

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ
ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Кафедра системного програмування і
спеціалізованих комп'ютерних систем

КУРСОВИЙ ПРОЄКТ

з дисципліни «Комп'ютерні мережі» на тему:
Маршрутизація в мережі передачі даних

Студента IV курсу, групи КВ-21

за спеціальністю

123 «Комп'ютерна інженерія»

Мануйлова Д.Д.

Керівник: доцент, к.т.н. Мартинова О. П.

Національна оцінка _____

Кількість балів: _____ Оцінка: ECTS _____

Члени комісії _____ доцент, к.т.н. Мартинова О. П.

_____ асистент кафедри СПіСКС, PhD

Сергієнко П.А.

Київ- 2025 рік

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ
СІКОРСЬКОГО»**

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 123 «Комп'ютерна інженерія»

ЗАВДАННЯ

на курсовий проєкт студента

Мануйлова Дениса Денисовича

1. Тема проєкту: Маршрутизація в мережі передачі даних
керівник проєкту: к.т.н., доцент Мартинова Оксана Петрівна,
2. Термін подання студентом проєкту: листопад 2025 р.
3. Вихідні дані до проєкту: Структура мережі – 3 регіональні мережі, кожна з яких має мінімум 9 комунікаційних вузлів. Середній ступінь мережі – 2,5. Ваги каналів - 3, 5, 6, 7, 8, 10, 11, 15, 18, 21. Алгоритм – маршрутизації з урахуванням стану каналів зв'язку.
4. Зміст пояснювальної записки: Вступ. Основні теоретичні відомості. Опис заданого алгоритму маршрутизації. Основна частина. Опис процесу тестування передачі повідомлень. Аналіз та порівняння отриманих результатів. Висновки. Список використаної літератури.

Календарний план

№ з/п	Назва етапів виконання курсового проєкту	Тиждень семестру	Примітка
1.	Отримання теми та завдання	2	Виконано
2.	Підбір та вивчення літератури	3 - 4	Виконано
3.	Виконання завдання 1	5 - 6	Виконано
4.	Виконання завдання 2	7 - 8	Виконано
5.	Виконання завдання 3	9 - 10	Виконано
6.	Виконання завдання 4	11 - 12	Виконано
7.	Виконання завдання 5	13	Виконано
8.	Подання курсового проєкту на перевірку	13	Виконано
9.	Захист курсового проєкту	14-15	Виконано

Студент

(підпис)

Мануйлов Денис
(ім'я, прізвище)

Керівник проєкту

(підпис)

Оксана Мартинова
(ім'я, прізвище)

АНОТАЦІЯ

Курсовий проєкт включає пояснювальну записку (XX с., XX рис., XX табл., список використаних джерел з XX найменувань).

У даному курсовому проєкті розроблено програмне забезпечення для моделювання процесів маршрутизації в комп'ютерних мережах. Основною метою роботи є дослідження алгоритмів пошуку найкоротшого шляху на основі стану каналів зв'язку та аналіз ефективності передачі даних у мережах зі складною топологією.

Створена програма дозволяє моделювати передачу повідомлень у двох режимах: дейтаграмному та режимі віртуального каналу. Реалізовано функціонал для візуальної побудови графа мережі, що складається з трьох регіональних підмереж, налаштування параметрів каналів зв'язку (вага, тип, пропускна здатність) та симуляції процесу передачі пакетів.

Для визначення оптимального маршруту використано алгоритм Дейкстри, який враховує стан каналів (Link State). Програмний продукт дозволяє збирати статистичні дані, будувати графіки залежності часу передачі та службового трафіку від параметрів мережі, а також візуалізувати процес маршрутизації в реальному часі.

Графічний інтерфейс програми реалізовано мовою програмування C++ з використанням бібліотеки Qt.

Ключові слова: комп'ютерна мережа, маршрутизація, алгоритм Дейкстри, Link State, віртуальний канал, дейтаграмний режим, C++, Qt, симуляція.

ANNOTATION

The course project includes an explanatory note (XX p., XX figures, XX tables, a list of XX references).

In this course project, software has been developed to simulate routing processes in computer networks. The main goal of the work is to study shortest path algorithms based on link states and analyze data transmission efficiency in networks with complex topology.

The developed program allows modeling message transmission in two modes: datagram and virtual circuit mode. Functionality is implemented for visual construction of a network graph consisting of three regional subnets, configuration of communication channel parameters (weight, type, bandwidth), and simulation of the packet transmission process.

Dijkstra's algorithm, which considers the link state, is used to determine the optimal route. The software product allows collecting statistical data, plotting graphs of transmission time and service traffic dependence on network parameters, and visualizing the routing process in real-time.

The graphical interface of the program is implemented in the C++ programming language using the Qt framework.

Keywords: computer network, routing, Dijkstra's algorithm, Link State, virtual circuit, datagram mode, C++, Qt, simulation.

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Програмний комплекс моделювання маршрутизації в мережі передачі даних». Галузь застосування: навчальний процес, дослідження ефективності алгоритмів маршрутизації, аналіз мережевого трафіку та проектування топологій комп'ютерних мереж.

2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є навчальний план спеціальності 123 «Комп'ютерна інженерія» та завдання на курсовий проєкт з дисципліни «Комп'ютерні мережі», затверджене кафедрою.

3. МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою роботи є створення програмного забезпечення для моделювання процесів маршрутизації в складеній мережі (Link-State) та дослідження часових характеристик передачі даних. Призначення програми:

- Візуалізація топології мережі, що складається з трьох регіонів.
- Пошук оптимального маршруту за допомогою алгоритму Дейкстри.
- Симуляція передачі даних у режимах віртуального каналу та дейтаграмного режимі.
- Збір та візуалізація статистики (графіки залежностей).

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами розробки є технічна документація бібліотеки Qt, специфікації протоколів маршрутизації (OSPF), науково-технічна література з алгоритмів на графах.

5. ТЕХНІЧНІ ВИМОГИ 5.1. Вимоги до програмного продукту:

- Мова програмування: C++.
- Середовище розробки та бібліотеки: Qt Framework (Qt Widgets, Qt Charts).
- Реалізація графічного інтерфейсу користувача (GUI).
- Можливість генерації мережі згідно з варіантом (3 регіони, мінімум 27 вузлів).
- Підтримка редагування топології (додавання/видалення вузлів та зв'язків).
- Реалізація алгоритму Дейкстри для розрахунку таблиць маршрутизації.
- Анімація переміщення пакетів між вузлами.
- Можливість налаштування параметрів: розмір повідомлення, MTU, ймовірність помилок.

Зм	Лист	№ докум.	Підп.	Дата

ІАЛЦ.467200.002 ТЗ

Лист

1

5.2. Вимоги до апаратного забезпечення:

- Процесор: Intel Core i3 або аналог (тактова частота не менше 2.0 ГГц).
- Оперативна пам'ять: не менше 4 ГБ.
- Вільне місце на диску: не менше 100 МБ.
- Дисплей з роздільною здатністю не менше 1280x720.

ЕТАПИ РОЗРОБКИ

1. Аналіз предметної області та вибір засобів розробки.
2. Проєктування архітектури програми (класи Node, Edge, Network).
3. Реалізація алгоритму маршрутизації (Dijkstra).
4. Розробка графічного інтерфейсу та системи візуалізації.
5. Реалізація логіки симуляції передачі даних та збору статистики.
6. Тестування та налагодження програми.
7. Оформлення пояснювальної записки.

