УДК 004.75:004.41 **DOI:** 10.31673/2412-9070.2026.017407
О. В. КОРЕЦЬКИЙ, аспірант,
ORCID:0009-0001-4809-7556
Державний університет інформаційно-комунікаційних технологій, Київ

МОДЕЛЬ ВИЗНАЧЕННЯ ПРИСТРОЮ ТУМАННИХ ОБЧИСЛЕНЬ ФРЕЙМВОРКУ, ПОБУДОВАНОГО НА ПЛАТФОРМІ МІКРОСЕРВІСНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Необхідність забезпечення розвитку світових економічних відносин, інтенсивний розвиток науки та зміцнення оборонного сектору вимагають постійного вдосконалення та розвитку галузі інформаційних технологій у напрямку підвищення швидкості та якості передавання корисних даних. На даний момент впровадження інформаційно-комунікаційних мереж на основі концепції високонадійного зв'язку з мінімальними затримками (URLLC) є одним із найскладніших завдань, що стоять перед науково-технічною спільнотою. Основною вимогою до мереж класу URLLC є висока надійність передавання даних з мінімальними затримками.

Ключовим аспектом досягнення цих цілей та забезпечення цих вимог є ефективне використання програмного забезпечення, яке зрештою безпосередньо вирішує функціональні завдання, тим самим генеруючи відповідний трафік телекомунікаційної мережі. У публікації розглянуто програмне забезпечення, побудоване на основі мікросервісного архітектурного стилю розробки та впровадження програмного забезпечення, який в останні 5–6 років активно розвивається.

У роботі подані результати розробки моделі визначення пристрою туманних обчислень фреймворку, побудованого на платформі мікросервісного програмного забезпечення. Подана модель визначення пристрою туманних обчислень, яка на основі технології алгоритмів ройового інтелекту – PSO (Particle Swarm Optimization) дозволяє визначити потенціал цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу.

Одержані в роботі результати оцінки застосовності поданої моделі, показали її придатність до визначення пристрою туманних обчислень фреймворку, побудованого на платформі мікросервісного програмного забезпечення. Показано, що застосування поданого в роботі алгоритму ройового інтелекту (PSO) у запропонованому в роботі варіанті фреймворку дозволяє зменшити час виконання функції мікросервісу за рахунок раціонального розподілу ресурсів на величину до 70%.

Ключові слова: фреймворк; мікросервісне програмне забезпечення; пристрої туманних обчислень; розподілені туманні динамічні обчислення.

Вступ

Необхідність забезпечення розвитку світових економічних відносин, інтенсивний розвиток науки та зміцнення оборонного сектору вимагають постійного вдосконалення та розвитку галузі інформаційних технологій у напрямку підвищення швидкості та якості передавання корисних даних. Одним із напрямків досліджень у цій галузі є розробка та швидке впровадження перспективних концепцій мереж передавання даних, а саме стандарту 5G/IMT-2020 та в його розвитку наступних поколінь таких мереж. На даний момент впровадження інформаційно-комунікаційних мереж на основі концепції високонадійного зв'язку з мінімальними затримками (URLLC) є одним із найскладніших завдань, що стоять перед науково-технічною спільнотою. Основною вимогою до мереж класу URLLC є висока надійність передавання даних з мінімальними затримками [1, 2]. Ці сервіси впроваджуються у таких сферах, як електронна медицина (e-health), автономний транспорт, небезпечне промислове виробництво формату 4.0 та інші.

Забезпечення розвитку світових економічних відносин, інтенсивний розвиток науки та зміцнення оборонного сектору вимагають постійного вдосконалення та розвитку галузі інфор-

маційних технологій у напрямку підвищення швидкості та якості передавання корисних даних. Одним із напрямків досліджень у цій галузі є розробка та швидке впровадження перспективних концепцій мереж передавання даних, а саме стандарту 5G/IMT-2020 та в його розвитку наступних поколінь таких мереж.

На даний момент впровадження інформаційно-комунікаційних мереж на основі концепції високонадійного зв'язку з мінімальними затримками (URLLC) є одним із найскладніших завдань, що стоять перед науково-технічною спільнотою. Основною вимогою до мереж класу URLLC є висока надійність передавання даних з мінімальними затримками [1, 2].

Ці сервіси впроваджуються у таких сферах, як електронна медицина (e-health), автономний транспорт, небезпечне промислове виробництво формату 4.0 та інші.

Ключовим аспектом досягнення цих цілей та забезпечення цих вимог є ефективне використання програмного забезпечення, яке зрештою безпосередньо вирішує функціональні завдання, тим самим генеруючи відповідний трафік телекомунікаційної мережі.

У сучасній галузі розробки програмного забезпечення накопичено величезний обсяг знань, зокрема про те, як писати ефективний код, методи, принципи та шляхи впровадження IT-продуктів (з точки зору коду) [3, 4]. Крім того, існують різні архітектурні рішення, такі як монолітна архітектура, багаторівнева архітектура, подієво-керована архітектура, безсерверна архітектура та мікросервісна архітектура [3, 4].

