

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

**Інститут прикладного системного аналізу**

**Кафедра системного проектування**

«На правах рукопису»  
УДК \_\_\_\_\_

До захисту допущено:  
Завідувач кафедри  
\_\_\_\_\_ А.І. Петренко  
«\_\_\_» \_\_\_\_\_ 2021 р.

**Магістерська дисертація  
на здобуття ступеня магістра  
за освітньо-науковою програмою «Інтелектуальні сервіс-орієнтовані  
розділені обчислювання»  
зі спеціальності 122 «Комп’ютерні науки»  
на тему: «Організація міграції мікросервісів в адаптивній туманній  
платформі обробки даних медичних датчиків»**

Виконав (-ла):

студент (-ка) 6 курсу, групи ДА-91МН  
Борисов Руслан Олександрович \_\_\_\_\_

Керівник:

Завідувач кафедри, д.т.н., професор,  
Петренко Анатолій Іванович \_\_\_\_\_

Рецензент:

Завідувач кафедри, д.б.н., к.т.н., с.н.с.  
Настенко Євген Арнольдович \_\_\_\_\_

Засвідчую, що у цій магістерській  
дисертації немає запозичень з праць  
інших авторів без відповідних посилань.  
Студент \_\_\_\_\_

Київ – 2021 року

**Національний технічний університет України**  
**«Київський політехнічний інститут імені Ігоря Сікорського»**  
**Інститут прикладного системного аналізу**  
**Кафедра системного проектування**

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 «Комп’ютерні науки»

Освітньо-наукова програма «Інтелектуальні сервіс-орієнтовані розподілені обчислювання»

ЗАТВЕРДЖУЮ

Завідувач кафедри

\_\_\_\_\_ А.І. Петренко

«\_\_\_\_» \_\_\_\_\_ 2021 р.

**ЗАВДАННЯ**  
**на магістерську дисертацію студенту**  
**Борисову Руслану Олександровичу**

1. Тема дисертації «Організація міграції мікросервісів в адаптивній туманній платформі обробки даних медичних датчиків», науковий керівник дисертації Петренко Анатолій Іванович, д.т.н., професор, затверджені наказом по університету від «17» березня 2021 р. № 851-с

2. Термін подання студентом дисертації \_\_\_\_\_

3. Об’єкт дослідження:

Генетичний алгоритм NSGA-II

4. Предмет дослідження

Ефективність генетичного алгоритму NSGA-II як засобу організації міграції мікросервісів в адаптивній туманній платформі

5. Перелік завдань, які потрібно розробити

1. Огляд засобів забезпечення адаптивності інформаційних систем;
2. Розробка моделей симуляції;
3. Аналіз результатів симуляції;
4. Розробка стартап-проекту.

**6. Орієнтовний перелік графічного (ілюстративного) матеріалу**

Презентація на тему «Організація міграції мікросервісів в адаптивній туманній платформі обробки даних медичних датчиків»

**7. Орієнтовний перелік публікацій**

**9. Дата видачі завдання 01.02.2021**

**Календарний план**

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	01.02.2021	
2	Огляд засобів забезпечення адаптивності	15.02.2021	
3	Розробка моделей симуляції	05.04.2021	
4	Аналіз результатів симуляції	19.04.2021	
5	Розробка стартап-проекту	26.04.2021	
6	Оформлення дипломної роботи	10.05.2021	
7	Отримання доступу до захисту та подача роботи в ДЕК	12.05.2021	

Студент

Борисов Р.О.

Науковий керівник

Петренко А.І.

# **РЕФЕРАТ НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ**

виконану на тему: «Організація міграції мікросервісів в адаптивній туманній платформі обробки даних медичних датчиків»

Магістерська дисертація виконана на 110 сторінках, містить 32 ілюстрації та 30 таблиць. При підготовці дисертації використано літературу з 43 джерел.

## **Актуальність теми**

Вимоги низької затримки, безпеки при обробці та зберіганні даних, необхідність зменшення навантаження на комп’ютерну мережу роблять привабливим розгортання програмних додатків у туманних обчислювальних системах. Але у випадку, коли користувачі додатків мають властивість мобільності та можуть змінювати своє місце положення, виникає необхідність підтримувати відповідну якість обслуговування для кожного місцезнаходження користувача. Актуальним є дослідження механізму для забезпечення міграції контексту клієнту та модулів додатків між обчислювальними пристроями як способу вирішення цієї проблеми.

## **Мета і завдання дослідження**

Метою роботи є дослідження ефективності механізму організації інтелектуальної міграції для моделі адаптивної туманної платформи з медичним додатком. Необхідно адаптувати та дослідити ефективність багатоцільового генетичного алгоритму NSGA-II як основи для організації міграції.

## **Рішення поставлених завдань і досягнуті результати**

В ході роботи було визначено загальну трирівневу архітектуру адаптивної туманної платформи для додатків із мобільними користувачами, створено її модель для проведення подальшого дослідження. Описано схеми інжекції ідеалізованих рішень в початкову популяцію та зменшення розмірності

множини рішень алгоритму з метою покращення результатів його роботи. Запропоновано розглядати механізм організації міграції з точки зору характеру ініціації процесу міграції, виділивши періодичну (заплановану) та реактивну міграції. Підготовлено сценарії симуляції для дослідження зазначених концепцій.

Згідно з виконаним аналізом результатів симуляцій, інжекція ідеалізованих рішень в початкову популяцію здатна суттєво покращити якість і швидкість роботи генетичного алгоритму в випадку одноцільової оптимізації. Також виправдано зменшення розмірності множини рішень генетичного алгоритму, розбиваючи множину всіх змінних системи на підмножини та проводячи оптимізацію для кожної з них. Продемонстровано можливість застосування NSGA-II для організації багатокритеріальної міграції на прикладі двох цільових функцій: вартості обчислень та середньої затримки.

## **Об'єкт досліджень**

Генетичний алгоритм NSGA-II.

## **Предмет досліджень**

Ефективність генетичного алгоритму NSGA-II як засобу організації міграції мікросервісів в адаптивній туманній платформі.

## **Наукова новизна та практичне значення**

Новизна та практична цінність роботи полягає в дослідженні можливості застосування генетичного алгоритму для організації міграції модулів додатків з мобільними користувачами в туманних plataформах.

## **Ключові слова**

Міграція додатків, туманні обчислення, генетичні алгоритми, МАРО.

# **РЕФЕРАТ НА МАГИСТЕРСКУЮ ДИССЕРТАЦИЮ**

выполненную на тему: «Организация миграции микросервисов в адаптивной туманной платформе обработки данных медицинских датчиков»

Магистерская диссертация выполнена на 110 страницах, содержит 32 иллюстрации и 30 таблиц. При подготовке диссертации использована литература из 43 источников.

## **Актуальность темы**

Требования низкой задержки, безопасности обработки и хранения данных, необходимость уменьшения нагрузки на компьютерную сеть делают привлекательным развертывание программных приложений в туманных вычислительных системах. Но в случае, когда пользователям приложений свойственна мобильность и возможность изменять свое положение, возникает необходимость поддерживать соответственное качество обслуживания для каждого местоположения пользователя. Актуальным является исследование механизма для обеспечения миграции модулей приложений между вычислительными устройствами в качестве способа решения данной проблемы.

## **Цель и задание исследования**

Целью работы является исследование механизма организации интеллектуальной миграции для модели адаптивной туманной платформы с медицинским программным приложением. Необходимо адаптировать и исследовать эффективность многоцелевого генетического алгоритма NSGA-II в качестве основы для организации миграции.

## **Решение поставленных задач и достигнутые результаты**

В ходе работы было определено общую трёхуровневую архитектуру адаптивной туманной платформы для приложений с мобильными пользователями, создано её модель для проведения дальнейшего исследования.

Описано схемы инжекции идеализированных решений в начальную популяцию и уменьшения размерности множества решений алгоритма с целью улучшения результатов его работы. Предложено рассматривать механизм организации миграции с точки зрения характера инициации процесса миграции, выделив периодическую (запланированную) и реактивную миграции. Подготовлено сценарии симуляции для исследования указанных концепций.

Согласно проведённому анализу результатов симуляций, инжекция идеализированных решений в начальную популяцию способна существенно улучшить качество и скорость работы генетического алгоритма в случае одноцелевой оптимизации. Также оправдано уменьшение размерности множества решений генетического алгоритма путём разделения множества всех переменных системы на подмножества и проведения оптимизации для каждой из них отдельно. Продемонстрировано возможность использования NSGA-II для организации многокритериальной миграции на примере двух целевых функций: цены вычислений и средней задержки.

### **Объект исследований**

Объектом исследований является генетический алгоритм NSGA-II.

### **Предмет исследований**

Эффективность генетического алгоритма NSGA-II в качестве средства организации миграции микросервисов в адаптивной туманной платформе.

### **Научная новизна и практическое значение**

Новизна и практическая ценность работы заключается в исследовании возможности использования генетического алгоритма для организации миграции моделей приложений с мобильными пользователями в туманных платформах.

### **Ключевые слова**

Миграция приложений, туманные вычисления, генетические алгоритмы, МАРО.

# **ABSTRACT ON MASTER'S THESIS**

on topic: “Organizing the microservices migration in adaptive fog platform for medical sensor data processing”

Master's thesis is expounded on 110 pages, consists of 32 figures and 30 tables. The paper references 43 distinct sources.

## **Topicality**

Demands of low latency, data processing and storing security, necessity of network traffic congestion decrease make it attractive to deploy applications in fog computing systems. However, in case when application users are mobile being able to change their location it is necessary to ensure the corresponding quality of service at each user's location. What is topical considering the given problem is a research on the mechanism for providing migration of both client context and application modules between computing devices.

## **Purpose and tasks**

The purpose of this paper is to investigate the efficiency of the intellectual migration organizing mechanism for the adaptive fog platform model with a medical application. The adjustment and effectiveness analysis of multi-objective genetic algorithm NSGA-II as a basis for migration organizing must be performed.

## **Solutions and achieved results**

In the scope of the paper the generic three-layer architecture of adaptive fog platform for applications with mobile users was designed, and its model with the aim of further researches was established. The schemes of idealized solutions injection into initial population and reduction of solutions set dimension were presented. It was suggested to consider the mechanism for migration organizing in the aspect of migration process initiation, including periodic (or planned) and reactive migrations.

According to the simulation results analysis, the idealized solutions injection into initial population can significantly improve quality and increase speed of genetic algorithm in case of single-objective optimization. Furthermore, the reduction of solutions set dimension by dividing the whole set of system variables into subsets and performing a separate optimization for each of them is justified. The ability of applying NSGA-II for organizing the multi-objective migration is illustrated for total cost and average latency minimization as target objectives.

### **Object of research**

The object of research is the NSGA-II genetic algorithm.

### **Subject of research**

The subject of research is the efficiency of NSGA-II genetic algorithm applied for organizing the microservice migration in adaptive fog platform.

### **Scientific novelty and practical value**

Scientific novelty and practical value of the paper consists in research of the application capabilities of the genetic algorithm for organizing the process of migration for modules of applications with mobile users in fog platforms.

### **Keywords**

Application migration, fog computing, genetic algorithm, MAPO.

## ЗМІСТ

ВСТУП .....	12
1 ОГЛЯД ЗАСОБІВ ЗАБЕЗПЕЧЕННЯ АДАПТИВНОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ .....	14
1.1 Визначення поняття адаптивності системи.....	14
1.2 Класичні підходи забезпечення адаптивності.....	15
1.2.1 Метод PDCA .....	16
1.2.2 Метод OODA .....	18
1.3 Сучасні підходи забезпечення адаптивності інформаційних систем .....	20
1.3.1 Цикл MAPE.....	20
1.3.2 Класифікація систем забезпечення адаптивності .....	23
1.3.3 Узгодження паралельних додатків та систем забезпечення адаптивності.....	25
1.3 Технології забезпечення адаптивності розподілених систем обробки даних в умовах мобільності користувачів .....	30
1.3.1 Туманні обчислення для забезпечення адаптивності.....	31
1.3.2 Міграція модулів додатку .....	32
1.3.3 Організація інтелектуальної міграції .....	34
1.3.4 Мікросервісна архітектура інформаційних систем .....	37
1.3.5 Архітектура адаптивної платформи .....	39
1.4 Висновки до розділу 1 .....	41
2 РОЗРОБКА МОДЕЛЕЙ СИМУЛЯЦІЇ .....	42
2.1 Вхідні дані та умови функціонування платформи.....	43
2.1.1 Модель медичного додатку.....	43

2.1.2 Мобільність користувачів платформи .....	47
2.1.3 Розміщення базових станцій оператора стільникового зв'язку .....	48
2.1.4 Визначення параметрів апаратного забезпечення платформи .....	52
2.1.4.1 Мережеві параметри .....	54
2.1.4.2 Параметри обчислювальних пристройів .....	58
2.2 Опис сценаріїв моделювання .....	62
2.2.1 Дослідження інжекції ідеалізованих рішень .....	62
2.2.2 Дослідження міграції на основі генетичного алгоритму .....	65
2.2.3 Сценарій багатоцільової оптимізації .....	67
2.3 Висновки до розділу 2 .....	69
<b>3 АНАЛІЗ РЕЗУЛЬТАТІВ СИМУЛЯЦІЇ.....</b>	<b>70</b>
3.1 Аналіз ефективності інжекції ідеалізованих рішень .....	70
3.2 Аналіз міграції на основі генетичного алгоритму.....	76
3.3 Аналіз результатів багатоцільової оптимізації .....	82
3.4 Висновки до розділу 3 .....	85
<b>4 РОЗРОБКА СТАРТАП-ПРОЄКТУ «АДАПТИВНА ПЛАТФОРМА ДЛЯ МЕДИЧНИХ ДОДАТКІВ».....</b>	<b>87</b>
4.1 Опис ідеї проєкту .....	87
4.2 Технологічний аудит ідеї проєкту.....	89
4.3 Аналіз ринкових можливостей .....	89
4.4 Розробка ринкової стратегії стартап-проєкту .....	97
4.5 Розробка маркетингової програми стартап-проєкту .....	99
4.6 Висновки до розділу 4 .....	102
<b>ВИСНОВКИ.....</b>	<b>103</b>
<b>ПЕРЕЛІК ПОСИЛАНЬ .....</b>	<b>106</b>

## ВСТУП

На сьогоднішній день можна стверджувати, що одною з ключових характеристик галузі інформаційних технологій є її всепроникність як у соціальне та побутове життя громадян, так і в сфері підприємницької діяльності. Однією з рушійних сил в даному контексті є парадигма Інтернету речей, яка ставить на меті забезпечити взаємодію з навколошнім світом шляхом створення та управління віртуальним представленням фізичних речей в інформаційному просторі. Прийнято конкретизувати парадигму Інтернету речей в контексті певної галузі його застосування, як-от індустріальний Інтернет речей для фабричного виробництва або медичний Інтернет речей для охорони здоров'я громадян. Щільно пов'язаною з Інтернетом речей концепцією є хмарні обчислення.

Але із подальшим технічним розвитком стало можливим створювати сценарії Інтернету речей, які вимагають низької затримки обробки даних або здатні генерувати значну кількість даних. Зокрема, з появою технології мобільного зв'язку 5G [1], яка значно зменшує затримку та збільшує пропускну спроможність бездротового зв'язку, у перспективі в галузі охорони здоров'я буде можливо реалізувати операції у віддаленому режимі, аналіз показників пацієнта в реальному часі тощо. Але із можливістю забезпечити низьку затримку з технологією 5G, все більш критичною стає затримка, утворена зв'язком до хмарного центру обробки даних. Сучасним підходом для подолання даної перешкоди є надання переваги туманним обчисленням замість хмарних.

Одним із аспектів, які підлягають врахуванню в туманних розподілених обчислювальних системах, в яких виконуються додатками з вимогою до низької затримки, пов'язаний з мобільністю кінцевих користувачів. Виникає необхідність підтримувати заданий рівень якості обслуговування для користувача в процесі постійної зміни його положення. Рішення даної проблеми є організація процесу міграції в першу чергу контексту користувача,

а в другу чергу програмних додатків між обчислювальними пристроями відповідно до переміщення користувача. В останній час детально досліджується як сама організація міграції [2, 3], так і вплив мобільності користувачів на характеристики системи [4].

Метою даної роботи є дослідження механізму організації інтелектуальної міграції на основі генетичного алгоритму NSGA-II в межах моделі адаптивної туманної платформи для медичних додатків. Алгоритм вже було успішно застосовано для вирішення задачі оптимального розміщення модулів додатків у туманній системі [5], а його особливістю є застосування в задачах багатоцільової оптимізації, що робить можливим враховувати декілька критеріїв якості обслуговування для адаптивної платформи. У роботі проаналізовано доцільність використання та запропоновано шляхи адаптації генетичного алгоритму для вирішення задачі організації міграції.

Робота складається з чотирьох розділів. У першому розділі приділено увагу поняттю адаптивності для туманної платформи, визначено її загальну архітектуру та стек технологій, який забезпечуватиме її роботу. В другому розділі досліджено і запропоновано модифікації механізму організації міграції на основі генетичного алгоритму NSGA-II. Для цього було створено модель адаптивної платформи з медичним додатком та підготовлено три сценарії для симуляції її роботи. У третьому розділі проведено аналіз отриманих в ході симуляції результатів, зроблено висновок про ефективність запропонованих модифікацій та перспективи реального втілення даного механізму організації міграції. У четвертому розділі висунуто ідею та створено відповідну документацію для стартап-проекту адаптивної платформи. В кінці підбито підсумки виконаної роботи.

# 1 ОГЛЯД ЗАСОБІВ ЗАБЕЗПЕЧЕННЯ АДАПТИВНОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ

## 1.1 Визначення поняття адаптивності системи

Адаптивність будь-якої системи – це властивість змінювати свою поведінку та/або структуру у процесі функціонування під впливом зовнішнього середовища чи внутрішніх факторів [6]. Під впливом збуджуючих факторів можуть змінюватися параметри компонентів системи, зникати та виникати нові компоненти та зв'язки між ними. Ключовим поняттям адаптивної поведінки є ланцюг зворотного зв'язку, який відповідає за введення причино-наслідкових відносин усередині системи (рисунок 1.1).

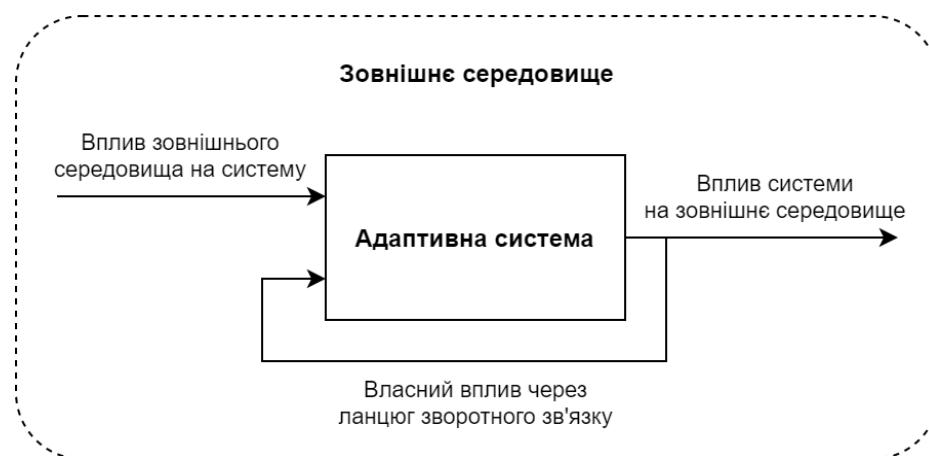


Рисунок 1.1 – Загальна схема адаптивної системи

Зокрема, для інформаційних систем визначення адаптивності залишиться тим самим. Однак, підлягає уточненню, що є зовнішнім середовищем для адаптивної інформаційної, з яких компонентів вона може складатися та яким чином можна реалізувати механізм зворотного зв'язку. Відповідно до наведеної у [6] метамоделі, зовнішнє середовище складається із так званих «сущностей», кожна з яких зберігає власний стан, який в свою чергу може змінюватись під час функціонування системи. Сущностей запропоновано поділити на:

- Розумні сутності – такі, що діють цілеспрямовано та певною мірою раціонально. У цьому середовищі адаптивна система є якраз такою розумною сутністю. До них можна віднести сторонні сутності, які діють в межах даного середовища, як-от інтелектуальні агенти, особи, що приймають рішення;
- Примітивні сутності – сутності, що не мають ознак інтелектуальності, виконують дії за наперед визначенім алгоритмом або являють собою певного роду ресурси. До них відносять фізичні пристрой (наприклад пристрой обробки даних, пристрой зберігання даних, мережеве обладнання тощо) та програмні компоненти (операційні системи, різноманітні додатки);
- Пасивні сутності – інші цифрові або фізичні об'єкти. Доцільно висунути припущення, що з ними можуть взаємодіяти розумні та примітивні сутності, але вони власне не виконують активних дій.

## **1.2 Класичні підходи забезпечення адаптивності**

Класичні підходи до забезпечення адаптивності ґрунтуються на працях другої половини двадцятого століття, в яких сформульовані досить універсальні принципи. Підтвердженням універсальності є їхнє широке застосування у різноманітних галузях людської діяльності, починаючи від управління підприємством, контролю якості продуктів та закінчуючи стратегічним плануванням у військовій справі. В основі багатьох підходів лежить ітераційний процес, який в загальному випадку дозволяє реагувати на зміну обстановки та поступово знаходити оптимальний режим функціонування, а у випадку конкретного застосування в якості механізму адаптивності дозволяє підлаштовуватись до впливу зовнішнього середовища та переходити в оптимальний стан функціонування. Далі на прикладах методів PDCA, OODA та MAPE будуть сформульовані основні властивості та характерні риси такого типу підходів.

### 1.2.1 Метод PDCA

Метод PDCA являє собою ітеративний процес із чотирьох складових, що може застосовуватися для контролю та постійного покращення якості операційних процесів та продукції [7]. Ідея, що лежить в основі методу, була сформульована Френсісом Беконом у 1620 році в якості методу наукового пізнання як послідовних стадій висування гіпотези, проведення експерименту, перевірки гіпотези. Адаптацію цієї ідеї у 1920-х роках здійснив Волтер Шухарт, який для покращення якості продукту на виробництві запропонував послідовність розробки специфікації, виробництва і приймального контролю. Нарешті, популярність принцип здобув завдяки роботі Вільяма Едвардса Демінга 1986 року, в якій він спочатку називав метод Циклом Шухарта, але свою назву PDCA отримав дещо пізніше.

Перед використанням методу PDCA підлягають визначення наступні питання [8]:

- Що необхідно досягти: формулюється мета/ціль застосування методу;
- Як з'ясувати, що внесена зміна є покращенням: визначаються метрики оцінювання об'єкту;
- Які зміни можна вносити: визначаються концепції та методики внесення змін у об'єкт;

Складовими PDCA є чотири процеси:

- Plan (планування) – визначаються задачі та процеси, які потенційно дозволять наблизитись до заданої цілі;
- Do (виконання) – втілюються заплановані на попередньому етапі заходи, розпочинається аналіз даних;
- Check (перевірка) – розраховуються заздалегідь визначених метрик об'єкту, завершується збір та аналіз даних, здійснюється порівняння очікуваних та отриманих результатів, отримуються нові знання про об'єкт;

- Act/Adjust (виконання/пристосування) – робляться висновки стосовно виконаних операцій, затверджуються у вигляді стандартів або відхиляються проведені зміни, проводяться підготовчі заходи для наступної ітерації, приймається рішення щодо поточної реалізації об'єкту (продукту).

Важливим елементом методу PDCA є його ітеративний характер (рисунок 1.2). Таким чином реалізується механізм зворотного зв'язку, який дозволяє врахувати результати та застосувати нові знання, отримані на попередніх ітераціях. Враховуючи це, метод PDCA може бути використаний в якості механізму адаптації для певного виду систем.

Однак, при його втіленні в інформаційну систему виникає необхідність у компоненті з високим рівнем раціональності, яка мала б проявлятися в першу чергу на стадії планування, а в другу – на стадії пристосування. Альтернативою могла бстати заздалегідь спроектована база знань, набір стратегій тощо. Але також слід зауважити, механізм стандартизації як закріплення досягнутого рівня якості продукту припускає стабільність умов, в яких перебуває об'єкт, з точки зору зовнішнього впливу. Таким чином, PDCA застосовується більшою мірою ефективно, коли період одного циклу коротший у порівнянні з тривалістю впливу зовнішнього середовища на цільовий об'єкт.

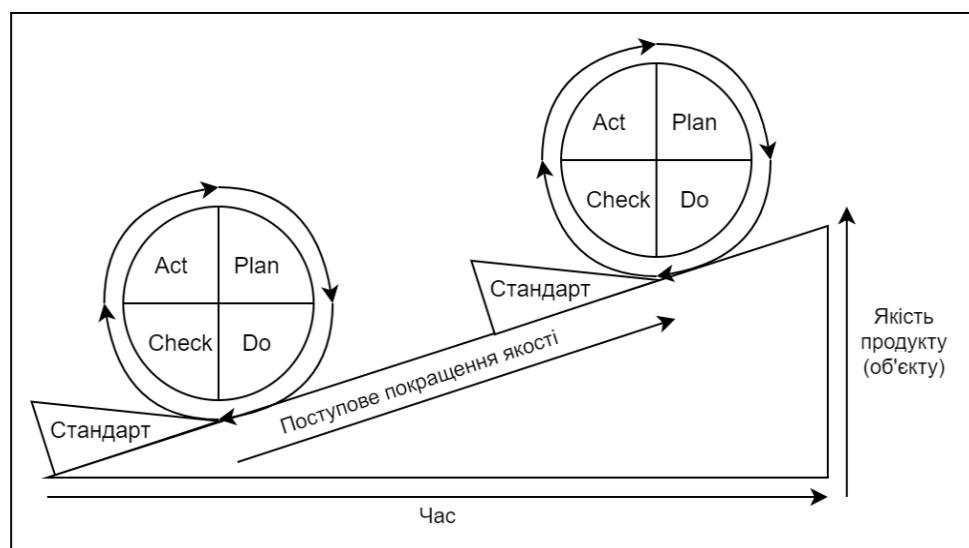


Рисунок 1.2 – Застосування методу PDCA для покращення якості об'єкту

### 1.2.2 Метод OODA

Аналогічно попередньому методу, OODA є також складається з чотирьох послідовних етапів, які повторюються від ітерації до ітерації. OODA був сформульований [9] Джоном Бойдом, полковником повітряних сил США, 1995 року. Першочергово метод застосовувався як елемент військової стратегії, але концепції, закладені в нього, знайшли широке застосування в інших галузях, де важливе місце займає процес прийняття рішень.

Метод OODA, схематично зображений на рисунку 1.3, включає наступні стадії [10]:

- **Observe** (спостереження) – отримання інформації про поточну ситуацію;
- **Orient** (орієнтування) – осмислення отриманої на попередньому етапі інформації.
- **Decision** (прийняття рішення) – елемент розгалуження, в межах якого приймаються рішення про перехід на наступну стадію або здійснення повторного спостереження чи орієнтування.
- **Action** (виконання) – втілення дій, що чинять вплив на систему.

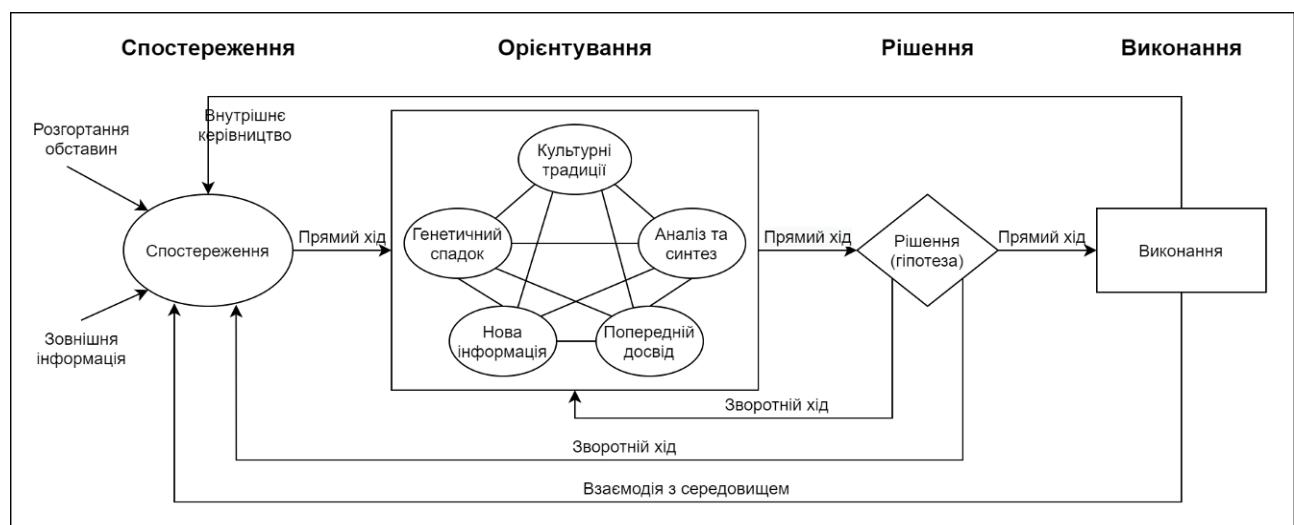


Рисунок 1.3 – Схема методу OODA [10]

На стадії спостереження джерелами цієї інформації є люди, результати тестів та моделювання, та інші дані. Важливим є зауваження, що для інформації, яка підлягає збору на даній стадії, характерні динамічний еволюційний розвиток, наявність протиріч та мінливість, неточність та недостовірність, неповність, залежність від суб'єкту, який здійснює спостереження.

На процес осмислення інформації людьми впливають попередній досвід, здатності до аналізу та синтезу, культурні традиції і генетичний спадок. Для підвищення ефективності даної стадії можна виділити наступні способи. По-перше, розголошення прихованого знання, яким володіють керівництво, може позитивно вплинути на ефективність оцінки інформації суб'єктом, сприяти приверненню уваги до дійсно важливої інформації тощо. По-друге, доцільним, хоча і не завжди можливим способом є застосування формалізованих методів і моделей обробки даних, зокрема теорії ймовірності та математичної статистики, методів нечіткої логіки, елементів теорії прийняття рішень. Це дозволить враховувати мінливість, неточність і нечіткість інформації. Нарешті, вже на цій стадії слід враховувати всі можливі варіанти дій на стадії виконання.

Стадія прийняття рішень включає стандартні заходи: формулювання набору альтернатив, їхня оцінка на системою критеріїв, вибір оптимальної. Їм передують підготовчі процеси для визначення, які області задачі слід взяти до уваги в першу чергу, та узагальнення поточних умов для можливості їх повного осмислення. Ефективність методу OODA визначається виконанням прийнятих рішень як реакції на внутрішні і зовнішні обставини.

