Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Системотехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Другий (магістерський)
(рівень вищої освіти)

«Дослідження методів підтримки прийняття рішень з реінжинірингу корпоративних комп'ютерних мереж»
(тема)

Виконав: студент II курсу, групи СПРм-20-1
Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми Освітньо-професійна
Освітня програма Системне проектування
(повна назва освітньої програми)
Лясковка Я. І.
(прізвище, ініціали)
Керівник проф. Безкоровайний В.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри системотехніки _____ Гребеннік І.В.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет *Автоматики та комп'ютеризованих технологій*

Кафедра *Системотехніки*

Рівень вищої освіти *Другий (магістерський)*

Спеціальність *122 Комп'ютерні науки*
(код і повна назва)

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

Освітня програма *Системне проектування*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Лясковці Яну Ігоровичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *«Дослідження методів підтримки прийняття рішень з реінжинірингу корпоративних комп'ютерних мереж»*
затверджена наказом університету від *«08» листопада 2021 р. № 1665 Ст*
2. Термін подання студентом роботи до екзаменаційної комісії *«13 грудня 2021 р.*
3. Вихідні дані до роботи *Об'єкт дослідження – корпоративні комп'ютерні мережі. Предмет дослідження – топологічні структури корпоративних комп'ютерних мереж. Функція – оптимізація топологічних структур централізованих трирівневих корпоративних мереж. Розмірність задач – до 60 елементів. Технічне забезпечення: ІВМ-сумісний персональний комп'ютер. Програмний засіб з графічним інтерфейсом.*
4. Перелік питань, що потрібно опрацювати в роботі *Вступ; аналіз сучасного стану проблеми підтримки прийняття рішень з реінжинірингу корпоративних комп'ютерних мереж; комп'ютерні мережі та їх топології; корпоративні комп'ютерні мережі як об'єкти реінжинірингу; технології проектування комп'ютерних мереж; постановка мети та задач дослідження; моделі та методи підтримки прийняття проектних рішень; висновки; перелік джерел посилання; обґрунтування вибору архітектури додатку та мови програмування; результати експериментів; текст програми; опис програми; графічна частина; відомість кваліфікаційної роботи.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п. 5 включається до завдання за рішенням випускової кафедри) *Кресленики, схеми, плакати та/або комп'ютерні ілюстрації (слайди) на аркушах формату А4, що включаються до тексту пояснювальної записки або складу додатків: схема зв'язку категорій "елемент", "відношення", "топологія" та "властивість" у процесі системної оптимізації мереж; схема декомпозиції проблеми синтезу корпоративних мереж; схема алгоритму розв'язання задачі; екранні форми програмного засобу..*

6. Консультанти розділів роботи (п.6 включається до завдання 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	Терміни виконання етапів роботи	Примітка
1	*Отримання, аналіз завдання, уточнення плану*		
	роботи	*01.09.21*	*Виконано*
2	*Аналіз корпоративних комп'ютерних мереж*		
	як об'єктів реінжинірингу	*08.09.21*	*Виконано*
3	*Огляд існуючих методів реінжинірингу*		
	корпоративних комп'ютерних мереж	*15.09.21*	*Виконано*
4	*Розробка математичного забезпечення задачі*	*29.09.21*	*Виконано*
5	*Розробка програмного забезпечення задачі*	*13.10.21*	*Виконано*
6	*Проведення експериментальних досліджень*	*27.10.21*	*Виконано*
7	*Підготовка публікацій за результатами*		
	дослідження	*10.11.21*	*Виконано*
8	*Оформлення пояснювальної записки*	*24.11.21*	*Виконано*
9	*Подання закінченої роботи науковому керівникові*	*03.12.21*	*Виконано*
10	*Усунення зауважень наукового керівника*	*05.12.21*	*Виконано*
11	*Підготовка презентації*	*07.12.21*	*Виконано*
12	*Подання роботи на рецензування*	*08.12.21*	*Виконано*
13	*Попередній захист*	*14.12.21*	*Виконано*
14	*Подання роботи до екзаменаційної комісії*	*16.12.21*	*Виконано*

Дата видачі завдання *«01» вересня 2021 р.*

Студент _____
(підпис)

Керівник роботи _____ *проф. Безкоровайний В. В.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до магістерської кваліфікаційної роботи: 77 с., 13 рис., 5 табл., 38 джерел інформації.

ГЕНЕТИЧНИЙ АЛГОРИТМ, ЕВОЛЮЦІЙНИЙ ПОШУК, КОРПОРАТИВНА КОМП'ЮТЕРНА МЕРЕЖА, МЕТОДИ ПРИЙНЯТТЯ ПРОЕКТНИХ РІШЕНЬ, РЕІНЖИНІРИНГ.

Об'єкт дослідження – корпоративні комп'ютерні мережі.

Предмет дослідження – топологічні структури корпоративних комп'ютерних мереж.

Мета роботи – підвищення ефективності систем підтримки прийняття проектних рішень за рахунок удосконалення методу оптимізації топологічних структур корпоративних комп'ютерних мереж.

Методи дослідження – системний, агрегативно-декомпозиційний, блочно-ієрархічний підходи, методи автоматизації проектування, спрямованого перебору, еволюційного синтезу на основі генетичного алгоритму.

Запропоновано удосконалення еволюційного методу спрямованого перебору, побудованого на основі генетичного алгоритму, для оптимізації топологічних структур корпоративних комп'ютерних мереж. Розроблено програмну реалізацію методу для оптимізації централізованих трирівневих мереж.

Отримані результати можуть бути використані для розв'язання задач оптимізації корпоративних комп'ютерних мереж офісів компаній, виробничих підприємств, наукових, навчальних лабораторій тощо. Їх використання дозволить скорочувати час розв'язання задач оптимізації, розв'язувати задачі більшої розмірності і за рахунок цього підвищувати якість проектних рішень і зменшувати витрати на створення й експлуатацію мереж.

ABSTRACT

Master’s Thesis: 77 pages, 5 tables, 13 figures, 38 title.

CORPORATE COMPUTER NETWORK, EVOLUTIONARY SEARCH, GENETIC ALGORITHM, PROJECT DECISION-MAKING METHODS, REENGINEERING.

The object of research is corporate computer networks.

The subject of research is topological structures of corporate computer networks.

The purpose of the work is to increase the efficiency of project decision support systems by improving the method of optimizing the topological structures of corporate computer networks.

Research methods – system, aggregative-decompositional, block-hierarchical approaches, methods of automation of design, directed search, evolutionary synthesis based on genetic algorithm.

It is proposed to improve the evolutionary method of directed search, based on a genetic algorithm, to optimize the topological structures of corporate computer networks. The software implementation of the method for optimization of centralized three-level networks has developed.

The obtained results can be used to solve problems of optimization of corporate computer networks of offices of companies, manufacturing enterprises, research, training laboratories, etc. Their use will reduce the time to solve optimization problems, solve larger problems, and thus improve the quality of design solutions and reduce the cost of creating and operating networks.

ЗМІСТ

Перелік скорочень 7
Вступ 8
1 Аналіз сучасного стану проблеми підтримки прийняття рішень з реінжинірингу корпоративних комп'ютерних мереж 10
1.1 Комп'ютерні мережі та їх топології 10
1.2 Корпоративні комп'ютерні мережі як об'єкти реінжинірингу 12
1.3 Технології проектування комп'ютерних мереж 20
1.4 Моделі і методи підтримки прийняття проектних рішень 28
1.5 Постановка мети та задач дослідження 36
2 Постановка та метод розв'язання задачі реінжинірингу корпоративних комп'ютерних мереж 38
2.1 Постановка задачі оптимізації мережі 38
2.2 Метод розв'язання задачі 40
2.3 Процедури відбору, схрещування та мутації генетичного алгоритму 45
3 Експериментальна частина 48
3.1 Обґрунтування вибору архітектури додатку та мови програмування 48
3.2 Опис життєвого циклу додатку 55
3.3 Результати експериментів 63
Висновки 72
Перелік джерел посилання 74
Додаток А Графічний матеріал атестаційної роботи 78
Додаток Б Текст програми 88
Додаток В Заява щодо самостійності виконання роботи та можливості її публікації 100
Додаток Г Протокол перевірки тексту пояснювальної записки електронною системою на плагіат 102
Додаток Д Відомість кваліфікаційної роботи 104

ПЕРЕЛІК СКОРОЧЕНЬ

ККМ – корпоративна комп’ютерна мережа.

ОПР – особа, яка приймає рішення.

ФКЧК – функція корисності часткових критеріїв.

ФУК – функція узагальненої корисності.

AMQP – Advanced Message Queuing Protocol.

CLI – Command-line interface.

HTML – HyperText Markup Language.

HTTP – HyperText Transfer Prorocol.

JDBC – Java Data Base Connector.

JVM – Java Virtual Machine.

MVC – Model View Controller.

REST – Representational state transfer.

UML – Unified Modeling Language.

WAR – Web application Archive.

XML – eXtensible Markup Language.

ВСТУП

Основу інфраструктури систем управління сучасних компаній складають корпоративні комп’ютерні мережі. Зміна кількості користувачів, переліку задач, які підлягають розв’язанню, елементної бази чи технологій обробки і передачі інформації на певному етапі призводять до необхідності модернізації мереж. Найбільш радикальні міри в мережах відбуваються при їх реінжинірингу. При цьому можуть розглядатись різні аспекти внутрішньої будови мережі: функціональна, організаційна, топологічна, інші види структур.

Ефективність мереж багато в чому визначається способом їх структурної організації. Корпоративна мережа може бути організована на різних наборах елементів (комп’ютерів, свічів, концентраторів, маршрутизаторів), різних архітектурах та типах каналів зв'язку між елементами. При цьому вартісні та функціональні характеристики багато в чому визначаються їхньою територіальною чи просторовою організацією.

В процесі реінжинірингу виникає необхідність розв’язання комплексу задач, що об’єднані у проблему структурно-функціонально-топологічної оптимізації. У ній рішення щодо кількості елементів, розподілу функцій між ними, технології їх реалізації та взаємозв'язки елементів мережі приймаються виходячи з їх просторового розташування. Це обумовлює актуальність задач оптимізації топологічних структур мереж.

Метою роботи є підвищення ефективності систем підтримки прийняття проектних рішень за рахунок удосконалення методу оптимізації топологічних структур корпоративних комп’ютерних мереж.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз сучасного стану проблеми підтримки прийняття проектних рішень з реінжинірингу корпоративних комп’ютерних мереж;

– сформулювати постановку задачі структурно-топологічної оптимізації централізованих трирівневих мереж;

– удосконалити обраний метод оптимізації топологічних структур корпоративних мереж;

– програмно реалізувати метод оптимізації мереж та провести дослідження його ефективності.

Об'єктом дослідження є корпоративні комп'ютерні мережі.

Предметом дослідження є топологічні структури корпоративних комп'ютерних мереж.

Методи дослідження – системний підхід, методи автоматизації проектування, структурно-топологічної оптимізації, сучасні інформаційні технології.

У роботі запропоновано удосконалення еволюційного методу спрямованого перебору, побудованого на основі генетичного алгоритму, для оптимізації топологічних структур корпоративних комп'ютерних мереж. Розроблено програмну реалізацію методу для оптимізації централізованих трирівневих мереж.

Отримані результати можуть бути використані для розв'язання задач оптимізації корпоративних комп'ютерних мереж офісів компаній, виробничих підприємств, наукових, навчальних лабораторій тощо. Їх використання дозволить скорочувати час розв'язання задач оптимізації, розв'язувати задачі більшої розмірності і за рахунок цього підвищувати якість проектних рішень і зменшувати витрати на створення й експлуатацію мереж.

За результатами, отриманими в процесі виконання дослідження, підготовлено матеріали для збірнику студентських наукових статей «Автоматизація та приладобудування» («Automation and Development of Electronic Devices» ADED-2021) та доповідь на Всеукраїнську науково-практичну конференцію здобувачів вищої освіти і молодих учених «Комп'ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» (м. Харків, 2021).

1 АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ З РЕІНЖИНІРИНГУ КОРПОРАТИВНИХ КОМП'ЮТЕРНИХ МЕРЕЖ

1.1 Комп'ютерні мережі та їх топології

Комп'ютерна мережа – це кластер комп'ютерів по спільному шляху зв'язку, які працюють для обміну ресурсами від одного комп'ютера до іншого, наданих вузлами мережі або розташованих на них.

До теперішнього часу набули популярності такі типи комп'ютерних мереж [1]:

– персональна мережа (PAN);

– локальна мережа (LAN);

– глобальна мережа (WAN);

– бездротова локальна мережа (WLAN);

– мережа кампусу (CAN);

– мережа столичного району (MAN);

– мережа зберігання даних (SAN);

– системна мережа (SAN);

– пасивна оптична локальна мережа (POLAN);

– корпоративна приватна мережа (EPN);

– віртуальна приватна мережа.

Топологія визначає структуру мережі, як усі компоненти взаємопов'язані один з одним. Існує два типи топології: фізична та логічна.

Фізична топологія – це геометричне представлення всіх вузлів мережі.

Існують наступні типи фізичних топологій:

– топологія «точка-точка» – це найпростіша топологія, яка з'єднує два вузли безпосередньо разом за допомогою загального зв'язку. Вся пропускна здатність загального каналу зарезервована для передачі між цими двома вузлами. Для з'єднання «точка-точка» для з'єднання двох кінців використовується

фактична довжина дроту або кабелю, але можливі й інші варіанти, наприклад, супутникові лінії [2];

– топологія «шина» – розроблена таким чином, що всі станції підключаються за допомогою одного кабелю, відомого як магістральний кабель. Кожен вузол або підключений до магістрального кабелю за допомогою відводного кабелю, або безпосередньо підключений до магістрального кабелю. Коли вузол хоче надіслати повідомлення через мережу, він розміщує повідомлення через мережу. Усі станції, доступні в мережі, отримають повідомлення незалежно від того, адресовано воно чи ні;

– топологія «кільце» – схожа на топологію шини, але зі сполученими кінцями. Вузол, який отримує повідомлення від попереднього комп’ютера, повторно передає його до наступного вузла. Дані надходять в одному напрямку, тобто односпрямовані;

– топологія «зірка» – це структура мережі, в якій кожен вузол підключений до центрального концентратора, комутатора або центрального комп’ютера. Центральний комп'ютер відомий як сервер, а периферійні пристрої, підключені до сервера, відомі як клієнти. Топологія зірка є найпопулярнішою топологією в реалізації мережі;

– топологія «дерево» – поєднує характеристики топології шини та топології зірки. Топологія дерева – це тип структури, в якій всі комп'ютери пов'язані один з одним ієрархічним чином. Між двома вузлами для передачі даних існує тільки один шлях. Таким чином, він формує ієрархію батьків і дітей;

– топологія «меш» – це структура мережі, в якій комп'ютери з'єднані один з одним за допомогою різних резервних з'єднань. Існує кілька шляхів від одного комп’ютера до іншого. Вона не містить комутатора, концентратора чи будь-якого центрального комп’ютера, який виступає як центральна точка зв’язку. Прикладом топології сітки є Інтернет [3];

– топологія «гібрид» – об’єднує дві або більше топології таким чином, що отримана мережа не має жодної зі стандартних топологій (наприклад, шина,

зірка, кільце тощо). Наприклад, деревовидна мережа (або мережа зірка-шина) є гібридною топологією, в якій мережі зірки з'єднані між собою за допомогою шинних мереж [4]. Однак мережа дерева, підключена до іншої мережі дерева, все ще топологічно є мережею дерева, а не окремим типом мережі. Гібридна топологія завжди створюється, коли підключаються дві різні базові топології мережі. Мережа «зірка-кільце» складається з двох або більше кільцевих мереж, з’єднаних за допомогою багатостанційного блоку доступу (MAU) як централізованого концентратора. Топологія «Сніжинка» — це зіркова мережа зіркових мереж. Два інших типи гібридних мереж: гібридна сітка та ієрархічна зірка [1].

1.2 Корпоративні комп’ютерні мережі як об’єкти реінжинірингу

В ринкових умовах, де постійні лише зміни, існує гостра постійна необхідність в нових інструментах і методах, які здатні допомогти підприємствам стати більш ефективними. Постійно зростаюча конкуренція ініціює потребу суб’єктів ринку наздогнати інших і стати першими поки не пізно, а для цього потрібні механізми, які можуть спростити дуже складні речі. Як видно з досвіду, найбільш успішними є ті зміни, які відбуваються в критичні моменти, тобто чим більша реальна небезпека кризи чи банкрутства, тим більша ймовірність успіху, оскільки в останньому випадку стимулюється ініціатива та активне впровадження змін на підприємстві, створення абсолютно нових та більше ефективних бізнес-процесів.

Концепція покращення бізнес-процесів ґрунтується на чотирьох підходах:

- методика швидкого аналізу рішення (FAST);
- бенчмаркінг процесу;
- перепроектування процесу;
- реінжиніринг процесу.

Методика швидкого аналізу рішення ґрунтується на способі покращення, вперше використаним компанією IBM в середині 80-х. В 90-х роках цей підхід був удосконалений компанією Дженерал Електрик. Методика швидкого аналізу рішення – проривний підхід, який концентрує увагу групи на певному процесі в ході одно-дводенної наради для визначення способів, якими група може покращити цей процес впродовж наступних 90 днів. Перед закінченням наради керівництво погоджує або відхиляє запропоновані покращення. Методика швидкого аналізу рішення може застосовуватись до заходів будь-якого рівня, починаючи з основних процесів і закінчуючи рівнем заходів. FAST-підхід до покращення бізнес-процесів зосереджується на одно- чи дводенній зустрічі, в ході якої визначаються причини проблеми та заходи, які не додають цінності в бізнес-процесі. Типовими покращеннями при застосуванні FAST-підходу є зниження витрат, тривалості циклу та рівня помилок на 5-15% за тримісячний період [5].

