Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

Галузь знань 12 – Інформаційні технології

Спеціальність 123 –Комп'ютерна інженерія

на тему «Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень»

КвРКІП. 180123.22.1.09 ПЗ

Виконав: студент 2 курсу, група КІ2м-22-1 _____ Шудрик А.О.
Підпис Ініціали, прізвище

Керівник доктор техн. наук, професор _____ Лисенко С.М.
Науковий ступінь, вчене звання Підпис Ініціали, прізвище

До захисту допускаю:
Зав. кафедри КІПС, д.т.н., проф
Т.О. Говорущенко
14 05 2024 р.

Хмельницький, 2024

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Комп'ютерної інженерії та інформаційних систем

Освітній рівень магістр

Галузь знань 12 Інформаційні технології

Спеціальність 123 Комп'ютерна інженерія

Освітня програма Освітньо-наукова програма «Комп'ютерна інженерія та програмування»

ЗАТВЕРДЖУЮ

Зав. кафедри Т.О.Говорущенко

"01" 09 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

Шудрику Андрію Олександровичу

Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Керівник проекту (роботи) Лисенко С.М., д.т.н., професор

Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 01.01.2024 р. № 1

2. Строк подання студентом проекту (роботи) на кафедру 01.05.2024 р.

3. Вихідні дані до проекту (роботи) Завдання на дипломне проектування

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз відомих методів оптимізації IoT із використанням туманних обчислень

Аналіз сфер застосування туманних обчислень для оптимізації IoT інфраструктури

Удосконалення методу оптимізації IoT інфраструктури

Дослідження роботи удосконаленого методу оптимізації IoT інфраструктури із використанням туманних обчислень.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

6. Консультанти розділів кваліфікаційної роботи магістра

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Лисенко С.М., професор кафедри КІІС		
Антиплагіат	Нічепорук А.О., доцент кафедри КІІС		

7. Дата видачі завдання «_01_»_____09_____2023р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) кваліфікаційної роботи магістра	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики КвРМ з керівником	01.09.2023	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.10.2023	виконано
3	Робота над розділом 1 – аналіз відомих моделей, методів за темою; постановка задачі	01.11.2023	виконано
4	Робота над розділом 2 – розробка моделей для вирішення поставленої задачі	01.12.2023	виконано
5	Робота над науковою статтею	01.02.2024	виконано
6	Робота над розділом 3 – розробка методів для вирішення поставленої задачі	15.02.2024	виконано
7	Робота над розділом 4 – проектування та розробка ПЗ для вирішення поставленої задачі, експериментальна частина	01.04.2024	виконано
8	Оформлення пояснювальної записки згідно вимог	18.04.2024	виконано
9	Попередній захист ДРМ	29.04.2024	виконано
10	Захист ДРМ на засіданні ЕК	До 15.05.2024	

Студент _____ Підпис Шудрик А.О. Ініціали, прізвище

Керівник роботи _____ Підпис Лисенко С.М. Ініціали, прізвище

РЕФЕРАТ

Тема Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Автор роботи: Шудрик А.О.

Керівник роботи: Лисенко С.М.

Пояснювальна записка: 76 с., 31 рис., 13 табл., 2 дод., 85 джерел.

IoT, туманні обчислення, ACO, PSO, QoS, DISSECT-CF.

Об’єктом дослідження є процес IoT інфраструктури із застосуванням туманних обчислень.

Предметом дослідження є метод оптимізації IoT інфраструктури із застосуванням туманних обчислень.

Метою кваліфікаційної роботи магістра є оптимізація IoT інфраструктури із застосуванням туманних обчислень.

Для розв’язання поставлених задач використовувалися методи ….

Наукова новизна отриманих результатів:

− набув подальшого розвитку метод оптимізації IoT інфраструктури із застосуванням туманних обчислень, який на відміну від відомих методів здійснює мінімізацію затримки опрацювання даних, отриманих від датчиків та розвантаження мережі хмарного середовища, та дозволяє зменшення затримки відповіді для задач, що вимагають найнижчої затримки, для задоволення QoS.;

− набули подальшого розвитку програмно-технічні засоби оптимізації IoT інфраструктури із застосуванням туманних обчислень.

На основі проведених досліджень удосконалено метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Практична значимість отриманих результатів полягає у зменшенні затримки опрацювання завдань для задоволення QoS.

Було розглянуто концепцію туманних обчислень, її структура, переваги, недоліки та області застосувань.

Було проведено аналіз сучасних методів оптимізації функціонування IoT інфраструктури із застосуванням туманних обчислень.

З метою оптимізації затримки в IoT системі із використанням туманних обчислень було удосконалено метод планування завдань, що базується на алгоритмі ACO.

Із метою реалізації засобів та удосконалення методу планування завдань в IoT інфраструктурі із застосуванням туманних обчислень було змодельвано IoT систему із застосуванням туманних обчислень в середовищі моделювання DISSECT-CF

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

IoT – інтернет речей

QoS – якість обслуговування

ACO – алгоритм мурашиної колонії

PSO – алгоритм рою часток

ОЗП – оперативний запам’ятовуючий пристрій

SDN – програмно визначена мережі

D2D – пристрій - пристрій

PCO – алгоритм контролю і оптимізації процесів

ВСТУП

Якість послуг, ефективність та безпека є важливими цілями в світі обчислень. До появи концепції туманних обчислень хмарні обчислення розглядалися як перспективне застосування завдяки гнучкості та масштабованості. Проте, централізованість хмарних обчислень є перешкодою для обчислень, що чутливі до затримок. Велика кількість IoT систем, що використовують хмарні обчислення схильна до неефективності базових обчислювальних та комунікаційних вимог, до яких відносяться: обізнаність про місцезнаходження, підтримка мобільності та низька затримка. Ці проблеми призвели до впровадження туманних обчислень.

Термін «Туманні обчислення» був запропонований компанією Cisco у 2012 році, оскільки виникла потреба у розширенні можливостей хмарних обчислень в роботі зі значною кількістю пристроїв інтернету речей, великими даними та застосунками які повинні працювати в режимі реального часу із низькою затримкою.

За даними консорціуму OpenFog, периферійні пристрої стикаються з двома значними проблемами з боку всіх хмарних сервісів:

- кількість даних, створених пристроями IoT зростає в геометричній прогресії, що спричинить перевантаження мережі та проблеми з продуктивністю на периферії інфраструктури;
- існують завдання, для вирішення яких недостатньо лише хмарних рішень через такі фактори як продуктивність, безпека, пропускна здатність та надійність.

Актуальність роботи полягає в удосконаленні методу оптимізації IoT інфраструктури із використанням туманних обчислень шляхом зменшення затримки відповіді із метою задоволення QoS.

Мета роботи – оптимізація IoT інфраструктури із застосуванням туманних обчислень.

Поставлена мета досягається розв’язанням таких основних задач:

– дослідити методи оптимізації в IoT інфраструктурі із застосуванням туманних обчислень;

– проаналізувати сучасні методи оптимізації функціонування IoT інфраструктури із застосуванням туманних обчислень;

– дослідити та описати засоби планування завдань в IoT інфраструктурі із застосуванням туманних обчислень;

– удосконалити метод та засоби планування завдань IoT інфраструктурі із застосуванням туманних обчислень;

– реалізувати засоби планування завдань в IoT інфраструктурі із застосуванням туманних обчислень.

Об’єктом дослідження є процес IoT інфраструктури із застосуванням туманних обчислень.

Предметом дослідження є метод оптимізації IoT інфраструктури із застосуванням туманних обчислень.

Наукова новизна отриманих результатів:

1. Набув подальшого розвитку метод оптимізації IoT інфраструктури із застосуванням туманних обчислень, який на відміну від відомих методів здійснює мінімізацію затримки опрацювання даних, отриманих від датчиків та розвантаження мережі хмарного середовища, та дозволяє зменшення затримки відповіді для задач, що вимагають найнижчої затримки, для задоволення QoS.

2. Набули подальшого розвитку програмно-технічні засоби оптимізації оптимізації IoT інфраструктури із застосуванням туманних обчислень.

В результаті виконаного наукового дослідження було удосконалено метод оптимізації IoT інфраструктури із застосуванням туманних обчислень. Удосконалений метод оптимізації IoT інфраструктури із застосуванням туманних обчислень знаходить своє застосування у IoT системах із великою кількістю гетерогенних завдань різних класів. Оптимізація досягається шляхом надання пріоритету опрацювання класам завдань, до яких застосована вимога найменшої затримки.

Для розв’язання поставлених задач використовуються основні положення теорії комп’ютерних мереж та систем, системного аналізу, моделювання, методів аналізу даних, теорії математичної статистики, теорії дискретної математики, теорії еволюційних алгоритмів.

Завданнями роботи є:

- проаналізувати архітектуру туманних обчислень;
- дослідити функціонування туманних обчислень;
- розглянути сфери використання туманних обчислень з метою оптимізації
- проаналізувати характеристики інфраструктури, побудованої із застосуванням концепції туманних обчислень;
- дослідити алгоритми оптимізації;
- удосконалити метод оптимізації IoT інфраструктури.

За темою кваліфікаційної роботи магістра опубліковані тези у матеріалах конференції XXIV Всеукраїнської науково-технічної конференції молодих вчених, аспірантів та студентів «Стан, досягнення та перспективи інформаційних систем і технологій» 18-19 квітня 2024 р., Одеса, Україна [1].

1 ДОСЛІДЖЕННЯ КОНЦЕПЦІЇ ТУМАННИХ ОБЧИСЛЕНЬ

1.1 Огляд концепції туманних обчислень

Із метою дослідження методів оптимізації IoT інфраструктури із застосуванням туманних обчислень було розглянуто концепцію туманних обчислень, її структура, переваги, недоліки та області застосувань.

Туманні обчислення представляють собою архітектуру системного рівня, що допомагає досягти оптимального розподілу мережі, обчислювальних потужностей та зберігання інформації. Туманні обчислення об’єднують у собі переваги хмарних та периферійних обчислень для забезпечення високої якості обслуговування, зниження затримок, забезпечення мобільності та інші функції, які використовуються в сучасних обчислювальних системах. Застосування туманних обчислень забезпечує більшу швидкість опрацювання даних, оскільки обчислювальні потужності знаходяться ближче до пристроїв IoT на географічному рівні [2][3].

Метою використання концепції туманних обчислень є досягнення належного балансу між основними характеристиками та оптимально організувати мережу [4]. Туманні обчислення розширюють можливості хмарних обчислень завдяки перенесенню обчислень на периферійні пристрої або пристрої інтернету речей [5]. Окрім цього, туманні обчислення розширюють хмарні технології управління віртуалізацією та безпекою на стороні периферійних пристроїв за допомогою шарів туманних обчислень [6][7].

Використання концепції туманних обчислень в побудові IoT інфраструктури покращує швидкодію, масштабованість, ефективність та безпеку для важливих застосунків в IoT інфраструктурі [8][9]. Завдання туманних обчислень:

- обчислення;
- зберігання даних;
- надання мережевих сервісів на периферійних пристроях на стороні користувача інфраструктури.

Архітектура мережі IoT із застосуванням туманних обчислень зображено на рисунку 1.2 [10][11].

Парадигма туманних обчислень забезпечує значне зниження затримок та більшу обізнаність вузлів системи в контексті виконання задач та підтримує вертикально ізольовані застосунки, чутливі до затримок за допомогою безперервного мережевого підключення та масштабування [12][13][14]. На рисунку 1.1 наведено характеристики які включають в себе туманні обчислення:

- низька затримка – завдяки близькому розташуванню до кінцевих пристроїв досягається висока швидкість реагування та обробки даних;
- багата та різнорідна підтримка кінцевих пристроїв – досягається завдяки близькому розташуванню обчислювальних вузлів до кінцевих пристроїв;
- багатокористувацьке контрольоване середовище – завдяки високо віртуалізованій розподіленій платформі;
- підтримка мобільності – завдяки прямій комунікації застосунку в туманному середовищі із мобільними пристроями;
- контекстність – пристрої та туманні вузли мають усю інформацію про середовище в якому знаходяться;
- географічна розподіленість – завдяки розподіленості туманне середовище забезпечує високу якість потокових послуг;
- бездротовий доступ - підходить для бездротових сенсорних пристроїв, які потребують розподіленого в часі аналізу та зв’язку;
- підтримка гетерогенності – вузли туманного середовища можуть мати різні форм-фактори та розгортатися в різних розподілених середовищах;
- сумісність пристроїв – функціональна сумісність із пристроями різних виробників в різних галузях;
- аналітика в реальному часі – доступна завдяки близькому розташуванню туманних вузлів до пристроїв IoT;
- промислове використання – має широкий спектр промислових застосувань завдяки обробці даних в реальному часі.

1.1.1 Порівняння туманних обчислень із хмарними обчисленнями

Туманні та хмарні обчислення це дві різні парадигми у сфері опрацювання даних. Обидві концепції надають обчислювальні ресурси та сервіси кінцевим користувачам, хоча відрізняються архітектурно, функціонально та випадками використання [15].

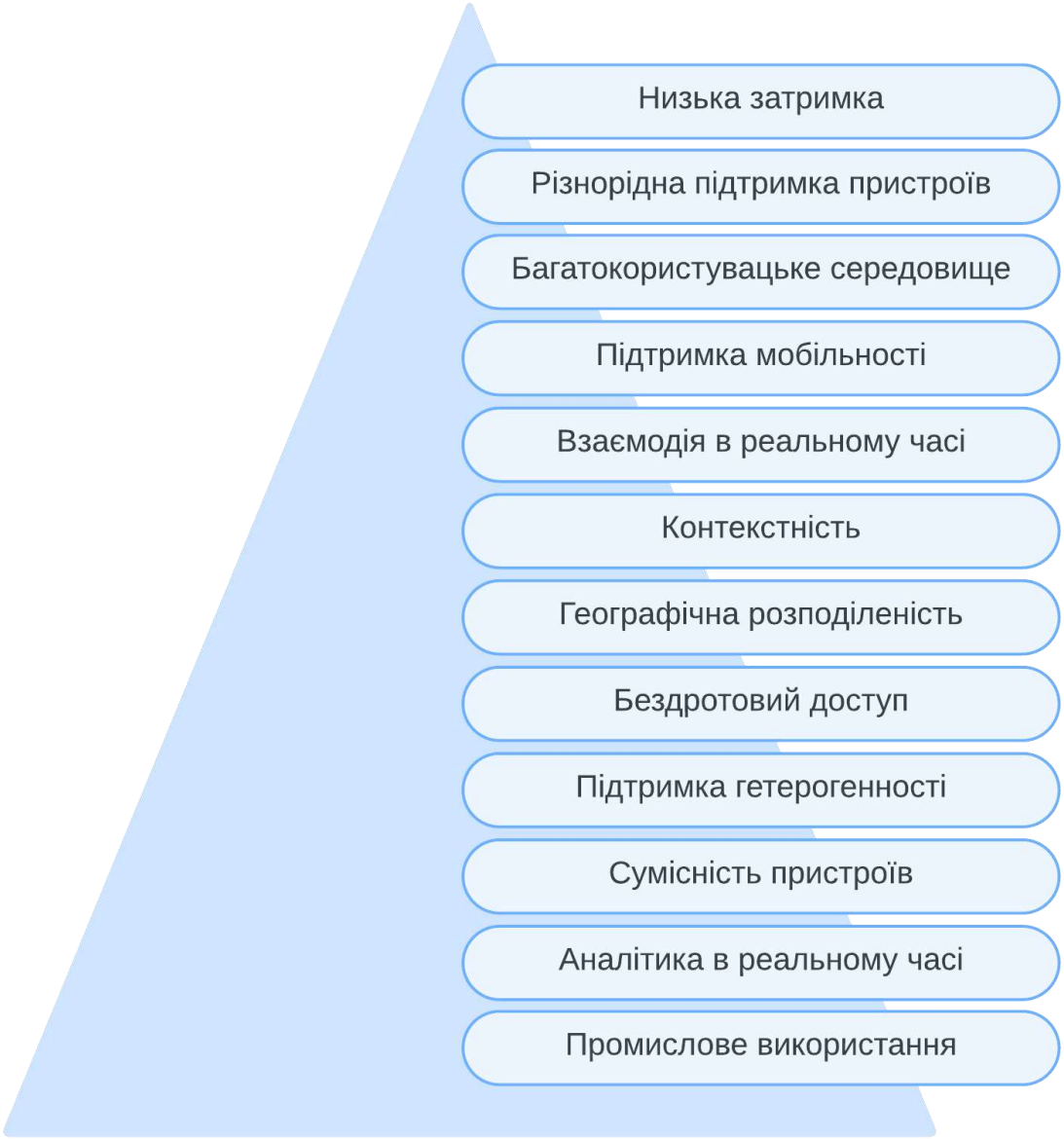


Рисунок 1.1 – Характеристика туманних обчислень

Хмарні обчислення це централізована модель обчислень в якій дані зберігаються та обчислюються у хмарному середовищі. Архітектура концепції хмарних обчислень наведена на рисунку 1.2 [16][17]. Концепція туманних обчислень має децентралізовану архітектуру в якій данні опрацьовуються на стороні IoT пристроїв [18].

Хмарні обчислення характеризуються більшою затримкою відповіді, оскільки географічно дата центр знаходиться на великій відстані від кінцевих IoT пристроїв [19][20]. На противагу цьому туманні обчислення здатні обробляти данні в реальному часі, що робить цю концепцію ефективнішою для використання із застосунками які є чутливими до затримок [20][21][22]

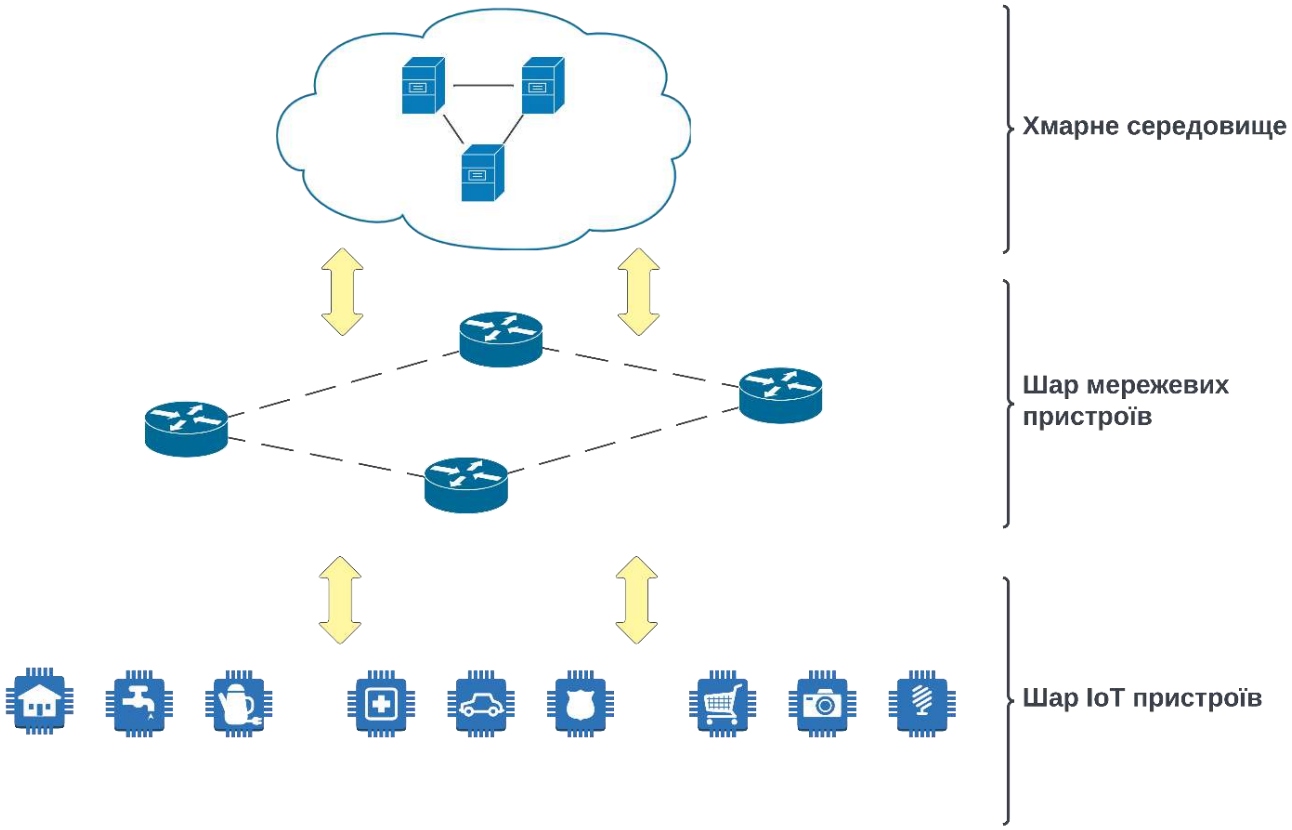


Рисунок 1.2 – Архітектура хмарних обчислень

Системи побудовані на основі хмарних обчислень дуже просто і швидко масштабуються, завдяки чому задовольняють потребу застосунків в обробці та збереженні великої кількості даних [23][24]. Система, що використовує туманні

обчислення є менш масштабованими. Проте, такі системи можуть надавати додаткові обчислювальні потужності та функціонал пристроям, що знаходяться на рівні периферійних обчислень [25].

Хмарні середовища захищені просунутими заходами безпеки, в той час як туманні обчислення сфокусовані на тому, щоб захистити данні на рівні периферійних обчислень [26].

1.1.2 Порівняння концепцій туманних та периферійних обчислень

Ціллю застосування концепції периферійних обчислень є обчислення даних якомога ближче до джерела даних. Периферійні обчислення базуються на опрацюванні даних на рівні IoT пристроїв, що робить їх протилежними до хмарних обчислень.

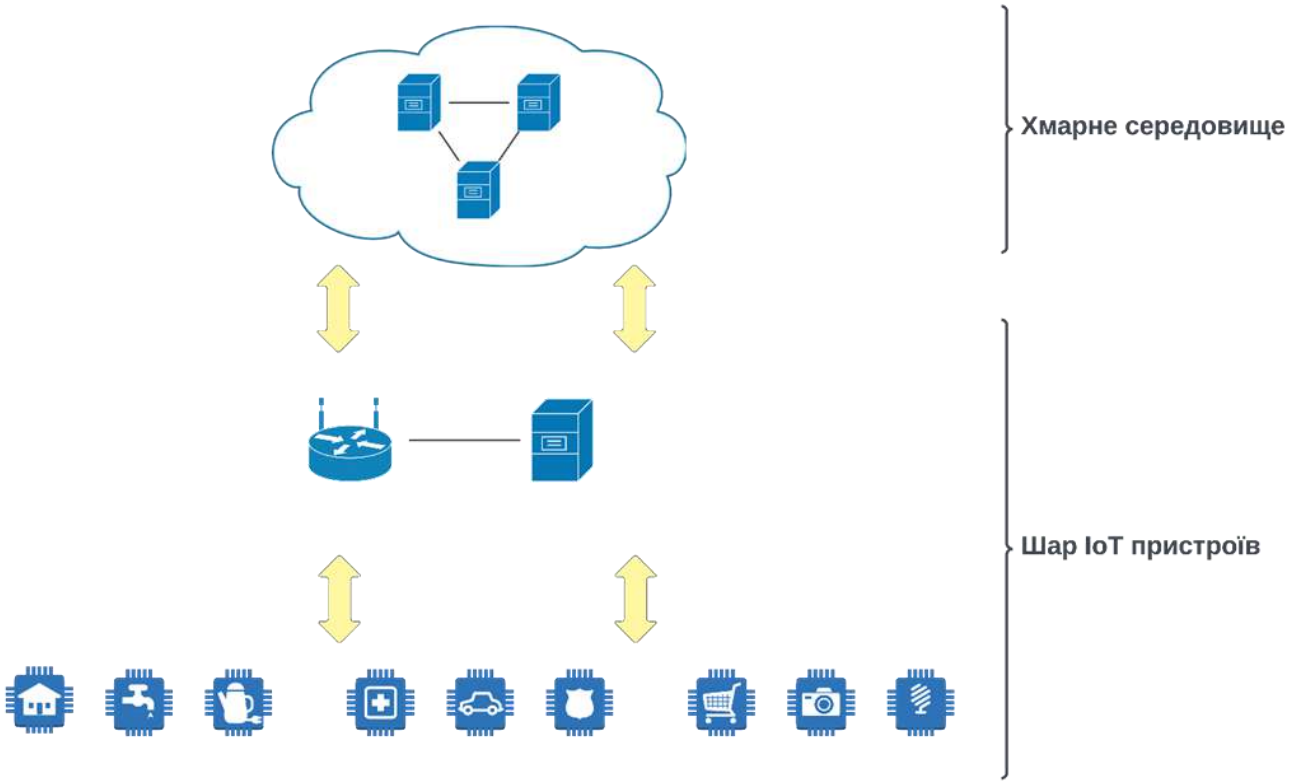


Рисунок 1.3 – Архітектура периферійних обчислень

Метою периферійних обчислень є зменшення кількості даних які передаються до хмарного середовища, що в результаті зменшує затримку в обчисленнях та збільшує загальну швидкодію системи [27][28][29]. Архітектура концепції периферійних обчислень зображена на рисунку 1.3 [30].

1.1.3 Архітектура туманних обчислень

Туманні обчислення поєднують в собі периферійні та хмарні обчислення, використовуючи переваги кожної із концепцій [31]. На рівні периферійних обчислень дані агрегуються, відправляються на обробку на рівень туманних обчислень [31][32].

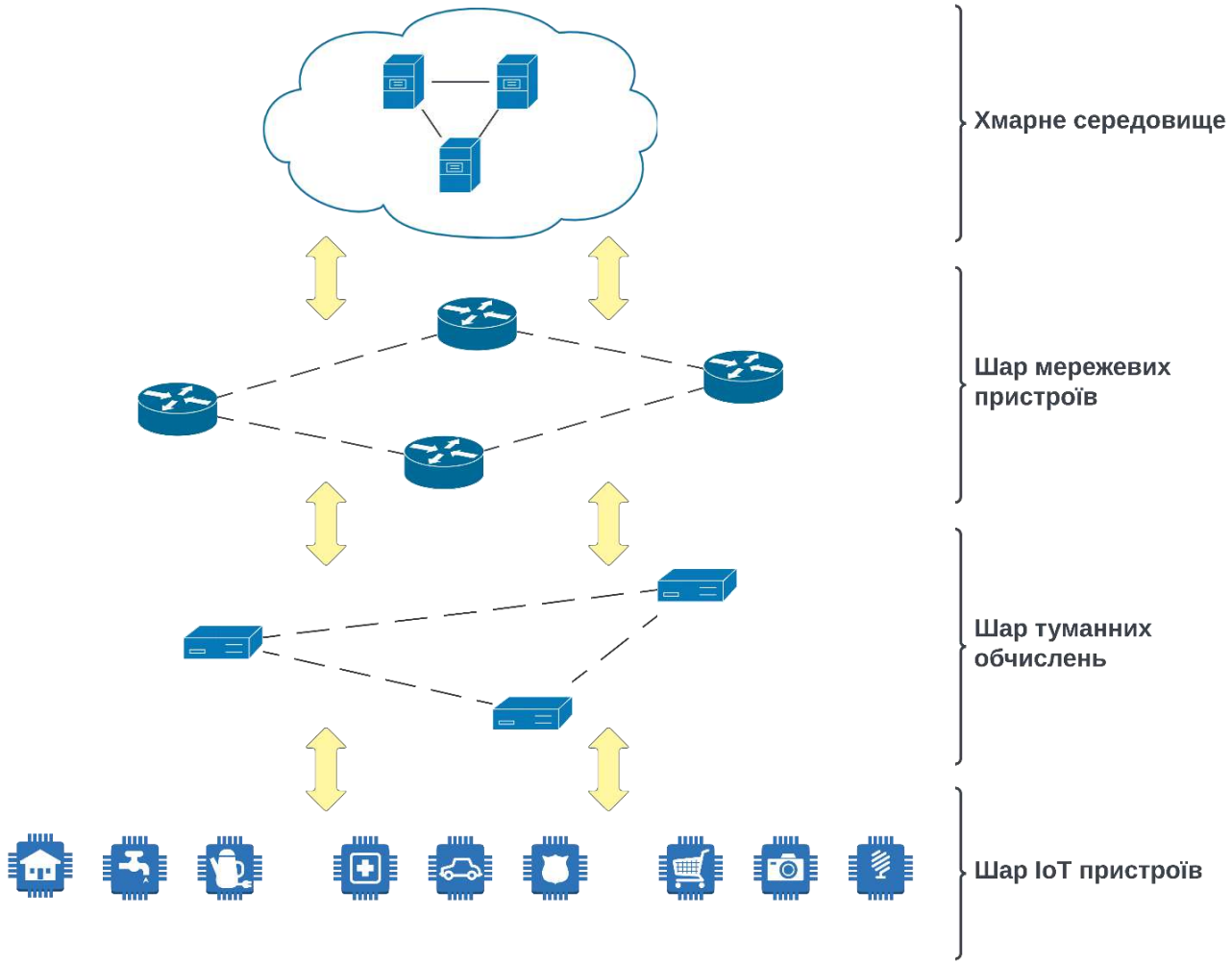


Рисунок 1.4 – Архітектура туманних обчислень

Рівень хмарних обчислень надає додаткові потужності для опрацювання та зберігання даних. Приклад архітектури туманних обчислень наведено на рисунку 1.4 [33].