У порівнянні з описаним вище методом PDCA, OODA має ряд властивостей, які роблять його використання в якості механізму адаптивності інформаційних систем більш доцільним. По-перше, оригінальна ідея ґрунтується на поняттях оперативності та динаміки: в ситуації протистояння двох сторін, та сторона, що виконує ітерації циклу швидко і якісно, досягає переваги. Таким чином, адаптивність до зовнішніх умов може забезпечуватись оперативним спостереженням за їх змінами та відповідним реагуванням. По-

друге, гнучкість реагування забезпечується більшою кількістю ланцюгів зворотного зв'язку, адже на стадії прийняття рішень можна повернутись до стадії спостереження і орієнтування. По-третє, той факт, що метод PDCA починається з висування гіпотез і закінчується вимірюванням, свідчить про певною мірною апріорний підхід до вирішення поставленої задачі (в тому числі адаптації). Натомість реакція на зовнішні і внутрішні чинники в OODA має апостеріорний характер, тобто здійснюється на основі отриманої інформації про них. Такий підхід, орієнтований на дані, є природним для інформаційних систем, що робить OODA більш привабливим при реалізації у них адаптивності. Нарешті, в методі OODA заохочується широке використання формалізованих методів обробки, аналізу даних і прийняття рішень, що є перевагою у випадку використання обчислювальних систем. Недолік цього методу [10] проявляється, коли на етапі прийняття рішень регулярно вирішується повернутися на стадію спостереження, дуже рідко виконуючи активні дії.

### **1.3 Сучасні підходи забезпечення адаптивності інформаційних систем**

#### **1.3.1 Цикл МАРЕ**

Ключові ідеї, покладені в основу підходу під назвою «цикл МАРЕ», дуже нагадують OODA. Навіть можна вважати, що МАРЕ є певною мірою реалізацією останнього, призначеного для використання конкретно в розподілених інформаційних системах. Між складовими циклу МАРЕ [11], наведеними на рисунку 1.4, є повна відповідність з основними етапами OODA:

- Monitor (моніторинг) – отримання даних про поточний статус паралельного додатку та стану обчислювальних вузлів, на яких він виконується. Такими даними можуть бути кількість проміжних задач, що виконуються або вже були виконані на обчислювальних вузлах, кількість використаних ресурсів, наприклад завантаженість

процесорів, об'єм використаної оперативної та постійної пам'яті, навантаження на мережу тощо;

- Analyze (аналіз) – проводиться аналіз поведінки паралельного додатку, порівнюється із очікуваним;
- Plan (планування) – приймається рішення щодо необхідної стратегії регулювання;
- Execute (виконання) – втілення обраної стратегії.

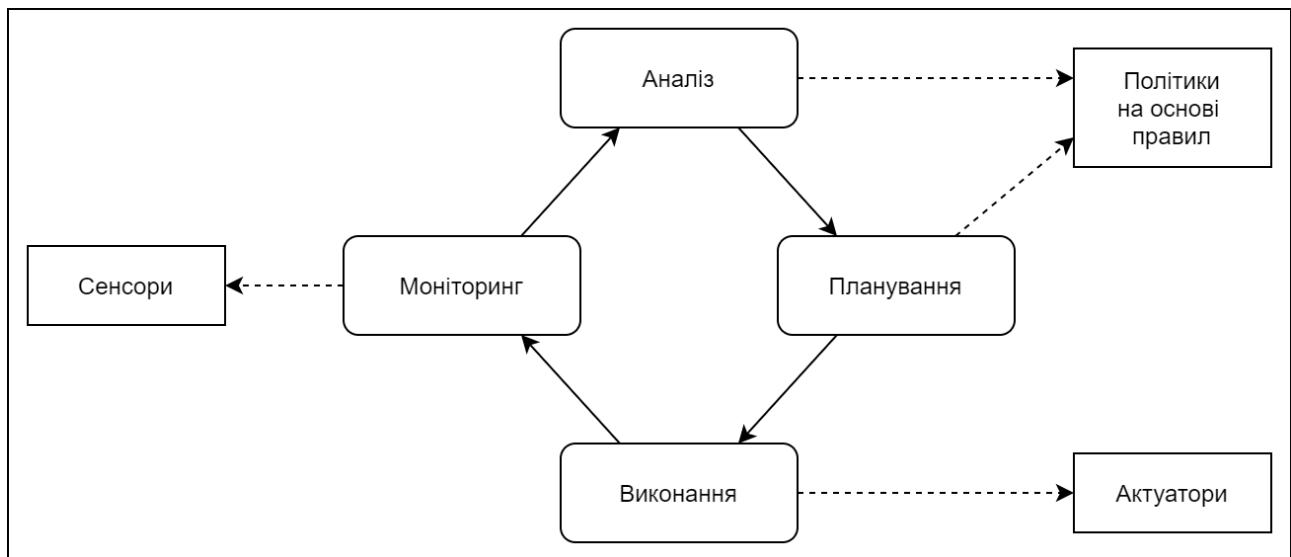


Рисунок 1.4 – Цикл МАРЕ для адаптивності системи [11]

Рушієм активних заходів в циклі МАРЕ є необхідність забезпечення нефункціональних вимог інформаційної системи. В той час як функціональні компоненти системи зосереджені на виконанні бізнес-логіки додатку та відповідають на питання, що саме має виконуватися, нефункціональні вимоги визначають, як саме має відбуватися це виконання. До нефункціональних вимог відносять наступні [11]:

- Продуктивність – чи не найбільш суттєва вимога у розподілених системах вимога, може полягати в мінімізації чи то мережевої затримки при передачі даних, чи то загального часу обслуговування;

- Безпека – у контексті параметру, що підлягає керуванню, означає вибір вузлів передачі, обробки та збереження даних користувача відповідно до чутливості цих даних: наприклад, відкриті дані можуть оброблятися у публічних ЦОД, а конфіденційні дані – як найближче до користувача або на приватних серверах;
- Відмовостійкість – у масштабних обчислювальних комплексах, задіяних у роботу паралельного додатку, висока ймовірність виходу зі строю хоча б одного апаратного модуля. Тому доцільно використовувати різноманітні заходи для забезпечення коректної роботи системи, як-от алгоритми з контрольними точками, введення елементів збитковості тощо;
- Управління енергоспоживанням – для багатьох пристройів, особливо в яких відсутнє підключення до зовнішнього постійного джерела енергії, є критичним і підлягає оптимізації об'єм електроенергії, що споживається під час виконання додатку.

Активності в основу циклі МАРЕ залежать від компонентів трьох типів: сенсори, актуатори та політики. Сенсори дозволяють вимірювати основні показники як власне запущеного додатку, так і обчислювальних вузлів, задіяних у його виконання. Аaktuатори у свою чергу є певним механізмом, за допомогою якого можна впливати на параметри компонентів системи: призупиняти та відновлювати виконання, переміщувати програмні компоненти між вузлами, залучати та звільнити ресурси (процесори, пам'ять, частину пропускної спроможності мережевого каналу). Рішення щодо управління нефункціональними вимогами приймається на основі політик: правил поведінки системи в певній ситуації, спроектованих заздалегідь або генерованих під час функціонування системи.

Таким чином, основний цикл в методі МАРЕ значною мірою співпадає з наведеним вище методом OODA. Особливістю МАРЕ є його безпосередня орієнтованість на вирішення задачі адаптивності в розподілених інформаційних

гетерогенних системах. Аналогічно ООДА, при своєчасному, постійному та оперативному виконанні цикла МАРЕ можна здійснювати адаптивний контроль нефункціональних вимог під час надання користувачам інформаційних послуг.

### **1.3.2 Класифікація систем забезпечення адаптивності**

Окрім методу МАРЕ з його варіаціями існують інші методи забезпечення адаптивності. Одні методи намагаються уточнити або модифікувати етапи циклу, в тому числі здійснюючи структурну декомпозицію; інші вдосконалюють метод, доповнюючи його новими концепціями. Із цим різноманіттям доцільно здійснити класифікацію адаптивних систем, абстрагувавшись від методів її забезпечення.

Таку класифікацію під назвою «метамодель» було запропоновано Сабатучі та ін. [6]. Метамодель виділяє 4 типи (рисунок 1.5) адаптивних систем залежно від присутності певних компонентів та збільшення рівня їхньої складності інтелектуальності. Кожна система наступного типу має у своєму складі елементи попереднього типу.

До типу 1 належать найпростіші системи для реалізації адаптивності. Вони вимагають визначення всіх можливих станів зовнішнього середовища та відповідних реакцій ще на стадії проєктування. Під час функціонування системи дані про зовнішнє середовище надходять через Env Monitor, а вихідний вплив здійснюється через Effector. Таким чином, використання систем первого типу доцільно за умов, коли кількість станів зовнішнього середовища є невеликою. Крім того, ці системи не рекомендується використовувати саме для нефункціональних вимог до інформаційних систем та за необхідності гнучких стратегій при рішенні задач.

Тип 2 включає більш складні системи, які розширяють тип 1, додаючи можливість обирати певну стратегію реакції на зовнішні умови вже у процесі функціонування. У свою чергу це вимагає знання про функціональні та нефункціональні аспекти додатку та вплив на них потенційних операцій, але звільняє від необхідності прораховувати всі можливі стани середовища на етапі

проектування. Системи типу 2 використовуються в ситуаціях, коли у реальному часі необхідно забезпечувати відповідність заздалегідь визначеним вимогам обслуговування. Але такі системи не можуть функціонувати в умовах динамічних функціональних або нефункціональних вимог.

Адаптивні систему типу 3 здатні функціонувати, коли знання про зовнішнє середовище є неповним, а вимоги до додатку заздалегідь невідомі та змінюються. Характерним компонентом цих систем є SolutionBuilder, який на основі репозиторію операцій здатний будувати нові стратегії для вирішення непередбачуваних проблем у процесі виконання систем. Обмеженням для таких механізмів адаптивності є, по-перше, застосування в якості компонентів систем реального часу та, по-друге, здатність рефлексивного дослідження та автономного вдосконалення доступних операцій.

До типу 4 належать найбільш складні та інтелектуальні адаптивні системи. Для них характерна спроможність аналізувати власну поведінку та вносити зміни у власний процес виконання. Теоретично, їхнє застосування відбувається в умовах, коли відсутні повна інформація і щодо закономірностей поведінки зовнішнього середовища, і щодо набору стратегій та їхнього впливу. Натомість, системи здатні постійно генерувати, переглядати та вдосконалювати власні стратегії.

Таким чином, із ускладненням адаптивних систем простежується тенденція перенесення процесу отримання знання про зовнішнє середовище і власний вплив на нього зі стадії проектування системи до стадії її безпосереднього функціонування. Із підвищеннем інтелектуальності вони отримують можливість долати невизначеність та функціонувати на більш високому рівні абстракції. Також можна зауважити, що головний цикл у методі MAPE жодним чином не суперечить наведений класифікації. При заміні набору політик як компоненту систему на відповідний, можна отримати системи кожного з чотирьох типів. Сенсори і актуатори у методі MAPE відповідають компонентам Env Monitor та Effect, а інші виконують аналіз і планування,

результатом чого може бути вибір оптимальної стратегії відповідно до метамоделі.

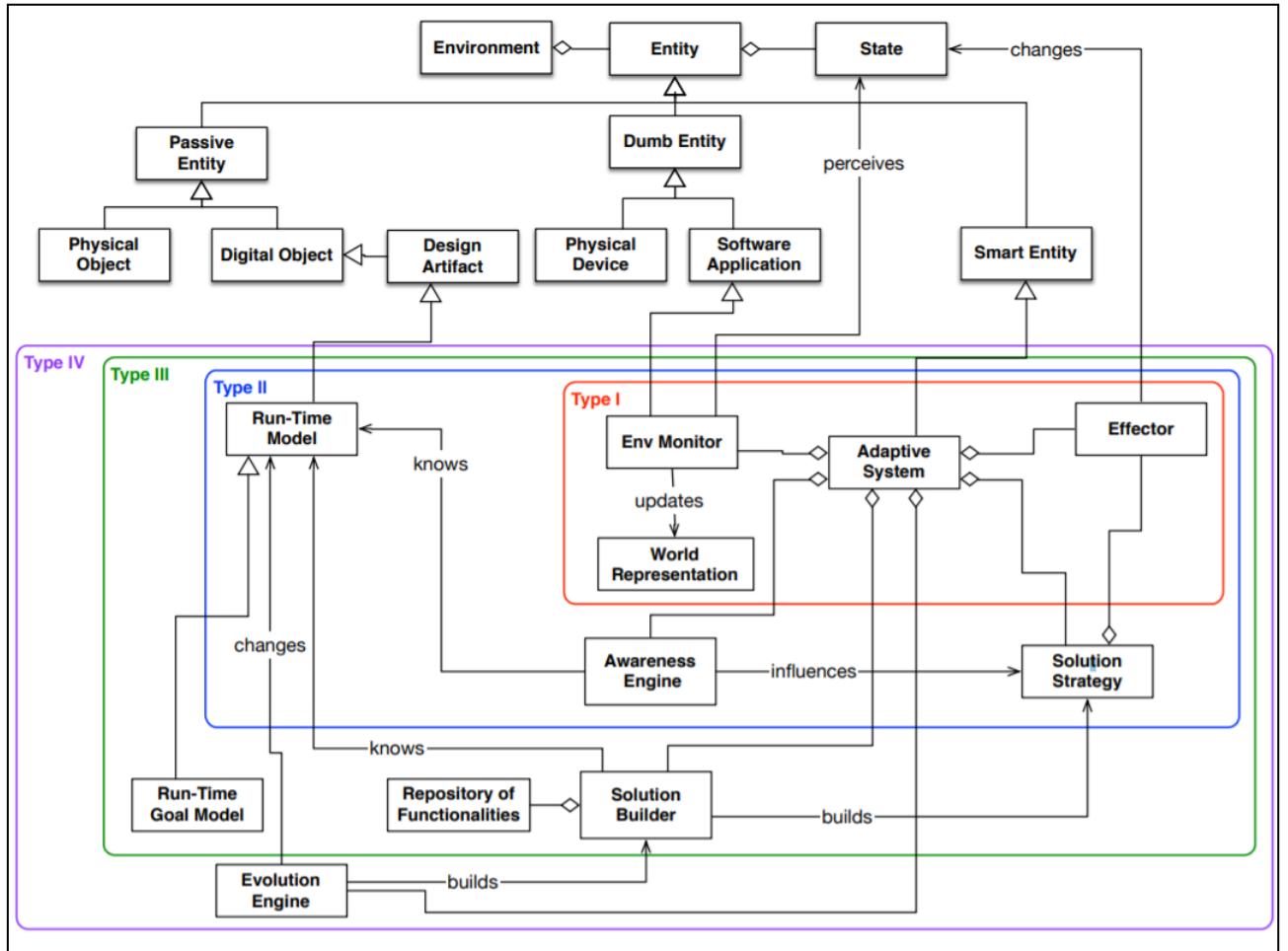


Рисунок 1.5 – Метамодель адаптивних систем [6]

### 1.3.3 Узгодження паралельних додатків та систем забезпечення адаптивності

Із ілюстрації метамоделі (рисунок 1.5) можна зробити висновок, що адаптивна система є окремим компонентом по відношенню до всього розгорнутого як апаратного, так і програмного комплексу. Сенсори і актуатори виконують роль інтерфейсу [11] між системою забезпечення адаптивності (системою управління певного роду) та її об'єктом – програмним додатком із

його функціональними та нефункціональними вимогами. Між ними двома, суб'єктом і об'єктом адаптації, має існувати узгодженість.

У контексті розподілених систем Інтернету речей було виділено чотири шаблони залежно від характеру розподілення [12]. Класично, в додатку Інтернету речей можна виділити декілька рівнів, серед яких важливо виділити:

- Perception layer (рівень сприйняття) – на цьому сенсори, під'єднані до фізичних речей, збирають дані, а актуатори, виконуючи операції, здійснюють вплив на фізичний світ;
- Processing & Storage layer (рівень обробки та збереження даних) – забезпечують доступ до зібраної на попередньому рівні інформації;
- Application layer (рівень додатків) – визначається сервісами, які на основі даних проводять аналіз, підтримують прийняття рішень тощо;
- Business layer (рівень бізнесу) – на цьому рівні сервіси об'єднуються між собою у вигляді бізнес-процесів, що функціонують з певною метою.

Відповідно до взаємозв'язків, які існують між цими рівнями, можна виділити чотири типи архітектур (шаблонів, рисунок 1.6):

- Централізована – взаємодія відбувається лише між сусідніми рівнями. Центральним елементом виступає хмарний центр обробки даних, виключно через який можна отримати доступ до сервісів;
- Колаборативна – декілька централізованих компонентів взаємодіють між собою, щоб забезпечувати кращу якість послуг;
- З'єднані корпоративні мережі (Інtranети) – дані з елементу рівня сприйняття доступні як для локальних, так і для глобальних централізованих послуг. Якщо зв'язок з центральним компонентом зникає, локальні послуги все ще залишаються доступними. Але елементи на рівні додатків, розміщені у межах корпоративної мережі, не можуть бути використані ззовні;

- Повністю розподілена – зв'язки існують між усіма рівнями додатку Інтернету речей, що сприяє спільному обміну даними та користуванню спільними локальними ресурсами для досягнення загальної мети.

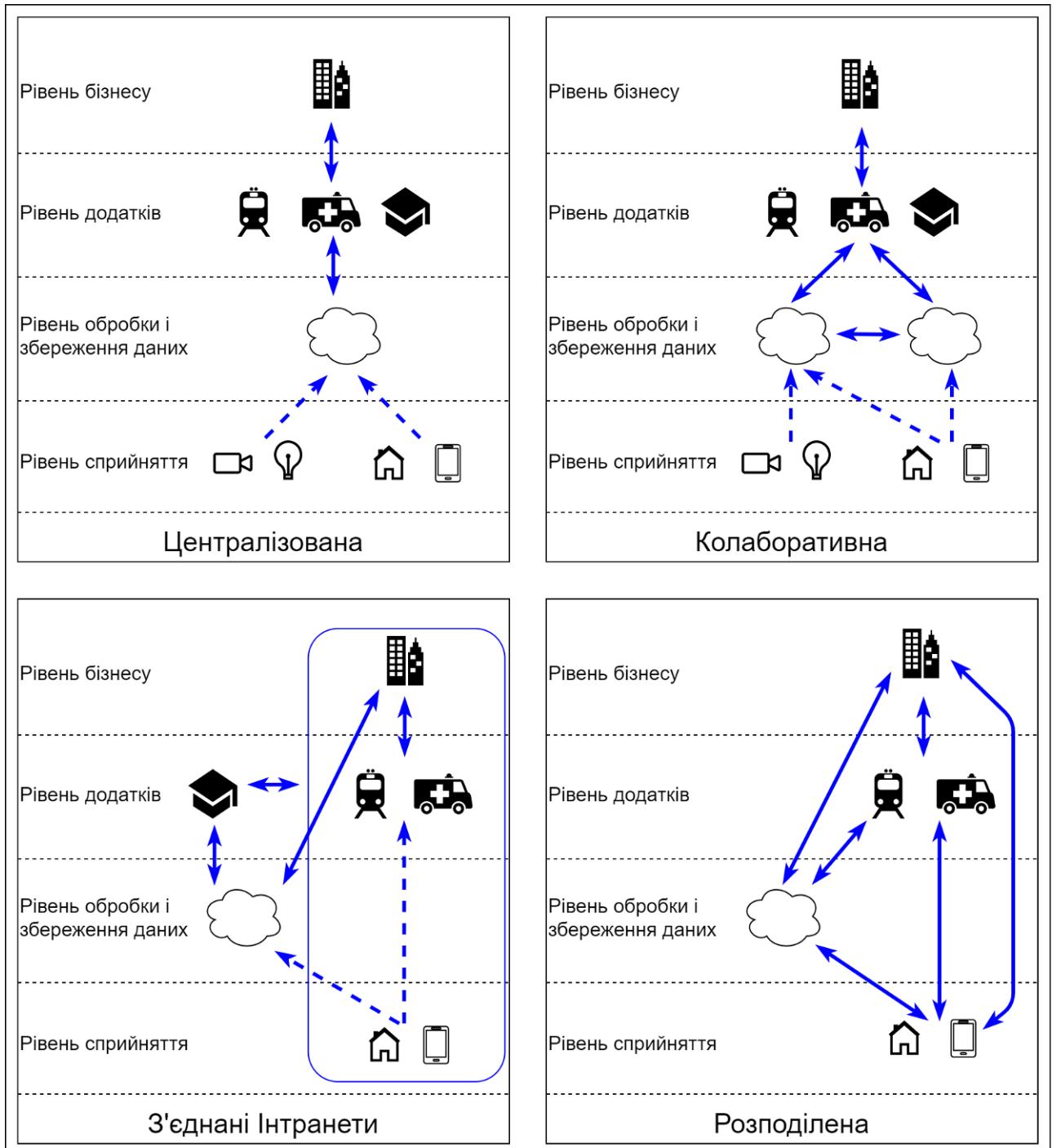


Рисунок 1.6 – Типи архітектур систем Інтернету речей [12]

Було виділено 6 можливих моделей взаємодії систем управління для забезпечення автономності керованих систем (КС) в залежності від характеру децентралізації [12, 13], проілюстрованих на рисунках 1.7 та 1.8:

- Централізована система – всі стадії циклу MAPE виконуються централізованому елементі;
- Система типу Master/Slave – стадії моніторингу та виконання відбуваються локально, а аналіз та планування здійснюються централізованим елементом;
- Система з розподілом інформації – на стадії моніторингу компоненти обмінюються даними, а інші стадії виконуються незалежно;
- Система з координацією – обмін даними відбувається на всіх стадіях циклу MAPE в розподіленій системі управління;
- Система з регіональним плануванням – зона розгортання системи поділяється на регіони, обмін інформацією відбувається лише на стадії планування з метою пошуку оптимальної глобальної стратегії;
- Ієрархічна система – будується ієархія циклів MAPE та адаптація здійснюється за принципом «розділяй і володарюй».

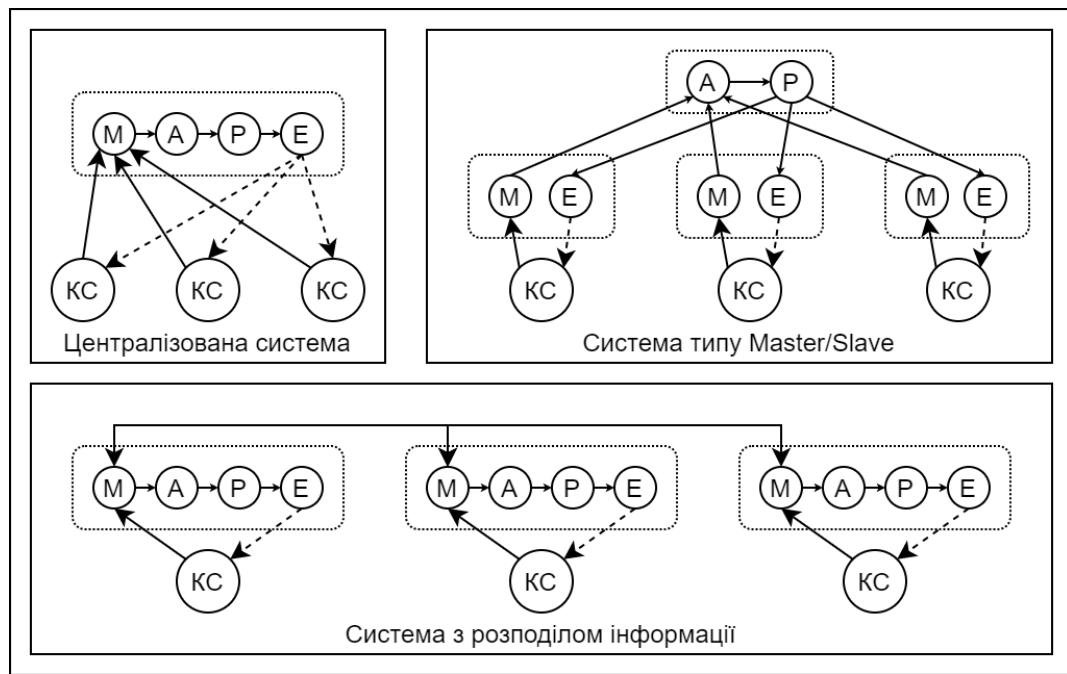


Рисунок 1.7 – Моделі систем керування для забезпечення адаптивності [12] (1)

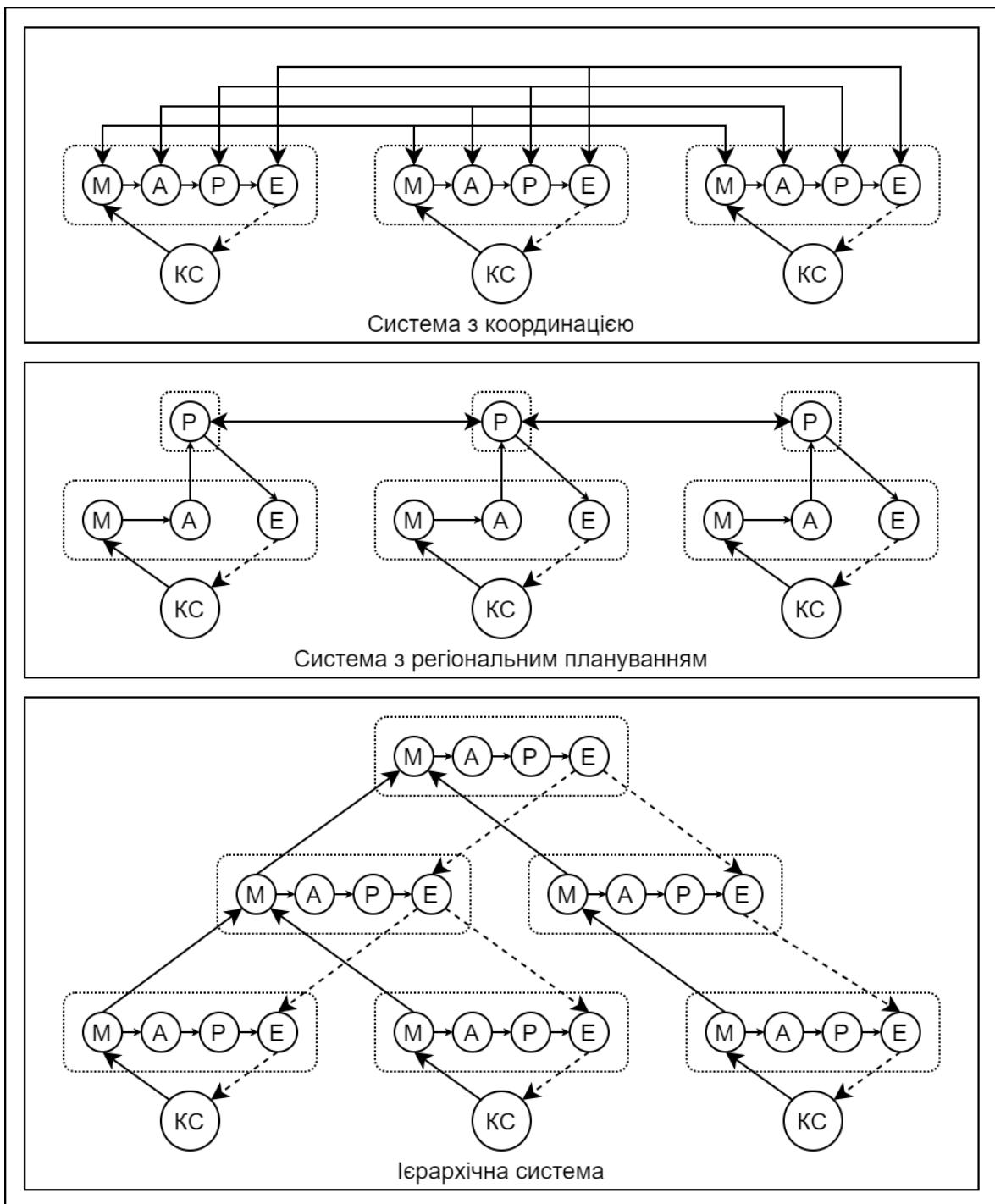


Рисунок 1.8 – Моделі систем керування для забезпечення адаптивності [12] (2)

Існує певна узгодженість між типами архітектур систем Інтернету речей та моделей систем керування для забезпечення їхньої адаптивності, наведена у таблиці 1.1. Її дотримання підвищить ефективність сумісного функціонування системи управління та керованої системи, а нехтування в результаті призведе до зменшення ефективності їх комбінації та, можливо, появи вразливих місць.

Таким чином врахування цього аспекту є вкрай важливим і у випадку, коли до існуючого додатку в межах концепції Інтернету речей впроваджують механізм адаптивності, і у випадку, коли для існуючої адаптивної платформи планується розгортання нового додатку.

Таблиця 1.1 – Узгодження архітектур IoT з моделями систем управління

		Архітектури Інтернету речей			
		Централізована	Колабора- тивна	З'єднані Інtranети	Розподілена
Моделі систем управління	Централізована	+	-	-	-
	Типу Master/Slave	+	-	-	-
	З координацією	-	-	-	+
	З розподілом інформації	-	-	+	-
	З региональним плануванням	-	+	-	-
	Ієрархічна	-	-	-	+

### 1.3 Технології забезпечення адаптивності розподілених систем обробки даних в умовах мобільності користувачів

Розподіленою системою називають систему, розміщення елементів якої суттєво впливає на її функціонування. Розподіленою є і система обробки даних медичних датчиків: покази останніх передаються на компонент обробки даних, а віддаленість між ними впливає часову затримку обробки, мережеве навантаження та безпеку переданих даних, адже вона здійснюється по публічним каналам зв'язку. Також, важливою властивістю такої системи є мобільність датчиків, яка спричиняє динамічні зміни в архітектурі розподіленої

системи. Тому, одним із аспектів її адаптивності можна вважати зміну конфігурації системи з урахуванням мобільності її користувачів. Вплив мобільності користувачів на ефективність роботи системи продемонстровано у [4], в умовах, коли користувачі переміщуються з двох крайніх областей до центральної, аналогічно типовому урбаністичному руху між житловими районами міста та його центром.

