Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочної форми навчання
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА

## Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи моделювання систем хмарних обчислень

(тема)

Виконав:

студент II курсу, групи КСМзм-22-1

Баляба Ю.В.
(прізвище, ініціали)

Спеціальність

123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

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

Освітня програма

Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: доц. Саранча С.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет навчально-науковий центр заочної форми навчання

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

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

Освітня програма Комп'ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

# ЗАВДАННЯ

## НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Балябі Юлії Вікторівні
(прізвище, ім'я, по батькові)

1. Тема роботи Методи моделювання систем хмарних обчислень

затверджена наказом по університету від " 03 " листопада 2023 р. № 244 Стз

2. Термін подання студентом роботи до екзаменаційної комісії 15 січня 2024 р.

3. Вхідні дані до роботи

Моделі хмарних обчислень.

Постачальники хмарних послуг (AWS, GCP, Azure).

Методи оцінки продуктивності хмарних платформ.

4. Перелік питань, що потрібно опрацювати у роботі

Аналіз предметної області.

Аналіз та вибір моделей хмарних сервісів.

Проведення експериментів

Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Презентація 14 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1 )

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| | | | |
| | | | |

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз предметної області | 07.11.23 – 15.11.23 | |
| 2 | Розробка моделей | 16.11.23 – 30.11.23 | |
| 3 | Реалізація алгоритмів | 01.12.23 – 8.12.23 | |
| 4 | Розробка структури програмних засобів | 09.12.23 – 13.12.23 | |
| 5 | Розробка програмних модулів | 14.12.23 – 25.12.23 | |
| 6 | Оформлення матеріалів кваліфікаційної роботи | 26.12.23 – 02.01.24 | |
| 7 | Подання кваліфікаційної роботи керівникові та | 03.01.24 – 10.01.24 | |
| | попередній захист | | |
| 8 | Подання кваліфікаційної роботи на рецензу- | 11.01.24 – 15.01.24 | |
| | вання | | |
| | | | |
| | | | |

Дата видачі завдання 06 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Саранча С.М.
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 58 с., 15 рис., 7 табл., 1 дод., 69 джерел.

МЕТОДИ МОДЕЛЮВАННЯ, ХМАРНІ ОЧИСЛЕННЯ, СТАБІЛЬНАСТЬ, ПОТОКИ ЗАВДАНЬ, МОДЕЛІ ОБСЛУГОВУВАННЯ.

Хмарні обчислення – це нова технологія, яка пропонує інформаційні послуги через Інтернет. Клієнти користуються послугами, які їм потрібні, коли вони потребують і там, де вони хочуть, і платять лише за те, що вони спожили. Отже, хмарні обчислення пропонують багато переваг, особливо для бізнесу. Глибоке вивчення та розуміння цієї нової системи та її компонентів дуже допомагає визначити, що потрібно зробити, щоб покращити її продуктивність. В кваліфікаційній роботі ми спочатку представляємо хмарні обчислення та їх компоненти, а потім описуємо ідею, яка намагається оптимізувати керування системою хмарних обчислень, яка складається з багатьох центрів обробки даних. В основу досліджень покладено методи моделювання складних систем.

Для програм, розміщених у системі хмарних обчислень, де є багато серверів, для обробки вхідних викликів програм, дуже важлива оцінка стабільності хмари. Однак загальну оцінку завжди важко досягти, оскільки досі немає стандартних визначень, методів, які використовуються в хмарних обчисленнях. У кваліфікаційній роботі представлено середовище моделювання хмарних обчислень, яке надає змогу оцінити логічну стабільність хмари за різних конфігурацій без проведення експериментів у реальному хмарному середовищі. Коректність моделювання підтверджується результатами теоретичних розрахунків відомої системи масового обслуговування.

# ABSTRACT

Master’s thesis: 58 pages, 15 figures, 7 tables, 1 appendice, 69 sources.

SIMULATION METHODS, CLOUD COMPUTING, STABILITY, TASK FLOWS, SERVICE MODELS.

Cloud computing is a new technology that offers information services over the Internet. Customers use the services they need, when they need them, where they want them, and pay only for what they use. Hence, cloud computing offers many benefits, especially for businesses. A deep study and understanding of this new system and its components is very helpful in determining what needs to be done to improve its performance. In the benchmark paper, we first introduce cloud computing and its components, and then describe an idea that tries to optimize the management of a cloud computing system consisting of many data centers. The research is based on methods of modeling complex systems.

For applications hosted in a cloud computing system where there are many servers to handle incoming application calls, cloud stability assessment is very important. However, a general assessment is always difficult to achieve because there are still no standard definitions of the methods used in cloud computing. The attestation work presents a cloud computing simulation environment, which makes it possible to evaluate the logical stability of the cloud under various configurations without conducting experiments in a real cloud environment. The correctness of the modeling is confirmed by the results of theoretical calculations of a well-known mass service system.

ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІТ – інформаційні технології

США – Сполучені Штати Америки

MOPS – мільйон операцій за секунду

ПЗ – програмне забезпечення

ПК – персональний комп'ютер

ЦП – центральний процесор

API – Application Programming Interface

AWS – Amazon Web Services

CRM – Customer Relationship Management

EC2 – Elastic Compute Cloud

GCP – Google Cloud Platform

HPC – High Performance Computing

IaaS – Infrastructure as a Service

IoT – Internet of Things

HTTP – HyperText Transfer Protocol

MIPS – Million Instructions Per Second

MPI – Message Passing Interface

MPICH – Message Passing Interface CHameleon

NPB – NAS parallel benchmark

PaaS – Platform as a service

SaaS – Software as a service

SOAP – Simple Object Access Protocol

SSD – Solid-State drive

WDS – бездротова розподільча система (англ., Wireless Distribution System)

USD – United States dollar

# ВСТУП

На сьогодні ще немає стандартного, формалізованого визначення концепції хмарних обчислень. Вони є природним кроком в еволюції послуг і продуктів інформаційних технологій на вимогу [1]. Хмарні обчислення, безсумнівно, починають мати значний вплив як на окремих кінцевих користувачів, так і на підприємства. Вони також привертають все більше уваги для досліджень і розробок з боку IT-експертів в комерційних секторах економіки. Використаємо визначення з [2], оскільки воно дає більш повну картину особливостей хмарних обчислень. «Хмари – це великий пул віртуалізованих ресурсів, які легко використовувати та які є доступними віддалено (такі як апаратне забезпечення, платформи розробки та/або служби). Ці ресурси можна динамічно переконфігурувати для адаптації до змінного навантаження (масштабу), що також забезпечує оптимальне використання ресурсів. Цей пул ресурсів зазвичай використовується за допомогою моделі оплати за використання, у якій гарантії надаються постачальником інфраструктури за допомогою індивідуальних угод про рівень обслуговування». Як описано в звіті Horizon Report [3], де хмарні обчислення названо однією з шести нових технологій, на які варто звернути увагу, «Поява великомасштабних «ферм даних» – великих кластерів, мережевих серверів – забезпечує доступність величезних об’ємів обчислювальної потужності та ємності для зберігання даних. Недорогі, прості рішення для зовнішнього зберігання, масштабування багатокористувацьких додатків, хостингу та багатопроцесорних обчислень відкривають двері до абсолютно інших способів мислення про комп’ютери, програмне забезпечення та файли».

Користувачі споживають електроенергію, не знаючи, як ця електроенергія була отримана. Хмарні обчислення, це сучасне слово, є прийняттям цієї концепції в галузях інформаційних технологій (IT). Отже,

хмарні обчислення – це бізнес-модель, де IT-послуги пропонуються споживачеві. Найважливіша мета цієї системи – легко надавати обчислювальні послуги, як-от вода, електроенергія, газ і телефон. Таким чином, споживачі використовують ресурс та/або послугу, коли їм це потрібно, відповідно до обсягів своїх потреб і, отже, платять на основі рівня їх використання. Власне, хмарні обчислення – це новий спосіб споживання інформатики. Він являє собою обчислювальну модель, у якій ресурс (програмне або апаратне забезпечення) розміщується, запускається, контролюється та адмініструється через Інтернет у великих центрах обробки даних. Цей ресурс надається як послуга. Так, хмарні обчислення є відповіддю на нові вимоги в IT. Насправді можна сказати, що хмарні обчислення є еволюцією концепції «грід-обчислень». Найважливіша відмінність між ними полягає в методі управління [4]. У грід-обчисленнях користувач повинен керувати всією системою (сервером, елементом мережі, операційною системою, програмним забезпеченням і т.і.). Але в хмарних обчисленнях система пропонується як послуга. Таким чином, користувач займається лише тим, що йому потрібно, і не переймається іншими послугами/питаннями. Це означає, що хмарні обчислення можна використовувати дружньо [5]. Насправді хмарні обчислення приймають концепцію корисних обчислень. Таким чином, більшість людей можуть використовувати його без будь-яких конкретних знань про те, як працює система, або потреби будь-чого керувати.

Отже, хмарні обчислення пропонують багато переваг, особливо для бізнесу. Глибоке вивчення та розуміння цієї нової системи та її компонентів дуже допомагає визначити, що ми повинні зробити, щоб покращити її продуктивність. В кваліфікаційній роботі ми спочатку представляємо хмарні обчислення та їх компоненти, а потім описуємо ідею, яка намагається оптимізувати керування системою хмарних обчислень, яка складається з багатьох центрів обробки даних. В основу досліджень покладено методи моделювання складних систем.

Для програм, розміщених у системі хмарних обчислень, де є багато серверів, для обробки вхідних викликів програм, дуже важлива оцінка стабільності хмари. Це важливо як для планування нових програм, так і для розширення існуючих програмних систем. Однак загальну оцінку завжди важко досягти, оскільки досі немає стандартних визначень, методів, які використовуються в хмарних обчисленнях. Фактична система хмарних обчислень може бути дуже великою та складною. У кваліфікаційній роботі представлено середовище моделювання хмарних обчислень. Це дає змогу оцінити логічну стабільність хмари за різних конфігурацій без проведення експериментів у реальному хмарному середовищі. Коректність моделювання підтверджується результатами теоретичних розрахунків відомої системи масового обслуговування.

Метою роботи є підвищення ефективності хмарних систем шляхом розробки методів моделювання обчислювальних процесів для оцінки стабільності хмари.

Для реалізації мети робити необхідно вирішити наступні задачі:

- провести дослідження сучасних хмарних систем від різних постачальників хмарних послуг та методів оцінки їх продуктивності;
- провести аналіз, порівняння та вибір моделей хмарних сервісів;
- розробити середовище моделювання хмарних обчислень, яке надає змогу оцінити логічну стабільність хмари за різних конфігурацій;
- провести експерименти по оцінці стабільності та продуктивності хмарних систем.

Об'єктом досліджень є процес управління ресурсами в хмарних системах.

Предмет досліджень: методи моделювання систем хмарних обчислень для оцінки стабільності та продуктивності.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Визначення в галузі хмарних обчислень

Термін «хмара» використовувався багато разів тому, щоб описати особливість використання інформаційних технологій через Інтернет. Хмарні обчислення насправді не є новою технологією. Це не просто нова технологія, а новий спосіб використання. Тому зараз ми знаходимо кілька запропонованих визначень. Кожен намагається знайти визначення, але воно охоплює лише певні аспекти технології. Робота [6] намагається узагальнити їх. Нарешті, пропонується таке визначення: хмари – це множина віртуальних і легкодоступних ресурсів (апаратного та/або програмного забезпечення). Ці ресурси можна динамічно регулювати залежно від усіх потреб. У цій моделі ми платимо лише те, що споживаємо. Національний інститут стандартів і технологій США (NIST) пропонує [7] таке визначення: «Хмарні обчислення – це модель для забезпечення зручного мережевого доступу на вимогу до спільного пулу конфігурованих обчислювальних ресурсів (наприклад, мереж, серверів, сховищ, програми та служби), які можна швидко надати та випустити з мінімальними зусиллями адміністратора або взаємодії постачальника послуг».

Це визначення, яке, здається, охоплює загальноприйняті аспекти хмарних обчислень, було прийнято більшістю дослідників. Важливо узгодити одне визначення, щоб обмежити обсяг дослідження та підкреслити потенційні переваги для бізнесу.

Іншими словами, головну причину існування цих різних визначень можна підсумувати тим фактом, що хмарні обчислення насправді не є новою технологією, а є новим впровадженням уже існуючих технологій для ведення бізнесу по-новому. Хмарні обчислення базуються на кількох старих

концепціях, таких як [1 і 8]: сервіс-орієнтована архітектура (SOA), розподілені та мережеві обчислення і віртуалізація.

Хмарні сервіси формують стиль обчислень, у якому широкомасштабовані можливості надаються «як послуга» багатьом клієнтам. Проте, на відміну від попередніх моделей ліцензування IT, ці послуги зазвичай виставляються клієнтам на основі споживання, таким чином перетворюючи традиційну модель капітальних витрат, поширену сьогодні в центрах обробки даних, на модель операційних витрат. Послуги, що надаються хмарними обчислювальними середовищами, сильно базуються на використанні віртуалізації різних типів обчислювальних технологій, але загальною темою є використання Інтернету для задоволення обчислювальних потреб користувачів. Типи обчислювальних технологій, які віртуалізуються, призводять до наступних чотирьох категорій хмарних служб: базові обчислювальні послуги, IaaS, PaaS і SaaS.

