МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Кваліфікаційна наукова
праця на правах рукопису

ЖИДКА ОЛЬГА ВАЛЕРІЇВНА

УДК 004.738.5:004.912:004.62:004.942

ДИСЕРТАЦІЯ

МОДЕЛІ І АЛГОРИТМИ ІНФОРМАЦІЙНОЇ ВЗАЄМОДІЇ ДЛЯ МЕРЕЖ ІНТЕРНЕТУ РЕЧЕЙ

Спеціальність 123 − Комп’ютерна інженерія
Галузь знань 12 − Інформаційні технології
Подається на здобуття наукового ступеня
доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____ О.В. Жидка

Науковий керівник: СТОРЧАК Каміла Павлівна, доктор технічних наук, професор

Київ - 2025

АНОТАЦІЯ

Жидка О. В. Моделі і алгоритми інформаційної взаємодії для мереж Інтернету речей. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії в галузі знань 12 – Інформаційні технології, за спеціальністю 123 – Комп'ютерна інженерія. – Державний університет інформаційно-комунікаційних технологій. – Київ, 2025.

Дисертаційна робота присвячена актуальній науковій задачі розробки моделей і алгоритмів інформаційної взаємодії, які забезпечують ефективність, надійність і безпеку обміну даними в мережах Інтернету речей (IoT) з урахуванням їх децентралізованої структури та обмежених ресурсів.

Тематика дисертаційного дослідження відповідає стандарту та фаховим компетентностям освітньо-наукової програми підготовки докторів філософії з комп'ютерної інженерії Державного університету інформаційно-комунікаційних технологій Міністерства освіти і науки України, а саме: фундаментальним науковим дослідженням теоретико-методологічних, науково-методичних та прикладних засад розробки моделей і алгоритмів, спрямованих на підвищення ефективності, надійності та безпеки інформаційної взаємодії в мережах Інтернету речей, а також вдосконалення процесу впровадження новітніх інформаційних технологій в системах децентралізованої обробки даних.

Актуальність дослідження моделей і алгоритмів інформаційної взаємодії для мереж Інтернету речей, можна обґрунтувати наступним чином:

1. Зростання кількості IoT-пристроїв. З розвитком технологій IoT спостерігається стрімке збільшення кількості підключених сенсорних пристроїв і систем, що генерують великі обсяги даних та вимагають ефективної інформаційної взаємодії. Це ставить перед дослідниками завдання оптимізації методів комунікації та передачі даних, щоб забезпечити надійну роботу мереж з великою кількістю вузлів.

2. Обмежені ресурси IoT-пристроїв. Багато сенсорних вузлів мають обмежену потужність, пропускну здатність, пам'ять та енергетичні ресурси. Тому важливо розробляти алгоритми і моделі, які оптимізують використання цих ресурсів та забезпечують тривалу і стабільну роботу систем.

3. Надійність і безперебійність взаємодії. У реальних умовах існують проблеми, пов'язані з виходом з ладу окремих елементів мережі, колізіями під час передачі даних, зміною топології мережі або непередбачуваними перешкодами в каналах зв'язку. Ефективні моделі та алгоритми інформаційної взаємодії повинні забезпечувати стійкість мережі до таких проблем, що є ключовим аспектом для IoT.

4. Якість обслуговування. У багатьох застосуваннях IoT (розумні міста, промисловий Інтернет речей, системи охорони здоров’я) важливим є забезпечення певного рівня якості обслуговування: мінімізація затримок, гарантія доставки даних та захист від втрат. Моделі, які дозволяють оцінювати і оптимізувати ці параметри, є важливими для створення надійних і ефективних IoT-систем.

5. Актуальність для майбутніх технологій. Розвиток технологій 5G/6G, а також поява нових стандартів для бездротових сенсорних мереж створюють нові вимоги до мереж IoT. Це включає покращену пропускну здатність, менші затримки і вищу енергоефективність. Тому необхідність дослідження нових моделей і алгоритмів інформаційної взаємодії є невід'ємною частиною розвитку інфраструктури майбутнього.

6. Мультиагентні системи та самоорганізація мережі. В умовах масштабних і динамічних мереж IoT важливо, щоб пристрої могли самостійно адаптуватися до змін середовища і взаємодіяти один з одним без постійного втручання людини. Розробка моделей та алгоритмів самоорганізації допомагає вирішити ці завдання, підвищуючи автономність і ефективність системи.

Таким чином, дослідження моделей і алгоритмів інформаційної взаємодії для мереж Інтернету речей є вагомим кроком до створення більш надійних, масштабованих і ефективних IoT-систем, що здатні відповідати

сучасним викликам і забезпечувати стабільне функціонування в умовах складної та динамічної мережі.

Специфіка інформаційної взаємодії в Інтернеті речей визначається ключовими характеристиками IoT, зокрема: зв'язаністю (можливістю підключення речей до глобальної мережі), гетерогенністю (поєднанням різних платформ і мереж), динамічними змінами стану пристроїв, а також масштабом (десятки трильйонів речей і мереж). Ці особливості забезпечують надання послуг для фізичних і віртуальних об'єктів, зокрема автономно.

Перелічені особливості не дозволяють застосовувати методи та алгоритми, на основі яких функціонують інфраструктурні обчислювальні мережі, до Інтернету речей. Це зумовлено кількома причинами: зв'язаність і динамічні зміни роблять топологію мережі нерегулярною, використання бездротових технологій, «туманних» і «хмарних» обчислень спричиняє колізії джерел даних при доступі до ресурсів IoT, а гетерогенність і вимоги до енергоефективності впливають на ймовірнісно-часові характеристики інформаційної взаємодії в мережах Інтернету речей.

Отже, зважаючи на особливості мереж IoT як об'єкта дослідження та потребу у впровадженні нових моделей і алгоритмів, була визначена мета дисертаційного дослідження: розробка моделей і алгоритмів, що забезпечують вибір оптимальних режимів інформаційної взаємодії в мережах IoT.

Відповідно до поставленої мети в роботі були сформульовані наступні наукові завдання:

1. Дослідити організацію мереж IoT, включаючи їх архітектуру, види та способи інформаційно взаємодії, класифікацію протоколів передачі даних, а також аспекти інформаційної безпеки.

2. Розробити імітаційну модель інформаційної взаємодії в мережі IoT.

3. Розробити комплекс математичних моделей для оцінки ймовірнісно-часових характеристик інформаційної взаємодії в мережі IoT.

4. Дослідити залежності ймовірнісно-часових характеристик від параметрів мережі IoT.

5. Розробити ймовірнісні алгоритми самоорганізації мережі IoT, зокрема генетичні алгоритми для оптимізації маршрутизації та пошуку альтернативних маршрутів.

У першому розділі дисертації проведено аналіз сучасних технологій і архітектурних рішень IoT-мереж, зокрема їх ключових компонентів: пристроїв, вузлів, шлюзів та методів взаємодії (бездротових і провідних технологій). Розглянуто централізовані, децентралізовані й розподілені архітектури, а також протоколи MQTT, CoAP, LoRaWAN, які забезпечують ефективну комунікацію. Особливу увагу приділено викликам, таким як енергоспоживання, масштабованість, безпека даних і управління мережею. Проаналізовано загрози безпеці IoT-систем і надано рекомендації щодо їх захисту та забезпечення цілісності мережі.

У другому розділі здійснено аналіз та порівняння сучасних засобів моделювання IoT. Також запропоновано імітаційну модель інформаційної взаємодії в мережі Інтернету речей. Одним з важливих аспектів цієї моделі є можливість моделювання ймовірних сценаріїв відключення або повторного підключення агентів до мережі, що є типовим для мобільних або динамічних середовищ IoT. Імітаційна модель дозволяє враховувати такі фактори, як періодичне відключення від мережі або затримки в передачі даних через зміну топології.

У розділі розглянуто і проаналізовано два підходи до оцінки ймовірнісно-часових характеристик інформаційної взаємодії в мережах IoT. Основну увагу було зосереджено на методах моделювання таких систем, а також на особливостях передачі даних і управління ресурсами в умовах різних архітектур IoT, зокрема «туманних обчислень» і мережевих топологій mesh.

У третьому розділі здійснено аналіз та оцінювання різних алгоритмів самоорганізації бездротових сенсорних мереж з метою визначення найбільш ефективного для застосування в мережах Інтернету речей.

Також в розділі детально розглянуто та запропоновано для використання в мережах IoT два алгоритми, що базуються на еволюційній теорії. Алгоритм

формування альтернативних маршрутів враховує накопичену статистику щодо доставки даних між вузлами. Якщо ймовірність доставки даних нижча за заданий поріг, канал не використовується для побудови альтернативного маршруту. Алгоритм самоорганізованого розміщення сенсорних пристроїв дозволяє оптимізувати розташування сенсорних вузлів у мережі, враховуючи вимоги до покриття, енергоефективності та мінімізації втрат даних. Для цього було досліджено стратегії маршрутизації даних у безпровідних сенсорних мережах і запропоновано генетичний алгоритм для оптимального визначення маршруту між вузлами мережі.

Були отримані наступні наукові результати:

1. Вперше розроблено адаптивну математичну модель доступу в туманних обчисленнях, яка за рахунок коефіцієнту завантаженості мережі дозволяє зменшити середній час передачі даних в мережах IoT.

2. Удосконалено ймовірнісну модель інформаційної взаємодії в мережі IoT з топологією mesh, яка на відміну від існуючих дозволяє зменшити час встановлення з'єднання за рахунок впливу інтервалу втрати роутера.

3. Вперше розроблено метод самоорганізації мережі IoT на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, який за рахунок коефіцієнту покриття території та знаходження найкоротшого маршруту дозволяє зменшити час передачі даних.

Дисертація виконувалась в Державному університеті інформаційно-комунікаційних технологій. Обраний напрям досліджень відповідає тематиці науково-дослідних робіт Державного університету інформаційно-комунікаційних технологій.

Ключові слова: Інтернет речей, IoT, інформаційна взаємодія, методи множинного доступу, туманні обчислення, хмарні обчислення, оптимізація, сенсорні пристрої, канали зв’язку, модель, інформаційна безпека, безпека мережі, системи захисту інформації, бездротовий зв’язок, бездротова сенсорна мережа.

ABSTRACT

Zhidka O. V. Models and algorithms of information interaction for Internet of Things networks. – Qualifying scientific work on manuscript rights.

Dissertation for obtaining the scientific degree of Doctor of Philosophy in the field of knowledge 12 − Information technologies, specialty 123 − Computer engineering. − State University of Information and Communication Technologies. − Kyiv, 2025.

The dissertation is devoted to the actual scientific task of developing models and algorithms of information interaction that ensure the efficiency, reliability and security of data exchange in Internet of Things networks, taking into account their decentralized structure and limited resources.

The subject of the dissertation research aligns with the standards and professional competencies of the educational and scientific program for training Doctors of Philosophy in Computer Engineering at the State University of Telecommunications under the Ministry of Education and Science of Ukraine. Specifically, it pertains to fundamental scientific research in the theoretical-methodological, scientific-methodological, and applied foundations for developing models and algorithms aimed at improving the efficiency, reliability, and security of information interaction within Internet of Things networks. Additionally, it addresses the enhancement of processes for integrating innovative information technologies into decentralized data processing systems.

The relevance of researching models and algorithms for information interaction in IoT networks can be justified as follows:

1. Growth in the number of IoT devices. The rapid expansion of IoT technologies has led to an exponential increase in connected sensor devices and systems, generating vast amounts of data and requiring efficient information interaction. This poses challenges for researchers to optimize communication and data transfer methods to ensure reliable operation in networks with a high number of nodes.

2. Limited resources of IoT devices. Many sensor nodes have constrained capabilities, such as limited power, bandwidth, memory, and energy resources. Developing algorithms and models that optimize resource usage and ensure long-term, stable system operation is therefore critical.

3. Reliability and continuity of interaction. In real-world conditions, issues such as node failures, data transmission collisions, network topology changes, or unpredictable channel interference arise. Effective models and algorithms for information interaction must ensure network resilience to such challenges, a key aspect for IoT systems.

4. Quality of service (QoS). In numerous IoT applications − smart cities, industrial IoT, and healthcare systems, for instance − ensuring a certain level of service quality is crucial. This includes minimizing latency, guaranteeing data delivery, and protecting against losses. Models that evaluate and optimize these parameters are essential for creating reliable and efficient IoT systems.

5. Relevance to future technologies. The development of 5G/6G technologies and the emergence of new standards for wireless sensor networks impose new requirements on IoT networks, such as enhanced bandwidth, reduced latency, and increased energy efficiency. Therefore, researching new models and algorithms for information interaction is a vital component of developing future-proof infrastructure.

6. Multi-agent systems and network self-organization. In large-scale, dynamic IoT networks, it is critical for devices to independently adapt to environmental changes and interact with each other without continuous human intervention. Developing self-organizing models and algorithms addresses these needs, increasing system autonomy and efficiency.

Research into models and algorithms for information interaction in IoT networks is a vital step towards creating more reliable, scalable, and efficient IoT systems. Such systems must meet modern challenges and ensure stable operation in complex and dynamic network environments.

The specifics of information interaction in the Internet of Things are

determined by its key characteristics, including connectivity (the ability to connect things to the global network), heterogeneity (integration of various platforms and networks), dynamic changes in device states, and scale (tens of trillions of devices and networks). These features enable the provision of services for physical and virtual objects, including autonomous operation.

The listed characteristics prevent the application of methods and algorithms used in traditional infrastructure computing networks to the Internet of Things. This is due to several factors: connectivity and dynamic changes make network topology irregular, the use of wireless technologies, fog, and cloud computing causes data source collisions when accessing IoT resources, and heterogeneity combined with energy efficiency requirements impacts the probabilistic and temporal characteristics of information interaction in IoT networks.

Thus, considering the specifics of IoT networks as a research object and the need for new models and algorithms, the goal of the dissertation research was formulated: to develop models and algorithms that facilitate the selection of optimal modes of information interaction in Internet of Things networks.

According to the stated goal, the following scientific tasks were formulated in the research:

1. Investigate the organization of Internet of Things networks, including their architecture, types, and methods of information interaction, classification of data transmission protocols, as well as aspects of information security.

2. Develop a simulation model of information interaction in the IoT network.

3. To develop a set of mathematical models for estimating the probabilistic-temporal characteristics of information interaction in the IoT network.

4. Investigate the dependence of probability-time characteristics on IoT network parameters..

5. Develop probabilistic algorithms for IoT network self-organization, in particular genetic algorithms for routing optimization and finding alternative routes.

The first chapter of the dissertation analyzes modern technologies and architectural solutions for IoT networks, focusing on their key components: devices,

nodes, gateways, and interaction methods (wireless and wired technologies). Centralized, decentralized, and distributed architectures are examined, along with protocols such as MQTT, CoAP, and LoRaWAN, which enable efficient communication. Special attention is given to challenges such as energy consumption, scalability, data security, and network management. The chapter also analyzes security threats to IoT systems and provides recommendations for their protection and ensuring the integrity of the network.

The second chapter analyzes and compares modern IoT modeling tools. Additionally, a simulation model of information interaction in the Internet of Things network is proposed. One of the key aspects of this model is its ability to simulate probable scenarios of agent disconnection or reconnection to the network, which is typical for mobile or dynamic IoT environments. The simulation model accounts for factors such as periodic network disconnections or data transmission delays caused by changes in network topology.

The chapter examines and analyzes two approaches to evaluating the probabilistic and temporal characteristics of information interaction in IoT networks. The primary focus is on modeling methods for such systems, as well as the specifics of data transmission and resource management within various IoT architectures, particularly fog computing and mesh-based network topologies.

The third chapter presents an analysis and evaluation of various self-organizing algorithms for wireless sensor networks to identify the most effective ones for application in IoT networks.

The chapter also details and proposes two algorithms based on evolutionary theory for use in IoT networks. The alternative route formation algorithm considers accumulated statistics on data delivery between nodes. If the probability of data delivery falls below a set threshold, the channel is not used for constructing an alternative route. The self-organizing sensor placement algorithm optimizes the positioning of sensor nodes within the network, taking into account coverage requirements, energy efficiency, and data loss minimization. To achieve this, data routing strategies for wireless sensor networks were investigated, and a genetic

algorithm was proposed for optimal route search between network nodes.

The following scientific results were obtained:

1. For the first time, an adaptive mathematical model of access in fog computing was developed, which allows reducing the average data transfer time in IoT networks due to the network load factor.

2. The probabilistic model of information interaction in the IoT network with mesh topology has been improved, which, unlike the existing ones, allows to reduce the connection establishment time due to the influence of the router loss interval.

3. For the first time, a method of self-organization of the IoT network was developed based on probabilistic algorithms based on genetic approaches, which, due to the coefficient of territory coverage and finding the shortest route, allows reducing the time of data transmission.

Keywords: Internet of things, IoT, information interaction, multiple access methods, fog computing, cloud computing, optimization, sensor devices, communication channels, model, information security, network security, information protection systems, wireless communication, wireless sensor network.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Наукові праці, у яких опубліковані основні наукові результати дисертації:

1. Бичков В.В., Жидка О.В., Дослідження протоколів та технологій передачі даних у мережах IoT, Зв’язок, № 3(169), с. 28–32, 2024.

2. Жидка О.В., Андрійченко Т.Р., Інформаційна безпека систем IoT, Зв’язок, № 4(170), с. 65−69, 2024.

3. Olga Zhydka, Myroslav Riabyi, Tetiana Okhrimenko, Andriy Fesenko, Lazat Kydyralina, Research on data transmission technologies and information security in IoT networks, CEUR Workshop Proceedings, Social Communication and Information Activity in Digital Humanities 2024, 3851, ISSN 1613-0073.

4. Жидка О.В., Дакова Л.В., Даков С.Ю., Поляшенко Д.В., Analysis of the use of software honeypots in Internet of Things, Наукові записки ДУІКТ №2, с. 92-98, 2024.

5. Жидка О.В., Бондаренко Д. А., Модель оцінки ймовірнісно-часових параметрів інформаційної взаємодії в Інтернеті речей, Зв’язок, № 5(171), с. 59−66, 2024.

6. Жидка О.В., Ймовірнісна модель встановлення інформаційної взаємодії в мережі Інтернету речей з топологією mesh, Телекомунікаційні та інформаційні технології, № 4, с. 43-52, 2024.

7. XIII міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій»; Машинне навчання; 21 жовтня 2021р., Державний університету телекомунікацій, м. Київ.

8. I Всеукраїнська науково-технічна конференція «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу»; Найбільші виклики безпеці та рішення для IoT; 28 листопада 2023р., Державний університету телекомунікацій, м. Київ.

9. V Науково-технічна конференція «Сучасний стан та перспективи розвитку IoT»; Глобальна бізнес-можливість Інтернету речей; 18 квітня

2024р., Державний університету телекомунікацій, м. Київ.

10. VII Міжнародна науково-практична конференція «Проблеми кібербезпеки інформаційно-телекомунікаційних систем»; Методика забезпечення інформаційної безпеки IoT; 26 квітня 2024р., Київський національний університет ім. Тараса Шевченка, м. Київ.

11. IV Всеукраїнська науково-технічна конференція молодих вчених, аспірантів та студентів «Комп'ютерні ігри і мультимедіа як інноваційний підхід до комунікації − 2024»; IoT в комп'ютерних іграх і мультимедіа; 26−27 вересня 2024р., Одеський національний технологічний університет, м. Одеса.

12. Всеукраїнська науково-практична конференція здобувачів вищої освіти і молодих учених «Моделювання та практичне забезпечення розвитку складових професіоналізму педагогів закладів освіти»; Інтернет речей у здоров'язбережувальних технологіях початкової школи; 15-16 жовтня 2024р., Комунальний заклад вищої освіти «Хортицька національна навчально-реабілітаційна академія» Запорізької обласної ради, м. Запоріжжя.

13. II Всеукраїнська науково-технічна конференція «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу»; Аналіз режимів доступу до ресурсів у мережах Інтернету речей; 18 листопада 2024р., Державний університету телекомунікацій, м. Київ.

14. VII Всеукраїнська науково-технічна конференція «Комп'ютерні технології: інновації, проблеми, рішення»; Аналіз засобів моделювання мережі Інтернету речей; 02-03 грудня 2024р., Державний університет «Житомирська політехніка», м. Житомир.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 16
ВСТУП 18
1 ДОСЛІДЖЕННЯ ОРГАНІЗАЦІЇ МЕРЕЖ ІНТЕРНЕТУ РЕЧЕЙ 22
1.1 Аналіз технології побудови мереж Інтернету речей 22
1.2 Архітектура мережі Інтернету речей 29
1.3 Огляд способів взаємодії в мережі Інтернету речей 37
1.4 Види взаємодії в Інтернеті речей 46
1.5 Протоколи взаємодії в Інтернеті речей 49
1.5.1 Класифікація протоколів передачі даних в мережах IoT 49
1.5.2 Технології та протоколи передачі даних на довгі відстані в IoT-мережах 52
1.6 Інформаційна безпека систем IoT 61
1.7 Постановка завдання та мети дослідження 66
1.8 Висновки до розділу 68
2 МОДЕЛІ ОЦІНКИ ЙМОВІРНОСНО-ЧАСОВИХ ХАРАКТЕРИСТИК ІНФОРМАЦІЙНОЇ ВЗАЄМОДІЇ В МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ 70
2.1 Аналіз засобів моделювання мережі Інтернету речей 70
2.2 Імітаційна модель інформаційної взаємодії 76
2.3 Моделі доступу в «туманних обчисленнях» з роздільною здатністю колізій джерел даних 79
2.4 Ймовірнісна модель встановлення інформаційної взаємодії в мережі Інтернету речей з топологією mesh 90
2.5 Висновки до розділу 101
3 ЙМОВІРНІСНІ АЛГОРИТМИ САМООРГАНІЗАЦІЇ МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ 104
3.1 Аналіз алгоритмів самоорганізації мережі Інтернету речей 104
3.2 Генетичний алгоритм самоорганізації мережі IoT 112
3.3 Генетичний алгоритм пошуку альтернативних маршрутів у мережі IoT 117
3.4 Висновки до розділу 127
4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ВЗАЄМОДІЇ В МЕРЕЖАХ ІНТЕРНЕТУ РЕЧЕЙ 130

4.1 Аналіз результатів моделювання доступу у туманних обчисленнях ... 130
4.1.1 Опис моделі туманного обчислення ... 130
4.1.2 Оцінка часу відповіді туманного вузла на запити сенсорних пристроїв у різних режимах доступу ... 132
4.1.3 Оцінка залежності кількості підключених пристроїв від навантаження вузла ... 136
4.1.4 Оцінка залежності середнього часу передачі даних від навантаження вузла ... 140
4.2 Аналіз результатів моделювання інформаційної взаємодії в мережі Інтернету речей з топологією mesh ... 143
4.2.1 Опис моделі mesh-мережі ... 143
4.2.2 Аналіз часу встановлення з'єднання в mesh-мережі в залежності від надійності з'єднання ... 146
4.2.3 Аналіз часу встановлення з'єднання в mesh-мережі в залежності від надійності вузлів ... 148
4.3 Оцінка працездатності алгоритмів самоорганізації ... 151
4.3.1 Опис моделі генетичного алгоритму розміщення СП ... 151
4.3.2 Аналіз ефективності вирішення задачі заповнення площини СП .. 153
4.3.3 Аналіз ефективності вирішення задачі в залежності від ймовірності мутації ... 155
4.4 Висновки до розділу ... 159
ВИСНОВКИ ... 161
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ... 164
ДОДАТОК А ... 177
ДОДАТОК Б ... 187
ДОДАТОК В ... 189
ДОДАТОК Г ... 190

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АЦП − аналого-цифровий перетворювач

API – (Application Programming Interface) інтерфейс програмування додатків

БСМ − бездротова сенсорна мережа

BLE – (Bluetooth Low Energy) блютуз з низьким енергоспоживанням

ГА – генетичний алгоритм

EDGE − (Enhanced Data rates for GSM Evolution) технологія мобільного зв'язку

GSM – (Global System for Mobile Communications) глобальна система мобільного зв'язку

GPS – (Global Positioning System) глобальна система супутникової навігації

HTTP – (Hypertext Transfer Protocol) протокол передавання гіпертексту

HTTPS − (Hypertext Transfer Protocol Secure) безпечний протокол передачі гіпертексту

IoT – Інтернет речей

ІВ – інформаційна взаємодія

I2C – (Inter-Integrated Circuit) міжінтегральна схема

LTE − (Long Term Evolution) технологія мобільного зв'язку четвертого покоління

LISP – (Locator/ID Separation Protocol) мережевий протокол

LPWAN − (Low Power Wide Area Network) мережа широкого покриття з низьким енергоспоживанням

MITM − (Man-in-the-Middle) людина посередині

МСЕ-Т − Міжнародна спілка електрозв'язку – Технічний сектор

NAS − (Non-Access Stratum) стратум без доступу

NFC – (Near Field Communication) комунікація на короткому радіусі

NS − (Network Simulator 3) симулятор мереж

OSPF – (Open Shortest Path First) відкритий найкоротший шлях першим

PAN – (Personal Area Network) персональна зона мережі

ПЛС − перетворення Лапласа-Стілтьєса

QoS – (Quality of service) якість обслуговування

RESTful API − (Representational State Transfer) інтерфейс програмування додатків на основі передавання стану представлення

СП – сенсорний пристрій

СВ – сенсорний вузел

СМО − система масового обслуговування

TCP − (Transmission Control Protocol) протокол управління передачею

UWB – (Ultra Wideband) ультраширокосмугова

UMTS − (Universal Mobile Telecommunications System) універсальна система мобільних телекомунікацій

USB − (Universal Serial Bus) універсальна послідовна шина

Wi-Fi − (Wireless Fidelity) технологія бездротового зв'язку

WiMAX − (Worldwide Interoperability for Microwave Access) технологія бездротового широкосмугового зв'язку

Z-Wave − технологія бездротової комунікації для автоматизації будинків і смарт-мереж

ВСТУП

Дисертаційне дослідження присвячено актуальній науковій задачі підвищення ефективності функціонування мереж Інтернету речей шляхом розробки моделей і алгоритмів, що оптимізують інформаційну взаємодію.

Технологія IoT знаходиться на етапі активних досліджень, її розвиток обіцяє значні перспективи в різних галузях промисловості – виробництві, логістиці, медицині, енергетиці, транспорті, міському господарстві, управлінні надзвичайними ситуаціями, а сфери її застосування продовжують стрімко розширюватися.

На міжнародному рівні концепція IoT вже почала набувати рис сформованої технології – ведеться активна робота у напрямку стандартизації архітектури, технічних компонентів і застосувань. Однак одночасно виникає потреба в розробці нових моделей і алгоритмів, які враховують особливості інформаційної взаємодії в мережах Інтернету речей.

Особливості інформаційної взаємодії визначаються фундаментальними характеристиками IoT-технології, до яких, відповідно до рекомендацій Міжнародного Союзу Електрозв'язку, належать:

- зв'язність – можливість будь-якої речі бути підключеною до глобальної інфокомунікаційної структури;

- забезпечення речей послугами – надання мережевих послуг без обмежень, у тому числі автономно для фізичних і віртуальних речей;

- гетерогенність – характеристика Інтернету речей, що визначає побудову IoT-пристроїв на різних апаратних, програмних платформах та мережах;

- динамічні зміни – характеристика, що визначає динамічні зміни статусу речей, наприклад, перехід від сплячого стану до активного, від підключених до певної мережі до непідключених тощо. Кількість речей, їх місцезнаходження, швидкість та інші параметри також можуть змінюватися динамічно;

- велика кількість речей – характеристика, яка ґрунтується на прогнозних оцінках, відповідно до яких йдеться про десятки трильйонів речей і трильйонні мережі.

Перераховані особливості роблять неможливим застосування методів і алгоритмів, на яких працюють традиційні інфраструктурні обчислювальні мережі, до Інтернету речей. Це пояснюється кількома причинами: зв'язність і динамічні зміни створюють нерегулярну топологію мережі, використання бездротових технологій, «туманних» і «хмарних» обчислень призводить до появи колізій джерел даних при доступі до ресурсів IoT, а гетерогенність та інші вимоги, такі як енергоефективність, впливають на ймовірнісно-часові характеристики інформаційної взаємодії в мережах Інтернету речей.

Отже, специфіка мереж IoT як об'єкта дослідження та необхідність розробки нових моделей і алгоритмів для оптимізації організації інформаційного взаємодії в таких мережах визначають актуальність теми дисертаційної роботи.

Об'єктом дослідження є процес інформаційної взаємодії в комп’ютерних мережах.

Предмет дослідження − моделі і алгоритми інформаційної взаємодії для комп’ютерних мереж.

Метою дисертаційного дослідження є розробка моделей і алгоритмів, що сприяють вибору оптимальних режимів інформаційної взаємодії в мережах Інтернету речей.

Методами дослідження є математичне моделювання, ймовірнісно-статистичний аналіз, методи оптимізації, комп’ютерне моделювання.

Відповідно до поставленої мети були сформульовані наступні наукові завдання:

1. Дослідити організацію мереж IoT, включаючи їх архітектуру, види та способи інформаційно взаємодії, класифікацію протоколів передачі даних, а також аспекти інформаційної безпеки.

2. Розробити імітаційну модель інформаційної взаємодії в мережі IoT.

3. Розробити комплекс математичних моделей для оцінки ймовірнісно-часових характеристик інформаційної взаємодії в мережі IoT.

4. Дослідити залежності ймовірнісно-часових характеристик від параметрів мережі IoT.

5. Розробити ймовірнісні алгоритми самоорганізації мережі IoT, зокрема генетичні алгоритми для оптимізації маршрутизації та пошуку альтернативних маршрутів.

Рішення проблеми, сформульованої в дисертаційній роботі, щодо розробки моделей і алгоритмів інформаційної взаємодії в мережах Інтернету речей базується на методах системного аналізу, теорії ймовірностей, випадкових процесів і математичної статистики, методах числового аналізу та іммітаційного моделювання.

Для досягнення поставленої мети дисертаційної роботи були вирішені завдання, що визначають новизну дослідження:

1. Вперше розроблено адаптивну математичну модель доступу в туманних обчисленнях, яка за рахунок коефіцієнту завантаженості мережі дозволяє зменшити середній час передачі даних в мережах IoT.

2. Удосконалено ймовірнісну модель інформаційної взаємодії в мережі IoT з топологією mesh, яка на відміну від існуючих дозволяє зменшити час встановлення з'єднання за рахунок впливу інтервалу втрати роутера.

3. Вперше розроблено метод самоорганізації мережі IoT на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, який за рахунок коефіцієнту покриття території та знаходження найкоротшого маршруту дозволяє зменшити час передачі даних.

Особистий внесок здобувача: основні положення та результати дисертаційної роботи отримані автором самостійно. Автор виконав усі теоретичні та практичні дослідження, що становлять основу дисертаційної роботи.

Основні результати за темою дисертаційного дослідження опубліковані в 14 наукових публікаціях. Серед праць 5 наукових статей опубліковані в

фахових виданнях України та 1 стаття опублікована у наукових виданнях Scopus. Апробація відбувалась на конференціях різного рівня в період з 2021 по 2024 роки, матеріали яких опубліковано у 8 працях наукових конференцій.

Структура та обсяг дисертації : дисертація складається з анотації, змісту, переліку умовних скорочень, вступу, чотирьох розділів, загальних висновків, списку використаних джерел і має 162 сторінки основного тексту, 57 рисунків, 4 таблиці. Список використаних джерел містить 105 найменувань і займає 12 сторінок. Загальний обсяг дисертаційної роботи – 190 сторінок.

Практична цінність дисертації полягає у розробці моделей та алгоритмів, які підвищують ефективність, надійність і адаптивність мереж Інтернету речей. Запропоновані рішення дозволяють оптимізувати передачу даних, зменшити затримки, уникнути колізій і забезпечити стабільну роботу систем навіть у складних умовах.

Залежності ймовірнісно-часових характеристик, отримані в роботі, є цінним інструментом для інженерів на етапах проєктування IoT-систем, сприяючи оптимальному вибору параметрів і зниженню витрат на впровадження.

1 ДОСЛІДЖЕННЯ ОРГАНІЗАЦІЇ МЕРЕЖ ІНТЕРНЕТУ РЕЧЕЙ

1.1 Аналіз технології побудови мереж Інтернету речей

Для реалізації цілей технології Інтернету речей, яка охоплює інтеграцію різноманітних сенсорних пристроїв та гетерогенних комп'ютерних і сенсорних мереж, потрібна відповідна архітектура. У рамках діяльності сектора стандартизації телекомунікацій Міжнародного союзу електрозв'язку (МСЕ-Т) рекомендована структура IoT поділяється на чотири основні рівні:

- рівень взаємодії з середовищем – включає сенсори і пристрої, що безпосередньо взаємодіють з фізичним оточенням;
- мережевий рівень – відповідає за забезпечення передачі даних між пристроями та іншими компонентами системи;
- сервісний рівень – забезпечує обробку, аналіз та перетворення зібраної інформації;
- рівень взаємодії з користувачем або додатками – забезпечує інтерфейси для взаємодії користувачів або програм з мережею IoT [1].

Ця архітектура відповідає важливим характеристикам IoT-систем:

- легкість модернізації;
- модульність;
- масштабованість мережі;
- адаптивність, яка дає змогу системам управління пристосовуватися до змін або модифікувати мережу залежно від умов її функціонування.

З позиції системного аналізу під час проєктування IoT-мереж необхідно визначити:

- топологію мережі;
- структуру та склад ключових елементів;
- основні взаємозв'язки між елементами;
- визначати призначення та завдання мережі;
- визначити критерії та індикатори оптимальної організації

інформаційної взаємодії.

Технологія IoT виникла на основі територіально розподілених сенсорних мереж, які спершу розроблялися як локальні системи збору даних з поступовим виходом до глобальних мереж, таких як Інтернет або GSM. У науковій літературі сенсорні мережі (рис. 1.1) описуються як «розподілені мережі, що складаються з бездротових вузлів, кожен з яких спеціалізується на певних задачах і розміщений у великій кількості випадковим чином на певній площі або території» [2].

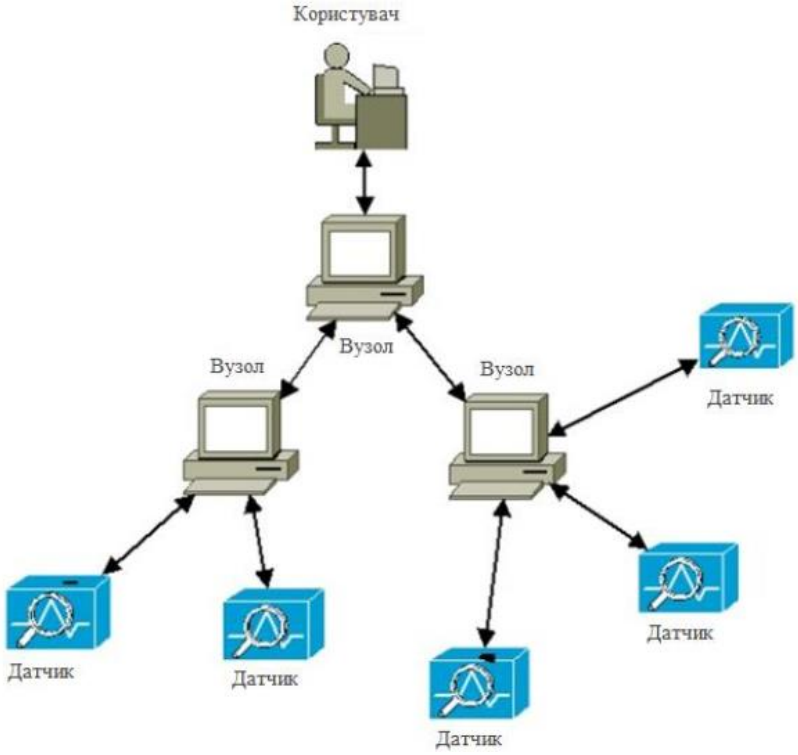


Рисунок 1.1 – Приклад простої сенсорної мережі

Таким чином, сенсорні мережі побудовані на основі спільної роботи великої кількості мініатюрних вузлів – сенсорних пристроїв (СП), які розміщені у певній області з високою щільністю. У межах покриття радіосигналу кожного СП має знаходитися щонайменше один сусідній вузол. Чим більше сусідів у кожного СП, тим вищою є точність і надійність роботи сенсорної мережі. Технології радіодоступу, що використовуються в сенсорних пристроях і базуються на стандарті IEEE 802.15.4, дозволяють передавати дані на відстань до кількох десятків метрів.

Сенсорний пристрій складається з чотирьох основних компонентів (рис. 1.2) [3]:

- блоки збору;
- блоки обробки;
- блоки передачі даних;
- блоки живлення.

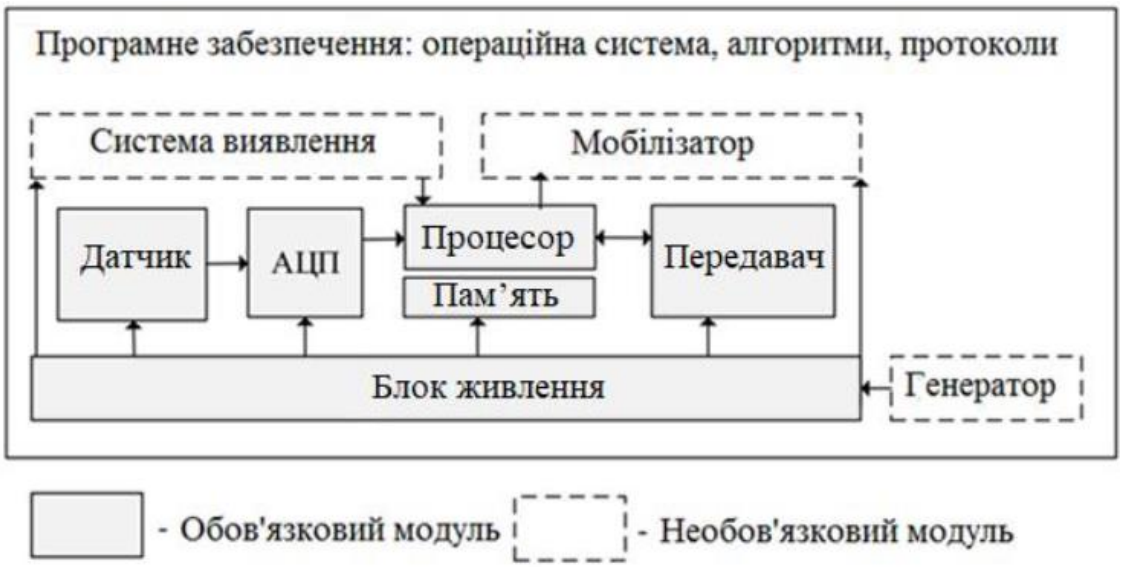


Рисунок 1.2 – Компоненти сенсорного пристрою

Також у складі сенсорного пристрою можуть бути модулі визначення місцезнаходження, силовий генератор і мобілізатор. Їх використання залежить від призначення, під яке розробляється сенсорна мережа.

Модуль збору даних складається з датчика і аналого-цифрового перетворювача (АЦП). У процесі спостереження датчик генерує аналоговий сигнал, який перетворюється на цифровий за допомогою АЦП. Далі цифровий сигнал передається до блоку обробки.

Модуль обробки, який включає процесор і пам'ять, виконує операції, необхідні для спільної роботи з іншими вузлами мережі з метою реалізації завдань спостереження. Завдяки наявності модуля обробки в складі сенсорного пристрою можливе перетворення вихідних даних за допомогою простих обчислень, що дозволяє отримувати тільки необхідні та частково

оброблені дані.

Передавач (трансивер) забезпечує інтерфейс сенсорного пристрою з мережею. Більшість сенсорних пристроїв потребують точної визначеності свого місцезнаходження, тому в схему інтегрується модуль визначення місця. Якщо сенсорний пристрій потребує переміщення, в нього може бути вбудований мобілізатор. Зазвичай всі ці модулі розміщуються в одному корпусі, розмір якого може варіюватися від кількох десятків кубічних сантиметрів до одного кубічного сантиметра, і при цьому він має бути досить легким, щоб залишатися в повітрі.

Окрім розмірів, до сенсорних пристроїв висуваються й інші вимоги, зокрема:

- низьке енергоспоживання;
- здатність взаємодіяти з багатьма подібними СП на малих відстанях;
- низька собівартість;
- автономність, тобто можливість роботи без нагляду;
- адаптивність до навколишнього середовища.

Блок живлення є одним із ключових компонентів сенсорного пристрою. Оскільки сенсорні пристрої можуть час від часу переходити в «сплячий» режим, тривалість роботи сенсорної мережі безпосередньо залежить від джерела живлення окремих СП, оскільки енергія виступає обмеженим ресурсом. Наприклад, загальний запас енергії смарт-вузла може становити приблизно 1 Дж. Для бездротових інтегрованих мереж датчиків (WINS) середнє споживання енергії повинно залишатися меншим за 30 мкА, щоб забезпечити тривалу автономну роботу для забезпечення тривалої роботи. Тривалість роботи сенсорних мереж можна збільшити, використовуючи акумулятори з можливістю підзарядки, наприклад, енергію можна поповнювати з навколишнього середовища за допомогою сонячних батарей.

Передавач СП може бути пасивним або активним оптичним компонентом, а також радіочастотним передавачем. У випадку радіочастотної передачі обов'язковими є процеси модуляції сигналу в певній смузі

пропускання, фільтрації та демодуляції, що ускладнює конструкцію СП і підвищує їх вартість. Також можуть бути втрати під час передачі даних між вузлами через близьке розташування антен до землі. Однак радіозв’язок є кращим, оскільки передача даних відбувається на низьких частотах (зазвичай менше 1 Гц), а цикли передачі є частими завдяки малим відстаням. Це дозволяє використовувати низькі частоти. Розробка радіопередавачів із низьким рівнем енергоспоживання є технічно складним і ресурсозатратним завданням. Хоча технології Bluetooth можуть застосовуватися у проєктуванні сенсорних пристроїв, вони не є достатньо ефективними для сенсорних мереж через високий рівень енергоспоживання.

Окрім традиційної архітектури сенсорних мереж, існують альтернативні підходи, які враховують потребу не тільки у моніторингу або контролі параметрів, що вимірюються, але й у здійсненні активного впливу на об’єкт вимірювання. Елемент, здатний здійснювати такий вплив, називається актором (рис. 1.3). Відмінною особливістю актора є наявність активного компонента, який під управлінням контролера взаємодіє з навколишнім середовищем та приймає рішення про вплив. Наприклад, це може бути пристрій для автоматичного введення інсуліну пацієнтам, які страждають на діабет.

Ще однією особливістю сенсорних мереж є здатність окремих сенсорних пристроїв взаємодіяти та працювати спільно завдяки вбудованим модулям обробки даних. Сучасні процесори стають дедалі компактнішими та потужнішими, однак обробка й зберігання інформації в сенсорних пристроях залишаються їхнім слабким місцем. У багатьох завданнях, пов’язаних із моніторингом, важливо визначати точне місцезнаходження сенсорних вузлів. Для цього в кожен вузол може бути інтегрований модуль глобального позиціонування (GPS) з точністю до 5 метрів. В альтернативному варіанті лише частина сенсорних пристроїв оснащується GPS, а інші визначають своє положення, використовуючи допоміжні сигнали від цих вузлів.

Розміщення сенсорних пристроїв не обов’язково має бути попередньо

розрахованим. Ця особливість дозволяє випадкове розміщення пристроїв, наприклад, у важкодоступних місцевостях або для швидкого розгортання мережі під час надання допомоги протягом певного часу. Це означає, що протоколи сенсорної мережі та алгоритми роботи сенсорних пристроїв повинні підтримувати самоорганізацію, що реалізується через систему визначення місця розташування [4].

Рисунок 1.3 – Компоненти акторного вузла

Взаємодія сенсорних пристроїв у межах однієї сенсорної мережі утворює топологію типу mesh (з англ. mesh – сітка). У такій структурі сенсорні вузли з'єднуються через численні канали зв’язку, формуючи розгалужену та стійку до збоїв сітчасту топологію (рис. 1.4) [5].

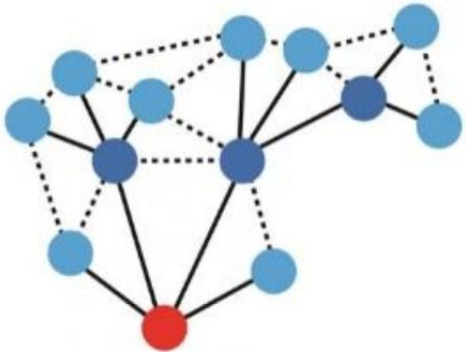


Рисунок 1.4 – Приклад топології mesh

Мережа з топологією mesh, яка здатна до самоорганізації, надає такі можливості:

- створення зон суцільного інформаційного покриття на великих площах, що дозволяє забезпечити доступ до даних у широкому діапазоні;
- масштабованість мережі, що означає можливість збільшення площі зони покриття та щільності інформаційних потоків у режимі самоорганізації, без необхідності значних змін в інфраструктурі;
- використання бездротових транспортних каналів для зв'язку в режимі «кожен з кожним», що дозволяє сенсорним пристроям обмінюватися інформацією безпосередньо один з одним;
- стійкість мережі до втрати окремих елементів, що підвищує надійність системи, оскільки з'єднання між пристроями залишаються активними, навіть якщо деякі з них виходять з ладу.

Ще однією важливою технологією, що реалізує концепцію IoT, є технологія M2M (Machine to Machine) (рис. 1.5) [7]. M2M – це загальна назва технологій, які забезпечують обмін інформацією між машинами або передачу даних в автоматичному режимі між пристроями без втручання людини. Ці технології можуть включати як дротові, так і бездротові системи моніторингу датчиків або різних параметрів пристроїв.

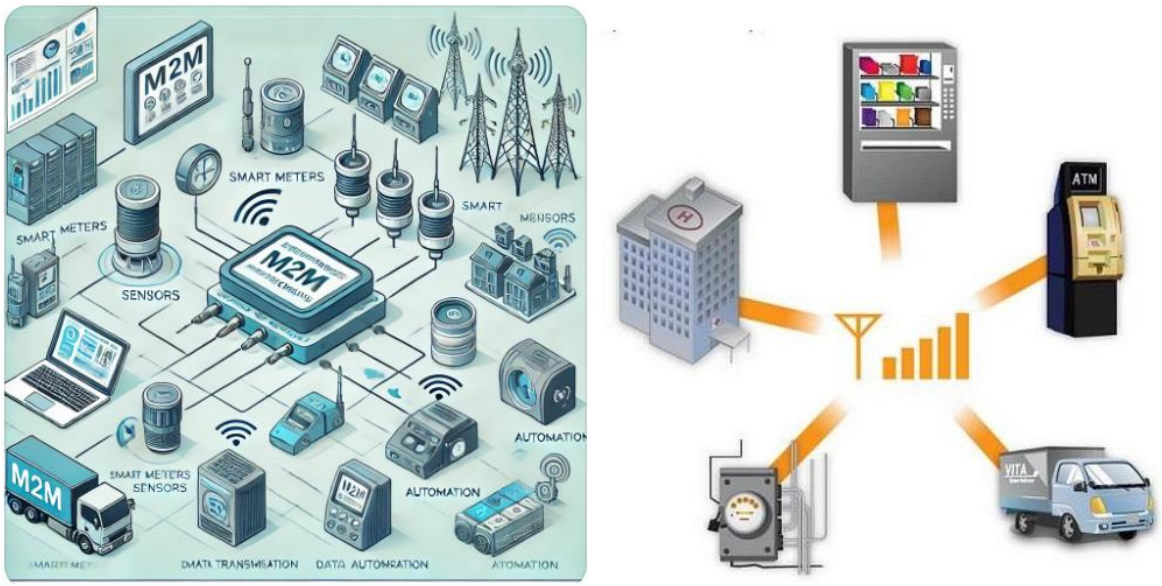


Рисунок 1.5 – Приклад технології Machine to Machine

Банкомати та платіжні термінали можуть автоматично передавати дані через GSM-мережі. Технологія M2M знаходить застосування в системах безпеки, охорони здоров'я, а також у системах відстеження рухомих об'єктів з використанням GPS. Завдяки вдосконаленню та широкому поширенню цієї технології, її можна використовувати у будь-якому мобільному пристрої, включаючи вузли сенсорних мереж.

