

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова
праця
на правах рукопису

ЛЯШЕНКО АНДРІЙ ВОЛОДИМИРОВИЧ

УДК 004.75

ДИСЕРТАЦІЯ

МОДЕЛІ ТА МЕТОДИ АДАПТИВНОЇ МАРШРУТИЗАЦІЇ В ТРАНСПОРТНИХ ІОТ МЕРЕЖАХ

Спеціальність 172 — Телекомунікації та радіотехніка
Галузь знань 17 — Електроніка та телекомунікації

Подається на здобуття наукового ступеня доктора філософії

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

Ляшенко А.В.

Науковий керівник Глоба Лариса Сергіївна, доктор технічних наук,
професор

Київ — 2026

АНОТАЦІЯ

Ляшенко А.В. Моделі та методи адаптивної маршрутизації в транспортних IoT мережах. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 172 – Телекомунікації та радіотехніка. – Навчально-науковий інститут телекомунікаційних систем КПІ ім. Ігоря Сікорського, Київ, 2026.

У дисертаційній роботі вирішено актуальну науково-прикладну задачу управління інформаційними потоками автономних транспортних засобів в інформаційно-комунікаційній IoV-мережі розумного міста.

Сучасна IoV-мережа об'єднує мобільні IoT-вузли, придорожні модулі, вузли попередньої обробки даних, базові станції та хмарні сервіси в єдину інформаційно-комунікаційну систему. У цій системі транспортний засіб виступає не лише об'єктом маршрутизації, а насамперед джерелом інтенсивних потоків V2X-повідомлень, телеметрії, сенсорних даних і службових метаданих.

Для таких потоків розроблено методи адаптивної оцінки поточного QoS-стану сегментів мережі, просторово-часового прогнозування інформаційного навантаження та маршрутизації інформаційних потоків, які генеруються мобільними IoT-вузлами та обслуговуються інфраструктурними елементами IoV-мережі. Стандарти ETSI ITS і 3GPP C-V2X задають для них жорсткі вимоги до затримки доставки та актуальності даних.

Постановка проблеми визначається просторово-часовою нерівномірністю інформаційного навантаження в IoV-мережі. У години пік концентрація автономних транспортних засобів на окремих сегментах мережі призводить до різкого зростання інтенсивності V2X-повідомлень, телеметрії та потоків сенсорних даних, тоді як на інших сегментах зберігається помірний режим роботи. Таке навантаження є нестаціонарним, зашумленим і схильним до швидкого поширення між суміжними ребрами графа, тому для оперативного керування мережею необхідні методи адаптивної оцінки стану інформаційних потоків, їх просторово-часового прогнозування та пошуку оптимального шляху обслуговування.

Аналіз існуючих підходів показав, що класичні методи статисти-

чного усереднення, зокрема пакетна обробка, ковзне вікно та ЕМА з фіксованим коефіцієнтом згладжування, не забезпечують необхідного компромісу між шумостійкістю та швидкістю реакції на сплески інформаційного навантаження. Статичні алгоритми пошуку шляху в часозалежному графі ігнорують очікуваний майбутній стан сегментів мережі, а моделі прогнозування без урахування топології не здатні відтворити поширення навантаження між сусідніми сегментами. Промислові підходи не забезпечують явного узгодження між реактивною оцінкою поточного стану та проактивним прогнозом майбутнього інформаційного навантаження.

Таким чином, актуальною є задача адаптивної маршрутизації інформаційних потоків в умовах динамічно змінного QoS-стану сегментів IoV-мережі, що потребує розробки нових моделей і методів, здатних одночасно оцінювати поточний стан каналу, прогнозувати його зміну та приймати маршрутне рішення в межах часових обмежень, визначених стандартами ETSI ITS.

У роботі задача формалізується у вигляді часозалежної зваженої IoV-мережі, в якій вершини відображають вузли телекомунікаційної інфраструктури (RSU, gNB, хмарні агрегатори), а ребра — сегменти передавання даних між суміжними вузлами мережі. Вага ребра визначається як **час обслуговування інформаційного потоку на сегменті** — тобто час перебування мобільного IoT-вузла в зоні активної генерації V2X-даних. Вимірювані QoS-індикатори стану сегмента визначають адаптивне оновлення оцінки швидкості, на основі якої обчислюється ця вага. Таке подання дозволяє поєднати реактивне оновлення стану за поточними вимірами з прогнозом майбутнього навантаження та процедурою вибору маршруту обслуговування інформаційних потоків. Формальною основою роботи є телекомунікаційна модель IoV-мережі у вигляді часозалежного зваженого графа $G_T = (V, E, W(t), T)$ та модель вхідного потоку даних S_{in} .

У роботі розроблено телекомунікаційну модель IoV-мережі та два взаємопов'язані методи. **Телекомунікаційна модель IoV-мережі** подає мережеву інфраструктуру у вигляді часозалежного зваженого орієнтованого графа $G_T = (V, E, W, T)$, в якому вага ребра визначається

як час обслуговування інформаційного потоку на сегменті. Вимірювані QoS-індикатори стану сегмента — коефіцієнт пропускної здатності η , коефіцієнт завантаження L та коефіцієнт зміни завантаження каналу τ — визначають адаптивне оновлення оцінки швидкості, на основі якої обчислюється ця вага.

Першим є **метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі**. У ньому коефіцієнт згладжування α визначається автоматично за результатами кластеризації QoS-стану сегмента у просторі ознак (η, L, τ) . Для побудови бази нечітких знань використано пакетно-інкрементальну кластеризацію, метрику Махалано-біса, коефіцієнт Бхаттачар'ї та нечіткий висновок Такагі–Сугено. Це забезпечує автоматичне формування правил без експертного налаштування та дозволяє змінювати α від режиму стійкої фільтрації шуму до режиму миттєвої реакції на сплеск інформаційного навантаження.

Другим є **метод просторово-часового прогнозування інформаційного навантаження сегментів IoV-мережі та пошуку оптимального маршруту обслуговування інформаційних потоків**. Прогнозування реалізовано на основі графової нейронної мережі з дифузійною згорткою (DiffusionConv) та блоками GRU, яка враховує топологію мережі та кореляції між суміжними сегментами. Прогноз використовується як часозалежна евристика алгоритму A^* , що оцінює очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку, та забезпечує суттєве скорочення простору пошуку при збереженні оптимальності маршруту.

Розроблені модель і методи утворюють єдиний контур обробки даних. На основі телекомунікаційної моделі IoV-мережі формується часозалежний граф; метод адаптивної оцінки формує згладжену реактивну оцінку поточного QoS-стану сегментів; метод прогнозування генерує короткостроковий прогноз навантаження; алгоритм A^* з часозалежною евристикою визначає маршрут з мінімальним очікуваним сумарним часом обслуговування інформаційного потоку. Завдяки цьому рішення приймається не на основі одиничного зашумленого виміру, а на основі узгодженої реактивно-прогновної оцінки стану IoV-мережі.

Метою дисертаційної роботи є підвищення ефективності управ-

ління інформаційними потоками автономних транспортних засобів в IoV-мережі розумного міста за рахунок розробки телекомунікаційної моделі IoV-мережі та методів автоматичної оцінки поточного QoS-стану сегментів мережі, їх просторово-часового прогнозування та пошуку оптимального маршруту обслуговування на основі модифікованого алгоритму A^* , що забезпечують скорочення часу реакції системи на зміну інформаційного навантаження та зменшення обчислювальних витрат при маршрутизації.

Для досягнення мети у дисертації поставлено та вирішено сім задач: проведено аналіз архітектури IoV-мереж та особливостей формування інформаційних потоків; досліджено обмеження існуючих методів оцінки стану інформаційного навантаження; розроблено телекомунікаційну модель IoV-мережі; розроблено метод адаптивної оцінки стану; розроблено метод просторово-часового прогнозування та удосконалено метод пошуку оптимального маршруту; створено програмний комплекс та проведено експериментальні дослідження на імітаційному стенді IoV-мережі м. Ірпінь та наборі даних METR-LA.

Об'єктом дослідження є процеси формування, передавання та управління інформаційними потоками автономних транспортних засобів в інформаційно-комунікаційній IoV-мережі розумного міста.

Предметом дослідження є методи адаптивної оцінки стану, просторово-часового прогнозування та реактивно-проактивної маршрутизації інформаційних потоків у часозалежній IoV-мережі.

Для розв'язання поставлених задач застосовано методи теорії графів для побудови часозалежної моделі IoV-мережі та реалізації алгоритмів пошуку шляху; методи машинного навчання та штучних нейронних мереж для просторово-часового прогнозування показника стану інформаційного потоку; методи теорії нечітких множин для формування адаптивної оцінки; методи імітаційного моделювання для побудови експериментального стенду IoV-мережі на основі реальної топології та синтетичних інформаційних потоків; методи інженерії програмного забезпечення для створення мікросервісної архітектури та структур даних в оперативній пам'яті.

Експериментальну перевірку виконано у двох різних, але вза-

ємопов'язаних середовищах. Перше — імітаційний стенд ІoV-мережі м. Ірпінь, побудований за реальною топологією дорожнього графа та синтетичними сценаріями навантаження. Він призначений для дослідження локальних перевантажень, інцидентів і пікової інтенсивності. Друге — еталонний набір даних METR-LA, що дає змогу зіставити результати із сучасними роботами у галузі просторово-часового прогнозування. Програмну реалізацію виконано як мікросервісну систему з Apache Kafka, CSR-графом в оперативній пам'яті, сервісом TGNN-прогнозування та сервісом маршрутизації реального часу.

Експериментальні результати підтвердили ефективність розроблених методів. Метод адаптивної оцінки стану інформаційних потоків демонструє зростання помилки лише у **1,2 рази** при масових сплесках навантаження проти **2,7 рази** для ковзного вікна, а на бенчмарку METR-LA — час реакції близько **9 с** проти **171 с** та шумостійкість **+10%** при $\sigma = 3$ mph. Компактна модель TGNN містить лише **205 тис. параметрів**, що на **32%** менше, ніж у DCRNN, і забезпечує **24%** виграв за MAE (**2,10** проти **2,77** mph). У наборі METR-LA ця метрика вимірюється в милях на годину, а швидкість v_{seg} виступає фізичним індикатором стану інформаційного навантаження сегмента. Метод пошуку оптимального шляху скорочує простір пошуку на **56–68%** при емпірично зафіксованій втраті оптимальності не більше **0,00–0,08%**. Навантажувальне тестування (**583 повід./с**, **363 000** повідомлень) показало, що програмний комплекс обробляє **96,6%** потоку із затримкою **1,19 мс**, тоді як підхід на основі повного перерахунку з БД обробляє лише **27,2%**.

Наукова новизна одержаних результатів полягає в тому, що:

1. Вперше розроблено телекомунікаційну модель ІoV-мережі у вигляді часозалежного зваженого орієнтованого графа, в якій, на відміну від класичних графових моделей маршрутизації, вага ребра визначається як час обслуговування інформаційного потоку на сегменті та формується на основі вимірюваних QoS-індикаторів стану сегмента, що забезпечує формальну основу для задачі маршрутизації інформаційних потоків з урахуванням динаміки телекомунікаційного навантаження;

2. Отримало подальшого розвитку метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі автономних транспортних засобів, в якому, на відміну від традиційних підходів з експертним заданням бази нечітких знань, реалізовано автоматичне формування бази правил за результатами кластеризації QoS-стану сегментів з використанням метрик Махаланобіса та коефіцієнта Бхаттачар'ї і нечіткого висновку Такагі–Сугено, що забезпечує миттєву реакцію на зміну інтенсивності інформаційних потоків без участі експерта;
3. Удосконалено метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі алгоритму A^* за рахунок використання часозалежної прогнозової евристики на основі розробленого методу прогнозування, що оцінює очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку, що забезпечує суттєве скорочення простору пошуку при збереженні оптимальності маршруту та виконання пошуку в межах часових обмежень, визначених стандартами ETSI ITS.

Практичне значення одержаних результатів полягає у створенні програмного комплексу управління інформаційними потоками в IoV-мережі на основі Apache Kafka, In-Memory CSR-графа, сервісів адаптивної оцінки, TGNN-прогнозування та пошуку оптимального шляху. Розроблені методи й програмні засоби можуть бути використані для задач обробки високочастотної телеметрії та підтримки рішень у реальному часі.

Ключові слова: ІНТЕРНЕТ РЕЧЕЙ (IOT), INTERNET OF VEHICLES (IOV), АДАПТИВНА МАРШРУТИЗАЦІЯ, НЕЙРОННІ МЕРЕЖІ, НЕЧІТКА ЛОГІКА, ТЕЛЕКОМУНІКАЦІЙНІ МЕРЕЖІ, УПРАВЛІННЯ ІНФОРМАЦІЙНИМИ ПОТОКАМИ, ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ, ПРОСТОРОВО-ЧАСОВЕ ПРОГНОЗУВАННЯ, РОЗУМНЕ МІСТО.

ABSTRACT

A.V. Liashenko Models and Methods of Adaptive Routing in Transport IoT Networks. – Qualifying scientific work on manuscript rights.

Thesis for the degree of Doctor of Philosophy in specialty 172 – Telecommunications and Radio Engineering. – Educational and Scientific Institute of Telecommunication Systems of Igor Sikorsky KPI, Kyiv, 2026.

The dissertation addresses an urgent scientific and practical problem of managing information flows of autonomous vehicles in a smart-city Internet of Vehicles (IoV) network. The developed methods provide adaptive estimation of the current QoS state of each network segment, spatio-temporal forecasting of information load, and routing of information flows generated by mobile IoT sources and served by IoV infrastructure.

The problem statement is driven by the spatial and temporal unevenness of information load in IoV networks. During peak hours, the concentration of connected vehicles on specific network segments causes a sharp growth of V2X messages, telemetry packets, and sensor-data streams, while other segments remain moderately loaded. This load is non-stationary, noisy, and propagates through adjacent edges of the graph. Therefore, efficient operation of the network requires adaptive estimation of the current state of information flows, forecasting of future load distribution, and search for an optimal service path.

The analysis of existing approaches showed that batch processing, sliding-window methods, and EMA with a fixed smoothing coefficient do not provide the required balance between noise suppression and fast response to bursts of information load. Static path-search algorithms ignore the future state of network segments, while forecasting models without graph topology cannot represent load propagation between neighboring edges. Industrial solutions do not explicitly synchronize reactive estimation with proactive forecasting of future network conditions.

Thus, the problem of adaptive routing of information flows under dynamically changing QoS state of IoV network segments is relevant, requiring new models and methods capable of simultaneously estimating the current channel state, forecasting its change, and making routing decisions within the time constraints defined by ETSI ITS standards.

In the dissertation, the problem is formalized as a time-dependent

weighted IoV network in which vertices represent telecommunication infrastructure nodes (RSU, gNB, cloud aggregators), whereas edges correspond to data-transfer segments between adjacent nodes. The edge weight is defined as the **service time of the information flow on the segment** — that is, the time an OBU spends in the zone of active V2X data generation — and is formed on the basis of measurable QoS indicators of the segment state. Such a representation makes it possible to combine reactive state updating from streaming measurements with future-load forecasting and route selection for information-flow servicing.

The dissertation develops a telecommunication model of the IoV network and two interconnected methods. The **telecommunication model of the IoV network** represents the infrastructure as a time-dependent weighted directed graph $G_T = (V, E, W, T)$, in which the edge weight is defined as the service time of the information flow on the segment and is formed on the basis of measurable QoS indicators of the segment state — the channel capacity coefficient η , the channel load coefficient L , and the channel load change coefficient τ .

The first method is the **method of adaptive state estimation of information flows on IoV network segments**. In this method, the smoothing coefficient α is determined automatically from clustering results of the QoS state of segments in the three-dimensional feature space (η, L, τ) . The method uses batch-incremental clustering, Mahalanobis distance, the Bhattacharyya coefficient, and Takagi–Sugeno fuzzy inference to build the knowledge base automatically and to switch between stable noise filtering and immediate reaction to load bursts.

The second method is the **method of spatio-temporal forecasting of information load on IoV network segments and optimal service route search**. Forecasting is implemented using a graph neural network with Diffusion Convolution and GRU blocks that account for network topology and inter-segment correlations. The forecast is used as a time-dependent heuristic for the A* algorithm that evaluates the expected QoS state of segments at the moment of flow passage through them, providing significant reduction of the search space while preserving route optimality.

The developed model and methods form a unified data-processing

loop: the telecommunication model provides the time-dependent graph; the adaptive estimation method produces a smoothed reactive estimate of the current QoS state; the forecasting method generates a short-term load forecast; and the A* algorithm with the time-dependent heuristic determines the route with the minimum expected total service time. As a result, decisions are made not from a single noisy measurement but from a consistent reactive-predictive estimate of IoV-network state.

The aim of the dissertation is to improve the efficiency of information flow management for autonomous vehicles in a smart-city IoV network by developing a telecommunication model of the IoV network and methods for automatic estimation of the current QoS state of network segments, their spatio-temporal forecasting, and optimal service route search based on a modified A* algorithm, providing reduced system reaction time to changes in information load and lower computational cost of routing.

The dissertation solves seven tasks: analysis of IoV architecture and information-flow formation; identification of limitations of existing state-estimation methods; development of the telecommunication model of the IoV network; development of the adaptive estimation method; development of the TGNN-based forecasting method and improvement of the optimal-path search method; and implementation plus experimental validation on the Irpin IoV network simulation testbed and the METR-LA benchmark.

The **object of research** is the processes of formation, transmission, and management of information flows of autonomous vehicles in a smart-city information and communication IoV network.

The **subject of research** is methods of adaptive state estimation, spatio-temporal forecasting, and reactive-proactive routing of information flows in a time-dependent IoV network.

To solve the stated tasks, the work applies graph-theory methods for time-dependent IoV network modelling and path-search algorithms; machine learning and neural-network methods for spatio-temporal prediction of information flow state indicators; fuzzy-set methods for adaptive state estimation; simulation methods for building an experimental IoV network testbed based on real topology and synthetic information flows; and software-engineering methods for microservice architecture and in-memory

data structures.

Experimental validation was conducted in two complementary environments. The first one is the Irpin IoV network simulation testbed, built from the real road-graph topology and synthetic load scenarios and intended for studying local overloads, incidents, and peak-intensity scenarios. The second one is the benchmark METR-LA dataset, which enables direct comparison with modern spatio-temporal forecasting studies. The software implementation was developed as a microservice system based on Apache Kafka, an in-memory CSR graph, a TGNN forecasting service, and a routing service operating in real time.

Experimental results confirm the effectiveness of the proposed methods. The adaptive state-estimation method increases its error only by **1.2 times** during mass load bursts versus **2.7 times** for the sliding-window baseline, and on the METR-LA benchmark the reaction time is about **9 s** versus **171 s**. The compact TGNN model has only **205 k parameters**, which is **32%** fewer than DCRNN, and improves MAE by **24%** (**2.10** vs **2.77** mph). The optimal-path search method reduces the search space by **56–68%** with empirically observed optimality loss not exceeding **0.00–0.08%**. Under a high-intensity stream (**583 msg/s**, **363,000** messages), the software complex processes **96.6%** of the flow with **1.19 ms** latency, whereas the full DB recalculation approach processes only **27.2%**.

The obtained results show that the practical effect of the proposed approach lies not only in reducing estimation error and shrinking the search space, but also in improving the stability of IoV-system operation in real time. Faster reaction to load bursts makes it possible to detect the risk of segment overload earlier, while reducing the number of expanded vertices lowers computational costs under mass request servicing. Therefore, the proposed solutions are applicable to decision-support services, urban mobility monitoring platforms, and streaming telemetry management systems for connected and autonomous vehicles.

The scientific novelty of the obtained results consists in the following:

1. for the first time, a telecommunication model of the IoV network in the form of a time-dependent weighted directed graph has been developed, in which, unlike classical graph routing models, the edge weight is defined

as the service time of the information flow on the segment and is formed on the basis of measurable QoS indicators of the segment state, providing a formal basis for the problem of routing information flows with account for telecommunication load dynamics;

2. the method of adaptive state estimation of information flows on IoV network segments of autonomous vehicles has been further developed, in which, unlike traditional approaches with expert-defined fuzzy knowledge bases, automatic generation of the rule base is implemented from clustering results of the QoS state of segments using Mahalanobis distance, the Bhattacharyya coefficient, and Takagi–Sugeno fuzzy inference, ensuring immediate response to changes in information flow intensity without expert involvement;
3. the method of optimal service route search for information flows based on the A* algorithm has been improved by using a time-dependent predictive heuristic based on the developed forecasting method that evaluates the expected QoS state of segments at the moment of flow passage through them, ensuring significant reduction of the search space while preserving route optimality and search execution within time constraints defined by ETSI ITS standards.

The practical significance of the results lies in the creation of a software complex for information-flow management in an IoV network based on Apache Kafka, an in-memory CSR graph, adaptive estimation services, TGNN forecasting, and optimal path search. The developed methods and software tools can be used for high-frequency telemetry processing and real-time decision support.

From the viewpoint of telecommunication-system design, the proposed approach is important because it transforms heterogeneous V2X telemetry, service messages, and sensor reports into a unified state representation that can be used for online control. This reduces the semantic gap between data acquisition, stream processing, forecasting, and routing decisions. In contrast to isolated optimization modules, the developed solution preserves continuity between monitoring and control layers and thereby supports more reliable operation of IoV services under rapidly changing urban conditions.

Another important result is the reproducibility and transferability of the developed solution. The software complex was implemented as a modular set of interoperable services with explicit data contracts and can be adapted to different city topologies, traffic densities, and deployment scales without changing the core logic of the proposed methods. This creates the basis for further application of the obtained results in urban mobility platforms, traffic-management centers, and intelligent transport telematics services that require stable real-time processing of distributed telemetry.

Keywords: INTERNET OF THINGS (IOT), INTERNET OF VEHICLES (IOV), ADAPTIVE ROUTING, NEURAL NETWORKS, FUZZY LOGIC, TELECOMMUNICATION NETWORKS, INFORMATION FLOW MANAGEMENT, SIMULATION, SPATIO-TEMPORAL FORECASTING, SMART CITY.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. Liashenko A., Globa L. Accelerating A^* algorithm based search in time-dependent graphs with learned heuristics // *Radioelectronic and Computer Systems*. — 2026. — Vol. 2026, no. 1. — P. 179–191. — ISSN: 1814-4225. — E-ISSN: 2663-2012. — DOI: 10.32620/reks.2026.1.12. (Scopus, Q3).

Особистий внесок здобувача: сформульовано задачу, розроблено алгоритм прискореного пошуку A^* з навченою евристикою, проведено обчислювальні експерименти, написано статтю. **Внесок співавтора Л. Глоби:** визначено напрям дослідження та здійснено наукове рецензування рукопису.

2. Globa L., Liashenko A. Developing the Fuzzy Logic Rules Based on Clustering Algorithms for Data Analysis in the IoT Network // *Lect. Notes Electr. Eng.*, vol. 1198. — Cham : Springer, 2024. — P. 782–803. — ISSN: 1876-1100. — E-ISSN: 1876-1119. — DOI: 10.1007/978-3-031-61221-3_38. (Scopus).

Особистий внесок здобувача: розроблено метод побудови нечітких логічних правил на основі кластеризації, реалізовано програмний прототип, підготовлено рукопис. **Внесок співавтора Л. Глоби:** визначено постановку задачі та здійснено редагування тексту.

3. Globa L., Novogrudska R., Liashenko A. The clustering and fuzzy logic methods complex for Big Data processing // *Proceedings of International Conference on Applied Innovation in IT (ICAIIIT)*. — Anhalt University of Applied Sciences, 2022. — Vol. 10, Issue 1. — ISSN: 2199-8876. — P. 69–79. — DOI: 10.25673/76934. (Scopus).

Особистий внесок здобувача: розроблено метод комплексного застосування кластеризації та нечіткої логіки, проведено експерименти на реальних даних. **Внесок співавтора Л. Глоби:** визначено концепцію роботи. **Внесок співавтора Р. Новогрудської:** участь в аналізі та інтерпретації результатів.

4. Liashenko A., Globa L. A Comprehensive Method for Anomaly Detection in Complex Dynamic IoT Systems // *Proceedings of International Conference on Applied Innovation in IT (ICAIIIT)*. — Anhalt University of Applied Sciences, 2025. — Vol. 13, Issue 1. — P. 101–107. — ISSN: 2199-8876. — ISBN: 978-3-96057-182-7. — DOI: 10.25673/119221. (Scopus).

Особистий внесок здобувача: розроблено метод виявлення аномалій у динамічних IoT-системах, проведено валідацію на реальних наборах даних, написано статтю. **Внесок співавтора Л. Глоби:** здійснено наукове рецензування та редагування.

5. Liashenko A., Globa L. Comparative Analysis of Consumer Strategies for Real-Time Traffic Graph Updates in IoT Systems // *Proceedings of International Conference on Applied Innovation in IT (ICAIIIT)*. — Anhalt University of Applied Sciences, 2025. — Vol. 13, Issue 5. — P. 151–159. — ISSN: 2199-8876. — DOI: 10.25673/122849. (Scopus).

Особистий внесок здобувача: розроблено та порівняно стратегії споживачів для оновлення графа в реальному часі, підготовлено рукопис. **Внесок співавтора Л. Глоби:** здійснено наукове рецензування та редагування.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

6. Liashenko A., Globa L. A Method for Dynamic Graph Weight Adaptation in IoV Networks Using Fuzzy-Based Exponential Smoothing // *2026 IEEE 18th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*. — Lviv, Ukraine, 2026. — P. 1–6. — DOI: 10.1109/TCSET65181.2026.11461094. (Scopus).

Особистий внесок здобувача: розроблено метод динамічної адаптації ваг графа на основі нечіткого експоненційного згладжування, проведено апробацію на реальних траєкторіях. **Внесок співавтора Л. Глоби:** здійснено наукове рецензування.

7. Globa L., Bugayenko Yu., Liashenko A., Grebinichenko M. Analysis of clustering algorithms for use in the universal data processing system // *Proceedings of the 10th Int. Sci.-Tech. Conf. on Ontology-Based Intelligent Systems (OSTIS-2020)*. — Minsk : BSUIR, 2020. — ISBN: 978-985-543-464-5. — С. 101–104.

Особистий внесок здобувача: проведено аналіз алгоритмів кластеризації, підготовлено тези. **Внесок співавторів Л. Глоби та Ю. Бугаєнка:** визначено постановку задачі. **Внесок співавтора М. Гребініченко:** участь в обговоренні результатів.

8. Ляшенко А.В. Підхід для побудови нечітких логічних правил для великих даних // *Матеріали XIV Міжнар. наук.-техн. конф. «Перспективи телекомунікацій 2020»*. — Київ : КПП ім. Ігоря Сікорського, 2020. — ISSN: 2663-502X, E-ISSN: 2664-3057. — С. 247. — URL: <https://conferenc-journal.its.kpi.ua/article/view/201702>
9. Бугаєнко Ю.М., Ляшенко А.В. Метод кластеризації для обробки великих обсягів даних // *Матеріали XIII Міжнар. наук.-техн. конф. «Перспективи телекомунікацій»*. — Київ : КПП ім. Ігоря Сікорського, 2019. — ISSN: 2663-502X. — С. 37–39.

Особистий внесок здобувача: розроблено метод кластеризації, реалізовано програмний прототип. **Внесок співавтора Ю. Бугаєнка:** визначено постановку задачі та участь в обговоренні результатів.

10. Globa L., Bugayenko Yu., Ishchenko I., Liashenko A. Approach to determining the number of clusters in a data set // *Proceedings of the 9th Int. Sci.-Tech. Conf. on Ontology-Based Intelligent Systems (OSTIS-2019)*. — Minsk : BSUIR, 2019. — ISBN: 978-985-543-427-0.

Особистий внесок здобувача: розроблено алгоритм визначення кількості кластерів. **Внесок співавторів Л. Глоби та Ю. Бугаєнка:** визначено постановку задачі. **Внесок співавтора І. Іщенка:** участь в обговоренні результатів.

11. Бугаєнко Ю.М., Ляшенко А.В. Архітектурне рішення щодо системи аналізу даних обчислювальних процесів // *Матеріали XV Міжнар. наук.-техн. конф. «Перспективи телекомунікацій»*. — Київ : КПІ ім. Ігоря Сікорського, 2021. — ISSN: 2663-502X, E-ISSN: 2664-3057. — С. 218–221.

Особистий внесок здобувача: розроблено архітектурне рішення, підготовлено тези. **Внесок співавтора Ю. Бугаєнка:** визначено вимоги до системи та участь в обговоренні результатів.

12. Ляшенко А.В. Адаптивна маршрутизація в ІoV-мережах на основі часо-просторової графової нейронної мережі // *Матеріали XX Міжнар. наук.-техн. конф. «Перспективи телекомунікацій» (ПТ-2026)*. — Київ : КПІ ім. Ігоря Сікорського, 2026. — ISSN: 2663-502X, E-ISSN: 2664-3057. — С. 349.

Наукові праці, які відображають окремі наукові результати дисертації:

13. Глоба Л.С., Щелест Є.В., Ляшенко А.В. Підхід щодо отримання нечітких логічних правил із набору статистичних даних // *Інформ.-комунік. технології та сталий розвиток : матеріали XXI Міжнар. конф. (14–16.11.2022)*. — Київ : ТОВ «Юстон», 2022. — С. 36–38.

Особистий внесок здобувача: розроблено підхід, підготовлено текст розділу. **Внесок співавтора Л. Глоби:** визначено напрям дослідження. **Внесок співавтора Є. Щелеста:** участь в аналізі результатів.

ЗМІСТ

АНОТАЦІЯ.....	2
ABSTRACT.....	8
СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ.....	13
ЗМІСТ	17
СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ	21
ВСТУП.....	24
РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ УПРАВЛІННЯ ІНФОРМАЦІЙНИМИ ПОТОКАМИ В ІОВ-МЕРЕЖАХ.....	31
1.1 Архітектура Іов-мережі як інформаційно-комунікаційної системи	31
1.2 Особливості функціонування Іов-мереж в умовах високого інформаційного навантаження та стохастичності трафіку.....	37
1.2.1 Особливості формування інформаційних потоків в Іов-мережах	37
1.2.2 Проблема затримок та актуальності даних у навігаційних системах реального часу	41
1.2.3 Обчислювальні обмеження та вимоги до алгоритмічної складності в системах реального часу	43
1.2.4 Технології потокової обробки даних для Іов-систем	44
1.3 Аналіз методів оцінки стану інформаційних потоків в Іов-мережах.....	46
1.3.1 Обмеження систем пакетної обробки та методів ковзного вікна	47
1.3.2 Проблема ідентифікації режимів руху та дрейфу концепції у вхідному потоці.....	49
1.3.3 Аналіз підходів промислових навігаційних сервісів до оцінки стану сегментів Іов-мережі	52
1.4 Аналіз методів маршрутизації інформаційних потоків від мобільних Іот-джерел	55
1.4.1 Ефективність класичних алгоритмів пошуку (Dijkstra, A*) в умовах змінного середовища	56

1.4.2	Проблеми застосування статичних евристик та методів прямого прогнозування часу прибуття (ETA)	58
1.4.3	Аналіз методів машинного навчання для прогнозування параметрів маршруту	60
1.5	Аналіз методів прийняття рішень та обробки даних в умовах невизначеності	68
1.5.1	Доцільність застосування апарату нечіткої логіки для адаптивного керування параметрами системи	69
1.5.2	Порівняльний аналіз алгоритмів кластеризації для автоматичної ідентифікації режимів руху	72
1.6	Математична постановка задачі управління інформаційними потоками в IoV-мережі	77
1.6.1	Формалізація телекомунікаційної моделі IoV-мережі	78
1.6.2	Модель вхідного потоку даних	78
1.6.3	Постановка оптимізаційної задачі	81
РОЗДІЛ 2 МЕТОД АДАПТИВНОЇ ОЦІНКИ СТАНУ ІНФОРМАЦІЙНИХ ПОТОКІВ НА СЕГМЕНТАХ IOV-МЕРЕЖІ		86
2.1	Телекомунікаційна модель IoV-мережі	87
2.2	Автоматична побудова бази нечітких правил на основі кластеризації	89
2.2.1	Пакетно-інкрементальна кластеризація та геометрична оптимізація кількості станів	96
2.2.2	Керування життєвим циклом патернів на основі статистичних метрик	102
2.3	Нечіткий контролер адаптивного згладжування телеметрії	111
2.3.1	Структура нечіткого виведення типу Такагі-Сугено для згладжування телеметрії	111
2.3.2	Аналіз обчислювальної складності	115

РОЗДІЛ 3 МЕТОД ПРОГНОЗУВАННЯ ІНФОРМАЦІЙНОГО НАВАНТАЖЕННЯ ІОВ-МЕРЕЖІ ТА ПОШУКУ ОПТИМАЛЬНОГО ШЛЯХУ НА ОСНОВІ МОДИФІКОВАНОГО АЛГОРИТМУ A^*	117
3.1 Метод просторово-часового прогнозування інформаційного навантаження на основі TGNN	117
3.1.1 Архітектура прогнозовної моделі TGNN	118
3.1.2 Побудова композитної евристичної функції	127
3.1.3 Кешування прогнозів та оптимізація обчислень	131
3.2 Результати навчання TGNN та порівняння з існуючими моделями	132
3.2.1 Умови навчання та протокол експерименту	132
3.2.2 Результати на тестовій вибірці та порівняння з еталонними моделями	133
3.2.3 Аналіз процесу навчання	135
3.3 Метод пошуку оптимального шляху на основі модифікованого алгоритму A^*	137
3.3.1 Формалізація методу пошуку оптимального шляху	137
3.3.2 Алгоритм пошуку оптимального шляху P^*	138
3.3.3 Вплив TGNN-евристики на простір пошуку	140
РОЗДІЛ 4 РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ СИСТЕМИ АДАПТИВНОЇ МАРШРУТИЗАЦІЇ ІНФОРМАЦІЙНИХ ПОТОКІВ В ІОВ-МЕРЕЖІ	143
4.1 Архітектура програмного комплексу IoT для керування інформаційними потоками	144
4.1.1 Обґрунтування вибору технологічного стеку та протоколів	146
4.1.2 Взаємодія компонентів системи	150
4.1.3 Структури повідомлень для обміну даними	152
4.2 Програмна реалізація ключових мікросервісів	154
4.2.1 Реалізація сервісу кластеризації режимів руху	154
4.2.2 Реалізація сервісу маршрутизації	156
4.2.3 Підсистема навчання моделі TGNN	160

4.3	Експериментальне дослідження ефективності системи	164
4.3.1	Постановка задачі та система метрик	164
4.3.2	Програмно-апаратний стенд та імітаційна модель	165
4.3.3	Результати моделювання та аналіз сценаріїв	168
4.3.4	Веб-інтерфейс та засоби моніторингу.....	185
	ЗАГАЛЬНІ ВИСНОВКИ.....	188
	СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	191
	ДОДАТКИ	208
	Додаток А. Матеріали щодо програмної реалізації та репозиторію проєкту.....	208
	Додаток Б. Довідка про апробацію результатів дисертаційної роботи.	209