Бенчмаркінг процесу – систематичний метод визначення, усвідомлення та творчого розвитку товарів, проектів, послуг, обладнання, процесів та процедур (встановлених принципів) більш високої якості для покращення поточної діяльності організації через вивчення того, як різні організації виконують однакові чи схожі операції. Зазвичай, бенчмаркінг знижує витрати, тривалість циклу та кількість помилок на 20-50%. При реалізації типового проекту бенчмаркінга процеса розробка найбільш вигідного, націленого на майбутнє рішення, займає від 4-х до 6-ти місяців. При бенчмаркінгу бізнес-процесів ключові процеси ідентифікуються, осмислюються і порівнюються з кращими еквівалентними процесами для визначення небажаних відхилень. Як правило, ґрунтуючись на порівняльному аналізі, визначають декілька компаній, які функціонують краще, ніж компанія, що проводить дослідження. Після чого, команда, яка проводить бенчмаркінг, оцінює процеси іншої компанії для того, щоб з’ясувати, чому вони функціонують краще, ніж процеси організації, яка проводить дослідження. В результаті використовуються дані результати для розробки і впровадження покращених процесів, які поєднують в собі риси

процесів «еталонних» компаній і при цьому часто створюють процеси, які виявляються кращими, ніж будь-який із досліджуваних ними раніше.

Підхід до перепроектування процесу концентрує зусилля на вдосконаленні існуючого процесу. Перепроектування як правило використовується для тих процесів, які досить успішні в даний момент. Такий підхід використовується в тому випадку, якщо покращення показників діяльності на 30-60% (як правило до таких результатів призводить перепроектування) дозволить компанії отримати конкурентні переваги [6].

Реінжиніринг бізнес-процесів – найбільш радикальний із всіх чотирьох підходів до покращення бізнес-процесів. Його також часто називають інновацією процесу, оскільки його успіх в основному ґрунтується на інноваціях та творчих здібностях команди по покращенню процесу. Такий підхід забезпечує новий погляд на цілі процесу і повністю ігнорує існуючий процес і структуру організації. Все починається з чистого листка паперу так само, якби ви тільки почали розробляти цей процес. Реінжиніринг БП, якщо його проводити правильно, знижує витрати та тривалість циклу на 60-90% і рівень помилок на 40-70%. Даний підхід використовується в тих випадках, коли процес на теперішній момент настільки застарілий, що не варто навіть намагатись його зберегти або поліпшувати. Команда по покращенню процесу ніби відходить назад і оглядає процес свіжим оком, ставлячи перед собою питання, як би спланували цей процес, якби не було жодних обмежень. Підхід використовує можливості, які надаються доступними інструментами процесу, включаючи самі останні досягнення в сфері механізації, автоматизації та інформаційних технологій і одночасно покращують ці інструменти. Часто цей процес стимулює команду по покращенню процесу до розробки принципово нового проекту процесу, який стає справжнім проривом [7].

У процесі проектування чи створення будь-якої корпоративної комп’ютерної мережі, її реорганізації чи плануванні її розвитку неминуче виникають завдання синтезу структури. При цьому можуть розглядатись різні аспекти внутрішньої будови мережі: організаційна, топологічна (просторова),

функціональна, інші види структур [8, 9]. Особливої важливості завдання структурного синтезу набувають для територіально розподілених (великомасштабних) корпоративних комп’ютерних мереж. Для них збільшення відстаней між функціональними підсистемами призводить до появи нової системної властивості, не властивого для територіально зосереджених комп’ютерних мереж. Воно пов'язано з тим, що структурні, вартісні та функціональні характеристики розподілених комп’ютерних мереж багато в чому визначаються топологією (розміщенням) їх підсистем та елементів. Топологія підсистем та елементів, своєю чергою, визначає топологію комунікаційних зв'язків, які забезпечують функціонування корпоративної комп’ютерної мережі як єдиного цілого, реалізуючи обмін між елементами та підсистемами ресурсами, енергією, інформацією [10]. До числа територіально розподілених можуть бути віднесені багато з сучасних технічних, організаційно-технічних, соціально-економічних систем: інформаційно-обчислювальні мережі, системи розподілу інформації, спеціалізовані транспортні системи, розподілені системи обслуговування, виробничо-збутові комплекси, територіальні АСК, інші системи [11].

Корпоративна комп’ютерна мережа (ККМ), що формально подається кортежем $s = <E, R>$ (де E – множина елементів мережі; R – множина безпосередніх зв’язків між елементами), може мати множину різних топологічних реалізацій G^*. Кожній з топологічних реалізацій (варіантів розміщення елементів) мережі s, буде відповідати свій набір функціональних і вартісних показників:

$$\phi: (E, R, G) \rightarrow P, \qquad (1.1)$$

де ϕ – деяке відображення, яке ставить у відповідність варіанту мережі набір її характеристик.

При вирішенні завдань оптимізації формальне подання корпоративних мереж має відображати її топологічні властивості. Для цього доцільно подавати їх опис у розширеному вигляді:

$$s = < E, R, G >, \tag{1.2}$$

де G – топологія структури $<E, R>$.

На етапі аналізу мети створення (оптимізації) комп'ютерної мережі виділяється множина найважливіших властивостей (оперативність, надійність, живучість тощо) P', якими вона має володіти. Виділена множина властивостей P' є підмножиною універсальної множини властивостей. До універсальної множини належать властивості, які можуть бути отримані на універсальних множинах елементів E^U (комп'ютерів, іншого обладнання) , відношень (архітектур) R^U і топологій (місць розташування елементів мережі) G^U:

$$P^U = \phi\ (E^U, R^U, G^U) \tag{1.3}$$

Множина E^U в (1.3) включає всі можливі типи елементів, на яких може бути побудована мережа. Множина відношень (архітектур) R^U визначається можливими принципами побудови мережі, розподілом функцій між її елементами і відображає можливі схеми безпосередніх зв'язків між елементами множини E^U. Склад множини безпосередніх зв'язків R^U визначається складом універсальної множини елементів E^U, а склад множини універсальної множини місць розташування G^U визначається складом множини елементів E^U і архітектур R^U (рис. 1.1).

Відображення P' на множині елементів корпоративної мережі E^U, відношень між ними R^U і їхньої топології G^U визначає підмножини елементів E', архітектур R' і топологій G', на яких може бути побудована мережа з виділеними властивостями P'.

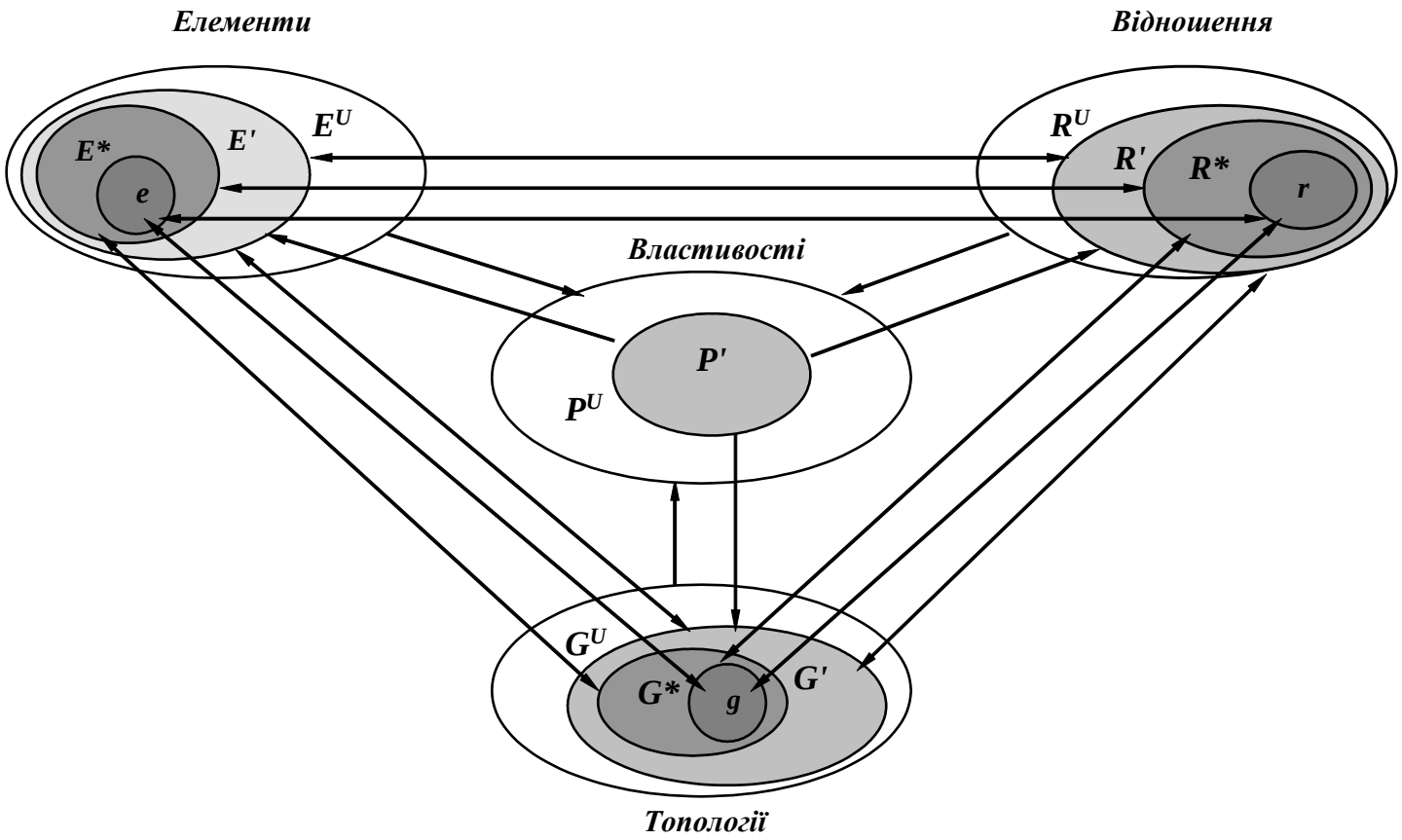

Рисунок 1.1 – Схема зв'язку категорій "елемент", "відношення", "топологія" та "властивість" у процесі системної оптимізації мережі

У такий спосіб формується сфера існування мережі $S'=\{s\}$, яка з урахуванням обмежень звужується до допустимої множини варіантів її побудови $S^*=\{s\}, S*\subseteq S', E*\subseteq E'\subseteq E^U,\ R*\subseteq R'\subseteq R^U, G*\subseteq G'\subseteq G^U$ [12].

На наступних етапах завдання оптимізації полягає у виборі таких підмножин елементів $E^o\subseteq E^*$, архітектур $R^o\subseteq R^*$ і місць розташування елементів $G^o\subseteq G^*$ із допустимої множини $S^*=\{s\}$, які забезпечують з мінімальними витратами C^o досягнення необхідних функціональних характеристик мережі P'.

Множина формалізованих властивостей $P'=\{p_i\}$ є множиною часткових (локальних) критеріїв ефективності варіантів побудови мережі. Серед найбільш загальних вимог, що пред'являються до корпоративних комп'ютерних мереж, виділяються: якість, терміни, вартість, надійність виконання функцій та живучість [10, 13].

Як показники якості можуть розглядатись достовірність і точність розв'язання інформаційно-управляючих задач. Рівень якості мережі визначається складом типових елементів $E\subseteq E^*$.

Виконання покладених на мережу функцій має здійснюватися у мінімальні $\tau\rightarrow min$ або обмежені терміни $\tau\leq\tau^*$ (де τ^* – допустимий час реалізації функцій мережі).

Показник вартості C оцінюється витратами на створення та (або) експлуатацію мережі. Іноді під вартістю розуміється її питомий показник, наприклад, витрати на реалізацію однієї функції (розв'язання однієї задачі).

На практиці прагнуть до інтегральності часткових критеріїв $K=\{k_1, k_2,..., k_m\}$, тобто, щоб їх кількість була меншою за кількість найважливіших властивостей $|K|<|P|$. Ці умови зазвичай виконуються, бо деякі властивості є взаємозалежними (змінюються узгоджено). Так вимога неможливості відмови у розв'язанні інформаційно-обчислювальних задач призводить до того, що ненадійність одних елементів чи каналів зв'язку мережі збільшує час виконання функції з використанням інших, а збільшення кількості чи потужності елементів

мережі знижує рівень її завантаження, збільшуючи витрати на її створення й експлуатацію.

Оцінка варіантів побудови мережі може бути здійснена з використанням функціонально-вартісного аналізу [14]. Метою оптимізації корпоративної мережі є максимізація її ефективності (отримання максимального співвідношення ефекту від її використання Q і витрачених ресурсів C). Зазвичай вдається отримати узагальнені оцінки ефекту та вартості мережі:

$$Q = F_1(E,\ R,\ G), \tag{1.4}$$

$$C = F_2(E,\ R,\ G) \tag{1.5}$$

де $E,\ R,\ G$ – множини елементів, відношень між елементами та їх топологія мережі.

Функціональний ефект від використання мережі є неспадаючою функцією від витрачених на його досягнення ресурсів (вартості мережі):

$$\bar{Q} = F(\bar{C}),$$

де $\bar{Q}$ і $\bar{C}$ – скалярні оцінки ефекту та витрат;

F – оператор, який відображає стратегію використання ресурсів, витрачених на побудову чи експлуатацію мережі.

На ранніх етапах проектування виникає завдання вибору варіанта побудови ККМ за критерієм "ефект-вартість":

$$K_{QC} = \underset{Q,\,C,\,F}{opt}\ \Theta(Q, C, F)\ , \tag{1.6}$$

де $opt\Theta$ – оператор, що визначає вид узагальненого критерію ефективності.

В умовах обмежень на показники ефекту та витрат задача оптимізації мережі може бути подана у таких формах:

$$s_1^o = arg \max_{s \in S*} (\bar{Q}(s) - \bar{C}(s):\ \bar{Q}(s) \geq \bar{Q}^*,\ \ \bar{C}(s) \leq \bar{C}^*), \quad (1.7)$$

$$s_2^o = arg \max_{s \in S*} (\bar{Q}(s)/\bar{C}(s):\ \bar{Q}(s) \geq \bar{Q}^*,\ \ \bar{C}(s) \leq \bar{C}^*), \quad (1.8)$$

де $\bar{Q}^*$, $\bar{C}^*$– граничні рівні оцінок ефекту та витрат на мережу;

$S*=\{s\}$ – множина допустимих варіантів побудови мережі.

Частковими випадками задач оптимізації (1.7) – (1.8) є задачі з послабленими обмеженнями на показники ефекту чи витрат [10, 12]:

– для заданих обмежень на витрати визначити варіант побудови ККМ, що максимізує її наведений ефект:

$$s_3^o = arg \max_{s \in S*} (\bar{Q}(s):\ \ \bar{C}(s) \leq \bar{C}^*); \quad (1.9)$$

– для заданих обмежень на наведений ефект від використання мережі витрати визначити варіант її побудови, що мінімізує витрати:

$$s_4^o = arg \min_{s \in S*} (\bar{C}(s):\ \bar{Q}(s) \geq \bar{Q}^*). \quad (1.10)$$

Проблема оптимізації ККМ включає комплекс задач вибору структури, топології, технології функціонування, параметрів елементів та зв'язків, оцінки та вибору найкращого варіанту її побудови за множиною показників. Описи ККМ та задач їх синтезу у вигляді (1.1) – (1.10) є досить загальними. Для отримання проектних рішень потрібна їх деталізація з урахуванням особливостей мереж, що оптимізуються.

1.3 Технології проектування комп’ютерних мереж

З метою забезпечення ефективності рішень на всіх етапах життєвого циклу ККМ розроблена методології їх структурного синтезу як територіально

(просторово) розподілених об’єктів. В ній передбачається декомпозиція проблеми оптимізації мережі на комплекси завдань, які відносяться до різних рівнів її опису та етапів синтезу, розробку комплексу математичних моделей, проектних процедур, логічної схеми та технології проектування.

У загальному випадку корпоративна мережа може складатися з великої кількості елементів зі складною архітектурою взаємозв'язків між ними. Створення єдиного опису для оптимізації мережі на всіх етапах її життєвого циклу є досить складним завданням [15]. Проблема синтезу мережі складається з сукупності не повністю визначених завдань проектування. Для них заранне сконструйовано схеми проектування і не синтезовано відповідні моделі. Виходячи з цього, проблему синтезу корпоративних мереж як територіально розподілених об’єктів мереж відносять її до слабоструктурованих [16, 17].

Складність більшості корпоративних мереж не дозволяє створювати їх єдиний формалізований опис (математичну модель) і знаходити по ньому ефективні проектні рішення з використанням загальної проектної процедури.

Методологія структурного синтезу корпоративних мереж заснована на методології агрегативно-декомпозиційного та блочно-ієрархічного підходів. Вони передбачають декомпозицію опису мережі на ієрархічні рівні та аспекти. При цьому сам процес проектування розбивається на групи проектних процедур. Такі процедури призначені для отриманням та перетворенням описів мережі (рішень) щодо виділених рівнів та аспектів. На заключній стадії такі описи агрегуються для отримання на відповідному рівні рішень щодо мережі в цілому [8, 18].

Проблема синтезу корпоративної мережі подається як метазавдання *MetaTask*, яке складається з множин задач на різних рівнях декомпозиції. Такі задачі мають множини зв’язків між собою за вихідними даними та результатами розв’язання (рис. 1.2):

$$MetaTask = \{Task^l\},\quad Task^l = \{Task_i^l\},\ i = \overline{1, i_l},\ \ l = \overline{1, n_l}, \tag{1.11}$$

де $Task^l$ – множина задач синтезу мережі, що відносяться до l-го рівня декомпозиції;

n_l – кількість рівнів опису мережі;

i – номер задачі;

i_l – кількість задач на l-му рівні декомпозиції.

Кожна з задач на цьому етапі розглядається як деякий перетворювач своїх вхідних даних у вихідні:

$$Task_i^l: = In_i^l \rightarrow Out_i^l, i = \overline{1, i_l},\ l = \overline{1, n_l}, \tag{1.12}$$

де $In_i^l,\ Out_i^l$ – вхідні та вихідні дані i-ї задачі, що відноситься до l-го рівня.

Кожна з виділених задач $Task_i^l,\ i = \overline{1, i_l},\ l = \overline{1, n_l}$ за необхідності може бути подана у вигляді множини підзадач $Task_i^l$=$\{Task_{ij}^l\},\ j = \overline{1, j_i}$, де j_i – кількість підзадач i-ї задачі l-го рівня декомпозиції.

Системологічний аналіз проблеми синтезу ККМ та огляд її сучасного стану дозволяють зробити висновок про доцільність використання у конструкторському та техніко-економічному аспектах трьох ступенів деталізації їх опису на мета-, макро- та мікрорівнях.