Вважається, що ця технологія дала початок терміну «Інтернет речей», який позначає обчислювальне середовище, де пристрої самостійно взаємодіють між собою і надають користувачам результати своєї спільної роботи. Враховуючи появу цього терміну, сенсорні пристрої, інтегровані в об'єкти, також стали називати частиною IoT.

1.2 Архітектура мережі Інтернету речей

На сьогодні існує велика кількість архітектурних рішень для сенсорних мереж, але єдиного стандарту поки не вироблено. Це зумовлено тим, що перші спроби стандартизації в цій сфері були зроблені лише кілька років тому, і процес їх впровадження ще не завершений.

Відсутність єдиних загальноприйнятих стандартів спричинила появу численних розрізнених систем, таких як «розумний будинок», розроблених різними компаніями. Це, своєю чергою, створює низку проблем:

- виникнення надмірної кількості стандартів;
- надлишкове регламентування найпростіших об'єктів і процесів;
- велика кількість організацій і стандартів створює умови для лобіювання інтересів окремих компаній, що суперечить загальним завданням стандартизації.

Довгі терміни розробки стандартів призводять до їх морального старіння, так як вони не встигають за розвитком технологій, особливо на їх ранньому етапі.

Таким чином, варто розглянути дві основні архітектури: програмну і

фізичну.

Умовно програмна архітектура складається з семи рівнів.

Фізичний рівень − це датчики і електронні пристрої, які здатні підключатися до «речей» і отримувати дані від них.

Датчики збирають дані, але необхідно перетворити їх в зрозумілий формат і підключити цей пристрій до системи, використовуючи протокол обміну даними, який потрібно налаштувати.

Підключення до мережі − підключення пристрою до бездротової або провідної мережі. Ця можливість підключення змінюється на основі контексту і домену.

Акумуляція даних. Можна сказати, що даний шар відповідає за рівень безпеки та доступ до даних. Даний шар повинен бути досить «мобільний» для внесення необхідних змін.

Абстрагування даних. Зібрані дані використовуються для прийняття рішення або для цілей звітності. Це важливий шар, в який входить фактичний створюване рішення і бізнес-логіка.

Рівень додатків. На даному рівні відбувається контроль, аналіз і подання звітів системи. Грунтуючись на цих даних, можна відображати звіти або застосовувати машинне навчання, якусь спеціальну логіку або використовувати інтелектуальне рішення і посилати сигнал назад на датчики.

Останній рівень − це шар призначеного для користувача інтерфейсу.

Це короткий опис організації програмної частини Інтернету речей (рис. 1.6) [8].

Фізична архітектура IoT демонструє, як різні інфокомунікаційні технології, що забезпечують функціонування Інтернету речей, пов’язані між собою. Вона складається з чотирьох функціональних рівнів: рівня додатків IoT, рівня підтримки додатків і послуг, мережевого рівня і рівня пристроїв (рис. 1.7) [9].

Рівень пристроїв (сенсори і сенсорні мережі) є найнижчим рівнем архітектури IoT і складається з «розумних» об’єктів, інтегрованих зі

системами управління. Сенсори забезпечують з'єднання між фізичним і віртуальним (цифровим) світами, здійснюючи збір та обробку інформації в реальному часі.

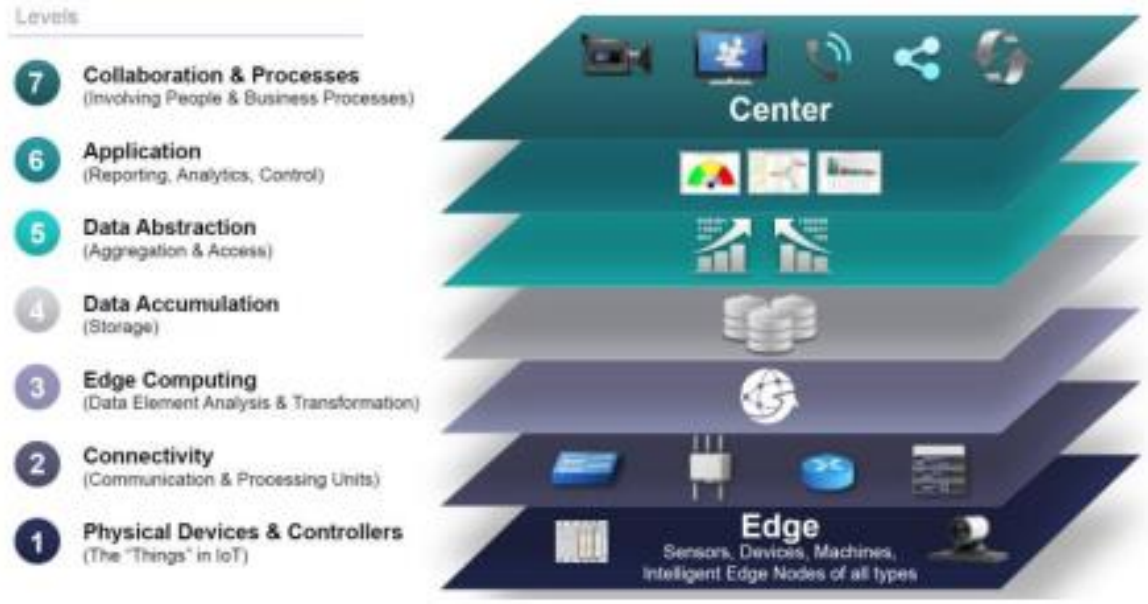


Рисунок 1.6 − Приклад програмної архітектури Інтернету речей

Більшість сенсорів потребують підключення до агрегатора сенсорів (шлюзу), який може бути реалізований за допомогою локальної обчислювальної мережі, такої як Ethernet, Wi-Fi або персональної мережі (PAN, Personal Area Network).

Основна мета рівня пристроїв полягає у виявленні різних явищ у навколишньому середовищі за допомогою периферійних пристроїв та зборі даних із реального світу. Цей рівень включає різноманітні датчики, використання яких є ключовою функцією IoT-пристроїв.

Датчики зазвичай інтегруються через концентратора, який слугує основною точкою з'єднання для кількох датчиків. Концентратор акумулює та передає дані з датчиків до блоку обробки даних у пристрої. Для передачі даних між датчиками та додатками концентратор використовує різноманітні транспортні механізми, такі як Inter-Integrated Circuit (I2C) чи Serial Peripheral Interface (SPI). Вибір конкретного механізму залежить від типу IoT-пристроїв

і створює канал зв’язку між датчиками та додатками збору даних.

Датчики в IoT-пристроях можна класифікувати на три широкі категорії. Першою категорією є датчики руху, які вимірюють зміни в русі та орієнтацію пристроїв. Вони поділяються на два типи: лінійний і кутовий рух. Лінійний рух відноситься до переміщення пристрою в певному напрямку, тоді як кутовий рух описує його обертальні зміни.

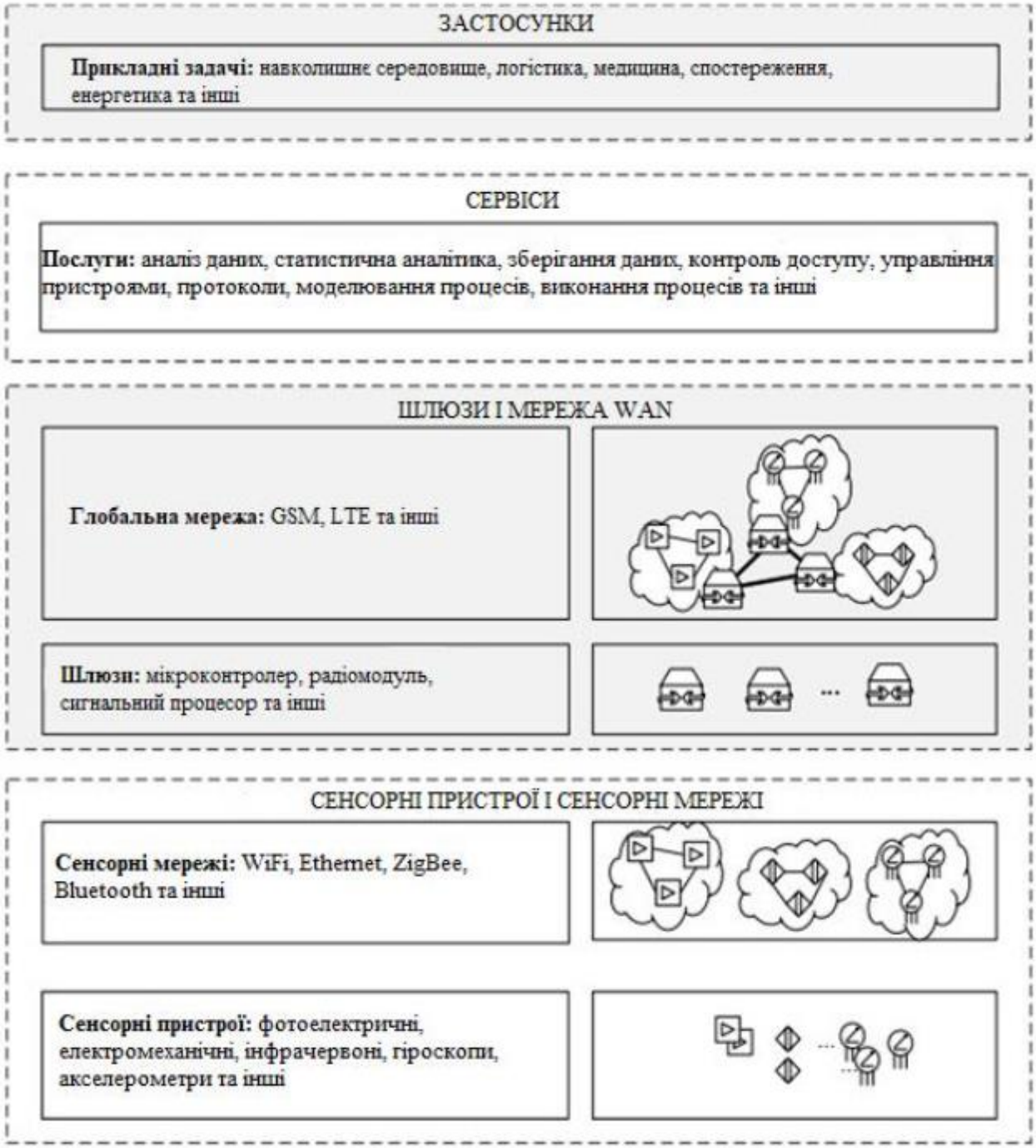


Рисунок 1.7 − Архітектура IoT

Другою категорією є датчики навколишнього середовища, до яких належать пристрої, що реагують на зміни параметрів, такі як світло та тиск. Вони допомагають IoT-пристроям приймати автономні рішення, наприклад, у розумних замках, системах домашньої автоматизації та освітлення.

Третю категорію складають датчики місцеперебування, які взаємодіють із фізичним розташуванням пристрою. Найпоширеними серед них є магнітні датчики, що працюють як цифрові компаси, та GPS-датчики, які застосовуються для навігаційних завдань у IoT-пристроях. Ці датчики допомагають виявити орієнтацію та місцезнаходження, що є критично важливим для багатьох додатків IoT.

Мережевий рівень забезпечує транспортування великих обсягів даних, що генеруються численними системами управління на першому рівні IoT. Він базується на конвергентній мережевій інфраструктурі, яка створюється шляхом об’єднання різнорідних мереж у єдину мережеву платформу.

В основі мережевого рівня лежить його функція як комунікаційного каналу, що дозволяє передавати дані, зібрані з датчиків, до інших підключених пристроїв. Реалізація цього рівня в IoT-пристроях відбувається через використання різноманітних комунікаційних технологій, таких як Z-Wave, LoRa, Wi-Fi, Bluetooth та інші. Ці технології забезпечують ефективну передачу даних між пристроями в межах самої мережі, що є важливим для забезпечення стабільності та надійності роботи IoT-систем. Таким чином, мережевий рівень відіграє критичну роль у з’єднанні та обміні інформацією між різними компонентами IoT-інфраструктури [10].

Сервісний рівень, або рівень обробки даних, охоплює набір інформаційних послуг, що забезпечують автоматизацію технологічних бізнес-операцій в IoT, зокрема:

- підтримка операційної та бізнес-діяльності;
- аналітична обробка інформації;
- зберігання даних;
- забезпечення інформаційної безпеки;

- управління бізнес-процесами та інші функції.

Даний рівень виконує ключову роль у взаємодії між зібраними даними та прийняттям рішень, він є головним блоком обробки даних IoT-пристрою і відповідає за аналіз даних, отриманих з рівня датчиків.

Після збору даних, рівень обробки виконує їх аналіз і на основі отриманих результатів приймає рішення. У багатьох IoT-пристроях, таких як смарт-годинники або розумні домашні концентратори, цей рівень також зберігає результати попереднього аналізу для подальшого використання. Крім того, рівень обробки даних має можливість передавати результати аналізу з одного пристрою на інший через мережевий рівень, що забезпечує інтеграцію та взаємодію між різними компонентами IoT-системи. Таким чином, рівень обробки даних не лише виконує функцію аналізу, але й підтримує ефективну комунікацію та обмін інформацією в рамках мережі.

Рівень додатків охоплює різноманітні типи програм, розроблені для конкретних промислових секторів і сфер діяльності, таких як енергетика, транспорт, торгівля, медицина, освіта тощо.

Додатки можуть бути «вертикальними», тобто розробленими для конкретної галузі промисловості, або «горизонтальними», які застосовуються в різних секторах економіки.

Архітектура системи Інтернету речей може варіюватися в залежності від поставлених завдань і використовуваних пристроїв. При її створенні важливо враховувати сучасний розвиток стандартів, а також можливість подальшої підтримки і розвитку системи.

Архітектура мереж Інтернету речей може бути побудована на основі різних підходів, таких як централізована, децентралізована та розподілена архітектури. Кожен з цих підходів має свої особливості, переваги і недоліки, які визначаються конкретними вимогами до мережі та її застосуванням.

Централізована архітектура IoT передбачає наявність єдиного центрального вузла (хабу, сервера або шлюзу), через який проходить весь трафік від пристроїв до центральної системи обробки [11]. Цей центральний

вузол виконує функції координації, управління і зберігання даних. Усі IoT-пристрої підключаються до цього вузла і передають свої дані до нього для подальшої обробки.

Переваги:

- спрощене управління мережею: всі пристрої контролюються через єдиний центр;
- легкість у реалізації: менша складність при проєктуванні, оскільки обробка даних централізована;
- надійне зберігання даних: усі дані збираються в одному місці, що спрощує їх обробку, аналіз і зберігання.

Недоліки:

- вразливість до збоїв: вихід з ладу центрального вузла може призвести до зупинки всієї мережі;
- проблеми з масштабуванням: із збільшенням кількості підключених пристроїв навантаження на центральний вузол значно зростає;
- затримки при передачі даних: пристрої можуть перебувати далеко від центрального вузла, що збільшує затримки при передачі.

Централізована архітектура зазвичай використовується в невеликих або локальних мережах з обмеженою кількістю пристроїв, де надійність центрального вузла легко забезпечується.

У децентралізованій архітектурі відсутній єдиний центральний вузол. Натомість, дані передаються між вузлами безпосередньо або через локальні шлюзи, і обробка інформації може виконуватися на декількох вузлах одночасно. Це дозволяє знизити навантаження на окремі частини мережі і зменшити затримки в передачі даних.

Переваги:

- підвищена стійкість до збоїв: мережа не залежить від одного центрального вузла. При виході з ладу окремих вузлів, інші можуть продовжувати роботу;
- краще масштабування: нові пристрої можуть додаватися до мережі

без значного збільшення навантаження на окремі елементи;

- зменшені затримки: оскільки обробка даних може відбуватися локально, затримки при передачі значно менші.

Недоліки:

- складність управління: у децентралізованій мережі складніше координувати роботу великої кількості пристроїв;

- ускладнення безпеки: більше точок взаємодії означає більше вразливостей для атак;

- вища складність обробки даних: децентралізовані системи потребують ефективних механізмів синхронізації та обміну даними між вузлами.

Цей підхід широко використовується в масштабованих і надійних системах, де кожен вузол може самостійно виконувати частину обчислень або обробки даних (наприклад, у великих міських сенсорних мережах).

Розподілена архітектура є найгнучкішим варіантом. У ній вся обробка даних і управління мережею розподіляються між численними вузлами IoT. Кожен вузол може як збирати, так і обробляти дані локально, а також взаємодіяти з іншими вузлами для обміну інформацією. Це забезпечує високу гнучкість, масштабованість і адаптивність мережі.

Переваги:

- висока надійність і відмовостійкість: розподілені системи майже не залежать від окремих вузлів, тому мережа продовжує працювати навіть при виході з ладу деяких елементів;

- відмінне масштабування: мережа легко масштабується за рахунок додавання нових вузлів, оскільки обробка даних відбувається паралельно;

- підвищена ефективність: обробка даних може виконуватися ближче до джерела інформації, що знижує затримки та навантаження на центральні системи.

Недоліки:

- висока складність управління: розподілені системи вимагають

складних механізмів для синхронізації та координації між вузлами;

- складна безпека: необхідно забезпечити безпеку всіх вузлів та їхньої взаємодії, що робить такі системи вразливими до атак;

- потребує більше ресурсів: розподілені системи можуть вимагати більше обчислювальних ресурсів на рівні кожного вузла.

Розподілені мережі особливо ефективні у великих, географічно розподілених системах IoT, таких як мережі смарт-міст, промисловий IoT або розумні будинки, де необхідна автономність і швидка обробка даних на місці.

Вибір архітектури IoT-мережі залежить від конкретних вимог до продуктивності, масштабованості та надійності системи. Централізовані архітектури простіші у впровадженні, але обмежені в масштабах і надійності. Децентралізовані мережі забезпечують кращу масштабованість та надійність, але складніші в управлінні. Розподілені системи надають максимальну гнучкість і ефективність, але потребують значних ресурсів для належної роботи.

1.3 Огляд способів взаємодії в мережі Інтернету речей

Одним із ключових аспектів організації Інтернету речей є розробка методів взаємодії між системами управління (інтернет-протоколами). У практиці використовують три основні підходи до взаємодії: прямий доступ, доступ через шлюз та доступ через сервер.

При прямому доступі до пристроїв IoT комунікація здійснюється безпосередньо з клієнтського застосування через IP-адресу, яка ідентифікує конкретну інтернет-річ. Інтерфейс взаємодії представлений у вигляді вебресурсу з графічним оформленням, доступним для користувачів через веббраузер. Також може використовуватися спеціалізоване програмне забезпечення.

Цей спосіб має свої переваги, зокрема простоту реалізації, оскільки користувачі можуть легко підключитися до IoT-пристрою за відомою IP-

адресою. Вебінтерфейси часто забезпечують інтуїтивно зрозумілу навігацію та зручність використання.

Проте існують і суттєві недоліки цього підходу:

- необхідність постійного підключення до мережі з фіксованою IP-адресою, що залежить від інтернет-провайдера. При зміні адреси або проблемах зі зв'язком IoT-пристрій може потребувати звернення до проксі-сервера для ідентифікації, що збільшує час реакції системи. Зрештою, цей підхід є вразливим до збоїв, оскільки у разі проблем з провайдером користувач може втратити доступ до пристроїв. В якості альтернативи можна використовувати мережевий псевдонім (аліас IP-адреси), однак це потребує регулярних запитів пристрою до спеціалізованого сервера для оновлення IP-адреси відповідно до псевдоніма;

- обмеження на кількість підключень до пристрою, що пов'язано з низькою якістю мережевого зв'язку або слабкими обчислювальними можливостями самих пристроїв. Це питання може бути вирішено через інтеграцію високопродуктивного обладнання та підключення пристроїв до стабільного інтернет-з'єднання. Однак це вимагає значно більшого енергоспоживання, що часто змушує робити такі пристрої стаціонарними, з підключенням до постійних джерел живлення.

З огляду на ці обмеження, прямий доступ часто вважається менш ефективним, особливо в умовах динамічних мереж, де потрібна гнучкість і швидкість взаємодії.

Однією з альтернатив прямому доступу є взаємодія через шлюз, що вважається більш ефективним підходом, особливо для організації зв'язку між «туманними» і «хмарними» обчисленнями.

Доступ до пристроїв Інтернету речей через шлюз є більш ефективним та раціональним способом організації взаємодії, особливо у випадках, коли необхідно інтегрувати бездротові сенсорні мережі або мережі IoT з глобальною мережею Інтернет. На відміну від прямого доступу, цей підхід стає незамінним у випадках, коли сенсорні мережі використовують власні

протоколи, не сумісні з IP-протоколами, що характерно для багатьох сучасних стандартів бездротових мереж.

Більшість протоколів бездротових сенсорних мереж (наприклад, Zigbee, Z-Wave, LoRaWAN) не підтримують традиційні IP-протоколи через обмежені ресурси таких мереж, зокрема їх енергоспоживання, обчислювальну потужність та вимоги до пропускної здатності. Замість цього вони застосовують спеціалізовані протоколи для ефективної взаємодії між сенсорами. Для забезпечення можливості передачі даних з сенсорної мережі до глобальної мережі Інтернет необхідно використовувати шлюз − пристрій, який виконує функції ретрансляції та трансляції протоколів.

Шлюз працює як посередник між сенсорними пристроями та зовнішньою мережею, конвертуючи дані з локальних протоколів у стандартні IP-формати, доступні для Інтернет-застосунків. Наприклад, сенсори можуть використовувати низькоенергетичний протокол передачі даних всередині мережі, тоді як шлюз конвертує ці повідомлення у HTTP або MQTT-протоколи для передачі через Інтернет.

Основні переваги використання шлюзу включають:

- можливість інтеграції різнорідних пристроїв, що використовують різні протоколи, в єдину систему;

- оптимізацію енергоспоживання сенсорних пристроїв за рахунок зменшення навантаження на них у процесі комунікації з Інтернетом;

- забезпечення масштабованості мережі IoT, оскільки шлюзи дозволяють ефективно управляти великою кількістю підключених пристроїв, знижуючи кількість безпосередніх IP-з’єднань.

Однак, разом з цим підходом зберігаються деякі недоліки, характерні і для прямого доступу. Основними з них є:

- залежність від шлюзу як критичного елемента інфраструктури. Якщо шлюз виходить з ладу або його продуктивність недостатня, це може призвести до порушення роботи всієї мережі;

- проблеми з масштабуванням та забезпеченням надійності, оскільки

велика кількість пристроїв може перевантажити шлюз, що призведе до зниження швидкості обробки даних або навіть до збоїв;

- збільшення затримок передачі даних через необхідність додаткової обробки та трансляції протоколів.

Таким чином, хоча шлюз є необхідним компонентом для ефективної взаємодії між бездротовими сенсорними мережами та Інтернетом, його використання потребує ретельного проєктування мережевої інфраструктури з урахуванням вимог до продуктивності та надійності системи.

Існує кілька підходів до організації шлюзів для сенсорних мереж.

Перший спосіб передбачає використання комп'ютерів, які мають доступ до глобальної мережі Інтернет, де кожна підключена мережа з'єднується з таким комп'ютером. Головними недоліками цього підходу є його висока вартість і громіздкість, оскільки сенсорні мережі зазвичай складаються з компактних датчиків і повинні працювати автономно. Проте, при такій схемі мережа втрачає автономність, оскільки її робота залежить від наявності живлення і доступу до Інтернету через комп'ютер.

Другий підхід передбачає використання спеціального пристрою-шлюзу, який з'єднує сенсорну мережу з найближчою провідною мережею, зазвичай Ethernet, що забезпечує доступ до Інтернету. Такий шлюз оснащений приймачем, сумісним із сенсорною мережею, портом для підключення до Ethernet, а також мікроконтролером, який конвертує пакети даних між мережами. Цей метод є менш затратним і більш компактним порівняно з першим, однак він потребує більшого енергоспоживання через використання стандартних провідних мереж, які не оптимізовані для низького енергоспоживання. Крім того, шлюз не гарантує наявність точки доступу до Інтернету в найближчій провідній мережі.

Третій спосіб передбачає використання повністю автономного пристрою-шлюзу, який самостійно надає доступ до Інтернету через бездротові технології передачі даних. Такий шлюз оснащений одним приймачем для сенсорної мережі і другим − для глобальної бездротової мережі, як-от GSM

або WiMAX, яка покриває зону дії сенсорної мережі. Використання GSM є більш економічним у плані енергоспоживання. Також є шлюзи, які забезпечують підключення до найближчих Wi-Fi мереж для пошуку точки доступу до Інтернету.

Отже, якщо потрібно організувати повністю автономну територіально-розподілену сенсорну мережу, найкращим рішенням є третій спосіб. Якщо ж сенсорна мережа є частиною більшої провідної мережі, автономність не є обов'язковою, і можна використовувати перший або другий спосіб.

Найбільш поширеним способом організації шлюзів в Інтернеті речей є використання пристроїв, що мають апаратно-програмні засоби для роботи в мережах ZigBee та GSM, а також можливість підключення до Інтернету через GPRS/EDGE канали. Такий підхід особливо актуальний через те, що локальні мережі Інтернету речей часто базуються на стеку протоколів ZigBee. У зв'язку з цим найбільш вживаним типом шлюзу є ZigBee-GSM шлюз.

ZigBee-GSM шлюз складається з двох основних компонентів: вузла мережі ZigBee та вузла мережі GSM, з'єднаних через послідовний інтерфейс UART. Вузлами можуть бути вбудовані модулі, розроблені спеціально для таких мереж. Для мереж ZigBee, такі модулі розробляють компанії, як Jennic або Digi, а для мереж GSM – Siemens або Sierra Wireless. Ці вбудовані модулі випускаються для конструкторських розробок, автоматизованих систем та апаратно-програмних комплексів, що використовуються в IoT для забезпечення надійної взаємодії між пристроями та мережею Інтернет.

Спрощена схема ZigBee-GSM шлюзу (рис. 1.8) вкдючає основні компоненти: вузол мережі ZigBee, вузол мережі GSM, їх з'єднання через інтерфейс UART, а також підключення до Інтернету через GSM/EDGE. Мережа ZigBee взаємодіє з IoT-пристроями, а шлюз забезпечує передачу даних до Інтернету [12].

В даний час шлюзи в сенсорних мережах широко застосовуються для інтеграції та взаємодії різних технологій. Протягом останніх років багато компаній розробили різні варіанти шлюзів для сенсорних мереж, що

поєднують сучасні бездротові технології, такі як Wi-Fi, LTE, WiMAX, GSM/GPRS, Bluetooth та GPS.

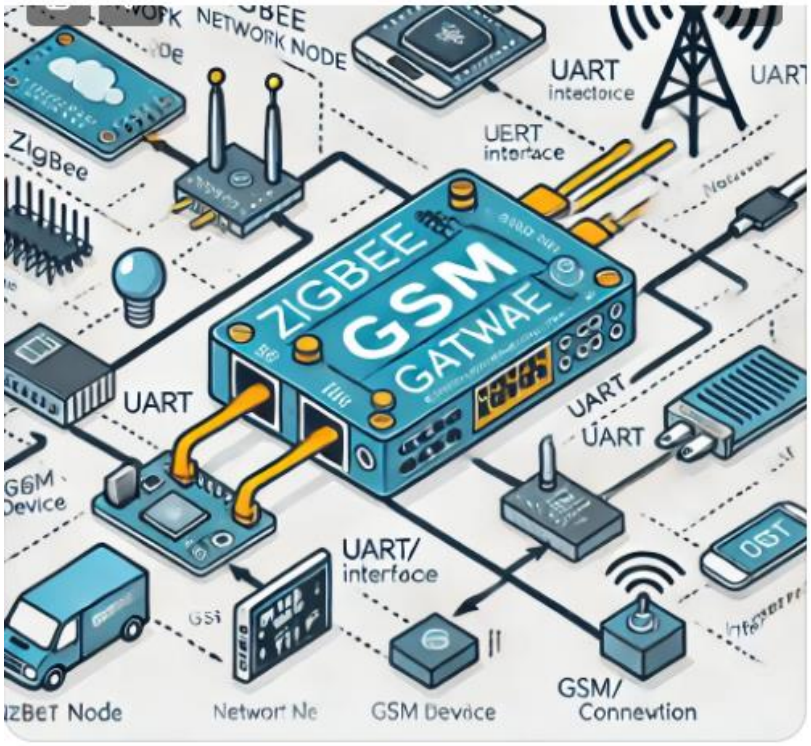


Рисунок 1.8 – Функціональна схема ZigBee-GSM шлюзу

Сучасні тенденції в розробці шлюзів зосереджуються на інтеграції новітніх бездротових технологій в одному пристрої. Однак, враховуючи вимоги Інтернету речей, головний акцент залишається на зниженні вартості та енергоспоживання. Тому шлюзи повинні бути максимально простими й доступними за ціною, без зайвої складності.

Що стосується енергоспоживання пристроїв, які приймають і передають дані, процес передачі інформації на сервер включає кілька ключових етапів:

- накопичення даних;
- активація GSM/GPRS модуля або вихід з режиму сну;
- встановлення зв’язку з сервером;
- передача даних;
- перехід у сплячий режим або вимкнення.

Дослідження показали, що GSM/GPRS пристрої можуть витрачати від

кількох до десятків секунд на встановлення з’єднання з мережею. Зокрема, якщо використовується стек TCP/IP, потрібен додатковий час на отримання IP-адреси і встановлення з’єднання з сервером. Тому для досягнення ефективності сплячий режим повинен тривати значно довше, ніж час, необхідний для встановлення зв’язку з сервером.

Доступ до інтернет-речей через сервер передбачає використання посередника між пристроями IoT і кінцевими користувачами. Цей посередник – сервер, який виконує такі основні функції:

- приймає повідомлення від інтернет-речей і передає їх користувачам;
- зберігає отриману інформацію та обробляє її;
- забезпечує інтерфейс для користувача, який дозволяє двосторонній обмін між користувачем і пристроєм IoT.

Цей метод є дуже ефективним і широко використовуваним, оскільки він дозволяє зменшити навантаження на самі інтернет-речі, переміщуючи обробку запитів користувачів на центральний сервер. Це, в свою чергу, допомагає зменшити навантаження на радіоканали зв’язку IoT-пристроїв і оптимізує використання провідних каналів для зв’язку між сервером і користувачами.

Централізований сервер також забезпечує надійне зберігання і обробку даних, дозволяючи інтернет-речам взаємодіяти між собою і використовувати можливості хмарних обчислень. Для з’єднання локальних бездротових мереж з сервером можуть застосовуватися шлюзи, які служать важливими точками доступу для організації такої комунікації.

Централізований сервер управління інтернет-речами складається з таких основних компонентів:

- модуль обробки інформації, що відповідає за різні операції з даними, такі як розпакування стиснених файлів, виконання арифметичних і логічних обчислень, класифікацію і перетворення даних у зручний для сприйняття формат;
- база даних для зберігання отриманої інформації, що дозволяє

зберігати статистику для моніторингу об’єктів та надавати актуальні дані користувачам, незалежно від часу їх підключення;

- інтерфейс взаємодії з інтернет-речами, який базується на певному протоколі, що підтримується всіма підключеними пристроями;

- система контролю доступу до інтернет-речей, яка також дозволяє управляти їх ієрархією, параметрами та функціональними можливостями.

Останній елемент може бути виведений на окремий сервер і відокремлений від основної бази даних, що зберігає інформацію, що збирається.

Процес управління інтернет-речами також здійснюється через сервер: користувач надсилає команду на сервер, який, у свою чергу, передає її потрібному пристрою. Важливо, щоб набір команд управління був мінімальним і загальним для всіх інтернет-речей, що забезпечує повне відділення управлінських функцій від обробки даних. Це значно полегшує обмін інформацією між користувачами та пристроями, а також сприяє стандартизації методів взаємодії в рамках Інтернету речей.

Відокремлення керуючої інформації від даних передбачає мінімізацію набору команд та сутностей, з якими ці команди можуть виконуватися. Наприклад, одна і та сама команда може використовуватись для управління різними пристроями, такими як вимикання лампочки, включення тостера чи зміна режиму відеокамери. Даними в такому випадку будуть поточні стани пристроїв, які можна отримати за допомогою команди для читання їх стану.

При використанні централізованого сервера управління, топологію Інтернету речей можна розглядати з двох перспектив: реальна − фізична топологія, що відображає фактичні зв’язки між елементами, та логічна топологія, яка враховує віртуальні з’єднання між ними.

Фізична топологія Інтернету речей − зірка (Star topology), де всі пристрої або групи пристроїв підключені окремо до центрального сервера управління. У такій топології сервер є основним вузлом, через який проходить комунікація між усіма речами, що підключені до мережі (рис. 1.9) [9].

Логічна топологія Інтернету речей може бути значно більш варіативною і складною. Вона описує те, як пристрої можуть взаємодіяти між собою через непрямі зв’язки, зокрема через сервер управління. Тобто, два пристрої можуть бути з’єднані не безпосередньо (як в фізичній топології), а через сервер, що забезпечує комунікацію між ними.

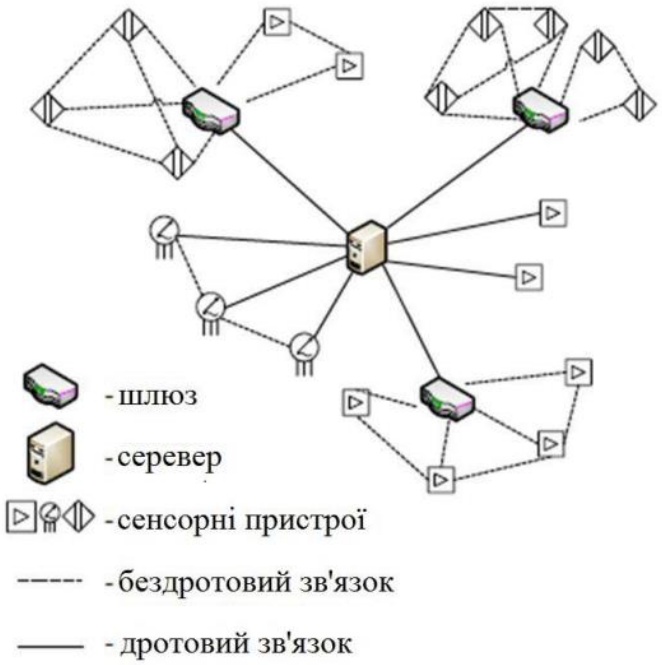


Рисунок 1.9 – Фізична топологія Інтернету речей з використанням централізованого сервера

Логічна топологія може включати використання локальних бездротових каналів між пристроями для прямого зв’язку та віддалених зв’язків через сервер управління, що забезпечує непрямий зв’язок.

У цьому випадку сервіс, наданий одним пристроєм іншому, може бути або прозорим (прихованим від користувача, без врахування фізичної відстані між пристроями), або з розподілом на два різні протоколи для локальної мережі та віддаленого зв’язку.

Таким чином, фізична топологія фокусується на реальному підключенні пристроїв, а логічна топологія більш орієнтована на те, як ці пристрої можуть взаємодіяти один з одним через різні канали і протоколи.

1.4 Види взаємодії в Інтернеті речей

Інтернет речей є обчислювальною мережею, що включає три основні компоненти: інтернет-речі, сервери керування інтернет-речами та призначені для користувачів вузли, такі як мобільні та персональні обчислювальні пристрої. Зважаючи на це, можна виділити три основні види взаємодії в Інтернеті речей:

- взаємодія між інтернет-речами;
- взаємодія між користувачами та інтернет-речами;
- взаємодія між віддаленим сервером і інтернет-речами.

Більшість інтернет-речей використовують радіочастотний канал для передачі даних, що часто має обмежену пропускну здатність, незважаючи на великий обсяг переданих даних у деяких випадках. Крім того, інтернет-речі мають обмеження щодо обчислювальних ресурсів, що створює певні проблеми для протоколів обміну даними між пристроєм та концентратором (Концентратор даних − це сервер, шлюз або інший пристрій, який приймає, обробляє, маршрутизує або зберігає дані, отримані від пристроїв Інтернету речей.). Це призводить до кількох ключових вимог до протоколів обміну:

- мінімізація трафіку;
- максимальне зменшення помилок передачі даних і трансляції команд;
- підтримка функції делегування завдань іншим обчислювальним одиницям.

На сьогодні основним протоколом мережевого рівня для Інтернету речей є IP, який може використовуватися разом із модифікованими протоколами транспортного рівня. Однак обговорюється можливість заміни протоколу IP для оптимізації алгоритмів маршрутизації та забезпечення мобільності пристроїв Інтернету речей. Як приклади альтернатив, розглядаються протоколи Locator/ID Separation Protocol (LISP) та Six/One [10].

Протокол LISP, розроблений компанією Cisco, спрямований на розділення функціональності, що асоціюється з IP-адресами. Він поділяє

ідентифікатори хостів та локатори маршрутизації, що дозволяє створювати тунельні маршрутизатори та додавати LISP-заголовки в пакети, коли вони рухаються мережею.

Протокол Six/One, розроблений компанією Ericsson, є доповненням до протоколу IPv6. Він забезпечує хостам постійні IP-адреси, які змінюються лише в старших бітах в залежності від того, до якого провайдера підключений хост на той момент. Старші біти автоматично оновлюються, коли пакети проходять через нового провайдера.

На прикладному рівні існує низка протоколів, створених спеціально для потреб Інтернету речей. Один з найбільш відомих і популярних протоколів − MQTT, розроблений компанією IBM, який активно використовується в середовищах IoT для забезпечення надійної та ефективної передачі даних.

Концепція Інтернету речей передбачає не лише збір даних і керування віддаленими об'єктами, а й обмін інформацією між ними, розподіл завдань і планування з урахуванням доступних сервісів у зоні охоплення. Об'єкти Інтернету речей мають здатність створювати локальні бездротові мережі для спільного вирішення завдань. Це можливо завдяки властивості самоорганізації бездротових сенсорних мереж.

снують складні завдання, що потребують значних обчислювальних ресурсів, наприклад, обробка відео в реальному часі. У разі помірного інформаційного потоку локальна мережа «розумних камер» може обробляти відео самостійно. Проте зі збільшенням навантаження локальна мережа здатна звертатися до обчислювальних потужностей хмари. Хмарні технології для Інтернету речей поділяються на два основних типи:

- хмарні обчислення (Cloud Computing);
- туманні обчислення (Fog Computing).

Туманні обчислювальні ресурси представляють собою локальні сенсорні мережі, що є частиною Інтернету речей. Вузли цих мереж можуть взаємодіяти між собою для спільного вирішення завдань без звернення до віддалених хмарних серверів. Туман в даному контексті означає наближення

обчислювальних ресурсів до «землі» або кінцевих пристроїв, замість їх централізації в хмарних дата-центрах. Таким чином, Fog Computing доповнює Cloud Computing, а не замінює його. У деяких випадках обидва підходи можуть працювати разом, наприклад, коли хмара надає послугу для туманної обчислювальної мережі [11].

Fog Computing можна охарактеризувати як віртуалізовану платформу, яка забезпечує три основні типи послуг для взаємодії між пристроями M2M (машина до машини): обчислення, зберігання даних і мережеву інфраструктуру. Основна мета туманних обчислень − організація ефективної взаємодії між мільярдами пристроїв та хмарними дата-центрами, забезпечуючи швидший обмін даними та оптимізацію ресурсів.

Туман можна уявити як трирівневу модель:

- верхній рівень − тисячі хмарних дата-центрів, які забезпечують ресурси для виконання складних завдань, таких як аналітика;

- середній рівень − десятки тисяч децентралізованих керівних дата-центрів, що містять «інтелект» туманних обчислень;

- нижній рівень − мільйони індивідуальних пристроїв, що виконують локальні обчислення.

Парадигма Fog Computing має кілька суттєвих відмінностей від Cloud Computing:

- розподілене обчислення в реальному часі: обчислювальні ресурси можна розміщувати на краю мережі, що сприяє зниженню затримок у процесі обміну даними і здійснювати обробку даних в реальному часі;

- географічний розподіл компонентів: модель Fog Computing характеризується меншою централізованістю ніж у Cloud Computing. Пристрої можуть обмінюватися даними і ділитися ресурсами напряму один з одним;

- значний обсяг зовнішніх даних: пристрої з великою кількістю сенсорів здатні генерувати величезні обсяги даних в реальному часі;

- складна топологія: мільйони географічно розподілених вузлів здатні формувати різноманітні й часто непередбачувані зв'язки;

- мобільність і різноманітність: переміщення пристроїв потребує застосування альтернативних протоколів для забезпечення ефективного обміну даними між пристроями в різних умовах.

1.5 Протоколи взаємодії в Інтернеті речей

1.5.1 Класифікація протоколів передачі даних в мережах IoT

Класифікація протоколів і технологій для IoT зазвичай ґрунтується на різних рівнях взаємодії в IoT-інфраструктурі, включаючи передачу даних, взаємодію з користувачами, енергоспоживання, масштабованість і типи мереж. Нижче наведено основні категорії та приклади протоколів і технологій для IoT.

1. Протоколи для передачі даних від сенсорів до серверів (збір даних). Ця група включає протоколи, що дозволяють сенсорам і пристроям передавати зібрані дані на центральні сервери або хмарні платформи для подальшого аналізу [12].

MQTT (Message Queuing Telemetry Transport) − протокол легкий, оптимізований для мереж з низькою пропускною здатністю та нестабільними з’єднаннями. Використовується в телеметрії та зборі даних.

CoAP (Constrained Application Protocol) − призначений для пристроїв з обмеженими ресурсами та працює за схемою «клієнт-сервер». Використовується для низької пропускної здатності та зменшеного споживання енергії.

HTTP/HTTPS (Hypertext Transfer Protocol) − протокол вебзапитів, що може бути використаний для передачі даних між IoT-пристроями і серверами через інтернет.

AMQP (Advanced Message Queuing Protocol) − протокол обміну повідомленнями для передачі даних від пристроїв на сервери з високою надійністю та гарантованою доставкою.

2. Протоколи для з’єднання сенсорних пристроїв з кінцевими

користувачами. Ці протоколи забезпечують зв’язок між сенсорами або пристроями і кінцевими користувачами через мобільні або стаціонарні пристрої.

Wi-Fi − використовується для високошвидкісної передачі даних на короткі дистанції, зазвичай для підключення пристроїв у межах будинку або офісу.

Bluetooth/Bluetooth Low Energy (BLE) − короткодистанційний бездротовий протокол для передачі даних, використовується для мобільних і носимих пристроїв з низьким енергоспоживанням.

ZigBee − стандарт для бездротових мереж з низьким енергоспоживанням. Зазвичай використовується для домашньої автоматизації та мереж з великою кількістю сенсорів.

5G/LTE (Long-Term Evolution) − стандарти мобільного зв’язку для високошвидкісної передачі даних. Підтримують масове підключення IoT-пристроїв і забезпечують мобільний зв’язок для кінцевих користувачів.

NFC (Near Field Communication) − технологія короткодистанційної бездротової передачі даних, що використовується для швидкої взаємодії між пристроями, наприклад, для платежів [13].

3. LPWAN (Low Power Wide Area Network)-технології використовуються для передачі даних на далекі відстані з низьким енергоспоживанням. Вони ідеальні для великомасштабних IoT-додатків, таких як розумні міста, сільське господарство або промислові рішення.

LoRaWAN (Long Range Wide Area Network) − забезпечує дальню бездротову передачу даних для сенсорів із низьким енергоспоживанням. Використовується для моніторингу інфраструктури, розумного міста, смарт-лічильників.

SigFox − енергоефективна технологія для передачі малих обсягів даних на великі відстані. Популярна у сфері IoT для збору даних від сенсорів.

NB-IoT (Narrowband IoT) − стандарт стільникового зв’язку, призначений для роботи з IoT-пристроями, які споживають мало енергії і мають низьку

швидкість передачі даних.

4. Протоколи для з’єднання серверів між собою. Ці протоколи використовуються для обміну даними між різними серверами, хмарними платформами та обчислювальними центрами в IoT-інфраструктурі.

HTTP/2 − покращена версія HTTP для більш ефективного обміну даними між серверами.

WebSocket − протокол для двостороннього зв’язку між клієнтами і серверами в реальному часі, використовується для обміну даними з мінімальними затримками.

DDS (Data Distribution Service) − протокол для розподілу даних у реальному часі між кількома пристроями або серверами.

5. Протоколи для обміну даними в локальних мережах. Ці протоколи використовуються для побудови локальних мереж між IoT-пристроями.

6LoWPAN (IPv6 over Low power Wireless Personal Area Networks) − протокол для передавання IPv6-пакетів через мережі з низьким енергоспоживанням, таких як ZigBee або Bluetooth.

Thread − протокол для створення надійних і безпечних бездротових мереж IoT-пристроїв з низьким енергоспоживанням, зазвичай використовується в розумних будинках.

6. Протоколи для хмарних IoT платформ. Протоколи та API, які використовуються для інтеграції IoT-пристроїв із хмарними платформами для зберігання, обробки та аналізу даних.

RESTful API (Representational State Transfer) − стандарт вебсервісів для інтеграції IoT-пристроїв з хмарними платформами для збору та аналізу даних.

GraphQL − API для запитів до даних на хмарних платформах IoT. Протокол дозволяє клієнтам запитувати тільки ті дані, які їм потрібні, що мінімізує трафік і оптимізує використання ресурсів. У порівнянні з RESTful API, де сервер повертає фіксовані набори даних, GraphQL забезпечує більшу гнучкість, дозволяючи комбінувати дані з різних джерел у одному запиті [13].

Класифікація протоколів і технологій для IoT може бути поділена на

різні рівні залежно від мети використання: для збору даних, передачі між пристроями, взаємодії з кінцевими користувачами, обміну даними між серверами або для підключення до хмарних платформ. Кожна з цих технологій має свою специфіку та оптимізована для певних умов використання.

1.5.2 Технології та протоколи передачі даних на довгі відстані в IoT-мережах

У найближчому майбутньому до IoT будуть підключені мільярди пристроїв. Більша частина цих пристроїв буде працювати від батарейок. У зв'язку з цим, однією з основних вимог до таких пристроїв є тривалість роботи без втручання людини.

Для того, щоб забезпечити тривалу роботу пристроїв IoT, були розроблені нові мережі, які називаються LPWAN (Long Power Wide Area Network). Ці мережі характеризуються низькою потужністю передачі даних, що дозволяє їм працювати від батарейок протягом тривалого часу. До основних технологій цих мереж належать NB-IoT, Weightless, LoRa, SigFox та інші. Перераховані технології забезпечують тривалий час автономної роботи пристроїв та низьку вартість. Вони використовуються в таких сферах, як розумні міста, сільське господарство та охорона здоров'я [13].

Технологія LoRaWAN викликала великий інтерес у галузі бездротового зв'язку. Це призвело до необхідності створення єдиного стандарту для глобальних мереж з низьким енергоспоживанням, таких як LPWAN [14]. Абревіатура LoRa означає «Long Range». Цей метод модуляції дозволяє передавати дані на великі відстані з низьким енергоспоживанням. LoRa розроблений і запатентований компанією Semtech і використовується в мережах LPWAN. Протокол LoRaWAN є відкритим, що означає, що його може використовувати будь-яка компанія. Це сприяє поширенню технології LoRaWAN і її використанню в різних сферах. Технологія LoRaWAN має великий потенціал для розвитку IoT. Вона дозволяє підключати велику

кількість пристроїв з низьким енергоспоживанням, що відкриває нові можливості для автоматизації та контролю.

Метод модуляції LoRa і заснований на технології розширення спектру. Це означає, що дані передаються у вигляді широкосмугових імпульсів з частотою, яка змінюється з часом. Таке рішення має кілька переваг. По-перше, воно робить передавач і приймач більш стійкими до перешкод. По-друге, це дозволяє застосовувати недорогі компоненти, наприклад, кварцові резонатори. LoRa працює в субгігагерцовому діапазоні частот, що також сприяє його енергоефективності [14].