СПИСОК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

ACID	Atomicity, Consistency, Isolation, Durability — набір властивостей транзакцій БД
AoI	Age of Information — вік інформації
API	Application Programming Interface — інтерфейс прикладного програмування
ARIMA	AutoRegressive Integrated Moving Average — авторегресійне інтегроване ковзне середнє
CAM	Cooperative Awareness Message — повідомлення про кооперативну поінформованість
CPM	Collective Perception Message — повідомлення колективного сприйняття
CPU	Central Processing Unit — центральний процесор
CSR	Compressed Sparse Row — формат стиснених розріджених рядків
C-V2X	Cellular Vehicle-to-Everything — стільниковий стандарт V2X (3GPP)
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DCRNN	Diffusion Convolutional Recurrent Neural Network — дифузійна згорткова рекурентна нейронна мережа
DENM	Decentralized Environmental Notification Message — повідомлення про події навколишнього середовища
DSRC	Dedicated Short-Range Communications — виділений короткодіапазонний зв'язок (IEEE 802.11p)
eNB	Evolved Node B — базова станція LTE
EMA	Exponential Moving Average — експоненціальне ковзне середнє
ETA	Estimated Time of Arrival — очікуваний час прибуття
ETSI	European Telecommunications Standards Institute — Європейський інститут телекомунікаційних стандартів
FCM	Fuzzy C-Means — нечіткий метод C-середніх
FIFO	First-In, First-Out — черга типу «перший прийшов — перший вийшов»

FWHM	Full Width at Half Maximum — повна ширина на рівні половини висоти
GCN	Graph Convolutional Network — графова згорткова мережа
GNN	Graph Neural Network — графова нейронна мережа
gNB	Next Generation Node B — базова станція 5G NR
GPS	Global Positioning System — глобальна система позиціонування
GPU	Graphics Processing Unit — графічний процесор
GRU	Gated Recurrent Unit — вентильний рекурентний блок
HTTP	Hypertext Transfer Protocol — протокол передачі гіпертексту
IoT	Internet of Things — Інтернет речей
IoV	Internet of Vehicles — Інтернет транспортних засобів
ITS	Intelligent Transportation Systems — інтелектуальні транспортні системи
JSON	JavaScript Object Notation — формат серіалізації даних
LGBM	Light Gradient Boosting Machine — легка градієнтна бустингова модель
LSTM	Long Short-Term Memory — мережа довгої короткочасної пам'яті
MAE	Mean Absolute Error — середня абсолютна помилка
MAPE	Mean Absolute Percentage Error — середня абсолютна відсоткова помилка
MEC	Multi-access Edge Computing — крайові обчислення з множинним доступом
ML	Machine Learning — машинне навчання
MQTT	Message Queuing Telemetry Transport — протокол обміну телеметрією
MSE	Mean Squared Error — середньоквадратична помилка
OBD-II	On-Board Diagnostics II — бортова діагностика другого покоління
OBU	On-Board Unit — бортовий пристрій транспортного засобу
OSRM	Open Source Routing Machine — маршрутизатор з відкритим кодом
QoS	Quality of Service — якість обслуговування

RAM	Random-Access Memory — оперативна пам'ять
RDBMS	Relational Database Management System — СКБД реляційного типу
REST	Representational State Transfer — архітектурний стиль API
RMSE	Root Mean Squared Error — корінь середньоквадратичної помилки
RSU	Road Side Unit — придорожній модуль збирання та ретрансляції даних
SMA	Simple Moving Average — просте ковзне середнє
STGCN	Spatio-Temporal Graph Convolutional Network — просторово-часова графова згорткова мережа
TCP	Transmission Control Protocol — протокол керування передачею
TDG	Time-Dependent Graph — часозалежний граф
TGNN	Temporal Graph Neural Network — часова графова нейронна мережа
V2I	Vehicle-to-Infrastructure — комунікація транспортного засобу з інфраструктурою
V2N	Vehicle-to-Network — комунікація транспортного засобу з мережею
V2V	Vehicle-to-Vehicle — міжтранспортна комунікація
V2X	Vehicle-to-Everything — комунікація транспортного засобу з усім
WAVE	Wireless Access in Vehicular Environments — бездротовий доступ у транспортному середовищі
WCSS	Within-Cluster Sum of Squares — внутрішньокластерна сума квадратів
БД	База даних
ДТП	Дорожньо-транспортна пригода
ПЗ	Програмне забезпечення
ШІ	Штучний інтелект

ВСТУП

Актуальність теми. Функціонування сучасних інтелектуальних транспортних систем у середовищі Smart City дедалі більше залежить від якості управління інформаційними потоками, що генеруються автономними та підключеними транспортними засобами. Сучасна IoV-мережа об'єднує мобільні IoT-вузли, придорожні модулі, вузли попередньої обробки даних, базові станції та хмарні сервіси в єдину інформаційно-комунікаційну систему. У цій системі транспортний засіб виступає не лише об'єктом маршрутизації, а насамперед джерелом інтенсивних інформаційних потоків: V2X-повідомлень, телеметрії, даних камер, лідарів та службових метаданих для колективного сприйняття середовища. Стандарти ETSI ITS (EN 302 637, TS 103 324) та 3GPP Release 16/17 для C-V2X регламентують жорсткі вимоги до затримки доставки (менше 100 мс для DENM) та мінімального віку інформації (AoI), що безпосередньо визначає вимоги до методів оцінки стану і маршрутизації інформаційних потоків у мережі.

Інтенсивність зазначених потоків змінюється нерівномірно в просторі та часі. На магістральних сегментах і поблизу вузлів тяжіння трафіку інформаційне навантаження зростає стрибкоподібно, тоді як на периферії мережі може залишатися помірним. Динаміка навантаження залежить від щільності руху, добового профілю, інцидентів та якості бездротових каналів. Це створює жорсткі вимоги до систем обробки даних: вони мають швидко оцінювати поточний стан інформаційних потоків, прогнозувати його зміни та знаходити маршрут обслуговування з урахуванням очікуваного стану мережі.

Існуючі підходи до оцінки і маршрутизації переважно орієнтовані або на статичний граф, або на локальне усереднення зашумлених вимірів. Такі підходи не враховують просторово-часового поширення інформаційного навантаження між суміжними сегментами мережі та не забезпечують синхронізацію між реактивною обробкою поточних даних і проактивним прогнозом. У результаті система може працювати із застарілою оцінкою стану сегментів, перевантажувати канали зв'язку та втрачати обчислювальні ресурси на обслуговування вже неактуальних маршрутів.

Отже, актуальною є задача адаптивної маршрутизації інформаційних потоків в умовах динамічно змінного QoS-стану сегментів IoV-мережі, що потребує розробки нових моделей і методів, здатних одночасно оцінювати поточний стан каналу, прогнозувати його зміну та приймати маршрутне рішення в межах часових обмежень, визначених стандартами ETSI ITS.

Зв'язок роботи з науковими програмами, планами, темами. Дисертаційна робота виконувалась згідно з планом наукових досліджень Кафедри інформаційно-комунікаційних технологій та систем Навчально-наукового інституту телекомунікаційних систем:

1. В рамках держбюджетної теми: 2218п «Гетерогенна мережа збору, передачі та обробки інформації для системи розподіленої генерації MicroGrid» (номер державної реєстрації 0119U001184).
2. В рамках держбюджетної теми: 2313п «Побудова інформаційно-аналітичної платформи для супроводження функціонування кіберфізичних систем» (номер державної реєстрації 0120U102298).
3. В рамках Міжнародного проекту “Research Guidance System for Organizations Based on Self-improvement Domain Knowledge Map and Customized Industry Knowledge Graph” за співпраці НТУУ «КПІ ім. І. Сікорського» та Інституту інформаційних досліджень академії наук провінції Шандунь, КНР (номер договору 0305/55-M).

Мета і задачі дослідження. Метою дисертаційної роботи є підвищення ефективності управління інформаційними потоками автономних транспортних засобів в IoV-мережі розумного міста за рахунок розробки телекомунікаційної моделі IoV-мережі та методів автоматичної оцінки поточного QoS-стану сегментів мережі, їх просторово-часового прогнозування та пошуку оптимального маршруту обслуговування на основі модифікованого алгоритму A^* , що забезпечують скорочення часу реакції системи на зміну інформаційного навантаження та зменшення обчислювальних витрат при маршрутизації.

Для досягнення мети дослідження було поставлено та вирішено такі задачі:

1. Провести аналіз архітектури IoV-мереж розумного міста як інформаційно-комунікаційних систем, особливостей формування та передавання інформаційних потоків автономних транспортних засобів, існуючих методів оцінки стану інформаційного навантаження та алгоритмів пошуку маршрутів. Виявити обмеження методів статистичного усереднення та недоліки класичних алгоритмів пошуку шляху в умовах динамічної зміни інформаційного навантаження.
2. Розробити метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі на основі автоматичного формування бази нечітких знань за результатами кластеризації режимів інформаційного навантаження.
3. Запропонувати метод просторово-часового прогнозування інформаційного навантаження сегментів IoV-мережі на основі часо-просторової графової нейронної мережі.
4. Розробити метод пошуку оптимального шляху на основі модифікованого алгоритму A^* в IoV-мережі.
5. Розробити архітектуру та програмний комплекс системи адаптивної маршрутизації інформаційних потоків на основі подійно-орієнтованої взаємодії та структур даних в оперативній пам'яті.
6. Провести експериментальні дослідження розроблених моделей і методів та програмного комплексу на експериментальному цифровому двійнику IoV-мережі м. Ірпінь, побудованому за реальною топологією дорожнього графа та синтетичними сценаріями навантаження, та еталонному наборі даних METR-LA.

Об'єкт дослідження – процеси формування, передавання та управління інформаційними потоками автономних транспортних засобів в інформаційно-комунікаційній IoV-мережі розумного міста.

Предмет дослідження – методи адаптивної оцінки стану, просторово-часового прогнозування та реактивно-проактивної маршрутизації інформаційних потоків у часозалежній IoV-мережі.

Методи дослідження, застосовані для вирішення поставлених завдань:

1. Методи теорії графів – для побудови часозалежної моделі IoV-мережі, формалізації топологічних залежностей та реалізації алгоритмів пошуку оптимального шляху.
2. Методи машинного навчання та штучних нейронних мереж (зокрема GNN, TGNN) – для розробки моделей просторово-часового прогнозування показника стану інформаційного потоку та навчання прогнозної евристики.
3. Методи теорії нечітких множин та нечіткої логіки – для класифікації режимів інформаційного навантаження, адаптивної оцінки стану потоків та прийняття рішень при неповноті даних.
4. Методи імітаційного моделювання – для створення експериментального стенду IoV-мережі на основі реальної топології, генерації синтетичних інформаційних потоків та відтворення сценаріїв сплесків інформаційного навантаження.
5. Методи інженерії програмного забезпечення та системного аналізу – для проектування мікросервісної архітектури, розробки структур даних у оперативній пам'яті та оцінки ефективності використання обчислювальних ресурсів.

Наукова новизна отриманих результатів:

1. Вперше розроблено телекомунікаційну модель IoV-мережі у вигляді часозалежного зваженого орієнтованого графа, в якій, на відміну від класичних графових моделей маршрутизації, вага ребра визначається як час обслуговування інформаційного потоку на сегменті та формується на основі вимірюваних QoS-індикаторів стану сегмента, що забезпечує формальну основу для задачі маршрутизації інформаційних потоків з урахуванням динаміки телекомунікаційного навантаження.

2. Отримало подальшого розвитку метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі автономних транспортних засобів, в якому, на відміну від традиційних підходів з експертним заданням бази нечітких знань, реалізовано автоматичне формування бази правил за результатами кластеризації QoS-стану сегментів з використанням метрик Махаланобіса та коефіцієнта Бхаттачар'ї і нечіткого висновку Такагі–Сугено, що забезпечує миттєву реакцію на зміну інтенсивності інформаційних потоків без участі експерта.
3. Удосконалено метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі алгоритму A^* за рахунок використання часозалежної прогнозовної евристики на основі розробленого методу прогнозування, що оцінює очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку, що забезпечує суттєве скорочення простору пошуку при збереженні оптимальності маршруту та виконання пошуку в межах часових обмежень, визначених стандартами ETSI ITS.

Практичне значення одержаних результатів:

1. Усі теоретичні розробки дисертації представлені автором у вигляді методів адаптивної оцінки, прогнозування та маршрутизації інформаційних потоків, які можуть бути безпосередньо використані для підвищення ефективності функціонування IoV-мереж і сервісів реального часу в умовах стохастичного інформаційного навантаження.
2. На основі розроблених моделей та методів створено програмні засоби системи маршрутизації інформаційних потоків (включаючи сервіси TGNN-прогнозування та адаптивної оцінки), архітектура яких базується на структурах даних у оперативній пам'яті та Apache Kafka. Це дозволяє обробляти високочастотні потоки телеметрії (до 50 000 повідомлень/хв) із затримкою рівня мілісекунд.
3. Розроблений програмний комплекс може бути використаний у системах управління інформаційними потоками IoV-мереж на ба-

зі стандартів ETSI ITS та 5G V2X: зокрема, для адаптивного оновлення вагової моделі мережі на MEC-вузлах, маршрутизації телеметричних потоків між RSU та хмарними сервісами, а також моніторингу QoS-стану сегментів у режимі реального часу.

Особистий внесок здобувача. Дисертаційна робота узагальнює результати теоретичних та експериментальних досліджень, проведених автором самостійно. Основні результати, отримані автором особисто: проведено аналіз і порівняльне дослідження методів кластеризації для обробки великих обсягів даних у телекомунікаційних та IoT-системах, а також запропоновано підхід до автоматизованого визначення кількості кластерів у статистичних наборах даних [1, 3, 4, 8]; розроблено метод побудови нечітких логічних правил на основі статистичних даних, що дозволяє автоматизовано формувати базу нечітких знань без участі експерта [2, 6, 7, 13]; запропоновано та досліджено методи обробки поточкових даних у реальному часі для адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі, що забезпечують зменшення затримки обробки та обчислювальних витрат [11]; запропоновано підхід до прискорення алгоритму A^* у часово-залежних графах, що дозволяє зменшити кількість розширених вершин за практично збереженої оптимальності маршруту [10, 31]. Усі експериментальні дослідження, програмна реалізація та аналіз результатів виконані здобувачем особисто.

Апробація результатів дисертації. Основні положення та результати дисертаційної роботи були представлені й одержали схвалення на таких науково-технічних конференціях:

- Міжнародній науково-технічній конференції OSTIS (2019, 2020 рр.).
- XIV та XV Міжнародних науково-технічних конференціях «Перспективи телекомунікацій» (2019, 2020, 2021 рр.).
- XXI Міжнародній науково-практичній конференції «Інформаційно-комунікаційні технології та сталий розвиток» (2022 р.).
- International Conference on Applied Innovations in IT (ICAIIIT 2022).
- International Conference on Applied Innovations in IT (ICAIIIT 2025).

- International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET 2024).
- International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET 2026).
- Міжнародній науково-технічній конференції «Перспективи телекомунікацій» (2026 р.).

Публікації. За результатами досліджень опубліковано 13 наукових праць, у тому числі 1 статтю у науковому фаховому виданні України за спеціальністю 172 — Телекомунікації та радіотехніка, 5 статей у матеріалах міжнародних конференцій, 6 тез доповідей на науково-технічних конференціях та 1 розділ у колективній монографії.

Структура та обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел із 137 найменуваннями та 2 додатками. Робота містить 59 рисунків, 18 таблиць і 36 формул.

РОЗДІЛ 1

АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ УПРАВЛІННЯ ІНФОРМАЦІЙНИМИ ПОТОКАМИ В ІОВ-МЕРЕЖАХ

1.1 Архітектура Іов-мережі як інформаційно-комунікаційної системи

Іов-мережа розумного міста є багаторівневою інформаційно-комунікаційною системою, у якій мобільні транспортні засоби, придорожня інфраструктура, вузли попередньої обробки даних та хмарні сервіси об'єднані єдиним контуром формування, передавання й обробки інформаційних потоків [1, 2, 18, 21, 136]. Керування інформаційними потоками, що виникають на сегментах Іов-мережі, визначає вимоги до пропускної здатності, затримки і обчислювальних ресурсів інфраструктури.

З позицій телекомунікацій доцільно виділяти три функціональні рівні Іов-мережі: рівень сприйняття, мережевий рівень і прикладний рівень. Рівень сприйняття формують мобільні Іот-вузли та сенсори транспортних засобів; мережевий рівень реалізує доставку даних через V2X-канали, RSU, gNB/eNB та вузли попередньої обробки даних; прикладний рівень забезпечує аналітичну обробку, прогнозування стану мережі та підтримку прийняття рішень [2, 21, 136, 137]. Структурну взаємодію цих рівнів наведено на рис. 1.1.

Запропоноване трирівневе подання узгоджується з оглядовими роботами з архітектури Іов та WAVE/C-V2X-систем, у яких окремо виділяються контури сприйняття, мережевої взаємодії та сервісної аналітики [2, 21, 136, 137].

Джерелом проблеми є рівень сприйняття та мережевий рівень Іов-мережі, на яких формується й передається потік телеметрії та V2X-повідомлень. Обробка цих даних і прийняття рішень щодо оцінки стану, прогнозування та маршрутизації виконуються сервісами прикладного рівня програмного комплексу.

У межах цієї архітектури ключовими типами вузлів є: *OBU* (on-board unit) — бортовий модуль автономного транспортного засобу; *RSU* — придорожній вузол збирання й ретрансляції даних; *MEC*-вузол



Рис. 1.1 - Архітектура IoV-мережі як інформаційно-комунікаційної системи

— вузол локальної попередньої обробки даних; *gNB/eNB* — вузол радіодоступу мобільної мережі; *cloud* — хмарний рівень довготривалої аналітики, зберігання та оркестрації сервісів [14, 15, 17, 21, 22, 137]. Сукупність цих вузлів формує єдине інформаційне середовище, де потоки даних мають різний пріоритет, структуру і часові вимоги до доставки.

Типові інформаційні потоки, що генеруються автономними транспортними засобами та обслуговуються інфраструктурою IoV-мережі, наведено в табл. 1.1. Нормативну основу для повідомлень CAM, CPM, DENM та профілів V2X/WAVE і 5G V2X наведено у [129–135, 137].

Наведені типи потоків відрізняються за обсягом, частотою генерації, вимогами до затримки та допустимого рівня втрати пакетів. Тому задача управління інформаційними потоками в IoV-мережі не зводиться до одного показника швидкості руху. Вона охоплює оцінку поточного режиму інформаційного навантаження, виявлення критичних сегментів і перенаправлення потоків з урахуванням доступних телекомунікаційних та обчислювальних ресурсів.

Вимоги до характеристик цих потоків регламентуються стандартами **ETSI ITS** (Intelligent Transport Systems) — серією телекомунікаційних стандартів Європейського інституту телекомунікаційних стандартів, що визначають протоколи, формати повідомлень та параметри QoS для V2X-комунікацій [132, 133, 134]. Зокрема, для DENM-повідомлень (аварійні

Таблиця 1.1 - Типи інформаційних потоків CAV в IoV-мережі

Тип потоку	Стандарт / джерело	Призначення	Типові вимоги
CAM	ETSI EN 302 637-2	Періодичне повідомлення про положення, швидкість, напрямок та стан CAV	Низька затримка, висока регулярність доставки
CPM	ETSI TS 103 324	Кооперативне сприйняття середовища, передавання об'єктів і подій від сенсорів авто	Висока пропускна здатність, стійкість до пікових сплесків
DENM	ETSI EN 302 637-3	Події та попередження про небезпеку, аварії, обмеження руху	Пріоритетна доставка, мінімальний Age of Information
Сирі сенсорні дані	Камери, лідари, телеметрія OBU	Локальна аналітика, побудова карти оточення, навчання моделей	Високий обсяг даних, потреба в попередній локальній обробці

події) встановлено вимогу на затримку доставки не більше 100 мс [133], для CAM — інтервал передачі 100–1000 мс залежно від режиму руху [132]. Ці часові обмеження безпосередньо визначають вимоги до швидкодії методів оцінки стану сегментів мережі та алгоритмів маршрутизації інформаційних потоків.

Просторово-часову нерівномірність формування навантаження в реальній міській мережі показано на рис. 1.2.

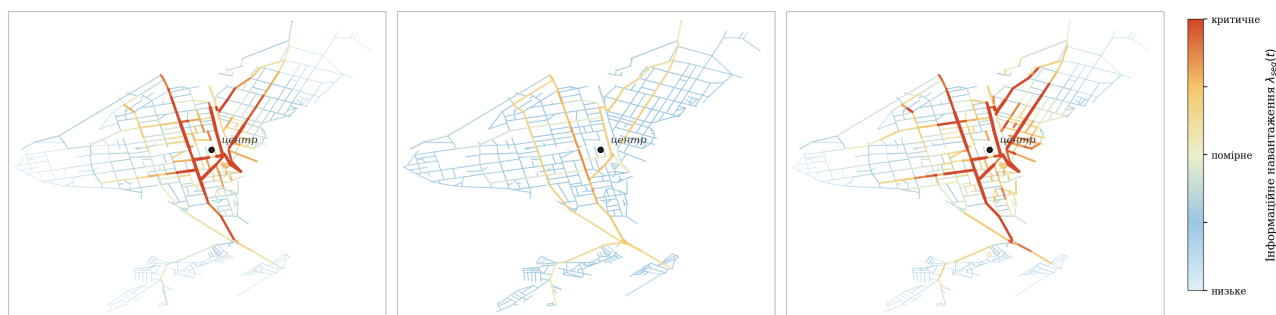


Рис. 1.2 - Просторово-часовий розподіл інформаційного навантаження в IoV-мережі м. Ірпінь: а — 08:00, ранкова година пік; б — 14:00, денний рівноважний режим; в — 18:00, вечірня година пік. У ранковий період максимуми $\lambda_{\text{seg}}(t)$ концентруються вздовж центральних магістралей, удень навантаження переходить у помірний збалансований стан, а ввечері зміщується на радіальні виходи з центру міста.

Рис. 1.2 демонструє, що інформаційне навантаження формується топологією міста, добовими патернами переміщення та концентрацією

мобільних IoT-джерел. Саме така просторово- часова нерівномірність унеможливорює використання статичних методів обробки і вимагає адаптивної оцінки та прогнозування стану мережі [1, 21, 136].

Для розуміння механізму передачі даних та керування інформаційними потоками в IoV-мережі необхідно розглянути повний контур взаємодії між компонентами системи. На рис. 1.3 наведено ілюстративну схему архітектури IoV-мережі з двома сегментами, що відображає основні типи вузлів, протоколи зв'язку та замкнений контур керування.

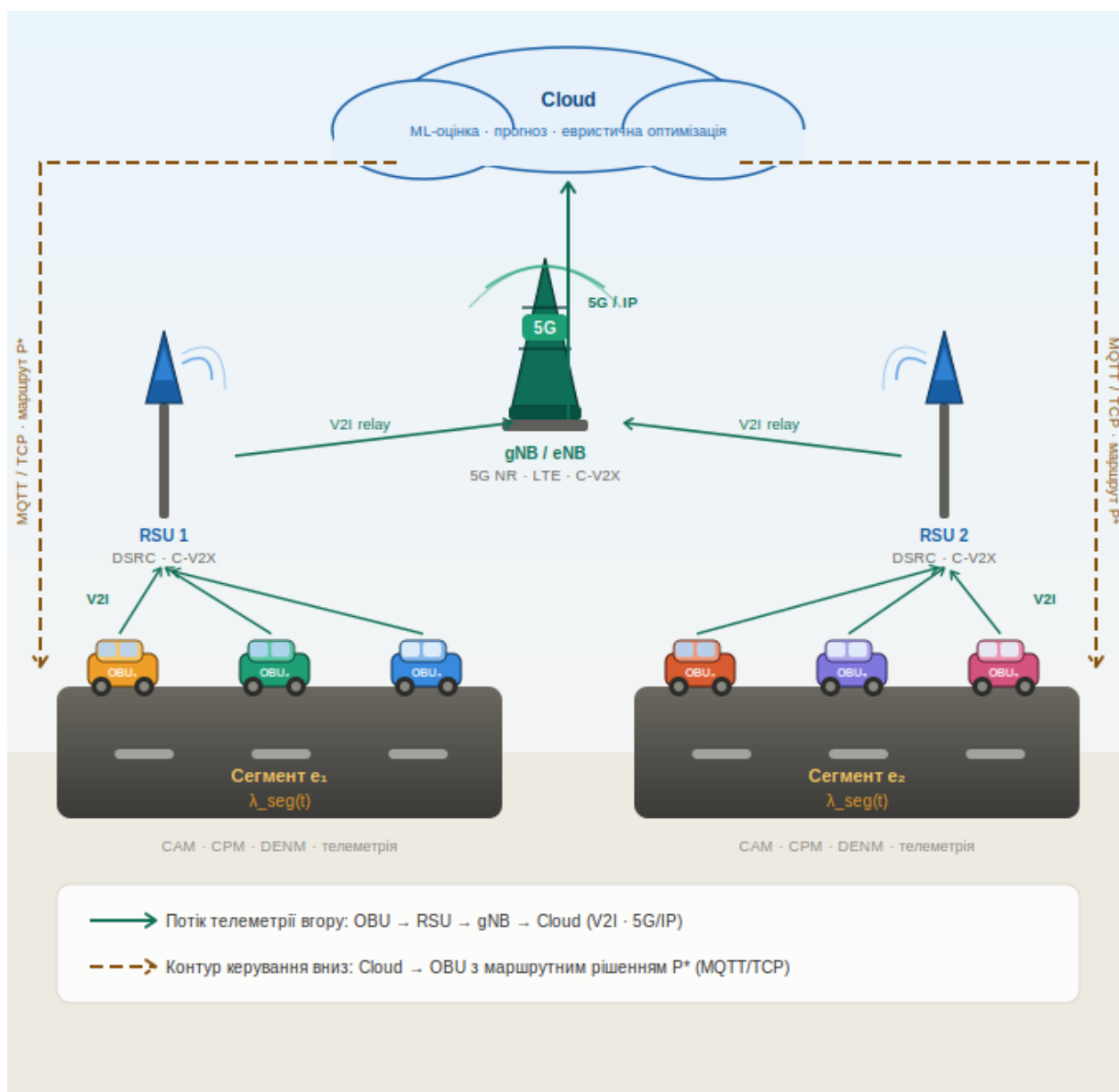


Рис. 1.3 - Архітектура IoV-мережі: передача даних та контур керування інформаційними потоками. Зелені стрілки — потік телеметрії від OBU до хмарного рівня (V2I, V2I relay, 5G/IP); помаранчеві пунктирні стрілки — контур керування від хмарного рівня до OBU з маршрутним рішенням P^* (MQTT/TCP)

Передача даних у IoV-мережі відбувається за трирівневою схемою і охоплює два напрямки: *висхідний потік телеметрії* від транспортних засобів до хмарного рівня та *низхідний контур керування* від хмарного рівня назад до транспортних засобів.

Висхідний потік ($\text{OBU} \rightarrow \text{RSU} \rightarrow \text{gNB} \rightarrow \text{Cloud}$). Кожен автономний транспортний засіб оснащений бортовим модулем OBU (On-Board Unit), який безперервно генерує V2X-повідомлення: *CAM* (Cooperative Awareness Message) з частотою 1–10 Гц, *CPM* (Collective Perception Message) та *DENM* (Decentralized Environmental Notification Message) для екстрених подій [132, 133, 134]. Ці повідомлення передаються до придорожного модуля RSU (Road Side Unit) по радіоінтерфейсу *V2I* (Vehicle-to-Infrastructure) на основі одного з двох стандартів: DSRC (Dedicated Short-Range Communications, IEEE 802.11p/WAVE) на частоті 5.9 ГГц або C-V2X (Cellular V2X, 3GPP Release 14+) через інтерфейс PC5. RSU виступає вузлом першого рівня агрегування: він збирає телеметрію від усіх OBU свого сегмента, формує вектор QoS-стану $[\eta, L, \tau]$ та передає дані далі до базової станції gNB/eNB по інтерфейсу X2/Xn [135] або безпосередньо через стек TCP/IP у хмарний рівень по каналу 5G NR (New Radio) або LTE з використанням протоколу MQTT-SN [130] чи REST API (HTTP/2) поверх TLS.

Хмарний рівень (Cloud). Після надходження телеметрії до хмарного рівня виконується комплексна обробка інформаційних потоків [2, 21, 136]. Сучасні IoV-платформи застосовують на цьому рівні методи машинного навчання для адаптивної оцінки поточного QoS-стану каналів, виявлення аномалій та сплесків навантаження, а також нечітку логіку для прийняття рішень в умовах неповноти і зашумленості вхідних даних [14, 15]. Для короткострокового прогнозування розподілу навантаження між сегментами застосовуються методи просторово-часового аналізу на основі графових нейронних мереж, що враховують топологію мережі та кореляції між сусідніми сегментами [39, 40, 41]. Оптимальний маршрут обслуговування інформаційного потоку P^* визначається евристичними методами пошуку на графі з урахуванням прогнозованого стану сегментів на момент проходження потоку через них [29, 30]. Всі обчислення виконуються в режимі реального часу з урахуванням часових обмежень

стандартів ETSI ITS [132, 133].

Низхідний контур керування (Cloud $\rightarrow$ OBU). Сформований маршрут P^* (послідовність вузлів графа G_T) доставляється назад до OBU через брокер повідомлень за протоколом *MQTT* (Message Queuing Telemetry Transport, ISO/IEC 20922) або *TCP/IP* з використанням власного бінарного протоколу на основі Protocol Buffers (protobuf). MQTT обрано як основний протокол зворотного каналу завдяки його малому заголовку (фіксовані 2 байти), підтримці QoS-рівнів (at-least-once, exactly-once) та нативній асинхронності [130]. Отримавши маршрут, OBU декодує відповідь, оновлює внутрішню таблицю маршрутизації та починає слідувати оновленому шляху. Якщо за час передачі QoS-стан змінився — OBU надсилає новий запит, і цикл повторюється. Саме ця замкненість контуру і забезпечує адаптивність маршрутизації в умовах динамічно змінного інформаційного навантаження [14, 15, 21].

Концептуальну модель управління інформаційними потоками в IoV-мережі подано на рис. 1.4.

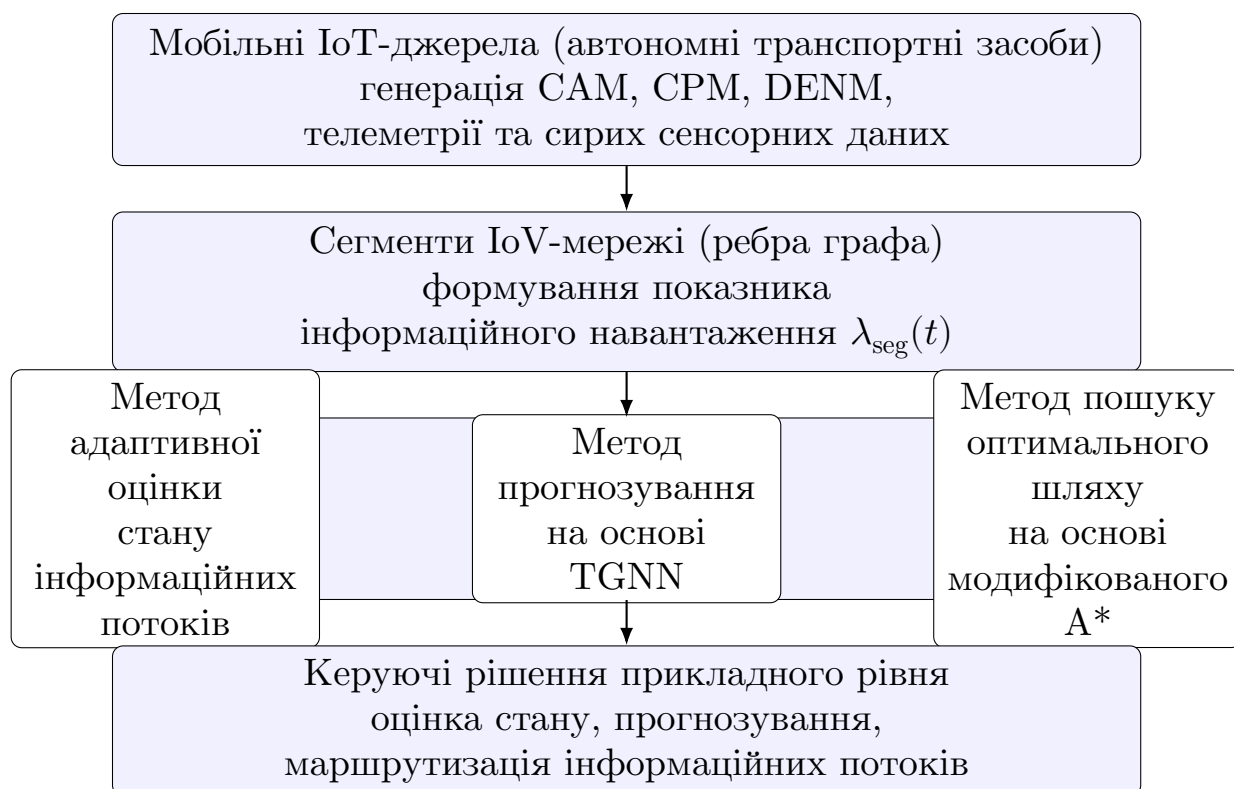


Рис. 1.4 - Концептуальна модель управління інформаційними потоками в IoV-мережі

IoV-мережа описується єдиною часозалежною графовою моделлю,

а не сукупністю окремих графів. Ключовою характеристикою сегмента є інформаційне навантаження $\lambda_{seg}(t)$, яке детерміновано пов'язане з щільністю транспортних засобів і параметрами генерації V2X-повідомлень:

$$\lambda_{seg}(t) = \rho_{seg}(t) \cdot L_{seg} \cdot (f_{CAM} + f_{CPM}) \cdot \overline{S_{msg}}.$$

Тут $\rho_{seg}(t)$ — щільність транспортних засобів на сегменті, L_{seg} — довжина сегмента, f_{CAM} та f_{CPM} — частоти генерації кооперативних повідомлень, а $\overline{S_{msg}}$ — середній розмір повідомлення. Ця залежність має фізичний характер: швидкість руху використовується не як самоціль, а як фізичний індикатор стану інформаційного навантаження, оскільки через фундаментальні діаграми потоку [80] вона пов'язана зі щільністю транспортних засобів і, відповідно, з інтенсивністю генерованих повідомлень.

У контексті задачі управління розглядається просторово-часовий розподіл інформаційного навантаження як ключова характеристика об'єкта дослідження. Саме вона визначає вимоги до методів адаптивної оцінки, прогнозування та маршрутизації інформаційних потоків, які далі аналізуються у підрозділах розділу 1.