### **1.3.1 Туманні обчислення для забезпечення адаптивності**

Важливим аспектом ефективного функціонування інформаційної системи з обробки датчиків є відповідність процесу виконання завдань їхнім вимогам. Так, заслужено популярна парадигма хмарних обчислень ефективно застосовується обчислювано містких задач, як-от аналіз великих даних та машинне навчання. Натомість, в умовах нестабільного підключення до мережі, необхідності обробки даних у реальному часі, потреби передачі значних обсягів даних або підвищених вимог безпеки, хмарні обчислення можуть бути слабким місцем інформаційної системи [14].

У випадку наведених вимог, доцільним є залучення для обробки даних обчислювальних потужностей, розташованих близче до кінцевих користувачів, в межах концепції краївих обчислень. У багатьох працях терміни країові та туманні обчислення вживаються як синоніми, хоча існують праці, в яких ці поняття мають відмінності. Так, у роботі [15] виділяють концепцію туманних обчислень як одну із реалізацій парадигми краївих обчислень водночас із концепціями мобільних краївих обчислень та обчисленнями у мікроцентратах з обробки даних (хмарках). У таблиці 1.2 автори порівнюють ці реалізації з точки зору типу залучених обчислювальних пристройів (вузлів), їхнього місце розташування в системі, архітектури програмного забезпечення, ступені обізнаності контексту, відстань до кінцевого користувача, технологій доступу та характеру комунікації між вузлами. Однак, у цій роботі все ж поняття краївих та туманних обчислень вживаються як взаємозамінні, у значенні парадигми в цілому, аніж якоїсь конкретної її реалізації.

Таблиця 1.2 – Порівняння реалізацій парадигми країових обчислень [15]

	<b>Туманні обчислення</b>	<b>Мобільні країові обчислення</b>	<b>Обчислення у мікро-ЦОД</b>
Обчислювальні вузли	Маршрутизатори, комутатори, точки доступу, шлюзи	Сервери на базових станціях	Мікроцентри обробки даних
Місце розташування пристройв	Будь-де між кінцевими користувачами та хмарним ЦОД	Біля контролерів радіомережі, на базові станції макрорівня	В будь-якому приміщенні або ззовні
Архітектура ПЗ	Абстракція туманного шару	Мобільне оркестрування	Агентна для мікро-ЦОД
Обізнаність контекстом	Середня	Висока	Низька
Відстань	Один або декілька мережевих «кроків»	Один мережевий «крок»	Один мережевий «крок»
Технологія доступу	Bluetooth, Wi-Fi, мобільні мережі	Мобільні мережі	Wi-Fi
Взаємодія між вузлами	Повна підтримка	Часткова підтримка	Часткова підтримка

Отже, проявом адаптивності інформаційної системи, побудованої в парадигмі туманних обчислень, в умовах мобільності користувачів може бути постійний моніторинг поточного стану користувачів системи із динамічним залученням і вивільненням апаратних ресурсів, відповідно до поточних потреб обробки даних.

### 1.3.2 Міграція модулів додатку

Застосування країових обчислень полягає в обробці даних якомога близче до кінцевого користувача, що забезпечується запуском програмних

компонент на відповідних обчислювальних вузлах. Принциповим є те, що в загальному випадку програмна компонента містить набір контекстів, станів, кожного користувача, який звертається до неї. По мірі переміщення користувача із одного місця в інше, розміщення програмних компонент перестає бути оптимальним та вносить небажані затримки обміну даними, що є суттєвим для чутливих систем реального часу. Тому виникає необхідність у процесі міграції – перенесення запущеної програмної компоненти із відповідним контекстом з одного обчислювального вузла на інший, згідно з переміщенням мобільного користувача.

Таким чином, процес міграції слід розглядати з двох боків: по-перше, як безпосереднє переміщення програмного середовища (віртуальної машини або контейнера) з одної апаратного ресурсу на інший; по-друге, як переміщення контексту користувача з одного програмного середовища в інше.

Існують дві техніки міграції програмного середовища [3]:

- Холодна міграція – віртуальна машина або контейнер повністю припиняє роботу, відбувається копія стану її/його регістрів процесора та всієї пам'яті з одного обчислювального вузла на інший, після чого вона/він відновлює роботу на новому пристрої. Для цього варіанту характерний значний час, коли програмне середовище є недоступним користувачеві;
- Жива міграція – відбувається копія лише регістрів процесора та мінімально необхідної пам'яті, після чого програмне середовище продовжує роботу на новому пристрої. Подальше перенесення стану віртуальної машини або контейнера відбувається уже в процесі її/його роботи. Така техніка міграції значно зменшує час недоступності програмного середовища, але потребує механізмів забезпечення синхронізації та може спричинити перенесення більшої кількості даних, аніж попередній.

Схожі техніки можна застосувати і для міграції контексту користувача, блокуючи його на початковому вузлі та здійснюючи повне або часткове його перенесення на цільовий, хоча це вимагає закладання концепції контексту ще на етапі створення програмного продукту.

Таким чином, при розробці реальної платформи, процес міграції для забезпечення адаптивності обов'язково слід враховувати у двох його аспектах, але при моделюванні такої платформи можна розглянути лише аспект переміщення контексту, абстрагувавшись від програмного середовища. Тоді кожному користувачу відповідає його унікальне програмне середовище, тотожне його контексту, міграція якого і стане предметом дослідження.

### **1.3.3 Організація інтелектуальної міграції**

Із використанням концепції туманних обчислень виникає задача про оптимальне розташування модулів додатку, що відрізняються своїми вимогами до обчислювальних, мережевих та інших ресурсів, на гетерогенних обчислювальних вузлах системи. Так, у [16] розробники запропонували «краєорієнтовану» стратегію, призначену для ієархічних систем із архітектурою у вигляді дерева. Коренем такої системи є хмарний ЦОД, до якого під'єднані вузли туманних обчислень першого рівня. Кожен вузол може містити довільну кількість дочірніх вузлів наступного рівня. Вузлами останнього рівня можуть бути персональні пристрої користувачів, а листами дерева є датчики. Краєорієнтована стратегія намагається розташувати модулі додатку, що обробляють дані з датчиків, якомога ближче до кінцевих користувачів, та лише за неможливості такого розміщення переходить навищий рівень ієархії. Альтернативно, у [17] було формалізовано задачу розміщення сервісів на вузлах туманних обчислень та продемонстровано її рішення методом лінійного програмування. Такий підхід дозволяє внести поняття якості обслуговування, наклавши на знайдені рішення умови задоволення зазначеним вимогам. Нарешті, у [5] було здійснене порівняння ефективності наведених вище

стратегій із запропонованим підходом багатоцільового розміщення додатків у туманному середовищі (МАРО), який не поступився альтернативним методам.

В основі МАРО покладено генетичний алгоритм NSGA-II, особливістю якого є можливість оптимізації множини цільових функцій на відміну від запропонованих рішень, пристосованих для єдиної або зваженій сумі декількох цільових функцій. Для задачі розміщення додатків на вузлах туманних обчислень в якості цільових функцій у [5] вибрано часову затримку обробки даних, споживання електроенергії туманними вузлами та загальну вартість обчислень. Для заданих функцій змінними є логічні  $x_{ij}$ , які визначають присутність модуля додатку  $j$  на вузлі  $i$ . За допомогою стандартних операцій мутації та комбінування генетичний алгоритм NSGA-II підтримує і поступово покращує популяцію рішень, наближаючись до фронту оптимальних за Парето (рисунок 1.9). Серед параметрів для алгоритму NSGA-II слід виділити розмір популяції, який визначає кількість рішень, які залишаються після елітарного відбору, та кількість обчислень набору цільових функцій, після якої алгоритм завершує свою роботу. Остаточне рішення щодо розташування модулів додатку приймається за допомогою методів без визначення відношення переваги або апріорних методів [18].

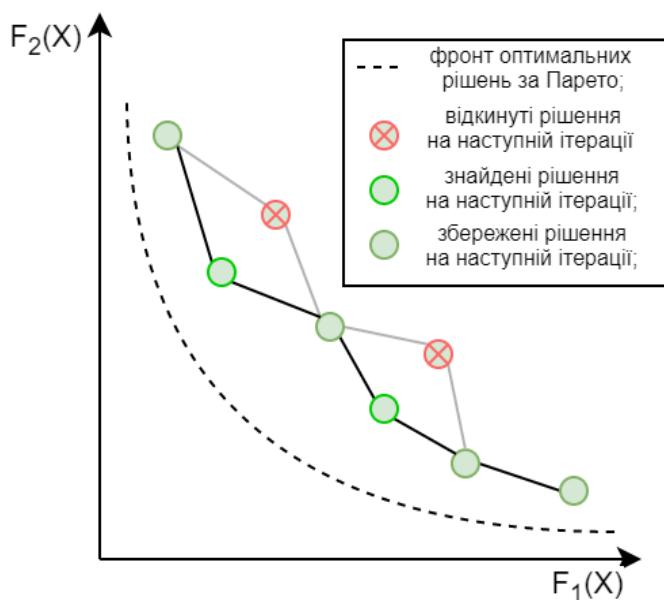


Рисунок 1.9 – Процес пошуку оптимальних рішень алгоритмом NSGA-II

Задачу організації інтелектуальної міграції програмних компонент в умовах мобільності користувачів можна сформулювати як необхідність багаторазового вирішення описаної вище задачі розподілу модулів додатку між обчислювальними вузлами. Процес оптимізації може відбуватися регулярно через визначені проміжки часу, а може ініціюватися лише при фактичних змінах в архітектурі системи в процесі переміщення кінцевого користувача. В останньому випадку міграція має реактивний характер. Також, з'являються широкі можливості для аналізу поведінки мобільного користувача з метою передбачення його положення в наступний момент часу для застосування «проактивної» міграції [2].

У даній роботі підлягає перевірці ефективність підходу МАРО для вирішення задачі інтелектуальної міграції. Для поліпшення роботи алгоритму з точки зору компромісу між якістю знайдених рішень та швидкістю обчислень слід дослідити наступні модифікації:

1. Інжекція ідеалізованих рішень в початкову популяцію: ідеалізованими рішеннями є такі рішення, які б за ідеальних умов при необмежених ресурсах, були б присутні у фронті Парето для визначених цільових функцій. Наприклад, для мінімізації загальної вартості системи ідеалізованим рішенням є розміщення всіх модулів на пристрої з найменшою вартістю обчислень. У випадку мінімізації навантаження на мережу за ідеальних умов всі модулі розміщаються на найближчих до кінцевих користувачів пристроях. В загальному випадку початкова популяція алгоритму NSGA-II є довільною, отже може ініціалізуватися випадково. Однак, можна висунути припущення, що існує можливість покращити якість отриманого результату, якщо частину популяції заповнити ідеалізованими рішеннями;
2. Зменшення розмірності рішення: у випадку, коли кількість ітерацій прямує до нескінченості, залучення всіх змінних системи, які для даної задачі визначають розміщення кожного модуля додатку на окремому вузлі, має дати краще рішення, аніж у випадку, коли

множина змінних системи розбивається на підмножини, для яких оптимізація відбувається окремо. Але можна висунути припущення, що в умовах обмеженого часу на прийняття рішень та динамічності системи, рішення, знайдене для всіх змінних може виявитися гіршим, аніж сума рішень, знайдених для окремих частин змінних системи. У даній роботі пропонується розбити множину всіх модулів додатків на так звані «кластери», та проводити оптимізацію окремо для кожного з них, таким чином перевіривши ефективність такої модифікації процесу рішення. Крім того, при використанні такого підходу існує перспектива раціонального та автоматизованого розподілу множини змінних на частини, що здатна привести до подальшого покращення результату;

3. Впровадження реактивного характеру організації міграції: скориставшись принципом реактивної міграції, оптимізацію розподілу модулів додатку можна проводити лише для тих користувачів, які спричинили зміну у конфігурації мережі. Застосування даного підходу значно зменшує область значень рішення, а тому пришвидшує роботу генетичного алгоритму. Однак, ця модифікація не буде призводити до втрати якості знайденого рішення лише за умови, що розташування інших модулів не втратить свою оптимальність при застосуванні остаточного рішення. У реальному ж випадку можна очікувати як певне погіршення якості ситуації у порівнянні з оригінальним алгоритмом, так і в середньому покращення ситуації, оскільки оптимізація невеликої частини загального рішення в умовах обмеженого часу на прийняття рішення може виявитися ефективнішою за тривалу оптимізацію самого рішення загалом.

#### **1.3.4 Мікросервісна архітектура інформаційних систем**

В умовах необхідності постійної міграції програмного середовища між обчислювальними вузлами системи, бажано забезпечити гнучкість системи

таким чином, щоб компоненти додатку могли легко переміщуватися між вузлами, швидко запускатися та припиняти свою роботу. Такі вимоги до ефективного застосування програмного забезпечення дозволяє задоволити парадигма мікросервісів. Окрім переваг на етапах розробки та вдосконалення, для мікросервісної архітектури у контексті безпосередньої роботи програмного продукту характерні властивості гнучкості, слабкої зв'язності та масштабованості [19].

Мікросервіс як самостійний програмний компонент має у своїй області відповідальності єдиний і цілісний аспект бізнес-процесу та може бути запущений швидко та незалежно від інших компонентів. Ця властивість збільшує гнучкість процесу міграції, дозволяючи вирішувати задачу оптимізації розташування окремого мікросервіса, не переймаючись про можливі приховані залежності з іншими компонентами. Однак, у контексті міграції більш гостро проявляється конфлікт між необхідним і достатнім рівнем гранулярності. З одного боку, збільшення загальної кількості мікросервісів дає можливість оперувати меншими за об'ємом модулями, підвищуючи гнучкість міграції. З іншого ж, завжди вказується, що збільшення гранулярності часто призводить до появи додаткових комунікацій між мікросервісами, що призводить до погіршення гнучкості системи, ускладнюючи оптимізацію розташування модулів.

Додаткову синергію між мікросервісною архітектурою та адаптивною міграцією забезпечує закладена властивість масштабованості, завдяки якій природно розміщувати різні екземпляри кожного мікросервіса, пропозиціючи від потенційно слабких місць. Ця задача може бути врахована у процесі більш глобальної задачі з оптимізації розташування модулів додатку.

Можна припустити, що в поєднанні з організацією міграції вкрай ефективно створювати свій додаток у парадигмі безсерверних обчислень. У цьому випадку розробник програмного продукту зможе повністю абстрагуватися від програмного середовища, чи то віртуальної машини, чи то контейнера, сконцентрувавшись лише на визначені контексту користувача. А

механізм інтелектуальної міграції зможе повністю перейняти контроль над середовищем виконання. Безсерверні обчислення підносяться як наступна ступінь розвитку після мікросервісної архітектури [20] та заслуговує бути врахованою в контексті адаптивної платформи туманних обчислень для мобільних користувачів.

### **1.3.5 Архітектура адаптивної платформи**

Враховуючи описаний стек технологій, можна визначити архітектуру платформи (рисунок 1.10), яка б забезпечувала властивість адаптивності в умовах мобільності її кінцевих користувачів. На нижньому рівні знаходиться інфраструктура, що складається з вузлів туманних обчислень та, можливо, хмарних центрів обробки даних. Із ним за допомогою бездротових технологій, більшою мірою 3G, 4G/LTE, 5G та меншою мірою Wi-Fi, взаємодіють різноманітні клієнтські пристрой та датчики: мобільні телефони, персональні пристрой моніторингу життєдіяльності, датчики для вимірювання конкретних параметрів тощо. На рівні проміжного програмного забезпечення реалізовані механізми повної віртуалізації або контейнеризації, адаптовані механізми міграції віртуальних середовищ та задіяні інтелектуальні методи організації процесу міграції, чи то на основі генетичного алгоритму, чи то з використанням іншого методу штучного інтелекту та машинного навчання. Прикладний рівень визначається модулями бізнес-додатків та вимогами до якості обслуговування користувачів. Про адекватність побудованої архітектури платформи свідчить її відповідність із представленаю у сценарії [5].

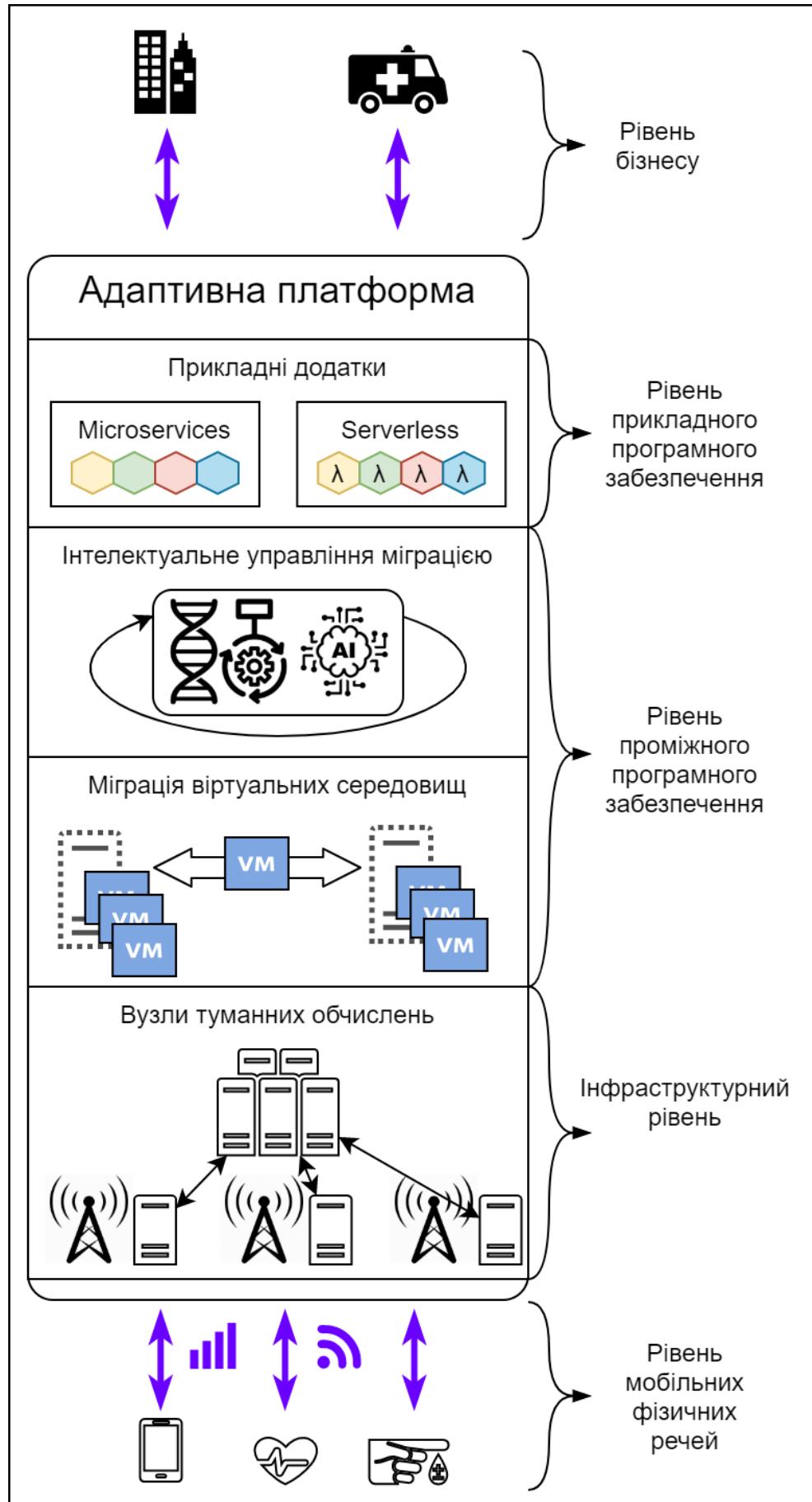


Рисунок 1.10 – Архітектура адаптивної туманної платформи

## 1.4 Висновки до розділу 1

В результаті, можна стверджувати, що типовим механізмом забезпечення адаптивної поведінки системи є циклічне вимірювання зовнішніх і внутрішніх характеристик, їх аналіз, прийняття рішення про необхідність виконання подальших дій та безпосереднє втілення останніх.

Доцільно оцінити повноту знань про вимоги до поведінки системи та зовнішнє середовище та обґрунтувати, які аспекти функціонування системи можна визначити на етапі проектування, а які мають проявлятися безпосередньо під час виконання. Це дозволить заздалегідь класифікувати цільову систему, визначити її властивості та необхідні компоненти.

Також, у процесі проектування як цільової системи із її функціональними та нефункціональними вимогами, так і системи забезпечення адаптивності та нефункціональних вимог, необхідно перевірити на узгодженість характер розподіленості обох систем.

Було визначено архітектуру адаптивної платформи Інтернету речей в умовах їхньої мобільності та обґрунтовано стек залучених технологій, що складається з парадигми туманних обчислень, механізмів міграції віртуального середовища, організації інтелектуального керування процесом міграції та орієнтованості на мікросервісну та без серверну архітектури прикладних додатків.

В якості основи інтелектуального компоненту обрано генетичний алгоритм та описано способи його модифікування з метою пристосувати його до вирішення задачі міграції.

## 2 РОЗРОБКА МОДЕЛЕЙ СИМУЛЯЦІЇ

Для дослідження моделі адаптивної платформи розроблено власне середовище проведення симуляцій на базі інструментарію iFogSim для моделювання та симуляції додатків в галузях Інтернету речей, туманних та крайових обчислень [16]. iFogSim дозволяє моделювати роботу додатків, представлених у формі орієнтованого графа в парадигмі розподілених потоків даних. Вузлами графа є датчики, що у визначені моменти часу ініціюють передачу даних, приводи (актуатори, діячі) – кінцеві пристрої в парадигмі Інтернету речей, відповідають тупиковим вершинам графа та закінчують цикл обробки інформації, та модулі – вершини, які обробляють дані, що надійшли на їхній вхід, та генерують відповідні вихідні дані. Також, iFogSim надає можливість моделювати мережі із вузлів туманних обчислень, оперуючи параметрами пропускної спроможності та затримки каналу, однак із обмеженням на строгу ієрархічну архітектуру мережі. Серед параметрів апаратного забезпечення можна задати потужність (у мільйонах інструкцій за секунду), енергоспоживання та вартість використання (в умовних одиницях за кожний мільйон виконаних інструкцій). Таким чином вихідними даними симуляції можуть бути сумарні значення спожитої енергії та вартості для кожного пристрою, навантаження на мережу (в одиницях об'єму даних, наприклад бітах, за одиницю часу), та середній час затримки між необхідними ділянками у графі додатку.

Розроблений симулятор на базі iFogSim розширює функціонал останнього у наступних аспектах. По-перше, він дозволяє моделювати переміщення вузлів, вносячи поняття двовимірної координатної площини, із визначенням їхньої поточної координати, вектору швидкості через його модуль та кут нахилу. По-друге, на координатній сітці можна виставити статичні бездротові точки доступу (базові станції) із визначеною областю покриття, до яких динамічно під'єднуються та від'єднуються мобільні пристрої на своєму шляху. По-третє, у симуляторі знято обмеження на ієрархічність побудованих

мереж та закладено можливість однорангової комунікації, що збільшує гнучкість його використання у різноманітних сценаріях. Нарешті, реалізовано мінімально необхідний функціонал міграції модулів з одного пристрою на інший та механізм управління цим процесом із можливістю подальшого розширення. Перераховані функції складають достатній інструментарій для проведення подальшого дослідження міграції у моделі адаптивної туманної платформи.

## **2.1 Вхідні дані та умови функціонування платформи**

### **2.1.1 Модель медичного додатку**

В якості програмного продукту в галузі медичного Інтернету речей взято спрощений прототип системи моніторингу стану населення на предмет серцево-судинних захворювань [21]. Наведена схема моніторингу пацієнта складається із семи параметрів із відповідними вимогами до їхньої частоти (таблиця 2.1). Виміри, в яких частоту або період взяття показів не визначено, у праці [21] стверджується, що їхнє взяття відбувається лише на вимогу, а не періодично.

**Таблиця 2.1 – Схема моніторингу серцево-судинних захворювань [21]**

<b>Вимірюваний параметр</b>	<b>Частота (період) вимірювання</b>
Електрокардіограма	128 Гц
Кров'яний тиск	2 секунди
Насиченість крові киснем	2 секунди
Частота серцевих скорочень	2 секунди
Рівень холестерину в крові	не визначено
Рівень глюкози в крові	не визначено
Локація пацієнта	не визначено

Хоча, враховуючи потенціал створення мініатюрних датчиків, які можуть вимірювати параметри холестерину та глюкози крові у будь-який момент часу, було прийняте рішення про доцільність їх періодичного вимірювання та обробки, як і користь постійної обізнаності системи про поточне положення пацієнта. Однак, для побудованої моделі додатку до періодичного моніторингу додано лише параметр локації, адже можливість її визначення вже не складає жодних труднощів на сучасному рівні технологічного розвитку.

Орієнтуючись на описаний прототип додатку, було створену його модель (рисунок 2.1) для використання у дослідженні. Множина датчиків для вимірювання окремих параметрів була замінена узагальненим датчиком пацієнта, який з періодом у дві секунди надсилає дані електрокардіограми, кров'яних тиску та насыщеності киснем, частоти серцевих скорочень та GPS-координати користувача. Далі покази датчика потрапляють на модуль, розміщений на смартфоні пацієнта, де відбувається їхнє впорядкування та підготовка до надсилання на подальшу обробку. Через бездротове з'єднання дані потрапляють зі смартфону до адаптивної платформи, де в першу чергу аналізуються відповідним модулем. Очікується виконання первинної фільтрації, порівняння даних з еталонними значеннями, збереження даних для подальшого використання та, за необхідності, здійснюється запит додаткової інформації до модуля, на якому зберігається електронна медична картка пацієнта. На наступному кроці узагальнена інформація надсилається до модулю визначення відхилень, від якого очікується більша ступінь інтелектуальності: за допомогою моделей штучного інтелекту, як-от класичних експертних систем або новітніх підходів на основі нейронних мереж, здійснюється більш детальний аналіз параметрів пацієнта. Врешті решт, висновки про його поточний стан надсилаються кінцевому споживачу. В даному випадку сам пацієнт отримує інформацію про свій стан у режимі реального часу, але цілком можливо надсилати її персональному лікарю або негайно повідомляти службу швидкої допомоги в екстрених випадках.

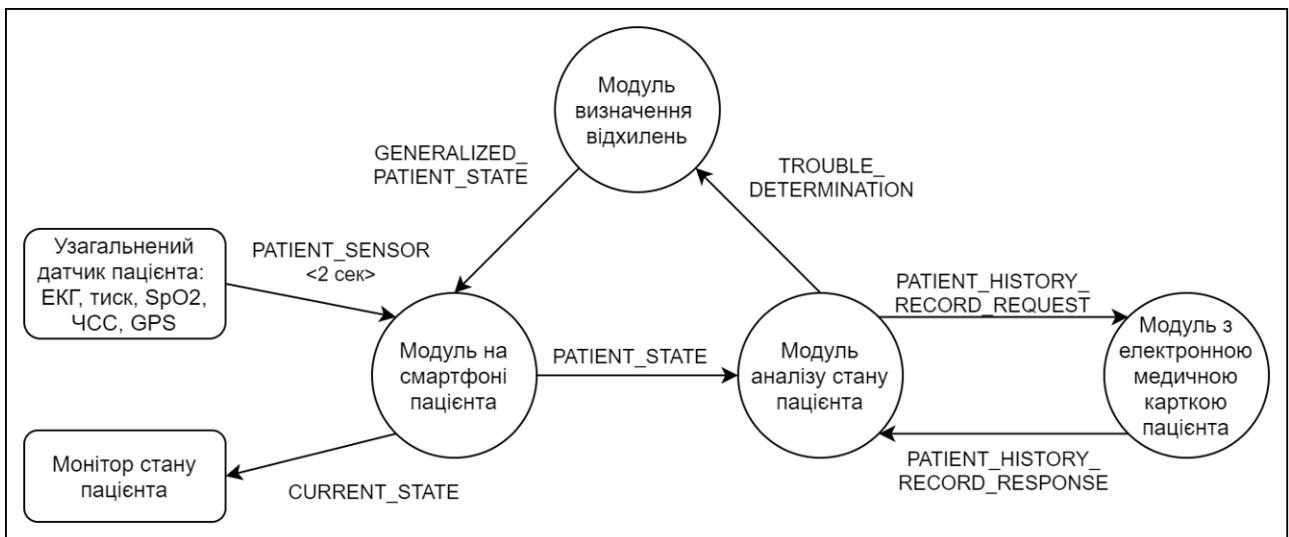


Рисунок 2.1 – Використана у дослідженні модель медичного додатку

Основними параметрами моделі додатку в симуляторі iFogSim є кількість інструкцій процесора  $INST_i$  (у мільйонах інструкцій) для відображення обчислювальної складності, об'єм даних  $NW_i$  (у бітах) для врахування мережової завантаженості, ймовірність утворення нової задачі після обробки й отримання результату попередньої  $p_{ij}$ , а також періодичність надсилання задач даного типу  $T_i$ , якщо ініціювання задачі відбувається датчиком або мимовільно визначенім модулем. Вважається, що кожне ребро графа визначає окремий тип задач, для яких зазначені параметри є спільними. Нижче проведено розрахунок необхідних параметрів для кожного ребра графа. Для даної моделі додатку параметр  $T_i$  має значення 2 секунди для задач, які надходять від узагальненого датчика, та відсутній для всіх інших, тому він не визначається у подальших розрахунках. У свою чергу, параметр ймовірності  $p_{ij}$  дорівнює одинці в усіх випадках, які співпадають з описаним циклом додатку, окрім ребра «PATIENT\_HISTORY\_RECORD\_REQUEST», який може відбуватись після обробки «PATIENT\_STATE» лише в 1% випадків.

Цикл додатку починається із відправки узагальненим датчиком даних по ребру «PATIENT\_SENSOR». Враховуючи необхідну частоту та тип параметра з таблиці 2.1, необхідно розрахувати сумарний об'єм даних який надходить з кожного вимірюваного параметру. Нехай для показів локації користувача

використовуються п'ять параметрів: три координати, швидкість та кут повороту. Таким чином, єдина передача даних від узагальненого датчика  $NW_1$  загалом складає 16,896 біт. Обчислювальну складність  $INST_1$  же прийнято за 500 млн. інструкцій, що у сукупності з визначеною пізніше обчислювальною потужністю смартфона призведе до приблизного часу обробки у 100 мс.

$$nw_{ECG} = 128 \frac{1}{c} \cdot 64 \text{ біт} \cdot 2 \text{ с} = 16,384 \text{ біт}, \quad (2.1)$$

$$nw_{SpO2} = nw_{ЧСС} = nw_{тиск} = 0.5 \frac{1}{c} \cdot 64 \text{ біт} \cdot 2 \text{ с} = 64 \text{ біт}, \quad (2.2)$$

$$nw_{GPS} = 0.5 \frac{1}{c} \cdot 5 \cdot 64 \text{ біт} \cdot 2 \text{ с} = 320 \text{ біт}, \quad (2.3)$$

$$NW_1 = nw_{ECG} + nw_{SpO2} + nw_{ЧСС} + nw_{тиск} + nw_{GPS} = 16,896 \text{ біт}. \quad (2.4)$$

Параметр об'єму даних  $NW_2$  для задач типу «PATIENT\_STATE» обчислено з розрахунку на те, що отримані значення датчиків надсилаються на подальшу обробку у форматі JSON, отже займають приблизно в 1,5 рази більше, аніж сирі дані. Врахувавши приблизно 135 байт на символльні назви полів структури файлу, отримано:

$$NW_2 = 1.5 \cdot NW_1 + 135 \text{ байт} \cdot 8 \frac{\text{біт}}{\text{байт}} = 26,424 \text{ біт}. \quad (2.5)$$

Інші параметрам було присвоєно приблизні значення із врахуванням значень [16, 5]. Таблиця 2.2 містить остаточні параметри моделі додатку для моніторингу серцево-судинних захворювань на базі медичних датчиків.