Завдяки високій чутливості (-148 dBm) технологія LoRa чудово підходить для пристроїв, які потребують низького енергоспоживання та забезпечення стабільного зв’язку на великих відстанях.

У LoRaWAN працюють різні типи пристроїв, які поділяються на три класи.

1. Двонаправлені кінцеві пристрої класу A (Bi-directional End Devices, Class A). Ці пристрої використовуються там, де необхідно мінімізувати енергоспоживання, з акцентом на передачу даних від кінцевого вузла до сервера. Кінцевий вузол розпочинає сеанс зв’язку шляхом надсилання пакета даних, після чого відкриваються два приймальні вікна для отримання відповіді від сервера. Обмін даними між сервером і кінцевим пристроєм можливий лише після ініціації сеансу з боку вузла.

2. Двонаправлені кінцеві пристрої класу B (Bi-directional End Devices, Class B). Відрізняються від пристроїв класу A тим, що вони мають можливість періодично відкривати додаткові вікна прийому за розкладом. Синхронізація розкладу здійснюється через спеціальний сигнал від шлюзу. Це дозволяє серверу обмінюватися даними з кінцевим пристроєм у визначені моменти часу.

3. Двонаправлені кінцеві пристрої класу C (Bi-directional End Devices, Class C). Ці пристрої мають практично постійно відкриті приймальні вікна, які закриваються лише під час передачі даних. Завдяки такій властивості, вони

ідеально підходять для завдань, пов’язаних з передачею великих обсягів даних.

Архітектура LoRaWAN включає основні компоненти: кінцеві пристрої, шлюзи, мережевий сервер і сервер додатків (рис. 1.10).

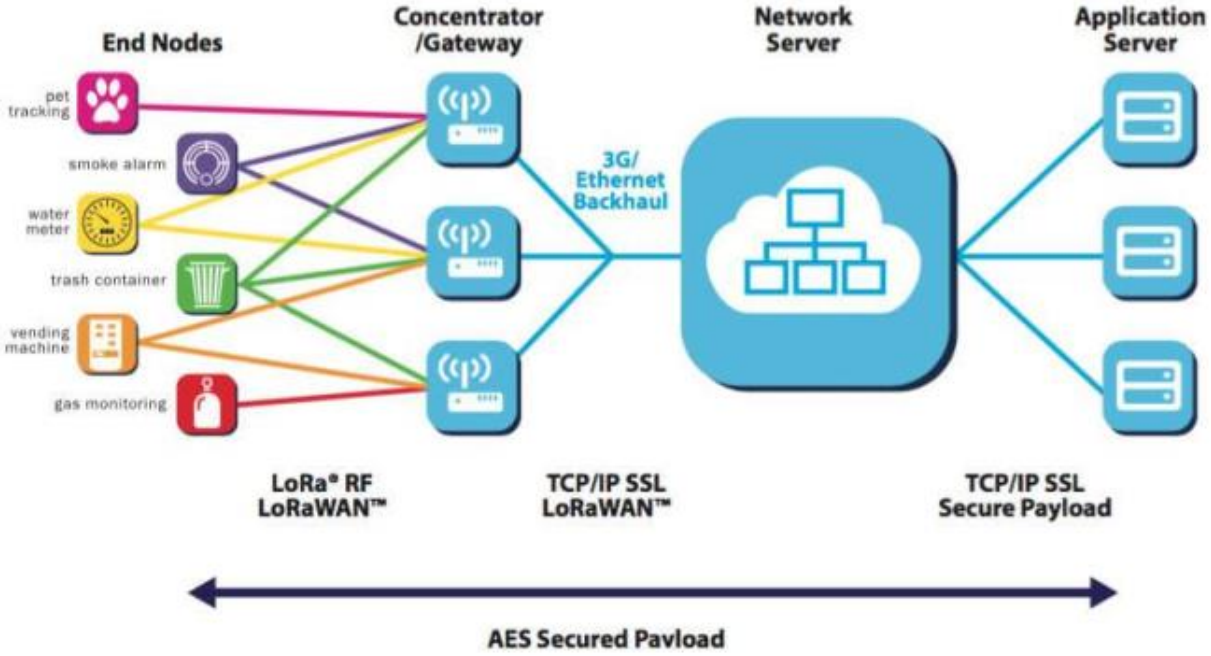


Рисунок 1.10 – Архітектура мережі LoRaWAN

Кінцеві вузли (End Nodes) – це пристрої, які виконують функції вимірювання, контролю та керування. Вузол складається з різноманітних датчиків для вимірювань та керуючих елементів. Зазвичай працює на батарейному живленні, тому для збереження енергії передача даних відбувається лише протягом короткого часу. Після цього відкриваються два тимчасових вікна для прийому даних [15].

Шлюз LoRa (Gateway/Concentrator) – пристрій, що отримує дані від кінцевих вузлів через радіоканал і передає їх у транзитну мережу, наприклад WiFi, Ethernet або стільникові мережі. Кінцеві вузли та шлюзи утворюють мережу з топологією типу «зірка». Багатоканальні приймачі та передавачі шлюзу можуть обробляти сигнали, які надходять одночасно з кількох каналів або від одного каналу, забезпечуючи повне покриття мережі та прозору

двосторонню передачу даних між мережею та кінцевими вузлами [16].

Мережевий сервер (Network Server) − це віддалений центр керування мережею, який відповідає за регулювання швидкості, аналіз, обробку та зберігання даних, отриманих зі шлюзів.

Сервер додатків (Application Server) − це пристрій, який збирає інформацію з кінцевих вузлів та здійснює віддалений контроль їх роботи.

Один LoRa-шлюз може обслуговувати до п’яти тисяч кінцевих пристроїв завдяки:

- особливостям мережевої топології;
- адаптивній швидкості передачі даних та вихідній потужності пристроїв, які встановлює мережевий сервер;
- тимчасовому поділу доступу до середовища;
- частотному поділу каналів;
- спеціальній LoRa-модуляції, яка дозволяє одночасно демодулювати сигнали на різних швидкостях у межах одного частотного каналу [14].

Енергоефективні бездротові технології LoRaWAN дозволяють створювати глобальні, але водночас прості мережі передачі даних із великою кількістю кінцевих вузлів. Їх основні переваги − розширений радіус дії, тривала автономна робота та надійне виявлення корисного сигналу, навіть за наявності перешкод [17].

SigFox − це технологія, яка забезпечує нову мережу та інформаційну стратегію для Інтернету речей. Її розробила компанія з Labège, Франція, яка також є оператором цієї мережі та займається впровадженням IoT в бізнес-індустрію. Архітектура мережі SigFox схожа на мережі стільникових операторів, таких як GSM і GPRS, але відрізняється нижчими витратами та підвищеною енергоефективністю (рис. 1.11) [18].

Зона покриття SigFox становить близько 30-50 км у сільській місцевості та 3-10 км у міських умовах через велику кількість радіошумів.

Технологія використовує надвузьку смугу частот (UNB − Ultra Narrow Band) для підключення пристроїв до глобальної мережі, що дозволяє знизити

рівень потужності передавача під час передачі даних. У Європі використовується частота 868,8 МГц, а у США − 915 МГц [19].

Пристрої передають дані на базові станції SigFox, де вони декодуються і передаються до хмарного сервера. Після цього хмарний сервер SigFox надсилає повідомлення на клієнтські сервери та IT-платформи через API.

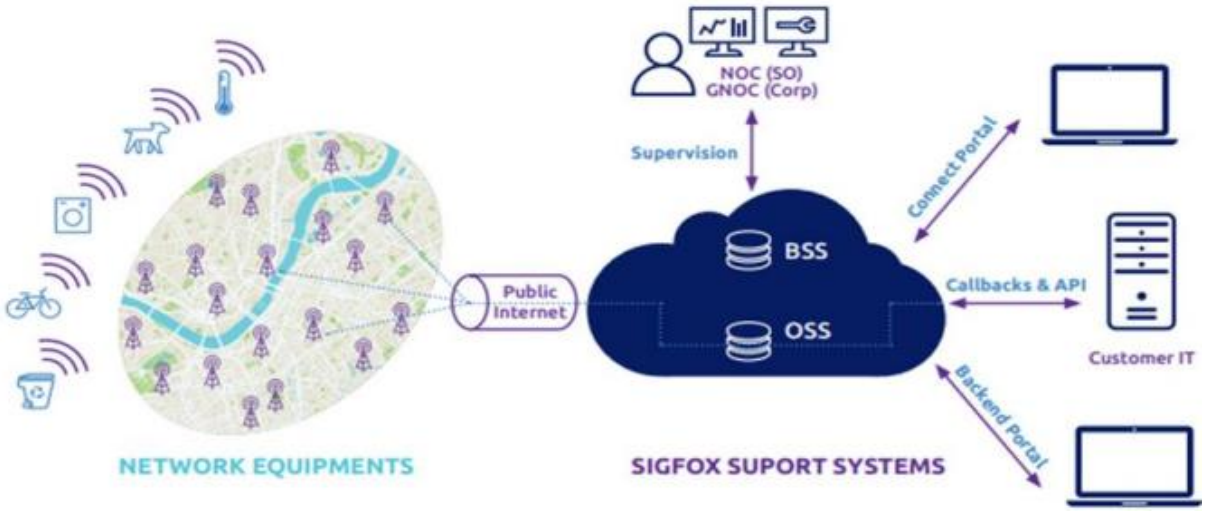


Рисунок 1.11 – Архітектура мережі SigFox

Технологія SigFox орієнтована на створення недорогих пристроїв з широким радіусом дії. Вона має застосування в різних сферах, де потрібен бездротовий зв’язок: домашні пристрої та споживчі товари, енергетика, охорона здоров’я, транспорт, віддалений моніторинг і управління, безпека.

SigFox має кілька переваг над іншими технологіями LPWAN мереж. До основних переваг належать:

- велике покриття;
- висока здатність проникнення сигналу;
- тривала робота від однієї батареї (сенсори можуть працювати до 20 років на двох батарейках AA);
- дуже низьке енергоспоживання;
- низька вартість.

Однак, як і всі сучасні технології, SigFox має свої недоліки:

- низька швидкість передачі даних;

- залежність від стільникової інфраструктури;
- обмежена стійкість до перешкод [20].

Цю технологію активно використовують у багатьох країнах Європи та США.

NB-IoT − це стандарт стільникового зв'язку, спеціально розроблений для пристроїв телеметрії, які передають невеликий обсяг даних. Він був створений організацією 3GPP як продовження розвитку мобільних стільникових мереж, і його перша версія з'явилася у 2016 році.

Протокол NB-IoT має низку важливих переваг:

- низьке енергоспоживання, яке забезпечує до 10 років роботи на одній батареї;
- велике покриття;
- можливість швидкого оновлення мережі;
- висока надійність.

NB-IoT представляє собою наступний крок у розвитку стільникового зв'язку, що дає операторам змогу впроваджувати рішення для IoT, такі як системи інтелектуального моніторингу та обліку.

Ця технологія розглядається як еволюція стільникового зв'язку до Інтернету речей, підтримуючи M2M додатки з низьким енергоспоживанням. NB-IoT може працювати в тих самих частотних діапазонах, що й 2G/3G/4G, зокрема в «низьких» частотах: 800 МГц (20-й діапазон), 900 МГц (8-й діапазон) та 1800 МГц (3-й діапазон). Використання вищих частот є менш доцільним через більшу втрату сигналу.

У технології NB-IoT є три основні способи виділення частотного ресурсу, які дозволяють операторам ефективно інтегрувати цю технологію у свої наявні мережі. Кожен з них має свої особливості щодо використання спектру.

1. В межах смуги (In-band deployment). Цей спосіб використовує вільні ресурси в межах існуючих LTE (Long-Term Evolution − це стандарт бездротового зв'язку четвертого покоління (4G), розроблений для

забезпечення високошвидкісної передачі даних в мобільних мережах. Його мета − покращити пропускну здатність, знизити затримки та забезпечити кращу якість обслуговування в порівнянні з попередніми поколіннями мобільного зв'язку (3G та 2G). LTE є еволюцією технологій GSM/EDGE та UMTS/HSPA) або 4G частотних діапазонів. NB-IoT займає підканали в межах смуги LTE, зазвичай один або кілька фізичних ресурсних блоків. Це рішення забезпечує високу ефективність використання спектра, оскільки не вимагає виділення окремого частотного діапазону для NB-IoT і дозволяє працювати в симбіозі з існуючою мережею LTE.

2. Гвардійська смуга (Guard-band deployment). В цьому випадку NB-IoT використовує ресурси, розташовані в гвардійських смугах LTE − простір між активними частотами LTE, який зазвичай не використовується для передачі даних. Завдяки цьому, NB-IoT може ефективно функціонувати без впливу на продуктивність LTE. Цей підхід дозволяє операторам використовувати неактивний частотний ресурс для впровадження IoT рішень без додаткових витрат на спектр.

3. Позасмуговий режим (Stand-alone deployment). У цьому сценарії NB-IoT працює в окремому частотному діапазоні, незалежно від LTE або 4G. Для цього часто використовують частоти, що раніше використовувалися для 2G (GSM) мереж. Це рішення забезпечує найкращу ізоляцію і мінімальний вплив на існуючі мобільні сервіси, особливо у випадках, коли існує потреба у наданні сервісів IoT у зонах з низькою пропускною здатністю або там, де LTE ресурси перевантажені.

Кожен з цих способів виділення частот має свої переваги і вибір залежить від доступних ресурсів оператора та конкретних вимог до покриття і пропускної здатності мережі.

Weightless-P − це технологія бездротового зв'язку з низьким енергоспоживанням, яка використовується для Інтернету речей. Вона розроблена для пристроїв, які потребують тривалої роботи від батарей, двостороннього зв'язку та підтримки великої кількості пристроїв [21]. В якості

особливостей цієї технології слід зазначити широку зону покриття, масштабованість мережі, довгий термін роботи батареї та безпеку.

Weightless-P підтримує кілька типів модуляції, але обмежений діапазон. Це дозволяє їй забезпечувати високу якість зв'язку, навіть у складних умовах. Одна базова станція Weightless-P може обслуговувати більше пристроїв, ніж базова станція іншої технології LPWAN. Крім того, базова станція Weightless- P має повний контроль над мережею та пристроями, що дозволяє їй забезпечувати більш високу безпеку та надійність.

Weightless-P − це більш розвинена технологія, ніж LoRa і SigFox. Вона підтримує гарантовану доставку повідомлень: повідомлення буде доставлено до кінцевого пристрою, навіть за умови слабкого сигналу та наявності перешкод. Це дозволяє уникнути повторного відправлення повідомлень, що економить заряд пристроїв. Крім того, Weightless-P використовує метод підтримки адаптивної швидкості передавання інформації. Це означає, що швидкість передавання даних змінюється в залежності від умов навколишнього середовища.

Weightless-P підтримує адаптивну зміну швидкості передачі даних. Це означає, що швидкість передачі даних для кожного пристрою регулюється в залежності від його відстані до базової станції. Чим ближче пристрій до базової станції, тим вища швидкість передачі даних. Більш оптимізований та невеликий по розмірам протокол Weightless-P дозволяє зменшити вартість системи та простоту експлуатації. Це пов'язано з тим, що для реалізації цієї технології потрібно менше апаратного забезпечення та програмного забезпечення [22-23].

Все це дозволяє підвищити продуктивність мережі та продовжити термін служби батареї пристроїв. Крім того, Weightless-P має більш оптимізований та невеликий по розмірам протокол, ніж NB-IoT та інші стільникові M2M системи. Це дозволяє зменшити вартість системи та простоту експлуатації.

Технологія Weightless-P знайшла застосування в різноманітних сферах,

включаючи системи безпеки та спостереження. В області охорони здоров’я ця технологія забезпечує безперервний моніторинг стану пацієнтів, дозволяючи своєчасно реагувати на зміни. У сфері розумних речей Weightless-P використовується для підключення побутових пристроїв до IoT, забезпечуючи їхню інтеграцію з іншими системами. У промисловому Інтернеті речей − сприяє автоматизації процесів і збору даних від датчиків у реальному часі.

Провівши аналіз відкритих джерел, можна зробити висновок про перспективність використання для даних цілей наступних технологій: NB-IoT, Weightless, LoRa, SigFox (табл. 1.1) [14].

Таблиця 1.1 − Порівняльна характеристика технологій передачі даних на довгі відстані в мережах IoT

<table>
<tr><td>Характеритстики</td><td>LoRaWAN</td><td>SigFox</td><td>NB-IoT</td><td>Weightless-P</td></tr>
<tr><td>Метод модуляції</td><td>CSS</td><td>DBPSK/GFSK</td><td>GFSK/BPSK/QPSK</td><td>GMSK/PSK</td></tr>
<tr><td>Діапазон</td><td>ISM</td><td>ISM</td><td>Ліцензований</td><td>ISM</td></tr>
<tr><td rowspan="2">Швидкість</td><td rowspan="2">0,3-50 кбіт/с</td><td rowspan="2">100 кбіт/с</td><td>UL: 1-144 кбіт/с</td><td rowspan="2">0,2-100 кбіт/с (адаптивна)</td></tr>
<tr><td>DL: 1-200 кбіт/с</td></tr>
<tr><td rowspan="2">Смуга</td><td>Штрокосмуг.</td><td>Вузькосмуг.</td><td>Вузькосмуг.</td><td>Вузькосмуг.</td></tr>
<tr><td>до 500 кГц</td><td>100 кГц</td><td>200 кГц</td><td>12,5 кГц</td></tr>
<tr><td>Максимальний час автономної роботи пристроїв</td><td>>10 років</td><td>До 20 років</td><td>До 10 років</td><td>3-5 років</td></tr>
<tr><td rowspan="3">Частота</td><td>868,8 МГц (Європа)</td><td>868,8 МГц (Європа)</td><td rowspan="3">800/900/1800 МГц</td><td rowspan="3">169/433/470/780/868/915 МГц</td></tr>
<tr><td>915 МГц (США)</td><td rowspan="2">915 МГц (США)</td></tr>
<tr><td>433 МГц (Азія)</td></tr>
<tr><td>Безпека</td><td>AES-64/128</td><td>AES з HMACs</td><td>AES-256</td><td>AES-128/256</td></tr>
<tr><td rowspan="2">Дальність</td><td>До 2,5 км у місті,</td><td>До 10 км у місті,</td><td rowspan="2">До 2 км</td><td rowspan="2">До 2 км у місті</td></tr>
<tr><td>До 45 км за містом</td><td>До 50 км за містом</td></tr>
</table>

1.6 Інформаційна безпека систем IoT

Кількість IoT-пристроїв − роутерів, камер, NAS-сховищ, компонентів «розумного будинку» зростає з кожним роком. Прогнозується, що до 2025 року загальна кількість встановлених розумних пристроїв досягне відмітки 30,9 мільярда [24-25]. Із збільшенням кількості підключених пристроїв зростає і необхідність їх захисту. Перші масові атаки на IoT-пристрої з використанням шкідливого програмного забезпечення були зафіксовані ще у 2008 році, а з тих пір подібних атак стає тільки більше.

Для запобігання атакам на пристрої Інтернету речей важливо з'ясувати різноманітні види атак, які можуть бути використані кіберзлочинцями.

1. Одним з таких видів атак є DDoS-атака. DDoS-атаки (атаки з відмовою в обслуговуванні) є серйозною загрозою для пристроїв IoT. У цих атаках ботнет, що складається з компрометованих пристроїв, надсилає велику кількість запитів до цільової системи або мережі, намагаючись перевантажити їхні ресурси. Через велику кількість запитів, які надходять від ботнету, мережа або цільова система може перевантажитися, що призводить до зниження її продуктивності або навіть до повного зупинення роботи. Вдало налаштована DDoS-атака може призвести до виникнення системних помилок або вразливостей безпеки, які можуть бути використані для отримання несанкціонованого доступу до системи. Кіберзлочинці можуть скомпрометувати пристрої IoT та використовувати їх для створення ботнету, який буде виконувати DDoS-атаки. Це особливо небезпечно, оскільки пристрої IoT можуть бути менш захищеними та менш виявленими для користувачів. Зловмисники можуть використовувати компрометовані пристрої IoT для внутрішніх атак у локальній мережі. Це може призвести до порушення безпеки всієї мережі та втрати конфіденційної інформації.

Усього за другу половину 2024 року аналітики виявили понад 700 оголошень про послуги з проведення DDoS-атак на різних форумах у даркнеті (рис.1.12) [26] .

Рисунок 1.12 − Розподіл кількості публікацій, пов’язаних з послугами з проведення DDoS-атак, по місяцям, 2024

2. Експлойти програмного забезпечення є серйозною загрозою для пристроїв IoT та можуть призвести до небезпечних наслідків. Кіберзлочинці можуть використовувати відомі вразливості у програмній частині пристроїв для здійснення атак. Це може включати в себе відкриті порти, недоліки в аутентифікації, неадекватні обмеження доступу, тощо. Не всі виробники пристроїв IoT завантажують оновлення програмного забезпечення вчасно. Це робить пристрої вразливими до атак, оскільки зловмисники можуть використовувати вразливості, які були виправлені у нових версіях програмного забезпечення. Не всі виробники надають достатню інформацію користувачам щодо програмного забезпечення, яке використовується в їхніх пристроях. Це може ускладнювати процес виявлення та усунення вразливостей, оскільки користувачі можуть бути несвідомі щодо потенційних загроз безпеці [27].

3. MITM-атака (Man-in-the-Middle attack) або атака посередника є однією з найбільш хитрих та широко використовуваних методів кібератак. Вона полягає в тому, що зловмисник вставляється між сторонами, які спілкуються, що зазвичай є пристроєм відправника та пристроєм отримувача, та перехоплює і маніпулює комунікаційним потоком. Зловмисники можуть

змінювати дані під час їх передавання між пристроями. Це дозволяє їм впливати на поведінку пристроїв та маніпулювати результатами комунікації. Багато смарт-пристроїв часто не зашифровані, що робить їх особливо вразливими до MITM-атак. З отриманими даними зловмисники можуть отримати несанкціонований доступ до систем, пристроїв або мереж, що може призвести до витоку чутливої інформації або виконання небажаних дій [28].

4. Фізичне втручання: просте підключення кіберзлочинцем USB-флешки із шкідливим кодом до зовнішнього пристрою IoT може стати достатньою умовою для поширення зловмисного програмного забезпечення через мережу та перехоплення комунікацій, які в ній здійснюються.

5. Брутфорс-атаки: той факт, що в компаніях часто недостатньо уваги приділяється безпеці паролів для IoT-пристроїв, робить їх уразливими до атак методом грубої сили («брутфорс»). Часто паролі залишаються незмінними після встановлення, що дає зловмисникам можливість легко їх зламати [29].

За першу половину 2023 року 97,91% спроб перебору паролів, зафіксованих ханіпотами, були пов’язані з протоколом Telnet та 2,09% − із SSH. Найбільше інфікованих пристроїв, які здійснювали ці атаки, перебували у Китаї, Індії та США (рис. 1.13), а за кількістю атак лідирують Китай та Пакистан [30-31].

Рисунок 1.13 − ТОР 10 країн та територій, в яких знаходилася більшість пристроїв, що атакували ханіпоти, 2023

6. Викрадення прошивки є серйозною загрозою для безпеки пристроїв IoT. Прошивка, яка контролює роботу пристрою, може містити критичні дані та налаштування, а також бути ключовим елементом захисту. Зловмисники можуть використовувати вкрадену прошивку для отримання контролю над пристроєм. Це може включати в себе виконання шкідливих команд або встановлення додаткового шкідливого програмного забезпечення. Якщо викрадена прошивка містить конфіденційні дані, такі як облікові записи або ключі шифрування, зловмисники можуть отримати доступ до цих даних і використовувати їх у своїх цілях, включаючи витік конфіденційної інформації. Зловмисники можуть модифікувати викрадену прошивку, щоб вона містила шкідливе програмне забезпечення. Після цього вони можуть поширювати це програмне забезпечення на інші пристрої, що працюють на тій же або подібній прошивці [32].

Аналіз наведених векторів атак на IoT свідчить про те, що ключові компоненти систем Інтернету речей є досить уразливими до дій зловмисників. Незалежно від масштабу та типу середовища, у яке інтегрується система IoT, забезпечення її безпеки повинно бути враховане ще на етапі проєктування. Особливу складність для інженерів та фахівців з інформаційної безпеки становить те, що через технологічні особливості IoT неможливо встановити агента для перевірки систем на зараження або вразливості [33].

Отже, щоб запобігти кібератакам на пристрої та знизити загальні ризики для безпеки компанії, варто дотримуватись кількох ключових рекомендацій.

1. Управління поверхнею атаки, інвентаризація та моніторинг усіх IoT-пристроїв є ключовими елементами захисту систем. Одним із першочергових завдань під час планування безпеки IoT є створення карти підключених пристроїв для їх інвентаризації. Команди з кібербезпеки мають володіти точною інформацією про кількість пристроїв, що використовуються, їх ідентифікатори виробників, серійні номери, а також версії обладнання та прошивки. Моніторинг у режимі реального часу, аналіз та регулярна звітність відіграють критично важливу роль у керуванні ризиками, пов’язаними з

Інтернетом речей. Однак традиційні засоби безпеки кінцевих точок, які покладаються на програмних агентів, не підходять для IoT-пристроїв. Натомість більш сучасні підходи пропонують безагентні рішення, такі як DeviceTotal. Ці інструменти дозволяють моніторити поверхню атаки в реальному часі, оцінювати рівень ризику та проводити неперервний аналіз поведінки й стану IoT-пристроїв. Деякі системи навіть надають можливість управляти ризиками потенційних «атак нульового дня» та прекогнітивними загрозами. Такі технології дають змогу організаціям повноцінно використовувати переваги IoT, мінімізуючи його головний недолік – недостатній рівень захисту [27].

2. Сегментація мережі. Під час успішної кібератаки зловмисник може отримати доступ до всієї мережевої інфраструктури організації. Сегментація допомагає уникнути цього, обмежуючи доступ і мінімізуючи потенційні збитки шляхом зменшення поверхні атаки. Сегментація мережі передбачає поділ внутрішньої мережі на кілька окремих підмереж. Хоча ці сегменти можуть періодично взаємодіяти, вони зазвичай залишаються незалежними та ізольованими один від одного. Такий підхід дозволяє приділяти більше уваги захисту критично важливих частин мережі, де зберігаються найбільш цінні або конфіденційні дані[34].

3. Встановлення надійних паролів для IoT є важливим кроком у забезпеченні безпеки пристроїв. Багато таких пристроїв постачаються зі стандартними слабкими паролями, які легко підібрати. Після першого підключення до мережі найкращою практикою є негайна заміна попередньо встановленого пароля на складніший. Він має бути стійким до підбору, унікальним для кожного пристрою та відповідати політикам безпеки, прийнятим у вашій організації. Надійний пароль допомагає захистити IoT-пристрої від несанкціонованого доступу та мінімізувати ризики для корпоративної мережі [36].

4. Захист IoT-пристроїв на фізичному рівні є критично важливим, оскільки пристрої, доступні ззовні, можуть стати ціллю для фізичного

втручання з боку зловмисників. Такий доступ може призвести до несанкціонованого втручання в роботу системи або завантаження шкідливого програмного забезпечення. Щоб запобігти цьому, необхідно забезпечити надійне розміщення пристроїв у місцях із обмеженим доступом, які виключають можливість фізичного контакту зі сторонніми особами. Цей крок значно знижує ймовірність компрометації пристроїв через пряме втручання.

5. Своєчасне оновлення прошивок є важливим заходом для підвищення безпеки IoT-пристроїв, оскільки нові версії можуть містити виправлення виявлених програмних вразливостей. Регулярне встановлення оновлень суттєво підвищує стійкість пристроїв до атак. Водночас важливо перевіряти автентичність оновлень, оскільки під виглядом офіційної прошивки зловмисники можуть поширювати шкідливе програмне забезпечення. Крім того, варто враховувати, що офіційні оновлення інколи можуть містити власні вразливості. Тому необхідно контролювати версійність прошивки та впроваджувати лише перевірені стабільні оновлення. Автоматизовані системи аналізу прошивок здатні значно полегшити цей процес, допомагаючи підтримувати пристрої в актуальному та безпечному стані [37-38].

1.7 Постановка завдання та мети дослідження

Аналіз IoT-технологій виявив специфіку інформаційної взаємодії в мережах Інтернету речей та показав неможливість прямого використання моделей і алгоритмів, що застосовуються у традиційних комп’ютерних мережах. Це пов’язано з особливостями роботи IoT-мереж, такими як децентралізована структура, низька потужність пристроїв, обмеженість пропускної здатності і підвищені вимоги до ефективного енергоспоживання.

На основі цих особливостей була сформульована задача дисертаційного дослідження: розробка моделей і алгоритмів, що сприяють вибору оптимальних режимів інформаційної взаємодії в мережах Інтернету речей. Мета полягає в тому, щоб адаптувати інформаційні процеси до умов IoT,

враховуючи специфіку протоколів і методів, які використовуються для передачі даних у таких мережах.

Пропонується використовувати засоби моделювання для оцінки ймовірносно-часових характеристик мережі Інтернету речей. Моделі повинні ґрунтуватися на оригінальних методах і алгоритмах, що враховують специфіку інформаційної взаємодії в IoT.

При вирішенні поставленої задачі пропонується оцінювати функціональну залежність між набором параметрів P, які кількісно визначають мережу Інтернету речей, та ймовірносно-часовими характеристиками H, що якісно описують інформаційну взаємодію:

$$H = f(P). \tag{1.1}$$

Ця постановка дозволяє вирішувати три основні задачі моделювання, необхідні для проєктування IoT-мереж:

- пряма задача − оцінка характеристик інформаційної взаємодії при наявних параметрах мережі IoT. Це дозволяє визначити, як зміна певних параметрів мережі впливає на якість зв’язку та взаємодії між елементами IoT;
- зворотна задача − проєктування мережі IoT, що відповідає заданим характеристикам. На основі допустимих значень характеристик H необхідно підібрати параметри мережі P, що дозволяють досягти цих характеристик;
- задача налаштування − побудова моделей, що виявляють функціональну залежність між множинами параметрів P та характеристиками H, на основі запропонованих методів і алгоритмів. Це дозволяє налаштовувати мережу для досягнення оптимальних режимів роботи, враховуючи специфічні умови та вимоги.

Запропоновані моделі дозволять визначати ключові параметри, що впливають на продуктивність мережі, такі як ймовірність успішної доставки даних і затримки передачі. Це допоможе оптимізувати роботу IoT-мережі, покращуючи якість інформаційної взаємодії.

Таким чином, дисертаційна робота спрямована на розробку нових рішень для підвищення ефективності інформаційної взаємодії в IoT-мережах з урахуванням обмежень і вимог, специфічних для таких середовищ.

1.8 Висновки до розділу

У першому розділі «Дослідження організації мереж Інтернету речей» здійснюється детальний аналіз сучасних технологій та архітектурних рішень, які використовуються для побудови IoT-мереж. Розглядаються ключові компоненти таких мереж, включаючи основні IoT-пристрої, вузли та шлюзи, які забезпечують їх функціонування. Особлива увага приділяється методам взаємодії між цими елементами, що включають як бездротові, так і провідні технології передачі даних, а також специфіку комунікаційних протоколів, які забезпечують ефективне та надійне функціонування IoT-мереж.

У розділі також обговорюються різні варіанти архітектури мережі, такі як централізовані, децентралізовані та розподілені підходи, що використовуються залежно від вимог до продуктивності, надійності та енергоспоживання. Проводиться аналіз найбільш популярних IoT-протоколів, таких як MQTT, CoAP, LoRaWAN та інші, які відповідають за транспортні, мережеві та прикладні рівні комунікації.

У процесі цього аналізу акцентується увага на основних викликах, пов’язаних з організацією інформаційної взаємодії в IoT-мережах, таких як забезпечення низького енергоспоживання, масштабованості, безпеки даних та управління мережею в умовах великої кількості підключених пристроїв.

У розділі розглядаються аспекти забезпечення доступу до захищених мереж IoT-пристроїв та ймовірні загрози, які можуть поставити під ризик безпеку таких систем. На основі аналізу технологій та найпоширеніших векторів атак наведено рекомендації для забезпечення цілісності мережі з IoT-пристроями. Ці рекомендації спрямовані на зниження ризиків, пов’язаних із вразливостями IoT, і підвищення загального рівня безпеки таких систем [39].

На основі проведеного аналізу формується постановка задачі дисертаційного дослідження. Основна мета дослідження полягає у розробці нових моделей та алгоритмів, які дозволяють оптимізувати режими інформаційної взаємодії в мережах Інтернету речей. Це включає в себе вибір оптимальних параметрів мережі для досягнення максимальних показників продуктивності, надійності та енергоефективності в різних умовах експлуатації IoT-інфраструктури.

2 МОДЕЛІ ОЦІНКИ ЙМОВІРНОСНО-ЧАСОВИХ ХАРАКТЕРИСТИК ІНФОРМАЦІЙНОЇ ВЗАЄМОДІЇ В МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ

2.1 Аналіз засобів моделювання мережі Інтернету речей

Імітаційне моделювання полягає в створенні та виконанні на комп’ютері програмної системи, яка відтворює структуру та функціонування об’єкта або явища в часі. Ця програмна система називається імітаційною моделлю і представляє об'єкти та сутності реального світу. Зв'язки між елементами об'єкта в реальності відображаються у взаємодіях об'єктів у моделі. Імітаційна модель є спрощеним аналогом реальної системи − як існуючої, так і тієї, яку планується створити. Вона реалізується у вигляді комп'ютерної програми, і її виконання можна вважати процесом симуляції поведінки оригінальної системи в часі. Проведення імітаційного моделювання вимагає проведення серії обчислювальних експериментів та їх подальшої статистичної обробки.

Основні цілі імітаційного моделювання в IoT:

1. Аналіз продуктивності. Оцінка ефективності IoT-систем за різних сценаріїв і умов. Наприклад, можна моделювати роботу мережі IoT при збільшенні кількості пристроїв і визначати, як це впливає на швидкість передачі даних або затримку.

2. Оптимізація ресурсів. Моделювання допомагає оптимізувати використання енергії, пропускної здатності мережі, обчислювальних потужностей та інших ресурсів в IoT. Це особливо актуально для систем з обмеженими ресурсами, таких як бездротові сенсорні мережі.

3. Безпека і стійкість. Імітація дозволяє тестувати стратегії безпеки та оцінювати стійкість IoT-систем до різних загроз і атак. Наприклад, можна досліджувати, як система реагує на спроби несанкціонованого доступу або DoS-атаки.

4. Планування розгортання. Імітаційне моделювання допомагає оцінити, як краще розмістити пристрої IoT в реальному середовищі для оптимальної

продуктивності системи. Це може включати розрахунок оптимальних місць розташування сенсорів або базових станцій.

5. Тестування нових технологій. Моделювання дозволяє випробовувати нові протоколи передачі даних або алгоритми управління без необхідності розгортання фізичних пристроїв. Це дає змогу економити ресурси та час на експериментальних дослідженнях [40].

Наразі пристрої IoT та їхні мережі стають дедалі складнішими і динамічнішими. Під час моделювання таких мереж розробники стикаються з наступними проблемами:

- затримка трафіку – часто виникають ситуації, коли дані не можуть бути своєчасно зібрані та оброблені, або коли необхідні дані взагалі не можуть бути отримані;
- підвищене енергоспоживання – через мобільність пристрої споживають більше енергії, при цьому їхні ресурси залишаються незмінними;
- проблеми, пов’язані з безпекою передачі даних – при динамічних змінах структури мережі з’являються нові можливості для атак.

Далі наведено огляд існуючих інструментів та платформ моделювання та проаналізовано їхні переваги та недоліки. Цей аналіз має на меті допомогти дослідникам у виборі правильного симулятора для розробки алгоритмів, протоколів та методів для бездротових мереж [41].

1. Симулятор мережі NS-2 − це симулятор з відкритим вихідним кодом, який дозволяє проводити об'єктно-орієнтоване моделювання подій. Цей симулятор призначений для досліджень у сфері мережевих технологій, зокрема для вивчення протоколів моделювання, розробки дротових, бездротових та супутникових мереж, а також для дослідження роботи TCP, UDP, телетайпа, FTP і вебсайтів.

NS-2 підтримує два мови програмування: C++ та OTcl. C++ є ефективним для реалізації дизайну, але має обмеження у візуалізації. У NS-2 рівень управління ізольований від рівня обробки даних. Для скорочення часу передачі пакетів та обробки даних планувальник подій і об'єкти базових

мережевих компонентів написані на C++. Таким чином, C++ використовується для реалізації різних протоколів маршрутизації даних, а OTcl − для побудови середовища моделювання.

За допомогою NS-2 можуть бути реалізовані моделі розповсюдження радіохвиль, моделі енергоспоживання; симулятор також підтримує інструменти мобільності, візуалізації та генерації топології [42].

Переваги:

- дозволяє взаємодіяти з реальною системою, підтримує широкий обсяг протоколів на всіх рівнях;
- доступна велика кількість моделей.

Недоліки:

- не підтримує створення методологічно обґрунтованих симуляцій;
- уповільнює моделювання при великій кількості вузлів у моделі (тому не підходить для реалізації моделей мобільних транспортних засобів).

2. NS-3 (Network Simulator 3) є однією з найпоширеніших платформ для моделювання мереж, зокрема бездротових IoT мереж. Дозволяє користувачу визначати топологію системи, додавати в модель концентратори та встановлювати з'єднання між ними. Реалізована на C++ з можливістю імпорту модулів мови програмування Python. NS-3 може функціонувати не тільки як симулятор, але й як емулятор у режимі реального часу, пов'язаний з реальним середовищем. Також вона підтримує використання кількох існуючих стандартів маршрутизації, таких як 802.15.4, 6LoWPAN і RP [43].

Переваги:

- відкритий код і активна спільнота;
- підтримка різноманітних мережевих технологій та протоколів;
- можливість інтеграції з реальними мережевими пристроями;
- гнучкість у налаштуванні різних мережевих параметрів.

Недоліки:

- високий рівень складності для новачків;
- відсутність вбудованих інструментів для візуалізації, що потребує

додаткових ресурсів [42].

3. OMNeT++ − симулятор дротових і бездротових мереж. Підтримує паралельне моделювання розподілу даних і надає інфраструктуру та інструменти для написання симуляцій. Використовується в різних сферах − для моделювання комунікаційних мереж, систем масового обслуговування, протоколів, а також для оцінки продуктивності програмних систем. Симулятор OMNeT++ працює з кількома користувацькими інтерфейсами. Графічна анімація інтерфейсів корисна для налагодження та демонстрації переміщення вузлів, а командний рядок оптимальний для пакетної обробки даних.

Переваги:

- потужна система модулів, що дозволяє розширювати функціонал;
- інтегровані інструменти для візуалізації та налагодження;
- підтримка багатьох протоколів і архітектур, зокрема для IoT.

Недоліки:

- висока складність моделювання складних топологій;
- деякі бібліотеки можуть мати обмежену підтримку сучасних IoT технологій;
- невеликий набір доступних протоколів;
- відсутність можливості перенесення симуляційних моделей у реальні реалізації [44].

4. Matlab/Simulink − проста у використанні платформа, де проблеми та рішення описуються за допомогою знайомих математичних позначень. Написана мовою C та є кросплатформенною. У MatLab надаються різні функції командного рядка, а також інструменти для вимірювання, аналізу та візуалізації даних (цифрова фільтрація, обробка та модуляція сигналів тощо). На базі MatLab існує кілька додатків для науковців, інженерів та дослідників, призначених для технічних обчислень. Завдяки цьому симулятору можна розробляти алгоритми цифрової обробки сигналів, моделювати системи та тестувати програмні й апаратні реалізації, аналізувати і оптимізувати мережеві

параметри, такі як енергоефективність, затримка передачі тощо.

Переваги:

- потужний інструмент для математичного моделювання;
- широкий спектр бібліотек для роботи з IoT мережами;
- простота візуалізації результатів.

Недоліки:

- висока вартість ліцензії;
- висока складність налаштування для специфічних мережевих протоколів.

5. Cooja (Contiki OS) − це симулятор для бездротових сенсорних мереж, який широко використовується для моделювання IoT. Працює разом із операційною системою Contiki, спеціально розробленою для IoT пристроїв .

Переваги:

- орієнтованість на симуляцію сенсорних мереж і IoT;
- можливість моделювати як на рівні окремих вузлів, так і всієї мережі;
- відкритий код і безкоштовність.

Недоліки:

- обмежений вибір протоколів порівняно з NS-3 або OMNeT++;
- вимагає значних ресурсів для моделювання великих мереж [45].

6. CupCarbon − це платформа для моделювання міських IoT мереж (Smart City). Вона дає змогу моделювати комунікацію між IoT пристроями, зокрема бездротовими сенсорними мережами [42].

Переваги:

- спеціалізованість на міських IoT сценаріях;
- інтегрована візуалізація топології мережі;
- можливість розробки сценаріїв для енергоефективних мереж.

Недоліки:

- обмежена гнучкість у порівнянні з більш універсальними симуляторами;
- вузька спеціалізація для сценаріїв розумного міста.

7. TOSSIM − це симулятор, створений для моделювання бездротових сенсорних мереж під операційною системою TinyOS, яка є популярною в IoT пристроях.

Переваги:

- орієнтований на симуляцію великих сенсорних мереж;
- оптимізований для обмежених за ресурсами пристроїв.

Недоліки:

- підтримка лише TinyOS, що обмежує його використання;
- відсутність багатьох сучасних протоколів і можливостей [46].

8. AnyLogic − це багатопарадигмальна платформа для моделювання, яка підтримує кілька типів моделювання: системна динаміка, дискретно-подієве та агентно-орієнтоване моделювання. Вона також використовується для моделювання мереж IoT, що дозволяє створювати складні гібридні моделі взаємодії пристроїв і систем [42].

Переваги:

- підтримка гібридного моделювання, що дає можливість моделювати не лише мережеву взаємодію, але й поведінку систем у більш широкому контексті;
- вбудовані інструменти для візуалізації процесів у реальному часі;
- можливість масштабного моделювання складних систем із багатьма елементами;
- інтеграція з вбудованим Java-середовищем програмування;
- зручний та інтуїтивно зрозумілий графічний інтерфейс;
- широкий і зручний довідковий матеріал, що полегшує роботу з системою.

Недоліки:

- платний доступ, особливо для комерційного використання;
- вища складність моделювання мережевих аспектів порівняно з спеціалізованими симуляторами, як NS-3 чи OMNeT++;
- не завжди оптимальний для детального моделювання низькорівневих

мережевих протоколів, оскільки фокусується на макрорівневому моделюванні.

Порівняння розглянутих симуляторів з відкритим вихідним кодом представлено у вигляді таблиці (табл. 2.1) [47].

Таблиця 2.1 − Порівняльна характеристика симуляторів з відкритим вихідним кодом

Симулятор	Область застосування	Мова програмування	Доступність	Мобільність	Графічна підтримка	Масштабованість
NS-2	Бездротова мережа	C++/OTcl	Бездротовий зв'язок і ad hoc	Підтримується, але обмежено	Обмежена	Невелика
NS-3	Бездротова мережа	C++/Python	Бездротовий зв'язок і ad hoc	Підтримується, але обмежено	Обмежена	Велика
OMNeT++	Бездротова мережа	C++/NED	Бездротовий зв'язок і ad hoc	Підтримується	Хороша	Велика
MATLAB	Бездротова мережа	C++	Бездротовий зв'язок і ad hoc	Підтримується	Відмінна	Дуже велика

Вибір симулятора залежить від цілей дослідження та типу мережі IoT. Для загальних мережевих досліджень підходять NS-3 і OMNeT++*, тоді як Cooja і TOSSIM оптимальні для сенсорних мереж. CupCarbon чудово підходить для міських IoT сценаріїв, а Matlab/Simulink пропонує великі можливості для математичного аналізу.

2.2 Імітаційна модель інформаційної взаємодії

При розробці моделі інформаційної взаємодії (ІВ) в IoT-мережах, з огляду на їх фундаментальні характеристики, доцільно використовувати імітаційне моделювання (ІМ). Практична реалізація ІМ базується на чотирьох основних парадигмах: дискретно-подієве моделювання, динамічне моделювання, системна динаміка за Форрестером та мультиагентний підхід. Ці парадигми представляють різні підходи до побудови моделей змін станів у часі, проте всі вони використовують причинно-наслідковий механізм для просування процесів. Відмінності полягають у виборі математичних і програмних об'єктів, але логіка симуляції залишається однаковою − побудова траєкторії функціонування об'єкта через системний час [48].

Імітаційна модель інформаційної взаємодії в IoT, заснована на мультиагентному підході − це складна система моделювання, яка використовує агентне моделювання для вивчення та аналізу взаємодії між об'єктами Інтернету речей [49]. В мультиагентному моделюванні кожен об'єкт системи (агент) діє як автономний учасник з власними правилами поведінки. У контексті IoT, агенти можуть бути представлені як окремі «речі» (датчики, пристрої), які взаємодіють між собою і з навколишнім середовищем. Кожен агент має свої цілі та стратегії, реагує на змінні умови та приймає рішення.

Дана імітаційна модель інформаційної взаємодії в IoT допомагає:

- аналізувати ефективність інформаційного обміну між пристроями;
- визначати потенційні вузькі місця та оптимальні стратегії взаємодії;
- тестувати нові технології або підходи до управління системами IoT;
- прогнозувати поведінку системи за різних сценаріїв.

Такий підхід широко використовується в логістиці, розумних містах, промисловому Інтернеті речей та охороні здоров'я.

З точки зору реалізації математичних і програмних об'єктів, було обрано мультиагентний підход для моделювання ІВ.

Модель, розроблена в середовищі імітаційного моделювання AnyLogic, працює таким чином: стохастично генеруються запити, що надходять від агентів мережі. Якщо агент успішно проходить валідацію та сертифікований для обміну інформацією, і якщо він передає запит уперше з моменту своєї реєстрації в мережі, його додають до карти мережі для подальшої ідентифікації у просторі. Після цього отримана інформація обробляється і зберігається в пам’яті, а сервер надсилає агенту керуючу команду. Агент повинен підтвердити отримання цієї команди, щоб завершити цикл обміну інформацією (рис. 2.1).

Під час проходження всіх етапів схеми можуть траплятися такі ситуації:

1. Відхилення запиту на передачу даних від агента може статися через конфлікти в джерелах даних або відсутність належного маршруту для їх передачі.

2. Багаторазове додавання одного й того самого агента до карти мережі. Ця проблема усувається шляхом видалення агента з карти, якщо протягом встановленого часу від нього не надходить відповідь.

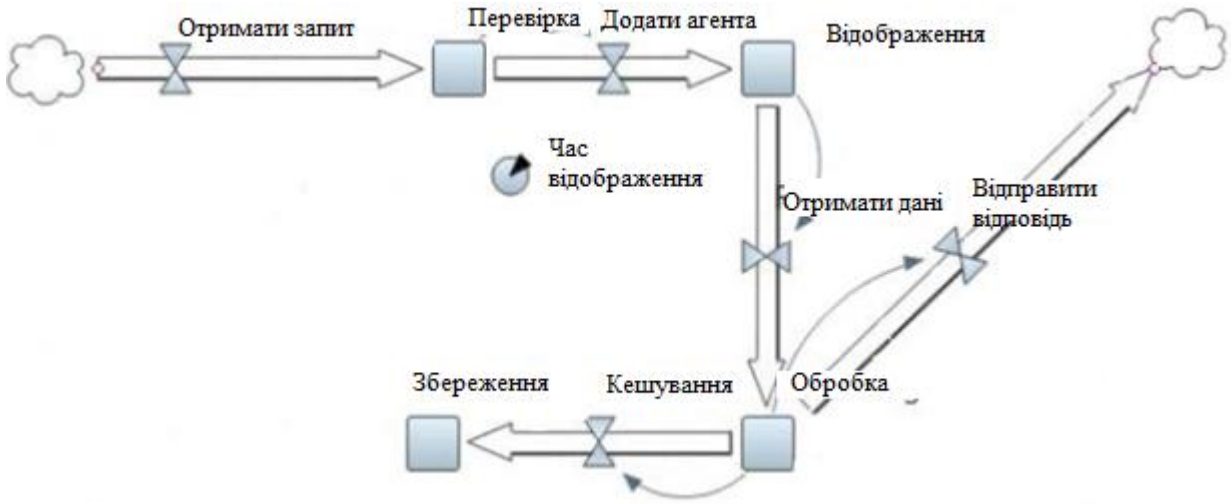


Рисунок 2.1 – Структура імітаційної моделі інформаційної взаємодії

3. «Зациклення» виникає, коли одна й та сама команда повторно надсилається агенту після його відключення або повторного підключення. Якщо агент не здатний виконати команду (через відсутність очікуваних даних), система буде продовжувати пропонувати виконати команду знову. Після визначеної кількості спроб ця інформація фіксується в пам'яті, і система більше не здійснює повторних спроб [50].