Протягом останніх 5–6 років архітектура мікросервісів у розробці та розгортанні програмного забезпечення швидко розвивалася, особливо після успішного пілотного застосування у відомих IT-продуктах, таких як Amazon та Netflix. Ця архітектура важлива для інфраструктурних рішень, таких як платформи Інтернету речей, веб-платформи, що реалізують більше одного продукту та інші [4, 5].

Постановка завдання

На рис. 1 показано приклад сервісу, реалізованого на основі мікросервісного підходу, який надає функції для покращення графічних зображень, сформованих на пристроях користувача. Такий сервіс реалізовано, наприклад, у додатку Google Фото.

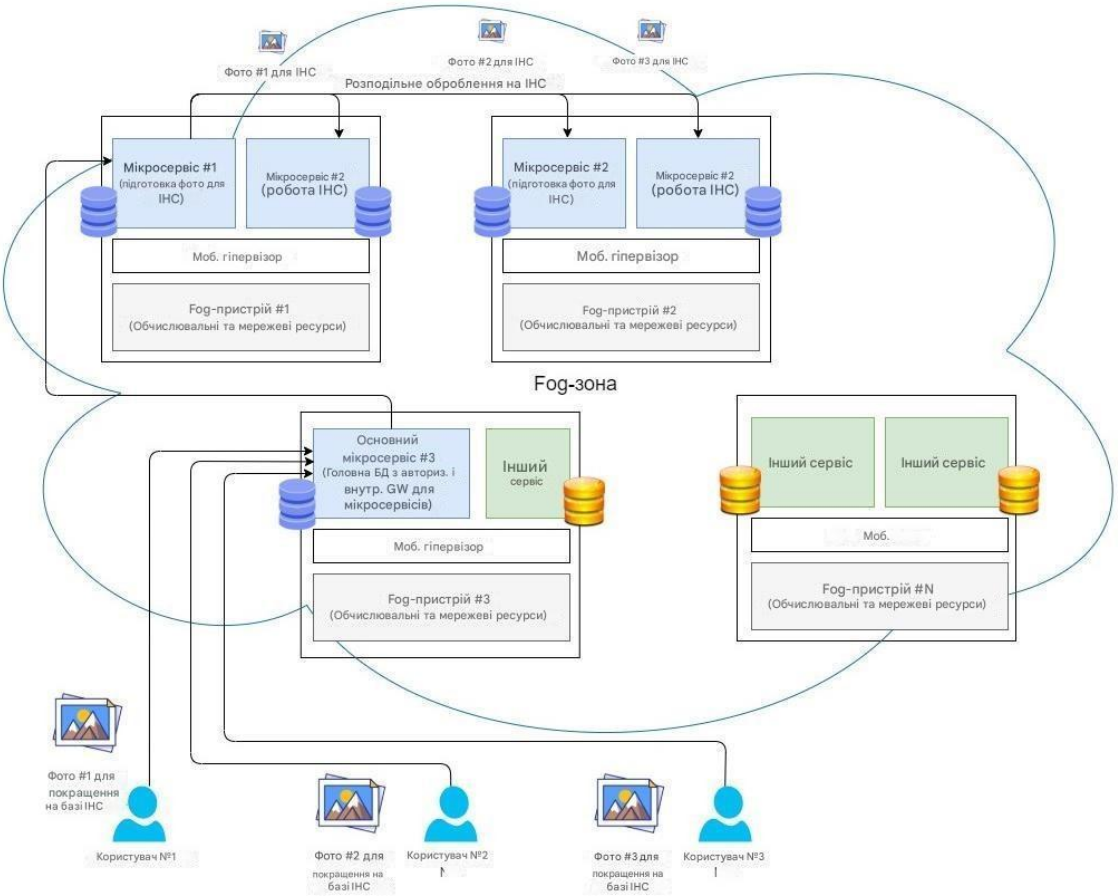

Рис. 1. Схема функціонування фреймворку, побудованого на платформі мікросервісного програмного забезпечення

Сервіс реалізовано у такому порядку. Після того, як користувач формує графічне зображення в вигляді окремого файлу застосунком мобільного пристрою, цей файл надсилається до найближчого мікросервісу, який готує його до обробки штучною нейронною мережею (наприклад, згорткового типу). Наступним кроком, після попередньої обробки, графічне зображення надсилається до мікросервісу, де реалізується обробка за допомогою штучних нейронних мереж [4, 5].

У цьому прикладі демонструється базовий сервіс, який можна розгорнути в рамках поданого фреймворку. Наприклад, більш цікавим та актуальним напрямком, якщо говорити про обробку відео/фото контенту, є застосування доповненої реальності (AR – Augmented Reality) та/або віртуальної реальності (VR – Virtual Reality) [4, 5, 6].