Слід зазначити, що парадигма туманних обчислень не є назалежною та існує як розширення хмарних та периферійних обчислень [34].

1.2 Туманні вузли

Фундаментальними елементами в архітектурі туманних обчислень є туманні вузли [35]. Туманні вузли – це модульні елементи апаратного та програмного забезпечення, що налаштовані на виконання конкретних задач або класів задач [36].

Туманні вузли є важливими децентралізованими елементами в архітектурі туманних обчислень. Кожен туманний вузол забезпечений власними потужностями для проведення обчислень та зберігання інформації. Обчислювальні потужності можуть бути розподілені на різних рівнях ієрархії в межах рівня туманних обчислень [37][38].

Налаштування туманних вузлів залежить від відстані їхнього розташування відносно центрального сервера, що знаходиться в хмарному середовищі [39].

Вузли які знаходяться ближче до кінцевих пристроїв IoT мають просте апаратне та програмне забезпечення та невеликі потужності. Вони включать в себе різні датчики, приводи та інші пристрої IoT [40].

Вузли що знаходяться ближче до хмарного середовища мають більш складне апаратне та програмне забезпечення із високими обчислювальними потужностями та можливістю концентрувати велику кількість інформації. Це можуть бути такі пристрої як розумні маршрутизатори та міні комп’ютери [41].

Туманні вузли з’єднані між собою у мережу в межах свого шару туманних обчислень та здійснюють підтримку трафіку між вузлами, балансування навантаження та забезпечують відмовостійкість. У таблиці 1.1 наведено приклади

апаратного забезпечення із їх конфігураціями, яке може бути використано для розгортання туманних вузлів [42].

З огляду на таблицю 1.1 можна зробити висновок, що апаратні пристрої можна підбирати за їх характеристиками під певні задачі та тип обчислень [43]. Також можуть бути підібрані інші апаратні пристрої з огляду на ціну, об’єм ОЗП, швидкість Ethernet, Bluetooth в залежності від потреб кінцевого користувача та функцій які повинен виконувати туманний вузол [44][45].

Таблиця 1.1 – Характеристики апаратного забезпечення яке може бути використане як туманний вузол.

Параметри	Cisco Edge and Fog Computing	Lattepanda alpha	RockPi 4	Raspberry pi 3 b+	Tinker board	Beaglebone black
Процесор	Six Cores	Dual Core, fourthread	Hexa Core ARM	Quad Core 64-bit	Quad Core	ARM Cortex-A8
ОЗП	2GB/core	8GB	1GB/2GB/4GB	1GB SRAM	2GB	512 MB DRAM
Швидкість Ethernet	In Gbs	Gb	Gbit LAN	Gigabit (max 300 mbps)	Gigabit	On chip 10/100
Bluetooth	-	4.2	5.0	4.2	4.0	4.1
USB	3.0	3.0	3.0	2.0	2.0	2.0
Вартість(UAH)	>46138	~11534	~1300-2300	~1384	~1845	~1845

Продовження таблиці 1.1 – Характеристики апаратного забезпечення яке може бути використане як туманний вузол.

Сховище даних	HDD	Micro sd	Micro sd	Micro sd	Micro sd	Micro sd
Операційна система	Ubuntu, Redhat, Windows	Ubuntu, Redhat, Windows	Raspbian OS and light weight OS	Raspbian OS and light weight OS	Raspbian OS and light weight OS	Raspbian OS and light weight OS
Мова програмування	Java, Python, Ruby, C++, dart,Scala	Java, Python, Ruby, C++, dart,Scala	Java, Python, Ruby, C++, dart,Scala	Java, Python, Ruby, C++, dart,Scala	Java, Python, Ruby, C++, dart,Scala	Java, Python, Ruby, C++, dart,Scala

1.3 Характеристика параметрів оптимізації IoT інфраструктури з використанням концепції туманних обчислень

Існують випадки у яких IoT інфраструктура, що базується на архітектурі туманних обчислень не потребує додаткової оптимізації [46]. Проте, у певних випадках застосування туманних обчислень стає найбільш оптимальним рішенням завдяки якому застосунок досягає поставлених цілей та повністю задовольняє усі вимоги QoS [47][48].

Характеристики за якими визначається, що IoT інфраструктура може бути покращена завдяки застосуванню туманних обчислень:

- географічна різноманітність – якщо розподілити туманні вузли по усій мережі можливо використати механізм кешування, з метою зберігання даних на стороні периферійних пристроїв щоб забезпечити швидку доставку даних [49];
- пропускна здатність мережі – завантаження великої кількості даних, що була отримана від пристроїв IoT до хмарного середовища може

перенавантажити мережу [50]. За допомогою використання туманних обчислень можна зберегти данні, отримані від різних датчиків та пристроїв, на стороні периферійних пристроїв, та завантажити до хмарного середовища вже опрацьовані данні [8]. Таким чином мережа хмарного середовища розвантажується [51][52];

- надійність – велика кількість IoT інфраструктур розгорнуті навколо застосунків, що виконують критичні завдання та відповідають за безпеку чи життя людей. Тому для таких застосунків важливо продовжувати функціонування навіть при втраті з'єднання із хмарним середовищем [52]. У таких випадках туманні вузли виконують критично важливі обчислення для підтримки роботи застосунку [53];
- багатофункціональні пристрої IoT – часто IoT застосунки переносять задачі обчислення та опрацювання даних на пристрої IoT. Таке використання IoT пристроїв призводить до того, що потреба в з'єднанні із хмарним середовищем повністю відпадає [54]. Однак, у такому випадку кінцеві IoT пристрої потребують великої кількості пам'яті та обчислювальних потужностей, що збільшує вартість датчиків [55];
- безпека – IoT застосунки можуть використовуватись із метою транспортування великої кількості конфіденційних даних до хмарного середовища [56]. У випадку атаки зловмисника на хмарне середовище, конфіденційні дані можуть бути скомпроментовані. У випадку використання туманних обчислень, туманні вузли знаходяться на стороні периферійних пристроїв мережі, тому імовірність компроментування конфіденційних даних значно зменшується. Окрім цього, туманні вузли можуть виконувати роль проксі-серверів, які використовують криптографію для шифрування конфіденційних даних [57]. Така криптографія не може бути використана на стороні периферійних пристроїв як датчики та приводи, оскільки їм не вистачить пам'яті та обчислювальних потужностей [58].

1.4 Потреба у використанні туманних обчислень

Велика кількість даних, що генерується сенсорами та іншими периферійними пристроями та зростання популярності концепції Інтернету Речей(рисунок 1.5) викликала потребу в використанні розподілених обчислень. Передача даних, згенерованих сенсорами, до хмарних середовищ вимагає великої пропускної здатності, тому є неефективною і негативно впливає на роботу системи[59].

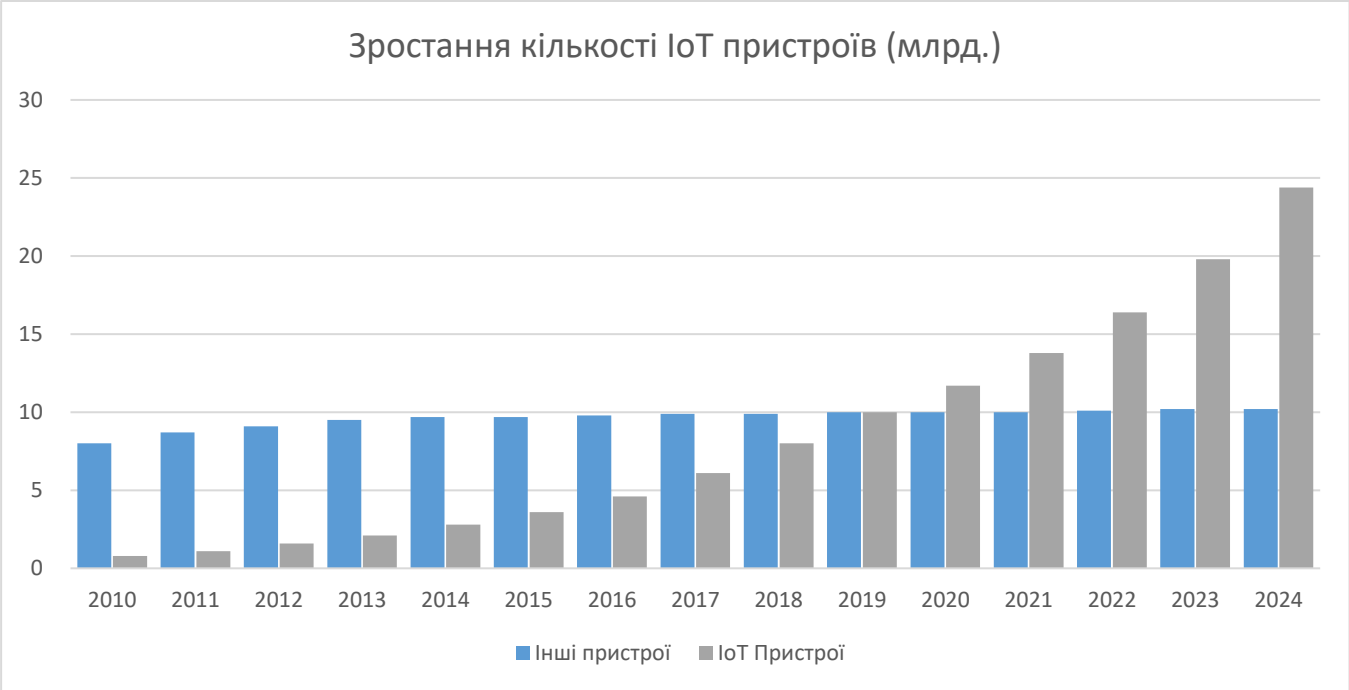


Рисунок 1.5 – Зростання кількості IoT пристроїв відносно інших електронних пристроїв

Причини впровадження концепції туманних обчислень в IoT інфраструктуру:

- розподіл функцій – частина функціоналу хмарного середовища зосереджується на стороні туманних обчислень, функції для різних застосунків стають розподіленими. Такі функції оптимально розподілені по мережі та допомагають виконувати управління IoT інфраструктурою [62];

- розподілена архітектура – туманні обчислення спрямовані на використання обчислювальної потужності периферійних пристроїв, завдяки чому забезпечується ефективне зберігання і обчислення даних [63];
- імерсивний розподіл – на відміну від хмарної архітектури IoT, туманні обчислення є децентралізованими, завдяки цьому гарантується, що ресурси які були отримані та збережені на рівні туманних обчислень будуть доступні усім вузлам мережі [64]. Такий підхід забезпечує гнучкість мережі IoT та простоту інтеграції нових пристроїв;
- низька затримка – використання обчислювальних потужностей та пам'яті пристроїв IoT займає важливу роль у зменшенні затримки відповіді, і тому є вагомою причиною для застосування туманних обчислень. Завдяки забезпеченню низької затримки, туманні обчислення дозволяють впроваджувати системи реального часу такі як: Штучний Інтелект, Віртуальна Реальність, Доповнена Реальність, аналітика потокових даних [65].

1.5 Сфери застосування туманних обчислень

Метою туманних обчислень є децентралізація обчислювальних процесів і забезпечення ефективності вузлів IoT. Туманні обчислення застосовуються на мережевому рівні локальної мережі, тому піддають себе викликам безпеки, довіри та конфіденційності, оскільки кількість пристроїв та даних постійно зростає. Значна кількість проектів відмовилась від використання хмарних технологій через ризики безпеки та конфіденційності [66].

Багато зусиль було вкладено в сферу безпеки туманних обчислень. Один із них це спосіб захисту від розподілених атак на відмову в обслуговуванні (DDoS) у промисловій інфраструктурі інтернету речей [67]. Підхід було успішно реалізовано, покращено обробку даних в реальному часі та обчислювальні можливості промислової інфраструктури інтернету речей [68].

Також, був створений протокол авторизації між хмарними туманними та периферійними пристроями який використовує валідацію ключа сесії [69].

1.5.1 Управління у сфері охорони здоров’я

Перевагою туманної інфраструктури в сфері охорони здоров’я є можливість мати декілька серверів, що дозволяє забезпечувати краще надання медичних послуг. Туманні обчислення забезпечують систему охорони здоров’я такими перевагами як зменшення енергоспоживання, мінімізація затримок та зменшення використання мережевого трафіку. На додачу, аналіз та зберігання даних локально в межах туманного шару покращує безпеку, оскільки важливі та конфіденційні дані залишаються в межах організації [70].

До впровадження туманних обчислень управління охороною здоров'я вже було успішним у хмарних обчисленнях.

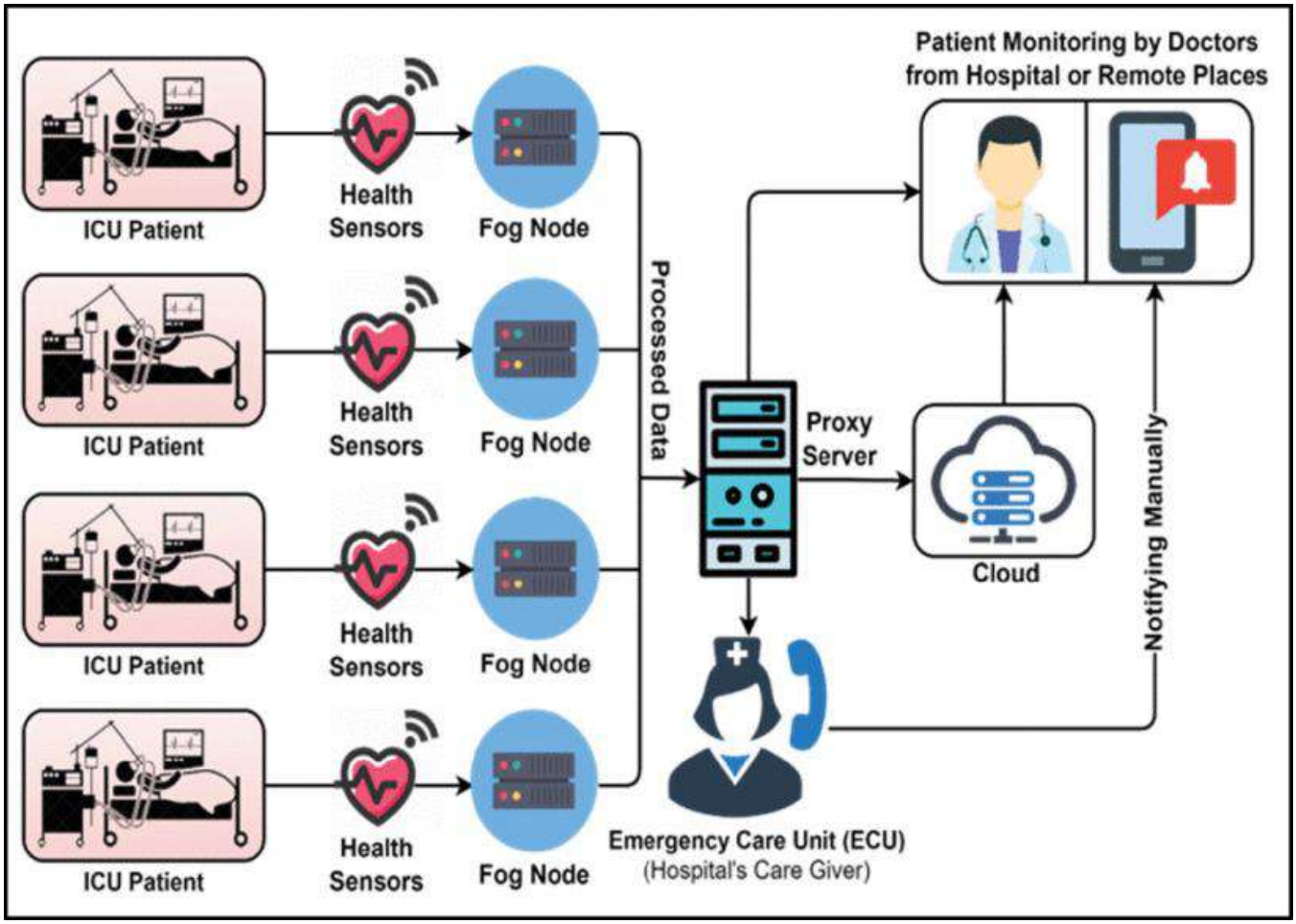


Рисунок 1.6 – Приклад використання туманних обчислень в медичній сфері [35]

Однак будь-який збій мережі або недостатня пропускна здатність у хмарній архітектурі призводить до тривалого часу реагування. Це неприйнятно для додатків у сфері охорони здоров'я, оскільки впливає на якість життя пацієнта і навіть може призвести до смерті. Можна стверджувати, що хмарні обчислення не завжди пропонують достатньо високий стандарт якості обслуговування для медичних застосунків.

Всесвітня організація охорони здоров'я прогнозує, що до 2035 року в усьому світі буде катастрофічно не вистачати 12,9 мільйона медичних працівників [40][71].

Для вирішення проблеми нестачі медичних працівників необхідні енергоефективні, недорогі та масштабовані медичні технології, які сприятимуть профілактиці та лікуванню захворювань [35][70].

Зараз туманні обчислення використовуються для оптимізації традиційних послуг, покращуючи час відгуку, контроль доступу, конфіденційність, а також збільшує якість надання послуг завдяки наближенню пристроїв до користувача і пацієнта (рисунок 1.6) [72].

Якщо новітні можливості туманних обчислень будуть ефективно застосовані в чутливих до часу додатках для охорони здоров'я, вони можуть прискорити виявлення раннього попередження про надзвичайні ситуації, що дозволить краще приймати рішення [73].

Також, функції туманних обчислень можуть бути використані для мінімізації затримки пошуку даних про пацієнтів [46].

1.5.2 Медичні пристрої

Використання медичних пристроїв стає все більш поширеним. Такі пристрої використовуються для надання телемедичних послуг, моніторингу стану пацієнтів та керування персоналом на місці під час хірургічних операцій [44]. Туманні обчислення не лише зменшують витрати в медичній сфері, а і постають наріжним

каменем сучасних систем охорони здоров’я за рахунок покращення якості передачі сигналів поміж пристроями [63].

Рисунок 1.7 – Прогнозування росту ринку медичних пристроїв

У 2022 році розмір глобального ринку медичних пристроїв сягав 30 міль’ярдів доларів та прогнозується, що до 2032 розмір ринку збільшиться майже у десять разів(рисунок 1.7) [52].

1.5.3 Система управління світлофорами

Туманні обчислення та транспортні мережі заміщають традиційну фізичну систему регулювання трафіку віртуальною системою керування трафіком, що є більш економним рішенням [74].

На додачу, туманні обчислення можуть забезпечити зменшення заторів, налаштовуючи дорожні сигнали, в залежності від ситуації на дорозі або за допомогою транспортних комунікацій, вказуючи водієві менш навантажений маршрут (рисунок 1.7) [53][68].

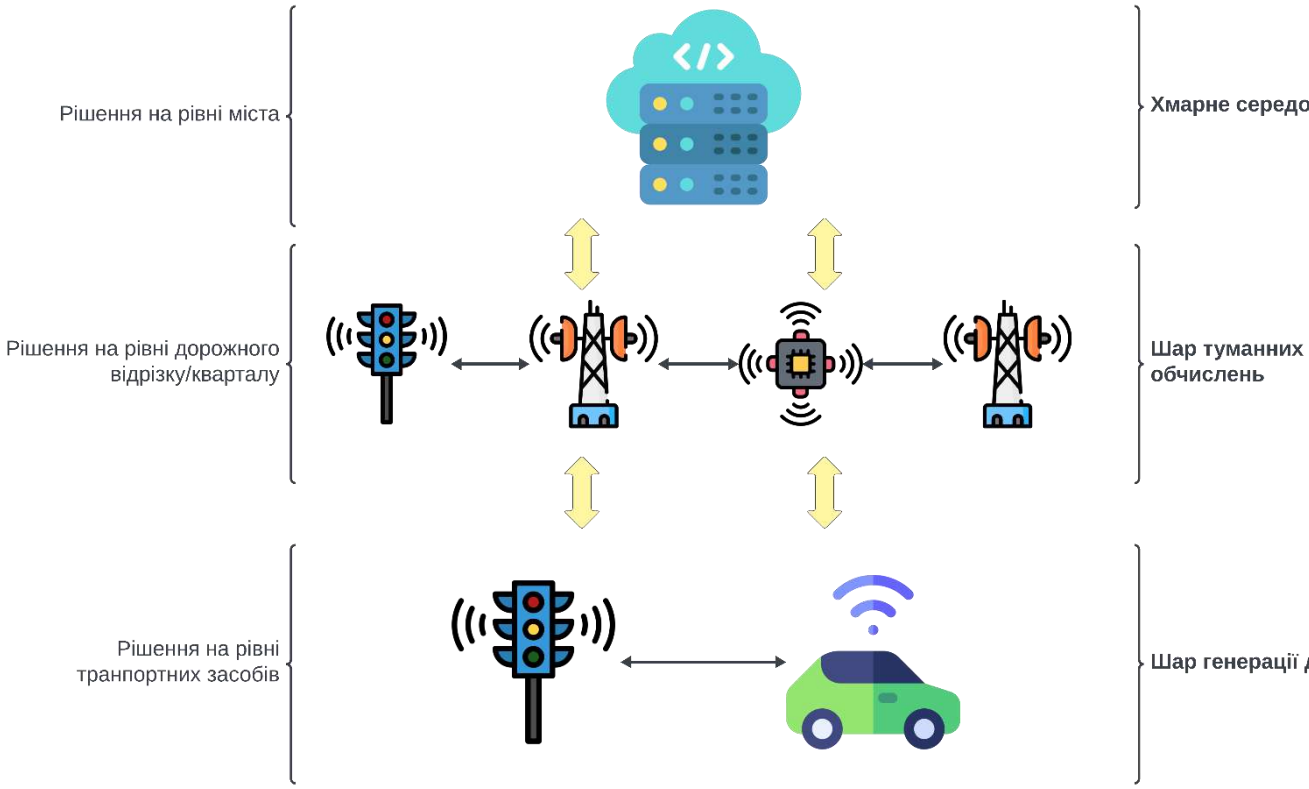


Рисунок 1.7 – Архітектура транспортної мережі із використанням концепції туманних обчислень

1.5.4 Розумні мережі

Розумні мережі допомагають керувати операційною діяльністю підключених пристроїв за допомогою зв'язку «машина-машина» та «людина-машина», що призводить до максимізації якості обслуговування, мінімізації рівня відмов, підвищення ефективності використання енергії та оптимізації безпеки [15][75].

1.5.5 Розумні будинки та міста

Основною метою застосунків для розумних міст є покращення управління міськими потоками та забезпечити зворотній зв'язок у реальному часі для вирішення проблем, що можуть виникати в операційній діяльності користувачів

[42]. Наявні рішення для розумних міст, що використовують хмарні обчислення, відповідають багатьом вимогам, однак, мають багато недоліків, які включають затримку, підтримку мобільності, масштабованість та локалізацію [43]. Туманні обчислення розширюють хмарні сервіси обробки даних, що призводить до зменшення затримок, покращення сервісу, контекстну обізнаність, балансування навантаження, ефективний розподіл даних та підтримку мобільності [76].

1.6 Проблеми та виклики що виникають при застосуванні туманних обчислень

Туманні обчислення стикаються зі структурними проблемами через свою гетерогенну природу [77]. Оскільки в якості туманного вузла може бути використана периферійна або центральна мережа, деякі вузли можуть бути не призначені для виконання обчислень загального призначення. Тому інтеграція функції загального призначення з її традиційною роллю може бути проблематичною [59]. Крім того, в умовах сервіс-орієнтованого розвитку великомасштабних додатків виникає проблема через обмеженість ресурсів деяких туманних вузлів [67]. Тому існує потреба в платформі програмування, яка допоможе в розробці розподілених застосунків. Реалізація системи безпеки для забезпечення цілісності, орієнтованої на дані, може суттєво вплинути на якість сервісу, що надає інфраструктура із використанням туманних обчислень [78]. Крім того, аутентифікація доступу до сервісів та обслуговування для збереження конфіденційності такої розгалуженої розподіленої мережі може виявитися складною [53].

Туманні обчислення стикаються з недоліком у комунікаціях між пристроями (D2D) [37]. Більшість досліджень зосередженні на комунікаціях «туман-хмара» та «IoT-туман». Однак існує необхідність включити дослідження комунікації "туман-туман", що є відкритим питанням в області туманних обчислень. Деякі приклади включають вибір туманного вузла, прогнозування завдань на основі історичних даних, використання ресурсів мережі туманних

обчислень, масштабованість, надійність і мобільність туманних вузлів. Також, дослідження потребують такі області як: продуктивність туманних екосистем, надійність, зберігання, економія енергії, використання розподіленого туману за допомогою поєднання пристроїв між пристроями (D2D) і стільникових мереж, мобільність і вибір вузлів, питання довіри і безпеки на всіх рівнях туманної парадигми [45][72][76].

Інтеграція інтернету речей і туманних сервісів почала покращувати якість послуг у галузі охорони здоров'я [77]. Це підвищує операційну ефективність, оптимізує енергоспоживання та мінімізує експлуатаційні витрати. За допомогою туманних застосунків електронної охорони здоров'я можна підвищити якість обслуговування пацієнтів і мінімізувати деякі медичні ускладнення [78]. Виклики, з якими стикається інтегрування туманних обчислень в інфраструктуру охорони здоров’я включають: захист конфіденційних даних, управління збоями системи, налаштування декількох систем і управління різноманітними системами, що мають численні виміри [79].

В інфраструктурі розумного міста управління інформаційно-комунікаційними ресурсами повинно бути більш ефективним, щоб виконувати аналіз великих обсягів даних в режимі реального часу близько до користувача і гарантувати безпеку інфраструктури та згенерованої інформації [80]. Інфраструктура має низку завдань які необхідно вирішити в областях надійності, стійкості до відмов та довговічності для обробки великих обсягів даних на межі мережі для поліпшення відмов вузлів, якості обслуговування та продуктивності розумного міста через зловмисну атаку, затори трафіку, а також надійності та доступності надання послуг [72].

1.7 Загроза безпеки в туманних обчисленнях

Більшість застосунків для туманних обчислень спрямовані на задоволення функційних потреб та потреб користувачів, ігноруючи вимоги безпеки або розглядаючи їх як другорядні [72]. Успадкувавши проблеми та недоліки безпеки

хмарних обчислень, туманні обчислення можуть бути вразливими для експлуатації [57]. Архітектура безпеки, визначена консорціумом OpenFog не є універсальною, вона лише описує всі механізми, що можуть бути застосовані із метою зробити туманний вузол захищеним, починаючи від процесора, закінчуючи програмним забезпеченням. Питання безпеки туманних обчислень все ще залишається актуальною проблемою як в академічному, так і в промисловому середовищі [72]. Крім того, рішення безпеки, реалізоване для хмарних обчислень, може бути неефективним для туманних обчислень, оскільки вони працюють на різних рівнях і їхня архітектура відрізняється [8].

Туманні обчислення схильні до атак, оскільки вони побудовані на традиційній мережі. Аутентифікація та конфіденційність є проблемними питаннями в концепції туманних обчислень. Комунікація вузлів туману може становити загрозу безпеці в мережі, оскільки заражений вузол може заразити інші [5]. Туманні обчислення вважаються легко вразливими, оскільки середовище в якому функціонують вузли туману знаходиться між хмарними центрами обробки даних і пристроями. Дослідження показують, що тема безпеки в туманних обчисленнях більше стосується автентифікації, захищеного обміну даними, атак на відмову в обслуговуванні та питань конфіденційності [68]. Питання автентифікації, безпечного обміну ключами та анонімність повинні бути належним чином вирішені для забезпечення належної безпеки та конфіденційності на рівні туманних обчислень [56].

1.8 Висновки

В розділі було проведено дослідження методів оптимізації IoT інфраструктури із застосуванням туманних обчислень. Було розглянуто концепцію туманних обчислень, її структура, переваги, недоліки та області застосувань. Було розглянуто архітектуру системи із використанням туманних обчислень, її порівняння із хмарними та периферійними обчисленнями.

Перевагами туманних обчислень є:

- зменшення затримки;
- збільшення пропускної здатності;
- безперервне надання послуг;
- широка підтримка пристроїв;
- ефективне використання ресурсів;
- зменшення обсягу передачі даних;
- легка масштабованість.

Недоліками туманних обчислень є:

- обчислювальна складність;
- складність розгортання та підтримки;
- вартість;
- обмежені ресурси.

Також, було визначено ряд вимог до IoT системи, для задоволення яких необхідно застосовувати туманні обчислення.

Було проаналізовано сфери застосування туманних обчислень та визначено мету їх застосування та проблеми які були вирішені за допомогою застосування туманних обчислень.

2 АНАЛІЗ ВИКОРИСТАННЯ КОНЦЕПЦІЇ ТУМАННИХ ОБЧИСЛЕНЬ ДЛЯ ОПТИМІЗАЦІЇ IOT ІНФРАСТРУКТУРИ

2.1 Аналіз методів оптимізації туманних обчислень в IoT інфраструктурі

Було проведено аналіз сучасних методів оптимізації функціонування IoT інфраструктури із застосуванням туманних обчислень. В ході проведення аналізу сучасних методів оптимізації IoT інфраструктури було розглянуто методи та алгоритми які можуть бути використані для зменшення затримки відповіді вузлів туману із метою задоволення QoS.