Але, як і в будь-якій іншій діяльності, що базується на послугах, незалежно від категорії послуги, вона в основному включає двох учасників: запитувача послуги та постачальника послуги. З точки зору запитувача послуг, система хмарних обчислень надає обчислювальну утиліту, доступну на вимогу. З точки зору постачальника послуг, на системному рівні хмарні обчислення містять надійну систему розподілу, яка раціонально розподіляє вхідні запити на певні комп’ютерні сервери в хмарі для обробки запиту. Будучи швидко еволюціонуючими, хмарні обчислення досі не мають узгоджених стандартів, тому будемо використовувати абстракції дуже високого рівня для сторони постачальника послуг. Буде звернена увага більше на логічну оцінку стабільності хмарних обчислень на основі конфігурацій системи, що буде описано у наступних розділах. Фактична система хмарних обчислень, безумовно, включатиме додаткові компоненти, такі як посередники послуг і хмарні федерації. Перші узгоджують відносини між запитувачами послуг і постачальниками послуг хмари, тоді як другі дуже необхідні для сайтів соціальних мереж, компоненти яких можуть

розміщуватися різними постачальниками хмарних обчислень, щоб оптимально обслуговувати клієнтів у різних географічних місцях [7]. Ці методи відіграють таку ж важливу роль для продуктивності всієї системи, що й ті сервери, які обробляють вхідні запити. Однак вони не є предметом цієї роботи. На рисунку 1.1 показана базова архітектура системи хмарних обчислень.

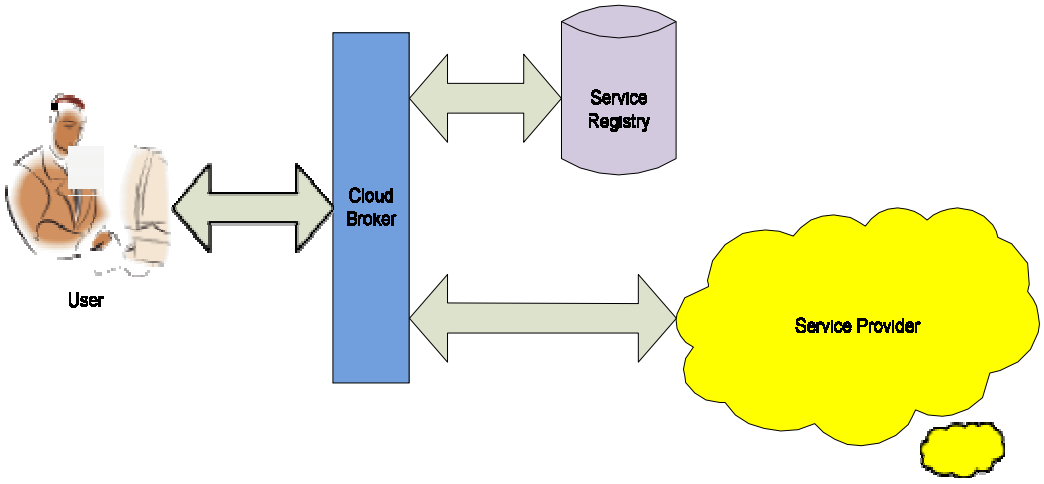


Рисунок 1.1 – Базова архітектура системи хмарних обчислень

Нашу аналітичну модель, представлену у другому розділі, можна просто розглядати як абстракцію, побудовану над постачальником послуг, який завжди містить велику мережу хмарних серверів, які виконують фактичні обчислювальні завдання.

## 1.2 Переваги хмарних обчислень

Є багато переваг, які користувачі послуг можуть отримати від хмарних обчислень. Шість основних переваг узагальнено в [9]:

- зменшена вартість;
- збільшення пам'яті;
- високий рівень автоматизації;

- гнучкість;
- великий ступінь мобільності;
- дозволяє використовувати новітні інформаційні технології.

Тим часом широкомасштабне застосування хмарних обчислень може приносити і додаткові проблеми. Однією з основних проблем переміщення програм у хмару є необхідність оволодіти кількома мовами та операційними середовищами. Крім того, хмарні обчислення викликають питання щодо конфіденційності, безпеки та надійності [10]. Багато дослідницьких питань представлено в [1]. Наприклад, відкриті виклики, пов'язані з даними про походження в хмарі, включають те, як зібрати інформацію про походження стандартизованим і безперебійним способом з мінімальними накладними витратами, як зберігати її постійно, щоб можна було повернутися до неї в будь-який час, і як логічно представити його користувачеві. Серед більшості проблем стабільність системи завжди є однією з головних проблем, тому технічна проблема, яку ми намагаємося розглянути тут, стосується оцінки стабільності системи. Отримані експериментальні результати також можна використовувати для планування потужностей на етапі проектування розробки хмарної системи

## 1.3 Моделі хмарних обчислень

Як було сказано у вступі, хмарні обчислення є відповіддю на нові вимоги в інформаційних технологіях. Модель бізнесу поєднує в собі пропозиції хмарних обчислень для споживачів. Автори в [11] намагаються описати можливі пропозиції. Вони дійшли висновку, що все є послугою, представленою як XaaS, наприклад SaaS (Програмне забезпечення як послуга), PaaS (Платформа як послуга), HaaS (Апаратне забезпечення як послуга), DaaS ([Розробка, база даних, робочий стіл] як послуга), IaaS (Інфраструктура як послуга). Більше прикладів можна знайти в [12]. У цій

моделі найчастіше використовуються три служби: IaaS, PaaS і SaaS [12, 14, 15, 16 і 17], які зображені на рисунку 1.2.

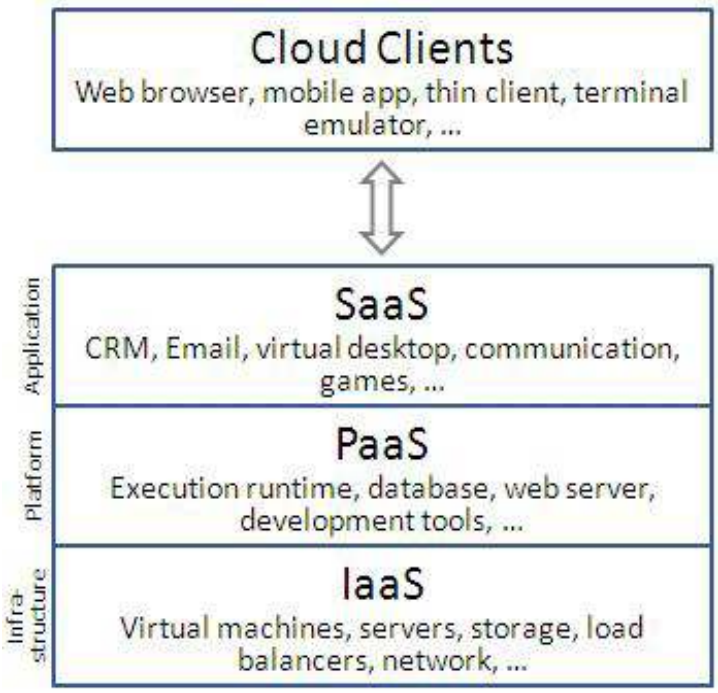


Рисунок 1.2 – Моделі хмарних обчислень

IaaS (інфраструктура як послуга): споживачі безпосередньо використовують IT-інфраструктуру (обчислювальну потужність, мережі, сховище). Ці ресурси надаються через технології віртуалізації. Фізичні ресурси інтегруються або розкладаються відповідно до попиту споживачів. Стратегія віртуалізації полягає у створенні віртуальних машин стільки, скільки потрібно. Таким чином, у цій службі постачальник керує лише ресурсами, а споживачі мають визначати операційну систему та програму, які будуть використовуватися.

PaaS (платформа як послуга): ця послуга надає програмний ресурс, включаючи операційну систему, платформу розробки. Отже, у цьому типі послуги споживач повинен розробляти та керувати лише своєю програмою. Постачальник послуг пропонує споживачеві всі необхідні інструменти, які дозволяють йому запускати свою програму.

SaaS (програмне забезпечення як послуга): припускає надання програм на вимогу через Інтернет. Таким чином, уся система, від апаратного рівня до кінцевої програми, адмініструється та контролюється постачальником послуг. Споживач використовує програму лише тоді, коли йому це потрібно, і йому нічого не потрібно створювати або керувати для виконання своїх потреб.

## 1.4 Модель розгортання хмарних обчислень

Модель розгортання складається в основному з 4 типів, визначених у хмарній спільноті [6, 7 і 16].

Інфраструктура публічної хмари. Постачальник системи цього типу пропонує набір ресурсів (апаратного та/ або програмного забезпечення) як послугу для широкої громадськості. Публічні хмари мають багато переваг, наприклад відсутність початкових капіталовкладень в інфраструктуру. Натомість ця інфраструктура має нижчий контроль системи з боку користувача, що перешкоджає ефективності в багатьох бізнес-сценаріях.

Інфраструктура приватної хмари. Цей тип розгортання використовується для одного користувач (організації). Управління цією системою може здійснюватися як самою організацією, так і третьою стороною. У приватній хмарі користувач має більше контролю над системою. Тому використання цього типу є кращим у бізнесі, особливо при першій інтеграції хмарних технологій.

Гібридна хмарна інфраструктура. У цій системі ми маємо комбінацію іншого типу розгортання хмарних обчислень. Він підходить для бізнесу: приватна хмара для основного використання та публічна хмара, коли потреба зростає. Отже, гібридна хмара може бути використана для оптимізації ресурсів користувачів та постачальників ресурсів залежно від фактичної діяльності (рисунок 1.3).

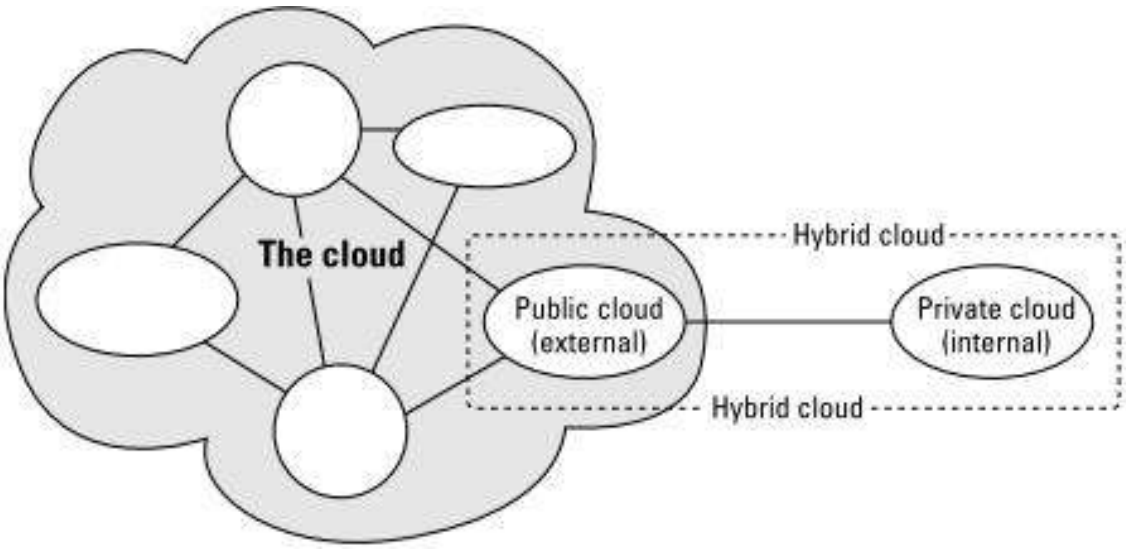


Рисунок 1.3 – Приклад розгортання гібридної хмари

Хмарна інфраструктура у цьому випадку підтримує конкретну спільноту зі спільною функцією чи метою. Вона може бути створений однією організацією або декількома організаціями, які мають взаємні інтереси, як-от місія, політика, безпека. Хмарою спільноти може керувати організація (організації), що входить до складу, або третя сторона.

# 2 МОДЕЛІ ТА МЕТОДИ ОЦІНКИ СТАБІЛЬНОСТІ СИСТЕМ ХМАРНИХ ОБЧИСЛЕНЬ

## 2.1 Оцінка стабільності

Стабільність для системи хмарних обчислень, як і для будь-якої іншої системи, є надзвичайно важливою. Оцінка стабільності необхідна протягом усього життєвого циклу системи, від початкового проектування та впровадження до фактичної експлуатації, управління та розширення. Клієнти не потерплять повільної відповіді від хмарного додатка, оскільки вони платять за його використання на вимогу. Постачальники послуг повинні надавати якісні системи, які можуть повністю використовувати наявні ресурси, одночасно обслуговуючи кожен запит вчасно, щоб залучити більше клієнтів.

