МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

РУДИЧ Михайло Валентинович

ІНФОРМАЦІЙНА МОДЕЛЬ КОРПОРАТИВНОЇ КОМП'ЮТЕРНОЇ МЕРЕЖІ НА ОСНОВІ СИСТЕМ МАСОВОГО ОБСЛУГОВУВАННЯ / INFORMATION MODEL OF A CORPORATE COMPUTER NETWORK BASED ON MASS SERVICE SYSTEMS

спеціальність: 123 - Комп’ютерна інженерія
освітньо-професійна програма - Комп’ютерна інженерія
Кваліфікаційна робота

Виконав студент групи КІм- 21
РУДИЧ Михайло Валентинович
_____

Науковий керівник
д.т.н., професор Березький О.М.
_____

Тернопіль - 2023

РЕЗЮМЕ

Кваліфікаційна робота на тему «Інформаційна модель корпоративної комп'ютерної мережі на основі систем масового обслуговування» зі спеціальності 123 «Комп’ютерна інженерія» освітнього ступеня «магістр» написана обсягом 70 сторінок і містить 9 ілюстрацій, 6 таблиць, 3 додаткіи та 50 джерел за переліком посилань.

Метою кваліфікаційної роботи є розроблення інформаційної моделі корпоративної комп'ютерної мережі на основі систем масового обслуговування.

Методи дослідження: методи системного аналізу, математичного моделювання, проектування комп’ютерних мереж.

Наукова новизна одержаних результатів. Розроблено інформаційної моделі корпоративної комп'ютерної мережі на основі систем масового обслуговування, що дозволило інтегрувати параметри мережевої інфраструктури з концепціями систем масового обслуговування. Вузол мереж виступає як сервер, обробляючи пакети даних (запити). Пристрої, що ініціюють передачу даних розглядаються як клієнти. Черга відповідає буферу в мережі, де зберігаються пакети, що чекають на передачу або обробку. Частота, з якою вузли мережі генерують пакети даних, відповідає інтенсивності потоку запитів. Здатність вузла мережі обробляти або передавати пакети даних відповідає інтенсивності обслуговування сервера.

КЛЮЧОВІ СЛОВА: СИСТЕМА МАСОВОГО ОБСЛУГОВУВАННЯ, ІНФОРМАЦІЙНА МОДЕЛЬ, КОМУТАТОР, МАРШРУТИЗАТОР, ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ.

RESUME

Qualification work on « Information model of a corporate computer network based on mass service systems » in the specialty 123 "Computer Engineering" educational degree "Master" is written in 70 pages and contains 7 illustrations, 6 tables, 3 appendices and 53 sources by reference.

The purpose of the qualification work is to develop an information model of a corporate computer network based on queuing systems.

Research methods: methods of system analysis, mathematical modeling, design of computer networks.

Scientific novelty of the results. An information model of a corporate computer network based on queuing systems has been developed, which allowed to integrate the parameters of network infrastructure with the concepts of queuing systems. A network node acts as a server, processing data packets (requests). Devices that initiate data transmission are considered clients. A queue corresponds to a buffer in the network where packets waiting for transmission or processing are stored. The frequency with which network nodes generate data packets corresponds to the intensity of the request flow. The ability of a network node to process or transmit data packets corresponds to the intensity of server service.

KEYWORDS: QUEUING SYSTEM, INFORMATION MODEL, SWITCH, ROUTER, SIMULATION MODELING.

ЗМІСТ

ВСТУП

Актуальність. У сучасному світі, де комп'ютерні мережі стають все більш складними та критично важливими для бізнес-операцій, потреба в ефективному моделюванні та аналізі мережевих структур є ключовою для забезпечення їхньої надійності, продуктивності та безпеки.

Системи масового обслуговування (СМО) знаходить широке застосування в комп'ютерних науках та інформаційних технологіях. Наприклад, у мережевих технологіях черги є невід'ємною частиною маршрутизаторів та комутаторів, де пакети накопичуються для передачі. Застосовуючи принципи теорії черг, проектувальники можуть оптимізувати ці системи, забезпечуючи оперативну продуктивність та ефективне використання ресурсів.

Системи масового обслуговування є актуальними для симуляції трафіку та протоколів у мережах, оскільки вони надають потужний інструментарій для аналізу та моделювання різних аспектів мережевої взаємодії. Використання СМО дозволяє точно відтворювати та аналізувати поведінку мережевих ресурсів під час обробки великих обсягів даних, враховуючи варіативність та непередбачуваність мережевого трафіку. Це сприяє оптимізації пропускної спроможності, ефективності протоколів та загальної продуктивності мережі, враховуючи складність сучасних мережевих середовищ.

Метою кваліфікаційної роботи є розроблення інформаційної моделі корпоративної комп'ютерної мережі на основі систем масового обслуговування.

Об'єкт дослідження – процес передачі даних в комп'ютерних мережах.

Предмет дослідження – моделі комп’ютерних мереж на основі систем масового обслуговування.

Поставленої мети роботи можна успішно досягти шляхом виконання таких завдань:

- здійснити аналітичний огляд інформаційних моделей, алгоритмів та засобів імітаційного моделювання комп’ютерних мереж;

- розробити структуру інформаційної моделі на основі систем масового обслуговування;

- розробити структури представлення мережевого обладнання;

- розробити і реалізувати програмне забезпечення імітаційного моделювання мереж.

Наукова новизна одержаних результатів. Розроблено інформаційної моделі корпоративної комп'ютерної мережі на основі систем масового обслуговування, що дозволило інтегрувати параметри мережевої інфраструктури з концепціями систем масового обслуговування.

Практичне значення отриманих результатів. Розроблені алгоритми дозволяють аналізувати та оптимізувати процеси розподілу ресурсів та обробки даних у мережі, що є важливим для управління великими обсягами мережевого трафіку та забезпеченням високого рівня сервісу.

Апробація роботи. Отримані результати опубліковані в межах VIII Науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі», опубліковано тези доповіді [1,2].

Впровадження результатів ДР. Результати роботи рекомендовано до впровадження на підприємстві (додаток Б).

Кваліфікаційна робота складена із трьох розділів, висновків, списку використаних джерел та додатків.

Перший розділ, присвячений моделюванню комп'ютерних мереж, включає огляд основних концептів, літератури та засобів імітаційного моделювання. Це створює фундаментальну базу для подальшої розробки моделі.

Другий розділ фокусується на структурі та реалізації інформаційної моделі мережі, враховуючи різні структури представлення мережевого обладнання та алгоритми систем масового обслуговування. Цей розділ має велике значення для розуміння того, як можна оптимізувати розподіл ресурсів у мережі.

Третій розділ присвячений розробці програмного забезпечення для імплементації розробленої моделі. Він охоплює вибір засобів розробки, структуру проекту та процес розроблення ПЗ. Також включає проведення експериментів для перевірки та валідації моделі.

1 МОДЕЛЮВАННЯ КОМП’ЮТЕРНИХ МЕРЕЖ

1.1 Функції та призначення інформаційних моделей

Інформаційна модель у наукових джерелах України, як і в міжнародному контексті, визначається як упорядкована структура для представлення даних, процесів, правил та відносин в інформаційній системі [4-5]. Ця модель є ключовим елементом у процесі аналізу, проектування та реалізації інформаційних систем. Ось декілька ключових аспектів інформаційної моделі:

1. Структуризація даних. Інформаційна модель визначає, як дані організовані та взаємодіють між собою. Це включає визначення сутностей, їх атрибутів та зв'язків.

2. Представлення процесів**.** Вона може також включати опис процесів обробки даних, включаючи алгоритми, робочі процедури, правила бізнес-логіки тощо.

3. Формалізація. Інформаційна модель дозволяє формалізувати знання та інформацію у структурованому вигляді, що полегшує аналіз, обмін даними, та автоматизацію процесів.

4. Уніфікація підходів**.** Вона сприяє стандартизації та уніфікації підходів до збору, зберігання, обробки та передачі інформації в межах організації або системи.

5. Підтримка рішень. Інформаційні моделі використовуються для підтримки прийняття рішень, планування та управління в інформаційних системах.

Представлення моделі корпоративної комп'ютерної мережі на основі систем масового обслуговування може бути виконане шляхом аналогії між компонентами мережі та елементами системи масового обслуговування. Ось як це може бути зроблено:

Елементи корпоративної комп'ютерної мережі як система масового обслуговування:

1. Вхідний потік (Заявки). У мережі це можуть бути запити від користувачів або внутрішніх систем до серверів, мережевих пристроїв або баз даних. Запити можуть виникати випадково і мати різні характеристики, подібно до клієнтів у системах масового обслуговування.

2. Черга (Лінія очікування). Це може бути буфер у мережевому комутаторі або черга обробки запитів на сервері. Тут запити "очікують" на обробку.

3. Обслуговуючі канали (Сервери). У корпоративній мережі це можуть бути сервери, маршрутизатори або інші мережеві пристрої, які обробляють вхідні запити.

4. Дисципліна обслуговування. В мережі це можуть бути алгоритми керування трафіком або пріоритетність обслуговування в мережевому обладнанні.

5. Час обслуговування. У мережі це відповідає часу, необхідному для обробки запиту сервером або мережевим пристроєм.

Ця модель може бути використана для аналізу та оптимізації продуктивності мережі, наприклад, для визначення оптимальної кількості серверів або потужності обробки для мінімізації часу очікування та забезпечення стабільної роботи мережі.

Розберемо показники продуктивності.

Термін "потужності обробки для мінімізації часу очікування" у контексті корпоративних мереж та обчислювальних систем відноситься до використання достатньої обчислювальної потужності для забезпечення ефективної обробки даних або запитів з метою скорочення часу очікування користувачів або процесів. В англійській мові цей термін можна виразити як "processing capacity for minimizing waiting time" або "computational power to reduce wait times" [2-5].

У цьому контексті, "потужності обробки" може включати кількість і швидкість процесорів, обсяг оперативної пам'яті, швидкість мережевих з'єднань тощо. Це особливо важливо в високонавантажених системах, де велика кількість запитів потребує швидкої обробки для забезпечення високої продуктивності та задоволення потреб користувачів.

Дискретно-подієве моделювання (discrete event simulation) та системи масового обслуговування є тісно пов'язаними, але не ідентичними поняттями. Обидва вони належать до області моделювання та аналізу систем, але їхні фокуси та застосування відрізняються.

Дискретно-подієве моделювання (ДПМ) це метод моделювання, при якому стан системи змінюється тільки у відповідь на події, які відбуваються в конкретні моменти часу. Використовується для моделювання широкого спектру систем, де важливі дискретні події, наприклад, в виробництві, логістиці, телекомунікаціях, комп'ютерних мережах тощо.

Системи масового обслуговування (СМО) це спеціалізований клас моделей, який зосереджений на аналізі процесів обслуговування, де клієнти або заявки обслуговуються в чергах. Використовується для аналізу систем обслуговування, таких як каси в магазинах, телефонні центри, мережеві сервери, системи обробки запитів і т.д. Основна увага приділяється характеристикам черг, дисциплінам обслуговування, часу обслуговування, інтенсивності потоків запитів та іншим факторам, що впливають на ефективність та продуктивність системи.

Зв'язок між ДПМ та СМО:

– дискретно-подієве моделювання часто використовується як методика для моделювання систем масового обслуговування, оскільки воно дозволяє точно відобразити прибуття клієнтів, обробку заявок та інші події в системі;

– обидва підходи дозволяють виконувати аналіз "що-якщо" (what-if analysis) для оцінки реакції системи на зміни в параметрах та умовах роботи.

Отже, хоча дискретно-подієве моделювання та системи масового обслуговування є взаємопов'язаними, вони не є однаковими; перше є більш загальним методом моделювання, тоді як друге є спеціалізованою областю з власним набором теорій та методів [3-8].

Схема комп’ютерної мережі у вигляді системи масового обслуговування.

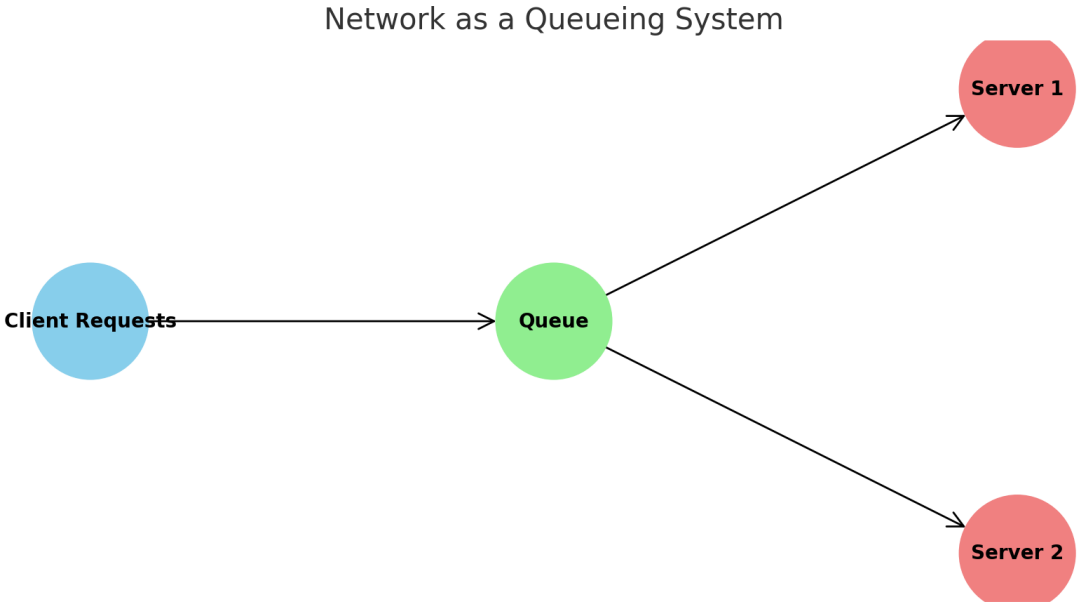

Рисунок 1.1 – Загальне представлення ділянки мережі у вигляді СМО

На цій діаграмі представлено спрощену схему корпоративної комп'ютерної мережі, розглянутої як система масового обслуговування:

– Client Requests зображає вхідний потік запитів або даних від клієнтів до мережі.

– Queue відображає чергу, де запити накопичуються та очікують обробки. Це може бути буфер у мережевому обладнанні або сервері.

– Server 1 та Server 2 представляють обслуговуючі канали - сервери, які обробляють надходження запитів. В системі може бути один або декілька серверів, що працюють паралельно для оптимізації часу обробки та зменшення часу очікування.

Ця модель допомагає у візуалізації та розумінні, як запити обробляються в корпоративній мережі та як можна оптимізувати процеси для підвищення ефективності та продуктивності мережевої інфраструктури.

1.1.1 Види трафіку при моделюванні мереж

При моделюванні комп'ютерних мереж важливо розглядати різні види трафіку, оскільки кожен тип має унікальні характеристики та вимоги до мережі. Різноманітність трафіку може включати використання різних типів протоколів,

режимів роботи протоколів, реакції на кібератаки та інші сценарії. Детальніше розглянемо основні види трафіку:

Розглянемо типи протоколів [2, 10-15].

– TCP (Transmission Control Protocol) трафік: Надійний, забезпечує контроль за доставкою пакетів. Використовується для додатків, яким потрібна точна доставка (наприклад, веб-браузери, електронна пошта).

– UDP (User Datagram Protocol) трафік. Ненадійний, але швидший. Підходить для додатків, де швидкість важливіша за надійність (наприклад, потокове відео, онлайн-ігри).

– ICMP (Internet Control Message Protocol) трафік. Використовується для передачі повідомлень управління та помилок між пристроями в мережі (наприклад, ping-запити).

Розглянемо режими роботи протоколів

– Unicast трафік. Передача даних між одним відправником та одним одержувачем.

– Broadcast трафік. Передача даних від одного відправника до всіх пристроїв у мережі.

– Multicast трафік. Передача даних від одного відправника до багатьох одержувачів, які є членами групи multicast.

Розглянемо аспекти трафіку мережевих кібератак:

– Трафік DDoS (Distributed Denial of Service) відображає навмисне перевантаження мережі великою кількістю запитів, що призводить до відмови в обслуговуванні.

– Трафік «Сканування портів» виникає, коли зловмисник намагається визначити відкриті порти на мережевих пристроях для подальших атак.

– Трафік мережевих черв'яків Виникає, коли самореплікуючі програми (черв'яки) поширюються в мережі, використовуючи її ресурси.

Розглянемо інші сценарії генерації трафіку [15-20]:

– Трафік VOIP (Voice over IP) показує передачу голосових даних через мережу Інтернет. Вимагає стабільності та низької затримки.

– Трафік IoT (Internet of Things) пристроїв часто характеризується низькою пропускною спроможністю та високою кількістю пристроїв.

– Потоковий трафік (Streaming traffic) для потокового відео та аудіо, вимагає високої пропускної спроможності та стабільності з'єднання.

Значення для моделювання. Розуміння цих різних видів трафіку є критично важливим при моделюванні мереж, оскільки кожен вид трафіку ставить унікальні вимоги до мережевої інфраструктури. Наприклад, моделювання мережі, орієнтованої на VOIP, вимагатиме особливої уваги до затримки та стабільності, тоді як мережа для IoT пристроїв вимагатиме підтримки великої кількості одночасних з'єднань з низькою пропускною спроможністю.