2.1.1 Мінімізація затримок

З метою мінімізації затримки обслуговування застосунків, що були розгорнуті із використанням архітектури IoT – туманні обчислення – хмарне середовище використовуються методи що враховують навантаження на вузли туману та типи робочих навантажень, що генеруються IoT пристроями. У випадку, якщо розрахована затримка обслуговування на основі поточного навантаження на туманний вузол менше за порогове значення, завдання буде прийнято на туманному вузлі; в іншому випадку завдання буде запланованим найбільш оптимальний туманний вузол із найменшим навантаженням в даний момент часу. Найбільш оптимальний сусідній вузол визначається шляхом обраховування очікуваної затримки обслуговування і затримки поширення всіх сусідніх вузлів. Коли завдання досягає максимального значення розвантаження, воно буде вивантажено в хмару. Проте, для більшості методів передбачається, що вузли туману з’єднані між собою напряму. З цього слідує, що метод мінімізації затримок не враховує вартості зв’язку.

Окрім цього існують евристичні методи, що базуються на основі лінійного програмування з метою розподілу завдань медичних кіберфізичних систем Запропонований в роботі [82] алгоритм використовує розгортання віртуальних машин для віртуальних медичних пристроїв і забезпечує необхідний QoS

медичних застосунків в режимі реального часу, враховуючи вартість зв'язку, вартість обчислень, розміщення віртуальних машин і ціна розподілу завдань. Недоліком запропонованого алгоритму є те, що запропонована архітектура мережі туманних обчислень базується на стільниковій мережі, у якій вузли туману представлені як базові станції стільникової мережі. Окрім того, не враховано можливості фактичного планування завдань та балансування навантаження, а лише розміщення віртуальних машин. Як результат, метод не може бути узагальнений на архітектуру туманних обчислень.

Також є методи, що використовують цілочисельне нелінійне програмування для спільної оптимізації планування завдань і розміщення сховища.

2.1.2 Розподілення навантаження

Із метою розподілення навантаження використовуються евристичні алгоритми для визначення розміщення завдань на вузлах туману з метою розвантаження мережі на основі функції вартості, яка враховує вартість зв'язку, вартість обчислень та енергоспоживання.

Також широко використовуються методи, що використовують схему управління сервісами – технологію програмно-визначених мереж (SDN) для зменшення навантаження на мережу. Окрім цього використовуються евристичні алгоритми для прийняття диференційованих рішень щодо розвантаження хмарних обчислень з використанням оптимізаційної функції, що базуються на енергоспоживанні зв'язку, енергоспоживанні обчислень, моделях затримок завдань.

Для планування запитів від розумних пристроїв, таких як розумні лічильники, на віртуальних машинах вузлів туману використовуються різні алгоритми, в тому числі циклічний, дросельний, оптимізація рою частинок (PSO), оптимізація мурашиних колоній (ACO), а також гібрид штучної бджолиної колонії (PSO) та алгоритму (ACO), що демонструє найкращі результати.

2.1.3 Оптимальне розміщення пристроїв

Широко використовуються методи оптимального розміщення пристроїв туманної мережі на визначеній великій площі з метою оптимізації енергоспоживання вузлів туману при одночасному задоволенні вимоги QoS застосунків IoT. Із метою знаходження компромісу між фізичним розташування туманних вузлів, енергоспоживанням та QoS для розвантаження задач мережі використовують генетичні алгоритми. В дослідженні [83] розглядається проблема розташування периферійних серверів, що використовують метод оптимізації PCO, заснований на багатоцільовій функції із метою зменшення загального енергоспоживання та підтримання задовільної затримки.

Також, використовуються генетичні алгоритми для прийняття рішень про розвантаження завдань в туманних обчисленнях. Проте, більшість досліджень не розглядають питання балансування навантаження між туманними вузлами.

2.2 Архітектура туманної мережі

В роботі розглянуто метод ефективного планування завдань в IoT інфраструктурі із застосуванням туманних обчислень із використанням мета-евристичного ACO алгоритму. Головною метою якого є мінімізація часу відповіді виконання завдань із використанням даних про пропускну здатність мережі, затримку та існуюче навантаження на пристрої туманної мережі.

Архітектура IoT-туманної мережі складається з трьох шарів: IoT-шару, туманного шару і хмарного шару, як показано на рисунку 6. Така ж архітектура використовується в дослідженні [84]. У шарі IoT, ряд географічно розподілених сенсорних вузлів з'єднані із використанням локальної мережі. Рівень IoT збирає різні дані з різних додатків, такі як погода, якість повітря та інтенсивність руху. Рівень IoT з'єднаний з рівнем туманних обчислень, що містить ряд вузлів туману. Кожен вузол туману використовується для агрегування, фільтрації та опрацювання зібраних даних із датчиків. Кожен вузол туману містить локального

агента, який відповідає за збір даних про продуктивність, таких як швидкість надходження даних з датчиків та швидкість обслуговування датчиків.

Датчики надсилають запити на розвантаження завдань на вузли туману, а вузли туману пересилають ці запити на головний вузол туману, який відповідає за планування розвантаження завдань на вузли туману, використовуючи агреговані дані від агентів туману. Хмарний рівень забезпечує потужні обчислювальні можливості та можливості зберігання даних. Туманний рівень підключений до хмарного рівня через мережу Інтернет, а не через локальну мережу. Хмарний рівень використовується для інтенсивної обробки і зберігання даних.

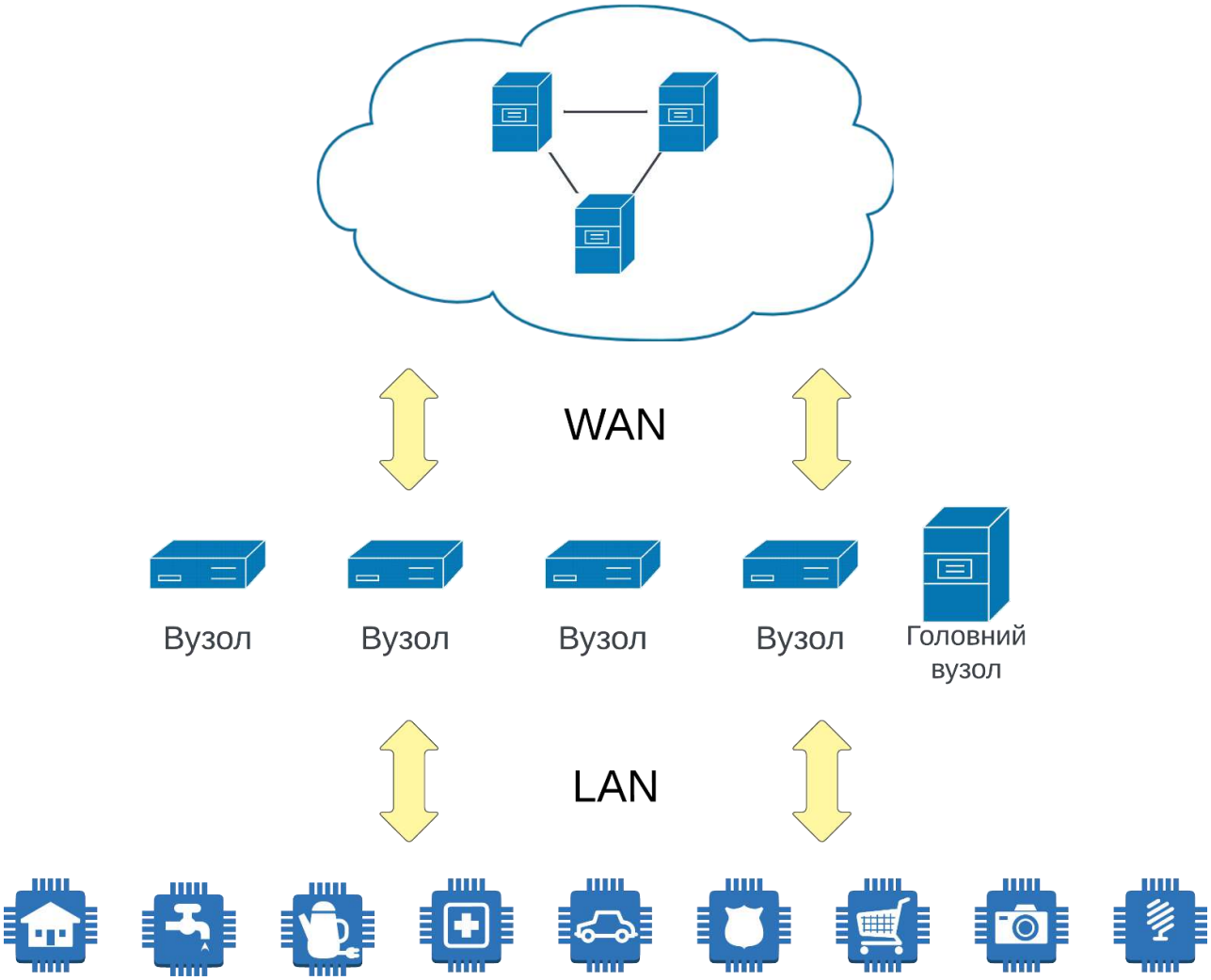


Рисунок 2.1 – Архітектура IoT інфраструктури із використанням туманних обчислень та хмарного середовища

2.2.1 Формулювання моделі туманних обчислень в IoT інфраструктурі

Рівень пристроїв IoT складається із довільної кількості розподілених датчиків $S_1, S_2, \ldots, S_m$, де кожен датчик S_k генерує дані зі швидкістю λ_i. Рівень туманної мережі складається із довільної кількості туманних вузлів $fg_1, fg_2, \ldots, fg_n$. У таблиці 2.1 наведено позначення, що використовуються у формулах.

Таблиця 2.1 – Позначення, що використовуються у формулах

Символ	Значення
S_k	Сенсор номер k
λ_y	Кількість даних згенерованих сенсором S_i
C	Клас застосування
fg_y	Туманний вузол номер j
λ_y	Кількість даних, що надходять до туманного вузла fg_j
λ_{yc}	Кількість даних, що надходять до туманного вузла fg_j від усіх датчиків, що належать до класу застосування c
L_{ky}	Затримка відповіді через мережу між S_i та fg_j
$Dsize_y$	Кількість даних, згенерованих датчиком S_i
BW_l	Пропускна здатність мережі
M_{sens}	Загальна кількість датчиків
N_{nod}	Загальна кількість вузлів
L_{yc}	Затримка відповіді між fg_j та хмарним середовищем
BW_c	Пропускна здатність мережі хмарного середовища
μ_{ky}	Швидкість обробки вузлом fg_j даних, згенерованих датчиком S_i
μ_{yc}	Швидкість обробки вузлом fg_j даних класу застосування c

Вартість передачі даних між датчиком S_k і вузлом туману fg_y, як показано в рівнянні (2.1), є сумою мережевої затримки L_{ky} між датчиком S_i і вузлом туману fg_y і часу передачі даних між S_k і fg_y, який розраховується як $Dsize_y/BW_l$, де $Dsize_y$ – розмір даних, що генеруються датчиком S_k, а BW_l – пропускна здатність локальної мережі.

$$FogCCost_{ky} = L_{ky} + \frac{Dsize_k}{BW_C}. \quad (2.1)$$

Вартість зв’язку між туманним вузлом fg_y і хмарою, як показано в рівнянні (2.2), є сумою затримки хмари L_{jc} і часу передачі даних між туманним вузлом fg_j і хмарою, що розраховується як $Dsize_y /BW_c$, де BW_c – пропускна здатність хмарної мережі.

$$CloudCCost_{ky} = L_{yc} + \frac{Dsize_k}{BW_c}. \quad (2.2)$$

Загальна вартість зв'язку для розвантаження датчика S_i розраховується за допомогою рівняння (2.3).

$$\mathrm{CommCost}_{ky} = \mathrm{CloudCCost}_{ky} + \mathrm{FogCCost}_{ky}. \quad (2.3)$$

Кожен датчик S_k генерує дані відповідно до розподілу Пуассона зі швидкістю λ_k . Згенеровані дані надсилаються до туманних вузлів для виконання фільтрації, агрегації та простого аналізу. Нарешті, туманні вузли надсилають дані до хмари, де виконується складніший аналіз, дорога обробка та зберігання.

Час обслуговування вивантаження завдань від датчика S_k до вузлів fg_k розраховується відповідно до моделі масового обслуговування M/M/1 за допомогою рівняння (2.4).

$$ST_{ky} = \frac{1}{\mu_{ky} - \lambda_k}, \quad (2.4)$$

де μ_{ky} - швидкість обслуговування,

а λ_k - швидкість надходження даних від датчика S_k ,

Для великої кількості датчиків, де кожен датчик належить до певного класу застосування c , час обслуговування розраховується за допомогою рівняння (2.5).

$$ST_{yc} = \frac{1}{\mu_{yc} - \lambda_{yc}}. \quad (2.5)$$

де μ_{yc} - швидкість обслуговування у вузлі туману fg_k для певного класу додатків с,

а λ_{yc} - сумарна швидкість надходження даних від датчиків, що належать до певного додатку с, яка обчислюється за допомогою рівняння (2.6).

$$\lambda_{yc} = \sum_{i \in C} \lambda_{yk}. \quad (2.6)$$

Коефіцієнт використання вузла туману fg_k , U_y , є сумою коефіцієнтів використання розвантажених завдань від усіх датчиків S_k , що розраховується за допомогою рівняння (2.7).

$$U_y = \sum_c U_{jc} = \sum_c \sum_{i \in c} x_{ky} \times \frac{\lambda_{yk}}{\mu_{yk}}, \quad (2.7)$$

де x_{ky}=1, якщо завдання датчика S_k вивантажуються на туманний вузол fg_k.

Загальний час відгуку робочого навантаження датчика S_k в результаті розвантаження завдань є сумою вартості зв'язку IoT-туман, вартості зв'язку туман-хмара та середнього часу обслуговування на вузлах туману, який розраховується за допомогою рівняння (2.8).

$$R_{ki} = CommCost_{ky} + ST_{kc} = L_{yc} + \frac{Dsize_i}{BW_c} + L_{ky} + \frac{Dsize_i}{BW_l} + \frac{1}{\mu_{kc} - \sum_{i \in c} \lambda_{ky}}. \quad (2.8)$$

Середній час відгуку вузла туману fg_j в результаті виконання всіх завдань, вивантажених на цей вузол, обчислюється за допомогою рівняння (2.9).

$$R_y = \sum_c \frac{R_{yc}}{1 - U_{yc}}. \quad (2.9)$$

Навантаження fg_j на вузол туману fg_j відносно всіх вузлів туману обчислюється за допомогою рівняння (2.10).

$$load_k = 1 - \frac{R_y - R_{avg}}{\sum_k R_k}. \quad (2.10)$$

2.3 Еволюційні алгоритми для розвантаження завдань туманної мережі в IoT інфраструктурі

Проблема розвантаження завдань IoT-туману є критично важливою для застосунків IoT, що вимагають обробки даних в реальному часі. Дані, що генеруються пристроями IoT, повинні бути збалансовані між вузлами туману, враховуючи затримку в мережі, пропускну здатність мережі, час обробки на вузлах туману і існуюче навантаження на вузли туману.

Таким чином, рішення задачі планування повинно вибирати цільовий вузол туману, який гарантовано задовольняє визначені обмеження QoS, зокрема, час відгуку, враховуючи характеристики мережі та поточне навантаження на вузли туману. Ця задача є NP-важкою, в якій складність зростає експоненціально зі збільшенням кількості датчиків та вузлів туману. Тому використання традиційних методів такого пошуку є складним завданням.

До розгляду представлено два еволюційні метаевристичні алгоритми, що використовують оптимізацію мурашиних колоній та оптимізацію рою частинок,

щоб подолати цю проблему. Для вирішення задачі оптимізації використано алгоритми ACO та PSO.

2.3.1 Алгоритм ACO для розподілення задач в туманній мережі

Алгоритм ACO виявився ефективним як оптимізаційна метаевристика для декількох NP-важких дослідницьких задач, включаючи задачі комівояжера та планування роботи майстерні. Крім того, алгоритм ACO використовувався в подібних задачах планування в хмарі, включаючи планування віртуальних машин на хмарних ресурсах та планування завдань на віртуальних машинах з метою балансування навантаження на віртуальні машини та зменшення часу відгуку завдань. Крім того, алгоритм ACO використовується для планування завдань IoT у хмарі. Також, алгоритм ACO використовується із метою планування завдань з урахуванням дедлайнів для туманних обчислень у багаторівневій інфраструктурі IoT. Запропонований алгоритм робить фокус на максимізації прибутку постачальника туманних послуг та враховує обмеження термінів виконання завдань IoT.

У пошуках перекладання обчислення даних із датчика S_k на вузли туману fg_y з метою мінімізації часу відгуку, k-а мураха обирає вузол туману fg_k для перекладання навантаження, згенерованого датчиком S_k, з ймовірністю, що задається рівнянням (2.11).

$$P_{ky}^{k}(t) = \frac{(\tau_{ky}(t))^{\alpha}(\eta_{ky}(t))^{\beta}}{\sum_s(\tau_{ks}(t))^{\alpha}(\eta_{ky}(t))^{\beta}}, \quad (2.11)$$

де α і β - евристичні константи; $\alpha \geq 0$ - евристичний параметр, що контролює вплив кількості феромону,

а $\beta \geq 1$ - евристичний параметр, який визначає важливість якості розвантаження завдання,

$\eta_{ij}(t)$ - евристична функція, яка представляє якість розвантаження завдання і обчислюється за допомогою рівняння (2.12).

$$\eta_{ij}(t) = \frac{load_j}{R_{ij}}, \tag{2.12}$$

де R_{ij} розраховується за допомогою рівняння (8),

а $load_j$ являє собою навантаження на вузол туману j і розраховується за допомогою рівняння (2.10).

Очевидно, що зі збільшенням R_j зменшується $load_j$ і зменшується $\eta_{ij}(t)$. Як наслідок, ймовірність перекладання обчислень датчика і на вузол туману fg_j буде невеликою. $\tau_{ij}^k\ (t)$ - це кількість феромонного сліду для розвантаження завдання для мурашки k в ітерації (t), а $\tau_{ij}^k\ (t+1)$ - це феромонний слід від розвантаження цього завдання для мурашки k в ітерації $t+1$, визначений за допомогою рівняння (2.13).

$$\tau_{ij}^k(t+1) = (1-\rho)\tau_{ij}^k(t) + \rho\Delta\tau_{ij}^k(t), \tag{2.13}$$

де $\Delta\tau_{ij}^k(t)=1/R_{ij}$ і ρ - константа, яка представляє швидкість випаровування феромону, що імітує ефект випаровування феромонів на кожному кроці.

Окрім цього, феромонний слід оновлюється глобально після того, як всі мурахи завершили виконання усіх своїх можливих завдань, тобто повну ітерацію. Глобальне оновлення феромону відбувається за допомогою рівняння (2.14).

$$\tau_{ij}^k(t+1) = \left(1-\rho_g\right)\tau_{ij}^k(t) + \rho_g\Delta\tau_{ij}^k(t), \tag{2.14}$$

де $\Delta\tau_{ij}^k(t)=1/L_{best}$, L_{best} - найкраще знайдене завдання завантаження,

а ρ_g - глобальна швидкість випаровування.

Початкове значення феромонного сліду розраховується за допомогою рівняння (2.15).

$$\tau_{ky}^{k}(0) = \frac{R_{ky}}{R_{avg}}. \quad (2.15)$$

Метаевристичний алгоритм, що використовує ACO для пошуку ефективного планування завдань для датчиків IoT на доступних вузлах туману, який гарантує, що задані обмеження QoS, а саме час відгуку, задовольняються з врахуванням характеристик мережі, часу обслуговування та поточного навантаження на вузли туману зображено у вигляді блок-схеми(Додаток А). Даний алгоритм починається із задання значень евристичних параметрів α, β, ρ, ρ_g, кількості мурах M_{ants} та встановлення максимальної кількості ітерацій N_{iter} . Крім того, алгоритм ініціалізує кількість сенсорних вузлів N_{nod}, інтенсивність роботи кожного сенсора λ_i та інтенсивність обслуговування вузла туману μ_{ky} .

На другому кроці обчислюється час відгуку R_{ky} вузла туману fg_y за допомогою рівняння (8). Третій та четвертий кроки відповідають за ініціалізацію початкового феромонного сліду $\tau_{ky}^{k}(0)$ та евристику розвантаження завдання $\eta_{ky}^{k}(t)$ для кожної мурахи ant_k за допомогою рівнянь (2.12) та (2.15). На кожній ітерації мураха ant_k перекладає обчислення навантаження датчика S_k на вузол туману fg_k з ймовірністю $P_k^k(t)$, обчисленою за допомогою рівняння (2.11). Вибір вузла туману fg_k здійснюється за допомогою методу колеса рулетки. Із допомогою методу колеса рулетки обчислюється ймовірність для кожного рішення, а кумулятивні ймовірності розташовуються в порядку зростання. Генерується випадкове число $\in[0,1]$, яке порівнюється з розрахованими кумулятивними ймовірностями. Нарешті, знаходиться відповідний розв'язок, що відповідає згенерованому випадково числу. Цей метод виявився ефективним у відборі потенційно корисних рішень і часто використовується в генетичних алгоритмах.

Коли мураха завершує вивантаження всіх сенсорів, локальна матриця сліду феромонів оновлюється за допомогою рівняння 2.13. Після кожної ітерації

алгоритму ACO оновлюється глобальну матрицю сліду феромонів $\tau_{ky}^{k}(t)$, використовуючи рівняння (2.14). Алгоритм працює поки не буде досягнуто максимальної кількості ітерацій.

2.3.2 Алгоритм PSO для розподілення задач в туманній мережі

PSO - це метаевристичний алгоритм оптимізації, що базується на популяції індивідів, які співпрацюють. Така популяція називається роєм часток, розміром N_{swarm}. Кожна особина в рої, яка називається часткою, представляє рішення, яке є позицією в просторі пошуку. Частинка P_y у D-вимірному просторі пошуку виражається у вигляді D-вимірного вектора $P_k=\{p_{k1},p_{k2},....,p_{kd}\}$. Частинки рою випадковим чином роблять пошук можливих розв'язків задачі. Рух пошуку кожної частинки залежить від локального оптимуму її історичного пошуку та знайденого глобального оптимуму всіх частинок. Кожна частинка під час процесу пошуку, P_i, оновлює нове положення, $X_k(t+1)$, на основі обчислення швидкості, $V_k(t+1)$, як показано в рівняннях (2.16) і (2.17).

$$V_k(t+1) = \omega \times V_k(t) + C_1 \times \alpha \times (Pl_k - P_k) + C_2 \times \beta \times \left(P_g - P_k(t)\right), \quad (2.16)$$

де $k=1,2,...,N_{swarm}$,

і $t=1,2,....,itermax$ - номер ітерації,

ω - інерційна вага початкової швидкості, яка балансує між розвідкою і розробкою.

$$P_k(t+1) = P_k(t) + V_k(t+1). \quad (2.17)$$

Якщо на початку процесу пошуку використовується велике значення інерційної ваги, то перевага надається розвідці, тоді як менше значення інерційної ваги сприяє більшій експлуатації. C1 і C1 - коефіцієнти навчання. α і β - випадкові

числа, рівномірно розподілені на $\in[0,1]$. PSO спочатку було запропоновано для розв'язання задач у неперервних областях. Оскільки планування розвантаження мобільних завдань відбувається в дискретному просторі пошуку, PSO повинен бути модифікований відповідно до доменної області.

Позиція частинки p_{ky} представляє розвантаження вузла датчика S_k на вузол туману fg_k. Представлення розвантажених сенсорних задач та вузлів туману є дискретними. На рисунку 2.2 показано приклад кодування частинок. У прикладі завдання вузла датчика S_1 вивантажується на вузол туману fg_3, завдання S_2 вивантажується на fg_1, завдання S_3 вивантажується на fg_2, завдання S_4 вивантажується на fg_2, а завдання S_5 вивантажується на fg_1.

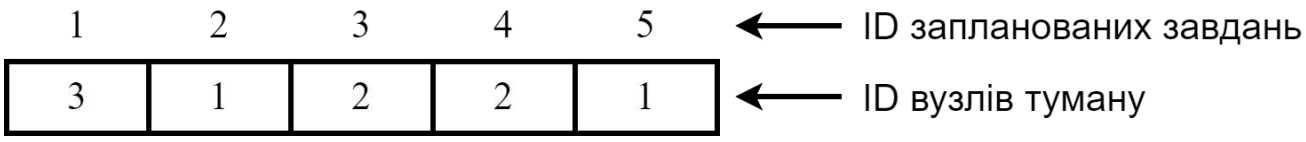


Рисунок 2.2 – Приклад кодування частинок

Під час кожної ітерації положення та швидкість частинки оновлюються за допомогою рівнянь (2.16) та (2.17). Однак кожне значення положення частинки має бути перетворене в дискретне числове значення за допомогою рівняння (2.18).

$$p_{ky} = \begin{cases} \lfloor |p_{ky}| \rfloor & 0 > p_{ky} \leq N_{nod} \\ \lfloor |p_{ky}| \rfloor \,\%N_{nod} & otherwise \end{cases} \quad (2.18)$$

де N_{nod} – кількість вузлів туману,

а $\lfloor |p_{ky}| \rfloor$ - верхня межа абсолютного значення p_{ky} для частинки p_k .

Загальний вигляд запропонованої системи підтримки прийняття рішень показано на алгоритмі що зображено на рисунку 2.4. Алгоритм починається з ініціалізації максимальної кількості ітерацій N_{iter}, параметрів оновлення швидкостей, включаючи α,β,i_ω, та ймовірності мутації P_{mut} . Далі запропонований

PSO-алгоритм проводить ініціалізацію положення частинок P_k та швидкості V_k випадковими значеннями.

$$Maximizef = \sum_{y=1}^{N_{nod}} \frac{load_y}{R_y}. \quad (2.19)$$

Кожна частинка оцінюється за допомогою фітнес-функції, яка представляє якість розв'язку, що виражається кодуванням частинки. значення фітнес-функції обчислюється за допомогою рівняння (2.19).

Чим вище значення фітнес-функції, тим краще рішення. Після обчислення фітнесу кожної частинки встановлюється локальний оптимум кожної частинки. Найкраща фітнес-функція серед рою встановлюється як глобальний оптимум. Алгоритм працює протягом певної кількості ітерацій N_{iter} .

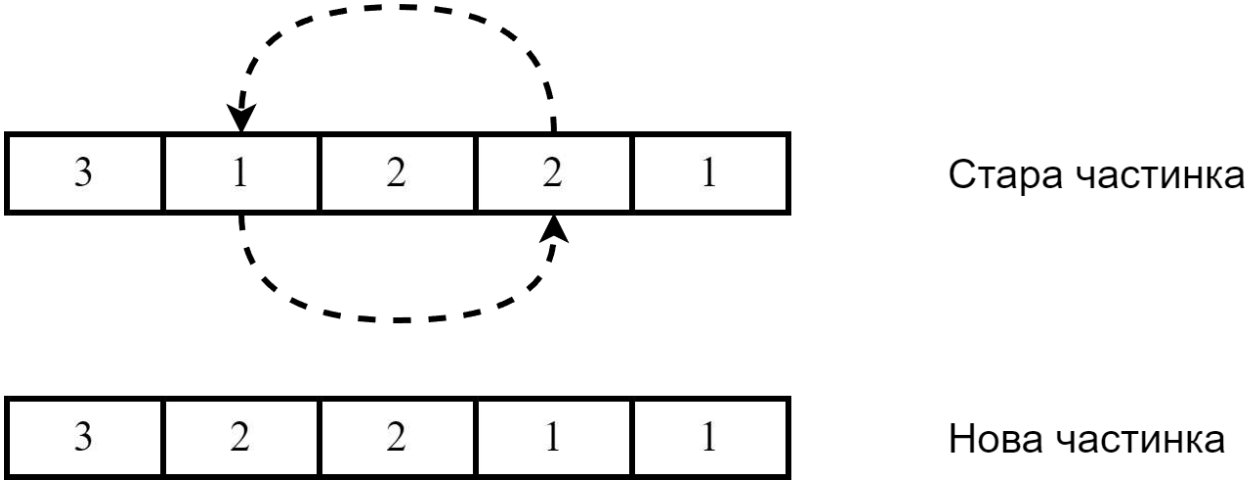


Рисунок 2.3 – Приклад мутації частинок

Під час кожної ітерації швидкість і положення кожної частинки оновлюються за допомогою рівнянь (2.16) і (2.17), а положення кожної частинки перетворюється в дискретне значення за допомогою рівняння (2.18). Алгоритм PSO підтримує нові локальні та глобальні оптимуми. Нарешті, до кожної частинки застосовується випадкова мутація. У процесі мутації значення двох випадкових вузлів частинки обмінюються місцями, щоб згенерувати нову задачу розвантаження частинок, як показано на рисунку 2.3.