Зміст

ВСТУП.....	9
1. ТЕОРЕТИЧНА ЧАСТИНА	10
1.1. Постановка задачі та аналіз предметної області	10
1.2. Огляд алгоритмів маршрутизації.....	10
1.3. Маршрутизація з урахуванням стану каналу (Link-State).....	10
1.4. Алгоритм Дейкстри.....	11
2. ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ	13
2.1. Призначення та структура програми.....	13
2.2. Опис основних класів та алгоритмів	15
2.3. Інтерфейс користувача.....	16
2.4. Функціональні можливості та інтерфейс користувача	18
2.4.1. Ручне створення та редагування топології.....	18
2.4.2. Налаштування параметрів мережі.....	19
2.4.3. Аналіз таблиць маршрутизації	20
2.4.4. Візуалізація симуляції передачі даних.....	21
3. ДОСЛІДНИЦЬКА ЧАСТИНА	23
3.1. Постановка задачі дослідження та опис топології мережі	23
3.2. Аналітичне дослідження характеристик передачі.....	24
3.3. Практичне тестування передачі (Симуляція)	26
3.4. Визначення оптимального розміру пакету	29
ВИСНОВКИ.....	30
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	31
ДОДАТОК	32

					ІАЛЦ. 467200.003 ПЗ		
		№ докум.	Підпис		<div>Маршрутизація в мережі передачі даних.</div> <div>Пояснювальна записка</div>		
Розроб.		Мануйлов Д.Д.					
Перев.		Мартінова О. П.					
Н. контроль							
Затвердив							
					Лім.	Лист	Листів
						3	27
					НТУУ «КПІ ім. Ігоря Сікорського», ФПМ, КВ-21		

ВСТУП

Комп'ютерні мережі є основою сучасного цифрового простору, а ефективність їхньої роботи залежить від алгоритмів маршрутизації. Актуальність роботи полягає у дослідженні протоколів із урахуванням стану каналів (Link State), які забезпечують високу надійність та швидку збіжність мережі.

Метою курсового проєкту є розробка програмного забезпечення для моделювання маршрутизації на базі C++/Qt та аналізу ефективності передачі даних у режимах дейтаграми та віртуального каналу.

Для досягнення мети поставлено такі завдання:

1. Проаналізувати принципи роботи алгоритмів Link State (Dijkstra) для розрахунку шляху.
2. Розробити програмну модель мережі довільної топології, включаючи три регіональні підмережі.
3. Створити графічний інтерфейс користувача (GUI) та реалізувати анімацію передачі пакетів.
4. Провести серію експериментів для порівняння службового трафіку та часу доставки пакетів залежно від MTU та режиму комутації.

Об'єктом дослідження є процеси маршрутизації пакетів у мережах. Предметом — алгоритм Дейкстри та ефективність передачі даних у різних режимах комутації.

Практичне значення роботи полягає у створенні наочного інструменту для симуляції та аналізу мережевих протоколів, що має навчальну цінність.

1. ТЕОРЕТИЧНА ЧАСТИНА

1.1. Постановка задачі та аналіз предметної області

Завданням курсового проєкту є моделювання мережі передачі даних, яка складається з трьох регіональних мереж. Кожна регіональна мережа містить мінімум 9 комунікаційних вузлів. Середній ступінь вузла в мережі складає 2,5, що вказує на помірну зв'язність топології.

Ключовою вимогою є реалізація маршрутизації з урахуванням стану каналів зв'язку. Це вимагає від кожного вузла знання повної топології мережі та ваг усіх каналів для прийняття рішень про маршрутизацію.

1.2. Огляд алгоритмів маршрутизації

Маршрутизація — це процес вибору шляху в мережі вздовж якого пакети даних передаються від джерела до призначення. Алгоритми маршрутизації можна класифікувати на:

1. Статичні (неадаптивні): маршрути задаються вручну адміністратором і не змінюються автоматично при зміні топології.
2. Динамічні (адаптивні): маршрути автоматично перераховуються при зміні навантаження або топології мережі.

Динамічні протоколи поділяються на два основні класи:

- * Дистанційно-векторні (Distance Vector), наприклад RIP. Кожен вузол знає лише відстань до сусідів.
- * Протоколи з урахуванням стану каналу (Link-State), наприклад OSPF, IS-IS. Кожен вузол будує повну карту мережі.

Для виконання даного проєкту обрано другий клас алгоритмів (Link-State), оскільки він є більш ефективним для складних ієрархічних мереж, що складаються з кількох регіонів.

1.3. Маршрутизація з урахуванням стану каналу (Link-State)

Концепція маршрутизації за станом каналу базується на ідеї, що кожен маршрутизатор повинен мати точну інформацію про граф мережі. Процес маршрутизації складається з наступних кроків:

1. Виявлення сусідів: Маршрутизатор надсилає Hello-пакети, щоб дізнатися, хто є його сусідами.

2. Вимірювання вартості лінії: Визначається метрика (вага) каналу до кожного сусіда (затримка, пропускна здатність тощо). У даному проєкті ваги обираються з фіксованого набору значень.
3. Створення пакету стану каналу (LSA - Link State Advertisement): Формується пакет, що містить інформацію про всіх сусідів та ваги зв'язків.
4. Розсилка пакетів: LSA розсилаються всім маршрутизаторам мережі методом «заливки» (flooding).
5. Обчислення нового маршруту: Коли маршрутизатор має повну базу даних графа мережі, він використовує алгоритм Дейкстри для побудови дерева найкоротших шляхів.

1.4. Алгоритм Дейкстри

Алгоритм Дейкстри (Dijkstra's algorithm) є основним математичним апаратом для протоколів Link-State (наприклад, OSPF). Він дозволяє знайти найкоротші шляхи від однієї вершини графа до всіх інших.

Принцип роботи алгоритму:

1. Кожній вершині приписується мітка - відстань від джерела до цієї вершини. Початкова мітка для джерела - 0, для інших - нескінченність.
2. Всі вершини діляться на два набори: відвідані та невідвідані.
3. На кожному кроці вибирається невідвідана вершина з найменшою міткою (поточна вершина).
4. Розглядаються всі сусіди поточної вершини. Для кожного сусіда розраховується нова відстань: (мітка поточної вершини) + (вага ребра).
5. Якщо розрахована відстань менша за поточну мітку сусіда, мітка оновлюється.
6. Поточна вершина позначається як відвідана.
7. Кроки повторюються, доки всі вершини не будуть відвідані або доки не буде знайдено шлях до цільового вузла.

Математично задача зводиться до пошуку шляху з мінімальною сумарною вагою ребер у зваженому графі:

Формула ініціалізації(початковий стан графа):

$$\begin{cases} D(s) = 0, & \text{для початкового вузла } s \\ D(v) = \infty, & \text{для всіх інших вузлів } v \neq s \end{cases}$$

Формула релаксації (як алгоритм оновлює маршрут, якщо знаходить коротший шлях через сусідню вершину):

$$D(v) = \min (D(v), D(u) + w(u, v))$$

$D(v)$ – поточна найкоротша відстань від почтакового вузла до вузла v .

$D(u)$ – найкоротша відстань до сусіднього вузла u .

$w(u, v)$ – вага ребра, що з'єднує вузли u та v .

Саме цей алгоритм буде програмно реалізований у наступному розділі для знаходження оптимального маршруту передачі повідомлень.