1.2 Особливості функціонування IoV-мереж в умовах високого інформаційного навантаження та стохастичності трафіку

1.2.1 Особливості формування інформаційних потоків в IoV-мережах

Функціонування мереж Інтернету транспортних засобів (IoV) відбувається в умовах екстремального інформаційного навантаження, яке має суттєві відмінності від трафіку в класичних клієнт-серверних системах. Якщо веб-системи проектуються з розрахунком на періодичну активність користувачів (трафік, ініційований людиною), то IoV-середовище характеризується безперервною машинною генерацією даних, де кожен транспортний засіб діє як активний генератор телеметрії [1, 2, 7, 21, 136].

Просторово-часовий розподіл такого навантаження детально наведено в підрозділі 1.1 (рис. 1.2). Для подальшого аналізу принципово важливо, що інформаційні потоки в IoV-мережі формуються під впливом реальної міської топології, добових міграційних патернів і концентрації

мобільних IoT-джерел поблизу вузлів тяжіння трафіку [1, 21, 136].

Аналіз архітектури та режимів роботи таких систем дозволяє виділити три критичні характеристики навантаження, які визначають вимоги до методів адаптивної оцінки стану інформаційних потоків і маршрутизації:

Висока інтенсивність та щільність потоку подій

На відміну від систем моніторингу з періодичним опитуванням, IoT-сенсори в транспорті (GPS-трекери, OBD-II сканери, V2X-модулі) генерують звіти про стан з частотою дискретизації f_s від 1 до 10 Гц, що спричиняє ефект мультиплікації навантаження [7]. Розглянемо сегмент мережі з кількістю активних транспортних засобів N_{veh} . Сумарна інтенсивність вхідного потоку λ_{in} визначається як: $\lambda_{in} = N_{veh} \cdot f_s \cdot S_{msg}$, де S_{msg} – середній розмір повідомлення телеметрії. Для 5000 активних автомобілів при частоті 10 Гц система отримує потік інтенсивністю до 50 000 подій на хвилину. Така щільність даних створює критичний тиск на підсистему введення-виведення серверної інфраструктури [8].

Зазначимо, що наведена вище формула є просторово-локалізованою версією виразу $\lambda_{in} = N_{veh} \cdot f_s \cdot S_{msg}$. Зв'язок між ними встановлюється через тотожність $N_{veh} = \rho_{seg}(t) \cdot L_{seg}$, де N_{veh} – кількість транспортних засобів на конкретному сегменті у момент t , а $\rho_{seg}(t)$ та L_{seg} – щільність і довжина сегмента. Таким чином, $\lambda_{seg}(t)$ є **розподіленням по ребрах графа** варіантом загальної інтенсивності λ_{in} : перша прив'язує інтенсивність інформаційного потоку до конкретного ребра e_{uv} IoV-мережі, тоді як друга описує сукупне навантаження в межах одного сегмента або сенсорної зони. Саме $\lambda_{seg}(t)$ є тією теоретичною характеристикою, що визначає вагу ребра $w_{uv}(t)$ у графовій моделі – через вимірювані QoS-індикатори $[\eta, L, \tau]$ (детальніше – у підрозділі 1.6) [1, 7, 21].

Традиційні реляційні бази даних (RDBMS) у таких сценаріях стають «вузьким місцем». Проблема полягає у механізмах забезпечення цілісності (ACID) та індексації: при кожній вставці нового запису про швидкість база даних змушена перебудовувати B-дерева індексів, що викликає блокування таблиць та різке падіння пропускної здатності на запис. Це обумовлює необхідність використання спеціалізованих потокових платформ, здатних забезпечити лінійне горизонтальне масштабування та

низьку затримку запису (детальний аналіз технологій потокової обробки наведено в п. 1.1.4) [7, 8, 18].

Стохастичність та нерівномірність навантаження

Генерація даних в IoV не підпорядковується розподілу Пуассона, а характеризується наявністю «вибухової» активності та важкими хвостами розподілу [9]. Обсяг вхідного трафіку може змінюватися на порядки протягом коротких проміжків часу внаслідок зовнішніх подій:

- *Ранкові та вечірні години пік*: синхронізована міграція тисяч автомобілів.
- *Дорожні інциденти*: різке гальмування групи авто генерує шквал подій про зміну швидкості та прискорення

Ця особливість призводить до явища «дрейфу концепції» (Concept Drift) у вхідних даних [10]. Статистичні властивості потоку (математичне сподівання швидкості μ_v , дисперсія σ_v^2) змінюються непередбачувано. Статичні моделі обробки (наприклад, з фіксованим розміром буфера) в таких умовах стають неефективними, оскільки не здатні адаптуватися до миттєвої зміни інтенсивності потоку, що спричиняє переповнення буферів та неконтрольоване зростання черг повідомлень (відставання обробника від потоку), коли час обробки події перевищує інтервал її актуальності.

Високий рівень шуму та невизначеність даних

Третьою суттєвою проблемою навантаження в IoV є низька якість («достовірність» або *Veracity*) вхідних даних на етапі їх отримання. Телеметрія, що надходить до системи маршрутизації, формується гетерогенними джерелами: від високоточних бортових комп'ютерів до GPS-модулів у смартфонах водіїв [85], що зумовлює значну шумову компоненту у вхідному потоці [11].

Аналіз природи цього шуму дозволяє виділити два типи невизначеності, які створюють паразитне навантаження на алгоритми оновлення графа:

- *Апаратна похибка вимірювання*. В умовах щільної міської забудови проявляється ефект «міських каньйонів», коли сигнал супутників

відбивається від висотних будівель. Це призводить до похибок позиціонування (drift), які можуть сягати 10–30 метрів. Для системи це виглядає як «стрибки» автомобіля між паралельними вулицями або різкі зміни миттєвої швидкості, які фізично неможливі (наприклад, прискорення з 0 до 60 км/год за 1 секунду). Пряма агрегація таких даних без фільтрації спотворює ваги ребер графа.

- *Семантична невизначеність поведінки.* Навіть коректні дані про низьку швидкість не завжди свідчать про затор. Короткочасні події — зупинка на світлофорі, пропускання пішохода, паркування на узбіччі — генерують повідомлення з нульовою швидкістю $v \cong 0$. Прості детерміновані алгоритми (наприклад, усереднення у ковзному вікні) інтерпретують це як різке падіння пропускну здатності ребра.

Ця особливість навантаження вимагає від системи виконання ресурсоемних операцій фільтрації для кожного вхідного повідомлення. Система не може просто записати дані в базу; вона змушена виконувати обчислення статистичних метрик (таких як відстань Махаланобіса) «на льоту», щоб визначити, чи є поточний вимір валідним сигналом для оновлення графа, чи це шум, який слід відкинути. Внаслідок цього задача маршрутизації перетворюється із задачі, обмеженої продуктивністю диска, на задачу, обмежену процесором, що необхідно враховувати при проектуванні архітектури [12], [47].

Узагальнюючи викладене, можна стверджувати, що профіль навантаження в сучасних IoV-системах формується під сукупним впливом трьох факторів: екстремальної частоти транзакцій, стохастичної нерівномірності потоку та низької достовірності «сирих» даних. Проведений аналіз доводить, що класичні методи пакетної обробки інформації та статичні алгоритми фільтрації вичерпали ліміт своєї ефективності в умовах такого динамічного середовища. Вони не здатні гарантувати підтримку актуальності моделі графа IoV-мережі без порушення часових обмежень реального часу. Це обґрунтовує необхідність переходу до потокових архітектур, які передбачають перенесення логіки фільтрації та агрегації даних безпосередньо на етап їх отримання, ще до моменту

збереження в базу даних [7, 8, 12, 18, 21].

1.2.2 Проблема затримок та актуальності даних у навігаційних системах реального часу

Критичним параметром якості функціонування IoV-системи є часова затримка між моментом зміни фізичних параметрів руху (наприклад, зниженням швидкості потоку) та моментом актуалізації відповідної ваги ребра в графі доріг [13]. У динамічному середовищі, де стан транспортної мережі змінюється стохастично, будь-яка затримка призводить до десинхронізації цифрової моделі графа G_{model} з реальною топологією G_{real} [14].

Згідно з вимогами до систем класу *Soft Real-Time*, для ефективного керування інформаційними потоками цикл оновлення стану («подія $\rightarrow$ переоцінка стану сегмента») не повинен перевищувати 100–200 мс. Перевищення цього порогу призводить до того, що система маршрутизації починає оперувати застарілими даними, що робить процедуру пошуку шляху формально коректною, але практично неефективною. [15]

Показник «Віку інформації» (Age of Information) для ваг графа

Для кількісної оцінки актуальності графа доцільно використовувати метрику «Вік інформації» (*Age of Information, AoI*). [16] Стосовно ваги конкретного ребра $w_{uv}(t)$, ця метрика визначається як час, що минув з моменту генерації останнього застосованого оновлення τ_{last} :

$$\Delta_{AoI}(w_{uv}) = t_{current} - \tau_{last}, \quad (1.1)$$

де $\Delta_{AoI}(w_{uv})$ — показник «віку інформації» для вагового коефіцієнта ребра графа, що з'єднує вузли u та v ; $t_{current}$ — поточний системний час у момент запиту маршруту (або перевірки стану); τ_{last} — часова мітка генерації останнього валідного оновлення, яке було успішно застосовано до структури графа.

Цей показник характеризує ступінь часової десинхронізації цифрової моделі графа відносно реального фізичного стану транспортної мережі. В ідеальному (миттєвому) середовищі $\Delta_{AoI} \rightarrow 0$. Однак у реальних розподілених системах Δ_{AoI} завжди є додатною величиною, що акумулює всі затримки передачі, буферизації та обробки даних [14, 15,

16].

Проблема полягає в тому, що реальна вартість проїзду ребром $C_{real}(t)$ є неперервною функцією часу, тоді як вага у графі $w_{uv}(t)$ оновлюється дискретно. В умовах різкої зміни дорожньої обстановки (наприклад, аварійна зупинка потоку, коли швидкість падає з 50 км/год до 0 км/год за кілька секунд), велика затримка оновлення призводить до того, що граф продовжує містити «оптимістичну» вагу ребра. [17] Внаслідок цього виникає ефект «фантомної пропускної здатності»: система вважає дорогу вільною і продовжує спрямовувати туди транспортні потоки, фактично погіршуючи затор ще до того, як інформація про нього відобразиться в моделі [14, 15, 16].

Декомпозиція затримки актуалізації ваг

За відсутності черги повідомлень $\Delta_{AoI} \approx L_{update}$; при перевантаженні системи $\Delta_{AoI} \gg L_{update}$, оскільки повідомлення накопичуються в черзі. Тому мінімізація кожного компонента затримки є критичною для підтримання актуальності графа [14, 15, 16].

Загальна затримка актуалізації ваг L_{update} складається з мережевих та обчислювальних компонентів:

$$L_{update} = L_{net} + L_{ingest} + L_{compute} + L_{commit}, \quad (1.2)$$

де L_{net} та L_{ingest} — час транспортування телеметрії до центру обробки (включаючи буферизацію в чергах повідомлень), $L_{compute}$ — час, необхідний для фільтрації шуму та розрахунку нового значення ваги (агрегація), L_{commit} — час запису оновленого значення в структуру даних графа [14, 15, 18].

Несвоєчасне оновлення ваг графа призводить до хибних маршрутних рішень двох типів: по-перше, граф може відображати затор, який вже розсмоктався, що змушує систему будувати непотрібні об'їзні маршрути; по-друге, граф може відображати вільний рух на ділянці, де фактично утворився затор, що призводить до потрапляння нових автомобілів у пастку затору — цей сценарій є найбільш критичним [14, 15, 16, 17].

Відтак, забезпечення мінімальної затримки оновлення ваг є необхідною умовою адекватності графа транспортної моделі. Класичні

методи агрегації даних не здатні забезпечити синхронізацію стану графа з реальністю в межах 100 мс при пікових навантаженнях. Це вимагає розробки спеціалізованих потокових методів оновлення, які мінімізують обчислювальну затримку $L_{compute}$ та дозволяють підтримувати граф у актуальному стані в режимі м'якого реального часу [14, 15, 18, 19].

1.2.3 Обчислювальні обмеження та вимоги до алгоритмічної складності в системах реального часу

Забезпечення функціонування IoV-системи в межах жорстких часових обмежень (затримка менше 100 мс) накладає сурові вимоги не лише на архітектуру мережі, але й на фундаментальні характеристики застосовуваних алгоритмів [18]: їхню обчислювальну складність та ємнісну складність. В умовах, коли граф доріг $G(V, E)$ містить десятки тисяч вузлів, а частота оновлення ваг сягає тисяч операцій на секунду, класичні алгоритми часто виявляються непридатними для масштабування.

Вимоги до обчислювальної складності підсистеми оновлення даних

Алгоритми попередньої обробки потоку (фільтрація, агрегація, кластеризація) виконуються для *кожного* вхідного повідомлення. Це накладає обмеження на асимптотичну складність операцій [19].

- *Лінійна складність $O(N)$ або $O(1)$* : Ефективний алгоритм обробника повинен обробляти вхідні дані за один прохід. Алгоритми, що вимагають перебору історичного буфера розміром K при надходженні кожного нового елемента (наприклад, перерахунок ковзного середнього «в лоб»), створюють навантаження $O(K)$. Якщо K велике - це блокує процесор.
- *Інкрементальність*: Необхідно використовувати рекурсивні формули, які дозволяють оновлювати стан системи S_t на основі попереднього стану S_{t-1} та нового виміру x_t без звернення до всієї історії: $S_t = f(S_{t-1}, x_t)$. Це забезпечує константну складність $O(1)$ і є критичною вимогою для високочастотних потоків.

Вимоги до ємнісної складності та роботи з пам'яттю

У високонавантажених системах критичним ресурсом стає оперативна пам'ять. Зберігання повної історії «сирих» вимірів для кожного з тисяч ребер графа призводить до лінійного зростання споживання оперативної пам'яті $O(E \times T)$, де E — кількість ребер, T — глибина історії. Це створює підвищене навантаження на підсистему керування пам'яттю та вносить непередбачувані паузи в роботу програми. Ефективна модель даних повинна зберігати лише компактні статистичні дескриптори (наприклад, центр кластера та матрицю коваріації), що зменшує вимоги до пам'яті до $O(E)$ і стабілізує час відгуку [19, 25].

Алгоритмічні виклики підсистеми пошуку шляху

Пошук оптимального маршруту в графі з V вершинами та E ребрами за допомогою класичного алгоритму Дейкстри має складність $O(E + V \log V)$. У часозалежних графах кожна вершина набуває різної вартості залежно від моменту прибуття, тому простір станів розширюється з $|V|$ до $|V| \times |T|$, де $|T|$ — кількість дискретних часових кроків. Це явище відоме як «вибух простору пошуку» [6, 20] і робить критично важливою мінімізацію кількості розкритих вершин під час пошуку. Застосування евристичної функції $h(n)$ в алгоритмі A^* дозволяє спрямувати пошук, однак сама евристика повинна обчислюватися миттєво. Існує ризик, що використання складної глибокої нейромережі для розрахунку $h(n)$ займе більше часу, ніж простий перебір вершин. Тому до ML-моделей висувається вимога «легкої» архітектури та низької затримки формування прогнозу (менше 5–10 мс).

Отже, при розробці системи маршрутизації інформаційних потоків необхідно враховувати не лише точність моделей, а й їхню алгоритмічну ефективність. Пріоритет надається підходам, які забезпечують сублінійну складність пошуку, константну складність оновлення ваг та мінімальне використання оперативної пам'яті, що дозволяє системі масштабуватися в умовах обмежених серверних ресурсів [18, 19, 20].

1.2.4 Технології потокової обробки даних для IoV-систем

Аналіз характеристик інформаційного навантаження IoV-систем (п. 1.1.1–1.1.3) показав, що обсяг та швидкість надходження телеметричних даних унеможливають використання класичних

підходів типу «запит–відповідь» або періодичної пакетної обробки. Для обробки високочастотних потоків машинних даних у режимі реального часу сучасні ITS-платформи використовують архітектуру потокової обробки повідомлень (*Stream Processing*) [7, 78, 111, 114].

Ключовою технологією в цій парадигмі є розподілені брокери повідомлень, зокрема **Apache Kafka** — розподілена платформа потокової обробки, що реалізує модель «публікація–підписка» (*Publish–Subscribe*) [52, 100]. Архітектура Kafka базується на кількох фундаментальних принципах:

- **Розділений журнал** (*Partitioned Log*): потік повідомлень розподіляється між кількома розділами (*Partitions*), що забезпечує горизонтальне масштабування та паралельну обробку без взаємного блокування обробників;
- **Групи обробників** (*Consumer Groups*): механізм, що дозволяє кільком незалежним алгоритмам обробляти ідентичний потік даних паралельно — це необхідно для порівняльного дослідження різних методів оновлення ваг графа на одному потоці вхідних даних;
- **Асинхронна взаємодія**: джерело даних (IoT-пристрій або сенсор) надсилає повідомлення без очікування підтвердження від обробника. Це усуває залежність швидкості надсилання даних від швидкості їх обробки — вимога, що є визначальною для IoV-середовища, де транспортний засіб не може бути заблокований через перевантаження серверної інфраструктури.

Альтернативні платформи (Apache Pulsar, RabbitMQ, AWS Kinesis) також реалізують потокову обробку, однак Apache Kafka є де-факто стандартом для високонавантажених IoT-систем [53, 100] завдяки гарантованому збереженню порядку повідомлень у межах розділу, високій пропускній здатності та вбудованому механізму відмовостійкості через реплікацію.

Застосування потокової архітектури на базі розподілених брокерів повідомлень є необхідною інфраструктурною передумовою для побудови системи адаптивної маршрутизації, здатної обробляти високочастотний

потік IoT-телеметрії із забезпеченням горизонтальної масштабованості та ізоляції обробників [7, 8].

1.3 Аналіз методів оцінки стану інформаційних потоків в IoV-мережах

Ефективність адаптивної маршрутизації в IoV-системах є функцією від двох змінних: якості алгоритму пошуку шляху та актуальності оцінки стану сегментів графа W_t [30]. Якщо алгоритм пошуку (наприклад, A^*) відповідає за вибір оптимальної траєкторії [89], то підсистема адаптивної оцінки стану інформаційних потоків відповідає за адекватність відображення фізичної реальності у цифровій моделі.

В умовах Інтернету транспортних засобів вага ребра $w_{uv}(t)$ (яка зазвичай корелює з часом проїзду або оберненою швидкістю) є нестационарним стохастичним процесом [48, 10]. Задача підсистеми оновлення полягає у перетворенні високочастотного, зашумленого потоку дискретних подій $S = \{e_1, e_2, \dots, e_n\}$ у неперервну функцію вартості ребра $w_{uv}(t)$.

Типову добову динаміку такого навантаження наведено на рис. 1.5. Вона поєднує два виражені піки, міжпіковий помірний режим та шумові флуктуації на рівні окремих вимірів, що безпосередньо формує вимоги до адаптивного оцінювання стану потоку.

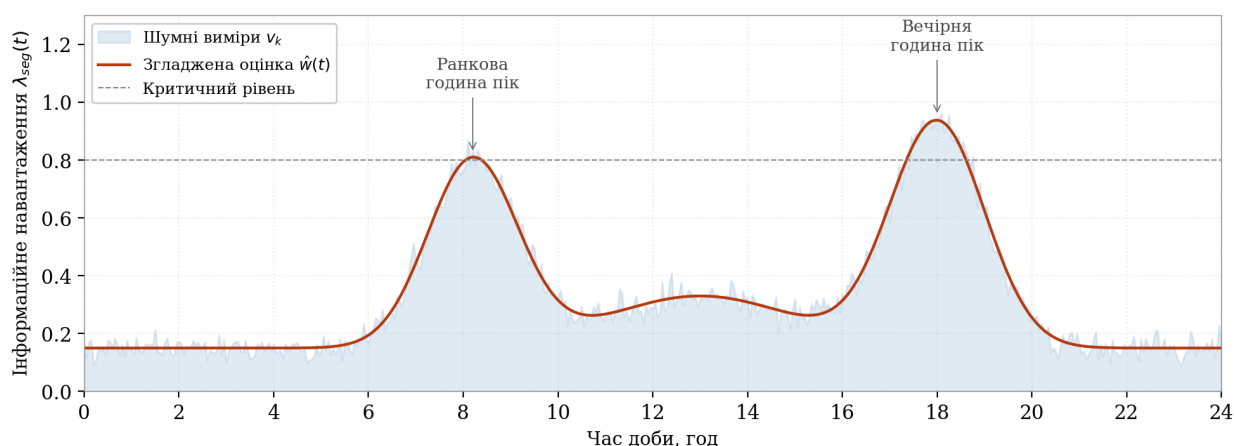


Рис. 1.5 - Добова динаміка інформаційного навантаження на сегменті IoV-мережі. Блакитною областю показано зашумлені вимірювання $\lambda_{seg}(t)$, червоною лінією — згладжену оцінку стану потоку, пунктиром — критичний рівень навантаження. Наявність ранкового та вечірнього максимумів обґрунтовує потребу в адаптивній оцінці, яка одночасно фільтрує шум і швидко реагує на наближення до перевантаження.

Аналіз науково-технічної літератури та існуючих реалізацій ITS дозволяє класифікувати підходи до вирішення цієї задачі на такі три основні групи [21]:

- *Пакетна обробка (Batch Processing)*: Періодична агрегація історичних даних із сховищ [22].
- *Віконні методи (Window-based approaches)*: Усереднення даних у ковзному вікні фіксованого розміру [23].
- *Інкrementальні методи (Incremental Smoothing)*: Рекурсивне оновлення стану на основі експоненційного згладжування [27, 74, 75, 77].

Нижче наведено аналіз перших двох груп методів, які на сьогодні є найбільш поширеними, та виявлено їхні системні обмеження в умовах високонавантажених IoV-мереж.

1.3.1 Обмеження систем пакетної обробки та методів ковзного вікна

Аналіз існуючих архітектурних рішень для ITS дозволяє виділити два домінуючі підходи до агрегації телеметричних даних: класичну пакетну обробку (*Batch Processing*) [22] та просту потокову агрегацію на основі ковзного вікна (*Sliding Window*) [23]. Незважаючи на широке розповсюдження, обидва підходи мають суттєві обмеження при роботі з високочастотними стохастичними потоками.

Системи на базі періодичного перерахунку

Традиційні системи моніторингу транспорту часто будуються за парадигмою «спочатку зберегти — потім аналізувати». У такій моделі потік телеметрії спочатку персистується (зберігається) у реляційній базі даних (RDBMS) або спеціалізованій базі часових рядів (Time-Series DB) [110] [24]. Оновлення ваг графа ініціюється планувальником завдань дискретно, з фіксованим інтервалом ΔT (наприклад, кожні 5 хвилин). Математично процес оновлення ваги ребра $w_{uv}(t)$ на момент часу t_{now} описується як операція агрегації множини історичних записів $S_{\Delta T}$: $w_{uv}(t) = \mathcal{F}\left(\frac{1}{|S_{\Delta T}|} \sum_{v_i \in S_{\Delta T}} v_i\right)$, де $\mathcal{F}$ — функція перетворення швидкості у вартість (наприклад, час обслуговування інформаційного потоку на сегменті).

Критичні недоліки підходу:

- *Часові «сліпі зони»:* Дискретність оновлення призводить до того, що стан графа є ступінчастою функцією, тоді як реальний трафік змінюється неперервно. Якщо критична подія (наприклад, ДТП) сталася на початку інтервалу ΔT , інформація про неї потрапить у граф лише під час наступного перерахунку. У проміжку $[t, t + \Delta T]$ система маршрутизації залишається «сліпою» до проблеми, продовжуючи спрямовувати автомобілі в затор.
- *Пікові навантаження та блокування:* Операція агрегації вимагає одночасного сканування індексів бази даних для тисяч ребер графа. Це створює багато запитів, що призводить до стрибкоподібного зростання навантаження на CPU та підсистему вводу-виводу. Експериментальні дослідження показують, що в моменти перерахунку середнє навантаження на процесор зростає до 20% і вище, а час відгуку системи суттєво зростає, що є неприпустимим для сервісів реального часу.

Методи ковзного вікна

Альтернативою пакетній обробці є методи потокової агрегації, де для кожного ребра підтримується буфер останніх k значень або значень за останні τ секунд [23]. Нова оцінка ваги розраховується як просте ковзне середнє (SMA) при надходженні кожного повідомлення. Хоча цей метод зменшує затримку порівняно з пакетним підходом, він стикається з проблемами масштабованості та налаштування чутливості:

- *Проблема масштабованості по пам'яті:* Для реалізації алгоритму необхідно зберігати в оперативній пам'яті повну історію вимірів для кожного ребра. У графі з $|E|$ ребрами загальний обсяг необхідної пам'яті становить $M \approx |E| \times kS_{item}$. При великій кількості ребер (масштаб мегаполісу) та достатній глибині вікна це призводить до значних витрат оперативної пам'яті [25]. Крім того, постійна ротація елементів у буферах (видалення найстарішого, додавання нового) створює підвищене навантаження на підсистему керування

пам'яттю, що вносить непередбачувані мікро-затримки в обробку потоку.

- «Дилема чутливості»: Фіксований розмір вікна створює конфлікт між здатністю фільтрувати шум та швидкістю реакції [26]. Велике вікно $k \uparrow$ забезпечує стабільну оцінку та добре працює з шумом GPS. Однак, система набуває високої інерційності (відставання). При різкому падінні швидкості (затор) ковзне середнє буде знижуватися повільно, лінійно залежачи від ширини вікна. Мале вікно $k \downarrow$ забезпечує низьку затримку детекції змін. Проте, система стає вразливою до викидів. Короткочасна зупинка одного транспортного засобу може суттєво змінити середнє значення у малому вікні, що призведе до хибного детектування затору та нестабільності маршрутів.

Як наслідок, ні пакетні методи (через високу затримку), ні методи ковзного вікна (через високу ресурсоємність та інерційність) не забезпечують оптимального вирішення задачі оновлення графа в умовах IoV. Необхідний перехід до інкрементальних методів з адаптивними параметрами, які здатні змінювати свою чутливість залежно від контексту [22, 23, 26, 27].

1.3.2 Проблема ідентифікації режимів руху та дрейфу концепції у вхідному потоці

Третім класом методів, що широко застосовується в інженерії трафіку для задач реального часу, є методи інкрементального оновлення (*Incremental Smoothing*). До них відносяться алгоритми простого експоненційного згладжування (Exponential Moving Average, ЕМА) та рекурсивні байєсівські фільтри (зокрема, фільтр Калмана) [27, 76, 93, 94, 107].

Основна ідея цих методів полягає у рекурсивному оновленні оцінки стану S_t на основі попереднього значення S_{t-1} та нового виміру x_t :

$$S_t = \alpha x_t + (1 - \alpha) S_{t-1}, \quad (1.3)$$

де $\alpha \in [0, 1]$ — коефіцієнт згладжування, що визначає вагу нових

даних.

Хоча ці методи є обчислювально ефективними (складність $O(1)$) і не вимагають зберігання історії, їх застосування в IoV-системах нашоується на структурну проблему, пов'язану з природою транспортного потоку — проблему дрейфу концепції (Concept Drift) [10, 27, 48].

Більшість класичних алгоритмів базуються на гіпотезі про квазістаціонарність процесу (тобто припущенні, що статистичні характеристики потоку — математичне сподівання швидкості та дисперсія — змінюються достатньо повільно, щоб на коротких інтервалах їх можна було вважати сталими) [27]. Однак у реальних умовах це припущення є помилковим [10, 48, 83]. Транспортний потік характеризується наявністю фазових переходів між різними режимами функціонування [84].

- *Вільний потік*: Висока швидкість, низька щільність, стабільна дисперсія.
- *Синхронізований потік*: Середня швидкість, висока щільність, сильна кореляція між сусідніми авто.
- *Затор*: Низька швидкість, стоп-старт режим, висока варіативність миттєвих значень.

Перехід між цими станами часто відбувається стрибкоподібно — наприклад, внаслідок аварії пропускна здатність падає миттєво. Статичні моделі не здатні своєчасно детектувати цей момент переходу. Класичні процедури виявлення точок зміни, зокрема CUSUM [79], здатні фіксувати момент зламу, але не надають механізму безперервного адаптивного налаштування параметра згладжування для подальшого супроводу потоку.

Головним недоліком існуючих інкрементальних методів є використання фіксованого значення коефіцієнта α . Це створює неусувний конфлікт, відомий як компроміс між стійкістю та чутливістю [27].

При малому $\alpha \rightarrow 0$ система стає інерційною (консервативною). Вона добре фільтрує шум GPS у стабільному режимі, але катастрофічно запізнюється з реакцією на різку зміну ситуації (відставання). Наприклад,

при утворенні затору система ще кілька хвилин транслюватиме "високу швидкість", спрямовуючи нові автомобілі в пастку [26, 27].

При великому $\alpha \rightarrow 1$ Система реагує миттєво, але стає нестабільною. Будь-який випадковий викид (шум вимірювання) або короткочасна зупинка інтерпретується як зміна градієнта швидкості, що призводить до постійного "мерехтіння" ваг графа і нестабільності маршрутів [26, 27].

Дослідження показують [26, 27], що неможливо підібрати єдине універсальне значення α , яке було б ефективним у всіх режимах руху.

Узагальнюючи аналіз методів оновлення графа, можна констатувати наступне:

1. Пакетні методи непридатні через високу затримку та пікові навантаження на БД.
2. Віконні методи обмежені високим споживанням пам'яті та лінійною складністю.
3. Інкрементальні методи з фіксованими параметрами не здатні адаптуватися до нестаціонарної динаміки трафіку («дрейфу концепції»).

Це обґрунтовує необхідність розробки адаптивного інкрементального методу, який поєднує обчислювальну ефективність експоненційного згладжування з інтелектуальним механізмом керування коефіцієнтом α на основі аналізу простору ознак [26, 27, 93, 94].

Окремо слід відзначити, що класичні прогнозні моделі часових рядів (ARIMA, SARIMA) [48] та рекурентні нейронні мережі (LSTM, GRU) [37, 63, 91, 119] також не вирішують задачу адаптивної оцінки стану інформаційних потоків на сегментах мережі. Ці методи обробляють кожен часовий ряд ізольовано, ігноруючи просторову топологію мережі: стан ребра моделюється без урахування пропагації заторів від сусідніх ребер. Крім того, ARIMA потребує стаціонарного ряду (або інтегрування для усунення нестаціонарності), що суперечить природі транспортних даних з різкими структурними зсувами. Автономні LSTM-моделі, хоча й здатні захоплювати нелінійні часові залежності, потребують значних обсягів навчальних даних та обчислювальних ресурсів для перенавчання при

зміні умов руху, що робить їх непридатними для потокового оновлення ваг у реальному часі.

використання фіксованого α створює неусувний компроміс: висока чутливість підсилює шум і спричиняє нестабільність маршрутів, тоді як низька чутливість забезпечує фільтрацію, але критично сповільнює реакцію на інциденти. Це обґрунтовує необхідність адаптивного механізму керування α на основі аналізу поточного режиму руху.

1.3.3 Аналіз підходів промислових навігаційних сервісів до оцінки стану сегментів IoV-мережі

Для об'єктивного оцінювання існуючих методів адаптивної оцінки стану сегментів необхідно проаналізувати підходи, що використовуються провідними комерційними навігаційними сервісами. На основі наукових публікацій та технічних звітів можна ідентифікувати три основні парадигми [54, 55, 56, 108].

HERE / TomTom — фільтр Калмана з адаптивним шумом процесу. Навігаційні сервіси HERE та TomTom базуються на рекурсивній баєсівській фільтрації, зокрема на одновимірному фільтрі Калмана [54, 108]. Стан системи — оцінка швидкості $\hat{x}$ як фізичного індикатора стану сегмента та невизначеність P — оновлюються за класичною двоетапною схемою «прогноз–корекція»:

$$\hat{x}_{t|t-1} = \hat{x}_{t-1}, \quad P_{t|t-1} = P_{t-1} + Q,$$

$$K = \frac{P_{t|t-1}}{P_{t|t-1} + R}, \quad \hat{x}_t = \hat{x}_{t|t-1} + K(z_t - \hat{x}_{t|t-1}),$$

де z_t — виміряна швидкість, R — дисперсія шуму вимірювання, Q — шум процесу, K — коефіцієнт посилення Калмана. Адаптація полягає у динамічному збільшенні Q при нестабільному стані сегмента. Коефіцієнт K є математично еквівалентним адаптивному α в ЕМА, проте визначається виключно статистичними характеристиками шуму і **не враховує причину зміни показників** — фільтр однаково реагує на випадкове коливання GPS-вимірювань і на реальну зміну стану інформаційного навантаження (наприклад, початок затору), оскільки не

містить механізму розпізнавання поточного режиму навантаження.

Слід зазначити, що математична еквівалентність $K \equiv \alpha$ теоретично дозволяє замінити калманівський механізм визначення K на будь-який інший, зокрема на нечітку кластеризацію режимів інформаційного навантаження (такий гібридний підхід досліджено у розділі 4 під назвою «Нечіткий-Калман»). Однак це не усуває надлишковості: якщо α вже визначається зовнішнім механізмом на основі режиму інформаційного навантаження, то калманівські рівняння для обчислення K через P , Q та R стають *зайвим шаром обчислень*, що не додає інформації. Пряме застосування $\hat{v}_t = \alpha(\vec{x}) \cdot z_t + (1 - \alpha(\vec{x})) \cdot \hat{v}_{t-1}$ з адаптивним α , визначеним через аналіз режиму інформаційного навантаження, є обчислювально простішим ($O(1)$ без підтримки стану P) та концептуально прозорішим, оскільки параметр α має безпосередню інтерпретацію — ступінь довіри до нових даних залежно від стабільності режиму інформаційного навантаження.

Waze — баєсівське злиття з виявленням подій. Сервіс Waze моделює швидкість на кожному сегменті як нормальний розподіл $\mathcal{N}(\mu, \sigma^2)$ [55], який оновлюється баєсівським зваженням за оберненою дисперсією. Ключовою відмінністю є механізм виявлення аномалій: якщо спостереження суттєво відхиляється від апіорного розподілу ($|z - \mu| > k \cdot \sigma$, типово $k = 3$), система інтерпретує це як інцидент та **скидає** апіорний розподіл, починаючи оцінку з нуля. Такий двозначний поділ («скидати або не скидати») відповідає принципу роботи сервісу Waze, де користувачі вручну повідомляють про інциденти на дорозі. Обмеженням є надмірна чутливість механізму скиду: на зашумлених даних скиди можуть відбуватися занадто часто (при кожному значному GPS-відхиленні), що знижує стабільність оцінки швидкості.