У моделюванні мережі також важливо враховувати міру вразливості до кібератак, особливо DDoS-атак та сканувань портів, оскільки це допомагає планувати заходи безпеки та резервування мережевих ресурсів.

Кожен тип трафіку має свої специфічні характеристики, які слід враховувати під час моделювання, щоб забезпечити оптимальну роботу та безпеку мережі.

Моделювання бездротових мереж вимагає особливої уваги до унікальних характеристик та викликів, які вони ставлять перед різними видами трафіку. Ось кілька ключових видів трафіку для моделювання в бездротових мережах:

1. Моделювання мобільного трафіку (Mobile Traffic) з високою густотою користувачів, як це відбувається в міських середовищах або на великих заходах. Особливості: Висока мобільність, переривчасте підключення, різноманітність пристроїв.

2. Wi-Fi Трафік Моделювання офісних мереж або громадських Wi-Fi зон. Особливості: Велика кількість одночасних користувачів, потреба в рівномірному покритті сигналу.

3. Трафік IoT (Internet of Things) це Моделювання мережі смарт-будинків, міських IoT систем. Особливості: Масова кількість невеликих пакетів, висока густота пристроїв, енергоефективність.

4. Трафік Реального Часу (Real-time Traffic). Моделювання VoIP дзвінків або потокового відео. Особливості: Вимоги до низької затримки та високої стабільності підключення.

5. Трафік Віддаленої Роботи (Remote Work Traffic). Моделювання зростаючого використання VPN та хмарних сервісів для віддаленої роботи. Особливості: Захищеність та надійність з'єднань, великі об'єми даних.

6. Екстрений Трафік (Emergency Traffic). Моделювання ситуацій надзвичайних станів, де необхідна швидка комунікація. Особливості: Високий пріоритет, забезпечення стабільності з'єднання в критичних умовах.

1.1.2 Особливості моделювання бездротових мереж.

При моделюванні бездротових мереж слід враховувати наступні фактори:

- Проблеми покриття та сили сигналу при моделюванні розподілу сигналу Wi-Fi або мобільного зв'язку, щоб забезпечити оптимальне покриття.
- Перешкоди та Затінення. Врахування фізичних перешкод, які можуть впливати на якість з'єднання.
- Щільність Користувачів. Моделювання мереж з високою щільністю користувачів, особливо в громадських місцях або офісах.
- Енергоефективність. Особливо важливо для IoT пристроїв, які часто мають обмежену батарею.
- Безпека. Моделювання потенційних загроз безпеці, таких як несанкціонований доступ або атаки.

Моделювання бездротових мереж дозволяє аналізувати та оптимізувати роботу мережі з урахуванням специфічних вимог та умов використання. Це допомагає планувати мережеву інфраструктуру, прогнозувати її продуктивність та забезпечувати необхідний рівень безпеки та надійності.

1.1.3 Функції моделювання трафіку

Формула дискретного розподілу Пуассона використовується для опису випадкових подій, що відбуваються з певною середньою частотою протягом фіксованого періоду часу. Цей розподіл є корисним в багатьох галузях,

включаючи теорію черг, статистику, фінанси та інші. Ймовірність того, що випадкова подія відбудеться рівно *k* разів за певний час, дається формулою:

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!},$$

де:

P(X=k) - ймовірність того, що подія відбудеться рівно *k* разів.

λ - середня кількість разів, коли подія відбувається за фіксований період.

e - основа натурального логарифму, приблизно дорівнює 2.71828.

k!- факторіал *k*.

Якщо, наприклад, в середньому один клієнт приходить у магазин кожні 10 хвилин, то середня кількість клієнтів за годину (λ) буде 6. За допомогою розподілу Пуассона ми можемо обчислити ймовірність того, що за годину прийде, наприклад, рівно 4 клієнти. Ймовірність, що 4 клієнти прийдуть за годину, становить приблизно 0.1339 або 13.39%.

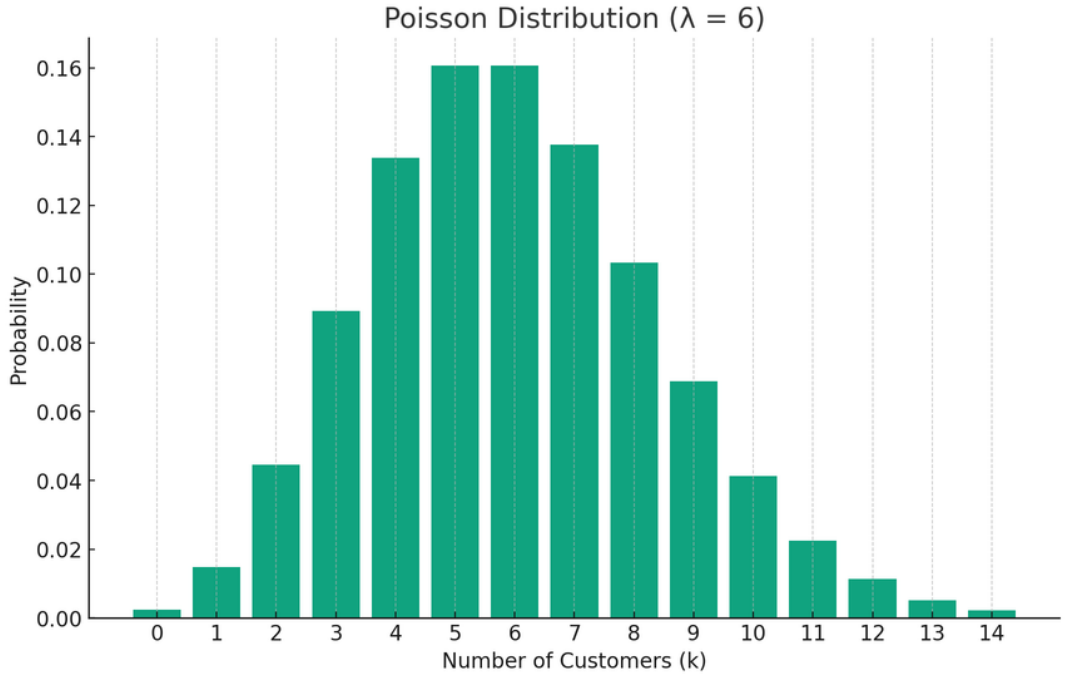

Рисунок 1.2 - Розподіл Пуассона

На графіку вище зображено розподіл Пуассона для середньої кількості подій λ=6. Кожна стовпчик представляє ймовірність того, що в конкретний

часовий проміжок відбудеться певна кількість подій (клієнтів). На графіку видно, що ймовірність того, що прийде близько 6 клієнтів за годину, є найвищою, що відповідає середньому значенню λ. Ймовірності зменшуються для більшої або меншої кількості клієнтів

Цей розподіл є корисним, коли події відбуваються незалежно одна від одної. Проте реальні процеси не завжди підпадають під розподіл Пуассона. Цей розподіл є ідеалізованою моделлю, яка відповідає певним умовам. Для того, щоб процес міг бути адекватно описаний розподілом Пуассона, мають бути виконані наступні умови:

1. Рідкісність подій. Події мають відбуватися з відносно низькою частотою порівняно з можливою кількістю спостережень. Наприклад, розподіл Пуассона використовується для опису кількості аварій на певній ділянці дороги або кількості дзвінків у кол-центр протягом години.
2. Незалежність подій. Кожна подія має відбуватися незалежно від інших.
3. Сталість середньої частоти. Середня частота подій має бути стабільною протягом розглядуваного періоду.

У моделюванні трафіку в комп'ютерних мережах використовують різні розподіли для опису згенерованого трафіку.

Експоненційний розподіл використовується для моделювання інтервалів між подіями (наприклад, прихідом пакетів у мережу). Його функція щільності ймовірності має вигляд:

Функція щільності ймовірності (PDF):

$$f(x;\lambda) = \lambda e^{-\lambda x},$$

де λ - параметр інтенсивності, який визначає середній інтервал між подіями.

Графік експоненційного розподілу:

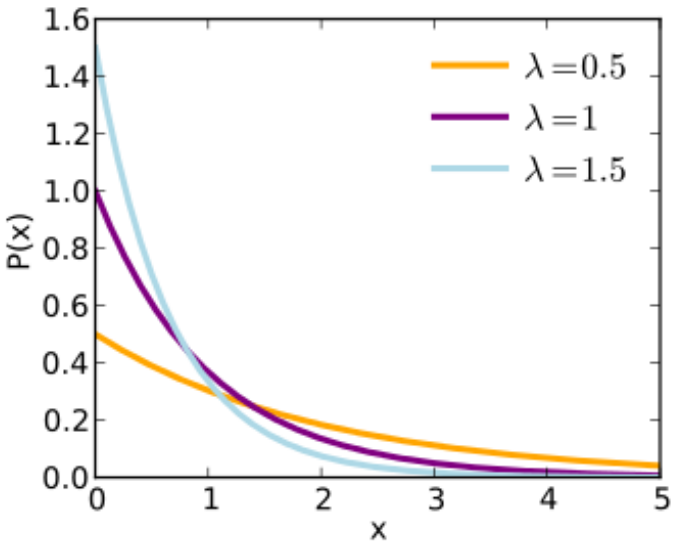

Рисунок 1.3 - Експоненційний розподіл

Нормальний (гаусовий) розподіл може бути використаний для моделювання величин, які мають нормальний розподіл, такі як розміри пакетів або час передачі. Функція щільності ймовірності має вигляд:

$$f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

де μ - середнє значення,

σ - стандартне відхилення.

Графік нормального розподілу:

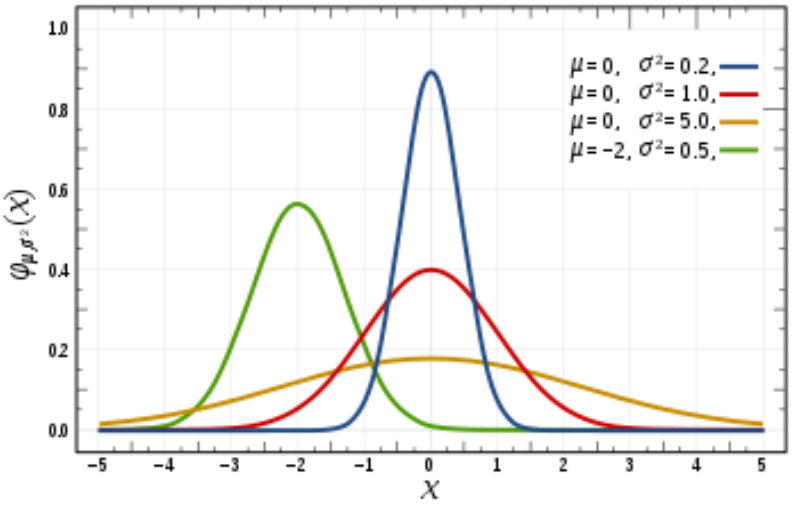

Рисунок 1.4 - Нормальний (гаусовий) розподіл

Графік розподілу Вейбулла (Weibull distribution) часто використовується у статистичному аналізі та інженерії для моделювання різних видів даних, включаючи трафік у комп'ютерних мережах. Цей розподіл характеризується двома параметрами: масштабом (scale) та формою (shape), які визначають форму графіка. Розподіл Вейбулла використовується для моделювання розподілу часу настання подій, який може бути більш або менше експоненційного. Функція щільності ймовірності має вигляд:

$$f(x;\lambda,k) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k},$$

де λ і k - параметри розподілу Вейбулла.

Графік розподілу Вейбулла:

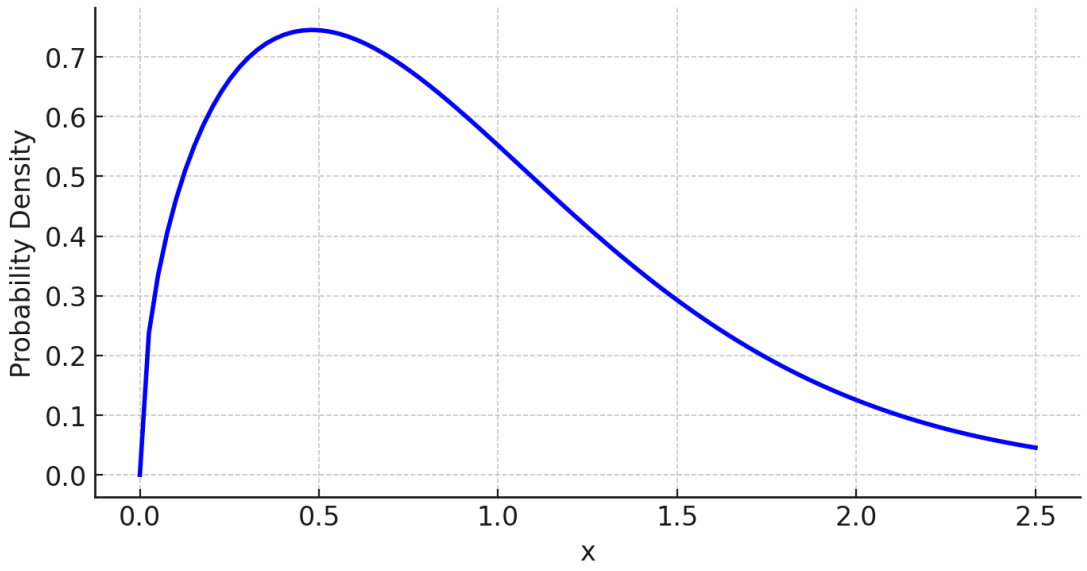

Рисунок 1.5 - Розподіл Вейбулла

Розглянемо, як параметри розподілу Вейбулла можуть бути налаштовані для конкретних вимог моделювання трафіку в комп'ютерній мережі:

Параметр форми часто позначається як k, визначає форму розподілу. Він впливає на "хвіст" розподілу та концентрацію значень довкола моди (найбільш частого значення).

– Якщо k<1, розподіл має важкий хвіст, що може відображати великі сплески трафіку або "пікові" періоди.

– Якщо k=1, розподіл схожий на експоненційний, ідеальний для моделювання випадкових подій.

– Якщо k>1, розподіл має легший хвіст, що може відображати більш стабільний і передбачуваний трафік.

Параметр масштабу (Scale Parameter) λ визначає масштаб розподілу. Цей параметр впливає на розкид даних.

– Велике значення λ вказує на більшу середню величину трафіку і вищу варіативність.

– Менше значення λ вказує на меншу середню величину трафіку і меншу варіативність.

Приклад застосування в аналізі трафіку:

– для моделювання високоваріативного трафіку (наприклад, трафік, що виникає під час особливих подій або атак), можна використати низьке значення для параметра форми та високе для параметра масштабу;

– для моделювання більш регулярного та передбачуваного трафіку, можна встановити вище значення параметра форми та нижче значення параметра масштабу.

Розглянуті нами розподіли можуть використовуватися для моделювання трафіку в комп'ютерних мережах залежно від конкретних потреб і властивостей мережі.

1.1.4 Показники продуктивності мережі

Розглянемо основні показники продуктивності мережі і з точки зору їх представлення в СМО. В Таблиці 1.1 - загальні характеристики продуктивності мережі. В Таблиці 1.2 характеристики з погляду СМО.

Ці таблиці допомагають зрозуміти ключові відмінності між дротовими та бездротовими мережами у контексті їх продуктивності, а також представляють основні метрики, які використовуються для аналізу продуктивності вузлів у мережах з погляду систем масового обслуговування.

Таблиця 1.1 - Загальні характеристики продуктивності мережі

Характеристика	Дротові мережі	Бездротові мережі	Одиниці вимірювання
пропускна здатність Throughput	Вища	Може бути обмежена	Mbps (мегабіти за секунду)
Затримка Latency	Менша	Вища	Мілісекунди (ms)
надійність Reliability	Вища	Може знижуватися	Відсоток (%)
Масштабованість Scalability	Обмежена	Гнучка	Кількість вузлів

Таблиця 1.2 - Характеристики з погляду СМО

Характеристика	Опис	Одиниці вимірювання
інтенсивність потоку запитів (Arrival Rate)	Частота надходження запитів	Запити за одиницю часу
інтенсивність обслуговування (Service Rate)	Швидкість обробки запитів	Запити за одиницю часу
час у системі (Time in System)	Загальний час у системі	Секунди (s)
довжина черги (Queue Length)	Кількість очікуючих запитів	Кількість запитів
вірогідність відмови (Blocking Probability)	Імовірність неможливості обслуговування	Відсоток (%)

Підсумуємо всі типи вузлів у дротових мережах:

1. Маршрутизатори (Routers);
2. Комутатори (Switches);
3. Хаби (Hubs);
4. Мости (Bridges);
5. Шлюзи (Gateways);
6. Брандмауери (Firewalls);
7. Модеми (Modems);

8. Пристрої доступу (Access Points, якщо використовуються для з'єднання з бездротовими мережами).

Типи вузлів бездротових мережах:

1. Бездротові Маршрутизатори (Wireless Routers);
2. Бездротові Доступні Точки (Wireless Access Points);
3. Бездротові Адаптери (Wireless Adapters);
4. Шлюзи для IoT (IoT Gateways);
5. Датчики (Sensors);
6. Бездротові Мости (Wireless Bridges);
7. Бездротові Ретранслятори (Wireless Repeaters);
8. Бездротові Модеми (Wireless Modems).