2. ОПИС РОЗРОБЛЕНОГО ПРОГРАМНОГО ПРОДУКТУ

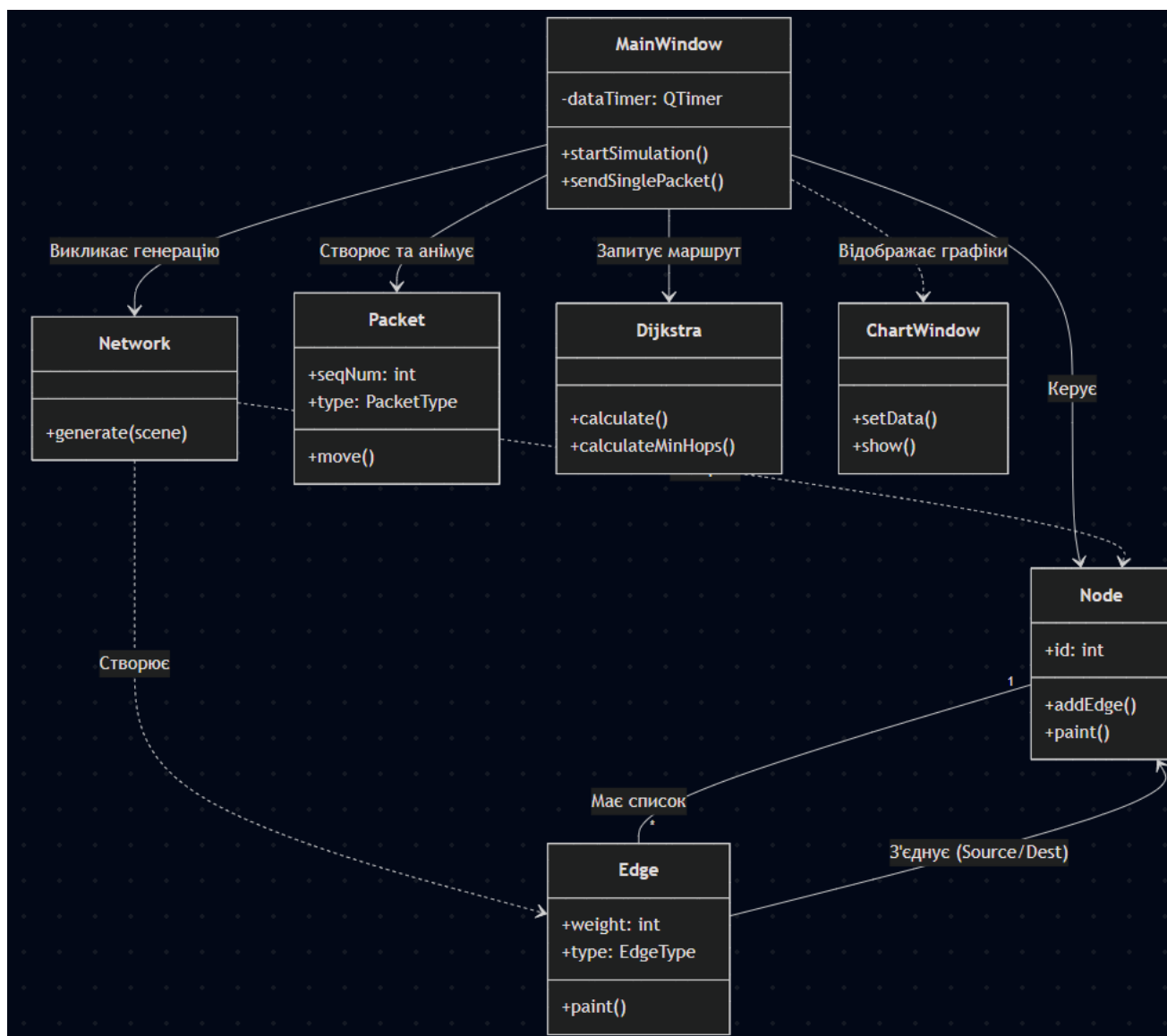
2.1. Призначення та структура програми

Розроблений програмний продукт призначений для симуляції роботи комп'ютерної мережі з використанням алгоритмів маршрутизації стану каналу (Link-State). Програма дозволяє користувачеві наочно дослідити принципи маршрутизації, вплив розміру пакетів (MTU) та ймовірності помилок на ефективність передачі даних.

Проект реалізовано мовою C++ з використанням фреймворку Qt. Архітектура програми побудована за модульним принципом і складається з наступних основних компонентів (файлів):

- `main.cpp` — точка входу в програму.
- `mainwindow.h/cpp` — головний клас інтерфейсу, що керує логікою симуляції, таймерами та взаємодією з користувачем.
- `network.h/cpp` — модуль генерації топології мережі відповідно до варіанту (3 регіони).
- `node.h/cpp` — клас, що описує маршрутизатор (вузол мережі) та його візуальне представлення.
- `edge.h/cpp` — клас, що описує канал зв'язку (ребро графа), його вагу та тип (дуплекс/напівдуплекс).
- `packet.h/cpp` — клас для візуалізації пакетів даних та анімації їх переміщення.
- `dijkstra.h/cpp` — реалізація алгоритму Дейкстри для знаходження найкоротшого шляху.
- `chartwindow.h/cpp` — модуль для побудови графіків та діаграм.

Схема взаємодії модулів представлена на рис. 2.1.



2.1. Схема взаємодії модулів.

2.2. Опис основних класів та алгоритмів

Клас Node (Вузол) Клас успадковується від QGraphicsItem і відповідає за відображення маршрутизатора на сцені. Він зберігає свій унікальний ідентифікатор (id) та список підключених каналів (edgeList). Реалізовано обробку події подвійного кліку миші (mouseDoubleClickEvent), яка викликає розрахунок та відображення таблиці маршрутизації для даного вузла.

Клас Edge (Канал) Клас відповідає за зв'язок між двома вузлами. Він має властивості: вага (weight) та тип з'єднання (EdgeType: Duplex або HalfDuplex). Візуально вага відображається числом над лінією зв'язку. Передбачена можливість зміни ваги каналу користувачем.

Клас Dijkstra (Алгоритм маршрутизації) Цей клас містить статичні методи для реалізації алгоритму пошуку шляху. Метод calculate(Node* startNode, ...) реалізує класичний алгоритм Дейкстри. Він використовує пріоритетну чергу (або ітеративний пошук мінімуму) для знаходження найкоротшого шляху на основі ваг ребер (метрика «Вартість»). Метод calculateMinHops(...) реалізує пошук в ширину (BFS) для знаходження шляху з найменшою кількістю переходів (метрика «Хопи»). Результат роботи алгоритму повертається у вигляді вектору структур RoutingEntry, що містять повний шлях та вартість доставки до кожного вузла.

Клас Network (Генерація мережі) Метод generate автоматично створює топологію мережі згідно з завданням. Алгоритм генерує 3 регіональні кластери, розміщуючи вузли за координатами, та з'єднує їх між собою магістральними каналами, утворюючи єдину мережеву інфраструктуру. Ваги каналів обираються випадковим чином із заданого набору значень.

Клас MainWindow (Логіка симуляції) Клас керує процесом симуляції. Основні методи:

- startSimulation(): ініціалізує параметри (розмір повідомлення, MTU), викликає алгоритм Дейкстри для пошуку маршруту та запускає процес передачі.
- sendSinglePacket(...): створює об'єкт пакету та анімацію його руху вздовж знайденого шляху.
- createPacketAnim(...): використовує QPropertyAnimation для плавної візуалізації руху пакетів. Також тут реалізована логіка імітації втрати пакетів залежно від заданого відсотка помилок.

