-адрес і розподіляє дані пропорційно доступній пропускній здатності цих з’єднань. Щоб повторно зібрати дані, що доставляються через різні з’єднання, кожному пакету або групі пакетів зазвичай призначаються додаткові порядкові номери.

Такий доступ розподіляється на чотири групи:

- перша група – підходи, що використовують один шлях;
- друга група – підходи з використанням різних шляхів;
- третя група – підходи на основі HTTP, поєднання HTTP і багатошляхової передачі;
- четверта група використовує підходи проміжного програмного забезпечення.

Методи з використанням різних шляхів, спрямовані на збільшення пропускної здатності додатків за допомогою кількох TCP-з’єднань через один і той же фізичний шлях. Тим не менш, якщо вони використовуються для розподілу даних по різних фізичних шляхах, проблема переупорядкування в приймачі зробить їх неефективними, оскільки вони не враховують проблему переупорядкування, спричинену неоднорідними шляхами.

У 2000-х роках дослідники почали шукати рішення для забезпечення агрегації пропускної здатності за різними фізичними шляхами. Простий підхід для досягнення цієї мети полягає в тому, щоб безпосередньо додати підтримку кількох інтерфейсів до певної програми, відкривши кілька сокетів TCP (по одному для кожного активного інтерфейсу) і виконавши розподіл даних між різними сокетами. Якщо інтерфейси клієнта підключені до незалежних мереж, одночасне використання кількох шляхів може досягти загальної пропускної здатності, близької до суми всієї пропускної здатності окремих інтерфейсів. З огляду на те, що популярні файли часто копіюються на кількох серверах, для клієнтів стає природним паралельне підключення до кількох дзеркальних серверів для отримання файлу (тобто за принципом «багато до одного»). Було виявлено, що багатошляхове потокове передавання має кращі характеристики втрат, ніж одноканальне потокове передавання.

Звичайний TCP/IP завжди використовує один «найкращий» шлях відповідно до певних показників маршрутизації, навіть якщо між двома кінцевими точками може бути більше одного шляху. Така поведінка призводить до недостатнього використання доступних мережевих ресурсів. Розповсюдження мобільних пристроїв, оснащених декількома інтерфейсами, представлених смартфонами, призводить до зростання кількості багатоінтерфейсних хостів в Інтернеті. Таким чином, це погіршує невідповідність між одношляховим транспортуванням і безліччю доступних мережевих шляхів.

4 БАЛАНСУВАННЯ НАВАНТАЖЕННЯ МІЖ ШЛЯХАМИ

Віртуальні послуги, які пропонуються в Інтернеті через комерційні хмари або в корпоративних мережах через корпоративні хмари, все більше споживаються користувачами. У наш час з’являється все більше нових програм, які пропонують необхідні послуги як для професійної діяльності, так і для розваг.

В кваліфікаційній роботі пропонується алгоритм балансування навантаженням між шляхами, з використанням SDN (Software Defined Networks). Модифікований алгоритм (МА) можна застосовувати в невеликих мережах, переважно з великою кількістю непересічних шляхів.

Той факт, що мережі SDN мають логічно централізовану архітектуру, дозволяє контролеру мати глобальне уявлення про топологію мережі, і завдяки цьому можна обчислити всі шляхи між джерелами та адресатами. Ще одна важлива характеристика полягає в тому, що традиційні балансувальники навантаження не програмуються, оскільки вони мають набори інструкцій, визначені власною архітектурою їх виробників.

Ця функція дозволяє мережевому адміністратору створювати інтелектуальні додатки, використовуючи дані, зібрані з мережі, які можуть приймати рішення, наприклад, стосовно маршрутизації мережевого потоку. Використання сучасного підходу, такого як SDN, дозволяє подолати перевантаження площини даних шляхом оптимізації трафіку даних, коли вибираються менш завантажені шляхи.

Мережі Ethernet покладаються на STP (Spanning Tree Protocol) для створення охоплюючого дерева лише з одним шляхом, що спричиняє пересилання трафіку даних з однієї точки в іншу в мережі без петель, створених надлишковими шляхами. Завдяки глобальному погляду на мережу, контролер SDN робить використання STP непотрібним і бере на себе роль керування мережевими шляхами.

Алгоритм MA базується на механізмі балансування навантаження, запропонованому Сетом у Seth [25]. Однак MA відрізняється від алгоритму Сета тим, що він має функцію під назвою «switch control», яка використовує деякі припущення для виконання зміни шляху, уникаючи модифікацій, які призводять до дуже неоптимальних рішень.

Контроль комутації перевіряє, чи поточна зайнятість шляху, тобто; найбільший обсяг даних, переданих і отриманих між ланцюгами шляху, перевищує відсоток його пропускної здатності, і чи обчислений потенційний новий шлях має заповненість принаймні на відсоткове значення, менше за заповненість поточного шляху. Крім того, MA виконує обчислення шляху з непересічними зв’язками на основі алгоритму Едмондса та Карпа [26]. Цей алгоритм виконує обчислення шляху за допомогою алгоритму пошуку в ширину, щоб знайти всі можливі шляхи між джерелом і пунктом призначення на графі. Потім він використовує теорію максимального потоку шляхом застосування методу Форда-Фулкерсона для вибору шляхів, які не мають спільних каналів [26]. Механізм Сета використовує непересічні зв’язки на основі адаптації алгоритму Дейкстри.

4.1 Використання багатошляхової маршрутизації

Методи багатошляхової маршрутизації використовують фізичні ресурси традиційної мережі, використовуючи кілька шляхів між джерелом і одержувачем для надсилання трафіку даних. Три основні компоненти багатошляхової маршрутизації це:

- обчислення шляхів;
- розподіл трафіку;
- вибір шляху.

Ці компоненти також можна застосовувати в контексті SDN, коли комутатор SDN може використовувати інформацію з різних рівнів і кілька шляхів для пересилання пакетів.

Обчислення шляху має на меті знайти всі існуючі шляхи між джерелом і пунктом призначення. Щоб процес пошуку був ефективним, алгоритм обчислення шляху повинен мати глобальні знання про топологію мережі. Після виявлення топології алгоритму необхідно визначити шляхи від джерела до пункту призначення на основі трьох сценаріїв.

Сценарій 1. Непересічні вузли – шлях складається з вузлів (окрім джерела та адресата), які не використовуються іншими шляхами. У разі відмови від спільного використання вузлів також не буде спільного використання каналів, за винятком широкомовних каналів. Ця конфігурація забезпечує більшу відмовостійкість, оскільки шляхи повністю незалежні. Однак це передбачає більші витрати на інфраструктуру, оскільки для отримання більшої кількості шляхів необхідно створити більше вузлів і каналів.

Приклад цього сценарію показано на рисунку 4.1, який представляє граф, що складається з 10 вузлів і 14 ребер, з вузлом «A» як джерелом і вузлом «J» як пунктом призначення двох потоків, які слідують шляхами, які є незалежними та вільними. від спільного використання вузлів і ребер.

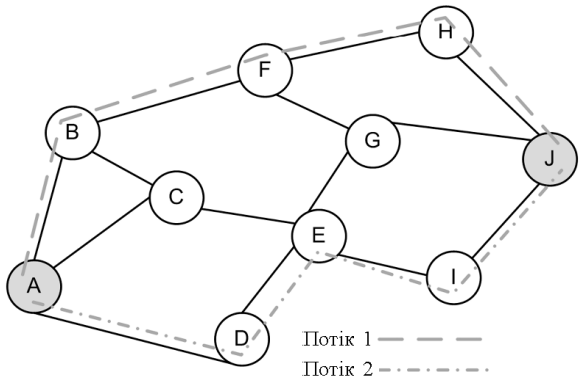

Рисунок 4.1 – Приклад сценарію обчислення шляху з непересічними вузлами

Сценарій 2. Непересічні зв’язки – шлях складається з вузлів, які можуть бути спільними для інших шляхів, але канали не є спільними. Цей сценарій має меншу відмовостійкість порівняно з попереднім сценарієм. Це

пояснюється тим, що коли вузол виходить з ладу, це впливає на всі шляхи, які його використовують. Те саме не станеться, якщо збій станеться в каналі. Використання непересічних каналів може допомогти збільшити загальну пропускну здатність і зменшити перевантаження мережі.