Таблиця 2.2 – Параметри моделі додатку

Назва ребра графа	Об'єм даних $NW_i$ , біт	Складність $INST_i$ , млн. інстр.
PATIENT_SENSOR	16,896	500
PATIENT_STATE	26,424	2000
PATIENT_HISTORY_ RECORD_REQUEST	8,192	5000
PATIENT_HISTORY_ RECORD_RESPONSE	819,200	1000
TROUBLE_DETERMINATION	108,344	1000
GENERALIZED_PATIENT_STATE	800	250
CURRENT_STATE	80	0

### 2.1.2 Мобільність користувачів платформи

У праці [3] з метою врахування в процесі моделювання реальних шаблонів переміщення користувачів було використано результати моделювання міського трафіка Люксембурга [22]. Для вирішення задачі моделювання трафіка використовувався інструментарій “Simulation of Urban Mobility” (SUMO) [23]. Однак, автори на сторінці із сценарієм моделювання для Люксембурга у GitHub [24] зазначають, що, хоча валідність даних була підтверджена для ранніх версій SUMO, вона не може гарантуватись для нових версій інструменту, та заохочують використовувати сумісний з сучасними версіями сценарій для моделювання трафіка Монако [25].

Враховуючи це, для отримання даних про мобільність користувачів в умовах міста було проведено моделювання сценарію для трафіка Монако за допомогою симулятора SUMO (рисунок 2.2). В основному, SUMO призначений для моделювання транспортного потоку, з метою визначення, наприклад, проблемних маршрутів, узгодження світлофорів, оптимізації маршрутів громадського транспорту та інших муніципальних задач. Наприклад, для

обраного сценарію Монако в моделюванні беруть участь як особисті автомобілі, так і велосипеди, мопеди, регулярні автобусні рейси за різними маршрутами, спеціальний транспорт як автомобілі швидкої допомоги, комерційні перевезення вантажівками, потяги до Франції та Італії, а також пішоходи. Хоч у даному дослідженні використано дані лише невеликої частки всіх пішоходів та особистого транспорту, існує можливість врахувати більшу кількість інформації, наприклад враховуючи переміщення цілих груп людей у громадському транспорті.



Рисунок 2.2 – Моделювання міського трафіка за допомогою SUMO

### **2.1.3 Розміщення базових станцій оператора стільникового зв'язку**

У попередніх пунктах відповідно до архітектури адаптивної платформи було визначено рівень мобільних фізичних речей. Наступним етапом є

визначення інтерфейсу їхньої взаємодії з платформою, що відбувається за допомогою технологій мобільного зв'язку. Для його забезпечення на карту Монако було накладено сітку базових станцій задля майже повного покриття території мобільним зв'язком. З метою дотримання адекватності моделі за допомогою відповідного інструментарію [26, 27] було знайдено зареєстровані базові станції оператора мобільного зв'язку «Orange Mobile» та оцінено їхню приблизну щільність розташування на території країни. Так, на рисунку 2.3 зображене розташування базових станцій на міській території, а на рисунку 2.4 – на приміській. Як видно із рисунку 2.5, територія країни повністю покрита зв'язком 4G, однак у даному дослідженні задля усереднення вважатиметься, що на приміській території знаходяться лише базові станції технології 3G.



Рисунок 2.3 – Розташування базових станцій Монако на території міста

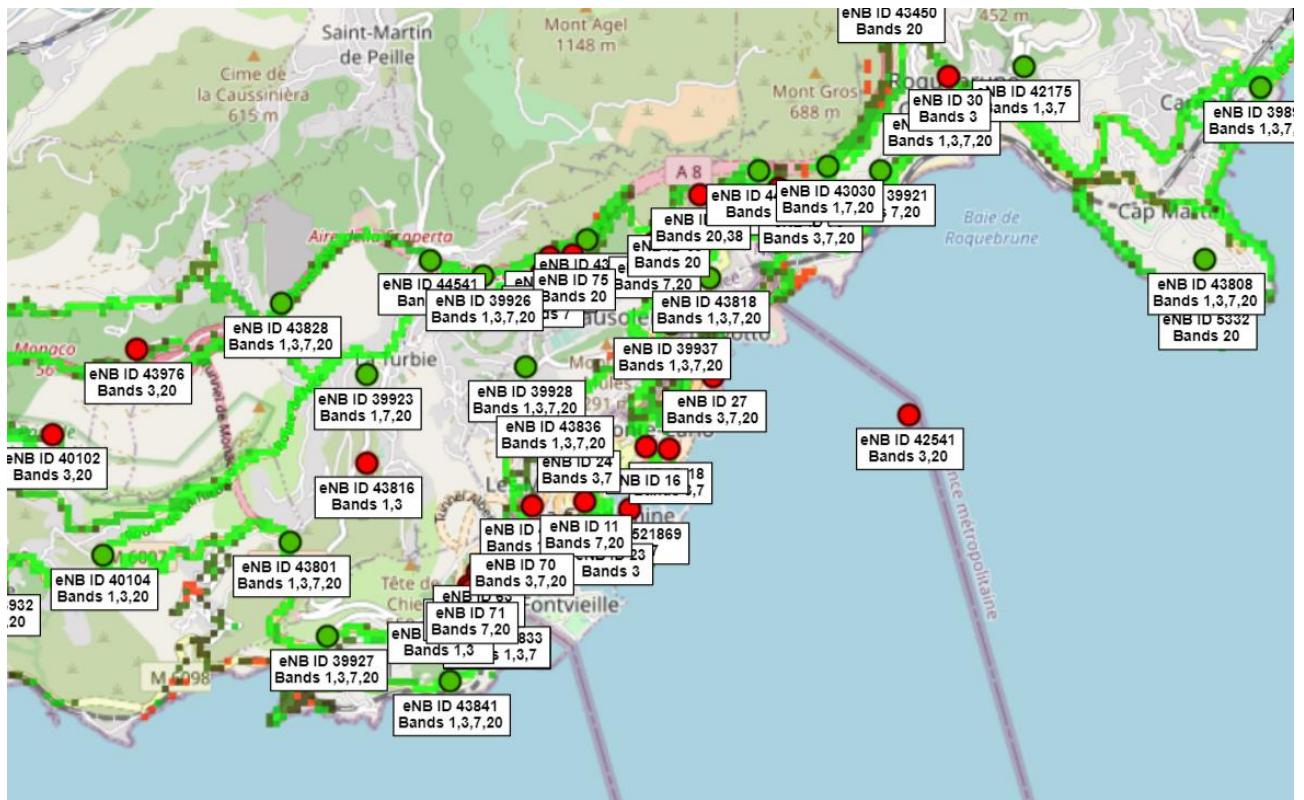


Рисунок 2.4 – Розташування базових станцій Монако на приміській території

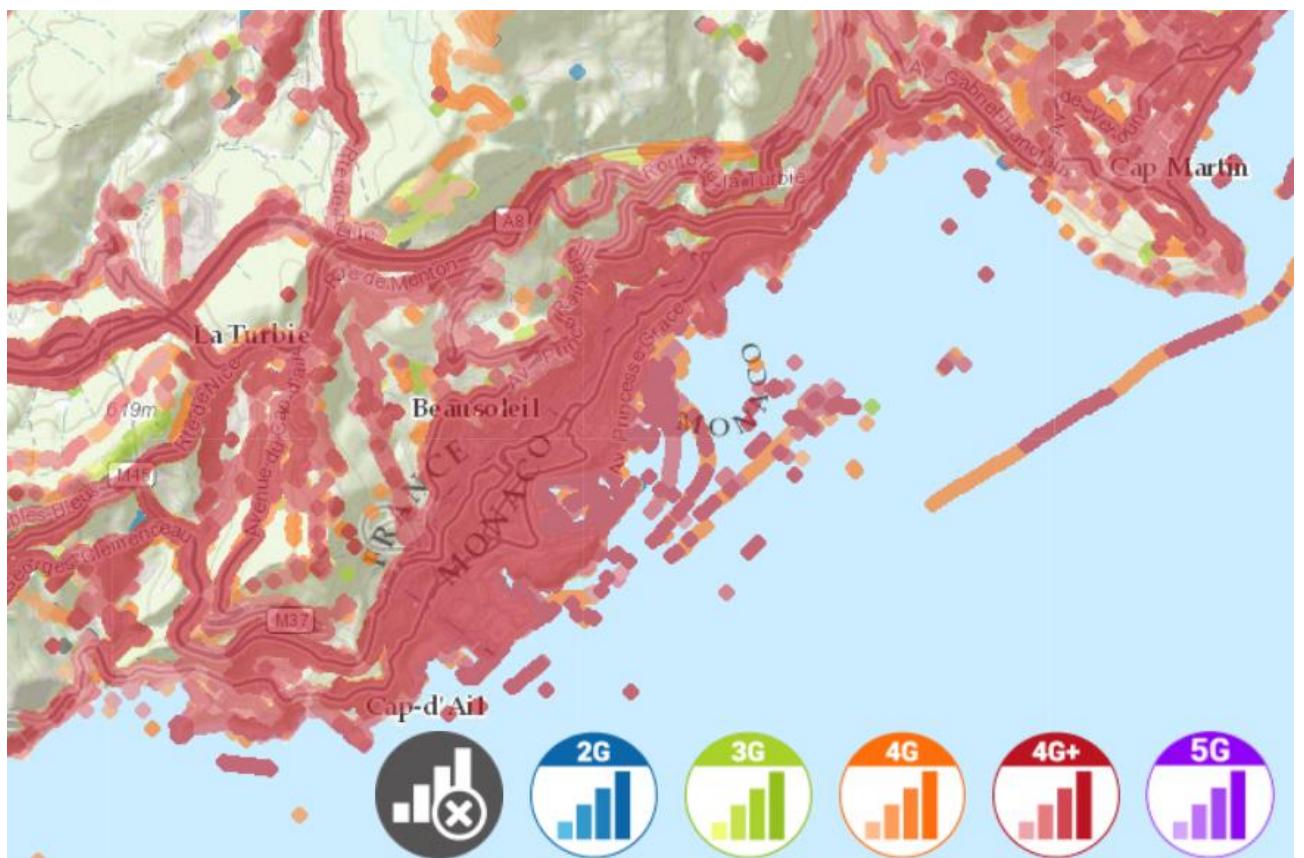


Рисунок 2.5 – Покриття території Монако з технологією 4G

Згідно з наведеною інформацією [28], розрізняють декілька типів базових станцій в залежності від радіусу їхньої зони покриття та варіанту використання (таблиця 2.3).

Таблиця 2.3 – Типи базових станцій [28]

<b>Тип базової станції</b>	<b>Радіус покриття</b>	<b>Призначення використання</b>
Фемтокомірка	10м	Дім, офіс
Домашній повторювач	100м	Дім, офіс, підприємство
Пікокомірка	200м	Висока будівля, готель, парковка
Мікрокомірка	1-2км	Торгівельні центри, транспортні вузли, околиці шахт, квартал міста, тимчасові масові заходи, заходи з усунення наслідків стихійних лих
Макрокомірка	5-32км	Міські, приміські, сільські території
Макрокомірка з розширенним радіусом	50-150км	Території передмістя та сіл

Таким чином, для базових станцій на міській території можна обрати радіус покриття 1-2км, а для базових станцій передмістя – 5км. Однак, такі параметри дещо суперечили б реальному розташуванню базових станцій. Тому задля збереження приблизної відповідності щільності розміщення базових станцій було вирішено, що радіус покриття базових станцій на території міста складатиме 385м, а в передмісті – 1км.

Остаточне розміщення базових станцій у сценаріях моделювання зображене на рисунку 2.6, де у місті розміщено станції 4G, а на решті території – 3G. Таке розташування забезпечує майже повне покриття території відповідно до переміщення кінцевих користувачів.

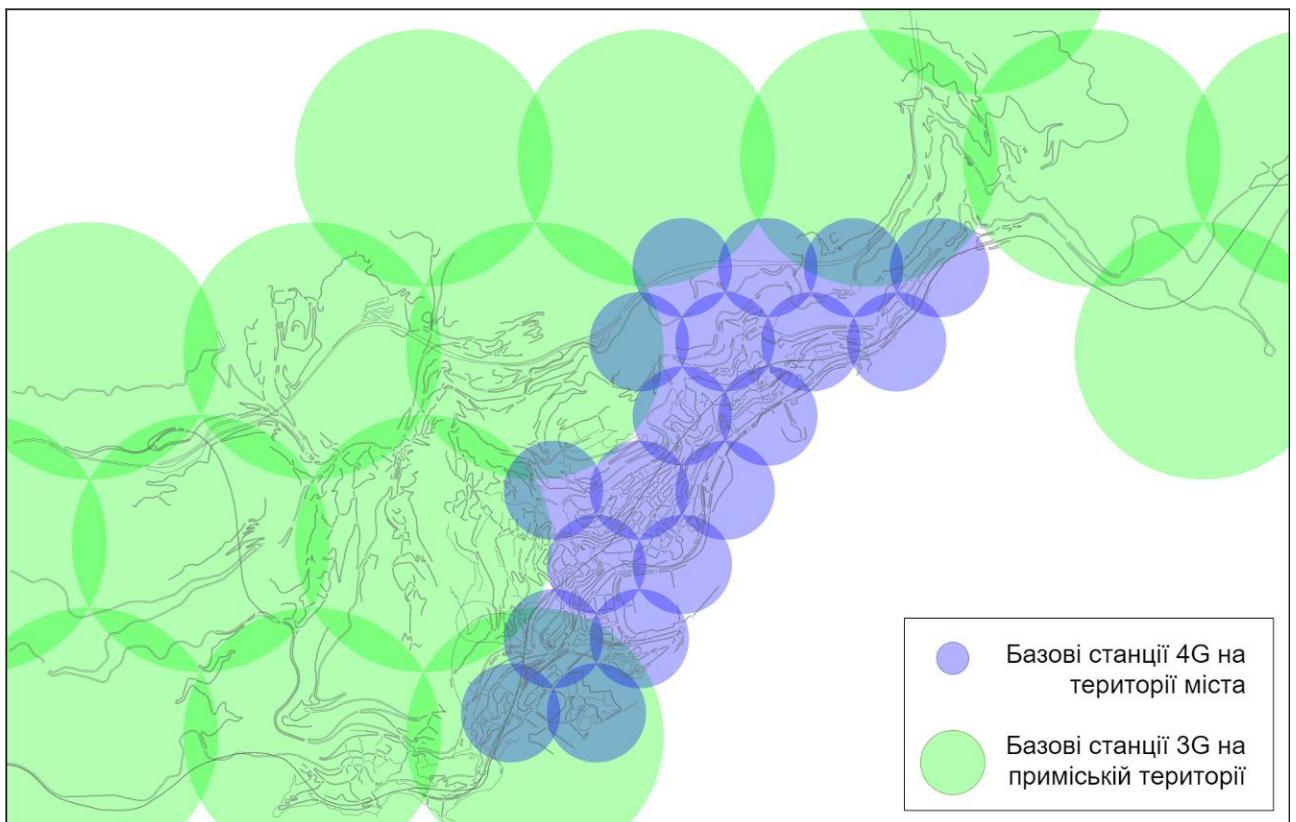


Рисунок 2.6 – Розташування базових станцій у сценаріях моделювання

#### 2.1.4 Визначення параметрів апаратного забезпечення платформи

Після опису моделі додатку, сценарію мобільності користувачів, розташування базових станцій бездротового зразку, наступним кроком у визначенні моделі адаптивної платформи є конкретизація її архітектури. Як вже було зазначено раніше, відповідно до стандартної концепції Інтернету речей, пристроями найнижчого рівня є датчики та актуатори. Інформація з перших потрапляє на смартфон користувача, обробляється та потрапляє в адаптивну платформу через технологію бездротового зв’язку. Базові станції, що її реалізують, згідно з концепцією туманних обчислень, окрім мережової функції містять апаратні ресурси, здатні проводити обробку інформації близче до кінцевих користувачів. Базові станції оператора зв’язку надсилають дані на шлюз Інтернет провайдера, який в загальному випадку також може містити обчислювальні ресурси у вигляді приватних центрів обробки даних довільного масштабу. Але в цій моделі адаптивної платформи даних ресурсів задіяно не

було, а шлюз Інтернет провайдера виконує лише функцію комунікації між всіма іншими вузлами. Можливим є розташування на території місцевої лікарні власного серверу із певною обчислювальною потужністю, яка за необхідності може бути використана у додатку. На найвищому рівні розміщено хмарний центр роботи даний, а між ним та шлюзом Інтернет провайдеру існує зв'язок через глобальну Інтернет мережу. Зображення описаної архітектури знаходиться на рисунку 2.7 та відповідає схожим сценаріям у [3, 5, 16].

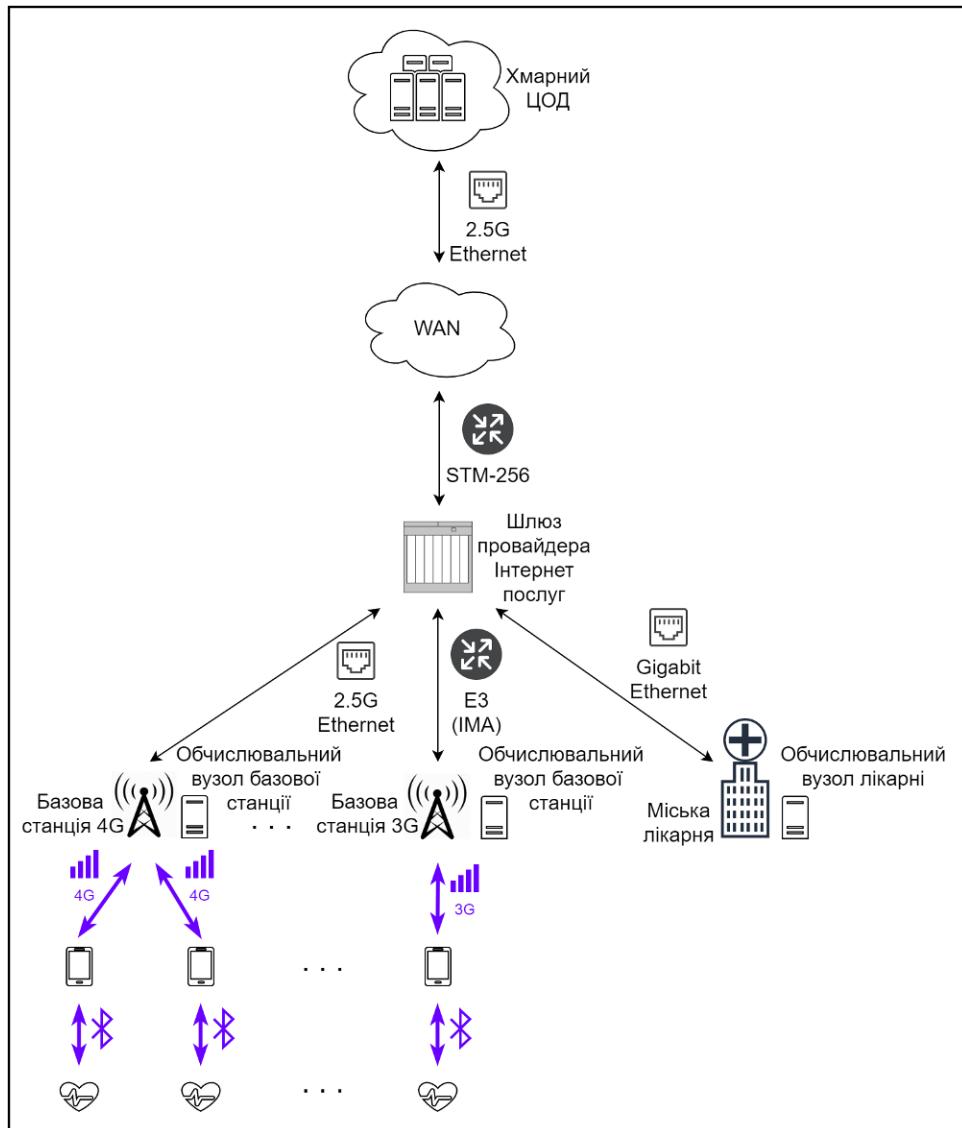


Рисунок 2.7 – Архітектура моделі адаптивної платформи

Слід розділити параметри апаратного забезпечення на дві категорії: параметри мережі та параметри обчислювальних вузлів. До параметрів першої

категорії належать пропускна здатність  $bw$  і затримка каналів зв'язку між пристроями платформи  $l$ . Друга категорія включає потужність обчислювальних вузлів  $MIPS$ , їхнє енергоспоживання  $W$ , кількість обчислювальних елементів  $CPU$  та вартість обчислень  $Pr$ .

#### 2.1.4.1 Мережеві параметри

Для визначення мережевих параметрів системи слід врахувати всі можливі комунікації між пристроями, з огляду на попередні пункти. Варто зазначити, що у сценаріях моделювання пропускна здатність задається у розмірності біт за секунду, а затримка – у мілісекундах.

Найнижчий рівень комунікації відповідає взаємодії між узагальненим датчиком, що вимірює параметри користувача, та смартфоном останнього. Нехай, для забезпечення цієї комунікації використано технологію з низьким енергоспоживанням Bluetooth LE. З огляду на дані [29] пропускну здатність було закладено у 6,000 байт/с, як певне усереднення між її значенням для пристрійв iOS та Android. Оскільки у симулаторі iFogSim саме для зв'язків від датчика до пристрою та від пристрою до діяча не закладений параметр пропускної здатності, а час передачі даних визначається лише часовою затримкою, було здійснене переведення цієї величини у затримку з врахуванням розміру стандартних пакетів даних для даного додатку та отримані значення 352 мс та 2 мс для датчика та актуатора відповідно.

$$l_1 = \frac{16,896 \text{ біт}}{6,000 \frac{\text{байт}}{\text{с}} \cdot 8 \frac{\text{біт}}{\text{байт}}} \cdot 10^3 \frac{\text{мс}}{\text{с}} = 352 \text{ мс}, \quad (2.6)$$

$$l_2 = \frac{80 \text{ біт}}{6,000 \frac{\text{байт}}{\text{с}} \cdot 8 \frac{\text{біт}}{\text{байт}}} \cdot 10^3 \frac{\text{мс}}{\text{с}} = 1.6 \text{ мс} \approx 2 \text{ мс}. \quad (2.7)$$

Наступним етапом в комунікації є передача даних зі смартфону до базової станції за допомогою бездротової стільникової технології зв'язку, чи то

третього, чи то четвертого покоління. Згідно з працею [30], реалістично прийняти пропускну спроможність 3G каналу в межах 0.5 та 5 Мбіт/с, а для 4G – від 1 до 50 Мбіт/с. Затримка в технології 3G складатиме від 100 до 500 мс, а в технології 4G – менше 100 мс. На ресурсі ж [31] наводяться величини затримки 212 мс (3G) та 98 мс (4G), що приблизно співвідноситься із зазначеними у [30], та уточнюється, що наведені дані визначені процедурою пінгування, яка, як відомо, враховує затримку в обох напрямках. Таким чином, було зафіковано середні значення пропускної спроможності 2.5 Мбіт/с для базових станцій 3G, та 25 Мбіт/с відповідно для базових станцій 4G. Стосовно значення затримки, було обрано, щоб затримка 200 мс у випадку 3G та 100 мс у випадку 4G, дорівнювала сумарній затримці від смартфона користувача до шлюзу Інтернет провайдера в прямий і зворотній бік разом. Тому значення затримки тільки на ділянці «смартфон – базова станція» прийнято 90 мс та 45 мс для базових станцій 3G та 4G, а затримку наступного зв'язку між базовою станцією та мережею Інтернет провайдера взято 10 мс та 5 мс відповідно.

Подальший зв'язок в розробленій моделі об'єднує базові станції зі шлюзом Інтернет провайдера. Згідно з інформацією, наведеною в [32], технології, якими базові станції під'єднано до шлюзу, є IMA для 3G мереж та Ethernet для 4G мереж. Також, у [33] вказано, що технологіями канального рівня для IMA можуть бути T1/E1 або T3/E3. Таким чином, для зв'язку між базовими станціями 3G зі шлюзом провайдера визначено пропускну спроможність 34.368 Мбіт/с [33]. Для випадку 4G цей параметр отримав значення 2.5 Гбіт/с згідно з відповідною технологією Ethernet. Однак, заявлені технологією значення пропускної здатності не гарантують передачу даних із відповідною швидкістю. На ресурсі [34] наводиться, що максимальна ефективність стандартного Gigabit Ethernet складає 94%, тобто максимальна швидкість передачі даних користувача дорівнює 940 Мбіт/с. В той же час, у [35] проведено дослідження, в одному з висновків якого стверджується, що забезпечення ефективності 70-80% для 10G Ethernet є складно досяжним та вимагає особливого обладнання. Тому, для усередненої поправки справжнього

значення пропускної здатності було прийнято коефіцієнт ефективності 85% для технології на цій та подальших ділянках мережі. В результаті, пропускна спроможність зв'язку між шлюзом провайдера Інтернет послуг та базовою станцією 3G склала 29.213 Мбіт/с, а для випадку 4G – 2.125 Гбіт/с.

$$bw_5 = 34.368 \cdot 10^6 \frac{\text{біт}}{\text{с}} \cdot 0.85 = 29.213 \cdot 10^6 \frac{\text{біт}}{\text{с}}, \quad (2.8)$$

$$bw_6 = 2.5 \cdot 10^9 \frac{\text{біт}}{\text{с}} \cdot 0.85 = 2.125 \cdot 10^9 \frac{\text{біт}}{\text{с}}. \quad (2.9)$$

За наявності в архітектурі системи локального сервера лікарні, його зв'язок із шлюзом Інтернет провайдера здійснюється за технологією Gigabit Ethernet з поправкою на ефективність 85% та часовою затримкою 5 мілісекунд.

$$bw_7 = 1000 \cdot 10^6 \frac{\text{біт}}{\text{с}} \cdot 0.85 = 850 \cdot 10^6 \frac{\text{біт}}{\text{с}}. \quad (2.10)$$

Останньою ланкою у мережі досліджуваної моделі є зв'язок між шлюзом провайдера та хмарним центром обробки даних через глобальну Інтернет мережу. У даному випадку зроблено припущення, що технологією для шлюзу Інтернет провайдера є високопродуктивний STM-256. Можливо, це не відповідає дійсності та типові міські провайдери Інтернету мають менш потужне з'єднання з глобальною мережею. Також, це суперечить схематичним зображенням із [32], де технологією канального рівня вихідного зв'язку шлюзу є Ethernet. Але все ж, на подальше дослідження ефективності генетичного алгоритму в адаптивній платформі ця неточність чинить вкрай опосередкований вплив. Врешті решт, пропускну спроможність даного зв'язку у напрямку глобальної мережі визначено для STM-256 та зафіковано як 39.81 Гбіт/с з коефіцієнтом ефективності 85%. Хмарний ЦОД у свою чергу з'єднаний із глобальною мережею виділеним каналом 2.5G Ethernet, що близче до

нижньої межі із [34], де вимоги ЦОД можуть знаходитися від 1 Гбіт/с до 40 Гбіт/с. Щодо затримки комунікації, то в різних працях її значення дорівнює менше 70 мс [5], 80 мс [1], 100 мс [16] та більше 100 мс [4]. Для даної роботи орієнтовним значенням було обрано 75 мс. Остаточні мережеві параметри системи моделі адаптивної платформи наведено в таблиці 2.4.

$$bw_8 = 39.81 \cdot 10^9 \frac{\text{біт}}{\text{с}} \cdot 0.85 = 33.839 \cdot 10^9 \frac{\text{біт}}{\text{с}}, \quad (2.11)$$

$$bw_9 = 2.5 \cdot 10^9 \frac{\text{біт}}{\text{с}} \cdot 0.85 = 2.125 \cdot 10^9 \frac{\text{біт}}{\text{с}}. \quad (2.12)$$

Таблиця 2.4 – Мережеві параметри системи

№	Зв'язок (технологія)	Пропускна здатність $bw_i$ , біт/с	Затримка передачі $l_i$ , мс
1	Датчик – смартфон (BLE)	–	352
2	Смартфон – актуатор (BLE)	–	2
3	Смартфон – базова станція (3G)	$2.5 \cdot 10^6$	90
4	Смартфон – базова станція (4G)	$25 \cdot 10^6$	45
5	Базова станція 3G – шлюз Інтернет провайдера (IMA/E3)	$29.213 \cdot 10^6$	10
6	Базова станція 4G – шлюз Інтернет провайдера (2.5G Ethernet)	$2.125 \cdot 10^9$	5
7	Сервер лікарні – шлюз Інтернет провайдера (Gigabit Ethernet)	$850 \cdot 10^6$	5
8	Шлюз Інтернет провайдера – хмарний ЦОД (STM-256)	$33.839 \cdot 10^9$	75
9	Хмарний ЦОД – шлюз Інтернет провайдера (2.5G Ethernet)	$2.125 \cdot 10^9$	75

#### 2.1.4.2 Параметри обчислювальних пристройв

Згідно із визначеною на рисунку 2.7 архітектурою моделі адаптивної платформи, обчислювальними пристроями системи є смартфони, сервери на базових станціях, за необхідності сервер місцевої лікарні, а також хмарний центр обробки даних. iFogSim дає можливість враховувати параметри пристройв, як-от обчислювальна потужність, кількість паралельно працюючих обчислювальних елементів, об'єм оперативної та постійної пам'яті, пропускну спроможність каналу, енергоспоживання та вартість проведення обчислень на ньому. Однак, у даній роботі не враховуються параметри пам'яті та пропускної спроможності: вони призвели б до ускладнення моделі системи та, можливо, чинили би неочікуваний вплив на її роботу. Отже, цим параметрам було присвоєно максимально можливі значення. Симулятор надає можливість розширення моделей енергоспоживання пристройв, але тут використовується стандартна лінійна модель, для якої визначається енергоспоживання в режимі очікування та в режимі повного навантаження, а інші можливі значення цієї величини лежать на прямій між цими двома точками. Таким чином, визначеню підлягають параметри обчислювальної спроможності  $MIPS_i$  у мільйонах операцій за секунду (принципово, що спосіб вимірювання цього показника є спільним для всіх процесорів та відбувається за бенчмарком Dhrystone), кількість обчислювальних елементів (ядер процесора)  $CPU_i$ , енергоспоживання  $W_{idle_i}$ ,  $W_{max_i}$  (режим очікування та максимального навантаження) у ватах та вартості обчислень  $Pr_i$  в умовних одиницях за один мільйон інструкцій.