2.3. Інтерфейс користувача

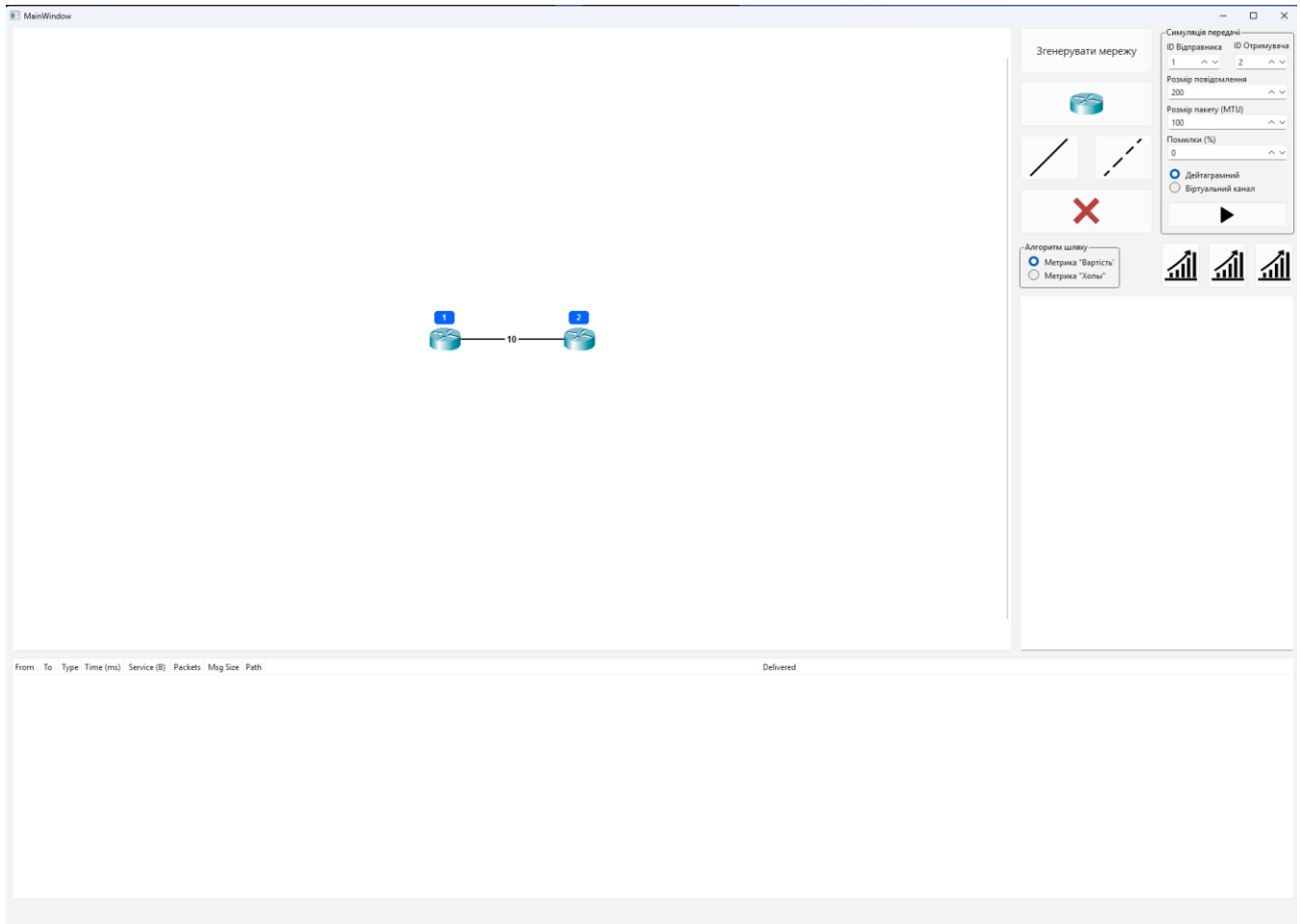


Рис. 2.1 – Головне вікно програми

Робоча область відображає графічну модель мережі. Вузли представлені іконками маршрутизаторів з ідентифікаторами, канали — лініями з позначенням ваги.

Панель керування містить наступні елементи:

1. Генерація: Кнопка «Згенерувати мережу» для швидкого створення топології (Заданної за варіантом).
2. Редагування: Інструменти для додавання вузлів, дуплексних та напівдуплексних каналів, видалення елементів.
3. Параметри симуляції:
 - Вибір ID відправника та отримувача.
 - Встановлення розміру повідомлення (в байтах) та розміру пакету (MTU).
 - Встановлення ймовірності помилок у каналі (%).
 - Перемикач режиму: «Дейтаграмний» або «Віртуальний канал».

Зм	Лист	№ докум.	Підп.	Дата

ІАЛЦ.467200.002 ТЗ

Лист
11

4. Алгоритм: Вибір метрики маршрутизації («Вартість» або «Хопи»).
5. Аналітика: Кнопки для виклику вікон з графіками залежностей(При наведенні описується яка саме залежність).

У нижній частині вікна розташована таблиця логування, куди в реальному часі записуються результати кожної симуляції: час передачі, обсяг службового трафіку, кількість пакетів та статус доставки.

2.4. Функціональні можливості та інтерфейс користувача

Головне вікно програми надає користувачеві повний набір інструментів для створення, редагування та аналізу комп'ютерної мережі. Нижче наведено детальний опис режимів роботи програми.

2.4.1. Ручне створення та редагування топології

Програма дозволяє не лише генерувати мережу автоматично, але й створювати власну топологію вручну.

- Створення вузлів: За допомогою кнопки «Додати вузол» на робочу область додаються нові маршрутизатори з автоматичною індексацією.
- Створення каналів: Користувач може з'єднати два обрані вузли каналом зв'язку. Реалізовано два типи каналів:
 - *Дуплексний (Full-Duplex)*: відображається суцільною лінією.
 - *Напівдуплексний (Half-Duplex)*: відображається пунктирною лінією.

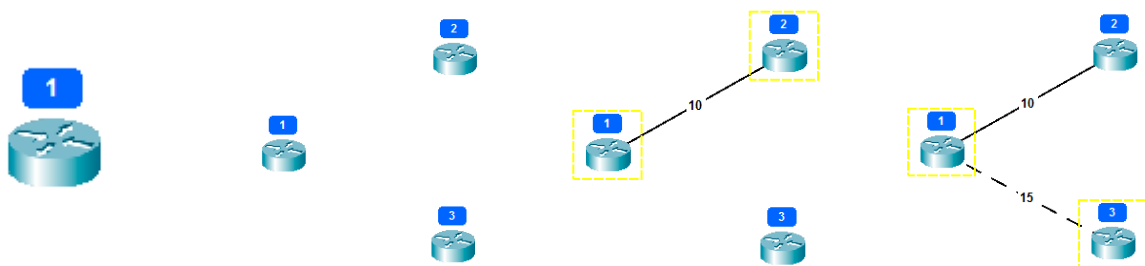


Рис. 2.2 – Ручне створення вузлів та каналів різних типів

2.4.2. Налаштування параметрів мережі

Кожен канал має вагу (вартість передачі). Програма дозволяє змінювати цей параметр вручну. При подвійному кліку на лінію зв'язку відкривається діалогове вікно для введення нової ваги. Це дозволяє моделювати ситуації зміни пропускну здатності каналів.

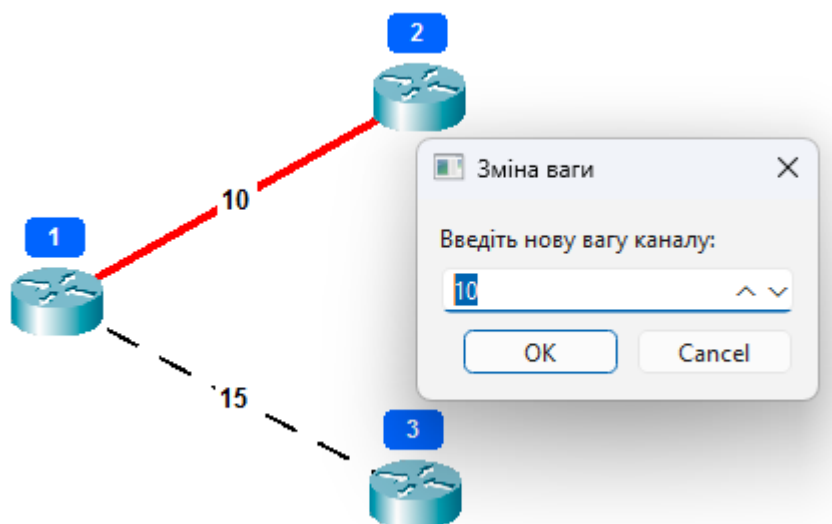


Рис. 2.3 – Ручне налаштування ваги каналу

Також реалізовано функцію видалення. При виділенні групи елементів (вузлів або каналів) та натисканні кнопки видалення, вони коректно прибираються з топології та пам'яті програми.