На метарівні проблема *MetaTask* розглядається загалом, аналізується її місце серед інших проблем управління муніципального, відомчого, регіонального чи іншого масштабу.

Більшість завдань макрорівня за своєю сутністю є завданнями системного проектування та відрізняються обмеженнями, що відображають специфіку основних етапів життєвого циклу ККМ [10]:

$$Task^1 = \{Task_i^1\},\ i = \overline{1,5.} \tag{1.13}$$

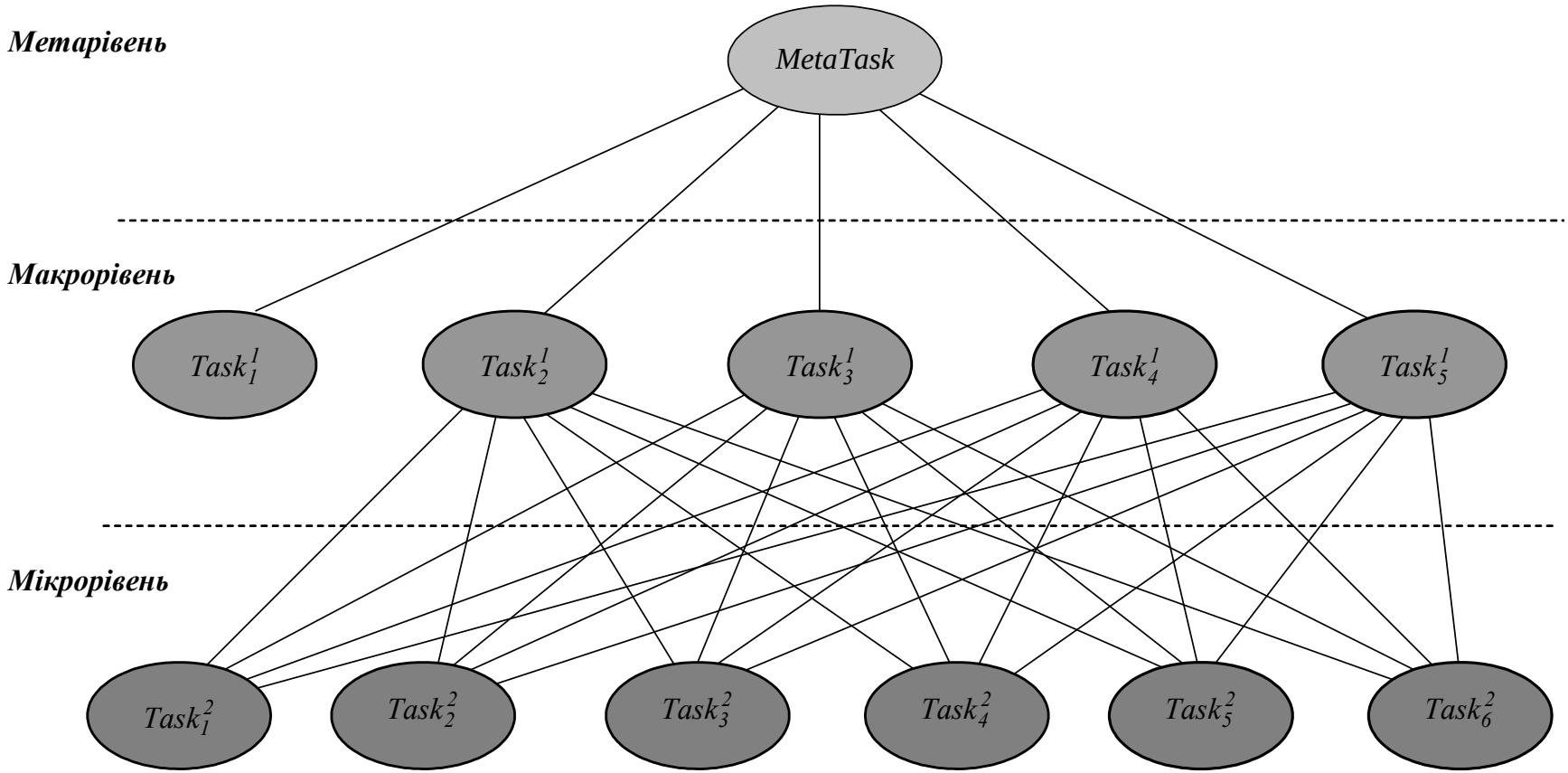

Рисунок 1.2 – Схема декомпозиції проблеми синтезу корпоративних мереж як територіально розподілених об’єктів

До основних задач макрорівня проблеми синтезу корпоративних мереж (1.13) відносяться:

– $Task_1^1$ – формування вимог до корпоративної мережі та розробка технічного завдання на проектування (оптимізацію);

– $Task_2^1$ – системне проектування мережі;

– $Task_3^1$ – планування розвитку мережі;

– $Task_4^1$ – структурна адаптація мережі;

– $Task_5^1$ – реінжиніринг мережі.

Розв'язання задачі $Task_1^1$ передбачає визначення головних цілей, задля досягнення яких створюється мережа. У процесі її вирішенні уточняється множина розв'язуваних нею завдань і визначаються принципи її побудови *П*. При цьому основна полягає у визначенні раціонального співвідношення показників ефекту від її використання Q^* та витрат на її створення й експлуатацію C^*. Це відповідає вирішенню задачі (1.6) та визначенню області існування мережі *S'*. Особливу важливість подібні задачі мають при створенні мереж нових видів, які не мають аналогів за цілями або виконуваними функціями. Для її рішення можна використати апарат функціонально-вартісного аналізу [14].

Особливістю цієї задачі є оціночний характер вихідних даних, їх низька достовірність. Це призводить до евристичності характеру отримуваних оцінок. Переважна більшість оцінювань вихідних даних отримується експертним методом.

Суть завдання системного проектування корпоративної мережі $Task_2^1$ полягає у визначенні найкращого за множиною обраних критеріїв *K* варіанта її побудови s^o. При цьому враховуються допустимі принципи її побудови *П*, а також задані структурні, топологічні, параметричні та технологічні обмеження $S'=\{s\}$, , а також допустимі рівні ефекту Q^* та витрат на мережу C^*. Ступінь визначеності вихідної інформації для цього завдання є вищою, ніж для попереднього.

Завдання планування траєкторії розвитку корпоративної мережі $Task_3^1$ полягає у виборі для заданої множини періодів часу $\{t\}$ зміни найкращих варіантів її побудови s_t^o при зміні вимог до неї, ресурсних обмежень C_t^*, обмежень на якість виконання нею функцій Q_t^*. Головна мета полягає у визначенні найкращої за множиною критеріїв K послідовності введення в експлуатацію її додаткових фрагментів, елементів та каналів зв'язку, що забезпечують на кожному відрізку часу необхідний рівень ефекту Q_t^* при виконанні обмежень на розміри витрат C_t^*.

Завдання адаптації мережі $Task_4^1$ вирішується в процесі її експлуатації. Воно пов'язане з необхідністю відносно незначних структурних, технологічних, топологічних або параметричних змін існуючого варіанту побудови мережі $s^o \in S'$ у зв'язку зі змінами множини та (або) характеристик її користувачів, виведенням з ладу деяких елементів (підмереж), підвищенням необхідного рівня ефекту $\tilde{Q}^*$ або зниження витрат на її експлуатацію.

Задача реінжинірингу корпоративної комп'ютерної мережі $Task_5^1$ вирішується в процесі її експлуатації. Вона виникає за необхідності внесення суттєвих змін до структури, технології функціонування, розміщення її елементів або параметрів елементів і каналів зв'язку. Це виникає внаслідок зміни множини та (або) вимог користувачів до засобів мережі, розширенням множини функціональних завдань мережі, удосконаленням елементної бази та (або) технологій збору, збереження, передачі й обробки інформації $\tilde{S}'$. Такі зміни можуть приводити до неефективності існуючого варіанту побудови мережі s^o [19]. При цьому допускається не лише модернізація елементів мережі і каналів зв'язку між ними, так і їхня заміна на більш сучасні (продуктивні).

Основні задачі, які підлягають розв'язанню на мікрорівн, присвячені системному проектуванню корпоративної мережі:

$$Task2 = \{Task_i^2\},\ i = \overline{1,6}, \tag{1.14}$$

де $Task_1^2$ – визначення принципів побудови корпоративної мережі;

$Task_2^2$ – вибір структури корпоративної мережі;

$Task_3^2$ – вибір топології елементів та зв'язків мережі;

$Task_4^2$ – вибір технології функціонування мережі;

$Task_5^2$ – визначення параметрів елементів та зв'язків між елементами мережі;

$Task_6^2$ – оцінка ефективності варіантів побудови мережі та вибір найкращого серед них.

Вибір принципів побудови та технології функціонування мережі π з множини допустимих Π здійснюється на підставі знань та досвіду проектувальників. При цьому множина варіантів, що визначає область існування корпоративної мережі S' (множини елементів E', відношень R' і топологій G'), скорочується до множини допустимих варіантів побудови корпоративної мережі $S^* \subseteq S'$. Вона визначається множинами допустимих елементів мережі $E^* \subseteq E'$, зв'язків між ними $R^* \subseteq R'$ і їхнім розміщенням (топологією) $G^* \subseteq G'$.

Завдання вибору структури корпоративної мережі $Task_2^2$ полягає у довизначенні варіанта її побудови s_{AB} (з заданими параметрами елементів та зв'язків $B = \psi_B(E, R)$, технологією її функціонування $A = \psi_A(E, R)$,) кількістю елементів $|E|$ та архітектурою $R \subseteq R^*$. Крім того здійснюється оцінка властивостей отриманого варіанта s_{ER} за визначеною множиною часткових критеріїв $K = \{k_1, k_2, ..., k_m\}$. Задача вирішується для попередньо визначених рівнів ефекту Q^* і витрат C^*.

Задача розміщення елементів мережі $Task_3^2$ полягає у визначенні для заданих її елементів E, каналів зв'язку між ними R, параметрів елементів і каналів B та технології функціонування A варіанта побудови корпоративної мережі s_{ERAB} з найкращим варіантом розміщення її елементів (топології) $G \subseteq G^*$.

Задача визначення найкращої технології функціонування мережі $Task_4^2$ присвячена довизначенню в умовах заданих множин її елементів E, каналів зв'язку між ними R, топології G, параметрів елементів і каналів $B = \psi_B(E, R)$

варіанту її побудови s_{ERGB} найкращим набором алгоритмів збору, передачі, зберігання й обробки інформації $A=\psi_A(E, R)$.

Задача визначення параметрів елементів та каналів зв'язку між елементами мережі $Task_5^2$ полягає у виборі варіанта побудови $s_B \in S^*$, який матиме найкращі їх значення B. Розв'язання цієї задачі здійснюється в умовах визначених структурних $<|E|, R>$, топологічних G та технологічних A характеристик мережі. Результати вирішення цієї задачі є основою для вибору типів вузлів мережі, елементів і каналів зв'язку між елементами серед визначених множин типових зразків.

Завдання визначення ефективності варіантів побудови мережі та вибору найкращого серед них $Task_6^2$ полягає в оцінці кожного з варіантів $s \in S^*$ за множиною часткових критеріїв $K=\{k_1, k_2, ..., k_m\}$ та розв'язанні задачі $s^o = \underset{s \in S^*}{arg\ opt}\ K(s)$. Її розв'язання здійснюється з урахуванням заданих структурних (E, R), топологічних G і технологічних A характеристик корпоративної мережі, а також параметрів її елементів та каналів зв'язку між ними B.

Подальша декомпозиція задач мікрорівня проблеми оптимізації мережі $Task_i^2=\{Task_{ij}^2\}$, $j=\overline{1, j_i}$ (де j_i – кількість підзадач задачі $Task_i^2$) призводить до комплексу нових завдань синтезу уже самих її елементів, її зв'язків та алгоритмів технології функціонування. У процесі системного проектування корпоративних мереж рішення щодо цього комплексу завдань вважаються відомими. Вони служать вхідними даними чи обмеженнями задач проблеми і подаються у вигляді множин допустимих значень функціональних і вартісних характеристик елементів, каналів зв'язку чи алгоритмів роботи B^*.

Для побудови ефективної технології оптимізації корпоративної мережі створюється логічна схема процесу її проектування. Для цього необхідна розробка комплексу моделей задач (1.11) з урахуванням їх зв'язків за вхідними і вихідними даними (1.12) – (1.14).

1.4 Моделі і методи підтримки прийняття проектних рішень

На всіх етапах проектування й експлуатації корпоративних комп'ютерних мереж виникає необхідність розв'язання підтримки задач прийняття рішень. Такі задачі формалізуються в термінах «умови – мета». Зазвичай «умови» розглядаються як стан ситуації прийняття рішень і способів переходу до іншого стану, а «мета» визначається як бажана ситуація. Формально задача прийняття рішень полягає у виборі альтернативи з множини допустимих $x \in X$, що призводить до деякого бажаного результату $u \in U$. Ефективність прийнятого рішення визначається ступенем відповідності отриманого наслідку з множини можливих $u \in U$ поставленій меті й оцінюється за множиною часткових критеріїв $K = \{ k_1, k_2, ..., k_m \}$ [23, 24].

Основними класами задач, що розв'язуються в процесі прийняття рішень є:

– задачі в умовах визначеності (кожному рішенню $x \in X$ відповідає єдиний наслідок з множини можливих $u \in U$. У цьому випадку існує однозначна залежність наслідків від прийнятих рішень;

– задачі в умовах ризику (кожному рішенню $x \in X$ відповідає декілька наслідків з множини можливих $u \in U_x$, які настають з певною ймовірністю);

– задачі в умовах невизначеності (кожному рішенню $x \in X$ відповідає декілька наслідків з множини можливих $u \in U_x$, але відсутня навіть стохастична залежність наслідків від прийнятих рішень).

У процесі розв'язання проблема прийняття проектних рішень щодо вибору варіантів оптимізації корпоративних мереж може розглядатися як деяка система *Pr*:

$$Pr = < Tasks, \ Rels >,$$

де *Tasks* – множина складових задачі прийняття рішень;

Rels – множина відношень між складовими задачі, що визначають взаємозв'язків між ними за вхідними і вихідним даними;

$$Tasks = \{ Task_i \}_{i=1}^{6},$$

де $Task_1$ – формалізація мети рішення, що приймається;

$Task_2$ – визначення універсальної множини проектних рішень X^U;

$Task_3$ – визначення множини допустимих проектних рішень $X \subseteq X^U$;

$Task_4$ – виділення підмножини ефективних проектних рішень $X^K \subseteq X \subseteq X^U$;

$Task_5$ – ранжирування множини проектних рішень $x \in X^K$;

$Task_6$ – вибору найкращого рішення $x^o \in X^K$.

У найпростішому випадку завдання формалізації мети прийняття рішень $Task_1$ полягає в побудові цільової функції $p(x) = k_1(x)$ з використанням лише одного критерію ефективності, значення якого задані на множині варіантів побудови мережі $x \in X$. У найбільш загальному випадку використовується множина суперечливих часткових критеріїв $K = \{ k_1, k_2, ..., k_m \}$. Часткові критерії $K = \{ k_1, k_2, ..., k_m \}$ зазвичай мають різні розмірність, фізичний зміст та інтервали зміни.

Розв'язання задачі встановлення універсальної множини варіантів побудови мережі X^U ($Task_2$) здійснюється з врахуванням специфіки мережі або ситуації прийняття рішення.

Задача визначення множини допустимих варіантів побудови мережі $X \subseteq X^U$ ($Task_3$) полягає у видаленні з універсальної множини варіантів побудови мережі X^U підмножини таких $\overline{X}$, які не задовольняють встановленим обмеження на функціональні чи вартісні показники $X = X^U \setminus \overline{X}$. З цією метою попередньо визначать оцінки функціональних та вартісних характеристик варіантів побудови мережі $x \in X^U$. Основним засобом для оцінки характеристик

варіантів побудови мережі $x \in X^U$ за множиною показників $K = \{k_1, k_2, ..., k_m\}$ є аналітичне моделювання. Для скалярного оцінювання якості варіантів побудови мережі *P(x)* використовуються методи багатофакторного оцінювання, побудовані з використанням функцій корисності часткових критеріїв (ФКЧК) [25].

Задача виділення підмножини ефективних варіантів побудови мережі $X^K \subseteq X$ ($Task_4$) полягає у вилученні з множини допустимих таких, що є гіршими хоча б за один з інших одночасно за всіма показниками $K = \{k_1, k_2, ..., k_m\}$. Проектна рішення називають ефективним або Парето-оптимальним $x^o \in X$, якщо не існує більш кращого рішення $x \in X$, тобто $x^o \succ x$ для всіх варіантів $x \in X$.

Розв'язання задачі ранжування варіантів побудови мережі $Task_5$ здійснюється на основі максимізації їх загальної корисності. Для її розв'язання використовують знання та досвід особи, яка приймає рішення (ОПР) або процедури формування узагальненого критерію ефективності *P(x)*. В обох підходах кожному з варіантів побудови мережі $x^o \in X^K$ приписується якісне або кількісне значення його цінності $P(x)$, яке визначають порядок варіантів за якістю:

$$\forall\, x,y \in X:\ x \sim y \leftrightarrow P(x) = P(y);$$

$$x \succ y \leftrightarrow P(x) > P(y);$$

$$x \succsim y \leftrightarrow P(x) \geq P(y).$$

Після цього задача вибору найкращого варіанту побудови мережі $x^o \in X^K$ ($Task_6$) зводиться до вибору крайнього елемента впорядкованого ряду.

Це відповідає розв'язку задачі екстремізації функції узагальненої корисності (ФУК):

$$x^o = \arg \underset{x \in X^K}{extr} P(x).$$

Її розв'язання може бути здійснено існуючими методами математичного програмування.

Процеси прийняття проектних рішень з оптимізації мереж можуть включати розв'язання інших задач, що пов'язані зі специфікою ситуації вибору. При цьому кожна з розглянутих вище задач може бути розбита на множину нових підзадач.

Загальна схема вирішення проблеми багатокритеріального вибору варіанту побудови корпоративної мережі з використанням підходу на основі кількісного оцінювання подається в такому вигляді [24]:

$$S \rightarrow A \rightarrow opt\ \Theta \rightarrow x^o,$$

де S – ситуація прийняття проектного рішення;

A – набір аксіом, які визначають принцип упорядкування варіантів побудови мережі;

$opt\ \Theta$ – схема компромісу, що задається шляхом визначення узагальненого критерію (функції загальної корисності);

x^o – ефективний варіант побудови мережі.