Протягом останніх трьох-чотирьох років у міжнародній науковій спільноті все частіше з'являються публікації, що обґрунтовують доцільність використання пристроїв з обмеженими обчислювальними та мережевими можливостями як повноцінних вузлів у розподіленому обчислювальному кластері. Такий підхід отримав назву «туманні обчислення» – за аналогією з природним явищем. У цьому контексті сукупність пристроїв (користувацькі термінали, пристрої Інтернету речей, елементи граничної мережі тощо), виділяючи частину своїх ресурсів, утворюють сукупність малих обчислювальних вузлів в архітектурі мережі – своєрідні «краплі туману» [7, 8].

Розподілені динамічні туманні обчислення можуть сприяти реалізації напрямку мікросервісів, забезпечуючи величезні обчислювальні можливості (завдяки сукупності оточуючих пристроїв), зменшуючи навантаження на мережу та мінімізуючи затримки передавання даних.

Міжнародна рекомендація МСЕ-Т Q.3745 «Протокол для застосунків Інтернету Речей із часовими обмеженнями поверх програмно-конфігурованих мереж» описує не лише протокол, але й відповідний фреймворк [9].

Запропонований фреймворк формалізує структуру системи та регламентує відповідні системні процеси, у межах яких використовується протокол взаємодії між сервером послуг Інтернету речей та системним застосунком програмно-конфігурованої мережі (SDN-додатком).

Фреймворк орієнтований на інтеграцію хмарних інфраструктур з багаторівневими пограничними (edge) та туманними (fog) обчисленнями, а також з мережевою інфраструктурою, побудованою на основі технологій SDN/NFV, з урахуванням рівня оркестрації ресурсів і сервісів.

У результаті формується єдина конвергентна інфраструктура, яка забезпечує реалізацію нових підходів до розподілу обчислювальних навантажень, зокрема на мережевому краї (edge) та безпосередньо на кінцевих пристроях, включно з вузлами Інтернету речей.

У межах розглянутої архітектури фреймворку виділяється сукупність функціональних елементів, що реалізують визначені обчислювальні функції. Структура МЕС характеризується ієрархічною організацією, у якій нижчі обчислювальні рівні підпорядковуються вищим хмарним шарам. Натомість у туманних (Fog) обчисленнях жорстко фіксована ієрархія, як правило, відсутня.

За умови коректного проєктування туманної інфраструктури з урахуванням технічних параметрів та обчислювальних можливостей кожного Fog-пристрою, їхнього динамічного просторового розподілу, а також використання технологій прямої взаємодії пристрій-пристрій (D2D, Device-to-Device), може бути досягнуто суттєвого синергетичного ефекту.

Слід відзначити, що в межах такої структури Fog-пристрої зазнають динамічного перерозподілу, що зумовлює зміну ресурсоємності окремих Fog-зон. Поверх цієї динамічної обчислювальної інфраструктури виконується міграція мікросервісів прикладних сервісів. У розробленому фреймворку визначено формалізовану логіку взаємодії функціональних елементів під час підключення нового Fog-пристрою. Відповідна послідовність обміну повідомленнями представлена у вигляді діаграм повідомлень на рис. 2.

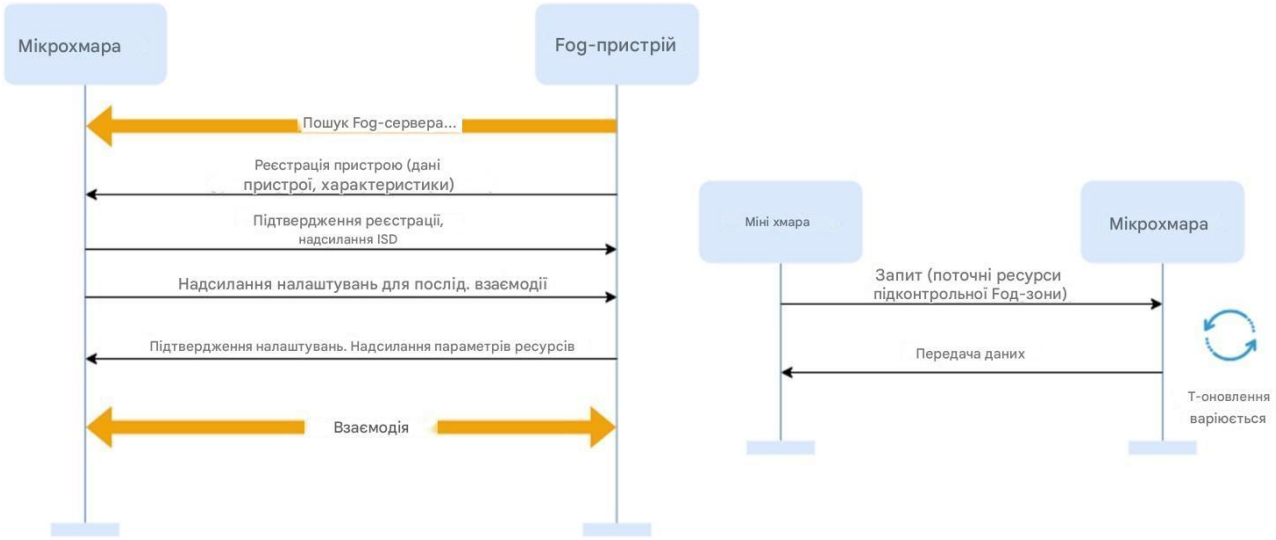

Рис. 2. Загальна діаграма взаємодії основних елементів фреймворку, побудованого на платформі мікросервісного програмного забезпечення

У межах запропонованого фреймворку передбачається розв'язання комплексної задачі динамічного перерозподілу Fog-пристроїв, які обираються для виконання визначених обчислювальних процедур у межах функціонування єдиної архітектурної платформи [6, 7].