Рисунок 2.4 – Дискретний PSO алгоритм, аркуш 1

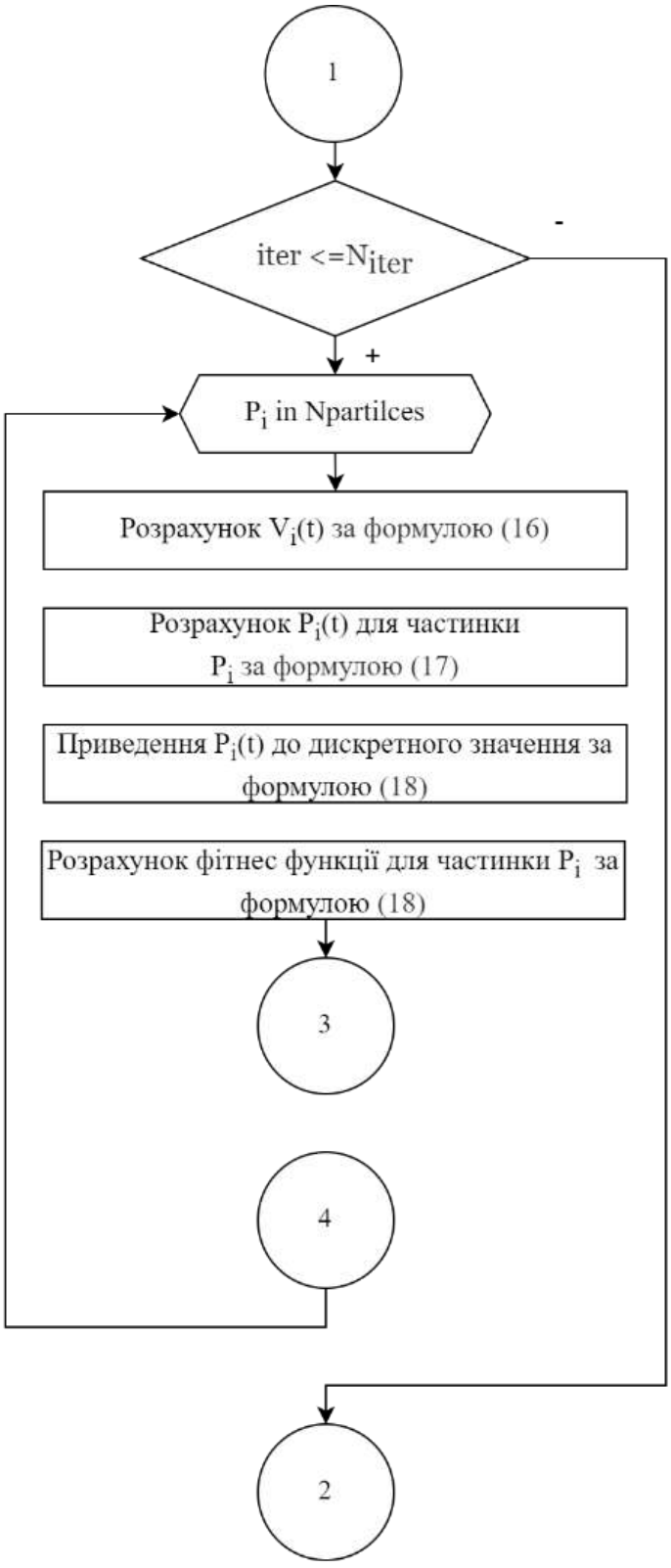


Рисунок 2.4 – Дискретний PSO алгоритм, аркуш 2

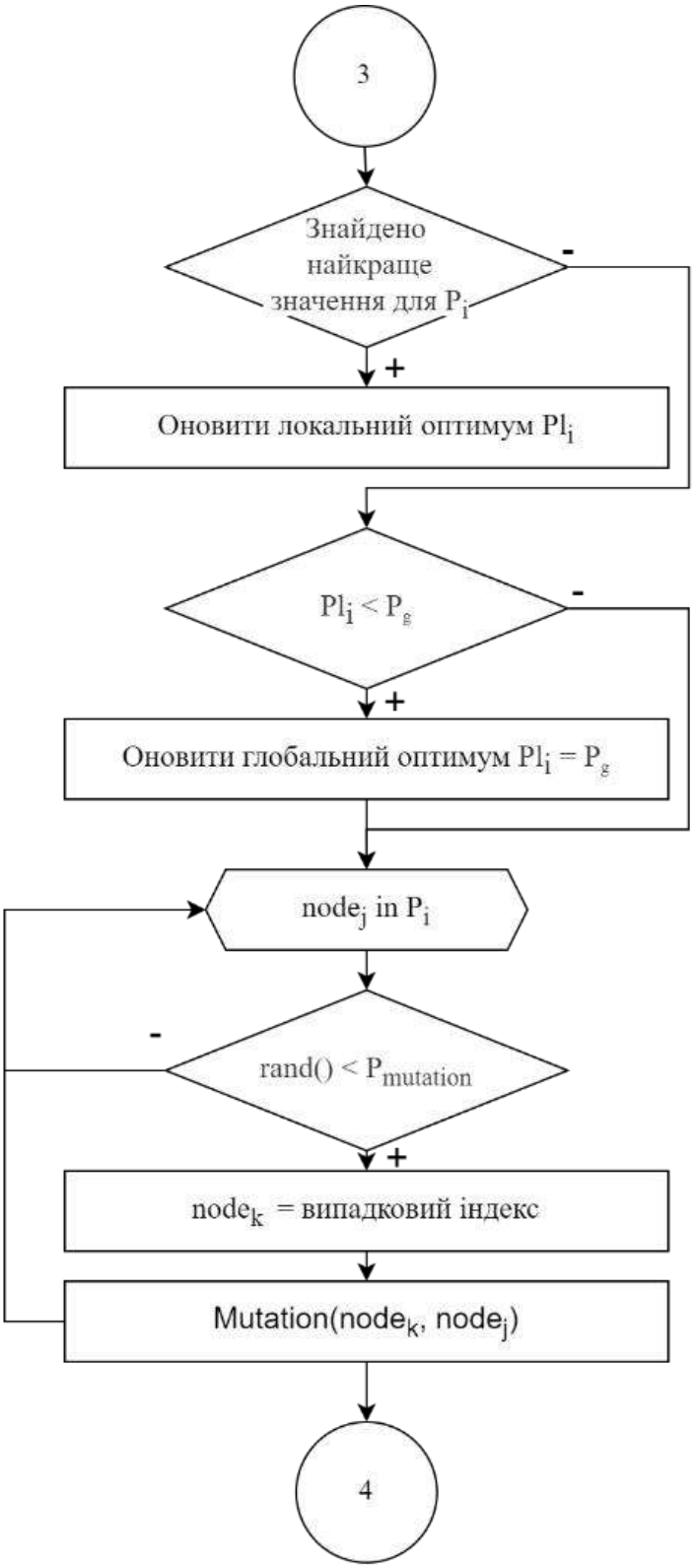


Рисунок 2.4 – Дискретний PSO алгоритм, аркуш 3

Мутація покращує можливості локального пошуку і зберігає різноманітність згенерованого нового рішення для розвантаження задач.

2.3.3 Порівняння результатів роботи системи на основі алгоритмів RR, ACO та PSO

Для перевірки ефективності запропонованих алгоритмів використовується тестовий сценарій IoT. Сценарій IoT-туман-хмара складається з трьох шарів, в якому перші шари складаються з великої кількості датчиків. Датчики включають три різних IoT-сенсори, які збирають різні дані про завдання, такі як температура, умови руху та відео з камер спостереження. 200-2000 датчиків IoT випадковим чином розподілені в радіусі 100-300 метрів від вузлів туману. Кожен датчик генерує 250-1024 КБ даних відповідно до завдання датчика. Допустима затримка виконання завдань - 100 мс. Другий рівень складається з 8-20 туманних вузлів, які використовуються для агрегації даних і обробки робочого навантаження, згенерованого першим рівнем. Передбачається, що кожний туманний вузол містить є ряд віртуальних машин для планування завдань, у якому кожна віртуальна машина використовується для планування певного класу IoT-застосунків. Хмарний рівень побудовано із одного центру опрацювання даних, що підключений до туманного рівня через мережу Інтернет.

Моделювання проводилося на комп'ютері з процесором Intel Core i7-4510 з тактовою частотою 2,60 ГГц та 8 ГБ оперативної пам'яті. В експериментах налаштування параметрів встановлювалися відповідно до значень, описаних у таблиці 2.2. Налаштування параметрів отримано шляхом проведення декількох попередніх експериментів.

Експерименти проводяться для оцінки запропонованого алгоритму планування завдань ACO згідно декількох оціночних метрик, а саме: середній час відгуку, ступінь дисбалансу та середньоквадратичне відхилення дисбалансу навантаження. Ступінь дисбалансу показує дисбаланс між доступними вузлами туману і розрахований із використанням рівняння 2.20.

Таблиця 2.2 – Параметри значень експерименту

Параметр	Значення
BW_{c1}	12 MB/s
BW_l	400 MB/s
$M_{sensors}$	300-3000
N_{nod}	8-12
λ_y	10-50
c	1-3
μ_{ij}	50-300
$Dsize_k$	250 kb -1 Mb
L_{ky}	U(2-20) ms
L_{yc}	30 ms
M_{ant}	30
N_{iter}	100
P	0.2
ρ_g	0.3
α	0.4
β	3

$$DI = \frac{Max(R_y) - Min(R_y)}{R_{avg}}, \ \ y = 1,2, \dots, N_{nod}. \tag{2.20}$$

Стандартне відхилення часу відгуку розвантажених завдань використовується для оцінки розподілу навантаження між вузлами туману, де менше значення означає високозбалансовані вузли був розрахований за формулою 2.21.

$$SD = \sqrt{\frac{\sum_y (R_y - R_{avg})^2}{N_{nod}}}. \tag{2.21}$$

Перший тестовий сценарій демонструє поведінку запропонованих алгоритмів для незначної кількості вузлів туману, N_{nod}=8. Параметри налаштування наступні: c=1 , λ_y=30 , та μ_{ky}=200 . На рисунку 2.5 продемонстровано середній час відгуку запланованих завдань для запропонованих алгоритмів планування PSO та ACO з різною кількістю датчиків, а також порівняння запропонованих алгоритмів з планувальником RR. У цьому сценарії середній час відгуку збільшується зі збільшенням кількості завдань за рахунок додавання більшої кількості IoT-датчиків. Планувальник RR порушує визначену прийнятну затримку у 100 мс. Запропонований алгоритм розвантаження ACO забезпечує менший середній час відгуку, ніж алгоритм PSO.

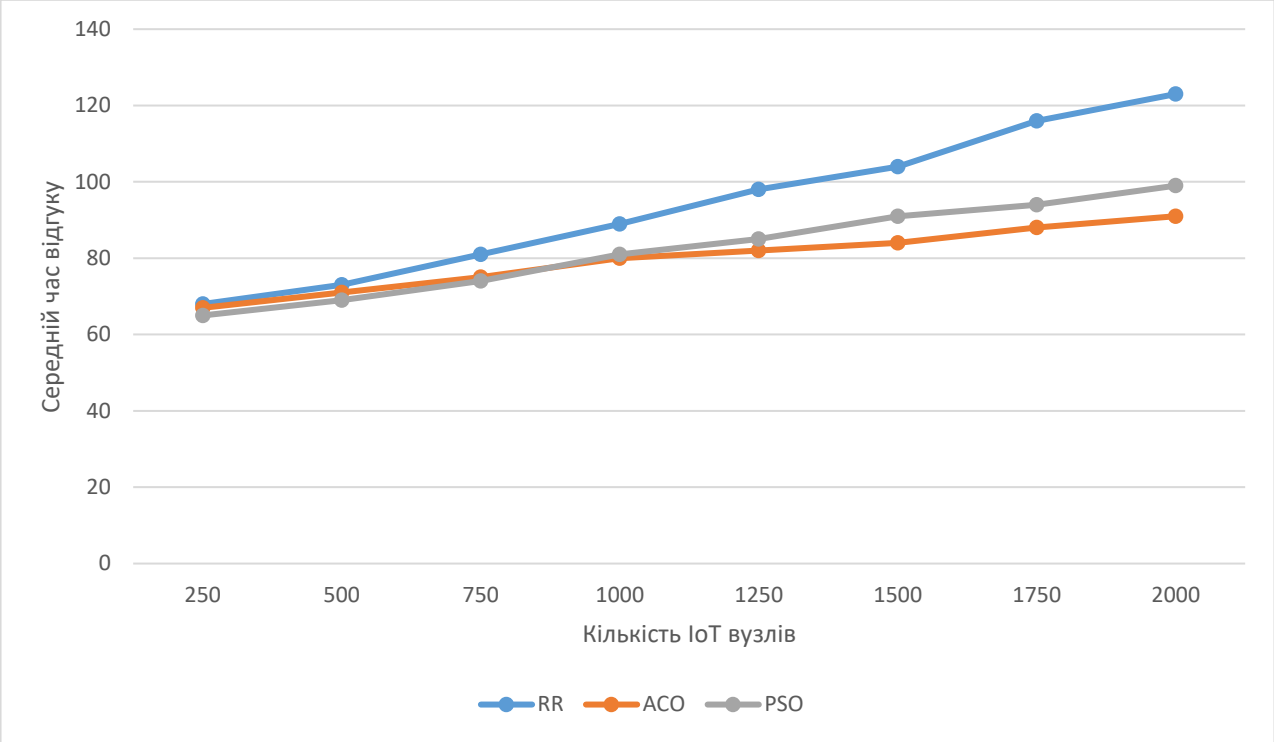


Рисунок 2.5 – Середній час відгуку вивантажених завдань при кількості туманних вузлів 8

Наступний тестовий сценарій показує поведінку запропонованих алгоритмів при більшій кількості вузлів туману, N_{nod}=20, та більшій швидкості передачі даних, λ_y=50. Експеримент проводиться для одного класу застосунків c=1, швидкість обслуговування вузлів туману μ_{ij}=300 . На рисунку 2.6 показано,

що запропонований алгоритм розвантаження ACO зберігає менший час відгуку при збільшенні кількості датчиків IoT і не порушує заданої допустимої затримки.

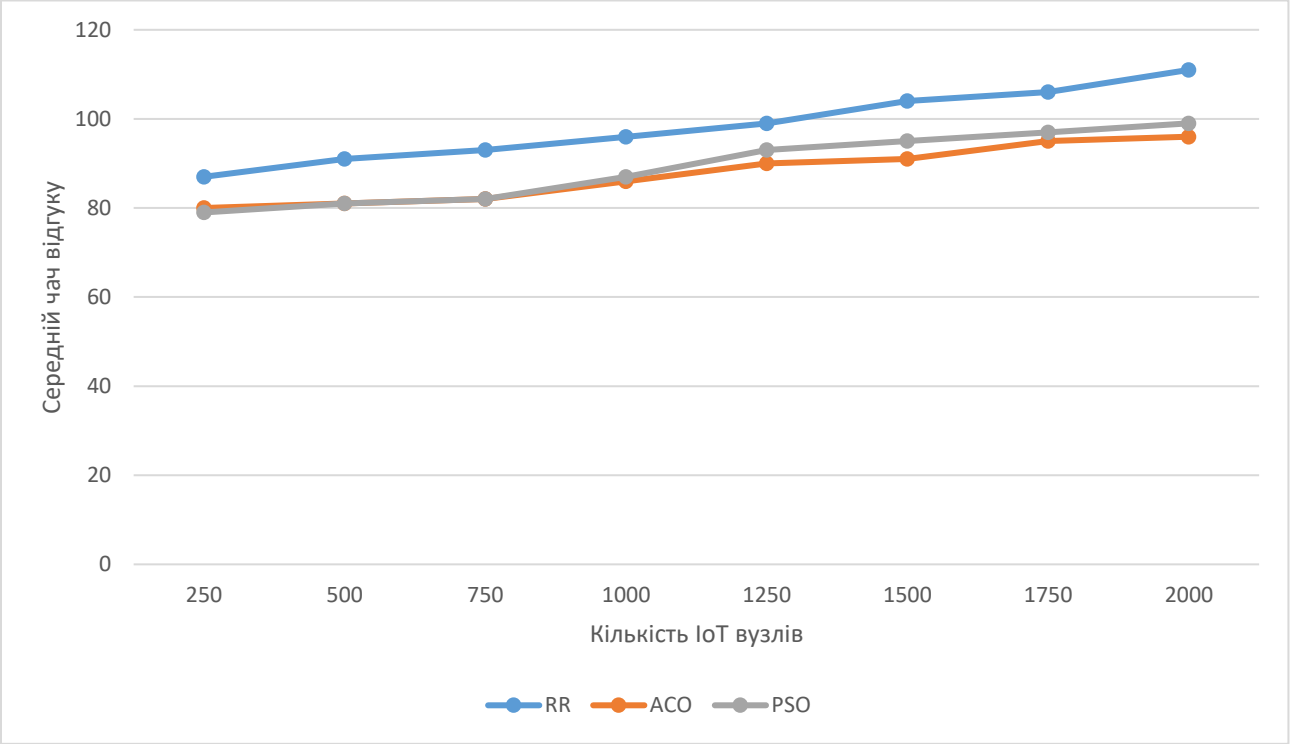


Рисунок 2.6 – Середній час відгуку вивантажених завдань при кількості туманних вузлів 20

На рисунку 2.7 показано середній час відгуку розвантажених завдань під час ітерацій одного запуску запропонованого алгоритму з такими налаштуваннями: c=3 , λ_y=10 для c=1, λ_y=5 для c=2, λ_y=1 для c=3, N_{nod}=20 , μ_{ky}=200 для c=1, μ_{ij}=50 для c=3, та μ_{ij}=10 для c=4 . На кожній ітерації записується середній час відгуку всіх мурах. Збіжність запропонованого алгоритму ACO відбувається швидше, ніж у алгоритму PSO: найкраще значення досягається одразу після ітерації номер 27. Цей результат свідчить про те, що запропонований алгоритм ACO досліджує простір пошуку можливих рішень і досягає найкращого значення за розумний проміжок часу порівняно з алгоритмом PSO.

У наступному тестовому сценарії три різні задачі з датчиками, що належать до трьох різних класів застосунків, c=3 . Кожна задача має різну швидкість передачі даних λ_y=10 для c=2, λ_y=20 для c=3, λ_i=40 для c=4.

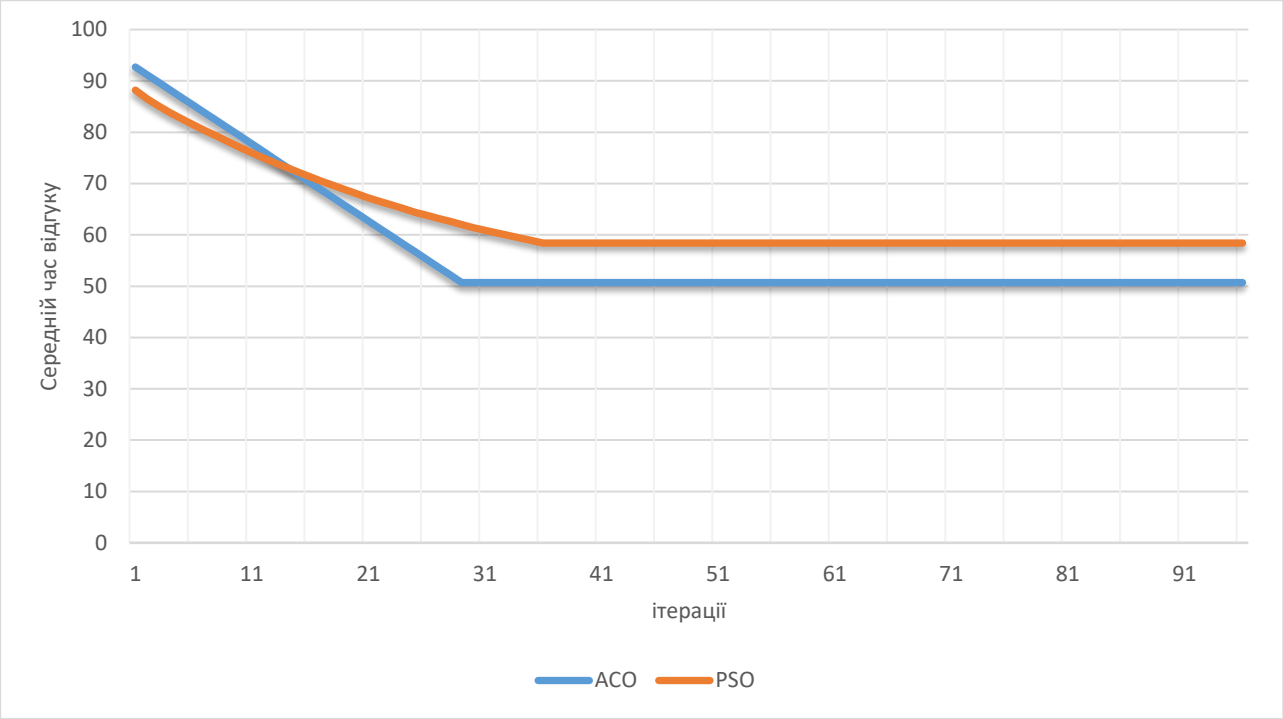


Рисунок 2.7 – Середній час відповіді за ітерацію

Кількість вузлів туману N_{nod}=10. Швидкість обслуговування для кожної задачі становить μ_{ky}=150 для c=2, μ_{ky}=100 для c=3 і μ_{ky}=50 для c=4.

На рисунку 2.8 показано, що запропонований алгоритм розвантаження ACO підтримує менший час відгуку при збільшенні кількості датчиків IoT порівняно з алгоритмом PSO та RR. У цьому сценарії і RR, і PSO порушують встановлену допустиму затримку. Цей сценарій показує, що запропонований алгоритм розвантаження завдань ACO здатний прийняти найкраще рішення щодо розвантаження порівняно з запропонованим алгоритмом PSO.

На рисунку 2.9 показано середньоквадратичне відхилення часу відгуку при збільшенні кількості вузлів IoT з використанням попередніх налаштувань параметрів. Середньоквадратичне відхилення часу відгуку - це відхилення часу відгуку всіх розвантажених завдань від середнього часу відгуку.

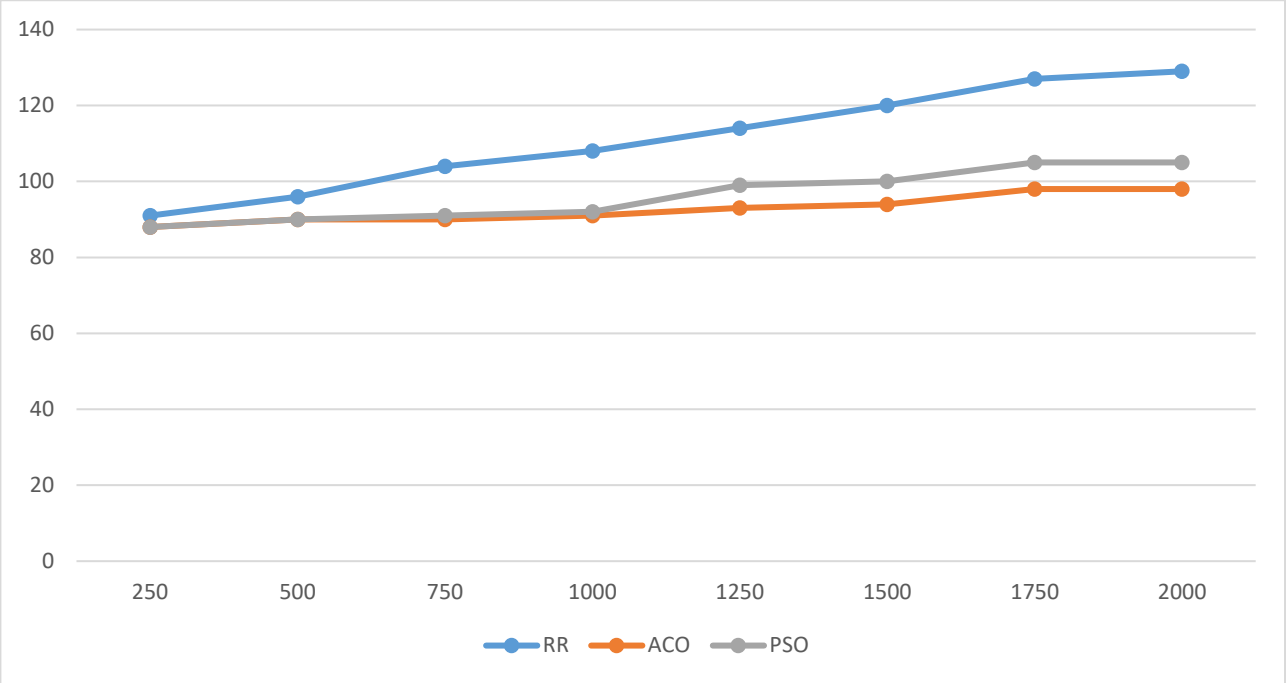


Рисунок 2.8 – Середній час відгуку для трьох різних класів задач

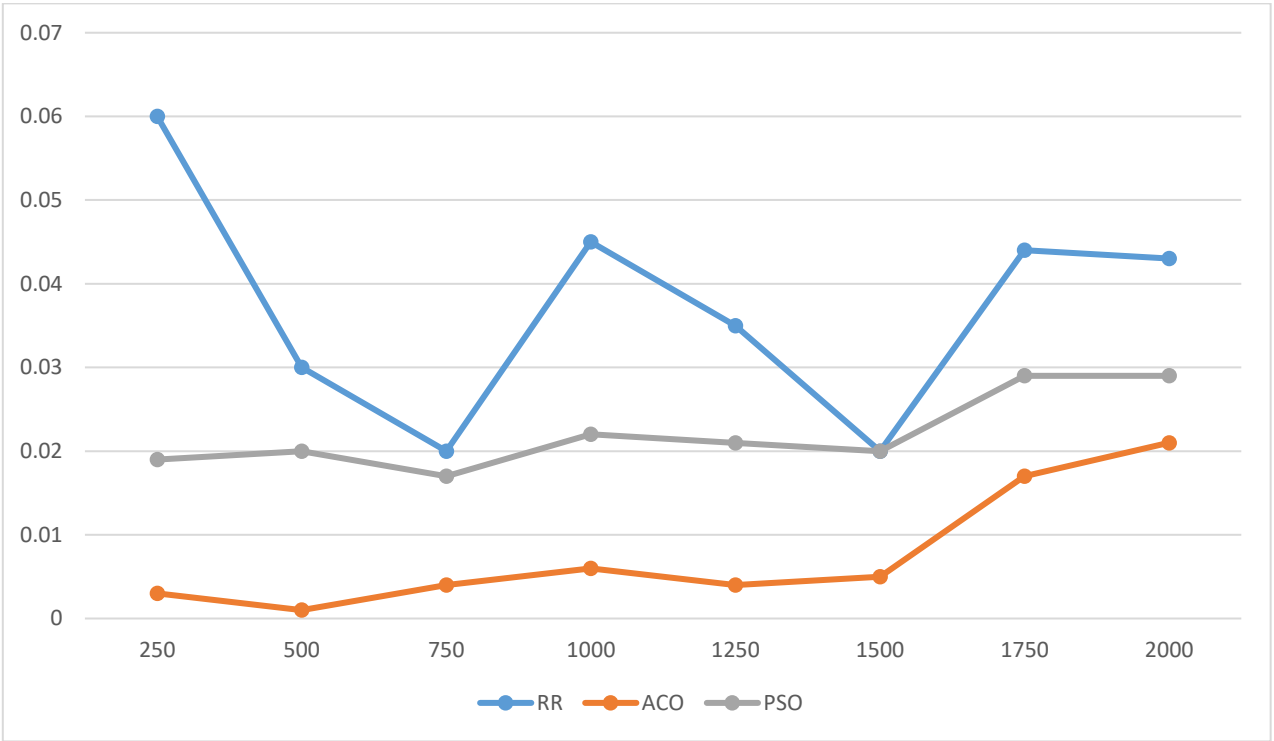


Рисунок 2.9 – Середньоквадратичне відхилення часу відгуку при збільшенні кількості вузлів IoT

На рисунку чітко видно, що запропонований алгоритм розвантаження завдань ACO збільшив середньоквадратичне відхилення порівняно з алгоритмами RR.