Наведемо таблиці із ключовими параметрами власне дротових та бездротових мережевих пристроїв .

Таблиця 1.3 - Технічні характеристики мережевих пристроїв

Пристрій	Розмір маршрутних таблиць	Швидкість обробки пакетів	Пропускна здатність	Додаткові характеристики
Маршрутизатор	Залежить від моделі (до декількох тисяч маршрутів)	До мільйонів pps (пакетів за секунду)	Зазвичай 1-100 Gbps	Підтримка різних протоколів маршрутизації
Комутатор	Обмежена (кілька сотень VLAN)	До декількох сотень Gbps	Зазвичай 1-100 Gbps	Висока швидкість локального комутування
Брандмауер	Не застосовується	Залежить від моделі	Залежить від моделі	Забезпечення безпеки мережі
Міст	Обмежена (для локальних мереж)	Висока (залежить від моделі)	Зазвичай 1-10 Gbps	Комутування на рівні кадрів даних

Наведемо таблиці з технічними характеристики бездротових мережевих пристроїв.

Таблиця 1.4 - Технічні характеристики бездротових мережевих пристроїв

Пристрій	Частотний діапазон	Максимальна швидкість даних	Дальність	Додаткові характеристики
Бездротовий Маршрутизатор Wireless Router	2.4 GHz, 5 GHz	До 1-10 Gbps (залежить від стандарту)	До 100-200 метрів	Підтримка різних стандартів (802.11ac, 802.11ax), гостьові мережі
Бездротова точка доступу Wireless Access Point	2.4 GHz, 5 GHz	До 1-10 Gbps (залежить від стандарту)	До 100-200 метрів	Підтримка PoE , множинні SSID
Бездротовий Адаптер Wireless Adapter	2.4 GHz, 5 GHz	Залежить від стандарту (наприклад, 802.11ac до 866 Mbps)	Зазвичай до 100 метрів	USB або PCIe інтерфейси, підтримка WPA3
Шлюз IoT IoT Gateway	Різні (включаючи LoRa, Zigbee)	Залежить від технології	Від кількох метрів до кількох кілометрів	Підтримка різних протоколів IoT, інтеграція з хмарними сервісами
Бездротові Мости Wireless Bridges	2.4 GHz, 5 GHz	До 1 Gbps	Залежить від моделі (до декількох кілометрів)	Використовується для з'єднання двох мереж або розширення мережі
Бездротові Ретранслятори Wireless Repeaters	2.4 GHz, 5 GHz	Залежить від стандарту (наприклад, 802.11ac до 1 Gbps)	Зазвичай до 100-200 метрів	Покращення покриття мережі, зменшення "мертвих зон"
Бездротові Модеми Wireless Modems	Залежить від мережі (наприклад, LTE, 5G)	Залежить від мережі (наприклад, LTE до 300 Mbps, 5G до 1-10 Gbps)	Залежить від мережі та умов зв'язку	Підключення до мобільних мереж, Wi-Fi готспот

Ці таблиці демонструють, як технічні характеристики мережевого обладнання впливають на продуктивність мережі. Реальні характеристики

можуть значно відрізнятися залежно від конкретної моделі та виробника обладнання.

1.2 Огляд літератури

Стаття [46] описує новий метод оцінки ємності пакетного буфера вузлів, використовуючи модель мережі чергування. Цей метод аналізує ємність буфера пакетів для кожного типу вузла, коли він перебуває в найкращому робочому стані.

Для оцінки ситуації з заторами в мережі чергування та отримання реальних ефективних показників швидкостей прибуття та передачі в моделі були додані вузли утримання, і еквівалентна модель мережі чергування була розширена. Була створена модель мережі чергування типу M/M/1/N з вузлами утримання для бездротових сенсорних мереж, а також розроблені приблизні ітеративні алгоритми. Експериментальні результати показують, що модель узгоджується з реальними даними [46].

Модель мережі чергування типу M/M/1/N є однією з фундаментальних моделей в теорії масового обслуговування, яка застосовується для аналізу систем чергування. Ось ключові характеристики цієї моделі:

1. Символи в записі «M/M/1» вказують на тип розподілу часу між прибуттями та обслуговуванням, а також кількість каналів обслуговування. Перший "M" (Markovian) означає, що час між прибуттями клієнтів є експоненційно розподіленим (або марковським). Другий "M" вказує, що час обслуговування також є експоненційно розподіленим. "1" означає, що в системі є лише один канал обслуговування.

2. Символ «**N**» вказує на максимальну кількість клієнтів, які можуть перебувати в системі одночасно, включаючи як клієнта, який обслуговується, так і клієнтів, які чекають у черзі. Коли кількість клієнтів досягає N, система перестає приймати нових клієнтів, доки деякі з них не покинуть систему.

3. Аналіз та Розрахунки. За допомогою моделі M/M/1/N можна розрахувати ряд важливих показників ефективності системи, таких як середній час очікування в черзі, ймовірність відмови (коли система не може прийняти клієнта, бо черга повна), середня кількість клієнтів у системі тощо.

4. Застосування. Модель M/M/1/N часто застосовується для аналізу систем, де простір для черги обмежений, наприклад, в комп'ютерних мережах для аналізу буферів пакетів, в телекомунікаціях для аналізу викликів, що очікують обслуговування, та в інших областях, де важливим є управління обмеженими ресурсами.

Модель M/M/1/N є цінним інструментом для визначення оптимальних параметрів систем масового обслуговування, забезпечуючи важливу інформацію для планування та оптимізації роботи цих систем.

Модель M/M/1/N використовується в комп'ютерних мережах для аналізу буферів пакетів, зокрема для визначення оптимальної ємності буфера, ймовірності втрати пакетів та загальної ефективності мережі. Ось як вона застосовується:

1. Моделювання буферу пакетів. У комп'ютерних мережах, буфер пакетів в маршрутизаторах або комутаторах можна розглядати як систему чергування. Пакети, які приходять у мережевий пристрій, стають у чергу на обробку або передачу. Буфер має обмежену ємність, тому він може вміщати лише певну кількість пакетів. Це відповідає "N" в моделі M/M/1/N.

2. Аналіз ймовірності втрати пакетів. Коли буфер заповнений (тобто в ньому вже є N пакетів), нові пакети, які прибувають, будуть відкинуті або втрачені, оскільки в буфері немає місця. Модель M/M/1/N дозволяє аналізувати ймовірність таких втрат пакетів залежно від швидкості прибуття та обслуговування пакетів.

3. Оптимізація ємності буфера. За допомогою моделі можна визначити оптимальний розмір буфера, що збалансує між потребою мінімізації втрати пакетів та вимогами до фізичного обсягу пам'яті. Це допомагає забезпечити ефективне використання ресурсів мережі.

4. Визначення затримок. Модель також дозволяє оцінювати середній час, який пакет проводить у черзі, що є важливим для оцінювання затримок у мережі. Для критичних за часом додатків, таких як голосовий трафік або відеоконференції, високі затримки можуть бути неприйнятними.

5. Експериментальне підтвердження та тестування. На основі цієї моделі можна проводити експерименти та симуляції для перевірки різних сценаріїв використання мережі, щоб переконатися, що мережева інфраструктура здатна ефективно обробляти вимоги реального трафіку.

Використання моделі M/M/1/N допомагає інженерам та проектувальникам мереж розробляти більш ефективні та надійні мережеві системи, оптимізуючи параметри буфера для забезпечення високої продуктивності та якості обслуговування.

Стаття [47] представляє дослідження у сфері використання теорії черг у контексті транспортного зв'язку (vehicular communication), використовуючи реальні дані трафіку LTE (Long-Term Evolution). Розглянемо його детальніше:

1. Аналіз доцільності для впровадження транспортного Зв'язку: Дослідження зосереджується на аналізі можливості використання системи транспортного зв'язку з використанням теорії черг. Це важливо для розуміння того, як можна оптимізувати комунікаційні процеси у сфері транспорту, особливо в умовах великого обсягу даних та запитів.

2. Модель M/M/m є класичною у теорії черг і використовується для аналізу систем з множинними серверами (тобто каналами зв'язку). Вона дозволяє оцінити ймовірність того, що всі канали будуть зайняті, коли транспортний засіб спробує їх використати.

3. Оцінка часу очікування та кількості перемикань каналів. Дослідження використовує модель M/M/m для визначення очікуваного часу очікування на комунікаційний канал та очікуваної кількості перемикань каналів. Це важливо для розуміння ефективності системи зв'язку та її здатності обробляти високий обсяг запитів.

4. Оптимізація маршрутизації через багатоетапне перенаправлення. Коли базова станція стає перевантаженою, у статті пропонується підхід до оптимізації

маршрутизації за допомогою багатоетапного перенаправлення. Це означає, що замість того, щоб використовувати один прямий канал до базової станції, дані можуть бути перенаправлені через декілька проміжних точок для зменшення навантаження.

Разом ці елементи створюють комплексний підхід до управління транспортним зв'язком, зосереджуючись на оптимізації використання каналів зв'язку та ефективності передачі даних. Транспортний зв'язок - це зростаюча сфера комунікацій між транспортними засобами, включаючи придорожню комунікаційну інфраструктуру [48]. Досягнення в галузі бездротового зв'язку роблять можливим обмін інформацією в режимі реального часу між транспортними засобами та інфраструктурою.

1.3 Засоби імітаційного моделювання комп’ютерних мереж

Аналіз і моделювання комп'ютерних мереж є ключовими елементами в процесі проектування, управління та оптимізації мережевої інфраструктури. Для цих цілей використовуються різні програмні засоби, кожен з яких має свої унікальні функції, переваги та недоліки. Нижче представлена порівняльна таблиця декількох популярних програм для аналізу і моделювання комп'ютерних мереж:

1. Wireshark;
2. Cisco Packet Tracer;
3. GNS3 (Graphical Network Simulator-3);
4. NetSim;
5. OPNET Modeler.

Таблиця 1.5 – ПЗ для моделювання

Критерій / Програма	Wireshark	Cisco Packet Tracer	GNS3	NetSim	OPNET Modeler
Основна функція	Аналіз пакетів	Моделювання мереж	Моделювання мереж	Моделювання мереж та аналіз	Моделювання мереж та аналіз
Інтерфейс	Графічний	Графічний	Графічний	Графічний	Графічний
Основні функції	Захоплення та аналіз пакетів	Створення та налагодження віртуальних мереж	Створення складних мережевих сценаріїв	Симуляція мереж різних типів	Детальне моделювання мереж та аналіз даних
Переваги	Детальний аналіз, Безкоштовний	Легкий у використанні, Ідеальний для навчання	Підтримка реального обладнання	Розширені можливості моделювання	Високий рівень деталізації, Гнучкість у моделюванні
Недоліки	Вимагає технічних знань	Обмежена функціональність	Складний у налаштуванні	Висока вартість	Висока вартість, Складний у вивченні
Типове використання	Моніторинг мережі	Навчання та освіта	Реальні мережеві сценарії	Дослідження та освіта	Корпоративні дослідження

Середовище Wireshark:

- Функції: Аналіз пакетів, фільтрація трафіку, розшифровка протоколів.
- Переваги: Надає дуже детальну інформацію про трафік, безкоштовний.
- Недоліки: Вимагає глибоких знань мережевих протоколів.

Середовище Cisco Packet Tracer:

- Функції: Візуальне моделювання мереж, тестування конфігурацій.
- Переваги: Інтуїтивно зрозумілий, відмінно підходить для навчальних цілей.
- Недоліки: Може бути обмежений для складних мережевих конфігурацій.

Середовище GNS3:

- Функції: Моделювання складних мереж, підтримка реального обладнання.

– Переваги: Дозволяє тестувати реальні мережеві конфігурації.

– Недоліки: Може бути складним для налаштування та вимагає додаткових ресурсів.

Середовище NetSim:

– Функції: Моделювання мереж, аналіз протоколів.

– Переваги: Високий рівень деталізації, підходить для наукових досліджень.

– Недоліки: Висока вартість, може бути складним для початківців.

Середовище OPNET Modeler:

– Функції: Детальне моделювання мережевих сценаріїв, аналіз продуктивності.

– Переваги: Глибокий рівень аналізу та моделювання.

– Недоліки: Висока ціна, складний у вивченні.

Кожен з цих інструментів має свої унікальні характеристики і може бути вибраний в залежності від конкретних потреб користувача. Для глибокого аналізу мережевого трафіку краще використовувати Wireshark. GNS3 є відмінним вибором для моделювання складних мережевих сценаріїв із використанням реального обладнання, тоді як NetSim і OPNET Modeler найкраще підійдуть для детальних наукових досліджень і корпоративного застосування.

Засоби імітаційного моделювання систем масового обслуговування.

У Python є декілька бібліотек, які можуть бути використані для моделювання систем масового обслуговування. Ось декілька прикладів:

1. Бібліотека SimPy **[**12**]**: для дискретно-подієвого моделювання, яка дуже добре підходить для моделювання складних систем масового обслуговування. Вона дозволяє моделювати черги, обслуговуючі процеси та різноманітні стратегії керування.

2. Бібліотека Ciw **[**11**]** спеціально створена для моделювання черг та систем масового обслуговування (discrete event simulation library). Вона дозволяє

визначати мережі черг, специфікувати розподіли часу обслуговування та прибуття, та аналізувати результати моделювання.

3. Бібліотека QueueSim [10] для моделювання черг та систем масового обслуговування в Python, яка надає інструменти для створення та аналізу різних типів черг.

4. Бібліотека Queueing-tool яка дозволяє створювати та аналізувати мережі черг. Вона підходить для моделювання різних сценаріїв чергування та має можливості візуалізації.

1.4 Постановка задач дослідження

Метою дослідження є розробити програмне забезпечення для аналізу та моделювання корпоративної безпровідної комп'ютерної мережі, використовуючи принципи систем масового обслуговування (СМО). Програма повинна дозволяти моделювати мережеві процеси, аналізувати продуктивність, визначати оптимальні конфігурації та оцінювати ефективність мережі.

Функціональні вимоги

2.1. Моделювання мережі

– Створення Мережевих Моделей: Забезпечення функціоналу для створення детальних моделей безпровідних мереж, включаючи вузли, з'єднання, пропускні здатності, та інші важливі параметри.

– Конфігураційні Варіанти: Можливість моделювання різних конфігурацій мережі для вивчення їх впливу на загальну продуктивність.

2.2. Аналіз систем масового обслуговування (СМО)

– Інтеграція алгоритмів СМО: Впровадження алгоритмів для аналізу черг, обробки запитів та розподілу ресурсів у мережі.

– Сценарії використання: Моделювання різних сценаріїв використання мережі, таких як висока завантаженість або збої в роботі окремих компонентів.

2.3. Оптимізація мережі

– Автоматизований Пошук Оптимальних Рішень: Алгоритми для визначення найкращих конфігурацій мережі, з урахуванням ефективності та вартості.

– Параметри Оптимізації: Включення варіабельних параметрів, таких як швидкість передачі даних, надійність, вартість обладнання, для налаштування оптимізації.

2.4. Візуалізація

– Графічне Представлення Мережі: Візуалізація архітектури мережі та її компонентів для легшого розуміння структури.

– Візуалізація Результатів: Графіки та діаграми для показу результатів аналізу, такі як продуктивність мережі під різним навантаженням.

Технічні вимоги

– Мова програмування: Python.

– Бібліотеки та інструменти: SymPy для символьних обчислень, NumPy для чисельних обчислень, Matplotlib для візуалізації, інші необхідні бібліотеки Python.

– Інтерфейс. Командний рядок або простий графічний інтерфейс користувача (за необхідності).

1.5 Висновки до розділу

Здійснено аналітичний огляд інформаційних моделей, алгоритмів та засобів імітаційного моделювання комп’ютерних мереж. Встановлено що особливістю мобільного трафіку є висока мобільність, переривчасте підключення, різноманітність пристроїв, потреба в рівномірному покритті сигналу. Виділено основні параметри моделей СМО при описі мереж: інтенсивність потоку запитів, інтенсивність обслуговування, час заявки у системі, довжина черги, вірогідність відмови.

2 ІНФОРМАЦІЙНА МОДЕЛЬ МЕРЕЖІ

2.1 Структура інформаційної моделі

Розглянемо моделі систем масового обслуговування.

Моделі СМО застосовуються як у керуванні виробничими процесами, так і в секторі послуг. Детальний аналіз черг, що включає оцінювання середньої довжини черги, часу очікування та обслуговування, допомагає глибше зрозуміти основні принципи функціонування систем обслуговування. В системах масового обслуговування виокремлюють три ключові фази, які проходить кожен запит:

1) момент появи запиту на вході системи;

2) час, протягом якого запит перебуває у черзі;

3) процес обслуговування запиту, після якого він залишає систему.

На кожній із цих стадій використовуються специфічні параметри, які необхідно аналізувати перед створенням математичних моделей системи.

Параметри вхідних даних включають:

1) загальну кількість запитів на вході (розмір популяції);

2) спосіб надходження запитів до системи;

3) поведінку споживачів послуг.