Приклад цього сценарію представлено на рисунку 4.2, де є три потоки, що проходять шляхи, які не мають спільних зв’язків, але вузол «E» є спільним для потоків 2 і 3.

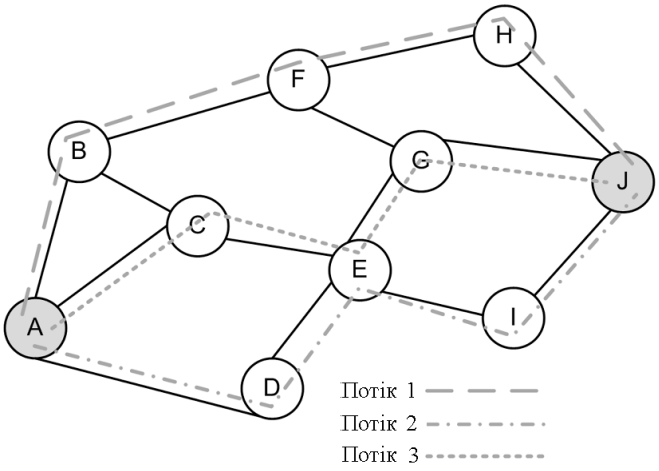

Рисунок 4.2 – Приклад сценарію обчислення шляху з непересічними ребрами

Сценарій 3. Пепересічні зв’язки: приклад зображено на рисунку 4.3, як вузли, так і ребра (канали) шляху можуть бути спільними для інших шляхів у мережі. Відсутність обмежень на використання загальних вузлів і каналів полегшує обчислення шляху. Однак цей сценарій вважається найгіршим із трьох, оскільки якщо вузол або канал виходить з ладу, цей збій вплине на всі шляхи, які спільно використовують такий фізичний ресурс.

Продуктивність алгоритму при обчисленні шляху залежить від кількості вузлів і зв’язків у топології. Встановлення ідеальної кількості шляхів може зменшити складність обчислень. Крім того, слід уникати пересічних шляхів, оскільки спільне використання вузлів і каналів означає низьку відмовостійкість.

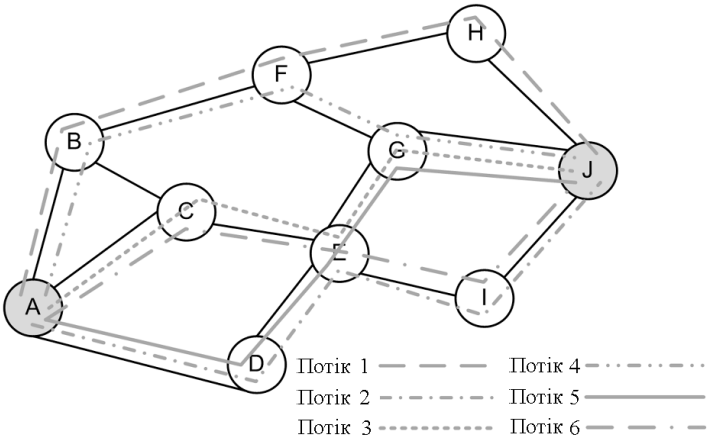

Рисунок 4.3 – Приклад сценарію обчислення шляху з пересічними вузлами та ребрами

Ще одна проблема стосується пропускної здатності, оскільки шляхи зі спільними каналами також матимуть спільну пропускну здатність.

Механізми балансування навантаження, представлені в кваліфікаційній роботі, використовують непересічні шляхи у випадку алгоритму Сета та шляхи з непересічними каналами у випадку запропонованого алгоритму.

Розподіл трафіку може здійснюватися на різних рівнях: пакет, потік, субпотік, суперпотік та субсуперпотік.. У кваліфікаційній роботі використовується розподіл трафіку за потоком на основі таких ідентифікаторів пакетів:

- IP джерела;
- IP призначення;
- протоколу;
- порту джерела;
- порту призначення.

Після визначення ідентифікатора потоку комутатори SDN використовують його під час пересилання пакетів. Вибір шляхів можна класифікувати відповідно до типу використаного селектора:

- Round robin – трафік пересилається всіма обчисленими шляхами в циклічній послідовності;

- інформація про пакет – трафік пересилається на основі інформації, що міститься в заголовку пакета;

- стан трафіку – може враховувати трафікове навантаження, пропускну здатність каналу, обсяг трафіку або кількість активних потоків на шляху;

- стан мережі – може враховуватися розмір черги, затримка, тремтіння або втрата пакетів на шляху.

Реалізації балансування навантаження, що використовуються в цій роботі, ґрунтуються на виборі шляхів за станом трафіку. Алгоритм MA (Multipath Load Balance) використовує пропускну здатність шляху, тобто обсяг інформації про дані, передані та отримані в 1-секундному інтервалі комутаторами SDN шляху.

Алгоритм Сета використовує обсяг трафіку на шляху, тобто обсяг даних, переданих та отриманих інтерфейсами комутаторів, які становлять шлях протягом 2-секундного інтервалу.

4.2 Алгоритми балансування навантаженням

В кваліфікаційній роботі використовуються два алгоритми балансування навантаженням між шляхами: модіфікований під назвою MA і алгоритм Сета.

Два алгоритми діють на прикладному рівні та взаємодіють із мережевим контролером за допомогою його північного інтерфейсу.

Крім того, вони використовують бібліотеку «NetworkX», написану на Python, для створення, обробки та вивчення графіків і мереж.

У нашому випадку вузли – це прості елементи (комутатори SDN), які використовують ідентифікатори потоку для пересилання пакетів. Традиційний канальний рівень 2 пересилає пакети, використовуючи лише MAC-адресу для вузла призначення, і не виконує балансування навантаженням.

4.2.1 Реалізація алгоритму Сета

Ця реалізація базується на алгоритмі Дейкстри для обчислення непересічних множинних шляхів, які мають найменшу кількість переходів між джерелом і одержувачем. Алгоритм працює лише для одного потоку з балансованням навантаженням. Крім того, перемикання враховує обсяг даних на поточному шляху, порівнюючи його з обсягом обчислених шляхів. Ми використовуємо механізм Сета, оскільки його вихідний код доступний за загальною публічною ліцензією [25]. Псевдокод алгоритму, отриманий з вихідного коду, представлено в лістингу 4.1.

Лістинг 4.1 – Псевдокод алгоритму Сета

```
begin
  src = source host
  dst = destination host
  while true do
    connected_nodes = proc_topology()
    G = networkx.graph(connected_nodes)
    paths = networkx.all_shortest(G, src, dst, dijkstra)
    for i from 1 to number(paths) do
      t1 = returns_data_tx_rx(OpenDaylight_restconf, paths[i])
      wait 2s
      t2 = returns_data_tx_rx(OpenDaylight_restconf, paths[i])
      path_cost[i] = t2 - t1
    end
    best_path = returns_index(min(path_cost))
    configure_flow(paths[best_path], source, destination)
  end
end
```

Вихідний і кінцевий хости визначаються на початку виконання, як показано в рядках 2 і 3. Реалізація алгоритму виконує балансування навантаження лише для одного потоку. У рядку 4 починається структура повторення, яка забезпечує постійне виконання програми. У рядку 5 функція «proc_topology()» запитує у контролера мережі інформацію про з’єднання, встановлені між комутаторами мережі.

Така інформація зазвичай відома контролеру, який використовує протоколи OFDP (OpenFlow Discovery Protocol) і LLDP (Link Layer Discovery Protocol) для виявлення існуючих мережевих пристроїв у топології.

У наступному рядку за допомогою інформації, отриманої від контролера, мережевий додаток збирає глобальний граф топології мережі. У рядку 7 структура графа «G» обробляється функцією «all_shortest» на основі алгоритму Дейкстри. У результаті всі обчислені шляхи між джерелом і одержувачем повертаються та зберігаються в змінній «path».

У рядку 8 починається цикл, який повторюватиметься з урахуванням кількості обчислених шляхів. Усередині циклу, у рядку 9, «t1» отримує від контролера в поточний момент часу інформацію про обсяг даних, переданих і отриманих інтерфейсами комутаторів, які складають шлях, і після 2 секунд очікування, «t2» також отримує від контролера ту саму інформацію про обсяг даних, як показано в рядку 11.