Google Maps — пряме прогнозування на базі графових нейромереж. Підхід Google Maps, описаний Derrow-Pinion et al. [56], концептуально відрізняється від описаних вище підходів. Замість коригування оцінки швидкості на основі минулих спостережень, цей метод **прогнозує майбутню** швидкість за допомогою просторово-часової графової нейронної мережі. Модель отримує на вхід послідовність станів усіх вузлів графової моделі IoV-мережі за попередні часові кроки

та формує прогноз, який безпосередньо використовується як оцінка стану сегмента та вага ребра для маршрутизації. Це робить підхід **випереджувальним**: система здатна передбачити затор до того, як він повністю сформується. Проте прогнозна швидкість може вносити систематичну похибку — модель може «передбачити» затор, якого в дійсності не відбудеться.

Результати порівняльного аналізу розглянутих методів оцінки стану сегментів IoV-мережі систематизовано в табл. 1.2.

Таблиця 1.2 - Порівняльний аналіз методів оцінки стану сегментів IoV-мережі

Метод	Затримка	Складність	Пам'ять	Адаптивність
Пакетна обробка	Висока (ΔT)	$O(N)$ за інтервал	Потрібна БД	Відсутня (фіксований ΔT)
Ковзне вікно (SMA)	Середня	$O(k)$ на подію	$O(E \times k)$	Фіксована (параметр k)
ЕМА (фіксований α)	Низька	$O(1)$ на подію	$O(E)$	Фіксована (параметр α)
Фільтр Калмана (HERE)	Низька	$O(1)$ на подію	$O(E)$	Часткова (за рівнем шуму)
Баєсівська оцінка (Waze)	Низька	$O(1)$ на подію	$O(E)$	Бінарна (норма-/аномалія)
TGNN-прогноз (Google)	Низька	$O(V \cdot K)$	Модель + граф	Повна (просторово-часова)

Проведений аналіз дозволяє виявити спільне обмеження промислових підходів: жоден з них не класифікує поточний стан інформаційного навантаження сегмента (вільний потік, формування затору, аварія) і не використовує цю класифікацію для вибору параметрів оцінки стану. Фільтр Калмана (HERE) регулює лише співвідношення між довірою до нового вимірювання та попередньої оцінки на основі рівня шуму;

Waze застосовує спрощений бінарний поділ на «норму» та «аномалію» без проміжних градацій; Google Maps повністю замінює фільтрацію на пряме прогнозування нейромережею. Це обґрунтовує доцільність розробки методу, в якому коефіцієнт згладжування визначається на основі автоматично розпізнаного режиму інформаційного навантаження — від стабільного вільного потоку до аварійної ситуації.

1.4 Аналіз методів маршрутизації інформаційних потоків від мобільних IoT-джерел

Після формування актуальної моделі стану IoV-мережі на основі оціненого інформаційного навантаження наступним етапом функціонування системи маршрутизації є безпосереднє знаходження оптимальної траєкторії руху від початкового вузла до цільового. В умовах Інтернету транспортних засобів (IoV) ця задача розв’язується у часозалежному графі (*Time-Dependent Graph*, *TDG*) [28].

Математичною особливістю таких графів є те, що вага ребра $w_{uv}(t)$ залежить від часу t входу в це ребро. Це накладає суттєві обмеження на застосовність стандартних алгоритмів [5]. По-перше, для коректного пошуку необхідно, щоб вагова функція задовольняла властивості FIFO (*First-In-First-Out*) [6]: рух не може відбуватися "назад у часі", і затримка на вході не повинна призводити до прибуття раніше, ніж при старті без затримки. По-друге, динаміка ваг робить статичні оцінки відстані (геометричні евристики) неефективними, оскільки найкоротший фізично шлях може виявитися найдовшим у часовому вимірі через затори.

Ключовим критерієм ефективності алгоритму в таких умовах стає не лише оптимальність знайденого рішення — мінімальне сумарне навантаження на сегменти маршруту, але й швидкодія самого пошуку (кількість обчислювальних операцій) [20]. В умовах високого навантаження, коли сервер повинен обробляти тисячі запитів маршрутизації на секунду, повний перебір варіантів стає неприпустимим.

Маршрутизаційні рушії з відкритим кодом. Окрему категорію становлять маршрутизаційні рушії з відкритим кодом — OSRM (Open Source Routing Machine), Valhalla та GraphHopper, які використовуються у

тисячах застосунків. Найпоширенішим є **OSRM**, що застосовує алгоритм скорочення ієрархій (Contraction Hierarchies) [44], який забезпечує пошук маршруту за мілісекунди. Однак ці рушії оперують **статичними профілями ваг**: для оновлення ваг графа необхідно виконати повний цикл перебудови (extract $\rightarrow$ partition $\rightarrow$ customize), що займає від хвилин до годин залежно від розміру мережі. Аналогічні обмеження мають Valhalla та GraphHopper. Це робить їх принципово непридатними для адаптивної маршрутизації в реальному часі на основі потокових IoT-даних, де ваги ребер змінюються кожну хвилину.

У цьому підрозділі проведено аналіз ефективності класичних детермінованих алгоритмів (Dijkstra, A^*) та сучасних підходів на базі машинного навчання в контексті їх застосування до динамічних IoV-мереж. Основну увагу приділено проблемі вибору евристичної функції $h(u)$, яка визначає напрямок та швидкість збіжності алгоритму пошуку.

1.4.1 Ефективність класичних алгоритмів пошуку (Dijkstra, A^*) в умовах змінного середовища

В основі більшості сучасних навігаційних систем лежать алгоритми пошуку на графах, які належать до класу методів «найкращий-перший» (*Best-First Search*). У контексті задач маршрутизації де-факто стандартами є алгоритм Дейкстри (*Dijkstra's Algorithm*) та його евристична модифікація — алгоритм A^* (*A-Star*) [29, 30].

Алгоритм Дейкстри

Алгоритм Дейкстри, запропонований у 1959 році, гарантує знаходження найкоротшого шляху у графі з невід'ємними вагами ребер. Його стратегія полягає у рівномірному розширенні фронту пошуку від початкового вузла s в усіх напрямках доти, доки не буде досягнуто цільового вузла d [29]. У часозалежних графах алгоритм Дейкстри залишається еталоном точності. Проте його головним недоліком є висока обчислювальна складність. Оскільки алгоритм є «непоінформованим», він досліджує простір станів «наосліп». Кількість відвіданих вершин $N_{explored}$ зростає квадратично або поліноміально залежно від відстані до цілі. В статті [31] зазначено, що в умовах високонавантажених IoV-систем, де потрібно обробляти тисячі запитів за секунду, такий підхід є надто

повільним, оскільки вимагає перебору значної частини вузлів графа, які апріорі не лежать на оптимальному шляху.

Алгоритм A^*

Алгоритм A^* є розвитком методу Дейкстри, який використовує евристичну інформацію для пріоритезації напрямку пошуку. [30] Функція оцінки вартості шляху $f(n)$ для кожного вузла n визначається як:

$$f(n) = g(n) + h(n), \quad (1.4)$$

де:

- $g(n)$ — точна вартість шляху від старту до вузла n (вже пройдена відстань або час);
- $h(n)$ — евристична оцінка вартості шляху від вузла n до цільового вузла (прогноз).

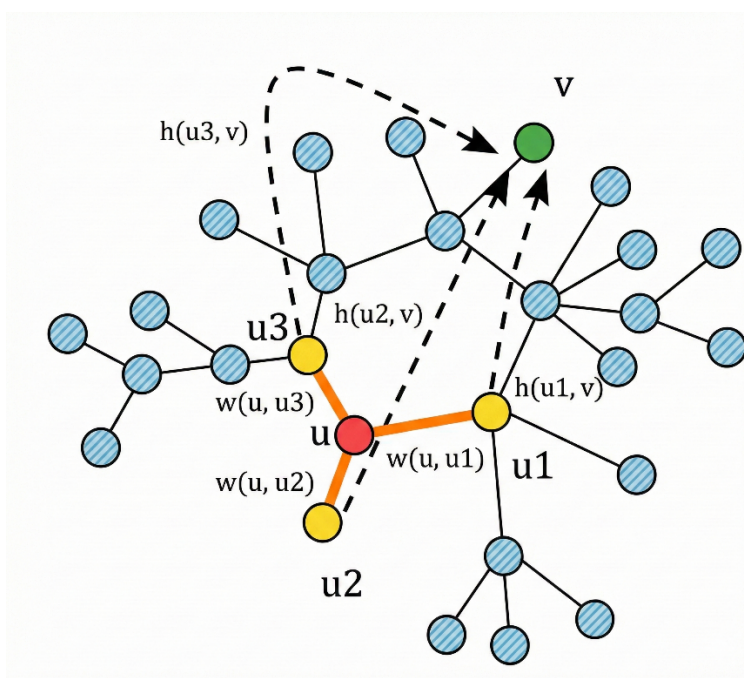


Рис. 1.6 - Принцип розкриття вершини графа в A^* алгоритмі

Принцип розкриття вершин показано на рис. 1.6. Ефективність A^* прямо залежить від якості евристичної функції $h(n)$. Якщо $h(n)$ є «допустимою» (ніколи не переоцінює реальну вартість) та «монотонною» (задовольняє нерівність трикутника), алгоритм гарантовано знайде оптимальний шлях, перевібивши при цьому значно меншу кількість вершин, ніж алгоритм Дейкстри [30].

1.4.2 Проблеми застосування статичних евристик та методів прямого прогнозування часу прибуття (ETA)

Проблема класичного підходу полягає в тому, що традиційні евристики є статичними. Найпоширенішим варіантом є геодезична евристика $h_{geo}(v) = d_{gc}(v, D) / v_{max}$, де v_{max} є фізичним індикатором **мінімального** інформаційного навантаження сегмента (стан вільного потоку, $\lambda_{seg} \rightarrow \lambda_{min}$). У статичному графі, де навантаження незмінне, така евристика працює ефективно. Однак у часозалежній IoV-мережі, де інформаційне навантаження $\lambda_{seg}(t)$ динамічно змінюється, виникає принципове протиріччя: евристика h_{geo} припускає, що всі сегменти перебувають у стані мінімального навантаження, і не відображає поточного стану інформаційних потоків.

Розглянемо типовий сценарій: сегмент на прямому геометричному шляху до цілі перебуває під критичним інформаційним навантаженням ($\lambda_{seg} \gg \lambda_{min}$, висока щільність IoT-джерел, низька швидкість). Евристика h_{geo} має мале значення для вузлів цього сегмента — вона «не бачить» підвищеного навантаження — тому A^* пріоритезує їх розкриття. Алгоритм розкриває вузли, обчислює реальну вагу $g(n)$ (велику через перенавантаженість), і лише тоді починає шукати обхідні шляхи через менш навантажені сегменти. У результаті A^* зі статичною евристикою не забезпечує ефективного управління інформаційними потоками в умовах динамічного навантаження.

В умовах динамічного інформаційного навантаження IoV-мережі класичний алгоритм A^* зі статичними евристиками втрачає свою головну перевагу — ефективність спрямованого пошуку. Статична оцінка $h(n)$ перестає корелювати з реальним станом інформаційного навантаження $\lambda_{seg}(t)$ на залишку маршруту, що унеможливорює проактивне обрання шляху через менш навантажені сегменти мережі. Це актуалізує задачу побудови **часозалежної евристики**, що враховує поточний і прогнозний стан $\lambda_{seg}(t)$ на сегментах маршруту. [31]

Сучасною альтернативою є спроба використати методи машинного навчання (ML) для апроксимації реального часу прибуття. Для цього використовуються моделі градієнтного бустингу (наприклад, LightGBM [32]) або прості нейронні мережі, які навчаються на історичних даних

мінімізувати середньоквадратичну помилку (MSE) прогнозу. Хоча такі моделі можуть забезпечувати високу точність прогнозування середнього часу поїздки, їхнє безпосереднє використання в алгоритмі A^* часто призводить до погіршення результатів [53]. Причиною цього є проблема неузгодженості. Для ефективної роботи A^* (без повторного розкриття вершин) евристика повинна бути монотонною [30] тобто задовольняти нерівність трикутника: $h(u) \leq w(u, v) + h(v)$, де $w(u, v)$ – вага ребра між сусідніми вузлами.

ML-моделі, що навчаються лише на мінімізацію похибки кінцевого часу, не враховують локальну структуру графа. Вони можуть видавати прогнози, які суперечать один одному для сусідніх вузлів (наприклад, прогноз часу від вузла u менший, ніж від сусіднього вузла v , хоча рух відбувається від u до v). Це порушення призводить до того, що порушується монотонність вздовж шляху, алгоритм A^* змушений виконувати процедуру повторного розкриття вершин, коли знайдено кращий шлях до вже закритої вершини [30, 53].

Отже, застосування статичних евристик у динамічному середовищі є неефективним через ігнорування поточного стану $\lambda_{\text{seg}}(t)$, а пряме використання регресійних ML-моделей для прогнозування залишкового часу обслуговування інформаційного потоку $h(n) = \hat{T}(n \rightarrow d)$ є ризикованим через порушення математичних властивостей (допустимості та монотонності), необхідних для коректної роботи A^* . Альтернативним підходом є **прогнозування швидкості потоку** $v_{\text{pred}}(n, t)$ — фізичного індикатора $\lambda_{\text{seg}}(t)$ — з побудовою евристики:

$$h(n, t) = \frac{d_{\text{gc}}(n, d)}{v_{\text{pred}}(n, t)},$$

де d_{gc} — геодезична відстань до цілі, яка завжди не перевищує реальну відстань по ребрах графа. Така конструкція забезпечує структурну тенденцію до **допустимості** (недооцінки реальної вартості обслуговування потоку), оскільки $d_{\text{gc}} \leq d_{\text{road}}$ і прогнозна швидкість не перевищує v_{max} . Це дозволяє A^* оцінювати **мінімальний сумарний час обслуговування інформаційних потоків** на залишку маршруту, орієнтуючи пошук через сегменти з очікуваним нижчим навантаженням λ_{seg} .

Перспективним напрямком є Neural A* [43] — диференційовна версія алгоритму A*, яка навчається формувати карту вартостей безпосередньо з даних. Проте цей підхід орієнтований на сіткові (grid) середовища та не адаптований до часозалежних графів IoV-мережі реального масштабу.

1.4.3 Аналіз методів машинного навчання для прогнозування параметрів маршруту

Для подолання обмежень статичних евристик сучасні дослідження фокусуються на застосуванні методів машинного навчання, здатних виявляти приховані залежності у транспортних даних. Нижче наведено порівняльний аналіз трьох груп методів — від найпростіших до найбільш виразних — з метою обґрунтування вибору архітектури для формування динамічної евристики [32, 37, 39, 40, 41, 43, 91, 101, 102, 103, 127].

Статистичні моделі на табличних даних (LightGBM)

Як базовий підхід для оцінки ефективності прогнозування часто використовують ансамблеві методи на основі дерев рішень. Зокрема, алгоритм LightGBM реалізує градієнтний бустинг, послідовно будуючи слабкі моделі для виправлення помилок попередніх ітерацій $F_{m+1}(x) = F_m(x) + h_m(x)$, де $h_m(x)$ - нове дерево рішень, що мінімізує функцію втрат (MSE). Перевагою LightGBM є висока швидкість навчання та точність на табличних даних [32]. Однак, це не графова модель: вона використовує лише числові ознаки окремих сегментів, ігноруючи топологічні зв'язки між дорогами та механізм просторово-часового поширення навантаження між суміжними ребрами. Це призводить до того, що прогнози для сусідніх ребер можуть бути неузгодженими, що є критичним для алгоритму A*. Саме ця фундаментальна обмеженість — відсутність врахування просторової структури графа — унеможливорює використання LightGBM як евристичної функції в запропонованій системі.

Причина такої обмеженості ілюструється на рис. 1.7. Локальний сплеск інформаційного навантаження на одному сегменті через малий інтервал часу поширюється на суміжні ребра, тому модель прогнозування повинна явно враховувати графову структуру мережі, а не лише локальні табличні ознаки [39, 40, 41].

Графові нейронні мережі (GNN)

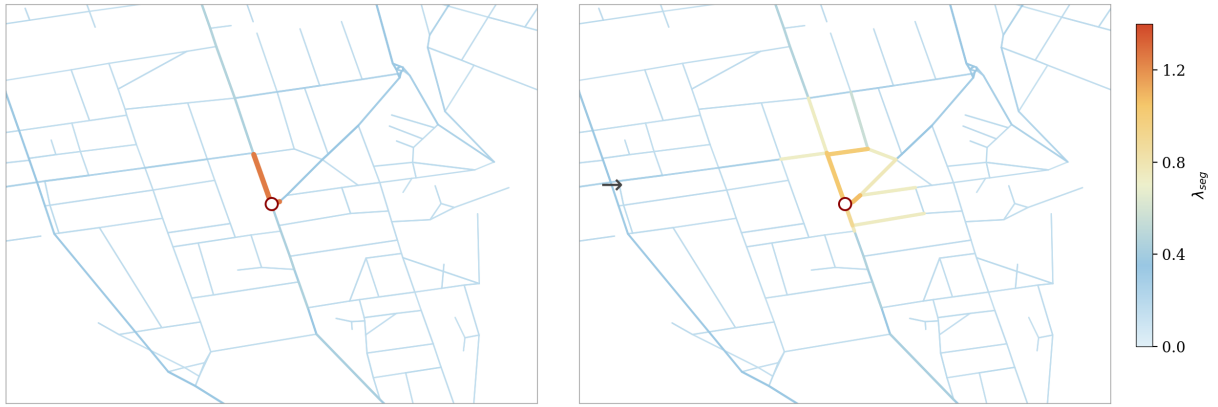


Рис. 1.7 - Просторово-часове поширення інформаційного навантаження між сегментами IoV-мережі: а — локальний сплеск на одному сегменті в момент t ; б — поширення навантаження на сусідні ребра в момент $t + \Delta t$. Така кореляція між суміжними сегментами обґрунтовує використання графових моделей TGNN для прогнозування майбутнього стану мережі.

Для того, щоб безпосередньо враховувати топологію дорожньої мережі застосовуються графові нейронні мережі. GNN обробляють дані у їхній природній структурі, що дає можливість вузлам агрегувати інформацію від своїх сусідів [33]. Загальний принцип шару GNN описується механізмом передавання повідомлень (message passing), який складається з двох етапів.

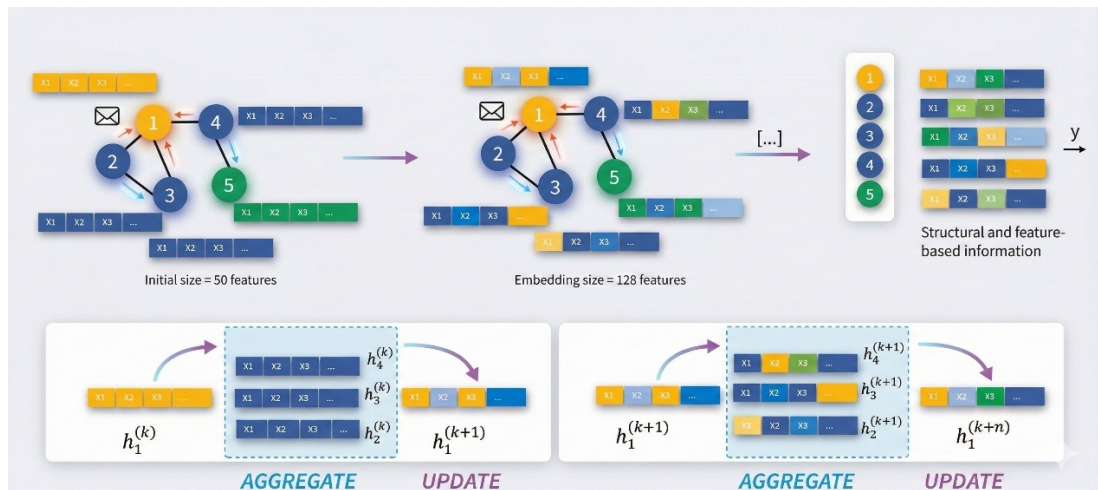


Рис. 1.8 - Механізм передавання повідомлень в GNN

Як показано на рис. 1.8, на етапі збирання інформації (AGG) кожний вузол «прослуховує» свої сусідні вузли. Він отримує їхні векторні ознаки (повідомлення) та, використовуючи агрегуювальну функцію (наприклад, середнє значення, суму або максимум), об'єднує їх в одне агреговане повідомлення. Це дає змогу вузлу отримати узагальнену

інформацію про стан свого локального оточення.

Після отримання агрегованого повідомлення вузол оновлює власний стан (UPD). Він поєднує свій поточний вектор ознак із агрегованим повідомленням, зазвичай за допомогою невеликої нейронної мережі. У результаті вузол отримує новий, більш інформативний вектор ознак, який містить знання не лише про сам вузол, а й про його сусідів.

Математично цей двоетапний процес може бути записаний як:

$$h_i^{(l+1)} = \text{UPD}^{(l)}(h_i^{(l)}, \text{AGG}^{(l)}(\{h_j^{(l)} : j \in N(i)\})), \quad (1.5)$$

де $h_i^{(l)}$ — вектор ознак (векторне представлення) вузла i на рівні l , $N(i)$ — множина сусідів вузла i .

Дві найбільш впливові індуктивні архітектури для представлення графів — GraphSAGE та Graph Attention Network (GAT).

GraphSAGE (Sample and Aggregate)

Індуктивна архітектура, яка навчається створювати універсальну агрегуювальну функцію для сусідніх ознак [34]. Як агрегатор середнього використовувалась формула:

$$h_i^{(l+1)} = \sigma\left(W \cdot \text{CONC}\left(h_i^{(l)}, \frac{1}{|N(i)|} \sum_{j \in N(i)} h_j^{(l)}\right)\right), \quad (1.6)$$

де $h_i^{(l+1)}$ — нове векторне представлення для вузла i на наступному рівні $l + 1$, σ — нелінійна активаційна функція (наприклад, ReLU), W — матриця ваг, яка навчається, CONC — операція конкатенації двох векторів, $\frac{1}{|N(i)|} \sum h_j^{(l)}$ — агреговане повідомлення сусідів.

Сусідні вектори усереднюються, об'єднуються та передаються через перетворювальну нейронну мережу. Це робить GraphSAGE стійким до змін структури графа.

Graph Attention Network (GAT)

Більш просунута архітектура, що використовує механізм уваги (attention) для динамічного визначення важливості сусідніх вузлів під час агрегування [35]. Коефіцієнти уваги α_{ij} обчислюються для кожного сусіда j вузла i :

$$\alpha_{ij} = \text{softmax}_j(\text{LeakyReLU}(\mathbf{a}^T [Wh_i \parallel Wh_j])), \quad (1.7)$$

де α_{ij} — коефіцієнт уваги (важливості), W — матриця ваг, h_i, h_j — вектори ознак вузлів i та j , $\parallel$ — оператор конкатенації, $\mathbf{a}^T$ — параметри механізму уваги. Нормування здійснюється функцією softmax, яка робить суму коефіцієнтів рівною 1.

Оновлений вектор вузла визначається як зважена сума перетворених ознак сусідів:

$$h'_i = \sigma \left(\sum_{j \in N(i)} \alpha_{ij} W h_j \right), \quad (1.8)$$

де h'_i — новий вектор ознак вузла i , $\sum \alpha_{ij} W h_j$ — агреговане повідомлення як зважена комбінація векторних представлень сусідів.

Незважаючи на здатність враховувати топологію, класичні GNN (GraphSAGE, GAT) обробляють лише один «знімок» графа і не моделюють часову динаміку. Для транспортних задач, де затори формуються та розсмоктуються протягом хвилин, це є суттєвим обмеженням: модель не здатна розрізнити затор, що наростає, від заторів, що вже зникає. Це мотивує перехід до просторово-часових архітектур.

Часові графові нейронні мережі (TGNN)

TGNN аналізують послідовності графових «знімків» у часі, що дозволяє враховувати динаміку дорожнього руху, наприклад поширення заторів [36].

Модель отримує на вхід послідовність станів графа за кілька попередніх часових кроків (наприклад $\{X^{(t-2)}, X^{(t-1)}, X^{(t)}\}$). Кожен такий стан або «знімок» є вектором ознак для всіх вузлів графа у певний момент часу.

Загальну архітектуру TGNN показано на рис. 1.9. Кожен «знімок» графа послідовно проходить через GNN. Завдання GNN полягає у захопленні просторової структури графа в даний момент. Вона агрегує інформацію з сусідніх вузлів, створюючи проміжні ознаки $X'^{(t)}$, які відображають локальне оточення вузла.

На рис. 1.10 зображено архітектуру моделі. Послідовність ознак $\{X'^{(t-2)}, X'^{(t-1)}, X'^{(t)}\}$ подається до шару Gated Recurrent Unit (GRU) [63]. GRU має внутрішню «пам'ять» — прихований стан $H^{(t-1)}$, що зберігає інформацію про попередні стани системи. На кожному кроці GRU

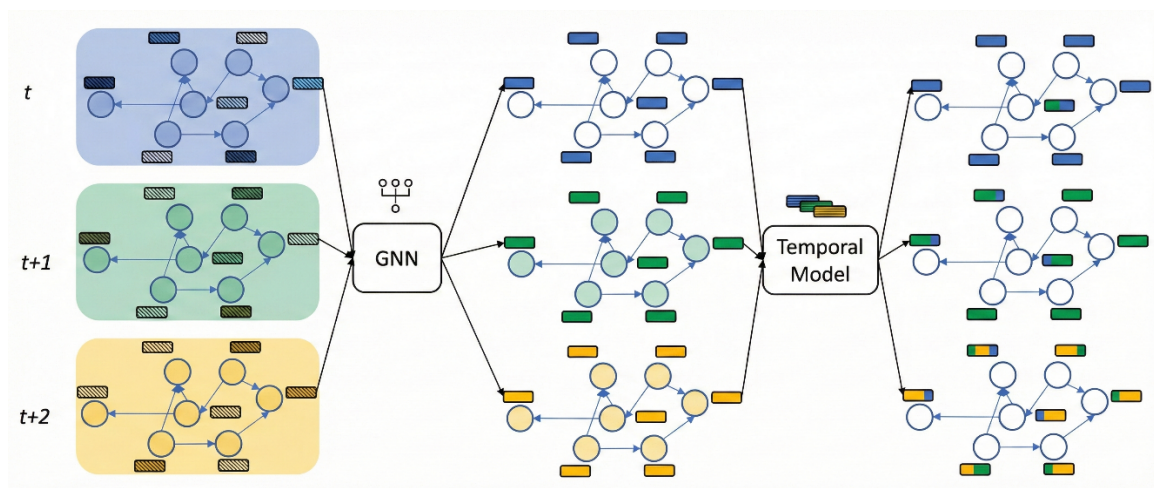


Рис. 1.9 - Загальна архітектура часової графової нейронної мережі (TGNN)

оновлює цей стан, включаючи нові просторові ознаки. Це дає змогу моделі виявляти часові закономірності та залежності (наприклад, формування або ріст заторів).

Фінальний прихований стан $H^{(t)}$ містить об'єднану просторово-часову інформацію. Він подається до лінійного шару, який формує підсумковий прогноз (наприклад, середньої швидкості кожного вузла на наступному часовому кроці).

Серед конкретних реалізацій архітектури TGNN для задач прогнозування трафіку найбільш впливовими є три моделі, що визначили напрямок розвитку цієї галузі.

DCRNN (Diffusion Convolutional Recurrent Neural Network). Модель DCRNN [39] моделює просторове поширення заторів як процес дифузії на графі. Просторова складова реалізована через дифузійну згортку (*Diffusion Convolution*), яка апроксимує марківський процес випадкового блукання на графі суміжності. Часова складова побудована на архітектурі «послідовність-в-послідовність» (*Seq2Seq*) з використанням GRU як кодера та декодера. DCRNN містить приблизно 300 тис. параметрів та забезпечує на бенчмарку METR-LA середню абсолютну помилку $MAE = 2,77$ mph для горизонту прогнозування 5 хвилин. Ця модель є де-факто еталоном для порівняння нових архітектур прогнозування трафіку.

STGCN (Spatio-Temporal Graph Convolutional Network). Модель STGCN [40] використовує принципово інший підхід до моделю-

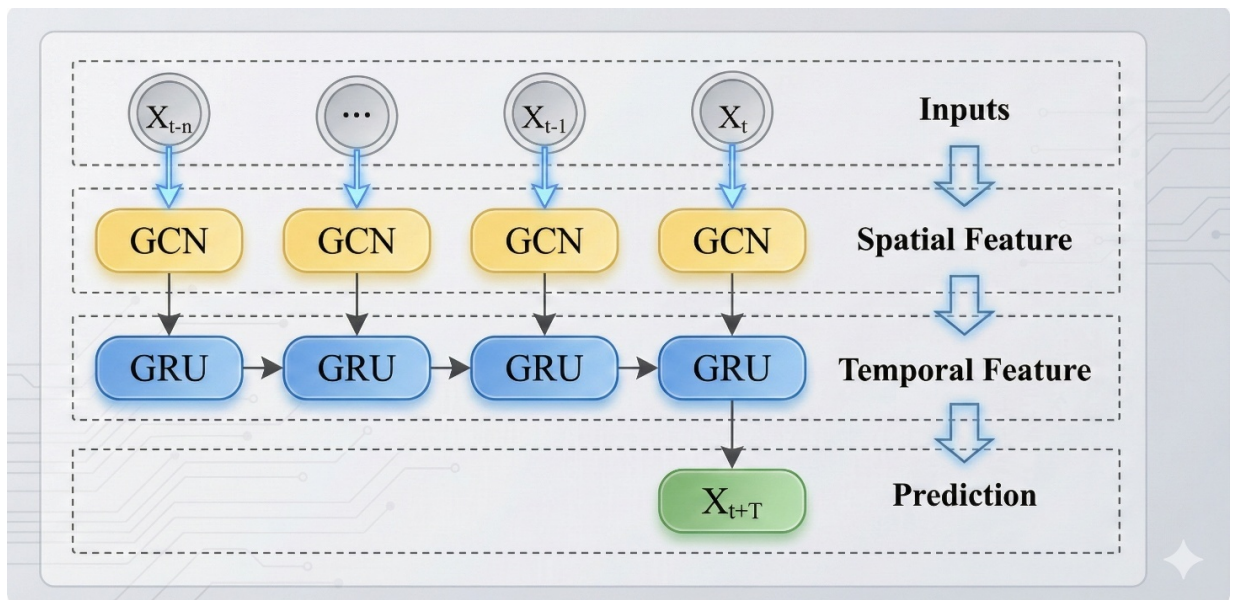


Рис. 1.10 - Архітектура просторово-часової моделі, що поєднує шари GCN для вилучення просторових ознак та блок GRU для аналізу часових залежностей

вання часових залежностей: замість рекурентних блоків (GRU/LSTM) застосовуються **часові згортки** (*temporal convolutions*) з механізмом вентиляційної лінійної одиниці (*Gated Linear Unit, GLU*). Просторова складова реалізована через спектральну графову згортку за перетворенням Чебишева. Перевагою STGCN є можливість паралелізації обчислень (на відміну від послідовного характеру GRU), що забезпечує вищу швидкість навчання та формування прогнозу [105]. Проте модель використовує фіксовану матрицю суміжності та не адаптується до змін у структурі графа.

Graph WaveNet [41]. Модель Graph WaveNet [41, 104] поєднує адаптивну матрицю суміжності, що навчається під час тренування, з розширеними причинно-наслідковими згортками (*dilated causal convolutions*). Ключовою інновацією є механізм **адаптивної суміжності**: замість використання лише фізичної топології графа, модель навчає додаткову матрицю прихованих зв'язків, що дозволяє враховувати взаємний вплив між вузлами, які не є безпосередніми сусідами у фізичній мережі доріг. На METR-LA модель демонструє $MAE = 2,69$ mph для горизонту 5 хвилин.

Усі три моделі належать до класу TGNN та демонструють високу точність на стандартних бенчмарках прогнозування трафіку.

Проведений аналіз підтверджує чітку еволюцію підходів: від статистичних моделей (LightGBM), які ігнорують топологію графа, через просторові GNN, які не враховують часову динаміку, до просторово-часових TGNN, які поєднують обидва аспекти. Саме ця послідовність обґрунтовує вибір архітектури TGNN для побудови часозалежної евристики алгоритму A^* .

Порівняльний аналіз методів

Результати порівняльного аналізу розглянутих підходів у контексті задачі формування евристики наведено в табл. 1.3.

Таблиця 1.3 - Порівняльний аналіз підходів моделей машинного навчання для пошуку евристик

Метод	Принцип роботи	Переваги	Обмеження
LightGBM [32]	Статистичний ансамбль дерев рішень (Gradient Boosting)	1. Висока швидкість обробки табличних даних. 2. Висока точність прогнозування.	1. Не використовує топологію графа (не «бачить» структуру мережі). 2. Може генерувати неузгоджені прогнози для суміжних вузлів.
GNN (Graph Neural Networks) [33]	Агрегація просторових ознак	Враховує топологію графа (структуру зв'язків між вузлами).	Не враховує часову динаміку (історію змін станів).

Метод	Принцип роботи	Переваги	Обмеження
DCRNN [39]	Дифузійна згортка + GRU (Seq2Seq)	1. Моделює просторове поширення заторів. 2. Еталон точності (MAE = 2,77 mph на METR-LA).	1. ~300 тис. параметрів. 2. Послідовний декодер обмежує паралелізацію.
STGCN [40]	Спектральна графова згортка + часові згортки (GLU)	1. Повна паралелізація обчислень. 2. Висока швидкість навчання.	1. Фіксована матриця суміжності. 2. Не адаптується до змін структури графа.
Graph WaveNet [41]	Адаптивна суміжність + розширені причинно-наслідкові згортки	1. Навчає приховані зв'язки між вузлами. 2. MAE = 2,69 mph на METR-LA.	Висока складність навчання адаптивної матриці.
TGNN (Temporal Graph Neural Networks) [36]	Просторово-часове прогнозування (комбінація GNN + GRU)	1. Враховує і топологію графа, і часову динаміку. 2. Найбільш повний опис системи.	Найвища обчислювальна складність (потребує більше ресурсів для навчання).

Аналіз підтверджує, що для побудови ефективної евристики в динамічному середовищі необхідна архітектура TGNN, яка здатна моделювати просторово-часові залежності.

При цьому існують роботи з навчання евристик для алгоритму A^* на основі нейронних мереж. Зокрема, Neural A^* [43] навчає диференційовану евристику на сіткових графах для задач роботизованої навігації. Проте цей підхід оперує на статичних сітках (grid maps) з незмінними вагами, тоді як транспортні графи є часозалежними з динамічними вагами ребер. Методи прогнозування трафіку на графових нейромережах (DCRNN, Graph WaveNet, STGCN) використовуються виключно для прогнозування швидкості або часу подорожі, а їх **інтеграція як часозалежної евристичної функції $h(n, t)$** в алгоритм A^* з адаптивними вагами ребер на графах IoV-мережі — залишається невирішеною задачею. В Розділі 3 буде запропоновано використання TGNN як евристичної функції A^* для задачі маршрутизації в часозалежних IoV-мережах.