Для розв'язання окремих підзадач цієї комплексної проблеми пропонується використовувати набір ефективних алгоритмів обробки даних.

Загальний алгоритм, покладений в основу фреймворку, а також логіка взаємодії функціональних елементів, наведена на рис. 2, вимагають розв'язання низки наукових задач, пов'язаних з розробкою моделей і методів. До яких віднесемо наступні.

1. Визначення центру скупчення користувачів (або їхніх запитів) для конкретного сервісу.

2. Оцінювання обчислювального потенціалу цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу.

У даній роботі буде розглянуте питання, пов'язане з оцінюванням обчислювального потенціалу цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу в межах одного фреймворку.

Аналіз останніх досліджень і публікацій

Питанню визначення обчислювального потенціалу цільового середовища туманних обчислень фреймворку, побудованого на платформі мікросервісного програмного забезпечення, присвячено ряд наукових робіт [8, 10-13].

У статті [8] подано огляд підходів до Fog-обчислень, коли вузли та користувачі мобільні, топологія змінюється. Огляд показує тренди та проблеми динаміки та розподіленості розрахунків, але безпосередні питання визначення обчислювального потенціалу цільового середовища туманних обчислень у даній роботі не розглядаються..

Стаття [10] присвячена Fog-обчисленням у динамічних бездротових Mesh-системах з використанням архітектури NDN (Named Data Networking). Враховується мобільність вузлів, стан мережевих зв'язків, що змінюється, розподілені ресурси. Подана модель не враховує безпосередні запити користувачів та визначення обчислювального потенціалу цільового середовища туманних обчислень, призначеного для задоволення їх запитів.

Огляд алгоритмів планування завдань у Fog-середовищах - характеристика, проблеми, напрямки майбутніх досліджень подано в роботі [11]. Питання розподілу користувачів та врахування їх щільності у зонах розподілу в даній роботі визначено, як одна з проблем планування завдань у Fog-середовищах. Безпосередню механізми визначення обчислювального потенціалу цільового середовища туманних обчислень в даній роботі не подано.

У статті [12] пропонується динамічний розподіл Fog-вузлів на групи (працюючі, резервні, «сплячі») з прогнозом навантаження, щоб оптимізувати енергоспоживання та затримки. Подано приклад динамічного управління ресурсами з екологічним акцентом але без акцентування на визначення обчислювального потенціалу цільового середовища туманних обчислень.

У роботі [13] представлено результати розробки моделі розподілу завдань та ресурсів у Fog-мережі з IoT-пристроями. Акцент в поданій моделі сконцентровано на комплексних системах з підходами та реалізаціями розподіленої динаміки розрахунків. Безпосередні моделі, призначені для визначення обчислювального потенціалу цільового середовища туманних обчислень у даній роботі не представлені.

Формулювання цілей статті

Метою статі є розробка моделі оцінювання обчислювального потенціалу цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу фреймворку, побудованого на платформі мікросервісного програмного забезпечення.

Основна частина

Одним із завдань, згідно з алгоритмом фреймворку, зображеним на рис. 2, є періодичне оцінювання обчислювального потенціалу цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу в межах одного фреймворку.

Вирішення цієї задачі пов'язане з процесом визначення Fog-пристроїв, які мають вільні необхідні обчислювальні ресурси та відповідають умовам міграції на них мікросервісів із подальшим розгортанням і включенням їх у загальну архітектуру фреймворку.

У межах цього підпроцесу фреймворку необхідно вирішити задачу оптимізації: для заданої функції, яка описує стан Fog-пристроїв через набір параметрів, серед усіх можливих значень цієї функції потрібно знайти таке значення, за якого функція досягає максимального (max) значення (мінімальні значення відкидаються).

Існує багато алгоритмів, які гарантовано знаходять екстремум функції, що є локальним мінімумом поблизу заданої початкової χ_0. До таких алгоритмів належать, наприклад, алгоритми градієнтного спуску. Проте в цьому завданні потрібно знайти глобальний максимум функції, яка в заданому діапазоні параметрів, окрім одного глобального екстремуму, має багато локальних екстремумів. Градієнтні алгоритми не можуть оптимізувати таку функцію, оскільки їхнє рішення сходиться до найближчого екстремуму біля початкової точки [14, 15].

Для задач пошуку глобального максимуму або мінімуму використовуються так звані алгоритми глобальної оптимізації. До таких алгоритмів належать алгоритми ройового інтелекту – PSO (*Particle Swarm Optimization*). Алгоритм PSO розроблений на основі принципу поведінки біологічних організмів, зокрема здатності груп деяких видів тварин працювати як єдине ціле для визначення бажаних позицій у заданій області.