Переривання веб-служб Amazon і недоступність Gmail лише на пару годин нещодавно призвели до низки скарг від користувачів, які мали проблеми з доступом до своїх облікових записів, і ці випадки викликали запитання та занепокоєння щодо стабільності системи хмарних обчислень.

Для великої системи хмарних обчислень існують численні фактори, які в основному визначають її стабільність, наприклад, стан програмного та апаратного забезпечення, що підтримує систему. Один виняток в операційній системі або короткочасна несправність мережевої карти може привести всю систему в нестабільний стан. Цей вид проблеми стабільності, викликаної несправностями або втратою обладнання, можна визначити як проблему фізичної стабільності. В кваліфікаційній роботі приділяється більше уваги логічній стабільності системи, тобто проблемам стабільності, викликаних різними конфігураціями системи з точки зору деяких основних показників, таких як швидкість надходження запитів, кількість серверів у хмарі та обчислювальна потужність кожного сервера та розглядається, як

використовувати симуляцію для його оцінки.

Також, як зазначено вище, для системи хмарних обчислень, чия обчислювальна потужність фіксована, стабільність має вирішальне значення, оскільки негативні наслідки можуть накопичуватися. Зазвичай це відбувається, коли поточна запитана обчислювальна потужність перевищує доступну потужність служби. Одним із прямих результатів є те, що ці запити не можуть бути надані негайно, і їм потрібно чекати. Якщо кількість запитів продовжує зростати з такою ж або навіть більшою швидкістю, система хмарних обчислень не зможе своєчасно обробляти вхідні запити, і кількість запитів, які очікують або ніколи не обслуговуються, зростатиме. Ми говоримо, що система в цій ситуації не стабільна. Ця ситуація погіршується, коли запити, що очікують, надсилаються повторно. Для існуючої системи ми можемо оцінити її стабільність, переглянувши її файли журналу та відстежуючи її продуктивність. Точність оцінки залежить від зібраних даних і обмежена наявними ресурсами. Насправді моніторинг хмарних систем також є активною темою для досліджень. Величезний розмір хмарного центру обробки даних і велика кількість його вузлів вимагають надійної системи моніторингу, щоб активно реагувати на збої [11]. Було б краще заздалегідь передбачити стан стабільності, ніж реагувати, коли дійсно виникає збій. Однак у період планування створення або розширення великої хмарної системи, як правило, у нас немає фізичних параметрів, за якими слід стежити, або ми не можемо дозволити собі використовувати реальну платформу для оцінки її стабільності. У цих випадках доводиться вдаватися до моделювання. Моделювання робить можливим повторювані результати та використання різних сценаріїв в обчислювальному середовищі.

## 2.2 Визначення системи

Незважаючи на те, що існує багато різних систем хмарних обчислень, які можуть використовувати різні стандарти та методи, базові архітектури та

процедури обробки фактичних постачальників послуг не сильно відрізняються і можуть бути розумно змодельовані, як описано в [5] для цілей теоретичного аналізу: розглянемо групу комп'ютерних серверів, організованих для співпраці в наданні послуги хмарних обчислень для потоку потоку повідомлень, які надходять у систему випадково, що означає виконання запитів/програм. Передбачається, що кожен сервер має однакову максимально можливу швидкість обчислення в операціях з плаваючою комою за секунду (FLOPS). Повідомлення можуть бути різних типів, і кожен тип повідомлення має визначену ємність обслуговування та вимагає певного середнього часу виконання.

Обробка повідомлень, що надходять, потребує певної потужності комп'ютера і буде розподілена на сервера та розпочне виконання негайно, якщо доступна необхідна потужність хмари. В іншому випадку його буде поставлено в чергу, де повідомлення буде чекатиме певний час, а потім повторить спробу обслуговування. Одночасно одне повідомлення може перебувати лише на одному хмарному елементі (комп'ютерному сервері). Коли швидкість надходження повідомлень перевищує пропускну здатність системи хмарних обчислень, черга матиме тенденцію до нескінченного зростання, і система визначається як нестабільна або насичена. На рисунку 2.1 показано, як наведені вище повідомлення обробляються на стороні постачальника послуг.

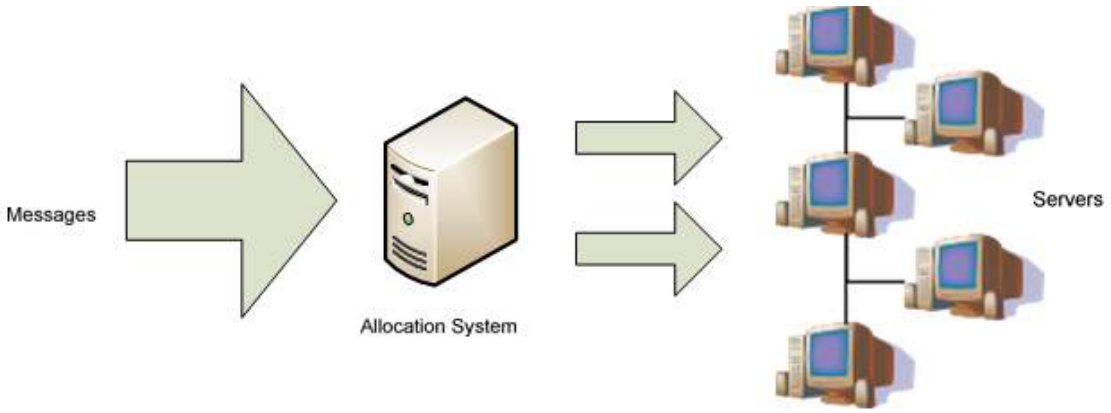


Рисунок 2.1 – Обробки повідомлень постачальником хмарних послуг

Базуючись на теоретичній моделі попереднього розділу, опишемо програму для моделювання роботи системи надання послуг хмарних обчислень.

Нижче наведено кілька необхідних припущень. Ми припускаємо, що всі сервери в хмарі ідентичні, тобто мають однакову обчислювальну потужність. Крім того, ми припускаємо, що існує щонайбільше два типи повідомлень, які надходять згідно з процесом Пуассона з заданими швидкостями. Припускається, що час їх обслуговування відповідає експоненціальному розподілу із заданими середніми значеннями. Ми також припускаємо, що політика вибору серверів є без заміни, тобто прибуле повідомлення спочатку випадковим чином вибирає один сервер для запиту обробки; якщо запит відхилено, повідомлення запитує послугу з інших серверів у випадковому порядку; якщо немає доступного сервера для спроби, відхилене повідомлення поміщається в чергу. Фактична система хмарних обчислень може мати набагато більше типів повідомлень, які надходять відповідно до інших і більш складні політики можуть існувати не лише для вибору серверу, а також для прийняття рішення про те, як надсилати повідомлення на конкретний сервер.

## 2.3 Розроблення моделі надходження завдань в хмарну систему

За основу візьмемо модель з теорії масового обслуговування (дисципліні в рамках математичної теорії ймовірностей), з назвою M/M/1, яка представляє довжину черги в системі з одним сервером. Час надходження повідомлення визначається процесом Пуассона, а час обслуговування завдань має експоненціальний закон розподілу. Назва моделі написана нотацією Kendall. Ця модель є найелементарнішою з моделей масового обслуговування і є привабливим об’єктом дослідження, оскільки в цій моделі можна отримати вирази закритої форми для багатьох цікавих метрик. Розширенням цієї моделі з більш ніж одним сервером є модель M/M/c.

Модель M/M/1 характеризується стохастичним процесом, чий простір станів – це набір {0,1,2,3,...}, де значення відповідає кількості клієнтів у системі, включно з будь-якими, які зараз обслуговуються.

Надходження відбуваються зі швидкістю $\lambda$ відповідно до процесу Пуассона та переміщують процес із стану $i$ в $i+1$.

Час обслуговування має експоненціальний розподіл із параметром $\mu$ у черзі M/M/1, де $1/\mu$ є середнім часом обслуговування.

Усі моменти прибуття та час надання послуг (зазвичай) вважаються незалежними один від одного.

Один сервер обслуговує клієнтів по одному з початку черги відповідно до дисципліни «першим прийшов, першим обслужено». Після завершення обслуговування клієнт виходить з черги, і кількість клієнтів у системі зменшується на одного.

Буфер черги має нескінченний розмір, тому кількість клієнтів, які він може містити, не обмежена.

Модель можна описати як неперервний у часі ланцюг Маркова, що описує безперервний стохастичний процес, у якому для кожного стану процес змінюватиме стан відповідно до експоненціальної випадкової змінної, а потім переходитиме до іншого стану, як визначено ймовірностями стохастичної матриці. Еквівалентне формулювання описує процес як зміну стану відповідно до найменшого значення набору експоненційних випадкових змінних, по одній для кожного можливого стану, до якого він може перейти, з параметрами, визначеними поточним станом. Тоді матриця швидкості переходів виглядає наступним чином:

$$P=\begin{vmatrix} -\lambda & \lambda & 0 & 0 & \cdots & 0 \\ \mu & -(\mu+\lambda) & \lambda & 0 & \cdots & 0 \\ 0 & \mu & -(\mu+\lambda) & \lambda & \cdots & 0 \\ & & \cdots & & & \\ 0 & 0 & 0 & 0 & \cdots & \lambda \end{vmatrix}. \qquad (2.1)$$

Це такий же безперервний часовий ланцюг Маркова, як і в процесі народження-смерті. Діаграма простору станів для цього ланцюга наведена на рисунку 2.1.

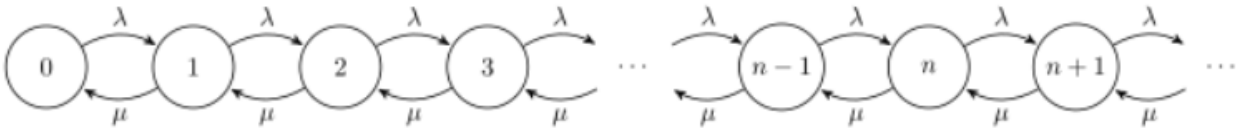


Рисунок 2.1 – Діаграма простору станів

Слід зазначити, що модель вважається стабільною, лише якщо $\lambda < \mu$. Якщо, в середньому, надходження відбуваються швидше, ніж завершення обслуговування, черга зростатиме нескінченно довго, і система не матиме стаціонарного розподілу. Стаціонарний розподіл є граничним розподілом для великих значень t.

Різні показники продуктивності можуть бути обчислені явно для черги M/M/1. Ми пишемо $\rho = \lambda/\mu$ для використання буфера та вимагаємо $\rho < 1$, щоб черга була стабільною. $\rho$ представляє середню частку часу, протягом якого сервер зайнятий.

Імовірність того, що стаціонарний процес знаходиться в стані $i$ (містить і клієнтів, включаючи тих, що обслуговуються), становить

$$\pi_i = (1-\rho)\rho^i . \qquad (2.2)$$

Бачимо, що кількість споживачів у системі геометрично розподілена з параметром $1 - \rho$. Таким чином, середня кількість клієнтів у системі дорівнює $\rho/(1-\rho)$, а дисперсія кількості клієнтів у системі дорівнює $\rho/(1-\rho)^2$. Цей результат справедливий для будь-якого режиму обслуговування, що зберігає роботу, наприклад спільного використання процесора.

Період зайнятості – це період часу, який вимірюється між моментом, коли клієнт приходить у порожню систему, до моменту, коли клієнт залишає порожню систему. Період зайнятості має функцію щільності

$$f(t)=\begin{cases}\dfrac{1}{t\sqrt{\rho}}e^{-(\lambda+\mu)t}I_1(2t\sqrt{\lambda\mu}), & якщо\ t>0\\ 0, & якщо\ t\le 0\end{cases}, \qquad (2.3)$$

де $I_1$ – є модифікованою функцією Бесселя першого роду.

Середній час відповіді або час перебування (загальний час, який клієнт проводить у системі) не залежить від дисципліни планування та може бути обчислений за допомогою закону Літтла як $1/(\mu - \lambda)$. Середній час очікування становить $1/(\mu - \lambda) - 1/\mu = \rho/(\mu - \lambda)$. Розподіл часу відповіді залежить від дисципліни планування.

Для клієнтів, які приходять і залишають чергу як стаціонарний процес з дисципліною планування FIFO, час перебування в системі, який вони відчувають (сума часу очікування та часу обслуговування) дорівнює $(\mu - \lambda)/(s + \mu - \lambda)$, а отже функція щільності:

$$f(t)=\begin{cases}(\mu-\lambda)e^{-(\lambda+\mu)t}, & якщо\ t>0\\ 0, & якщо\ t\le 0\end{cases}. \qquad (2.4)$$

Оскільки програма моделювання забезпечує гнучкість налаштування значень більшості параметрів, ми можемо дозволити їй приблизно імітувати роботу стандартної системи масового обслуговування M/M/1, фіксуючи номера серверів постійними для одного типу повідомлень.