Першим обчислювальним вузлом у моделі медичного додатку є смартфон пацієнта. В якості цільового процесора цього пристрою обрано Cortex-A55 сімейства ARM середнього цінового сегменту 2017 року випуску [35]. Для його попередника, Cortex-A53 обчислювальна потужність була визначена у [36, 37] та склала 2.3 DMIPS/MHz. На офіційному ресурсі [38] стверджується, що модель 2017 року ефективніша за попередника на 15%, тому його ефективність

$MIPS_1$  приблизно оцінено в 5,300 DMIPS, зафіксувавши частоту роботи на 2 ГГц. Аналогічно у порівнянні з Cortex-A53 оцінено енергоспоживання процесора у 0.02 Вт для режиму низького споживання енергії 0.85 Вт для моменту найактивнішого використання. Залишаючи збільшення ефективності на 10-15% у порівнянні з попередником, для мобільного процесору було визначено параметри  $W_{idle_1} = 0.018$  Вт,  $W_{max_1} = 0.7225$  Вт. Вважається, що клієнтський додаток займає лише одне ядро процесора смартфона ( $CPU_1 = 1$ ), а вартість обчислень на пристрой не підлягає розрахунку і дорівнює нулю.

$$MIPS_1 = 2.3 \frac{MIPS}{\text{МГц}} \cdot 0.85 \cdot 2,000 \text{ МГц} = 5,300 \text{ MIPS}, \quad (2.13)$$

$$W_{idle_1} = 0.02 \text{ Вт} \cdot 0.9 = 0.018 \text{ Вт}, \quad (2.14)$$

$$W_{max_1} = 0.85 \text{ Вт} \cdot 0.85 = 0.7225 \text{ Вт}. \quad (2.15)$$

Для спрощення моделі платформи прийнято, що як на обчислювальних вузлах базових станцій, на сервері лікарні, так і у хмарному центрі обробки даних влаштовані процесори одного типу: Intel Core i7-5960X, який випустили на ринок 2014 року. Згідно з результатами тестування [39], його обчислювальна потужність оцінюється у 298,190 DMIPS, або 300,000 MIPS після округлення. Але, як наведено в [40] оцінка потужності за Dhrystone є одноядерним, або точніше однопотоковим, тестом. Тому зазвичай його показник складається для всіх ядер процесора, яких в Intel Core i7-5960X всього 16, враховуючи технологію Intel Hyper-Threading. Таким чином, на одне ядро припадає 18,750 DMIPS. Стосовно енергоспоживання обраного процесора, у режимі очікування він споживає 69 ват, а в режимі максимального навантаження – 189 ват [41]. Це значення також стосується усіх ядер процесора загалом, тоді як цільовим є значення спожитої електроенергії за одиницю часу єдиним логічним ядром. Звідси й отримано значення  $W_{idle_2} = 4.3125$  Вт,  $W_{max_2} = 11.8125$  Вт.

$$MIPS_2 \approx 300 \cdot 10^3 \cdot MIPS \cdot \frac{1}{16} = 18,750 \text{ MIPS}, \quad (2.16)$$

$$W_{idle_2} = 69 \text{ Вт} \cdot \frac{1}{16} = 4.3125 \text{ Вт}, \quad (2.17)$$

$$W_{max_2} = 189 \text{ Вт} \cdot \frac{1}{16} = 11.8125 \text{ Вт}. \quad (2.18)$$

Кількість ядер інших обчислювальних вузлів не є фіксованою величиною та може змінюватись від одного сценарію до іншого. Але в загальному випадку в хмарному центрі обробки даних можуть бути доступні від 16 до 128 повноцінних процесорів, тобто загальна кількість ядер  $CPU_2$  складатиме від 256 до 2048. Туманні вузли обчислень, представлені серверами на базових станціях, містять значну меншу кількість ресурсів: доступними можуть бути від декількох ядер до одного або двох повноцінних процесорів. У такому випадку  $CPU_3$  становитиме від 3 до 16 ядер. За наявності у сценарії моделювання міської лікарні з приватним сервером, можна розраховувати на кілька повноцінних процесорів, тоді значення  $CPU_4$  дорівнюватиме 64.

Тепер визначеню підлягає лише вартість використання зазначених обчислювальних ресурсів. Симулятор iFogSim припускає, що мірою плати за використання апаратних потужностей обчислювального центру є певна кількість умовних одиниць за один мільйон інструкцій, виконаний на обчислювальному обладнанні. Однак така міра більш характерна для мейнфреймів, аніж для хмарних обчислень з орендованими віртуальними машинами або контейнерами, тому вимагає додаткових обчислень. Із цих міркувань здійснено певний перехід від плати за оренду віртуальної машини визначеної потужності до плати за аналогічну потужність у середовищі мейнфреймів. У [42], посилаючись на працю [43], було визначено, що аналогічна потужність у 15,200 MIPS у середовищі Amazon коштує 168,050 \\$ щорічно. Ці дані було взято за еталонні, щоб розрахувати вартість  $Pr_2$  мільйону інструкцій для модельованого хмарного ЦОД. Для обчислювальних вузлів,

розміщених на базових станціях, було обрано значення вартості мільйону інструкцій  $Pr_3$ , яке на 10% перевищує вартість у хмарному ЦОД. Сервер місцевої лікарні, у свою чергу, вважається приватним, отже не вимагає грошових витрат  $Pr_4$  за обчислювальні ресурси. Остаточні параметри всіх обчислювальних елементів визначені в таблиці 2.5.

$$Pr_2 = \frac{168,050 \frac{\text{у. о.}}{\text{рік}}}{15,200 \text{ MIPS} \cdot 365 \cdot 24 \cdot 3600 \frac{\text{c}}{\text{рік}}} = 351 \cdot 10^{-9} \frac{\text{у. о.}}{\text{млн. інстр.}}, \quad (2.19)$$

$$Pr_3 = 1.1 \cdot Pr_2 = 386 \cdot 10^{-9} \frac{\text{у. о.}}{\text{млн. інстр.}}, \quad (2.20)$$

$$Pr_4 = 0. \quad (2.21)$$

Таблиця 2.5 – Параметри обчислювальних вузлів системи:

<b>№</b>	<b>Тип пристрою</b>	<b>Потужність <math>MIPS_i</math>, млн. інстр./с (1 процесор)</b>	<b>Кількість процесорів (ядер), <math>CPU_i</math></b>	<b>Енергоспоживання <math>W_{idle_i}/W_{max_i}</math>, Вт</b>		<b>Вартість <math>Pr_i</math>, у.о./млн.інстр.</b>
1	Смартфон	5,300	1	0.018	0.723	0
2	Хмарний ЦОД	18,750	256 – 2048	4.313	11.813	$351 \cdot 10^{-9}$
3	Сервер базової станції	18,750	3 – 16	4.313	11.813	$386 \cdot 10^{-9}$
4	Сервер міської лікарні	18,750	64	4.313	11.813	0

## 2.2 Опис сценаріїв моделювання

### 2.2.1 Дослідження інжекції ідеалізованих рішень

Метою проведення даного сценарію є дослідження впливу початкової інжекції ідеалізованих на якість роботи генетичного алгоритму. Як зазначалось раніше, в загальному випадку немає обмежень на початкову популяцію. Але існує ймовірність, що наповнення початкової популяції ідеалізованими рішеннями може значно покращити отримані генетичним алгоритмом результати. Схема інжекції наведена нижче на рисунку 2.8.

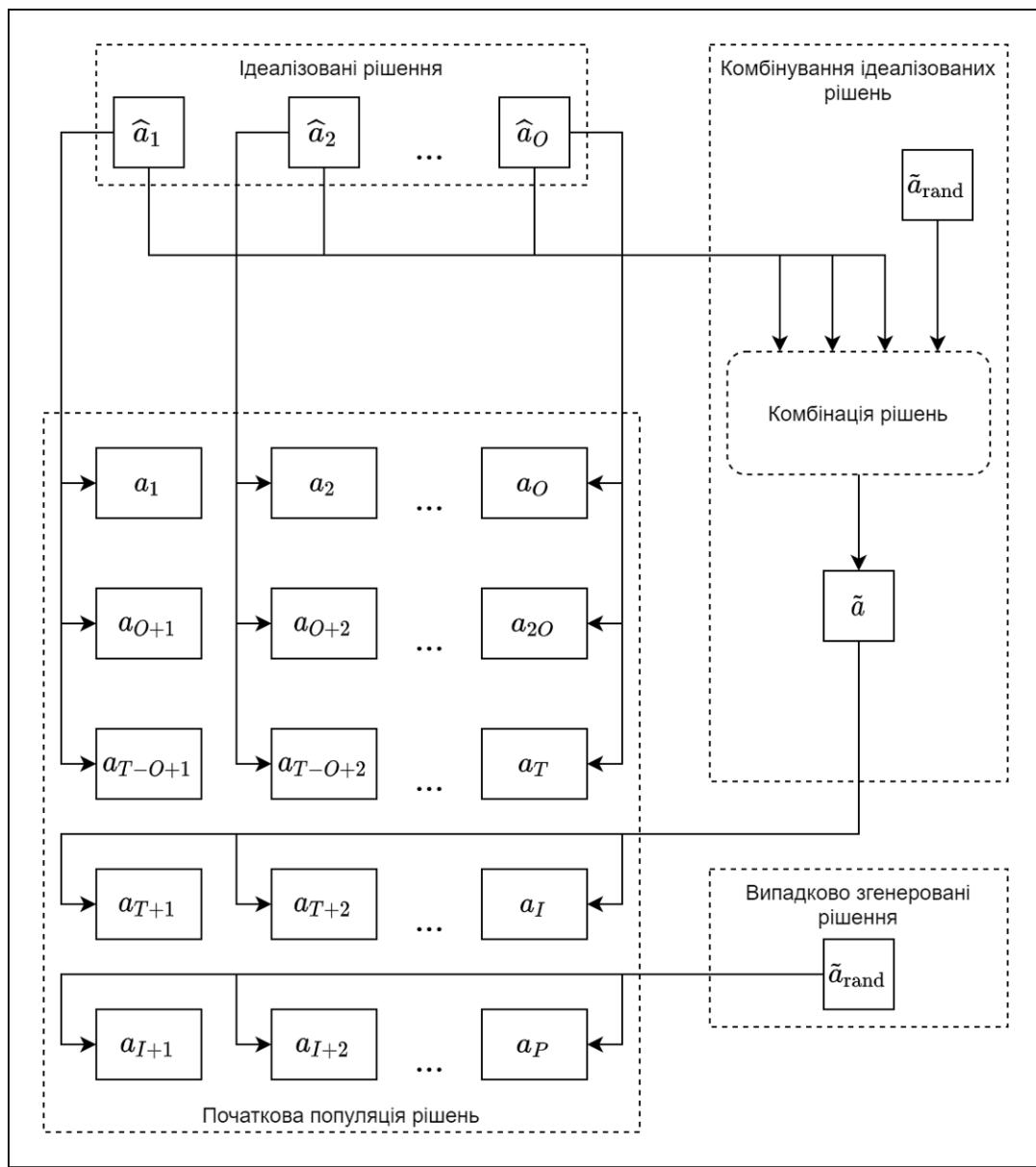


Рисунок 2.8 – Схема інжекції ідеалізованих рішень

У звичайному випадку для кожної цільової функції у відповідність ставиться одне ідеалізоване рішення. Наприклад, для мінімізації енергоспоживання ідеальним буде розмістити всі модулі додатку на пристрой з найменшим споживанням електроенергії. Для мінімізації вартості обчислень слід обрати пристрій з найбільш дешевою ціною за обчислювальний ресурс. Для зменшення затримки можна розмістити модулі додатку на найближчому до них обчислювальному вузлі. Але в загальному випадку кількість ідеалізованих рішень  $O$  може відрізнятись від кількості цільових функцій.

Частина початкової популяції  $T$  заповнюється незмінними копіями цих рішень та визначається рівнянням 2.22. Варійованим параметром виступає коефіцієнт впевненості, який визначає скільки незмінених ідеалізованих рішень потрапить в популяцію.

$$T = \text{conf} \cdot I, \quad (2.22)$$

де  $T$  – кількість ідеалізованих рішень у популяції,

$\text{conf}$  – коефіцієнт впевненості,

$I$  – загальна кількість інжектованих рішень

$P$  – розмір початкової популяції.

Після заповнення перших  $T$  рішень відбувається перехід на другий етап: інжекції комбінацій ідеалізованих рішень. Решта  $I - T + 1$  рішень в популяції заповнюється шляхом вибору «гену» в термінах генетичного алгоритму із довільно вибраного ідеалізованого рішення або рішення, яке було згенеровано випадково. Альтернативою описаного процесу комбінування є базові для генетичних алгоритмів операції кросоверу та мутації. Значення загальної кількості інжектованих рішень  $I$  визначається співвідношенням 2.23. У ньому керованим параметром є  $\epsilon_{\text{inj}}$  – частка інжектованих рішень відносно загального розміру популяції алгоритму.

$$I = inj \cdot P, \quad (2.23)$$

де  $inj$  – частка інжектованих рішень.

У даному експериментальному сценарії досліджено лише частковий випадок цієї модифікації: проводиться мінімізація часової затримки для обробки задач користувачів, тобто однієї цільової функції. Множина ідеалізованих рішень в цьому сценарії складається з одного елементу, який визначає розміщення модулів додатків на обчислювальному пристрої базової станції найближчої до смартфону пацієнта. Цей експериментальний сценарій передбачає єдиний запуск генетичного алгоритму, який відбувається після початкового розміщення користувачів на карті. Таким чином, генетичний алгоритм повинен оптимізувати розміщення одразу всіх модулів додатку, що підлягають міграції (модуль аналізу стану пацієнта, модуль з електронною медичною карткою та модуль аналізу відхилень), між обчислювальними вузлами. Після цього симуляція триває ще 50 секунд, а в кінці обчислюється середня для всіх користувачів затримка обробки даних, яка і стане критерієм ефективності генетичного алгоритму.

Параметр  $conf$  не підлягає дослідженню в межах цього сценарію та має постійне значення 0.66. У свою чергу варійованим параметром є  $inj$ , який визначає загальну частку популяції генетичного алгоритму, яку займатимуть ідеалізовані рішення, та змінюється у межах від 0 до 1 включно з кроком 0.2. Також, до варійованих параметрів належать загальні для генетичних алгоритмів розмір популяції  $P$  в межах від 2 до 102 включно з непостійним кроком дискретизації та максимальна кількість ітерацій алгоритму  $Iter_{max}$ , яка прийматиме значення від 5 до 100 також з непостійним кроком. Останнім варійованим параметром є кількість користувачів системи  $N$ , які беруть участь в моделюванні: від 10 до 40 з кроком 2. З огляду на зазначені параметри буде досліджено ефективність описаної модифікації генетичного алгоритму.

Було внесено зміни у деякі вхідні параметри моделі із попереднього підрозділу. По-перше, затримку лінії зв'язку між шлюзом Інтернет-провайдера та хмарним центром обробки даних було підвищено до 200 мс, щоб зробити вкрай непривабливим. У такому випадку генетичний алгоритм має оптимально розмістити модулі додатку лише по обчислювальним вузлам базових станцій. По-друге, кількість доступних ядер на цих обчислювальних пристроях визначено як 3. Обмеженість в цьому ресурсі очікувано спричинить зростання затримки обробки даних із ростом користувачів, але такі умови не мають створити непередбачуваного впливу на систему. По-третє, ймовірність запиту інформації з електронної картки пацієнта було встановлено в 100%, щоб виключити вплив цієї невизначеності на систему. З цією ж метою мережевим параметрам базових станцій 4G було присвоєно параметри 3G: в іншому випадку це призвело до великих відхилень значень затримок кожного користувача від загального середнього значення. Нарешті, із сценарію моделювання було виключено сервер міської лікарні, адже він не є необхідним для оцінки ефективності генетичного алгоритму в межах даного сценарію.

### **2.2.2 Дослідження міграції на основі генетичного алгоритму**

Метою цього експериментального сценарію є дослідження ефективності ітераційного застосування генетичного алгоритму для підтримання оптимального значення цільової функції в динамічній системі. У цьому випадку динамічний аспект системи визначається зміною базових станцій, до яких під'єднані користувачі під впливом їхньої мобільності. Як і в минулому сценарії вибрана єдина цільова функція: зменшення затримки обробки даних користувачів в основному циклі додатку.

Усі модифікації, які досліджено в межах цього сценарію слід розглянути в двох аспектах (рисунок 2.9): по-перше, з точки зору характеру ініціації процесу оптимізації, а по-друге, з точки зору поділу множини всіх змінних системи на кластери. Серед типів ініціації виділено періодичну (або заплановану) ініціацію, яка запускає процес оптимізації через визначені

проміжки часу, та реактивну ініціацію, яка запускає процес оптимізації в той момент, коли в системі відбулися зміни (в даному контексті мобільний користувач змінив базову станцію). Множиною змінних системи, які підлягають оптимізації є розміщення модулів додатків. У загальному випадку можна поділити множину змінних на довільну кількість кластерів  $C$ , а процес оптимізації проводити для модулів окремого кластера. Тоді оптимізація розташування всіх модулів одночасно буде частковим випадком описаної схеми, коли кількість кластерів дорівнює одиниці.

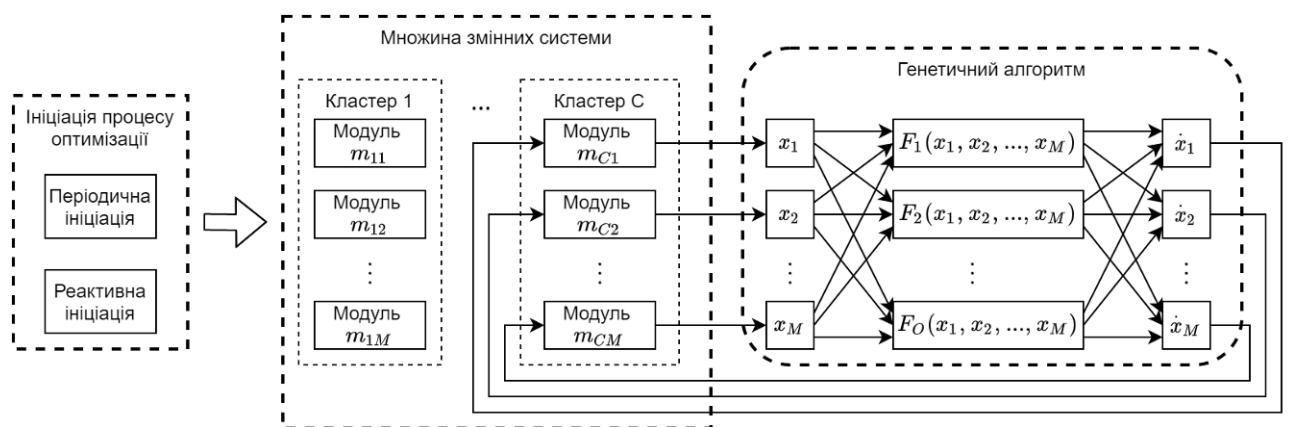


Рисунок 2.9 – Оптимізація динамічної системи

Згідно з наведеною класифікацією було для дослідження було обрано п'ять моделей оптимізації, які описано у таблиці 2.6. Модель 1 містить єдиний кластер для всіх модулів, проводячи оптимізацію для них одночасно та ініціює цей процес реактивно. Модель 2 протилежна попередній у тому аспекті, що в ній зміна базової станції мобільним користувачем ініціює оптимізацію виключно його модулів (а таких, що здатні мігрувати, всього три у даному додатку). Моделі 3 та 4 обмежую розмір кластеру 30 елементами, але у третьої процес ініціації не залежить від змін в системі та відбувається періодично, в той час як в моделі 4 задіяна реактивна ініціація. Модель 5 є поєднанням характеру ініціації моделей 3 та 4, суміщаючи реактивну та періодичну. Модель 6 є складеною із моделей 2 та 3.

Таблиця 2.6 – Моделі оптимізації для експерименту

№ моделі	Характер ініціації оптимізації (період)	Кількість модулів в кластері, $M$
1	Реактивна	Всі
2	Реактивна	3
3	Періодична (1 хвилина)	30
4	Реактивна	30
5	Періодична (1 хвилина) та реактивна	30, 30
6	Періодична (1 хвилина) та реактивна	30, 3

Для всіх моделей було створено середовище аналогічне попередньому сценарію з усіма його припущеннями та спрощеннями. Час моделювання збільшено до однієї години, кількість користувачів фіксована та дорівнює 60. Дляожної моделі було здійснено 9 запусків з різними параметрами розміру популяції  $P \in \{10, 40, 102\}$  та кількості ітерацій  $Iter_{max} \in \{10, 30, 60\}$ . Однак, слід зазначити, що, оскільки модель 6 є поєднанням моделей 2 та 3, параметри  $P$  та  $Iter_{max}$  стосуються лише частини моделі з періодичним характером (2), а для реактивної частини моделі параметрам присвоєно значення  $P = 40$ ,  $Iter_{max} = 10$ . Параметр частки інжектованих рішень  $inj$  встановлено у 100%, а коефіцієнт впевненості залишено  $conf = 0.66$ . Аналізу підлягає об'єм (кількість модулів) та частота міграції упродовж часу моделювання. Також необхідно порівняти значення середньої затримки для всіх пристрій наприкінці моделювання та час, витрачений на моделювання, який дозволить оцінити ефективність застосування моделей.

### 2.2.3 Сценарій багатоцільової оптимізації

Як вже було зазначено, у роботі [5] генетичний алгоритм NSGA-II дозволив отримати задовільні рішення задачі розташування модулів додатку на обчислювальних вузлах із врахуванням трьох цільових функцій. Метою даного сценарію є перевірка доцільності використання цього генетичного алгоритму

для організації багатоцільової інтелектуальної міграції. В якості цільових функцій системи обрано часову затримку обробки даних користувачів та загальної вартості обчислень.

Сценарій симуляції включає чотири запуски моделі з кількістю користувачів  $N = 100$ , яка триватиме 4 години 45 хвилин модельного часу. Перший запуск здійснюється зі стратегією хмарного розміщення додатків: модулі аналізу стану пацієнту та визначення відхилень розміщуються виключно на хмарному ЦОД. При такій стратегії очікується найдешевша вартість обчислень, але й велика часова затримка. Другий запуск відповідає крайовому розміщенню додатків: зазначені два модулі кожного користувача запускаються на обчислювальному вузлі найближчої базової станції та здійснюють реактивну міграцію із переключенням користувача з одної базової станції до іншої. Очікується, що в результаті цього запуску буде отримано найшвидше з точки зору затримки, але й найдорожче рішення. У третьому та четвертому запусках використано стратегію на основі генетичного алгоритму NSGA-II у формі третьої моделі з попереднього сценарію. Для третього запуску період для запланованої ініціації процесу оптимізації встановлено на 5 хвилин, для четвертого – на 1 хвилину модельного часу. Інші параметри для них є спільними: розмір популяції  $P = 40$ , кількість ітерацій  $Iter_{max} = 30$ , частка інжектованих рішень  $inj = 100\%$ , а коефіцієнт впевненості  $conf = 66\%$ , кількість модулів у кластері  $M = 30$ . Від цих стратегій очікується отримання компромісних рішень для заданих цільових функцій.

На відміну від попередніх двох сценаріїв, параметрам моделі адаптивної платформи було присвоєно початкові значення. Затримку зв'язку між шлюзом Інтернет провайдера та хмарним ЦОД було встановлено як 75 мс, а станціям базовим станціям 4G було повернуто відповідні параметри. Це має спричинити значно менші значення середньої затримки для користувачів у порівнянні з попередніми сценаріями. З огляду на більшу кількість користувачів кількість ядер хмарного ЦОД було збільшено до 2048, на обчислювальних вузлах

базових станцій – до 16, а на сервері міської лікарні – до 64. Накладено додаткове обмеження про неможливість міграції модулю з електронною книжкою пацієнта із міркувань, що вона має зберігатись на безпечному сервері міської лікарні, тому в міграції беруть участь лише два модулі для кожного користувача. Значення ймовірності запитів до модулю електронної картки повернено до 1%.

## 2.3 Висновки до розділу 2

У межах цього розділу було описано та визначено основні параметри моделі адаптивної туманної платформи для медичних додатків. Спершу, в парадигмі розподілених потоків даних у формі орієнтованого графу було розроблено модель медичного додатку, з датчиком, діячем, модулем смартфону пацієнта і трьома функціональними модулями, які запускаються на адаптивній платформі та підлягають міграції. Опираючись на реальні дані, було підготовлено сценарії мобільності користувачів та розташування базових станцій 3G та 4G зв’язку. Наведено архітектуру моделі адаптивної платформи, визначено мережеві параметри зв’язків між пристроями з огляду на існуючі мережеві стандарти та параметри обчислювальних пристройів.

Детальніше описано модифікації та підходи, які підлягають дослідженню в експериментальних сценаріях симуляцій. По-перше, сформульовано підхід, що полягає в інжекції ідеалізованих рішень в початкову популяцію генетичного алгоритму, та визначено умови для першого сценарію моделювання. Далі розроблено другий сценарій для перевірки ефективності застосування генетичного алгоритму у вигляді шести різних моделей для оптимізації динамічної системи. Нарешті, описано третій сценарій, який має продемонструвати доцільність використання генетичного алгоритму NSGA-II для організації інтелектуальної міграції з необхідністю мінімізації критеріїв затримки обробки даних та загальної вартості обчислень.

## 3 АНАЛІЗ РЕЗУЛЬТАТІВ СИМУЛЯЦІЇ

### 3.1 Аналіз ефективності інжекції ідеалізованих рішень

Як було описано в минулому розділі, у першому сценарії моделювання вхідними параметрами були кількість користувачів системи  $N \in [10; 40]$ , розмір популяції генетичного алгоритму  $P \in [2; 102]$ , максимальна кількість ітерацій алгоритму  $Iter_{max} \in [5; 100]$  та частка інжектованих ідеалізованих рішень  $inj \in [0; 1]$ . Вихідним значенням моделі є середня затримка обробки даних в головному циклі додатку  $\bar{d}(N, P, Iter_{max}, inj)$ .

На рисунку 3.1 зображене результат першого сценарію моделювання у вигляді групи тривимірних точкових діаграм. Окрема діаграма відповідає фіксованому значенню параметра  $inj$ , трьома осями діаграм є  $N$ ,  $P$  та  $Iter_{max}$ , а колір визначається середньою затримкою  $\bar{d}$ : жовтий колір означає нижчу затримку та відповідно кращий результат; фіолетовий колір – вищу затримку та гірший результат. До значень середньої затримки застосовано згладжування фільтром Гауса з параметром середньоквадратичного відхилення  $\sigma = 0.5$ .

Із наведених діаграм можна зробити декілька висновків. По-перше, зменшення затримки зі збільшенням кількості ітерацій при довільних фіксованих значеннях кількості користувачів та розміру популяції свідчить про правильну роботу алгоритму. По-друге, в середньому зі збільшенням кількості користувачів, що спричиняє ускладнення задачі оптимізації, необхідна більша кількість ітерацій генетичного алгоритму, щоб досягти задовільних значень середньої затримки. Нарешті, застосування модифікації з інжекцією ідеалізованих рішень значно покращує ефективність роботи генетичного алгоритму: для досягнення задовільного результату необхідна менша кількість ітерацій та менший розмір популяції, що значно дає змогу пришвидшити роботу алгоритму. Таким чином можна зробити припущення, що доцільно вибирати якомога більший параметр частки інжектованих рішень аж до 100%.