Для вирішення проблеми підтримки прийняття проектних рішень рекомендується використання аналітико-евристичного підходу. Він передбачає вибір аксіоматики A особою, яка приймає рішення, на основі евристичних міркувань.

У процесі реінжинірингу топологічної структури корпоративних комп'ютерних мереж вирішується завдання розміщення центру, вузлів, її елементів та каналів зв'язку між ними на певній заданій території. Методи вирішення задачі розміщення вузлів при реінжинірингу представлені нижче.

Метод на основі покоординатної оптимізації топологічної структури мережі. Суть методу полягає у циклічному покращенні початкового варіанту

мережі шляхом послідовної оптимізації місць розміщення кожного з її u вузлів за умови фіксованого розміщення решти $u - 1$ вузла.

Алгоритм цього методу для оптимізації мережі за показником витрат подається такими кроками.

1. Задання вихідних даних: множина допустимих місць розміщення вузлів мережі; кількості вузлів мережі u; індексу поточного вузла $– j := 1$; значення ітерації – $i := 0$; значення лічильника проходів по точках розміщення l; значення показника якості для найкращого варіанта побудови мережі w^o; кращого поточного показника якості $\Delta C(w_i^l) = \infty$.

2. Сформувати початкове розміщення вузлів мережі w_i^l, розрахувати значення показника якості $\Delta C(w_i^l)$.

3. Збільшити значення лічильника $i := i + 1$; для вузла j в варіанті w_i^l змінити місце його розміщення при фіксованих розміщеннях інших u–1 вузлів.

4. Розрахувати значення показника якості $\Delta C(w_i^l)$. Якщо $\Delta C(w_i^l) \le \Delta C(w_{i-1}^l)$, то запам’ятати краще значення $\Delta C(w^l) := \Delta C(w_i^l)$, $w^o := w_i^l$ і перейти до кроку 5.

5. Збільшити значення індексу поточного вузла $j := j + 1$. Якщо $j < u$ перейти до кроку 3, в іншому випадку перейти до кроку 6.

6. Якщо l=0, присвоїти $w_i^{l+1} := w_i^l$, $l := l+1$, $j := 1$, і перейти до кроку 3, в іншому випадку перейти до кроку 7.

7. Якщо $\Delta C(w^l) \le \Delta C(w^{l-1})$, то $w_i^{l+1} := w_i^l$, $l := l+1$, $j := 1$ і перейти до кроку 3, в іншому випадку перейти до кроку 8.

8. Закінчення алгоритму: отримано варіант побудови мережі (розміщення вузлів) з найкращим із розглянутих значенням показника якості $\Delta C(w^o)$.

Метод імітації відпалу. Цей метод заснований на імітації процесу кристалізації речовини [26]. Основна ідея методу полягає в можливості переходів між варіантами побудови мережі у напрямку погіршення показника

якості рішень для можливості виходу з локальних екстремумів. Імовірність виконання такий дій зменшується в процесі пошуку кращих варіантів.

У процесі розв'язання задачі оптимізації мережі цим методом здійснюється розрахунок значення зміни енергії процесу *ΔE*, імовірності переходу до отриманого значення функції *P(ΔE)* та функції зниження температури процесу відпалу *T*.

Алгоритм цього методу можна подати такими кроками.

1. Завдання вхідних даних: множина місць допустимого розміщення вузлів корпоративної мережі; значень мінімальної t_{min}, максимальної t_{max} та початкової $t_1 = t_{max}$ температури відпалу; значення поточної ітерації алгоритму i:=0; кількості вузлів мережі $u := u_{max}$; кількості вибраних вузлів $k := u_{max}$.

2. Формування початкового варіанту: місць розміщення вузлів мережі w_0, $w^0 := w_0$; матриці зв'язків елементів, вузлів та центру мережі $x = [x_{ij}],\ i, j = \overline{1, n}$.

3. Перевірка умови про закінчення роботи алгоритму: якщо $t_1 \le t_{max}$, то необхідно перейти до кроку 9, в іншому випадку перейти до кроку 4.

4. Зміна значення поточної ітерації алгоритму $i := i + 1$ та генерація $w_i(k)$.

5. Визначення зміни енергії процесу ΔE (крок 4).

6. Якщо значення зміни енергії процесу $\Delta E \le 0$, то найкращий варіант побудови мережі отримує оцінку $w^0 := w_i$ і перейти до кроку 3, в іншому випадку перейти до кроку 7.

7. Розрахувати ймовірність переходу $p(\Delta E)$ (крок 5) і перейти, враховуючи її значення до отриманого варіанту.

8. Зменшити значення температури процесу $t_{i+1} := T(i)$ (крок 2), зменшити значення кількості вибраних вузлів k та перейти до кроку 3.

9. Закінчення роботи алгоритму: отримано варіант побудови мережі w^o з мінімальним з розглянутих показником значенням витрат $\Delta C(w^o)$.

Метод оптимізації варіанту побудови мережі на основі пошуку із заборонами. Ідея методу полягає в аналізі матриці найближчих сусідів по мережі. Для виходу з локальних екстремумів цільової функції здійснюється аналіз списку заборон, куди входить передісторія пошуку варіантів.

Алгоритм цього методу можна подати сукупністю таких кроків.

1. Завдання вхідних даних: множина місць допустимого розміщення вузлів мережі; списку заборон; околиці пошуку найближчих сусідів по мережі; кількості ітерацій i:=0; найкращого поточного значення критерію оцінки варіантів побудови мережі $\Delta C(w_i^1) = \infty$.

2. Сформувати список «найближчих сусідів» по мережі.

3. Сформувати початковий варіант побудови мережі w_0, $w^o := w_0$.

4. Якщо умову зупинки роботи алгоритму виконано, перейти до кроку 8, в іншому випадку перейти до кроку 5.

5. Сформувати варіант побудови мережі w_i на основі матриці «найближчих сусідів» та заданого списку заборон T.

6. Якщо знайдено кращий варіант побудови мережі $\Delta C(w_i) < \Delta C(w_{i-1})$, то $w^0 := w_i$ і додати елементи множини T до існуючого списку заборон.

7. Змінити значення кількості ітерацій $i := i + 1$ і перейти до кроку 4.

8. Закінчення роботи алгоритму: отримано варіант побудови мережі w^o з мінімальним з розглянутих показником значенням витрат $\Delta C(w^o)$.

Для припинення роботи алгоритму можна використовувати одну з таких умов [27]:

– виконання фіксованої кількості ітерацій;

– виконання фіксованої кількості ітерацій, за яких значення функції не покращується;

– по досягненню заданої точності рішення.

Одним з найбільш ефективних методів розв’язання задач оптимізації топологічних структур корпоративних мереж є еволюційні метод на основі

генетичних алгоритмів, що імітують процеси розмноження та відбору у живих організмах. Вони використовують специфічні оператори створення популяції, схрещування особин популяції (кросовер), мутації і селекції (відбору) [28-29].

Метод кластеризації k-means. Суть цього методу полягає в мінімізації віддалення груп комп'ютерів мережі від вузлів (точок доступу, локальних центрів, серверів) [30]. Це реалізується за рахунок розбиття множини елементів корпоративної мережі на задану кількість підмножин (кластерів). На кожній ітерації алгоритму здійснюється визначення центру мас кластера, який був отриманий у попередньому циклі алгоритму. В наступному циклі комп'ютери мережі повторно розбиваються на підмножини (кластери) за показником близькості (вартості підключення) до вузлів (центрів кластерів).

Алгоритм цього методу може бути поданий такою послідовністю кроків.

1. Завдання вхідних даних: множина місць допустимого розміщення вузлів мережі, кількості кластерів; множини найкращих місць розміщення вузлів $w^o = \varnothing$; значення поточної ітерації алгоритму i:=0.

2. Змінити значення поточної ітерації алгоритму $i := i + 1$ і згенерувати початкову множину центроїдів для елементів мережі w_0.

3. Для центроїдів визначити відстані до кожного з комп'ютерів мережі і сформувати кластери.

4. Визначити нові центроїди (місця розміщення вузлів мережі) за середнім значенням координат усіх точок кластерів (комп'ютерів).

5. Перевірка умов закінчення роботи алгоритму: якщо $w_i = w_{i-1}$, перехід до кроку 3, в іншому випадку перехід до кроку 6.

6. Визначити найкращий варіант побудови мережі w^o шляхом визначення для кожної підмножини комп'ютерів (кластера) точки, найближчої до центроїду (вузла мережі).

7. Закінчення роботи алгоритму: отримано варіант побудови мережі w^o з мінімальним з розглянутих показником значенням витрат $\Delta C(w^o)$.

1.5 Постановка мети та задач дослідження

За результатами огляду й аналізу сучасно стану проблеми підтримки прийняття рішень з реінжинірингу корпоративних комп’ютерних мереж встановлено, що ефективність мереж багато в чому визначається способом їх структурної організації. Корпоративні мережі можуть бути організовані на різних наборах елементів (комп’ютерів, свічів, концентраторів, маршрутизаторів), різних архітектурах та типах каналів зв'язку між елементами. При цьому вартісні та функціональні характеристики багато в чому визначаються їхньою територіальною чи просторовою організацією.

У процесі реінжинірингу виникає необхідність розв’язання комплексу задач, що об’єднані у проблему структурно-функціонально-топологічної оптимізації. У ній рішення щодо кількості елементів, розподілу функцій між ними, технології їх реалізації та взаємозв'язки елементів мережі приймаються виходячи з їх просторового розташування. Це обумовлює актуальність задач оптимізації топологічних структур мереж. Для розв’язання задач реінжинірингу топологічних структур корпоративних комп’ютерних мереж до теперішнього часу розроблено множину методів, що розрізняються за точність та часовою складністю. Виникають задачі дослідження методів оптимізації топологічних структур з метою їх удосконалення і вибору найкращого з них, виходячи з необхідної точності розв’язання задачі та наявних обчислювальних ресурсів.

Метою роботи є підвищення ефективності систем підтримки прийняття проектних рішень за рахунок удосконалення методу оптимізації топологічних структур корпоративних комп’ютерних мереж.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести аналіз сучасного стану проблеми підтримки прийняття проектних рішень з реінжинірингу корпоративних комп’ютерних мереж;

– сформулювати постановку задачі структурно-топологічної оптимізації централізованих трирівневих мереж;

– удосконалити обраний метод оптимізації топологічних структур корпоративних мереж;

– програмно реалізувати метод оптимізації мереж та провести дослідження його ефективності.

Об'єктом дослідження є корпоративні комп'ютерні мережі.

Предметом дослідження є топологічні структури корпоративних комп'ютерних мереж.

2 ПОСТАНОВКА ТА МЕТОД РОЗВ’ЯЗАННЯ ЗАДАЧІ РЕІНЖИНІРИНГУ КОРПОРАТИВНИХ КОМП’ЮТЕРНИХ МЕРЕЖ

2.1 Постановка задачі оптимізації мережі

Характеристики комплексу завдань:

– даний комплекс задач вирішує проблему побудови топологічної структури під час реінжинірингу корпоративних комп’ютерних мереж;

– до переліку об’єктів під час управління якими вирішують комплекс завдань є сервери та комутатори, що переадресовують запити;

– періодичність запитів – відповідно до використовуваної технології роботи мережі;

– тривалість розв'язання залежить від кількості серверів та комутаторів;

– умовою зупинки вирішення є досягнення максимальної кількості поколінь;

– даний комплекс задач дуже тісно пов’язаний з вирішенням задач в логістичних сферах;

– умови та тимчасові характеристики розв'язання задачі визначено загальним алгоритмом функціонування системи.

Вихідна інформація:

– побудований граф з підписаними вершинами, після закінчення роботи алгоритму;

– логування допоміжної інформації, після генерації наступного покоління (час виконання, найкращий індивід тощо).

Вхідна інформація.

Вузли комп’ютерної мережі можуть розташовуватися лише у місцях розташування комп’ютерів мережі (КМ). Зв'язок між КМ та центральним вузлом (ЦВ) може здійснюватися лише через один вузол. Наприклад, територіальне

розміщення ЦВ та КМ задається їх координатами (табл. 2.1). Координати виражені в міліметрах, масштаб 1: 1000000.

Наведені витрати на 1 км каналу могли становити c_l = 0,88 тис. умовних одиниць (у.о.). У мережі передбачається використання однотипних вузлів з достатньою продуктивністю для своєчасної обробки відповідних потоків запитів. Наведені витрати за кожен вузол могли становити 16,3 тис. у.о. У мережі передбачається використання ЦВ з продуктивністю достатньою для обслуговування всіх запитів, що надходять, з необхідним рівнем якості. Центральний вузол та всі КМ вже були обладнані. Наведені витрати на ЦВ становили 104,6 тис. у.о., на один КМ – 4,9 тис. у.о.

Формально завдання полягає у виборі для заданого набору значень інтервальних вхідних параметрів найкращого варіанта топологічної структури розподіленої мережі та значень її інтервальних вихідних параметрів. З урахуванням цього вартість топологічної структури будемо розраховувати за наступною формулою:

$$C = c_C + c_U \cdot \sum_{i=1}^{n_E} r_{ii} + c_E \cdot n_E + \sum_{i=1}^{n_E} \sum_{j=1}^{n_E} c_{ij} \cdot r_{ij} \rightarrow \min_{r_{ij}} \qquad (2.1)$$

де C – функція пристосованості;

c_C – вартість ЦВ;

n_E – кількість елементів, що входять до складу системи;

c_U – вартість типового вузла мережі;

r_{ij} – елемент матриці суміжності (r_{ij} = 1, якщо пункти i і j мережі безпосередньо пов'язані між собою, r_{ij} = 0 – в іншому випадку);

c_E – вартість типового комп’ютера мережі;

c_{ij} – вартість зв'язку між пунктами мережі i і j .

Таблиця 2.1 – Найменування та координати пунктів комп'ютерної мережі

Номер пункту	Найменування пункту	Координати		Номер пункту	Найменування пункту	Координати	
		X, мм	Y, мм			X, мм	Y, мм
1	ЦВ	205,0	295,0	14	КМ-13	312,0	238,0
2	КМ-1	194,5	317,0	15	КМ-14	109,5	180,0
3	КМ-2	219,0	237,0	16	КМ-15	340,0	308,0
4	КМ-3	259,0	264,0	17	КМ-16	365,0	241,0
5	КМ-4	165,0	245,0	18	КМ-17	372,5	269,0
6	КМ-5	175,0	345,0	19	КМ-18	158,5	141,0
7	КМ-6	114,0	326,0	20	КМ-19	323,0	147,0
8	КМ-7	134,5	267,0	21	КМ-20	83,5	153,0
9	КМ-8	287,0	350,0	22	КМ-21	206,5	92,0
10	КМ-9	201,0	183,0	23	КМ-22	361,5	181,0
11	КМ-10	77,5	313,0	24	КМ-23	232,0	85,0
12	КМ-11	272,0	190,0	25	КМ-24	288,0	93,0
13	КМ-12	146,0	168,0				

2.2 Метод розв’язання задачі

У переважній більшості задач синтезу та оптимізації ККМ як єдиний чи основний критерій виступають витрати на її створення та (або) експлуатацію. Їх можна вважати такими, що складаються з витрат на елементи, вузли, центр і зв'язки між ними. Будемо вважати, що вузли мережі (включаючи центральний) утворюються на основі її елементів. Характер залежності функції витрат при фіксованих значеннях кількості вузлів мережі у діапазоні від 1 до u_{max} може бути представлено одноекстремальною функцією. Пошук мінімуму функції витрат залежно від особливостей задачі може здійснюватися з допомогою стратегій збільшення або зменшення кількості вузлів у мережі.

Описаний вище підхід може успішно застосовуватися до вирішення завдань структурно-топологічної оптимізації у традиційній постановці. У процесі реінжинірингу необхідні додаткові витрати на створення нових та реконструкцію існуючих елементів мережі, її вузлів, центру та зв'язків між ними (рис. 2.1).

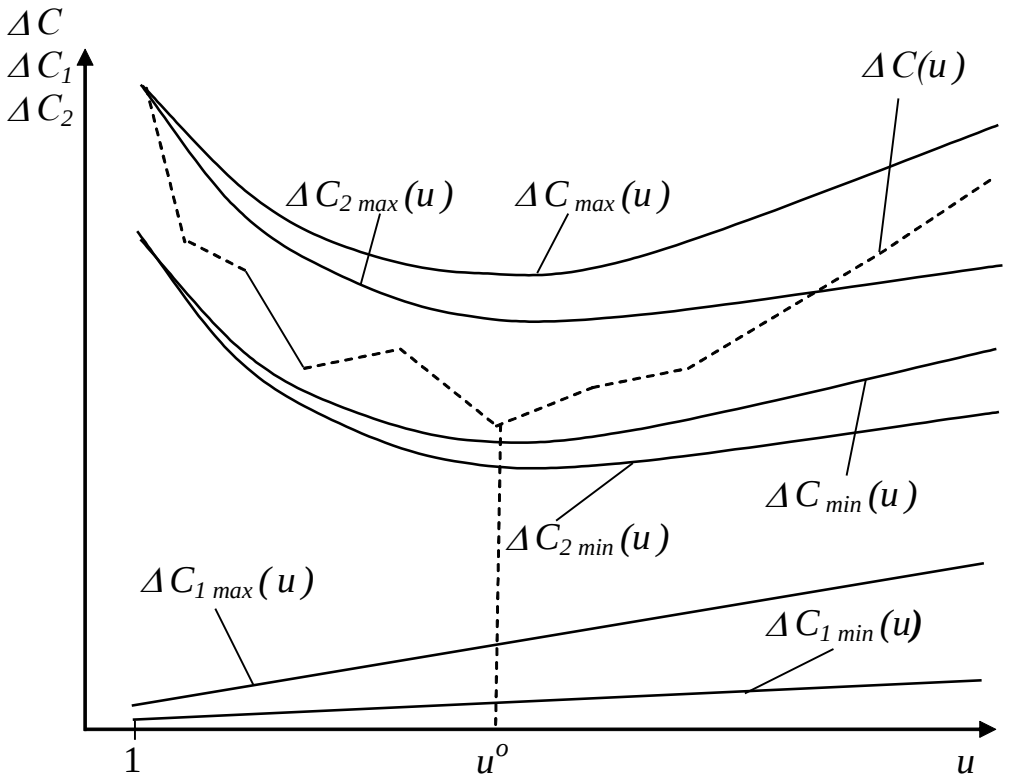

Рисунок 2.1 – Огинаюча локальних мінімумів функції витрат на реінжиніринг

Тому в процесі реінжинірингу топологічних структур корпоративних мереж як цільову функцію витрат доцільно використовувати суму необхідних додаткових витрат ΔC. Огинаюча локальних мінімумів функції витрат для задачі реінжинірингу може бути багатоекстремальною (рис. 2.2).