На практиці хмарна інфраструктура, так і програмне забезпечення пропонуються на віртуальних платформах третіми сторонами, а компанії-клієнти укладають контракти на послуги, такі як зберігання або обробка даних. Іншою можливістю є використання онлайн-систем, що замінюють

необхідність внутрішньої розробки чи придбання рішень. Питання стабільності в обох випадках різняться.

Теоретично існують добре обґрунтовані математичні рішення для розрахунку вимірювань, таких як середня довжина черги, середня кількість запитів у системі, середній час, проведений у системі, середній час, проведений у черзі, і так далі, щодо до середньої швидкості прибуття та середньої швидкості обслуговування в системі M/M/1. Як результат, M/M/1 добре підходить для перевірки результатів моделювання.

# 3 ІНТЕГРАЦІЯ МОДЕЛІ В ХМАРНЕ СЕРЕДОВИЩЕ ТА ПРОВЕДЕННЯ ЕКСПЕРИМЕНТІВ

В цьому розділі наведено результати експериментів Основна мета – показати статистику продуктивності та вартість запуску програм з різними моделями навантаження. При розробці програмного забезпечення для моделювання процесів в системі, було реалізовано гнучке налаштування параметрів та характеристик, що надало можливість достатньо точно моделювати роботу стандартної хмарної системи, яка отримує завдання за законами масового обслуговування M/M/1.

## 3.1 Конфігурація експериментального кластеру та методологія оцінки

Моделювання відбувалось на 32 (A10) віртуальних машинах, які працювали на платформі Azure (фірми Microsoft) з 256 ядрами. Таблиці 3.1 містить конфігурацію для кожної віртуальної машини.

Таблиця 3.1 – Сервіси інфраструктури

| Хмарна платформа | Тип | Ядра | Процесор | Пам'ять | Тип пам'яті | Віртуалізація | Мережа |
|---|---|---|---|---|---|---|---|
| Azure | A10 | 8 | IntelXeon E5-2670@ 2.6 GHz | 64 GB | DDR3 | Hyper-V | 10 gigabit ethernet |

Для експериментів було обрано операційну систему CentOS 6.5. Оцінка продуктивності, масштабованості та впливу використання багаторівневого паралелізму проводилася з використанням гібридних кодів MPI+OpenMP.

Робочі навантаження перевіряли ефективність наявного програмного забезпечення управління хмарними обчисленнями, з великим впливом на

продуктивність. Було реалізовано великі робочі навантаження, які дозволяла система управління віртуальними машинами A10 VM.

Характеристики, які використовувалися для оцінки навантаження, це швидкість і MOPS. MOPS фактично надає результати рівня бенчмарку. Також аналізувалось прискорення – співвідношення між часом, який потрібен для послідовного виконання, та часом розподіленого виконання.

Результати оцінки перевірялися шляхом повторних випробувань, щоб збільшити точність оцінки.

## 3.2 Обмеження та налаштування системи моделювання

Експерименти з оцінювання обробки завдань в хмарну систему M/M/1 проводилися з такими обмеженнями:

- надходження є процесом Пуассона ($\lambda$);
- час обслуговування експоненціально розподілено ($1/\mu$);
- є один сервер для обробки одного типу повідомлень;
- довжина черги, в якій надходять запити чекають перед тим, як бути обслугованими, є нескінченною;
- кількість доступних запитів на приєднання до системи нескінченна;
- визначено $\rho = \lambda/\mu$.Тоді ми можемо використовувати таке рівняння, щоб знайти кількість запитів у черзі:

$$N = \frac{\rho^2}{1-\rho} \tag{3.1}$$

Щоб змоделювати систему масового обслуговування M/M/1, ми маємо такі значення параметрів: встановлюємо номер сервера рівним 1, частоту надходження повідомлень рівним 1,0 і час повторної спроби повідомлення рівним 0,5. Зокрема, ми встановлюємо ємність на сервер рівною 5. Запитана ємність дорівнює 5, а час обслуговування становить 0,75 для обох типів

повідомлень. Таким чином, система має лише один тип повідомлень, і в будь-який час лише одне повідомлення може обслуговуватися на сервері, а фактична швидкість обслуговування $\mu$ можна розрахувати як:

$$\mu = 1/0{,}75 = 20/15 \approx 1{,}3333.$$

Тоді така система дуже близька до системи масового обслуговування M/M/1, за винятком того, що кожне повідомлення має час повторення. У стандартній системі M/M/1 час повторення для повідомлення в черзі має бути нульовим, тобто, як тільки сервер стає доступним, повідомлення буде негайно надіслано на цей сервер.

Коли ми маємо параметри, встановлені таким чином, ми можемо запустити кілька моделювань і використовувати фактичні результати для обчислення фактичних значень для $\rho$, а потім розрахувати $N$, теоретична середня довжина черги в довгостроковому періоді.

Зрештою, ми можемо порівняти обчислене $N$ із фактичною середньою довжиною черги на виході. Ступінь близькості покаже, чи справді наша система обробляє повідомлення належним чином, чи програма правильно симулює систему черги.

## 3.3 Інтеграція програми моделювання в систему хмарних обчислень

У цьому підрозділі ми представляємо ідею оптимізації управління системою хмарних обчислень. Інфраструктура як послуга (IaaS) пропонує ресурси як послугу. Як правило, ці ресурси представлені у вигляді віртуальної машини (VM). Користувачам потрібна послуга (тобто набір ресурсів (сховище, обчислювальна потужність, пропускна здатність), і постачальник IaaS намагається задовольнити потреби. Ресурси провайдера зазвичай організовані в різних дата-центрах. Як правило, центр обробки даних складається з кількох фізичних серверів, які взаємопов’язані та

віртуалізовані для оптимізації їх використання. Клієнти можуть розташовуватися в різних місцях. Отже, постачальник послуг, зацікавлений у створенні дата-центрів у різних місцях (рисунок 3.1) .

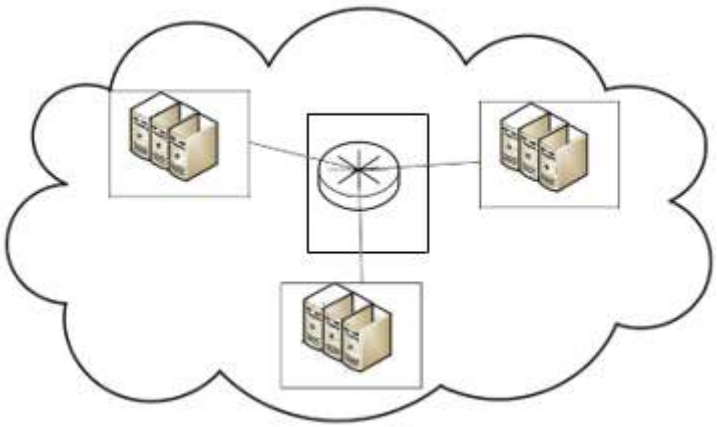

Рисунок 3.1 – Розподіл обробки повідомлень між датацентрами

Реалізуємо ідею, яка полягає в інтеграції багатоагентної системи для керування різними центрами обробки даних, щоб оптимізувати керування системою та максимально відповідати попиту користувачів.

Хмарні обчислення все ще знаходяться на стадії розробки. Фостер описує в [4] поточний стан розподілених систем і способи управління ними. Крім того, у хмарних обчисленнях ми можемо використовувати технології, які використовувалися в грід-обчисленнях. У роботі [17] запропоновано структуру на основі агентів для масштабованості в хмарних обчисленнях. Він зосереджений на тому, щоб міжагент зв'язку задовольнив вимоги користувача, навіть якщо ми використовуємо додаткову послугу іншого постачальника для задоволення запиту. Таким чином, вони намагаються використовувати агентську технологію для оптимізації роботи споживача. Наскільки нам відомо, не було попередніх робіт, які б намагалися мати справу з кількома центрами обробки даних, що належать одному постачальнику.

Дана робота зосереджена на тому, як ефективно використовувати ресурси в різних розташуваннях центрів обробки даних. Ідея полягає в тому, щоб знайти стратегію відповіді на максимальну кількість запитів клієнтів, зберігаючи при цьому високу якість послуг. Як правило, коли клієнт вимагає набір ресурсів, загальною ідеєю є звернути цей запит до найближчого до нього центру обробки даних. Але це рішення не може бути найкращим вибором у всіх випадках, особливо коли є затримка мережі або центр обробки даних перевантажений. Отже, провайдеру потрібна стратегія, щоб вплинути на запит клієнта до найкращого центру обробки даних. У цьому випадку знання глобального стану (прогнозованого стану) системи хмарних обчислень є важливою інформацією, яка впливає на вибір найкращого центру обробки даних для використання залежно від визначених критеріїв. Насправді, щоб отримати цей результат, необхідна система автоматичного зв'язку для забезпечення управління хмарною системою. Він відповідатиме за всі комунікації та переговори між різними центрами обробки даних, щоб мати можливість вибрати найкращий. Отже, мультиагентна система може бути рішенням для керування системою хмарних обчислень і вибору найкращого центру обробки даних за запитом споживача. В роботі використано два типи агентів: Global Cloud Agent (GCA) і Local Agent (LA) (рисунок 3.2).

Для кожного центру обробки даних у хмарній системі створюється локальній агент. Цей агент відповідає за контроль і керування системою локального центру обробки даних. У будь-який момент він надає будь-яку інформацію, яка потрібна Global Cloud Agent, як-от вільні ресурси, стан мережі та ін. Крім того, LA зберігає історію стану центру обробки даних. Отже, за допомогою цієї інформації він може передбачити та надати GCA більше даних, які можуть допомогти краще керувати системою хмарних обчислень. У цьому випадку системи хмарних обчислень локальний агент виконує наглядову роль, а не контролюючу (управлінську) роль.

У системі хмарних обчислень реалізовано лише однин Global Cloud Agent. Цей агент буде відповідати за контроль і управління всією системою.

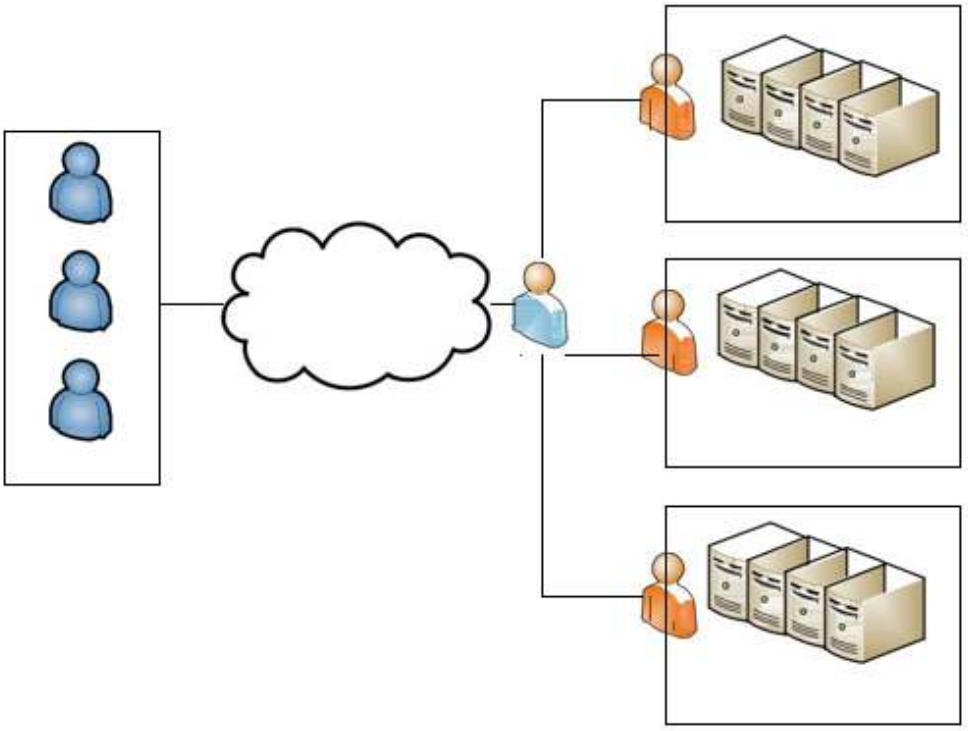

Рисунок 3.2 – Інтеграція агента в систему хмарних обчислень

Для кожного запиту клієнта GCA збирає інформацію від LA, а потім вирішує, який центр обробки даних є найкращим для цього клієнта. Це рішення може бути прийнято відповідно до політики, визначеної постачальником хмарних обчислень.

Агент може вирішувати задачу переспрямування повідомлень як між датацентрами так і між конкретними серверами. В нашому випадку є сформований кластер в середовище Azure с фіксованою кількістю серверів, між яких виконується розподіл завдань.

Наприклад, ми можна вибрати балансування навантаження між центрами обробки даних (серверів) або використовувати центр обробки даних лише тоді, коли попередній буде повністю завантажений. Інша стратегія полягає в тому, щоб спочатку використовувати центр обробки даних з нижчими витратами.