4. Запит був успішно оброблений, інформаційна взаємодія завершилася.

У моделі роботи агента, описаній в термінах AnyLogic (рис. 2.2), передбачається, що мобільний об'єкт може переміщатися вільно або слідувати певному алгоритму. Під час руху агент може входити в зону покриття мережі або виходити з неї, залежно від стандарту і використовуваного частотного діапазону. Радіус покриття може варіюватися, наприклад, у випадку стандарту IEEE 802.11.

Перші два блоки схеми деталізують процес переміщення агента та його підключення до мережі. З моменту підключення стає доступним обмін

інформацією з іншими агентами та учасниками мережі. Після цього агент починає обмінюватися повідомленнями, надсилаючи та отримуючи дані.

Слід зауважити, що процес валідації агента (надання дозволу на прийом і передачу інформації) здійснюється на рівні всієї системи, тому в цій блок-схемі вона не відображена. Якщо відповідь на повідомлення не надходить, агент вважається відключеним від мережі. До того ж може виникати конфлікт між кількома об'єктами, які одночасно намагаються передати інформацію.

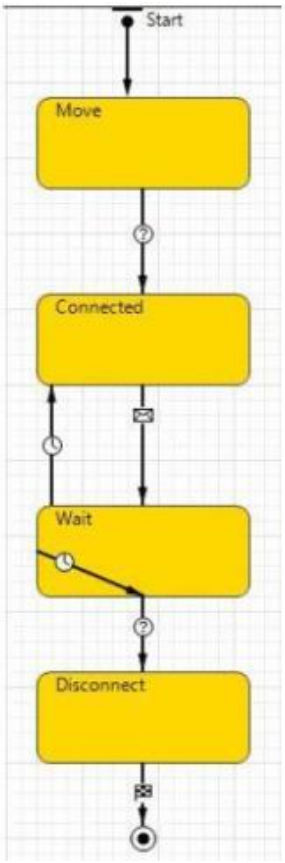


Рисунок 2.2 – Модель роботи агента

2.3 Моделі доступу в «туманних обчисленнях» з роздільною здатністю колізій джерел даних

Одним із ключових аспектів ефективного функціонування IoT є точне прогнозування ймовірнісно-часових характеристик обміну даними, таких як затримки в передачі, ймовірність втрати пакетів, відмови системи та інші

параметри, що впливають на надійність і стабільність мережі.

Розробка моделей для оцінки цих характеристик має велике значення для забезпечення надійної роботи IoT-систем. Від точності оцінки цих параметрів залежить безпека і ефективність процесів у реальному часі. При побудові IoT-мережі важливо враховувати можливість одночасної передачі даних від кількох пристроїв управління, що може призвести до колізій джерел даних. Програмне забезпечення сервера повинно вміти виявляти та усувати такі ситуації. В даному підрозділі запропоновано модель, яка дозволяє оцінити час передачі даних з урахуванням колізій, враховуючи різні режими доступу пристроїв до сервера.

Існує кілька режимів організації доступу до спільних ресурсів, таких як канал зв'язку та обчислювальні потужності сервера: опитування, переривання та множинний доступ [51]. Необхідно розглянути ці режими детальніше для забезпечення коректного моделювання та порівняння результатів залежно від навантаження та інших параметрів IoT.

В режимі опитування сенсорні пристрої починають передачу даних лише після запиту з боку сервера. Якщо у сенсорного пристрою немає готового для передачі пакета даних, спеціальний логічний механізм формує пакет стану та повідомляє сервер про свою працездатність. Час, необхідний для виконання опитування, залежить від фізичних властивостей каналу зв'язку, які визначають затримку поширення сигналу [52].

Час обслуговування t_s сенсорного пристрою у режимі опитування формується так:

$$t_s = t_p + \frac{\bar{b}}{C}, \tag{2.1}$$

де t_p − час, що витрачається на сигнал опитування, с; $\bar{b}$ − середнє значення довжини пакета, біт; C − пропускна спроможність каналу, біт/с [51].

Відповідно, повний цикл взаємодії N сенсорних пристроїв і сервера дорівнює $T_c = Nt_s$. Часова діаграма реалізації режиму опитування в мережі IoT демонструє, як сервер посилає запити до кожного сенсора по черзі, і кожен

сенсор відповідає в своєму часовому слоті, передаючи дані або статус (рис. 2.3) [53].

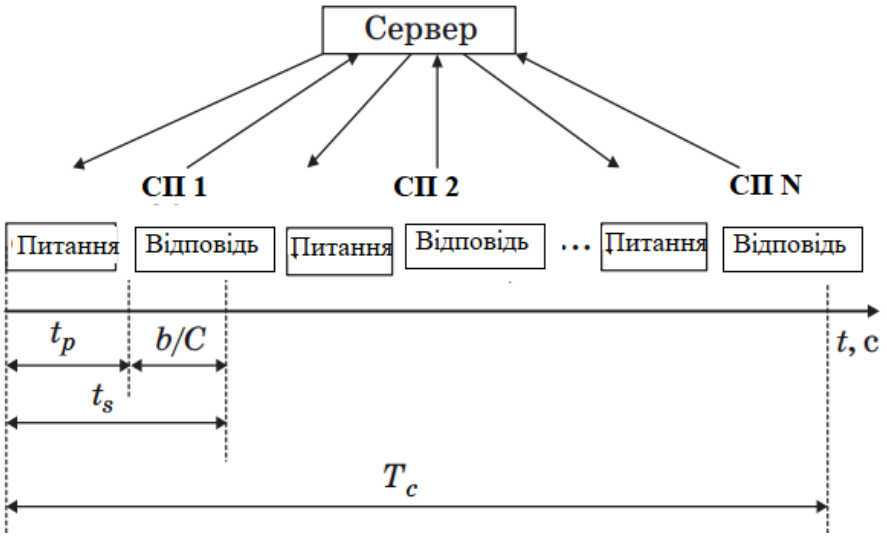


Рисунок 2.3 – Часова діаграма реалізації режиму опитування

Час t_p включає як час формування запиту, так і затримку розповсюдження. Співвідношення b/C – це час, необхідний для передачі b біт зі швидкістю C біт на секунду. Якщо сенсорний пристрій, який опитують, не має готового повідомлення для передачі, то значення b є дуже малим, оскільки передається стандартне керуюче повідомлення про відсутність даних для передачі. У випадку, якщо в буфері накопичено пакет даних, значення b дорівнює кількості біт у пакеті. Співвідношення (2.1) дозволяє оцінити середній час обслуговування одного СП за умови, що дисперсія b значно менша, ніж $\bar{b}$ [51].

Якщо кожен СП передає в середньому λ пакетів на секунду, то для системи, що містить N сенсорних пристроїв, загальна інтенсивність потоку даних складе λN пакетів на секунду, а середній інтервал між їх надходженнями дорівнює $1/\lambda N$ секунд. Отже, для уникнення безмежного зростання черги, час обслуговування $\bar{t}_s$ повинен відповідати умові:

$$t_s = t_p + \frac{b}{C} \leq \frac{1}{\lambda N}. \qquad (2.2)$$

Розв'язуючи рівняння (2.2) відносно C, можна отримати умову, яка

гарантує відсутність черг для кожного сенсорного пристрою:

$$C \geq \frac{\lambda N \bar{b}}{1 - \lambda N t_p}. \quad (2.3)$$

Згідно з цією умовою, кожен сенсорний пристрій опитується кожні $1/\lambda$ секунд, і в той же час у ньому формується черговий пакет даних [51].

У режимі переривань сервер отримує і ставить у чергу сигнали від сенсорних пристроїв про готовність до передачі даних, замість надсилання запитів. Це створює конкуренцію між пристроями за право передати дані. Пристрої спонтанно відправляють запити на сервер, який формує чергу і надсилає підтвердження. Якщо підтвердження не отримано, запит повторюється. Сервер приймає дані від пристрою при вільному каналі і обробляє наступний запит у черзі [54]. Схему та часову діаграму реалізації режиму переривань зображено нижче (рис. 2.4).

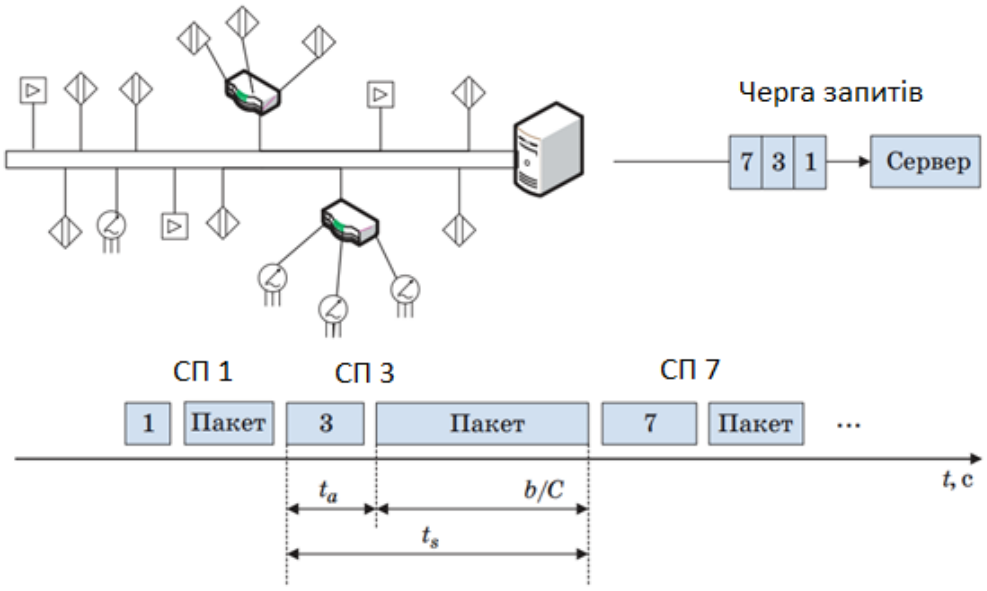


Рисунок 2.4 – Часова діаграма реалізації режиму переривань

Для випадкових запитів на передачу даних ймовірність перетину двох або більше запитів

$$P_c = 1 - e^{-N\lambda T}, \quad (2.4)$$

де T – час передачі пакету довжиною b бітів.

При повторних запитах фактична інтенсивність надходження перевищить λ. Відповідно до закону Пуассона, можна визначити ймовірність виникнення конфлікту:

$$P_c = 1 - e^{-\Lambda T}, \tag{2.5}$$

де Λ − реальна інтенсивність надходження пакетів у T-секундному інтервалі, $\Lambda = N\lambda + P_c\Lambda$.

Тоді продуктивність такої системи дорівнює $N\lambda T = \Lambda T e^{-\Lambda T}$. Щоб знайти максимальну продуктивність, потрібно продиференціювати $N\lambda T$ по ΛT і результат прирівняти до нуля:

$$\frac{d(N\lambda T)}{d\Lambda T} = e^{-\Lambda T} - \Lambda T e^{-\Lambda T} = 0; \quad 1 - \Lambda T = 0, \Lambda T = 1. \tag{2.6}$$

Підставляючи це значення ΛT у вираз для $N\lambda T$, можна визначити максимальну кількість сенсорних пристроїв N_{max}, при якій за заданої пропускної здатності можна уникнути критичного рівня колізій: $(N\lambda T)_{max} = 1/e$. Враховуючи, що $T = b/C$:

$$N_{max} = \frac{C}{e\lambda b}. \tag{2.7}$$

Ці рівняння отримані на основі припущення про наявність окремого каналу для надсилання запитів [51].

Час перебування запитів у черзі t_a можна визначити за допомогою співвідношення інтенсивностей їх надходження та обслуговування. Загальний час обслуговування t_s включає в себе час доступу t_a і час передачі даних $\frac{\bar{b}}{C}$. Таким чином:

$$t_s = t_a + \frac{\bar{b}}{C}. \tag{2.8}$$

Це рівняння аналогічне до рівняння (2.1) для режиму опитування. Як міру ефективності такого режиму взаємодії можна взяти коефіцієнт навантаження ρ.

Якщо $\rho > 1$, то запити надходять швидше, ніж їх встигають обробляти,

що призводить до безмежного зростання черг. При $\rho \leq 1$ черга залишається обмеженою. Фізичний зміст ρ полягає у відношенні середнього часу обслуговування до середнього інтервалу між надходженням запитів [55].

В режимі множинного доступу доступ до сервера розподіляється між сенсорними пристроями через ймовірнісний арбітраж. Якщо пристрій має дані, він починає передавати пакет на сервер. Якщо відбувається конфлікт із даними інших пристроїв, передача переривається і планується повторно. У разі відсутності конфліктів пакет передається успішно. Для уникнення повторних конфліктів пристрої передають дані у випадкові інтервали часу t_r. Інтервали передачі коригуються на основі історії конфліктів. Множинний доступ може бути реалізований у тактованому або нетактованому режимі, залежно від інтервалів передачі. Часову діаграму реалізації режиму множинного доступу зображено нижче (рис. 2.5) [51].

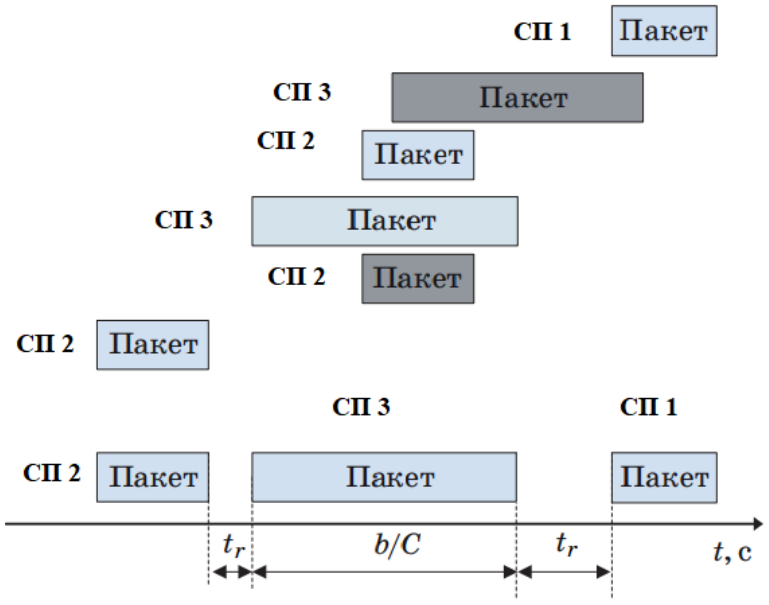


Рисунок 2.5 − Часова діаграма реалізації режиму множинного доступу

Такт дорівнює максимальному інтервалу часу між моментом початку передачі СП і моментом виявлення конфлікту:

$$P = (1 - \frac{1}{N})^{(N-1)}, \tag{2.9}$$

де P − ймовірність того, що рівно один СП спробує передати пакет протягом

такту і отримає доступ до сервера, а N − кількість сенсорних пристроїв у черзі на передачу.

Ефективність режиму множинного доступу, або частка часу, протягом якого пакети передаються без повторних спроб, визначається наступним чином:

$$\rho = \frac{\bar{b}/C}{\bar{b}/C + WT_T}, \tag{2.10}$$

де W − середня кількість тактів, що пройшли під час конкуренції, поки сенсорний пристрій не отримає доступ для передачі даних, $W = (1 - P)/P$; T_T − тривалість такту.

Нехай задано: N − кількість сенсорних пристроїв, що взаємодіють із сервером; ΔT − період дискретизації. За допомогою методів імітаційного моделювання можна оцінити час передачі даних від сенсорних пристроїв на сервер за різних режимів доступу в умовах виникнення колізій [51].

Кожному сенсорному пристрою в мережі IoT присвоюється унікальний ідентифікатор ID, який базується на його IP-адресі. Після цього для кожного СП за допомогою генератора випадкових чисел (RND) обирається випадкове значення часу t початку передачі даних. Це значення вибирається з певного діапазону $[L, R]$, де L і R − мінімальні та максимальні межі часу, між якими пристрій може почати передачу.

$$t = r\Delta T, \tag{2.11}$$

де r− випадкове число, $r \in RND$.

Таке випадкове розподілення часу передачі мінімізує ймовірність колізій між пристроями, які одночасно намагаються передати дані на сервер. Це дозволяє рівномірніше розподілити навантаження на мережу і підвищити її ефективність. Таке планування може адаптуватися до змін навантаження або умов роботи сенсорних пристроїв, що підвищує загальну продуктивність IoT-мережі.

Сервер встановлює з'єднання з тим СП, для якого $\min_t[t_i = \overline{1, n.}]$, нехай ID цього сенсора дорівнює k. Для решти СП новий час початку передачі

визначається за формулою

$$J_i \Delta T = J_{i-1} \Delta T + r_{pak} \Delta T + r \Delta T, \quad (2.12)$$

де J_i –точка відліку початку наступної передачі даних − це момент, коли сенсорний пристрій починає чергову передачу; J_{i-1} − це точка початку передачі даних попереднього, тобто $(k-1)$-го сенсорного пристрою; r_{pak} − кількість точок відліку часу, необхідних для передачі даних (це стала величина, яка залежить від довжини пакета даних); r − число, що визначає кількість точок відліку, необхідних для встановлення наступної передачі [51].

Ці параметри визначають, скільки часу витрачається на різні етапи взаємодії між сервером і сенсорними пристроями в залежності від обраного режиму доступу:

- час t_p, необхідний для запиту k-го СП під час реалізації режиму опитування;
- час доступу t_a при реалізації режиму переривань;
- випадкову затримку щодо завершення передачі даних $(k-1)$-м СП під час реалізації режиму множинного доступу.

Вираз (2.12) є основою для реалізації імітаційної моделі. Нижче розглянуто деякі особливості запропонованої моделі.

Розподіл початку передачі даних від сенсорних пристроїв у часі для кожної ітерації представлений як двовимірна матриця $||p_{ij}||_{N \times M}$, де M − це відліки часу. Для кожного СП визначається випадкова затримка $t_i, i = \overline{1,N}$, що позначає час початку звернення до сервера відносно завершення передачі даних попереднім сенсором, у вигляді випадкового числа $r \in RND[L,R]$.

Тобто, для кожного пристрою моделюється випадкова затримка перед початком передачі, що дозволяє оптимізувати роботу системи, зменшуючи ймовірність колізій між СП. Наприклад, «1» у першому рядку матриці $||p_{ij}||$ означає, що для СП з $ID = 0$ затримка початку передачі $t = 3$. Наявність «1» у другому рядку матриці $||p_{ij}||$ означає, що для СП з $ID = 1$ затримка початку передачі $t = 1$, і так далі:

$$\left\|\begin{matrix}0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\end{matrix}\right\|. \quad (2.13)$$

Далі виконується пошук сенсорного пристрою з ID, який має мінімальну затримку початку передачі. У наведеному прикладі це сенсор з $ID = 1$. У разі виявлення колізії визначаються нові точки відліку початку передачі відповідно до формули (2.12) [56]. Якщо колізії немає, управління передається серверу для зчитування даних із подальшою їх обробкою та зберіганням. Дозвіл на зчитування даних визначається за такими умовами:

- якщо $\left(J_{i_k} - J_{i-1_k}\right) > r, k = \overline{1,N}$, то СП з $ID = k$ передає дані;
- якщо $\left(J_{i_k} - J_{i-1_k}\right) \leq r, k = \overline{1,N}$, це означає, що СП з $ID = k$ потрапив у колізію, і йому призначається нове значення j_i для наступної спроби передачі.

Результати проведених експериментів свідчать про те, що середній час передачі даних зменшується зі збільшенням параметра r_{pak}. Це явище можна пояснити підвищенням точності обчислення часу початку передачі, оскільки з ростом r_{pak} період дискретизації ΔT зменшується. У результаті зменшується «вага» кожної точки відліку в загальному часі моделювання, що призводить до більш точних розрахунків моментів, коли сенсорні пристрої можуть почати передачу своїх даних.

Збільшення r_{pak} дозволяє отримати більше точок вимірювання, що в свою чергу покращує оцінку часу початку передачі. Це важливо для систем, які працюють в умовах конкуренції між сенсорними пристроями за доступ до сервера, оскільки зниження ймовірності колізій сприяє підвищенню ефективності передачі даних.

Проте слід зазначити, що зі збільшенням r_{pak} час моделювання істотно зростає. Це пов'язано з тим, що в системі стає більше параметрів для обробки, що вимагає додаткових обчислювальних ресурсів і часу. Таким чином, існує

певний компроміс між точністю розрахунків і швидкістю виконання моделювання.

Отже, під час проєктування та оптимізації систем Інтернету речей важливо враховувати цей баланс між точністю передачі даних і тривалістю моделювання, щоб досягти найкращих результатів у функціонуванні мережі в цілому. Розуміння цього взаємозв'язку може допомогти інженерам і розробникам знайти оптимальні рішення для забезпечення ефективної роботи систем IoT [57].

Зі збільшенням кількості сенсорних пристроїв, що взаємодіють із сервером, середній час передачі даних також зростає. Це свідчить про те, що модель може бути використана для визначення максимально можливого числа СП, які можуть бути підключені до сервера, виходячи з вимог до часу передачі.

Залежність максимального числа СП від навантаження ρ можна зобразити графічно (рис. 2.6).

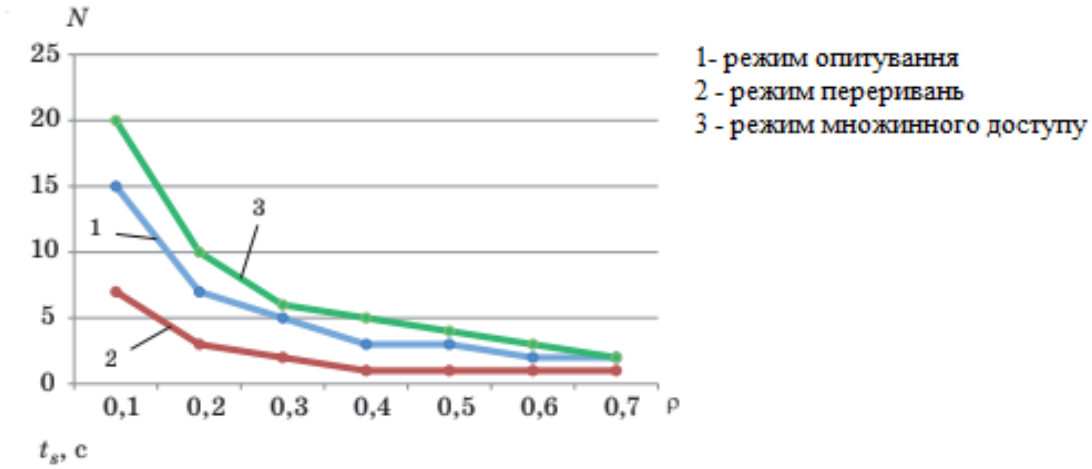


Рисунок 2.6 – Залежність N від ρ

Порівняльні результати показують, що режим множинного доступу має переваги при організації взаємодії СП з сервером за умов низького та середнього навантаження. Це підтверджує, що множинний доступ є ефективним рішенням для роботи з обмеженими ресурсами в мережах IoT, особливо в ситуаціях, коли кількість підключених пристроїв зростає. Таким

чином, при плануванні організації кластерної структури мережі IoT, модель може бути використана для вибору оптимального режиму взаємодії в залежності від прогнозованого навантаження в кластері. Це дозволить максимально ефективно використовувати ресурси мережі та зменшити ймовірність виникнення колізій, що, у свою чергу, сприятиме підвищенню загальної продуктивності системи.

Результати залежності середнього часу передачі даних $\bar{t_s}$ від навантаження ρ при $N = 10$ для різних режимів взаємодії СП та сервера (інтенсивність λ при цьому змінювалася від 0,1 до 100 пакетів/с) подано графічно (рис. 2.7).

Аналіз показує переваги режиму множинного доступу при навантаженнях понад $\rho > 0{,}1$. При високому навантаженні СП постійно активні через підвищену ймовірність колізій, що вимагає їх вирішення. Часті спроби отримати доступ до сервера змушують пристрої залишатися активними. Однак такі умови малореалістичні, оскільки сенсори повинні чергувати сплячий режим з активністю для економії енергії і продовження їхнього терміну служби.

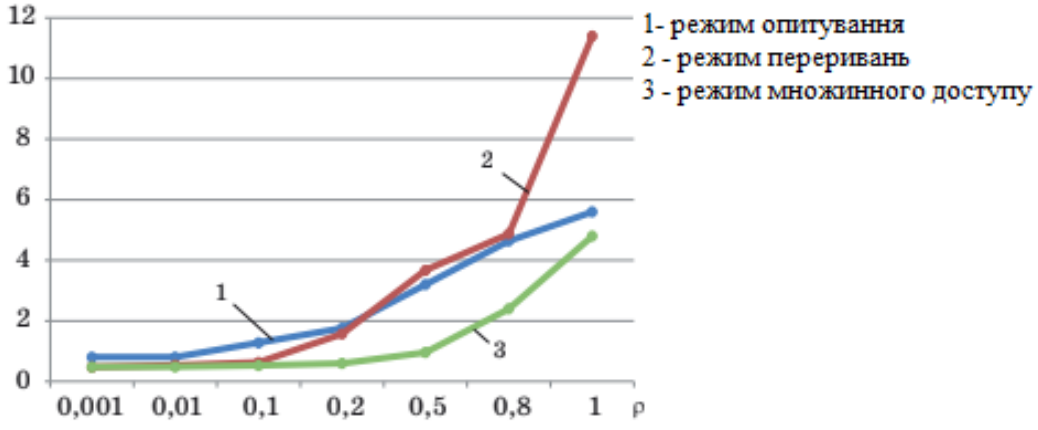


Рисунок 2.7 – Залежність $\bar{t_s}$ від ρ

З іншого боку, при малих навантаженнях $\rho < 0{,}1$, що більше відповідає реальним умовам, різниця в характеристиках часу між режимами є незначною. Проте в таких умовах режим опитування може бути кращим вибором, оскільки всі дії виконує сервер, а сенсорні пристрої можуть залишатися в сплячому

режимі, якщо немає даних для передачі, що допомагає зберегти енергію.

Також цікаво порівняти режими опитування та переривань: при $\rho < 0{,}2$ час передачі даних є меншим при організації взаємодії сенсорних пристроїв з сервером у режимі переривань. Проте з підвищенням ρ взаємодія в режимі опитування стає більш вигідною [58-59].

Проєктування систем Інтернету речей пов'язане з низкою завдань, які вимагають моделювання процесів інформаційної взаємодії. Це дозволяє при прогнозованому трафіку розробляти оптимальні режими функціонування таких систем.

Запропонована модель імітує різні режими доступу сенсорних пристроїв до сервера для передачі даних і дає змогу оцінити ймовірнісно-часові характеристики інформаційної взаємодії з урахуванням стохастичного характеру процесу доступу і ймовірності виникнення колізій. Модель є інваріантною щодо кількості сенсорних пристроїв, формату пакета даних та середнього часу передачі даних в умовах колізій [60].

Ця модель може бути застосована на ранніх етапах проєктування систем Інтернету речей для покращення ефективності їх роботи.

2.4 Ймовірнісна модель встановлення інформаційної взаємодії в мережі Інтернету речей з топологією mesh

У бездротових сенсорних мережах кількість сенсорних пристроїв може змінюватися від кількох сотень до тисяч, залежно від поставленого завдання. Наприклад, у специфікаціях ZigBee кількість СП в одній зоні може досягати 64 000. Мережі такого великого масштабу та високої щільності з обмеженою пропускною здатністю мають забезпечувати послуги з певним рівнем якості обслуговування. На відміну від традиційних мереж, бездротові сенсорні мережі організовуються випадковим чином, а взаємозв'язки між сенсорними пристроями змінюються в часі.

Сенсорні вузли можуть виходити з ладу через різні фактори, такі як

недостатній рівень електроживлення, критичні умови у зовнішньому середовищі або поломка апаратної частини. Вихід з ладу декількох сенсорних вузлів не повинен істотно впливати на роботу всієї мережі або її окремих частин. Іншими словами, Інтернет речей повинен бути стійким до відмов окремих СП і продовжувати підтримувати необхідний рівень якості обслуговування [61].

Для обчислювальних мереж якість обслуговування визначається через ймовірнісно-часові характеристики. Відповідно до моделі інформаційної взаємодії, оцінка ймовірнісно-часових характеристик у мережі Інтернету речей включає сумарний час взаємодії, який можна представити як:

$$t_{i\text{в}} = t_{\text{вз}} + t_{\text{пд}}, \qquad (2.14)$$

де $t_{\text{ів}}$ − загальний час інформаційної взаємодії, $t_{\text{вз}}$ − час встановлення зв'язку між пристроями, $t_{\text{пд}}$− час передачі даних.

Ця формула відображає загальний час, необхідний для встановлення зв'язку та подальшої передачі даних в мережі Інтернету речей. Оцінка цих характеристик дозволяє зрозуміти рівень якості обслуговування в таких мережах і виявити можливі вузькі місця у передачі інформації.

Для оцінки часу встановлення зв'язку $t_{\text{вз}}$ пропонується використовувати ймовірнісну модель встановлення з'єднання в мережі Інтернету речей. Оцінка часу передачі даних $t_{\text{пд}}$ проводиться за допомогою апарату систем масового обслуговування.

Процес встановлення ІВ у мережі Інтернету речей може займати певний час через її динамічний характер: топологія мережі може багаторазово змінюватися в процесі експлуатації через введення нових вузлів, вихід з ладу існуючих або критичні зміни у зовнішньому середовищі. Усі ці аспекти необхідно враховувати при розробці моделі встановлення з'єднання для інформаційної взаємодії [62].

Встановлення ІВ означає побудову логічного каналу, що з'єднує сенсорний пристрій з шлюзом отримувача для подальшої передачі даних. Завдяки mesh-топології мережі Інтернету речей, між джерелом і отримувачем

може бути кілька таких логічних каналів (рис. 2.8) [63].

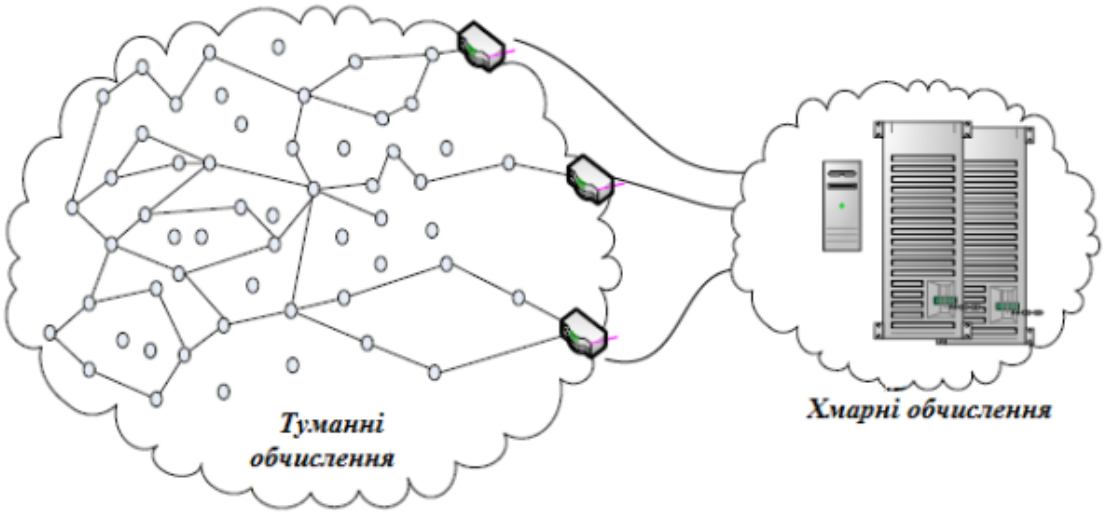


Рисунок 2.8 − Логічні канали для передачі даних від СП в «хмару»

Логічний канал складається з фізичних каналів, що з'єднують сусідні сенсорні пристрої. Побудова логічного каналу відбувається шляхом надсилання керуючого сигналу на встановлення з'єднання. Під час проходження виклику через сенсорні пристрої стан кожного фізичного каналу може бути одним із двох: «1» − канал зайнятий, і передача даних неможлива, або «0» − канал вільний для передачі даних.

Повторні спроби встановлення з'єднання додають стохастичний характер цьому процесу, оскільки виникають вимушені повернення до попередніх сенсорних пристроїв, а кількість фізичних каналів, пройдених викликом при його доставці до адресата, стає випадковою величиною. Врахування цього аспекту дозволяє використовувати імітаційне моделювання для дослідження процесу встановлення логічного каналу для інформаційної взаємодії.

Таким чином, час встановлення інформаційної взаємодії $t_{iв}$ можна розглядати як випадкову величину, яка може бути визначена наступним виразом:

$$t_{i\text{в}} = \sum_{i=1}^{n_k} t_{k_i} + \sum_{i=1}^{n_\text{н}} t_{\text{н}_i} + n_c t_c, \qquad (2.15)$$

де n_k –кількість пройдених фізичних каналів, $n_\text{н}$ – ймовірність успішного встановлення з'єднання, $n_\text{с}$ – число спроб в зафіксованій реалізації процесу встановлення інформаційної взаємодії, в загальному випадку $0 \leq n_\text{с} < n_\text{доп}$, де $n_\text{доп}$ – допустима кількість спроб встановлення взаємодії, t_{k_i} – час проходження i-го фізичного каналу; $t_{\text{н}_i}$ – час зворотного проходження i-го фізичного каналу; $t_\text{с}$ – час перемикання на інший логічний канал [63].

Кожен експеримент на імітаційній моделі генерує три випадкові величини: n_k, $n_\text{н}$ і $n_\text{с}$, що дозволяє оцінити час встановлення інформаційної взаємодії $t_\text{ів}$ відповідно до виразу (2.15).

Оцінивши ці величини в кожному експерименті, можна визначити загальний час встановлення ІВ в мережі Інтернету речей як випадкову величину, що враховує всі можливі фактори, які впливають на процес передачі даних.

Отримане значення часу встановлення інформаційної взаємодії $t_\text{ів}$ визначає результат встановлення зв'язку:

- якщо $t_\text{ів} \leq t_\text{доп}$, то ІВ успішно встановлена в межах допустимого часу;
- якщо $t_\text{ів} > t_\text{доп}$, то зв'язок встановлено, але з низькою якістю обслуговування; це особливо критично для даних термінової доставки, оскільки вони можуть втратити свою актуальність;
- якщо кількість колізій $n_c > n_\text{доп}$, то з'єднання не вдалося встановити, що свідчить про невдалі спроби передачі даних через перевищення допустимої кількості колізій [63].

Таким чином, час встановлення та кількість колізій є ключовими параметрами для оцінки якості обслуговування в мережах Інтернету речей.

Вхідними даними для моделювання процесу встановлення інформаційної взаємодії є:

- множина логічних каналів L_{ij}, які можуть бути побудовані від

джерела i до адресата j. Це набір можливих маршрутів для передачі даних;

- характеристика фізичних каналів, що включає час передачі сигналу-виклику по фізичному каналу в прямому та зворотному напрямках;
- ймовірність повної зайнятості фізичних каналів (тобто, ймовірність того, що канал не доступний для передачі даних);
- допустимий час для встановлення ІВ;
- кількість спроб встановлення ІВ, тобто кількість можливих повторних спроб, якщо з'єднання не було встановлено з першого разу;
- час на повторну спробу, який включає перемикання на інший логічний канал при невдалому встановленні з’єднання.

У моделі накопичуються статистичні дані, які дозволяють оцінити ймовірність встановлення ІВ у межах допустимого часу $t_{\text{доп}}$ та середні та середньоквадратичні значення часу встановлення ІВ $t_{\text{ів}}$, що характеризують процес встановлення ІВ в мережі Інтернету речей.

Ці статистичні показники дають змогу аналізувати ефективність встановлення зв'язку та оцінювати якість обслуговування в мережах IoT [64].

На всій множині логічних каналів L_{ij}, виконується їх упорядкування за кількістю зайнятих фізичних каналів с, тобто за принципом $\text{с} = \text{с}_{min}, \dots, c_{max}$. В окремому експерименті визначається кількість зайнятих фізичних каналів з множини L_{ij}, після чого на отриманій реалізації імітується процес проходження виклику від джерела i до адресата j. Після встановлення ІВ фіксуються такі величини:

- $n_{\text{тр}}$ – кількість успішно переданих запитів;
- $n_{\text{от}}$ – кількість отриманих відповідей;
- $n_{\text{п}}$ – кількість колізій або перешкод, що виникли під час передачі.

Ця процедура повторюється N разів для збору статистичних даних. На основі отриманих результатів розраховується середнє значення часу встановлення ІВ, що дозволяє оцінити ефективність зв'язку в мережі Інтернету речей.

Реалізація кількості втрачених фізичних каналів с здійснюється шляхом

випадкового вибору номерів каналів із множини можливих каналів d, $c \in d$. Номер чергового втраченого фізичного каналу z визначається за формулою:

$$z = [Ud + 1], \tag{2.16}$$

де U – випадкове число, $U \in [0,1]$, отримане за допомогою датчика випадкових чисел. Оператор [ ] означає округлення до найближчого цілого числа з недостачею [63].

Після визначення номера каналу z, цьому фізичному каналу присвоюється значення «1» у множині L_{ij}, що означає його зайнятість або втрату. Процедура визначення z повторюється с раз для всіх втрачених каналів.

Задача розрахунку характеристик встановлення інформаційної взаємодії зводиться до задачі оцінки математичного очікування $M\xi$ дискретної випадкової величини $\xi = f(\alpha)$, де $\alpha = (\alpha_1, \dots, \alpha_d)$ – це вектор параметрів, що має відомий закон розподілу ймовірностей p (тобто $\alpha \sim p$). Випадкова величина ξ може приймати два значення $\xi \in \{0,1\}$, де:

- $\xi = 0$ відповідає успішному встановленню з'єднання;
- $\xi = 1$ – не встановленню з'єднання.

Математичне очікування $M\xi = P\{\xi = 1\}$ має сенс ймовірності неуспішного встановлення з'єднання. При цьому випадкова величина ξ є невиродженою, що означає, що $0 < M\xi < 1$, тобто існує певна ймовірність як успішного, так і неуспішного встановлення з'єднання [65].

Цей підхід дозволяє оцінити ймовірність невдачі при встановленні ІВ і є корисним для аналізу ефективності роботи мережі Інтернету речей, де встановлення з'єднань може бути випадковим і залежати від різних факторів.

Випадкову величину α можна інтерпретувати як вектор $\alpha = (\alpha_1^v, \dots, \alpha_d^v)$, що відображає стан фізичних каналів у множині L_{ij} в контексті їх доступності: канал є або його немає. Кожна випадкова величина $\alpha_i^v \in \{0, 1\}, i = \overline{1, d}$, d – число фізичних каналів в множині L_{ij}. $\alpha_i^v = 1$ означає втрату i-го фізичного каналу зі швидкістю v.

Вважається, що випадкові величини α_i^v незалежні, і якщо α має кінцеву множину значень $\alpha \in X$, де $X = \{x_j; j = 1, \dots, n; n = 2^d\}$, то розподіл $p(x), x \in X$, задається набором ймовірностей $p(x_j = P\{\alpha = x_j\} = p_j \geq 0, j = \overline{1,n})$.

З урахуванням класифікації множини L_{ij} за кількістю зайнятих каналів c, оцінку математичного очікування $M\xi$ можна представити у вигляді суми:

$$M\xi = \sum_{c_{min}}^{c_{max}} M\zeta(c), \tag{2.17}$$

де $M\xi$ — це ймовірність неуспішного встановлення ІВ за наявності c фізичних каналів, що не можуть проводити виклик. Ця сума дозволяє оцінити загальну ймовірність невдачі в процесі встановлення зв'язку в мережі Інтернету речей.

Для задання станів $L_{ij}(c)$ на k-му випробуванні використовується випадковий вибір фізичних каналів c, які не проводять виклик на встановлення інформаційної взаємодії. У результаті вектор α набуває конкретної реалізації x, що містить c одиниць (канали, які зайняті) і $(d - c)$ нулів (вільні канали). Згідно з правилом проходження виклику через множину альтернативних логічних каналів L_{ij} та правилом встановлення ІВ, обчислюється значення:

$$\xi(x|c)_k = f[L_{ij}^k], \tag{2.18}$$

де $\xi(x|c)_k$ − результат встановлення або не встановлення ІВ на k-му випробуванні (для k -ої реалізації множини L_{ij}), $\xi(x|c)_k \in (0,1)$, і ймовірність отриманої реалізації $p(x) = f[L_{ij}^k]$.

Значення $\xi(x|c)_k = 1$ має місце в наступних випадках:

- сигнал виклику не досяг адресата, тобто жоден логічний канал з множини L_{ij} не був побудований успішно;

- кількість спроб перевищила допустиме значення, що означає, що з'єднання не вдалося встановити в межах допустимих спроб;

- інформаційна взаємодія була встановлена, але з низькою якістю обслуговування − це означає, що час встановлення ІВ $t_{\text{ів}}$ перевищив

допустимий час $t_{\text{доп}}$, що є критичним для термінової передачі даних.

Оцінка ймовірності неуспішного встановлення ІВ між СП при наявності с втрачених фізичних каналів на множині всіх логічних каналів L_{ij} може бути виражена наступним чином:

$$M\xi(c) = \frac{C_d^c}{N_{2c}} \sum_{k=1}^{N_{2c}} p(L_{ij}^k | \xi(x|c) = 1), \tag{2.19}$$

де N_c − число розіграних станів (реалізацій) множини L_{ij} .

Імовірність $p(L_{ij})$ розраховується за формулою:

$$p(L_{ij}) = \prod_{i=1}^{h} p(\alpha_i = 1) \prod_{\substack{j=1 \\ j \neq 1}}^{d-h} [1 - p(\alpha_i = 1)], \tag{2.20}$$

Ймовірності $p(\alpha_i = 1), i = \overline{1, d}$ задаються під час вирішення задачі розподілу потоків [63].

Остаточна оцінка ймовірності неуспішного встановлення з'єднання між парою (i, j) матиме наступний вигляд:

$$M\xi(c) = \sum_{c=c_{min}}^{c_{max}} \frac{C_d^c}{N_c} p(L_{ij}^k | \xi(x|c) = 1), \tag{2.21}$$

Таким чином, вираз (2.21) повністю відображає ймовірнісний підхід до встановлення інформаційної взаємодії в мережах Інтернету речей з топологією mesh.

Для оцінки ймовірнісно-часових характеристик доставки даних використовують математичний апарат систем масового обслуговування (СМО) (рис. 2.9). Особливості застосування теорії СМО для моделювання обчислювальних мереж вимагають врахування різних типів взаємодії між окремими СМО, які моделюють вузли мережі. Основний вид взаємодії між вузлами − це послідовне поетапне обслуговування. У випадку встановленого з'єднання така взаємодія відображається у вигляді ланцюжка сенсорних пристроїв.

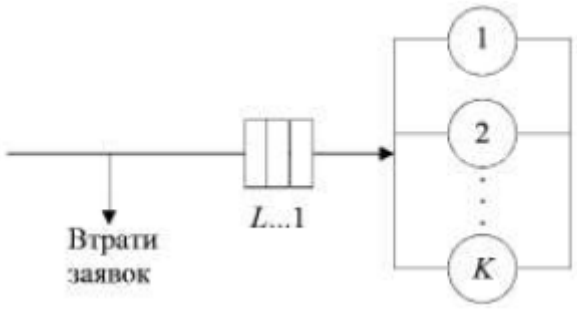


Рисунок 2.9 – Графічне позначення СМО

Система масового обслуговування характеризується такими параметрами:

- $A(t)$ − функція розподілу інтервалів часу між надходженнями заявок, де $0 \leq t$;
- $B(t)$ − функція розподілу часу обслуговування заявок;
- K − кількість каналів обслуговування,$K \geq 1$;
- L − довжина черги, $0 \leq L \leq \infty$.

Ці параметри є ключовими для моделювання та оцінки ефективності роботи мережі Інтернету речей.

Характеристики, які оцінюються за допомогою СМО включають: $P(n)$ − ймовірність втрати заявки; $T_{\text{оч}}$ − середній час очікування; L − середню кількість заявок у черзі; M − середню кількість заявок у системі.

У теорії СМО для оцінки продуктивності використовується апарат перетворення Лапласа-Стілтьєса (ПЛС). Перетворенням Лапласа-Стілтьєса випадкової величини x з функцією розподілу $B(x)$ називають функцію $\beta(s)$, яка визначається так:

$$\beta(s) = \int_0^{\infty} e^{-sx} dB(x), \tag{2.22}$$

де $s \geq 0$ − параметр ПЛС.

Якщо x − неперервна випадкова величина, то ПЛС

$$\beta(s) = \int_0^\infty e^{-sx} dB(x) d(x), \tag{2.23}$$

Для оцінки ймовірнісно-часових характеристик застосовуються такі відомі властивості перетворення Лапласа-Стілтьєса:

1. ПЛС суми незалежних випадкових величин дорівнює добутку ПЛС кожної з цих величин, тобто, якщо дві незалежні випадкові величини мають ПЛС функцій розподілів $\beta_1(s)$ і $\beta_2(s)$, то ПЛС функції розподілу їх суми дорівнює $\beta_1(s)\beta_2(s)$.

2. Якщо B_k – це k-й момент випадкової величини щодо початку координат, то

$$B_k = (-1)^k \frac{d^k \beta(s)}{ds^k}|_{s=0}, \tag{2.24}$$

тобто, моменти випадкової величини можна отримати шляхом багаторазового диференціювання перетворення Лапласа-Стілтьєса функції розподілу в точці $s = 0$. Перший центральний момент характеризує математичне сподівання цієї випадкової величини,

$$\bar{t} = \beta = \frac{\beta(s)}{ds}, \tag{2.25}$$

а другий центральний момент використовується для обчислення дисперсії випадкової величини.

$$\sigma^2 = B_2 - B_1^2 = \frac{d^2\beta(s)}{ds^2}|_{s=0} - \left[-\frac{d\beta(s)}{ds}|_{s=0}\right]^2. \tag{2.26}$$

3. $\lim\limits_{s\to 0} \beta(s) = \lim\limits_{t\to\infty} B(t)$.

4. Величина $e^{-sx}B(x_i)$ є ймовірністю складної події, яка полягає в тому, що випадкова величина не перевищить значення x_i (співмножник $B(x_i)$), і до того ж, на інтервалі $[0, x_i]$ не відбудеться жодної «катастрофи» (співмножник e^{-sx}). Параметр s інтерпретується як інтенсивність «катастроф». Інтегрування по всьому діапазону дає $\int_0^\infty e^{-sx}\, dB(x) = \beta(s)$.

Таким чином, перетворення Лапласа-Стілтьєса при оцінці середнього часу затримки $t_{\text{пд}}$ має ймовірнісний сенс: воно визначає ймовірність того, що

за час $t_{\text{доп}}$ вдасться передати всі необхідні дані.

У контексті Інтернету речей час передачі даних описується через перетворення ПЛС [66]. Передача між елементами відбувається одночасно в K незалежних пуассонівських потоках даних, кожен з яких має інтенсивність $\lambda_k, k = 1, \dots, K$, і обробляється за довільним законом. Тобто кожний сегмент маршруту моделюється як система масового обслуговування типу $M|G|1$.