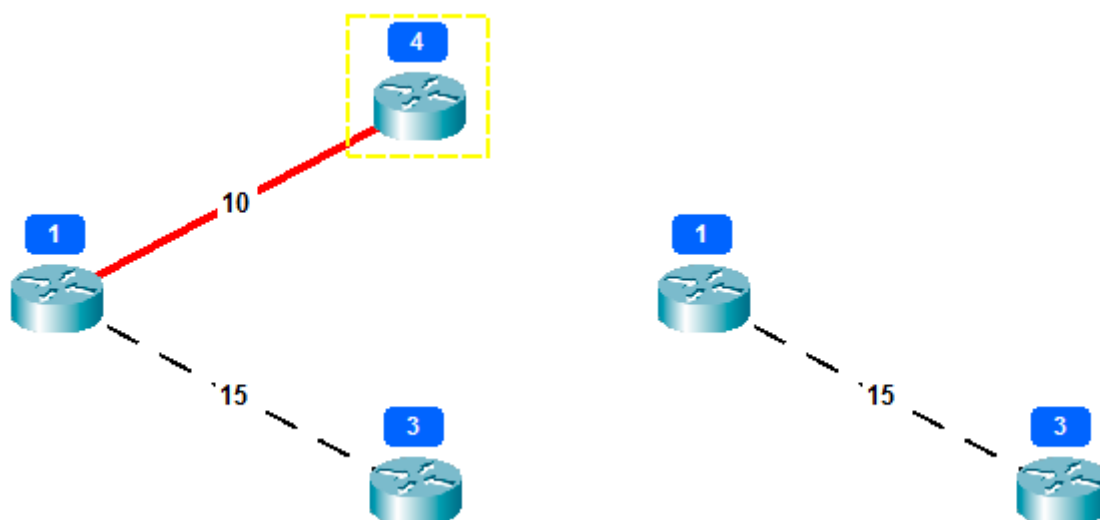


Рис. 2.4 – Виділення групи елементів для видалення

2.4.3. Аналіз таблиць маршрутизації

Однією з ключових особливостей програми є візуалізація роботи алгоритму Дейкстри. Користувач може обрати критерій побудови маршруту:

- Метрика «Вартість»: враховує ваги каналів.
- Метрика «Хопи»: враховує лише кількість транзитних ділянок.

При подвійному кліку на будь-який вузол відкривається його таблиця маршрутизації, розрахована згідно з обраною метрикою.

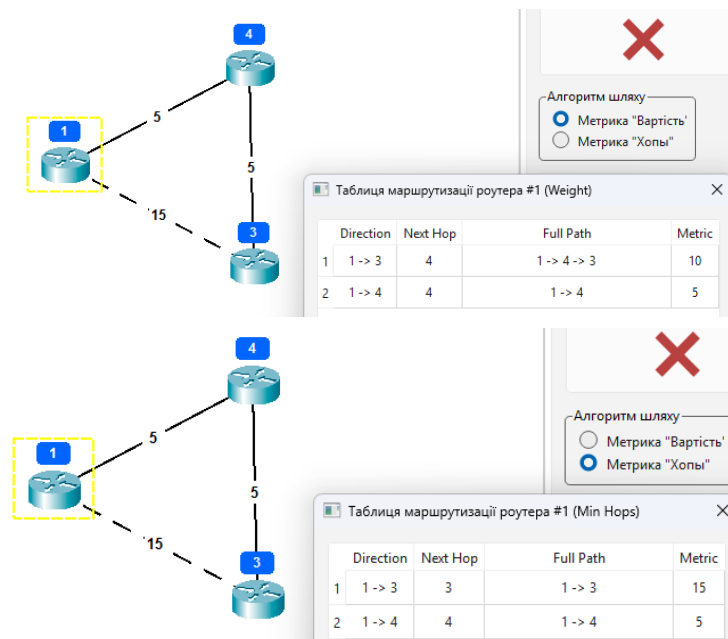


Рис. 2.5 – Порівняння таблиць маршрутизації для різних метрик (зверху - за вартістю, знизу - за кількістю хопів)

2.4.4. Візуалізація симуляції передачі даних

Процес передачі даних відображається анімованими пакетами, що рухаються вздовж розрахованого маршруту.

- Дейтаграмний режим: Пакети (жовтого кольору) передаються потоком без попереднього встановлення з'єднання.
- Режим віртуального каналу: Симуляція проходить у три етапи, що візуалізується різними типами пакетів:
 1. Запит на з'єднання (синій пакет).
 2. Підтвердження з'єднання (зелений пакет).
 3. Передача даних (жовті пакети).
 4. Розрив з'єднання (червоний пакет).

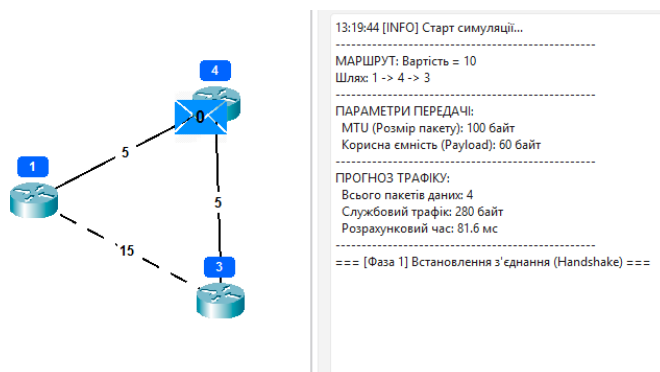


Рис. 2.6.1 – Запит на з'єднання.

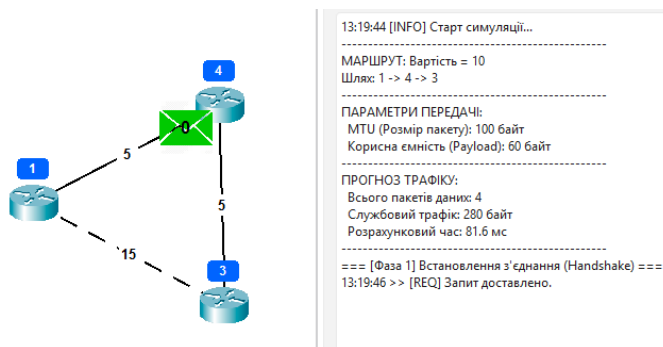
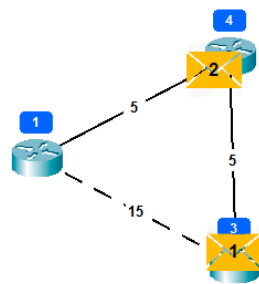


Рис. 2.6.2 – Підтвердження з'єднання.



```

13:19:44 [INFO] Старт симуляції...

-----
МАРШРУТ: Вартість = 10
Шлях: 1 -> 4 -> 3
-----

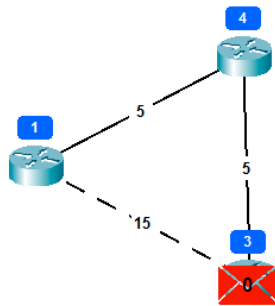
ПАРАМЕТРИ ПЕРЕДАЧІ:
MTU (Розмір пакету): 100 байт
Корисна ємність (Payload): 60 байт
-----

ПРОГНОЗ ТРАФІКУ:
Всього пакетів даних: 4
Службовий трафік: 280 байт
Розрахунковий час: 81.6 мс
-----

=== [Фаза 1] Встановлення з'єднання (Handshake) ===
13:19:46 >> [REQ] Запит доставлено.
13:19:48 >> [ACK] З'єднання встановлено!
=== [Фаза 2] Передача даних (Потік) ===

```

Рис. 2.6.3 – Передача даних.



```

13:19:44 [INFO] Старт симуляції...

-----
МАРШРУТ: Вартість = 10
Шлях: 1 -> 4 -> 3
-----

ПАРАМЕТРИ ПЕРЕДАЧІ:
MTU (Розмір пакету): 100 байт
Корисна ємність (Payload): 60 байт
-----

ПРОГНОЗ ТРАФІКУ:
Всього пакетів даних: 4
Службовий трафік: 280 байт
Розрахунковий час: 81.6 мс
-----

=== [Фаза 1] Встановлення з'єднання (Handshake) ===
13:19:46 >> [REQ] Запит доставлено.
13:19:48 >> [ACK] З'єднання встановлено!
=== [Фаза 2] Передача даних (Потік) ===
13:19:51 >> [DATA] Пакет #1 доставлено.
<< [ACK] Підтвердження для пакету #1 відправлено.
13:19:52 >> [DATA] Пакет #2 доставлено.
<< [ACK] Підтвердження для пакету #2 відправлено.
13:19:53 >> [DATA] Пакет #3 доставлено.
<< [ACK] Підтвердження для пакету #3 відправлено.
13:19:54 >> [DATA] Пакет #4 доставлено.
<< [ACK] Підтвердження для пакету #4 відправлено.
=== [Фаза 3] Розрив з'єднання ===

```

Рис. 2.6.4 – Розрив з'єднання.