Кількість запитів на вході може бути як обмеженою, так і необмеженою. Необмежена популяція припускає, що загальна кількість запитів, що надійшли з моменту запуску до будь-якого моменту часу, становить лише малу частку від потенційно можливої кількості клієнтів [25-30].

Щодо надходження запитів, вони можуть входити в систему за різними сценаріями: згідно із заздалегідь встановленим розкладом або непередбачувано. Поведінка клієнтів зазвичай передбачає, що кожен новий запит приєднується до черги та залишається в системі до моменту його обслуговування. Однак на практиці клієнти можуть виходити з черги, якщо вона здається їм надто довгою.

Характеристики самої черги включають її довжину та правила обслуговування. Довжина черги може бути обмежена фізичними умовами або не мати обмежень. У випадку досягнення максимально можливої довжини, черга перестає приймати нові запити, що призводить до відмови. Більшість реальних

систем використовують принцип "першим прийшов – першим обслужений" (FIFO). Проте у деяких випадках, наприклад у медичних установах, до цього принципу можуть додаватися різні пріоритети, в залежності від стану пацієнтів. Схожий принцип пріоритетів застосовується і при запуску комп'ютерних програм.

Особливості процесу обслуговування включають конфігурацію системи (кількість каналів обслуговування та фаз) та режим обслуговування.

Системи обслуговування можуть мати різноманітну конфігурацію, визначену кількістю каналів обслуговування. Так, у банках чи поштових відділеннях, де відкрито кілька вікон для обслуговування клієнтів, які чекають у спільній черзі, створюється ситуація багатоканальної однофазової системи. В такій системі кожен клієнт підходить до першого звільненого вікна для обслуговування.

З іншого боку, системи обслуговування характеризуються кількістю фаз обслуговування. Система вважається однофазовою, коли клієнт отримує всі послуги в одному місці і після цього залишає її. Проте, якщо в ресторані клієнт проходить кілька етапів — замовляє їжу, оплачує в іншому місці та отримує її в третьому, то це приклад багатофазової системи обслуговування, яка в даному випадку містить три фази.

Системи обслуговування різної конфігурації наведено на рисунку 1.1.

На рисунку 2.1 представлено чотири типи систем масового обслуговування:

а) Одноканальна однофазова система: показує один потік заявок, що проходить через один пункт обслуговування.

б) Одноканальна двофазова система: ілюструє один потік заявок, що проходить послідовно через два пункти обслуговування.

в) Двоканальна однофазна система: зображує один потік заявок, який розділяється на два паралельних пункти обслуговування.

г) Двоканальна двофазна система: показує один потік заявок, який розділяється на два паралельних пункти обслуговування у першій фазі, а потім кожен з них проходить через другий пункт обслуговування в другій фазі.

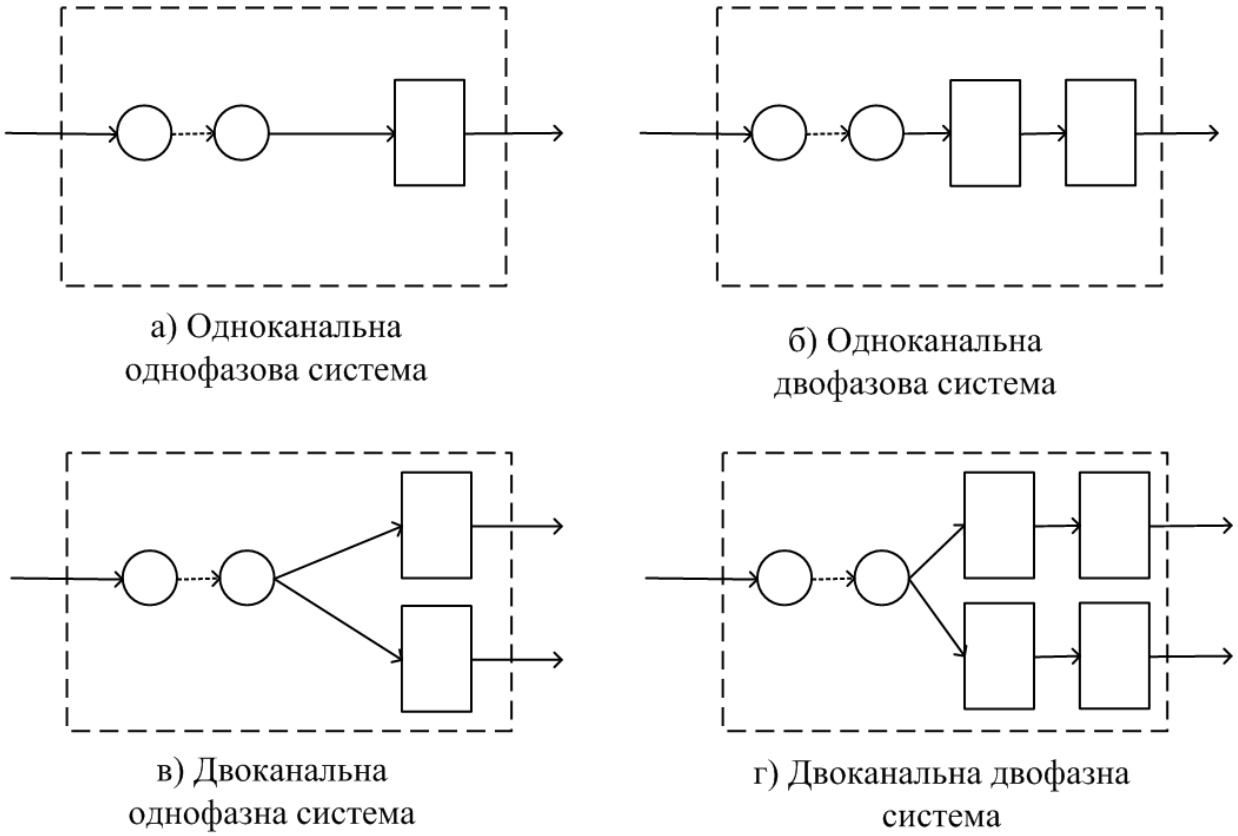

Рисунок 2.1 - Типи систем масового обслуговування

Режим обслуговування. Як і режим надходження заявок, режим обслуговування може характеризуватися або постійним, або випадковим часом обслуговування. За постійного часу на обслуговування будь-якого клієнта витрачається однаковий час. Така ситуація може спостерігатися на автоматичній мийці автомобілів. Однак частіше трапляються ситуації, коли час обслуговування має випадковий розподіл.

Опис мережі у вигляді СМО

Для створення структури інформаційної моделі, що поєднує моделі систем масового обслуговування (СМО) з концепціями безпровідних мереж, нам потрібно спочатку визначити основні поняття та їх взаємозв'язки у кожній з цих областей. Потім ми можемо інтегрувати ці поняття таким чином, щоб безпровідні мережі були описані в термінах СМО [30-40].

Основні поняття СМО:

- сервер - елемент який обробляє запити;

- клієнт - генератор запитів у системі;
- черга - місце, де зберігаються запити, що чекають обробки;
- процес обслуговування - дія, що виконується сервером для обробки запиту;
- інтенсивність потоку запитів - частота, з якою клієнти генерують запити;
- інтенсивність обслуговування -спроможність сервера обробляти запити за одиницю часу.

Основні поняття безпровідних мереж:

- Вузол мережі: Пристрій у безпровідній мережі, що може виступати як передавач або приймач.
- Пакет даних: Одиниця інформації, що передається у мережі.
- Канал зв'язку: Медіум, через який відбувається передача даних.
- Пропускна спроможність: Максимальна кількість даних, що може бути передана через канал за одиницю часу.

Інтеграція двох систем понять:

- Сервер (СМО) ↔ Вузол мережі (мережі). У контексті безпровідної мережі, вузол мережі може виступати як сервер у СМО, обробляючи пакети даних (запити).
- Клієнт (СМО) ↔ Передавач (мережі). Пристрої, що ініціюють передачу даних, можуть розглядатися як клієнти у СМО.
- Черга (СМО) ↔ Буфер (мережі): Черга в СМО відповідає буферу в безпровідній мережі, де зберігаються пакети, що чекають на передачу або обробку.
- Інтенсивність потоку запитів (СМО) ↔ Частота генерації пакетів (мережі). Частота, з якою вузли мережі генерують пакети даних, відповідає інтенсивності потоку запитів у СМО.
- Інтенсивність обслуговування (СМО) ↔ Пропускна спроможність (безпровідні Мережі). Здатність вузла мережі обробляти або передавати пакети даних відповідає інтенсивності обслуговування сервера у СМО.

Цей підхід дозволяє розглядати мережі через призму теорії СМО, інтегруючи технічні деталі мережевої інфраструктури з абстрактними концепціями систем масового обслуговування.

На нижче представленій діаграмі візуалізовано інтеграцію онтологій систем масового обслуговування (СМО) та безпровідних мереж. Вузли діаграми відображають ключові концепції з обох областей, а стрілки показують, як елементи СМО відповідають або представляють елементи безпровідних мереж. Ця діаграма допомагає зрозуміти, як різні аспекти СМО можуть бути застосовані до концепцій, пов'язаних з безпровідними мережами.

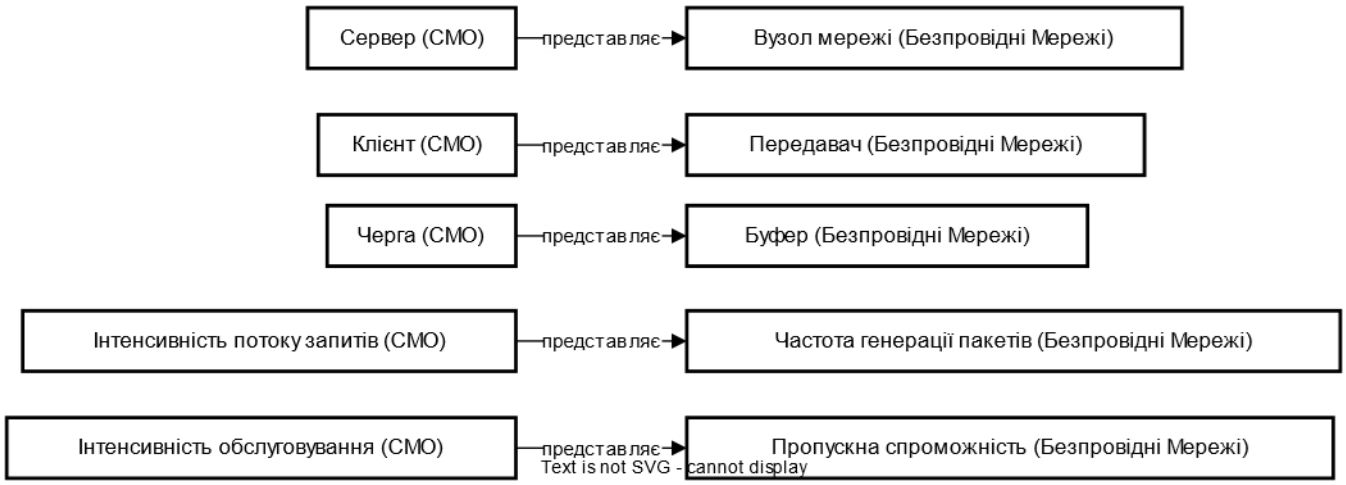

Рисунок 2.2 – Розроблення інформаційної моделі

На цій діаграмі:

- Вертикальні блоки представляють елементи з онтологій СМО та безпровідних мереж.
- Стрілки між блоками ілюструють відносини "представляє" між елементами СМО та відповідними елементами безпровідних мереж.

Додаймо технічні характеристики моделі СМО:

1) середній час, який клієнт проводить у черзі;

2) середня довжина черги;

3) середній час, який клієнт проводить у системі обслуговування (час очікування плюс час обслуговування);

4) середня кількість клієнтів у системі обслуговування;

5) імовірність того, що система обслуговування виявиться незайнятою;

6) імовірність певного числа клієнтів у системі.

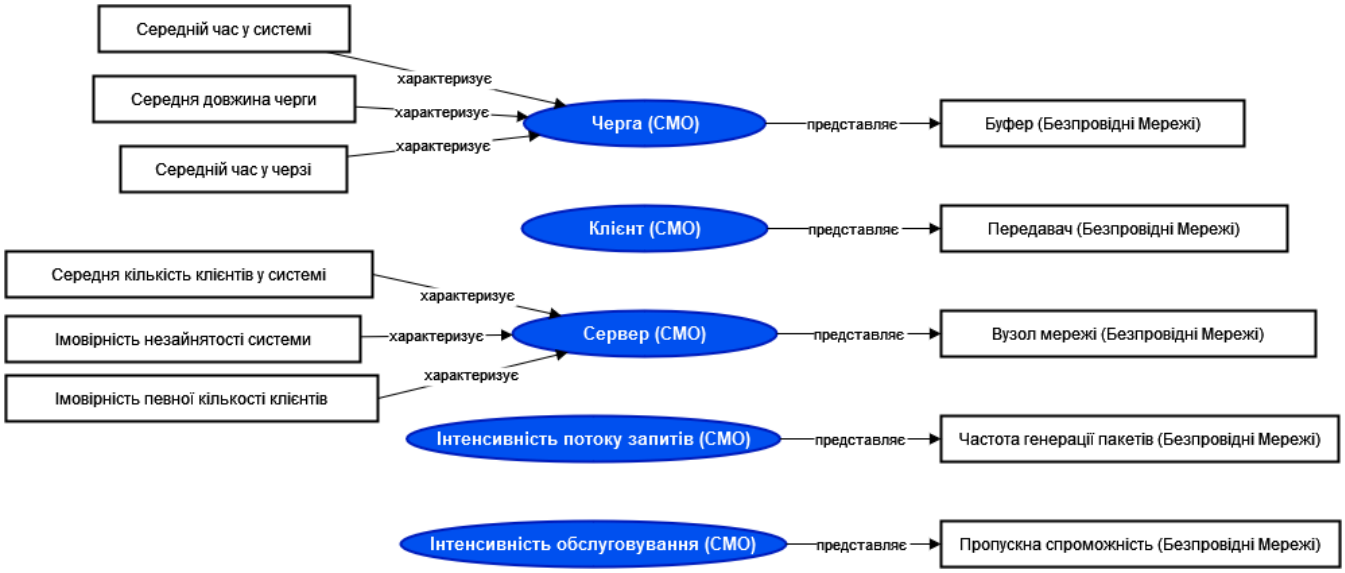

Рисунок 2.3 – Інформаційна модель

Додані нові відносини "характеризує", які вказують, як концепції, пов'язані з часом у черзі, довжиною черги, часом у системі, кількістю клієнтів у системі та імовірностями, пов'язані з елементами СМО, такими як "Черга" та "Сервер".

Ці відносини допомагають краще зрозуміти, як різні метрики та характеристики систем масового обслуговування можуть бути застосовані у контексті безпровідних

Представлення у вигляді форми Бекуса-Наура

Інформаційна модель системи масового обслуговування (СМО) може бути представлена у вигляді форми Бекуса-Наура (BNF), яка є стандартом для формального представлення синтаксичних правил мов програмування, протоколів, інформаційних моделей тощо. BNF використовується для визначення елементів моделі та їх взаємозв'язків у структурованому і строгому форматі [13-15].

Приклад опис СМО у формі Бекуса-Наура:

```
<СМО> ::= <Вхідний_потік> <Черга> <Обслуговуючі_канали>
<Дисципліна_обслуговування>
<Вхідний_потік> ::= "Пуассонівський потік з інтенсивністю"
<інтенсивність_потоку>
<Черга> ::= "Черга з максимальною довжиною" <максимальна_довжина_черги>
<Обслуговуючі_канали> ::= "Кількість обслуговуючих каналів"
<кількість_каналів>
<Дисципліна_обслуговування> ::= "FIFO" | "LIFO" | "Priority"
<інтенсивність_потоку> ::= <число>
<максимальна_довжина_черги> ::= <число> | "без обмежень"
<кількість_каналів> ::= <число>
```

У цьому прикладі:

- <СМО> представляє цілу систему масового обслуговування.
- <Вхідний_потік> описує характеристики вхідного потоку клієнтів.
- <Черга> визначає властивості черги, такі як її максимальна довжина.
- <Обслуговуючі_канали> описує кількість каналів (серверів) у системі.
- <Дисципліна_обслуговування> визначає дисципліну обслуговування в черзі.

Форма Бекуса-Наура дозволяє чітко та однозначно описати структуру та правила будь-якої інформаційної системи, забезпечуючи зрозумілу та стандартизовану основу для аналізу, проектування та реалізації.

Зазначений BNF опис можна перетворити в XML-структуру, яка відображатиме компоненти системи масового обслуговування (СМО). Приклад у форматі XML:

```
<СМО>
    <Вхідний_потік інтенсивність="10">Пуассонівський потік з
інтенсивністю</Вхідний_потік>
    <Черга максимальна_довжина="5">Черга з максимальною
довжиною</Черга>
    <Обслуговуючі_канали кількість="3">Кількість обслуговуючих
каналів</Обслуговуючі_канали>
    <Дисципліна_обслуговування
тип="FIFO">FIFO</Дисципліна_обслуговування>
</СМО>
```

У цьому XML-документі:

– Елемент <СМО> є кореневим елементом, який включає у себе всі частини опису системи масового обслуговування.