## 3.4 Опис проведення та результатів експериментів

Начальний запуск програми моделювання виконувався десять разів з очікуваною швидкістю надходження λ = 1,0 і часом повторної спроби 0,5. Таблиця 3.2 показує фактичну швидкість прибуття та обчислену середню довжину черги, а також фактичну середню довжину черги. Ми використовуємо *λ,* щоб представити швидкість надходження повідомлень, μ представляти службу обслуговування, $L_1$ і $L_2$, щоб представити середню довжину черги за двома різними способами обчислення. $L_1$-$N$ і $L_2$-$N$ показують відмінності теоретичних та фактично отриманих результатів.

Таблиця 3.2 – Очікувана швидкість прибуття $\lambda = 1.0$, час повтору дорівнює 0.5, у випадку одного сервера (5000 секунд)

| Номер | Фактичний вхід | | Теоретичний Обчислення | | Фактичний вихід | | Різниця | |
|---|---|---|---|---|---|---|---|---|
| | λ | μ | ρ | N | $L_1$ | $L_2$ | $L_1$-N | $L_2$-N |
| 1 | 0,9936 | 1,3416 | 0,7406 | 2,1146 | 2.2008 | 2,1511 | 0,0862 | 0,0365 |
| 2 | 1,0009 | 1,3353 | 0,7496 | 2,2436 | 2,8936 | 2,8366 | 0,6500 | 0,5930 |
| 3 | 0,9924 | 1,3203 | 0,7516 | 2,2749 | 2,7333 | 2,6875 | 0,4584 | 0,4126 |
| 4 | 0,9948 | 1,3237 | 0,7515 | 2,2731 | 2,7957 | 2,7260 | 0,5226 | 0,4529 |
| 5 | 1,0042 | 1,3327 | 0,7535 | 2,3034 | 2,3631 | 2,3020 | 0,0597 | 0,9986 |
| 6 | 1,0184 | 1,3436 | 0,7580 | 2,3736 | 2,1267 | 2,0698 | 0,7531 | 0,6962 |
| 7 | 1,0344 | 1,3429 | 0,7703 | 2,5827 | 2,6725 | 2,6215 | 0,0898 | 0,0388 |
| 8 | 0,9952 | 1,3241 | 0,7516 | 2,2742 | 2,0875 | 2,0299 | 0,1933 | 0,2557 |
| 9 | 0,9872 | 1,3122 | 0,7523 | 2,2852 | 2,1084 | 2,0671 | 0,1232 | 0,3819 |
| 10 | 0,9982 | 1,3488 | 0,7401 | 2,1071 | 2,1316 | 1,8718 | 0,0345 | 0,3647 |
| Середній | 1,00193 | 1,33252 | 0,7519 | 2,2788 | 2,3913 | 2,33633 | 0,1125 | 0,11575 |

З таблиці 3.2 можна побачити, що фактична швидкість прибуття 1,00193 і час обслуговування 1,33252 дуже близькі до встановлених нами (очікуваних) значень λ = 1,0 і μ =1,3333, а вихідна середня довжина черги $L_1$ і $L_2$ лише трохи більше, ніж теоретично обчислене значення. Різниця приблизно: на 0.11 більше. Найважливіше те, що в десяти випадкових прогонах система стабільно видає значення, які знаходяться у фіксованому діапазоні, близькому до результатів теоретичних обчислень.

В наступному експерименті використовуємо ті самі значення параметрів, за винятком іншого часу повторної спроби. Ми зменшуємо час повторення з 0,5 до 0,05. Ми все ще випадково запускаємо програму моделювання десять разів і отримуємо десять наборів результатів, показаних у таблиці 3.3.

Таблиця 3.3 – Очікувана швидкість прибуття $\lambda = 1.0$, час повтору дорівнює 0.05, у випадку одного сервера (5000 секунд)

| Номер | Фактичний вхід | | Теоретичний Обчислення | | Фактичний вихід | | Різниця | |
|---|---|---|---|---|---|---|---|---|
| | λ | μ | ρ | N | $L_1$ | $L_2$ | $L_1$-N | $L_2$-N |
| 1 | 1,0108 | 1,3347 | 0,7573 | 2,3634 | 2,4434 | 2,5117 | 0,08 | 0,1483 |
| 2 | 1,0110 | 1,3121 | 0,7705 | 2,5872 | 2,3938 | 2,4455 | - 0,193 | - 0,142 |
| 3 | 0,9874 | 1,3168 | 0,7498 | 2,2477 | 2,0524 | 2,1183 | - 0,195 | - 0,129 |
| 4 | 0,9938 | 1,3642 | 0,7285 | 1,9546 | 1,7877 | 1,8680 | - 0,167 | - 0,087 |
| 5 | 1,0046 | 1,3455 | 0,7466 | 2.2003 | 1,6393 | 1,7242 | - 0,561 | - 0,476 |
| 6 | 0,9924 | 1,3247 | 0,7492 | 2,2373 | 2,6178 | 2,6802 | 0,3805 | 0,4429 |
| 7 | 1,0126 | 1,3521 | 0,7489 | 2,2337 | 3,0532 | 3,1215 | 0,8195 | 0,8878 |
| 8 | 0,9844 | 1,3284 | 0,7410 | 2,1206 | 2,4230 | 2,4973 | 0,3024 | 0,3767 |
| 9 | 1,0070 | 1,3402 | 0,7514 | 2,2708 | 2,3774 | 2,4401 | 0,1066 | 0,1693 |
| 10 | 0,9998 | 1,3478 | 0,7418 | 2,1312 | 2,5778 | 2,6376 | 0,4466 | 0,5064 |
| Середній | 1,00038 | 1,33665 | 0,7484 | 2,2265 | 2,3366 | 2,40444 | 0,1101 | 0,1779 |

З таблиці 3.2 дуже очевидно, що різниця між фактичним результатом $L_1$ і $L_2$ та $N$ значно менший. Воно зменшується з приблизно 1 повідомлення приблизно до 0,1 повідомлення, дуже близького значення, коли час повторення зменшується з 0,5 до 0,05. Для подальшої перевірки правильності моделювання ми змінюємо очікувану частоту надходження на 0,5, а всі інші значення параметрів залишаються такими ж, як і раніше, тобто така ж очікувана швидкість обслуговування.

Таблиця 3.4 показує фактичні результати, де середня різниця може досягати 0,006, що дуже добре. З колібровкою моделі точність моделювання збільшується.

Таблиця 3.4 – Очікувана швидкість прибуття $\lambda = 0.5$, час повтору дорівнює 0.05, у випадку одного сервера (5000 секунд)

| Номер | Фактичний вхід | | Теоретичний Обчислення | | Фактичний вихід | | Різниця | |
|---|---|---|---|---|---|---|---|---|
| | λ | μ | ρ | N | $L_1$ | $L_2$ | $L_1$-N | $L_2$-N |
| 1 | 0,5082 | 1,2865 | 0,395 | 0,2579 | 0,2235 | 0,2839 | - 0,034 | 0,026 |
| 2 | 0,5190 | 1,3369 | 0,3882 | 0,2463 | 0,2203 | 0,2769 | - 0,026 | 0,0306 |
| 3 | 0,4936 | 1,3434 | 0,3674 | 0,2134 | 0,2183 | 0,2749 | 0,0049 | 0,0615 |
| 4 | 0,4906 | 1,3062 | 0,3756 | 0,2259 | 0,2707 | 0,3303 | 0,0448 | 0,1044 |
| 5 | 0,4864 | 1,3038 | 0,3731 | 0,2220 | 0,2387 | 0,2967 | 0,0167 | 0,0747 |
| 6 | 0,5058 | 1,3197 | 0,3833 | 0,2382 | 0,2687 | 0,3283 | 0,0305 | 0,0901 |
| 7 | 0,5002 | 1,3436 | 0,3723 | 0,2208 | 0,2267 | 0,2723 | 0,0059 | 0,0515 |
| 8 | 0,4864 | 1,3310 | 0,3654 | 0,2105 | 0,1943 | 0,2454 | - 0,016 | 0,0349 |
| 9 | 0,4996 | 1,3127 | 0,3806 | 0,2338 | 0,2559 | 0,3152 | 0,0221 | 0,0814 |
| 10 | 0,4990 | 1,3086 | 0,3813 | 0,235 | 0,2471 | 0,2986 | 0,0121 | 0,0636 |
| Середній | 0,49888 | 1,31924 | 0,3813 | 0,235 | 0,2364 | 0,29225 | 0,0014 | 0,0572 |

Коли швидкість надходження вище, ніж швидкість обслуговування системи, довжина черги повинна очевидно збільшитися. Було зроблено кілька прогонів із такими налаштуваннями та навіть довшим часом моделювання, 10 тисяч секунд, і результати збільшення зміни довжини черги можна побачити з таблиці 3.5.

Таблиця 3.5 – $\lambda > \mu$ у випадку 1 сервера випадку (10 тис. секунд)

| Номер експерименту | λ | μ | λ/ μ | $L_1$ | $L_2$ |
|---|---|---|---|---|---|
| 1 | 1,3490 | 1,3449 | 1,0030 | 156,5855 | 156,5745 |
| 2 | 1,3648 | 1,3151 | 1,0378 | 380,1344 | 380,1154 |
| 3 | 1,3698 | 1,3366 | 1,0248 | 332,4943 | 332,4885 |

З рисунків 3.3, 3.4 та 3.5 можна побачити, що довжина черги значно збільшується при збільшенні значення. Зокрема, чим вище співвідношення $\lambda/\mu$,тим крутіша форма графіку.

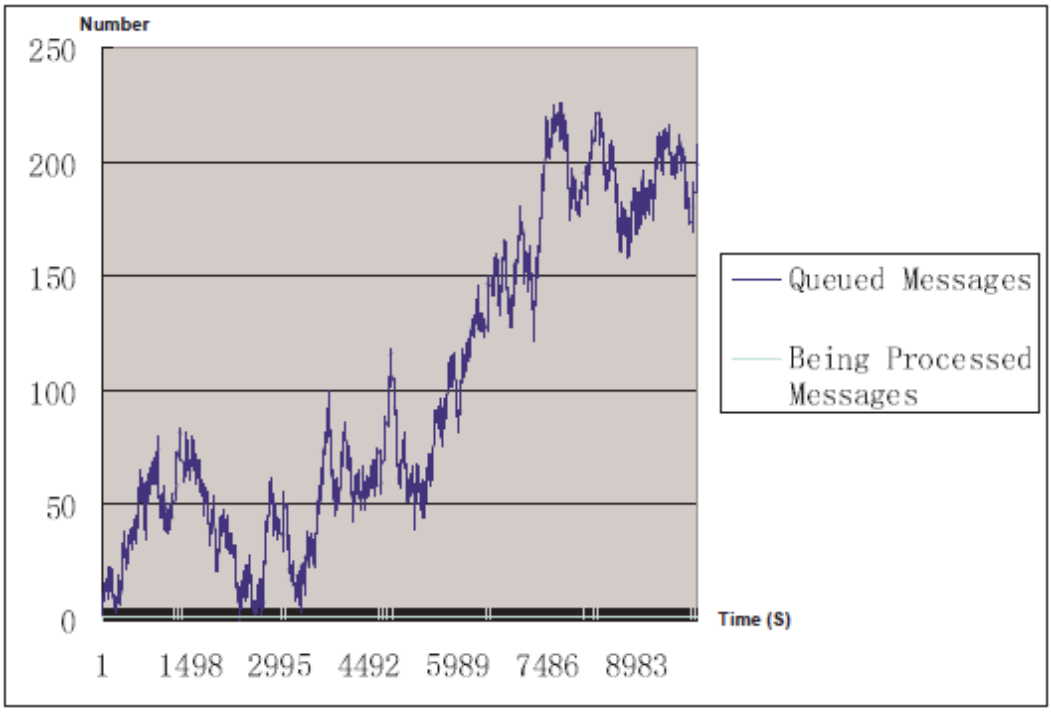


Рисунок 3.3 – Збільшення черги при λ =1,3490 (№1 у таблиці 3.5), за 10 тис. с.