У нижній частині вікна ведеться детальний лог подій, що дозволяє відстежити кожен етап передачі та переконатися у відсутності або наявності помилок.

3. ДОСЛІДНИЦЬКА ЧАСТИНА

3.1. Постановка задачі дослідження та опис топології мережі

Метою дослідницької частини є аналіз ефективності алгоритмів маршрутизації та режимів передачі даних. Відповідно до варіанту, за допомогою розробленого ПЗ було згенеровано топологію мережі (3 регіони, 27 вузлів).

На рис. 3.1 наведено візуалізацію згенерованої топології мережі у головному вікні програми.

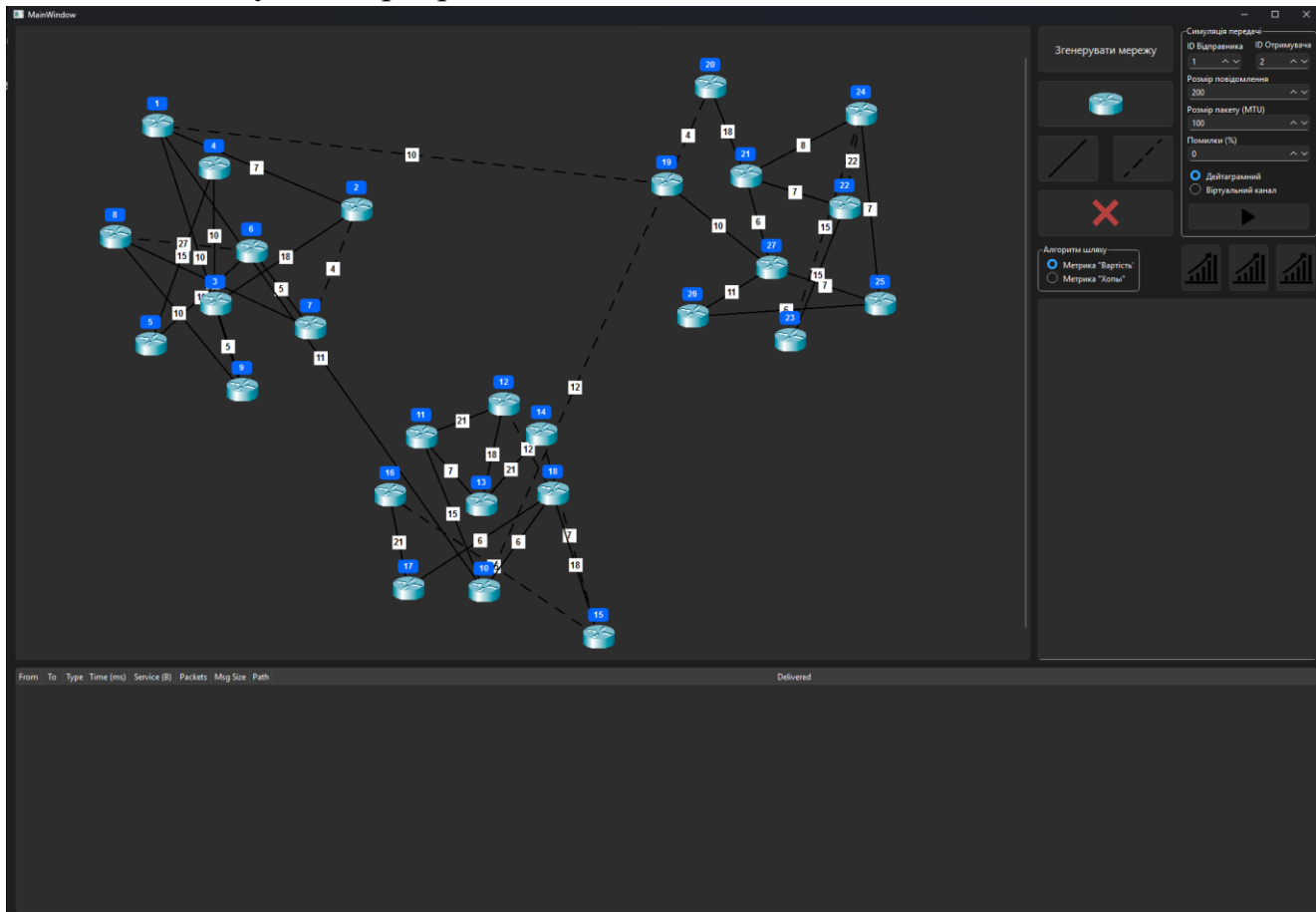


Рис. 3.1 – Топологія досліджуваної мережі (3 регіони)

3.2. Аналітичне дослідження характеристик передачі

У програмі реалізовано модуль аналітичного розрахунку, який дозволяє спрогнозувати навантаження на мережу залежно від вхідних параметрів без необхідності тривалої симуляції кожного окремого випадку.

Аналіз залежності кількості пакетів від MTU. Було побудовано графік теоретичної залежності кількості пакетів від розміру MTU для повідомлення фіксованого розміру (2000 байт).