1.5 Аналіз методів прийняття рішень та обробки даних в умовах невизначеності

Попередній аналіз особливостей навантаження IoV-систем виявив, що вхідний потік даних характеризується високим рівнем стохастичності, неповнотою та наявністю шуму. У такому середовищі параметри транспортного потоку є випадковими величинами, а межі між різними станами системи є розмитими.

Класичні детерміновані алгоритми керування, що базуються на чітких порогових правилах, виявляються неефективними в умовах невизначеності. Використання жорстких умов призводить до нестабільності системи та ефекту, коли незначні коливання вимірів викликають часті та суперечливі перемикання режимів роботи.

Для побудови адаптивної системи маршрутизації, здатної адекватно реагувати на нечітку динаміку трафіку, необхідне застосування методів м'яких обчислень та інтелектуального аналізу даних.

У цьому підрозділі проведено аналіз доцільності використання двох ключових груп методів для вирішення задачі адаптивного оновлення графа:

Теорія нечітких множин (Fuzzy Logic): для формалізації експертних знань та прийняття рішень в умовах невизначеності. Нечіткий контролер дозволяє замінити жорсткі порогові правила на гнучкий

механізм адаптивного керування коефіцієнтом згладжування.

Кластерний аналіз: для автоматичного виявлення типових режимів руху (шаблонів поведінки трафіку) без участі експерта. Кластеризація забезпечує формування бази знань для нечіткого контролера, що усуває необхідність ручного налаштування функцій належності для кожного ребра графа.

Варто врахувати, що методи глибокого навчання (TGNN) детально розглянуті у попередньому підрозділі (п. 1.3) і використовуються як складова запропонованих методів прогнозування та пошуку шляху. Математичний апарат багатовимірною статистичного аналізу, зокрема відстань Махаланобіса та коефіцієнт Бхаттачар'ї, які забезпечують оцінку близькості та подібності кластерів у просторі трафікових ознак, детально розглядається у розділі 2 при викладенні запропонованого методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі.

1.5.1 Доцільність застосування апарату нечіткої логіки для адаптивного керування параметрами системи

Аналіз природи даних у транспортних системах показує [12, 22], що більшість параметрів (наприклад, «інтенсивність руху» або «рівень затору») не мають чітких фізичних меж. Використання традиційного математичного апарату теорії множин, якому властива бінарна однозначність (елемент або належить множині, або ні), призводить до втрати інформації при описі таких «розмитих» станів [42].

У цьому контексті доцільним є використання апарату **нечіткої логіки** (*Fuzzy Logic*), запропонованого Л. Заде [42, 97, 109]. Концептуальною основою цього підходу є відмова від булевої логіки на користь багатозначної логіки, що дозволяє формалізувати експертні судження та оперувати лінгвістичними змінними в умовах невизначеності.

Математична формалізація нечіткої множини

На відміну від класичної («чіткої») множини, де характеристична функція приймає лише значення $\{0, 1\}$, у теорії нечіткої логіки вводиться поняття **функції належності**. Нехай X — універсальна множина, що містить усі можливі значення вхідного параметра (наприклад, швидкості). Нечітка множина $\tilde{A}$ на X визначається сукупністю впорядкованих пар:

$\tilde{A} = \{ (x, \mu_{\tilde{A}}(x)) \mid x \in X \}$, де $\mu_{\tilde{A}}(x) : X \rightarrow [0, 1]$ функція належності, яка ставить у відповідність кожному елементу x ступінь його належності нечіткому множині $\tilde{A}$. Це положення дозволяє побудувати логічну систему, в якій істиннісні значення висловлювань можуть набувати будь-яких дійсних значень з інтервалу $[0; 1]$, що забезпечує математичний опис понять «частково істинно» або «майже затор».

Структура процесу нечіткого керування

Процес обробки даних у системі адаптивної оцінки стану сегментів мережі на базі нечіткої логіки реалізується як послідовність трьох етапів: фазифікації, нечіткого логічного виведення та дефазифікації (рис. 1.11).

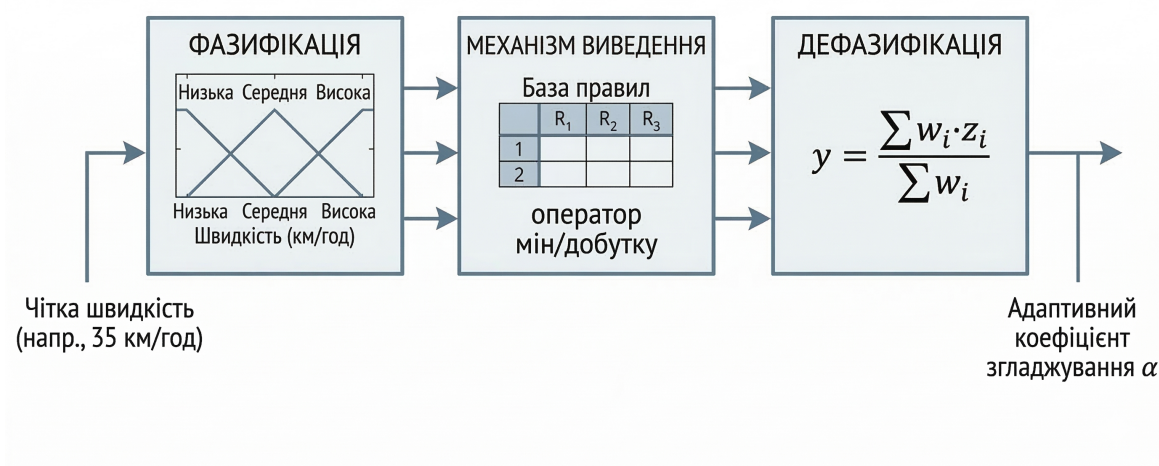


Рис. 1.11 - Структура нечіткого контролера Такагі–Сугено для адаптивної оцінки стану сегментів IoV-мережі

На етапі фазифікації відбувається перетворення чітких вхідних величин (наприклад, виміряна швидкість $v = 15$ км/год, дисперсія $\sigma = 5,2$) у лінгвістичні змінні за допомогою функцій належності. Для задач реального часу, де критичною є обчислювальна складність, найчастіше використовуються кусково-лінійні функції (трикутні або трапецієподібні) [42]. Вони характеризуються простотою обчислення та дозволяють однозначно встановити область визначення термів (наприклад, «Низька», «Середня», «Висока»). Результатом фазифікації є вектор ступенів істинності для кожного терма.

На етапі нечіткого логічного виведення потрібно побудувати базу

правил. Ядром системи є база нечітких правил, яка формалізує залежності між входами та виходами у вигляді продукційних правил типу: IF $(x_1 \text{ is } A_1)$ AND $(x_2 \text{ is } A_2)$ THEN $(y \text{ is } B)$

де x_i — вхідні змінні (метрики потоку), а y — вихідна змінна (коефіцієнт згладжування α).

Механізм виведення визначає ступінь активації кожного правила. Для цього застосовуються t-норми (операція MIN або добуток) для агрегації умов в антецеденті правила («IF» частина). Отриманий рівень істинності "відсікає" або масштабує функцію належності висновку («THEN» частина). У задачах керування часто використовується механізм виведення Мамдані (*Mamdani*) або більш обчислювально ефективний механізм Такагі-Сугено (*Takagi-Sugeno*), де висновок задається лінійною функцією, а не нечіткою множиною. Метод Мамдані - це класичний підхід, де і антецеденти (умови), і консеквенти (висновки) правил задаються нечіткими множинами. Перевагою цього методу є висока інтерпретованість та можливість моделювати людську логіку («Якщо швидкість НИЗЬКА, то затор СИЛЬНИЙ»). Недоліком є висока обчислювальна складність на етапі дефазифікації. Для отримання чіткого значення необхідно обчислювати центр тяжіння складної геометричної фігури (інтегрування функції належності $\mu_{\Sigma}(y)$), що є ресурсоемною операцією. У методі Такагі-Сугено висновок правила задається не нечіткою множиною, а чіткою функцією (зазвичай лінійною або константою) від вхідних змінних: IF $(x_1 \text{ is } A_1)$ THEN $y = f(x)$. Метод Такагі-Сугено [45, 46, 60, 66, 67] забезпечує обчислювальну ефективність за рахунок того, що процес дефазифікації зводиться до обчислення середньозваженого значення (Weighted Average) $y = \frac{\sum w_i z_i}{\sum w_i}$, де w_i - ступінь активації i -го правила, z_i - вихід рівняння.

Враховуючи обмеження, сформульовані у п. 1.1.3 (необхідність обробки в режимі м'якого реального часу з мінімальним навантаженням на CPU), використання методу Мамдані є недоцільним через повільну процедуру дефазифікації. Метод Такагі-Сугено є більш придатним для задач адаптивного керування інформаційними потоками, оскільки він забезпечує компактність обчислень та гарантує неперервність вихідної поверхні керування [60].

Фінальним етапом є перетворення результируючої нечіткої множини (або сукупності активованих висновків) назад у чітке числове значення, необхідне для оновлення ваги ребра графа. Найпоширенішим методом дефазифікації є метод центру тяжіння, який забезпечує найбільш плавний відгук системи. Математично чітке значення виходу y^* обчислюється як: $y^* = \frac{\int_{\min}^{\max} y \cdot \mu_{\Sigma}(y) dy}{\int_{\min}^{\max} \mu_{\Sigma}(y) dy}$, де $\mu_{\Sigma}(y)$ — агрегована функція належності вихідної змінної. У дискретному вигляді ця формула трансформується у відношення зважених сум [61].

Застосування описаного математичного апарату дозволяє вирішити проблему «жорстких порогів» у класичних алгоритмах. Нечіткий контролер виступає як інтелектуальний згладжуючий фільтр, який автоматично адаптує коефіцієнт реакції системи α залежно від ступеня впевненості у поточному режимі трафіку, забезпечуючи баланс між стабільністю та чутливістю [45].

1.5.2 Порівняльний аналіз алгоритмів кластеризації для автоматичної ідентифікації режимів руху

Застосування апарату нечіткої логіки, обґрунтоване в попередньому підрозділі, вимагає наявності чітко сформованої бази правил та функцій належності. Враховуючи гетерогенність транспортної мережі, де кожне ребро має унікальні характеристики пропускної здатності, ручне експертне налаштування термів (наприклад, меж «затору») є неможливим. Це породжує необхідність застосування методів інтелектуального аналізу даних для автоматичного виявлення типових шаблонів поведінки трафіку без учителя.

Основним інструментом для вирішення задачі розбиття вхідного потоку на однорідні групи (режими руху) є кластерний аналіз. Проте специфіка великих даних в IoV-системах накладає суворі обмеження на вибір алгоритму [46].

Базуючись на принципах обробки великих даних (обсяг, швидкість, різноманітність), можна сформулювати критичні вимоги до шуканого методу:

1. **Стійкість до обсягу та швидкості даних.** Алгоритм повинен мати низьку часову складність, в ідеалі — лінійну $O(N)$. Методи, що

вимагають побудови повної матриці відстаней $O(N^2)$, є неприйнятними, оскільки при збільшенні кількості вимірів час обробки зростає експоненційно, що порушує вимоги реального часу.

2. **Робота з числовими даними для фазифікації.** Це специфічна вимога, обумовлена подальшим використанням нечіткої логіки. Етап фазифікації вимагає переходу від числового значення до лінгвістичного терма. Тому алгоритм кластеризації повинен працювати у метричному просторі з неперервними числовими величинами (швидкість, щільність), а не з категоріальними атрибутами.
3. **Мінімальна кількість вхідних параметрів.** Алгоритм не повинен вимагати складного налаштування десятків гіперпараметрів для кожного ребра графа окремо, оскільки це унеможливило б масштабування системи.

Порівняльний аналіз класів алгоритмів

Для систематичного порівняння основних алгоритмів кластеризації [99] за сформульованими критеріями складено табл. 1.4, що містить часову складність, тип вхідних даних та кількість обов'язкових параметрів для кожного алгоритму.

Таблиця 1.4 - Порівняльний аналіз алгоритмів кластеризації

Категорії	Volume			Variety	Velocity	
	Розмір набору даних	Обробка великої розмірності	Обробка викидів	Тип набору даних	Часова складність	Вхідні параметри
Роздільні алгоритми						
K-Means	Великі	Так	Ні	Числові	$O(nkd)$	1
K-modes	Великі	Ні	Ні	Категоріальні	$O(n)$	1
K-medoids	Малі	Так	Так	Категоріальні	$O(n^2dt)$	1
PAM	Малі	Ні	Ні	Числові	$O(k(n-k)^2)$	1
CLARA	Великі	Ні	Ні	Числові	$O(k(40+k)^2 + k(n-k))$	1
CLARANS	Великі	Ні	Ні	Числові	$O(kn^2)$	2
FCM	Великі	Так	Ні	Числові	$O(ndk^2)$	1
Ієрархічні алгоритми						
BIRCH	Великі	Ні	Ні	Числові	$O(n)$	2
CURE	Великі	Так	Ні	Числові	$O(n^2 \log n)$	2
ROCK	Великі	Ні	Так	Категор. і числові	$O(n^2 + n^2 \log n)$	1
Chameleon	Великі	Так	Ні	Всі типи даних	$O(n^2)$	3
ECHIDA	Великі	Ні	Ні	Змішані дані	$O(N B(1 + \log B(m)))$	2
На основі щільності						
DBSCAN	Великі	Ні	Так	Числові	$O(n \log n)$ or $O(n^2)$	2
OPTICS	Великі	Ні	Ні	Числові	$O(n \log n)$	2
DBCLASD	Великі	Ні	Ні	Числові	$O(3n^2)$	-
DENCLUE	Великі	Так	Ні	Категоріальні	$O(\log D)$	2
На основі сітки						
Wave-Cluster	Великі	Ні	Ні	Спеціальні дані	$O(n)$	3
STING	Великі	Ні	Ні	Спеціальні дані	$O(n)$	1
CLIQUE	Великі	Так	Ні	Числові	$O(Ck + mk)$	2
OptiGrid	Великі	Так	Ні	Спеціальні дані	$O(nd)$ and $O(nd \log n)$	3

Як видно з табл. 1.4, існуючі алгоритми кластеризації належать до чотирьох основних класів: ієрархічні, щільнісні, сіткові та роздільні (ітераційні). Нейромережеві підходи на кшталт ART [106] також застосовуються для адаптивного розпізнавання патернів, однак у цій задачі вони поступаються за інтерпретованістю та простотою побудови функцій належності. Проведемо їх аналіз на відповідність сформульованим критеріям.

- **Ієрархічні алгоритми (CURE, BIRCH [50]).** Як показано у табл. 1.4, часова складність цих методів складає від $O(N^2)$ до $O(N^2 \log N)$, що суттєво перевищує допустимий рівень для IoV-систем (критерій 1). Крім того, ці методи будують деревоподібну структуру вкладених кластерів (дендрограму), головним недоліком якої є незворотність операції злиття/розділення: якщо об'єкт помилково віднесено до кластера на ранньому етапі, це неможливо виправити пізніше. У динамічному потоці, де режими руху змінюються (дрейф концепції), така жорсткість є критичним обмеженням.
- **Алгоритми на основі щільності (DBSCAN [49, 112], OPTICS).** Хоча середня складність DBSCAN складає $O(N \log N)$ (критерій 1 виконується), ці методи потребують ретельного налаштування параметра сусідства ε та мінімальної кількості точок MinPts (критерій 3 не виконується). Оскільки щільність трафіку на магістралі та у житловій зоні відрізняється на порядки, використання єдиних налаштувань DBSCAN для всього міста неможливе, а динамічний підбір ε для кожного ребра є обчислювально складною задачею.
- **Алгоритми на основі сітки (STING, Wave-Cluster) [56].** Ці методи є швидкими (складність залежить від кількості комірок, а не точок), проте їхня точність критично залежить від розмірності сітки (критерій 3 — необхідність підбору розміру комірки для кожного виміру). Для тривимірного простору ознак трафіку (η, L, τ) це може призвести до втрати важливих мікро-патернів поведінки. Крім того, квантування простору на комірки порушує критерій 2, оскільки

результатом є дискретна сітка, а не неперервні центроїди, придатні для побудови функцій належності.

Як показано вище, три з чотирьох класів алгоритмів не задовольняють повну сукупність критеріїв. Четвертий клас — **роздільні (ітераційні) алгоритми** — представлений методами K-Means [38, 57, 68, 69] та його нечіткою модифікацією FCM [86, 87, 88]. Суть методу полягає у розбитті набору даних на k попередньо визначених груп шляхом мінімізації сумарного квадратичного відхилення точок від центрів (центроїдів) кластерів.

Перевірка відповідності K-Means кожному з критеріїв:

- **Критерій 1 (швидкодія).** Часова складність однієї ітерації складає $O(N \times k \times d)$, де d — розмірність даних. Це лінійна залежність від N (при фіксованих k та d), що є найнижчою серед усіх проаналізованих алгоритмів (табл. 1.4) і дозволяє використовувати алгоритм для потокової обробки великих масивів телеметрії.
- **Критерій 2 (числові дані для фазифікації).** Результатом роботи алгоритму є координати центрів кластерів — середні арифметичні всіх точок групи у неперервному метричному просторі. Ці центри мають чіткий фізичний зміст (наприклад, «типова швидкість у заторі») і можуть бути безпосередньо використані як опорні точки для побудови трикутних функцій належності [95] на етапі фазифікації.
- **Критерій 3 (мінімум параметрів).** Єдиним обов'язковим параметром є кількість кластерів k , що підтверджується табл. 1.4 [70, 71, 72, 98, 99]. На відміну від ієрархічних методів, K-Means також легко адаптується до роботи з потоками даних через механізм міні-пакетного оновлення, де центроїди оновлюються ітеративно при надходженні нових порцій даних, що вирішує проблему обмеженої пам'яті.

Відкриті обмеження алгоритму K-Means в умовах IoV

Незважаючи на відповідність усім трьом критеріям, класичний K-Means має ряд обмежень, які залишаються невирішеними саме в контексті динамічних IoV-систем:

- *Статичність кількості кластерів k .* Алгоритм вимагає апріорного визначення числа кластерів. Для статичних даних це вирішується методом ліктя (Elbow method): алгоритм виконується для $k = 2, 3, \dots, k_{\max}$, і для кожного обчислюється інерція $J(k) = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$. Значення k^* , після якого $J(k)$ перестає суттєво зменшуватись («злам»), обирається як оптимальне. Однак у поточному режимі статистичні характеристики трафіку змінюються з часом (дрейф концепції), тому оптимальне k не є сталим — воно має динамічно адаптуватися до поточного стану мережі.
- *Чутливість до ініціалізації.* Результат роботи залежить від початкового розміщення центроїдів. Процедура K-Means++ [38, 57, 68, 69] зменшує цей ефект для одноразового запуску, але при інкрементальному оновленні кластерів на потоці даних проблема повторної ініціалізації потребує окремого механізму.
- *Відсутність механізму інкрементального оновлення.* Класичний K-Means працює з повним набором даних (batch-режим). У потоковій IoV-системі, де дані надходять безперервно, необхідний механізм інкрементального оновлення центроїдів без повного перерахунку, що виходить за межі стандартного алгоритму.

Відповідно, проведений порівняльний аналіз (табл. 1.4) показує, що серед чотирьох класів алгоритмів кластеризації саме роздільний метод K-Means найбільш повно відповідає критеріям обчислювальної ефективності, роботи з числовими даними та мінімальної кількості параметрів. Проте його класична (пакетна) реалізація не враховує динамічність транспортного потоку — зміну оптимальної кількості режимів k у часі та необхідність інкрементального оновлення. Ці невирішені обмеження формують потребу в розробці адаптивної модифікації кластеризації, яка зберігає переваги K-Means (лінійна складність, інтерпретовані центроїди) та водночас забезпечує автоматичну адаптацію до дрейфу концепції в IoV-даних. Розробка такого методу є предметом розділу 2.

1.6 Математична постановка задачі управління інформаційними потоками в IoV-мережі

Проведений у попередніх підрозділах аналіз дозволив виявити системні обмеження існуючих методів оцінки стану інформаційних потоків, пошуку шляхів та ідентифікації режимів інформаційного навантаження. На основі цього аналізу в даному підрозділі формулюється математична постановка задачі, яка визначає формальні вимоги до системи адаптивної маршрутизації.

Основою будь-якої навігаційної системи є телекомунікаційна модель IoV-мережі. В умовах високої динаміки, де параметри руху змінюються кожну хвилину, класичне представлення графа зі статичними вагами $G(V, E)$ є недостатнім. Необхідно перейти до моделі **часозалежного графа** (*Time-Dependent Graph, TDG*) [6].

Концептуальна модель такого переходу наведена на рис. 1.12 і складається з трьох послідовних етапів:

1. **Отримання стохастичного потоку S_{in} .** Сенсори транспортних засобів генерують послідовність дискретних вимірів швидкості, які містять шум вимірювання та стохастичну варіацію, зумовлену динамікою транспортного потоку. Внаслідок цього спостережувані значення мають значну дисперсію відносно істинного стану дорожнього руху, що обумовлює необхідність фільтрації.
2. **Оцінка ваги ребра $\widehat{w}_{uv}(t)$.** Адаптивний контролер виконує функцію апроксимації $\mathcal{F}$: перетворює зашумлені дискретні події на гладку неперервну функцію часу із довірчим інтервалом. Ключовою проблемою є визначення динамічного коефіцієнта згладжування α , який регулює баланс між реактивністю на нові виміри та стійкістю до шуму.
3. **Формування часозалежного графа G_T .** Результируючий орієнтований граф, де кожному ребру присвоєно не фіксовану константу, а динамічний профіль $w_{uv}(t)$, що відображає прогнозований час обслуговування інформаційного потоку на сегменті як функцію моменту входу на ребро.

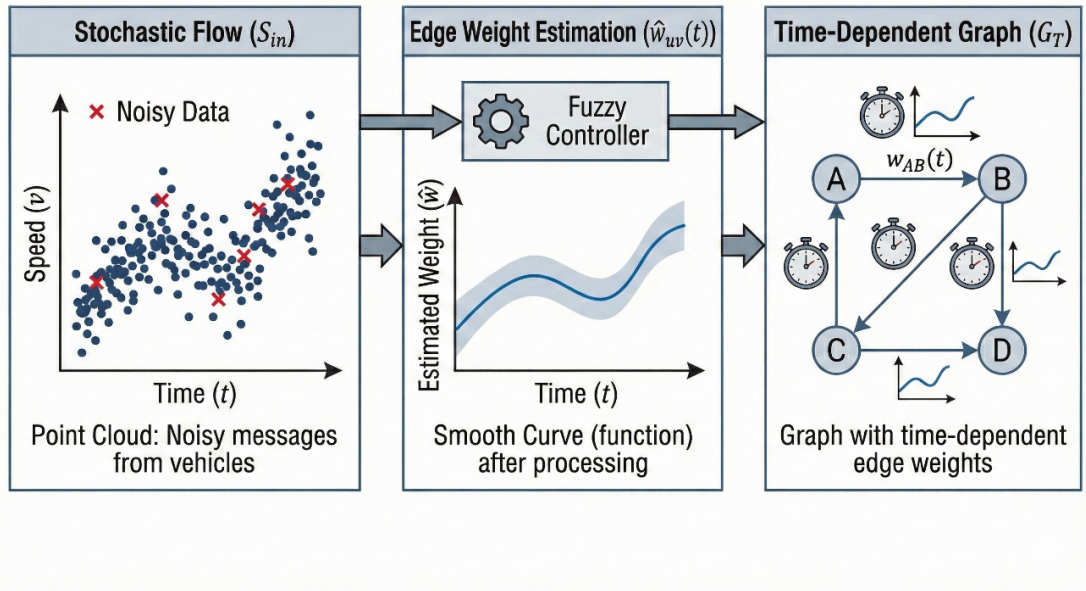


Рис. 1.12 - Концептуальна модель часозалежної IoV-мережі з інформаційними потоками

1.6.1 Формалізація телекомунікаційної моделі IoV-мережі

Для задачі управління інформаційними потоками фізична інфраструктура IoV-мережі (OBU, RSU, gNB/eNB, хмарні сервіси) подається у вигляді часозалежного зваженого орієнтованого графа $G_T = (V, E, W, T)$. Ключова відмінність від класичних графових моделей маршрутизації — вага ребра визначається як **час обслуговування інформаційного потоку на сегменті**, а не як відстань чи час проїзду. Детальний формальний опис моделі, визначення QoS-індикаторів стану сегмента $[\eta, L, \tau]$ та властивості вагової функції наведено у п. 2.1.

1.6.2 Модель вхідного потоку даних

Стан графа G_T не є відомим апріорі, а відновлюється на основі потоку вхідних даних S_{in} , що генеруються сенсорами транспортних засобів. Потік формалізується як нескінченна послідовність кортежів:

$$S_{in} = \{\dots, \mathbf{m}_k, \dots\}, \quad k \in \mathbb{N}. \quad (1.11)$$

Кожне повідомлення $\mathbf{m}_k$ характеризується вектором параметрів:

$$m_k = \langle id_{edge}, t_k, \mathbf{x}_k, v_k^{raw} \rangle, \quad (1.12)$$

де id_{edge} — унікальний ідентифікатор ребра графа, що дозволяє

однозначно прив'язати подію до топології; t_k — момент часу реєстрації події; v_k^{raw} — виміряна швидкість транспортного засобу, яка підлягає згладжуванню; $\mathbf{x}_k$ — вектор QoS-стану ребра $[\eta_k, L_k, \tau_k]^T$, що використовується для визначення коефіцієнта адаптації α .

Стохастична природа потоку полягає в тому, що спостережуване значення швидкості v_k^{raw} є виміряним значенням істинної швидкості потоку v_k^* , спотвореним випадковими похибками:

$$v_k^{\text{raw}} = v_k^* + \varepsilon_k, \quad \varepsilon_k \sim \mathcal{N}(0, \sigma_\varepsilon^2), \quad (1.13)$$

де ε_k — випадкова величина шуму вимірювання, що включає похибки GPS-позиціонування, індивідуальну поведінку водіїв та дискретизацію сенсорів. Наявність цього шуму обумовлює неможливість прямого використання v_k^{raw} як ваги ребра та потребу в адаптивному згладжуванні.

Вектор $\mathbf{x}_k \in \mathbb{R}^3$ формується з трьох нормалізованих метрик, отриманих на етапі попередньої обробки:

$$\mathbf{x}_k = \begin{bmatrix} \eta_k \\ L_k \\ \tau_k \end{bmatrix}, \quad (1.14)$$

де компоненти мають такий фізичний зміст:

- **Коефіцієнт пропускної здатності сегмента (η_k):** Безрозмірна величина $\eta_k \in [0, 1]$, що відображає відношення поточної швидкості до максимально дозволеної (швидкість вільного потоку) на даному ребрі $\eta_k = \frac{v_k^{\text{raw}}}{v_{\text{max}}}$. Вона характеризує якість каналу передачі: $\eta \approx 1.0$ відповідає вільному потоку з мінімальною конкуренцією IoT-пристроїв за канал, а $\eta \approx 0$ — глухому затору з максимальним навантаженням.
- **Коефіцієнт завантаження (L_k):** Відношення поточної щільності IoT-джерел до пропускної здатності сегмента $L_k = \frac{\rho_{\text{current}}}{C_{\text{road}}}$. Безпосередньо відображає ступінь насиченості V2X-каналу: при $L_k \rightarrow 1$ кількість активних передавачів наближається до граничної, що спричиняє зростання колізій і затримок доставки пакетів.

- **Коефіцієнт зміни завантаження каналу (τ_k):** Динамічний показник $\tau_k \approx \frac{dv}{dt}$, що відображає тенденцію зміни навантаження. Від'ємні значення ($\tau_k < 0$) сигналізують про формування ударної хвилі затору і прогнозоване погіршення QoS, тоді як $\tau_k > 0$ свідчить про розсмоктування затору і відновлення пропускної здатності каналу.

Таким чином, вектор $\mathbf{x}_k = [\eta_k, L_k, \tau_k]^T$ є **вектором QoS-стану сегмента**: він кількісно описує поточну якість обслуговування інформаційного потоку на ребрі з трьох взаємодоповнюючих точок зору — миттєвої якості каналу (η_k), його завантаженості (L_k) та динаміки зміни (τ_k). Саме цей вектор є вимірюваним представленням теоретичного показника $\lambda_{\text{seg}}(t)$ через спостережувані сенсорами величини.

Принцип розділення відповідальності реалізується так: вектор $\mathbf{x}_k$ визначає ступінь довіри до поточного виміру через коефіцієнт $\alpha(\mathbf{x}_k)$, а v_k^{raw} є вхідним спостереженням для обчислення згладженої оцінки швидкості. Оновлення виконується за формулою адаптивного експоненційного згладжування:

$$\hat{v}_{uv}(t_k) = \alpha(\mathbf{x}_k) \cdot v_k^{\text{raw}} + (1 - \alpha(\mathbf{x}_k)) \cdot \hat{v}_{uv}(t_{k-1}), \quad (1.15)$$

де

- $\hat{v}_{uv}(t_k)$ — оновлена згладжена оцінка швидкості на ребрі в момент t_k ; на її основі розраховується оцінка ваги $\hat{w}_{uv}(t_k) = L_{uv}/\hat{v}_{uv}(t_k)$, яка записується в граф;
- $\hat{v}_{uv}(t_{k-1})$ — попередня згладжена оцінка швидкості, що зберігає «пам'ять» системи про стан інформаційного навантаження;
- v_k^{raw} — нове спостереження швидкості з повідомлення m_k ;
- $\alpha(\mathbf{x}_k) \in [0, 1]$ — **динамічний коефіцієнт адаптації**, що визначається QoS-станом $\mathbf{x}_k$ і є центральною невідомою задачі.

Значення α визначає компроміс між реактивністю та стабільністю: при $\alpha \rightarrow 1$ система миттєво реагує на нові виміри (висока чутливість до шуму), при $\alpha \rightarrow 0$ — ігнорує їх (висока інерція). Цей ефект

проілюстровано на рис. 1.13: фіксоване $\alpha = 0,05$ забезпечує гладку криву, але з великою затримкою при переході до затору; $\alpha = 0,80$ — швидко реакцію, але з високим рівнем пропущеного шуму. Адаптивний коефіцієнт $\alpha(\mathbf{x}_k)$ (нижня панель) автоматично збільшується під час перехідних процесів і зменшується у стаціонарному режимі, забезпечуючи оптимальний компроміс. Як показано в п. 1.2, жодне фіксоване значення α не є оптимальним для всіх режимів руху, що мотивує розробку адаптивного методу визначення $\alpha(\mathbf{x}_k)$ на основі нечіткого виведення (розділ 2).

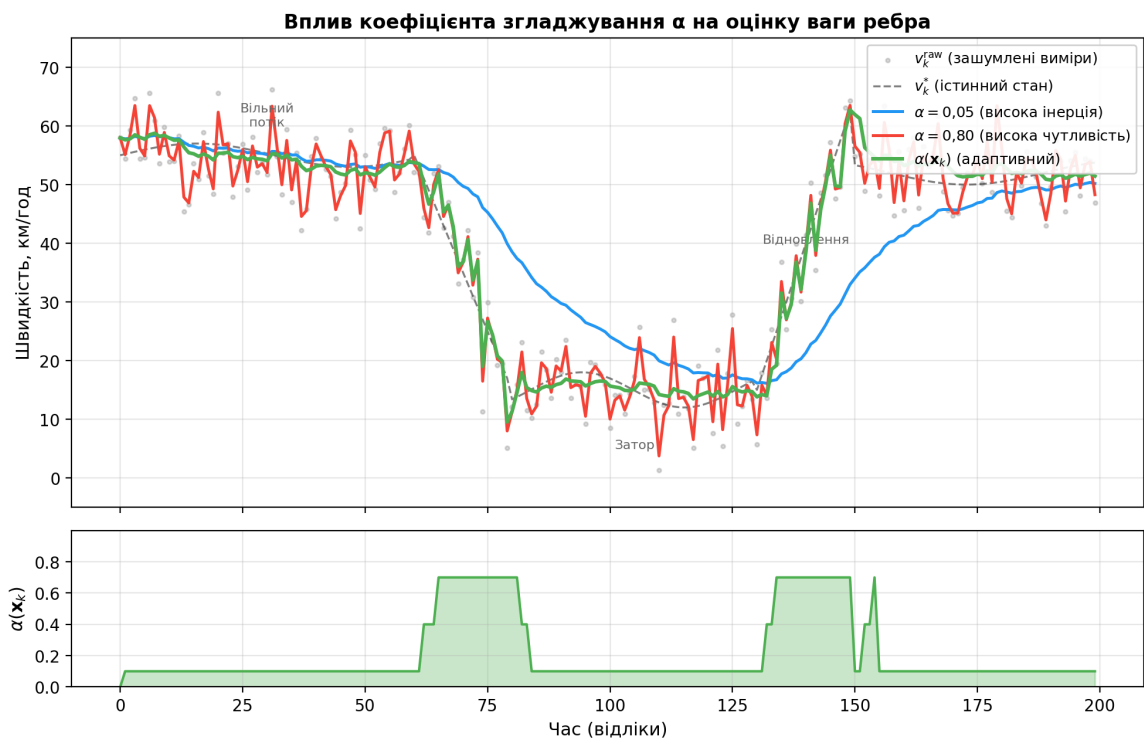


Рис. 1.13 - Вплив коефіцієнта згладжування α на оцінку ваги ребра: фіксовані $\alpha = 0,05$ (висока інерція) та $\alpha = 0,80$ (висока чутливість) порівняно з адаптивним $\alpha(\mathbf{x}_k)$. Нижня панель — динаміка коефіцієнта адаптації

1.6.3 Постановка оптимізаційної задачі

Під **ефективністю управління інформаційними потоками** в IoV-мережі розуміється сукупність трьох взаємопов'язаних показників: **час реакції системи** на зміну QoS-стану сегментів, **оптимальність маршруту** обслуговування інформаційного потоку та **обчислювальна вартість** пошуку маршруту. Задача підвищення ефективності полягає

у мінімізації цих показників одночасно в умовах динамічно змінного навантаження на сегменти IoV-мережі.

Задача адаптивної маршрутизації формулюється як пошук маршруту P^* — послідовності суміжних вершин $P = (v_1, v_2, \dots, v_K)$, де $v_1 = s$ (старт), $v_K = d$ (ціль), що мінімізує сумарний час обслуговування інформаційних потоків на сегментах IoV-мережі у часозалежному графі. Час прибуття у кожен проміжну вершину визначається рекурсивно:

$$a_1 = t_{\text{start}}, \quad a_{i+1} = a_i + \widehat{w}_{v_i, v_{i+1}}(a_i), \quad i = 1, \dots, K-1, \quad (1.16)$$

де $\widehat{w}_{v_i, v_{i+1}}(a_i)$ — оцінка часу обслуговування потоку на ребрі, отримана за формулою адаптивного згладжування (1.15). Вартість шляху відповідає сумарному часу обслуговування інформаційного потоку:

$$C(P, t_{\text{start}}) = a_K - a_1 = \sum_{i=1}^{K-1} \widehat{w}_{v_i, v_{i+1}}(a_i). \quad (1.17)$$

Оптимальний маршрут:

$$P^* = \arg \min_{P \in \mathcal{P}_{sd}} C(P, t_{\text{start}}), \quad (1.18)$$

де $\mathcal{P}_{sd}$ — множина всіх допустимих шляхів від s до d .