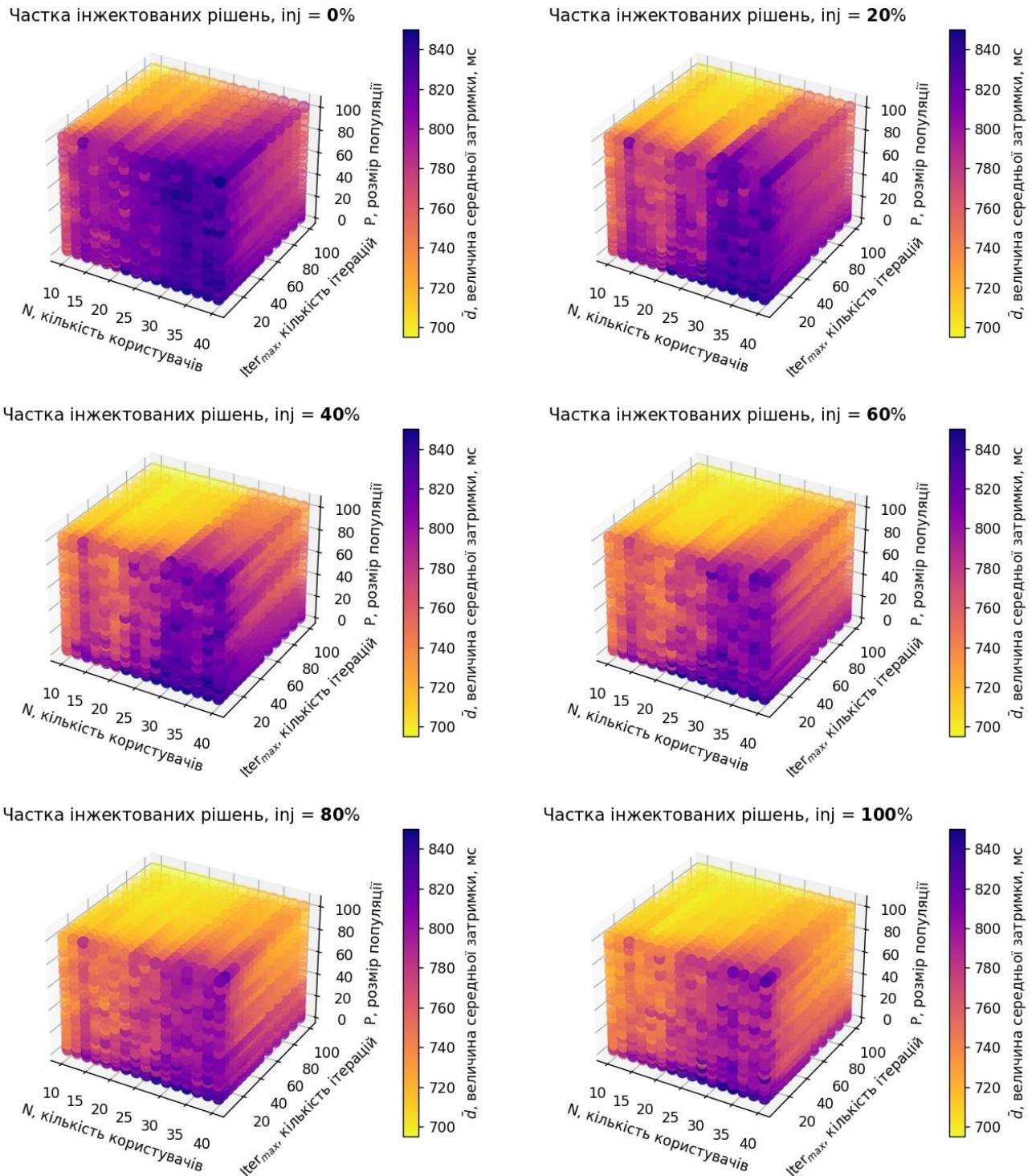


Рисунок 3.1 – Діаграми затримки для різної частки інжектованих рішень

Доцільно розглянути отримані дані з точки зору покращення ступеня покращення, яке можна отримати зі збільшенням параметрів  $N$ ,  $P$ ,  $Iter_{max}$  та  $inj$ . Для цього на наступних візуалізаціях наводяться двовимірні кольорові карти. Всі цих діаграм відповідають двом обраним параметрам. Колір

відповідної області карти визначається значенням часткової похідної середнього значення затримки  $\bar{d}$  по першому параметру але з усередненням за третім параметром та нормуванням. Таким чином колір на діаграмі двох параметрів  $p_1$  та  $p_2$  визначається деяким значенням  $\tilde{g}(p_1, p_2)$ , яке обчислюється за співвідношенням 3.2: перший множник є частковою похідною  $\tilde{d}$  по  $p_1$ , а другий відповідає за нормування цієї похідної, щоб можна було співставляти значення похідної  $\tilde{d}$  по  $p_1$  для різних значень параметрів  $p_2$ . Такий підхід дозволить оцінити наскільки ефективно покращується знайдене рішення при зміні параметра  $p_1$  для всіх значень параметра  $p_2$ , а аналіз двовимірної кольорової карти проводити легше, аніж тривимірної точкової діаграми, як на рисунку 3.1.

$$\tilde{d}(p_1, p_2, inj) = \frac{1}{\|p_3\|} \cdot \sum_{p_3} \bar{d}(p_1, p_2, p_3, inj), \quad (3.1)$$

$$\tilde{g}(p_1, p_2, inj) = \frac{\partial \tilde{d}(p_1, p_2, inj)}{\partial p_1} \cdot \frac{1}{\sum_{p_2} \tilde{d}^2(p_1, p_2, inj)} \quad (3.2)$$

Таким чином, на рисунку 3.2 зображена залежність  $\tilde{g}(Iter_{max}, N, inj)$ : для кожного обчисленого значення частки інжектованих рішень  $inj$  побудовано кольорову кару, на якій можна оцінити, наскільки швидко зменшується затримка  $\bar{d}$  при збільшенні максимальної кількості ітерацій  $Iter_{max}$  окремо для всіх значень кількості користувачів  $N$ . Також слід зазначити, що застосовано фільтр Гауса з параметром  $\sigma = 1$ . Серед висновків, по-перше, можна відмітити, що для більшості значень кількості користувачів збільшення кількості ітерацій до 20 супроводжується найбільшим ростом якості знайдених рішень. Подальше збільшення кількості ітерацій не призводить до значного покращення для малої кількості користувачів але є доцільним для великої. По-друге, порівнюючи кольорові карти для різних значень  $inj$ , можна зробити висновок, що збільшення частки інжектованих рішень призводить до збільшення швидкості

зростання якості на малих кількостях ітерацій (до 20-25) і уповільнення зростання якості для більших значень. В цілому, це свідчить про те, що в середньому інжекція ідеалізованих рішень з високим параметром  $inj$  дозволяє досягти прийнятної якості знайденого рішення за меншу кількість ітерацій генетичного алгоритму.

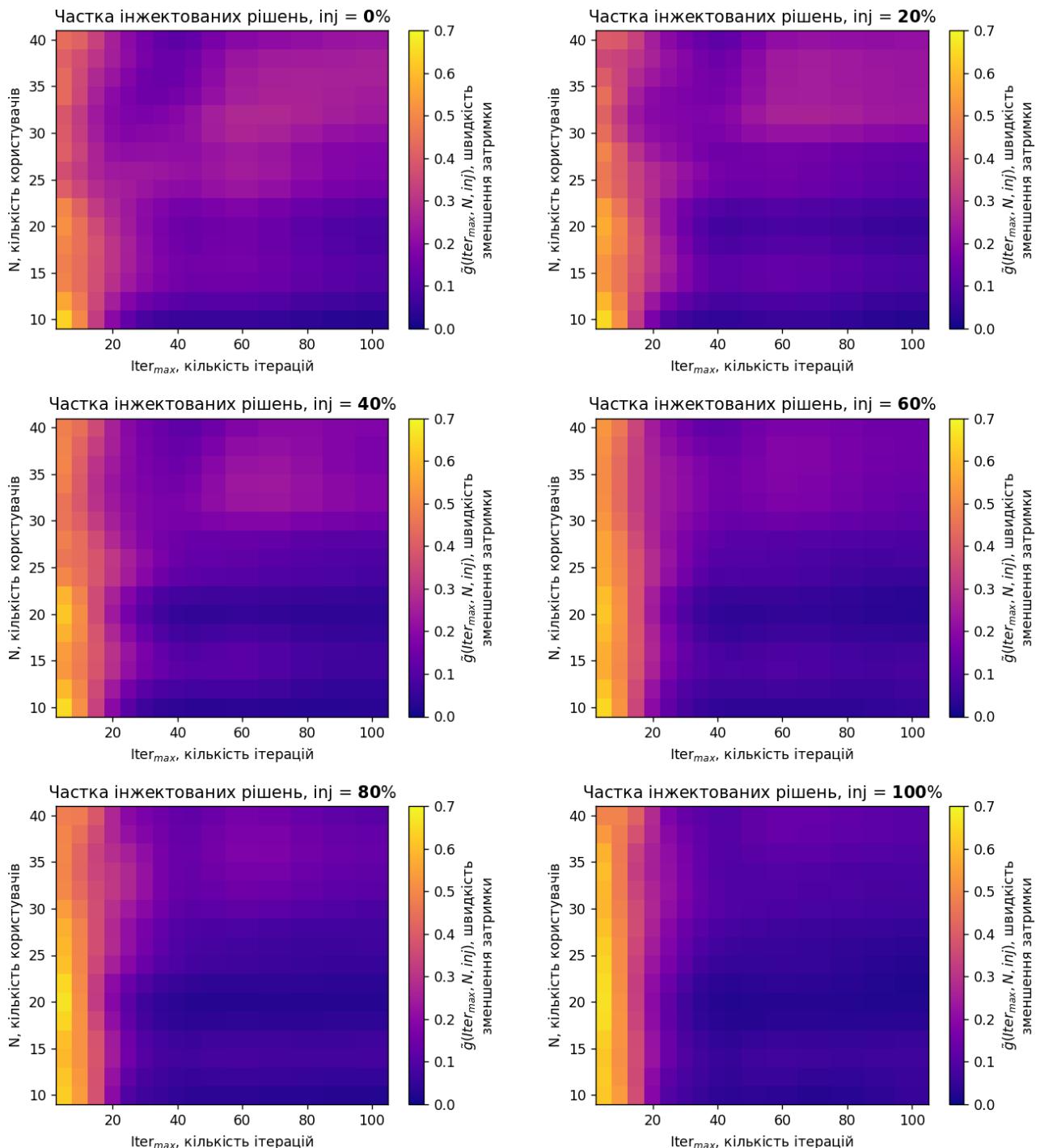


Рисунок 3.2 – Швидкість зменшення затримки при зростанні кількості ітерацій

Аналогічно, на рисунку 3.3 наведено кольорову карту залежності  $\tilde{g}(P, N, inj)$ ,  $\sigma = 2$ . На відміну від попереднього графіку, збільшення розміру популяції може привести до збільшення затримки. В цілому ж, можна стверджувати, що за відсутності інжекції ідеалізованих рішень зростання кількості елементів в популяції не призводить до стрімкого зростання якості роботи алгоритму. Натомість, при появі інжектованих рішень вкрай доцільно збільшувати розмір популяції алгоритму. Як і в попередньому рисунку при високому значенні  $inj$  висока ступінь зростання якості досягається за менший розмір популяції, що знову ж таки, пришвидшить роботу алгоритму.

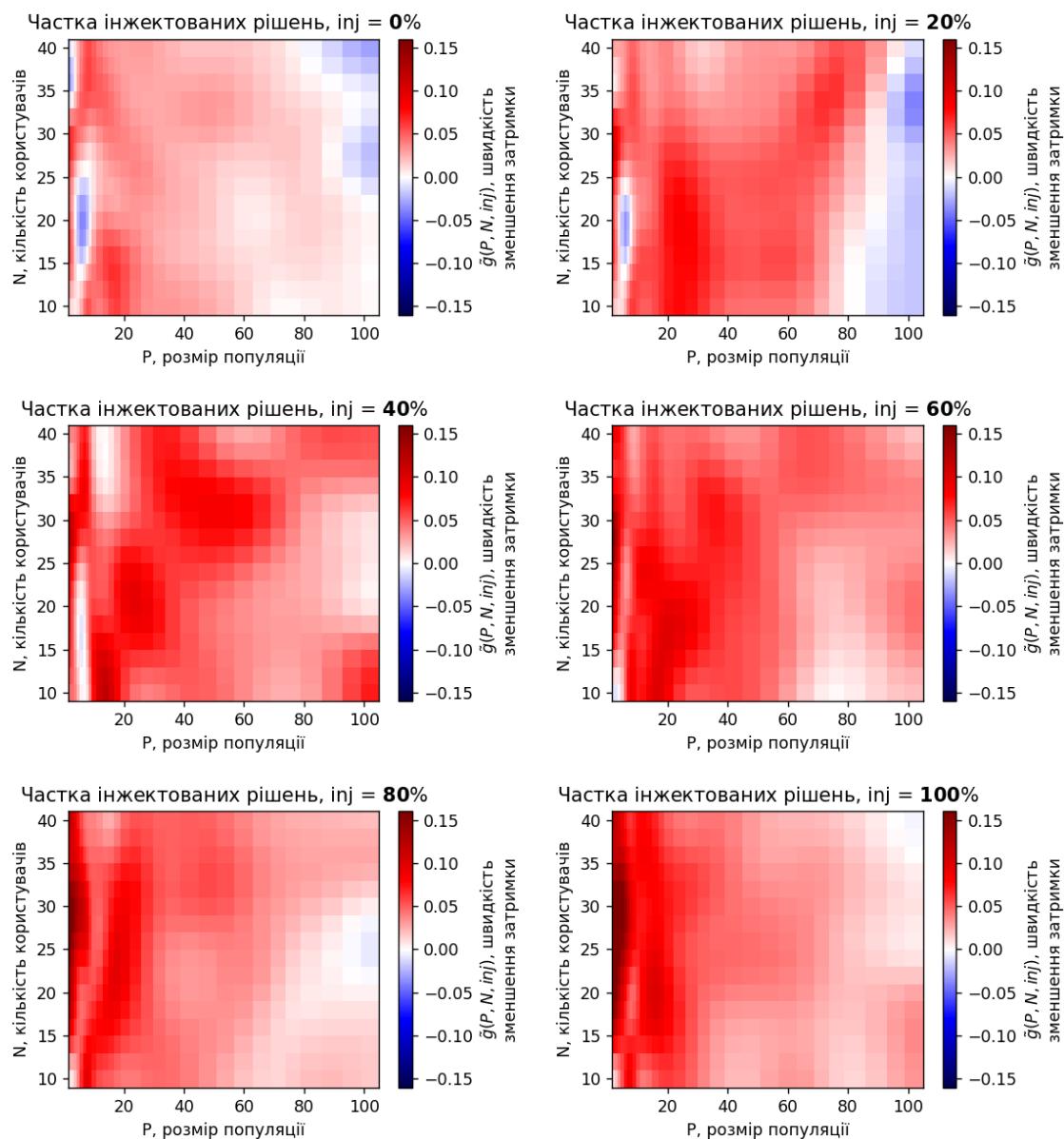


Рисунок 3.3 – Швидкість зменшення затримки при зростанні розміру популяції

Фінальним етапом у аналізі результатів даного сценарію є залежність  $\tilde{g}(Iter_{max}, P, inj)$  (рисунок 3.4),  $\sigma = 1$ . На даній карті продемонстровано, наскільки швидко покращується рішення при збільшенні кількості ітерацій для різних за умови усереднення значення затримки для всіх значень розміру популяції  $P$  кількості користувачів  $N$ . Як і у випадку  $\tilde{g}(Iter_{max}, N, inj)$ , впровадження інжекції ідеалізованих рішень дозволяє значно отримати високу ступінь зменшення середньої затримки за невелику кількість ітерацій, що ще раз підтверджує ефективність даного підходу.

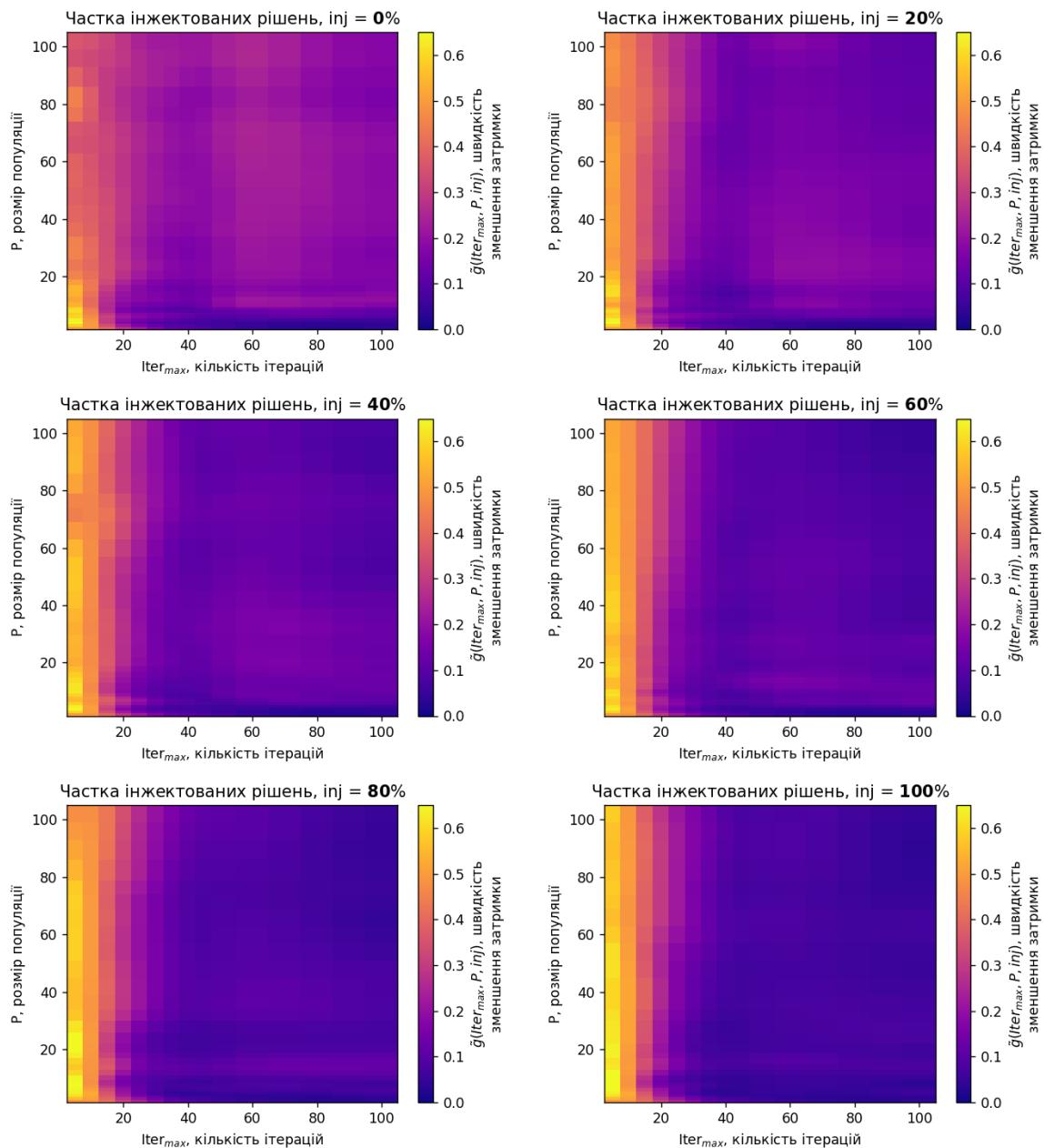


Рисунок 3.4 – Швидкість зменшення затримки при зростанні кількості ітерацій

Таким чином було обґрунтовано, що модифікація, яка полягає в інжекції ідеалізованих рішень виявилася вкрай ефективною, принаймні для одноцільової оптимізації. По-перше, вона робить можливим досягнення набагато кращих значень цільової функції, аніж при використанні випадкових рішень для формування початкової популяції. По-друге, вона призводить до покращення якості зі збільшенням розміру популяції, що зовсім не гарантується при випадковій ініціалізації. По-третє, вона дозволяє досягати прийнятних рішень за значно меншу кількість ітерацій генетичного алгоритму.

### **3.2 Аналіз міграції на основі генетичного алгоритму**

Перша частина результатів включає порівняння обраних шести моделей оптимізації з точки зору частоти прийняття рішень про міграцію та кількості модулів, які змінили своє положення у заданий проміжок часу. Так, на рисунках 3.5, 3.6, 3.7, 3.8, 3.9 та 3.10 кожний стовпчик відповідає результатам, отриманим для конкретної моделі. Перший рядок зображує загальну кількість підключень з одної базової станції на іншу, яка є однаковою для всіх моделей. На спільній горизонтальній осі відкладений проміжок часу від нуля до 3600 секунд. Діаграми являють собою комбінацію стовпчиків, які визначають, що в даний момент часу було здійснено міграцію заданої на вертикальній осі кількості модулів додатку. Синім кольором відображаються міграції, які були ініційовані періодично, а червоним – ініційовані реактивно (для моделей 5 та 6 присутні обидва кольори). На рисунках 3.5, 3.6 і 3.7 розміщено результати для моделей 1, 2 та 3, а на рисунках 3.8, 3.9, 3.10 – для моделей 4, 5 та 6. Для рисунків 3.5 і 3.8 однакове значення параметру  $Iter_{max} = 10$ , для рисунків 3.6 та 3.9  $Iter_{max} = 30$ , а для 3.7 та 3.10  $Iter_{max} = 60$ .

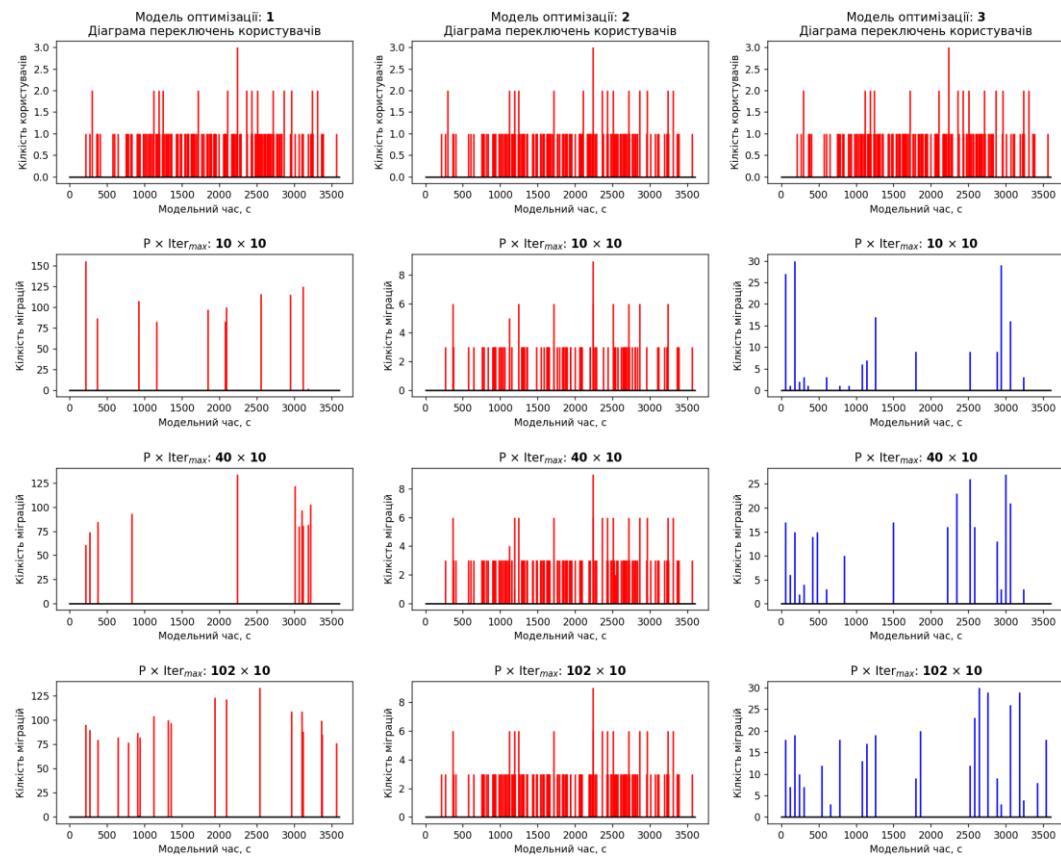


Рисунок 3.5 – Частота міграцій для моделей оптимізації 1, 2, 3 ( $\text{Iter}_{\max} = 10$ )

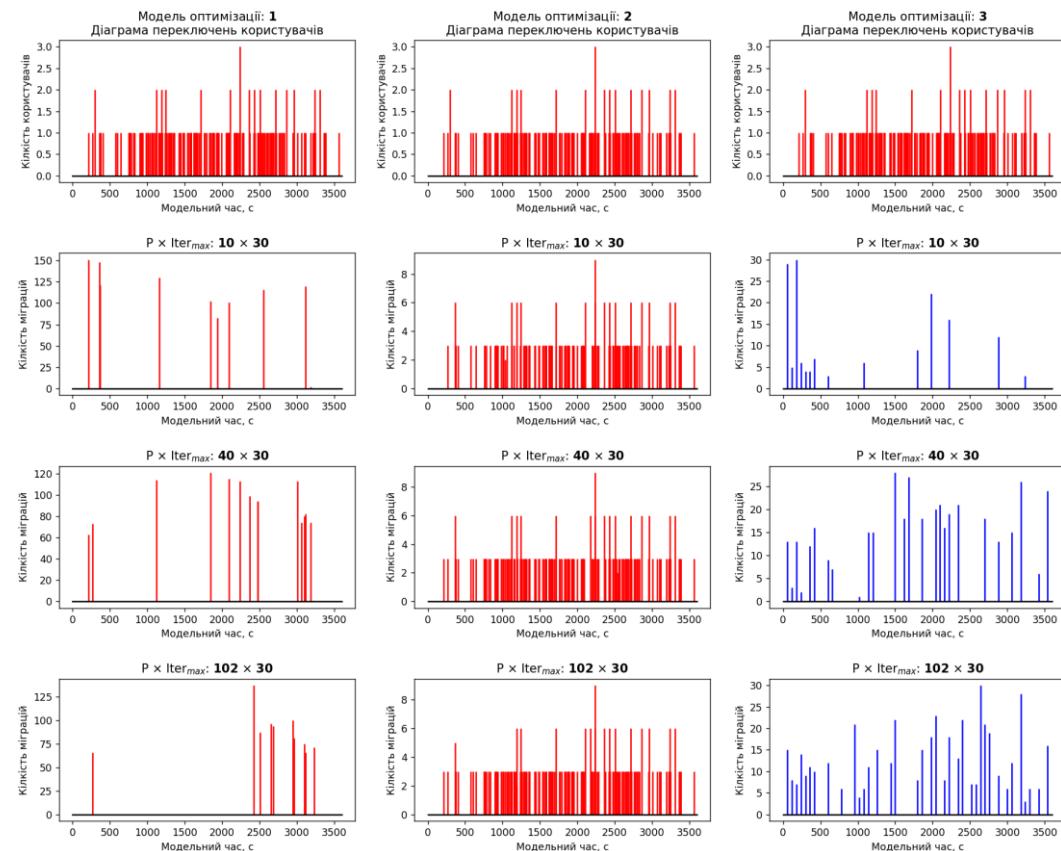


Рисунок 3.6 – Частота міграцій для моделей оптимізації 1, 2, 3 ( $\text{Iter}_{\max} = 30$ )

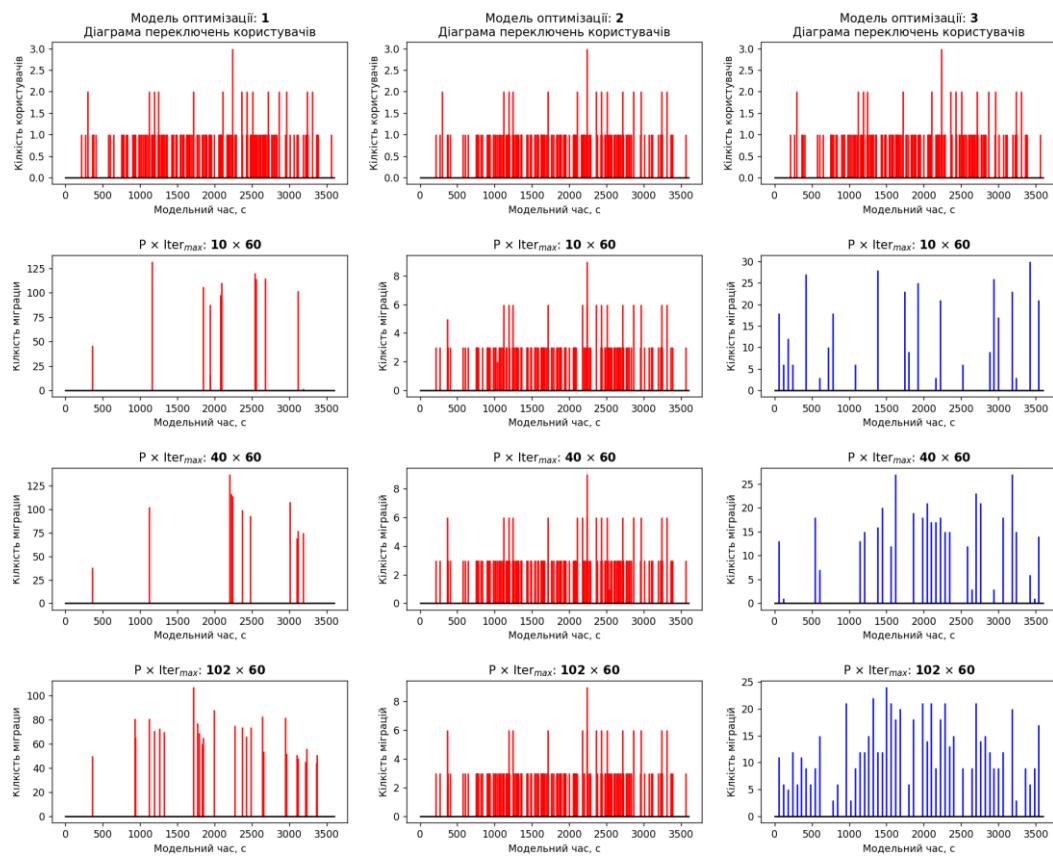


Рисунок 3.7 – Частота міграцій для моделей оптимізації 1, 2, 3 ( $\text{Iter}_{\max} = 60$ )

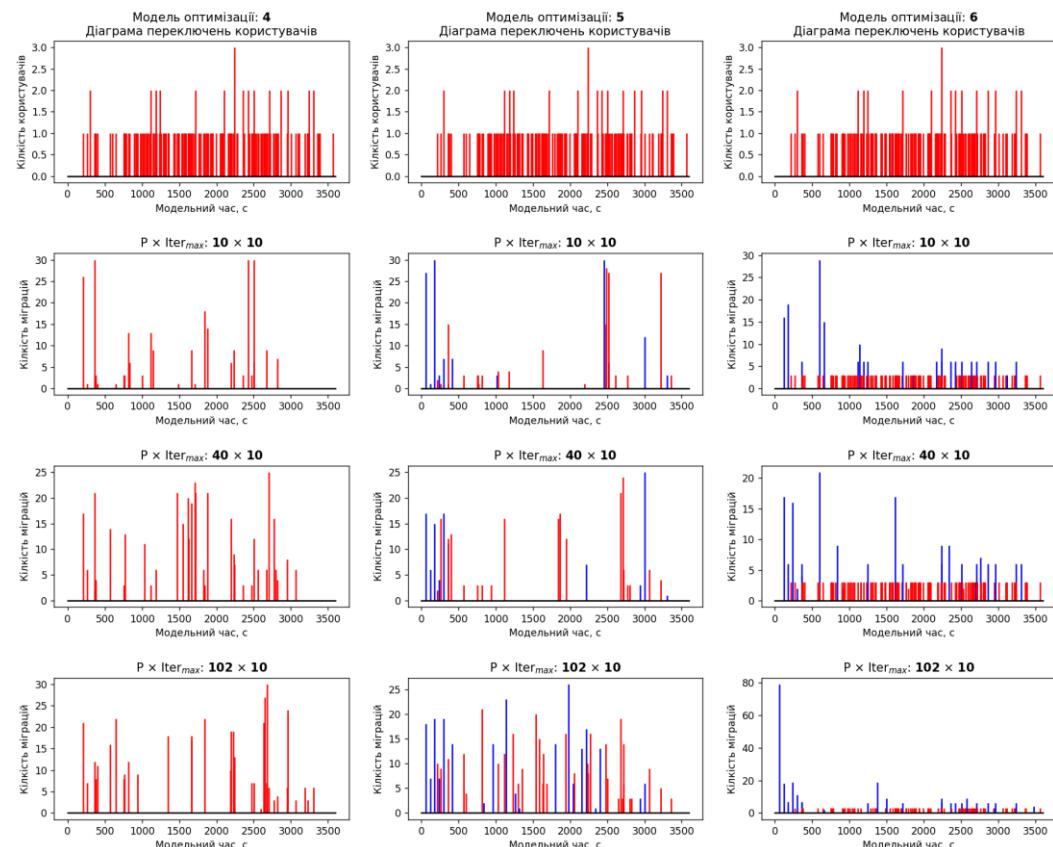


Рисунок 3.8 – Частота міграцій для моделей оптимізації 4, 5, 6 ( $\text{Iter}_{\max} = 10$ )