Потім у рядку 12 значення “t2” віднімається на “t1” і результат зберігається у векторі «path_cost». Змінна «best_path» отримує шлях із найменшим обсягом записаних даних у рядку 14. Ідея полягає в тому, щоб використовувати шлях із меншою кількістю переданих і отриманих даних протягом цього 2-секундного інтервалу. І, нарешті, у рядку 15 програма передає контролеру інформацію про новий шлях, який має використовувати поток.

4.2.2 Впровадження алгоритму

Реалізація алгоритму базується на алгоритмі Едмондса та Карпа [26], який обчислює кілька роз’єднаних шляхів між джерелом і пунктом призначення. Їхній алгоритм застосовує алгоритм пошуку в ширину, теорію максимального потоку та мінімальний розріз зв’язків із пропускною здатністю, що дорівнює 1, для обчислення найкоротших непересічних шляхів між джерелом і пунктом призначення.

Реалізація МА відрізняється від алгоритму Сета не лише типом обчислення шляху, але й кількістю потоків, які вона може обробляти. Крім того, МА має функцію «управління комутацією», яка перевіряє, чи перевищує поточне заняття шляху відсоткове значення (назване «condition1») його пропускної спроможності, і чи потенційний новий шлях, обчислений МА, має зайнятість принаймні на відсоткове значення («condition2») менше, ніж у поточного шляху. Псевдокод о алгоритму представлено в лістингу 4.2.

Лістинг 4.2 – Псевдокод модифікованого алгоритму

```
begin
  src = source switch
  dst = destination switch
  no_flows = number of flows
  path_capacity = 10,000,000
  current[flow] = ∅
  while true do
    connected_nodes = proc_topology()
    G = networkx.graph(connected_nodes)
    for flow from 1 to no_flows do
      paths = networkx.edge_disjoint(G, src, dst, edmonds_karp)
      for i from 1 to number(paths) do
        paths_cost[i] =
        calculate_occupation(OpenDaylight_restconf, paths[i])
      end
      best_path = returns_index(min(paths_cost))
      if current[flow] = ∅ then
        configure_flow(paths[best_path], src, dst, flow)
        current[flow] = best_path
      end
      else if paths_cost[current[flow]] ≥ condition1×
              path_capacity & path_cost[best_path]≤condition2×
              path_cost[current[flow]] then
            configure_flow(paths[best_path], src, dst, flow)
            current[flow] = best_path
      end
    end
  end
end
```

У рядках 2 і 3 визначено комутатори джерела та призначення. Рядок 4 визначає кількість потоків, які будуть збалансовані. У рядках 5 і 6 визначені опорні значення для функції керування комутацією. У рядку 7 починається

структура повторення, яка підтримує постійне виконання програми. Рядок 10 представляє структуру повторення, залежно від кількості використовуваних потоків.

У рядку 11 структура графа G обробляється функцією «edge_disjoint()» за допомогою методу обчислення шляхів Едмондса та Карпа. У рядку 12 починається цикл, який повторюватиметься з урахуванням кількості обчислених шляхів. У цьому випадку функція «calculate_occupation»" оцінює зайнятість на обчислених шляхах, використовуючи інформацію про обсяг даних, переданих і отриманих за 1 с комутаторами кожного шляху. У цей момент мережева програма реєструє найвище навантаження даних, передане між каналами шляху, а потім повторює в усіх обчислених шляхах.

У рядку 15 змінна «best_path» отримує шлях із найменшою зайнятістю. І, нарешті, рядок 16 перевіряє, чи це перше виконання програми для поточного потоку. Якщо так, шлях потоку налаштовується в рядку 17, а «поточний» вектор отримує індекс найкращого шляху для потоку.

Якщо ні, рядок 20 представляє функцію керування комутацією, яка дозволяє перемикати поток на новий шлях, якщо поточне заповнення шляху більше або дорівнює значенню, визначеному в змінній «condition1», і новий шлях має завантаженість, яке принаймні на «condition2» менше завантаженості поточного шляху.

У випадках, коли задовольняються обидві умови, новий альтернативний шлях буде встановлено для поточного потоку в рядку 21, а «поточний» вектор отримає індекс найкращого шляху для поточного потоку, як показано в рядку 22.

4.3 Оцінка продуктивності

Розглянемо оцінку продуктивності контролера ODL, який не використовує механізми балансування навантаження, алгоритми Сета та модифікований алгоритм.

Оцінюваними показниками є:

- сукупна пропускна здатність – використання балансування навантаження має тенденцію до збільшення сукупної пропускної здатності (для його обчислення потоки використовують протокол TCP (Transmission Control Protocol);

- справедливість – використання шляху більш ніж одним потоком вважається справедливим, якщо всі потоки використовують рівні частини доступної смуги пропускання (індекс справедливості повертає значення від 0 до 1 і якщо всі потоки мають майже однакову пропускну здатність між собою, результат наближається до 1, а якщо існують значні відмінності в пропускній здатності потоку, результат має тенденцію наближатися до 0);

- втрата пакетів – втрата пакетів в основному пов’язана з перевантаженням на шляху.

Індекс справедливості [27] визначається як:

$$f(x_1, x_2, ..., x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2}, \qquad (4.1)$$

де n – загальна кількість потоків, що змагаються за пропускну здатність, а x_i – значення пропускної здатності потоку і.

Три алгоритми, використані в експериментах:

- ODL – алгоритм без балансування навантаження (з використанням контролера ODL усі потоки комутуються за шляхом за замовчуванням, визначеним у охоплюючому дереві);

- алгоритм Сета виконує балансування навантаження на основі непересічних шляхів;

- модифікований алгоритм виконує балансування навантаження на основі шляху з непересічними зв’язками та функцією керування комутацією.

Алгоритм Сета виконує балансування навантаження лише одного з потоків, у даному випадку основного потоку. Алгоритм MA може збалансувати більше потоків. Однак, щоб зробити цю оцінку справедливою, проведені експерименти розділені на дві групи, які називаються двопотоковими та чотирипотоковими експериментами.

В експериментах із двома потоками MA налаштований на балансування лише основного потоку, як у алгоритмі Сета. Крім того, використовується інший потік, який відповідає фоновому трафіку, і цей трафік не збалансований. Для експериментів з двома потоками використовуються два типи моделей трафіку:

- модель TCP-2f складається з двох потоків, головний потік має хост «д1» в якості джерела, а хост «а5» на порту TCP/5001 в якості пункту призначення, а джерелом фонового потоку є хост «д2» і відповідно «а6» як пункт призначення на порту TCP/5002 (рисунок 4.4) (тривалість потоків зафіксована на рівні 600 с);
- модель UDP-2f: складається з двох потоків, головний потік має хост «д1» як джерело та хост «а5» на порту UDP/5001 як місце призначення, а фоновий потік має як джерело «д2» і як пункт призначення «а6» на порту UDP/5002 (рисунок 4.4) (тривалість потоків також зафіксована на 600 с).

В експериментах із чотирма потоками MA налаштовано для балансування трьох потоків, які є основними. Четвертий потік – це фоновий трафік, який комутується шляхом за замовчуванням, створеним у охоплюючому дереві. Для експериментів із чотирма потоками використовуються два типи моделей трафіку. Модель TCP-4f: складається з чотирьох потоків, три з яких це основні потоки з джерелами, пунктами призначення та портами, визначеними таким чином:

- для потоку 1 джерелом є хост «д1», а пункт призначення – хост «а5» на порту TCP/5001;
- для потоку 2 джерелом є хост «д2», а пунктом призначення – хост «а6» на порту TCP/5002;

- для потоку 3 джерелом є хост «д3», а пунктом призначення – хост «а7» на порту TCP/5003;

- фоновий потік має хост «д4» як джерело та хост «а8» на порту TCP/5004 як місце призначення (рисунок 4.4).

Модель UDP-4f: складається з чотирьох потоків, три з яких основні потоки з джерелами, адресатами та портами, визначеними таким чином:

- потік 1, джерелом є хост «д1», а пунктом призначення є хост «а5» на порт UDP/5001;

- потік 2, джерелом є хост «д2», а пунктом призначення – хост «а6» на порту UDP/5002;

- потік 3, джерелом є хост «д3», а пунктом призначення – хост «а7» на порту UDP/5003;

- фоновий потік визначається хостом «д4» як джерелом та хостом «а8» на порту UDP/5004 як пунктом призначення (рисунок 4.4).