– Кожен компонент СМО описаний як окремий елемент, наприклад <Вхідний_потік>, <Черга>, <Обслуговуючі_канали>, і <Дисципліна_обслуговування>.

– Атрибути кожного елемента, такі як інтенсивність, максимальна_довжина, кількість, та тип, визначають специфічні параметри для кожного компонента системи.

2.2 Структури представлення мережевого обладнання

Представлення реальної мережі у контексті моделювання системи масового обслуговування вимагає додавання додаткових параметрів та характеристик для робочих станцій, комутаторів, маршрутизаторів та серверів. Це включає специфікацію їхньої пропускної здатності, часу обробки, а також визначення поведінки мережевого трафіку [40-50]. Ось кілька способів, як можна це зробити:

1. Визначення параметрів обладнання. Кожен мережевий компонент (наприклад, сервер або комутатор) може мати свої унікальні характеристики, такі як максимальна пропускна здатність, час обробки запиту, обсяг черги та інші.

2. Моделювання різних типів трафіку. В реальній мережі трафік може бути різноманітним, включаючи легкі запити до сервера, важкі потоки даних, відеоконференції тощо. Можна задати різні типи та інтенсивності трафіку для симуляції.

3. Пріоритети обслуговування. У реальних мережах можуть використовуватися різні механізми пріоритезації трафіку, які можна відтворити в моделі.

4. Симуляція мережевих збоїв та відновлень. Реальні мережі часто зіштовхуються з перервами у роботі та необхідністю відновлення. Можна включити логіку для імітації цих подій.

5. Візуалізація мережі. Використання інструментів візуалізації для представлення структури мережі та динаміки трафіку.

Код моделювання:

```
class NetworkDevice:
    def __init__(self, env, capacity, processing_time):
        self.server = simpy.Resource(env, capacity)
        self.processing_time = processing_time

def client(env, name, device):
    print(f'{name} прибуває у {env.now:.2f}')
    with device.server.request() as request:
        yield request
        print(f'{name} починає обслуговування у {env.now:.2f}')
        yield env.timeout(device.processing_time)
        print(f'{name} закінчує обслуговування у {env.now:.2f}')

# ... (код для налаштування середовища і запуску симуляції)

# Приклад створення мережевих пристроїв з різними параметрами
router = NetworkDevice(env, capacity=2, processing_time=2)
switch = NetworkDevice(env, capacity=3, processing_time=1)
```

Цей код включає клас NetworkDevice, який може відображати різні мережеві пристрої з їх властивостями.

2.2.1 Параметри комутатора.

Моделювання процесу опрацювання заявки-пакету в сервері СМО в контексті комп'ютерних мереж, де сервер функціонує як комутатор, може бути представлене наступним чином (основні компоненти моделі):

1. Заявки-Пакети. У випадку мережевого комутатора, заявки-пакети – це мережеві пакети даних, які надходять до комутатора для обробки.

2. Черга**.** Комутатор має чергу (буфер), де пакети можуть накопичуватися, чекаючи на обробку.

3. Комутатор виконує функцію процесу обслуговування, тобто, обробляє пакети, визначаючи їхній напрямок та пересилаючи їх до відповідних портів.

Процес моделювання складається:

1. Прибуття Пакетів:
    - Пакети приходять до комутатора відповідно до певного розподілу часу прибуття (наприклад, Пуассонівський розподіл).
    - Час між прибуттями пакетів може бути модельований як випадкова величина.
2. Черга:
    - Якщо комутатор зайнятий обробкою іншого пакета, нові пакети потрапляють у чергу.
    - Черга може бути обмежена за розміром, що може призводити до втрати пакетів при переповненні.
3. Обробка Пакетів:
    - Кожен пакет обробляється комутатором із певною швидкістю, яка може бути модельована як випадкова величина або фіксоване значення (наприклад, час, необхідний для комутації пакета).
4. Передача Пакетів:
    - Після обробки пакети передаються до відповідних виходів комутатора.

Параметри Моделі:

– Інтенсивність потоку запитів (λ): Частота приходу пакетів до комутатора.

– Інтенсивність обслуговування (μ): Середній час обробки одного пакета.

– Максимальна довжина черги: Обмеження на кількість пакетів, що можуть очікувати в черзі.

Таблиця, що відображає ключові аспекти процесу обробки заявок-пакетів в сервері СМО, який функціонує як комутатор у комп'ютерній мережі.

Ця таблиця допомагає візуалізувати та зрозуміти процес обробки заявок-пакетів у СМО, де сервер функціонує як комутатор.

Таблиця 2.1 - Обробка заявок-пакетів комутатором

Компонент моделі	Опис	Приклади параметрів
прибуття пакетів	Пакети надходять до комутатора за певним розподілом часу прибуття.	Інтенсивність потоку запитів (λ), наприклад, кількість пакетів за секунду.
черга	Нові пакети потрапляють у чергу, якщо комутатор зайнятий обробкою іншого пакета.	Максимальна довжина черги, політика обслуговування черги.
обробка пакетів	Комутатор обробляє кожен пакет із певною швидкістю.	Інтенсивність обслуговування (μ), наприклад, час обробки одного пакета.
передача пакетів	Пакети передаються до відповідних портів після обробки.	Швидкість передачі даних, порти виходу комутатора.

2.2.2 Параметри маршрутизатора

Таблиця, яка відображає ключові аспекти процесу обробки заявок-пакетів у сервері системи масового обслуговування (СМО), де сервер функціонує як маршрутизатор у комп'ютерній мережі.

Таблиця 2.2 - Обробка заявок-пакетів маршрутизатором

Компонент Моделі	Опис	Приклади Параметрів
прибуття пакетів	Пакети надходять до маршрутизатора за певним розподілом часу прибуття.	Інтенсивність потоку запитів (λ), наприклад, кількість пакетів за секунду.
обробка маршрутів	Маршрутизатор аналізує заголовки пакетів для визначення їхнього напрямку.	Час обробки маршруту, розмір маршрутної таблиці.
чергування	Нові пакети потрапляють у чергу, якщо маршрутизатор зайнятий обробкою іншого пакета.	Максимальна довжина черги, політика обслуговування черги.
передача пакетів	Пакети передаються до відповідних інтерфейсів виходу після обробки.	Швидкість інтерфейсів, порти виходу маршрутизатора.

У контексті маршрутизатора, ключовим відмінним аспектом є процес обробки маршрутів, який включає аналіз заголовків пакетів та визначення

їхнього напрямку на основі маршрутної таблиці. Цей процес може бути модельований як частина СМО, де маршрутизатор виступає в ролі обслуговуючого пристрою, що обробляє "заявки" від мережевих пакетів.

2.2.3 Параметри бездротових пристроїв

Таблиці, які відображають ключові аспекти процесу обробки заявок-пакетів для IoT Gateway та Wireless Repeaters у контексті СМО.

Таблиця 2.3 - Обробка заявок-пакетів IoT Gateway

Компонент моделі	Опис	Приклади параметрів
прибуття пакетів	Пакети від різних IoT пристроїв надходять до шлюзу.	Інтенсивність потоку запитів (λ), наприклад, кількість пакетів за секунду.
обробка пакетів	Шлюз аналізує та пересилає пакети до мережі або хмари.	Час обробки пакета, швидкість обробки даних.
чергування	Пакети чекають на обробку у випадку завантаженості шлюзу.	Максимальна довжина черги, політика обслуговування черги.
передача пакетів	Пакети передаються до хмарного сервісу або іншої частини мережі.	Швидкість інтерфейсів, протоколи передачі даних.

Таблиця 2.4 - Обробка заявок-пакетів бездротові ретранслятори

Компонент Моделі	Опис	Приклади Параметрів
Прибуття Пакетів	Пакети надходять від основного маршрутизатора або інших бездротових пристроїв.	Інтенсивність потоку запитів (λ), наприклад, кількість пакетів за секунду.
Чергування	Пакети можуть чекати на пересилання у випадку завантаженості ретранслятора.	Максимальна довжина черги.
Пересилання Пакетів	Ретранслятор передає пакети, збільшуючи дальність покриття мережі.	Час передачі пакета, покращення сигналу.

2.3 Алгоритми реалізації моделей системи масового обслуговування

Моделювання процесу опрацювання заявки-пакету в сервері системи масового обслуговування (СМО) в контексті комп'ютерних мереж, де сервер функціонує як комутатор, може бути представлене наступним чином:

Основні компоненти моделі.

1. Заявки-Пакети. У випадку мережевого комутатора, заявки-пакети – це мережеві пакети даних, які надходять до комутатора для обробки

2. Комутатор має чергу (буфер), де пакети можуть накопичуватися, чекаючи на обробку

3. Процес обслуговування. Комутатор обробляє пакети, визначаючи їхній напрямок та пересилаючи їх до відповідних портів

Етапи алгоритму процесу моделювання:

1. Прибуття пакетів. Пакети приходять до комутатора відповідно до певного розподілу часу прибуття (наприклад, Пуассонівський розподіл). Час між прибуттями пакетів може бути модельований як випадкова величина.

2. Чергування. Якщо комутатор зайнятий обробкою іншого пакета, нові пакети потрапляють у чергу. Черга може бути обмежена за розміром, що може призводити до втрати пакетів при переповненні.

3. Обробка Пакетів. Кожен пакет обробляється комутатором із певною швидкістю, яка може бути модельована як випадкова величина або фіксоване значення (наприклад, час, необхідний для комутації пакета).

4. Передача Пакетів. Після обробки пакети передаються до відповідних виходів комутатора.

Опишемо параметри моделі СМО:

- Інтенсивність потоку запитів (λ). Частота приходу пакетів до комутатора.
- Інтенсивність обслуговування (μ). Середній час обробки одного пакета.
- Максимальна довжина черги. Обмеження на кількість пакетів, що можуть очікувати в черзі.

Моделі СМО.

Моделі систем масового обслуговування використовують стандартну нотацію Кендалла, яка складається з п'яти символів у форматі A/B/C/D/E, де:

- *A* визначає час між прибуттями заявок у систему (розподіл інтервалів прибуття),
- *B* визначає час обслуговування заявок (розподіл часу обслуговування),
- *C* - кількість каналів обслуговування (кількість одночасних заявок, що можуть обслуговуватися),
- *D* - максимальна довжина черги (обмеження черги),
- *E* - обсяг джерела заявок (розмір популяції клієнтів).

Деякі з найбільш відомих моделей СМО, які використовуються для різних типів комп'ютерних мереж в таблиці .

Таблиця 2.4 - Відомі моделі СМО

Модель СМО	Час між прибуттями	Час обслуговування	Кількість каналів	Макс. довжина черги	Розмір популяції
M/M/1	Експоненційний	Експоненційний	1	∞ (необмежена)	∞ (необмежена)
M/M/c	Експоненційний	Експоненційний	c	∞ (необмежена)	∞ (необмежена)
M/M/1/N	Експоненційний	Експоненційний	1	N	∞ (необмежена)
M/M/c/N	Експоненційний	Експоненційний	c	N	∞ (необмежена)
M/G/1	Експоненційний	Загальний	1	∞ (необмежена)	∞ (необмежена)
G/M/1	Загальний	Експоненційний	1	∞ (необмежена)	∞ (необмежена)
G/G/1	Загальний	Загальний	1	∞ (необмежена)	∞ (необмежена)

Тут "M" означає "Memoryless" і вказує на експоненційний розподіл, тобто без пам'яті. "G" вказує на "General" або загальний розподіл, де часи між прибуттями або часи обслуговування можуть мати будь-який розподіл. Числа "1", "c", "N" вказують відповідно на кількість каналів обслуговування та обмеження черги. "∞" означає, що черга або популяція не мають обмежень.

Ці моделі використовуються для аналізу різних аспектів мережевої поведінки, таких як продуктивність, затримка, пропускна спроможність та надійність. Вибір моделі залежить від специфіки мережевої ситуації, яку необхідно проаналізувати.

Реалізації моделі M/M/1/N

Для реалізації моделі M/M/1/N в Python з метою моделювання безпровідної мережі використаємо бібліотеку `SimPy`. Алгоритм який відображає роботу системи M/M/1/N для безпровідної мережі:

```
python
import simpy
import random

# Параметри моделі
lambda_rate = 5  # швидкість прибуття (пакети за одиницю часу)
mu_rate = 8      # швидкість обслуговування (пакети за одиницю часу)
N = 10           # максимальна кількість пакетів у системі (обмеження
буфера)
simulation_time = 100  # час моделювання

# Функція для генерації пакетів
def packet_generator(env, server, lambda_rate):
    while True:
        yield env.timeout(random.expovariate(lambda_rate))  # Генерація
інтервалу часу до наступного пакету

        if len(server.items) < N:  # Перевірка на переповнення буфера
            packet = f'пакет_{env.now}'
            print(f'Час {env.now:.4f}: {packet} прибув')
            env.process(packet_processing(env, server, packet))

# Функція для обробки пакетів
def packet_processing(env, server, packet):
    with server.request() as request:
        yield request
        yield env.timeout(random.expovariate(mu_rate))  # Час
обслуговування пакету
        print(f'Час {env.now:.4f}: {packet} оброблено')
```

Створення середовища моделювання та ресурсів

```
env = simpy.Environment()
server = simpy.Resource(env, capacity=1)  # Створення одного сервера
(каналу обслуговування)
# Запуск генератора пакетів та моделювання
env.process(packet_generator(env, server, lambda_rate))
env.run(until=simulation_time)
```

У цьому коді packet_generator створює пакети згідно з розподілом Пуассона, а packet_processing обробляє ці пакети на сервері з експоненційно розподіленим часом обслуговування. Якщо кількість пакетів у черзі досягає максимальної межі (N), нові пакети не додаються до черги, що імітує поведінку системи M/M/1/N.

Цей код може бути адаптований під конкретні потреби моделювання, змінивши параметри моделі або додавши додаткову логіку для імітації специфічних аспектів безпровідної мережі.

Реалізації Моделі M/M/m.

У контексті транспортного зв'язку (vehicular communication) [47], використали Модель M/M/m. Приклад коду для симуляції цього випадку.

```
import simpy
import random
import numpy as np

def mm1_queue(env, number_servers, arrival_rate, service_rate):
    """M/M/m queue model in SimPy."""
    server = simpy.Resource(env, capacity=number_servers)
    times = []

    def vehicle(env, name, server, arrival_rate, service_rate):
        """Process for each vehicle."""
        arrival_time = env.now
        with server.request() as request:
            yield request
            yield env.timeout(np.random.exponential(1/service_rate))
            times.append(env.now - arrival_time)

    while True:
        yield env.timeout(np.random.exponential(1/arrival_rate))
        env.process(vehicle(env, 'Vehicle %d' % env.now, server, arrival_rate,
service_rate))

# Parameters
number_servers = 3 # Number of servers
arrival_rate = 2 # Vehicles per minute
service_rate = 1 # Service rate per server
```

```
# Setup and run the simulation
env = simpy.Environment()
env.process(mm1_queue(env, number_servers, arrival_rate, service_rate))
env.run(until=100) # Run the simulation for 100 minutes

# Results
average_waiting_time = np.mean(times)
print(f"Average Waiting Time: {average_waiting_time:.2f} minutes")
```

Система M/M/m, де `m` - це кількість серверів (каналів зв'язку) в системі. Ми будемо розраховувати такі метрики, як середній час очікування у черзі, ймовірність того, що всі канали будуть зайняті, та очікувану кількість перемикань каналів.

Моделювання процесу бездротової передачі даних

Схема процесу показана на рисунку 2.X. Вузол 1 передає пакет. Симульована мережа потім запитує дані трасування, щоб визначити приймаючі вузли - у цьому випадку вузли 2 і 3. Подія прийому пакета потім планується для цих двох вузлів у майбутньому (в залежності від розміру пакета). Кожен вузол складається з одного процесу SimPy (радіо), який містить кілька об'єктів Python (мережевий стек і додаток). У радіо є черга передачі, але немає черги прийому. Це тому, що при сигналізації події прийому пакета радіопередавальний рівень негайно передає пакет вгору в мережевий стек. Для послідовної передачі пакетів через радіо потрібна черга передачі. Якщо ця довжина черги перевищена, радіопередавальний рівень відхиляє (відкидає) пакети з верхніх рівнів. Ці верхні рівні можуть або не мати своєї черги в залежності від симульованого мережевого стеку.

Для симуляції передачі пакета вихідний вузол спочатку перевіряє, чи він відправляв пакет недавно (налаштований параметр, ми використовуємо менше 32 мілісекунд). Якщо так, то він використовує наступний послідовний номер послідовності в даних сліду (переходячи з кінця на початок, якщо потрібно) і сигналізує початок прийому пакета на кожному вузлі, вказаному в сліду. Якщо ні, то він вибирає новий випадковий номер послідовності зі сліду і використовує ці дані для ідентифікації одержувачів. Ця процедура підтримує часові артефакти

в даних сліду при потребі і видаляє будь-яку часову кореляцію, якщо це необхідно.