PSO - один із найефективніших і універсальних алгоритмів ройового інтелекту. Його ключові сильні сторони - простота, універсальність, відсутність потреби у градієнті, здатність працювати з великими та складними просторами рішень, можливість паралелізації. Завдяки цьому PSO підходить для багатьох реальних задач у науці, інженерії та машинному навчанні.

До основних переваг алгоритму PSO віднесем наступне [14, 15, 16]:

- **Гнучкість і універсальність** - PSO є метаевристичним методом і він не потребує похідних, гладкості функції чи навіть аналітичного опису проблеми. Це дає змогу застосовувати його у дуже різноманітних проблемах, навіть таких, де цільова функція змінюється, шумна, або складна.
- **Проста структура та відносно невеликі обчислювальні витрати** - базовий PSO простіший для реалізації, ніж складні еволюційні алгоритми і має невелику кількість параметрів.

- **Можливість швидкого пошуку та глобальна оптимізація** - за рахунок одночасної роботи багатьох частинок, які досліджують простір рішень, PSO добре справляється з багатовимірними просторами й має шанси знайти глобальне (або близьке до нього) рішення, навіть якщо функція складна.
- **Потенціал модифікації та адаптації до динамічних умов** - як зазначено у статті, існують різні варіанти PSO, пристосовані до змін: наприклад, мульти-ройові підходи, перезапуск, адаптивна зміна параметрів тощо.
- **Можливість гібридизації** - PSO часто комбінують з іншими методами (еволюційними або евристичними), що дає змогу підсилити стабільність і продуктивність алгоритму при нестабільних або складних умовах.

Отже, PSO є «базовим інструментом», на якому можна будувати спеціалізовані, адаптивні рішення, що добре підходять для задач з динамічним характером.

Основні компоненти PSO [14]:

- позиція x_{i1} окремої частинки рою;
- швидкість v_{i1} окремої частинки рою;
- найкраща знайдена позиція p_{i1} окремої частинки рою;
- найкраща знайдена позиція g всього рою.

Алгоритм ройового інтелекту – PSO для визначення Fog-пристроїв побудуємо наступним чином.

Кожна окрема частинка рою складається з трьох векторів: її положення в D-вимірному просторі пошуку визначається положенням:

$$\overline{x_i} = (x_{i1}, x_{i2}, \ldots\ldots, x_{iD}). \tag{1}$$

Найкращу знайдену позицію визначимо як:

$$\overline{p_i} = (p_{i1}, p_{i2}, \ldots\ldots, p_{iD}). \tag{2}$$

Направлена швидкість руху визначається як:

$$\overline{v_i} = (v_{i1}, v_{i2}, \ldots\ldots, v_{iD}). \tag{3}$$

Під час запуску алгоритму частинки рівномірно й випадково ініціалізуються в усьому просторі пошуку, причому швидкість частинок також ініціалізується випадковим чином. Сформовані частинки переміщуються простором пошуку за допомогою простого набору рівнянь оновлення векторів частинки.

Алгоритм оновлює весь рій на кожному часовому кроці, оновлюючи швидкість і положення кожної частинки у кожному вимірі за наступними правилами:

$$v_{id}^{t+1} = v_{id}^{t} + c\varepsilon_1\left(p_{id}^{t} - x_{id}^{t}\right) + c\varepsilon_2\left(p_{gd}^{t} - x_{id}^{t}\right) \tag{4}$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t}, \tag{5}$$

де C – коефіцієнт прискорення; ε_1 й ε_2 – випадкові числа в межах [0; 1]; p_{id}^{t} – найкраще положення, знайдене всіма частинками; p_{gd}^{t} – положення, знайдене будь-якою сусідньою частинкою.

Процес оновлення коротко описаний у псевдокоді, приведеному у таблиці.

Алгоритм оновлення рою (*PSO)*

for кажного кроку *t* **do**
for кожної частинки *i* в рою **do**
оновити позицію *xt* до виразу (4) й (5)
рорзрахувати фітнес-функцію для *xt* f(xt)оновити *pi , pg*
end forend for

Варто зазначити, що в алгоритмі швидкість частинок фіксується на максимальному значенні $V_{\max}$. Без фіксації алгоритм схильний не сходитися, оскільки обчислення значень (4) і (5) призводили б до швидкого зростання швидкості та, відповідно, положення частинок, що наближаються до нескінченності. Параметр $V_{\max}$ не дозволяє системі увійти в такий стан, обмежуючи швидкість усіх частинок.

У результаті необхідно визначити параметри, що описують кожен із досліджуваних Fog-пристроїв. При цьому деякі параметри оцінюються на рівні порівняння з граничними значеннями, наприклад, обсяг виділеної логічної оперативної пам'яті (RAM), необхідної для роботи мікросервісу, що готується до міграції.

Для вирішення поточного завдання визначимо параметри, що описують Fog-вузол з точки зору забезпечення якості обслуговування.