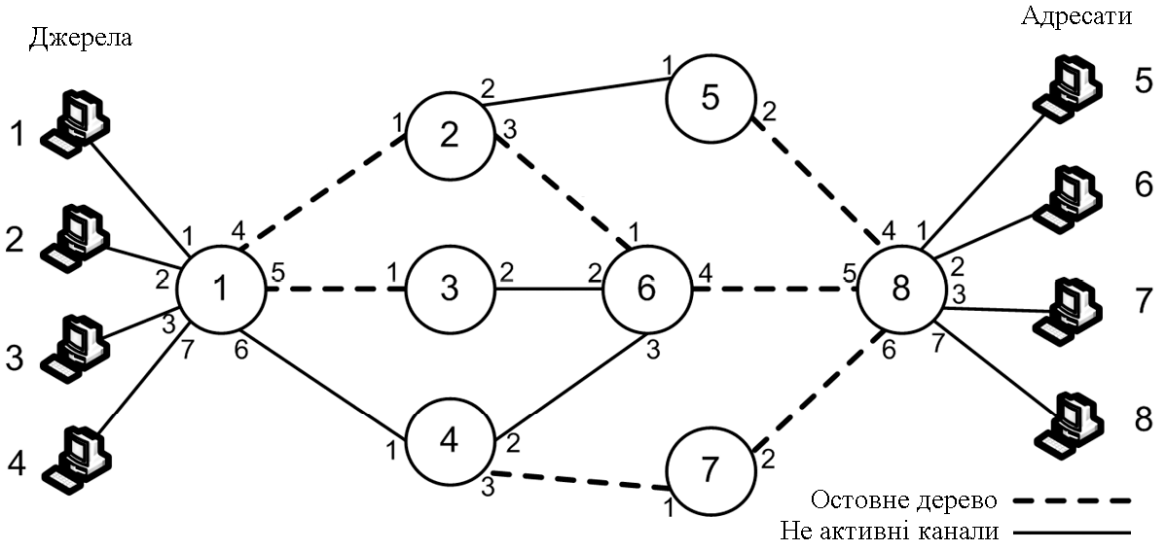

Рисунок 4.4 – Топологія експериментальної мережі

Пунктирні канали вказують на шляхи за замовчуванням, визначені охоплюючим деревом, створеним ODL, обидва пов’язані зі статусом каналів під час використання ODL.

Потоки TCP використовуються для вимірювання сумарної пропускної здатності та індексу справедливості, тоді як потоки UDP використовуються для вимірювання втрати пакетів. Потоки UDP налаштовані на передачу даних зі швидкістю 9,5 Мбіт/с; 95% пропускної здатності каналів мережі.

Цю швидкість передачі було обрано після виконання кількох тестів, під час яких вимірювалася втрата пакетів. Швидкість передачі послідовно збільшувалася, доки не було досягнуто ліміту передач без втрати пакетів; 9,5 Мбіт/с в експериментах. Оцінювалася втрата пакетів, коли мережа майже перевантажена, навіть якщо надсилається лише один потік. Усі потоки генеруються за допомогою інструменту iPerf. Кожен тест виконується 30 разів.

На рисунку 4.4 показана топологія мережі, що використовується в експериментах, яка складається з восьми вузлів, трьох непересічних шляхів і п’яти непересічних шляхів з урахуванням джерел і пунктів призначення. Імітаційне моделювання виконується в Mininet 2.3.0 [28]. Вузли мережі емулюються за допомогою віртуального комутатора Open vSwitch.

Пропускна здатність усіх каналів мережі налаштована на 10 Мбіт/с завдяки наявним обчислювальним ресурсам. Контролер OpenDaylight був обраний тому, що це відкритий проект, який уже використовується в академічному та корпоративному світі [29]. Розроблений на мові програмування Java, він має нативну програму під назвою «l2switch», що складається з модуля «Loop Remover», відповідального за генерацію охоплюючого дерева мережі, подібного до того, що створюється протоколом STP. Охоплююче дерево складається зі стандартних шляхів, які охоплюють усі вузли в мережі, і спрямоване на усунення надлишкових шляхів, які можуть спричинити петлі. ODL підтримує інвентаризацію охоплюючого дерева зі «статусом» STP «пересилання» для активних каналів і «відкидання» для неактивних каналів. У прикладі на рисунку для зв’язку хоста «д1» з хостом «а5» використовується шлях 1-2-6-8. Статус каналів можна перевірити за допомогою веб-консолі керування ODL [30].

4.3.1 Двопотокові експерименти

По-перше, параметри «condition1» і «condition2» модифікованого змінюються для оцінки їх впливу на сукупну пропускну здатність за допомогою моделі трафіку TCP-2f. Прийнята «умова1» ∈ {30%, 40%, 50%, 60%, 70%} і «умова2» ∈ {10%, 30%, 50%, 70%}.

На рисунку 4.5 показано, що немає істотного впливу обох параметрів на сукупну пропускну здатність. Таким чином, було прийнято рішення встановити першу умову на 50%, щоб перемикання могло відбуватися лише тоді, коли поточний шлях має середню або велику зайнятість смуги пропускання. Роблячи це, можна уникнути поділу великих потоків і, як наслідок, затримок, спричинених надмірним перевпорядкуванням пакетів у пункті призначення через цей поділ. Аналогічно, друга умова встановлюється на 10%, щоб уникнути перемикання між шляхами з дуже близькою пропускною здатністю, що також може сприяти появі непорядкових пакетів.

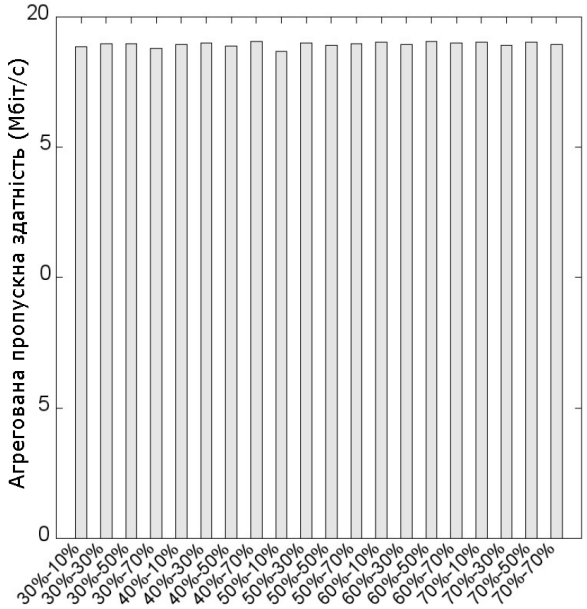

Рисунок 4.5 – Агрегована пропускна здатність для модернізованого алгоритму для різних значень пари умова 1-умова 2 та двох потоків

На рисунку 4.6 представлені результати сукупної пропускної здатності. Можна спостерігати, що використовуючи лише ODL, загальна пропускна здатність становить приблизно 10 Мбіт/с. У цьому випадку основний потік і фоновий поділяють смугу пропускання стандартного шляху, створеного ODL (остовним деревом), утвореним вузлами 1-2-6-8. Таким чином, оскільки немає балансування навантаження, сукупна пропускна здатність буде меншою або дорівнюватиме пропускній здатності стандартного шляху.

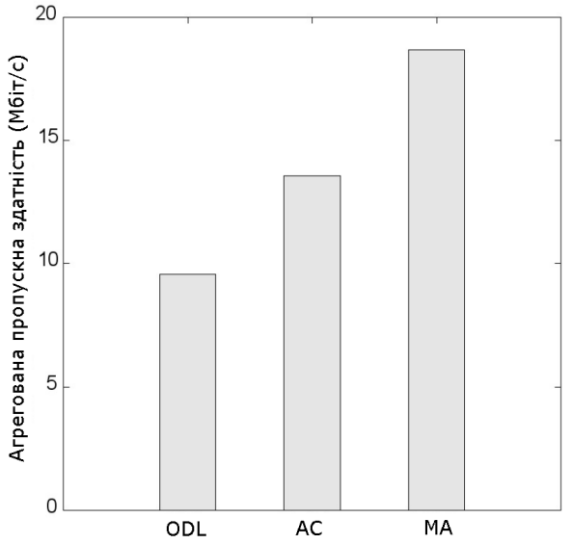

Рисунок 4.6 – Агрегована пропускна здатність для ODL та алгоритмів Сета і MA коли кількість потоків дорівнює 2

У балансуванні навантаження алгоритму Сета сукупна пропускна здатність збільшена на 41% порівняно з ODL і досягає 13,5 Мбіт/с. Це пояснюється тим, що балансування Сета виконує перемикання основного потоку на новий альтернативний шлях, який передає менший обсяг трафіку. Основний потік тепер перемикається на шлях без спільних посилань із фоновим потоком, утвореним вузлами 1-4-7-8, на шлях за замовчуванням, який використовується фоновим потоком, або на шляхи з посиланнями, спільними з сформованим шляхом за замовчуванням вузлами 1-2-5-8, 1-3-6-8 і 1-4-6-8.