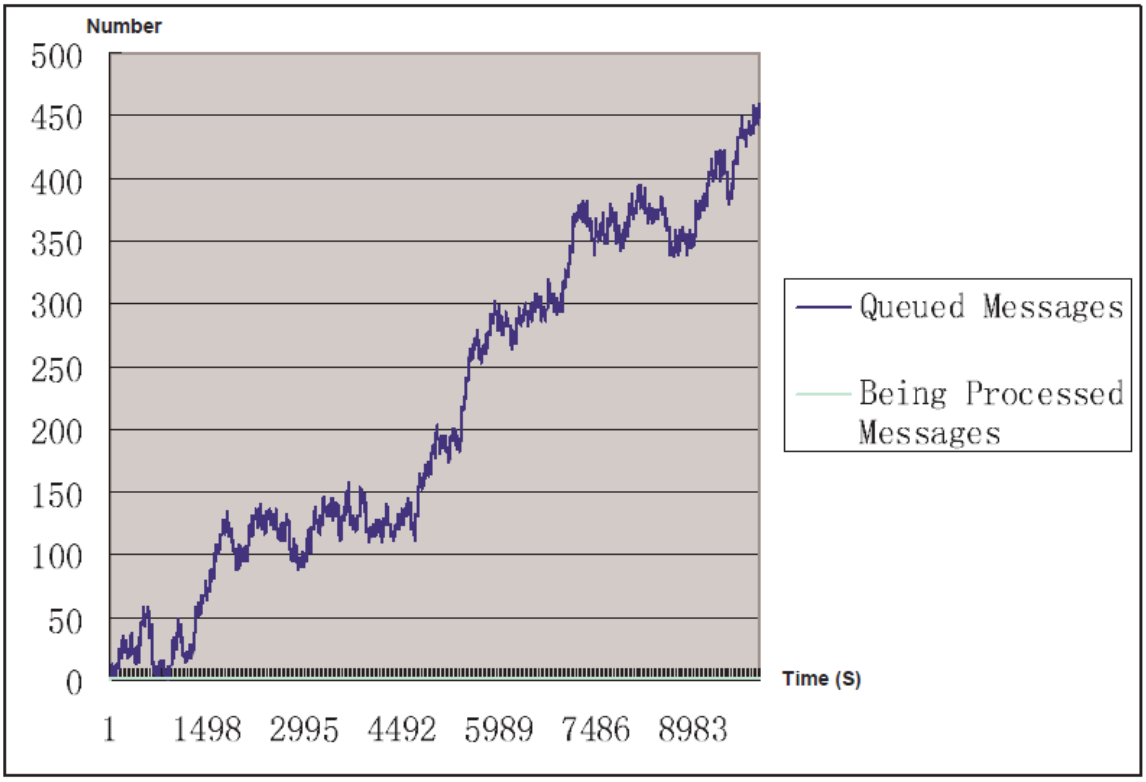


Рисунок 3.4 – Збільшення черги при λ= 1.3648 (№2 у таблиці 3.5), за 10 тис. с.

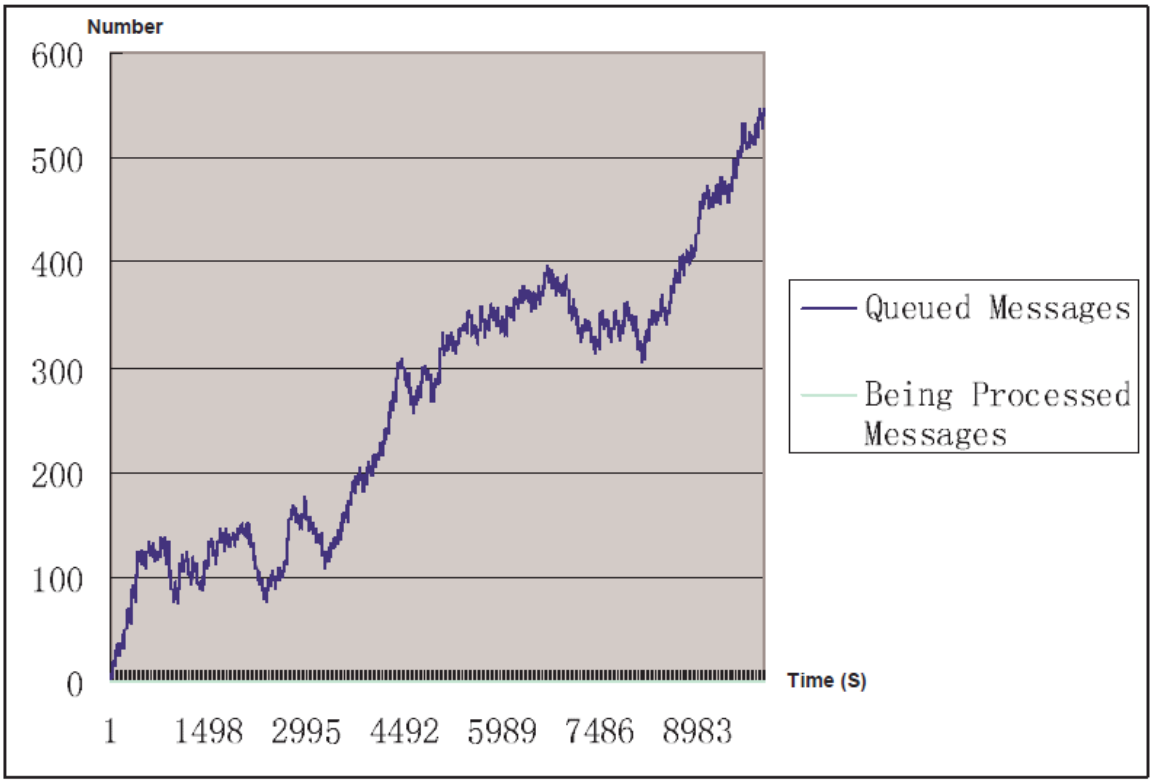


Рисунок 3.5 – Збільшення черги при λ 1.369 (№3 у таблиці 3.5), за 10 тис. с.

Якщо ми скинемо деякі значення параметрів, наприклад, Capacity Per Server на 100, Requested Capacity на 5 і Service Time на 15 для обох двох типів повідомлень, тобто система все ще матиме лише один тип повідомлень, але в будь-який час більше ніж одне повідомлення (насправді це може бути не більше 20 повідомлень) може бути обслуговано на сервері, тоді здається, що фактична швидкість обслуговування $\mu$ можна розрахувати як:

$$\mu = 20/15 = 1/0,75 \approx 1,3333,$$

що дорівнює швидкості обслуговування, яку ми маємо в системі, і яка промодельована раніше. Але результати фактичних прогонів показують, що фактичний рівень обслуговування цієї системи дещо вищий, ніж очікувалось та розраховувалось. Наприклад, оберемо один зразок із частотою надходження вхідних даних 1,34, фактичною частотою надходження 1,3519 і розрахованою швидкістю обслуговування 1,3456, що виконується за 10 тисяч секунд. На рисунку 3.6 показано коливання довжини черги в циклі, де ми не знаходимо такого ж шляху збільшення, як показано на рисунках 3.3, 4.3 і 3.5, навіть якщо ми маємо 1,3519 > 1,3456, чиє співвідношення λ/μ =1,0047, що навіть вище, ніж зразок №1 у таблиці 3.5.

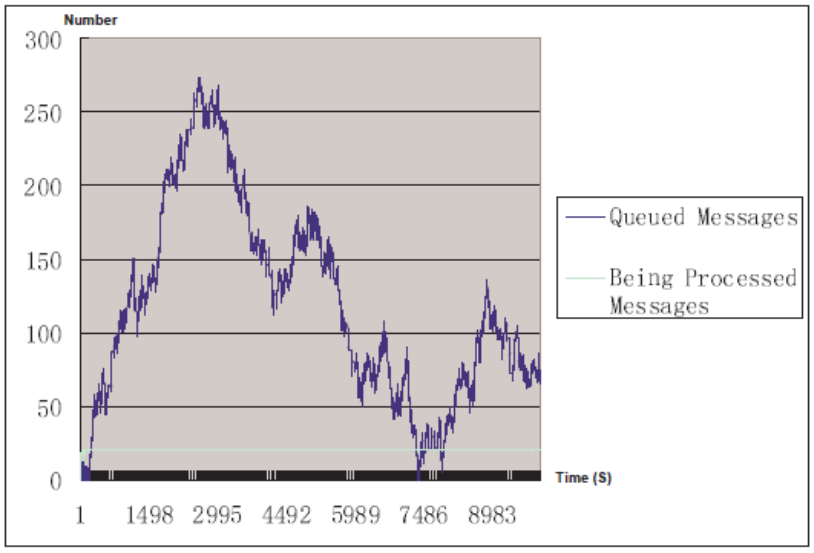


Рисунок 3.6 – $\lambda$ =1,3519, $\mu$ =1,3456,за 10 тисяч секунд

## 3.5 Моделювання у випадку множини серверів

Розглянеми більш складний випадок. Промоделюємо систему з 500 серверами та 2 типами повідомлень, як визначено в таблиці 3.6.

Таблиця 3.6 – Значення параметрів для типів повідомлень

| Повідомлення | Тип 1 | Тип 2 |
|---|---|---|
| Запитувана ємність (FLOPS) | 5 | 10 |
| Середній час виконання/обслуговування (секунди) | 15 | 12 |
| Ймовірність повідомлення | 0.5 | 0.5 |

Кожен сервер має однакову місткість 100 FLOP. Зробимо три прогона циклу моделювання.

Оскільки існує два типи повідомлень, ми не маємо простого способу, як раніше, дізнатися можливу швидкість обслуговування, але тепер ми можемо приблизно визначити її значення на основі результатів моделювання. Ми виявили, що система перебуває в стабільному стані, коли швидкість надходження становить приблизно 510. Фактична швидкість надходження та виходи показані в таблиці 3.7. На рисунку 3.7 показано, що довжина черги дуже мала, коли $\lambda$ менше 510 (що відповідає № 4 у таблиці 3.7).

Таблиця 3.7 – Два типи повідомлень у випадку 500 серверів за 10 тис. і 5 тис. секунд окремо

| Час | Номер | $\lambda$ | $L_1$ | $L_2$ |
|---|---|---|---|---|
| 10000сек. | 1 | 509.8101 | 145.3111 | 145.2435 |
| | 2 | 510.1773 | 143.6217 | 143.6174 |
| | 3 | 510.1916 | 169.2893 | 169.2670 |
| | 4 | 499.9239 | 2.2004 | 2.1804 |
| 5000сек. | 5 | 514.5486 | 5384.8341 | 5384.7288 |

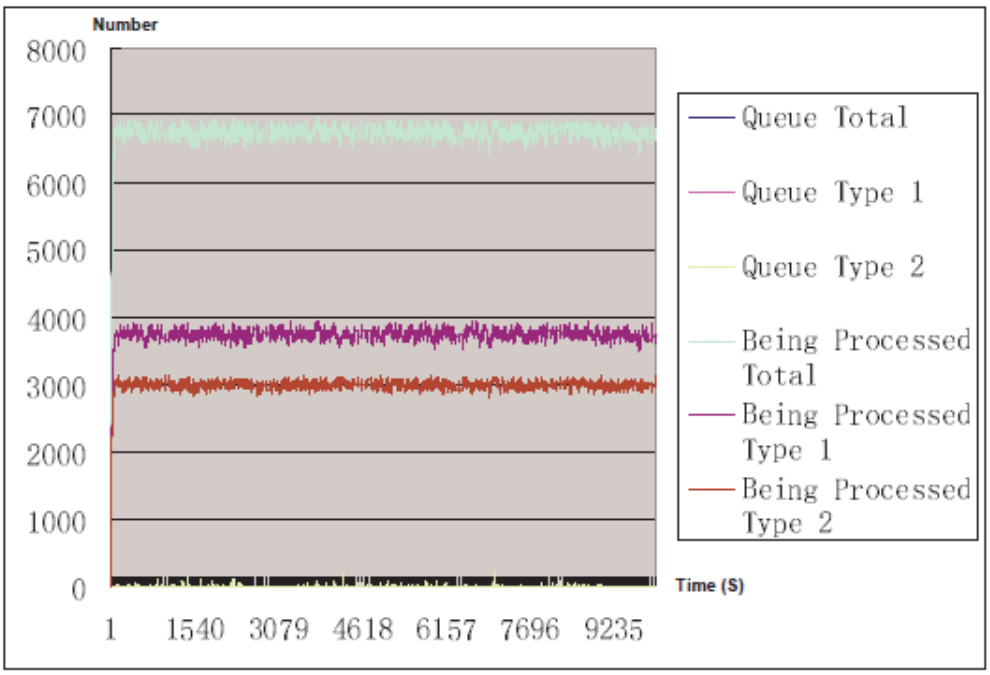


Рисунок 3.7 – $\lambda$ =499.9239 (№4 у таблиці 3.7), 500 серверів, 10 тисяч секунд

Коли λ наближається до 510, ми бачимо, що довжина черги сильно коливається, як показано на малюнках 3.8, 3.9 і 3.10 (що відповідає №1, №2, №3 у таблиці 3.7 відповідно), але система все ще залишається в стабільному стані.