Для системи $M|G|1$ використовується рівняння Поллачека-Хинчина для ПЛС функції розподілу часу очікування $W(s)$

$$W(s) = \frac{s(1-\rho)}{s - \lambda + \lambda\beta(s)}, \tag{2.27}$$

де ρ – коефіцієнт завантаження каналу; λ – інтенсивність надходження пакетів у канал, а $\beta(s)$ – ПЛС часу обробки пакета на серверному пристрої. Щоб обчислити $W(s)$, потрібно знайти ПЛС функції розподілу часу обробки пакета $\beta(s)$. Для випадкової величини t, яка представляє час обробки пакета, з експоненційним розподілом ймовірностей

$$B_k(t) = 1 - e^{-\mu_k t}, \tag{2.28}$$

де $\mu_k = C/l_k$ – пропускна здатність серверного пристрою в пакетах$/c$; C – пропускна здатність СП в бітах$/c$; l_k – довжина пакета даних k-го класу в бітах, де $k = 1, \dots, K$.

Перетворення Лапласа-Стілтьєса

$$\beta(s) = \frac{\mu}{(\mu + s)} = \frac{1}{(1 + st)}. \tag{2.29}$$

Для випадкової величини t з рівномірною функцією розподілу

$$B_k(t) = \frac{t - \alpha}{b - \alpha}, \text{при } \alpha \le t \le b, \tag{2.30}$$

перетворення Лапласа-Стілтьєса

$$\beta(s) = \frac{e^{-s\alpha} - e^{-sb}}{s(b - a)}. \tag{2.31}$$

Відповідно до розглянутих властивостей ПЛС тривалість передачі даних k-го класу від вузла i до вузла j визначається як

$$\beta_k(t) = \prod_{d=1}^{N} W_d(t)\beta_d(t), \qquad (2.32)$$

де $\beta_d(t)$ − ПЛС функції розподілу часу обробки пакета в d-му каналі маршруту; $W_d(t)$ − ПЛС функції розподілу часу очікування $W(s)$ в d-му каналі маршруту.

Таким чином, цей розподіл часу безперервної передачі залежить від векторів інтенсивностей надходження та обслуговування пакетів і визначається як функція $\beta_k(t) = f(\lambda_k, \mu_k)$.

Метод забезпечує розподіл часу перебування даних k-го класу в мережі Інтернету речей, враховуючи заданий вектор ймовірностей виникнення помилок на елементах маршруту для всіх пар взаємодіючих серверних пристроїв. Перетворення Лапласа-Стілтьєса дозволяє, при встановлених обмеженнях на час $t_{\text{доп}}$ для часу $t_{\text{пд}}$ певного маршруту, оцінити можливе навантаження на цей маршрут і вибрати відповідний алгоритм самоорганізації мережі [67].

2.5 Висновки до розділу

У другому розділі «Моделі оцінки ймовірносно-часових характеристик інформаційної взаємодії в мережі Інтернету речей» здійснюється аналіз сучасних засобів моделювання IoT. Проведено порівняння наявних інструментів, таких як імітаційні платформи (OMNeT++, NS-3, Cooja), які дозволяють моделювати поведінку мережі, протоколи передачі даних, і процеси управління ресурсами в умовах обмежень, характерних для IoT. Імітаційні середовища є важливим інструментом для дослідження поведінки складних систем IoT до їх фактичного розгортання, що дозволяє вивчати взаємодію пристроїв і оптимізувати їхнє функціонування в різних середовищах.

Також в розділі детально розглянута імітаційна модель інформаційної взаємодії в мережі IoT. Цей підхід дозволяє моделювати динаміку взаємодії

між пристроями IoT (агентами), оцінювати продуктивність системи за допомогою симуляції, а також виявляти можливі проблеми, такі як затримки при передачі даних, колізії або проблеми маршрутизації. Використання мультиагентних моделей є ефективним для відображення складних процесів, де кожен пристрій виступає як окремий агент, що діє автономно і взаємодіє з іншими агентами або серверами.

Одним з важливих аспектів цієї моделі є можливість моделювання ймовірних сценаріїв відключення або повторного підключення агентів до мережі, що є типовим для мобільних або динамічних середовищ IoT. Імітаційна модель дозволяє враховувати такі фактори, як періодичне відключення від мережі або затримки в передачі даних через зміну топології.

У розділі було розглянуто і проаналізовано два підходи до оцінки ймовірнісно-часових характеристик інформаційної взаємодії в мережах IoT. Основну увагу було зосереджено на методах моделювання таких систем, а також на особливостях передачі даних і управління ресурсами в умовах різних архітектур IoT, зокрема «туманних обчислень» і мережевих топологій на основі mesh.

Запропонована модель доступу в «туманних обчисленнях» з роздільною здатністю колізій джерел даних імітує різні режими доступу сенсорних пристроїв до сервера для передачі даних та дозволяє оцінити ймовірнісно-часові характеристики інформаційної взаємодії з урахуванням стохастичного характеру процесу доступу і ймовірності виникнення колізій. Модель інваріантна до кількості сенсорних пристроїв, формату пакета даних та середнього часу передачі даних за умов виникнення колізій. Вона може знайти застосування на ранніх етапах проєктування систем Інтернету речей.

Запропонована ймовірнісна модель встановлення інформаційної взаємодії в мережі Інтернету речей за топологією mesh, заснована на мультиагентному підході, побудована з урахуванням основних характеристик технології IoT. Модель дозволяє оцінити як абсолютні, так і ймовірнісні показники інформаційної взаємодії. У моделі враховані реальні умови процесу

обміну даними: наявність непрацездатних каналів і точок доступу, обмежена кількість спроб для повторного встановлення з'єднань, а також можливість використання альтернативних маршрутів.

Для оцінки часу передачі даних запропоновано використовувати апарат перетворення Лапласа-Стілтьєса. Перший центральний момент ПЛС дозволяє обчислити середній час передачі даних в умовах встановленої інформаційної взаємодії. Ймовірнісний зміст ПЛС дозволяє оцінити ймовірність успішної доставки даних.

У результаті аналізу та порівняння різних моделей було зроблено висновок, що імітаційне моделювання та ймовірнісні підходи є важливими інструментами для дослідження й оптимізації систем IoT. Вони дозволяють виявляти та вирішувати проблеми, пов'язані з передачею даних, доступом до ресурсів і управлінням конфліктами в умовах реального часу. Важливу роль у цьому відіграють туманні обчислення, які забезпечують ефективне використання ресурсів, та mesh-топології, що гарантують високу надійність і масштабованість систем IoT.

3 ЙМОВІРНІСНІ АЛГОРИТМИ САМООРГАНІЗАЦІЇ МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ

3.1 Аналіз алгоритмів самоорганізації мережі Інтернету речей

Ймовірнісні алгоритми самоорганізації мережі Інтернету речей − це математичні методи, які використовують ймовірнісні підходи для автоматичного налаштування та оптимізації мережі IoT без прямого втручання людини. У таких алгоритмах сенсорні пристрої можуть самостійно визначати оптимальні маршрути передачі даних, розподіляти ресурси та координувати роботу між собою, виходячи зі стану мережі та доступних ресурсів. Це дозволяє мережі адаптуватися до змін, таких як зміна кількості пристроїв або умов передачі, що особливо важливо для великих динамічних мереж [68].

Основні переваги ймовірнісних алгоритмів самоорганізації:

- адаптивність − мережа може швидко реагувати на зміни, зокрема на втрату зв'язку з окремими пристроями;
- енергоефективність − дозволяє економити енергію, оскільки пристрої можуть знижувати активність, коли це можливо, або визначати найкращі канали зв'язку з мінімальним споживанням енергії;
- масштабованість − підходить для великих мереж IoT, оскільки не вимагає централізованого управління.

Алгоритми самоорганізації розвиваються з урахуванням основних вимог бездротових сенсорних мереж (БСМ). По-перше, вони мають забезпечувати масштабованість, тобто мережа повинна підтримувати роботу з десятками, а іноді й сотнями тисяч вузлів. З часом вузли можуть виходити з ладу або приєднуватися до мережі через додавання нових компонентів (мотів), тому алгоритми самоорганізації повинні швидко реагувати на такі зміни та адаптуватися до реструктуризації.

По-друге, алгоритми самоорганізації повинні бути розподіленими та децентралізованими, що допомагає уникнути порушень у функціонуванні

мережі через відмову одного або кількох вузлів.

По-третє, важливою є вимога енергоефективності, оскільки енергія є критичним ресурсом для мереж з автономним живленням. Крім того, незважаючи на розвиток швидкісного бездротового зв’язку, пропускна здатність спільного середовища передачі залишається обмеженим ресурсом, який потребує раціонального використання.

З точки зору структурування мережі, найефективнішими є алгоритми, що ґрунтуються на кластеризації вузлів [69]. Їхній принцип полягає у функціональній різнорідності вузлів: мережа ділиться на кластери за певними правилами, що створює функціональні області в межах мережі (рис. 3.1).

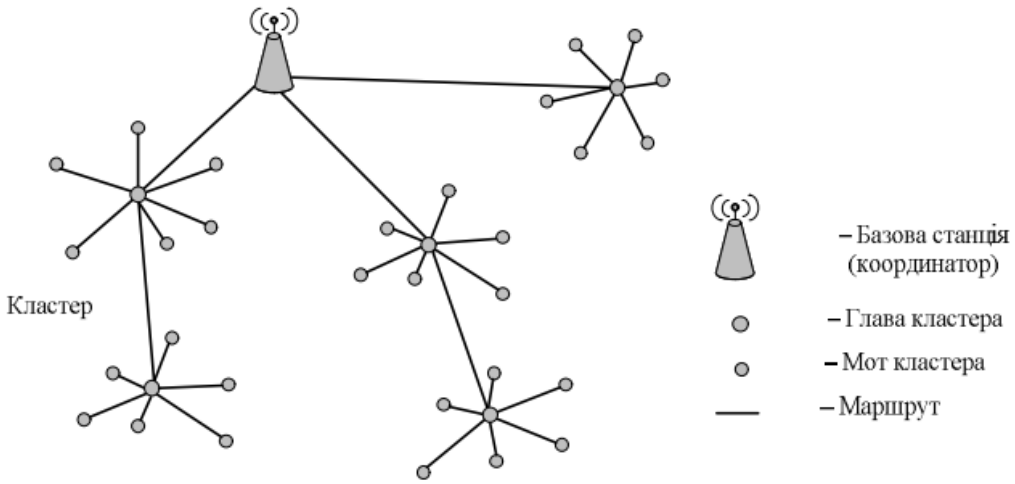


Рисунок 3.1 − Кластерна організація БСМ

Іншим типом алгоритмів самоорганізації є алгоритми формування ланцюжків зв’язків [70]. Основна ідея цих алгоритмів полягає у створенні послідовних маршрутів від одного вузла до іншого, так що кожен вузол передає інформацію лише найближчому сусідньому моту (рис. 3.2).

До інших типів алгоритмів самоорганізації належать алгоритми з деревоподібною та чарунковою структурою [71], а також ті, що використовують інформацію про місцезнаходження або ґрунтуються на гетерогенності вузлів [72].

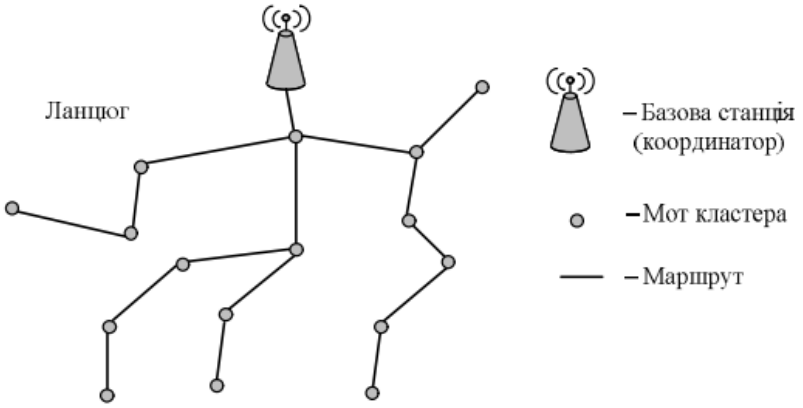


Рисунок 3.2 − Побудова ланцюжка зв’язків

Як показує аналіз літератури, питання порівняльного дослідження алгоритмів самоорганізації для БСМ та вибору найбільш підходящого алгоритму залишаються недостатньо вивченими, що робить цю тему актуальною для проєктування бездротових сенсорних мереж.

Розвитком алгоритмів просторової кластеризації для самоорганізації однорангових сенсорних мереж є алгоритм Optics based [70]. Він самостійно організовує сенсорні вузли, утворюючи кластери за допомогою процедури впорядкування, яка схожа на процес, застосовуваний у раніше відомому алгоритмі Optics. Цей підхід використовує оптичні технології для комунікації між вузлами. Часто застосовується у високошвидкісних мережах передачі даних, де оптичні зв’язки забезпечують стабільну передачу інформації з мінімальними затримками.

Серед алгоритмів кластеризації обмеженого розміру є ті, що базуються на розширенні кільця пошуку. Алгоритм Expanding Ring проводить пошук у раундах, починаючи з мінімального значення межі хопів (кількість переходів), яке дорівнює одиниці. Спочатку перевіряються сусідні вузли, і якщо шлях не знайдено, алгоритм розширює пошук на більшу відстань. Використовується для ефективного пошуку маршрутів у мережах.

В алгоритмі Rapid [73] ініціатору призначається бюджет B, частину якого він залишає собі, а решту розподіляє між своїми сусідами, надсилаючи повідомлення кожному з них. Після вичерпання бюджету батьківський вузол

отримує інформацію про всі свої дочірні вузли, і алгоритм завершується. Використання підтверджень допомагає визначити розмір і глибину дерева, а також максимальну кількість хопів.

Алгоритм Persistentє покращеною версією Rapid, з додатковою функцією управління розміром кластерів.

Алгоритм Payment scheme працює за принципом «схеми оплати», де використовується потужність передавання як медіатор, і він враховує місцезнаходження вузлів [70]. На основі індикатора якості зв'язку кожен вузол може розрахувати мінімальну потужність для надійного передавання даних іншим вузлам або координатору. Існує потенціал для економії енергії за рахунок залучення ретранслятора між передавальним вузлом і отримувачем.

Алгоритм Bio-Inspired Mechanisms based є міждисциплінарним і використовує негативний зворотний зв'язок, що базується на перенесенні принципів клітинної біології в мережеву взаємодію вузлів сенсорної мережі [71] . Його мета − забезпечити самоорганізацію в умовах ненадійних каналів зв'язку, допомагаючи вузлам мережі отримувати інформацію про доступні ресурси та маршрути для передачі контрольних повідомлень.

Алгоритм SIDA (Self-Organized ID Assignment) дозволяє оптимізувати обмін даними між датчиками і координатором за рахунок самоорганізації з використанням ідентифікації змінної довжини. Основна ідея полягає в створенні накладеного бінарного дерева.

Технологія ультраширокосмугового (UWB) передавання має переваги для самоорганізації в сенсорних мережах, зокрема, низьку потужність передавання та енергоефективність. Використання UWB-TR допомагає знизити вимоги до точності синхронізації і якості бездротового каналу.

Алгоритм BOOTUP спрямований на економію енергії шляхом зменшення кількості повідомлень, необхідних для побудови мережі [72]. Це досягається об'єднанням фази виявлення зв'язків із фазою підключення, що усуває потребу в загальній синхронізації мережі.

Результати порівняльного аналізу методів вибору оптимального

проєктного варіанту показали, що метод аналізу ієрархій є більш ефективним для визначення найкращого алгоритму з огляду на середньоквадратичну похибку прийняття рішень, отриманих від групи експертів.

Суть методу аналізу ієрархій полягає в тому, щоб визначити вектор глобальних пріоритетів, за максимальним значенням компонент якого можна виявити відповідний переважний алгоритм самоорганізації БСМ. Цей вектор глобальних пріоритетів визначається через головний власний вектор, який відповідає максимальному власному числу, обчисленому на основі матриці парних порівнянь

$$A = \begin{pmatrix} a_{11} \dots a_{1j} \dots a_{1n} \\ a_{21} \dots a_{2j} \dots a_{2n} \\ \dots \dots \dots a_{ij} \dots \dots \dots \\ a_{n1} \dots a_{nj} \dots a_{nn} \end{pmatrix}, \quad (3.1)$$

де a_{ij} – оцінки парних порівнянь алгоритмів самоорганізації, надані експертами; n – кількість порівнюваних алгоритмів.

У результаті обробки отриманої матриці (3.1) можна визначити компоненти власного вектора V_j та вектора глобальних пріоритетів P_j.

$$P_j = \frac{V_j}{S}, V_j = \sqrt[n]{\prod_{i=1}^{n} a_{ij}}, j = \overline{1,n}, S = \sum_{j=1}^{n} V_j. \quad (3.2)$$

Після визначення головного власного значення матриці парних порівнянь розраховують оцінку узгодженості думок експертів (ОС). Якщо значення $OC \leq 1{,}5$, ступінь узгодженості суджень експертів вважається прийнятним. В іншому випадку експертам рекомендується переглянути свої оцінки.

Формалізовану процедуру вибору переважного алгоритму самоорганізації бездротових сенсорних мереж на основі методу аналізу ієрархій було реалізовано програмно в середовищі Excel.

Для вибору оптимального алгоритму самоорганізації БСМ були використані зазначені вище дев’яти алгоритмів: №1 − Optics based algorithm, №2 − Persistent algorithm, №3 − Payment scheme algorithm, №4 − Rapid

algorithm, №5 − Bio-inspired Mechanisms based algorithm, №6 − SIDA algorithm, №7 − UWB technology based algorithm, №8 − Expanding ring algorithm, №9 − BOOTUP algorithm.

Нижче представлено матрицю парних порівнянь алгоритмів самоорганізації БСМ, отриману на основі оцінок 26 експертів (табл. 3.1). Також наведено результати обробки цієї матриці, зокрема, обчислення компонент власного вектора V_j і вектора пріоритетів P_j. Оцінка узгодженості ОС становить 0,046, що знаходиться в межах норми.

Таблиця 3.1 − Результати порівнянь алгоритмів самоорганізації методом аналізу ієрархій

Експерт № 1-26	№1	№2	№3	№4	№5	№6	№7	№8	№9	V_j	P_j
№1	1	0,16	0,14	0,19	0,32	1,75	2,57	0,24	0,47	0,45	0,03
№2	6,23	1	0,45	2,28	4,24	7,15	8,23	3,33	5,09	3,11	0,23
№3	7,15	2,23	1	3,14	5,63	8,23	9,13	4,32	6,13	4,35	0,32
№4	5,15	0,44	0,32	1	3,73	6,85	7,14	2,28	4,13	2,20	0,16
№5	3,13	0,24	0,18	0,27	1	4,12	5,38	0,42	2,14	0,96	0,07
№6	0,567	0,14	0,12	0,15	0,24	1	2,13	0,19	0,27	0,32	0,02
№7	0,39	0,12	0,11	0,14	0,19	0,47	1	0,16	0,24	0,24	0,02
№8	4,23	0,3	0,23	0,44	2,37	5,24	6,34	1	3,34	1,48	0,11
№9	2,15	0,19	0,16	0,24	0,47	3,67	4,12	0,29	1	0,69	0,04
					ОС=4,6%						

Аналіз компонент вектора пріоритетів P_j показує, що кращим алгоритмом самоорганізації БСМ є алгоритм № 3 – Payment scheme algorithm, якому відповідає максимальна компонента вектора пріоритетів, шо застосовує схему оплати, використовуючи потужність передавання як медіатор, і вимагає знання місця розташування вузлів [74].

Для адаптації до різних умов і сценаріїв мереж IoT постійно розробляються нові підходи. Існуючі алгоритми часто модифікуються та комбінуються для створення гібридних підходів. Можна сказати, що загалом існують десятки варіантів ймовірнісних алгоритмів самоорганізації, які застосовуються до конкретних завдань, таких як кластеризація, маршрутизація

та балансування навантаження.

Генетичний алгоритм (ГА) для самоорганізації розміщення сенсорних вузлів (СВ) займає особливе місце серед алгоритмів самоорганізації завдяки своїм можливостям еволюційної оптимізації [75]. У мережах IoT, а також у багатьох інших розподілених системах, його використовують для досягнення кількох важливих цілей:

- оптимізація маршрутизації: забезпечення мінімальної затримки і енергоспоживання під час передачі даних між вузлами;
- управління енергоспоживанням: збільшення часу автономної роботи вузлів, особливо для мереж, що працюють на батареях;
- розташування вузлів: вибір оптимальних місць для встановлення сенсорів для досягнення максимальної площі покриття;
- балансування навантаження: рівномірний розподіл трафіку між вузлами для запобігання перевантаження мережі;
- глобальна оптимізація: на відміну від таких підходів, як Expanding Ring Algorithm, генетичний алгоритм має здатність знаходити глобально оптимальні рішення, що знижує ризик потрапити в локальні мінімуми і підвищує ефективність мережі в цілому.

Основні етапи роботи генетичного алгоритму для мереж IoT:

1. Ініціалізація популяції: алгоритм починає з набору випадкових конфігурацій мережі, які представляють можливі рішення задачі. Кожна конфігурація може описувати розташування вузлів, енергетичні параметри, схеми маршрутизації або інші параметри мережі IoT.

2. Оцінка придатності (фітнес-функція): для кожної конфігурації обчислюється значення фітнес-функції, яка відображає якість цього рішення. Наприклад, для мережі IoT фітнес-функція може враховувати такі метрики, як мінімізація енергоспоживання, зниження затримки передачі даних, покращення пропускної здатності або максимальне покриття мережі.

3. Селекція (відбір): конфігурації, які демонструють кращі показники за фітнес-функцією, отримують більше шансів бути обраними для подальшого

схрещування і утворення нових рішень. Це нагадує процес природного відбору.

4. Кросовер (схрещування): кращі конфігурації комбінуються між собою для створення нових варіантів мережевих налаштувань. Це дозволяє об'єднувати корисні параметри різних конфігурацій, що може привести до створення більш ефективних рішень.

5. Мутація: в окремих конфігураціях випадково змінюються окремі параметри, що сприяє дослідженню нових варіантів і запобігає потраплянню в локальні оптимуми. Наприклад, можна змінити позиції певних вузлів або коригувати параметри енергозбереження.

6. Заміна покоління: нова популяція замінює стару, і процес повторюється, поки не буде досягнуто бажаного результату − наприклад, мінімізації енергоспоживання, оптимального маршруту або заданого рівня покриття мережі.

Генетичний алгоритм самоорганізації розміщення сенсорних вузлів займає місце серед еволюційних і біоінспірованих алгоритмів у списку (як Bio-inspired Mechanisms based algorithm) і відповідає потребам у задачах глобальної оптимізації. У порівнянні з іншими алгоритмами, такими як Rapid Algorithm або BOOTUP, він може бути більш ресурсомістким, оскільки вимагає обчислень для кількох поколінь, але при цьому забезпечує високу якість рішення. Тому генетичний алгоритм часто застосовується на етапі проєктування і планування мережі, коли важливо досягти глобально оптимального розміщення вузлів [76].

Таким чином, генетичний алгоритм доповнює інші алгоритми самоорганізації, забезпечуючи еволюційний підхід до оптимізації і здатність до адаптації, що робить його корисним для довготривалих рішень у розподілених мережах, таких як IoT. Крім того, його еволюційна природа дозволяє розвивати системи в реальному часі, забезпечуючи їхню стабільність і ефективність у тривалій перспективі.

3.2 Генетичний алгоритм самоорганізації мережі IoT

Ефективна організація інформаційної взаємодії в Інтернеті речей значною мірою залежить від створення альтернативних маршрутів і оптимального розташування сенсорних пристроїв. З огляду на базові особливості Інтернету речей, алгоритми, що використовуються для пошуку альтернативних маршрутів і розміщення пристроїв, повинні бути ймовірнісними за своєю природою.

Одним із підходів до вирішення таких завдань є використання еволюційних алгоритмів. Генетичні алгоритми зарекомендували себе як конкурентоспроможні при вирішенні багатьох NP-важких задач, особливо в практичних застосуваннях, де математичні моделі мають складну структуру і застосування стандартних методів, таких як метод гілок і меж, динамічне або лінійне програмування, є значно ускладненим [77].

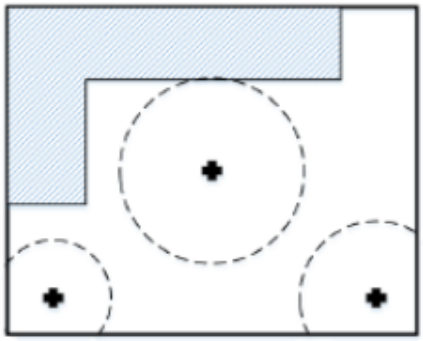
Для самоорганізації сенсорних пристроїв у просторі необхідно використовувати алгоритм, який дозволяє їм автономно розподілятися для досягнення максимально можливого покриття території.

Передбачається, що простір поділяється на окремі фрагменти, кожен із яких може містити три основні сутності: вільний простір, перешкоди та сенсорні пристрої з певним радіусом дії R.

Якби сенсорні пристрої були однорідними, можна було б застосовувати відомі алгоритми, такі як задача упаковки куль. Однак зазвичай СП є неоднорідними, тобто мають різний радіус дії і призначені для моніторингу різних видів інформації. Для вирішення цього завдання запропоновано наступний генетичний алгоритм [78].

Для спрощення область розміщення сенсорних пристроїв можна представити як площину, на якій розташовані перешкоди і самі пристрої. Цю площину можна математично описати у вигляді матриці A, де порожнє місце на площині позначається нулем, перешкоди позначаються одиницею, а різні СП − цифрами від 2 до $N + 1$, де N − кількість різних сенсорних пристроїв

(рис. 3.3) [79].



1	1	1	1	0
1	0	0	0	0
1	0	3	0	0
0	0	0	0	0
2	0	0	0	2

Рисунок 3.3 − а) площина з перешкодами і СП б) площина у вигляді матриці

Потім матриця розбивається на дві множини: множину B, що складається з розташованих підряд осередків без перешкод, та допоміжну множину C, яка дозволяє відновити з B оригінальну матрицю (3.3). Поділ здійснюється наступним чином: проводиться послідовний обхід матриці A, під час якого для кожної комірки, що не є перешкодою, в множину C записуються її координати (індекси) по осях матриці. У множину B записується числове значення комірки, яке може бути нулем або номером сусіднього вузла. Розмір множини B позначається як N_i.

$$A = \begin{Vmatrix} 1\,1\,1\,1\,0 \\ 1\,0\,0\,0\,0 \\ 1\,0\,3\,0\,0 \\ 0\,0\,0\,0\,0 \\ 2\,0\,0\,0\,2 \end{Vmatrix} \Rightarrow \begin{matrix} C = [15\ 22\ 23\ 24\ 25\ 32\ \ldots] \\ B = [\underbrace{0\ 0\ \ldots\ 0\ 2}_{N_i}] \end{matrix} \tag{3.3}$$

Множина B містить усі позиції, де можливо розмістити сенсорні пристрої. Для кодування індексу СП у цій множині потрібно витратити N_b біт, де $N_b = \log_2 N_i$. Закодований у вигляді двійкового рядка індекс виступає як один ген хромосоми.

Хромосома являє собою послідовність індексів у бітовому коді, що описує розташування набору СП в множині B. Через принцип роботи генетичного алгоритму, розмір усіх хромосом повинен бути однаковим, тому за один прохід алгоритму може розглядатися лише фіксована кількість СП.

Фітнес-функція F реалізована наступним чином: оскільки розв’язується

задача покриття площини, необхідно відновити її. Гени з хромосоми перевіряються як позиції СП у множині B, після чого відновлюється матриця A. На основі таблиці характеристик визначаються зони покриття датчиків. Потім обчислюється сумарна площа покриття датчиками, і це значення виступає як оцінка фітнес-функції [80].

Розрахунок площі покриття відбувається наступним чином. Для рішення, записаного в хромосомі Ch_i, формується двовимірний булевий масив покриття X, розмір якого відповідає площині, тому індексація його аналогічна. Далі здійснюється обхід по датчикам: для кожного датчика, залежно від його радіуса, визначаються чотири межі на площині в різні боки (якщо отримана межа виходить за границі матриці, вона обмежується крайніми осередками матриці). З отриманими межами проводиться обхід квадратної області. Далі перевіряється, чи задовольняє осередок умові, що визначається простим рівнянням кола:

$$(x-a)^2+(y-b)^2 \le R^2, \tag{3.4}$$

де a, b − координати сенсорного пристрою;

x, y − координати осередку в обхідній області;

R − радіус дії СП.

Якщо осередок задовольняє цій умові, у X з аналогічним індексом записується значення $true$. Таким чином, створюються «зони покриття» навколо кожного датчика (рис. 3.4) [81].

Значення фітнес-функції F можна обчислити, підсумувавши кількість осередків масиву X, які покриті принаймні одним сенсором. Іншими словами, F дорівнює кількості осередків із значенням $true$ в масиві X, що позначають зони покриття датчиками.

$$F(Ch_i)=\sum_{i=0}^{N} x_i, x_i \in (0;1). \tag{3.5}$$

На етапі оцінки використовується механізм збереження найкращого рішення. Це означає, що після оцінювання популяції найкращий кандидат

поточного покоління порівнюється з попередньо запам'ятованим найкращим рішенням.

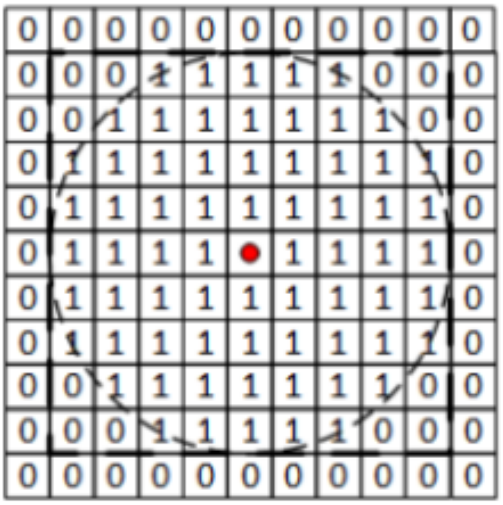


Рисунок 3.4 − Створення масиву покриття для площини

Якщо нове рішення виявляється кращим, воно замінює найгіршу хромосому в поточному поколінні. Якщо ж найкращим залишається попередній кандидат, він оновлює значення найкращого рішення [82].

Відбір здійснюється таким чином: для кожної хромосоми визначається ймовірність p_i:

$$p_i = \frac{F(Ch_i)}{\sum_{i=0}^{N} F(Ch_i)}. \quad (3.6)$$

Сума ймовірностей p_i всіх хромосом дорівнює одиниці. Потім на основі цих ймовірностей відбирається набір хромосом, кількість яких дорівнює розміру популяції. Таким чином, хромосоми з вищим значенням p_i мають більшу ймовірність потрапити до відбору для рекомбінації.

Рекомбінування в алгоритмі реалізовано стандартними методами: випадкові пари хромосом, які пройшли відбір, піддаються одноточковому схрещуванню [83]. У процесі схрещування ділянки хромосом побітно змінюються в межах випадково вибраних сегментів, відповідно до ймовірності схрещування p_{cr}, де $p_{cr} \geq 0{,}5$. Далі для кожної хромосоми з ймовірністю мутації $p_{mt} = 0{,}1$ один з її генів випадковим чином змінюється.

Зважаючи на те, що гени, залежно від вхідних параметрів, можуть мати

заборонені значення, кожен ген у кожній отриманій хромосомі перевіряється на можливу помилку. Якщо в гені виявляється помилка, він замінюється випадково вибраним геном з іншої, «здорової» хромосоми.

В алгоритмі передбачено дві умови завершення:

- досягнуто максимальну кількість ітерацій, і оптимальним рішенням розміщення вважається найкращий кандидат;
- виконано повне покриття, і знайдене рішення не можна поліпшити.

Таким чином, алгоритм поступово оптимізує рішення, поки не отримає остаточний результат, який неможливо покращити (рис. 3.5) [84].

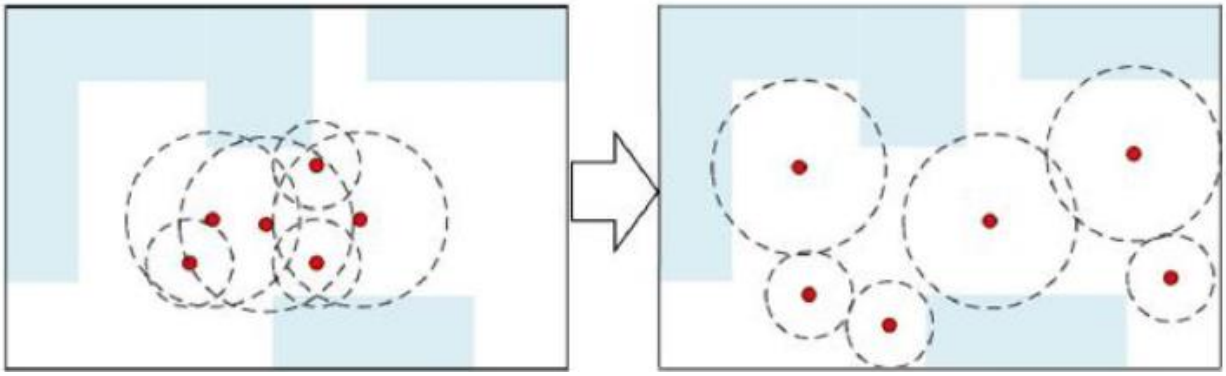


Рисунок 3.5 − Результат роботи алгоритму

Цей алгоритм можна реалізувати й для тривимірного простору. У цьому випадку для опису території використовуються тривимірні масиви A і X. Крім того, щоб визначити, чи знаходиться осередок у межах радіусу дії сенсорного пристрою під час створення масиву X, використовується рівняння сфери:

$$(x-a)^2+(y-b)^2+(z-c)^2 \le R^2, \qquad (3.7)$$

де a, b, c − координати сенсорного пристрою;

x, y, z − координати осередку в обхідній області;

R − радіус дії СП.

Таким чином, кожен сенсор покриває сферичну область у тривимірному просторі, дозволяючи застосовувати алгоритм для складніших структур. Розмірність множин B і C для тривимірного простору не зміниться.

3.3 Генетичний алгоритм пошуку альтернативних маршрутів у мережі IoT

Маршрутизаційна стратегія визначає загальний підхід до вибору шляху для передачі даних, спираючись на різні критерії, такі як відстань, затримка, споживання енергії тощо. Вибір стратегії залежить від специфічних вимог і характеристик системи мереж. Існують різноманітні класифікації стратегій маршрутизації, які забезпечують оптимальну роботу мережі в різних умовах і для різних застосувань (рис. 3.6) [85-87].

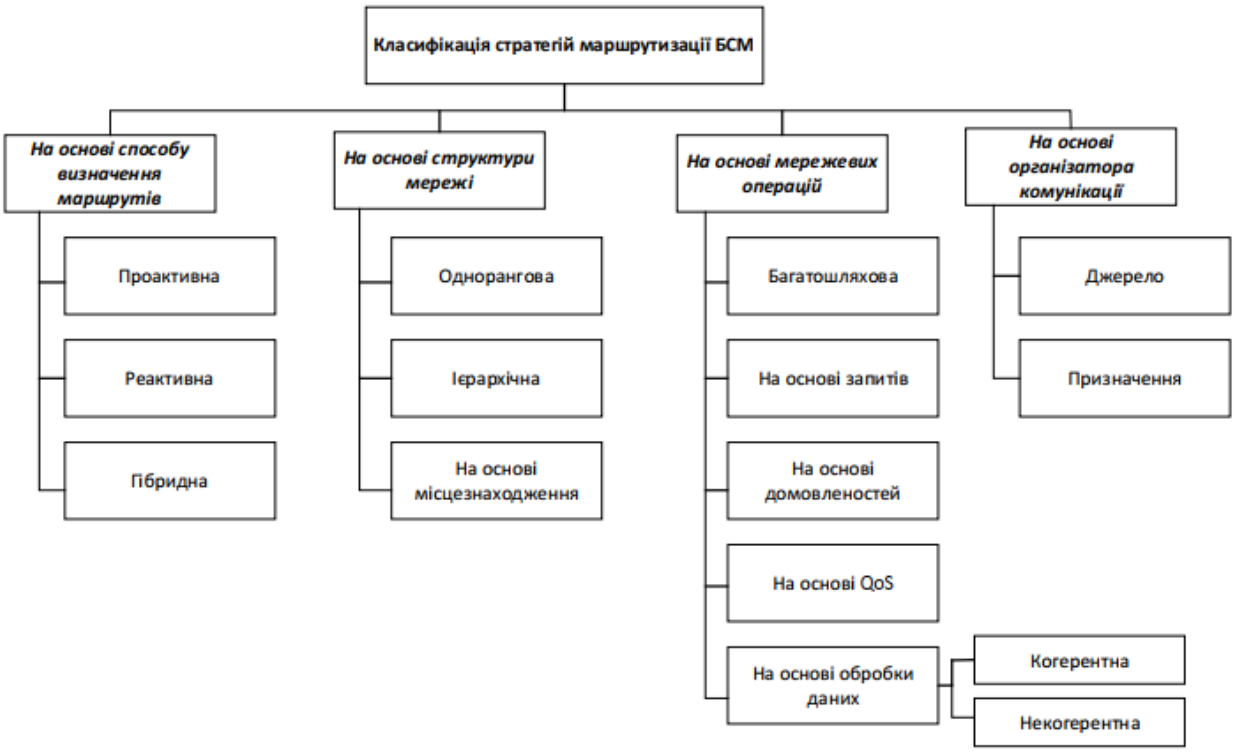


Рисунок 3.6 − Класифікація стратегій маршрутизації даних БСМ

Багатошляхова маршрутизація, яка включає генетичний алгоритм, передбачає використання кількох маршрутів для передачі даних, що підвищує ефективність роботи мережі за рахунок її надійності та стійкості до відмов. Якщо один із маршрутів стає недоступним, дані перенаправляються альтернативними шляхами. Для забезпечення коректної роботи цієї стратегії мережеві вузли постійно відстежують стан мережі та доступність маршрутів,

збираючи інформацію про затримки, перешкоди тощо. На основі цих даних визначається оптимальний маршрут, а у випадку змін у стані мережі або недоступності певного маршруту відбувається динамічне оновлення маршрутів.

Переваги багатошляхової маршрутизації:

- надійність передачі даних: завдяки наявності альтернативних маршрутів забезпечується стійкість до збоїв у мережі;
- адаптивність − мережа швидко реагує на зміни, як-от вихід з ладу або додавання нових вузлів, перевантаження тощо.

Недоліки багатошляхової маршрутизації:

- складність управління − для підтримки множини маршрутів необхідні більш складні алгоритми;
- високі вимоги до ресурсів − підтримка інформації про кілька маршрутів потребує більших обчислювальних потужностей, включно з розширеними таблицями маршрутизації.

Багатошляхова маршрутизація вважається ефективною стратегією для мереж, де надійність є ключовим параметром.

Узагалі, вибір стратегії маршрутизації відображає специфічні вимоги та виклики бездротових сенсорних мереж, зокрема їх обмежені ресурси, динамічну топологію та вимоги до якості обслуговування. Від правильного вибору стратегії залежить стабільна та ефективна робота мережі.

Для налаштування роботи генетичного алгоритму використовуються чотири основні параметри:

1. Розмір популяції (N) − це кількість особин у популяції, кожна з яких представляється хромосомою. Цей параметр визначає, скільки варіантів рішення одночасно аналізує ГА. Розмір популяції залежить від складності задачі та доступних обчислювальних ресурсів і встановлюється експериментально. При цьому збільшення кількості особин дозволяє краще досліджувати простір пошуку, підвищуючи ймовірність знаходження глобального оптимуму.

2. Кількість поколінь (G) − це кількість ітерацій, протягом яких відбувається еволюція популяції. Кількість поколінь визначається складністю задачі, обсягом вхідних даних, доступними ресурсами та бажаною точністю рішення. При цьому більша кількість поколінь дає алгоритму більше можливостей для покращення рішень.

3. Імовірність схрещування (P_{cross}) − визначає частоту обміну генетичним матеріалом між хромосомами, значення знаходиться у діапазоні від 0.0 до 1.0. Якщо $P_{cross} = 0.0$, нове покоління формується шляхом копіювання існуючих особин, без схрещування (за винятком мутацій). При цьому схрещування сприяє утворенню нових комбінацій генів, що дозволяє досліджувати більше можливих рішень.

4. Імовірність мутації (P_{mut}) − визначає частку хромосом, які зазнають мутації в одному поколінні, також знаходиться в діапазоні від 0.0 до 1.0. Мутація забезпечує різноманітність у популяції, запобігаючи передчасному зближенню алгоритму до локального оптимуму.

Ці параметри мають бути налаштовані відповідно до особливостей задачі, оскільки вони суттєво впливають на ефективність і швидкість роботи алгоритму.

Для створення ефективного генетичного алгоритму, адаптованого до специфіки роботи БСМ, було обрано відповідні генетичні оператори та розроблено функцію оцінки пристосованості. Ця функція дозволяє оцінювати якість кожної особини, яка в даному випадку представляє маршрут передачі даних.

Блок-схема ГА, призначена для вирішення задачі маршрутизації даних у БСМ (рис. 3.7), включає всі ключові етапи:

- ініціалізацію популяції (маршрутів);
- оцінювання пристосованості кожної особини;
- застосування генетичних операторів для формування нового покоління;
- перевірку умов завершення (наприклад, досягнення заданої кількості

поколінь або знаходження оптимального рішення).

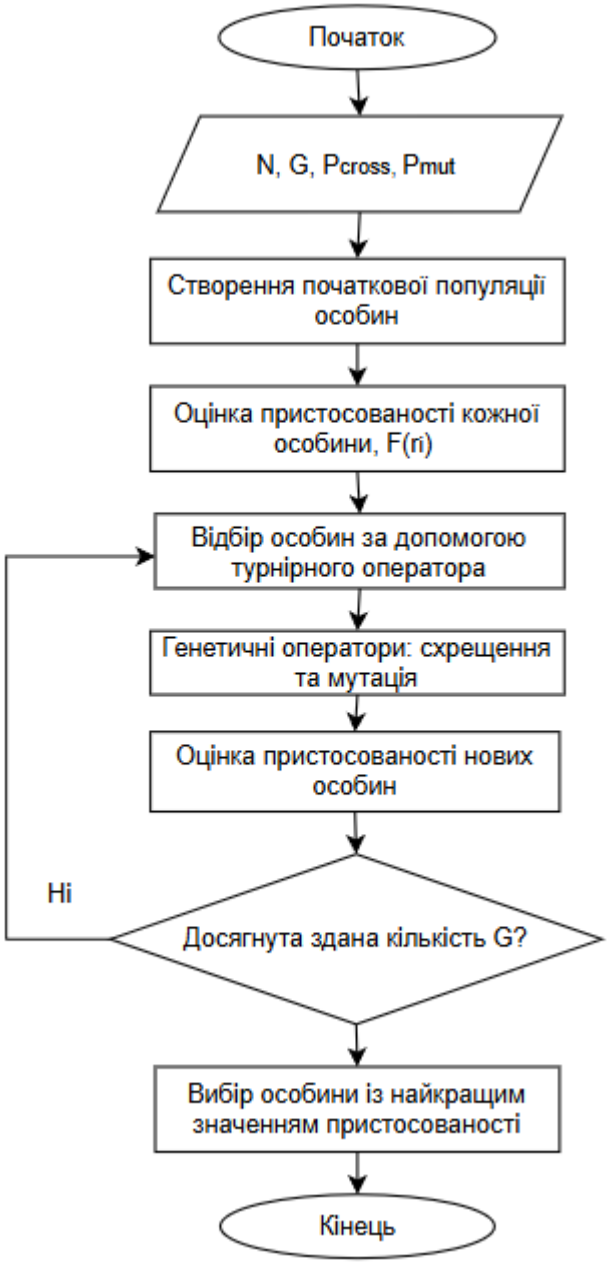


Рисунок 3.7 − Блок-схема сформованого ГА

$$F(r_i) = \frac{1}{\sum_{j=1}^{l_i-1} r_i(j,j+1)}, \tag{3.8}$$

де $F(r_i)$ − функція пристосованості особини r_i;

l_i − довжина маршруту (кількість точок у ньому);

$r_i(j, j+1)$ − відстань між послідовними вузлами маршруту j та $j+1$.

Таке формулювання дозволяє ефективно адаптувати ГА до потреб БСМ,

забезпечуючи баланс між продуктивністю та точністю.

Для побудови ГА було використано генетичні оператори турнірного відбору, впорядкованого схрещування та мутації перемішування.

Генетичний оператор турнірного відбору базується на принципі «змагання» між випадково вибраними особинами. Розмір турніру визначається кількістю особин, які беруть участь у відборі. З випадково обраних особин перемогу здобуває та, яка має найвищу пристосованість. Саме вона переходить до наступного покоління.

Цей метод забезпечує баланс між випадковістю та перевагою кращих особин, сприяючи еволюції популяції (рис. 3.8).

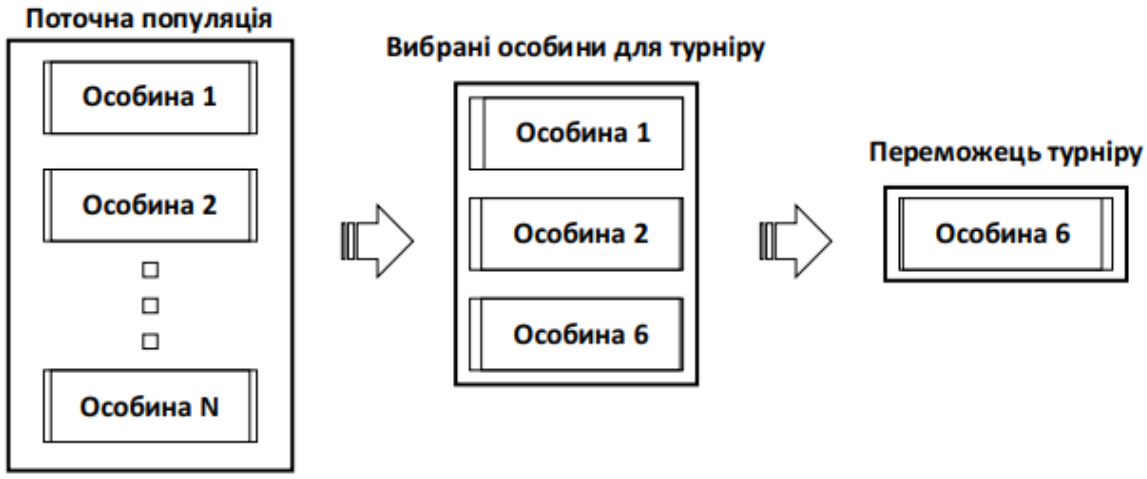


Рисунок 3.8 − Приклад роботи турнірного оператора відбору

На практиці перевага віддається невеликому розміру турніру, оскільки це дозволяє зберігати різноманітність у популяції. Параметр (розмір турніру) k може бути налаштований для досягнення оптимального балансу між різноманітністю і поліпшенням існуючих рішень. Зазвичай розмір турніру встановлюється на рівні 2 або 3 [88].

Оператор впорядкованого схрещування полягає в збереженні порядку генів у батьківських особинах, що є важливим для задач, де порядок генів критичний для правильного розв’язку.

Механізм впорядкованого схрещування:

- у батьківських особинах визначаються одна або кілька точок схрещування;

- генетичні послідовності розділяються на частини по точках схрещування;

- частини обмінюються між батьками, в результаті чого формується нове покоління, при цьому порядок генів у кожній послідовності залишається збереженим.