Розв'язання цієї задачі стикається з двома взаємопов'язаними проблемами:

1. **Проблема точності оцінки ваг.** Якість маршруту P^* визначається якістю оцінок ваг $\widehat{w}_{uv}(t)$. За наявності стохастичного шуму (1.13) завищений коефіцієнт α у формулі (1.15) пропускає шум у граф, занижений — ігнорує реальні зміни режиму інформаційного навантаження. В обох випадках алгоритм A^* знаходить оптимальний маршрут у *неточному* графі, який відрізняється від оптимального маршруту в реальній мережі.
2. **Проблема ефективності пошуку шляху.** Навіть за умови ідеальних ваг, пошук найкоротшого шляху в графі G_T з $|V| = N$ вершинами вимагає застосування евристичної функції $h(v, t)$, яка

скеровує алгоритм A^* у напрямку цілі. Без такої евристики A^* вироджується в алгоритм Дейкстри та розкриває надмірну кількість вершин, що порушує обмеження реального часу (1.19). Побудова допустимої (*admissible*) часозалежної евристики є нетривіальною задачею, оскільки стандартні статичні евристики (наприклад, геодезична відстань) не враховують динаміку транспортного потоку [30].

Цю проблему проілюстровано на рис. 1.14: статична евристика $h_{geo}(v)$, побудована на основі геодезичної відстані, скеровує A^* через вершину А, яка геометрично ближча до цілі. Проте ребро А–С перебуває в стані затору (20 хв замість очікуваних 5 хв), і результуючий маршрут $S \rightarrow A \rightarrow C \rightarrow G$ потребує 30 хв. Часозалежна евристика $h(v, t)$ враховує поточний стан трафіку й оцінює $h(A) = 18,2$, тому A^* обирає обхідний шлях $S \rightarrow B \rightarrow D \rightarrow G$ за 20 хв, розкриваючи при цьому менше вершин.

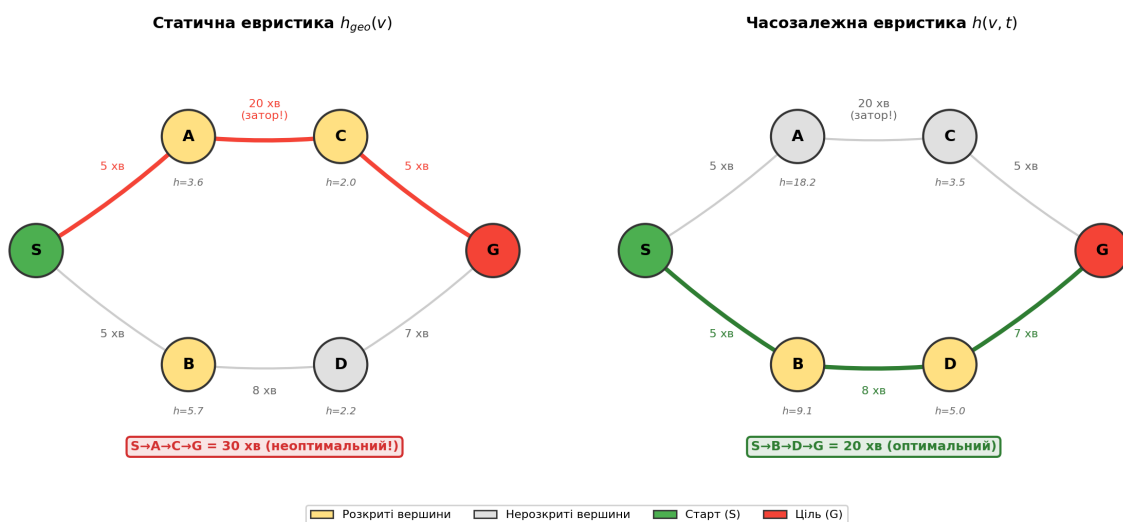


Рис. 1.14 - Порівняння статичної та часозалежної евристик: статична h_{geo} скеровує A^* у затор, динамічна $h(v, t)$ — в обхід

Обмеження задачі:

1. *Обмеження реального часу.* Сумарний час оновлення ваг T_{update} та розрахунку маршруту T_{calc} не повинен перевищувати допустимої затримки:

$$T_{update} + T_{calc} \leq T_{deadline}. \quad (1.19)$$

2. *Обмеження алгоритмічної складності.* Кількість розкритих вершин $N_{explored}$ при пошуку має бути суттєво меншою за повний перебір: $N_{explored} \ll |V|$.

Таким чином, наукова задача декомпонується на три підзадачі (рис. 1.15):

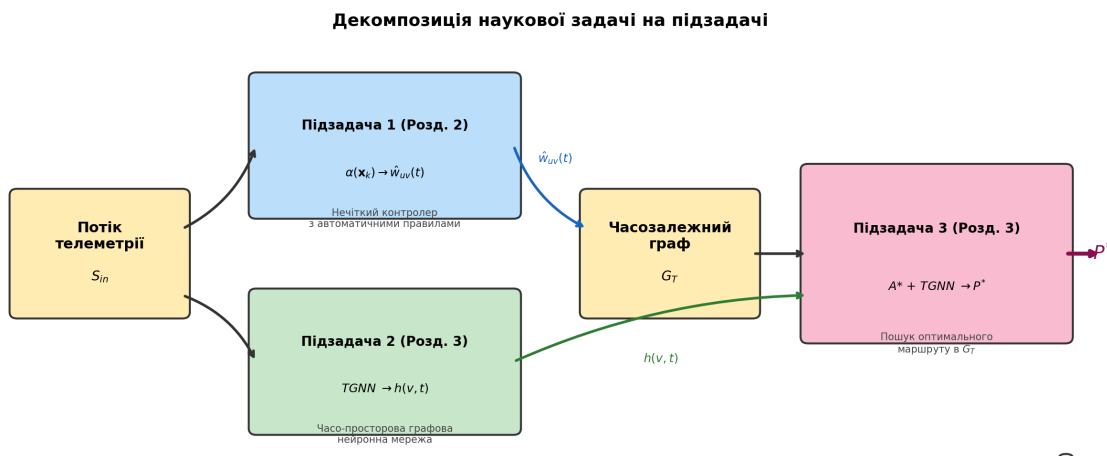


Рис. 1.15 - Декомпозиція наукової задачі на підзадачі та їх взаємозв'язок

1. Розробка методу адаптивної оцінки стану інформаційних потоків графа $\widehat{w_{uv}}(t)$ на основі нечіткого контролера з автоматично побудованою базою правил.
2. Розробка методу формування часозалежної евристики $h(v, t)$ на основі часо-просторової графової нейронної мережі (TGNN).
3. Інтеграція цих методів в узгоджений алгоритм маршрутизації інформаційних потоків для пошуку оптимального шляху P^* .

Висновки до розділу 1

1. Виявлено особливості формування інформаційних потоків в IoV-мережах розумного міста. Показано, що архітектура IoV поєднує мобільні IoT-джерела, RSU, базові станції, вузли попередньої обробки та хмарні сервіси в єдину інформаційно-комунікаційну систему, а навантаження має виражену просторово-часову нерівномірність.
2. Сформульовано задачу управління інформаційними потоками автономних транспортних засобів. У контексті задачі ключовою характеристикою об'єкта дослідження визначено просторово-часовий розподіл інформаційного навантаження на сегментах IoV-мережі, вимірюваним представленням якого є вектор QoS-стану сегмента — коефіцієнт пропускної здатності сегмента η , коефіцієнт завантаження L та коефіцієнт зміни завантаження каналу τ .
3. Проаналізовано існуючі методи оцінки стану інформаційних потоків, маршрутизації та прийняття рішень в умовах невизначеності. Встановлено, що пакетна обробка, ковзне вікно та фіксоване експоненціальне згладжування не забезпечують необхідної швидкодії і стійкості до шуму, а класичні алгоритми пошуку шляху не враховують очікуваного майбутнього стану сегментів мережі.
4. Обґрунтовано необхідність розробки нових моделей і методів: телекомунікаційної моделі IoV-мережі з QoS-орієнтованою вагою ребра, методу адаптивної оцінки стану інформаційних потоків, методу просторово-часового прогнозування інформаційного навантаження на основі TGNN та методу пошуку оптимального маршруту обслуговування інформаційних потоків на основі модифікованого алгоритму A^* .
5. Розроблено математичну постановку задачі управління інформаційними потоками в IoV-мережі на основі єдиної часозалежної моделі $G_T = (V, E, W(t), T)$, моделі вхідного потоку S_{in} та оптимізаційної задачі визначення маршруту P^* , що створює формальну основу для методів, поданих у розділах 2–4.

РОЗДІЛ 2

МЕТОД АДАПТИВНОЇ ОЦІНКИ СТАНУ ІНФОРМАЦІЙНИХ ПОТОКІВ НА СЕГМЕНТАХ ІОВ-МЕРЕЖІ

У першому розділі було встановлено, що класичні методи оцінки стану інформаційних потоків (фіксоване експоненціальне згладжування, ковзне середнє) [27, 74, 75, 76] використовують статичний коефіцієнт згладжування α , що спричиняє конфлікт між чутливістю до змін та фільтрацією шуму вимірювань параметрів інформаційного потоку [7, 8]. Промислові підходи — фільтр Калмана (HERE/TomTom) [54, 108], баєсівське злиття (Waze) — адаптують параметри лише на основі шумових характеристик сигналу, без урахування режиму інформаційного навантаження [1, 2, 13, 17].

У цьому розділі пропонується метод адаптивної оцінки стану інформаційних потоків на сегментах Іов-мережі, що складається з двох взаємопов'язаних компонентів:

1. **Автоматична побудова бази нечітких правил** на основі пакетно-інкрементальної кластеризації [45, 46, 64, 65] вхідного потоку Іот-даних у тривимірному просторі ознак станів інформаційного навантаження.
2. **Адаптивна оцінка стану потоку** на основі нечіткого контролера типу Такагі-Сугено [60, 66, 96], який використовує згенеровану базу правил для динамічного визначення коефіцієнта адаптації $\alpha \in [0,01; 1]$ залежно від поточного режиму інформаційного навантаження.

Метод адаптивної оцінки стану інформаційних потоків на сегментах Іов-мережі формує компоненту $g(n)$ — фактичну характеристику обслуговування вже пройденої частини шляху — у формулі оцінки вершини алгоритму A^* . Компонента $h(n, t)$ — прогнозна евристика на основі часо-просторової графової нейронної мережі (TGNN) — розглядається у розділі 3.

Математична постановка задачі адаптивної оцінки стану інформаційних потоків у стохастичному часозалежному графі, включаючи

формалізацію телекомунікаційної моделі IoV-мережі, вхідного потоку даних та оптимізаційної задачі, наведена у п. 2.1 та п. 1.6.

2.1 Телекомунікаційна модель IoV-мережі

У класичних графових моделях маршрутизації [5, 6, 30] вага ребра визначається як відстань або час проїзду — фізична характеристика дорожньої інфраструктури. Для телекомунікаційної задачі управління інформаційними потоками IoV-мережі така інтерпретація є недостатньою: вона не відображає динаміку завантаженості V2X-каналу і не дозволяє враховувати QoS-стан сегмента при виборі маршруту. У запропонованій моделі вага ребра має телекомунікаційний зміст — вона визначається як **час обслуговування інформаційного потоку на сегменті**, а вимірювані QoS-індикатори стану каналу визначають адаптивне оновлення оцінки швидкості, на основі якої ця вага обчислюється.

На фізичному телекомунікаційному рівні IoV-мережа включає мобільні OBU-вузли (бортові пристрої транспортних засобів — джерела інформаційних потоків), придорожні RSU-вузли (Road Side Unit — вузли першого рівня агрегування телеметрії), базові станції gNB/eNB (вузли другого рівня, що забезпечують зв'язок з хмарними сервісами) та хмарні сервіси, які взаємодіють через V2X-канали. Для задачі управління інформаційними потоками ця інфраструктура подається у вигляді часозалежного зваженого орієнтованого графа:

$$G_T = (V, E, W, T), \quad (2.1)$$

де

- $V = \{v_1, v_2, \dots, v_n\}$ — множина вершин (вузлів агрегування та маршрутизації інформаційних потоків: RSU, gNB/eNB, хмарні агрегатори), $|V| = N$;
- $E \subseteq V \times V$ — множина орієнтованих ребер (сегментів обміну між суміжними вузлами мережі), $|E| = M$;
- T — часовий вимір (неперервний або дискретизований з кроком Δt);

- $W : E \times T \rightarrow \mathbb{R}^+$ — функція ваги ребра, що задає час обслуговування інформаційного потоку на сегменті як функцію часу входу на ребро.

Структуру розробленої моделі наведено на рис. 2.1.

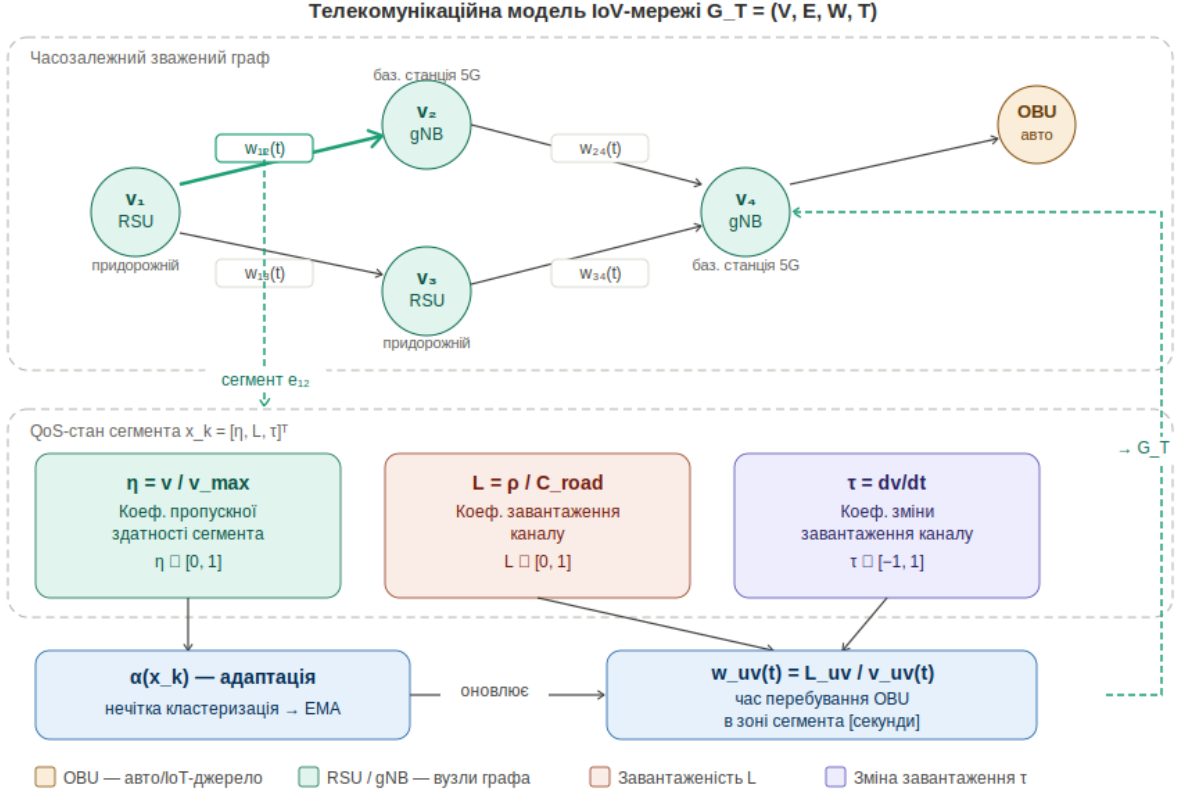


Рис. 2.1 - Телекомунікаційна модель IoV-мережі $G_T = (V, E, W, T)$: граф з вузлами OBU, RSU, gNB та QoS-індикаторами стану сегмента $[\eta, L, \tau]$, що визначають адаптивне оновлення оцінки швидкості та через неї — вагу ребра $w_{uv}(t)$

Для кожного ребра $e_{uv} \in E$ вагова функція визначає **час перебування мобільного IoT-вузла (OBU) в зоні активної генерації V2X-даних на сегменті**:

$$w_{uv}(t) = \frac{L_{uv}}{v_{uv}(t)}, \quad (2.1a)$$

де L_{uv} — довжина сегмента, $v_{uv}(t)$ — поточна швидкість потоку. Оскільки через фундаментальну діаграму транспортного потоку [80, 84] зростання щільності IoT-джерел спричиняє зниження швидкості, вага ребра є природним телекомунікаційним індикатором навантаження: чим більше OBU на сегменті, тим нижча швидкість і тим більший $w_{uv}(t)$.

QoS-стан сегмента описується вектором вимірюваних індикаторів $\mathbf{x}_k = [\eta_k, L_k, \tau_k]^T$, де:

- $\eta_k = v_k/v_{\max}$ — **коефіцієнт пропускної здатності сегмента**: відображає наскільки вільний канал для передачі даних; $\eta \approx 1$ — вільний потік, $\eta \approx 0$ — перевантаження;
- $L_k = \rho_{\text{current}}/C_{\text{road}}$ — **коефіцієнт завантаження каналу**: відображає насиченість каналу IoT-джерелами; $L \in [0, 1]$;
- $\tau_k \approx dv/dt$ — **коефіцієнт зміни завантаження каналу**: показує динаміку QoS — наростання ($\tau < 0$) або відновлення ($\tau > 0$) завантаженості; $\tau \in [-1, 1]$.

Вектор $\mathbf{x}_k$ є вимірюваним представленням теоретичного показника $\lambda_{\text{seg}}(t)$ через спостережувані сенсорами величини. QoS-індикатори використовуються для визначення коефіцієнта адаптивного згладжування $\alpha(\mathbf{x}_k)$ (детально — у п. 2.2), який регулює швидкість оновлення згладженої оцінки швидкості $\hat{v}_{uv}(t)$ залежно від поточного стану каналу; після цього вага ребра обчислюється як $w_{uv}(t) = L_{uv}/\hat{v}_{uv}(t)$.

Обов'язковою вимогою до функції $w_{uv}(t)$ є виконання умови **FIFO** (необгону) [5]: пізніший вхід на ребро не може дати раніше прибуття у вершину v :

$$\forall t_1, t_2 \in T : t_1 < t_2 \Rightarrow (t_1 + w_{uv}(t_1)) \leq (t_2 + w_{uv}(t_2)). \quad (2.2)$$

Виконання умови FIFO забезпечує коректність застосування алгоритму A^* для пошуку оптимального маршруту [30].

Таким чином, **мінімізація суми $\sum w_{uv}(t)$ вздовж маршруту рівнозначна вибору шляху через сегменти з мінімальним сумарним інформаційним навантаженням в IoV-мережі.**

2.2 Автоматична побудова бази нечітких правил на основі кластеризації

Ефективність функціонування підсистеми адаптивного оновлення графа, описаної в п. 1.5, критично залежить від якості налаштування бази знань нечіткого контролера. База знань складається з двох компонентів:

лінгвістичних змінних (функцій належності, що визначають терми «Затор», «Вільний потік» тощо) та правил логічного виведення.

У традиційних системах керування інформаційними потоками ці параметри задаються експертним шляхом на основі апріорних знань про пропускну здатність дороги [45, 46, 102]. Однак у масштабах мережі Інтернету транспортних засобів (IoV), яка охоплює тисячі ребер з гетерогенними характеристиками [21, 22], такий підхід стає неможливим з двох причин:

- *Проблема масштабування:* Ручне налаштування меж лінгвістичних термів для кожного окремого сегмента дорожньої мережі потребує неприпустимо великих часових витрат.
- *Проблема локального контексту:* Поняття «низька швидкість» є відносним. Для швидкісної магістралі зниження швидкості до 40 км/год свідчить про критичний затор, тоді як для житлової зони це є нормальним режимом руху. Використання універсальних (глобальних) порогів призводить до хибних спрацювань системи адаптації.

Для вирішення цих проблем у запропонованому методі адаптивної оцінки стану інформаційних потоків передбачено **автоматичну генерацію бази правил**. Основна гіпотеза методу полягає в тому, що сталі режими руху (патерни) формують компактні, статистично розрізнені групи (кластери) у багатовимірному просторі ознак станів транспортного потоку [51, 86, 90].

Метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі [45, 46] дозволяє перейти від експертного задання правил до їх автоматичного вилучення з історичних даних без участі вчителя [45, 46, 102]. Процес реалізується як конвеєр послідовних перетворень: формування простору ознак → пакетна кластеризація → імовірнісна валідація → генерація нечітких термів.

Метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі функціонує за конвеєрним принципом. Структурно він складається з п'яти послідовних функціональних блоків, кожен з яких виконує специфічну трансформацію інформаційного потоку: від «сирих»

векторів стану транспортного потоку до лінгвістично інтерпретованих правил керування. Узагальнена структурна схема методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі наведена на рис. 2.2.

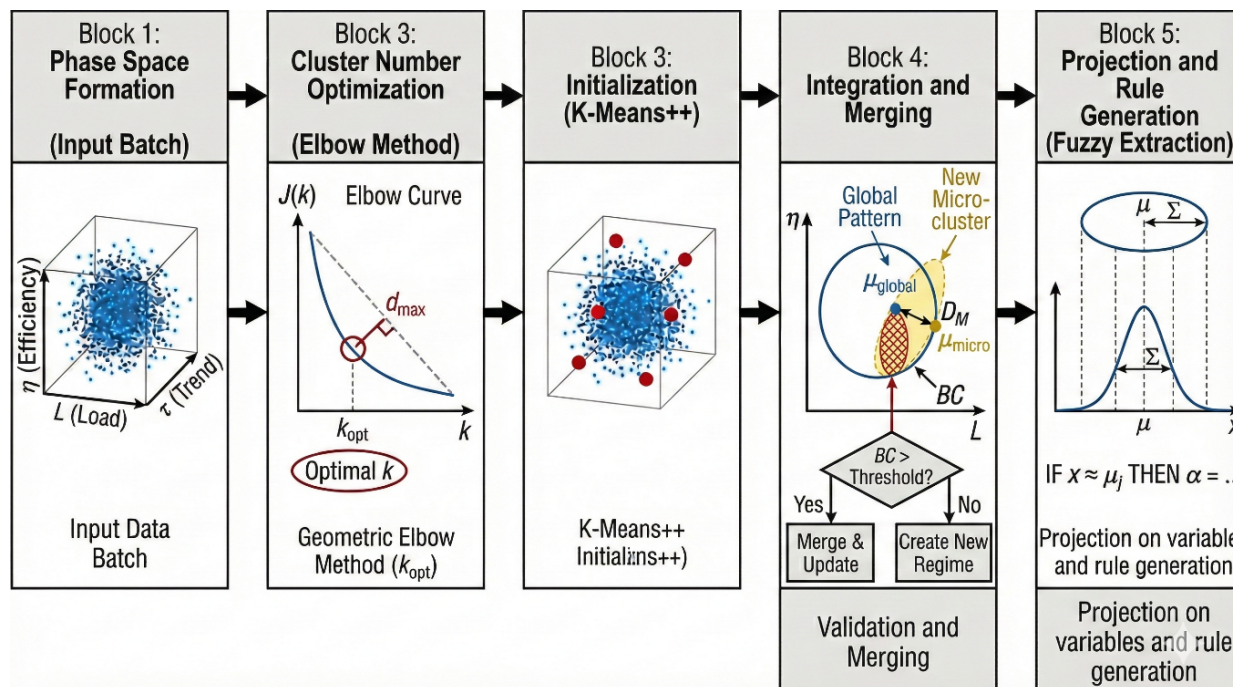


Рис. 2.2 - Структурна схема методу адаптивної кластеризації та генерації нечітких правил

Розглянемо детально функціональне призначення та логіку взаємодії представлених на схемі блоків.

Блок 1: Формування простору ознак. Перший етап відповідає за попередню обробку вхідних даних. З потоку телеметрії обчислюються три вимірювані QoS-індикатори стану сегмента, що кількісно виражають теоретичний показник $\lambda_{\text{seg}}(t)$ через спостережувані величини [11, 12]:

- **Коефіцієнт пропускну здатності сегмента (η):** характеризує поточну якість V2X-каналу — наскільки вільний сегмент для передачі даних;
- **Коефіцієнт завантаження (L):** відображає насиченість каналу IoT-джерелами — співвідношення активних передавачів до пропускну здатності сегмента;
- **Коефіцієнт зміни завантаження каналу (τ):** визначає динаміку зміни QoS — наростання затору або його розсіювання.

Формули розрахунку параметрів простору ознак (визначені у п. 1.5.2) зведені в табл. 2.1.

Таблиця 2.1 - QoS-індикатори стану сегмента IoV-мережі

Параметр	Формула	Діапазон	QoS-інтерпретація
Коефіцієнт пропускну здатності η	v/v_{limit}	$[0; 1,5]$	Якість V2X-каналу: 1 — вільний, 0 — перевантажений
Коефіцієнт завантаження L	n_{cars}/C	$[0; 1]$	Насиченість каналу IoT-джерелами
Гradient τ	$(v_t - v_{t-1})/v_{limit}$	$[-1; +1]$	Динаміка QoS: < 0 — погіршення, > 0 — відновлення

Примітка. Верхня межа $\eta > 1$ є реалістичною, оскільки на практиці транспортні засоби можуть рухатися зі швидкістю, що перевищує встановлений ліміт v_{limit} ділянки; діапазон розширено до 1,5 для коректного охоплення всіх спостережуваних режимів.

Важливою процедурою є нормалізація та групування даних у пакети (батчі), що дозволяє нівелювати різницю у фізичних розмірностях параметрів та забезпечити статистичну значущість вибірки для подальшого аналізу.

На рис. 2.3 наведено приклад розподілу точок у тривимірному просторі ознак (η, L, τ) , де кольором виділено автоматично ідентифіковані режими руху. Видно, що режими формують компактні, добре сепаровані групи, причому кожна група має характерну еліпсоїдальну форму, зумовлену кореляцією між компонентами вектора ознак.

Блок 2: Оптимізація кількості кластерів. Оскільки режим інформаційного навантаження є динамічним, кількість характерних режимів руху (наприклад, «вільний потік», «синхронізований потік», «затор») не є константою і може змінюватися залежно від часу доби чи погодних умов. Для автоматичного визначення поточної структури даних використовується метод ліктя [70, 71, 72, 98, 99]. Як показано на схемі, алгоритм аналізує криву внутрішньокластерної дисперсії $J(k)$ і знаходить точку максимального перегину (оптимальне k). Завдяки цьому

Розподіл точок фазового простору по кластерах

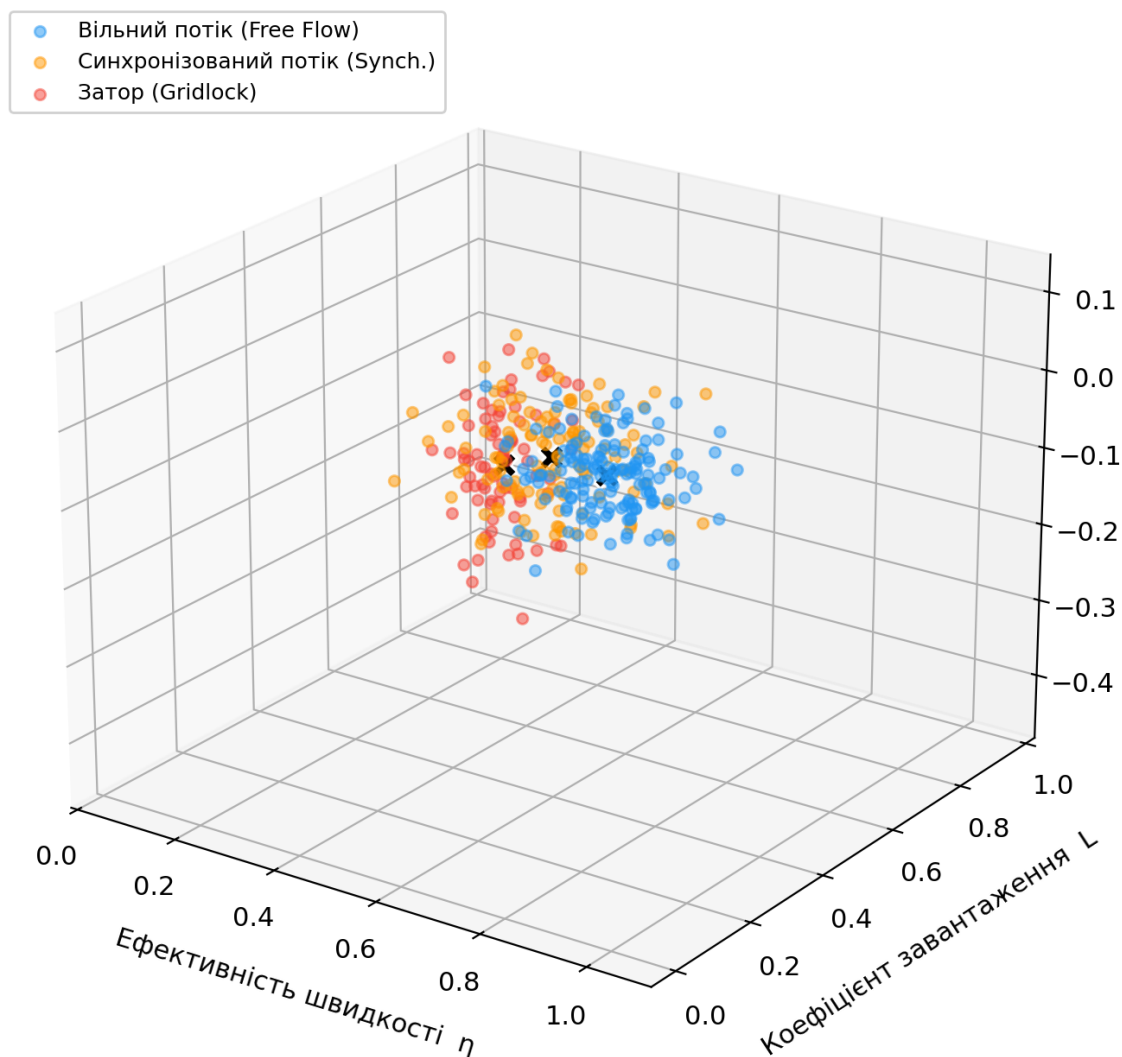


Рис. 2.3 - Розподіл точок простору ознак по кластерах: вільний потік (синій), синхронізований потік (помаранчевий), затор (червоний); чорні хрестики — центроїди

система адаптивно визначає, скільки саме правил потрібно згенерувати для поточного пакету даних, уникаючи як надмірного узагальнення, так і перенавчання (створення зайвих правил для шуму).

Блок 3: Ініціалізація та кластеризація K-Means++. На цьому етапі відбувається безпосереднє розбиття простору ознак на області, що відповідають виявленим режимам руху. Для забезпечення стабільності результатів та уникнення потрапляння у локальні мінімуми використовується покращений алгоритм ініціалізації **K-Means++** [38, 57, 68, 69]. Він гарантує, що початкові центроїди (показані на схемі

червоними точками) будуть рівномірно розподілені у хмарі даних, що критично важливо для коректного виявлення меж між станами «Затор» та «Синхронізований потік». Результатом роботи блоку є набір мікро-кластерів з чітко визначеними параметрами центру та дисперсії.

Блок 4: Інтеграція та злиття режимів. Цей блок реалізує механізм **інкрементального навчання** [64, 65, 97], що є ключовою відмінністю методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі від класичних статичних підходів. Система порівнює новостворені мікро-кластери (μ_{micro}) з уже існуючими в базі знань глобальними патернами (μ_{global}). На схемі продемонстровано логіку прийняття рішень на основі граничної умови (BC):

- Якщо новий кластер має значне перекриття з існуючим (гілка "Yes"), відбувається їх злиття (Merging) — це дає змогу уточнювати параметри існуючих правил (наприклад, розширювати діапазон поняття «нормальна швидкість» для конкретної ділянки дороги).
- Якщо перекриття відсутнє або мінімальне (гілка "No"), система ідентифікує це як новий, раніше невідомий режим роботи і ініціює створення нового правила. Такий підхід вирішує проблему «катастрофічного забування» [90], дозволяючи системі адаптуватися до змін у дорожній мережі без втрати раніше набутих знань.

Блок 5: Проекція та генерація правил. Фінальний етап забезпечує перехід від математичної абстракції до зрозумілої логіки керування. Виявлені багатовимірні кластери, що мають форму гіпер-еліпсоїдів, проектуються на осі окремих лінгвістичних змінних (η , L , τ). Як видно з графічної інтерпретації блоку, проекція перетворює статистичний розподіл даних у функції належності гаусового типу (μ , Σ). Це дозволяє автоматично сформулювати антецеденти (умови) та консеквенти (висновки) нечітких правил [60, 66, 96] виду: *«ЯКЩО Навантаження високе (відповідає проекції μ_L) І Коефіцієнт зміни завантаження каналу негативний, ТО Ефективність низька»*.

Узагальнений алгоритм роботи методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі наведено нижче.

Алгоритм. Адаптивна оцінка стану інформаційних потоків на сегментах IoV-мережі.

Вхід: потік IoT-повідомлень $x_k = [\eta, L, \tau]^T$ для ребра (u, v) .

Вихід: оновлена вага ребра $w_{uv}(t)$.

1. Накопичити пакет B_t з $N_{batch} = 400$ векторів.
2. Визначити оптимальну кількість кластерів k_{opt} за геометричним методом ліктя (п. 2.1.1).
3. Ініціалізувати центроїди за K-Means++ та виконати кластеризацію пакету B_t на k_{opt} мікро-кластерів $\{MC_j = \langle \mu_j, \Sigma_j, w_j \rangle\}$.
4. **Для кожного** мікро-кластера MC_j :
 - 4.1. Знайти найближчий глобальний кластер (формула 2.2):

$$k_{best} = \arg \min_m D_{Mahal}^2(\mu_j, \mu_m).$$
 - 4.2. Обчислити коефіцієнт Бхаттачар'ї BC (формула 2.3а).
 - 4.3. **Якщо** $BC > T_{merge}$: Байєсівське злиття — оновити μ_{global} , σ_{global} (п. 2.1.2).
Інакше: створити новий глобальний кластер.
 - 4.4. Обчислити $\alpha_j = \max(2(1 - BC), \alpha_{min})$.
5. **Для кожного** вхідного повідомлення x_k (потоківна обробка):
 - 5.1. Обчислити ступінь належності $\mu_m(x_k)$ до кожного з M глобальних кластерів (п. 2.2.1).
 - 5.2. Визначити $\alpha(x_k) = \sum_m \mu_m \alpha_m / \sum_m \mu_m$ (дефазифікація Такагі-Сугено).
 - 5.3. Оновити згладжену оцінку швидкості на ребрі: $\hat{v}_{uv}(t) = \alpha \cdot v_{raw} + (1 - \alpha) \cdot \hat{v}_{uv}(t - 1)$, а далі обчислити вагу $w_{uv}(t) = L_{uv} / \hat{v}_{uv}(t)$.