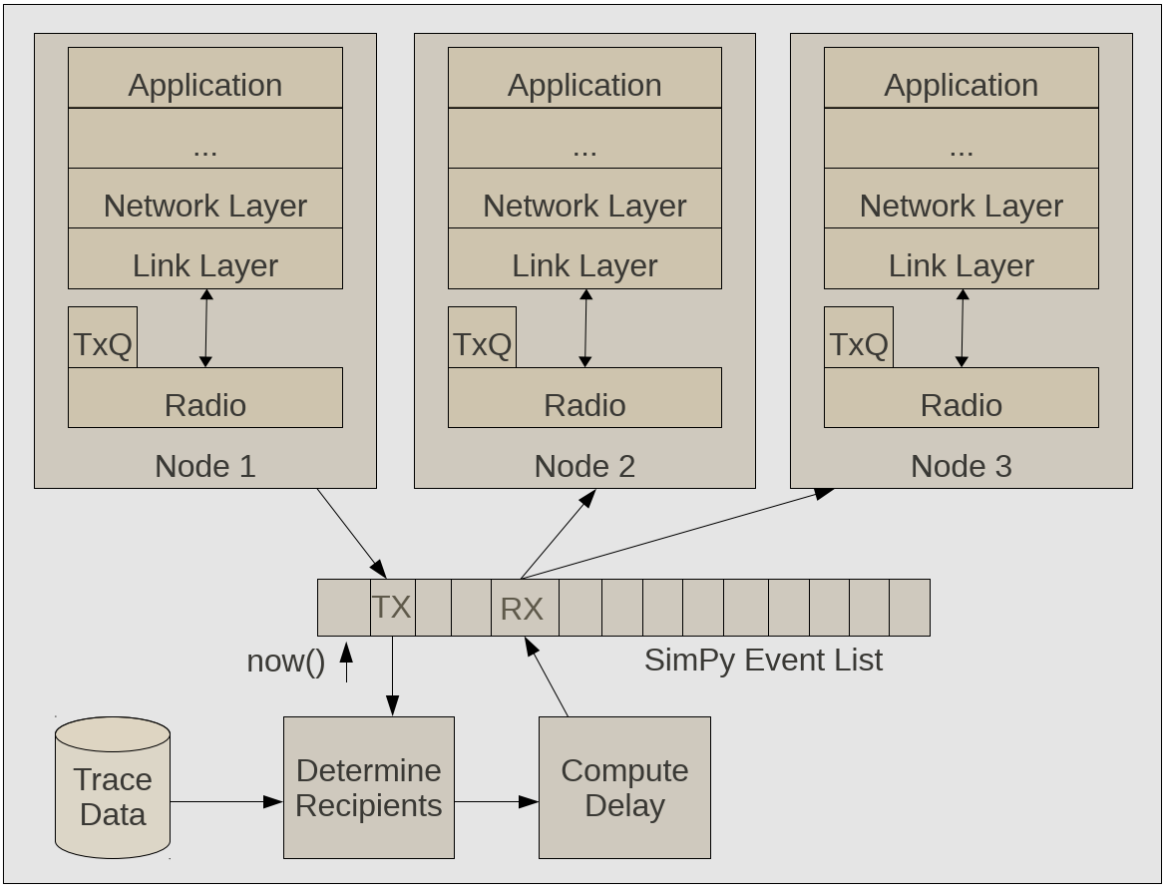

Рисунок 2. – Архітектура системи імітаційного моделювання та збору даних

Для правильної симуляції часу подій перед кожною сигналізацією доставки пакета вставляється затримка. Ми симулюємо радіо IEEE 802.15.4 зі швидкістю передачі даних 250 кбіт/с, тому ми можемо використовувати це для розрахунку затримок на основі приблизного розміру пакета (32 мікросекунди на байт). На короткій дальності спілкування радіопередавальників IEEE 802.15.4 затримки поширення значно менше 1 мікросекунди, тому загальна затримка визначається часом передачі, і ми не враховуємо затримку поширення. Ми також не враховуємо затримок обробки через її специфічний для додатка характер. Однак затримки обробки можна легко додати за допомогою наступного рядка в події прийому вузла:

```
yield hold, self,<симульований час обробки в мікросекундах>.
```

Усі передачі послідовність через симульований радіоприймач напівдуплексу. Це означає, що вузол, який передає, не отримує жодних повідомлень, відправлених до нього, поки не мине симульована затримка передачі. Зворотне також вірне; вузол, який приймає, не може передавати нове повідомлення, поки поточний прийом не завершиться.

Колізії виявляються незалежно на кожному приймаючому вузлі. Це дозволяє можливість зіткнення на одному вузлі, тоді як інші вузли успішно отримують пакети. Після прийому пакета вставляється вікно зіткнення з невеликою затримкою (яка є налаштованим параметром, ми використовуємо 32 мікросекунди). Якщо інший прийом пакета починається в межах цього вікна зіткнення, то всі колізії пакети відкидаються. Цей підхід передбачає ідеалізований рівень MAC, який може ідеально виявляти та перепланувати передачі, які можуть відбуватися поза вікном зіткнення. Це лише спрощена апроксимація зіткнень, однак наші результати (розділ 4) показують, що ця апроксимація є досить точною на практиці. Ця апроксимація менше відповідає для високих швидкостей передачі даних. Для точної симуляції застосунків з високою швидкістю передачі даних потрібен більш реалістичний рівень MAC.

2.4 Висновки до розділу

Вузол мереж виступає як сервер у СМО, обробляючи пакети даних (запити). Пристрої, що ініціюють передачу даних розглядаються як клієнти у СМО. Черга в СМО відповідає буферу в мережі, де зберігаються пакети, що чекають на передачу або обробку. Частота, з якою вузли мережі генерують пакети даних, відповідає інтенсивності потоку запитів у СМО. Здатність вузла мережі обробляти або передавати пакети даних відповідає інтенсивності обслуговування сервера у СМО.

Розроблено структури представлення комутаторів, маршрутизаторів, IoT шлюза та бездротового ретранслятора. Це дало змогу моделювання процесу опрацювання заявки-пакету в сервері системи масового обслуговування.

3 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МОДЕЛЮВАННЯ

3.1 Засоби розробки і структура проєкту

Бібліотека SimPy - це інструмент моделювання для Python, призначений для створення дискретних подійних моделей для дослідження імітаційних систем. Дозволяє представити складні системи масового обслуговування (СМО). Її можна використовувати для аналізу комп'ютерних мереж загалом і безпровідних мереж зокрема. Розглянемо більше переваг і недоліків цього підходу. Переваги:

1. Гнучкість і налаштовуваність. SimPy дозволяє створювати моделі, які відображають різні аспекти мережі та їх параметри. Це дозволяє досліджувати різні сценарії та ефективно аналізувати їх вплив на мережу.

2. Симуляція реального часу. SimPy надає можливість відтворювати часовий хід подій у моделях, що допомагає зрозуміти, як величезні обсяги даних обробляються та передаються в мережі.

3. Безпека та ефективність. Можна проводити тести та експерименти з різними стратегіями управління мережею, не ризикуючи реальним обладнанням. Це допомагає уникнути можливих негативних наслідків на реальній мережі.

Недоліки:

4. Витрати часу на розробку. Створення докладних моделей мережі в SimPy може бути часом і ресурсозатратним завданням, особливо для складних мереж з багатьма параметрами та конфігураціями.

5. Не завжди точне відтворення реальності. Моделі, створені в SimPy, є апроксимаціями реальних мереж. Іноді вони можуть не враховувати всі деталі або фактори, що впливають на реальну мережу.

6. Специфічний інструмент. Використання SimPy вимагає знань мови програмування Python та імітаційних моделей.

Загальна схема процесу налаштування та аналізу безпровідних мереж у середовищі SimPy деталізує основні кроки, включаючи створення екземплярів класів та виклик методів (рисунок 3.1).

Рисунок 3.1 - Схема процесу налаштування та аналізу безпровідних мереж

Розглянемо корпоративну мережу, де сервер обробляє запити від клієнтів. Модель може включати наступні елементи:

– Вхідний потік запитів від клієнтів до сервера, моделюється за допомогою Пуассонівського розподілу.

– Буферизована черга на сервері, де запити очікують на обробку.

– Обслуговуючі канали - процесори сервера або окремі сервери в кластері.

– Дисципліна обслуговування -пріоритетність або черговість обробки запитів.

– Час обслуговування, тобто час, який потрібен серверу для обробки запиту.

Підсумуємо спосіб яким можна співвіднести основні сутності СМО з класами, змінними та методами у SimPy наведено в таблиці 3.1.

Таблиця 3.1 – Основні сутності СМО

Сутність СМО	Співвідношення у SimPy	Опис у SimPy
Сервер	`Resource`	Представляє об'єкт, який може обслуговувати запити (клієнтів). Кожен `Resource` має обмежену кількість слотів для обслуговування.
Клієнт	Об'єкт, який створює запити	Процес, який звертається до ресурсу з метою обслуговування.
Черга	Черга в `Resource`	Запити (клієнти), які чекають на обслуговування в `Resource`, формують чергу.
Процес обслуговування	Процес у SimPy	Процес у SimPy — це генератор Python, який описує поведінку сутності протягом часу. В контексті СМО це може бути процес обслуговування клієнта.
Інтенсивність потоку запитів	Параметр для генерації клієнтів	Визначає, як часто клієнти звертаються до системи. Часто використовується з `env.timeout()` для моделювання часу між появами клієнтів.
Інтенсивність обслуговування	Параметр для обслуговування у `Resource`	Визначає, як швидко ресурс може обслуговувати клієнтів. Часто використовується у поєднанні з `env.timeout()` для моделювання часу обслуговування.

3.2 Розроблення алгоритмів моделювання

Для створення діаграми класів для програмного забезпечення, яке моделює корпоративну комп'ютерну мережу на основі систем масового обслуговування, можна використати наступну структуру. Основні Класи:

1. Клас NetworkModel:
   - Відповідає за створення та управління моделлю мережі.
   - Атрибути: список вузлів, з'єднань, налаштування мережі.
   - Методи: створення мережі, додавання/видалення вузлів, з'єднань.
2. Клас Node
   - Представляє вузол у мережі (наприклад, маршрутизатор, комутатор).
   - Атрибути: ID вузла, тип, параметри пропускної здатності.
   - Методи: обробка даних, взаємодія з іншими вузлами.
3. Клас Connection:
   - управляє з'єднаннями між вузлами;
   - атрибути: вузол-джерело, вузол-приймач, характеристики з'єднання;
   - методи: передача даних, моніторинг статусу з'єднання.
4. Клас QueueSystem:
   - Реалізує логіку систем масового обслуговування;
   - Атрибути: черги, стратегії обслуговування;
   - Методи: управління чергою, аналіз продуктивності.
5. Клас Optimizer:
   - займається оптимізацією мережі;
   - атрибути: параметри для оптимізації;
   - методи: алгоритми оптимізації, вибір оптимальних налаштувань.
6. Клас VisualizationTool:
   - Проводить візуалізацію мережі та результатів.
   - Атрибути: дані для візуалізації.
   - Методи: генерація діаграм, графіків.

7. Клас ReportGenerator:
   - Створює звіти з результатами аналізу.
   - Атрибути: формат звіту, дані для звіту.
   - Методи: генерація звіту, експорт у різні формати.

Таблиця з детальним описом класів, їхніх змінних та методів.

Таблиця 3.2 - Опис класів програмного забезпечення

Клас	Змінні / Методи	Опис
NetworkModel	`nodes: List[Node]` `connections: List[Connection]`	Управління загальною структурою та логікою мережевої моделі.

	`createNetwork()`
	`addNode(Node)`
	`removeNode(Node)`
	`addConnection(Connection)`
	`removeConnection(Connection)`	

Node	`id: String`
	`type: String`
	`capacity: Double`
		Представлення вузла в мережі, наприклад, маршрутизатора або комутатора.

	`processData()` `interactWith(Node)`	

Connection	`sourceNode: Node`
	`targetNode: Node`
	`characteristics: String`
		Управління з'єднаннями між вузлами мережі.

	`transferData()` `monitorStatus()`	
QueueSystem	`queues: List[Queue]` `serviceStrategies: List[Strategy]`	Реалізація логіки систем масового обслуговування для мережі.
	`manageQueue()` `analyzePerformance()`	
Optimizer	`optimizationParameters: Map[String, Double]`	Оптимізація мережевих параметрів.
	`runOptimizationAlgorithms()` `selectOptimalSettings()`	
VisualizationTool	`data: Map[String, Object]`	Візуалізація мережі та результатів аналізу.
	`generateDiagrams()` `generateGraphs()`	
ReportGenerator	`reportFormat: String` `reportData: Map[String, Object]`	Створення звітів з аналізу мережі.
	`generateReport()` `export(String)`	

Опис класів, змінних та методів.

Клас NetworkModel управляє загальною структурою та логікою мережевої моделі. Змінні:

– nodes: Список вузлів у мережі.

– connections: Список з'єднань між вузлами.

Методи:

– createNetwork(): Ініціалізує мережу.

– addNode(Node): Додає новий вузол до мережі.

– removeNode(Node): Видаляє вузол з мережі.

– addConnection(Connection): Створює з'єднання між вузлами.

– removeConnection(Connection): Видаляє з'єднання між вузлами.

Клас Node представляє вузол у мережі, наприклад, маршрутизатор або комутатор. Змінні:

– id: Унікальний ідентифікатор вузла.

– type: Тип вузла (наприклад, маршрутизатор, комутатор).

– capacity: Пропускна здатність вузла.

Методи:

– processData(): Обробляє дані, що проходять через вузол.

– interactWith(Node): Встановлює взаємодію з іншими вузлами.

Клас Connection для управління з'єднаннями між вузлами мережі. Змінні:

– sourceNode: Вузол-джерело з'єднання.

– targetNode: Вузол-приймач з'єднання.

– characteristics: Характеристики з'єднання (наприклад, швидкість, надійність).

Методи:

– transferData(): здійснює передачу даних через з'єднання.

– monitorStatus(): моніторинг стану з'єднання.

Клас QueueSystem Реалізує логіку системи масового обслуговування для мережі. Змінні:

– queues: список черг в системі.

– serviceStrategies: Стратегії обслуговування в чергах.

Методи:

– manageQueue(): Управління чергами та процесами в них.

– analyzePerformance(): Аналіз продуктивності системи черг.

Клас Optimizer займається оптимізацією мережевих параметрів. Змінні:

– optimizationParameters: Параметри для оптимізації.

Методи:

– runOptimizationAlgorithms() Виконує алгоритми оптимізації;

– selectOptimalSettings() Вибір найкращих налаштувань мережі.

Клас VisualizationTool Проводить візуалізацію мережі та результатів аналізу. Змінні:

– data: дані для візуалізації.

Методи:

– generateDiagrams() генерація діаграм мережі;

– generateGraphs() створення графіків продуктивності та інших показників.

Клас ReportGenerator створює звіти з аналізу мережі. Змінні:

– reportFormat: формат звіту;

– reportData: дані для включення в звіт.

Методи:

– generateReport() генерація звіту;

– export(String) експорт звіту в обраний формат.

Для створення каркасу програми, яка використовує бібліотеку SimPy для моделювання корпоративної комп'ютерної мережі на основі систем масового обслуговування, можна використовувати такий код. Я згенерую код для класу `NetworkModel` і покажу, як він може взаємодіяти з користувачем для отримання параметрів мережі.