Нарешті, для балансування модернізованого алгоритму сукупна пропускна здатність досягає 18,6 Мбіт/с, 37%-приріст порівняно з алгоритмом Сета.

Ця краща продуктивність пояснюється використанням шляхів з непересічними зв’язками та функцією керування перемиканням, яка дозволяє перемикатися для основного потоку лише тоді, коли зайнято 50% або більше його смуги пропускання, а новий шлях має зайняту смугу пропускання, яка становить щонайменше на 10% менше поточного шляху. Це запобігає переходу на новий шлях, рівень зайнятості якого подібний до поточного шляху.

На рисунку 4.7 представлені результати індексу справедливості з використанням моделі трафіку TCP-2f. Для ODL суперечка за використання пропускної здатності між потоками призвела до індексу справедливості 0,99, що означає, що обидва потоки мають однакову пропускну здатність: пропускна здатність фонового потоку досягає 4,8 Мбіт/с проти 4,7 Мбіт/с основного потоку.

У балансуванні навантаження Сета індекс справедливості також становить 0,99; який демонструє, що потоки мають однакову пропускну здатність, навіть із перемиканням шляхів, спричиненим балансуванням навантаження.

Це означає, що в моменти, коли фоновий потік і основний потік використовують стандартний шлях, пропускна здатність залишається приблизно рівною, а в моменти, коли основний потік перемикається на альтернативний шлях, обидва потоки також мають однакову пропускну здатність. Середня пропускна здатність для фонового потоку становить 6,8 Мбіт/с проти 6,7 Мбіт/с для збалансованого основного потоку.

Для балансування MA індекс справедливості також становить 0,99. Обидва потоки досягають 9,3 Мбіт/с. Слід зазначити, що навіть фоновий потік опосередковано виграє від балансування навантаженням, оскільки він використовує виключно шлях 1-2-6-8 протягом більшої частини часу.

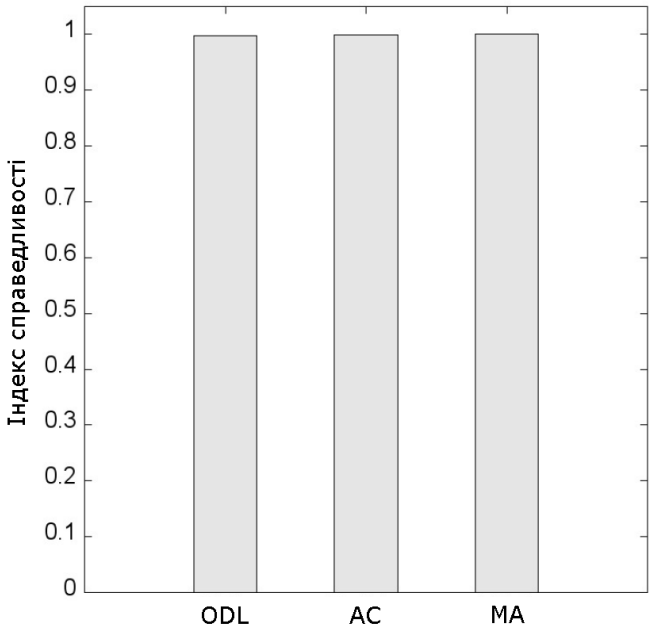

Рисунок 4.7 – Індекс справедливості для алгоритмів ODL, Сета та MA, коли кількість потоків дорівнює 2

Нарешті, на рисунку 4.8 представлені результати втрати пакетів за допомогою моделі трафіку UDP-2f. При використанні ODL втрати пакетів наближаються до 50%. Це високе значення можна пояснити тим фактом, що потоки використовують стандартний шлях, утворений вузлами 1-2-6-8. Пропускна здатність шляху за замовчуванням спільно використовується двома потоками UDP, які не підлягають контролю перевантаження. Таким чином, два хости-джерела передають потік з постійною швидкістю 9,5 Мбіт/с, що призводить до великої втрати пакетів.

При балансуванні алгоритмом Сета втрата пакетів наближається до 27%, оскільки втрата в основному виникає, коли збалансований основний потік розділяє стандартний шлях із фоновим потоком або коли він використовує альтернативний шлях із каналами, спільними за стандартним шляхом; наприклад, для шляхів, утворених вузлами 1-2-5-8, 1-3-6-8 і 1-4-6-8. З перемиканням основного потоку на шлях, утворений вузлами 1-4-7-8, потоки більше не конкурують один з одним за ширину смуги, і втрата пакетів незначна.

У MA балансова втрата пакетів становить менше 5%. Це пояснюється тим, що функція керування перемиканням дозволяє перемикання основного потоку лише тоді, коли існує різниця в 10% або більше в зайнятості поточного каналу порівняно з новим каналом.

Коли смуга пропускання шляхів зайнята однаковими рівнями навантаження, перемикання не відбувається. Таким чином, у випадку, коли основний потік слідує шляхом, який не має спільного зв'язку зі стандартним шляхом, таким як шлях, утворений вузлами 1-4-7-8, заняття шляхів відбуватиметься подібним чином. Це відбувається тому, що за допомогою керування перемиканням потоки залишатимуться довше за різними шляхами без будь-якого перемикання. І таким чином, оскільки між потоками немає суперечок щодо пропускної здатності, більше пакетів успішно доставляються до місць призначення, і, отже, втрата пакетів зменшується. MA показав втрату пакетів у 13 разів менше, ніж у алгоритму Сета, і в 24 рази менше, ніж у ODL.

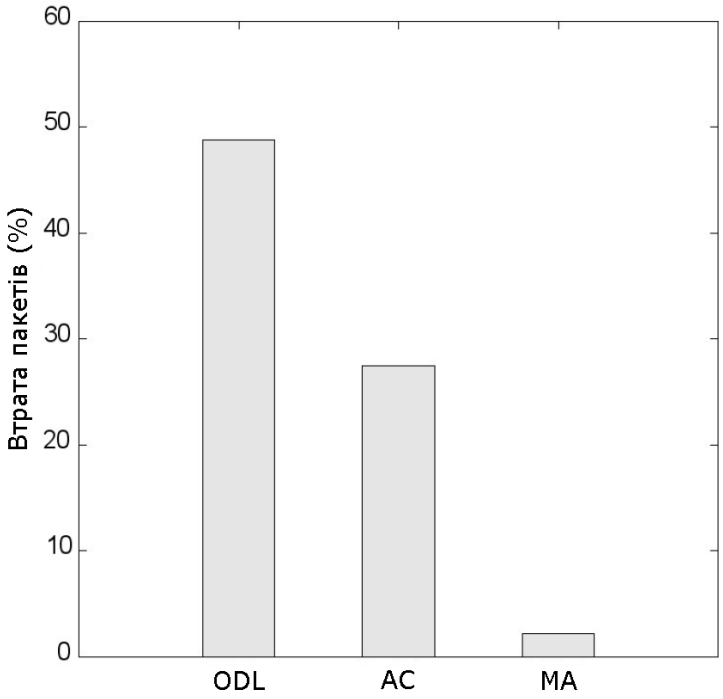

Рисунок 4.8 – Втрата пакетів для алгоритмів ODL, Сета та MA, коли кількість потоків дорівнює 2

4.3.2 Експерименти з чотирма потоками

Знову ж таки, параметри «умови1» і «умови2» алгоритму MA змінюються для оцінки їх впливу на сукупну пропускну здатність, але з використанням моделі трафіку TCP-4f. Використовується той самий набір значень, що й у експерименті з двома потоками.

На рисунку 4.9 показано аналогічні результати для сукупної пропускної здатності, за винятком випадків, коли «умова2» дорівнює 70%. Цей результат можна пояснити труднощами пошуку нового шляху з менш ніж 30% зайнятості поточного шляху, що використовується, дуже низьким значенням для сценарію зі значною кількістю потоків і, як наслідок, значним обсягом трафіку.