Для пошуку глобального оптимального рішення задачі синтезу топологічної структури в процесі реінжинірингу суть традиційного методу полягає у визначенні відрізку, який гарантовано містить оптимум функції додаткових витрат. З цією метою знаходиться оптимум функції максимальних додаткових витрат, що дозволяє визначити мінімум витрат за умов повної зміни

топологічної структури мережі. Таке рішення може бути покращено за умов часткової або повної модернізації топологічної структури корпоративної мережі. Для отримання найкращого розв’язку задачі необхідно пройти відрізок, на якому знаходиться оптимальна кількість вузлів мережі, збільшуючи чи зменшуючи значення їх кількість.

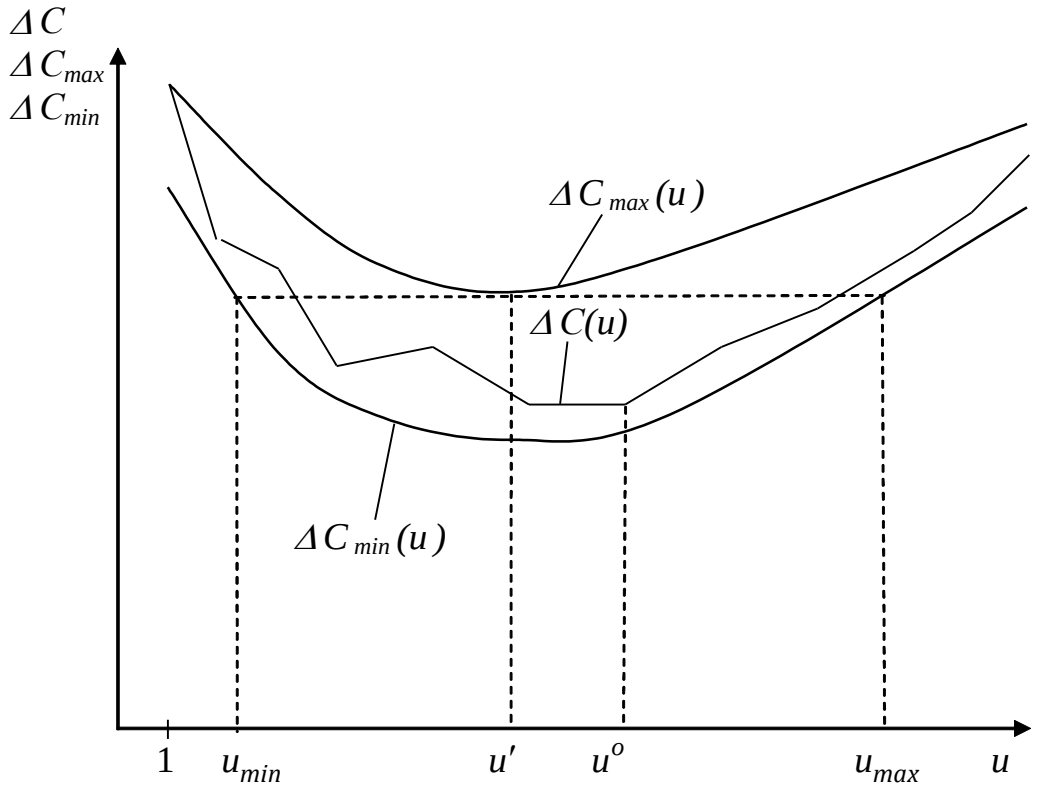

Рисунок 2.2 – Схема пошука мінімуму функції витрат на реінжиніринг

Такий метод передбачає розв'язання трьох задач (пошук мінімуму функції максимальних витрат, визначення відрізка, на якому знаходиться оптимальна кількість вузлів мережі, і пошук мінімуму функції витрат на реінжиніринг.

Для пошуку глобального рішення пропонується модифікувати еволюційний метод з використанням генетичного алгоритму, який дозволить здійснювати пошук відразу у всьому діапазоні допустимої кількості вузлів.

Для вирішення поставленої задачі було обрано реалізувати генетичний алгоритм, тому що GA дозволяє будувати декілька топологій одночасно, що зменшує складність пошуку результату.

При реалізації генетичного алгоритму використано хромосоми, які набувають значення у заданому інтервалі [0,1]. Кожна хромосома представляє множину місць можливого розміщення вузлів. Під геном розуміється код індексу вузла. Як функція пристосованості використовується функція витрат на реінжиніринг корпоративної комп'ютерної мережі (2.2).

Суть операторів, які використовуються при генерації топології, полягає в наступному:

– селектор – оператор що відбирає найкращих індивідуумів, за їх пристосованістю;

– кросовер – оператор схрещування, застосування якого призводить до світу з двох хромосом кількох нових. При одноточковому кросовері випадковим чином вибирається точка розриву, яка розбиває батька на два сегменти, які формують нового нащадка;

– мутатор – оператор, що мутує деяку кількість генів в хромосомі.

Алгоритм методу еволюційного пошуку представлений такими кроками:

1. Завдання вихідних даних: множина місць можливого розміщення вузлів, розміру популяції хромосом N, ймовірності мутації p; номери ітерації (популяції) $i := 0$; кращого поточного значення критерію $\Delta C(w) = \infty$.

2. За допомогою генератора випадкових чисел одержати початкову популяцію;

3. Розрахунок пристосованості населення за критерієм (2.1).

4. Якщо виконується умова закінчення роботи алгоритму перейти до кроку 10, в іншому випадку – до кроку 5.

5. Відбір індивідуумів в зал слави, якщо елітизм увімкнено.

6. Селекція хромосом, за допомогою турнірного відбору.

7. Застосування операції одноточкового схрещування.

8. Застосування операції мутації інверсії біту.

9. Формування нової популяції. Перехід до кроку 3.

10. Закінчення роботи алгоритму, отримано найкраще з розглянутих рішення.

На рисунку 2.3 алгоритм роботи методу представлено в графічному вигляді.

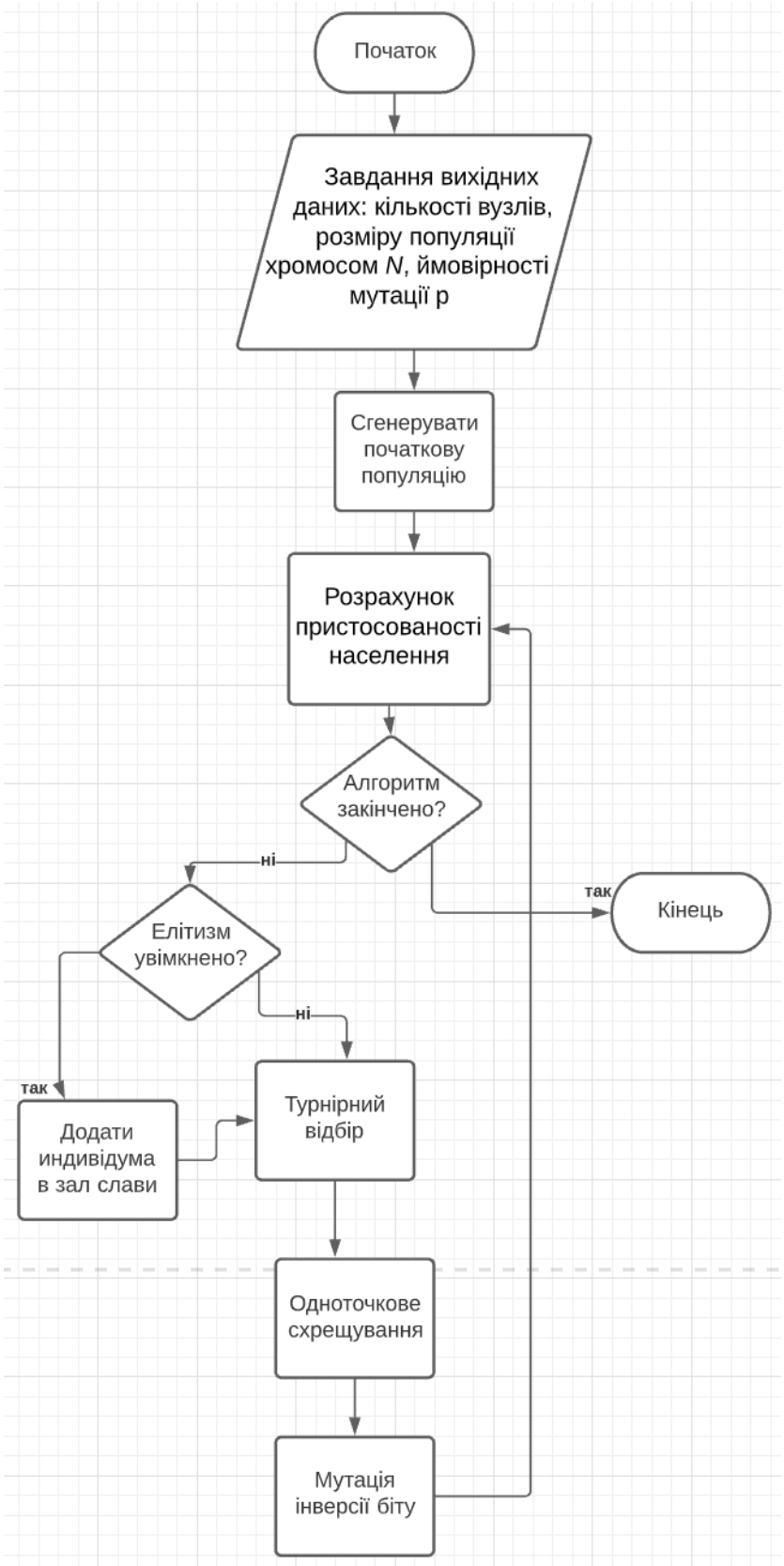

Рисунок 2.3 – Схема генетичного алгоритму еволюційного методу

2.3 Процедури відбору, схрещування та мутації генетичного алгоритму

Відбір – це етап генетичного алгоритму, на якому з популяції вибираються окремі геноми для подальшого розведення (за допомогою оператора кросинговеру).

Загальна процедура відбору може бути реалізована таким чином:

– вибір колеса рулетки – при виборі колеса рулетки ймовірність вибору особи для розведення наступного покоління пропорційна її придатності, чим краще пристосованість, тим вище шанс, що ця особина буде обрана. Вибір індивідуумів можна зобразити як обертання рулетки, яка має стільки кишень, скільки особин у поточному поколінні, з розмірами залежно від їхньої ймовірності. Імовірність вибору індивіда i дорівнює $p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$, де f_i– відповідність i та N – це розмір поточного покоління (в цьому методі одну особину можна відтворити кілька разів);

– вибір рангу – вибір рангу також працює з від'ємними значеннями придатності і в основному використовується, коли особини популяції мають дуже близькі значення придатності (це зазвичай відбувається в кінці прогону алгоритму). Це призводить до того, що кожна особина має майже рівну частку сектору кола (наприклад, у випадку пропорційного відбору придатності). У такому випадку кожна особина, незалежно від того, наскільки вона є подібною до інших, має приблизно однакову ймовірність бути обраною в якості «батька». Це призводить до втрати впливу відбору на краще підібраних особин, що призводить до не дуже гарного вибору «батьків»;

– стійкий відбір – у кожному поколінні відбирається кілька хороших (з високою пристосованістю) хромосом для створення нового потомства. Потім деякі погані (з низькою пристосованістю) хромосоми видаляють, і на їх місце розміщують нове потомство. Решта популяції доживає до нового циклу алгоритму;

– турнірний відбір – це метод вибору особи з набору особин. Для виконання кросовера обирається переможець кожного турніру;

– відбір елітарності – часто для отримання кращих параметрів використовуються стратегії з частковим відтворенням. Однією з них є елітарність, при якій невелика частина найкращих особин з останнього покоління передається (без змін) до наступного;

– вибір Больцмана – безперервна змінна температура контролює швидкість вибору відповідно до попередньо встановленого графіка. Температура починається з високої, а це означає, що вплив на вибір низький. Температуру поступово знижують, що поступово збільшує вплив на вибір, що дозволяє алгоритму звужуватися ближче до кращої частини простору пошуку, зберігаючи відповідний ступінь різноманітності.

У генетичних алгоритмах та еволюційних обчисленнях кросовер, також званий рекомбінацією, є генетичним оператором, який використовується для об’єднання генетичної інформації двох батьків для створення нового потомства:

а) одноточковий кросовер – точку на хромосомах обох батьків вибирають випадковим чином і позначають «точкою кросовера». Біти праворуч від цієї точки міняються місцями між двома батьківськими хромосомами. Це призводить до двох нащадків, кожен з яких несе певну генетичну інформацію від обох батьків;

б) двоточковий і k-точковий кросовер – дві точки кросинговеру вибираються випадковим чином з батьківських хромосом. Біти між двома точками міняються між батьківськими організмами. Двоточковий кросовер еквівалентний виконанню двох одноточкових кросоверів з різними точками перетину. Цю стратегію можна узагальнити до k-точкового перетину для будь-якого натурального числа k, вибираючи k точок перетину.

в) однорідний кросовер – у рівномірному кросовері, як правило, кожен біт вибирається з одного з батьків з однаковою ймовірністю. Іноді використовуються інші пропорції змішування, що призводить до нащадків, які успадковують більше генетичної інформації від одного з батьків, ніж від іншого.

г) кросовер для впорядкованих списків – у деяких генетичних алгоритмах не всі можливі хромосоми є дійсними рішеннями. У деяких випадках можна використовувати спеціалізовані оператори схрещування та мутації, які призначені для уникнення порушення обмежень проблеми.

Використання наведених вище кросинговерів часто призводить до хромосом, які порушують деякі обмеження задачі. Таким чином, генетичні алгоритми, що оптимізують упорядкування заданого списку, вимагають різних операторів кросоверу, які дозволять уникнути генерування недійсних рішень. Відомо досить багато таких кросоверів:

- частково зіставлений кросовер (PMX);
- циклічний кросовер (CX);
- оператор кросовера замовлення (OX1);
- Оператор кросовера на основі порядку (OX2);
- Оператор кросовера на основі позиції (POS);
- оператор перехресного рекомбінації голосування (VR);
- Оператор схрещування змінного положення (AP);
- оператор послідовного конструктивного кросовера (SCX);
- імітований двійковий оператор кросовера (SBX).

Мутація – це генетичний оператор, який використовується для підтримки генетичного різноманіття від одного покоління популяції хромосом генетичного алгоритму до іншого. Це аналог біологічної мутації.

Загальна процедура мутації може бути реалізована у такий спосіб:

– мутація бітового рядка – мутація бітових рядків виникає через перевертання бітів у випадкових позиціях. Імовірність мутації біта дорівнює $\frac{1}{l}$, де l – довжина двійкового вектора. Таким чином, досягається частота мутації *1* на мутацію та індивідуума, обраного для мутації;

– збігання – цей оператор додає випадкове число, взяте з гауссового розподілу із середнім значенням, рівним початковому значенню кожної змінної рішення, що характеризує початковий вектор входу.

3 ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

3.1 Обґрунтування вибору архітектури додатку та мови програмування

MVC розшифровується як Model View Controller. Це архітектура або шаблон проектування програмного забезпечення, який полегшує створення величезних програм. Він не належить до певної мови програмування чи фреймворку, але це концепція, яку ви можете використовувати при створенні будь-яких додатків чи програмного забезпечення на будь-якій мові програмування.

Як працює архітектура MVC на показано на рисунку 3.1.

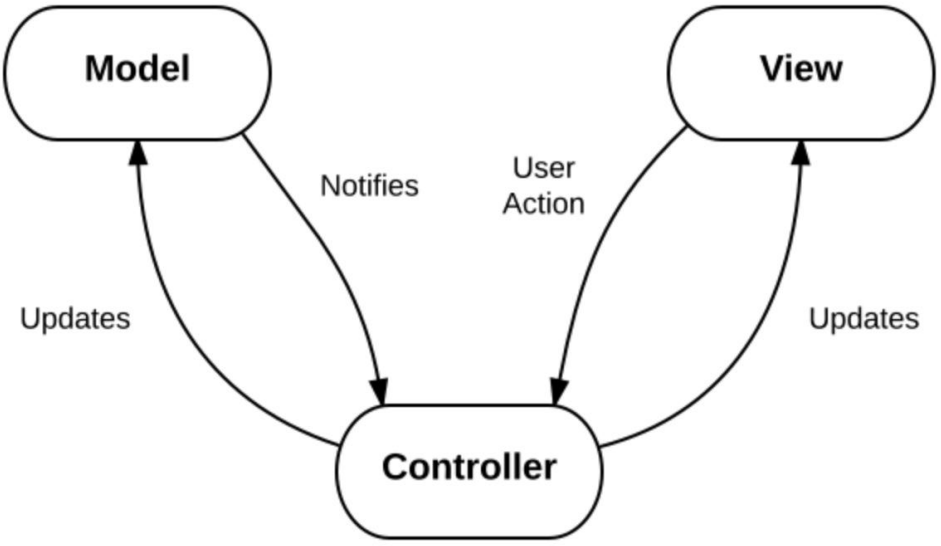

Рисунок 3.1 – Схема роботи додатку за архітектурою MVC

Модель працює безпосередньо з базою даних. Вона не повинна мати справу з інтерфейсом користувача або обробкою даних. У реальному світі ви просто використовуєте модель для отримання, вставки, оновлення та видалення даних із вашої бази даних.

Щоб пояснити це практично, уявіть, що ми створюємо програму для керування завданнями, яка просто дозволить користувачеві організовувати

завдання на основі дати та часу. Це означає, що в нашій базі даних будуть користувачі та завдання для керування. На мові MVC, User і Task є моделями в нашому додатку.