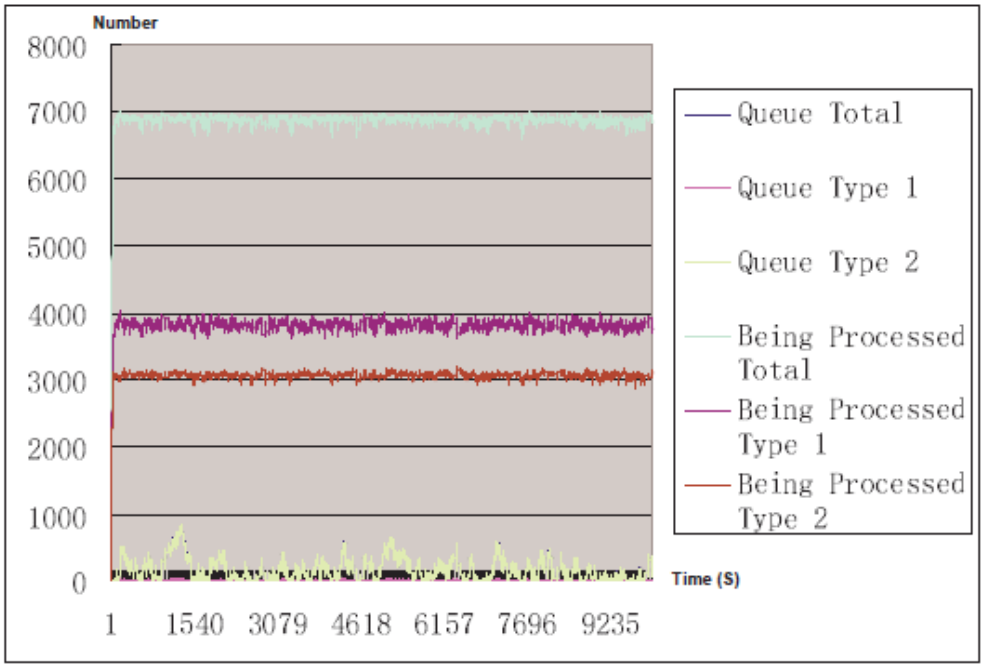


Рисунок 3.8 – $\lambda$ =509.8101 (№1 у таблиці 3.7), 500 серверів, 10 тисяч секунд

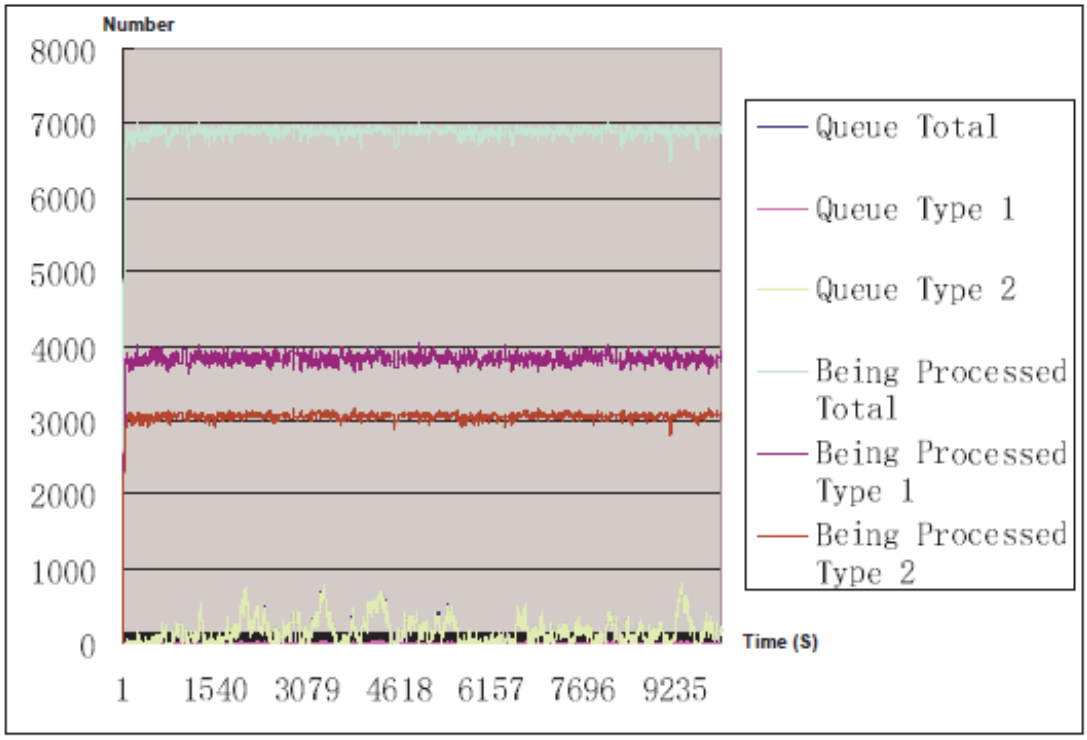


Рисунок 3.9 – $\lambda$ =510.1773 (№2 у таблиці 3.7), 500 серверів, 10 тисяч секунд

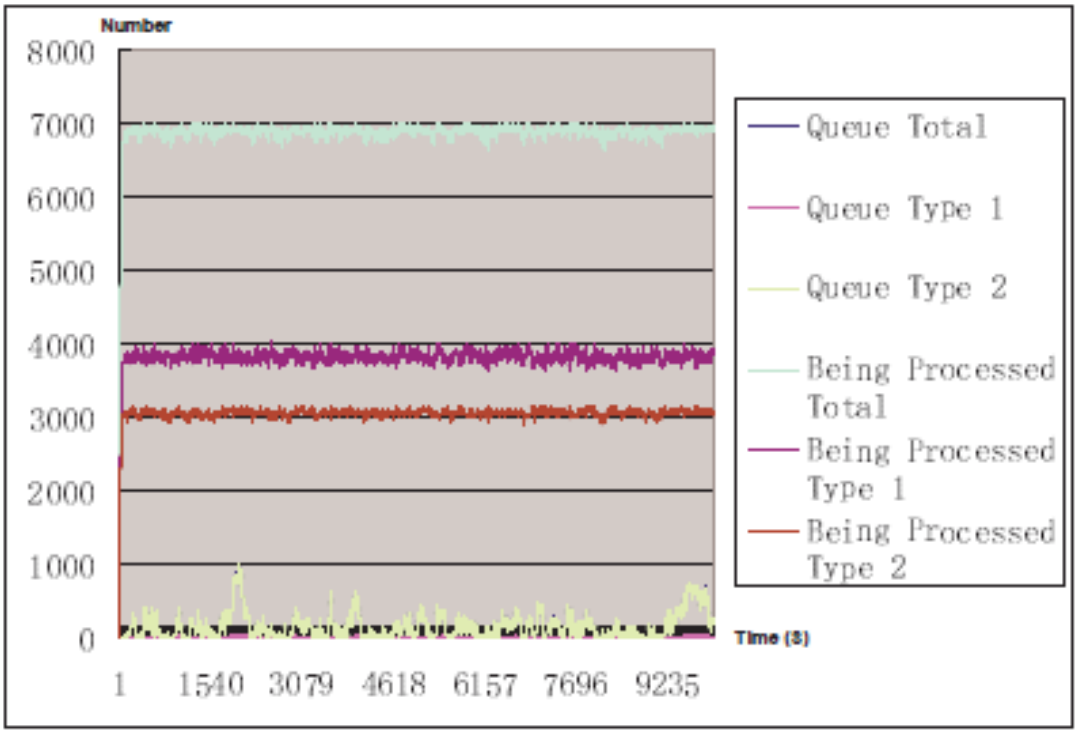


Рисунок 3.10 – $\lambda$ =510.1916 (№3 у таблиці 3.7), 500 серверів, 10 тисяч секунд

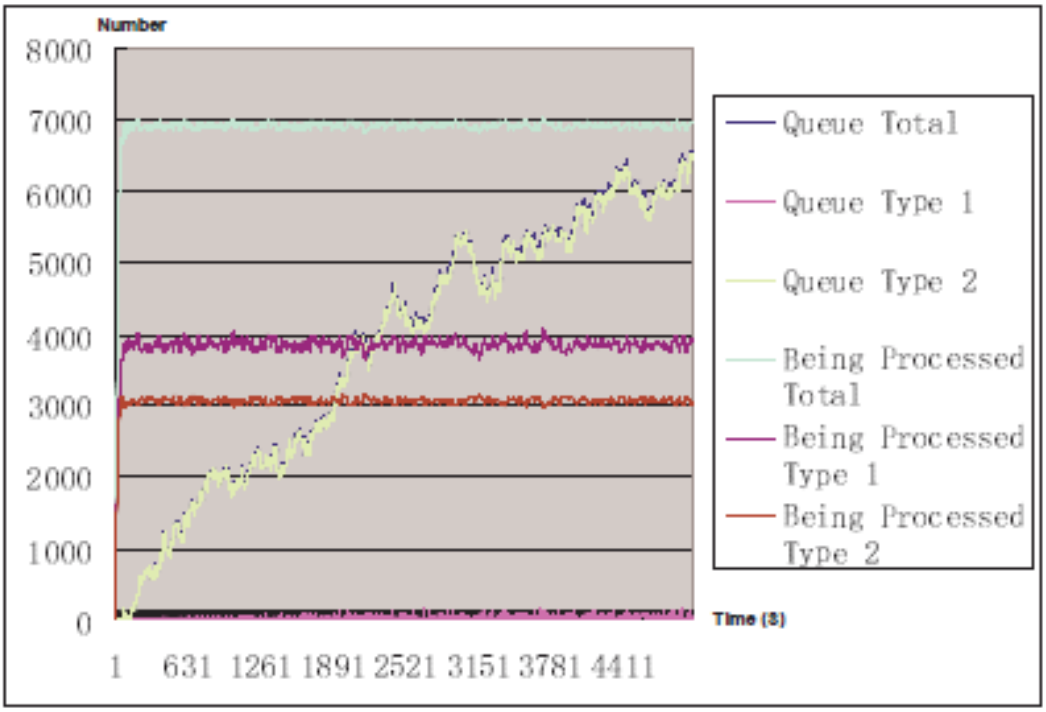


Рисунок 3.11 – $\lambda$ = 514.5486 (№3 у таблиці 3.7), 500 серверів, 10 тисяч секунд

Коли ми вибираємо більше значення для очікуваної швидкості надходження, наприклад, 515, ми можемо побачити, очевидно, що довжина черги прямо зростає зі збільшенням часу, навіть протягом менших 5000 секунд (рисунок 3.11)

# ВИСНОВКИ

У кваліфікаційній роботі представлено метод та програму моделювання для оцінки стабільності системи хмарних обчислень. Оцінка стабільності на їх основі також може допомогти нам знайти фактичний рівень обслуговування.

Програма моделювання забезпечила гнучкість у встановленні значень критичних параметрів. Єдиним обмеженим параметром є кількість типів повідомлень (наразі два). Але зі способу проектування ми бачимо, що розширити програму для більшої кількості типів повідомлень зовсім не важко. На відміну від інших обчислювальних методів у розподіленому середовищі, таких як Grid-обчислення, для яких було запропоновано кілька симуляторів, включаючи GanSim і SimGrid, для підтримки досліджень і розробок, ми не знайшли багато робіт для парадигми хмарних обчислень, особливо для аналізу його стабільності.

Наявні методи базуються на аналізі глобальної системи та намагається змоделювати більшість аспектів системи хмарних обчислень. Замість стабільності ці дослідження більше зосереджені на проблемах масштабованості такої комплексної системи та кількісній оцінці її продуктивності з використанням деякої базової конфігурації, якій бракує природної різноманітності. Крім того, хмарні обчислення все ще є науковою сферою, що швидко розвивається, і існує гостра нестача визначених стандартів. Результати моделювання зазвичай обмежені їхніми власними архітектурами та стандартами та зменшили загальність для інших систем. Крім того, у багатьох роботах вони не надають жодного способу перевірки симулятора, що є критичним для більшості симуляцій. У даній роботі, було зосереджено на спеціальній проблемі, яка менша, але застосовна до різних систем хмарних обчислень, оскільки базова архітектура системного рівня, яка змодельована, не буде сильно відрізнятися.

Ми використали добре відому теорію системи масового обслуговування M/M/1, щоб перевірити правильність моделювання.

Існує ряд шляхів для майбутніх досліджень за допомогою моделювання хмарних обчислень а саме:

- покращити програму моделювання, щоб приймати більше типів повідомлень;
- вивчення можливості постановки в чергу повідомлень щодо їх типів.

За результатами експерименту можна побачити, що тип повідомлень, які потребують більшої ємності, матиме більшу ймовірність потрапити в чергу, навіть якщо їм потрібен менший час виконання. Можна вивчити кореляції між цими двома ознаками, щоб знайти поріг, який має детермінований вплив

Ще одним напрямком досліджень може бути вивчення впливу архітектури системи на її фактичні показники обслуговування. Як було помічено раніше, для однієї серверної системи відношення необхідної ємності для одного повідомлення до загальної ємності сервера, тобто кількість повідомлень, які сервер може обробити за один момент, вплинуло на фактичну швидкість обслуговування системи.

Для проведення експериментів та впровадження в хмарну систему, була використана ідея мультиагентної системи. У майбутніх роботах планується дослідити зв'язок між агентами, а потім запропонувати реалізацію цієї системи, щоб перевірити її реальний потенціал.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. M. A. Vouk. Cloud computing : Issues, research and implementations. In Information Technology Interfaces, 2018. ITI 2018. 30th International Conference on, , June 2018. P. 31–40.

2. L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: Toward a cloud definition. ACM SIGCOMM Computer Communication Review, 39(1). 2019. P.50–55.