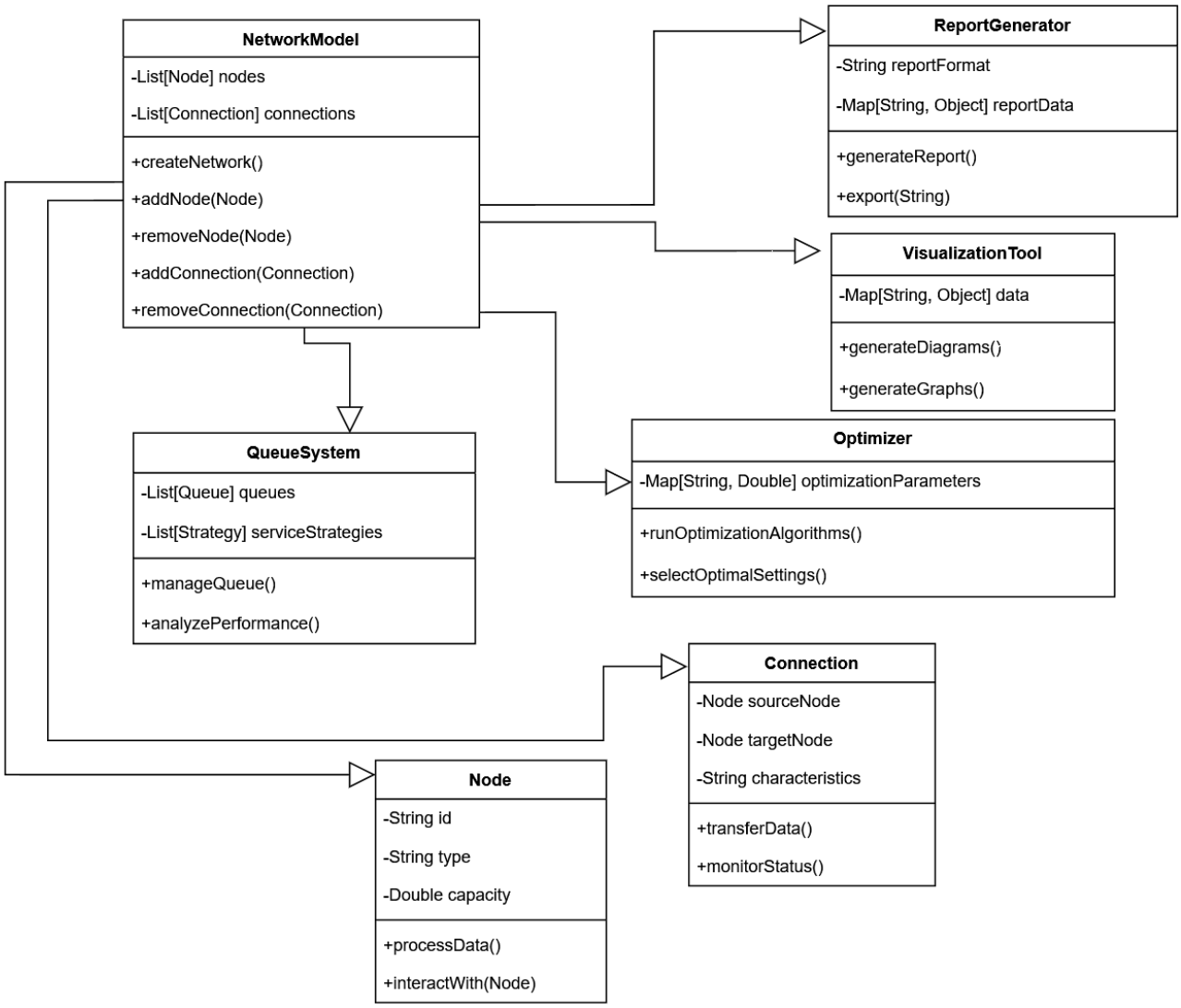

Рисунок 3.2 – Діаграма класів

Код Пайтон сценарію. Підключаємо потрібні бібліотеки

```
python
import simpy
```

Оголошуємо основний клас NetworkModel

```
class NetworkModel:
    def __init__(self, env):
        self.env = env
        self.nodes = []
        self.connections = []

    def createNetwork(self):
        # код для створення мережі
        pass
    def addNode(self, node):
        self.nodes.append(node)
```

```
    def removeNode(self, node):
        self.nodes.remove(node)

    def addConnection(self, connection):
        self.connections.append(connection)

    def removeConnection(self, connection):
        self.connections.remove(connection)
```

Далі ідуть визначення інших класів для тестування і візуалізації.

Створення середовища SimPy.

```
env = simpy.Environment()

# Створення екземпляру моделі мережі
network_model = NetworkModel(env)

# запит параметрів мережі від

# запуск моделі

env.run(until=100)  # Запуск на 100 кроках симуляції
```

Наша програма використовує бібліотеку SimPy для створення симуляційного середовища і демонструє, як можна ініціалізувати клас NetworkModel. Можна додати додаткові класи, такі як Node, Connection, QueueSystem, тощо, та інтегрувати їх з цим каркасом. Для збору параметрів мережі від користувача, можна використовувати функцію input() або інший механізм вводу. Після отримання необхідних даних, можна налаштувати симуляцію і запустити її.

Повернемось до методу `createNetwork(self)` для класу `NetworkModel`. Ми повинні визначити, що саме цей метод повинен робити в контексті моделі СМО та безпровідних мереж. В контексті моделювання СМО (Систем Масового Обслуговування) у безпровідних мережах, `createNetwork` може відповідати за створення ініціальної структури мережі, включаючи вузли мережі (наприклад, маршрутизатори або комутатори) та з'єднання між ними. Цей метод також може встановлювати початкові параметри для кожного вузла, такі як пропускна

спроможність, інтенсивність потоку запитів та інші характеристики, пов'язані зі СМО.

```
    def createNetwork(self, network_config):
        # Створення вузлів мережі
        for node_config in network_config["nodes"]:
            node = Node(node_config["id"], node_config["type"],
                        node_config["capacity"])
            self.addNode(node)

        # Створення з'єднань між вузлами
        for conn_config in network_config["connections"]:
            source_node = self.getNodeById(conn_config["source"])
            target_node = self.getNodeById(conn_config["target"])
            connection = Connection(source_node, target_node,
conn_config["characteristics"])
            self.addConnection(connection)

    def addNode(self, node):
        self.nodes.append(node)

    def getNodeById(self, node_id):
        return next((node for node in self.nodes if node.id ==
node_id), None)

    def addConnection(self, connection):
        self.connections.append(connection)
```

Клас createNetwork приймає конфігурацію мережі (network_config), яка може містити інформацію про вузли та з'єднання. Вона створює вузли та з'єднання на основі цієї інформації. Я припускаю, що класи Node та Connection вже визначені в іншому місці вашої програми.

Приклад конфігурації мережі може виглядати так:

```
network_config = {
    "nodes": [
        {"id": "1", "type": "router", "capacity": 100},
        {"id": "2", "type": "switch", "capacity": 100}
        # Інші вузли...
    ],
    "connections": [
        {"source": "1", "target": "2", "characteristics": "high-
speed"}
        # Інші з'єднання...
    ]
}
```

Класи Node та Connection визначені окремо.

Визначимо класи Node та Connection для використання у вашій моделі мережі. Ці класи будуть представляти вузли мережі (наприклад, маршрутизатори або комутатори) та з'єднання між ними.

Клас Node може включати інформацію про кожен вузол мережі, таку як ідентифікатор, тип вузла, пропускна здатність та інші властивості, які можуть бути важливі для вашої моделі.

```
class Node:
    def __init__(self, node_id, node_type, capacity):
        self.id = node_id
        self.type = node_type
        self.capacity = capacity
        # Можна додати інші властивості та методи, якщо це необхідно
```

Клас Connection відображатиме з'єднання між вузлами. Він може включати інформацію про вузол-джерело, вузол-приймач та характеристики з'єднання.

```
class Connection:
    def __init__(self, source_node, target_node, characteristics):
        self.source_node = source_node
        self.target_node = target_node
        self.characteristics = characteristics
        # Можна додати додаткові методи для управління з'єднанням
```

Ці класи є основою для представлення структури мережі у вашій моделі. Ви можете додати додаткові методи та властивості, які відображають специфіку вашої задачі та деталі мережевої моделі.

Імпорт та експорт інформаційної моделі

З огляду на систему понять СМО, і вимогу імпорту/експорту налаштувань моделі СМО у формі BNF, яка може бути збережена у форматі XML, розширимо програмний каркас новим класом, який буде відповідати за ці функції. Назвемо цей клас `SMOSettingsManager`. Він буде відповідати за імпорт та експорт налаштувань моделі СМО.

Підключаємо бібліотеку

```
import xml.etree.ElementTree as ET
```

Клас `SMOSettingsManager`

```
class SMOSettingsManager:
    def __init__(self, filename):
        self.filename = filename

    def importSettings(self):
        # Імпорт налаштувань з XML файлу
        tree = ET.parse(self.filename)
        root = tree.getroot()
        # Парсинг XML та встановлення налаштувань моделі
        # ...

    def exportSettings(self, settings):
        # Експорт налаштувань у XML файл
        root = ET.Element("SMOSettings")
        # Створення/додавання елементів на основі налаштувань моделі
        # ...
        tree = ET.ElementTree(root)
        tree.write(self.filename)


# Створення екземпляру менеджера налаштувань СМО
smo_settings_manager = SMOSettingsManager("settings.xml")
# Імпорт налаштувань СМО
smo_settings_manager.importSettings()
# ..налаштування моделі мережі на основі імпортованих налаштувань
```

Цей код демонструє базову структуру класу SMOSettingsManager, який може імпортувати налаштування моделі СМО з XML-файлу і експортувати їх назад у файл. Це забезпечує зв'язок між онтологією СМО-WSN та програмним забезпеченням для моделювання. Конкретні деталі імплементації залежать від структури XML-файлу та специфіки моделі СМО, яку ви хочете використовувати.

Функція importSettings() в класі SMOSettingsManager, імпортує налаштування моделі СМО з XML-файлу. У файлі XML зберігається інформація про різні параметри СМО, такі як інтенсивність потоку запитів, інтенсивність обслуговування тощо. Функція виглядає так:

```
import xml.etree.ElementTree as ET
class SMOSettingsManager:
```

```
    def __init__(self, filename):
        self.filename = filename
    def importSettings(self):
        # Словник для збереження налаштувань
        settings = {}
        # Імпорт налаштувань з XML файлу
        tree = ET.parse(self.filename)
        root = tree.getroot()
        # Перебір елементів XML та збір налаштувань
        for param in root.findall('parameter'):
            name = param.get('name')
            value = param.text
            settings[name] = value
        return settings
```

У цьому прикладі importSettings() читає XML-файл, аналізує його та збирає налаштування у словник. Припускається, що у файлі XML є елементи <parameter>, які мають атрибут name і текстове значення.

Враховуючи раніше розроблену таблицю (??) для моделі СМО, структура XML-файлу може містити налаштування для різних параметрів, пов'язаних з серверами (СМО), клієнтами, чергами, інтенсивністю потоку запитів та інтенсивністю обслуговування. Крім того, можуть бути параметри для оцінки середнього часу у черзі, середньої довжини черги, середнього часу у системі, середньої кількості клієнтів у системі тощо.

Остаточний варіант структури XML-файлу:

```
<SMOSettings>
    <ServerSettings>
        <Parameter name="averageServiceTime" value="2.5"/>
        <Parameter name="serviceRate" value="0.4"/>
        <Parameter name="idleProbability" value="0.2"/>
        <Parameter name="clientsProbability" value="0.5"/>
    </ServerSettings>
    <ClientSettings>
        <Parameter name="requestRate" value="0.7"/>
    </ClientSettings>
    <QueueSettings>
        <Parameter name="averageQueueTime" value="3.5"/>
        <Parameter name="averageQueueLength" value="5"/>
        <Parameter name="averageSystemTime" value="6"/>
        <Parameter name="averageClientsInSystem" value="4"/>
    </QueueSettings> </SMOSettings>
```

Ця структура розділяє налаштування на три основні секції: `ServerSettings`, `ClientSettings`, і `QueueSettings`. Кожна секція містить параметри, які характеризують відповідні аспекти системи масового обслуговування.

Відповідна функція `importSettings()` для читання цієї структури може виглядати наступним чином:

```
import xml.etree.ElementTree as ET

class SMOSettingsManager:
    def __init__(self, filename):
        self.filename = filename

    def importSettings(self):
        settings = {"ServerSettings": {}, "ClientSettings": {},
"QueueSettings": {}}
        tree = ET.parse(self.filename)
        root = tree.getroot()

        for section in root:
            section_name = section.tag
            for param in section.findall('Parameter'):
                name = param.get('name')
                value = param.get('value')
                settings[section_name][name] = value
        return settings
```

У цій функції `importSettings()` читаються та аналізуються параметри з кожної секції (`ServerSettings`, `ClientSettings`, `QueueSettings`) та зберігаються у відповідних словниках. Кожен параметр має `name` та `value`, які використовуються для збору інформації.

3.3 Експериментальне дослідження алгоритмів

Створення тестового сценарію, спочатку створимо тестовий XML файл smo_settings.xml з певними значеннями моделі СМО. Потім додамо логіку для зчитування цього файлу та запуску симуляції.

```
<SMOSettings>
    <ServerSettings>
        <Parameter name="averageServiceTime" value="2.5"/>
        <Parameter name="serviceRate" value="0.4"/>
```

```
    <Parameter name="idleProbability" value="0.2"/>
    <Parameter name="clientsProbability" value="0.5"/>
</ServerSettings>
<ClientSettings>
    <Parameter name="requestRate" value="0.7"/>
</ClientSettings>
<QueueSettings>
    <Parameter name="averageQueueTime" value="3.5"/>
    <Parameter name="averageQueueLength" value="5"/>
    <Parameter name="averageSystemTime" value="6"/>
    <Parameter name="averageClientsInSystem" value="4"/>
</QueueSettings> </SMOSettings>
```

Цей файл містить різні параметри для налаштувань сервера, клієнта та черги, які можуть бути використані у моделі СМО.

Для візуалізації результатів роботи моделі СМО у проекті можна застосувати різні підходи. Найпоширенішими інструментами для візуалізації даних у Python є бібліотеки Matplotlib і Seaborn. Ось кілька прикладів того, як можна візуалізувати результати:

1. Графіки Черги. Якщо потрібно візуалізувати динаміку черги (наприклад, середню довжину черги протягом часу), ви можете збирати дані в процесі симуляції і потім використовувати Matplotlib для створення графіка.

2. Гістограми можуть бути корисними для візуалізації розподілу таких параметрів, як час обслуговування або час очікування в черзі.

3. Теплові карти Для візуалізації складних взаємозв'язків між різними параметрами (наприклад, як зміна інтенсивності потоку впливає на середню довжину черги) можна використовувати теплові карти.

Для збору даних потріьно використовувати об'єкти або змінні всередині моделі SimPy, які оновлюються на кожному кроці симуляції. Наприклад, ви можете додати код, який реєструє поточну довжину черги на кожному кроці симуляції.

Для візуалізації безпровідної сенсорної мережі (WSN) з 5 вузлами, де кожен вузол моделюється як частина СМО, можна створити алгоритм, який виконує наступні кроки.

1. Ініціалізація Вузлів WSN - кожен вузол має параметри, характерні для СМО, наприклад, інтенсивність обслуговування та інтенсивність потоку запитів.

2. Створення імітаційного середовища

3. Моделювання процесів у вузлах. Кожен вузол виконує СМО-подібні процеси, такі як прийом запитів і обслуговування (наприклад, збір даних, обробка запитів).

4. Збір даних процесів. Збирати дані з кожного вузла протягом симуляції, такі як час у черзі, час обслуговування тощо.

5. Візуалізація структури мережі. Використати бібліотеку візуалізації (Matplotlib або NetworkX у Python) для візуалізації структури WSN.

6. Візуалізація результатів симуляції. Створити графіки або діаграми, що відображають зібрані дані (наприклад, середній час у черзі для кожного вузла).

7. Аналіз та Інтерпретація Результатів. Оцінити ефективність кожного вузла у мережі з точки зору СМО. Зробити висновки щодо загальної продуктивності мережі

Тестові дані: Для тестування були використано середній час очікування в черзі для кожного з п'яти вузлів мережі.

Візуалізація показує структуру мережі з п'ятьма вузлами та з'єднаннями між ними. Використано бібліотеку networkx для створення та візуалізації графа мережі.

Аналіз інтенсивності обслуговування та потоку запитів у мережі вирішує ряд актуальних завдань, що мають важливе значення для забезпечення стабільності та ефективності мережевих систем. Деякі з цих завдань включають:

1. Оцінка пропускної здатності. Визначення максимальної кількості запитів, які мережа може обробити без втрати якості обслуговування, щоб планувати розширення або модернізацію інфраструктури.

2. Мінімізація затримок. Аналіз часу очікування запитів у черзі та часу їх обслуговування для зниження затримок та підвищення швидкості відгуку мережі.

3. Планування Ресурсів**.** Розподіл мережевих ресурсів на основі аналізу потоків даних, щоб забезпечити оптимальне використання обладнання та каналів зв'язку.

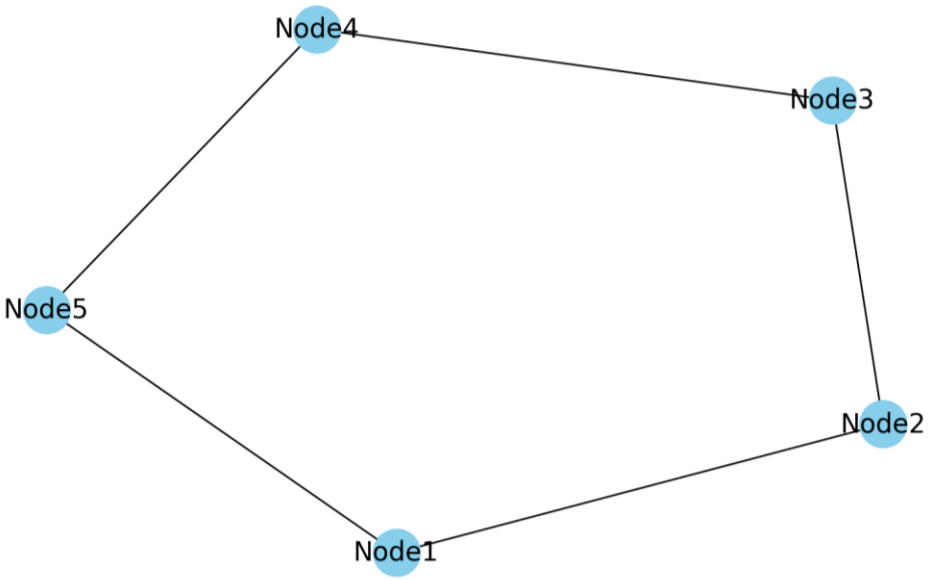

Рисунок 3.3 - Графове представлення тестової мережі

4. Виявлення та попередження перевантажень**.** Визначення моментів, коли інтенсивність потоку запитів перевищує можливості мережі, що дозволяє своєчасно вживати заходів для запобігання перевантажень.

5. Оптимізація відкладеного трафіку. Аналіз інтенсивності потоків дозволяє регулювати передачу даних для мінімізації впливу на критичні додатки у пікові години.