Глобальне завдання сформульоване таким чином: необхідно підтримувати час обслуговування шляхом вибору Fog-вузла, на який потрібно мігрувати мікросервіс. Таким чином, мінімізована функція пристосування виглядає так:

$$T = \sum_{i=1}^{n} W_i \, TimeSlot_i , \tag{6}$$

де T – розрахований параметр в [мс]; W_i- вага відповідного параметру; $TimeSlot_i$ – (fitness-функція) параметри, що описують стан Fog-пристрою; n-кількість таких параметрів.

Для побудови алгоритму використаємо два параметри:

- $TimeSlot_1$ (затримка розповсюдження);
- $TimeSlot_2$ (час обробки запиту мікросервісом).

Оскільки обидва параметри мають однаковий вплив на оцінюваний параметр, вага кожного з параметрів (W_i) дорівнює 0.5, причому сума ваг не повинна перевищувати 1.

Ці параметри обчислюються в [мс], тому немає потреби приводити їх до однієї області значень, як це необхідно при оцінці параметрів із різних областей визначення.

У результаті функція пристосування в межах цієї роботи виглядає так:

$$T = \sum_{i=1}^{n} W_i \, TimeSlot_i = 0.5 \, TimeSlot_1 + 0.5 \, TimeSlot_2 . \tag{7}$$

Перший параметр ($TimeSlot_1$) визначається за допомогою тайм-трекера на рівні фреймворку.

Другий параметр ($TimeSlot_2$) надсилається пристроєм як час обробки задачі.

Для моделювання алгоритму ройової оптимізації (PSO) застосована програмна модель на мові Python із застосуванням відповідних бібліотек і фреймворків, таких як NumPy, Pandas, Matplotlib, Math та інші.

За їх допомогою було згенеровано скупчення користувацьких пристроїв, розташовані у 5-ти кластерах. Припустимі центри скупчення користувачів були розподілені випадковим чином

відповідно до принципу найбільш імовірних місць у просторі скупчення людей. Загалом, були згенеровано координати пристроїв досліджуваних регіонів відповідно визначених кластерів.

У кластері №1 було згенеровано 10 користувацьких пристроїв, у 2-му кластері – 55, у 3-му – 17, у 4-му – 32, а у 5-му – 55 користувацьких пристроїв.

Для наочності та оцінки застосованості запропонованої моделі були також згенеровані Fog-вузли.

Згідно із заданою функцією придатності (fitness-функцією) та вище поданим алгоритмом PSO (1) – (6) були отримані результати для 5-го кластера, що представлені на рис.3.

У 5-му кластері знаходиться 55 пристроїв, кожний з яких характеризується набором параметрів, у тому числі часових, що були згенеровані у наступних межах:

$$TimeSlot_1 \in [0.5, 10]\ ms, \qquad TimeSlot_2 \in [0.2, 2]\ ms \tag{8}$$

Результат роботи алгоритму PSO показав, що 13-й Fog-пристрій з показниками $TimeSlot_1$ = 1.15 [ms], $TimeSlot_2$ = 0.69 [ms] і значенням fitness-функції 0.86 має найменший необхідний показник для подальшого розміщення мікросервісу, а також відповідає необхідному обчислювальному та мережевому потенціалу.

Показники цього пристрою позначено зеленим хрестиком на рис.3. Також цей графік візуалізує дані, розраховані для кожного Fog-пристрою (згенеровані дані у відповідних межах і відповідне значення fitness-функції).

Як доповнення до отриманих результатів варто зазначити, що на основі значень fitness-функції існує можливість розподілу множини Fog-пристроїв однієї Fog-зони на мікрокластери. Кожен мікрокластер може задовольняти необхідним показникам якості послуг. Приклад такого розподілу для наочності представлений на рис.4, де кожен із «мікрокластерів» забезпечує необхідний рівень варійованої якості у відповідних межах послуги.

Мікрокластери позначено у вигляді бежевих овалів на рис.4.

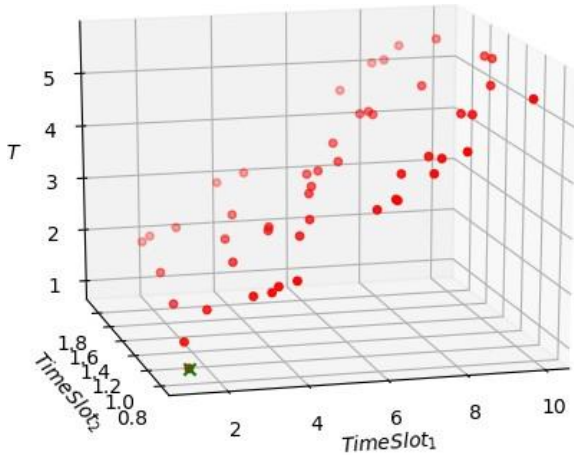

Рис. 3. Візуалізація даних розрахунку fitness-функції та результату моделювання алгоритму ройового інтелекту – PSO