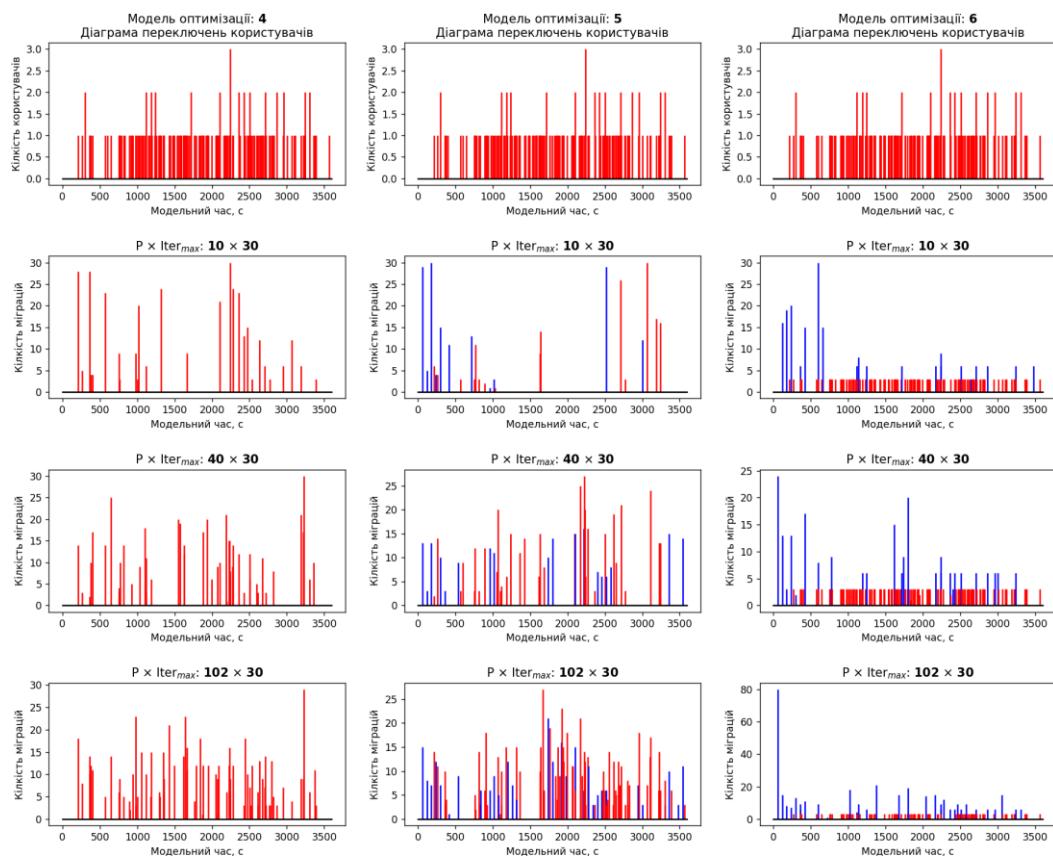


Рисунок 3.9 – Частота міграцій для моделей оптимізації 4, 5, 6 ( $Iter_{max} = 30$ )

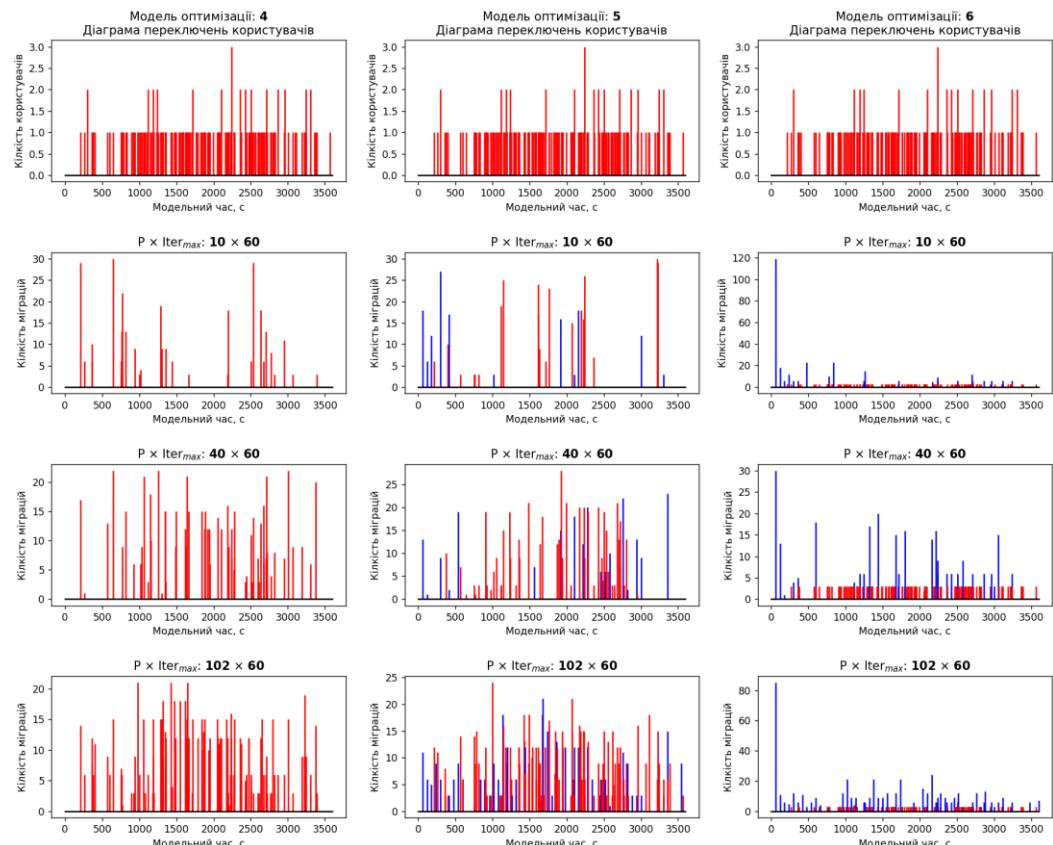


Рисунок 3.10 – Частота міграцій для моделей оптимізації 4, 5, 6 ( $Iter_{max} = 60$ )

У свою чергу, друга частина результатів вже демонструє порівняння ефективності застосування кожної моделі оптимізації з точки зору якості знайдених рішень та часу, який було витрачено на обчислення. На рисунку 3.11 наведено три групи графіків для кожного параметру  $P \in \{10, 40, 102\}$ , за якими можна оцінити результиуюче значення затримки  $\bar{d}$  та витрачений час для його досягнення.

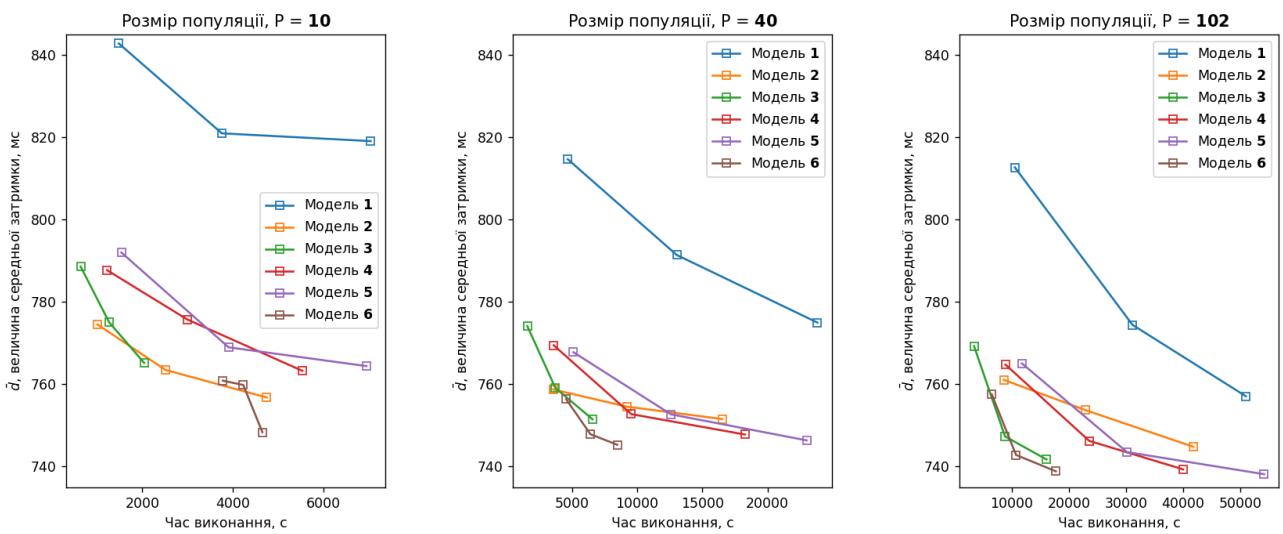


Рисунок 3.11 – Ефективність моделей оптимізації

Як видно з діаграм, використання генетичного алгоритму для динамічної оптимізації з високою кількістю змінних дає незадовільні результати. Про це свідчать результати, отримані для моделі 1: у порівнянні з іншими, для неї характерне найбільше значення затримки  $\bar{d}$ , при тому ж значенні тривалості обчислень. Це пояснюється тим, що генетичний алгоритм здатний поліпшити наступне рішення необхідно або втратити значно більше часу, або знайдене попереднє рішення має втратити свою оптимальність, що супроводжується низькою частотою і високим об'ємом міграції.

Модель 2 за рахунок дуже малої кількості змінних, що беруть участь в процесі оптимізації за малих та середніх значень розміру популяцій та кількості ітерацій, а отже і нижчого часу обчислень, надає задовільні результати: в даних умовах вони кращі або приблизно співпадають з результатами моделей 4 та 5.

Частота та об'єм міграції для моделі 2 майже повністю повторює діаграму переключення користувачів, отже у динамічній системі модель не втрачає можливості дещо поліпшити вже знайдене рішення. Це робить привабливим її застосування для локального покращення при невеликих змінах стану системи.

Модель 3 за характером ініціації не прив'язана до змін у системі та намагається періодично вдосконалити розміщення лише частини від всіх наявних модулів додатку. На відміну від оптимізації всіх змінних одночасно, як це відбувається у моделі 1, покращення лише частини змінних вимагає меншої кількості часу обчислення. Таким чином для середніх та високих значень параметрів  $P$  та  $Iter_{max}$  отримано достатньо високу частоту міграцій об'ємом приблизно 20 модулів, що вимагає значно меншого часу обчислення, аніж деякі альтернативні моделі.

Моделі 4 та 5 мають схожі результати ефективності для всіх параметрів системи: при їх застосуванні отримані приблизно однакові значення середньої затримки  $\bar{d}$ ; модель 5 стабільно потребує дещо більшого часу на виконання. Разом з моделлю 3, моделі 4 та 5 відрізняються між собою характером ініціації процесу оптимізації. Модель 3 носить виключно періодичний характер, модель 4 – виключно реактивний, а модель 5 об'єднує обидва підходи, що і пояснює довший час її роботи. Однак якість отриманих рішень для цих моделей або приблизно співпадає з отриманою для моделі 3, або потребує значно більшого часу обчислення. Таким чином, можна зробити висновок, що для глобального покращення рішень генетичним алгоритмом варто обрати періодичний (запланований) характер ініціації, покращувати його якість шляхом зменшення періоду роботи алгоритму та збільшенням параметрів  $P$  та  $Iter_{max}$ , аніж реагуючи на невеликі зміни в системі.

Найкращим з точки зору ефективності можна вважати модель 6. Увібравши в себе переваги моделей 2 та 3, даний підхід продовжує вдосконуватися глобально шляхом періодичної ініціації процесу оптимізації, що супроводжується розбиттям усієї множини змінних системи на невеликі

кластери, та швидко реагувати на зміни в динамічній системі, реактивно запускаючи процес оптимізації з дуже маленькою кількістю змінних. Для середніх та великих значень параметрів  $P$  та  $Iter_{max}$  комбінація моделей 2 та 3 виявилась ефективнішою за будь-яку з них, застосовану окремо.

Слід зауважити, що хоча локальна оптимізація, яка, наприклад, ініціюється реактивно, демонструє свою ефективність в даному сценарії, поведінка таких моделей вимагає додатково дослідження для випадку багатоцільової оптимізації. Адже ймовірно, що в ході пошуку компромісу між цільовими функціями, ефективність цього процесу значно знизиться зі зменшенням кількості змінних, що беруть участь в оптимізації.

### **3.3 Аналіз результатів багатоцільової оптимізації**

У ході третього сценарію симуляції були отримані результати, які мають підтвердити доцільність використання генетичного алгоритму NSGA-II для організації багатоцільової інтелектуальної міграції модулів додатків. Основним показником життєздатності такого підходу є отримання компромісного рішення між вартістю обчислень та часовою затримкою для третьої та четвертої стратегії. На рисунку 3.12 наведені результати чотирьох запусків для даного сценарію. Вісь абсцис відповідає значенням першої цільової функції – середньої часової затримки обробки даних в основному циклі додатку. На вісі ординат відкладено значення другої цільової функції – вартості обчислень за весь час моделювання, яка складається з вартості обчислень у хмарному середовищі та на кожній базовій станції. Як і було припущене, хмарна стратегія є оптимальною для першої цільової функції, а крайова – для першої. Із графіків можна зробити висновок, що організація багатоцільової міграції на основі генетичного алгоритму NSGA-II дозволяє підтримувати компроміс між загальною вартістю та величиною середньої затримки упродовж всього часу функціонування системи. Також наведено діаграму з частотою та об'ємом міграції для даного сценарію симуляції (рисунок 3.13).

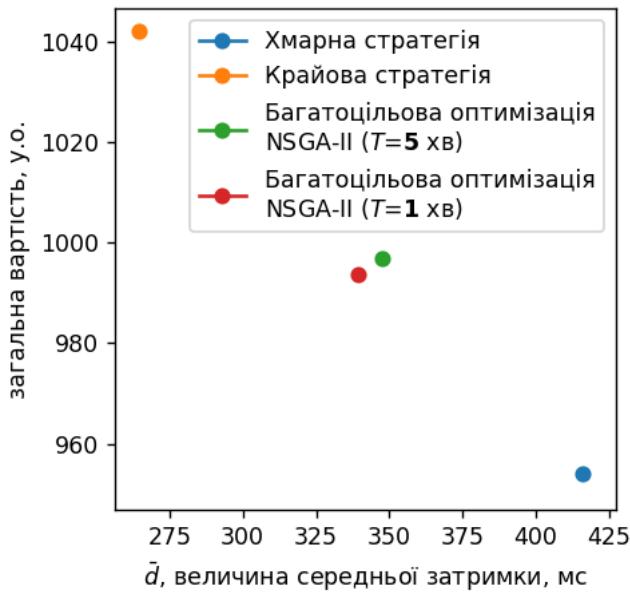


Рисунок 3.12 – Результати багатоцільової оптимізації

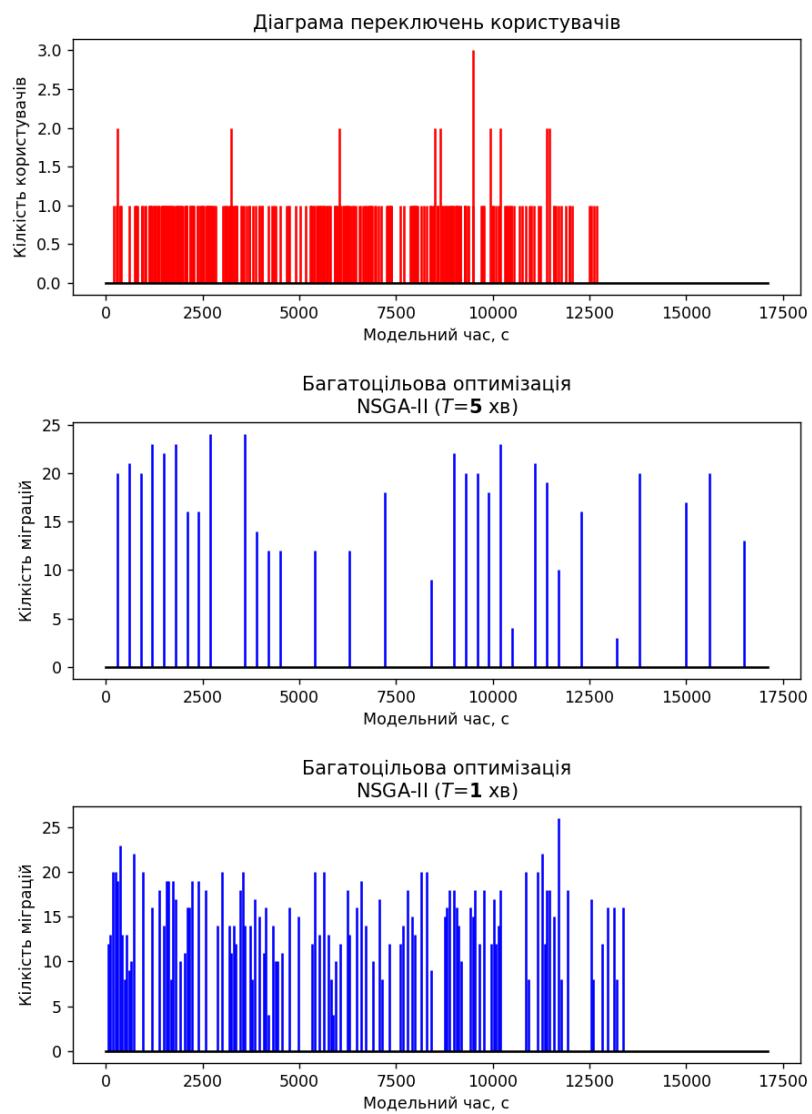


Рисунок 3.13 – Частота міграцій для стратегій багатоцільової оптимізації

Опосередкованими, але все ж вартими уваги, показниками є інші вихідні значення моделювання: кількість спожитої обчислювальними пристроями електроенергії та завантаженість мережі, яка прямо пропорційна об'єму даних, яку генерує система за одиницю часу. Разом із цільовими значеннями, усі вихідні дані наведені в таблиці 3.1.

Таблиця 3.1 – Вихідні дані третього сценарію симуляції

Стратегія	Середня затримка, мс	Загальна вартість, у.о.	Спожита електроенергія, Дж	Мережеве навантаження, кбайт/с
Хмарна	416.061	954.187	198,265,032	23.864
Крайова	264.561	1042.011	198,257,283	8.161
NSGA-II (період 5 хв)	347.641	996.972	198,257,960	19.844
NSGA-II (період 1 хв)	339.166	993.8	198,256,339	16.363

Виходячи із наведених в таблиці 3.1 даних можна зробити декілька висновків. По-перше, при порівнянні хмарної та крайової стратегій обробка даних близче до кінцевих користувачів може значно зменшити час затримки та навантаження на мережу, що характерно для парадигми туманних обчислень. По-друге, значення цільових функцій, які забезпечують стратегії на основі багатоцільової оптимізації, знаходяться на приблизно однаковій відстані від радикальних стратегій 1 та 2. Однак, неможна стверджувати подібне про значення мережевого навантаження, яке лише опосередковано враховується цільовою функцією часової затримки. Тому даний нецільовий параметр суттєво відрізняється для стратегій 3 та 4, але в будь-якому випадку є меншим за відповідне значення для хмарної і більшим за значення для крайової стратегій. Нарешті, значення спожитої енергії відрізняється несуттєво (в межах 0.005%).

Оскільки в побудованій моделі споживання електроенергії однакове для кожного ядра як хмарного ЦОД, так і обчислювального вузла базової станції та сервера лікарні, це свідчить про приблизно однакову загальну кількість виконаних задач для всіх користувачів. Незначне відхилення пов'язане з ймовірністю впливом запиту до модуля з електронною карткою пацієнта.

Загалом, можна зробити висновок, що використання стратегій на основі генетичного алгоритму NSGA-II для організації інтелектуальної багатокритеріальної міграції в адаптивній платформі є доцільним. Для подальшого покращення ефективності цих стратегій можна розглянути: впровадження паралельної реалізації генетичного алгоритму; розподіл системи на окремі області (кластери, регіони), кожен з яких автономно керує міграцією в межах своєї області; організувати адаптивну платформу у вигляді ієрархічної системи. Адже у даній роботі досліджено лише варіант централізованого управління міграцією. Таким чином, зазначені підходи дозволять будувати адаптивні платформи, враховуючи узгодженість між архітектурами додатків Інтернету речей та відповідними моделями систем забезпечення адаптивності, наведеними в першому розділі.

### **3.4 Висновки до розділу 3**

У даному розділі було наведено та проаналізовано дані, які отримано для кожного сценарію, описаного в розділі 2. Метою було дослідження та обґрунтування ефективності використання модифікацій та моделей застосування генетичного алгоритму при вирішенні задачі організації міграції модулів додатків.

Спершу було проаналізовано вплив інжекції ідеалізованих рішень на якість одно цільової оптимізації, виконаної генетичним алгоритмом. При збільшенні частки інжектованих рішень було продемонстровано зменшення середньої затримки для всіх варіантів кількості користувачів системи. Також в ході аналізу було обґрунтовано, що інжекція ідеалізованих рішень сприяє

швидшому отриманню прийнятних рішення, отже дозволяє зменшити необхідну кількість ітерацій та, за необхідності, розмір популяції. На основі цього було зроблено висновок про успішне застосування інжекції ідеалізованих рішень для рішення задачі генетичним алгоритмом принаймні у випадку одноцільової оптимізації.

По-друге, було проаналізовано ефективність генетичного алгоритму з точки зору характеру ініціації та кількості змінних системи, що підлягають оптимізації. Було продемонстровано, що оптимізація одночасно всіх змінних системи потребує багато часу та може надавати незадовільні результати. Натомість ефективніше проводити оптимізацію невеликої частини змінних системи, що дозволить отримати рішення, близчі до оптимального. Також було обґрунтовано доцільність поєднання періодичної (планової) ініціації з більшою кількістю змінних та реактивної ініціації з дуже маленькою: це дозволить підтримувати субоптимальний стан системи, витрачаючи менше часу на роботу алгоритму, та швидко досягти локальних покращень при маленьких динамічних змінах у системі. Ефективність зазначених підходів потребує додаткового дослідження для випадку задач багатокритеріальної оптимізації.

По-третє, наведено результати використання генетичного алгоритму для організації багатокритеріальної міграції для третього сценарію. В той час як хмарна та крайова стратегії розміщення модулів дозволяють мінімізувати одну з цільових функцій, стратегії на базі обраного генетичного алгоритму NSGA-II можуть знаходити і підтримувати компромісні значення для багатьох цільових функцій. Зроблено висновок про доцільність використання алгоритму для стратегій з метою організації багатокритеріальної міграції, а також наведено варіанти реалізації таких стратегій в розподілених системах із дотриманням узгодженості з архітектурою додатків Інтернету речей.

## 4 РОЗРОБКА СТАРТАП-ПРОЄКТУ «АДАПТИВНА ПЛАТФОРМА ДЛЯ МЕДИЧНИХ ДОДАТКІВ»

### 4.1 Опис ідеї проєкту

Поточний розділ присвячено розробці ідеї та плану впровадження стартап-проєкту, цільовим об'єктом якого є описана в роботі адаптивна платформа. Метою розділу є визначення технологічної та економічної перспективи реалізації інноваційної платформи, її конкурентної спроможності та плану просування продукту на ринку. Основну ідею стартап-проєкту «Адаптивна платформа для медичних додатків» у з інноваційною складовою наведено у таблиці 4.1.

Таблиця 4.1 – Опис ідеї стартап-проєкту

<b>Зміст ідеї проєкту</b>	<b>Напрямки застосування продукту</b>	<b>Вигоди для користувача продукту</b>
Створення на основі обчислювальних ресурсів базових станцій адаптивної платформи з механізмом міграції для розгортання медичних додатків	Запуск власних та сторонніх медичних додатків з вимогами до низької затримки	Забезпечення низької затримки обробки даних шляхом втілення концепції туманних обчислень
	Запуск власних та сторонніх медичних додатків в умовах мобільності користувачів	Забезпечення відповідного рівня якості послуг для мобільних користувачів додатку

Таким чином, основними користувачами системи є власники медичних інформаційних систем, серед яких потенційно приватні та громадські лікарні або інші медичні заклади, специфіка додатків яких припускає суттєві вимоги до

затримки обробки даних або небажаного мережевого навантаження в умовах мобільності користувачів.

Слабкі і сильні сторони ідеї стартап-проекту сформульовано в таблиці 4.2, а числові значення 1 до 5 означають оцінки відповідних параметрів. Сильні сторони власної реалізації адаптивної платформи є здатність забезпечити низькі затримки для встановлених додатків та інтелектуальну міграцію модулів додатку. Однак, обрана реалізація обмежена в апаратних ресурсах .

Таблиця 4.2 – Слабкі та сильні сторони ідеї «Адаптивної платформи»

№	Техніко-економічні характеристики ідеї	Потенційні платформи конкурентів				Слабка сторона W	Нейтральна сторона N	Сильна сторона S
		Власна плат-ма	K-т 1	K-т 2	K-т 3			
1	Здатність забезпечити низьку затримку	5	2	1	3			+
2	Наявність адаптивної міграції	5	3	1	1			+
3	Масштабованистъ	3	5	3	3		+	
4	Обмежені апаратні ресурси	2	5	4	3	+		
5	Необх-ть заключення сторонніх договорів	3	4	4	4		+	

## 4.2 Технологічний аудит ідеї проєкту

Підставою для реалізації адаптивної платформи є аудит сучасних представлених на ринку технологій. Його результати наведено в таблиці 4.3. Всі наведені технології є безкоштовними та доступними, але з огляду на критерії ефективності та зручності було обрано найбільш продуктивний варіант реалізації, зважаючи на відносну простоту алгоритму.

Таблиця 4.3 – Технологічна здійсненність ідеї стартап-проєкту

<b>№</b>	<b>Ідея проєкту</b>	<b>Технології реалізації</b>	<b>Наявність технології</b>	<b>Доступність технології</b>
1	Організація інтелектуальної міграції в адаптивній платформі	Високопродуктивна, низькорівнева мова C, власна реалізація алгоритму	Наявна	Безкоштовна, доступна
		Середньопродуктивна, високорівнева мова Java, MOEA Framework	Наявна	Безкоштовна, доступна
		Низькопродуктивна, високорівнева мова Python, DEAP Framework	Наявна	Безкоштовна, доступна
Для реалізації інтелектуального компоненту міграції обрано мову C та власну реалізацію алгоритму через ключову властивість високої продуктивності у порівнянні з альтернативами.				

## 4.3 Аналіз ринкових можливостей

Для успішного просування розробленого продукту в умовах ринку слід визначити можливі напрямки розвитку стартап-проєкту у контексті ринкового середовища, сформулювати вимоги від потенційних користувачів системи та

пропозиції наявних конкуруючих проектів. Попередній аналіз попиту на ринку адаптивних платформ наведено у таблиці 4.4. Останнє значення середньої рентабельності у галузі ринку майже вдвічі перевищує поточну процентну ставку на депозит, що свідчить про привабливість входження на ринок.

Таблиця 4.4 – Попередня характеристика потенційного ринку проєкту

<b>№</b>	<b>Показники стану ринку</b>	<b>Характеристика</b>
1	Кількість головних гравців, од	3
2	Загальний обсяг продажів, грн./у.о. на рік	500 тис. грн./у.о. на рік
3	Якісна оцінка динаміки ринку	Зростає
4	Наявність обмежень для входу на ринок	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Конфіденційність даних відповідно до вимог додатків
6	Середня норма рентабельності по ринку, %	$R = \frac{500 \cdot 10^3 \cdot 4}{10 \cdot 10^6} \cdot 100\% = 20\%$

Проведено аналіз потенційних користувачів адаптивної платформи, а його результати представлено в таблиці 4.5.

Таблиця 4.5 – Характеристика потенційних користувачів платформи

<b>№</b>	<b>Потреба, що формує ринок</b>	<b>Цільова аудиторія користувачів</b>	<b>Відмінності у поведінці потенційних груп користувачів</b>	<b>Вимоги клієнтів до продукту</b>
1	Адаптивна платформа з підтримкою міграції модулів додатків	Власники чутливих до часових затримок медичних додатків	Додатки державних лікарень мають більшу кількість користувачів, аніж приватні клініки. Натомість останні склонні до інновацій.	Низькі затримки обробки даних, врахування мобільності клієнтів

Відмінності у поведінці груп є важливим аспектом, з якого можна зробити висновок про доцільність співпраці з приватними лікарнями на перших етапах впровадження проєкту.

У таблиці 4.6 наведено результати аналізу загроз для стартап-проєкту при його виході на ринок платформ для медичних додатків. Серед ризиків виділено як стандартні, як наприклад конкуренція та складність у пошуку перших користувачів, так і специфічні для проєкту, як-от брак обчислювальних потужностей або необхідність внесення змін у механізм організації міграції.

Таблиця 4.6 – Фактори загроз для стартап-проєкту

<b>№</b>	<b>Фактор</b>	<b>Зміст загрози</b>	<b>Можлива реакція</b>
1	Брак апаратних ресурсів	Відсутність або недостатня кількість обчислювальних ресурсів на базових станціях операторів зв'язку	Розгортання власних мікро-ЦОД поблизу базових станцій
2	Відсутність перших користувачів платформи	На перших етапах впровадження можливі труднощі у пошуку приватних клінік з медичними додатками з метою партнерства	Розробка власного або спільного з однією чи декількома лікарнями тестового додатку
3	Зміна потреб окремих користувачів	Виникнення потреби врахування нових умов забезпечення якості послуг або критеріїв адаптивності	Закладання в адаптивну платформу можливості розширення
4	Конкуренція	Випуск конкуруючої платформи великою компанією	Використання інфраструктури великої компанії для подальшого розвитку платформи

Важливим аспектом для врахування потенційних можливостей стартап-проєкту на ринку є конкурентність середовища. Тому в таблиці 4.7 наведено результати аналізу наявної на ринку конкуренції.