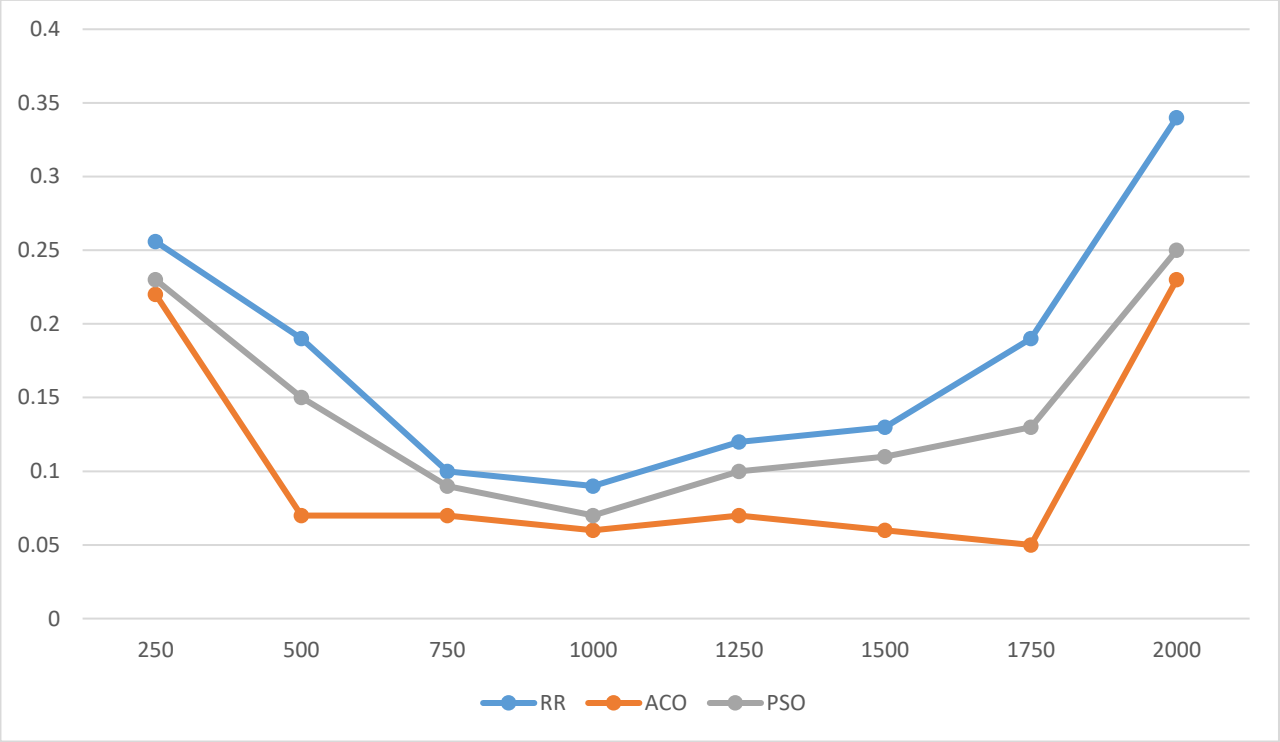


Рисунок 2.10 – Ступінь дизбалансу для алгоритмів RR, ACO, PSO

На рисунку 2.10 показано ступінь дисбалансу для алгоритмів RR та запропонованого алгоритму зі збільшенням кількості вузлів IoT з використанням тих самих попередніх налаштувань параметрів. З рисунку видно, що запропонований алгоритм ACO підтримує менші значення. Це означає, що запропонований алгоритм ефективно балансує навантаження на вузли туману.

2.4 Висновки

В розділі було проаналізовано сучасні методи оптимізації функціонування IoT інфраструктури із застосуванням туманних обчислень. Було визначено такі методи оптимізації туманних обчислень:

- мінімізація затримок;

- розподілення навантаження;
- оптимальне розміщення пристроїв.

Було досліджено та описано засоби оптимізації планування завдань в IoT інфраструктурі із застосуванням туманних обчислень. Було описано основні алгоритми які використовуються для оптимізації планування завдань. Також було проведено порівняння алгоритмів планування завдань:

- ACO;
- PSO;
- RR.

За результатами порівняння було визначено, що із використанням алгоритму мурашиної колонії (ACO) досягається нижчих показників часу затримки та часу відповіді. Окрім того, алгоритм ACO демонструє найнижчий ступінь дизбалансу та середньоквадратичного відхилення. Оскільки алгоритм мурашиної колонії демонструє найнижчий показник середньоквадратичного відхилення при збільшення кількості вузлів, він є найбільш вдалим вибором із розрахунку на подальше масштабування інфраструктури.

Було вирішено проводити вдосконалення методу оптимізації планування завдань в IoT інфраструктурі із застосуванням туманних обчислень на основі алгоритму мурашиної колонії.

3 МЕТОД ОПТИМІЗАЦІЇ ЗАТРИМКИ В IOT СИСТЕМІ З ВИКОРИСТАННЯМ ТУМАННИХ ОБЧИСЛЕНЬ

3.1 Пріоритетність виконання запланованих завдань

З метою оптимізації затримки в IoT системі із використанням туманних обчислень було удосконалено метод планування завдань, що базується на алгоритмі ACO, шляхом впровадження пріоритету виконання для різних класів завдань в системі.

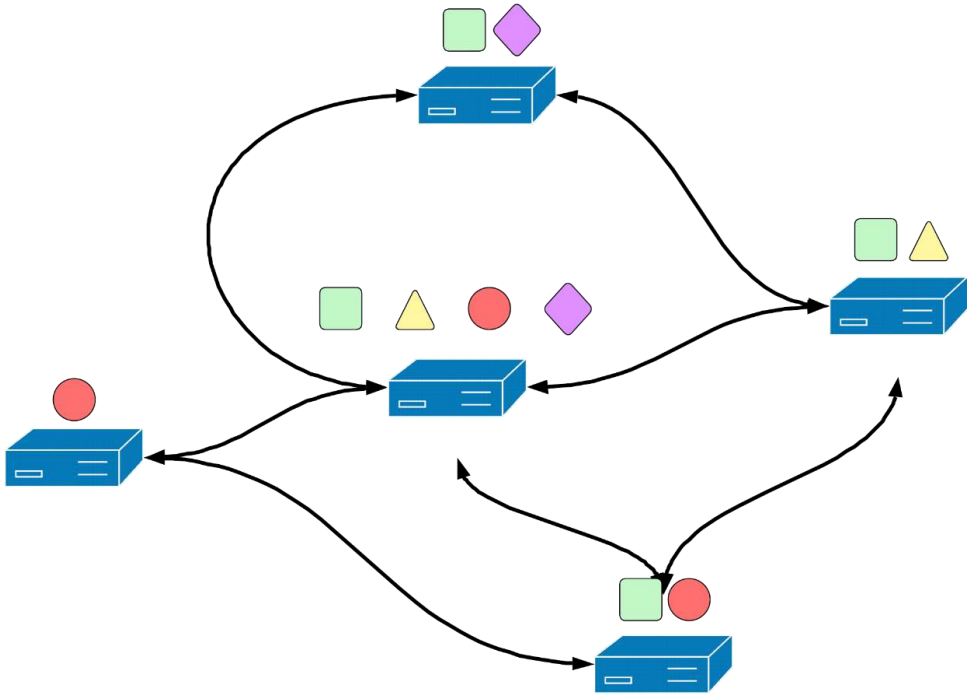


Рисунок 3.1 – З’єднання туманних вузлів за класами завдань

Також, було визначено основні кроки удосконаленого методу оптимізації IoT інфраструктури із застосуваннями концепції туманних обчислень:

1. Визначення класів завдань які можуть бути опрацьовані вузлами.
2. Розподілення пріоритетів між класами завдань в межах кожного із вузлів туману.
3. Визначення затримки відповіді вузлів за допомогою використання алгоритму ACO.

4. Корегування результатів визначення ефективності вузлів із допомогою використання пріоритетів класів завдань та актуального навантаження вузлів туману.

В роботі використано алгоритм ACO який показав найкращі результати в оптимізації зменшення часу відгуку при планування завдань. Із метою зменшення середнього часу опрацювання завдань було вирішено додати умову пріоритетності завдань на вузлах туману. Таким чином, вузол туману буде зосереджувати обчислювальні потужності на певних класах завдань та відповідно до цього обробляти більшу їх кількість за одиниць часу. На рисунку 3.1 показано мережу туманних вузлів, що з'єднані за класом завдань. Класи завдань позначені геометричними фігурами. Таким чином оптимізація буде відбуватись за класом завдань. Звідси випливає, шо замість загальної кількості туманних вузлів пошук найшвидшого буде відбуватись серед набору вузлів, що виконують конкретний клас або класи завдань.

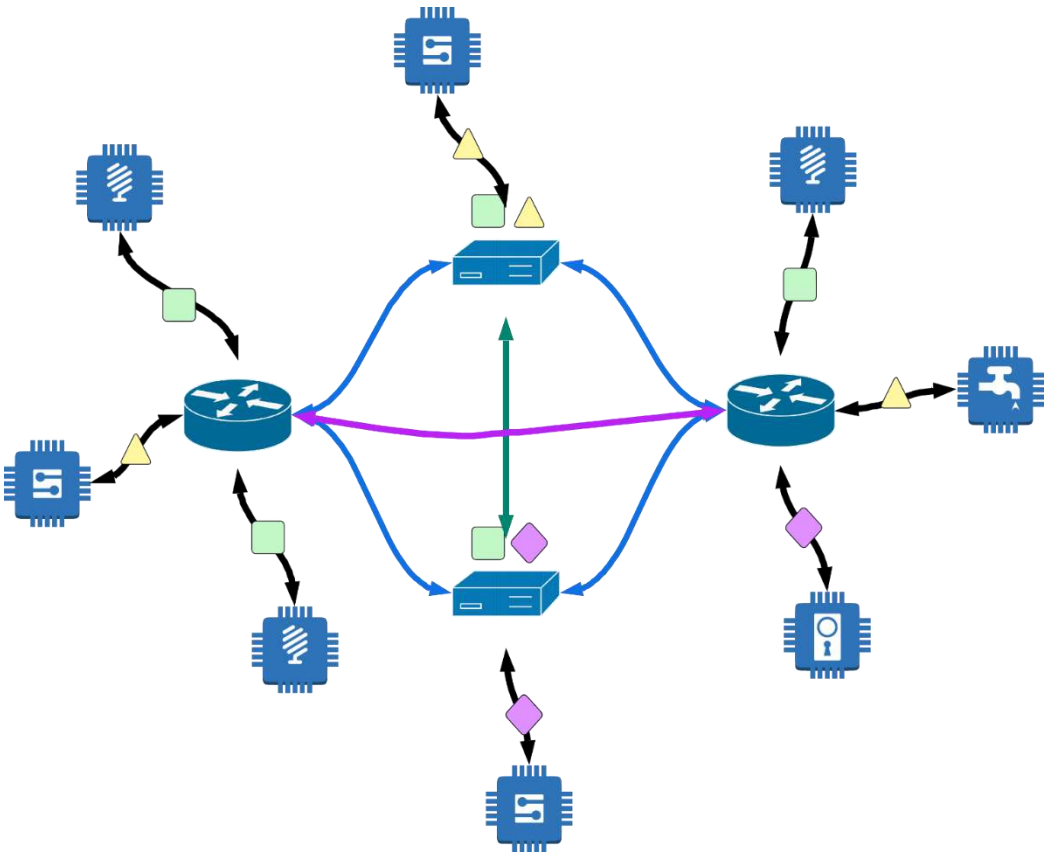


Рисунок 3.2 – Архітектура шлюзів та IoT пристроїв у системі

Відповідне гетерогенне середовище вимагає того, що в архітектурі повинні бути присутні шлюзи із компенсаторами навантаження. Такі шлюзи будуть виступати в ролі планувальників завдань. Оскільки, кожен шлюз буде мати інформацію про кожен вузол туману, його класи завдань, навантаження та час відповіді, він може ефективно планувати розподіленням завдань.

На рисунку 3.2 зображено архітектуру у якій шлюзи виступають у ролі планувальників завдань. Вони отримують завдання від IoT пристроїв та планують завдання, в залежності від її класу, використовуючи мережеву таблицю затримок відповідей туманної мережі.

Таблиця 3.1 – Мережева таблиця затримок відповідей

N_i	S_j	L_{ij}
1	A	12
1	B	34
1	C	78
1	D	40
2	C	37
2	D	140
3	E	22

З таблиці 3.1 N_i – вузол туману, S_j – клас завдання, L_{ij} – затримка відповіді виконання вузлом N_i завдання класу S_j. Можна побачити, що вузол 1 та вузол 2 виконують задачі C та D. Проте, вузол 1 повинен мати більший пріоритет для виконання задачі D, а вузол 2 повинен мати більший пріоритет для виконання задачі C. Відповідно до цього маємо формулу 3.1:

$$N_{min} = Min\big(L(N_i, S_j)\big), \tag{3.1}$$

де N_{min} – вузол із найменшою затримкою відповіді.

Датчики можуть генерувати велику кількість даних із високою частотою, в такому разі можливі випадки у яких усі вузли туману та хмарне середовище перенавантажені. У такому разі можлива втрата даних, отриманих від датчиків. Часто така втрата даних може бути не критичною, проте є сфери використання IoT у яких низька затримка є основною вимогою від системи. Для такого випадку пропонується ввести пріоритетність виконання класів задач.

Візьмемо до уваги вузол, що опрацьовує задачі класу A та B. У випадку, якщо виконання задачі класу A займає 40 мс, а класу B – 500 мс, за час виконання задачі класу B можна виконати 12.5 задач класу A. В такому випадку можна накласти вимогу якомога нижчої затримки задач класу A та невисоку важливість виконання задач класу B. Реалізація такої поведінки вимагає евристики, яка буде визначати, що у певного класу задач є пріоритетність над іншими у випадку, перевантаження задачами.

Така евристика може бути досягнена за допомогою впровадження таблиці пріоритетів задач.

Таблиця 3.2 – Пріоритети завдань

N_i	S_j	P_s
1	A	50
1	B	25
1	C	10
1	D	15
2	C	80
2	D	20
3	E	100

Із даних з таблиці 3.2 можемо вивести формулу (3.2), за якою вираховується сума пріоритетів.

$$\sum P\left(N_i, S_j\right) = 100\%. \quad (3.2)$$

Із даних у формулі 3.2 можна ввести поняття як цикл пріоритетів. Цикл пріоритетів – це цикл планування задач, який буде задовольняти формулу 3.2, тобто за одне виконання циклу буде виконуватись 100% задач по запланованих пріоритетах.

На рисунку 3.3 зображено блок схему алгоритму планування задач. Блок-схема демонструє алгоритм планування задач в межах одного вузла, який бере до обробки усі завдання в порядку надходження, допоки не буде вичерпано квоту пріоритетності для класу задачі. Як тільки квоти усіх класів задач буде вичерпано лічильники оновляться і розпочнеться наступний цикл опрацювання задач. Якщо ж до вузла надходять лише задачі одного класу, планувальник продовжить їх обробляти, оновлюючи лічильник, до появи нових задач інших класів.

Для планування задач використовується глобальні значення таблиці пріоритетів, що зображені у таблиці 3.2 та таблиці із локальними для туманного вузла значеннями, що містить лічильники виконання класів задач. Лічильники використовуються із метою балансування виконання задач, інакше задачі, що генеруються із великою частотою можуть перевантажити туманний вузол, що призведе до затримки виконання задачі. Така поведінка може призвести до втрати актуальності даними які були опрацьовані із затримкою.

Із метою визначення планувальником вузла із найменшою затримкою необхідно вести лічильник класів задач які були опрацьовані вузлом. Для досягнення цього доповнимо таблицю 3.1 даними із внутрішнього циклу опрацювання задач вузла. Окрім цього необхідно ввести лічильник отримання вузлом задач для збереження інформації не лише про навантаження вузла по класу задач, а і загальне навантаження вузла задачами.

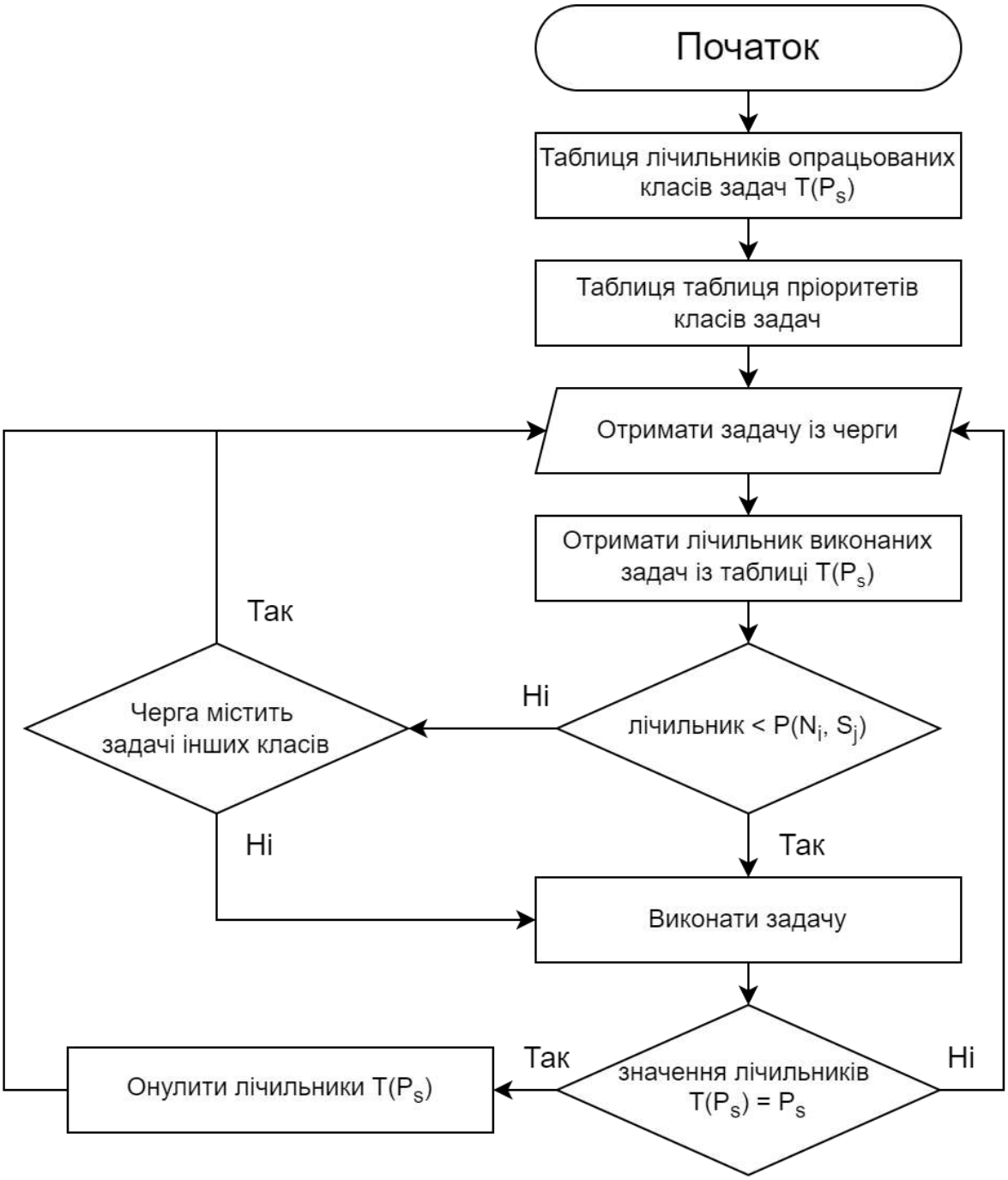


Рисунок 3.3 – Алгоритм планування задач з використанням таблиці пріоритетів

З метою визначення загального навантаження вузла задачами введемо формулу 3.3, яка буде враховувати як кількість виконаних задач, так і складність їх опрацювання.

$$n(P_i) = \sum \frac{\sum P(N_i,S_j)\,[t(S_j) \geq t_{current} - t(n)]}{t(n) \times L_{ij}}, \tag{3.3}$$

де $n(P_i)$ – загальне навантаження вузла,

n = вікно часу, у якому ведуться обрахунки, яке необхідне для того, щоб тримати в пам’яті лише актуальні дані про опрацювання задач,

$t(S_j)$ – час коли задача була виконана.

Формула 3.3 обраховує навантаження вузла яке не залежить від класу задачі та враховує затримку виконання задачі. Звідси отримає нову мережеву таблицю 3.3, яка буде відображати повну інформацію про стан роботи вузла туманного середовища.

Таблиця 3.3 – Мережева таблиця пріоритетів із загальним навантаженням вузлів

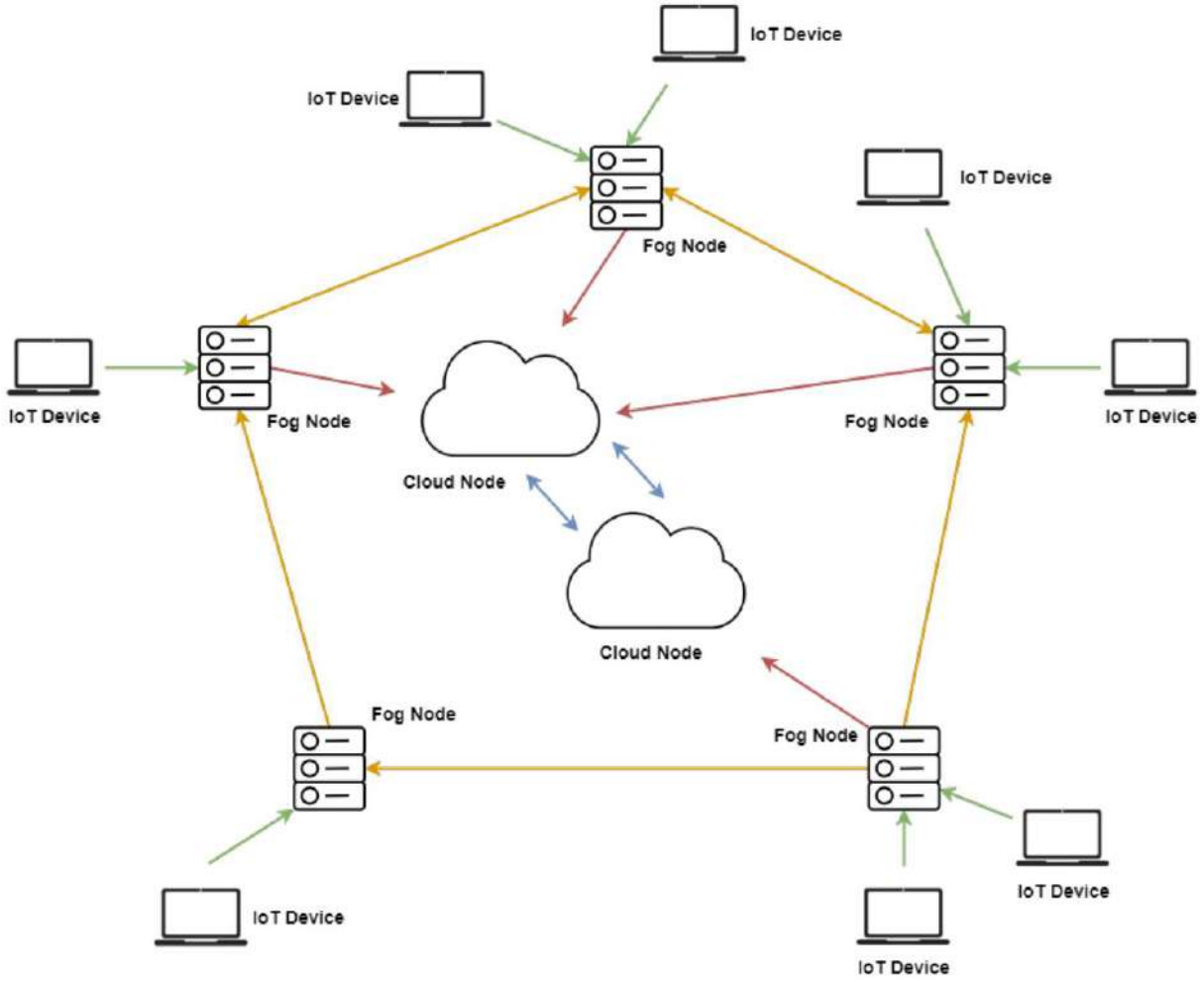
N_i	S_j	L_{ij}	P_s	$P(N_i, S_j)$	$n(P_i)$
1	A	12	50	11	23.03
1	B	34	25	3	
1	C	78	10	6	
1	D	40	15	12	
2	C	37	80	40	36.4
2	D	140	20	7	
3	E	22	100	98	74

Використовуючи таблицю 3.3 планувальник задач зможе враховувати загальне навантаження вузлів. Значення таблиці 3.3 вирахувані з використанням формули 3.3, де за значення n виступає 60 секунд та значення лічильників $P(N_i, S_j)$ були згенеровані в межах часового вікна n. Звідси за формулою 3.3 випливають значення $n(P_i)$, що показують загальне навантаження вузла N_i.

Слід зазначити, що для формули 3.3 значення $n(P_i)$ буде наближатися до 1 коли використання ресурсів вузла буде близьким до максимального. Також, коли значення $n(P_i)$ будуть перевищувати 1 буде означати, що вузол має високу обчислювальну потужність. У випадку якщо значення значно менше від 1 означає, що вузол вільний для навантаження новими задачами.

3.2. Середовище проведення експериментальних досліджень роботи інфраструктури IoT із використанням туманних обчислень

Для проведення експериментальних досліджень роботи IoT інфраструктури було використано середовище моделювання IoT систем – DISSECT-CF-Fog. DISSECT-CF-Fog – програмне середовище із відкритим вихідним кодом. Середовище пропонує функціонал детального моделювання системи у якій є можливість керувати різним її параметрами, що дозволяє налаштувати середовище близьке до реального світу.



- Рисунок 3.4 – Архітектура IoT мережі із використанням туманних обчислень побудована у середовищі DISSECT-CF-Fog [85]

Інструмент моделювання надає можливість використання різних алгоритмів планування задач, беручи до уваги використання енергії, географічне розташування пристроїв IoT середовища та ціну ресурсів хмарного середовища, що близька до реальних хмарних сервісів.

DISSECT-CF – є фреймворком для моделювання дискретних подій (DES), тому він підтримує прийняття рішень для складних сценаріїв, пов'язаних з часом. Програмне забезпечення здатне моделювати внутрішні процеси розподілених систем, підтримуючи паралельні та розподілені обчислення, незважаючи на послідовне виконання. На рисунку 4.4 зображено схему IoT мережі яку можна побудувати із використанням середовища моделювання.

Розширення DISSECT-CF-Fog, що використовується для моделювання туманної мережі складається із основних компонентів:

- application – компонент, що забезпечує механізм планування задач, який зважає на завантаженість вузлів, та розподіляє між ними задачі. Характеристики компонента зображені у таблиці 3.4;
- microcontroller – модуль, що дозволяє контролювати використання енергії в системі. Із допомогою цього модулю можна визначити періоди максимального та мінімального використання енергії, для визначення перевантаження системи або простоювання. Характеристики пристрою зображені у таблиці 3.5;
- device – пристрій IoT, що генерує дані. Його характеристики налаштувань зображено у таблиці 3.6.

Таблиця 3.4 - Параметри компоненту applicaton

Параметр	Опис
Frequency	Інтервал часу між двома розподілами даних (завдань) на віртуальні машини
Task Size	Вхідні дані організовуються в задачі, які мають заздалегідь визначений розмір

Продовження таблиці 3.4 - Параметри компоненту applicaton

Instance	Ресурси віртуальної машини
Instruction Count	Максимальна кількість інструкцій, виконаних на віртуальній машині, яку може представляти повністю завантажене завдання
Threshold	Вище цього значення нерозподілені завдання можуть бути перенаправлені на інший вузол
Application Strategy	Політика, яка визначає, на який вузол пересилати нерозподілені завдання
IoT Price	Вартість пристрою, який підключено до цього застосунку

Таблиця 3.5 – Параметри компоненту microcontroller

Параметр	Опис
CPU	Кількість ядер CPU
Memory	Кількість пам’яті мікроконтроллера
Processing Power	Обчислювальні можливості одного ядра процесора, що вимірюються в інструкціях за такт
Repository	Характеристика накопичувача даних
Turn On Process	Кількість інструкцій для імітації процесу ввімкнення
Turn Off Process	Кількість інструкцій для імітації процесу вимкнення
Minimum Power	Потужність енергії, коли пристрій повністю вимкнений, але підключений до джерела енергії.
Idle Power	Потужність пристрою, коли він увімкнений, але не виконується жодне активне завдання.
Maximum Power	Енергетична потужність, коли його процесор повністю завантажений.

Таблиця 3.6 – Параметри компоненту device

Параметр	Опис
Start Time	Коли IoT-пристрій починає працювати
Stop Time	Коли IoT-пристрій припиняє роботу
File Size	Розмір одного вимірювання
Sensor Count	Кількість датчиків, які має поточний IoT-пристрій
Device Strategy	Політика для визначення, з яким IoT-застосунком поточний IoT-пристрій зв'язується
Frequency	Інтервал часу між двома вимірюваннями датчиків
Latitude	Фізичне положення об'єкта
Longitude	Фізичне положення об'єкта
Microcontroller	Посилання на мікроконтроллер
Actuator	Відображає програмний або апаратний об'єкт, розташований поруч з цим IoT-пристроєм
Latency	Затримка для пакету даних, що подорожує мережею
Sensor Frequency	Тривалість одного вимірювання датчика
Mobility Strategy	Політика, яка описує переміщення пристрою

Середовище моделювання складається із чотирьох модулів:

- веб застосунок(Angular);
- серверний застосунок(Node.js);
- база даних(MongoDB);
- серверний застосунок моделювання(Java).

3.3 Висновки

У розділі було проведено удосконалення методу та засобів планування завдань в IoT інфраструктурі із застосуванням туманних обчислень. Із метою удосконалення було вибрано метод планування завдань на основі алгоритму мурашиної колонії (ACO).

Було введено поняття пріоритету виконання завдань вузлами туману. Також було розроблено формули для визначення вузла, що проведе обрахунки із найменшим часом та затримкою.

Також було розглянуто середовище проведення експериментальних досліджень DISSECT-CF. Було описано його модулі, абстаркції та їх налаштування.

4. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ УДОСКОНАЛЕНОГО МЕТОДУ ОПТИМІЗАЦІЇ ЗАТРИМОК ОПРАЦЮВАННЯ ЗАВДАНЬ В IOT ІНФРАСТРУКТУРІ ІЗ ВИКОРИСТАННЯМ ТУМАННИХ ОБЧИСЛЕНЬ

4.1 Налаштування середовища експериментальних досліджень IoT системи із використанням туманних обчислень за допомогою інструменту моделювання DISSECT-CF

Із метою реалізації засобів та удосконалення методу планування завдань в IoT інфраструктурі із застосуванням туманних обчислень було використано середовище моделювання DISSECT-CF яке надає функціонал побудови IoT системи із використанням туманних обчислень.