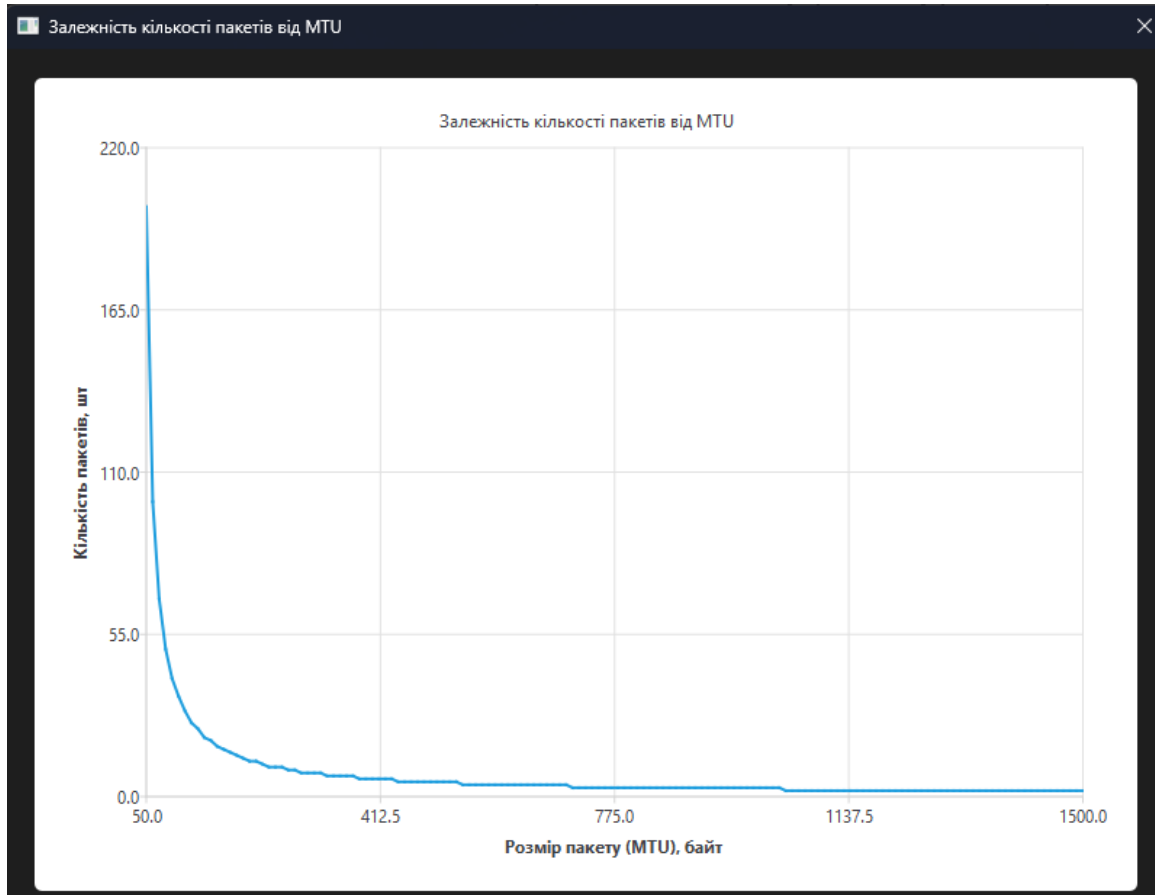


Рис. 3.2 – Теоретична залежність кількості пакетів від розміру MTU

Графік (рис. 3.2) показує гіперболічну залежність. При малих значеннях MTU (50-100 байт) відбувається фрагментація повідомлення на велику кількість дрібних пакетів (понад 20 шт.), що значно навантажує маршрутизатори операціями обробки заголовків.

Аналіз службового трафіку На рис. 3.3 наведено розрахунок обсягу службової інформації (заголовків).

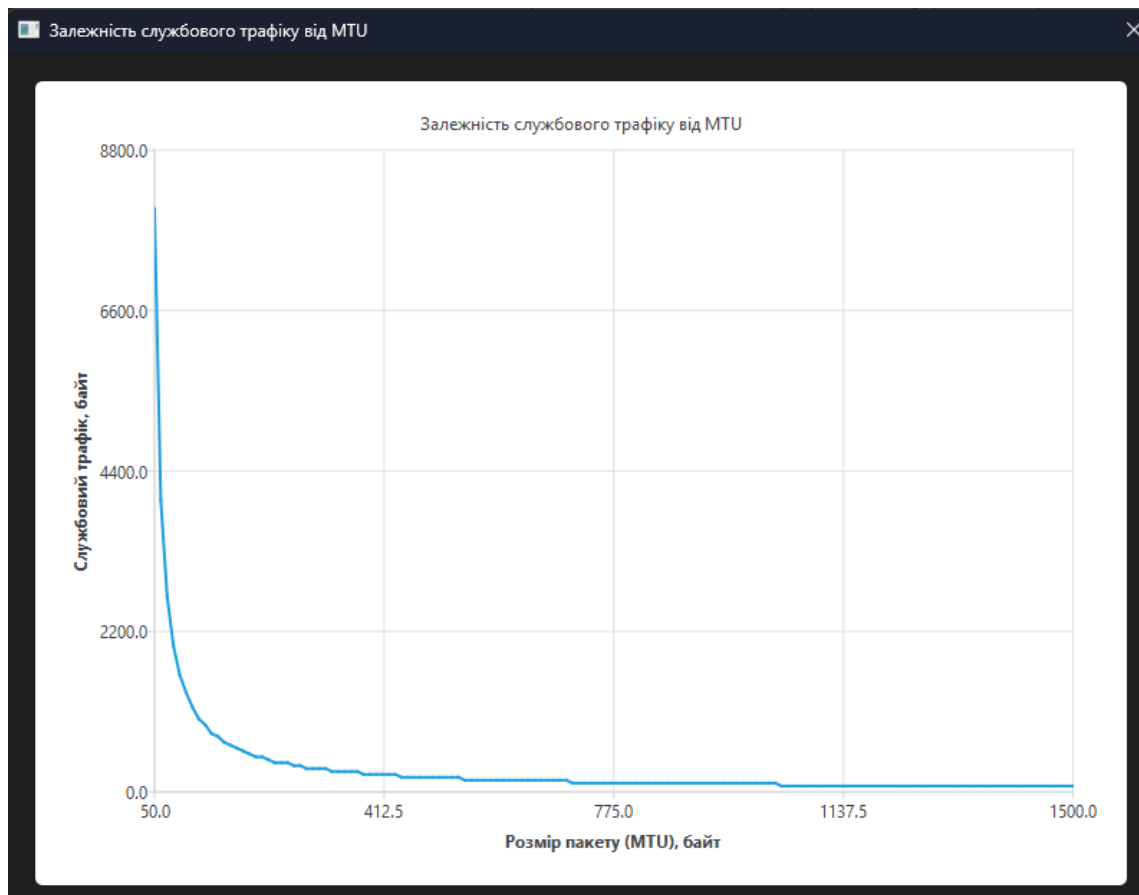


Рис. 3.3 – Залежність обсягу службового трафіку від MTU

Як видно з графіка, використання малих пакетів є неефективним: обсяг службових даних різко зростає. Наприклад, при малому MTU службовий трафік може складати до 50% від корисного навантаження. Зі збільшенням MTU накладні витрати стають мінімальними.

3.3. Практичне тестування передачі (Симуляція)

Для підтвердження теоретичних розрахунків було проведено серію запусків симуляції передачі повідомлення розміром 2000 байт із різними параметрами MTU. Результати занесено у таблицю логування програми (рис. 3.4).

	From	To	Type	Time (ms)	Service (B)	Packets	Msg Size	Path	
1	1	25	Datagram	366.00	1360	34	2000	1->19->27->25	Yes
2	1	25	Datagram	238.00	80	2	2000	1->19->27->25	Yes

Рис. 3.4 – Результати симуляції передачі даних

Аналіз результатів симуляції:

1. Експеримент 1 (Рядок 1): При малому розмірі корисної частини пакету (фрагментація на 34 пакети) загальний час передачі склав 366 мс. Це зумовлено великим обсягом службового трафіку (1360 байт заголовків), який довелося передати каналом зв'язку.
2. Експеримент 2 (Рядок 2): При збільшенні MTU (фрагментація всього на 2 пакети) загальний час зменшився до 238 мс. Обсяг службового трафіку впав до 80 байт.
3. Висновок: Різниця у часі (128 мс) витрачена виключно на передачу зайвих заголовків у першому випадку. Це підтверджує, що для швидкої передачі даних вигідніше використовувати максимально можливий MTU, щоб зменшити накладні витрати на заголовки.

Візуальне порівняння режимів комутації Під час експериментів було зафіксовано різницю у процедурі передачі даних для Дейтаграмного режиму та Віртуального каналу.

На рис. 3.5 наведено фрагмент логу програми для Дейтаграмного режиму. Видно, що передача починається миттєво з пакетів типу DATA.

```

13:31:12 [INFO] Старт симуляції...
-----
МАРШРУТ: Вартість = 75
Шлях: 20 -> 19 -> 1 -> 9 -> 8 -> 7 -> 6
-----
ПАРАМЕТРИ ПЕРЕДАЧІ:
MTU (Розмір пакету): 100 байт
Корисна ємність (Payload): 60 байт
-----
ПРОГНОЗ ТРАФІКУ:
Всього пакетів даних: 4
Службовий трафік: 160 байт
Розрахунковий час: 96 мс
-----
=== [Фаза 1] Дейтаграмний режим ===
13:31:18 >> [DATA] Пакет #1 доставлено.
13:31:19 >> [DATA] Пакет #2 доставлено.
13:31:20 >> [DATA] Пакет #3 доставлено.
13:31:21 >> [DATA] Пакет #4 доставлено.
-----
[FINISH] Передачу завершено.