6. Керування пріоритетами. Налаштування пріоритетів для різних типів трафіку, забезпечуючи, щоб критично важливі дані оброблялися першочергово.

7. Аналіз надійності. Оцінювання здатності мережі протистояти коливанням в інтенсивності трафіку, що допомагає в підвищенні загальної надійності системи.

8. Прогнозування потреб. Аналіз поточних тенденцій допомагає у прогнозуванні майбутніх вимог до мережі та підготовці до зростання обсягів даних.

9. Розробка мережевої політики. Встановлення правил для контролю трафіку на основі аналізу, що забезпечує дотримання встановлених стандартів QoS (Quality of Service).

10. Ідентифікація ненормальних зростань в трафіку, які можуть бути ознаками DDoS-атак, та вживання заходів для їх нейтралізації.

Аналіз інтенсивності обслуговування та потоку запитів є фундаментальним для розуміння поведінки та виявлення потенційних проблем у мережевих системах, що підтримує високий рівень продуктивності та ефективності в умовах швидкого розвитку технологій та змін у патернах користування.

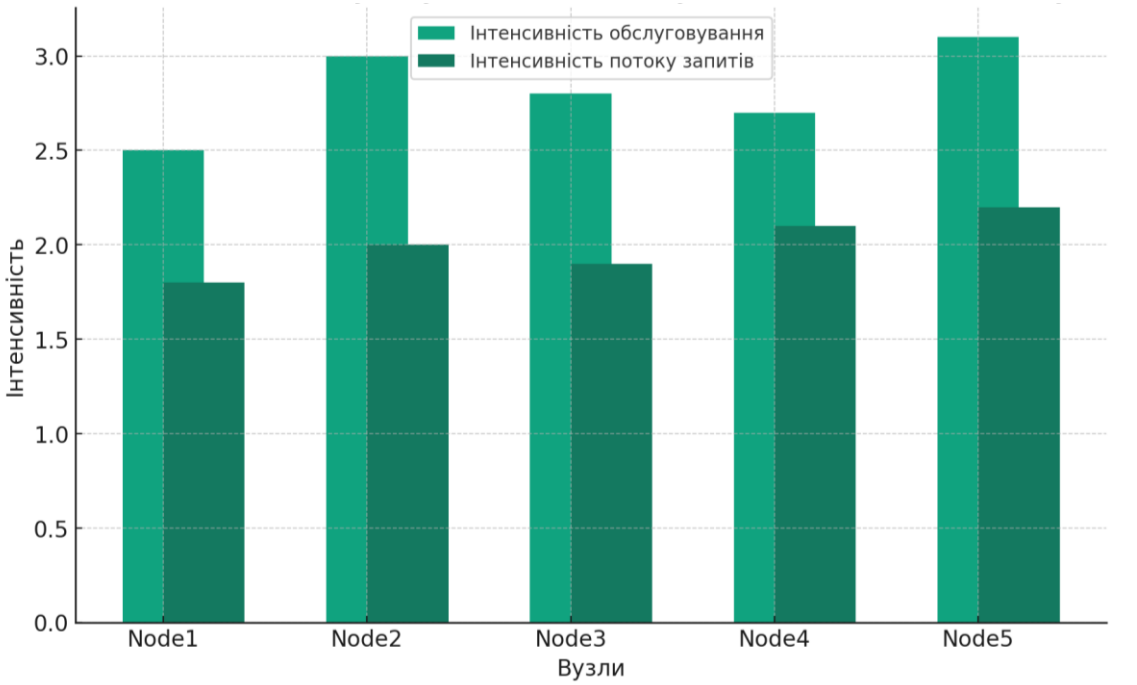

Рисунок 3.4 - Характеристики вузлів мережі

Цей графік ілюструє інтенсивність обслуговування та потоку запитів для кожного вузла у мережі.

3.4 Висновки до розділу 3

Розроблено та реалізовано програмне забезпечення імітаційного моделювання мереж на основі систем масового обслуговування. Для реалізації алгоритмів і моделей обрано Python на бібліотеку SimPy. Проведено експериментальне дослідження інтенсивності обслуговування та потоку запитів.

ВИСНОВКИ

1. Здійснено аналітичний огляд інформаційних моделей, алгоритмів та засобів імітаційного моделювання комп'ютерних мереж. Встановлено що особливістю мобільного трафіку є висока мобільність, переривчасте підключення, різноманітність пристроїв, потреба в рівномірному покритті сигналу. При моделюванні мереж з допомогою СМО використовують розподіли Пуассона, Експоненційний, Нормальний, та Вейбулла. Виділено основні параметри моделей СМО при описі мереж: інтенсивність потоку запитів, інтенсивність обслуговування, час заявки у системі, довжина черги, вірогідність відмови.

2. Розроблено інформаційну модель на основі систем масового обслуговування. Вузол мереж виступає як сервер у СМО, обробляючи пакети даних (запити). Пристрої, що ініціюють передачу даних розглядаються як клієнти у СМО. Черга в СМО відповідає буферу в мережі, де зберігаються пакети, що чекають на передачу або обробку. Частота, з якою вузли мережі генерують пакети даних, відповідає інтенсивності потоку запитів у СМО. Здатність вузла мережі обробляти або передавати пакети даних відповідає інтенсивності обслуговування сервера у СМО.

3. Розроблено структури представлення мережевого обладнання. Розроблено структури представлення комутаторів, маршрутизаторів, IoT шлюза та бездротового ретранслятора. Це дало змогу моделювання процесу опрацювання заявки-пакету в сервері системи масового обслуговування.

4. Розроблено та реалізовано програмне забезпечення імітаційного моделювання мереж на основі систем масового обслуговування. Для реалізації алгоритмів і моделей обрано Python на бібліотеку SimPy. Проведено експериментальне дослідження інтенсивності обслуговування та потоку запитів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Рудич М. В. Застосування пакету SimPy для моделювання роботи комп’ютерної мережі. Матеріали науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (ІКСМ-2023)., м. Тернопіль, 5 груд. 2023 р. С. 43.

2. Рудич М. В. Інформаційні моделі комп'ютерних мереж. Матеріали науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (ІКСМ-2023)., м. Тернопіль, 5 груд. 2023 р. С. 51.

3. Березький О.М., Дубчак Л.О., Мельник Г.М. Методичні рекомендації до виконання кваліфікаційної роботи освітнього ступеня «Магістр». Спеціальність: комп’ютерна інженерія. Магістерська програма – «Комп’ютерна інженерія» / Під ред. О.М. Березького. Тернопіль: ЗУНУ, 2020. 32 с.

4. Гураль І.В., Дубчак Л.О. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп’ютерна інженерія» / Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.

5. Коваленко О. Оптимізація архітектур модульних систем електронного навчання. *Математичне та комп’ютерне моделювання. Технічні науки*. 2008. Т. 1. С. 95-99. URL: 10.32626/2308-5916.2008-1.95-99.

6. Vlasova T., Kovalenko O., Kosolapov V. Organizational-Information Technology for Providing and Decisions Making in Situational Management. *Telecommunications and Computer Engineering, TCSET 2018: 14th International Conference on Advanced Trends in Radioelectronics*. 2018. P. 152-157. URL: 10.1109/TCSET.2018.8336176.

7. Коваленко О. Моделювання навантаження мережної інфраструктури систем ситуаційного управління. *Математичні машини і системи*. 2019. № 2. С. 101-110.

8. Литвинов А. Л. Теорія систем масового обслуговування. Харків : Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова, 2018. 141 с.

9. Литвинов А. Л. Розробка та дослідження ймовірнісних моделей оцінювання якості інформаційно-управляючих систем. *Комунальне господарство міст*. 2016. Т. 126. С. 22-26.

10. Бібліотека QueueSim. URL: https://github.com/A-Herzog/QueueSim.

11. Бібліотека Ciw. URL: https://ciw.readthedocs.io/en/latest/.

12. Бібліотека SimPy. URL: https://simpy.readthedocs.io/.

13. Модель телекомунікаційної мережі "інтелектуального будинку" / В. М. Теслюк та ін. *Науковий вісник НЛТУ України*. 2016. Т. 26.1. С. 351-357. URL: http://nbuv.gov.ua/UJRN/nvnltu_2016_26.1_56.

14. Використання XML для систем автоматизованого генерування моделей на основі мереж Петрі / В. М. Теслюк та ін. *Моделювання та інформаційні технології*. 2013. Т. 70. С. 129-136. URL: http://nbuv.gov.ua/UJRN/Mtit_2013_70_21.

15. Теслюк В. М. Моделі та інформаційні технології синтезу мікроелектромеханічних систем. Львів : Вежа і Ко, 2008. 192 с.

16. Ткаченко В. П., Огірко І. В., Огірко О. І. Інформаційна модель Грід технології. *Системи обробки інформації*. 2017. Т. 4(150). С. 88—91.

17. Любий Є., Бєлецька О. Експериментальні дослідження затримок транспортних засобів при виїзді з прилеглих територій. *Automash*. 2021. Т. 1, № 16. С. 106--116.

18. Shevchenko N. Y., Bagach S. G., Potapov D. S. Математичне обґрунтування вибору архітектури обчислювальної мережі як елемента інформаційної інфраструктури освітнього закладу. *Herald DSEA*. 2019. № 2 (46). С. 165--170.

19. Захарчук Д., Бурбела С., Мисик А. Особливості проведення спеціальних заходів з пошуку правопорушників на морській (річковій) ділянці державного кордону. *Збірник наукових праць Національної академії Державної прикордонної служби України. Серія: військові та технічні науки*. 2021. Т. 83, № 2. С. 92--114.

20. Фесенко Г. Особливості визначення можливостей наземного пункту управління БПЛА щодо реагування на позаштатні ситуації під час моніторингу територій з урахуванням помилок операторів та їх рівня підготовки. *enm*. 2020. № 3(13). С. 118--125.

21. Hohol T., Kindrat V. Теоретико-методологічне обґрунтування моделі формування логістичної компетентності майбутніх тренерів-викладачів. *Phys. Educ. Sport Health Cult. Mod. Soc.* 2022. № 2(58). С. 3--13.

22. Математична модель опису фону автомобільних транспортних засобів, що функціонують у контрольованій зоні біля стратегічного об'єкта / O. Azarenko та ін. *Journal of Scientific Papers "Social Development and Security"*. 2022. Т. 12, № 6. С. 180--190.

23. Semenov S., Lipchanska O., Lipchanskyi M. Аналіз методів управління передачею відеопотоку даних та вимог до якості їх передавання. *Системи управління навігації та зв'язку Збірник наукових праць*. 2018. Т. 3, № 49. С. 139--142.

24. Догтєва І., Шиян А. Відновлення групи реагування на інциденти інформаційної безпеки в умовах наростання інтенсивності кібератак. *Measuring*

and computing devices in technological processes. 2021. № 2. С. 21–29. URL: 10.31891/2219-9365-2021-68-2-3.

25. Шестопалов С. В., Кунуп Т. В., Пустовий Б. Л. Моделювання процесів управління наданням інтелектуальних сервісів в NGN. *Refrigeration Engineering and Technology*. 2018. Т. 54, № 3. URL: 10.15673/ret.v54i3.1117.

26. The Analysis of Performance for Priority Distinction Double-queue and Double-server Communication Network / J. L. Xiong et al. *International Journal of Communication*. 2014. Vol. 3. URL: https://api.semanticscholar.org/CorpusID:62382654.

27. Du J.-q. The strategy research and method realization for the computer network flow control. *International Conference on Graphic and Image Processing*. 2013. URL: https://api.semanticscholar.org/CorpusID:62720079.

28. Gao F., Liu Q., Zhan H. Research of SIP DoS Defense Mechanism Based on Queue Theory. *International Conference on Computer Science, Environment, Ecoinformatics, and Education*. 2011. URL: https://api.semanticscholar.org/CorpusID:14887144.

29. Minkevi{\vc}ius S., Sakalauskas L. On the law of iterated logarithm for extreme queue length in an open queueing network. *International Journal of Computer Mathematics: Computer Systems Theory*. 2021. Vol. 6. P. 220 - 235. URL: https://api.semanticscholar.org/CorpusID:237548367.

30. Stuckey N. Stochastic Estimation and Control of Queues within a Computer Network. 2012. URL: https://api.semanticscholar.org/CorpusID:59897689.

31. RouteNet-Erlang: A Graph Neural Network for Network Performance Evaluation / M. F. Galm{\'e}s et al. *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*. 2022. P. 2018-2027. URL: https://api.semanticscholar.org/CorpusID:247159041.

32. The Analysis of Performance for Priority Distinction Double-queue and Double-server Communication Network / J. L. Xiong et al. *International Journal of Communication*. 2014. Vol. 3. URL: https://api.semanticscholar.org/CorpusID:62382654.

33. Andrej{\vc}i{\vc} M. Analysis of the influence of queue type on packet delay in computer networks. 2015. URL: https://api.semanticscholar.org/CorpusID:195971842.

34. Queue Control Model in a Clustered Computer Network using M/M/m Approach / A. Ejem et al. *International Journal of Computer Trends and Technology*. 2016. Vol. 35. P. 12-20. URL: https://api.semanticscholar.org/CorpusID:38081651.

35. Розрахунок ефективності використання обчислювальних ресурсів самовідновлювальної комп’ютерної системи / N. Kuchuk та ін. *Системи управління, навігації та зв’язку. Збірник наукових праць*. 2021. Т. 3, № 65. С. 92–95. URL: 10.26906/sunz.2021.3.092.

36. Poslaiko N. I. Дослідження стаціонарного режиму в системі масового обслуговування типу gi /m /1 зі слабкою післядією. *Problems of applied mathematics and mathematic modeling*. 2022. URL: 10.15421/322119.

37. Літвін Н. Потік втрачених вимог у двоканальній системі масового обслуговування при малій інтенсивності вхідного потоку. *InterConf*. 2021. С. 431–436. URL: 10.51582/interconf.21-22.04.2021.049.

38. Konovaliuk V., Klyus I. Підхід до аналізу надійності функціонування комп'ютерної мережі системи організаційного управління. *Proceedings of the National Aviation University*. 2015. Т. 64, № 3. С. 111–114. URL: 10.18372/2306-1472.64.9036.

39. Volodymyr A. Клієнт-серверна модель інформаційної системи масового оповіщення населення. *TECHNICAL SCIENCES AND TECHNOLOG IES*. 2018. № 4 (14). С. 136–144. URL: 10.25140/2411-5363-2018-4(14)-136-144.

40. Коба О. В. Стаціонарні характеристики системи масового обслуговування слсл із поверненням при обслуговуванні в порядку черги. *Proceedings of the National Aviation University*. 2003. Т. 16, № 1. URL: 10.18372/2306-1472.16.11644.

41. Догтєва І., Шиян А. Відновлення групи реагування на інциденти інформаційної безпеки в умовах наростання інтенсивності кібератак. *MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES*. 2021. № 2. С. 21--29. URL: 10.31891/2219-9365-2021-68-2-3.

42. Гнатушенко В., Витовтов Г. Аналіз систем масового обслуговування при стрибкоподібній зміні інтенсивностей потоків інформації. *Applied Questions of Mathematical Modeling*. 2023. Т. 4, № 2.1. С. 76--83. URL: 10.32782/kntu2618-0340/2021.4.2.1.7.

43. Najim A. H., Mansour H. S., Abbas A. H. Characteristic Analysis of Queue Theory in Wi-Fi Applications Using OPNET 14.5 Modeler. *Eastern-European Journal of Enterprise Technologies*. 2022. Vol. 2, no. 9. P. 116. URL: 10.15587/1729-4061.2022.255520.

44. Queuing Theory Approach for Evaluating Rate of Transmission in Wireless Network Using Network Coding / M. Z. Ahmed et al. 2017.

45. Ray S., Starobinski D., Carruthers J. B. Performance of wireless networks with hidden nodes: A queuing-theoretic analysis. *Computer Communications*. 2005. Vol. 28, no. 10. P. 1179--1192.

46. A packet buffer evaluation method exploiting queueing theory for wireless sensor networks / T. Qiu et al. *Comput. Sci. Inf. Syst.* 2011. Vol. 8. P. 1028-1049. URL: https://api.semanticscholar.org/CorpusID:7164937.

47. Analysis of vehicular wireless channel communication via queueing theory model / S. Fowler et al. *2014 IEEE International Conference on Communications (ICC)*. 2014. P. 1736-1741. URL: 10.1109/ICC.2014.6883573.

48. Теслюк В. М. Дослідження і проектування комп’ютерних систем та мереж. Конспект лекцій. 62-ге вид. Тернопіль, 2012. URL: http://www.dgma.donetsk.ua/docs/kafedry/avp/metod/ПДКСМ%20Конспект%20лекц_й%20Теслюк.pdf (дата звернення: 01.11.2023).

49. Liu Y., Wu Y., Zhang J. Z. Modeling of C3I Communication Network based on Real-time Queuing Theory. 2011 IEEE International Conference on Computer Science and Automation Engineering. 2011. Vol. 1. P. 109-112. URL: https://api.semanticscholar.org/CorpusID:2362913.

50. Joint queue-perturbed and weakly coupled power control for wireless backbone networks / T. O. Olwal et al. International Journal of Applied Mathematics and Computer Science. 2012. Vol. 22. P. 749 - 764. URL: https://api.semanticscholar.org/CorpusID:11693778.