Першим етапом двоточкового схрещування є вибір двох випадкових точок розрізу, які розділяють генетичну послідовність батьківських хромосом на три частини. Гени, розташовані між цими точками, без змін передаються у відповідні позиції хромосом нащадків. Далі здійснюється заповнення решти позицій у хромосомах нащадків шляхом обходу генів батьківських хромосом у їх первинному порядку, починаючи з позиції, що слідує після другої точки розрізу. Якщо ген у поточній позиції ще не входить до хромосоми нащадка, його додають. У разі, якщо ген вже присутній, його пропускають і переходять до наступного в послідовності. Наприклад, у першого батька після другої точки розрізу розташований ген 7, який відсутній у нащадку, тому його додають. Наступний ген 8 вже присутній у нащадку, отже, його пропускають. Після цього додають ген 2, оскільки він ще не входить до хромосоми нащадка. Аналогічно заповнюються всі інші вільні позиції. У результаті формуються нащадки з унікальними генетичними послідовностями, які поєднують ознаки обох батьків (рис. 3.9) [89-90].

Цей метод є особливо корисним у випадках, де важливість має не лише зміст генів, але й їх послідовність.

Оператор мутації перемішування випадковим чином змінює порядок генів у вибраній послідовності, що дозволяє збільшити різноманітність популяції (рис. 3.10). При цьому пристосованість кожної особини визначається обернено пропорційно вазі маршруту. Під вагою маршруту можна розуміти будь-який показник, що характеризує його якість, наприклад, відстань, енергетичний запас вузлів або відношення сигнал/шум.

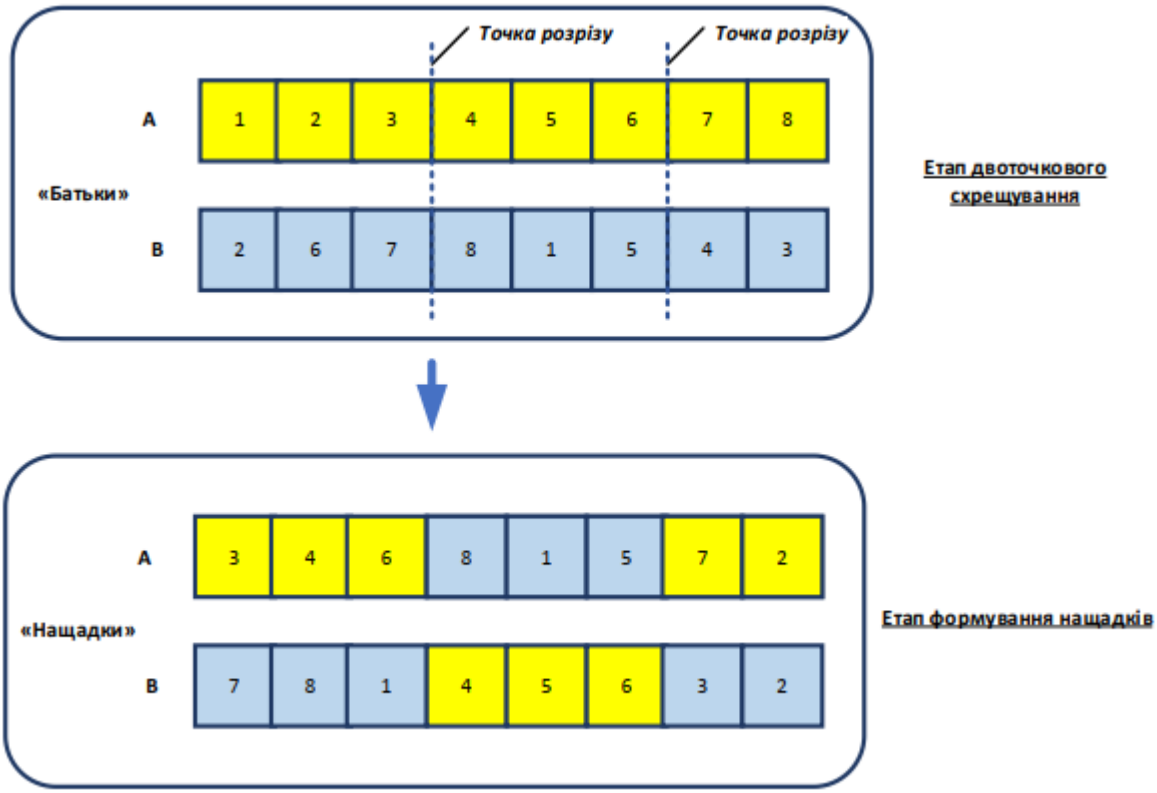


Рисунок 3.9 − Приклад роботи впорядкованого оператора схрещування

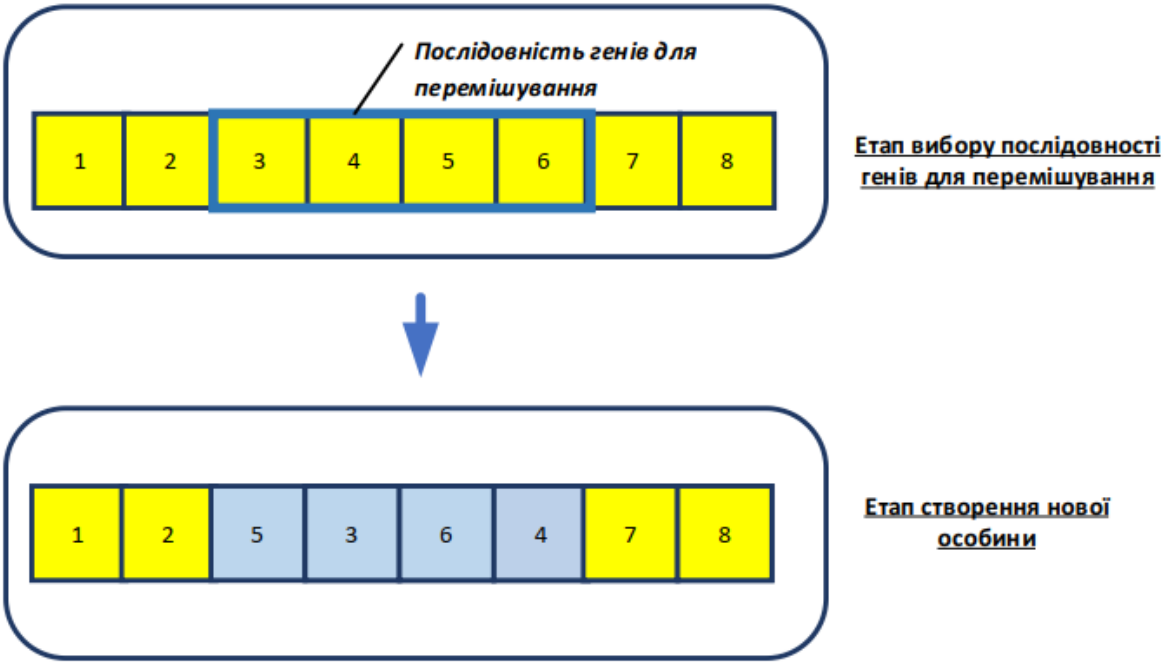


Рисунок 3.10 − Приклад роботи оператора перемішування

Таким чином, генетичний алгоритм належить до стратегії багатошляхової маршрутизації, оскільки його використання дозволяє

формувати множину маршрутів для передачі даних. Завдяки цьому, навіть у випадку неможливості передачі по найкращому маршруту, з множини доступних маршрутів можна вибрати альтернативне рішення, що особливо важливо у разі виходу з ладу основного маршруту.

Для перевірки працездатності розробленого генетичного алгоритму було виконано моделювання з використанням створеного програмного продукту на мові Python із застосуванням бібліотеки DEAP. Для цього були задані такі параметри: розмір популяції $N = 300; 400; 500; 600$; кількість поколінь $G = 50, \dots, 600$ із кроком 50; ймовірність схрещування $P_{cross} = 0.8$ та ймовірність мутації $P_{mut} = 0.2$. Значення P_{cross} і P_{mut} було вибрано на основі аналізу відповідних параметрів у літературі, тоді як розмір популяції та кількість поколінь підбиралися експериментально [91-92].

Налаштування параметрів ГА здійснювалося з урахуванням особливих вимог сенсорної мережі. Наприклад, для мереж моніторингу в складних умовах можуть використовуватися параметри, що сприяють мінімізації енергоспоживання та підвищенню надійності. Це налаштування є важливим для досягнення ефективної маршрутизації даних відповідно до поточних потреб.

У моделюванні використано мережу із 25 вузлів, розміщених випадковим чином у вільному просторі на ділянці розміром 100×100 м (рис. 3.11). Радіус дії кожного вузла становить 30 м. У разі, якщо відстань між вузлами перевищує цей радіус, передача даних неможлива. Для таких випадків вводиться штраф на відстань у 1000 м, що спонукає ГА шукати більш короткі маршрути.

На основі використання ГА було побудовано матрицю вузлів мережі, яка відображає топологію та взаємозв’язки між вузлами (рис. 3.12), визначено маршрут передачі даних від 4-го до 20-го вузла (рис. 3.13). Отримані результати подано таблично (табл. 3.2). Часова складність ГА при пошуку маршруту між зазначеними вузлами представлена графічно (рис. 3.14). Відповідний математичний апарат описано у [93].

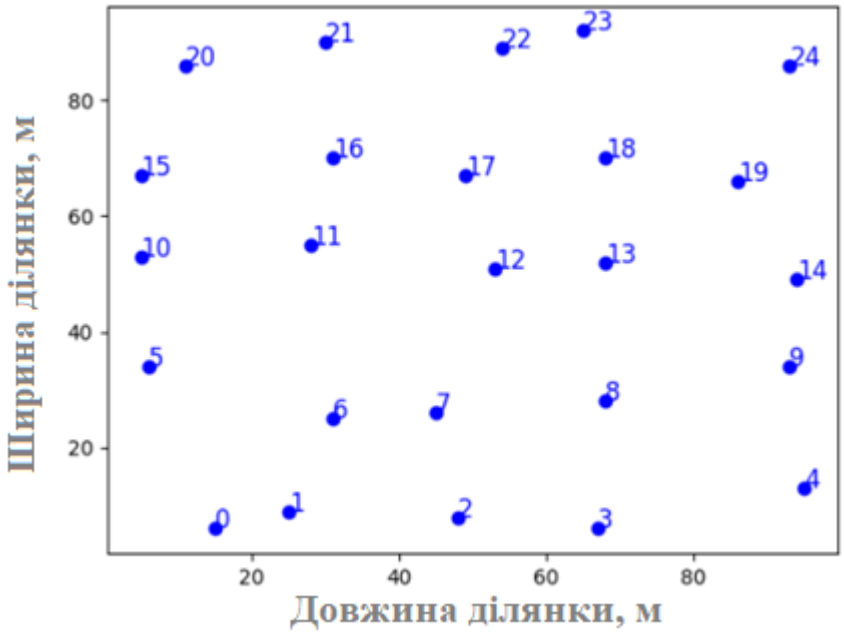


Рисунок 3.11 − Топологія досліджуваної мережі

Distance Matrix

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	10	1000	1000	1000	29	24	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
2	10	0	23	1000	1000	1000	17	26	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
3	1000	23	0	19	1000	1000	24	18	28	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
4	1000	1000	19	0	28	1000	1000	29	22	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
5	1000	1000	1000	28	0	1000	1000	1000	30	21	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
6	29	1000	1000	1000	1000	0	26	1000	1000	1000	19	30	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
7	24	17	24	1000	1000	26	0	14	1000	1000	1000	30	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
8	1000	26	18	29	1000	1000	14	0	23	1000	1000	1000	26	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
9	1000	1000	28	22	30	1000	1000	23	0	25	1000	1000	27	24	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
10	1000	1000	1000	1000	21	1000	1000	1000	25	0	1000	1000	1000	30	15	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
11	1000	1000	1000	1000	1000	19	1000	1000	1000	1000	0	23	1000	1000	1000	14	1000	1000	1000	1000	1000	1000	1000	1000	1000
12	1000	1000	1000	1000	1000	30	30	1000	1000	1000	23	0	25	1000	1000	25	15	24	1000	1000	1000	1000	1000	1000	1000
13	1000	1000	1000	1000	1000	1000	1000	26	27	1000	1000	25	0	15	1000	1000	29	16	24	1000	1000	1000	1000	1000	1000
14	1000	1000	1000	1000	1000	1000	1000	1000	24	30	1000	1000	15	0	26	1000	1000	24	18	22	1000	1000	1000	1000	1000
15	1000	1000	1000	1000	1000	1000	1000	1000	1000	15	1000	1000	1000	26	0	1000	1000	1000	1000	18	1000	1000	1000	1000	1000
16	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	14	25	1000	1000	1000	0	26	1000	1000	1000	19	1000	1000	1000	1000
17	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	15	29	1000	1000	26	0	18	1000	1000	25	20	29	1000	1000
18	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	24	16	24	1000	1000	18	0	19	1000	1000	29	22	29	1000
19	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	24	18	1000	1000	1000	19	0	18	1000	1000	23	22	29
20	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	22	18	1000	1000	1000	18	0	1000	1000	1000	1000	21
21	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	19	25	1000	1000	1000	0	19	1000	1000	1000
22	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	20	29	1000	1000	19	0	24	1000	1000
23	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	29	22	23	1000	1000	24	0	11	1000
24	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	29	22	1000	1000	1000	11	0	28
25	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	29	21	1000	1000	1000	28	0

Рисунок 3.12 − Матриця вузлів для сенсорної мережі із 25 вузлів

Цей підхід дозволяє ефективно оптимізувати маршрутизацію в умовах заданих обмежень та параметрів мережі.

Таблиця 3.2 – Отримані результати пошуку маршруту (відстань /час) між 4 та 20 вузлом при k=3

Кількість поколінь	Розмір популяції			
	300	400	500	600
50	1000/7.674	1000/9.769	1000/11.716	1000/14.106
100	1000/13.775	1000/17.990	121/22.007	1000/26.025
150	111/18.477	111/25.339	121/30.317	111/36.875
200	111/24.222	111/32.643	116/40.696	111/48.393
250	111/30.230	111/41.328	116/52.776	111/61.160
300	111/37.335	111/48.937	116/60.752	111/72.612
350	111/42.361	111/56.537	116/70.739	111/85.025
400	111/49.483	111/65.444	116/80.583	111/97.175
450	111/55.017	111/73.323	116/91.222	111/110.051
500	111/62.742	111/81.001	116/100.910	111/12.410
550	111/67.932	111/88.989	116/111.296	111/134.318
600	111/72.875	111/100.076	116/121.602	111/145.762

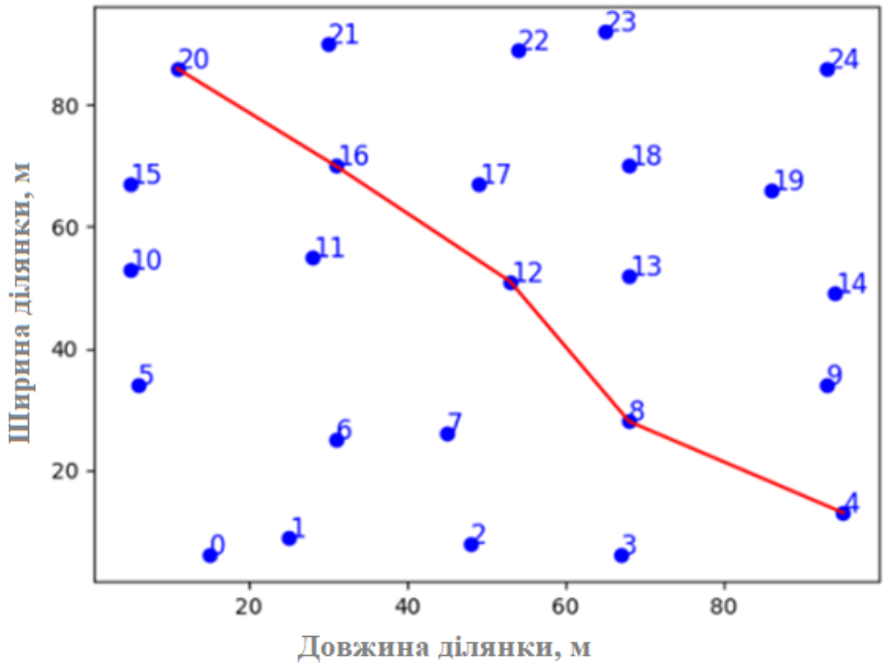


Рисунок 3.13 – Отриманий маршрут найменшої довжини між 4 і 20 вузлом

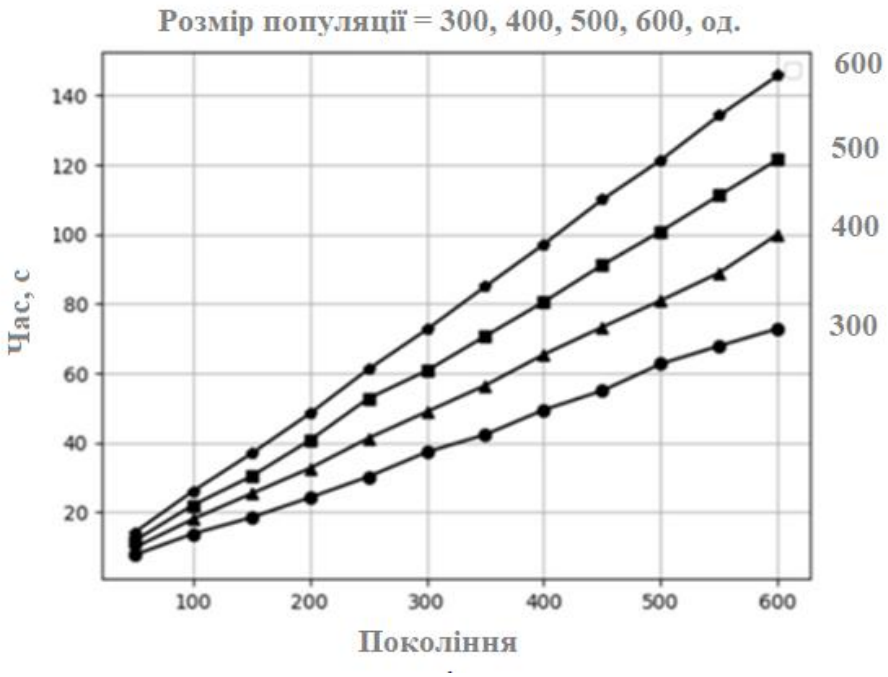


Рисунок 3.14 − Часова складність ГА

Як показує аналіз, зі збільшенням кількості поколінь часова складність алгоритму зростає лінійно. На основі даних (табл. 3.2), найкоротший маршрут між вузлами 4 і 20 проходить через точки [4; 8; 12; 16; 20] із загальною відстанню 111 м, причому час пошуку цього маршруту становить 18,477 с. Зі збільшенням кількості поколінь і розміру популяції тривалість пошуку маршруту також зростає лінійно. У цьому випадку оптимальним рішенням є використання невеликої кількості поколінь (150) і середнього розміру популяції (300), що дозволяє досягти найкращих результатів.

3.4 Висновки до розділу

У третьому розділі «Ймовірнісні алгоритми самоорганізації мережі Інтернету речей» здійснюється аналіз даного класу алгоритмів. У ході дослідження було здійснено оцінювання різних алгоритмів самоорганізації бездротових сенсорних мереж з метою визначення найбільш ефективного для застосування в мережах Інтернету речей. Метод аналізу ієрархій дозволив систематизувати процес вибору, враховуючи судження експертів у вигляді

парних порівнянь.

Порівняльний аналіз дев’яти алгоритмів самоорганізації виявив енергоефективний алгоритм як найкращий. Він забезпечує високу продуктивність завдяки використанню потужності передавання як медіатора в схемі оплати. Результати дослідження підтверджують його оптимальність для самоорганізації мереж IoT.

Також в розділі детально розглянуто та запропоновано для використання в мережах IoT два алгоритми, що базуються на еволюційній теорії.

Генетичний алгоритм формує альтернативні маршрути, враховуючи статистику доставки даних. Канали з ймовірністю передачі нижче порога виключаються, що підвищує надійність мережі, використовуючи лише стабільні зв’язки.

Генетичний алгоритм побудови множини альтернативних маршрутів, які з'єднують джерело і одержувача, заснований на операціях із трьома основними типами об'єктів: маршрут, покоління та мережа. Маршрут являє собою впорядкований список вузлів мережі, де першим вказується початковий вузол, а останнім − кінцевий. Покоління − це масив маршрутів, до яких застосовуються генетичні операції, такі як схрещування і мутація. Розмір покоління є змінним. Мережа моделюється у вигляді графа, описаного двома матрицями: перша містить дані про затримку передачі між вузлами, друга − ймовірності втрат пакетів даних. Матриця ймовірностей ініціалізується випадковими значеннями в діапазоні $[0 \div 1]$, що відображає динамічні властивості мережі Інтернету речей. У процесі роботи алгоритму ймовірності втрат змінюються випадковим чином. Ініціалізація початкового покоління передбачає визначення всіх можливих маршрутів від джерела до одержувача, для яких ймовірності втрат не перевищують заданий поріг. Далі виконується селекція для приведення розміру популяції до заданого значення. Вона реалізується за допомогою методу рулетки. З кожного покоління відбираються маршрути з найменшим прогнозним часом доставки даних, які вважаються «здоровими хромосомами» і залучаються до операції схрещування без

додаткового відбору. Схрещування генерує нові маршрути на основі існуючих із заданою ймовірністю. Мутація передбачає випадкове видалення вузлів із маршруту, причому ймовірність мутації фіксована. Критерієм оцінки (фітнес-функцією) є час доставки даних від джерела до одержувача. На вхід алгоритму подається граф, описаний матрицею інцидентності із значеннями часу затримки між вузлами в мілісекундах. Результатом роботи алгоритму є список альтернативних маршрутів у вигляді впорядкованих списків вузлів.

Другий – це алгоритм самоорганізованого розміщення сенсорних пристроїв, що також базується на еволюційній теорії. Цей підхід дозволяє оптимізувати розташування сенсорних вузлів у мережі, враховуючи вимоги до покриття, енергоефективності та мінімізації втрат даних. Для цього було досліджено стратегії маршрутизації даних у безпровідних сенсорних мережах і запропоновано генетичний алгоритм для оптимального визначення маршруту між вузлами мережі. Для оцінки ефективності цього підходу було створено програмний продукт на мові Python з використанням бібліотеки DEAP, яка забезпечує необхідний функціонал для реалізації генетичних алгоритмів. У рамках експерименту створено сенсорну мережу, що складається з 25 вузлів, кожен із яких має радіус дії 30 метрів. Для мережі розроблено та представлено матрицю вузлів, яка містить детальну інформацію про її топологію та зв’язки між вузлами.

Результати імітаційного моделювання показали, що найкоротший маршрут між вузлами 4 і 20 проходить через точки [4; 8; 12; 16; 20], загальна відстань складає 111 метрів. Оптимальні параметри для алгоритму було досягнуто при 150 поколіннях та розмірі популяції 300. Час пошуку маршруту зростає лінійно зі збільшенням кількості поколінь і чисельності популяції. Це підтверджує ефективність генетичного підходу для задач маршрутизації навіть у складних мережах.

Запропоновані алгоритми сприяють підвищенню продуктивності та адаптивності мережі Інтернету речей, забезпечуючи її стабільне функціонування в динамічних умовах.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ВЗАЄМОДІЇ В МЕРЕЖАХ ІНТЕРНЕТУ РЕЧЕЙ

4.1 Аналіз результатів моделювання доступу у туманних обчисленнях

4.1.1 Опис моделі туманного обчислення

Модель туманного обчислення в рамках аналізу часу отримання доступу до центрального вузла сенсорними пристроями базується на трьох ключових компонентах: сенсорні пристрої, туманні вузли (сервери) та оркестратор (балансувальник) [94].

Сенсорний пристрій – це кінцевий об'єкт IoT-мережі, який генерує дані й передає їх до туманного вузла для обробки. Основні характеристики СП:

- частота генерації даних (τ): визначає, з якою періодичністю СП відправляє пакети даних (в секундах);

- розмір пакета (σ): обсяг даних, що передається одним пакетом (у кілобайтах);

- з’єднання: СП можуть встановлювати зв’язок із туманним вузлом через доступний канал зв’язку.

СП в моделі генерують дані з постійним або змінним інтервалом часу. Після формування пакета, пристрій перевіряє наявність з’єднання з туманним вузлом. У разі відсутності з’єднання пакет ставиться в чергу.

Туманний вузол виконує роль сервера, що обробляє отримані дані від підключених СП. Туманний вузол приймає пакети даних, обробляє їх із врахуванням своєї потужності та передає результати далі у мережу або безпосередньо до кінцевого споживача.

Основні параметри туманного вузла:

- потужність (P): визначає максимальну кількість даних, яку сервер може обробити за одиницю часу (у кілобайтах на секунду);

- навантаження (I): обчислюється як співвідношення обсягу оброблених даних до потужності сервера;

- черга: якщо потужність вузла перевищена, надлишкові пакети даних ставляться в чергу на обробку, що призводить до затримок.

Оркестратор − це механізм розподілу навантаження, що забезпечує ефективну роботу всієї мережі.

Функції оркестратора:

- моніторинг: відслідковує навантаження кожного туманного вузла в реальному часі;

- перерозподіл: за необхідності переадресовує СП до менш завантажених вузлів для забезпечення рівномірного розподілу ресурсів.

Процес роботи оркестратора:

- визначає поточне навантаження всіх вузлів за допомогою формули;

- якщо навантаження на певний вузол перевищує допустимий поріг, оркестратор перенаправляє нові з'єднання до менш завантажених вузлів;

- у разі критичного перевантаження вузла, оркестратор тимчасово блокує нові запити до цього вузла.

Взаємодія між компонентами моделі здійснюється за такими етапами:

- запит з'єднання: СП ініціює запит до найближчого туманного вузла. У разі перевантаження вузла оркестратор перенаправляє запит;

- передача даних: після встановлення з'єднання СП відправляє пакет даних (σ) до туманного вузла;

- обробка даних: туманний вузол обробляє отримані пакети з урахуванням своєї потужності (P). У разі перевищення потужності пакети ставляться в чергу;

- аналіз навантаження: оркестратор постійно аналізує завантаженість вузлів і при необхідності перенаправляє трафік.

Для спрощення моделювання передбачається наступне: усі СП мають однаковий розмір пакета (σ) та частоту генерації (τ); потужність усіх туманних вузлів є постійною; оркестратор працює без затримок у передачі даних.

Модель дозволяє оцінити середній час доступу до туманного вузла для різних режимів доступу і залежність цього часу від навантаження.

4.1.2 Оцінка часу відповіді туманного вузла на запити сенсорних пристроїв у різних режимах доступу

Експеримент проводиться з метою оцінки часу відповіді туманного вузла на запити сенсорних пристроїв у різних режимах доступу: опитування, переривання та множинний доступ. Параметри експерименту базуються на спрощеній моделі туманного обчислення, представленій у попередньому підрозділі. Оркестратор у даному випадку не враховується.

Модель експерименту:

- туманний вузол: сервер із потужністю обробки 250 кілобайтів за секунду;
- сенсорні пристрої: кожен пристрій генерує пакети фіксованого розміру 50 кілобайтів.

Режими доступу взаємодії між СП і сервером моделюються окремо:

- опитування: сервер послідовно опитує СП, обробляючи їхні запити в порядку черги;
- переривання: СП ініціюють передачу даних на сервер, і кожен запит обробляється негайно після надходження;
- множинний доступ: всі СП передають запити одночасно, і сервер обробляє їх у порядку надходження, залежно від своєї потужності.

Параметри і змінні експерименту:

- кількість СП (N): варіюється від 1 до 50;
- розмір пакета (σ): 50 кілобайтів;
- потужність сервера (P): 250 кілобайтів на секунду;
- час відповіді (t): час, необхідний для обробки запиту від кожного СП, включаючи можливу затримку через черги.

На початковому етапі на сервер підключається певна кількість СП, яка варіюється від 1 до 50. Кожен пристрій генерує один пакет даних. Далі моделюється процес передачі та обробки даних між СП і сервером для кожного з режимів доступу. Після цього розраховується загальний час

відповіді для кожного СП, включаючи середнє та максимальне значення. Отримані значення часу відповіді для кожної кількості СП і режиму доступу фіксуються, зберігаються у вигляді таблиці та використовуються для побудови графіка залежності часу відповіді від кількості СП.

Очікувані особливості поведінки:

- опитування: зростання часу відповіді t пропорційно збільшенню кількості СП, оскільки сервер опрацьовує запити послідовно;

- переривання: зростання t має бути менш вираженим, оскільки запити обробляються у момент надходження, але можливе утворення черги при великій кількості СП;

- множинний доступ: початково t має залишатися стабільним, але при перевищенні потужності сервера ($N > 5$) час відповіді різко збільшується через накопичення черг [95].

Нижче наведено графіки функції розподілу проміжків часу початку відповіді СП на опитування центрального вузла для доступу в режимі опитування (рис. 4.1), режимі переривання (рис. 4.2) та режимі множинного доступу (рис. 4.3).

Рисунок 4.1 – Графік функції часу відповіді для доступу в режимі опитування

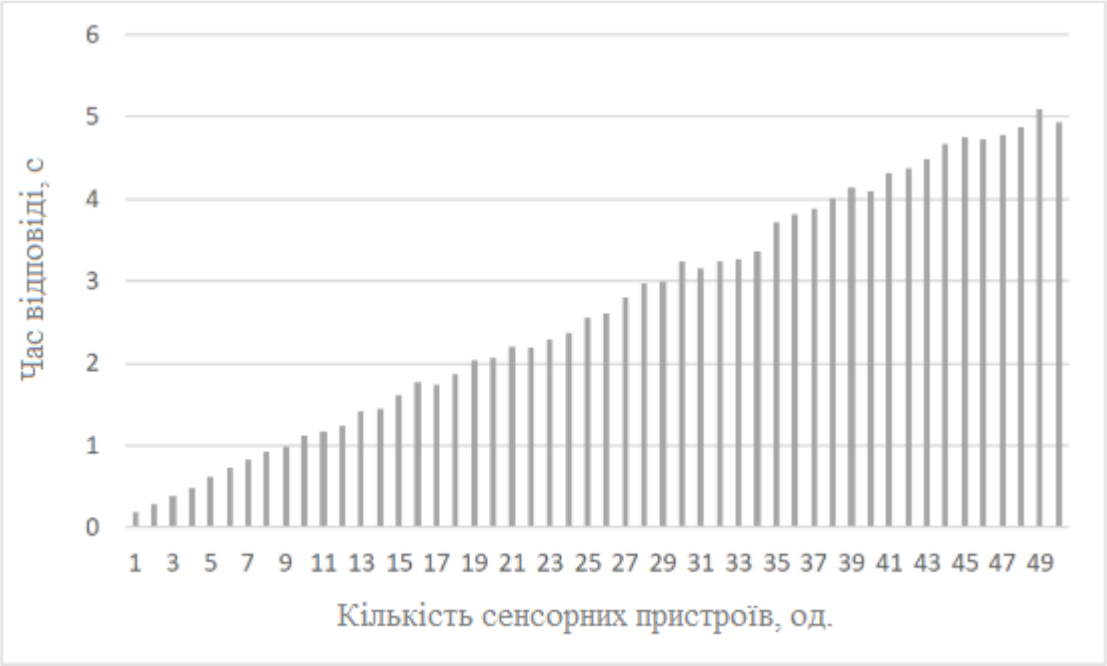


Рисунок 4.2 – Графік функції часу відповіді для доступу в режимі переривання

Рисунок 4.3 – Графік функції часу відповіді для доступу в режимі множинного доступу

Нижче наведено графік порівняння функцій часу для різних режимів доступу (рис. 4.4).

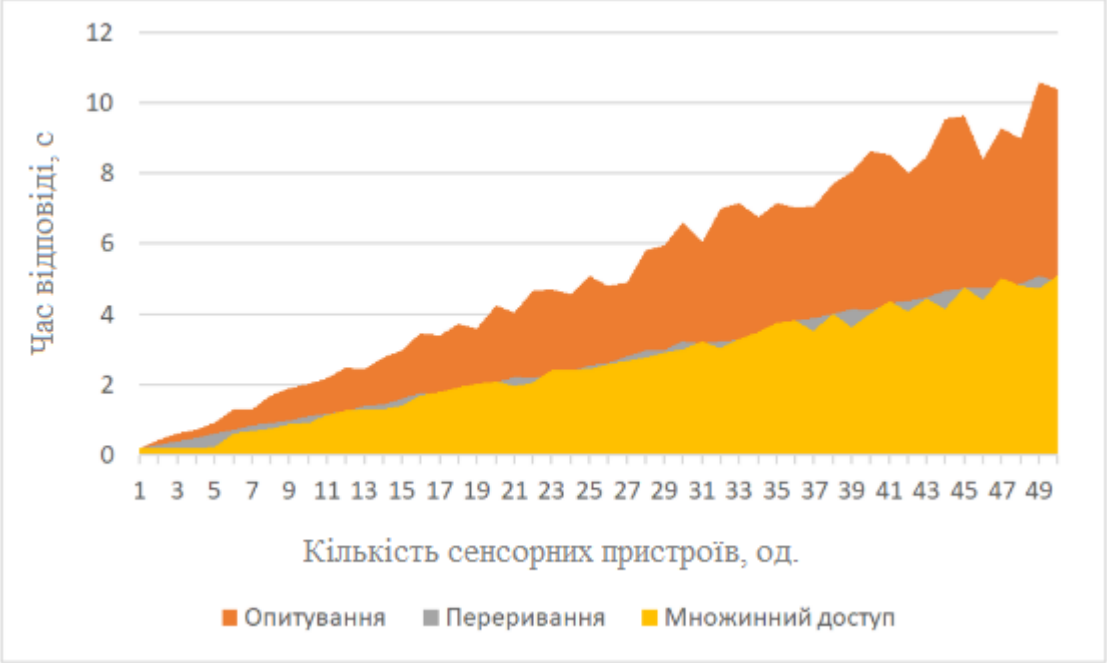


Рисунок 4.4 – Графік порівняння функцій часу відповіді для різних режимів доступу

Висновки за результатами експерименту:

- режим опитування: час відповіді в цьому режимі показує лінійне зростання залежно від кількості підключених пристроїв. Це обумовлено тим, що сервер послідовно опрацьовує кожен запит, що спричиняє затримки при великій кількості СП. Цей режим є найменш ефективним за умов високого навантаження;

- режим переривання: час відповіді в режимі переривання є меншим, ніж у режимі опитування, оскільки запити обробляються негайно після їх надходження. Проте, при значній кількості пристроїв можуть утворюватися черги, що зумовлює поступове зростання часу відповіді;

- режим множинного доступу: у цьому режимі сервер початково демонструє стабільний час відповіді незалежно від кількості СП. Однак, коли кількість пристроїв перевищує потужність сервера, час відповіді різко зростає через перенавантаження та накопичення черг. Цей режим є найбільш ефективним за умов помірного навантаження.

Кожен із розглянутих режимів має свої переваги та недоліки, залежно

від кількості підключених пристроїв. Для мереж із низьким і середнім навантаженням найбільш оптимальним є режим множинного доступу. У випадках високого навантаження доцільно використовувати режим переривання для зниження затримок. Режим опитування слід розглядати як варіант для невеликих мереж із прогнозованою кількістю пристроїв.

4.1.3 Оцінка залежності кількості підключених пристроїв від навантаження вузла

Мета експерименту – вивчення залежності кількості підключених пристроїв до туманного вузла від його навантаження за умови динамічного управління мережею оркестратором. У цьому дослідженні моделюється сценарій, за якого оркестратор поступово відключає сенсорні пристрої від вузла у разі перевищення допустимого рівня навантаження, забезпечуючи стабільну роботу системи.

Вхідні параметри експерименту:

- потужність туманного вузла: 800 кілобайтів на секунду;
- розмір пакета даних: варіюється від 5 кілобайтів до 5000 кілобайтів;
- початкова кількість підключених пристроїв: 25 пристроїв;
- таймаут очікування відповіді для пристроїв: 5 секунд.

Алгоритм оркестратора. Оркестратор постійно відслідковує навантаження вузла, яке залежить від обсягу даних, що передаються пристроями, та його потужності. У разі перевищення допустимого рівня навантаження, оркестратор поступово відключає частину пристроїв, знижуючи навантаження до прийнятного рівня [96].

Методика проведення експерименту:

- всі 25 пристроїв підключені до вузла, кожен пристрій генерує дані з однаковим розміром пакета;
- поступово збільшується розмір пакета даних, що імітує зростання обсягу інформації, яку генерують пристрої;

- оркестратор контролює рівень навантаження на вузол. У разі перевищення допустимого значення, частина пристроїв автоматично відключається для стабілізації роботи вузла;

- визначається кількість активних підключень до вузла для кожного розміру пакета. Дані фіксуються та використовуються для побудови графіка залежності кількості активних пристроїв від розміру пакета.

Очікувані результати:

- за малих розмірів пакета всі пристрої залишаються підключеними до вузла;

- зі збільшенням розміру пакета навантаження на вузол зростає, що призводить до поступового відключення пристроїв оркестратором;

- кількість активних пристроїв зменшується з підвищенням обсягу даних, досягаючи мінімального значення при найбільшому розмірі пакета.

Нижче наведено графіки залежності кількості підключених пристроїв від навантаження на вузол для різних режиму доступу (рис. 4.5 −4.7).

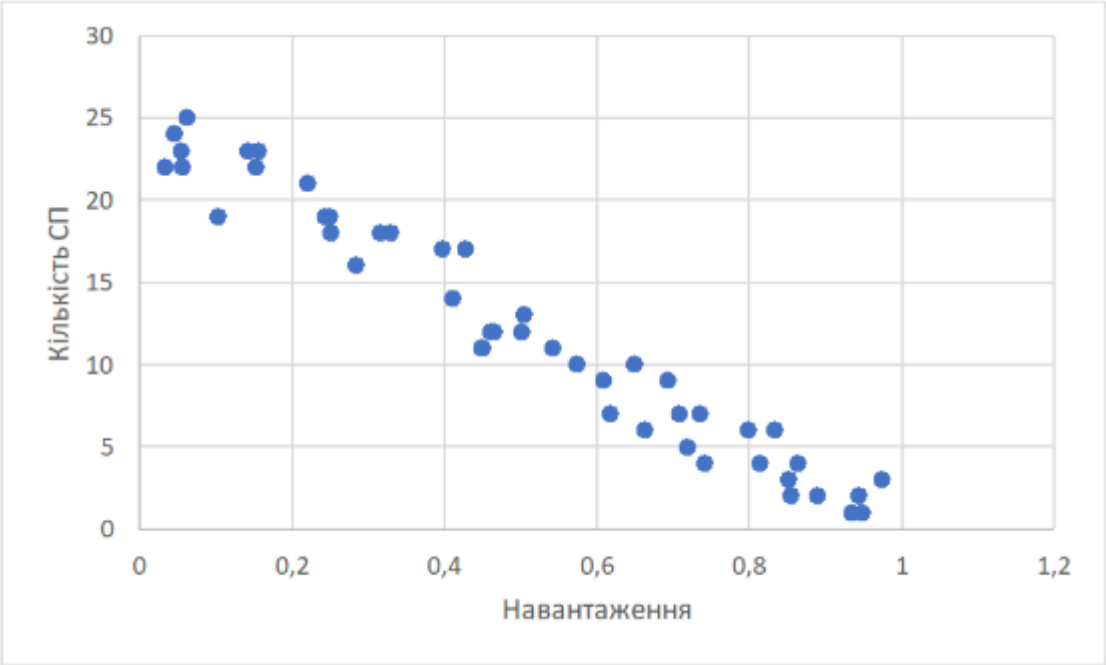


Рисунок 4.5 − Графік залежності кількості підключених пристроїв від навантаження на вузол для доступу в режимі опитування

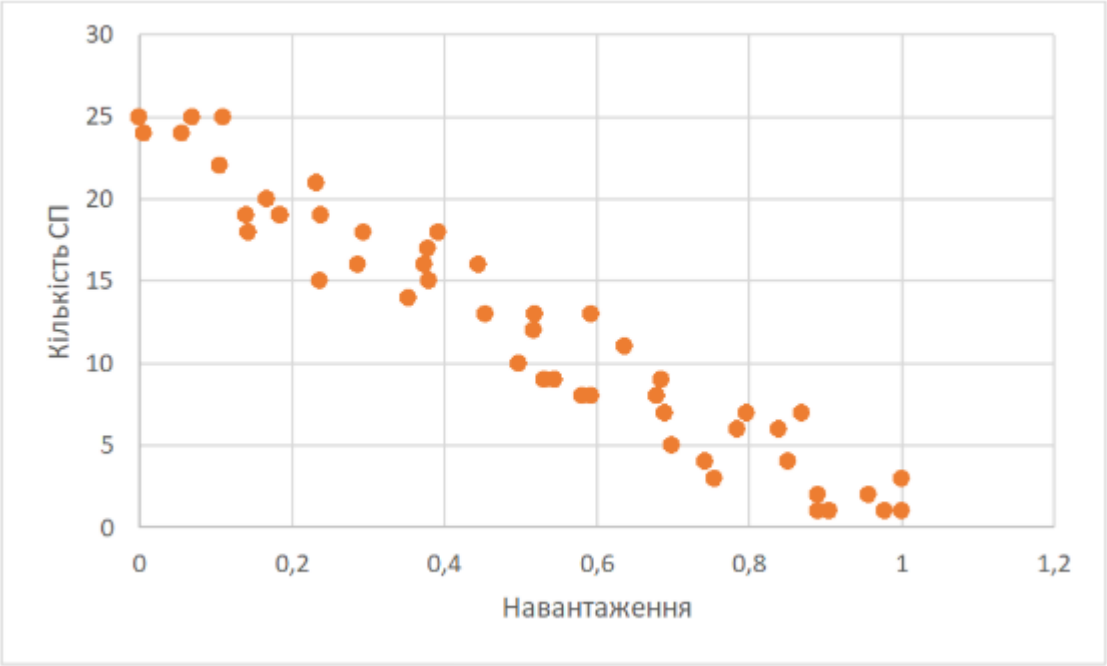


Рисунок 4.6 − Графік залежності кількості підключених пристроїв від навантаження на вузол для доступу в режимі переривань

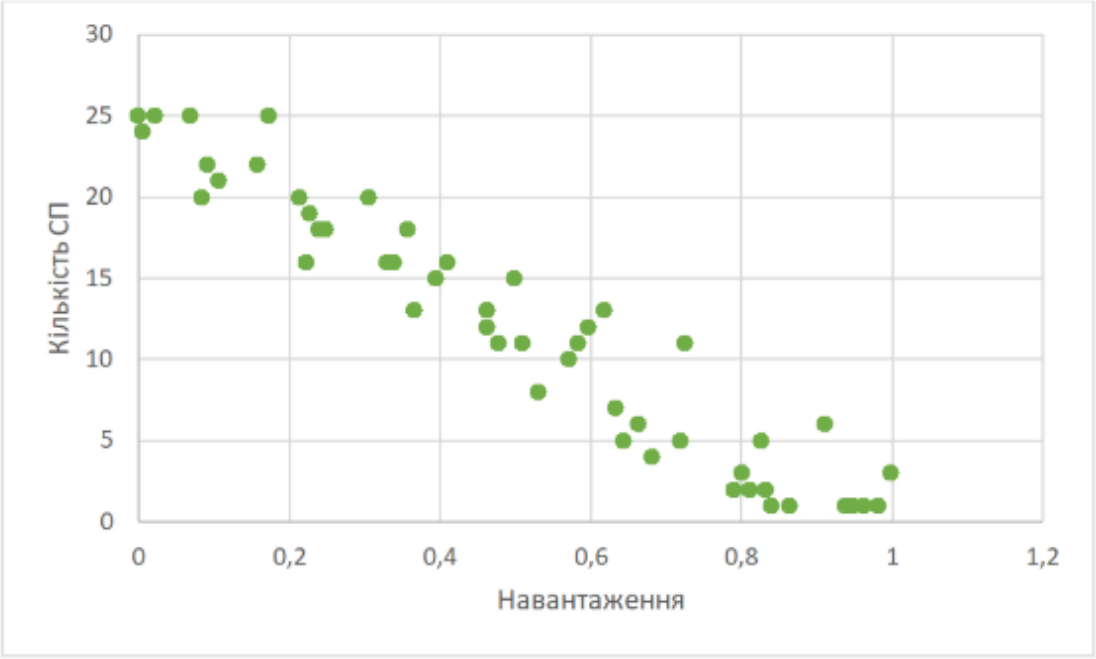


Рисунок 4.7 − Графік залежності кількості підключених пристроїв від навантаження на вузол для доступу в режимі множинного доступу

Нижче наведено графік порівняння залежності кількості підключених пристроїв від навантаження на вузол для різних режимів доступу (рис. 4.8).

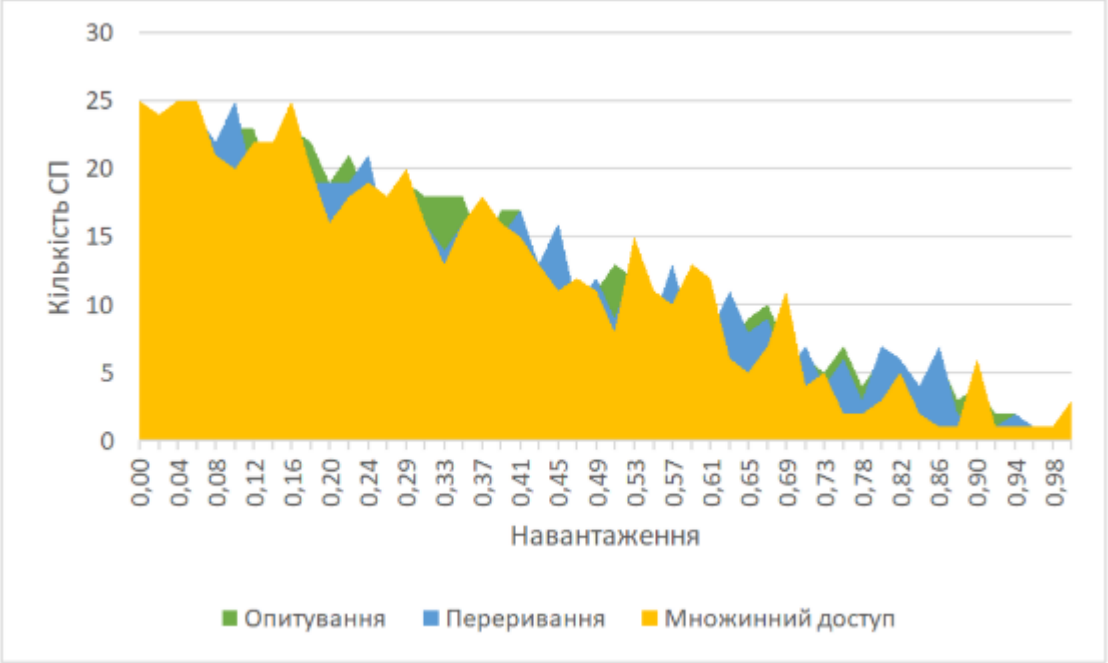


Рисунок 4.8 – Графік порівняння залежності кількості підключених пристроїв від навантаження на вузол для різних режимів доступу

На основі результатів експерименту виявлено, що режими доступу мають різну ефективність залежно від навантаження:

- режим опитування демонструє поступове зменшення кількості підключених пристроїв із зростанням навантаження, оскільки вузол обробляє запити послідовно. Це створює значні затримки, що робить цей режим менш ефективним для великих мереж;

- режим переривання є більш гнучким, оскільки обробляє запити одразу після їх надходження. Це забезпечує стабільну роботу при середньому навантаженні, але при критичних перевантаженнях також відбувається зменшення кількості підключених пристроїв;

- режим множинного доступу виявився найбільш ефективним за помірного навантаження, забезпечуючи стабільну кількість пристроїв. Однак при перевищенні порогу потужності вузла кількість активних пристроїв різко зменшується через перевантаження.

Режим множинного доступу є найкращим вибором для мереж із помірним навантаженням. Для інтенсивних навантажень більш доцільним є

режим переривання, який забезпечує кращу адаптивність. Режим опитування підходить лише для невеликих та стабільних мереж із прогнозованою кількістю пристроїв.