```

Рис. 3.5 – Лог передачі у дейтаграмному режимі

На рис. 3.6 наведено фрагмент логу для режиму **Віртуального каналу**. Чітко простежуються фази встановлення з'єднання (пакети CONN_REQ, CONN_ACK) перед початком передачі корисних даних, а також фаза завершення (DISCONNECT). Це підтверджує теоретичні відомості про накладні витрати часу на обслуговування сесії у цьому режимі.

```

13:33:59 [INFO] Старт симуляції...
-----
МАРШРУТ: Вартість = 75
Шлях: 20 -> 19 -> 1 -> 9 -> 8 -> 7 -> 6
-----
ПАРАМЕТРИ ПЕРЕДАЧІ:
MTU (Розмір пакету): 100 байт
Корисна ємність (Payload): 60 байт
-----
ПРОГНОЗ ТРАФІКУ:
Всього пакетів даних: 4
Службовий трафік: 280 байт
Розрахунковий час: 129.6 мс
-----
=== [Фаза 1] Встановлення з'єднання (Handshake) ===
13:34:05 >> [REQ] Запит доставлено.
13:34:11 >> [ACK] З'єднання встановлено!
=== [Фаза 2] Передача даних (Потік) ===
13:34:18 >> [DATA] Пакет #1 доставлено.
<< [ACK] Підтвердження для пакету #1 відправлено.
13:34:19 >> [DATA] Пакет #2 доставлено.
<< [ACK] Підтвердження для пакету #2 відправлено.
13:34:20 >> [DATA] Пакет #3 доставлено.
<< [ACK] Підтвердження для пакету #3 відправлено.
13:34:21 >> [DATA] Пакет #4 доставлено.
<< [ACK] Підтвердження для пакету #4 відправлено.
=== [Фаза 3] Розрив з'єднання ===
13:34:27 >> [FIN] З'єднання розірвано.
-----
[FINISH] Симуляцію завершено успішно.

```

Рис. 3.6 – Лог передачі у режимі віртуального каналу

Візуалізація втрати пакетів та ретрансмісії Окрему увагу було приділено перевірці надійності протоколу в умовах завад. У програмі реалізовано механізм імовірнісного знищення пакетів під час анімації.

На рис. 3.7 зображено роботу симуляції при встановленій ймовірності помилок 20%. У лозі подій зафіксовано подію [LOSS], яка означає, що пакет не дійшов до отримувача.

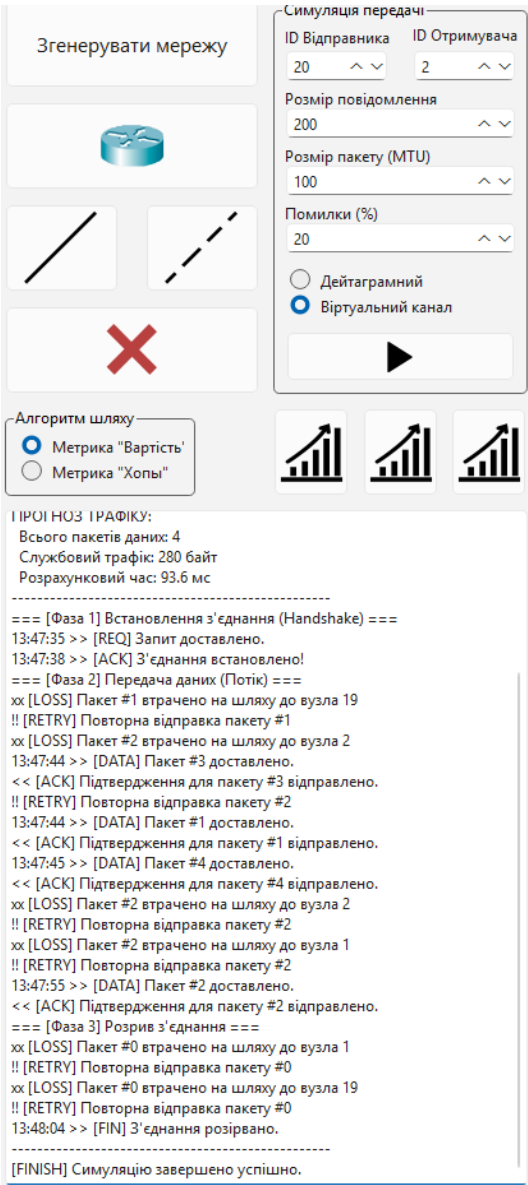


Рис. 3.7 – Фіксація втрати пакету та автоматична повторна передача

3.4. Визначення оптимального розміру пакету

На основі аналітичних графіків та результатів практичної симуляції можна зробити висновок, що оптимальне значення MTU для даної мережі знаходиться в діапазоні 500–1 500 байт. Використання пакетів менше 200 байт призводить до суттєвого сповільнення передачі (на 30-50%) через надлишковий службовий трафік, що чітко видно у таблиці результатів (рис. 3.4).

ВИСНОВКИ

У ході виконання курсового проєкту було розроблено програмний комплекс для моделювання маршрутизації в комп'ютерних мережах та досліджено ефективність передачі даних у мережі складної конфігурації.

У результаті роботи отримано наступні результати:

1. Розроблено програмне забезпечення: Створено додаток мовою C++ з використанням бібліотеки Qt, який дозволяє візуалізувати топологію мережі, що складається з трьох регіональних сегментів (мінімум 27 вузлів). Реалізовано зручний графічний інтерфейс для редагування мережі та спостереження за процесом передачі даних.
2. Реалізовано алгоритми: Успішно імплементовано алгоритм Дейкстри для знаходження найкоротшого шляху на графі (Link-State Routing). Програма дозволяє обирати метрику маршрутизації (вартість каналу або кількість хопів), що наочно демонструє зміну маршрутів залежно від критеріїв оптимізації.
3. Порівняно режими передачі: Встановлено, що дейтаграмний режим є більш ефективним для передачі коротких повідомлень через відсутність накладних витрат на встановлення з'єднання. Режим віртуального каналу, хоч і має початкову затримку на фазу «рукоштовування» (handshake), забезпечує надійність та фіксований маршрут, що є критичним для потокової передачі даних.
4. Досліджено вплив MTU: Аналіз показав, що використання малих пакетів (менше 200 байт) призводить до значного зростання службового трафіку (до 50% від загального обсягу) та збільшення часу обробки. Оптимальним діапазоном MTU для досліджуваної мережі визначено 500–1500 байт.
5. Проаналізовано вплив помилок: Експериментально підтверджено, що при збільшенні ймовірності помилок у каналах зв'язку навантаження на мережу зростає нелінійно через механізми повторної передачі (ARQ).

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Кулаков Ю. О., Луцький Г. М. Комп'ютерні мережі: Підручник. — К.: Юніор, 2003. — 400 с.
2. Таненбаум Е., Везеролл Д. Комп'ютерні мережі. — 5-те вид. — К.: Видавнича група BHV, 2012. — 960 с.
3. Буров Є. В. Комп'ютерні мережі: Підручник. — Львів: Магнолія 2006, 2010. — 262 с.
4. Qt Documentation [Електронний ресурс]. — Режим доступу: <https://doc.qt.io/> (дата звернення: 30.11.2025).
5. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms. — 3rd ed. — MIT Press, 2009. — 1312 p.

ДОДАТОК

Повний вихідний код доступні в репозиторії GitHub за посиланням:

<https://github.com/LoPHarp/NetworkRoutingSim.git>

					ІАЛЦ.467200.002 ТЗ	Лист 27
Зм	Лист	№ докум.	Підп.	Дата		