Нижче наведено детальний опис етапів реалізації методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі.

2.2.1 Пакетно-інкрементальна кластеризація та геометрична оптимізація кількості станів

Вхідними даними для методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі є потік векторів простору ознак $x_k \in R^3$, структура яких визначена у п. 1.5.2. Оскільки потік є потенційно нескінченним, збереження повної історії спостережень у пам'яті є неможливим [7, 64, 65]. Для вирішення цієї проблеми застосовано стратегію **пакетно-інкрементальної кластеризації** [64, 65].

Процес обробки дискретизується на послідовність пакетів $B_1, B_2, \dots, B_t$. Кожен пакет B_t містить фіксовану кількість векторів N_{batch} .

Вибір розміру пакету базується на статистичній оцінці похибки вимірювання. Згідно з центральною граничною теоремою, довірчий інтервал для оцінки середнього визначається як $\bar{x} \pm z_\gamma \cdot \frac{\sigma}{\sqrt{N_{batch}}}$, де z_γ — квантиль нормального розподілу для рівня довіри γ . Для забезпечення 95%-го рівня довіри ($z_{0,975} = 1,96$) при типовому коефіцієнті варіації транспортних даних $CV \approx 0,5$ та допустимій відносній похибці $\varepsilon = 5\%$, мінімально необхідний розмір вибірки складає: $N_{batch} \geq \left(\frac{z_{0,975} \cdot CV}{\varepsilon} \right)^2 = \left(\frac{1,96 \times 0,5}{0,05} \right)^2 \approx 384$. Для зручності обрано $N_{batch} = 400$. При $N = 400$ статистичний шум нівелюється, дозволяючи коректно оцінити коваріаційну матрицю Σ у 3-вимірному просторі, уникаючи ситуації, коли кластер стає статистично незначущим (схлопується до точки). З точки зору швидкодії, при вхідному потоці ≈ 800 подій/с, накопичення такого пакету займає менше 1 с, що задовольняє вимогам реального часу.

Спираючись на результати порівняльного аналізу, проведеного у підрозділі 1.4.2, для реалізації підсистеми автоматичного виявлення шаблонів обрано **алгоритм K-Means** [57, 68, 69]. Його вибір як базового методу обумовлений лінійною обчислювальною складністю $O(N)$, необхідною для обробки потоку, та здатністю формувати чіткі числові характеристики центроїдів (середнє значення μ , стандартне відхилення σ), що є необхідною умовою для подальшої побудови функцій належності (фазифікації) [95, 96]. На відміну від DBSCAN, який не гарантує параметрів μ/σ для побудови функцій належності, та GMM, що

потребує $O(N \cdot k \cdot d^2)$ для оцінки повних коваріаційних матриць, K-Means забезпечує оптимальний баланс між інформативністю та швидкодією.

У межах кожного пакету даних B_t задача кластеризації формалізується як оптимізаційна задача мінімізації сумарного квадратичного відхилення точок від центрів відповідних кластерів. Цільова функція має вигляд:

$$J(C) = \sum_{j=1}^k \sum_{x_i \in C_j} \|x_i - \mu_j\|^2 \rightarrow \min \quad (2.1)$$

де:

- k кількість кластерів;
- C_j множина точок, віднесених до j -го кластера;
- μ_j центроїд j -го кластера, що розраховується як середнє арифметичне координат точок.

Мінімізація цього функціоналу дозволяє досягти максимальної компактності груп, де точки всередині кластера є максимально схожими, а самі кластери — максимально відмінними один від одного.

Попри ефективність, класична реалізація K-Means [57, 71] має два суттєві недоліки при застосуванні до потокових даних:

- Висока чутливість до початкового розміщення центроїдів (проблема локальних мінімумів).
- Необхідність апріорного задання кількості кластерів k , яка у транспортному потоці є динамічною величиною.

Для адаптації алгоритму K-Means до умов IoV-системи у роботі застосовано дві процедури оптимізації: ініціалізацію **K-Means++** та динамічний пошук k за **геометричним методом ліктя**.

Геометричний метод ліктя

Критичним недоліком класичного алгоритму K-Means є необхідність апріорного задання кількості кластерів k . У транспортному потоці ця величина є динамічною: вночі потік є однорідним ($k=1$), тоді як у

години пік він може розшаровуватися на кілька режимів (вільний потік, синхронізований потік, затор). Жорстке задання k призводить до помилок моделювання: при заниженому k відбувається змішування різних режимів, При завищеному k відбувається штучне дроблення єдиного режиму. Для автоматичного визначення оптимального k_{opt} у кожному пакеті даних у літературі використовують внутрішні критерії якості кластеризації — метод ліктя, індекси Caliński–Harabasz та Davies–Bouldin, а також silhouette-аналіз [70, 71, 72, 98, 99]. У роботі застосовується аналіз кривої зміни цільової функції помилки.

Побудова графіка внутрішньокластерної суми квадратів (WCSS). Першим кроком є розрахунок суми квадратів внутрішньокластерних відстаней для діапазону можливих значень k (наприклад, від 1 до 10): $WCSS(k) = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} \|\mathbf{x} - \mu_j\|^2$.

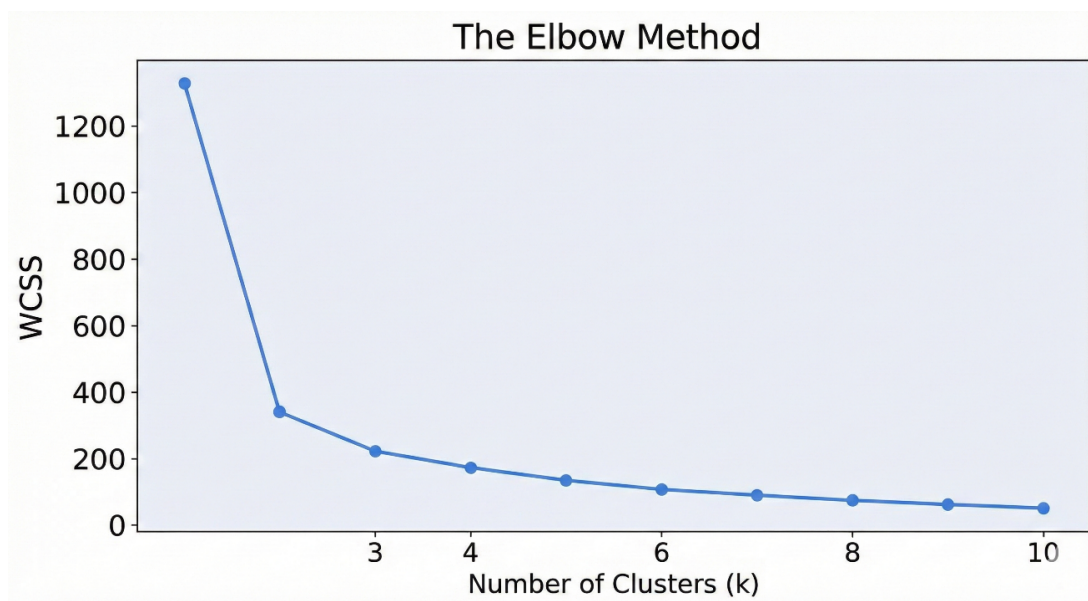


Рис. 2.4 - Графік залежності WCSS від кількості кластерів

Графік цієї залежності (рис. 2.4) є монотонно спадним. Точка, де відбувається різка зміна нахилу графа («лікоть»), вважається оптимальною. Вона відповідає стану, коли додавання нового кластера вже не дає значного приросту в якості розділення даних.

Оскільки візуальний пошук точки перегину неможливий в автоматизованій системі, застосовано **геометричний метод ліктя**. Суть методу полягає у розгляді графіка $WCSS(k)$ як геометричної кривої на площині. Алгоритм складається з наступних кроків:

- Значення k та $WCSS(k)$ нормалізуються у діапазон $[1]$ для коректного обчислення геометричних відстаней.
- Проводиться пряма лінія (вектор), що з'єднує першу $(1, WCSS(1))$ та останню $(K_{\max}, WCSS(K_{\max}))$ точки графа (на рис. 2.5 позначена зеленою пунктирною лінією).
- Для кожного проміжного k обчислюється перпендикулярна відстань $d(k)$ від точки на кривій до побудованої прямої (червона лінія на рис. 2.5). Математично критерій вибору оптимального кластера записується як: $k_{\text{opt}} = \arg \max_k (d(k))$. Саме ця точка відповідає максимальній кривизні графіка, що дозволяє алгоритмічно, без участі експерта, визначити момент, коли подальше ускладнення моделі стає недоцільним.

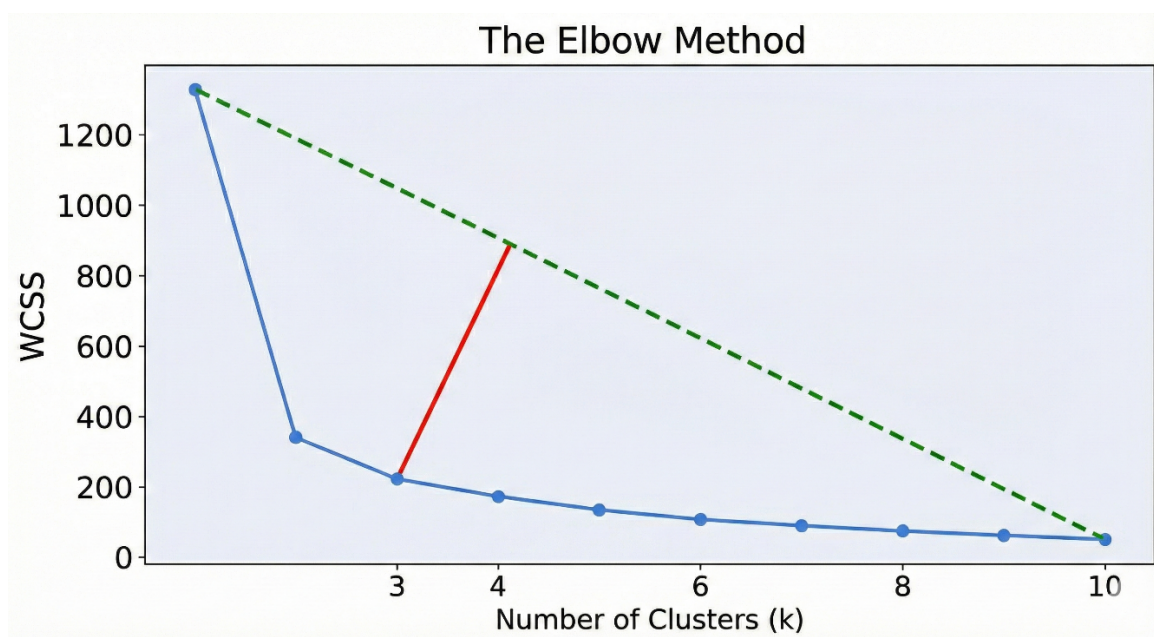


Рис. 2.5 - Геометричний метод пошуку «ліктя»

Проблема початкової ініціалізації центрів кластерів

Стандартна реалізація алгоритму K-Means використовує випадковий вибір початкових центроїдів. Однак цей підхід має суттєвий недолік: існує висока ймовірність того, що початкові центри будуть обрані у безпосередній близькості один від одного. Для задачі аналізу транспортних потоків це створює два ризики:

- алгоритм може некоректно розділити простір ознак (наприклад, розбити один режим «вільного потоку» на два підкластери, проігнорувавши режим «затору»).
- близьке розташування центрів вимагає значно більшої кількості ітерацій для їх «розведення» у правильні позиції, що збільшує час обробки пакету.

Для вирішення цієї проблеми у роботі застосовано техніку ініціалізації K-Means++ [38, 57, 68, 69]. Її основна ідея полягає у послідовному виборі центрів таким чином, щоб максимізувати відстань між ними.

Алгоритм ініціалізації реалізується за 4 кроки:

1. Перший центроїд μ_1 обирається випадковим чином з наявних точок пакету даних B_t (рівномірний розподіл).
2. Для кожної точки $\mathbf{x}_i$ у пакеті обчислюється найкоротша відстань $D(\mathbf{x}_i)$ до найближчого з вже обраних центроїдів: $D(\mathbf{x}_i) = \min_{j \in \{1 \dots m\}} \|\mathbf{x}_i - \mu_j\|$
3. Наступний центроїд μ_{m+1} обирається зі множини точок з імовірністю $P(\mathbf{x}_i)$, яка є пропорційною квадрату відстані до існуючих центрів: $P(x_i) = \frac{D(x_i)^2}{\sum_{x_j \in B_t} D(x_j)^2}$. Це ключовий етап: точки, що знаходяться найдалі від поточних кластерів (наприклад, аномальні стани трафіку або протилежні режими руху), мають найвищу ймовірність стати новими центрами.
4. Кроки 2 і 3 повторюються доти, доки не буде обрано k центроїдів.

На рис. 2.6 показано роботу алгоритму. Застосування K-Means++ дозволяє гарантувати, що початкові гіпотези про режими руху будуть добре сепаровані у просторі ознак (охоплюючи весь діапазон від $\eta \approx 0$ до $\eta \approx 1$). Експериментально доведено, що це прискорює збіжність алгоритму в середньому у 2–3 рази [38, 73] порівняно з випадковою ініціалізацією — відповідно до вимог реального часу.

Застосування комбінації геометричного методу ліктя (для визначення k) та ініціалізації K-Means++ дозволяє повністю автоматизувати

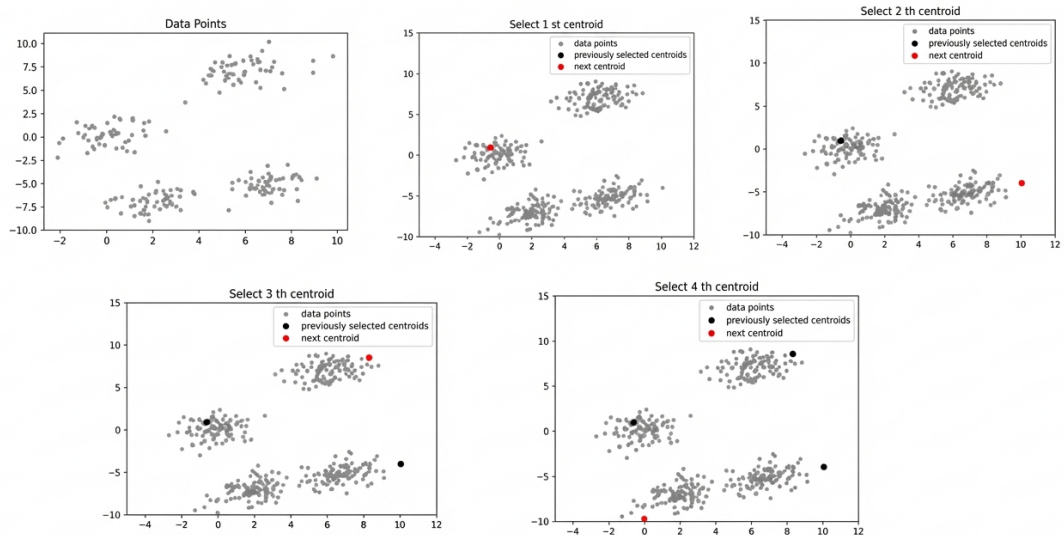


Рис. 2.6 - Візуалізація роботи алгоритму K-Means++

процес первинної сегментації потоку даних, забезпечуючи формування «кандидатів» у режими руху, які добре сепаровані у просторі ознак та охоплюють весь діапазон станів мережі (від $\eta \approx 0$ до $\eta \approx 1$). Отримані мікро-кластери передаються на наступний етап обробки — імовірнісну валідацію та злиття.

По завершенню ітерацій алгоритму K-Means для поточного пакету даних B_t , отримані групи точок трансформуються у компактні структури — мікро-кластери. Вони виступають у ролі короткострокових «прото-режимів», узагальнюючи локальну структуру трафіку. Кожен сформований мікро-кластер MC_j формалізується як кортеж статистичних параметрів: $MC_j = \langle \mu_j, \sigma_j, w_j \rangle$, де $\mu_j \in \mathbb{R}^3$ — центроїд кластера (вектор середніх значень ознак $[\bar{\eta}, \bar{L}, \bar{\tau}]$), що визначає положення режиму у просторі ознак. $\sigma_j \in \mathbb{R}^3$ — вектор стандартних відхилень вздовж кожного виміру. Він описує форму (еліпсоїд розсіювання) кластера і запобігає виродженню метрики Махаланобіса на наступних етапах. w_j — (кількість точок, віднесених до нього), що характеризує статистичну значущість даного режиму.

Для підвищення стійкості системи до шуму застосовується правило фільтрації: мікро-кластери з недостатньою статистичною підтримкою ($w_j < w_{\min}$) відкидаються як випадковий шум або аномальні викиди, де $w_{\min} = \lfloor 0,05 \cdot N_{batch} \rfloor = 20$ — мінімальний обсяг вибірки, необхідний

для надійної оцінки μ_j та σ_j у тривимірному просторі. Це узгоджується зі стратегіями обслуговування поточних даних, дозволяючи стиснути тисячі "сирих" вимірів телеметрії до кількох статистично значущих об'єктів [51, 67].

Отриманий набір валідних мікро-кластерів [64, 65, 77] $\{MC_j\}$ передається на наступний етап — процедуру глобальної інтеграції та злиття.

2.2.2 Керування життєвим циклом патернів на основі статистичних метрик

Сформовані на етапі мікро-кластеризації об'єкти є короткостроковими сутностями, що описують локальну структуру поточного пакету даних. Для формування стабільної бази знань їх необхідно інтегрувати у довгострокові **глобальні режими**, що зберігаються в пам'яті системи між ітераціями [66, 67].

Процес інтеграції реалізується як послідовна стратегія, що складається з трьох етапів: ідентифікації кандидата, перевірки гіпотези про злиття та Байєсівського оновлення параметрів.

2.2.2.1 Ідентифікація режиму-кандидата на основі метрики Махаланобіса. Першим кроком алгоритму інтеграції є визначення того, до якого з існуючих історичних режимів найбільш подібний новий мікро-кластер. Цей етап реалізує стратегію **пошуку найближчого сусіда** у статистичному просторі.

У просторі ознак станів транспортного потоку змінні є сильно корельованими. Наприклад, зростання коефіцієнта завантаження (L) неминуче призводить до зниження коефіцієнта пропускної здатності сегмента (η). Через наявність цього фізичного зв'язку кластери набувають форми не сфер, а витягнутих еліпсоїдів, орієнтованих під кутом до осей координат.

Використання класичної Евклідової відстані в таких умовах є некоректним, оскільки вона ігнорує форму розподілу даних. Точка може знаходитися геометрично далеко від центру, але статистично (вздовж головної осі еліпсоїда) належати до цього режиму.

Для вирішення цієї проблеми застосовано **відстань Махаланобіса**

з об'єднаною дисперсією [58]. Ця метрика дозволяє оцінити відстань між центрами розподілів у одиницях стандартного відхилення, автоматично адаптуючись до форми кластера.

Розрахунок виконується за наступною формулою:

$$D^2(\mu_{\text{micro}}, \mu_{\text{global}}) = \sum_{i=1}^{\text{dim}} \frac{(\mu_{\text{micro},i} - \mu_{\text{global},i})^2}{\sigma_{\text{global},i}^2 + \sigma_{\text{micro},i}^2} \quad (2.2)$$

де:

- $\mu_{\text{micro},i} - \mu_{\text{global},i}$ - координати центрів мікро-кластера та глобального режиму по i -му виміру;
- $\sigma_{\text{global},i}^2 + \sigma_{\text{micro},i}^2$ - об'єднана дисперсія. Вона виконує роль адаптивного знаменника: якщо кластери мають велику невизначеність (розкид), знаменник зростає, роблячи метрику більш толерантною до відхилень.

Суттєвою особливістю етапу інтеграції є те, що він не перевіряє кластери у випадковому порядку. Для кожного нового мікро-кластера виконується розрахунок вектора відстаней до всіх існуючих глобальних режимів: $\mathbf{D} = [D_1^2, D_2^2, \dots, D_M^2]$

Після цього обирається режим-кандидат з індексом k_{best} , що забезпечує глобальний мінімум відстані: $k_{\text{best}} = \arg \min_{j \in \{1..M\}} (D_j^2)$

На рис. 2.7 показано геометричну інтерпретацію відстані Махаланобіса у проекції простору ознак (η, L) . Сині концентричні еліпси відображають зони впевненості глобального режиму (1σ , 2σ , 3σ), червоний еліпс — мікро-кластер поточного пакету. Зелена стрілка демонструє розраховану відстань Махаланобіса, сіра пунктирна — Евклідову відстань для порівняння. Видно, що хоча Евклідова відстань між центрами є значною, відстань Махаланобіса враховує витягнутість еліпсоїда та об'єднану дисперсію, дозволяючи коректно ідентифікувати приналежність точки до даного режиму.

Отриманий індекс k_{best} визначає єдиного кандидата, який передається на наступний етап — валідацію злиття.

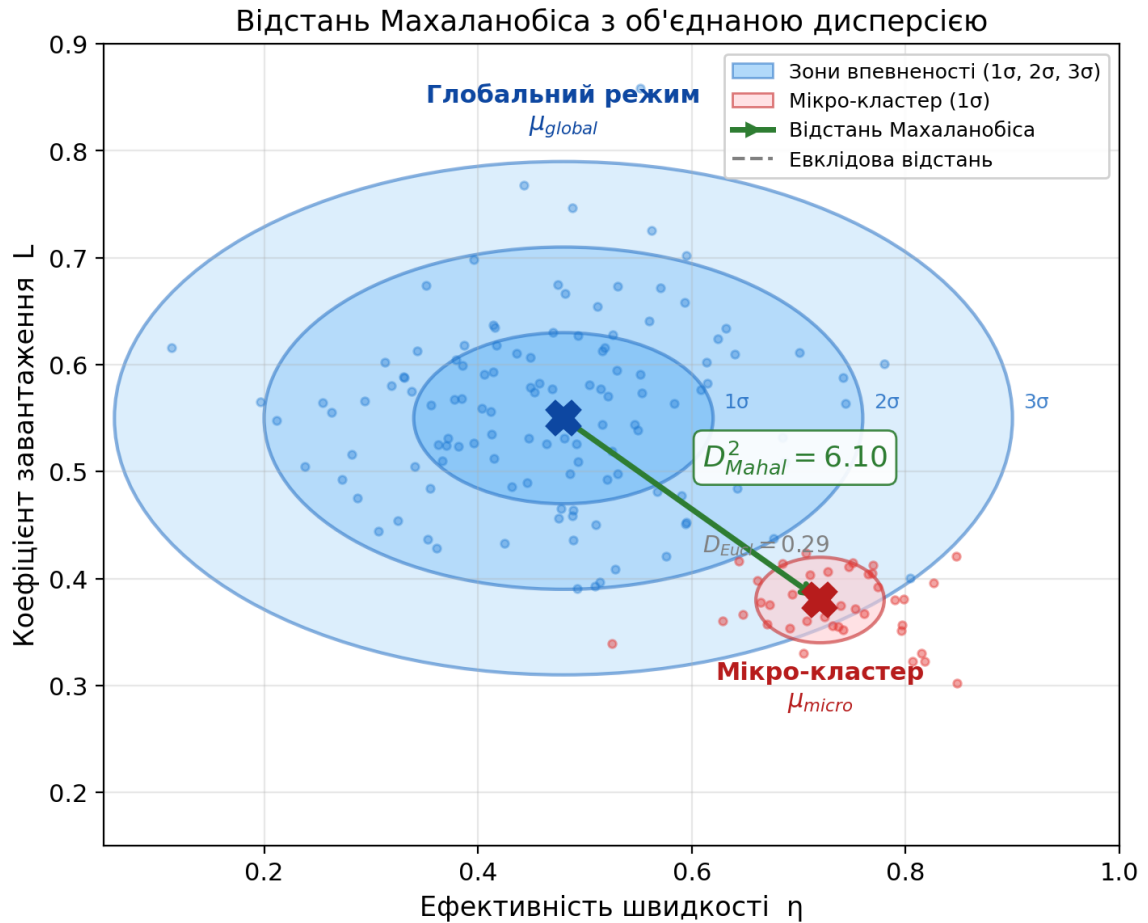


Рис. 2.7 - Приклад знаходження відстані Махаланобіса з об'єднаною дисперсією для двох кластерів

2.2.2.2 Валідація рішення про злиття на основі подібності Бхаттачар'ї. Геометрична близькість центрів кластерів, визначена на попередньому етапі, є необхідною, але недостатньою умовою для їх об'єднання. Щоб уникнути помилкового злиття різнорідних режимів (наприклад, стабільного потоку та хаотичного шуму), необхідно порівняти не тільки центри, але й форми розподілів.

Для цього використовується **коефіцієнт Бхаттачар'ї (BC)** [59, 73], який оцінює площу перекриття двох імовірнісних розподілів.

Оскільки компоненти вектора ознак (коефіцієнт пропускну здатності сегмента, коефіцієнт завантаження каналу, коефіцієнт зміни завантаження каналу) обробляються як незалежні змінні, розрахунок коефіцієнта виконується шляхом добутку подібностей по кожному виміру. Формула має вигляд добутку двох факторів:

$$BC(p, q) = \underbrace{\sqrt{\frac{2\sigma_p\sigma_q}{\sigma_p^2 + \sigma_q^2}}}_{\text{Фактор форми}} \cdot \underbrace{\exp\left(-\frac{(\mu_p - \mu_q)^2}{4(\sigma_p^2 + \sigma_q^2)}\right)}_{\text{Фактор відстані}} \quad (2.3)$$

де:

- μ_p, μ_q - центри мікро-кластера та глобального режиму
- σ_p, σ_q - їхні стандартні відхилення (ширина розкиду даних)

Фізичний зміст компонентів:

- *Фактор форми*: Цей доданок відповідає за порівняння дисперсій. Він стає максимальним, тільки якщо $\sigma_p = \sigma_q$. Якщо один кластер вузький, а інший широкий, цей коефіцієнт зменшується, запобігаючи злиттю стабільних даних з шумом.
- *Фактор відстані*: Експоненційний член, який швидко спадає до нуля, якщо відстань між центрами $(\mu_p - \mu_q)$ зростає відносно їхньої спільної ширини.

Слід зазначити, що метрика Махаланобіса (2.2) автоматично адаптується до масштабу кожної ознаки через нормалізацію об'єднаною дисперсією: якщо деяка компонента (наприклад, градієнт τ) має малу природну варіабельність, знаменник $\sigma_{p,i}^2 + \sigma_{q,i}^2$ буде малим, що автоматично підсилить чутливість метрики до відмінностей саме по цій осі. Відтак, ручне зважування ознак є надлишковим, а метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі залишається повністю автоматичним — без додаткових гіперпараметрів.

Повний коефіцієнт Бхаттачар'ї для тривимірного простору ознак обчислюється як добуток подібностей по кожному виміру $j \in \{1, 2, 3\}$:

$$BC = \prod_{j=1}^3 BC_j, \quad BC_j = \sqrt{\frac{2\sigma_{p,j}\sigma_{q,j}}{\sigma_{p,j}^2 + \sigma_{q,j}^2}} \cdot \exp\left(-\frac{(\mu_{p,j} - \mu_{q,j})^2}{4(\sigma_{p,j}^2 + \sigma_{q,j}^2)}\right) \quad (2.3a)$$

На рис. 2.8 наведено тристадійну візуалізацію процесу валідації та злиття. Панель (а) демонструє обчислення коефіцієнта Бхаттачар'ї: синя

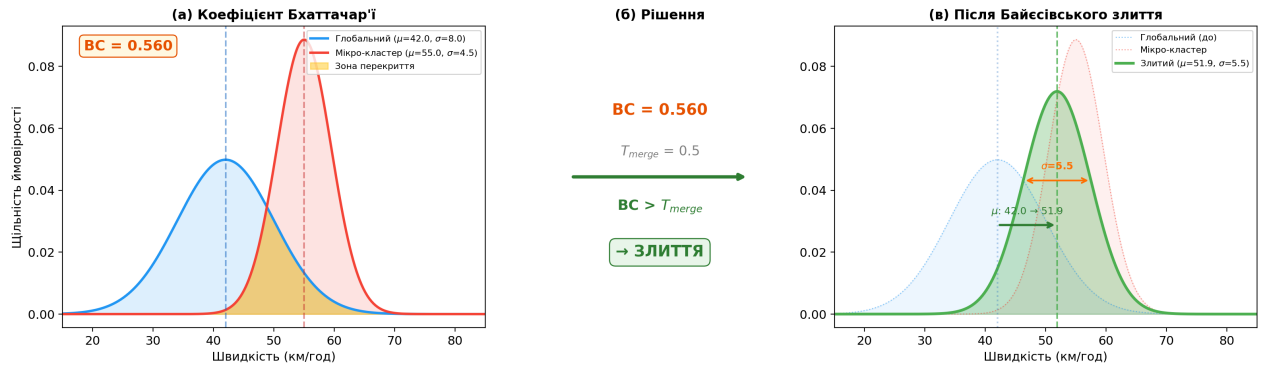


Рис. 2.8 - Повний цикл валідації та злиття кластерів: (а) обчислення коефіцієнта Бхаттачар'ї, (б) прийняття рішення про злиття, (в) результат Байєсівського оновлення параметрів

крива — щільність розподілу глобального кластера (історичний шаблон), червона — мікро-кластер з поточного пакету, заштрихована жовта область — зона перекриття, площа якої чисельно дорівнює значенню BC . Панель (б) показує логіку прийняття рішення: якщо $BC > T_{merge}$, виконується злиття. Панель (в) відображає результат Байєсівського оновлення — зелений розподіл зі зміщеним центром (μ_{new}) та звуженою дисперсією (σ_{new}). Коефіцієнт BC виступає індикатором статистичної спорідненості режимів:

- *Велике перекриття* ($BC \rightarrow 1$): Свідчить про те, що новий пакет даних описує той самий режим руху, що й історичний шаблон. Це є підставою для злиття.
- *Мале перекриття* ($BC \rightarrow 0$): Вказує на суттєву відмінність у параметрах (середньому значенні або дисперсії), що інтерпретується системою як виникнення нового, унікального режиму руху.

Поріг коефіцієнта Бхаттачар'ї ($T_{merge} = 0,5$) обрано на основі критерію **розрізнюваності сигналів** (критерій Релея).

У спектральному аналізі два піки вважаються нерозрізненими, коли відстань між їхніми центрами стає меншою за повну ширину на рівні половини висоти (FWHM). Для нормального розподілу ця ширина є константою: $FWHM = 2\sqrt{2\ln 2} \sigma \approx 2,35 \sigma$.

Покажемо, що $BC = 0,5$ відповідає саме цій граничній відстані. Для

двох однакових Гауссіанів ($\sigma_p = \sigma_q = \sigma$) формула (2.3) спрощується до:

$$BC = \exp\left(-\frac{(\mu_p - \mu_q)^2}{8\sigma^2}\right).$$

Підставляючи $BC = 0,5$ та розв'язуючи відносно відстані:

$$0,5 = \exp\left(-\frac{d^2}{8\sigma^2}\right) \Rightarrow d = \sqrt{8 \ln 2} \sigma = 2\sqrt{2 \ln 2} \sigma = FWHM.$$

Отже, поріг $T_{merge} = 0,5$ має чітку геометричну інтерпретацію: система об'єднує кластери лише тоді, коли відстань між їхніми центрами не перевищує $FWHM$, тобто коли розглядати їх як окремі режими руху стає статистично недоцільно. При $BC < 0,5$ розподіли сепаровані достатньо, щоб відповідати різним режимам.

Однак, у разі позитивного рішення про злиття, просте арифметичне усереднення параметрів старого та нового кластерів є некоректним, оскільки воно ігнорує різну статистичну вагу (точність) вимірювань. Для математично обґрунтованого оновлення параметрів μ та σ застосовується процедура **Байєсівського злиття**, описана у наступному пункті.

2.2.2.3 Байєсівське оновлення параметрів кластерів та розрахунок коефіцієнта адаптації. У разі прийняття позитивного рішення про об'єднання ($BC > 0.5$), виникає задача оновлення параметрів глобального кластера ($\mu_{macro}, \sigma_{macro}$) з урахуванням нових даних ($\mu_{micro}, \sigma_{micro}$). Просте арифметичне усереднення в даному випадку є некоректним, оскільки воно ігнорує різну статистичну значущість (надійність) джерел інформації: історичний режим, сформований на тисячах вимірів, має значно вищу вагу, ніж миттєвий пакет даних.

Для вирішення цієї задачі застосовано метод зважування, оберненого до дисперсії. Цей підхід базується на принципах байєсівського виведення, де дисперсія розглядається як міра невизначеності: чим менша дисперсія, тим вища точність і, відповідно, вага внеску.

Нове положення центру кластера визначається як зважена сума координат, де вагами виступають обернені дисперсії (або, для спрощення

запису, перехресні дисперсії):

$$\mu_{\text{new}} = \frac{\mu_{\text{macro}} \sigma_{\text{micro}}^2 + \mu_{\text{micro}} \sigma_{\text{macro}}^2}{\sigma_{\text{macro}}^2 + \sigma_{\text{micro}}^2} \quad (2.4)$$

Якщо новий мікро-кластер є дуже «гострим» (мала σ_{micro} , висока точність), він перетягне центр μ_{macro} сильніше в свій бік, ніж якщо він є розмитим (шумним), завдяки чому система швидко адаптуватися до чітких змін трафіку, ігноруючи при цьому ненадійні дані.

Нова ширина розподілу розраховується за **модифікованою формулою** злитого кластера, що є ключовою відмінністю методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі:

$$\sigma_{\text{new}} = \sqrt{\frac{2 \sigma_{\text{macro}}^2 \sigma_{\text{micro}}^2}{\sigma_{\text{macro}}^2 + \sigma_{\text{micro}}^2}} \quad (2.5)$$

Множник $\sqrt{2}$ коригує гармонійне середнє дисперсій і є **єдиним значенням**, яке забезпечує стабільність масштабу.