Таблиця 4.7 – Ступеневий аналіз конкуренції на ринку

<b>Хар-ка конкурентного середовища</b>	<b>Особливість конкурентного середовища</b>	<b>Прояв характеристики</b>	<b>Вплив на діяльність стартап-проєкту</b>
Тип	Чиста конкуренція	Через специфіку використання локальних ресурсів проект відокремлений від гігантів ІТ індустрії	Якомога скоріше укласти договори з потенційними партнерами, захопивши лідерство у сфері
Рівень конкурентної боротьби	Локальний	Платформа задіює аппаратні ресурси базових станцій, тому конкуренція відбувається в межах конкретного міста	При успішному запуску платформи в першому місті слід швидко пошири свій вплив у інших містах
Галузева ознака	Змішана	Існують властивості і міжгалузевої (сучасні підходи AI/ML, прогрес бездротових 5G мереж), і внутрішньогалузевої	Спостерігати за інноваціями в суміжних галузях та впроваджувати новітні підходи в платформу

<b>Хар-ка конкурентного середовища</b>	<b>Особливість конкурентного середовища</b>	<b>Прояв характеристики</b>	<b>Вплив на діяльність стартап-проєкту</b>
За видом товарів	Товарно-видова	Спільний вид: надання послуг розміщення додатків на адаптивній платформі в умові мобільності користувачів	Опосередкована конкуренція з потужнім IT компаніями-постачальниками хмарних послуг
Характер конкурентних переваг	Змішана	Чинять вплив як вартість розміщення додатків, так і якість обслуговування, яку може забезпечити платформа	Конкурувати доведеться у декількох аспектах
Інтенсивність	Не марочна	Відсутність представлення аналогічних послуг від відомих IT-компаній	Першими захопивши перевагу цілком можливо конкурувати з платформами-аналогами

Виходячи із таблиці 4.7 було проведено аналіз конкуренції за Портером. Його результати із відповідними висновками представлені в таблиці 4.8. Загалом поточної конкуренція на ринку відсутня, але існує ряд важливих факторів і залежностей від партнерів (операторів зв'язку, на базових станціях яких розгортається платформа) та постачальників хмарних послуг, які не слід ігнорувати.

Таблиця 4.8 – Аналіз конкуренції за Портером

<b>Складові аналізу</b>	<b>Прямі конкуренти в галузі</b>	<b>Потенційні конкуренти</b>	<b>Постачальники</b>	<b>Клієнти</b>	<b>Послуги-замінники</b>
	Відсутні	Великі ІТ-компанії	Оператори зв'язку	Власники медичних додатків	Хмарні послуги
Висновки	На даний момент прямі конкуренти для цільового міста відсутні, що робить можливим захоплення початкової переваги	Вихід на ринок великих ІТ-компаній в межах цільового міста є мало-ймовірним, хоча й не виключається.	В даному контексті постачальниками можуть вважатися оператори мобільного зв'язку: власники базових станцій, від яких залежить успішність втілення проєкту	Цільовими клієнтами є власники медичних додатків: приватні та державні лікарні: їхніми вимогами є якість забезпечення високої якості обслуговування	Із розвитком мобільних мереж постачальники хмарних послуг зможуть забезпечити прийнятний рівень затримки.

Обґрунтування факторів конкурентоспроможності стартап-проєкту адаптивної платформи (таблиця 4.9) складено на основі характеристик ідеї проєкту (таблиця 4.2) та аналізу конкуренції в галузі (таблиці 4.7 та 4.8).

Таблиця 4.9 – Обґрунтування факторів конкурентоспроможності

<b>№</b>	<b>Фактор конкурентоспроможності</b>	<b>Обґрунтування фактору конкурентоспроможності</b>
1	Здатність забезпечити низьку затримку обчислень	У порівнянні із сучасними хмарними платформами, адаптивна туманна платформа здатна зменшити часову затримку шляхом наближення обчислень до кінцевого користувача
2	Наявність адаптивної міграції	Здатність підтримувати відповідний рівень часової затримки та інших вимог до якості обслуговування в умовах мобільності користувачів додатків, розміщених на платформі

Наступним етапом є порівняння слабких та сильних сторін стартап-проекту у порівнянні з потенційними конкурентами на основі даних таблиці 4.2. Відповідні результати наведені у таблиці 4.10.

Таблиця 4.10 – Порівняльний аналіз слабких і сильних сторін

<b>№</b>	<b>Фактор конкурентоспроможності</b>	<b>Бали</b> <b>1-5</b>	<b>Рейтинг конкурентних платформ</b>						
			-3	-2	-1	0	+1	+2	+3
1	Низька затримка	5	+						
2	Адаптивна міграція	5	+						
3	Обмеженість ресурсів	2							+

Стандартним у плануванні стартап-проекту і кінцевим в дослідженні його ринкового потенціалу етапом є його SWOT-аналіз (таблиця 4.11). У вигляді матриці для стартап проекту адаптивної платформи розміщення медичних додатків подано сильні та слабкі сторони у поєднанні із загрозами та можливостями у ході реалізації проекту.

Таблиця 4.11 – SWOT-аналіз проєкту адаптивної платформи

<b>S</b>	Здатність забезпечити низьку затримку для вимогливих медичних додатків в умовах мобільності їхніх користувачів	Обмеженість апаратних ресурсів, розміщених на базових станціях операторів зв’язку	<b>W</b>
<b>O</b>	Співпраця як зі схильними до інноваційних рішень приватними лікарнями; залучення державної підтримки у співпраці з державними лікарнями; створення власного медичного додатку як частини MVP проєкту	Складність укладання угод із власниками базових станцій; необхідність розміщення додаткового обчислювального обладнання на базових станціях	<b>T</b>

З огляду на виконаний SWOT-аналіз було розроблено альтернативні стратегії виходу адаптивної платформи на ринок, які відрізняються темпами розвитку (таблиця 4.12). Більш привабливим є стратегія помірного розвитку.

Таблиця 4.12 – Альтернативні стратегії виходу платформи на ринок

№	Альтернативний комплекс заходів ринкової поведінки	Ймовірність отримання ресурсів	Сроки реалізації проєкту
1	Стрімкий розвиток: укладання договорів з кількома операторами зв’язку, кількома приватними фірмами	41%	10 місяців
2	Помірний розвиток: укладання договору з єдиним оператором зв’язку, залучення однієї клініки	64%	7 місяців

#### 4.4 Розробка ринкової стратегії стартап-проекту

Для детального аналізу обраної ринкової стратегії необхідно визначити цільові групи користувачів платформи. Відповідні дані наведено у таблиці 4.13.Хоча приватні лікарні є більш привабливими цільовими користувачами, державні медичні заклади також заслуговують уваги.

Таблиця 4.13 – Цільові групи потенційних користувачів системи

№	Опис профілю цільової групи потенційних користувачів	Готовність споживачів сприйняти продукт	Очікуваний попит в цільовій групі	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Приватні лікарні	Вище середньої: приватний бізнес більш інноваційний	Середній: через бажання забезпечити вищий рівень послуг	Низька: відсутність явних конкурентів	Висока: немає труднощів в укладання договорів
2	Державні лікарні	Середня: характерна менша ступінь інновації	Середній: через високу кількість пацієнтів	Низька: відсутність явних конкурентів	Середня: вимагає укладення договорів з держ. установами

Наступним кроком є конкретизація базової стратегії розвитку, яка вимагає визначення стратегії охоплення ринку, зазначення ключових аспектів конкурентоспроможності проекту та відповідної базової стратегії розвитку (таблиця 4.14). Через високу утилітарність адаптивної платформи найбільш

перспективною стратегією є диверсифікація, яка супроводжується співпрацею з підприємствами в галузях, відмінних від медичної.

Таблиця 4.14 – Визначення базової стратегії розвитку стартап-проекту

<b>№</b>	<b>Обрана альтернатива розвитку проекту</b>	<b>Стратегія охоплення цільового ринку</b>	<b>Ключові конкуренто-спроможні пропозиції</b>	<b>Базова стратегія розвитку</b>
1	Помірний розвиток	Стратегія розвитку ринку	Поступове нарощування користувальської бази клінік, з якими проводиться співпраця	Стратегія концентрованого розвитку
2	Помірний розвиток	Стратегія концентричної диверсифікації	Розширення сфери використання використання розробленої платформи для інших галузей підприємництва.	Стратегія диверсифікованого росту

Визначеню підлягає базової стратегії конкурентної поведінки стартап-проекту адаптивної платформи (таблиця 4.15). Обрано домінуючу стратегію: скориставшись відсутністю конкурентів захопити велику частку ринку.

Таблиця 4.15 – Базова стратегія конкурентної поведінки

<b>№</b>	<b>Чи є проект першо-проходцем</b>	<b>Пошук нових користувачів</b>	<b>Копіювання характеристик у конкурентів</b>	<b>Стратегія конкурентної поведінки</b>
1	Так	Помірний пошук та боротьба за особливо привабливих	Так: за появи конкурентів та вдалих підходів	Домінуюча стратегія

Останнім етапом у розробці ринкової стратегії для стартап-проекту є визначення стратегії позиціонування продукту, наведеної в таблиці 4.16. За ним можна сформулювати основні асоціації, за якими здійснюватиметься позиціонування адаптивної платформи на ринку. Основними асоціаціями стануть можливість забезпечення низької затримки обробки даних та її забезпечення в умовах мобільності кінцевих користувачів додатків, розміщених на платформі.

Таблиця 4.16 – Стратегія позиціонування стартап-проекту

<b>№</b>	<b>Вимоги до продукту від користувачів</b>	<b>Базова стратегія розвитку</b>	<b>Ключові конкуренто-спроможні пропозиції</b>	<b>Асоціації для комплексної позиції стартап-проекту</b>
1	Дотримання часової затримки в межах визначеної якості послуг	Стратегія диверсифікованого росту	Розширення сфери використання використання розробленої платформи для інших галузей підприємництва.	Низька затримка, врахування мобільності користувачів

## 4.5 Розробка маркетингової програми стартап-проекту

Для визначення маркетингової програми стартап-проекту спершу необхідно сформулювати маркетингову концепцію товару (таблиця 4.17) на основі попереднього аналізу, разом з його перевагами перед потенційними конкурентами: постачальниками хмарних платформ як AWS, Microsoft Azure, Heroku та потенційних місцевих постачальників послуг з обробки даних.

Таблиця 4.17 – Ключові переваги концепції потенційного товару

<b>№</b>	<b>Потреба</b>	<b>Вигода продукту</b>	<b>Ключові переваги</b>
1	Низька затримка	Розміщення платформи на обчислювальних пристроях на базових станціях операторів мобільного зв'язку	Застосування концепції туманних обчислень для забезпечення низької затримки
2	Врахування мобільності користувачів	Реалізація механізму інтелектуальної міграції	Використання генетичного алгоритму для оптимізації розташування модулів додатків

На підставі результатів аналізу побудовано трирівневу маркетингову модель товару в таблиці 4.18. Також необхідно зазначити, що спосіб захисту продукту полягає в оформленні відповідного патенту.

Таблиця 4.18 – Трирівнева маркетингова модель товару

<b>Рівні товару</b>	<b>Сутність та складові</b>		
1. Товар за задумом	Адаптивна платформа для розміщення медичних додатків з низькою затримкою в умовах мобільності користувачів		
2. Товар у реальному виконанні	Використання туманних обчислень для низької затримки	Hm	Tx
	Застосування інтелектуальної міграції	Hm	Tx
	Безпека оброблюваних даних	Hm	Bp
	Якість: згідно зі стандартами IEEE-829 чи ISO/IEC 9126		
3. Товар із підкріпленням	Безкоштовний місяць користування платформою з обмеженим функціоналом		
	Платний тариф з повною підтримкою		

У таблиці 4.19 наведено результати аналізу ціноутворення для стартап-проєкту, що визначає орієнтовані межі цін за послуги платформи та ґрунтуються на цінах інфраструктурних послуг AWS. Разом із цим, у таблиці 4.20 наведено інформацію щодо запланованої системи збути продукції у вигляді послуг платформи. Основним способом придбання продукту є купівля місячної підписки на користування платформою на сайті компанії.

Таблиця 4.19 – Встановлення цінових меж за послуги

№	Рівень цін на послуги-замінники, грн./день	Рівень цін на послуги-аналоги, грн./день	Рівень доходів користувачів платформи, грн./день	Верхня та нижня межі встановлення ціни на послугу, грн./день
1	750	1250	3000	700-2500

Таблиця 4.20 – Система збути послуг

№	Специфіка закупівельної поведінки користувачів	Функції постачальника щодо збути	Глибина каналів збути	Оптимальна система збути
1	Придбання місячної підписки на користування платформою	Продаж, консультація користувачів, оцінка вигоди для користувачів	0 (виключно на сайті компанії)	Стандартна для галузі надання інформаційних послуг

Завершальною частиною маркетингової програми стартап-проєкту адаптивної платформи є концепція маркетингових комунікацій (таблиця 4.21), в якій в якості основного каналу комунікацій вказано Інтернет та аспекти рекламиування послуг.

Таблиця 4.21 – Концепція маркетингових комунікацій

№	Специфіка закупівельної поведінки користувачів	Канали комунікацій з клієнтами	Ключові аспекти позиціонування	Завдання рекламного повідомлення	Рекламне звернення
1	Придбання місячної підписки на користування платформою	Сайт компанії в Інтернеті	Низька затримка розміщених додатків, врахування мобільності користувачів	Зробити наголос на перевагах платформи в порівнянні з хмарними сервісами	Наведення на сайті інформації щодо варіантів використання платформи

## 4.6 Висновки до розділу 4

У даному розділі було обґрутовано доцільність реалізації досліджуваної адаптивної платформи для медичних додатків у вигляді стартап-проекту. Було сформульовано основну ідею проекту, оцінено можливість її технічної реалізації.

Із зазначених сильних та слабких сторін проєкту у ході порівняння з потенційними конкурентами зроблено висновок про конкурентоспроможність платформи та ключові аспекти її забезпечення. На основі проведеного аналізу здійснено прогнозування розвитку проєкту в ринковому середовищі, визначено потенційні загрози та можливі заходи для їх усунення. Загальні перспективи стартап-проекту адаптивної платформи було сформульовано в результаті проведення SWOT-аналізу. Було визначено та оцінено основні сегменти користувачів продукту, на основі чого розроблено ринкову та маркетингову стратегії просування проєкту.

## ВИСНОВКИ

У ході роботи було досліджено спосіб організації інтелектуальної міграції модулів додатків як частину та механізм забезпечення адаптивності туманної платформи для запуску медичних додатків. Було запропоновано розгляdatи задачу організації інтелектуальної міграції модулів додатків у вигляді циклічного рішення задачі оптимального розташування модулів на вузлах обчислення. З огляду на успішне вирішення останньої задачі генетичним алгоритмом NSGA-II в одному з літературних джерел, цей алгоритм було обрано як основу механізму адаптивності.

На основі сучасних концепцій і технологій в галузі комп’ютерних наук та з огляду на дослідження схожих систем в літературі було синтезовано узагальнену модель адаптивної туманної платформи із трьох рівнів: на першому інфраструктурному рівні розміщені обчислювальні та мережеві ресурси; на другому рівні проміжного програмного забезпечення реалізовано механізм міграції віртуального середовища та/або міграції контексту користувача, а також систему інтелектуального керування цим процесом; на третьому рівні розміщені прикладні додатки (на сьогоднішній день з мікросервісною архітектурою, в перспективі – в парадигмі безсерверних обчислень).

Для дослідження моделі адаптивної туманної платформи було створено власний симулятор на основі iFogSim, який дозволяє моделювати переміщення мобільних користувачів, їхнє динамічне підключення та відключення від базових станцій мобільного зв’язку; міграцію модулів додатку між обчислювальними пристроями.

В межах розробленого середовища симуляції було визначено модель адаптивної туманної платформи. Цільовим прикладним додатком, який підлягає запуску в цій платформі, було обрано додаток з моніторингу серцево-судинних захворювань на базі датчиків ЕКГ, SpO<sub>2</sub>, ЧСС, тиску, холестерину та глюкози. Було підготовлено вхідні дані для подальшого моделювання:

маршрути жителів міста на основі сценарію Монако для інструментарію SUMO; розміщення базових станцій 3G та 4G зв'язку на міських та приміських територіях; мережеві параметри пропускної здатності та затримки на основі існуючих мережевих протоколів; параметри обчислювальних пристройів (смартфонів користувачів, обчислювальних вузлів на базових станціях, серверу міської лікарні). Таким чином була здійснена спроба найбільшою мірою наблизитись до реальних умов розгортання платформи.

Було запропоновано, досліджено та за результатами моделювання підтверджено ефективність підходу інжекції ідеалізованих рішень в початкову популяцію алгоритму. Віддавши перевагу заповненню початкової популяції рішеннями, які були б оптимальними за ідеальних умов, аніж випадковими, можна очікувати покращення абсолютних значень цільової функції. Крім цього, модифікація дозволяє пришвидшити знаходження прийнятних рішень, витрачаючи значно меншу кількість ітерацій та пришвидшуючи роботу генетичного алгоритму. Також, у порівнянні з випадковою ініціалізацією популяції, при ініціалізації з інжекцією значно зростає ефект від збільшення розміру популяції. Таким чином, цілком виправдано встановлення значення параметру частки інжектованих рішень на 100% у випадку одноцільової оптимізації, але додаткового дослідження потребує вплив коефіцієнту впевненості на якість отримуваних рішень та поведінка модифікації у випадку багатокритеріальної оптимізації.

Наступним етапом було розділено організацію міграції за характером ініціації процесу оптимізації на періодичну, або в ширшому значенні заплановану, та реактивну, в сенсі реакції на зміни в динамічній системі. У цьому контексті було запропоновано розглядати підтримання оптимального розташування модулів з двох точок зору: глобальна оптимізація для значної частини змінних системи, яка має проводитись заплановано, наприклад періодично через визначені проміжки часу; локальна оптимізація для незначної частини змінних системи, яка може відбуватися швидко, носити реактивний характер на невеликі зміни в динамічній системі та призводити до невеликого

локального покращення стану системи. Доцільність такої концепції було обґрунтовано з огляду на отримані в процесі моделювання експериментальні дані.

Також, в межах другого сценарію було висунуто припущення, що в умовах обмеженої кількості ітерацій генетичного алгоритму рішення отримане шляхом оптимізації всіх наявних змінних може виявиться менш якісним, аніж таке рішення, що отримане оптимізацією для певних підмножин змінних окремо. Останнє підтвердження було підтверджено в межах другого сценарію експерименту, коли значення цільової функції при оптимізації всіх 180 змінних виявилося суттєво гіршим, аніж при поступовій оптимізаціїожної підмножини зі всього 30 змінними.

Нарешті, з урахуванням попередніх результатів, було створено модель механізму організації інтелектуальної багатоцільової міграції в адаптивній туманній платформі. По-перше, було використано інжекцію ідеалізованих рішень в початкову популяцію генетичного алгоритму. По-друге, обрано періодичний характер ініціації процесу оптимізації з періодом в 1 та 5 хвилин модельного часу. По-третє, всю множину змінних системи було розбито на підмножини (кластери) з 30 змінних, щоб в єдиний момент часу генетичний алгоритм здійснював оптимізацію лише однієї підмножини всіх змінних. Таким чином, із застосуванням стратегії з багатокритеріальним алгоритмом NSGA-II було отримано компромісне рішення між хмарною та крайовою стратегіями для випадку мінімізації визначених двох цільових функцій: середньої затримки обробки даних користувачів та загальної вартості обчислень.

Наочанок, представлені міркування щодо потенційної реалізації механізму на основі NSGA-II у формі системи з регіональним плануванням або ієрархічної системи для узгодження з додатками, що мають архітектуру відмінну від централізованої, наприклад колаборативну або розподілену.

## ПЕРЕЛІК ПОСИЛАНЬ

1. N. Hassan. Edge Computing in 5G: A Review / N. Hassan, K. A. Yau, C. Wu // IEEE Access, серпень 2019. – V.7. – P. 127276-127289.
2. C. Puliafito. Container Migration in the Fog: A Performance Evaluation / C. Puliafito, C. Vallati, E. Mingozzi, G. Merlino, F. Longo, A. Puliafito // Sensors березень 2019.
3. C. Puliafito. MobFogSim: Simulation of mobility and migration for fog computing / C. Puliafito, D. Gonçalves, M. Lopes, L. Martins, E. Madeira, E. Mingozzi, O. Rana, L.F. Bittencourt // Simulation Modelling Practice and Theory, травень 2020. – V. 101.
4. L. F. Bittencourt. Mobility-Aware Application Scheduling in Fog Computing / L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, M. Parashar // IEEE Cloud Computing, березень 2017, Piscataway, NJ, USA. – V. 4. – P.26-35.
5. M. Narges. MAPO: A Multi-Objective Model for IoT Application Placement in a Fog Environment / M. Narges, D. Kimovski, R. Prodan // IoT 2019: Proceedings of the 9th International Conference on the Internet of Things, жовтень 2019, Bilbao, Spain. – P.1-8.
6. L. Sabatucci. The Four Types of Self-Adaptive Systems: A Metamodel / L. Sabatucci, V. Seidita, M. Cossentino // Smart Innovation, System and Technologies, травень 2018, Cham, Switzerland. – P.440-450.
7. N. Tague. The quality toolbox / Nancy Tague. – Milwaukee: ASQ Quality Press, 2005. – 545 с. – (2).
8. Plan, Do, Study, Act (PDSA) cycles and the model for improvement. – Режим доступу: <https://improvement.nhs.uk/documents/2142/plan-do-study-act.pdf>. – Дата доступу: 02.03.2021.
9. The Essence of Winning and Losing. – Режим доступу: [https://fasttransients.files.wordpress.com/2010/03/essence\\_of\\_winning\\_losing.pdf](https://fasttransients.files.wordpress.com/2010/03/essence_of_winning_losing.pdf). – Дата доступу: 02.03.2021.

10. D. Ullman. "OO-OO-OO!" The sound of a broken OODA loop / D. Ullman // CrossTalk, січень 2006. – V.10.
11. M. Aldinucci. Managing Adaptivity in Parallel Systems. / M. Aldinucci, M.Danelutto, P. Kilpatrick, C. Montangero, L. Semini // Formal Methods for Components and Objects. FMCO 2011. Lecture Notes in Computer Science, січень 2013, Springer, Heidelberg. – P.1-20.
12. H. Muccini. Self-adaptive IoT architectures: an emergency handling case study / H. Muccini, R. Spalazzese, M. Moghaddam, M. Sharaf // Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings, вересень 2018, Madrid, Spain. – P.1-6.
13. D. Weyns. On Patterns for Decentralized Control in Self-Adaptive Systems / D. Weyns, B. Schmerl, V. Grassi, S. Malek, R. Mirandola, C. Prehofer, J. Wuttke, J. Andersson, H. Giese, K. Göschka // Software Engineering for Self-Adaptive Systems II. Lecture Notes in Computer Science, жовтень 2010, Springer, Heidelberg. – P.76-107.
14. IEC White Paper: Edge intelligence – en, ko – IEC Basecamp – Режим доступу: <https://basecamp.iec.ch/download/iec-white-paper-edge-intelligence-en>. – Дата звернення: 23.04.2021 р.
15. K. Dolui. Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing / K. Dolui, S. K. Datta // 2017 Global Internet of Things Summit (GIoTS), червень 2017, Geneva, Switzerland. – P.1-6.
16. H. Gupta. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments / H. Gupta, A.V. Dastjerdi, S. Ghosh, R. Buyya // Software: Practice and Experience, жовтень 2016. – V. 47.
17. O. Skarlat. Towards QoS-Aware Fog Service Placement / O. Skarlat, M. Nardelli, S. Schulte, S. Dustdar // 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), травень 2017, Madrid, Spain. – P. 89-96.

18. A Priori Methods in Multiobjective Optimization. Markus Hartikainen, PhD.  
 Режим доступу: [http://users.jyu.fi/~jhaka/uppsala/Lecture4\\_apriorimethods.pdf](http://users.jyu.fi/~jhaka/uppsala/Lecture4_apriorimethods.pdf).  
 – Дата доступу: 24.04.2021.
19. S. Daya. Microservices from Theory to Practice: Creating Applications in IBM Bluemix Using the Microservices Approach / S. Daya, N. V. Duy, K. Eati, C. M. Ferreira, D. Glozic, V. Gucer, M. Gupta, S. Joshi, V. Lampkin, M. Martins, S. Narain, R. Vennam. – An IBM Redbooks publication, 2015. – 176 с.
20. C. Fan. Microservices vs Serverless: A Performance Comparison on a Cloud-native Web Application / C. Fan, A. Jindal, M. Gerndt // 10th International Conference on Cloud Computing and Services Science, січень 2020, Prague, Czech Republic. – P. 204-215
21. C. Li. The IoT-based heart disease monitoring system for pervasive healthcare service / C. Li, X. Hu, L. Zhang // International Conference on Knowledge Based and Intelligent Information and Engineering Systems, вересень 2017, Marseille, France. – V. 112. – P. 2328-2334.
22. L. Codeca. Luxembourg SUMO Traffic (LuST) Scenario: Traffic Demand Evaluation / L. Codeca, R. Frank, S. Faye, T. Engel // IEEE Intelligent Transportation Systems Magazine, квітень 2017. – V. 9. – P. 52-63.
23. P. A. Lopez. Microscopic Traffic Simulation using SUMO / P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. -P. Flötteröd, R. Hilbrich, L. Lücke, J. Rummel, P. Wagner, E. Wiessner // 21st International Conference on Intelligent Transportation Systems (ITSC), листопад 2018, Maui, HI, USA. – P. 2575-2582.
24. Luxembourg SUMO Traffic (LuST) Scenario. – Режим доступу: <https://github.com/lcodeca/LuSTScenario>. – Дата доступу: 26.04.2021.
25. L. Codeca. Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS / L. Codeca, J. Härry // SUMO 2018 - Simulating Autonomous and Intermodal Transport Systems, травень 2018, Berlin, Germany. – V. 2. – P.43-55.

26. Orange (France) - Cellular Coverage and Tower Map. – Режим доступу: <https://www.cellmapper.net/>. – Дата доступу: 27.04.2021.
27. Orange Mobile 3G / 4G / 5G coverage map, France. – Режим доступу: <https://www.nperf.com/en/map/FR/-/21.Orange-Mobile/signal/?ll=43.72531090909004&lg=7.397987369913609&zoom=12>. – Дата доступу: 27.04.2021.
28. Mobile Phone Base Stations, How do mobile base stations work, Mobile Base Stations in Australia, Cell Tower, Mobile Phone Tower. – Режим доступу: [https://mobilenetworkguide.com.au/mobile\\_base\\_stations.html](https://mobilenetworkguide.com.au/mobile_base_stations.html). – Дата доступу: 27.04.2021.
29. Maximizing BLE Throughput on iOS and Android - Punch Through. – Режим доступу: <https://punchthrough.com/maximizing-ble-throughput-on-ios-and-android/>. – Дата доступу: 27.04.2021.
30. I. Grigorik. High Performance Browser Networking: What every web developer should know about networking and web performance / I. Grigorik. – O'Reilly Media, 2013. – 400 с.
31. 5G Latency – 1ms – Is It Possible | LinkedIn. – Режим доступу: <https://www.linkedin.com/pulse/5g-latency-1ms-possible-akshay-mahajan/>. – Дата доступу: 27.04.2021.
32. Bandwidth Transport, Optimization and Protection for Wireless Backhaul: Meeting 3G & 4G service delivery challenges. Exar Corporation. – Режим доступу: <https://www.maxlinear.com/Files/Documents/nhwp101609.pdf>. – Дата доступу: 28.04.2021.
33. Cisco - Inverse Multiplexing for ATM Trunk Module. – Режим доступу: [http://storage.library.opu.ua/online/external/cisco/products/790/mgx8220/imatm\\_ds.htm](http://storage.library.opu.ua/online/external/cisco/products/790/mgx8220/imatm_ds.htm). – Дата доступу: 28.04.2021.
34. A. Mahimkar. Bandwidth on demand for inter-data center communication / A. Mahimkar, A. Chiu, R. Doverspike, M. Feuer, P. Magill, E. Mavrogiorgis, J. Pastor, S. L. Woodward, J. Yates // Proceedings of the 10th ACM Workshop on Hot Topics in Networks, листопад 2011, Cambridge, MA, USA. – P. 1-6.

35. Cortex-A55 – Arm. . – Режим доступу: <https://www.arm.com/products/silicon-ip-cpu/cortex-a/cortex-a55>. – Дата доступу: 29.04.2021.
36. Cortex-Ax vs performance | [www.bitkistl.com](http://www.bitkistl.com/2015/03/cortex-ax-vs-performnace.html). – Режим доступу: [https://www.bitkistl.com/2015/03/cortex-ax-vs-performnace.html](http://www.bitkistl.com/2015/03/cortex-ax-vs-performnace.html). – Дата доступу: 29.04.2021.
37. Energy Efficient Processors – ARM big.LITTLE Technology. – Режим доступу: [http://www.ziti.uni-heidelberg.de/ziti/uploads/ce\\_group/seminar/2015-Philipp\\_Gsching.pdf](http://www.ziti.uni-heidelberg.de/ziti/uploads/ce_group/seminar/2015-Philipp_Gsching.pdf). – Дата доступу: 29.04.2021.
38. Arm Cortex-A55: high efficiency, sustained performance - Processors blog - Processors - Arm Community. – Режим доступу: <https://community.arm.com/developer/ip-products/processors/b/processors-ip-blog/posts/arm-cortex-a55-efficient-performance-from-edge-to-cloud>. – Дата доступу: 29.04.2021.
39. Core i7-5960X Extreme Edition Review: Intel's Overdue Desktop 8-Core Is Here. – Режим доступу: <https://techgage.com/print/core-i7-5960x-extreme-edition-review-intels-overdue-desktop-8-core-is-here>. – Дата доступу: 29.04.2021.
40. AM64x Benchmarks. – Режим доступу: <https://www.ti.com/lit/an/spracv1/spracv1.pdf?ts=1618149795335>. – Дата доступу: 29.04.2021.
41. Core i7 5960X - 5930K and 5820K processor review - Power Consumption. – Режим доступу: <https://www.guru3d.com/articles-pages/core-i7-5960x-5930k-and-5820k-processor-review,8.html>. – Дата доступу: 29.04.2021.
42. 15,200 MIPS on AWS with Heirloom (Autoscaling an IBM Mainframe Application to 1,018 Transactions/Second) | LinkedIn. – Режим доступу: <https://www.linkedin.com/pulse/15200-mips-aws-heirloom-paas-autoscaling-ibm-mainframe-gary-crook/>. – Дата доступу: 29.04.2021.
43. Analytics: Key to Mainframe Modernization. – Режим доступу: <https://www.gartner.com/imagesrv/media-products/pdf/rsd/RSD-1-4MFWW15.pdf>. – Дата доступу: 29.04.2021.