Зберігається перша умова рівною 50%, а друга – рівною 10% для наступних експериментів через ті ж причини, що описані в попередньому пункті.

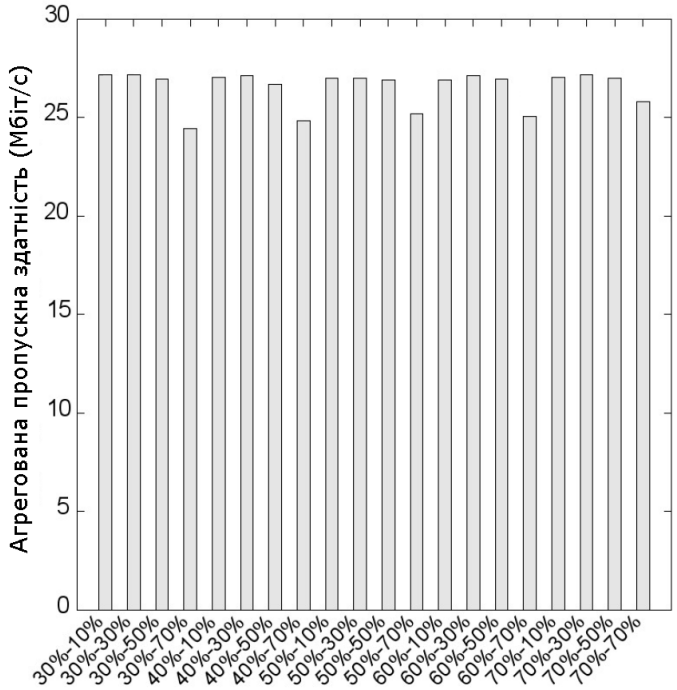

Рисунок 4.9 – Агрегована пропускна здатність для MA при використанні різних значень пари «умова1»-«умова2» і чотирьох потоках

Крім того, беручи до уваги контролер ODL і алгоритм Сета, на рисунку 4.10 представлені результати сукупної пропускної здатності. Можна спостерігати, що загальна пропускна здатність за допомогою ODL становить менше 10 Мбіт/с. Без балансування навантаження три основні потоки та фоновий потік направляються через шлях за замовчуванням.

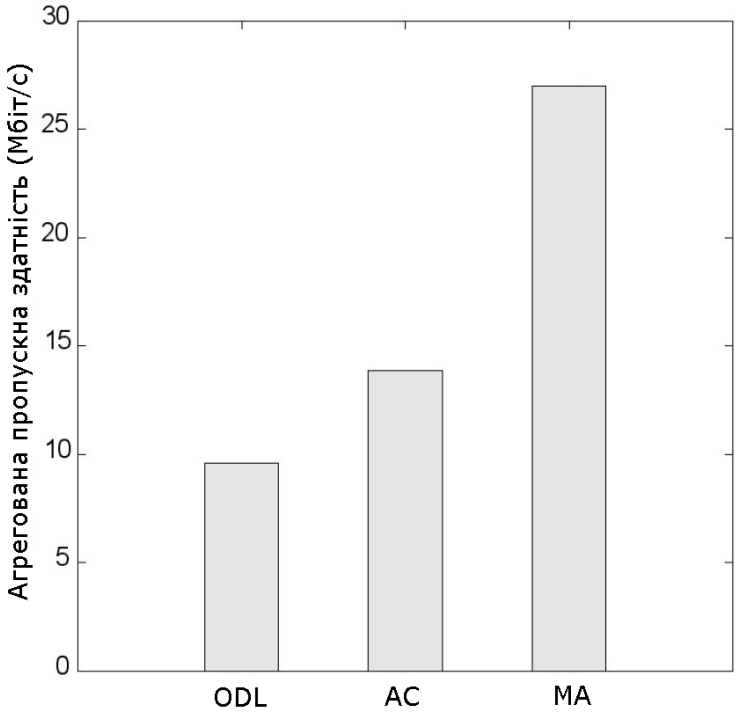

Рисунок 4.10 – Агрегована пропускна здатність для алгоритмів ODL, Сета та MA, коли кількість потоків дорівнює 4

У алгоритма Сета балансування навантаженням дозволяє сумарній пропускній здатності наближатися до 14 Мбіт/с. У цьому алгоритмі лише один із основних потоків збалансований, інші незбалансовані основні потоки маршрутизуються шляхом за замовчуванням, утвореним вузлами 1-2-6-8 разом із фоновим потоком, як можна побачити на рисунку 4.11. Крім того, під час балансування збалансований основний потік перемикається на шлях за замовчуванням, починаючи суперечку щодо пропускної здатності з іншими потоками. Це відбувається тому, що механізм завжди перевіряє, який із шляхів містить найменший обсяг переданих даних. Таким чином, будь-яка

незначна зміна значень обсягу даних змушує алгоритм Сета виконувати перемикання між ними, ситуація, яка має тенденцію відбуватися з великою частотою.

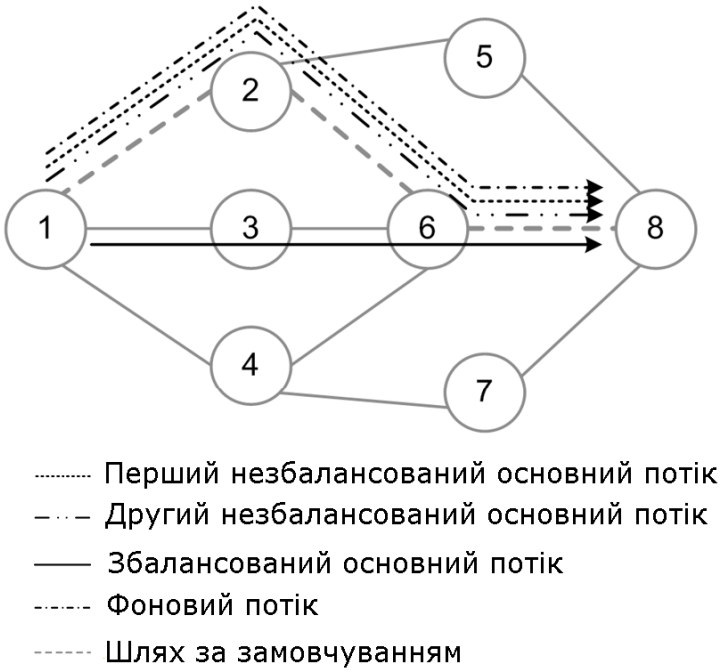

Рисунок 4.11 – Перемикання одного з основних потоків на шлях 1-3-6-8 з використанням алгоритму Сета

У алгоритмі МА сукупна пропускна здатність досягає 27 Мбіт/с. Цей результат можна пояснити наступними фактами:

- балансування навантаження застосовується до трьох основних потоків;
- обчислені шляхи не мають спільних зв’язків;
- функція керування перемиканням запобігає перемиканню, коли його робочі умови не виконуються.

Один із основних потоків має середню пропускну здатність 8,8 Мбіт/с, інші два основні потоки мають середню пропускну здатність 8,5 Мбіт/с, а фоновий потік досягає 1,2 Мбіт/с. Відповідно до прикладу перемикання шляхів, показаного на рисунку 4.12, припустимо наступне. Перший головний

потік передається шляхом, утвореним вузлами 1-4-7-8. Другий основний потік передається шляхом, утвореним вузлами 1-3-6-8, виконуючи спір щодо пропускної здатності з фоновим потоком, що складається з вузлів 6-8.