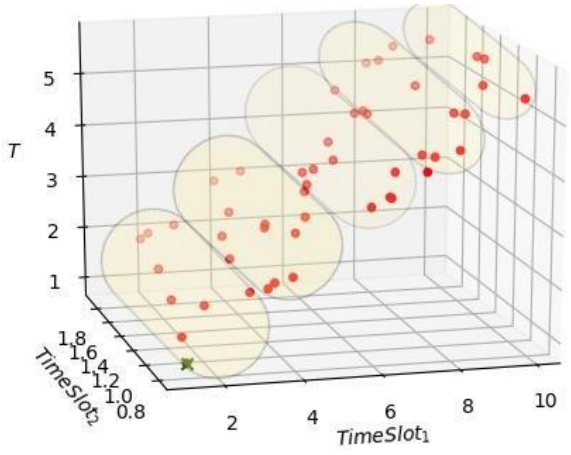

Рис.4. Розподіл зони туманних обчислень на мікрокластери послуг

Цей розподіл є віртуальним для систем моніторингу та управління у рамках запропонованого фреймворку з метою оперативної роботи алгоритмів з вибору Fog-пристроїв у Fog-зоні для подальшої міграції мікросервісів із збереженням якості наданих послуг.

Для оцінки ефекту застосування ройового алгоритму проведемо наступні розрахунки. Вище було наведено параметри та описано алгоритм для раціонального вибору Fog-пристрою з метою подальшої міграції мікросервісу. Як параметри приймаються наступні:

- $TimeSlot_1$ (затримка поширення);
- $TimeSlot_2$ (час обробки запиту мікросервісом).

У результаті моделювання було визначено, що 13-й пристрій має мінімальні значення цих параметрів: $TimeSlot_1$ = 1.15 [*ms*], $TimeSlot_2$ = 0.69 [*ms*], де час виконання функції оцінюється як сума цих параметрів:

$$T_{функц} = TimeSlot_1 + TimeSlot_2 . \quad (9)$$

Таким чином, мінімальний час виконання функції на визначеному Fog-пристрої:

$$T_{функц} = 1.16 + 0.69 = 1.85ms . \quad (10)$$

Якщо здійснити вибір Fog-пристрою рівновісно з повного набору пристроїв (при розрахунку прийнято, що їх в зоні розосередження 55 пристроїв), то час виконання функції можна оцінити середнім значенням функції.

Вираз для її розрахунку подамо у вигляді:

$$T_{функ.ср} = \frac{1}{N}\sum_{i=1}^{n} TimeSlot_{1i} + TimeSlot_{2i} = \frac{1}{N}\sum_{i=1}^{N} T_{функ_i} . \quad (11)$$

Прийнявши до уваги згенеровані дані у поданій вище моделі, середній час виконання функції мікросервісом при рівновісному розподілі функції на пристрої Fog-зони складе $T_{функ.ср}$ =6.25 *ms.*

Проведемо точкову оцінку ефекту застосування алгоритму вибору Fog-пристрою щодо часу виконання функції мікросервісом через відношення отриманого мінімального часу виконання функції $T_{функ.\min}$ до середнього значення виконання у випадку рівноймовірного вибору пристрою $T_{функ.ср}$.

$$T_{функ.\min} \frac{\left(T_{функ.\min} *100\%\right)}{T_{функ.сер}} = 28.8\% . \quad (12)$$

Таким чином, можна зробити висновок, що застосування поданого в роботі алгоритму ройового інтелекту (PSO) у запропонованому фреймворку дозволяє зменшити час виконання функції мікросервісу за рахунок раціонального розподілу ресурсів на величину до 70%.

Висновки

У роботі подані результати розробки моделі визначення пристрою туманних обчислень фреймворку, побудованого на платформі мікросервісного програмного забезпечення. Подана модель визначення пристрою туманних обчислень, яка на основі технології алгоритмів ройового інтелекту – PSO (*Particle Swarm Optimization*) дозволяє визначити потенціал цільового середовища туманних обчислень з метою вибору Fog-пристрою, на який доцільно здійснити міграцію відповідного мікросервісу. Висвітлені в роботі результати оцінки застосовності поданої моделі показали її придатність до визначення пристрою туманних обчислень фреймворку, побудованого на платформі мікросервісного програмного забезпечення. Показано, що застосування поданого в роботі алгоритму ройового інтелекту (PSO) у запропонованому в роботі варіанті фреймворку дозволяє зменшити час виконання функції мікросервісу за рахунок раціонального розподілу ресурсів на величину до 70%.

Список літератури

1. Shusta, V. S., Susla, A. I., & Biganych, V. Yu. (2024). Transformation of network technologies: from 4G to 5G. Scientific papers of V. I. Vernadsky TNU. Series: Technical sciences, 3, 94–99.

2. ITU-T. (2018). ITU-T GSTP-TN5G: Transport network support of IMT-2020/5G (Technical Report SG15-TD338/PLEN).

3. ITU-T. (2018). Draft recommendation G.ctn5g: Characteristics of transport networks to support IMT-2020/5G (Draft Recommendation SG15-TD295/PLEN).