Інструмент моделювання DISSECT-CF дозволяє збирати інформацію про використання енергії вузлами туманної мережі та хмарного середовища.

Окрім цього, можна отримати інформацію про навантаження вузлів туманної мережі у різний час.

Для проведення експериментального дослідження було використано п’ять вузлів туманного середовища та один екземпляр розгорнутий у хмарі. Розташування вузлів зображено на рисунку 4.1. Також, на рисунку зображено маршрути пристроїв. Маршрути пристроїв були побудовані за принципом того, що один пристрій повинен бути охоплений декількома туманними вузлами.

Таблиця 4.1 – Параметри налаштувань вузлів мережі туману

Назва параметру	Значення
ОЗП	4 гб
Центральний процесор	1 ядро
Обчислювальна потужність	0.001 інструкцій/мс
Процес запуску	100 інструкцій
Твердотільний накопичувач	1 гб

Окрім цього було побудовано маршрут у якому на певних ділянках пристрій не буде охоплений жодним із туманних вузлів. У таблиці 4.1 зображено параметри вузлів мережі туману. Таблиця 4.2. містить опис параметрів хмарного середовища.

Таблиця 4.2 – Параметри налаштувань хмарного середовища

Назва параметру	Значення
ОЗП	8 гб
Центральний процесор	2 ядра
Обчислювальна потужність	0.002 інструкцій/мс
Процес запуску	100 інструкцій
Твердотільний накопичувач	50 гб

Далі необхідно налаштувати параметри програмного середовища, що буде опрацьовувати задачі.

Таблиця 4.3 – Параметри програмного середовища

Назва параметру	Значення
Розмір задачі	50000 байт
Частота	60000 мс
Кількість інструкцій	1000
Розподілення передачі даних	2

Параметри програмного середовища, що розгорнуте на вузлах мережі туману наведені у таблиці 4.3, де:

- розмір задачі - атрибут розміру завдання вказує на максимальний обсяг необроблених даних, який може бути упакований в одне обчислювальне завдання для виконання віртуальними машинами. Інструкції середовища моделювання рекомендують вхідні значення параметру більше 5000 байт;

- частота – на основі значення частоти служба-демон перевіряє сховище на наявність необроблених даних. За інструкціями середовища моделювання значення повинно бути більшим від 6000 мілісекунд;
- кількість інструкцій – параметр, що визначає максимальне значення яке може представляти одне завдання. Рекомендоване значення більше 1000;
- розподілення передачі даних – визначає кількість необроблених завдань може утримуватись в реальному застосунку, наступні завдання будуть пересилатись відповідно до одної із стратегій застосунку. Рекомендоване значення 1-5.

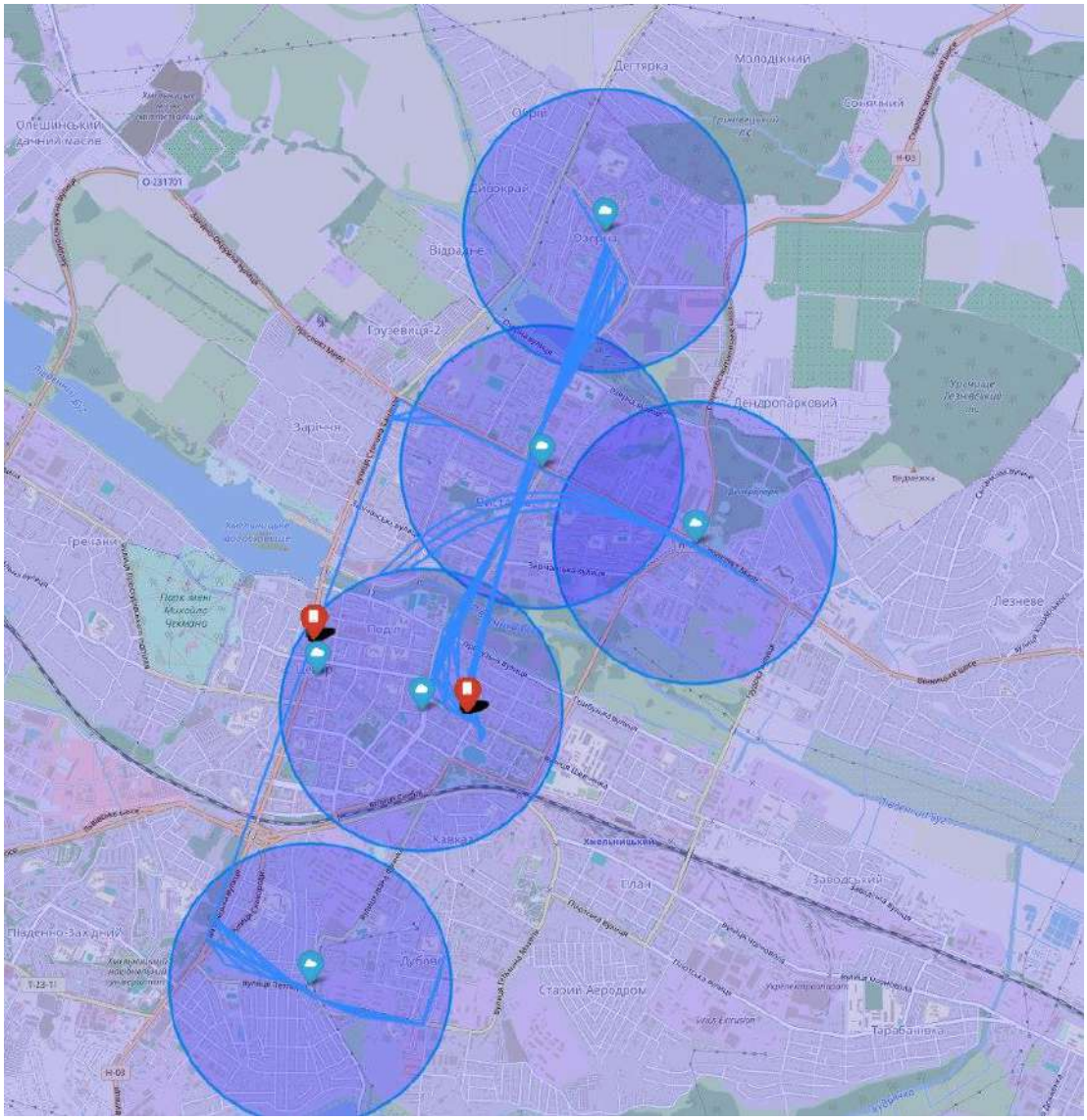


Рисунок 4.1 – Розташування вузлів мережі туману та маршрути пристроїв

. За допомогою функціоналу моделювання DISSECT-CF-Fog було згенеровано візуалізацію мережі на карті, зображену на рисунку 4.1.

Середовище моделювання має набір вбудованих стратегії розподілення задач у випадку коли кількість необроблених задач перевищить значення розподілення передачі даних:

- Random – вибирає вузол туману випадковим чином.
- Push Up – завжди вибирає батьківський вузол до якого приєднано вузол.
- Hold Down – відсилає задачу до вузла, що є найближчим до кінцевого користувача.
- Runtime – обирає вузол із найменшим значенням затримки відповіді, доступними ресурсами центрального процесору та загальними характеристиками загального процесора.
- Pliant – стратегія що обирає вузол за навантаженням вузла та ціною опрацювання даних.
- Стратегія Runtime реалізовує алгоритм мурашиної колонії (ACO) для визначення вузлів із найменшою затримкою та алгоритм рою часток (PSO) для визначення вузла із найбільш оптимальним навантаженням. Тому вона підходить для модифікації та використання разом із чергою пріоритетів. Таким чином алгоритм визначення вузла для опрацювання задачі буде мати вигляд:
- визначення затримок відповіді вузлів за алгоритмом мурашиної колонії(ACO);
- визначення доступних ресурсів за врахуванням енергетичних профілів вузлів за алгоритмом рою часток(PSO);
- визначення навантаження вузлів за формулою 3.3;
- визначення задачі, що буде виконуватись за розробленим алгоритмом планування задач(рис 3.3).

Таблиця 4.4 – Параметри кінцевих IoT пристроїв

Назва параметру	Значення
Розмір файлу	50 байт
Кількість сенсорів	1
Частота	60000 мс
Пропускна здатність	3250 байт\мс
Затримка	50 мс
Швидкість пересування	0.0025 м\с
Кількість ядер процесора	1
Обчислювальна потужність	0.001 інструкцій\мс
ОЗП	1 гб
Мінімальне використання енергії	0.025 ват
Використання енергії в стані спокою	0.155 ват
Максимальне використання енергії	0.225
Розмір твердотільного накопичувача	1 гб

Для тестування навантаження мережі вузлів туману використано 200 кінцевих пристроїв, що безперервно рухаються по маршрутах та надсилають данні із датчиків. У таблиці 4.4 наведено основні характеристики кінцевих пристроїв

4.3 Результати експериментальних досліджень фуфнкціонування IoT мережі із використанням концепції туманних обчислень за допомогою інструменту моделювання DISSECT-CF-Fog

Середовище моделювання DISSECT-CF-Fog має вбудований функціонал генерування графіків із даних про функціонування системи:

- використання енергії;

- вартість обчислювальних потужностей на популярних хмарних середовищах;
- використання ОЗП;
- кількість виконаних та запланованих задач;
- використання робочого часу процесорів вузлів;
- кількість оброблених даних.
- Для порівняння результатів було використано дві моделі, одна з яких базується на розробленому методі планування задач в IoT інфраструктурі із застосуванням туманних обчислень, друга базується на основі пошуку найближчого пристрою за стратегією планування Hold Down – вбудованої стратегії планування в середовищі DISSECT-CF-Fog.

4.3.1 Планування задач

На рисунку 4.2 зображено графік порівняння планування задач. Графік планування задач показує кількість запланованих задач в певних проміжках часу.

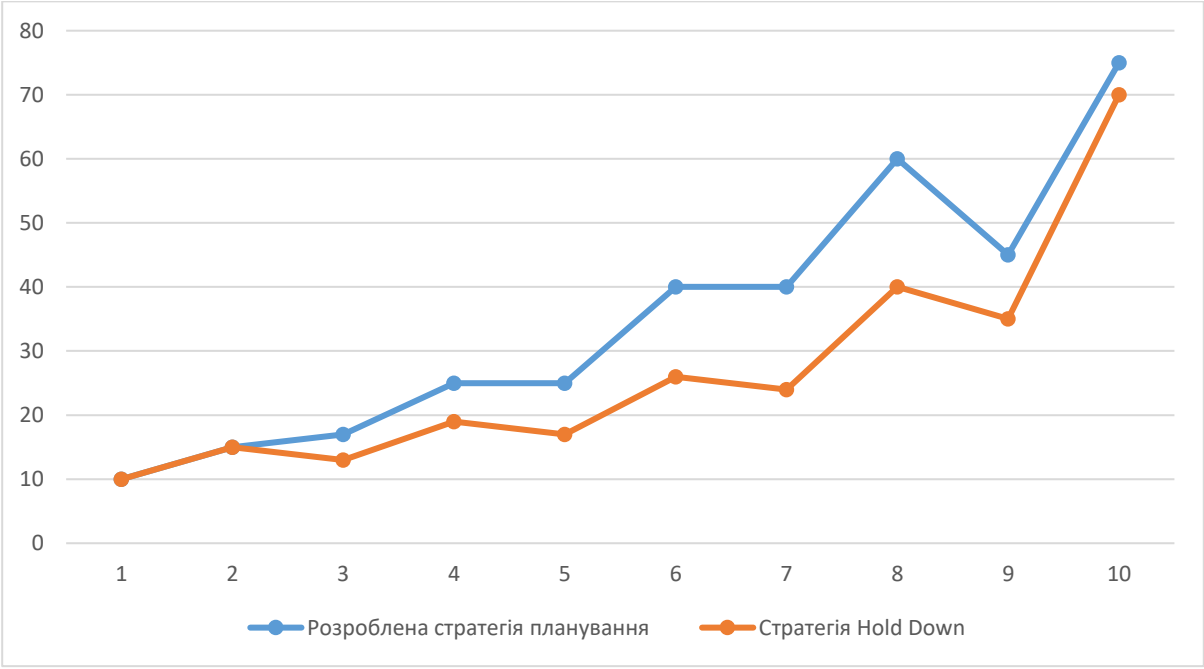


Рисунок 4.2 – Графік планування задач

Крива блакитного кольору, що відображає кількість запланованих задач для розробленої стратегії планування має розрив у кількості запланованих задач стратегією Hold Down.

Такий результат спричинений деталями роботи розробленого алгоритму на основі пріоритетів задач. На продовженій шкалі буде видно, що така поведінка циклічна. Шкала синього кольору буде віддалятись та наближатись до шкали оранжевого кольору.

Алгоритм планує задачі в порядку надходження, проте коли досягається максимальне значення лічильника для класу задачі, алгоритм починає планувати задачі які є більш пріоритетними.

У випадку цієї моделі пріоритет мають задачі із меншим часом опрацювання, тому шкала для розробленої стратегії демонструє більшу кількість запланованих задач, але у випадку якщо пріоритет будуть мати задачі, що мають більший час обробки результат буде протилежним.

Як висновок можна зазначити, що кращий результат буде досягатись у випадку якщо більший пріоритет виконання будуть мати задачі, на опрацювання яких витрачається менше часу.

4.3.2 Навантаження центрального процесора

За результатом виконання експерименту було побудовано графік використання енергії центральними процесорами вузлів туману.

На рисунку 4.3 продемонстровано отримані значення сумарного споживання енергії процесорами вузлів туману відповідно до розробленої стратегії планування та стратегії Hold Down.

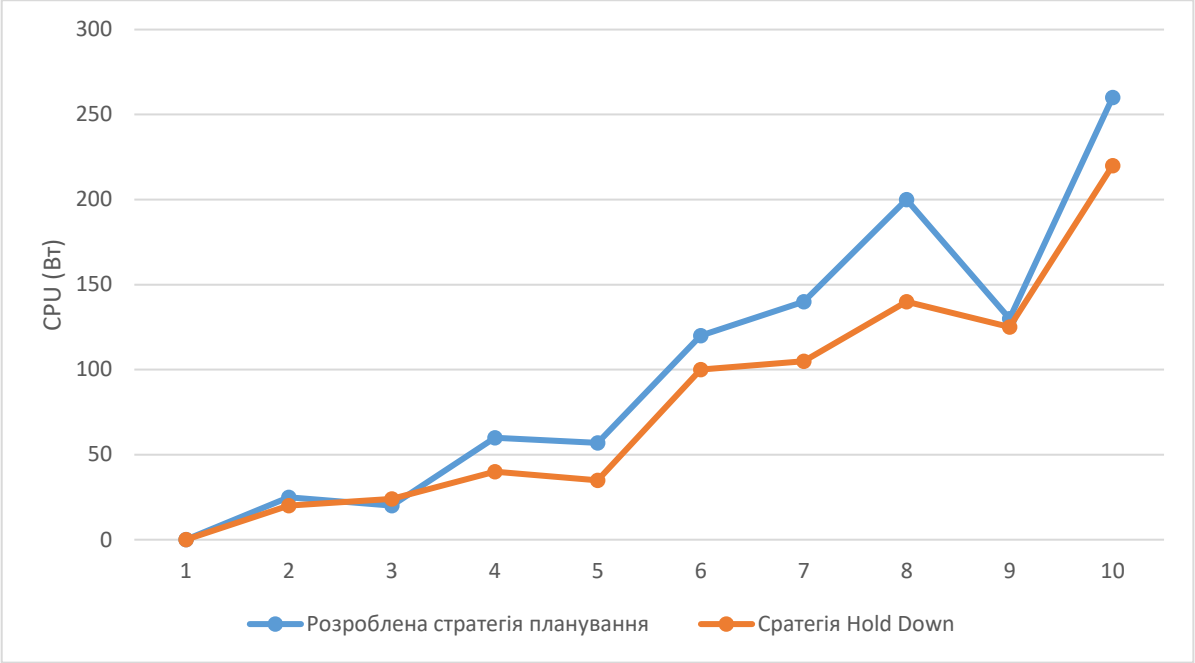


Рисунок 4.3 – Сумарне споживання електроенергії процесорами вузлів туману

Рисунок 4.3 повністю відображає поведінку розробленої стратегії планування, в якій починає роботу ітераційний процес алгоритму планування задач за пріоритетністю.

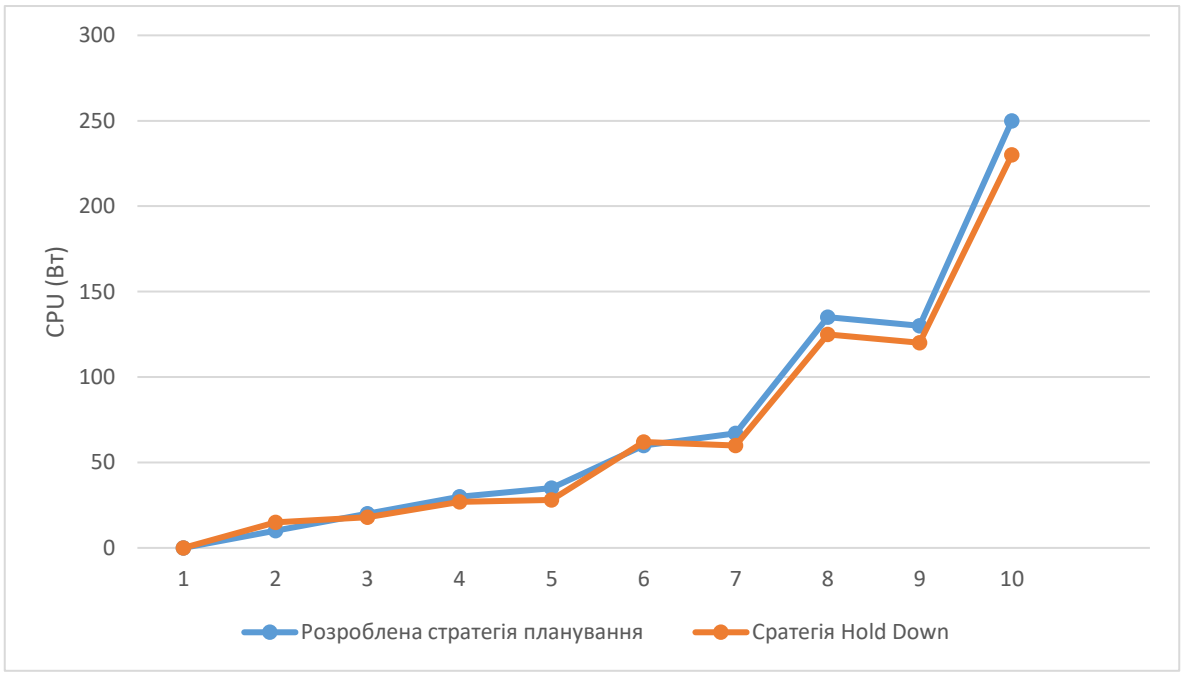


Рисунок 4.4 – Споживання енергії з пріоритетом на важкі задачі

Аналогічно до рисунку 4.1 значення споживання енергії процесорами вузлів для розробленої стратегії планування є більшими від стратегії Hold Down. Отриманий показник є результатом того, що алгоритм почав виконувати більше задач із високим пріоритетом та низьким часом оброки.

На відміну від Графіку планування задач, якщо пріоритет будуть мати задачі із більшим часом виконання результати значення споживання енергії із використанням розробленої стратегії планування набувають близьких значень до стратегії Hold Down, що зображено на рисунку 4.4.

4.3.3 Використання оперативної пам’яті

Також було згенеровано графік споживання ОЗП вузлами туману. Значення на графіку показують, що вузли в яких перетинаються маршрути пересування кінцевих пристроїв навантажені більше аніж решта вузлів системи. Статистику використання ОЗП зображено на рисунку 4.5.

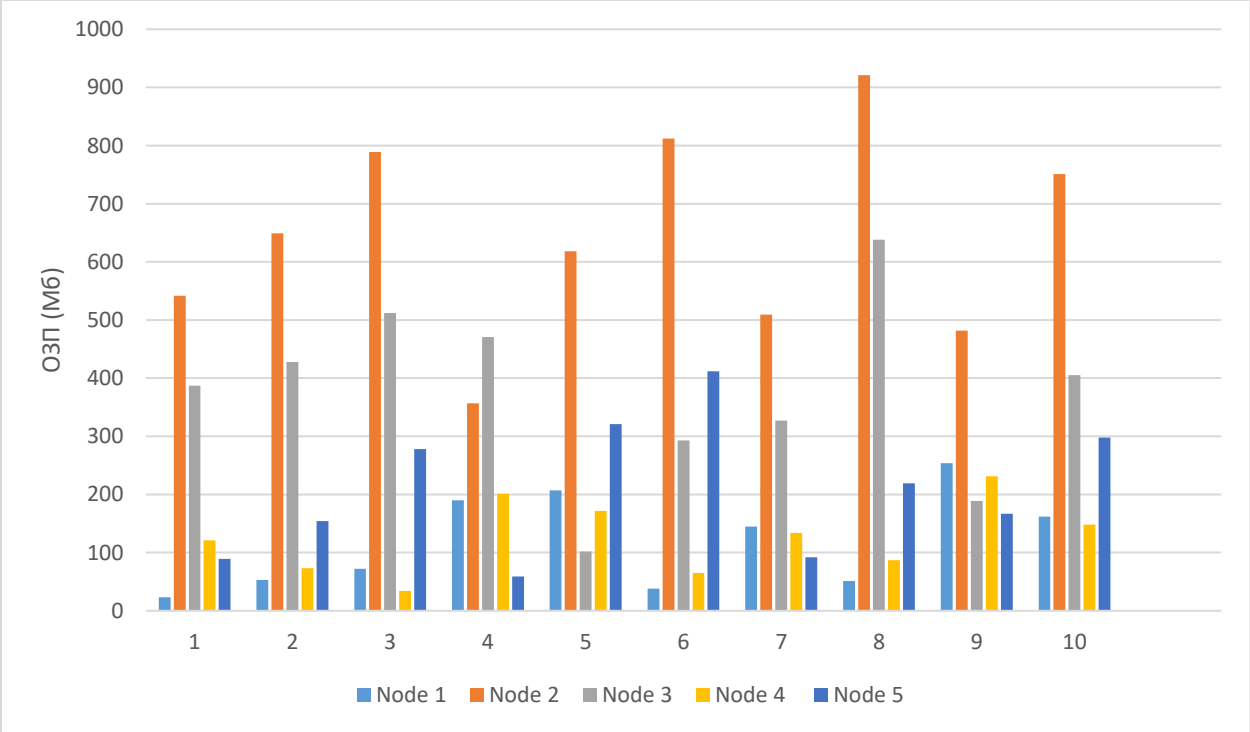


Рисунок 4.5 – Загальне споживання ОЗП вузлами туману

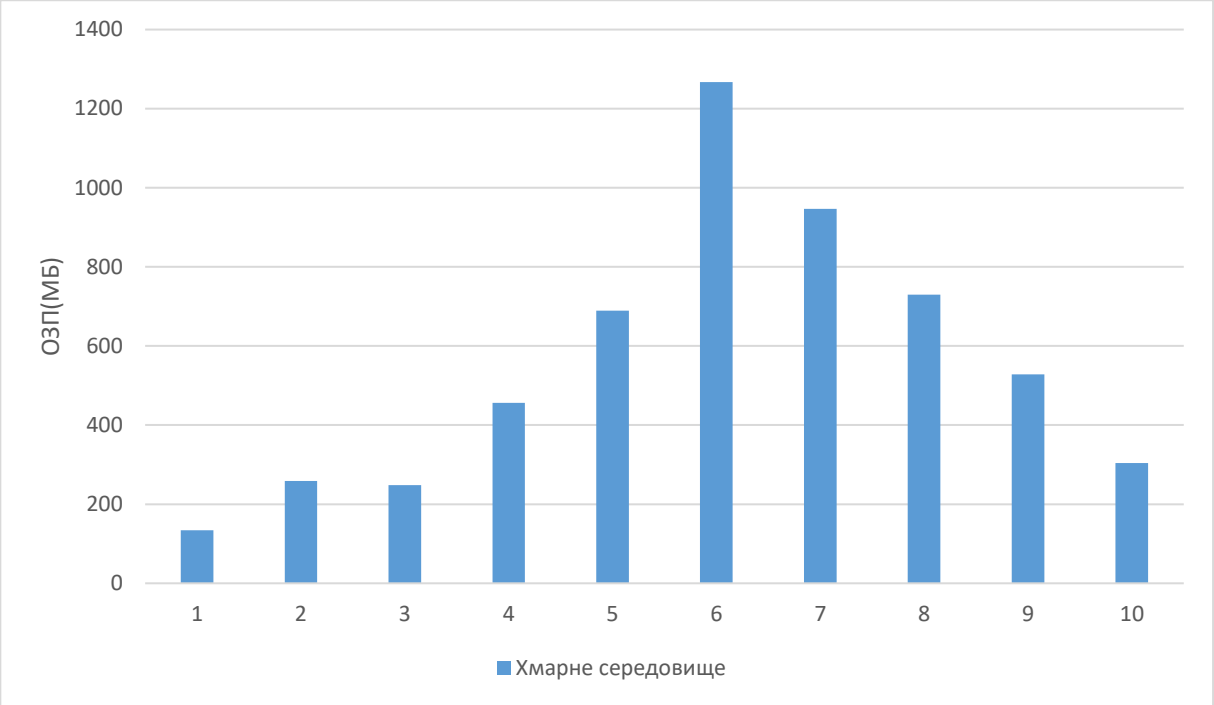


Рисунок 4.6 – Використання ОЗП хмарним середовищем

Окрім вузлів туману, обчислення поводились також на стороні хмари, куди відправлялись усі результати обчислень із вузлів туману, та обчислення у випадку коли кінцевий пристрій не був покритий мережею туману або коли хмарне середовище мало меншу затримку відповіді, аніж вузли туману. На рисунку 4.6 зображено графік використання оперативної пам’яті на стороні хмарного середовища.

4.3.4 Затримка відповіді опрацювання задачі вузлом туману

Основним параметром оптимізації методом, що був розроблений є затримка відповіді на опрацювання задачі вузлами туману. Було згенеровано статистику затримок відповіді для усіх вузлів туману та хмарного середовища.