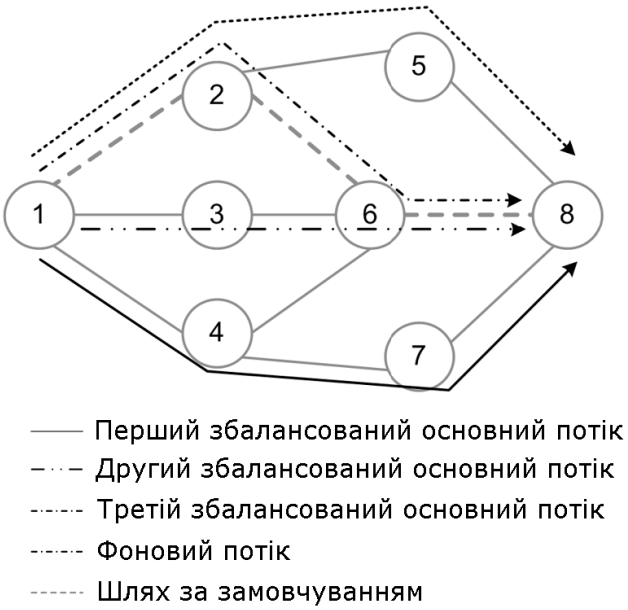

Рисунок 4.12 – Приклад перемикання для основних потоків за допомогою алгоритму MA

Третій основний потік передається шляхом, утвореним вузлами 1-2-5-8, вступаючи в суперечку з фоновим потоком у ланці, утвореній вузлами 1-2. Ця конфігурація перемикання потоків і подібні конфігурації (кожен потік слідує за іншим непересічним шляхом) забезпечують високе значення агрегованої пропускної здатності.

На рисунку 4.13 представлені результати індексу справедливості для моделі трафіку TCP-4f. ODL досягає 0,99, що означає, що чотири потоки однаково займають пропускну здатність шляху за замовчуванням. Середня пропускна здатність кожного потоку досягала 2,3 Мбіт/с.

Використання балансування навантаження алгоритму Сета призводить до індексу справедливості, близького до 0,89. Це можна пояснити тим фактом, що він балансує лише один основний потік, інші потоки

направляються через стандартний шлях, утворений вузлами 1-2-6-8. Коли збалансований потік ділиться пропускною здатністю з іншими потоками через те, що він знаходиться на стандартному шляху або тому, що він знаходиться на шляху, який має спільне з'єднання, зайнятість пропускної здатності між потоками, як правило, однакова.

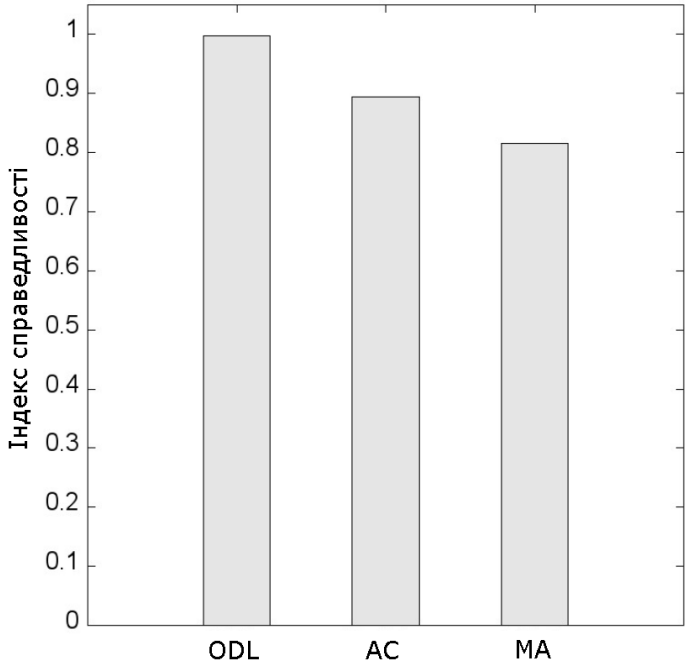

Рисунок 4.13 – Індекс справедливості для алгоритмів ODL, Сета і MA, коли кількість потоків дорівнює 4

Однак, коли збалансований потік перемикається на шлях, у якому немає спільного зв'язку з іншими потоками, наприклад шлях, утворений вузлами 1-4-7-8, як показано на рисунку 4.14, збалансований потік має тенденцію займати всю пропускну здатність цього шляху. При цьому зайнятість його смуги пропускання стає вищою порівняно з тим, що досягається іншими потоками, які продовжують спільно використовувати смугу пропускання стандартного шляху. У цьому випадку три потоки мають однакову пропускну здатність 2,8 Мбіт/с, тоді як збалансований основний потік досягає середньої пропускної здатності 5,5 Мбіт/с.

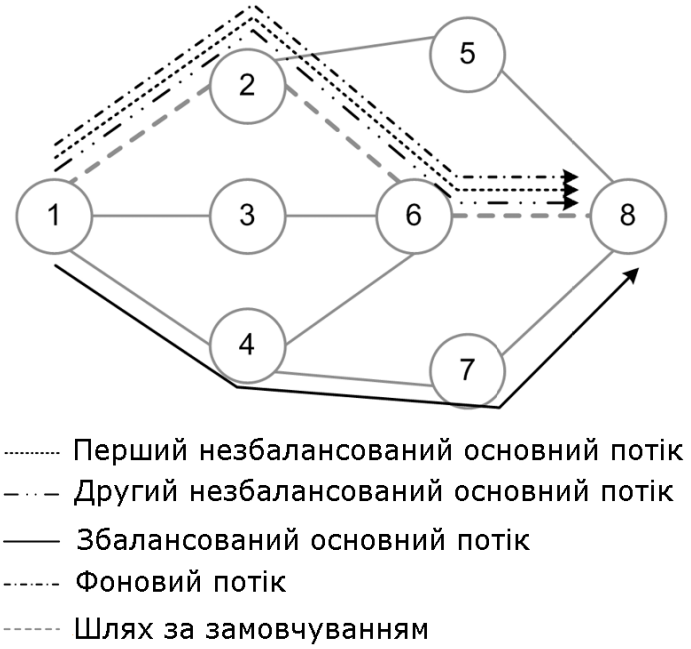

Рисунок 4.14 – Перемикання одного з основних потоків на шлях 1-4-7-8 з використанням механізму Сета

Для балансування навантаження алгоритмом MA індекс справедливості досяг 0,81. Хоча балансування навантаження трьох основних потоків дозволяє маршрутизувати кожен потік через один із трьох шляхів, утворених непересічними зв’язками, фоновий потік, маршрутизований через шлях за замовчуванням, споживає пропускну здатність двох шляхів із пересічними зв’язками, шляхів 1-2, -5-8 і 1-3-6-8 (рисунок 4.12). Таким чином, потоки, які передаються через ці два шляхи, конкурують за пропускну здатність із фоновим потоком, маршрутизованим через шлях за замовчуванням. У цьому випадку фоновий потік має середню пропускну здатність 1,2 Мбіт/с, тоді як один із збалансованих потоків має середню пропускну здатність 8,6 Мбіт/с, а інші два мають середню пропускну здатність 8,5 Мбіт/с.

На рисунку 4.15 представлені результати значень втрати пакетів для моделі трафіку UDP-4f. Результати, представлені на цьому рисунку, подібні до отриманих в експерименті з двома потоками, але з більш високими значеннями втрат, оскільки тепер передаються чотири потоки.

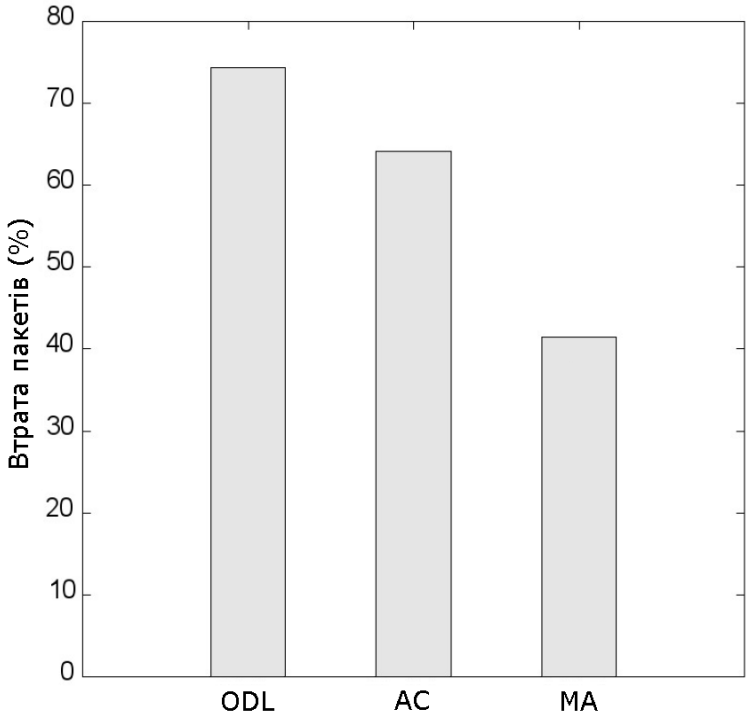

Рисунок 4.15 – Втрата пакетів для алгоритмів ODL, Сета і MA, коли кількість потоків дорівнює 4

Різниця між ODL і алгоритмом Сета зменшилася. ODL досяг 74% втрат пакетів, а алгоритм Сета має втрати близько 65%. Ймовірно, це пов’язано з перевантаженням, викликаним одночасною передачею чотирьох потоків. Виявляється, алгоритм Сета поводиться дуже подібно до ODL. Це тому, що, як уже згадувалося, у алгоритмі Сета збалансований лише один головний потік; решта передаються через стандартний шлях, подібний до ODL.