Отже, ми створимо дві моделі в нашій програмі під назвою User і Task. Зауважте, що моделі також мають стосунки між собою. У цьому випадку кожне завдання належить певному користувачеві, і користувач може мати кілька завдань. Таким чином, у нашій моделі користувача буде один метод для отримання всіх завдань користувача, а в моделі Task у нас буде один метод, який отримуватиме користувача.

View – це інтерфейс користувача, на якому наш користувач може виконувати певні дії. Він містить HTML, CSS, JS, XML або будь-яку іншу мову розмітки, яку ми можемо використовувати для створення красивого інтерфейсу користувача. Він також містить код для відображення даних, які він отримує від нашої програми.

Єдині дві речі, які має зробити View, це показувати дані користувачу в інтерфейсі та реагувати на події. Наприклад, що робити, коли користувач натискає кнопку Оновити або Видалити? Відповідь полягає в тому, що користувач повинен бути перенаправлений на форму оновлення або на спливаюче вікно підтвердження видалення.

Controller – це частина, в якій ми обробляємо дані після отримання запиту від View і до оновлення будь-якої бази даних за допомогою нашої моделі.

Уявіть собі, домашня сторінка нашої програми пропонує користувачеві ввести три завдання, якими він хоче керувати в нашому додатку. Після того, як він введе ці три завдання, його буде перенаправлено до форми реєстрації разом із цими трьома завданнями. І після реєстрації ці три завдання з’являться на інформаційній панелі [31].

Переваги архітектури MVC:

– розробка програми стає швидкою;

– легко співпрацювати та працювати кільком розробникам;

– простіше оновлювати програми;

– налагоджувати простіше, оскільки в додатку правильно написано кілька рівнів.

Недоліки архітектури MVC:

– важко зрозуміти архітектуру MVC;

– повинні мати суворі правила щодо методів.

У частині недоліків архітектури не так багато. А недоліки не такі вже й великі, і їх дуже легко ігнорувати в порівнянні з усіма перевагами, які ми отримуємо.

Java – це об'єктно-орієнтована мова програмування загального призначення, яка допомагає створювати програми та програми на будь-якій платформі. Java пропонує низку переваг, які дозволяють вам дотримуватися її.

Ключовими особливостями мови Java є [32]:

– Java проста – будь-яку мову можна вважати простою, якщо її легко вивчити і зрозуміти. Синтаксис Java простий, його легко писати, вивчати, підтримувати та розуміти, код легко налагоджується. Більше того, Java менш складна, ніж такі мови, як C і C++, оскільки багато складних функцій цих мов вилучено з Java;

– Java – це об'єктно-орієнтована мова програмування, яка допомагає нам підвищити гнучкість і можливість повторного використання коду. Використовуючи концепцію ООП, ми можемо легко повторно використовувати об’єкт в інших програмах. Це також допомагає нам підвищити безпеку, об’єднуючи дані та функції в єдиний блок і не даючи до них доступу зовнішньому світу. Це також допомагає організувати більші модулі на менші, щоб їх було легко зрозуміти;

– Java – безпечна мова. Java зменшує загрози безпеці та ризики, уникаючи використання явних покажчиків. Вказівник зберігає адресу пам'яті іншого значення, яке може спричинити несанкціонований доступ до пам'яті;

– Java дешева та економна в обслуговуванні – програми Java дешеві в розробці та обслуговуванні, оскільки ці програми залежать від конкретної

апаратної інфраструктури для виконання. Ми можемо легко виконати їх на будь-якій машині, що зменшує додаткові витрати на обслуговування;

– Java не залежить від платформи, вона пропонує своїм користувачам дуже ефективну користь, забезпечуючи функцію незалежності від платформи, як функцію Write Once Run Anywhere (WORA);

– Java є мовою програмування високого рівня, тобто, зрозуміла людині. Вона подібна до людської мови і має дуже простий і легкий в обслуговуванні синтаксис, схожий на синтаксис мови C++, але більш простий.

– Java має автоматичне керування пам’яттю, яке керується віртуальною машиною Java (JVM).

– Java – це багатопотокова мова, яка в Java може виконуватися одночасно більш ніж одним потоком. Потік — це найменша одиниця процесу. Багатопоточність допомагає нам отримати максимальне використання ЦП. Кілька потоків поділяють спільну область пам’яті і підвищують ефективність і продуктивність програми. Ці потоки працюють незалежно один від одного, не впливаючи один на одного.

– Стабільність – програми на Java більш стабільні в порівнянні з програмами інших мов. Крім того, миттєво виходить нова версія Java з розширеними функціями, що робить її більш стабільною.

– Java має ефективну стратегію розподілу пам'яті, оскільки вона ділить пам'ять в основному на дві частини - область купи і область стеку.

Основними недоліками Java є [32]:

– Java споживає пам'ять і значно повільніше, ніж рідні мови, такі як C або C++. Він також повільний порівняно з іншими мовами, такими як C і C++, оскільки кожен код повинен інтерпретуватися на коді машинного рівня;

– Java має не дуже привабливий вигляд і відчуття графічного інтерфейсу. Хоча в Java є багато конструкторів графічного інтерфейсу для створення графічного інтерфейсу, вони не підходять для створення складного інтерфейсу;

– Java вимагає значного або значного обсягу пам'яті в порівнянні з іншими мовами, такими як C і C++. Під час виконання збирання сміття ефективність пам’яті та продуктивність системи може впливати негативно;

Spring Boot дозволяє легко створювати окремі сервери, які ви можете «просто запустити».

Особливості Spring Boot [33]:

– зменшує час, витрачений на розробку, і підвищує загальну ефективність команди розробників;

– допомагає автоматично налаштувати всі компоненти для виробничої програми Spring;

– полегшує створення та тестування програм на основі Java, забезпечуючи налаштування за замовчуванням для модульних та інтеграційних тестів.

– допомагає уникнути будь-якої ручної роботи з написання шаблонного коду, анотацій та складних конфігурацій XML;

– постачається з вбудованими серверами HTTP, такими як Jetty і Tomcat, для тестування веб-додатків;

– інтеграція Spring Boot з екосистемою Spring, яка включає Spring Data, Spring Security, Spring ORM і Spring JDBC;

– забезпечує багато плагінів, які розробники можуть використовувати для плавної та легкої роботи з вбудованими базами даних і базами даних у пам’яті%

– дозволяє легко підключатися до баз даних і служб черги, таких як Oracle, PostgreSQL, MySQL, MongoDB, Redis, Solr, ElasticSearch, Rabbit MQ, ActiveMQ та багато інших;

– надає підтримку адміністратора – ви можете керувати за допомогою віддаленого доступу до програми;

– полегшує залежність і постачається з вбудованим контейнером сервлетів;

– пропонує гнучкість у налаштуванні конфігурацій XML, Java Beans і транзакцій баз даних;

– забезпечує легкий доступ до інтерфейсу командного рядка, що робить розробку та тестування програм Spring Boot, створених на Java або Groovy, гнучкими.

Найбільшою проблемою, з якою багато розробників стикаються під час використання Spring Boot, є відсутність контролю. Строгий стиль встановлює багато додаткових залежностей (які часто не використовуються), що збільшує розмір файлу розгортання.

Недоліки Spring Boot [33]:

– оновити застарілий код Spring може бути досить складно. Ви можете подолати цю проблему, використовуючи такі інструменти, як Spring Boot CLI (інтерфейс командного рядка), які допоможуть вам перетворити ваш застарілий код;

– якщо ви ніколи раніше не працювали з Spring і хочете дізнатися про проксі, ін’єкції залежностей і програмування AOP, не рекомендується починати зі Spring Boot, оскільки він не охоплює більшість цих деталей;

– ви дійсно повинні розуміти багато основних систем Spring (і трохи історії Spring також), а також деякі розширені теми, щоб змінити та усунути неполадки;

– якщо ви не знайомі з іншими проектами екосистеми Spring, такими як Spring Security, Spring AMQP, Spring Integration тощо), використання їх із Spring Boot змусить вас упустити багато концепцій, які ви зрозуміли б, якби почали використовувати їх самостійно.

Для розробки клієнтської частини було обрано фреймворк Angular. Фреймворк Angular полегшує розробку веб-додатків. Поєднуючи впровадження залежностей, декларативні шаблони, наскрізні інструменти та інтегровані передові методи, він вирішує майже всі проблеми під час створення веб-програми.

Плюси Angular [34]:

– реалізація архітектури MVC – надає цінність фреймворку під час створення клієнтської програми, але також створює основу для інших функцій, таких як прив’язка даних і області видимості. Завдяки архітектурі MVC можна ізолювати логіку програми від рівня інтерфейсу користувача та підтримувати розділення проблем. Контролер отримує всі запити до програми та працює з моделлю, щоб підготувати будь-які дані, необхідні для представлення. Подання використовує дані, підготовлені контролером, і відображає остаточну презентабельну відповідь;

– розширена архітектура дизайну. Деякі з великих веб-додатків містять багато компонентів. Angular спрощує спосіб керування цими компонентами, навіть якщо новий програміст приєднується до проекту після того, як процес розробки вже почався. Архітектура побудована таким чином, що допомагає програмісту легко знаходити та розробляти код;

– модулі – це механізмі, які групують пов’язані директиви, компоненти, канали та служби таким чином, що їх можна об’єднати з іншими модулями для створення програми. Програму на основі Angular можна розглядати як головоломку, де кожен модуль потрібно, щоб мати можливість побачити повну картину. Існує кілька способів додати різні елементи до модуля. Angular вирішує проблему використання глобальної функції, обмежуючи область дії всіх функцій модулем, у якому вона була визначена та використана.

– служби та ін'єкція залежностей (DI) – він розподіляє завдання між різними службами. Служба клієнта не створюватиме залежний об’єкт, скоріше він буде створений і введений за допомогою Angular-інжектора. Інжектор Angular відповідає за створення екземплярів служби та введення їх у такі класи, як компоненти та служби.

– кастомні директиви покращують функціональність HTML і підходять для динамічних програм на стороні клієнта. Усі вони починаються з префікса ng, щоб HTML міг їх ідентифікувати.

– TypeScript – Angular написаний за допомогою TypeScript, який є наднабором для JavaScript. Він повністю відповідає JavaScript, а також допомагає виявляти й усувати типові помилки під час кодування. У той час як невеликі проекти JavaScript не потребують такого покращення, додатки корпоративного масштабу потребують, щоб розробники зробили свій код чистішим і частіше перевіряли його якість.

Мінуси Angular [34]:

– основним недоліком використання Angular є обмежені параметри SEO та погана доступність для сканерів пошукових систем.

– Angular є багатослівним і складним. Часта скарга, яку ви почуєте від розробників Angular, - це багатослівність інструменту. І ця проблема не сильно змінилася з часів AngularJS.

– крута крива навчання. Якщо ви залучаєте нових розробників, які знайомі з JavaScript, для використання нового Angular, їм буде важко в порівнянні з адаптацією React або Vue. Це пояснюється тим, що набір тем і аспектів, які потрібно висвітлити, досить великий.

– у документації CLI бракує деталей. Деякі розробники висловлюють занепокоєння щодо поточного стану документації CLI. Хоча командний рядок дуже корисний для розробників Angular, ви не знайдете достатньо інформації в їхній офіційній документації на GitHub, і вам доведеться витрачати більше часу на вивчення потоків на GitHub, щоб отримати відповіді.

3.2 Опис життєвого циклу додатку

В UML діаграми варіантів використання моделюють поведінку системи та допомагають охопити вимоги системи.

Діаграми варіантів використання описують функції високого рівня та область застосування системи. Ці діаграми також визначають взаємодії між

системою та її акторами. Варіанти використання та дійові особи на діаграмах варіантів використання описують, що робить система і як учасники її використовують, але не те, як система працює всередині.

Діаграми варіантів використання ілюструють і визначають контекст і вимоги або всієї системи, або важливих частин системи. Ви можете змоделювати складну систему за допомогою однієї діаграми варіантів використання або створити багато діаграм варіантів використання для моделювання компонентів системи. Зазвичай ви розробляєте діаграми варіантів використання на ранніх етапах проекту і звертаєтеся до них протягом усього процесу розробки.

Діаграми варіантів використання корисні в таких ситуаціях [35]:

– перш ніж розпочати проект, можна створити діаграми варіантів використання для моделювання бізнесу, щоб усі учасники проекту мали розуміння працівників, клієнтів та діяльності підприємства;

– збираючи вимоги, ви можете створювати діаграми варіантів використання, щоб охопити системні вимоги та представити іншим, що має робити система;

– на етапах аналізу та проектування ви можете використовувати варіанти використання та акторів із ваших діаграм варіантів використання, щоб визначити класи, які потрібні системі;

– на етапі тестування ви можете використовувати діаграми варіантів використання для визначення тестів для системи;

Перелік основних елементів діаграм використання:

– кейси використання: овали в горизонтальній формі, які представляють різні види використання, які може мати користувач;

– актор представляє роль користувача, який взаємодіє з системою, яку ви моделюєте. Користувач може бути користувачем-людиною, організацією, машиною або іншою зовнішньою системою;

– асоціації – це зв’язок між елементами моделі, у вигляді лінії.

Діаграма варіантів використання, яка описує основні розподіл ролей в системі та функції системи, представлена на рисунку 3.2.

Користувач може ввести дані в форму або використовувати стандартне налаштування алгоритму. Після відправлення запиту, клієнт може переглянути топологічно-структурну модель корпоративної комп'ютерної мережі.

У програмній інженерії діаграма класів у уніфікованій мові моделювання (UML) — це тип діаграми статичної структури, яка описує структуру системи, показуючи класи системи, їхні атрибути, операції (або методи) та зв'язки між об'єктами.

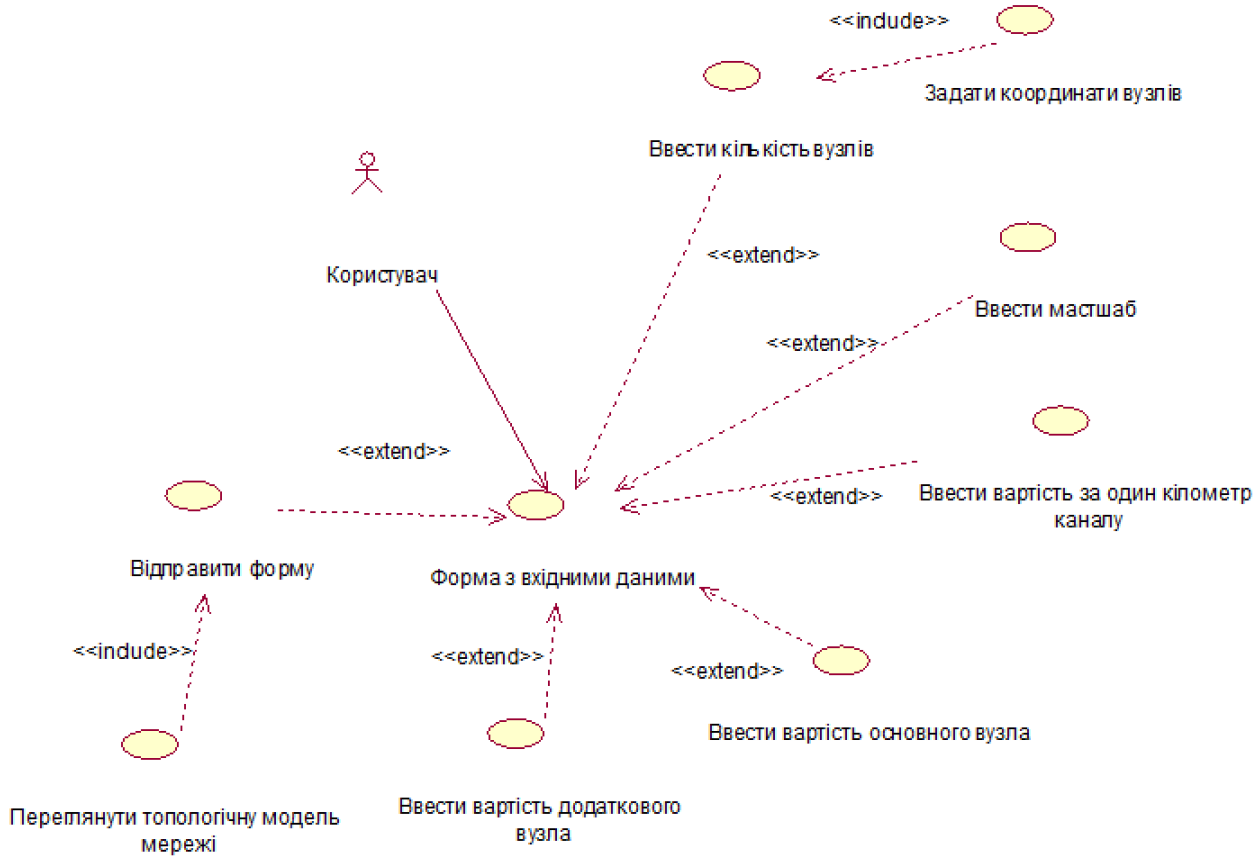

Рисунок 3.2 – Діаграма варіантів використання системи

Призначення діаграм класів [36]:

– показує статичну структуру класифікаторів у системі;

– діаграма надає базове позначення для інших структурних діаграм, передбачених UML;

– корисно для розробників та інших членів команди;

– бізнес-аналітики можуть використовувати діаграми класів для моделювання систем з точки зору бізнесу.

Позначення класу складається з трьох частин:

– назва класу;

– атрибути класу. Атрибути відображаються у другому розділі. Тип атрибута показується після двокрапки. Атрибути відображаються у змінних-членах (членах даних) у коді;

– операції класу (методи). Операції показані в третьому розділі. Це послуги, які надає клас. Тип повернення методу показано після двокрапки в кінці підпису методу. Тип повернення параметрів методу відображається після двокрапки після імені параметра. Операції відображають методи класу в коді

Клас може бути залучений до одного або кількох відносин з іншими класами. Зв’язок може бути одного з таких типів:

– успадкування (або узагальнення). Представляє відносини «є». Назва абстрактного класу показано курсивом. Суцільна лінія з порожнистим наконечником стрілки, що вказує від дитини до батьківського класу;

– проста асоціація. Структурний зв'язок між двома однолітковими класами. Суцільна лінія, що з'єднує два класи;

– агрегація. Особливий тип асоціації. Він представляє «частину» відносин. Суцільна лінія з незаповненим ромбом на кінці асоціації з'єднана з класом композиту;

– композиція. Особливий тип агрегації, коли руйнуються частини, коли руйнується ціле. Суцільна лінія з заповненим ромбом на асоціації, що з'єднується з класом композиту;

– залежність. Існує між двома класами, якщо зміни визначення одного можуть спричинити зміни до іншого (але не навпаки). Пунктирна лінія з відкритою стрілкою.