4.1.4 Оцінка залежності середнього часу передачі даних від навантаження вузла

Мета експерименту – вивчення залежності середнього часу передачі даних від сенсорних пристроїв до туманного вузла від рівня завантаження сервера. У процесі моделювання оркестратор не бере участі, кількість підключених пристроїв залишається сталою.

Вхідні параметри:

- кількість підключених СП (N): 10 пристроїв;
- потужність вузла (P): 2500 кілобайтів на секунду;
- розмір пакета (σ): змінюється від 50 кілобайтів до 5000 кілобайтів;
- час передачі (t): час, необхідний для передачі одного пакета даних від СП до сервера.

Алгоритм моделювання:

- на кожному кроці розмір пакета збільшується, що призводить до збільшення навантаження на вузол;
- для кожного розміру пакета розраховується середній час передачі пакета на основі співвідношення розміру пакета до потужності вузла та додаткових затримок через черги, якщо вузол перевантажений.

Якщо загальний обсяг даних, що передається одночасно ($N \times \sigma$), менший за потужність вузла (P), середній час передачі визначається емпіричним шляхом, виходячи з розрахунків завантаження вузла.

Нижче наведено графіки залежності середнього часу передачі даних від СП на сервер в залежності від навантаження вузла (рис. 4.9).

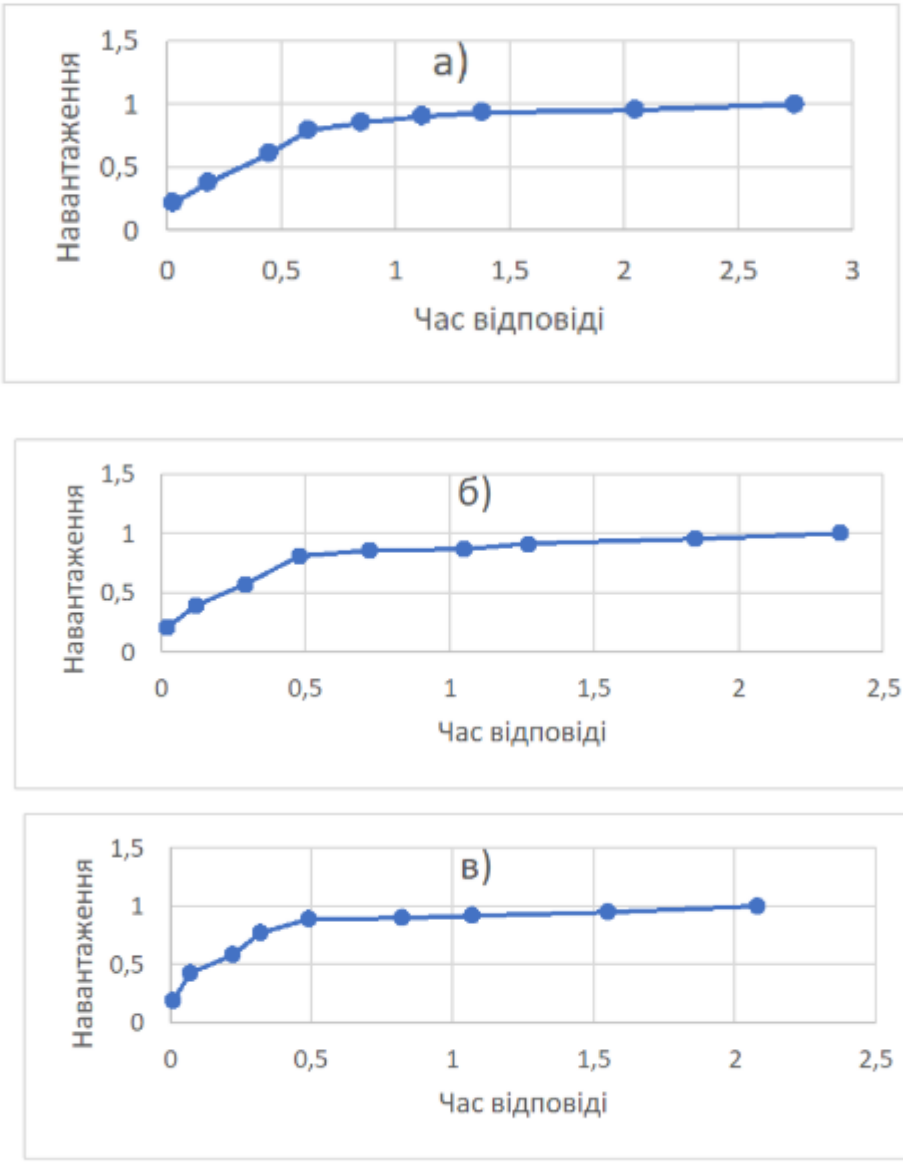


Рисунок 4.9 − Графіки залежності середнього часу передачі даних від СП на сервер в залежності від навантаження вузла

а) опитування; б) переривання; в) множинний доступ

Графік порівняння середнього часу доступу від навантаження вузла для кожного з режимів зображено нижче (рис. 4.10).

Результати моделювання демонструють, як різні режими доступу впливають на середній час передачі даних залежно від рівня завантаження вузла. У режимі опитування середній час передачі зростає найшвидше через те, що вузол обробляє запити послідовно. Для малих розмірів пакетів вузол працює без перевантажень, що забезпечує низькі затримки. Однак зі

збільшенням розміру пакета сервер перевантажується, що значно підвищує середній час передачі. Максимальний час досягається для найбільших розмірів пакетів, коли черги стають критичними.

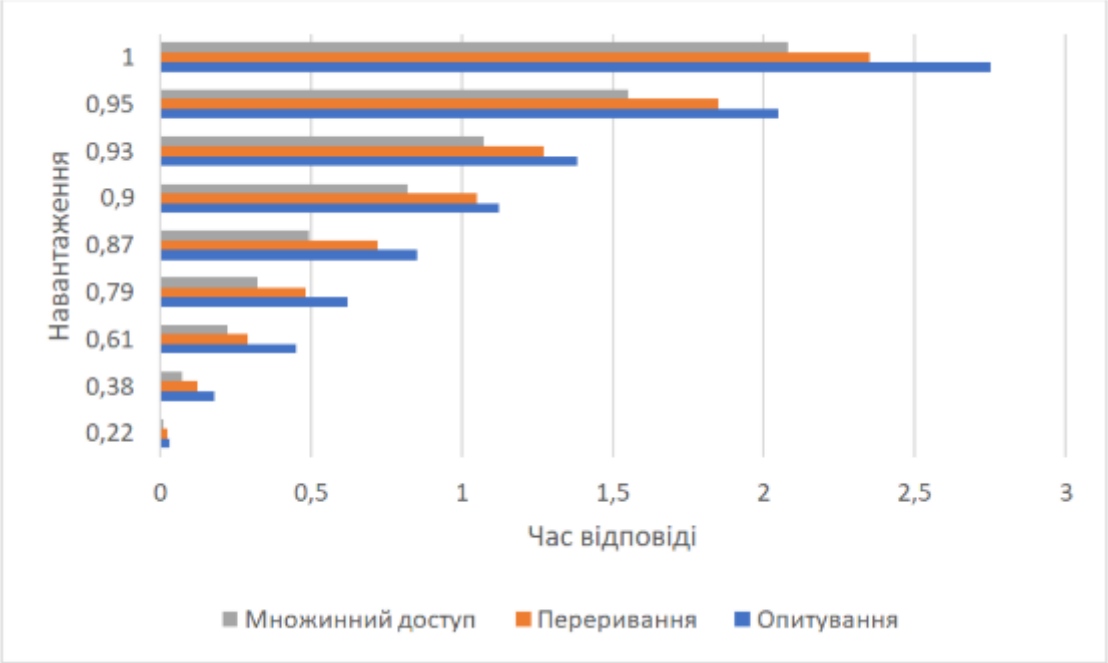


Рисунок 4.10 – Графік порівняння середнього часу доступу від навантаження вузла для кожного з режимів

Режим переривання демонструє кращі результати, оскільки запити обробляються одразу після надходження, що знижує ймовірність накопичення черг. Незважаючи на це, при великих розмірах пакетів і тут з'являються значні затримки через перевищення потужності вузла. Середній час передачі в цьому режимі зростає більш плавно порівняно з режимом опитування.

Найкращі результати спостерігаються у режимі множинного доступу, де вузол обробляє запити паралельно, що забезпечує стабільність середнього часу передачі для малих і середніх розмірів пакетів. Лише при значному перевищенні потужності вузла середній час передачі починає різко зростати через утворення черг.

4.2 Аналіз результатів моделювання інформаційної взаємодії в мережі Інтернету речей з топологією mesh

4.2.1 Опис моделі mesh-мережі

Дослідження mesh-мережі спрямоване на аналіз її властивостей у контексті встановлення з'єднань, стабільності та надійності роботи. Основна увага приділяється оцінці залежності параметрів мережі від таких змінних, як ймовірність втрати каналу зв'язку, втрати вузла та інтервалу втрати роутера.

Математично модель mesh-мережі представлена у вигляді матриці суміжності. Ця матриця відображає зв'язки між вузлами мережі, де кожен елемент вказує на наявність або відсутність зв'язку між конкретними вузлами, а також вагу цього зв'язку. Візуалізувати модель можна у вигляді графа, де вершини відповідають маршрутизаторам, а ребра − мережевим з'єднанням між ними. Вага ребра моделює «вагу» передачі даних, що є аналогом затримки, пропускної здатності або енергетичних витрат у реальних мережах. Введення такого параметра є доречним, оскільки в реальних умовах мережеві з'єднання відрізняються за якістю і продуктивністю, що впливає на вибір маршруту для передачі даних [97-98].

Інтервал втрати роутера − це параметр, який визначає час, протягом якого маршрутизатор вважається недоступним, якщо він не надсилає сигнали про своє існування. У контексті даної моделі, цей параметр є ключовим для оцінки стабільності мережі, оскільки від його значення залежить швидкість реакції на втрату вузла та час відновлення топології. Менший інтервал втрати сприяє швидшому оновленню топології, але може створювати додаткове навантаження на мережу через часті оновлення. Збільшений інтервал, у свою чергу, може призводити до затримок у виявленні проблем у мережі, що впливає на надійність передачі даних.

Функціонування mesh-мережі забезпечується протоколами динамічної маршрутизації. Ці протоколи дозволяють вузлам автоматично оновлювати маршрути передачі даних у разі змін у топології мережі, таких як поява нових

вузлів або втрата існуючих. У цій моделі використовується спрощена реалізація протоколу Open Shortest Path First (OSPF). OSPF − це протокол внутрішньої маршрутизації, який належить до категорії протоколів стану каналу (link-state). Основною його перевагою є можливість швидко адаптуватися до змін у топології мережі та забезпечувати маршрутизацію за найкоротшим шляхом.

Для пошуку найкоротшого шляху в моделі використовується алгоритм Дейкстри. Це ефективний алгоритм, який дозволяє знайти найкоротший шлях від одного вузла до всіх інших у графі з невід'ємними вагами ребер. Алгоритм працює за принципом послідовного розширення множини оброблених вузлів, обираючи кожного разу вузол із мінімальною поточною відстанню. У контексті mesh-мережі цей алгоритм забезпечує швидке й оптимальне визначення маршрутів для передачі даних [99].

Експеримент проведено на мережі з топологією mesh (рис. 4.11), що має відповідну матрицю суміжності (рис. 4.12).

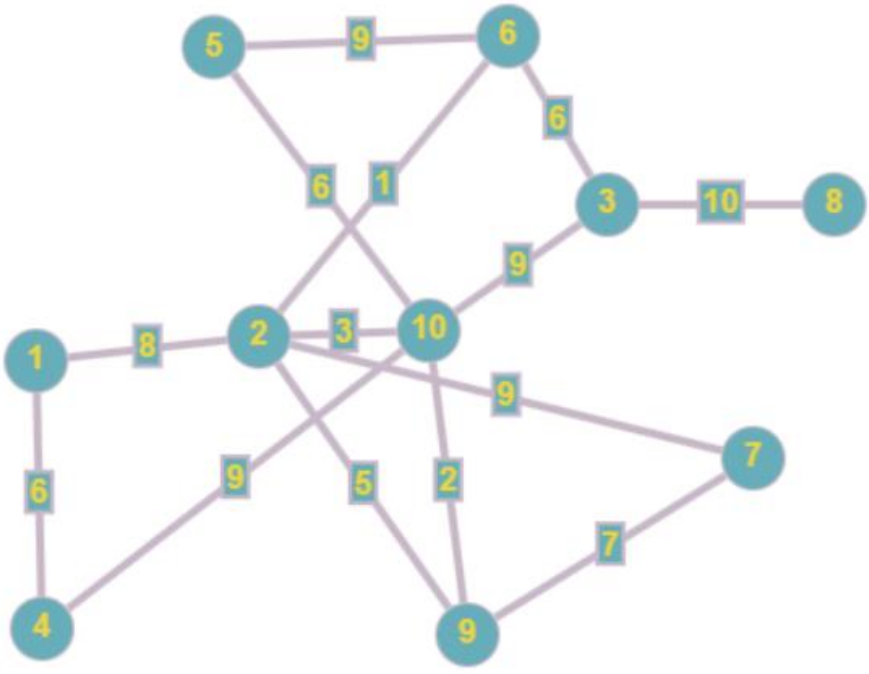


Рисунок 4.11 – Візуалізація mesh-мережі у вигляді графу

	1	2	3	4	5	6	7	8	9	10
1	0	8	0	6	0	0	0	0	0	0
2	8	0	0	0	0	1	9	0	5	3
3	0	0	0	0	0	6	0	10	0	9
4	6	0	0	0	0	0	0	0	0	9
5	0	0	0	0	0	9	0	0	0	6
6	0	1	6	0	9	0	0	0	0	0
7	0	9	0	0	0	0	0	0	7	0
8	0	0	10	0	0	0	0	0	0	0
9	0	5	0	0	0	0	7	0	0	2
10	0	3	9	9	6	0	0	0	2	0

Рисунок 4.12 – Матриця суміжності mesh-мережі

Програмна реалізація моделі mesh-мережі виконується мовою Python із використанням стандартних бібліотек, таких як NumPy. Граф мережі описується через матрицю суміжності, що дозволяє моделювати зв’язки між вузлами. Для організації роботи моделі створені три класи: MeshNetwork, Node і OSPFEmulator.

Клас MeshNetwork відповідає за управління мережею. Він забезпечує збереження структури мережі у вигляді матриці суміжності та надає функціональність для оновлення топології і розрахунку маршрутів. Зокрема, методи класу дозволяють моделювати втрату вузлів або каналів зв’язку та обчислювати найкоротші маршрути між вузлами за допомогою алгоритму Дейкстри.

Клас Node представляє окремий вузол мережі. У ньому зберігається інформація про ідентифікатор вузла та його статус. Це дозволяє відслідковувати активність вузлів та моделювати їхню поведінку в разі виходу з ладу.

Клас OSPFEmulator реалізує спрощену версію протоколу OSPF для маршрутизації в мережі. Він забезпечує періодичну розсилку інформації про топологію від головного вузла до всіх інших і моніторинг доступності вузлів. Розсилки виконуються з фіксованими інтервалами, що моделює поведінку реальних мережевих протоколів.

Реалізація моделі включає низку спрощень у порівнянні з реальним протоколом OSPF. Зокрема:

- відсутність hello-повідомлень та зворотних hello response. У стандартному OSPF ці повідомлення використовуються для встановлення та підтримки зв'язків між сусідніми вузлами. У моделі замість цього передбачено фіксований час у 1 секунду для встановлення з'єднання з працюючим вузлом;

- відсутність призначеного маршрутизатора. У OSPF призначений маршрутизатор використовується для централізованого поширення інформації про топологію мережі серед вузлів. У моделі цю роль виконує випадково обраний головний вузол, який надсилає інформацію про поточну топологію у всі напрямки з фіксованим інтервалом у 1,5 секунди. Передача даних у мережі можлива лише після того, як усі вузли отримають актуальну інформацію про топологію.

Ці спрощення роблять модель придатною для експериментального аналізу, зберігаючи при цьому основні характеристики реальної mesh-мережі. Вибраний підхід дозволяє зосередитися на ключових аспектах роботи мережі, таких як стабільність і адаптивність у змінних умовах.

4.2.2 Аналіз часу встановлення з'єднання в mesh-мережі в залежності від надійності з'єднання

Метою експерименту є дослідження часу, необхідного для встановлення з'днання між вузлами mesh-мережі в залежності від ймовірності відключення каналу зв'язку між маршрутизаторами. Для аналізу проводяться два окремі сценарії з різними значеннями параметру інтервалу втрати роутера :

- $t = 5$ секунд;
- $t = 1$ секунда.

Методика експерименту передбачає:

- вибір топології mesh-мережі;
- генерацію певного набору ймовірностей відключення каналів зв'язку

(від 0% до 50% з кроком 10%);

- проведення моделювання процесу встановлення з'єднання для кожного набору параметрів;
- вимірювання середнього часу встановлення з'єднання.

Для кожного значення параметру t результати фіксуються у вигляді таблиць.

За результатами експерименту було побудовано графіки залежності часу встановлення з’єднання в мережі від ймовірності втрати каналу зв’язку між вузлами (рис. 4.13).

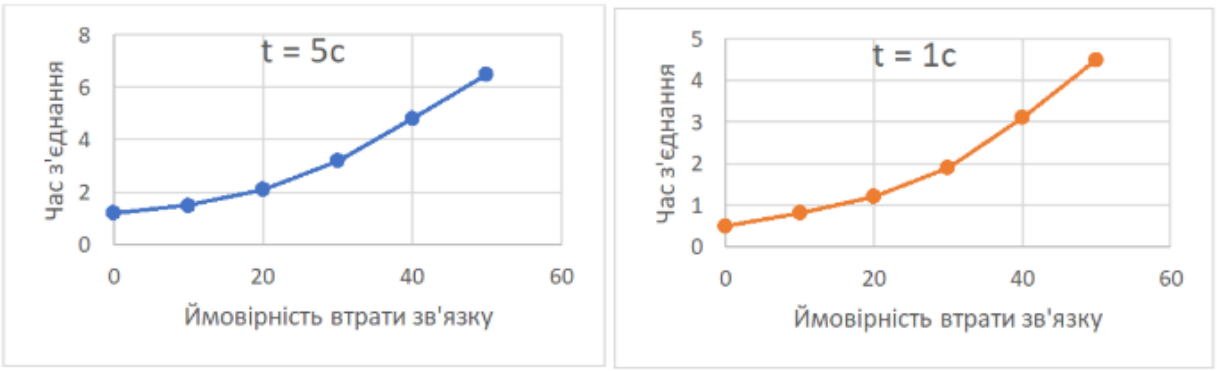


Рисунок 4.13 – Графіки залежності часу з’єднання від ймовірності втрати каналу

Результати експерименту дали можливість проаналізувати та порівняти час з’єднання для різних значень параметру t (рис. 4.14).

Аналіз отриманих даних демонструє, що зменшення параметру t (інтервалу втрати роутера) з 5 секунд до 1 секунди суттєво знижує середній час встановлення з'єднання для всіх значень ймовірності відключення каналів зв'язку. Це пояснюється тим, що при меншому значенні t мережа швидше реагує на втрати з'єднання та оновлює топологію. Однак, при високих значеннях ймовірності відключення (від 40% до 50%) ефективність зменшується через значне навантаження на процес налаштування маршрутів.

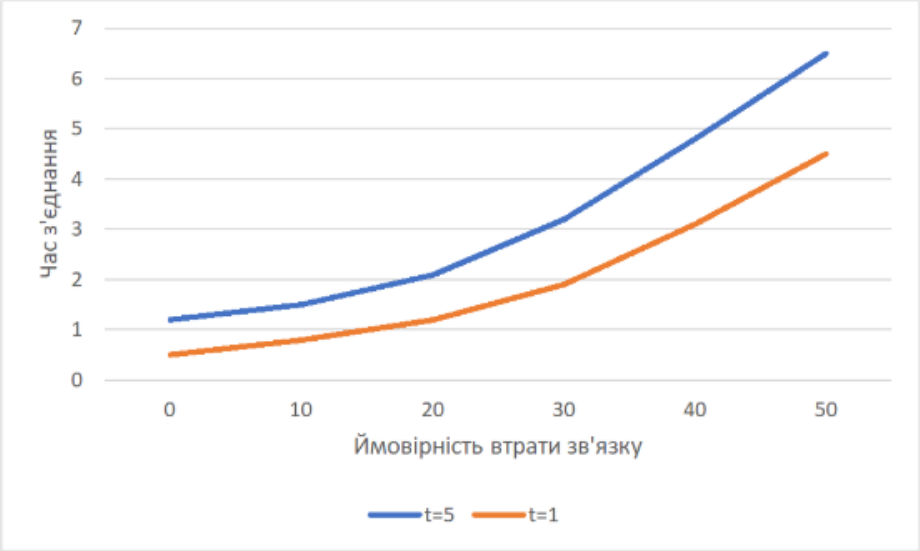


Рисунок 4.14 – Графіки порівняння часу з’єднання для різних значень t

Таким чином, для мережі з низькою ймовірністю відключення каналів доцільно використовувати короткий інтервал втрати роутера, тоді як для мережі з високою ймовірністю відключення оптимальним буде збільшення цього параметру з метою зменшення загального навантаження на маршрутизатори.

4.2.3 Аналіз часу встановлення з'єднання в mesh-мережі в залежності від надійності вузлів

Дослідження спрямоване на виявлення залежності часу встановлення з'єднання в mesh-мережі від надійності вузлів. Експериментальні розрахунки проводяться для двох різних сценаріїв з різними значеннями параметра інтервалу втрати роутера:

- $t = 5$ секунд;
- $t = 1$ секунда.

Методологія:

- задається топологія mesh-мережі;
- встановлюються різні значення ймовірності відключення вузлів (від 0% до 80% з кроком 10%);

- для кожного сценарію виконується моделювання процесу встановлення з'єднання;

- замірюється середній час встановлення з'єднання для кожного набору параметрів.

За результатами експерименту було побудовано графіки залежності часу встановлення з’єднання в мережі від ймовірності відключення вузла (рис. 4.15) та здійснено аналіз та порівняння часу з’єднання для різних значень параметру t (рис. 4.16).

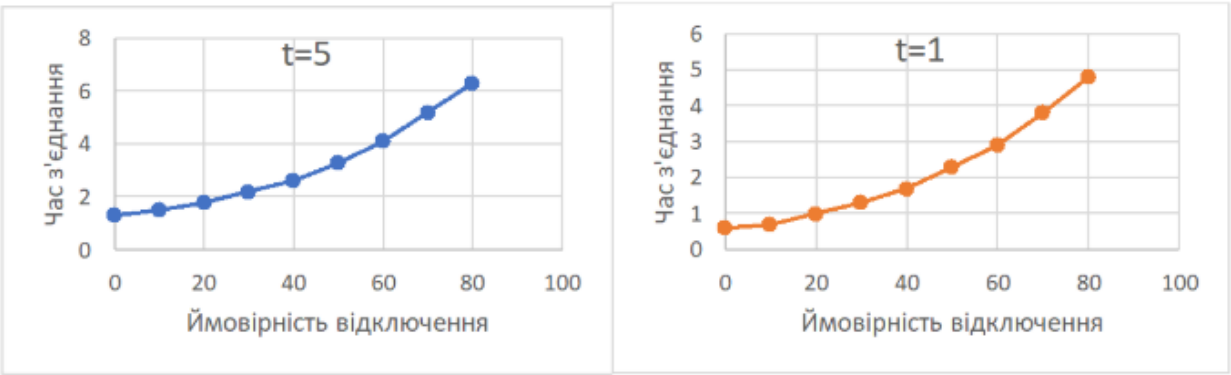


Рисунок 4.15 – Графіки залежності часу з’єднання від ймовірності відключення вузла

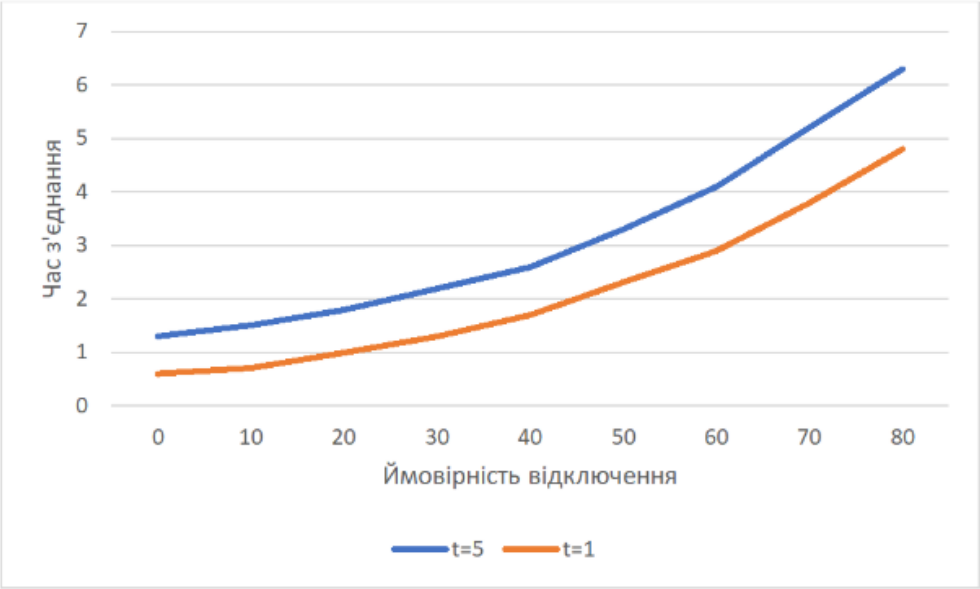


Рисунок 4.16 – Графіки порівняння часу з’єднання для різних значень t

Результати демонструють, що скорочення інтервалу втрати роутера з 5 секунд до 1 секунди сприяє суттєвому зниженню часу встановлення з'єднання. Загалом, стабільність mesh-мережі залежить як від ймовірності відключення вузлів, так і від швидкості реагування на втрати з'єднання. Менший інтервал дозволяє оперативніше адаптуватися до змін у топології, однак при високій ймовірності відключення це створює значне навантаження на маршрутизатори. Тому баланс між надійністю з'єднань і частотою налаштування є ключовим фактором ефективної роботи мережі. При цьому мережа оперативніше реагує на відключення вузлів, що знижує середній час встановлення з'єднання при низьких та середніх значеннях ймовірності відключення (від 0% до 40%) [100].

При високій ймовірності відключення вузлів (від 50% до 80%) мережа стикається з частими налаштуваннями маршрутів, що значно збільшує час встановлення з'єднання. Порівняно з втратою з'єднань між вузлами, відключення самих вузлів створює більш критичний вплив на загальну продуктивність мережі, оскільки це впливає не лише на маршрутизацію, але й на доступність вузлів для передачі даних. У таких умовах мережа вимагає більш збалансованих параметрів налаштування для зниження навантаження і забезпечення стабільної роботи.

Таким чином, на стабільність mesh-мережі значно більше впливає відключення вузлів, ніж втрата з'єднань між ними, оскільки перше призводить до втрати доступності ключових компонентів мережі. Для мереж із високою ймовірністю відмов вузлів слід обирати довший інтервал втрати роутера для забезпечення стабільності. Натомість коротший інтервал є ефективним у мережах із низькою ймовірністю відмов, дозволяючи швидко відновлювати з'єднання.

4.3 Оцінка працездатності алгоритмів самоорганізації

4.3.1 Опис моделі генетичного алгоритму розміщення СП

Головними завданнями моделі генетичного алгоритму для оптимізації розміщення сенсорних пристроїв на площині з перешкодами є аналіз спроможності ГА для розв'язання подібних задач, оцінка його ефективності в залежності від параметрів (ймовірність мутації, розмір популяції) та оцінка часу розв'язання задачі.

ГА широко використовуються для розв’язання складних оптимізаційних задач, де традиційні методи неефективні. Вони знаходять застосування у таких областях, як планування шляхів, проєктування мереж, конфігурація об’єктів та пошук параметрів [101].

Переваги ГА містять здатність знаходити глобальні оптимуми в багатовимірних просторах та можливість працювати з недосконалими моделями. Недоліки включають неоднорідність в збіжності та високі обчислювальні витрати.

Для реалізації задачі оптимального розміщення СП на площині з перешкодами була створена модель, що успішно представляє подібні умови.

Площина була спрощена до двовимірного масиву, який кодує різні типи об’єктів. Кодування має наступний вигляд:

- значення 0 відповідає порожньому місцю;
- значення 1 кодує перешкоду;
- значення $2 - N + 1$ свідчить про індекс СП, де N − кількість СП.

Уведення перешкод у модель було обумовлено неоднорідністю реальних умов передачі сигналу у мережах IoT. Умовні перешкоди представлені у вигляді прямокутників та можуть мати випадкову ширину та висоту.

Радіус дії СП та їх кількість визначаються випадково на початку симуляції ГА. При цьому сумарна площа покриття всіх СП не перевищує площі самої мапи. Розмір мапи завжди однаковий і складає 35×35 блоків.

Генерація перешкод також відбувається випадково, залежно від параметра максимального відсотка наповненості мапи перешкодами від загальної площі. Для симуляції використано значення не більше 40% [102].

У ролі фітнес-функції використовується відношення суми покриття СП для поточного рішення відносно максимально можливого для такої кількості СП із сумарною площею їх радіусів. Такий підхід дозволяє оцінити ефективність кожного індивіда в популяції та вибрати найкращі варіанти для подальшої еволюції.

Масив мапи (A) розділений на два допоміжні:

- B − зберігає розташування перешкод;
- C − зберігає позиції СП.

Це необхідно для того, щоб ГА мав доступ лише до зміни розміщення СП і не міг змінювати розташування перешкод. На виході алгоритм генерує масив C, який потім об'єднується з масивом B, формуючи початковий A, але з оновленим розташуванням СП.

Для реалізації моделі використано мову програмування Python. Додаток розбитий на дві частини:

- фронтенд − відповідає за візуалізацію роботи алгоритму;
- бекенд − займається обчисленнями, реалізацією ГА та управлінням класами представлення об'єктів мапи.

Для графічної візуалізації було використано бібліотеки Matplotlib та Tkinter. Matplotlib забезпечує створення графіків і наочного представлення роботи алгоритму, а Tkinter слугує для побудови графічного інтерфейсу користувача.

Програма побудована на основі модульної архітектури та містить такі основні компоненти:

- Obstacle − відповідає за генерацію та зберігання параметрів перешкод;
- Station − моделює станції передавання, їхні параметри та радіус дії;
- ГА − реалізація основних операцій генетичного алгоритму (селекція, кросовер, мутація, оцінка фітнес-функції);

- генерація мапи − відповідає за створення масивів A, B і C та інтеграцію результатів роботи алгоритму;

- візуалізація − відображає початкове та оптимізоване розташування СП та перешкод на площині.

Така структура забезпечує зручність розширення функціоналу, підтримку та модифікацію окремих компонентів програми.

4.3.2 Аналіз ефективності вирішення задачі заповнення площини СП

Для аналізу ефективності генетичного алгоритму у вирішенні задачі оптимального розміщення СП проведено серію симуляцій. Метою експерименту є визначення максимальної точності розв'язання задачі, що характеризується відношенням площі покриття до максимально можливої площі покриття для заданих параметрів [103].

Умови експерименту:

- розмір карти: 35×35 блоків;
- максимальний відсоток перекриття: 40%;
- ймовірність мутації: 10%;
- розмір популяції: 100 особин.

Експеримент проводився шляхом запуску ГА, де кожне покоління оцінювалося за допомогою фітнес-функції, яка визначає відсоток покриття площі мапи СП. Шляхом багатократного повторення експерименту були отримані середні значення оптимальності рішення (від 0 до 100 %) для кожного покоління (рис. 4.17).

Протягом експерименту спостерігається тенденція до покращення покриття з кожним наступним поколінням, що свідчить про ефективність генетичного алгоритму у знаходженні оптимальних рішень для розміщення СП на площині з перешкодами. Найвищі показники покриття були досягнуті на 1750-му поколінні, коли фітнес-функція досягла значення 96 %. Після цього

значення коливалося біля 93 %, що вказує на ефективність ГА у стабільному досягненні високих показників покриття на даній площині.

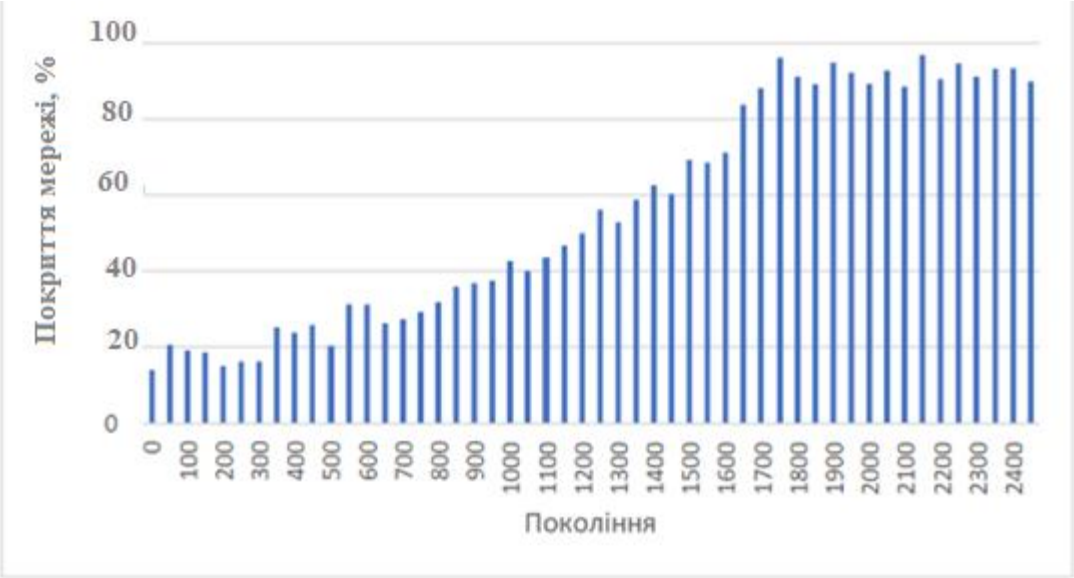


Рисунок 4.17 – Графік ефективності покриття мережі для кожного покоління

Проте, деякі покоління, зокрема 0, 100, 200, 250, 300, 500, показали тимчасове зниження значення фітнес-функції, що може свідчити про потребу в додатковій еволюції для подолання локальних мінімумів. Водночас загальна стабільність і ефективність алгоритму протягом експерименту дозволяють зробити висновок про його здатність адаптуватися до різних умов і наявності перешкод. Висока стабільність у досягненні покриття у фінальних поколіннях також підтверджує правильність налаштування параметрів ГА, що дозволяє досягти значних результатів при заданих умовах експерименту. Загалом, експеримент демонструє високу ефективність генетичного алгоритму в задачі оптимального розміщення СП на площині з перешкодами.

Оскільки в додатку реалізовано графічний інтерфейс, то є можливість візуально порівняти якість вирішення задачі для початкового та фінального поколінь (рис. 4.18).

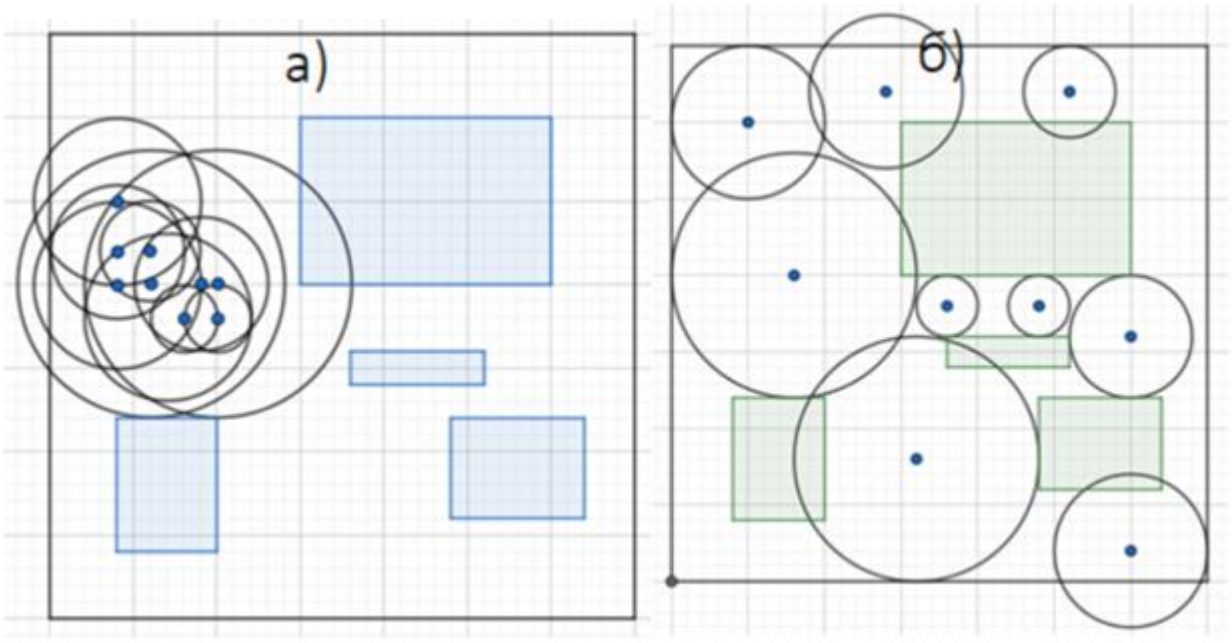


Рисунок 4.18 – Візуальне відображення рішення алгоритму на першому та останньому поколіннях
а) покоління 1; б) покоління 2500

Оскільки алгоритм заснований на випадкових числах, перше покоління обирає серед 100 згенерованих варіантів той, який забезпечує найбільше вільного простору для найбільшого кола. Цей варіант отримує найвищий пріоритет за результатами фітнес-функції та передає свій код наступному поколінню. Однак, це рішення не є оптимальним, оскільки воно не враховує взаємодію з іншими сенсорними пристроями і потенційні конфлікти. Результат останнього покоління, в свою чергу, демонструє оптимальне розміщення СП на площині, враховуючи перетин фігур, перешкоди, розмір мапи та радіус кожного СП, що дозволяє досягти найкращого покриття з урахуванням усіх обмежень.

4.3.3 Аналіз ефективності вирішення задачі в залежності від ймовірності мутації

Для оцінки ефективності ГА в задачі оптимального розміщення СП на площині з перешкодами було проведено серію експериментів, що дозволило

вивчити залежність результативності алгоритму від значення ймовірності мутації.

Експеримент полягає у варіюванні параметра ймовірності мутації від 10% до 80% з кроком 10%. Для кожного значення ймовірності мутації була проведена окрема симуляція, і для кожної симуляції зафіксовано максимальне значення фітнес-функції, що представляє ефективність розміщення СП з урахуванням перешкод на площині.

Вхідні параметри симуляції:

- розмір карти: 35×35 блоків;
- відсоток перекриття мапи: 40%;
- максимальна кількість поколінь: 2500;
- розмір популяції: 100;
- ймовірність мутації: варіація від 10% до 80%.

Методика проведення експерименту:

- для кожного значення ймовірності мутації проводиться 2500 поколінь, в межах яких оцінюється фітнес-функція для кожного індивідуума в популяції;
- ймовірність мутації варіюється від 10% до 80% з кроком 10%;
- для кожного значення ймовірності мутації записується максимальне значення фітнес-функції, що відповідає ефективності розв'язку задачі для відповідного налаштування мутації.

Результати для різних значень ймовірності мутації представлено в графічному вигляді (рис. 4.19).

На графіку можна спостерігати, що для ймовірності мутації, яка не перевищує 50%, генетичний алгоритм здатен знаходити оптимальні або близькі до оптимальних рішення для задачі розміщення сенсорних пристроїв. Однак, після перетину цього порогу, зростає ймовірність того, що мутації занадто часто порушують корисні особливості популяції, що були збережені в процесі еволюції. Це призводить до погіршення результатів.

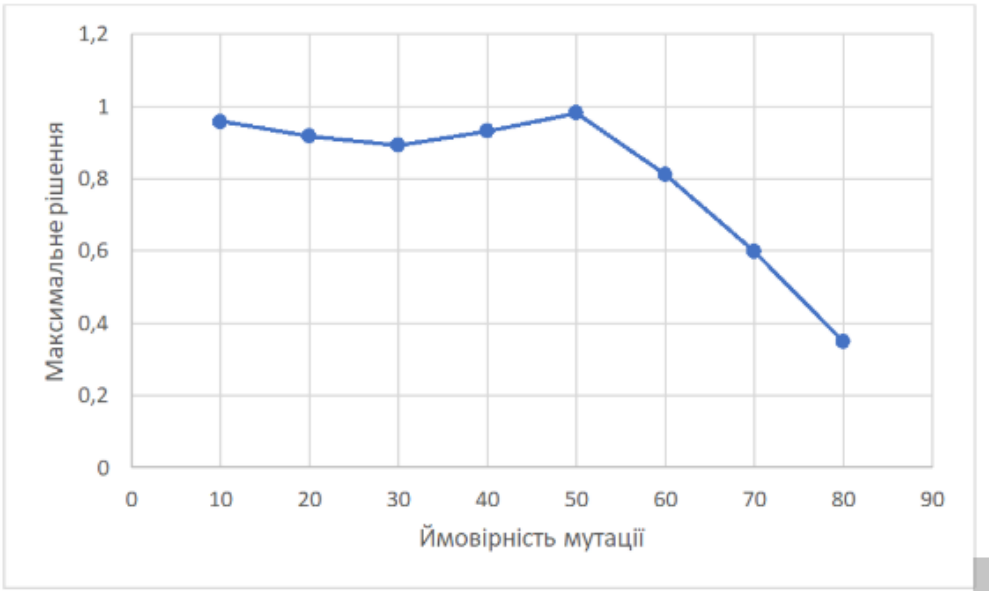


Рисунок 4.19 – Графік залежності максимального значення фітнес-функції від параметру ймовірності мутації

При ймовірності мутації понад 50%, значення фітнес-функції знижуються лінійно, і досягають рівня 0,35 при 80% мутації. Такий результат можна пояснити тим, що при високій ймовірності мутації індивідууми часто втрачають корисні риси, що дозволяють досягти ефективного покриття, і в результаті стають «хаотичними», не забезпечуючи сталого покращення розв'язку. Занадто велика ймовірність мутації знижує стабільність алгоритму, в той час як оптимальні значення для ймовірності мутації дозволяють алгоритму поступово вдосконалювати рішення, підтримуючи баланс між варіативністю та стабільністю [104].

Графік значення фітнес-функції відповідно до номеру популяції з ймовірністю мутації у 70% зображено нижче (рис. 4.20).

З графіку видно, що при ймовірності мутації 70% генетичний алгоритм досягає локального максимуму значення фітнес-функції на 1300-у поколінні, після чого значення стабілізується на рівні 0,60. Однак цей максимум на 30% нижчий, ніж при ймовірності мутації 10%. Висока ймовірність мутації викликає більші коливання значень фітнес-функції, оскільки часті зміни

індивідуумів порушують корисні риси, що вже були знайдені, і знижують стабільність еволюційного процесу.

Нижче також наведено порівняння графіків процесу вирішення задачі для однакової мапи з ймовірністю мутації 10% та 70% (рис. 4.21).

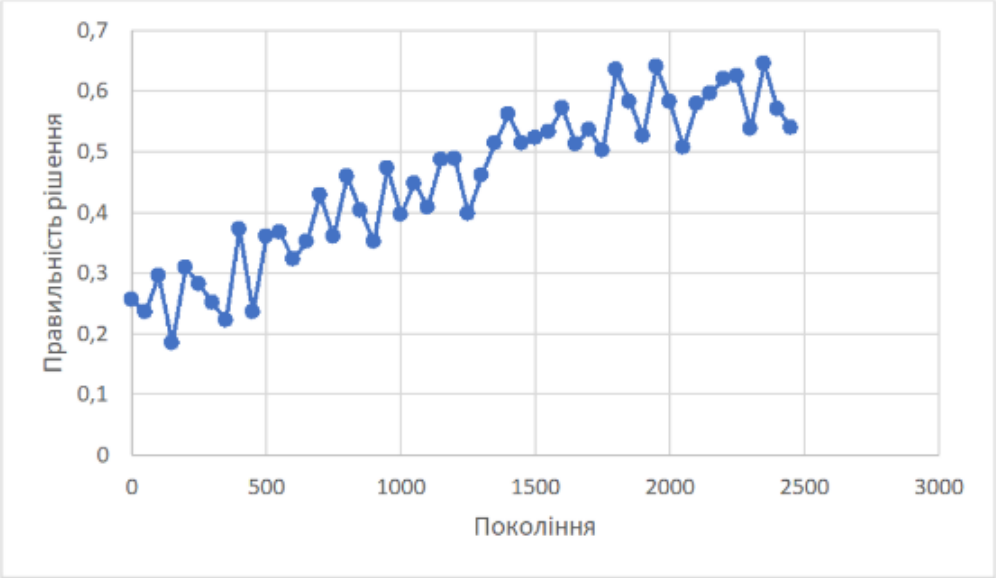


Рисунок 4.20 – Графік вирішення задачі з ймовірністю мутації у 70%

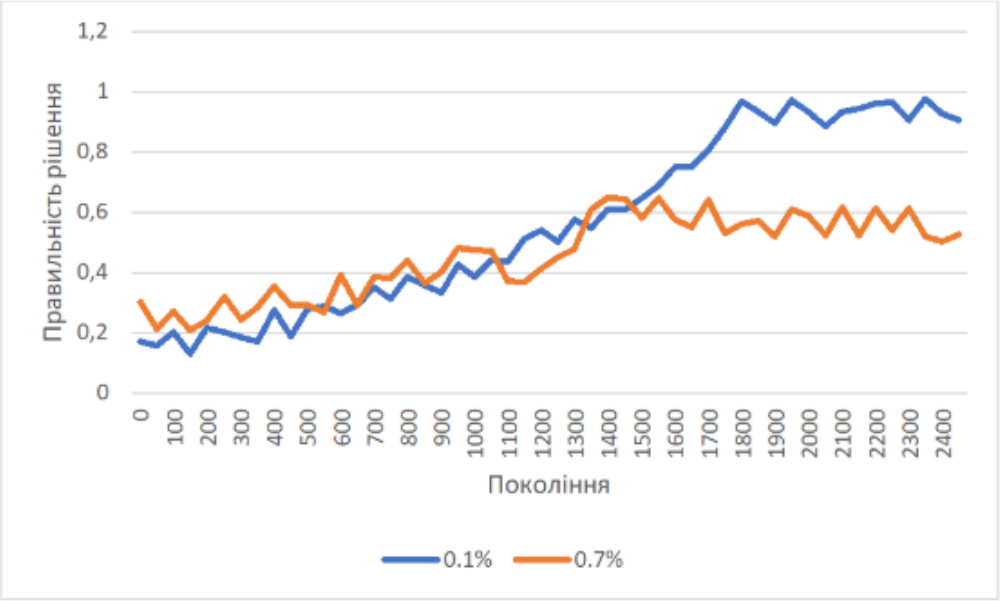


Рисунок 4.21 – Графіки рішення задачі для ймовірностей мутації 10% та 70%

З порівняння графіків для ймовірностей мутації 10% та 70% можна зробити висновок, що при 10% ймовірності значення фітнес-функції стабільно

зростає без великих коливань, та алгоритм досягає свого максимуму без значних коливань. Це свідчить, що при низькій ймовірності мутації алгоритм підтримує більш стійкий процес еволюції, що дозволяє знайти оптимальне рішення без надмірних змін у популяції.

Таким чином, висока ймовірність мутації призводить до швидших, але менш стабільних змін у популяції, що може знизити ефективність алгоритму. Натомість, низька ймовірність мутації забезпечує стабільніший і ефективніший пошук оптимальних рішень, хоча й повільніший [105].