Коефіцієнт втрати пакетів MA наближається до 42%. Це пояснюється тим, що три основні потоки перемикаються на менш завантажені шляхи. Балансування навантаження та керування комутацією MA дозволяють доставляти більше пакетів до місць призначення за менш завантаженими шляхами.

ВИСНОВКИ

Мета кваліфікаційної роботи полягала в поліпшенні продуктивності комп’ютерних мереж завдяки балансуванню навантаженням на мережевих каналах зв’язку.

В кваліфікаційній роботі запропоновано метод балансування навантаженням між шляхами, який включає функцію управління перемиканням потоків на недовантажені канали.

Запропонований алгоритм обчислює кілька шляхів із непересічними зв’язками, які мають найменшу кількість переходів між джерелом і одержувачем. Крім того, алгоритм має функцію «контролю перемикання», яка перевіряє, чи перевищує поточне завантаження шляху відсоток його пропускної здатності, і чи потенційний новий шлях, має завантаженість принаймні на порогове значення, менше, ніж значення поточного шляху.

Алгоритм перевершує за показниками стандартний режим роботи ODL і алгоритм Сета. Можна зробити висновок, що модифікований алгоритм балансування навантаженням є хорошим рішенням для уникнення перевантажень і таким чином, сприяє збільшенню сукупної пропускної здатності мережі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec. MPTCP is Not Pareto-optimal: Performance Issues and a Possible Solution. IEEE/ACM Transactions on Networking (TON), 21(5):1651–1665, October 2013.

2. N. F. Maxemchuk. DISPERSITY ROUTING IN STORE AND-FORWARD NETWORKS. PhD thesis, University of Pennsylvania, 1975. Available as http://repository. upenn.edu/dissertations/AAI7524101.

3. X. Huang and Y. Fang. Multiconstrained QoS multipath routing in wireless sensor networks. Wireless Networks, 14(4):465–478, 2008.

4. P. Key, L. Massoulie´, and D. Towsley. Combining multipath routing and congestion control for robustness. In Proceedings of the 40th Annual Conference on In- formation Sciences and Systems, pages 345–350. IEEE, 2006.

5. S. Shakkottai, E. Altman, and A. Kumar. The Case for Non-Cooperative Multihoming of Users to Access Points in IEEE 802.11 WLANs. In Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM ’06), 2006.

6. D. Wischik, M. Handley, and M. B. Braun. The resource pooling principle. ACM SIGCOMM Computer Communication Review, 38(5):47–52, 2008.

7. K. Sklower, B. Lloyd, G. McGregor, D. Carr, and T. Coradetti. The PPP Multilink Protocol (MP). IETF RFC 1990, August 1996.

8. W. Simpson. The Point-to-Point Protocol (PPP). IETF RFC 1661, July 1994.

9. H. Adiseshu, G. Parulkar, and G. Varghese. A reliable and scalable striping protocol. In Proceedings of the Applications, technologies, architectures, and protocols for computer communications, volume 26, pages 131–141. ACM, 1996.

10. S. Kandula, K. C.-J. Lin, T. Badirkhanli, and D. Katabi. FatVAP:

Aggregating AP Backhaul Capacity to Maximize Throughput. In Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, volume 8, pages 89–104, 2008.

11. T. Consortium et al. Openflow switch specifica- tion/version 1.1. 0, 2011.

12. T. N. Subedi, K. K. Nguyen, and M. Cheriet. OpenFlow-based in-network Layer-2 adaptive multipath aggregation in data centers. Computer Communications, 61:58–69, 2015.

13. D. Fedyk, P. Ashwood-Smith, D. Allan, A. Bragg, and P. Unbehagen. IS-IS Extensions Supporting IEEE 802.1 aq Shortest Path Bridging. IETF RFC 6329, 2012.

14. D. Eastlake, M. Zhang, A. Ghanwani, V. Manral, and A. Banerjee. Transparent Interconnection of Lots of Links (TRILL): Clarifications, Corrections, and Updates. IETF RFC 7180, May, 2014.

15. J. Sun, Y. Wen, and L. Zheng. On file-based content distribution over wireless networks via multiple paths: Coding and delay trade-off. In Proceedings of the IEEE INFOCOM, pages 381–385, 2011.

16. K.-H. Kim and K. G. Shin. Improving TCP performance over wireless networks with collaborative multi-homed mobile hosts. In Proceedings of the 3rd international conference on Mobile systems, applications, and services, pages 107–120. ACM, 2005.

17. P. Rodriguez, R. Chakravorty, J. Chesterfield, I. Pratt, and S. Banerjee. MAR: A commuter router infras- tructure for the mobile Internet. In Proceedings of the 2nd international conference on Mobile systems, applications, and services, pages 217–230. ACM, 2004.

18. K. Habak, K. A. Harras, and M. Youssef. OSCAR:A Collaborative Bandwidth Aggregation System. arXiv preprint arXiv:1401.1258, 2014.

19. R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host identity protocol. IETF RFC 5201, April 2008.

20. S. Mascolo, L. A. Grieco, R. Ferorelli, P. Camarda, and G. Piscitelli.

Performance evaluation of Westwood+ TCP congestion control. Performance Evaluation, 55(1):93–111, 2004.

21. D. Sarkar. A Concurrent Multipath TCP and Its Markov Model. In Proceedings of the IEEE International Conference on Communications (ICC ’06), pages 615–620, June 2006.

22. L. Magalhaes and R. Kravets. Transport level mecha- nisms for bandwidth aggregation on mobile hosts. In Proceedings of the 9th IEEE International Conference on Network Protocols, pages 165–171, 2001.

23. H.-Y. Hsieh, K.-H. Kim, Y. Zhu, and R. Sivakumar. A Receiver-centric Transport Protocol for Mobile Hosts with Heterogeneous Wireless Interfaces. In Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom ’03), pages 1–15, 2003.

24. A. Ford, C. Raiciu, M. Handley, and S. Barre. TCP Extensions for Multipath Operation with Multiple Ad- dresses. IETF Internet-Draft, May 2009.

25. Seth, N. SDN load balancing. Available at:https: //github.com/nayanseth/sdn-loadbalancing. (2022).

26. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. 2009. Introduction to Algorithms. MIT Press, 3 edition. Book.

27. Jain, R. The art of computer systems performance analysis: techniques for experimental design, measure- ment, simulation, and modeling. John Wiley & Sons, 1991. 1 edition.

28. Lantz, B., Heller, B., and McKeown, N. A network in a laptop: rapid prototyping for software-defined networks. In Proceedings of Workshop on Hot Topics in Networks (SIGCOMM), Hotnets-IX, 2010. pages 1–6.

29. Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothen- berg, C. E., Azodolmolky, S., and Uhlig, S. Software-Defined Networking: a comprehensive sur- vey. Proceedings of the IEEE, 2015. 103(1):14–76.

30. OpenDaylight Opendaylight - The L2 Switch project provides Layer2 switch functionality. 2022. Available at:https://docs.opendaylight.org/en/stable-fluorine/ user-guide/l2switch-user-guide.html.

31. Бояршинов Є.В., Соколовський С.О., Янковський О.А. Аналіз алгоритмів багатошляхової маршрутизації для програмно-визначених мереж//Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління. Чотирнадцята міжнародна науково-технічна конференція. Баку – Харків – Жиліна. 2024 р. Том 1. С. 111.

32. Соколовський С.О., Афанков М.В., Ключка М.І., Янковський О.А. Методи комплексна боротьба з перевантаженням в IP-мережах//Збірник тез доповідей дванадцятої міжнародної науково-технічної конференції «Проблеми інформатизації». Баку, Харків, Бельсько-Бяла. 2024. С. 78