На рисунку 3.3 представлена діаграма класів, яка описує класи серверної частини додатку.

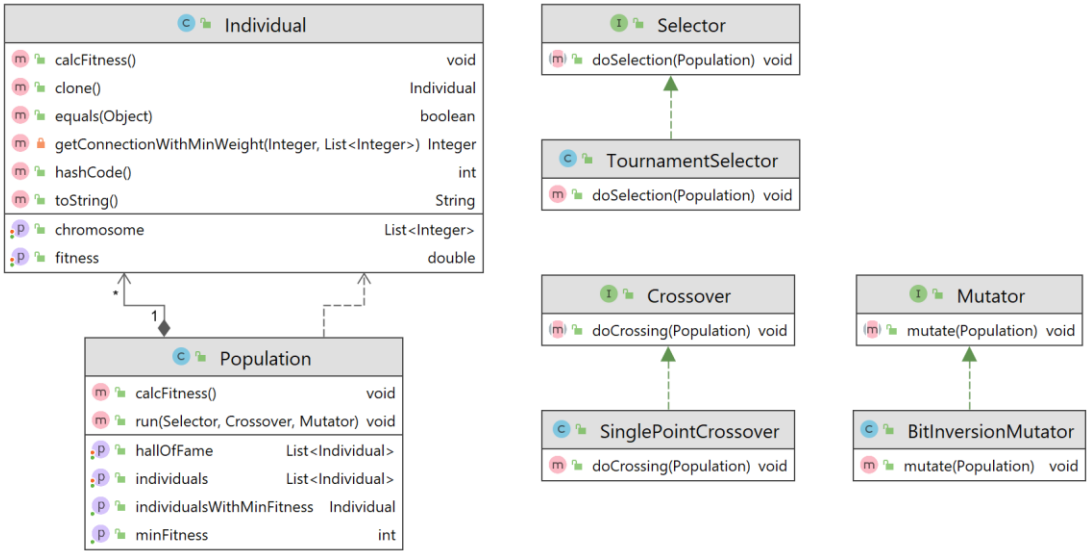

Рисунок 3.3 – Діаграма класів системи

Для підтримки введення даних за допомогою веб інтерфейсу було розроблено програмний код класу-компонента, який наведений в лістингу 3.1. Він містить дві функції. Перша для формування та відправлення запиту на серверну частину додатку, яка, в свою чергу, повертаю данні для побудови топологічну та структурну моделі корпоративної комп'ютерної мережі. Друга функція рисує графі екрані користувача.

```
@Component({
  selector: 'app-main',
  templateUrl: './main.component.html',
})
export class MainComponent {
  public form: FormGroup = new FormGroup({
    chromosomeLength: new FormControl(),
    scale: new FormControl(),
    costPer1KmOfCanal: new FormControl(),
    costPer1Node: new FormControl(),
    mainNodeCost: new FormControl(),
    additionalNodeCost: new FormControl()
  });
  public table: HTMLElement;
  public graph: Map<String, String>;
```

```
public errorMessage: String;
public showErrorMessage = false;

constructor(private mainService: MainService) {
}

public create() {
  this.showErrorMessage = false;
  const body = {
    chromosomeLength: this.form.value['chromosomeLength'],
    scale: this.form.value['scale'],
    costPer1KmOfCanal: this.form.value['costPer1KmOfCanal'],
    costPer1Node: this.form.value['costPer1Node'],
    mainNodeCost: this.form.value['mainNodeCost'],
    additionalNodeCost: this.form.value['additionalNodeCost'],
    coordinates: null
  };
  this.mainService.getGraph(body).subscribe(
    val => {
      if (val) {
        this.graph = val;
        this.drawGraph();
      }
    }, error => {
      if (error.error.message) {
        this.errorMessage = error.error.message;
      } else {
        this.errorMessage = error.error;
      }
      this.showErrorMessage = true;
    }
  )
}

private drawGraph() {
  // @ts-ignore
  const sys = window.sys;
  for (let entry of Object.entries(this.graph)) {
    const key = entry[0];
    const value = entry[1];
    sys.addEdge(value, key);
  }
}

public onChromosomeLengthChange() {
  const tabledata = [];
  for (let i = 0; i < this.form.value['chromosomeLength']; i++) {
    tabledata.push({number: 1 + i});
  }

  // @ts-ignore
  this.table = new Tabulator("#table", {
    data: tabledata,           //load row data from array
```

```
      layout: "fitColumns",      //fit columns to width of table
      responsiveLayout: "hide",  //hide columns that dont fit on
the table
      columns: [                 //define the table columns
        {title: "№", field: "number", editor: false},
        {title: "X, mm", field: "x", editor: "number"},
        {title: "Y, mm", field: "y", editor: "number"}
      ],
    });
  }}
```

Лістинг 3.1 – Програмний код класу MainComponent

Розроблено HTML код головної сторінки, яка містить вище описані форми для введення вхідних даних (рис. 3.4), в результаті отримали екранні форму головної сторінки (рис. 3.5 – 3.7).

```
<section id="home" class="home">
  <div style="...">
    <form [formGroup]="form" (ngSubmit)="create()" class="form-signin col-lg-4" autocomplete="off">
      <input type="number" id="inputChromosomeLength" class="form-control" placeholder="Chromosome Length"
             formControlName="chromosomeLength"
             required="required" (change)="onChromosomeLengthChange()">
      <input type="number" id="inputScale" class="form-control" placeholder="Scale" formControlName="scale"
             required="required">
      <input type="number" id="inputCostPer1KmOfCanal" class="form-control" placeholder="Cost Per 1 Km Of Canal"
             formControlName="costPer1KmOfCanal"
             required="required">
      <input type="number" id="inputCostPer1Node" class="form-control" placeholder="Cost Per 1 Node"
             formControlName="costPer1Node"
             required="required">
      <input type="number" id="inputMainNodeCost" class="form-control" placeholder="Main Node Cost"
             formControlName="mainNodeCost"
             required="required">
      <input type="number" id="inputAdditionalNodeCost" class="form-control" placeholder="Additional Node Cost"
             formControlName="additionalNodeCost"
             required="required">
      <button class="btn btn-lg btn-primary btn-block" type="submit">Submit</button>
      <p *ngIf="showErrorMessage">{{errorMessage}}</p>
    </form>
  </div>
</section>
```

Рисунок 3.4 – HTML код головної сторінки

На рисунку 3.5 наведена екранна форма вводу вхідних даних. Всі поля є не обов’язковими, якщо користувач не введе значення в поле, програма почне використовувати значення за замовчуванням. Поле «Chromosome Length» –

кількість генів у хромосоми, в нашій задачі кількість можливих місць розташувань вузлів. Поле «Cost Per 1 Km Of Canal» – наведені витрати на 1 км каналу передачі даних. Поле «Cost Per 1 Node» – наведені витрати за кожен вузол. Поле «Main Node Cost» – наведені витрати на розташування центрального вузла. Поле «Additional Node Cost» – наведені витрати на розташування ПК.

Рисунок 3.5 – Екранна форма вводу вхідних даних

На рисунку 3.6 наведена екранна форма вводу координат можливих місць розташувань вузлів. В першому стовбці вказаний порядковий номер місця, де можливо розташувати вузол. В другому стовбці користувач вказує координату розташування на осі абсцис в міліметрах. В третьому стовбці користувач вказує координату розташування на осі ординат в міліметрах. Кількість місць розташувань вузлів задається введеням значення в поле Chromosome Length екранної форми, що наведена на рисунку 3.5.

Після відправки та обробітки HTTP запиту користувач побачить граф, що наведено на рисунку 3.7. Відправити запит можна натиснувши на кнопку «Submit».

№	X, mm	Y, mm
1		
2		
3		
4		
5		

Рисунок 3.6 – Екранна форма вводу координат можливих місць розташувань вузлів

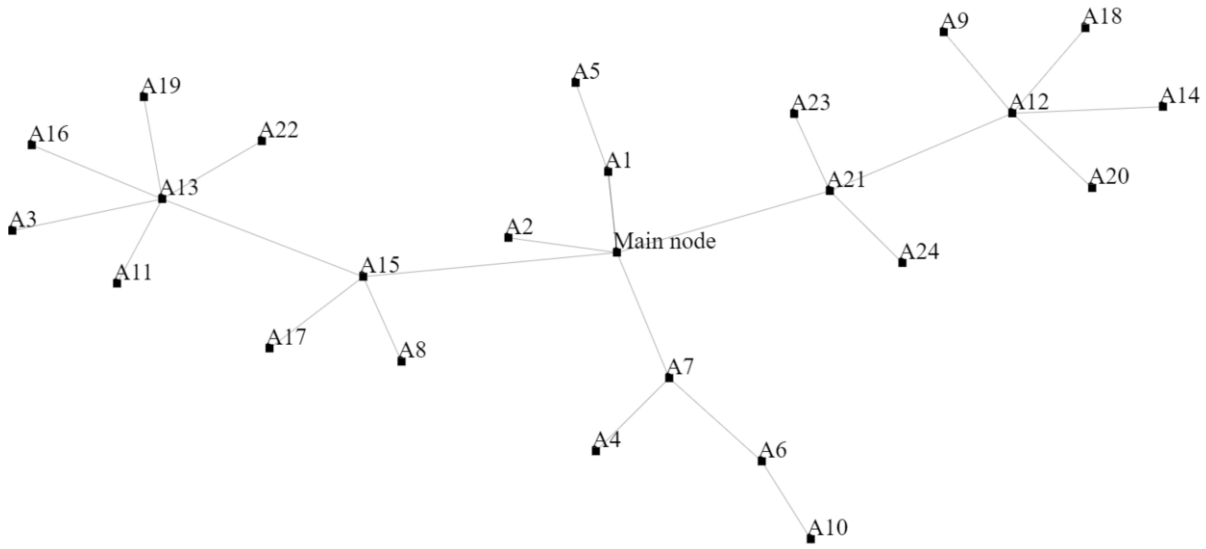

Рисунок 3.7 – Екранна форма виводу результатів оптимізації

3.3 Результати експериментів

В першому експерименті, для порівняння методу еволюційного пошуку та модифікованого методу еволюційного пошуку, було вирішено задачу описану нижче.

Вузли комп’ютерної мережі можуть розташовуватися лише у місцях розташування ПК. Зв'язок між ПК та центральним вузлом (ЦВ) може

здійснюватися лише через один вузол. Територіальне розміщення ЦВ та ПК задається їх координатами (табл. 3.1). Координати виражені в міліметрах, масштаб 1: 1000000.

Таблиця 3.1 – Найменування та координати пунктів комп'ютерної мережі

Номер пункту	Найменування пункту	Координати		Номер пункту	Найменування пункту	Координати	
		X, мм	Y, мм			X, мм	Y, мм
1	ЦВ	205,0	295,0	14	ПК-13	312,0	238,0
2	ПК-1	194,5	317,0	15	ПК-14	109,5	180,0
3	ПК-2	219,0	237,0	16	ПК-15	340,0	308,0
4	ПК-3	259,0	264,0	17	ПК-16	365,0	241,0
5	ПК-4	165,0	245,0	18	ПК-17	372,5	269,0
6	ПК-5	175,0	345,0	19	ПК-18	158,5	141,0
7	ПК-6	114,0	326,0	20	ПК-19	323,0	147,0
8	ПК-7	134,5	267,0	21	ПК-20	83,5	153,0
9	ПК-8	287,0	350,0	22	ПК-21	206,5	92,0
10	ПК-9	201,0	183,0	23	ПК-22	361,5	181,0
11	ПК-10	77,5	313,0	24	ПК-23	232,0	85,0
12	ПК-11	272,0	190,0	25	ПК-24	288,0	93,0
13	ПК-12	146,0	168,0				

Наведені витрати на 1 км каналу могли становити c_l = 0,88 ± 0,1 тис. умовних одиниць (у.о.). У мережі передбачається використання однотипних вузлів з достатньою продуктивністю для своєчасної обробки відповідних потоків запитів. Наведені витрати за кожен вузол могли становити = 16,3 ± 3,8 тис. у.о. У мережі передбачається використання ЦВ з продуктивністю достатньою для обслуговування всіх запитів, що надходять, з необхідним рівнем якості. Центральний вузол та всі ПК вже були обладнані. Наведені витрати на ЦВ становили 104,6 тис. у.о., але в одного ПК – 4,9 тис. у.о.

Всі експерименти будемо проводити на персональному комп’ютері з наступними характеристиками:

- процесор - Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz;
- кількість оперативної пам’яті – 24 Гб;
- операційна система – Windows 10.

В першому експерименті, для методу еволюційного пошуку було використано такі налаштування:

- елітизм – вимкнено;
- довжина хромосоми – 25;
- розмір популяції – 500;
- ймовірність схрещування – 90%;
- ймовірність мутації – 10%;
- ймовірність мутації гену – 0.04%;
- максимальна кількість епох – 50;
- максимальна кількість повторів мінімальної пристосованості підряд у різних епохах – 4;
- масштаб – 1:1000000;
- коефіцієнт для конвертації дистанції в км – 0.000001;
- витрати на 1 км каналу – 0.88;
- витрати за кожен вузол – 16.3;
- витрати на розташування центрального вузла – 104.6;
- витрати на розташування ПК – 4.9;
- максимальна кількість вузлів від 1 до 25.

Для модифікованого методу еволюційного пошуку було використано такі налаштування:

- елітизм – вимкнено;
- довжина хромосоми – 25;
- розмір популяції – 500;
- ймовірність схрещування – 90%;

- ймовірність мутації – 10%;
- ймовірність мутації гену – 0.04%;
- максимальна кількість епох – 50;
- максимальна кількість повторів мінімальної пристосованості підряд у різних епохах – 4;
- масштаб – 1:1000000;
- коефіцієнт для конвертації дистанції в км – 0.000001;
- витрати на 1 км каналу – 0.88;
- витрати за кожен вузол – 16.3;
- витрати на розташування центрального вузла – 104.6;
- витрати на розташування ПК – 4.9;
- максимальна кількість вузлів – вимкнено.

В результаті експерименту отримано результати наведені в таблиці 3.2.

Таблиця 3.2 – Результати першого експерименту

Найменування методу	Витрати на мережу, у.о.	Час розв’язання, с
Метод еволюційного пошуку	1767,1	2,45
Модифікований метод еволюційного пошуку	1767,1	0,22

Аналізуючи отримані результати видно, що модифікований метод еволюційного пошуку швидше в 11 разів. Модифікований метод еволюційного пошуку налаштовувався під конкретну задачу, тому було прийнято рішення збільшити кількість можливих місць розташувань вузлів до 50 та за допомогою генератора випадкових чисел згенерувати їх координат в діапазоні від 80,0 до 360,0. В результаті отримали координати наведені в таблиці 3.3.

Таблиця 3.3 – Найменування та координати пунктів комп'ютерної мережі

Номер пункту	Найменування пункту	Координати		Номер пункту	Найменування пункту	Координати	
		X, мм	Y, мм			X, мм	Y, мм
1	ЦВ	283.7	271.3	26	ПК-25	197.8	257.2
2	ПК-1	166.4	157.5	27	ПК-26	275.9	168.2
3	ПК-2	266.3	332.9	28	ПК-27	240.0	183.8
4	ПК-3	183.2	157.2	29	ПК-28	324.1	305.6
5	ПК-4	209.8	299.2	30	ПК-29	252.5	184.8
6	ПК-5	337.4	202.2	31	ПК-30	275.2	334.4
7	ПК-6	289.9	188.2	32	ПК-31	134.9	306.5
8	ПК-7	129.6	246.4	33	ПК-32	255.8	209.7
9	ПК-8	138.7	311.2	34	ПК-33	165.5	231.1
10	ПК-9	128.2	244.4	35	ПК-34	257.8	115.3
11	ПК-10	290.3	239.8	36	ПК-35	219.2	87.6
12	ПК-11	242.4	290.7	37	ПК-36	90.2	215.4
13	ПК-12	88.7	180.2	38	ПК-37	263.4	190.8
14	ПК-13	308.9	196.9	39	ПК-38	343.2	187.6
15	ПК-14	352.7	279.7	40	ПК-39	260.9	295.7
16	ПК-15	214.5	161.6	41	ПК-40	330.3	247.6
17	ПК-16	345.9	309.7	42	ПК-41	353.2	101.5
18	ПК-17	258.2	183.3	43	ПК-42	297.0	158.0

Продовження таблиці 3.3

Номер пункту	Найменування пункту	Координати		Номер пункту	Найменування пункту	Координати	
		X, мм	Y, мм			X, мм	Y, мм
19	ПК-18	180.8	201.7	44	ПК-43	331.7	184.6
20	ПК-19	208.0	212.3	45	ПК-44	200.7	173.0
21	ПК-20	211.3	311.6	46	ПК-45	168.2	203.9
22	ПК-21	122.2	313.4	47	ПК-46	107.0	288.6
23	ПК-22	208.8	158.5	48	ПК-47	127.8	305.5
24	ПК-23	134.8	130.1	49	ПК-48	119.1	106.0
25	ПК-24	322.3	216.2	50	ПК-49	87.5	233.5

В другому експерименті, для методу еволюційного пошуку було використано такі налаштування:

- елітизм – вимкнено;
- довжина хромосоми від 25 до 50 з шагом 5;
- розмір популяції – 500;
- ймовірність схрещування – 90%;
- ймовірність мутації – 10%;
- ймовірність мутації гену – 0.04%;
- максимальна кількість епох – 50;
- масштаб – 1:1000000;
- коефіцієнт для конвертації дистанції в км – 0.000001;
- витрати на 1 км каналу – 0.88;
- витрати за кожен вузол – 16.3;
- витрати на розташування центрального вузла – 104.6;

- витрати на розташування ПК – 4.9;
- максимальна кількість вузлів від 1 до 50.