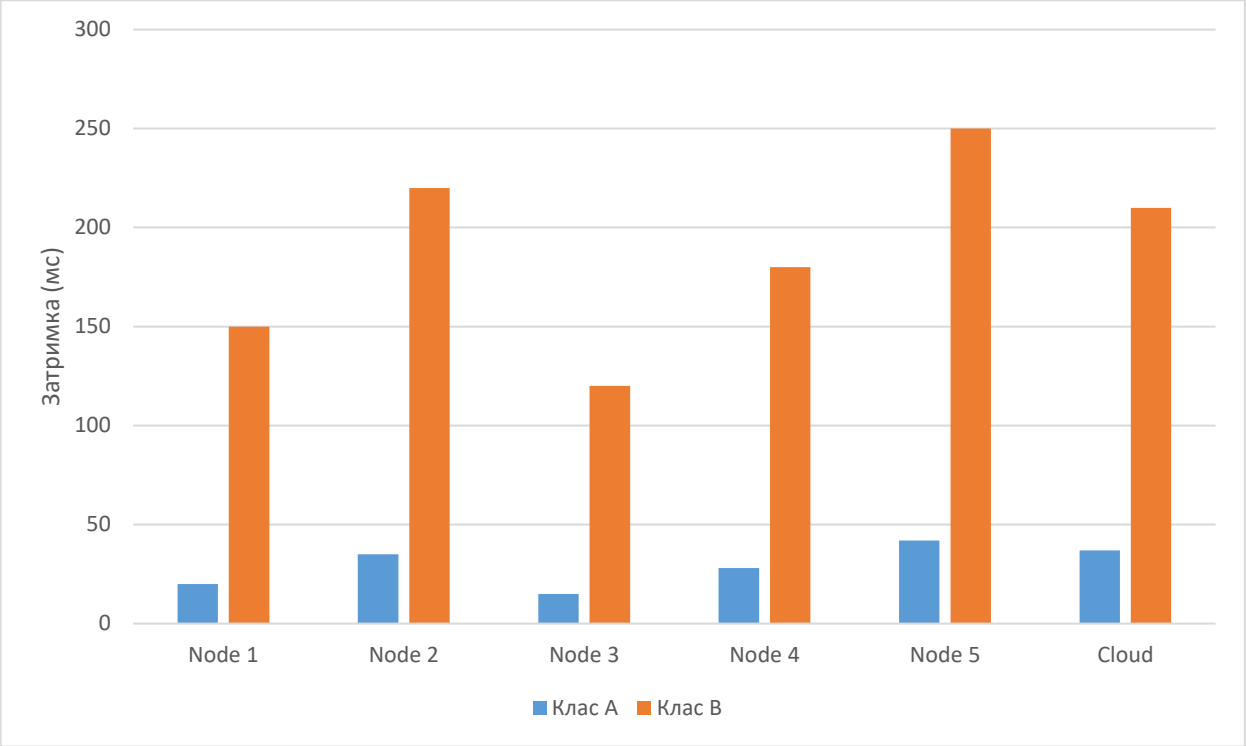


Рисунок 4.7 – Середнє значення затримки відповіді на опрацювання задач від вузлів до кінцевих пристроїв для розробленого методу оптимізації

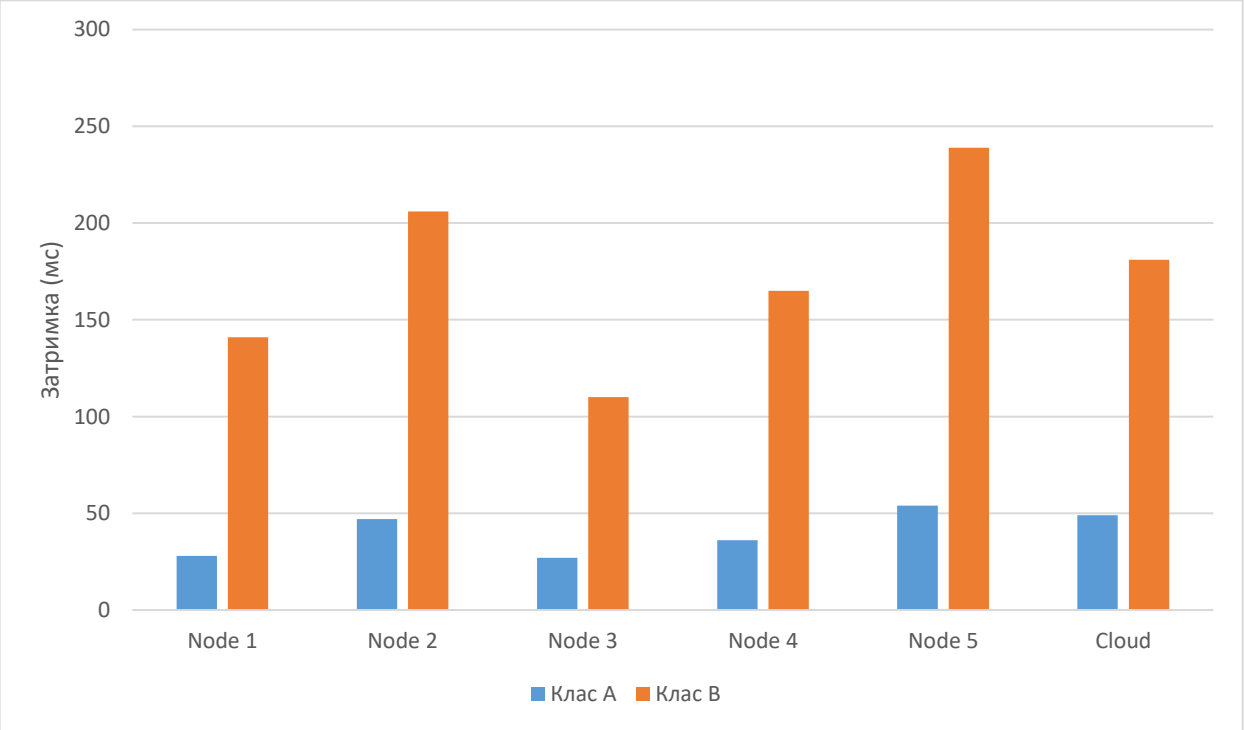


Рисунок 4.8 – Середнє значення затримки відповіді на опрацювання задач від вузлів до кінцевих пристроїв для стратегії Hold Down

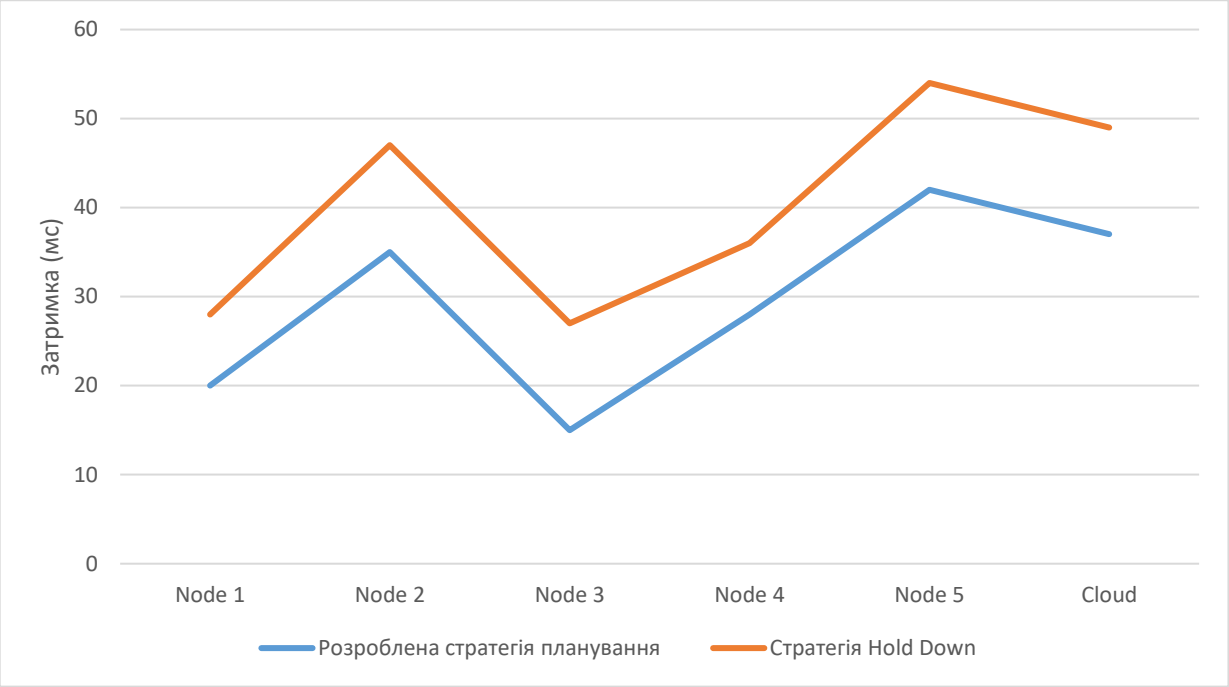


Рисунок 4.9 – Порівняння середнього значення затримки виконання задач класу А

Для експерименту було використано два класи задач:

- клас задач А має низький розмір байт, тому швидко доставляється мережею та опрацьовується вузлом.кількість виконаних та запланованих задач;
- клас задач В має велику кількість байт та довго завантажується мережею та більше часу опрацьовується вузлом.

Середнє значення затримки виконання задач для розробленого методу оптимізації представлено на рисунку 4.7. На діаграмі показано значення затримки для класів задач А та В.

На рисунку 4.8 зображено результати експерименту для стратегії планування Hold Down, на якій також зображено значення затримки для класів задач А та В.

Проведемо порівняння затримки для обох стратегій. На рисунку 4.9 зображено пряме порівняння середнього значення затримки виконання задач класу А.

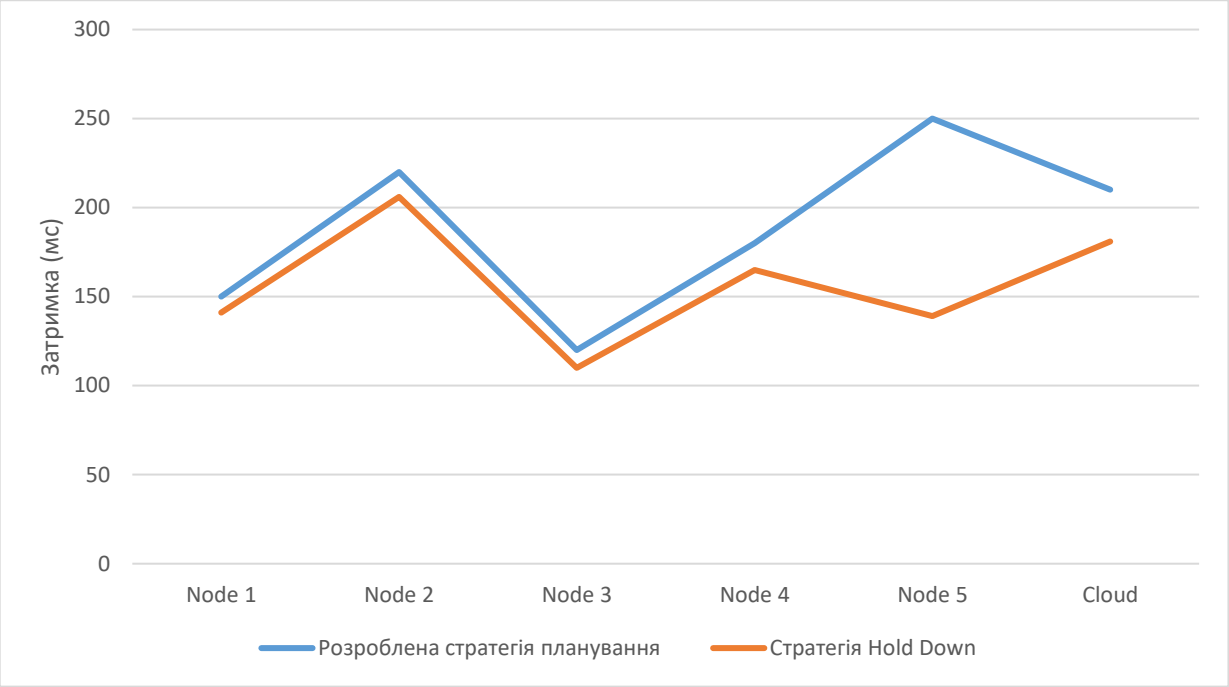


Рисунок 4.10 – Порівняння середнього значення затримки виконання задач класу B

На рисунку 4.9 видно, що розроблена стратегія планування задач показує кращий результат затримки виконання задач класу A. Такий результат зумовлений тим, що планування відбувається відповідно до алгоритму планування, що опирається на пріоритетність задач.

Рисунок 4.10 демонструє середнє значення затримки виконання задач класу B. Як результат, можемо спостерігати, що для стратегії Hold Down значення затримки виконання задач класу B є меншим, аніж в розробленій стратегії. Це спричинено тим, що більша частина часу виконання задач припадає на задачі класу A, який має більший пріоритет виконання.

Як висновок за результатом експерименту можна визначити, що збільшення пріоритетності виконання задач із меншим часом опрацювання призводить до більш продуктивної роботи IoT системи в цілому. Завдяки використанню розробленої стратегії планування задач, задачі із більшим пріоритетом виконуються із меншими затримками. Також, таких задач виконується більше, аніж могло б виконатись без використання пріоритетів.

З іншого боку у такому випадку збільшується затримка виконання класу задач на виконання яких витрачається більше часу. Хоча результат розробленої стратегії має не великий розрив значень із стратегією, що обирає найближчий вузол. Можна привести заключення, що розроблена стратегія збільшує ефективність IoT мережі у випадку, якщо переважна кількість задач, що генеруються мають низький час опрацювання.

4.4 Висновки

В розділі було реалізовано засоби планування завдань в IoT інфраструктурі із застосуванням туманних обчислень на основі удосконаленого методу планування завдань в мережі туманних обчислень.

Було проведено експериментальні дослідження функціонування IoT інфраструктури із застосуванням туманних обчислень із впровадженням удосконаленого методу планування завдань.

За результатами експериментальних досліджень було виявлено, що удосконалений метод планування завдань зменшує час відгуку вузлів туману за умов, що завдання які мають низький час опрацювання мають високий пріоритет виконання на вузлах туману, та такі завдання складають не менше 60 відсотків завдань, що генеруються датчиками IoT системи.

Експеримент дослідження функціонування IoT інфраструктури із застосуванням туманних обчислень із удосконаленим методом планування завдань показав, що час відгуку вузлів туману зменшився на 30% для класу задач із низьким часом опрацювання. Однак, час відгуку для задач, на опрацювання яких витрачається більше часу, збільшився на 6% відносно вбудованого методу планування завдань на найближчі вузли туману.

ВИСНОВКИ

Під час роботи було досліджено основні методи оптимізації IoT інфраструктури із використанням концепції туманних обчислень. Окрім цього було висвітлено основні переваги та недоліки методів оптимізації та їх порівняння.

В ході роботи було проаналізовано сучасні методи оптимізації функціонування IoT інфраструктури із застосуванням концепції туманних обчислень. Також було визначено основні характеристики концепції туманних обчислень та були визначені переваги та недоліки туманних обчислень з поміж інших концепцій оптимізації IoT.

Окрім цього, було визначено основні проблеми концепції туманних обчислень в межах сфер використання, та визначено напрямки подальших досліджень. Також, була розглянута архітектура IoT інфраструктури із використанням туманних обчислень та визначено місце туманних обчислень в IoT інфраструктурі.

Було удосконалено один із методів оптимізації IoT інфраструктури із використанням концепції туманних обчислень шляхом надання рівнів пріоритету на виконання різних класів завдань та врахування актуального навантаження вузлів туману.

Як результат роботи було реалізовано модель IoT інфраструктури із використанням туманних обчислень за допомогою середовища моделювання DISSECT-CF-Fog.

За результатами експериментальних досліджень було визначено, що використання удосконаленого методу оптимізації IoT інфраструктури із використанням туманних обчислень покращило якість обслуговування(QoS) системи за рахунок зменшення затримки опрацювання задач.

Було зроблено порівняння удосконаленого методу оптимізації IoT інфраструктури із використанням туманних обчислень із вбудованим в середовище моделювання методом планування задач до найближчого до

кінцевого пристрою вузла мережі туману. У інфраструктурі, що використовувала удосконалений метод оптимізації IoT інфраструктури із використанням туманних обчислень було зменшено час відгуку на 30% для класу задач із низьким часом опрацювання. Однак, час відгуку для задач, на опрацювання яких витрачається більше часу, було збільшено на 6%.

За результатами експериментальних досліджень було зроблено висновок, що удосконалений метод оптимізації IoT інфраструктури із використанням туманних обчислень підвищує ефективність інфраструктури у випадку, якщо переважна більшість задач, що генеруються мать низький час опрацювання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Шудрик А.О. Метод та програмно-технічні засоби оптимізації IoT інфрастуктури із застосуванням концепції туманних обчислень. *Стан, досягнення та перспективи інформаційних систем і технологій*: тези доп. всеукр. наук.-практ. конф. (м. Одеса, 18 квіт. 2024 р.). Одеса, 2021. С. 324-325.

2. Kavita S, Pethuru R Srirama SN. Chapter Eight - Edge platforms, frameworks and applications. *Elsevier*, 2022. . 237-258.

3. Asghari A, Sohrabi MK. Server placement in mobile cloud computing: a comprehensive survey for edge computing, fog computing and cloudlet. *Computer Science Review*, 2024. №.51. pp. 584-603.

4. Srirama SN. A decade of research in fog computing: Relevance, challenges, and future directions. Software: *Practice and Experience*. 2024. №.1. pp. 243-251.

5. Jumani AK, Shi J, Laghari AA, Hu Z, Nabi AU, Qian H. Fog computing security: A review. *Security and Privac*y. 2023. №.6. pp. 313-319.

6 Hussein WN, Hussain HN, Hussain HN, Mallah AQ. A deployment model for IoT devices based on fog computing for data management and analysis. *Wireless Personal Communications*. 2023. №20, 153-158.

7. Rani S, Srivastava G. Secure hierarchical fog computing-based architecture for industry 5.0 using an attribute-based encryption scheme. *Expert Systems with Applications*. 2024. №.235. pp. 1211-1220.

8. Mohammed BA, Al-Shareeda MA, Manickam S, Al-Mekhlafi ZG, Alayba AM, Sallam AA. Anaa-fog: A novel anonymous authentication scheme for 5g-enabled vehicular fog computing. *Mathematics*. 2023 №16. pp.1446-1553.

9. Wang Z, Goudarzi M, Gong M, Buyya R. Deep Reinforcement Learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Generation Computer Systems*. 2024. pp. 55-69.

10. Al-Shareeda MA, Alsadhan AA, Qasim HH, Manickam S. The fog computing for internet of things: review, characteristics and challenges, and open issues. *Bulletin of Electrical Engineering and Informatics*. 2024. №.13. pp. 1080-1089.

11. Pan S, Huang C, Fan J, Shi Z, Tong J, Wang H. Optimizing Internet of Things Fog Computing: Through Lyapunov-Based Long Short-Term Memory Particle Swarm Optimization Algorithm for Energy Consumption Optimization. *Sensors*. 2024. №.24. pp. 1165-1172.

12. Hazra A, Rana P, Adhikari M, Amgoth T. Fog computing for next-generation internet of things: fundamental, state-of-the-art and research challenges. *Computer Science Review*. 2023. №.48. pp. 549-554.

13. Apat HK, Nayak R, Sahoo B. A comprehensive review on Internet of Things application placement in Fog computing environment. *Internet of Things*. 2023. pp 866-873.

14. Zhang X, Xu M, Su J, Zhao P. Structural models for fog computing based internet of things architectures with insurance and risk management applications. *European Journal of Operational Research*. 2023. №.305. pp. 1273-1291.

15. Gulatas I, Kilinc HH, Zaim AH, Aydin MA. Malware threat on edge/fog computing environments from Internet of things devices perspective. *IEEE Access.* 2023. №.11. pp. 584-606.

16. Songhorabadi M, Rahimi M, MoghadamFarid A, Kashani MH. Fog computing approaches in IoT-enabled smart cities. *Journal of Network and Computer* Applications. 2023. №.211. C. 557-563.

17. Agnihotri AK, Gupta SK, Tiwari B. Healthcare Technology Evolution And Adoption Of Fog Computing In Healthcare: Review, Issue And Challenges. *Journal of Pharmaceutical Negative Results*. 2023. pp. 1419-1428.

18. Al-Mekhlafi ZG, Al-Shareeda MA, Manickam S, Mohammed BA, Alreshidi A, Alazmi M, Alshudukhi JS, Alsaffar M, Rassem TH. Efficient authentication scheme for 5G-enabled vehicular networks using fog computing. *Sensors*. 2023. № 28. pp. 3543-5549.

19. Jayanagara O, Wuisan DS. An Overview of Concepts, Applications, Difficulties, Unresolved Issues in Fog Computing and Machine Learning. *International Transactions on Artificial Intelligence*. 2023. pp. 213-229.

20. Pallewatta S, Kostakos V, Buyya R. Placement of microservices-based iot applications in fog computing: A taxonomy and future directions. *ACM Computing Surveys*. 2023. pp 428-437.

21. Naeem MA, Zikria YB, Ali R, Tariq U, Meng Y, Bashir AK. Cache in fog computing design, concepts, contributions, and security issues in machine learning prospective. *Digital Communications and Networks.* 2023. pp.1033-1052.

22. Padhy S, Alowaidi M, Dash S, Alshehri M, Malla PP, Routray S, Alhumyani H. Agrisecure: A fog computing-based security framework for agriculture 4.0 via blockchain. *Processes*. 2023. pp. 757-769.

23. Gulatas I, Kilinc HH, Zaim AH, Aydin MA. Malware threat on edge/fog computing environments from Internet of things devices perspective. *IEEE Access*. 2023. pp. 584-606.

24. Mohammed BA, Al-Shareeda MA, Manickam S, Al-Mekhlafi ZG, Alreshidi A, Alazmi M, Alshudukhi JS, Alsaffar M. FC-PA: fog computing-based pseudonym authentication scheme in 5G-enabled vehicular networks. *IEEE Access*. 2023. pp. 571-581.

25. Gowda NC, Manvi SS, Malakreddy B, Lorenz P. BSKM-FC: Blockchain-based secured key management in a fog computing environment. *Future Generation Computer Systems*. 2023. pp. 276-291.

26. Mohamed AA, Abualigah L, Alburaikan A, Khalifa HA. AOEHO: a new hybrid data replication method in fog computing for IoT application. *Sensors*. 2023. pp. 2189-2197.

27. Tuli S, Mirhakimi F, Pallewatta S, Zawad S, Casale G, Javadi B, Yan F, Buyya R, Jennings NR. AI augmented Edge and Fog computing: Trends and challenges. *Journal of Network and Computer Applications*. 2023. pp. 3648-3658.

28. Atiq HU, Ahmad Z, Uz Zaman SK, Khan MA, Shaikh AA, Al-Rasheed A. Reliable resource allocation and management for IoT transportation using fog computing. *Electronics*. 2023. pp. 1452-1467.

29. Tran-Dang H, Kim DS. Dynamic collaborative task offloading for delay minimization in the heterogeneous fog computing systems. *Journal of Communications and Networks.* 2023. pp. 583-598.

30. Behravan K, Farzaneh N, Jahanshahi M, Seno SA. A comprehensive survey on using fog computing in vehicular networks. *Vehicular Communications.* 2023. pp. 604-615.

31. Wu Q, Wang S, Ge H, Fan P, Fan Q, Letaief KB. Delay-sensitive task offloading in vehicular fog computing-assisted platoons. *IEEE Transactions on Network and Service Management.* 2023. pp. 732-743.

32. Almazroi AA, Aldhahri EA, Al-Shareeda MA, Manickam S. ECA-VFog: An efficient certificateless authentication scheme for 5G-assisted vehicular fog computing. *Plos one*. 2023. pp. 291-306.

33. Wei Z, Li B, Zhang R, Cheng X, Yang L. Many-to-many task offloading in vehicular fog computing: A multi-agent deep reinforcement learning approach. *IEEE Transactions on Mobile Computing*. 2023. pp. 462-479.

34. Chakraborty C, Othman SB, Almalki FA, Sakli H. FC-SEEDA: Fog computing-based secure and energy efficient data aggregation scheme for Internet of healthcare Things. *Neural Computing and Application*s. 2023. pp. 241-257.

35. Dogani J, Namvar R, Khunjush F. Auto-scaling techniques in container-based cloud and edge/fog computing: Taxonomy and survey. *Computer Communications*. 2023. pp. 834-847.

36. Sethi V, Pal S. FedDOVe: A Federated Deep Q-learning-based Offloading for Vehicular fog computing. *Future Generation Computer Systems*. 2023. pp. 96-105.

37. Abdulazeez DH, Askar SK. Offloading mechanisms based on reinforcement learning and deep learning algorithms in the fog computing environment. *Ieee Access*. 2023. pp. 2555-2586.

38. Asghari A, Sohrabi MK. Server placement in mobile cloud computing: a comprehensive survey for edge computing, fog computing and cloudlet. *Computer Science Review*. 2023. pp. 616-624.

39. Al-Mekhlafi ZG, Al-Shareeda MA, Manickam S, Mohammed BA, Alreshidi A, Alazmi M, Alshudukhi JS, Alsaffar M, Alsewari A. Chebyshev polynomial-based fog computing scheme supporting pseudonym revocation for 5G-enabled vehicular networks. *Electronics*. 2023. pp. 872-884.

40. Safa'a SS, Alansari I, Hamiaz MK, Ead W, Tarabishi RA, Khater H. iFogRep: An intelligent consistent approach for replication and placement of IoT based on fog computing. *Egyptian Informatics Journal*. 2023. pp. 327-339.

41. Singh J, Singh P, Hedabou M, Kumar N. An efficient machine learning-based resource allocation scheme for sdn-enabled fog computing environment. *IEEE Transactions on Vehicular Technology*. 2023. pp. 568-574.

42. Ortiz-Garcés I, Andrade RO, Sanchez-Viteri S, Villegas-Ch W. Prototype of an emergency response system using IoT in a Fog computing environment. *Computers*. 2023. pp. 181-198.

43. Mutlag AA, Abd Ghani MK, Mohd O, Abdulkareem KH, Mohammed MA, Alharbi M, Al-Araji ZJ. A new fog computing resource management (FRM) model based on hybrid load balancing and scheduling for critical healthcare applications. *Physical Communication*. 2023. pp. 2108-2124.

44. Li S, Liu H, Li W, Sun W. Optimal cross-layer resource allocation in fog computing: A market-based framework. *Journal of Network and Computer Applications*. 2023. pp. 3528-3539.

45. Burhan M, Alam H, Arsalan A, Rehman RA, Anwar M, Faheem M, Ashraf MW. A comprehensive survey on the cooperation of fog computing paradigm-based iot applications: layered architecture, real-time security issues, and solutions. *IEEE Access*. 2023. pp. 473-487.

46. Saif FA, Latip R, Hanapi ZM, Shafinah K. Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing. *IEEE Access*. 2023. pp. 635-646.

47. Tran-Dang H, Kim DS. Disco: Distributed computation offloading framework for fog computing networks. *Journal of Communications and Networks.* 2023. pp.121-131.

48. Wang Z, Goudarzi M, Gong M, Buyya R. Deep Reinforcement Learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Generation Computer Systems.* 2023. pp. 55-69.

49. Hosseinzadeh M, Azhir E, Lansky J, Mildeova S, Ahmed OH, Malik MH, Khan F. Task scheduling mechanisms for fog computing: A systematic survey. *IEEE Access*. 2023. pp. 782-797.

50. Zare M, Sola YE, Hasanpour H. Towards distributed and autonomous IoT service placement in fog computing using asynchronous advantage actor-critic algorithm. *Journal of King Saud University-Computer and Information Sciences.* 2023. pp. 368-381.

51. Li J, Li D, Zhang X. A secure blockchain-assisted access control scheme for smart healthcare system in fog computing. *IEEE Internet of Things Journal*. 2023. pp. 1134-1148.

52. Ostrowski K, Małecki K, Dziurzański P, Singh AK. Mobility-aware fog computing in dynamic networks with mobile nodes: A survey. *Journal of Network and Computer Applications.* 2023. pp. 3724-3738.

53. Li Y, Zhang S, Chang Y, Xu G, Li H. Privacy-Preserving and Poisoning-Defending Federated Learning in Fog Computing. *IEEE Internet of Things Journal*. 2023. pp. 832-845.

54. Machida F, Zhang Q, Andrade E. Performability analysis of adaptive drone computation offloading with fog computing. *Future Generation Computer Systems*. 2023. pp. 121-135.

55. Mohammed MA, Lakhan A, Abdulkareem KH, Garcia-Zapirain B. Federated auto-encoder and XGBoost schemes for multi-omics cancer detection in distributed fog computing paradigm. *Chemometrics and Intelligent Laboratory Systems*. 2023. pp. 932-948.

56. Siyadatzadeh R, Mehrafrooz F, Ansari M, Safaei B, Shafique M, Henkel J, Ejlali A. Relief: A reinforcement learning-based real-time task assignment strategy in emerging fault-tolerant fog computing. *IEEE Internet of Things Journal*. 2023. pp. 1238-1251.

57. Sabireen H, Neelanarayanan VJ. A review on fog computing: Architecture, fog with IoT, algorithms and research challenges. *Ict Express*. 2021. pp. 162-176.

58. Singh J, Singh P, Gill SS. Fog computing: A taxonomy, systematic review, current trends and research challenges. *Journal of Parallel and Distributed Computing.* 2021. pp. 56-85.

59. Costa B, Bachiega Jr J, de Carvalho LR, Araujo AP. Orchestration in fog computing: A comprehensive survey. *ACM Computing Surveys (CSUR).* 2022. pp. 101-134.

60. Habibi P, Farhoudi M, Kazemian S, Khorsandi S, Leon-Garcia A. Fog computing: a comprehensive architectural survey. *IEEE access*. 2020. pp. 105-135.

61. Martinez I, Hafid AS, Jarray A. Design, resource management, and evaluation of fog computing systems: A survey. *IEEE Internet of Things Journal*. 2020. pp. 2494-2516.

62. Laroui M, Nour B, Moungla H, Cherif MA, Afifi H, Guizani M. Edge and fog computing for IoT: A survey on current research activities & future directions. *Computer Communications*. 2021. pp. 210-231.

63. Mahmud R, Ramamohanarao K, Buyya R. Application management in fog computing environments: A taxonomy, review and future directions. ACM *Computing Surveys (CSUR).* 2020. pp. 1-43.

64. Varghese B, Wang N, Nikolopoulos DS, Buyya R. Feasibility of fog computing. *Handbook of Integration of Cloud Computing, Cyber Physical Systems and Internet of Things*. 2020. pp. 127-146.

65. Ometov A, Molua OL, Komarov M, Nurmi J. A survey of security in cloud, edge, and fog computing. *Sensors*. 2022. pp. 927-938.

66. Javadzadeh G, Rahmani AM. Fog computing applications in smart cities: A systematic survey. *Wireless Networks*. 2020. pp.1433-1457.

67. Das R, Inuwa MM. A review on fog computing: Issues, characteristics, challenges, and potential applications. *Telematics and Informatics Reports*. 2023. pp. 49-64.

68. Zhang C. Design and application of fog computing and Internet of Things service platform for smart city. *Future Generation Computer Systems*. 2020. pp. 630-640.

69. Haghi Kashani M, Rahmani AM, Jafari Navimipour N. Quality of service-aware approaches in fog computing. *International Journal of Communication Systems*. 2020. pp. 4340-4356.

70. Abdulqadir HR, Zeebaree SR, Shukur HM, Sadeeq MM, Salim BW, Salih AA, Kak SF. A study of moving from cloud computing to fog computing. *Qubahan Academic Journal*. 2021. pp. 60-70.

71. Bhambri P, Rani S, Gupta G, Khang A, editors. Cloud and fog computing platforms for internet of things. *CRC Press*; 2022. pp. 635-651.

72. Tange K, De Donno M, Fafoutis X, Dragoni N. A systematic survey of industrial Internet of Things security: Requirements and fog computing opportunities. *IEEE Communications Surveys & Tutorials*. 2020. pp. 2489-2520.

73. Moura J, Hutchison D. Fog computing systems: State of the art, research issues and future trends, with a focus on resilience. *Journal of Network and Computer Applications*. 2020. pp. 2784-2806.