4. Koretskyi, O. (2024). Architecture of multifunctional application software microservices. Modern Information Security, 3(59), 85–93. https://doi.org/10.31673/2409-7292.2024.030009

5. Richardson, C. (2018). Microservices patterns: With examples in Java. Manning.

6. Gabbrielli, M., Giallorenzo, S., Guidi, C., Mauro, J., & Montesi, F. (2016). Self-reconfiguring microservices. In Theory and practice of formal methods (Lecture Notes in Computer Science, Vol. 9660, pp. 194–210). Springer. https://doi.org/10.1007/978-3-319-30734-3_14

7. Khomenchuk, V. (2023). Analysis of the impact of cloud computing and edge computing on network performance. In Technological Horizons: Research and Application of Information Technologies for Technological Progress in Ukraine and the World (pp. 361–362). Kyiv, Ukraine.

8. Ostrowski, K., Małecki, K., Dziurzański, P., & Singh, A. K. (2023). Mobility-aware fog computing in dynamic networks with mobile nodes: A survey. Journal of Network and Computer Applications, 219, 103724. https://doi.org/10.1016/j.jnca.2023.103724

9. ITU-T. (2020). ITU-T Recommendation Q.3745: Protocol for time constraint Internet of things-based applications over software-defined networking. International Telecommunication Union.

10. Srikanteswara, S., Samuylov, A., Arrobo, G., Zhang, Y., Feng, H., Himayat, N., Spoczynski, M., & Koucheryavy, Y. (2024). Provisioning of fog computing over named-data networking in dynamic wireless mesh systems. Sensors, 24(4), 1120. https://doi.org/10.3390/s24041120

11. Matrouk, K., & Alatoun, K. (2021). Scheduling algorithms in fog computing: A survey. International Journal of Networking and Distributed Computing, 9, 59–74. https://doi.org/10.2991/ijndc.k.210111.001

12. Ali Kumar, D. S. N. K., Newaz, S. H. S., Rahman, F., & Au, T. W. (2022). Green demand aware fog computing: A prediction based dynamic resource provisioning approach. Electronics, 11(4), 608. https://doi.org/10.3390/electronics11040608

13. Faraji, F., Javadpour, A., Sangaiah, A., & Zavieh, H. (2023). A solution for resource allocation through complex systems in fog computing for the Internet of Things. Computing, 106, 1–25. https://doi.org/10.1007/s00607-023-01199-1

14. Zhang, Y., Wu, L., & Wang, S. (2015). A comprehensive survey on particle swarm optimization algorithm and its applications. Mathematical Problems in Engineering, 2015, *Article ID 931256. https://doi.org/10.1155/2015/931256*

15. Рабійчук, І. О., & Фечан, А. В. (2024). Основні виклики адаптаційності алгоритмів ройового інтелекту. Scientific Bulletin of UNFU, 34(5), 97-103. https://doi.org/10.36930/40340513

16. Karatanov, O. V., Ustymenko, O. V., Yena, M. V., Bova, Y. A., & Kalashnikova, V. I. (2021). Application of swarm intelligence algorithms in the design of control systems for groups of unmanned aerial vehicles. Young Scientist, 24(10), 98–103. https://doi.org/10.32839/2304-5809/2021-10-98-24.

O. Koretskyi

MODEL FOR DETERMINING A FOG COMPUTING DEVICE FOR A FRAMEWORK BUILT ON A MICROSERVICE SOFTWARE PLATFORM

The need to ensure the development of world economic relations, the intensive development of science and the strengthening of the defense sector require constant improvement and development of the information technology industry in the direction of increasing the speed and quality of useful data transmission. At the moment, the implementation of information and communication networks based on the concept of highly reliable communication with minimal delays (URLLC) is one of the most difficult tasks facing the scientific and technical community. The main requirement for URLLC class networks is high reliability of data transmission with minimal delays.

A key aspect of achieving these goals and ensuring these requirements is the effective use of software, which ultimately directly solves functional tasks, thereby generating the corresponding traffic of the telecommunications network.

The paper considers software built on the basis of the microservice architectural style of software development and implementation, which has been actively developing in the last 5–6 years.

The paper presents the results of solving one of the current tasks within the framework of the problem of implementing the microservice approach, namely, the development of a model for determining the fog computing device of a framework built on a microservice software platform.

A model for determining the fog computing device is presented, which, based on the technology of swarm intelligence algorithms - PSO (Particle Swarm Optimization), allows you to determine the potential of the target fog computing environment in order to select a Fog device to which it is advisable to migrate the corresponding microservice.

The results of the assessment of the applicability of the presented model, highlighted in the paper, showed its suitability for determining the fog computing device of a framework built on a microservice software platform.

It is shown that the use of the swarm intelligence algorithm (PSO) presented in the paper in the framework version proposed in the paper allows you to reduce the execution time of the microservice function by up to 70% due to the rational allocation of resources.

Keywords: framework; microservices software; fog computing devices; distributed fog dynamic computing.

Надійшла до редакції: 26.12.2025
Прийнята до друку: 06.02.2026
Опубліковано: 27.02.2026