Для модифікованого методу еволюційного пошуку було використано такі налаштування:

- елітизм – вимкнено;
- довжина хромосоми від 25 до 50 з шагом 5;
- розмір популяції – 500;
- ймовірність схрещування – 90%;
- ймовірність мутації – 25%;
- ймовірність мутації гену – 0.04%;
- кількість епох – 150;
- масштаб – 1:1000000;
- коефіцієнт для конвертації дистанції в км – 0.000001;
- витрати на 1 км каналу – 0.88;
- витрати за кожен вузол – 16.3;
- витрати на розташування центрального вузла – 104.6;
- витрати на розташування ПК – 4.9;
- максимальна кількість вузлів – вимкнено.

Для підвищення точності результатів у модифікованого методу еволюційного пошуку, було прийнято рішення видалити одну з умов зупинки алгоритму і залишити лише обмеження за кількість епох.

В результаті другого експерименту отримано результати наведені в таблиці 3.4. Де n – кількість можливих місць розташувань вузлів, C, C_1 – витрати на мережу в у.о., які розраховані за допомогою методу еволюційного пошуку та модифікованого методу еволюційного пошуку відповідно, t, t_1 – час затрачений на розв’язання в мілісекундах, методом еволюційного пошуку та модифікованим методом еволюційного пошуку відповідно, $\delta C = \frac{C - C_1}{C} \cdot 100,\ \delta t = \frac{t - t_1}{t} \cdot 100.$

Таблиця 3.4 – Результати другого експерименту

n	C, у.о.	t, мс	C_1, у.о.	t_1, мс	δC	δt
25	1367,6	6579	1367,6	899	0,00%	86,34%
30	1512,1	8248	1512,1	1045	0,00%	87,33%
35	1687,1	12844	1687,1	1148	0,00%	91,06%
40	1872,6	18235	1872,6	1273	0,00%	93,02%
45	2087,6	23715	2089,8	1632	-0,11%	93,12%
50	2307,9	31642	2305,2	1823	0,12%	94,24%

Залежність витрат від кількості вузлів мережі для методу еволюційного моделювання та його модифікації наведено на рисунку 3.8.

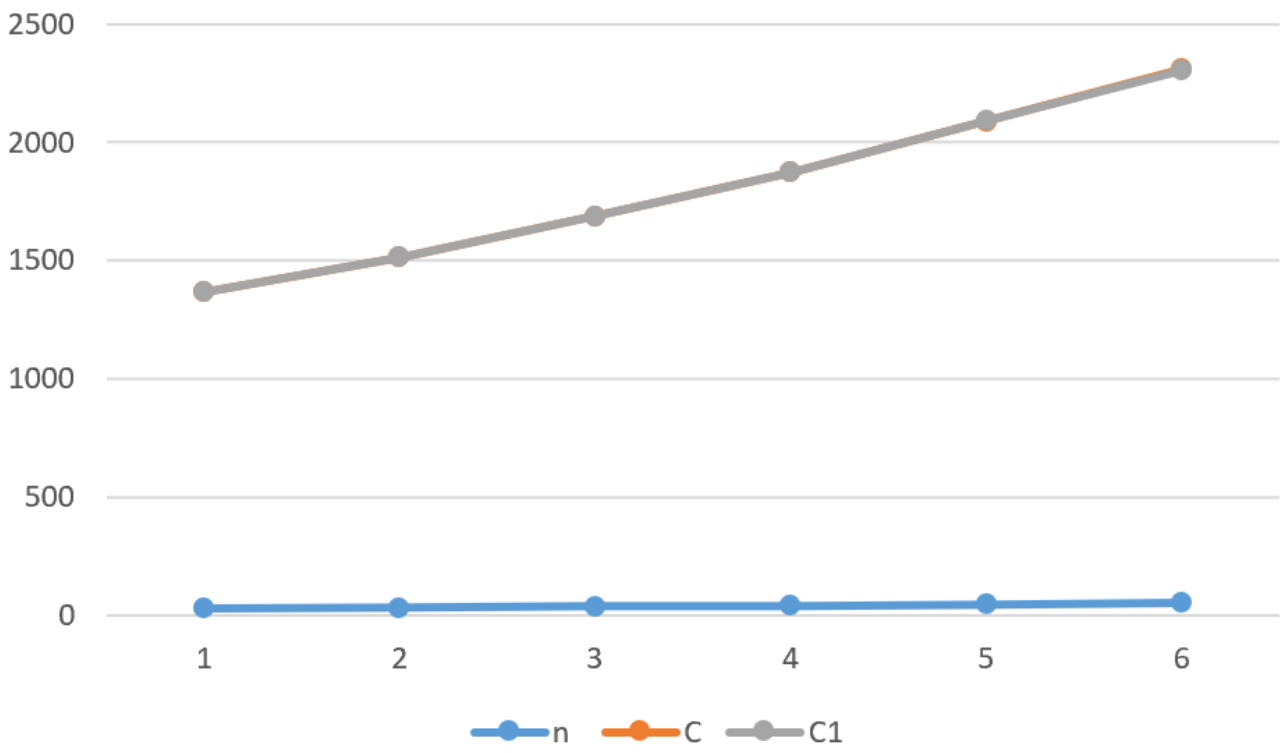

Рисунок 3.8 – Залежність витрат від кількості вузлів мережі

Залежність часу розв’язання задачі від кількості вузлів мережі для методу еволюційного моделювання та його модифікації наведено на рисунку 3.9.

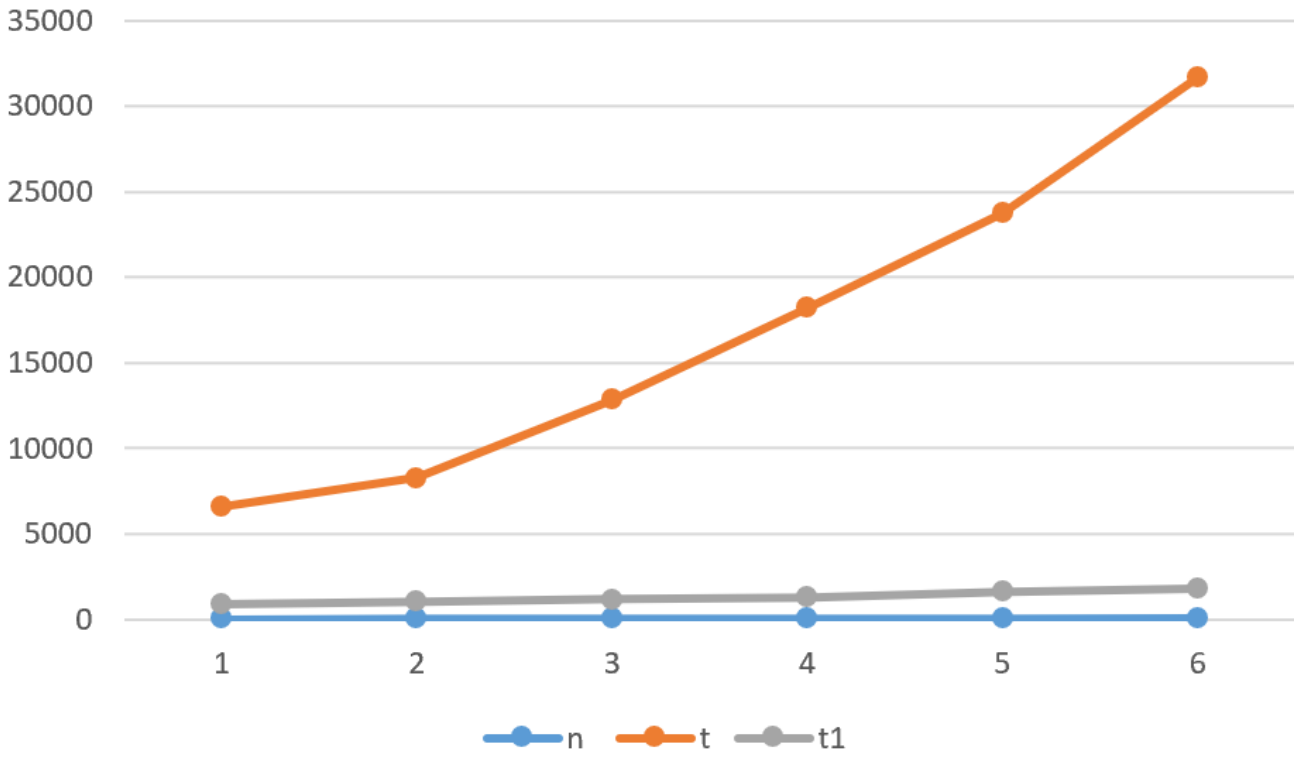

Рисунок 3.9 – Залежність З часу розв’язання задачі від кількості вузлів мережі

Аналізуючи отримані результати видно, що при збільшенні кількості можливих місць розташувань вузлів в 2 рази метод еволюційного пошуку сповільнився майже в 5 раз, тоді як модифікований лише в 2 рази. Порівнюючи витрати побудованих мереж, модифікований метод знайшов мережу, яка на 0,11% дорожча та мережу, яка на 0,12% дешевша, а ніж знайдені за допомогою звичайного методу. Проте порівнюючи час затрачений на виконання пошуку за допомогою модифікованого методу менше на 86,34-94,24%, аніж звичайний метод, що в свою чергу буде грати велику роль при великому n.

ВИСНОВКИ

В результаті виконання дослідження: проведено аналіз сучасного стану проблеми підтримки прийняття рішень з реінжинірингу корпоративних комп’ютерних мереж, розглянуто корпоративні комп’ютерні мережі як об’єкти реінжинірингу, сучасні технології проектування комп’ютерних мереж, моделі та методи підтримки прийняття проектних рішень з оптимізації мереж. За результатами аналізу виявлено множину факторів, що визначають ефективність мереж. Це дозволило формалізувати їх структурний опис як територіально розподілених об’єктів. Обрано схему декомпозиції проблеми структурного синтезу мереж, що включає множину завдань системного проектування, планування розвитку, адаптації та їх реінжинірингу. Визначено склад та схему взаємозв'язку за вхідними та вихідними даними задач системного проектування територіально розподілених мереж, проаналізовано методи їх оптимізації.

Запропоновано удосконалення еволюційного методу спрямованого перебору, побудованого на основі генетичного алгоритму, для оптимізації топологічних структур корпоративних комп’ютерних мереж. Розроблено програмну реалізацію методу для оптимізації централізованих трирівневих мереж.

Отримані результати можуть бути використані для розв’язання задач оптимізації корпоративних комп’ютерних мереж офісів компаній, виробничих підприємств, наукових, навчальних лабораторій тощо. Їх використання дозволить скорочувати час розв’язання задач оптимізації, розв’язувати задачі більшої розмірності і за рахунок цього підвищувати якість проектних рішень і зменшувати витрати на створення й експлуатацію мереж.

За результатами, отриманими в процесі виконання дослідження, підготовлено матеріали для збірнику студентських наукових статей «Автоматизація та приладобудування» («Automation and Development of Electronic Devices» ADED-2021) [37] та доповідь на Всеукраїнську науково-

практичну конференцію здобувачів вищої освіти і молодих учених «Комп’ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» (м. Харків, 2021) [38].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Types of Computer Networks. URL: https://www.geeksforgeeks.org/types-of-computer-networks/ (дата звернення 13.09.2021).

2. Arsal Jahejo. Point to Point Topology | Advantages and Disadvantages. URL: https://computernetworktopology.com/point-to-point-topology/ (дата звернення 13.09.2021).

3. Network Topologies. URL: https://ncertbookspdf.com/computer-network-topology/ (дата звернення 13.09.2021).

4. Sosinsky, Barrie A. Part 1: Network Basics. URL: https://books.google.com.ua/books?id=3DOREqRZejcC&pg=PA16&redir_esc=y#v=onepage&q&f=false (дата звернення 15.09.2021).

5. Н.М. Абдикеев, Т.П. Данько, С.В. Ильдеменов, А.Д. Киселев. Реинжиниринг бизнес-процессов. Полный курс MBA. Учебник 2-е изд., испр. М. : Эксмо, 2007. 592с.

6. Robson M. The Journey to Excellence. Wantage: MRA International, 1986.

7. Харрингтон Д. Оптимизация бизнес-процессов: документирование, анализ, управление, оптимизация. СПб.: АЗБУКА БМикро. 2002. 314 с.

8. Цвиркун А.Д., Акинфиев В.К., Филиппов В.А. Имитационное моделирование в задачах синтеза структуры сложных систем. Оптимизационно-имитационный подход. М.: Наука, 1985. 174 с.

9. Зайченко Ю.П., Гонта Ю.В. Структурная оптимизация сетей ЭВМ. Киев: Техника, 1986. 168 с.

10. Петров Э.Г., Писклакова В.П., Бескоровайный В.В. Территориально распределенные системы обслуживания. Киев: Техника, 1992. 208 с.

11. Бескоровайный В.В. Системологический анализ проблемы структурного синтеза территориально распределенных систем // Автоматизированные системы управления и приборы автоматики. 2002. Вып. 120. С. 29-37.

12. Овезгельдыев А.О., Петров Э.Г., Петров К.Э. Синтез и идентификация моделей многофакторного оценивания и оптимизации. К.: Наук. думка, 2002. 164 с.

13. Денисов А.А., Колесников Д.Н. Теория больших систем управления. Л.: Энергоиздат, 1982. 288 с.

14. Справочник по функционально-стоимостному анализу /А.П.Ковалев, Н.К.Моисеева, В.В.Сысун и др. /Под ред. М.Г.Карпунина, Б.И.Майданчика. М.: Финансы и статистика, 1988. 431 с.

15. Цвиркун А.Д., Акинфиев В.К. Структура многоуровневых и крупномасштабных систем. Синтез и планирование развития. М.: Наука, 1993. 160 с.

16. Построение современных систем автоматизированного проектирования /Жук К.Д., Тимченко А.А., Родионов А.А. и др. К.: Наук. думка, 1983. 248 с.

17. Тимченко А.А. Основи системного проектування та аналізу складних об'єктів: У 2-х кн. Кн. 1. Основи САПР та системного проектування складних об'єктів /За ред. В.І.Бикова. К.: Либідь, 2000. 272 с.

18. Норенков И.П., Маничев В.Б. Основы теории и проектирования САПР. М.: Высш. шк., 1991. 355 с.

19. Хаммер М., Чампи Дж. Реинжиниринг корпорации: Манифест революции в бизнесе: Пер. с англ. С.Пб.: Изд-во С.Пб. ун-та, 1997. 332 с.

20. Бескоровайный В.В., Имангулова З.А., Стадник И.А. Комплекс интерактивного проектирования топологических структур ИВС //Вестник ХГТУ. 1999. № 1(5). С. 33 – 36.

21. Бескоровайный В.В., Имангулова З.А. Алгоритмы оптимизации топологии ИВС на множестве радиально-узловых структур //Радиоэлектроника и информатика. 2000. №2. С. 100 – 104.

22. Бескоровайный В.В., Имангулова З.А. Математическая модель задачи синтеза централизованных информационных сетей //Вестник Харьковского государственного политехнического университета. 2000. Вып. 118. С. 11 – 14.

23. Конспект лекцій з дисципліни «Моделювання систем» для студентів спеціальності 122 – Комп'ютерні науки / Упоряд. В.В. Безкоровайний. Харків: ХНУРЕ. 2019. 174 с.

24. Основы системного анализа и проектирования АСУ / А.А. Павлов, С.Н. Гриша, В.Н. Томашевский [и др.]; Под общ. ред. А.А. Павлова. К.: Выща школа, 1991. 367 с.

25. Методы и модели принятия решений в условиях многокритериальности и неопределенности / Э. Г. Петров, Н. А. Брынза, Л. В. Колесник, О. А. Писклакова. Херсон: Гринь Д. С. 2014. 192 с.

26. Eglese R. W. Simulated annealing: a tool for operational research / R. W. Eglese // European journal of operational research. 1990. Т. 46, №. 3. P. 271–281.

27. Петров Э. Г. Территориально распределенные системы обслуживания / Э. Г. Петров, Пискалкова В. П., Бескоровайный В. В. К.: Техника, 1992. 208 с.

28. Mitchell, Melanie (1996). An Introduction to Genetic Algorithms. Cambridge, MA: MIT Press.

29. Beasley J. E. A genetic algorithm for the set covering problem / J. E. Beasley, P. C. Chu // European Journal of Operational Research. 1996. № 96 (2). P. 392–404.

30. Likas A. The global k-means clustering algorithm / A. Likas, N. Vlassis, J. J. Verbeek // Pattern recognition. 2003. Т. 36, №. 2. P. 451–461.

31. Jithin. What is MVC? Advantages and Disadvantages of MVC. URL: https://www.interserver.net/tips/kb/mvc-advantages-disadvantages-mvc/ (дата звернення 08.11.2021).

32. Advantages and Disadvantages of Java. URL: https://techvidvan.com/tutorials/pros-and-cons-of-java/ (дата звернення 13.11.2021).

33. Pros and Cons of Using Spring Boot .URL: https://scand.com/company/blog/pros-and-cons-of-using-spring-boot/ (дата звернення 17.11.2021).

34. Utkarsh Sidana. What are the Advantages and Disadvantages of Angular? URL: https://www.edureka.co/blog/advantages-and-disadvantages-of-angular/ (дата звернення 20.11.2021).

35. Use-case diagrams. URL: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case (дата звернення 24.11.2021).

36. What is Class Diagram? URL: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/ (дата звернення 01.12.2021).

37. Лясковка Я.І. Еволюційний пошук рішень у технологіях реінжинірингу виробничих комп'ютерних мереж / АВТОМАТИЗАЦІЯ ТА ПРИЛАДОБУДУВАННЯ («Automation and Development of Electronic Devices» ADED-2020) [Електронний ресурс]: збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. Харків : ХНУРЕ, 2021. Вип. 2. С. 131–134;

38. Лясковка Я.І., Безкоровайний В.В. Еволюційний пошук рішень у технологіях реінжинірингу корпоративних комп'ютерних мереж // Комп’ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві. Матеріали всеукраїнської науково-практичної конференції здобувачів вищої освіти і молодих учених. Харків, ХНАДУ, 2021. С. 148–152. URL: https://mf.khadi.kharkov.ua/departments/avtomatizaciji-ta-kompjuterno-integrovanikh-tekhnologii/konferencija-kit/kit-2021/