74. Alwakeel AM. An overview of fog computing and edge computing security and privacy issues. *Sensors*. 2021. pp. 1226-1240.

75. Rani S, Kataria A, Chauhan M. Fog computing in industry 4.0: Applications and challenges—A research roadmap. *Energy conservation solutions for fog-edge computing paradigms*. 2022. pp. 173-190.

76. Alzoubi YI, Osmanaj VH, Jaradat A, Al-Ahmad A. Fog computing security and privacy for the Internet of Thing applications: State-of-the-art. *Security and Privacy*. 2021. pp. 145-154.

77. Shakarami A, Shakarami H, Ghobaei-Arani M, Nikougoftar E, Faraji-Mehmandar M. Resource provisioning in edge/fog computing: A comprehensive and systematic review. *Journal of Systems Architecture*. 2022. pp. 2362-2375.

78. Tripathy SS, Bebortta S, Chowdhary CL, Mukherjee T, Kim S, Shafi J, Ijaz MF. FedHealthFog: A federated learning-enabled approach towards healthcare analytics over fog computing platform. *Heliyon*. 2024. pp. 352-374.

79. Ahmad I, Abdullah S, Ahmed A. IoT-fog-based healthcare 4.0 system using blockchain technology. *The Journal of Supercomputing.* 2023. №.79. pp. 3999-4020.

80. Songhorabadi M, Rahimi M, MoghadamFarid A, Kashani MH. Fog computing approaches in IoT-enabled smart cities. *Journal of Network and Computer Applications.* 2023. №.211. pp. 3557-3568.

81. Hazra A, Rana P, Adhikari M, Amgoth T. Fog computing for next-generation internet of things: fundamental, state-of-the-art and research challenges. *Computer Science Review.* 2023.№.48. pp.549-558.

82. A. Yousefpour, G. Ishigaki and J. P. Jue, "Fog computing: Towards minimizing delay in the Internet of Things", Proc. *IEEE Int. Conf. Edge Comput*. (EDGE), pp. 17-24, Jun. 2017.

83. L. Gu, D. Zeng, S. Guo, A. Barnawi and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system", *IEEE Trans. Emerg. Topics Comput*., vol. 5, no. 1, pp. 108-119, Jan. 2017.

84. R. Deng, R. Lu, C. Lai, T. H. Luan and H. Liang, "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption", *IEEE Internet Things* J., vol. 3, no. 6, pp. 1171-1181, Dec. 2016.

85. Markus, Andras, Ali Al-Haboobi, Gabor Kecskemeti, and Attila Kertesz. 2023. "Simulating IoT Workflows in DISSECT-CF-Fog" *Sensors* 23, no. 3: 1294

ДОДАТОК А (ОБОВ'ЯЗКОВИЙ)

ТЕЗИ

УДК 004.05

МЕТОД ТА ПРОГРАМНО-ТЕХНІЧНІ ЗАСОБИ ОПТИМІЗАЦІЇ IOT ІНФРАСТУКТУРИ ІЗ ЗАСТОСУВАННЯМ КОНЦЕПЦІЇ ТУМАННИХ ОБЧИСЛЕНЬ

ШУДРИК А.О. (andrii.sdr@gmail.com)
Хмельницький Національний Університет

Метою роботи є розробка евристичного методу оптимізації IoT інфраструктури із використанням туманних обчислень. Основними проблемами в туманних обчисленнях є планування завдань, затримка відповіді та споживання енергії. Розглянуто алгоритми оптимізації: PSO, ACO, GA, GWO. Визначено вимоги до методу, що розробляється та висунуто ідею щодо врахування пріоритетності виконання задач в мережі туманних обчислень.

Туманні обчислення – термін запропонований компанією Cisco, для визначення нової архітектури обчислень в IoT інфраструктурі. Туманні обчислення є високо віртуалізованою платформою, що знаходиться між шаром IoT пристроїв та традиційним хмарним середовищем, яка додає шар гнучких обчислень та зберігання даних (рис. 1).

Концепція туманних обчислень передбачає розміщення обчислювальних ресурсів на периферії мережі, фізично ближче до джерела даних та кінцевого користувача. Завдяки такій архітектурі досягаються зменшення навантаження на пропускну здатність мережі, що є важливою вимогою для застосунків які працюють в режимі реального часу.

Метою дослідження є розробка евристичного методу оптимізації планування задач в мережі туманних обчислень який досягне балансу між використанням енергії та мінімізацією часу виконання задачі.

Динамічна та гетерогенна природа туманних обчислень спричиняє проблеми для ефективного управління ресурсами та оптимізації робочого процесу. Для вирішення цих проблем можуть бути використані методи чисельної оптимізації, такі як: метод рою часток(PSO), мурашиний алгоритм(ACO), генетичний алгоритм(GA) та алгоритм зграї сірих вовків(GWO).

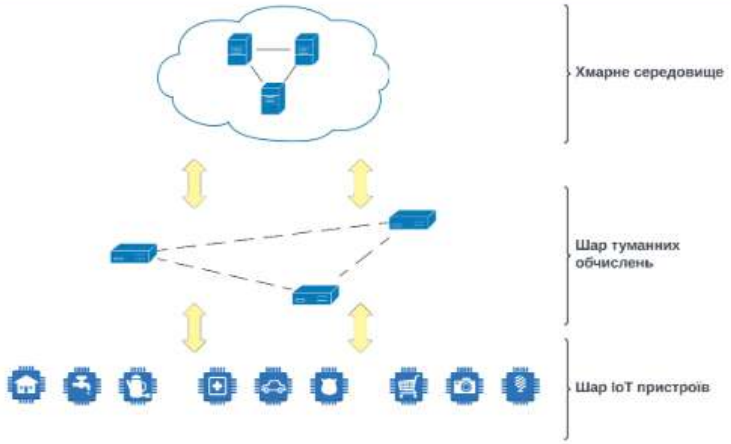


Рис. 1 Архітектура IoT із використанням туманних обчислень

Алгоритм рою часток(PSO) був розроблений із метою адаптації до невизначеностей, що притаманні туманному середовищу. Оптимізація полягає в тому, що задачі плануються динамічно, базуючись на доступних ресурсах та енергетичному профілі туманних вузлів. Метод визначає найкращий для виконання задачі туманний вузол, який визначається шляхом паралельного пошуку в якому кожен вузол є потенційно найкращим кандидатом на виконання задачі.

Алгоритм мурашиної колонії(ACO) використовується для пошуку найкоротшого шляху. В межах концепції туманних обчислень він може бути використаний для знаходження туманного вузла, який опрацює задачу із найменшою затримкою.

Генетичний алгоритм(GA) – еволюційний алгоритм, ідея якого базується на природному відборі генів. Алгоритм вирішує задачу пошуку найбільш оптимального туманного вузла для вирішення задачі шляхом перебору усіх варіантів та знаходження найкращого із них. За критерій якості в туманних обчисленнях виступає кількість часу, витрачена на обчислення задачі та повернення відповіді. Недоліком цього алгоритму є його неточність із невеликим набором даних.

Алгоритм зграї сірих вовків(GWO) базується на імітації ієрархії лідерства в зграї вовків та механізм полювання зграї. Алгоритм використовується для планування гетерогенних задач та розвантаження мережі.

За результатами дослідження було зроблено висновок, що метод оптимізації, який розробляється повинен базуватись на гібридному поєднанні декількох алгоритмів оптимізації та враховувати такі характеристики системи як: затримка відповіді, планування завдань та енергоспоживання. Окрім цього, пропонується додати критерій пріоритету виконання задачі, для контролю над навантаженням вузлів та підвищенню QoS для задач які вимагають найменшої затримки.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. S. -E. Chafi, Y. Balboul, M. Fattah, S. Mazer and M. El Bekkali, (2024). "Novel PSO-Based Algorithm for Workflow Time and Energy Optimization in a Heterogeneous Fog Computing Environment," in IEEE Access, vol. 12, pp. 41517-41530.

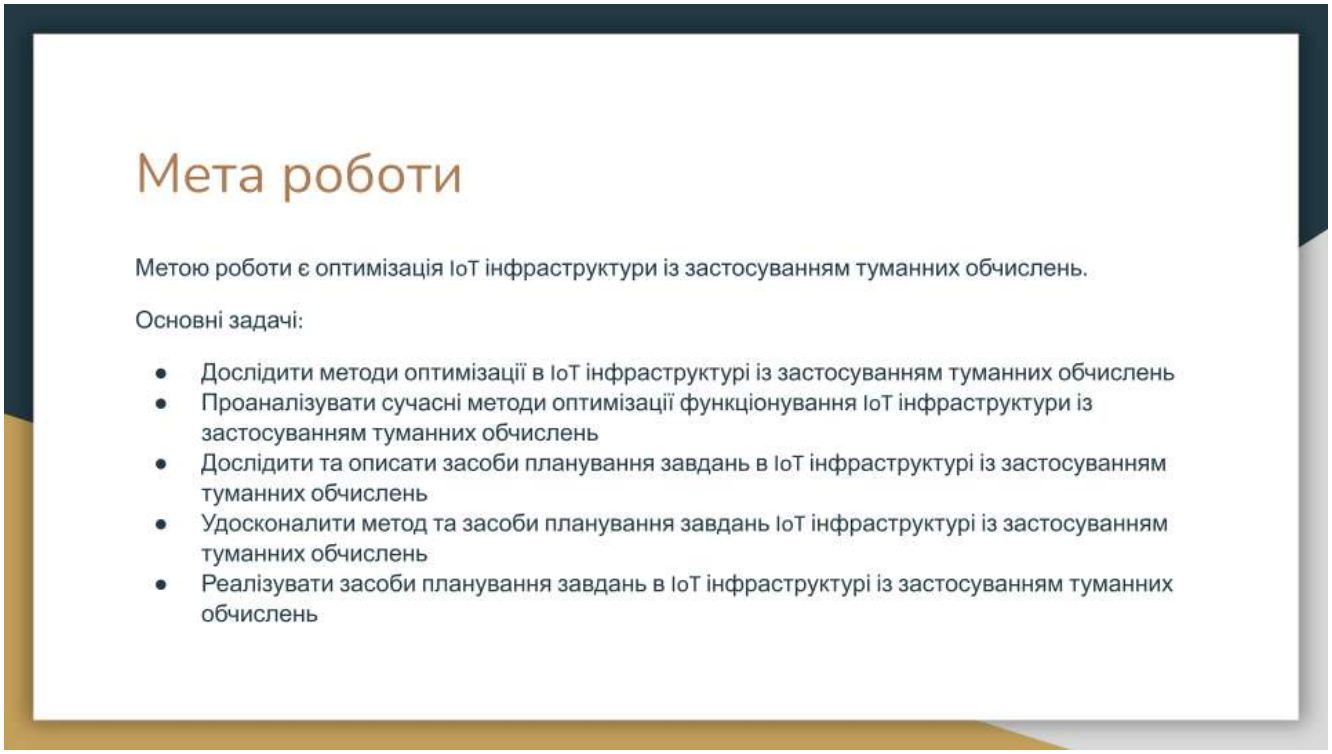
2. Zavieh, H., Javadpour, A., Ja'fari, F. et al. (2024). Enhanced Efficiency in Fog Computing: A Fuzzy Data-Driven Machine Selection Strategy. Int. J. Fuzzy Syst. 26, 368–389.

3. Saad, M., Enam, R., & Qureshi, R. (2024). Optimizing multi-objective task scheduling in fog computing with GA-PSO algorithm for big data application. Frontiers in Big Data, 7. DOI=10.3389/fdata.2024.1358486.

4. S. M. Hashemi, A. Sahafi, A. M. Rahmani and M. Bohlouli, (2022) "GWO-SA: Gray Wolf Optimization Algorithm for Service Activation Management in Fog Computing," in IEEE Access, vol. 10, pp. 107846-107863.

ДОДАТОК Б (ОБОВ'ЯЗКОВИЙ)

ПРЕЗЕНТАЦІЯ



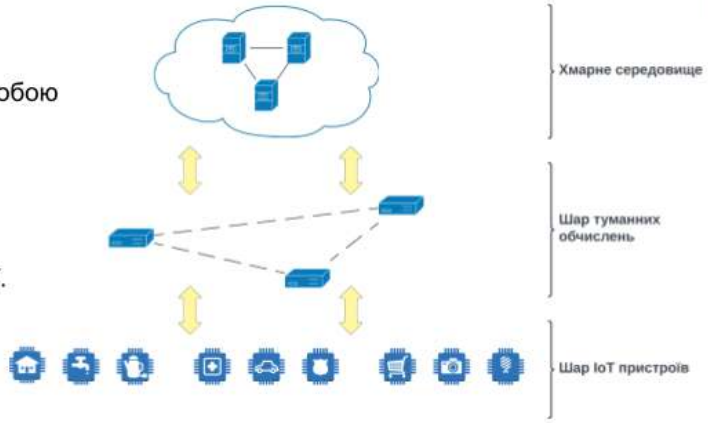
Ріст кількості IoT пристроїв

Оновні проблеми IoT

- Кількість даних, створених пристроями IoT зростає в геометричній прогресії, що спричинить перевантаження мережі та проблеми з продуктивністю на периферії інфраструктури;
- Існують завдання, для вирішення яких недостатньо лише хмарних рішень через такі фактори як продуктивність, безпека, пропускна здатність та надійність.

Концепція туманних обчислень

Туманні обчислення представляють собою архітектуру системного рівня, що допомагає досягти оптимального розподілу мережі, обчислювальних потужностей та зберігання інформації.



Завдання туманних обчислень

- обчислення;
- зберігання даних;
- надання мережевих сервісів на периферійних пристроях на стороні користувача інфраструктури.

Приклади апаратного забезпечення що може використовуватись в ролі вузла туманних обчислень

Rock PI 4

Beaglebone black

Cisco 5400 ENCS

ASUS Tinker board

Характеристика туманних обчислень

- низька затримка
- багата та різнорідна підтримка кінцевих пристроїв
- багатокористувацьке контрольоване середовище
- контекстність
- географічна розподіленість
- бездротовий доступ
- підтримка гетерогенності
- сумісність пристроїв
- аналітика в реальному часі

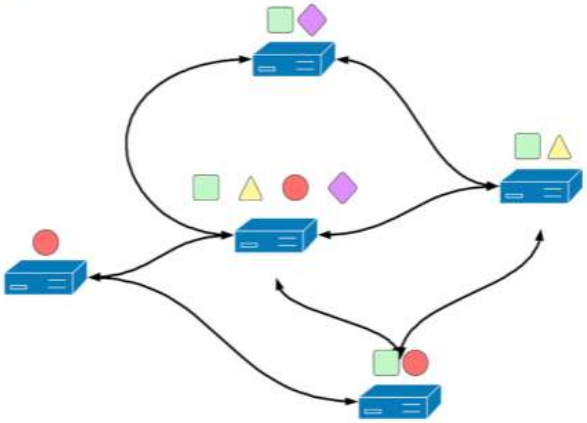
Переваги використання туманних обчислень

- зменшення затримки;
- збільшення пропускної здатності;
- безперервне надання послуг;
- широка підтримка пристроїв;
- ефективне використання ресурсів;
- зменшення обсягу передачі даних;
- легка масштабованість.

Недоліки використання туманних обчислень

- обчислювальна складність;
- складність розгортання та підтримки;
- вартість;
- обмежені ресурси.

Удосконалення методу планування завдань в мережі туманних обчислень

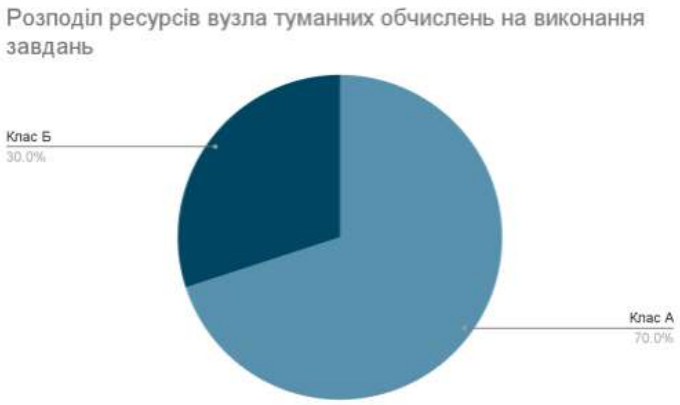


Удосконалення методу планування завдань в мережі туманних обчислень

Кроки знаходження найбільш ефективного вузла туманних обчислень

- Визначення класів завдань які можуть бути опрацьовані вузлами.
- Розподілення пріоритетів між класами завдань в межах кожного із вузлів туману.
- Визначення затримки відповіді вузлів за допомогою використання алгоритму ACO.
- Корегування результатів визначення ефективності вузлів із допомогою використання пріоритетів класів завдань та актуального навантаження вузлів туману.

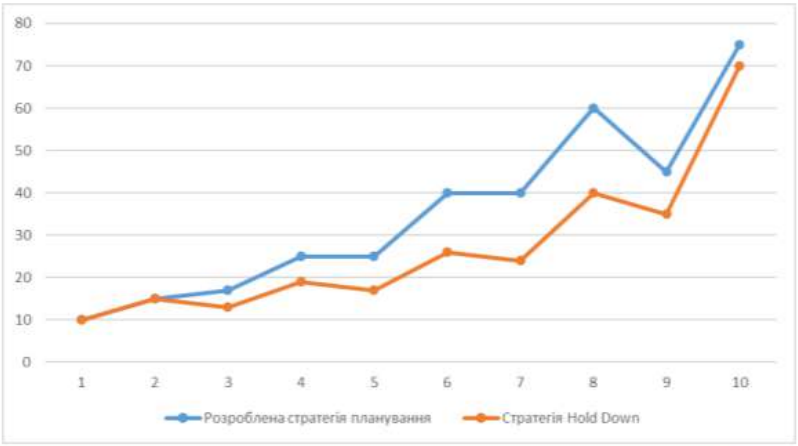
Удосконалення методу планування завдань в мережі туманних обчислень



Архітектура мережі туманних обчислень

З метою проведення експериментальних досліджень було побудовано мережу туманних вузлів та маршрути пристроїв IoT

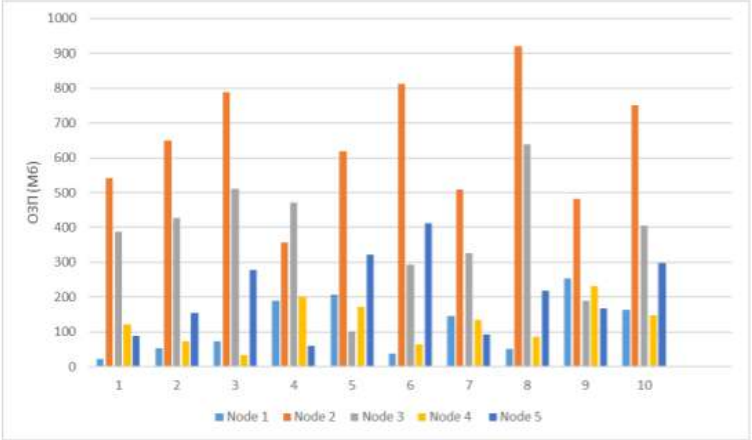
Порівняння кількості запланованих завдань



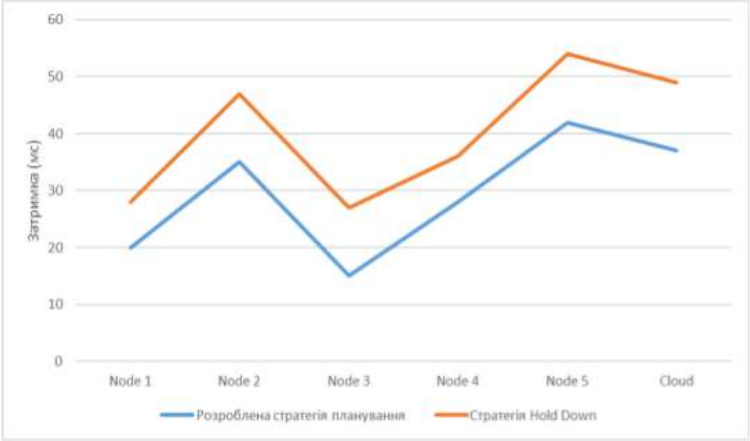
Архітектура мережі туманних обчислень

З метою проведення експериментальних досліджень було побудовано мережу туманних вузлів та маршрути пристроїв IoT

Загальне споживання озп вузлами туманних обчислень

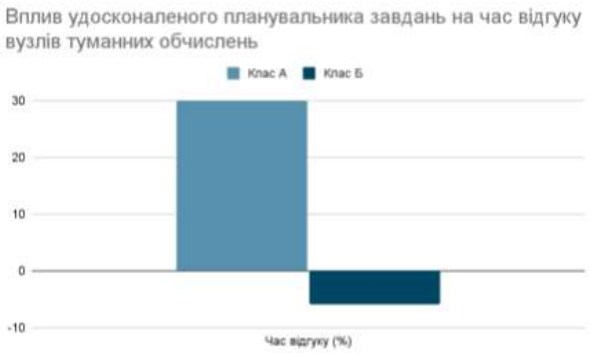


Порівняння затримки опрацювання завдань вузлами туманних обчислень



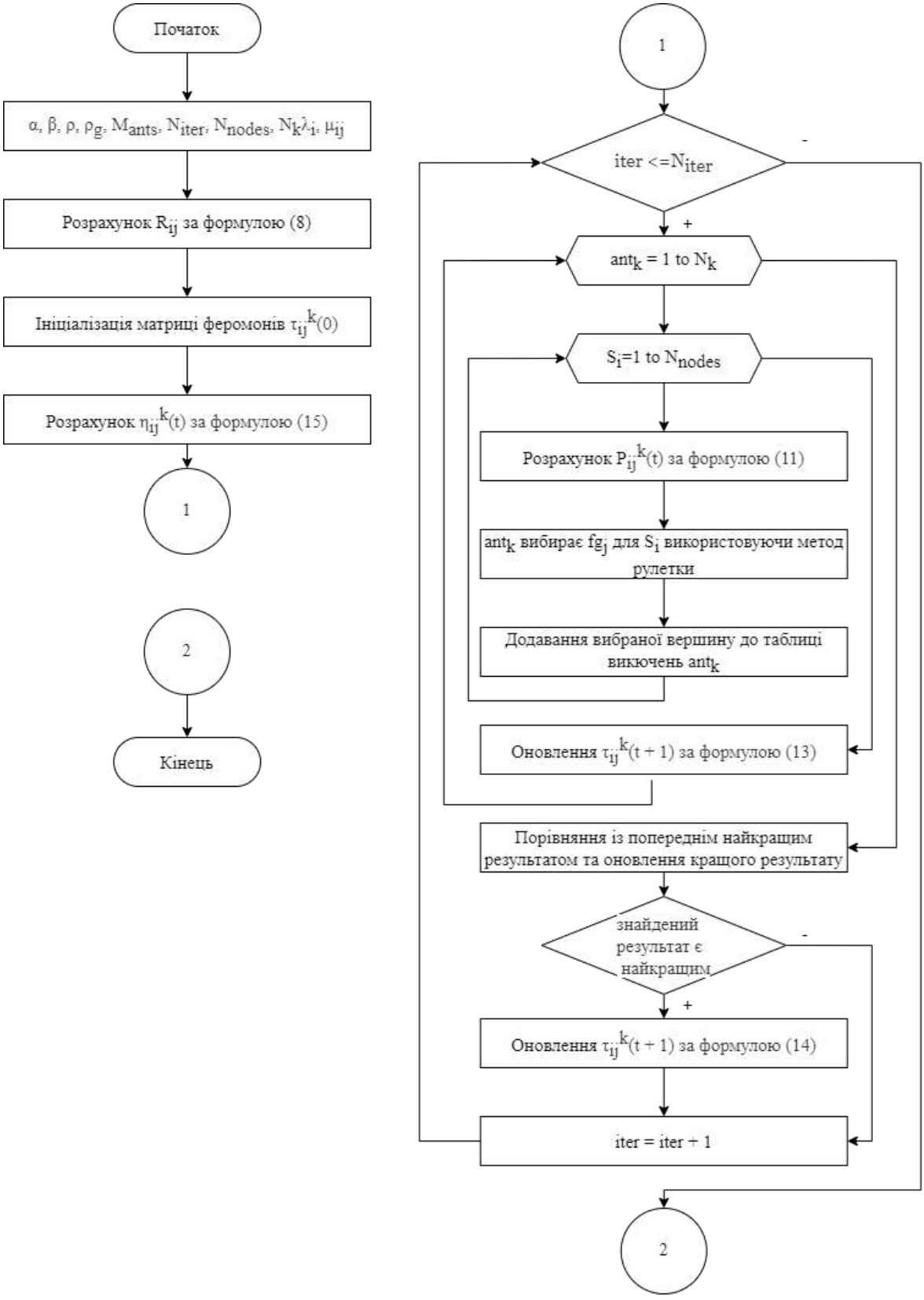
Висновки

За результатами експериментального дослідження було визначено, що завдяки впровадженню удосконаленого методу планування завдань було зменшено час відгуку вузлів туманних обчислень на 30%. Однак на опрацювання завдань що займають значну кількість ресурсів час відгуку зріз на 6%



ДОДАТОК В

МЕТАЕВРИСТИЧНИЙ АЛГОРИТМ З ВИКОРИСТАННЯМ АСО ДЛЯ ПОШУКУ ЕФЕКТИВНОГО РОЗВАНТАЖЕННЯ ЗАВДАНЬ



UNICHECK
by Turnitin

Ім'я користувача:
Кафедра КІ

ID перевірки:
1016247537

Дата перевірки:
13.05.2024 16:24:55 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
13.05.2024 16:29:28 EEST

ID користувача:
100005591

Назва документа: Шудрик_Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Кількість сторінок: 92 **Кількість слів:** 14553 **Кількість символів:** 113147 **Розмір файлу:** 2.01 MB **ID файлу:** 1016032711

3.51%
Схожість

Найбільша схожість: 1.07% з джерелом з Бібліотеки (ID файлу: 1016004947)

2.86% Джерела з Інтернету	207	Сторінка 94
2.12% Джерела з Бібліотеки	58	Сторінка 96

0.04% Цитат

Цитати	1	Сторінка 97
Посилання	1	Сторінка 97

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	130

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 1.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 10%**

	Документ		Сумарний збіг по Базі Даних	
ID: 126085 Назва: МКР Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень Додано в БД: 2024-05-13 Автора: Шудрик А.О. Керівники: Лисенко С.М. Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	88743	769	1948 (2%)	23 (3%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

Завідувачу кафедри КІІС
д-р.техн.наук, проф. Говорущенко Т. О.

Шудрика Андрія Олександровича
ПІБ здобувача вищої освіти

ФІТ, 2 курсу, групи КІ2м-22-1

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у Хмельницькому національному університеті» від 01.07.2022, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений(а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

22 квітня 2024 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач: Шудрик Андрій Олександрович

Тема: Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Спеціальність**:** 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи:

Кількість листів креслень _–_; кількість сторінок записки _76_

1. Короткий зміст роботи та прийнятих рішень _У роботі запропоновано метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

2. Висновок про відповідність роботи дипломному завданню _____ Кваліфікаційна робота магістра відповідає виданому завданню

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: _У першому розділі проведено огляд концепції туманних обчислень та її застосування у середовищі інтернету речей. Досліджено відомі рішення та засоби в цій сфері. У другому розділі визначено найбільш ефективний метод зменшення часу відгуку вузлів туманних обчислень. У третьому розділі запропоновано метод удосконалення планувальника завдань IoT системи із застосуванням туманних обчислень. У четвертому розділі запропоновано реалізацію засобів планування завдань в IoT інфраструктурі із застосуванням туманних обчислень на основі удосконаленого методу планування завдань в мережі туманних обчислень.

4. Позитивні сторони роботи: ___Запропонована система IoT із застосуванням туманних обчислень знаходить своє застосування у IoT системах із великою кількістю гетерогенних завдань різних класів складності.

5. Негативні сторони роботи: В першому розділі приділено недостатню увагу недолікам відомих методів оптимізації .

6. Оцінка графічного оформлення та пояснювальної записки роботи: –

7. Відгук про роботу в цілому: В загальному робота виконана на достатньому рівні.

8. Інші зауваження: –

9. Оцінка кваліфікаційної роботи:

Розглянувши позитивні та негативні сторони представленої кваліфікаційної роботи вважаю, що робота заслуговує оцінки «добре» 4.25 (B)

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) д.т.н.. професор, Бармак О.В., завідувач кафедри комп'ютерних наук

"13 травня" 2024р.

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОМАЦІЙНИХ СИСТЕМ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Метод оптимізації IoT інфраструктури із застосуванням туманних обчислень

Автор: Шудрик Андрій Олександрович

Спеціальність: 123 – Комп'ютерна інженерія

Освітня програма: освітньо-наукова

Науковий керівник: Лисенко С.М., д.т.н, професор

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як б'уде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
2) окремі виявлені збіги є загальновживаними фразами або виразами, про що свідчить посилання системи на збіг з джерелами на один фрагмент речення;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту. Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ідентичності/схожості Unicheck, складає 3.51% і адресується до 255 першоджерел; та системою Anti-Plagiarism складає 1%, що, з урахуванням наведених обгрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ С. М. Лисенко

Гарант ОП _____ О. С. Савенко

Завідувач кафедри КІС _____ Т. О. Говорущенко