4.4 Висновки до розділу

В четвертому розділі «Практична реалізація інформаційної взаємодії в мережах Інтернету речей» досліджується ефективність різних підходів до реалізації інформаційної взаємодії в мережах Інтернету речей. Аналіз моделей туманного обчислення показав, що для різних режимів доступу сенсорних пристроїв до туманних вузлів (опитування, переривання, множинний доступ) характерна різна продуктивність. Найбільш ефективним у мережах із помірним навантаженням виявився режим множинного доступу, тоді як режим переривання доцільно використовувати для високих навантажень, а режим опитування є оптимальним для невеликих та стабільних мереж.

Дослідження залежності продуктивності від навантаження продемонструвало, що використання оркестратора дозволяє підтримувати стабільну роботу мережі навіть за умов критичного перевантаження, оскільки він динамічно керує ресурсами, поступово відключаючи сенсорні пристрої.

У контексті mesh-мереж було встановлено, що стабільність і швидкість встановлення з'єднань залежать від таких параметрів, як ймовірність втрати каналів зв'язку чи вузлів, а також інтервалу втрати роутера. Менший інтервал дозволяє швидше відновлювати топологію мережі, однак створює підвищене навантаження на маршрутизатори.

Результати моделювання генетичного алгоритму підтвердили його

ефективність у знаходженні оптимальних рішень для задач розміщення сенсорних пристроїв в умовах обмежень. Оптимальні результати досягались при помірній ймовірності мутації (10–30%), що забезпечує баланс між варіативністю та стабільністю. Натомість висока ймовірність мутації призводить до хаотичних змін і знижує ефективність алгоритму.

На основі проведеного аналізу було розроблено рекомендації для оптимізації мереж. Для мереж із низькою ймовірністю втрат з’єднань доцільно встановлювати короткий інтервал втрати роутера, що покращує швидкість реагування системи. У мережах із високою ймовірністю відмов вузлів слід збільшувати цей інтервал для забезпечення стабільності. Генетичний алгоритм підтвердив свою перспективність у задачах планування та оптимізації розміщення сенсорних пристроїв у мережах Інтернету речей.

ВИСНОВКИ

У дисертаційній роботі вирішено наукову задачу розробки моделей і алгоритмів інформаційної взаємодії для мереж Інтернету речей, які враховують їх децентралізовану структуру, обмежені ресурси та специфічні особливості. Результати дослідження підтверджують актуальність та значущість обраного напрямку і можуть бути застосовані для створення ефективних IoT-систем, які відповідають сучасним викликам цифрової епохи.

Було проведено глибокий аналіз архітектурних особливостей, протоколів та технологій, що використовуються в мережах IoT. Виявлено ключові виклики, серед яких обмежені ресурси пристроїв, динамічна зміна топології мережі, необхідність забезпечення надійності та якості обслуговування. Узагальнено сучасні підходи до проєктування та управління IoT-мережами, визначено їх недоліки та можливості вдосконалення.

Було запропоновано імітаційну модель інформаційної взаємодії, яка враховує ймовірнісно-часові характеристики передачі даних у IoT-мережах. Модель дозволяє оцінювати продуктивність, затримки та надійність передачі інформації, враховуючи обмежені ресурси пристроїв та складність топології мережі.

Було розроблено адаптивну математичну модель доступу в туманних обчисленнях, яка враховує коефіцієнт завантаженості мережі, ймовірність колізій та середній час очікування доступу. Використання адаптивного вибору режиму дозволило зменшити середній час передачі даних на 8% порівняно з класичними моделями.

Було розроблено ймовірнісну модель встановлення інформаційної взаємодії в мережі IoT із топологією mesh, яка враховує інтервал втрати роутера та його вплив на час встановлення з'єднання. Дослідження показали, що скорочення параметра t з 5 с до 1 с дозволяє зменшити середній час встановлення з’єднання на 33% у мережах із низькою ймовірністю втрати зв’язку (до 20%). Однак при високих втратах (40–50%) короткі інтервали

можуть створювати додаткове навантаження, збільшуючи час встановлення з’єднання на 15–20%.

Було проведено моделювання інформаційної взаємодії в умовах «туманних» обчислень. Результати експериментальних досліджень підтвердили ефективність запропонованих методик. Запропоновані рішення сприяють зменшенню навантаження на мережу та покращенню якості обслуговування.

Було розроблено метод самоорганізації мережі IoT на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, який забезпечує ефективну маршрутизацію та мінімізує затримку передачі даних. Проведені експериментальні дослідження продемонстрували:

- покриття мережі досягло 96%, що перевищує ефективність традиційних методів (грід-методи – 70–85%, жадібні алгоритми – 80–90%, стохастичні методи – 85–93%);
- забезпечено швидку збіжність алгоритмів та ефективне використання сенсорів;
- довжина маршруту зменшена на 12% у порівнянні з традиційними алгоритмами, такими як алгоритм Дейкстри;
- затримка передачі даних мінімізована завдяки ефективному розміщенню вузлів та вдосконаленій маршрутизації;
- час передачі даних зменшено на 9% порівняно з існуючими методами.

Досліджено інваріантність залежностей ймовірнісно-часових характеристик від параметрів мережі IoT. Виконано імітаційне моделювання запропонованих моделей і алгоритмів у різних сценаріях функціонування IoT-мереж. Експериментально підтверджено їх ефективність у забезпеченні надійності, масштабованості та економії ресурсів. Проведено аналіз залежностей характеристик мережі від параметрів моделі, що дозволяє раціонально обирати конфігурацію системи на етапі проєктування.

Запропоновані підходи до проектування IoT-мереж базуються на інноваційних моделях та алгоритмах, які враховують специфіку сучасних і

майбутніх технологій (5G/6G). Це сприяє розвитку інтегрованих систем IoT у сферах «розумного міста», промислового Інтернету речей, охорони здоров’я тощо. Проведене дослідження дозволяє забезпечити ефективну інформаційну взаємодію в мережах Інтернету речей, що є важливим кроком для подальшого розвитку інтелектуальних систем і технологій майбутнього.

Наукова новизна та практична цінність роботи полягають в наступному:

1. Вперше розроблено адаптивну математичну модель доступу в туманних обчисленнях, яка за рахунок коефіцієнту завантаженості мережі, враховуючи ймовірність колізій та середній час очікування доступу, дозволяє зменшити середній час передачі даних в мережах IoT.

2. Удосконалено ймовірнісну модель інформаційної взаємодії в мережі IoT з топологією mesh, яка на відміну від існуючих дозволяє зменшити час встановлення з'єднання за рахунок впливу інтервалу втрати роутера.

3. Вперше розроблено метод самоорганізації мережі IoT на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, який за рахунок коефіцієнту покриття території та знаходження найкоротшого маршруту дозволяє зменшити час передачі даних.

Практичне значення роботи. Результати дослідження можуть бути використані інженерами для розробки нових рішень у сфері IoT, зокрема:

- оптимізації ресурсів у мережах із великою кількістю пристроїв;
- створення автономних систем самоорганізації;
- проєктування безпечних і надійних IoT-систем для промислового, побутового та соціального використання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ITU-T Recommendation X.509. Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks. – Geneva: ITU, 2021. – 130 p.

2. Губарев, В. Г. Інтернет речей: технології та застосування / В. Г. Губарев, Н. А. Соловйова. – Київ: Наукова думка, 2020. – 320 с.

3. A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1802.02041.pdf

4. Івченко, О. П. Технології побудови IoT-мереж / О. П. Івченко. – Харків: Видавництво ХНУ, 2019. – 256 с.

5. Ковальчук О. В., Ковальчук С. В. Аналіз побудови та роботи Mesh-технології // Збірник наукових праць ДУІКТ. – 2017. – №1. – С. 42–47.

6. Можаев О. В. Вивчення задач побудови мережі Інтернету речей / О. В. Можаев, Н. О. Кучук, Д. В. Штепа, Б. О. Соробей // Системи управління, навігації та зв'язку. – 2024. – №1. – С. 137–142. – DOI: https://doi.org/10.26906/SUNZ.2024.1.137. – Режим доступу: https://journals.nupp.edu.ua/sunz/article/view/3285.

7. Zheng K. Dynamic Beam-Based Random Access Scheme for M2M Communications in Massive MIMO Systems / K. Zheng, H. Yang, X. Xiong, J. Mei, K. Zhang // arXiv preprint arXiv:2209.14427. − 2022. − 28 с.

8. Журило О., Ляшенко О. Архітектура та системи безпеки IoT на основі туманних обчислень / О. Журило, О. Ляшенко // Innovative Technologies and Scientific Solutions for Industries. − 2024. − № 1(27). С. 54–60. DOI: 10.30837/ITSSI.2024.27.54.

9. Sethi P., Sarma S. N. Internet of Things: Architectures, Protocols, and Applications / P. Sethi, S. N. Sarma // Journal of Electrical and Computer Engineering. − 2017. − Vol. 2017. − Article ID 9324035. DOI: 10.1155/2017/9324035.

10. Tripathy B. Internet of Things (IoT): Technologies, Applications, Challenges and Solutions / B. Tripathy, J. Anuradha. − Florida : CRC Press, 2017. −334 c.

11. Bell Charles. Beginning IoT Projects: Breadboard-less Electronic Projects / Charles Bell. −Warsaw, VA, USA : Apress, 2021. − 870 p. − ISBN-13 (pbk): 978-1-4842-7233-6, ISBN-13 (electronic): 978-1-4842-7234-3. − DOI: 10.1007/978-1-4842-7234-3

12. A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications // електрон. текст. дані URL: A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications // електрон. текст. дані URL https://arxiv.org/pdf/1802.02041.pdf

13. MAC Layer Protocols for Internet of Things: A Survey // електрон. текст. дані URL:

14. Бичков В.В., Жидка О.В., Дослідження протоколів та технологій передачі даних у мережах IoT, Зв'язок, № 3(169), с. 28–32, 2024.

15. Perry Lea. IoT and Edge Computing for Architects 2nd edition / Lea Perry – Packt Publishing, 2020. – 632 p.

16. RFC 7252 – The Constrained Application Protocol (CoAP) [Електронний ресурс]. − Режим доступу: https://tools.ietf.org/html/rfc7252

17. RFC 7540 – Hypertext Transfer Protocol Version 2 [Електронний ресурс]. − Режим доступу: https://tools.ietf.org/html/rfc7540

18. Philip N. MQTT and CoAP: Underlying Protocols for the IoT [Електронний ресурс]. − Режим доступу: https://electronicdesign.com/iot/mqtt-andcoap-underlying-protocols-iot

19. Malokhvii E. Library for Bosch Sensortec BME280: combined temperature, pressure, humidity sensor [Електронний ресурс]. − Режим доступу: mhttps://github.com/malokhvii-eduard/arduino-bme280

20. Sigfox Technology // електрон. текст. дані URL:https://www.betasolutions.co.nz/Blog/17/Sigfox-Technology-Review

21. Z-Wave Technical Basics // електрон. текст. дані

URL:https://www.domotiga.nl/attachments/download/1075/Z-Wave%20Technical%20Basicssmall.pdf

22. Jack Tison, SVP Emerging Business, PanduitOctober 2015 3 Steps for Evolving IoT Architectures. [Електронний ресурс]. − Режим доступу: http://www.industrial-ip.org/en/industrial-ip/ internetof-things/3-steps-for-evolving-iot-architectures.

23. The advantages and disadvantages of Internet Of Things // електрон. текст. дані URL: https://e27.co/advantages-disadvantages-internet-things-20160615/

24. Поширені атаки на IoT та захист від них // електрон. текст. дані URL: https://corewin.ua/blog/attacks-on-iot-how-protect/

25. Безагентне управління вразливостями для IoT та OT // електрон. текст. дані URL: https://corewin.ua/blog/agentless-vulnerability-management-for-iot-and-ot/

26. 11 Biggest security challenges & solutions for IoT// електрон. текст. дані URL: https://www.peerbits.com/blog/biggest-iot-security-challenges.html

27. Жидка О.В., Дакова Л.В., Даков С.Ю., Поляшенко Д.В., Analysis of the use of software honeypots in Internet of Things, Наукові записки ДУІКТ №2, с. 92-98, 2024.

28. Mulyar I., Selyukov O., Dzhuliy V., & Kizyun B. A model for assessing the probabilistic and temporal characteristics of information interaction in the Internet of Things network // Collection of scientific papers of the Military Institute of Taras Shevchenko National University of Kyiv. – 2019. – No. 63. – P. 96–107.

29. The advantages and disadvantages of Internet Of Things // електрон. текст. дані URL: The advantages and disadvantages of Internet Of Things // електрон. текст. дані URL: https://e27.co/advantages-disadvantages-internet-things-20160615/

30. IoT security guide // електрон. текст. дані URL: https://www.dsci.in/files/content/knowledge-centre/2023/IoT-Security-Guide.pd

31. Gloukhovtsev, IoT_Security_Challenges_Solutions_and_Future_

Prospects , 2018KS [Електронний ресурс]. − Режим доступу:// https://education.dell.com/content/dam/dell-emc/documents/en-us/2018KS_ Gloukhovtsev -IoT_Security_Challenges_Solutions_and_Future _Prospects.pdf

32. IoT-Security-Challenges-and-Best-Practices // електрон. текст. дані URL: https://www.happiestminds.com/wp-content/uploads/2020/12/IoT-Security-Challenges-and-Best-Practices.pdf

33. IMDA IoT Cyber Security Guide Version 1, Jan 2019 [Електронний ресурс]. − Режим доступу: https://www.imda.gov.sg/-/media/imda/files/regulation-licensing-and-consultations/consultations/open-for-public-comments/consultation-for-iot-cyber-security-guide/imda-iot-cyber-security-guide.pdf

34. Жидка О. В., Андрійченко Т.Р., Інформаційна безпека систем IoT, Зв'язок, № 4(170), с. 65−69, 2024.

35. Д. Діденко, Поширені атаки на IoT та захист від них, 2023. URL:https://corewin.ua/blog/attacks-on-iot-how-protect/

36. Aleksander M., Korchenko O., Karpinski M., Odarchenko R., Vulnerability investigation for Internet of things sensor subnetworks architecture for different types of attacks, Ukrainian Scientific Journal of Information Security, 2016, vol. 22, issue 1, p. 12-19. URL: https://er.nau.edu.ua/bitstream/NAU/36403/1/10448-26997-1-SM.pdf

37. Безагентне управління вразливостями для IoT та OT. URL:https://corewin.ua/blog/agentless-vulnerability-management-for-iot-and-ot/

38. Olga Zhydka, Myroslav Riabyi, Tetiana Okhrimenko, Andriy Fesenko, Lazat Kydyralina, Research on data transmission technologies and information security in IoT networks, CEUR Workshop Proceedings, Social Communication and Information Activity in Digital Humanities 2024, 3851, ISSN 1613-0073.

39. Sohrabi K. Protocols for Self-Organization of a Wireless Sensor Network / K. Sohrabi, J. Gao, V. Ailawadhi, G. J. Pottie // Personal Communications, IEEE. − 2000. − Vol. 7, № 5. − С. 16–27.

40. Сокульський О. Є. Analysis of information technology capabilities

using machine-to-machine communication (M2M) for the Internet of Things / О. Є. Сокульський, Є. О. Топольськов, Ю. Д. Жданова // Вчені записки Таврійського національного університету імені В. І. Вернадського. Серія: Технічні науки. − 2024. − Т. 35, № 2. − С. 155–166.

41. Yang K., Huang W. Performance research of an improved hybrid routing protocol / K. Yang, W. Huang // IEEE. − 2019. − Vol. 19. − P. 1242–1247.

42. Аналіз засобів моделювання мережі Інтернету речей / VII Всеукраїнська науково-технічна конференція «Комп'ютерні технології: інновації, проблеми, рішення», 2–3 грудня 2024 р., Державний університет «Житомирська політехніка», м. Житомир. – Житомир: Державний університет «Житомирська політехніка», 2024.

43. Suwandhada K., Panyim K. ALEACH-Plus: An energy efficient cluster head based routing protocol for wireless sensor network / K. Suwandhada, K. Panyim // IEEE. − 2019. − Vol. 7, № 7. − P. 1–4.

44. Singh V., Lohani R. B. Mobility aware energy efficient clustering for wireless sensor network /V. Singh, R. B. Lohani//IEEE. − 2019. − Vol. 11. − P.1–6.

45. Zhou Y., Zhang H., Zhang L. FIS: An improved LEACH based on fuzzy inference system in MWSNs / Y. Zhou, H. Zhang, L. Zhang // IEEE. − 2018. − Vol. 16, № 9. − P. 699–703.

46. Rehman E., Sher M., Naqvi S. H. A. Energy efficient secure trust based clustering algorithm for mobile wireless sensor network / E. Rehman, M. Sher, S. H. A. Naqvi // Computer Networks and Communications. − 2017. − Vol. 1630673. − P. 1–8.

47. Sharma S., Mittal N. An improved LEACH-MF protocol to prolong lifetime of wireless sensor networks / S. Sharma, N. Mittal // IEEE. − 2018. − Vol. 18. − P. 174–179.

48. Yadrovskaya M. V., Porksheyan M. V., Sinelnikov A. A. Prospects of Internet of Things technology // Advanced Engineering Research. – 2021. – Vol. 21, No. 2. – P. 207–217. – URL: https://cyberleninka.ru/article/n/perspektivy-tehnologii-interneta-veschey

49. Astakhova T. N., Kolbanev M. O., Romanova A. A. Smart agricultural field based on the Internet of Things // Regional Informatics and Information Security. Proceedings Collection. – 2018. – Issue 5 / SPOISU. – St. Petersburg, 2018. – P. 201–202.

50. Нефедченко О.О. Модель інформаційної взаємодії в мережах IoT /О.О. Нефедченко // Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення: збірник тез доповідей, 12 жовтн. 2021р. − Тернопіль: 2021. − випуск 62. − С.42− 44.

51. Жидка О.В., Бондаренко Д. А., Модель оцінки ймовірнісно-часових параметрів інформаційної взаємодії в Інтернеті речей, Зв'язок, № 5(171), с. 59−66, 2024.

52. Miorandi, D., Sicari, S., De Pellegrini, F., & Chlamtac, I. (2012). Internet of Things: Vision, applications and research challenges. Ad Hoc Networks, 10(7), 1497-1516. https://doi.org/10.1016/j.adhoc.2012.02.016

53. Nižetić S., Šolić P., López-de-Ipiña González-de-Artaza D., Patrono L. Internet of Things (IoT): Opportunities, issues and challenges towards a smart and sustainable future // Journal of Cleaner Production. – 2020. – T. 274. – C. 122877. https://doi.org/10.1016/j.jclepro.2020.122877

54. Довгий С.О., Згуровський М.З., Лагутін А.А. Інформаційно-комунікаційні системи: основи побудови та перспективи розвитку. — Київ: Національний технічний університет України «КПІ», 2016. ¬ 450 с.

55. Kurose, J.F., Ross, K.W. Computer Networking: A Top-Down Approach. — Pearson, 2017. ¬ 864 pages.

56. Shafique K., Khawaja B.A., Sabir F., Qazi S., Mustaqim M. Internet of Things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for 5G-IoT scenarios // IEEE Access. – 2020. – Vol. 8. – P. 23022–23040.

57. Laroui M., Nour B., Moungla H., Cherif M.A., Afifi H., Guizani M. Edge and fog computing for IoT: A survey on current research activities & future directions // Future Generation Computer Systems. – 2020. – Vol. 109. – P. 924–

931. https://doi.org/10.1016/j.comcom.2021.09.003

58. Ali O., Ishak M.K., Bhatti M.K.L. New IoT domains, current standings and open research: A review // PeerJ Computer Science. – 2021. – Vol. 7. – Article e659. – DOI: https://doi.org/10.7717/peerj-cs.659.

59. Lin J., Yu W., Zhang N., Yang X., Zhang H., Zhao W. A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications // IEEE Internet of Things Journal. – 2017. – Vol. 4. – P. 1125–1142.

60. Ali O., Ishak M.H., Bhatti M.K.L., Khan I., Kim K.-I. A comprehensive review of Internet of Things: Technology stack, middlewares, and Fog/Edge computing interface // Sensors. – 2022. – Vol. 22. – Article 995. – DOI: https://doi.org/10.3390/s22030995.

61. Zakharov M. V., Kirichek R. V. Methods for constructing ultra-dense e-health networks using edge computing // Proceedings of the SPbNTORES Annual Conference. – 2020. – No. 1(75). – P. 145–147.

62. Borodin A. S., Kucheryavyy A. E., Paramonov A. I. Features of using D2D technologies depending on the density of users and devices // Elektrosvyaz (Electrosvyaz). – 2018. – No. 10. – P. 40–45.

63. Жидка О.В., Ймовірнісна модель встановлення інформаційної взаємодії в мережі Інтернету речей з топологією mesh, Телекомунікаційні та інформаційні технології, № 4, с. 43-52, 2024.

64. Verzun N., Kolbanev M., Shamin A. The architecture of the access protocols of the global infocommunication resources computers // Computers. – 2020. – Vol. 9, No. 2. – Article 49. – DOI: https://doi.org/10.3390/computers 9020049 (accessed: 25.07.11).

65. Verzun N. A., Kolbanev M. O., Omelyan A. V. Regulated multiple access in the wireless network of smart things // Omsk Scientific Bulletin. Series: Informatics, Computing Technology and Management. – 2016. – No. 4 (148). – P. 147–151.

66. Kucheryavyy A. E., Prokop'yev A. V., Kucheryavyy E. A. Self-organizing networks. – St. Petersburg: Lyubavich, 2011. – 312 p.

67. Bogatyrev V. A., Bogatyrev S. V., Derkach A. N. Timeliness of the reserved maintenance by duplicated computers of heterogeneous delay-critical stream // CEUR Workshop Proceedings. – 2019. – Vol. 2522. – P. 26–36.

68. Krivoulya G., Tokariev V., Ilina I, Shcherbak V. Mathematical Model for Finding Probability of Detecting Victims of Man-Made Disasters Using Distributed Computer System with Reconfigurable Structure and Programmable Logic / G. Krivoulya, V. Tokariev, I. Ilina, V. Shcherbak // IEEE International ScientificPractical Conference Problems of Infocommunications, Science and Technology: (PIC S&T), 06-09 oct. 2020y. - Kharkiv, 2020. - P.197 – 201.

69. Saaty T. L. The Analytic Hierarchy Process. – New York: McGraw-Hill, 1980. – 270 p.

70. Rogers A., David E., Jennings N. R. Self-organized routing for wireless microsensor networks // IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans. – 2005. – Vol. 35, No. 3. – P. 349–359.

71. Dressler F. Bio-Inspired Networking – Self-Organizing Networked Embedded Systems. – Berlin, Germany: Springer Berlin Heidelberg, 2008. – 320 p.

72. Sohrabi K., Gao J., Ailawadhi V., Pottie G. Self-Organizing Wireless Sensor Network // Proceedings of the 37th Annual Allerton Conference on Communication, Control, and Computing. – 1999. – P. 379–385.

73. Krishnan R. Efficient clustering algorithms for self-organizing wireless sensor networks // Ad Hoc Networks. – 2006. – Vol. 4, No. 1. – P. 36–52.

74. Batta M. S., Harous S., Louail L., Aliouat Z. A distributed TDMA scheduling algorithm for latency minimization in Internet of Things / M. S. Batta, S. Harous, L. Louail, Z. Aliouat // IEEE. − 2019. − Vol. 19. − P. 108−113.4.

75. Chen G., Hu H.-X. Finding the Optimal Network Topology for the Distributed Multi-Short-Paths Routing Algorithm – A Genetic Algorithm-Based Approach // Proceedings of the 2022 International Conference on Intelligent Systems and Computational Intelligence. China, 2022. C. 35–38. DOI: 10.1109/ICISCI53188.2022.9941373.

76. Chai Y., Zeng X.J. An effective routing with delay minimization for

multi-hop wireless mesh network // Proceedings of the IEEE Global Communications Conference. 2019. C. 1–6. DOI: 10.1109/GLOBECOM38437. 2019.9013776.

77. Praveen J. S., Kumanan T. Link Loss Identification and Congestion-Aware Routing in Mobile Wireless Sensor Network // Proceedings of the 2024 2nd International Conference on Device Intelligence, Computing and Communication Technologies. India, 2024. C. 585–589. DOI: 10.1109/DICCT61038. 2024.10532948.

78. Gripsy J.V., Jayanthiladevi A. Energy Optimization and Dynamic Adaptive Secure Routing for MANET and Sensor Network in IoT // Proceedings of the 2023 7th International Conference on Computing Methodologies and Communication (ICCMC). Erode, India, 2023. C. 1283–1290. DOI: 10.1109/ICCMC56507.2023.10083519.

79. Aluvala S., Reddy G., Al-Jawahry H.M., Ramadan G.M., Pareek P.K. Improved Sail Fish Optimizer for Energy Efficient Routing in Wireless Sensor Network // Proceedings of the 2023 International Conference on Integrated Intelligence and Communication Systems. India, 2023. C. 1–4. DOI: 10.1109/ICIICS59993.2023.10421391.

80. Joshi R.D., Banu S. Brief Survey on Wireless Sensor Network Routing // Proceedings of the 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS). Bangalore, India, 2023. C. 1–8. DOI: 10.1109/CSITSS60515.2023.10334097.

81. Bairagi P.P., Dutta M., Babulal K.S. Location based Routing Protocols and its Performances in Wireless Sensor Networks: An Investigation // Proceedings of the 2022 3rd International Conference on Electronics and Sustainable Communication Systems. India, 2022. C. 583–590. DOI: 10.1109/ICESC54411.2022.9885717.

82. Anitha T., Sridhar S. Novel Improved Communication Steadiness Routing for Wireless Sensor Network's Performance Analysis compared with Network Boundary Maintenance Routing // Proceedings of the 2023 International

Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering (ICECONF). Chennai, India, 2023. С. 1–8. DOI: 10.1109/ICECONF57129.2023.10083655.

83. Mishra J., Bagga J., Choubey S., Gupta I.K. Energy optimized routing for wireless sensor network using elitist genetic algorithm // Proceedings of the 2017 8th International Conference on Computing, Communication and Networking Technologies. India, 2017. С. 1–5. DOI: 10.1109/ICCCNT.2017.8204110.

84. Pyrih Y., Kaidan M., Tchaikovskyi I., Pleskanka M. Research of Genetic Algorithms for Increasing the Efficiency of Data Routing // Proceedings of the 2019 3rd International Conference on Advanced Information and Communications Technologies (AICT). Lviv, Ukraine, 2019. С. 157–160. DOI: 10.1109/AIACT.2019.8847814.

85. Rares M. Adaptive mutation in genetic algorithms for shortest path routing problem // Proceedings of the 2015 7th International Conference on Electronics, Computers and Artificial Intelligence (ECAI). Bucharest, Romania, 2015. С. 69–74. DOI: 10.1109/ECAI.2015.7301163.

86. Пиріг Я., Климаш М., Пиріг Ю., Лаврів О. Генетичний алгоритм як засіб розв’язання оптимізаційних задач // Інфокомунікаційні технології та електронна інженерія. 2023. №3(2). С. 95–107.

87. Bai Y. et al. A Deep Reinforcement Learning-Based Geographic Packet Routing Optimization // IEEE Access. 2022. Vol. 10. С. 108785–108796. DOI: 10.1109/ACCESS.2022.3213649.

88. Wang D., Li B. Analysis of a local routing in scale-free networks // Proceedings of the 27th Chinese Control and Decision Conference (CCDC). Qingdao, China, 2015. С. 1936–1939. DOI: 10.1109/CCDC.2015.7162236.

89. Min J., Kim W., Paek J., Govindan R. Effective Routing and Scheduling Strategies for Fault-Tolerant Time-Sensitive Networking // IEEE Internet of Things Journal. 2024. Vol. 11, №6. С. 11008–11020. DOI: 10.1109/JIOT.2023.3328626.

90. Rani S., Beniwal R., Saini K. Performance Evaluation on Various

Routing Strategies in IoT // Proceedings of the 2023 9th International Conference on Signal Processing and Communication (ICSC). NOIDA, India, 2023. С. 139–144. DOI: 10.1109/ICSC60394.2023.10441445.

91. Sun W. Data link network resource allocation method based on genetic algorithm // Proceedings of the 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). Chengdu, China, 2019. С. 1875–1880.

92. Gao M. Opposite and Chaos Searching Genetic Algorithm Based for UAV Path Planning // Proceedings of the 2020 IEEE 6th International Conference on Computer and Communications. China, 2020. С. 2364–2369.

93. Пиріг Я. Оцінка обчислювальної складності генетичного алгоритму // Інфокомунікаційні технології та електронна інженерія. 2024. №4(1). С. 52–60.

94. Assessment of energy characteristics of multiple access in wireless networks / N. A. Verzun, M. O. Kolbanev, A. A. Romanova, V. V. Tsehanovsky // Bulletin of SPbGETU “LETI”. – 2019. – No. 10. – P. 34–38.

95. Naraliyev N. A., Samal D. I. Review and analysis of standards and protocols in the Internet of Things area. Modern testing methods and issues of IoT information security // International Journal of Open Information Technologies. – 2019. – Vol. 7, No. 8. – P. 94–104.

96. Bogatyrev V. A., Bogatyrev A. V., Bogatyrev S. V. The Probability of Timeliness of a Fully Connected Exchange in a Redundant Real-Time Communication System // Wave Electronics and its Application in Information and Telecommunication Systems (WECONF 2020). – 2020. – URL: https://ieeexplore.ieee.org/document/9131517

97. Basics of modeling information systems for multiple access: Textbook / N. A. Verzun, M. O. Kolbanev, I. L. Korshunov, S. Yu. Mikadze. – St. Petersburg: SPbGEU, 2015. – 142 p.

98. Власенко В. О. Аналіз технологій побудови мережі передавання даних із високими вимогами щодо інформаційної безпеки, надійності та

затримки / В. О. Власенко, Ю. В. Щавінський, М. М. Запорожченко, В. С. Тищенко // Зв'язок. −2023. −№ 3. −DOI: 10.31673/2412-9070.2023.032030.

99. Elnashar, A. I. IoT Applications and Security / A. I. Elnashar. – Springer, 2020. – 350 p.

100. Можаев О. В. Вивчення задач побудови мережі Інтернету речей / О. В. Можаев, Н. О. Кучук, Д. В. Штепа, Б. О. Соробей // Системи управління, навігації та зв'язку. – 2024. – №1. – С. 137–142. – DOI: https://doi.org/10.26906/SUNZ.2024.1.137. – Режим доступу: https://journals.nupp.edu.ua/sunz/article/view/3285.

101. Самойленко М. Ю. Принципи застосування технології Інтернет речей у сучасному світі техніки / М. Ю. Самойленко // Вчені записки ТНУ імені В. І. Вернадського. Серія: Технічні науки. – 2020. – Т. 31(70), №6, ч. 1. – С. 142–148. – DOI: https://doi.org/10.32838/TNU-2663-5941/2020.6-1/24. – Режим доступу: https://www.tech.vernadskyjournals.in.ua/journals/2020/6_2020/part_

102. Кравченко О. В. Особливості побудови та реалізації мереж Інтернету речей в різних галузях промисловості / О. В. Кравченко // Електронний архів КПІ ім. Ігоря Сікорського. – 2020. – Режим доступу: https://ela.kpi.ua/items/d52e6375-bc14-4cb8-ade5-a234e9fb1b16. – Назва з екрана.

103. Савченко І. В. Аналіз проблем Інтернету речей при побудові мережі / І. В. Савченко, О. М. Бондаренко // Науковий вісник Чернігівського національного технологічного університету. – 2021. – №274. – С. 95–102. – Режим доступу: https://ir.stu.cn.ua/handle/123456789/27495. – Назва з екрана.

104. Кучук Н. О. Програмні технології Інтернету речей / Н. О. Кучук, О. В. Можаев // Факультет інформаційних технологій КНУ ім. Т. Шевченка. – 2020. – Режим доступу: https://fit.knu.ua/wp-content/uploads/2020/02/126_bak_FIT_2020_compressed.pdf. – Назва з екрана.

105. Петренко О. П. Інтернет речей: математичні методи та моделі / О. П. Петренко, І. В. Іванов // Кібернетика та системний аналіз. – 2020. – №4. – С.

12–25. – Режим доступу: https://www.kibernetika.org/PDFsE/2020/04/12.pdf.

ДОДАТОК А

SensorDevice.руш

```
import time
import random
import numpy as np
from queue import Queue

class SensorDevice:
def __init__(self, id, tau, sigma, fog_nodes):
self.id = id
self.tau = tau # Частота генерації даних (секунди)
self.sigma = sigma # Розмір пакета (КБ)
self.fog_nodes = fog_nodes # Список доступних туманних вузлів

def send_data(self):
while True:
fog_node = Orchestrator.select_fog_node(self.fog_nodes)
if fog_node:
fog_node.receive_data(self.sigma)
time.sleep(self.tau)
```

FogNode.py

```
class FogNode:
def __init__(self, id, capacity):
self.id = id
self.capacity = capacity # Потужність обробки (КБ/с)
self.queue = Queue()
```

```
self.processed_data = 0

def receive_data(self, data_size):
if self.queue.qsize() * data_size < self.capacity:
self.queue.put(data_size)
else:
print(f"FogNode {self.id} is overloaded! Data queued.")

def process_data(self):
while True:
if not self.queue.empty():
data_size = self.queue.get()
processing_time = data_size / self.capacity
time.sleep(processing_time)
self.processed_data += data_size
```

Orchestrator.py

```
class Orchestrator:
@staticmethod
def select_fog_node(fog_nodes):
node_loads = [(node, node.queue.qsize()) for node in fog_nodes]
node_loads.sort(key=lambda x: x[1])
return node_loads[0][0] if node_loads else None
@staticmethod
def monitor_load(fog_nodes):
loads = {node.id: node.queue.qsize() for node in fog_nodes}
return loads
```

mesh_network.py

```
import numpy as np
import time
import random

class Node:
def __init__(self, node_id: int):
self.node_id = node_id
self.active = True
def set_status(self, status: bool):
self.active = status

def is_active(self) -> bool:
return self.active

def __str__(self):
return f"Node({self.node_id}, {'active' if self.active else 'inactive'})"

class MeshNetwork:
def __init__(self, nodes: list):
# Зберігаємо вузли у словнику: id -> Node
self.nodes = {node.node_id: node for node in nodes}
self.n = len(nodes)
# Матриця суміжності: якщо з'єднання немає – weight = infinity
self.adj_matrix = np.full((self.n, self.n), float('inf'))
# Відстань від вузла до себе = 0
np.fill_diagonal(self.adj_matrix, 0)

def add_link(self, node1_id: int, node2_id: int, weight: float):
```

```
        """Додає двостороннє з'єднання між вузлами з вказаною вагою."""
        self.adj_matrix[node1_id][node2_id] = weight
        self.adj_matrix[node2_id][node1_id] = weight
        print(f"Link added between Node {node1_id} and Node {node2_id} with weight {weight}.")

    def remove_link(self, node1_id: int, node2_id: int):
        """Видаляє з'єднання між вузлами."""
        self.adj_matrix[node1_id][node2_id] = float('inf')
        self.adj_matrix[node2_id][node1_id] = float('inf')
        print(f"Link removed between Node {node1_id} and Node {node2_id}.")
    def disable_node(self, node_id: int):
        """Симулює відмову вузла, видаляючи усі його зв'язки."""
        if node_id in self.nodes:
        self.nodes[node_id].set_status(False)
        # Всі зв'язки з вузлом стають недоступними
        self.adj_matrix[node_id, :] = float('inf')
        self.adj_matrix[:, node_id] = float('inf')
        self.adj_matrix[node_id][node_id] = 0 # Діагональ залишається 0
        print(f"Node {node_id} disabled in the network.")

    def enable_node(self, node_id: int):
        """Активує вузол. (Відновлення зв'язків потрібно робити окремо)"""
        if node_id in self.nodes:
        self.nodes[node_id].set_status(True)
        print(f"Node {node_id} enabled in the network.")

    def dijkstra(self, start_node_id: int) -> dict:
        """Обчислює найкоротші шляхи від вузла start_node_id до всіх інших за допомогою алгоритму Дейкстри."""
```

```
        distances = {node_id: float('inf') for node_id in self.nodes}
        distances[start_node_id] = 0
        visited = set()

        for _ in range(self.n):
        current = None
        current_distance = float('inf')
        # Обираємо наступний вузол з мінімальною поточною відстанню
        for node_id in distances:
        if node_id not in visited and distances[node_id] < current_distance:
        current = node_id
        current_distance = distances[node_id]
        if current is None:
        break
        visited.add(current)
        # Оновлюємо відстані для сусідніх вузлів
        for neighbor in range(self.n):
        if self.adj_matrix[current][neighbor] != float('inf') and neighbor in self.nodes:
        if self.nodes[neighbor].is_active():
        new_distance = current_distance + self.adj_matrix[current][neighbor]
        if new_distance < distances[neighbor]:
        distances[neighbor] = new_distance
        return distances

        def __str__(self):
        return f"MeshNetwork with {self.n} nodes."
        class OSPFEmulator:
        def __init__(self, mesh_network: MeshNetwork, router_dead_interval: float
= 3.0):
        self.mesh_network = mesh_network
```

```
        self.router_dead_interval = router_dead_interval
        # Випадковий вибір головного вузла (без DR)
        self.main_node_id = random.choice(list(mesh_network.nodes.keys()))
        # Записуємо час останнього отримання "серцевого ритму" (heartbeat) для кожного вузла
        self.last_heartbeat = {node_id: time.time() for node_id in mesh_network.nodes}

        def broadcast_topology(self):
        """Симулює розсилку інформації про топологію від головного вузла."""
        print(f"\nMain node {self.main_node_id} broadcasting topology...")
        current_time = time.time()
        for node_id, node in self.mesh_network.nodes.items():
        if node.is_active():
        self.last_heartbeat[node_id] = current_time
        print("Topology broadcast complete.")

        def run(self, duration: float = 10):
        """Запускає симуляцію протоколу OSPF на вказаний час (duration у секундах)."""
        start_time = time.time()
        while time.time() - start_time < duration:
        self.broadcast_topology()
        time.sleep(1.5) # Інтервал розсилки 1.5 сек.
        self.check_nodes()

        if __nam__ == "__main__":
        # Створюємо вузли мережі
        nodes = [Node(i) for i in range(5)]
```

```
# Створюємо об'єкт мережі
network = MeshNetwork(nodes)

# Додаємо з'єднання між вузлами із заданими вагами
network.add_link(0, 1, 2)
network.add_link(0, 2, 5)
network.add_link(1, 2, 1)
network.add_link(1, 3, 2)
network.add_link(2, 3, 3)
network.add_link(3, 4, 1)

print("\n" + str(network))

# Обчислюємо найкоротші маршрути від вузла 0
distances = network.dijkstra(0)
print("\nShortest paths from Node 0:")
for node_id, distance in distances.items():
print(f"Node 0 -> Node {node_id}: {distance}")

# Запускаємо симуляцію OSPF (наприклад, на 10 секунд)
emulator = OSPFEmulator(network, router_dead_interval=3)
emulator.run(duration=10)
```

genetic_algorithm.py

```
import numpy as np
import matplotlib.pyplot as plt
import random

# Параметри мапи
```

```
MAP_SIZE = 35
OBSTACLE_PERCENT = 40
NUM_SENSORS = 10
SENSOR_RADIUS = 3
POPULATION_SIZE = 50
MUTATION_RATE = 0.1
GENERATIONS = 100

# Генерація випадкової мапи з перешкодами
def generate_map():
gri = np.zeros((MAP_SIZE, MAP_SIZE), dtype=int)
num_obstacles = (MAP_SIZE * MAP_SIZE * OBSTACLE_PERCENT) //
100
for _ in range(num_obstacles):
x, y = random.randint(0, MAP_SIZE - 1), random.randint(0, MAP_SIZE - 1)
grid[x, y] = 1
return grid

# Оцінка покриття
def evaluate_fitness(solution, obstacle_map):
coverage = np.zeros((MAP_SIZE, MAP_SIZE), dtype=int)
for x, y in solution:
for dx in range(-SENSOR_RADIUS, SENSOR_RADIUS + 1):
for dy in range(-SENSOR_RADIUS, SENSOR_RADIUS + 1):
nx, ny = x + dx, y + dy
if 0 <= nx < MAP_SIZE and 0 <= ny < MAP_SIZE and obstacle_map[nx, ny]
== 0:
coverage[nx, ny] = 1
return np.sum(coverage)
```

```
        # Генерація початкової популяції
        def generate_population():
        return [
        [(random.randint(0, MAP_SIZE - 1), random.randint(0, MAP_SIZE - 1)) for
_ in range(NUM_SENSORS)]
        for _ in range(POPULATION_SIZE)
        ]

        # Операція схрещування (кросовера)
        def crossover(parent1, parent2):
        point = random.randint(0, NUM_SENSORS - 1)
        child = parent1[:point] + parent2[point:]
        return child

        # Операція мутації
        def mutate(solution):
        if random.random() < MUTATION_RATE:
        index = random.randint(0, NUM_SENSORS - 1)
        solution[index] = (random.randint(0, MAP_SIZE - 1), random.randint(0,
MAP_SIZE - 1))

        return solution

        # Основний алгоритм ГА
        def genetic_algorithm():
        obstacle_map = generate_map()
        population = generate_population()
        best_solution = None
        best_fitness = 0
```

```
for _ in range(GENERATIONS):
population = sorted(population, key=lambda sol: evaluate_fitness(sol, obstacle_map), reverse=True)
if evaluate_fitness(population[0], obstacle_map) > best_fitness:
best_solution = population[0]
best_fitness = evaluate_fitness(best_solution, obstacle_map)

new_population = [population[0]] # Зберігаємо найкраще рішення
while len(new_population) < POPULATION_SIZE:
p1, p2 = random.choices(population[:POPULATION_SIZE // 2], k=2)
child = crossover(p1, p2)
child = mutate(child)
new_population.append(child)
population = new_population
return best_solution, obstacle_map

# Візуалізація результатів
def visualize(solution, obstacle_map):
plt.imshow(obstacle_map, cmap='gray_r')
for x, y in solution:
plt.scatter(y, x, c='red', marker='o')
plt.show()

# Запуск алгоритму
best_solution, obstacle_map = genetic_algorithm()
visualize(best_solution, obstacle_map)
```

ДОДАТОК Б

ЗАТВЕРДЖУЮ
Перший проректор Державного університету
інформаційно-комунікаційних технологій
Член-кореспондент НАН України, доктор технічних наук,
професор, лауреат Державної премії України в галузі науки і
техніки, Заслужений діяч науки і техніки України
_____ Олександр КОРЧЕНКО
"25" лютого 2025р.

АКТ

впровадження в освітній процес Навчально-наукового інституту Інформаційних технологій результатів дисертаційної роботи старшого викладача кафедри Інформаційних систем та технологій Державного університету інформаційно-комунікаційних технологій Жидкої Ольги Валеріївни на тему: «Моделі і алгоритми інформаційної взаємодії для мереж Інтернету речей» на здобуття наукового ступеня доктора філософії за спеціальністю 123 – Комп'ютерна інженерія.

Комісія у складі:

голова – директор Навчально-наукового інституту Інформаційних технологій, доктор технічних наук, професор Нестеренко К.С.; члени комісії: завідувач кафедри Комп'ютерної інженерії, кандидат технічних наук, доцент Лащевська Н.О.; завідувач кафедри Комп'ютерних наук, доктор технічних наук, професор Вишнівський В.В.; завідувач кафедри Інформаційних систем та технологій, доктор технічних наук, професор Сторчак К.П.

розглянула дисертаційну роботу Жидкої Ольги Валеріївни на тему: «Моделі і алгоритми інформаційної взаємодії для мереж Інтернету речей» та публікації автора за матеріалами дисертаційної роботи. Результати впроваджено в освітній процес Навчально-наукового інституту Інформаційних технологій, а саме:

1. Математичну модель доступу в туманних обчисленнях, яка за рахунок

оцінки ймовірнісно-часових характеристик дозволяє підвищити ефективність інформаційної взаємодії в мережах IoT.

2. Ймовірнісну модель встановлення інформаційної взаємодії в мережі Інтернету речей з топологією mesh, яка за рахунок аналізу ймовірнісно-часових характеристик дозволяє оптимізувати процес передачі даних.

На основі аналізу представлених матеріалів комісія встановила, що отримані наукові результати використовуються в освітньому процесі Державного університету інформаційно-комунікаційних технологій при викладанні навчальних дисциплін: «Комп'ютерне моделювання», «Технології Інтернету речей» спеціальності 126 «Інформаційні системи та технології».

Голова комісії

Директор Навчально-наукового інституту
Інформаційних технологій,
доктор технічних наук, професор Катерина НЕСТЕРЕНКО

Члени комісії

Завідувач кафедри Комп'ютерної інженерії,
кандидат технічних наук, доцент Наталія ЛАЩЕВСЬКА

Завідувач кафедри Комп'ютерних наук,
доктор технічних наук, професор Віктор ВИШНІВСЬКИЙ

Завідувач кафедри Інформаційних систем та технологій,
доктор технічних наук, професор Каміла СТОРЧАК

ДОДАТОК В

ТОВ «УКР-ОН»
м. Київ. вул. Харківське шосе, буд. 158, оф.86
ЕДРПОУ 41329131

ДОВІДКА
про впровадження результатів дисертаційної роботи
Жидкої Ольги Валеріївни

Актуальність впровадження наукових розробок обумовлена зростаючими вимогами до ефективності та надійності мереж IoT, особливо в умовах високої динамічності топології та обмежених ресурсів. Запропоновані моделі та методи сприяють підвищенню продуктивності інформаційної взаємодії, оптимізації використання мережевих ресурсів та зменшенню затримок при передачі даних, що є критично важливим для застосувань у сфері індустріального Інтернету речей, розумних міст та інших галузей. Впровадження результатів дослідження дозволить підвищити ефективність функціонування IoT-мереж, зменшити енергоспоживання та підвищити їхню стійкість до змін навантаження та можливих збоїв.

Наукові результати, що впроваджуються, були розроблені в рамках дослідження на тему " Моделі і алгоритми інформаційної взаємодії для мереж Інтернету речей". Запропоновані математичні моделі та метод самоорганізації дозволяють підвищити ефективність передачі даних у мережах IoT, оптимізувати процес маршрутизації та скоротити час пошуку альтернативних маршрутів.

Компанія бере на впровадження **розроблений метод самоорганізації мережі IoT на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, який забезпечує ефективну маршрутизацію та мінімізує затримку передачі даних**. Метод дозволяє збільшити покриття мережі до 96%; зменшити довжину маршруту на 12% у порівнянні з традиційними алгоритмами; зменшити час передачі даних на 9% порівняно з існуючими методами.

Цінність цього дослідження полягає в можливості підвищення надійності мереж, зменшення затримок і раціонального використання ресурсів у середовищах хмарних і туманних обчислень, що є особливо важливим для проєктування та налаштування IoT-інфраструктури.

Директор
ТОВ "УКР-ОН" _____ Гашко А.О

ДОДАТОК Г

ЗАТВЕРДЖУЮ
Заступник директора з юридичних питань
ТОВ "Хуавей Україна"
_____**Роман ЗОРІН**
(підпис, ініціали, прізвище)
«28» листопада **2025 р.**

АКТ
про реалізацію результатів дисертаційної роботи
Жидкої Ольги Валеріївни
на тему: «Моделі і алгоритми інформаційної взаємодії для мереж Інтернету речей»

Результати наукових досліджень Жидкої Ольги Валеріївни, а саме метод самоорганізації мережі Інтернету речей на основі ймовірнісних алгоритмів, заснованих на генетичних підходах, які забезпечують ефективну маршрутизацію та мінімізацію часу на пошук альтернативних маршрутів, випробувано в науково-дослідних і дослідно-конструкторських роботах, що ведуться в ТОВ "Хуавей Україна".

Застосування розроблених Жидкою О.В. математичних моделей й методу самоорганізації мереж IoT дозволять ефективно оптимізувати передачу даних, забезпечувати маршрутизацію та мінімізувати час пошуку альтернативних маршрутів. Це сприятиме підвищенню надійності, зменшенню затримок та оптимізації ресурсів у хмарних і туманних обчисленнях, що особливо корисно при проєктуванні та налаштуванні IoT-мереж.

Заступник директора з юридичних питань
ТОВ "Хуавей Україна" Роман ЗОРІН