3. L. Johnson, A. Levine, and R. Smith. The 2019 horizon report, 2019. Austin, Texas, The New Media Consortium.

4. Foster, Y. Zhao, I. Raicu and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared,” Grid Computing Environments Workshop (GCE '08), 2008.

5. R. Buyya, R. Ranjan and R. N. Calheiros, Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. Proceedings of the Conference on High Performance Computing and Simulation, Leipzig, Germany. IEEE Press: New York, U.S.A., 21–24 June 2019. pp.1–11.

6. L. Vaquero, L. Rodero-Merino , J. Caceres and M. Lindner, A break in the clouds: towards a cloud definition. SIGCOMM Computer Communication Review, Volume 39, Number 1, January 2019. pp.50-55.

7. NIST definition of cloud computing, http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf

8. L. Youseff, M. Butrico and D. Da Silva, Toward a Unified Ontology of Cloud Computing, Grid Computing Environments Workshop (GCE '08), 2008.

9. Six benefits of cloud computing. Available at http://web2.sys-con.com/node/640237.

10. B. Hayes. Cloud computing. Communications of the ACM, 51(7). July

2018. pp.9–11.

11. B. P. Rimal, E. Choi and I. Lumb, A Taxonomy and Survey of Cloud Computing Systems. Fifth International Joint Conference on INC, IMS and IDC. 2019, pp 44–51.

12. Gathering clouds of XaaS! https://www.ibm.com/developerworks/mydeveloperworks/blogs/sbose/entry/gathering_clouds_of_xaas?lang=en

13. B. Sonisky, chapitre 1, cloud computing bible,wiley publishing inc 2011

14. Q. Zhang, L. Cheng and R. Boutaba, Cloud computing : state of the art and research challenges. Journal of Internet Services and Applications, vol. 1, 2020 pp. 7–18.

15. C.N. Hoefer and G. Karagiannis, Taxonomy of cloud computing services. In: Proceedings of the 14th IEEE workshop on enabling the future service-oriented Internet (EFSOI'20), Workshop of IEEE GLOBECOM 2020, pp. 1345–1350.

16. C. N. Hofer and G. Karagiannis, Cloud computing services: taxonomy and comparison, Journal of Internet Services and Applications, Springer, Vol. 2, September 2011, No. 2. pp. 69-79.

17. A. Lenk, M. Klems, J. Nimis, S. Tai and T. Sandholm. What's Inside the Cloud? An Architectural Map of the Cloud Landscape. In ICSE Workshop on Software Engineering Challenges of Cloud Computing, May 2019.

18. L. Wang and G. von Laszewski, Scientific Cloud Computing: Early Definition and Experience," in 20th IEEE International Conference on High Performance Computing and Communications, HPCC'18. 2-18. pp.825-830.

19. Q. Zhang, L. Cheng and R. Boutaba, Cloud computing : state of the art and research challenges. Journal of Internet Services and Applications, vol. 1. 2020. pp. 7–18.

20. Chiang, M. and Zhang, T., 2016. Fog and IoT: An overview of research opportunities. IEEE Internet of Things Journal, 3(6), pp.854-864.

21. Chen, M., Ma, Y., Song, J., Lai, C.F. and Hu, B.,. Smart clothing: Connecting human with clouds and big data for sustainable health monitoring.

Mobile Networks and Applications, 21(5), 2016, pp.825-845.

22. Kumar, V., Laghari, A.A., Karim, S., Shakir, M. and Brohi, A.A., 2019. Comparison of Fog Computing & Cloud Computing. International Journal of Mathematical Sciences and Computing (IJMSC), 5(1), pp.31-41.

23. Jayaraman, P.P., Perera, C., Georgakopoulos, D., Dustdar, S., Thakker, D. and Ranjan, R., 2017. Analytics-as-a-service in a multi-cloud environment through semantically-enabled hierarchical data processing. Software: Practice and Experience, 47(8), pp.1139-1156.

24. Laghari, A.A., He, H., Karim, S., Shah, H.A. and Karn, N.K., 2017. Quality of experience assessment of video quality in social clouds. Wireless Communications and Mobile Computing, 2017.

25. Power, B. and Weinman, J., 2018. Revenue Growth is the Primary Benefit of the Cloud. IEEE Cloud Computing, 5(4), pp.89-94.

26. Benlian, A., Kettinger, W.J., Sunyaev, A., Winkler, T.J. and GUEST EDITORS, 2018. The transformative value of cloud computing: a decoupling, platformization, and recombination theoretical framework. Journal of management information systems, 35(3), pp.719-739.

27. Akherfi, K., Gerndt, M. and Harroud, H., 2018. Mobile cloud computing for computation offloading: Issues and challenges. Applied computing and informatics, 14(1), pp.1-16.

28. Mitra, A., O'Regan, N. and Sarpong, D., 2018. Cloud resource adaptation: A resource-based perspective on value creation for corporate growth. Technological Forecasting and Social Change, 130, pp.28-38.

29. Díaz, M., Martín, C. and Rubio, B., 2016. State-of-the-art, challenges, and open issues in the integration of the Internet of things and cloud computing. Journal of Network and Computer Applications, 67, pp.99-117.

30. Vrable, M., Savage, S. and Voelker, G.M., 2012, February. BlueSky: A cloud-backed file system for the enterprise. In Proceedings of the 10th USENIX conference on File and Storage Technologies (pp. 19-19). USENIX Association.

31. Coppolino, L., D'Antonio, S., Mazzeo, G. and Romano, L., 2017. Cloud

security: Emerging threats and current solutions. Computers & Electrical Engineering, 59, pp.126-140.

32. Marinescu, D.C., 2017. Cloud computing: theory and practice. Morgan Kaufmann.

33. Sultan, N., 2015. The implications of cloud computing for global enterprise management. Global enterprise management (pp. 39-56). Palgrave Macmillan, New York.

34. Galloway, S., 2017. The four: the hidden DNA of Amazon, Apple, Facebook and Google. Random House.

35. Gajbhiye, A. and Shrivastava, K.M.P., 2014, September. Cloud computing: Need, enabling technology, architecture, advantages and challenges. In 2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence) (pp. 1-7). IEEE.

36. Aruna Irani, A.R., Manjula, D. and Sugumaran, V., 2019. Task scheduling techniques in cloud computing: A literature survey. Future Generation Computer Systems, 91, pp.407-415.

37. Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E. and Wilkes, J., 2015, April. Large-scale cluster management at Google with Borg. In Proceedings of the Tenth European Conference on Computer Systems (p. 18). ACM.

38. Varghese, B. and Buyya, R., 2018. Next-generation cloud computing: New trends and research directions. Future Generation Computer Systems, 79, pp.849-861.

39. Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J. and Ghalsasi, A., 2011. Cloud computing—The business perspective. Decision support systems, 51(1), pp.176-189.

40. Laghari, A.A., He, H., Shafiq, M. and Khan, A., 2017, May. Impact of storage of mobile on quality of experience (QoE) at the user level accessing the cloud. In 2017 IEEE 9th international conference on communication software and networks (ICCSN) pp. 1402-1409.

41. Aljamal, R., El-Mousa, A. and Jubair, F., 2018, April. A comparative review of high-performance computing major cloud service providers. In 2018 9th International Conference on Information and Communication Systems (ICICS) (pp. 181-186). IEEE.

42. Ranjan, R., Benatallah, B., Dustdar, S. and Papazoglou, M.P., 2015. Cloud resource orchestration programming: overview, issues, and directions. IEEE Internet Computing, 19(5), pp.46-56.

43. Belgaum, M.R., Soomro, S., Alansari, Z. and Alam, M., 2018. Cloud service ranking using checkpoint-based load balancing in real-time scheduling of cloud computing. In Progress in Advanced Computing and Intelligent Engineering (pp. 667-676). Springer, Singapore.

44. Belgaum, M.R., Soomro, S., Alansari, Z., Musa, S., Alam, M. and Su'ud, M.M., 2017. Challenges: Bridge between cloud and IoT. In 2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICES) (pp. 1-5). IEEE.

45. Nayar, K.B. and Kumar, V., 2018. Cost-benefit analysis of cloud computing in education. International Journal of Business Information Systems, 27(2), pp.205-221.https://trends.google.com/trends/

46. Mohamed, K.S., 2019. IoT Cloud Computing, Storage, and Data Analytics. In The Era of Internet of Things (pp. 71-91). Springer, Cham.

47. Pérez, A., Moltó, G., Caballer, M. and Calatrava, A., 2018. Serverless computing for container-based architectures. Future Generation Computer Systems, 83, pp.50-59.

48. Figiel, K., Gajek, A., Zima, A., Obrok, B. and Malawski, M., 2018. Performance evaluation of heterogeneous cloud functions. Concurrency and Computation: Practice and Experience, 30(23), p.e4792.

49. Graupner, H., Torkura, K., Berger, P., Meinel, C. and Schnjakin, M., 2015, October. Secure access control for multi-cloud resources. In 2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops) (pp. 722-729). IEEE.

50. Joshi, N. and Shah, S., 2019. A comprehensive survey of services provided by prevalent cloud computing environments. In Smart Intelligent Computing and Applications (pp. 413-424). Springer, Singapore.

51. Lynn, T., Rosati, P., Lejeune, A. and Emeakaroha, V., 2017, December. A preliminary review of enterprise serverless cloud computing (function-as-a-service) platforms. In 2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom) (pp. 162-169). IEEE.

52. Laghari, A.A., He, H., Halepoto, I.A., Memon, M.S. and Parveen, S., 2017. Analysis of quality of experience frameworks for cloud computing. IJCSNS, 17(12), p.228.

53. McGrath, G., Short, J., Ennis, S., Judson, B. and Brenner, P., 2016, June. Cloud event programming paradigms: Applications and analysis. In 2016 IEEE 9th International Conference on Cloud Computing (CLOUD) (pp. 400-406). IEEE.

54. Laghari, A.A, RA Laghari, AA Wagan, AI Umrani;2019. Effect of Packet Loss and Reorder on Quality of Audio Streaming, SIS, EAI DOI: 10.4108/eai.13-7-2018.160390

55. Vakili, A. and Navimipour, N.J., 2017. A comprehensive and systematic review of the service composition mechanisms in cloud environments. Journal of Network and Computer Applications, 81, pp.24-36.

56. Islam, N. and Islam, Z., 2018. An economic perspective on major cloud computing providers. arXiv preprint arXiv:1810.05088.

57. Pérez-Arteaga, Pedro, Cristian Castellanos, Harold Castro, Dario Correal, Luis Guzmán, and Yves Denneulin. "Cost Comparison of Lambda Architecture Implementations for Transportation Analytics using Public Cloud Software as a Service." Special Session on Software Engineering for Service and Cloud Computing (2018): 855-862.

58. Dave, D., Meruliya, N., Gajjar, T.D., Ghoda, G.T., Parekh, D.H. and Sridharan, R., 2018. Cloud Security Issues and Challenges. In Big Data Analytics (pp. 499-514). Springer, Singapore.

59. Kotas, C., Naughton, T. and Imam, N., 2018, January. A comparison of

Amazon Web Services and Microsoft Azure cloud platforms for high-performance computing. In *2018 IEEE International Conference on Consumer Electronics (ICCE)* (pp. 1-4). IEEE.

60. Mukiri, R.R. and Prasad, D.B., 2019. Developing Secure Storage of Cloud with IoT Gateway. *Available at SSRN 3446640*.

61. Sharma, S., Chang, V., Tim, U., Wong, J. and Gadia, S., 2016. Cloud-based emerging services systems. *International Journal of Information Management*, pp.1-12.

62. Riti, P., 2018. Introduction to DevOps. In *Pro DevOps with Google Cloud Platform* (pp. 1-18). Apress, Berkeley, CA.

63. Lee, I., 2019. Pricing schemes and profit-maximizing pricing for cloud services. *Journal of Revenue and Pricing Management*, *18*(2), pp.112-122.

64. Jain, A. and Mahajan, N., 2017. Introduction to Cloud Computing. In *The Cloud DBA-Oracle* (pp. 3-10). Apress, Berkeley, CA.

65. Kash, I.A. and Key, P.B., 2016. Pricing the cloud. *IEEE Internet Computing*, *20*(1), pp.36-43.

66. Dewangan, M., Deshmukh, R.K. and Mishra, A., 2018. Comparative Study Between Existing Cloud Service Providers. *International Journal of Advanced Research in Computer Science*, *9*(2), p.537.

67. Safonov, V.O., 2016. *Trustworthy cloud computing*. John Wiley & Sons.

68. Meena, M. and Bharadi, V.A., 2018, August. A Novel Architecture of Hybrid Wavelet Techniques Used by CBIR System for Microsoft Azure Public Cloud SaaS Model. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)* (pp. 1-6). IEEE.

69. Саранча С.М., Якименко А.С., Баляба Ю.В., Сєрих О.О. Методи розподілення віртуальних машин за хмарними ресурсами. Проблеми інформатизації: Матеріали одинадцятої міжнародної науково-технічної конференції. –Баку – Харків – Бельсько-Бяла , 16 – 17 листопада 2023 року, с.50