Доведення. Класичне Байєсівське оновлення (добуток двох гаусіанів) дає дисперсію злитого розподілу:

$$\sigma_{\text{classic}}^2 = \frac{\sigma_{\text{macro}}^2 \sigma_{\text{micro}}^2}{\sigma_{\text{macro}}^2 + \sigma_{\text{micro}}^2}.$$

Введемо коригуючий множник k : $\sigma_{\text{new}}^2 = k \cdot \sigma_{\text{classic}}^2$. Поставимо вимогу **збереження масштабу**: якщо два кластери мають однаковий розкид ($\sigma_{\text{macro}} = \sigma_{\text{micro}} = \sigma$), то злиття не повинно змінювати ширину: $\sigma_{\text{new}} = \sigma$. Підставляємо:

$$\sigma^2 = k \cdot \frac{\sigma^2 \cdot \sigma^2}{\sigma^2 + \sigma^2} = k \cdot \frac{\sigma^2}{2} \Rightarrow k = 2.$$

Отже $\sigma_{\text{new}} = \sqrt{2 \sigma_{\text{classic}}^2}$, а множник під коренем дорівнює саме $\sqrt{2}$. При $k < 2$ кластер звужується з кожною ітерацією (схлопується до точки), при $k > 2$ — необмежено розширюється.

Тобто при $\sigma_{\text{macro}} = \sigma_{\text{micro}} = \sigma$ формула (2.5) дає $\sigma_{\text{new}} = \sigma$ (збереження масштабу), тоді як класичний добуток гаусіанів звужує розподіл до $\sigma/\sqrt{2}$. Результуюча дисперсія зменшується пропорційно

зростанню впевненості системи. Коли два незалежні джерела (історія та нові дані) підтверджують однакові параметри режиму, невизначеність звужується, що робить правило більш селективним.

Геометричну інтерпретацію результату злиття показано на панелі (в) рис. 2.8: центр зміщується у бік більш точного (вужчого) джерела, а результуюча дисперсія звужується порівняно з обома вихідними розподілами.

Застосування байєсівського злиття забезпечує системі дві критично важливі для IoV властивості:

- *Стабільність*: Система стійка до короточасних коливань. Поодинокі викиди з великою дисперсією (шум GPS [11, 54]) мають мінімальну вагу і майже не зміщують центри глобальних режимів.
- *Пластичність*: Система зберігає чутливість до реальних структурних змін. Якщо потік даних стабільно зміщується і формує щільні кластери в новій точці простору ознак, глобальний шаблон плавно «дрейфує» до нового стану.

Зауважимо, що глобальні кластери не потребують явного механізму видалення. Якщо певний режим руху зникає з потоку даних, жоден вхідний вектор не потрапляє у його зону чутливості, ступінь належності $\mu_k(\mathbf{x}) \rightarrow 0$, і кластер фактично виключається з обчислення α без втручання оператора. Кількість глобальних режимів обмежена параметром $K_{\max} = 10$, тому неактивні кластери не створюють значного обчислювального навантаження.

Крім фізичних параметрів, кожному кластеру присвоюється динамічний коефіцієнт адаптації α_j , який виступає вихідним значенням (консеквентом) для відповідного нечіткого правила. У розробленій системі α_j розраховується на основі метрики нестабільності сигналу, яка є оберненою до подібності Бхаттачар'ї.

Логіка розрахунку полягає в наступному: чим менше поточні дані схожі на ідеальний центр кластера (але все ще потрапляють у нього), тим вищою має бути швидкість адаптації, щоб відслідкувати потенційну зміну градієнта швидкості. Математично це реалізується через лінійне

нормування значення BC у діапазон $[1]$. Оскільки формула (2.6) застосовується лише до кластерів, що пройшли злиття ($BC_j > T_{merge} = 0,5$), значення BC_j завжди знаходиться у діапазоні $(0,5; 1]$, а відповідний α_j — у діапазоні $[0; 1)$.

Для лінійного відображення довільної величини $x \in [a, b]$ у нормований діапазон $y \in [0, 1]$ використовується стандартна формула min-max нормалізації:

$$y = \frac{x - a}{b - a}.$$

У нашому випадку $x = BC$, $a = T_{merge}$, $b = 1$, проте зв'язок є оберненим: більше BC (вища подібність) — менше α (повільніша адаптація). Тому:

$$\alpha = 1 - \frac{BC - T_{merge}}{1 - T_{merge}} = \frac{1 - BC}{1 - T_{merge}}.$$

Це єдине лінійне відображення, що задовольняє граничні умови $\alpha(BC=1) = 0$ та $\alpha(BC=T_{merge}) = 1$ без введення додаткових гіперпараметрів. При $T_{merge} = 0,5$ вираз спрощується до:

$$\alpha_j = 2(1 - BC_j) \quad (2.6)$$

Фінальне значення α_j визначається з урахуванням мінімального технічного порогу пульсації ($\alpha_{min} = 0,01$):

$$\alpha_j = \max(2(1 - BC_j), \alpha_{min}) \quad (2.6a)$$

Для типових режимів руху коефіцієнт адаптації набуває характерних значень:

- *Стабільний потік* ($BC \approx 0,95$): $\alpha \approx 0,10$ — система фільтрує сенсорний шум, зберігаючи плавність ваг;
- *Перехідний стан* ($BC \approx 0,70$): $\alpha \approx 0,60$ — помірна реакція на поступову зміну режиму руху;
- *Інцидент / різка зміна* ($BC \approx 0,55$): $\alpha \approx 0,90$ — максимальна швидкість адаптації для миттєвої реакції на дорожню подію.

Якщо новий пакет ідеально збігається з історією, система ”заморожує” ваги, ігноруючи дрібний шум. В іншому випадку дані знаходяться на межі розпізнавання (наприклад, різке ущільнення потоку), система встановлює максимальний коефіцієнт оновлення, забезпечуючи миттєву реакцію на зміну стану інформаційного навантаження.

Отже, сформований набір оновлених глобальних кластерів являє собою актуальну статистичну модель поведінки трафіку на ребрі, яка готова до фінального етапу — генерації нечітких правил.

2.3 Нечіткий контролер адаптивного згладжування телеметрії

Розроблена у п. 2.1 процедура кластеризації забезпечує автоматичне формування бази знань — узагальненої статистичної моделі режимів руху. Однак для задачі маршрутизації необхідно ефективно обробляти вхідний потік у реальному часі. Як зазначено у вступі до розділу, задача оновлення ваги ребра зводиться до визначення коефіцієнта $\alpha = f(\mathbf{x}_k)$ у формулі ЕМА, де $f(\cdot)$ — функція стану потоку.

Як апарат реалізації $f(\cdot)$ обрано **нечітку логіку** [60, 61, 66, 109]: перехід між режимами є неперервним, а оцінка достовірності даних природно описується функціями належності. Нижче описано архітектуру нечіткого контролера, який трансформує параметри кластерів (μ, Σ) у лінгвістичні змінні та реалізує динамічне керування коефіцієнтом адаптації.

2.3.1 Структура нечіткого виведення типу Такагі-Сугено для згладжування телеметрії

Для реалізації механізму прийняття рішень у реальному часі обрано архітектуру нечіткого контролера типу Такагі-Сугено [60, 66] 0-го порядку. Вибір даної моделі, на відміну від класичних систем Мамдані [61], зумовлений двома факторами:

- *Обчислювальна ефективність*: Відсутність складного етапу дефазифікації (через інтегрування площ) дозволяє виконувати розрахунки за кілька мікросекунд, що є критичним для обробки високочастотного потоку Kafka [52, 100].

- *Точність керування:* Виходом системи є не абстрактний лінгвістичний терм, а конкретне числове значення коефіцієнта $\alpha \in [0, 1]$, що дозволяє плавно регулювати інерційність фільтра.

Архітектура контролера базується на принципі **прямого відображення** «кластер $\rightarrow$ правило» [45, 46, 102], де кожен статистичний кластер, виявлений на етапі навчання, трансформується у відповідне продукційне правило. Структурно алгоритм складається з трьох етапів.

1. Фазифікація даних

На цьому етапі вхідний вектор простору ознак $x_k = [\eta, L, \tau]^T$ перетворюється на набір значень, що характеризують ступінь його належності до кожного з відомих режимів руху.

Функції належності (*Membership Functions, MF*) формуються автоматично на основі параметрів глобальних кластерів. Оскільки кластери описані векторами середніх значень μ_j та дисперсій σ_j^2 , функція належності набуває вигляду багатовимірної Гауссіани:

$$\mu_k(\mathbf{x}) = \exp \left(-\frac{1}{2} \sum_{i=1}^3 \frac{(x_i - c_{k,i})^2}{\sigma_{k,i}^2} \right) \quad (2.7)$$

де: $\mu_k \in [0, 1]$ — ступінь належності (вага правила), $c_{k,i}$ та $\sigma_{k,i}$ — центр та стандартне відхилення k -го кластера по i -му виміру. Знаменник $\sigma_{k,i}^2$ визначає «зону чутливості» правила: вузькі кластери (мала σ) реагують лише на точні збіги, тоді як широкі кластери допускають значні варіації вхідних даних. Оскільки дисперсія $\sigma_{k,i}^2$ автоматично відображає природну варіабельність кожної ознаки, ручне зважування не потрібне — метрика сама підсилює ознаки з малим розкидом. Геометрично це означає, що кожен режим руху представлений у просторі ознак як еліпсоїд.

На рис. 2.9 ліворуч показано проекцію простору ознак (η, L) з трьома кластерами та точкою Р, яка знаходиться між режимами «вільний потік» (*Free Flow* на рисунку) та «синхронізований потік» (*Traffic*), але геометрично ближче до центру останнього. Праворуч наведено результат фазифікації — стовпчикову діаграму ступенів належності μ_k . Система м'якої логіки не відносить точку жорстко до одного класу, а розраховує

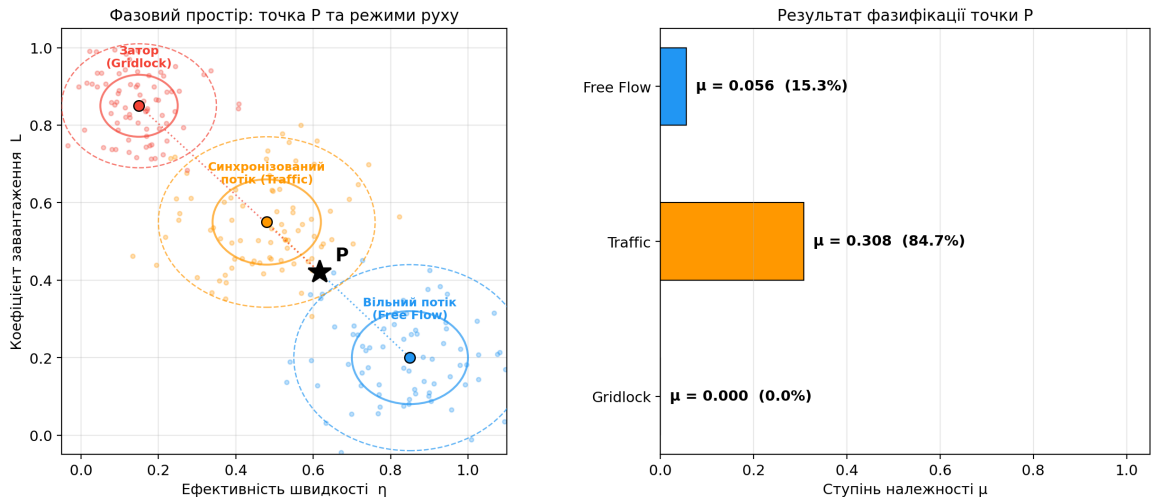


Рис. 2.9 - Візуалізація етапу фазифікації вхідних даних у проекції простору ознак

зважену Гауссову дистанцію до кожного центру, формуючи вектор ступенів належності.

2. База правил

Система використовує набір правил, кількість яких дорівнює кількості активних кластерів. Для моделі Такагі-Сугено 0-го порядку правила мають вигляд:

Правило R_j : ЯКЩО вхідний вектор x подібний до розподілу кластера j (з параметрами μ_j, Σ_j), ТО вихідний коефіцієнт $y = \alpha_j$.

Ключовою особливістю є те, що вихідне значення α_j (консеквент) не є випадковим числом. Це значення **нестабільності сигналу**, яке було розраховане на етапі злиття кластерів і збережене в базі знань. Відповідно, правило фактично стверджує: *"Якщо ситуація схожа на цей режим, використовуй властиву йому швидкість адаптації"*.

3. Дефазифікація та Агрегація результату

Фінальний етап полягає в агрегації виходів усіх правил для отримання єдиного керуючого впливу $\alpha(t)$. Використовується метод **зваженого середнього**, який забезпечує плавну інтерполяцію між режимами:

$$\alpha(\mathbf{x}) = \frac{\sum_{k=1}^M \mu_k(\mathbf{x}) \cdot \alpha_k}{\sum_{k=1}^M \mu_k(\mathbf{x})}, \quad \alpha_k = 2(1 - BC_k)$$

де M — кількість активних правил (кластерів), $\mu_k(\mathbf{x})$ — ступінь належності до k -го кластера, α_k — консеквент правила, який визначається через нестабільність сигналу ($BC_k \in [0,5; 1]$ для злитих кластерів, тож $\alpha_k \in [0; 1]$).

Якщо ситуація є перехідною (наприклад, між «Вільним потоком» та «Синхронізованим потоком»), ваги розподіляються між кількома правилами, а результуюче α стає компромісним значенням.

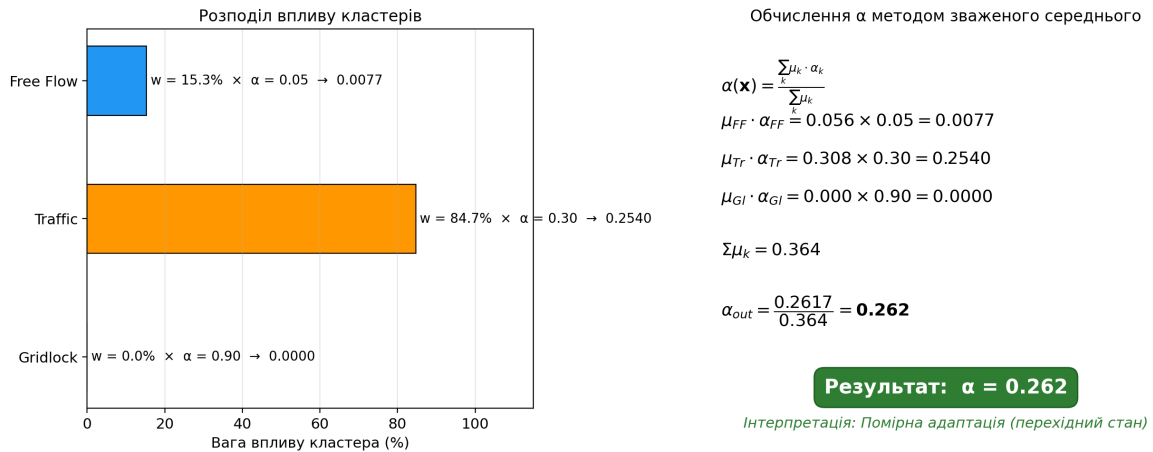


Рис. 2.10 - Результат дефазифікації керуючого впливу методом зваженого середнього на основі розрахованих ступенів належності

На рис. 2.10 показано покрокове обчислення дефазифікації для тієї ж вхідної точки Р. Ліва панель відображає розподіл впливу кластерів, права — формульний розрахунок α :

- Розподіл впливу:

- Кластер «Синхронізований потік» (*Traffic* на рис. 2.10) отримав вагу **84.6%** ($w_2 \approx 0.85$), оскільки точка Р знаходиться найближче до його центру.
- Кластер «Вільний потік» (*Free Flow* на рис. 2.10) отримав вагу **15.4%** ($w_1 \approx 0.15$), оскільки точка знаходиться на периферії його зони тяжіння.
- Кластер «Затор» (*Gridlock* на рис. 2.10) має **0.0%** впливу ($w_3 \approx 0$), оскільки точка знаходиться занадто далеко від його центру.

Розрахунок результату: Система виконує лінійну суперпозицію

виходів правил: $\alpha_{out} = \frac{(0.05 \cdot 0.154) + (0.30 \cdot 0.846) + (0.90 \cdot 0.0)}{0.154 + 0.846 + 0.0} \approx 0.0077 + 0.2538 = 0.261$

Отримане значення $\alpha = 0.261$ є математично обґрунтованим компромісом: система розуміє, що це переважно звичайний трафік ($\alpha \approx 0.3$), але з невеликим зсувом у бік вільного руху, тому результуючий коефіцієнт трохи зменшено для підвищення стабільності.

Такий підхід дозволяє системі уникати різких стрибків параметрів, забезпечуючи адаптивність, яка математично обґрунтована статистичною структурою реального трафіку.

2.3.2 Аналіз обчислювальної складності

Обчислювальна складність методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі визначається двома режимами функціонування: пакетною обробкою (оновлення бази знань) та потоковою обробкою (нечіткий висновок для кожного повідомлення).

Пакетна обробка (виконується кожні $N_{batch} = 400$ повідомлень):

- *Кластеризація K-Means*: $O(N_{batch} \cdot k \cdot d \cdot I)$, де $d = 3$ — розмірність простору ознак, k — кількість кластерів, I — кількість ітерацій збіжності.
- *Геометричний метод ліктя* виконує K-Means для $k = 2, \dots, K_{max}$, тому його сумарна складність: $\sum_{k=2}^{K_{max}} O(N_{batch} \cdot k \cdot d \cdot I) = O(K_{max}^2 \cdot N_{batch} \cdot d \cdot I)$, де $K_{max} = 10$.
- *Інтеграція мікро-кластерів*: $O(M \cdot d)$ на кожен мікро-кластер, де M — кількість глобальних режимів.

При типових значеннях ($k \leq 5$, $I \leq 20$, $K_{max} = 10$, $M \leq 10$) домінує складність методу ліктя. Підставляючи константи: $\sum_{k=2}^{10} (400 \cdot k \cdot 3 \cdot 20) \approx 1,3 \times 10^6$ операцій — менше 1 мс на сучасному процесорі, що підтверджує лінійну залежність від N_{batch} та придатність для обробки в реальному часі.

Потокова обробка (для кожного IoT-повідомлення): нечіткий висновок Сугено потребує $O(M \cdot d)$ операцій — обчислення ступеня

належності до кожного з M кластерів (по $d = 3$ вимірам) та зваженого усереднення. При типових $M \leq 10$, $d = 3$ це складає $O(1)$ — **постійний час** незалежно від розміру мережі, що підтверджується експериментальною затримкою 1,19 мс на повідомлення (розділ 4).

Таким чином, метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі забезпечує обробку в реальному часі: оновлення ваги кожного ребра графа потребує фіксованої кількості арифметичних операцій, а перенавчання бази знань відбувається на фоні, не блокуючи основний потік обробки.

Висновки до розділу 2

1. Вперше розроблено телекомунікаційну модель IoV-мережі у вигляді часозалежного зваженого орієнтованого графа $G_T = (V, E, W, T)$, в якій вага ребра має телекомунікаційну семантику — визначається як час обслуговування інформаційного потоку на сегменті та формується на основі вимірюваних QoS-індикаторів $[\eta, L, \tau]$. Модель задовольняє умову FIFO, що гарантує коректність застосування алгоритму A^* для пошуку оптимального маршруту.
2. Встановлено, що автоматичне формування бази нечітких правил на основі пакетно-інкрементальної кластеризації у просторі ознак (η, L, τ) з використанням метрик Махаланобіса та коефіцієнта Бхаттачар'ї усуває необхідність ручного налаштування параметрів нечіткого контролера і забезпечує автоматичну адаптацію до локальних характеристик кожного сегмента мережі без участі експерта.
3. Доведено, що обчислювальна складність потокової обробки методу адаптивної оцінки становить $O(1)$ на повідомлення, оскільки нечіткий висновок Такагі–Сугено потребує фіксованої кількості операцій незалежно від розміру мережі, а перенавчання бази знань виконується у фоновому режимі не блокуючи основний потік обробки. Це забезпечує теоретичну придатність методу для роботи в режимі реального часу в умовах IoV-мереж.

РОЗДІЛ 3

МЕТОД ПРОГНОЗУВАННЯ ІНФОРМАЦІЙНОГО НАВАНТАЖЕННЯ ІОВ-МЕРЕЖІ ТА ПОШУКУ ОПТИМАЛЬНОГО ШЛЯХУ НА ОСНОВІ МОДИФІКОВАНОГО АЛГОРИТМУ A^*

У розділі 2 розроблено телекомунікаційну модель Іов-мережі (п. 2.1) та метод адаптивної оцінки стану інформаційних потоків на сегментах Іов-мережі, який формує реактивну компоненту $g(n)$ у задачі пошуку оптимального шляху. Проте ефективність алгоритму A^* визначається не лише точністю поточної оцінки, а й якістю прогнозованої евристики $h(n, t)$, яка враховує очікуваний стан сегментів мережі та спрямовує пошук. У цьому розділі розроблено метод просторово-часового прогнозування інформаційного навантаження на основі часо-просторової графової нейронної мережі (TGNN) [33, 36] та показано, як його прогноз інтегрується в метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі модифікованого алгоритму A^* .

3.1 Метод просторово-часового прогнозування інформаційного навантаження на основі TGNN

Ефективність функціонування алгоритмів спрямованого пошуку (зокрема алгоритму A^*) [30] в Іов-мережах критично залежить від якості евристичної функції $h(n)$. Традиційні підходи використовують статичні евристики, засновані на геометричній відстані [5, 6, 30]: $h(n) = d(n, D)/v_{\max}$, де $v_{\max}$ — гранично допустима швидкість. Хоча такі методи гарантують виконання умов допустимості та монотонності [30], вони повністю ігнорують динаміку інформаційного навантаження, припускаючи постійну максимальну швидкість на всіх ділянках мережі. В умовах щільного міського середовища це призводить до суттєвого зниження ефективності пошуку: алгоритм продовжує розкривати вершини у напрямку, який є геометрично найкоротшим, навіть якщо цей напрямок відповідає сегменту з критичною концентрацією інформаційного навантаження.

Сучасні спроби вирішити цю проблему шляхом застосування машинного навчання для прямого прогнозування часу прибуття (ETA)

[56] стикаються з проблемою неузгодженості прогнозів. Передбачення нейромереж не гарантують монотонності евристики [30], що змушує алгоритм A^* повторно досліджувати вже відвідані вершини, нівелюючи теоретичний виграш від евристики.

Для усунення цього протиріччя у роботі розроблено **метод формування часозалежної евристики A^*** , суть якого полягає у зміні цільової змінної прогнозування. Замість абстрактного часу подорожі, модель навчається передбачати конкретний фізичний параметр — швидкість потоку, який надалі інтегрується у класичну кінематичну формулу $h = d/v$. Такий підхід зберігає фізичну інтерпретованість евристики та забезпечує узгодженість з метричними властивостями простору пошуку.

Реалізація цього підходу вимагає архітектури, здатної одночасно моделювати топологію дорожньої мережі та часову еволюцію транспортних потоків. У наступних підрозділах послідовно розглянуто: архітектуру прогнозної моделі TGNN (п. 3.1.1), побудову композитної евристичної функції (п. 3.1.2) та механізми кешування для забезпечення роботи в реальному часі (п. 3.1.3).

3.1.1 Архітектура прогнозної моделі TGNN

Для вирішення задачі просторово-часового прогнозування швидкості руху обрано архітектуру часо-просторової графової нейронної мережі (**TGNN** — Temporal Graph Neural Network). Ця модель аналізує послідовності станів графа («знімки») у часі [36], враховуючи як зв'язність перехресть, так і динаміку поширення заторів.

Розроблена архітектура складається з трьох послідовних блоків: просторового кодувальника на основі дифузійної згортки, часового кодувальника на основі керованих рекурентних блоків та прогнозного блоку.

1. Вхідні дані та вектор ознак

На вхід моделі подається тензор розмірності $(B \times T \times N \times F)$, де B — розмір пакету, $T = 12$ — кількість часових кроків (60 хвилин з інтервалом 5 хвилин), N — кількість вузлів графа (207 сенсорів для набору даних METR-LA [39]), $F = 6$ — розмірність вектора ознак для кожного вузла.

Вектор ознак вузла n у момент часу t містить шість компонент

(табл. 3.1):

Таблиця 3.1 - Вектор ознак вузла TGNN-моделі

#	Ознака	Формула / опис
0	Нормалізована швидкість	$(v_{n,t} - \mu_n) / \sigma_n$
1	Час доби (синус)	$\sin(2\pi \cdot t_{\text{day}} / 86400)$
2	Час доби (косинус)	$\cos(2\pi \cdot t_{\text{day}} / 86400)$
3	День тижня (синус)	$\sin(2\pi \cdot d_{\text{week}} / 7)$
4	День тижня (косинус)	$\cos(2\pi \cdot d_{\text{week}} / 7)$
5	Коефіцієнт зміни завантаження каналу	$v_{n,t}^{\text{norm}} - v_{n,t-1}^{\text{norm}}$

Нормалізація швидкості виконується **індивідуально для кожного датчика** з використанням його середнього значення μ_n та стандартного відхилення σ_n , обчислених на навчальній вибірці. На відміну від глобальної нормалізації, індивідуальна нормалізація для кожного датчика усуває систематичні зміщення, зумовлені різницею у характеристиках ділянок доріг (приміська автомагістраль із середньою швидкістю 60 mph та міська вулиця із 25 mph отримують порівнянні нормалізовані значення).

Циклічне кодування часу доби та дня тижня через пару ($\sin$, $\cos$) забезпечує неперервне представлення періодичних ознак: значення 23:55 та 00:05 є близькими у просторі ознак (на відміну від лінійного кодування, де вони були б максимально віддаленими). Це дозволяє моделі коректно моделювати періодичність транспортних потоків.

Коефіцієнт зміни завантаження каналу (δ -швидкість) відображає тенденцію зміни ситуації: додатне значення сигналізує про розвантаження, від'ємне — про формування затору.

2. Просторовий кодувальник: дифузійна згортка

Для моделювання просторових залежностей у транспортній мережі використано механізм **дифузійної згортки** (DiffusionConv) [39], який узагальнює процес випадкового блукання на орієнтованому графі. На відміну від стандартної графової згортки (GCN) [62, 118], що працює з симетричним оператором Лапласа, дифузійна згортка розрізняє **прямий** та **зворотний** напрямки поширення інформації.

Матриці переходу визначаються як:

$$P_f = D_{\text{out}}^{-1}A, \quad P_b = D_{\text{in}}^{-1}A^\top \quad (3.1)$$

де A — матриця суміжності орієнтованого графа дорожньої мережі, D_{out} та D_{in} — діагональні матриці степенів вихідних і вхідних ребер відповідно.

Прямий оператор P_f моделює поширення трафіку «за течією» (від поточної ділянки до наступних), а зворотний P_b — зворотний вплив (коли затор на виїзді уповільнює рух на попередніх ділянках). Така двонаправлена дифузія є критичною для IoV-мереж, де хвиля затору поширюється переважно у зворотному напрямку [39, 81, 82].

Вихід дифузійної згортки для K кроків формується конкатенацією $(2 \cdot + 1)$ доданків — початкового стану та K степенів кожної матриці переходу:

$$H' = \sum_{k=0}^K \theta_{f,k} P_f^k X + \sum_{k=0}^K \theta_{b,k} P_b^k X \quad (3.2)$$

де $\theta_{f,k}$, $\theta_{b,k}$ — навчальні параметри, $X \in \mathbb{R}^{N \times F}$ — матриця ознак вузлів. На практиці формула (3.2) реалізується ефективніше: $(2 \cdot + 1)$ доданків (включно з $P^0 X = X$) конкатенуються в один тензор, який обробляється єдиним лінійним перетворенням $W \in \mathbb{R}^{D \times (2 \cdot + 1)F}$, що є математичним еквівалентом, але зменшує кількість параметрів.

Просторовий кодувальник складається з двох послідовних шарів дифузійної згортки ($K = 2$) з нормалізацією шарів (LayerNorm) [115], активацією ReLU, регуляризацією відсіюванням та залишковим з'єднанням. Обробка кожного часового кроку $X \in \mathbb{R}^{N \times F}$ виконується у чотири послідовні кроки:

1. Перша дифузійна згортка: $Z_1 = \text{DC}_1(X)$;
2. Нормалізація та активація: $Z_2 = \text{ReLU}(\text{LN}_1(Z_1))$;
3. Друга дифузійна згортка: $Z_3 = \text{DC}_2(Z_2)$;
4. Нормалізація із залишковим з'єднанням:

$$\text{SpatEnc}(X) = \text{ReLU}(\text{LN}_2(Z_3) + W_{\text{res}} X),$$

де $W_{\text{res}} \in \mathbb{R}^{D \times F}$ — лінійна проекція, що перетворює вхідний вектор розмірності F до розмірності D прихованого простору, забезпечуючи коректне додавання у залишковому з'єднанні.

Обґрунтування вибору $K = 2$. Глибина дифузії $K = 2$ обрана з трьох міркувань.

По-перше, фізика дорожнього руху має властивість **локальності**: стан затору на конкретному перехресті сильно корелює зі станом його безпосередніх сусідів (1-hop) та сусідів другого порядку (2-hop). Вплив подій на відстані 20–30 перехресть є стохастичним і незначним для миттєвого прогнозу швидкості. Глибина $K = 2$ формує рецептивне поле, що охоплює мікрорайон — достатню область для виявлення тенденцій затору.

По-друге, збільшення глибини графової згортки призводить до **перезгладжування** [62, 101, 127]: операція згортки працює як фільтр низьких частот, і при $K \gg 2$ вектори прихованих станів усіх вузлів сходяться до єдиного стаціонарного значення, внаслідок чого модель втрачає здатність розрізняти завантажені та вільні ділянки.

По-третє, кожен додатковий крок дифузії вимагає множення матриць розмірності $N \times N$. Для задачі маршрутизації в реальному часі, де евристика обчислюється для кожного розкритого вузла, збільшення K призводить до неприпустимих затримок.

Обґрунтування вибору дифузійної згортки. Моделювання просторових залежностей на графах допускає декілька альтернативних підходів, які було розглянуто при виборі архітектури.

Спектральна згортка Чебишева [118], що лежить в основі моделі STGCN [40], апроксимує фільтр на графі поліномами Чебишева від симетричного нормалізованого лапласіану $L_{\text{sym}} = I - D^{-1/2} A D^{-1/2}$. Цей підхід має два суттєвих обмеження для IoV-мереж.

По-перше, лапласіан L_{sym} визначений лише для **неорієнтованих** графів (вимагає $A = A^T$) [118, 62], тоді як дорожня мережа є принципово орієнтованою: трафік рухається в одному напрямку, а затор поширюється у протилежному [39, 81, 82, 84].

По-друге, спектральний фільтр є **ізотропним** [33, 118] — тобто він агрегує інформацію від сусідів однаково з усіх напрямків, не розрізняючи,

звідки надійшов сигнал. Для транспортної мережі це означає, що модель не здатна відрізнити два фізично різні процеси: *прямий потік* (як трафік рухатиметься далі від поточної ділянки — «що попереду на маршруті?») та *зворотний потік* (як затор з наступних ділянок впливає на попередні — «що наближається ззаду?»).

Дифузійна згортка [39] усуває обидва обмеження: замість єдиного симетричного оператора вона використовує дві окремі матриці переходу (формула (3.1)): прямий оператор $P_f = D_{\text{out}}^{-1}A$, що моделює рух трафіку за напрямком доріг, та зворотний оператор $P_b = D_{\text{in}}^{-1}A^T$, що моделює зворотне поширення затору. Кожен оператор має власні навчальні параметри $\theta_{f,k}$ та $\theta_{b,k}$ (формула (3.2)), що дозволяє моделі самостійно визначити, наскільки важливим є прямий та зворотний сигнали для кожного кроку дифузії k .

Адаптивна матриця суміжності [41] (Graph WaveNet) навчає додаткову матрицю прихованих зв'язків $A_{\text{adpt}} = \text{softmax}(E_1 E_2^T)$ розмірності $N \times N$, де E_1, E_2 — навчальні вкладення вузлів. Для набору даних METR-LA ($N = 207$) це додає $2 \times 207 \times d$ параметрів та $O(N^2)$ обчислень на кожен крок. Хоча цей механізм підвищує точність (MAE = 2,69 проти 2,77 для DCRNN), у контексті задачі маршрутизації на великих графах ($N \sim 10^4$) квадратична складність є неприйнятною. Крім того, адаптивна суміжність моделює **статичні** приховані зв'язки, які не змінюються з часом, тоді як для евристики A^* необхідна реакція на **поточний** стан трафіку.

Відтак, дифузійна згортка є оптимальним компромісом між точністю моделювання фізики транспортних потоків, обчислювальною ефективністю та здатністю розрізняти прямий і зворотний напрямки поширення.

3. Часовий кодувальник: GRU

Послідовність просторових представлень $\{h_1, h_2, \dots, h_T\}$, де $h_t = \text{SpatEnc}(X_t) \in \mathbb{R}^{N \times D}$, обробляється для кожного вузла окремо блоком керованих рекурентних одиниць (GRU) [63] з двома шарами та прихованою розмірністю 128.

GRU на кожному часовому кроці t отримує два входи: просторове представлення h_t від кодувальника та власний прихований стан s_{t-1} з

попереднього кроку. Оновлення виконується у три етапи:

Вентиль оновлення — визначає, яку частку попереднього стану зберегти:

$$z_t = \sigma(W_z [h_t, s_{t-1}]).$$

Вентиль скидання — які компоненти пам'яті скинути:

$$r_t = \sigma(W_r [h_t, s_{t-1}]).$$

Оновлення прихованого стану:

$$s_t = (1 - z_t) \odot s_{t-1} + z_t \odot \tanh(W_s [h_t, r_t \odot s_{t-1}]), \quad (3.3)$$

де $[h_t, s_{t-1}]$ — конкатенація векторів, σ — логістична сигмоїда (виходи від 0 до 1), $\odot$ — поелементне множення, W_z, W_r, W_s — навчальні матриці ваг. Інтуїтивно, формула (3.3) реалізує зважене оновлення: компонент $(1 - z_t) \odot s_{t-1}$ зберігає попередню інформацію (наприклад, довготривалий тренд зміни швидкості), а компонент $z_t \odot \tanh(\dots)$ додає нову інформацію з поточного часового кроку.

Двошарова архітектура GRU з нормалізацією шарів на виході та регуляризацією відсіюванням ($p = 0,1$ між шарами) дозволяє моделі виявляти часові залежності різного масштабу [63, 119]: від короткочасних коливань швидкості в межах окремого часового кроку до середньомасштабних патернів (ранковий/вечірній пікові навантаження).

Обґрунтування вибору GRU. Для часового кодування послідовностей розглянуто три альтернативи.

LSTM (Long Short-Term Memory) [119] — класична рекурентна архітектура з трьома вентилями (введення, забування, виведення) та окремою коміркою пам'яті. GRU є спрощенням LSTM: два вентиля замість трьох, відсутність окремої комірки пам'яті. Для задачі прогнозування швидкості руху, де вхідна послідовність є короткою ($T = 12$ кроків), GRU забезпечує порівнянну або кращу точність при **на 33% менше параметрів** на один шар, що підтверджується систематичним порівнянням на задачах моделювання послідовностей [63]. Менша кількість параметрів є критичною для задачі маршрутизації, де модель має працювати у сервісі

з жорсткими вимогами до затримки.

Transformer [120] з механізмом самоуваги (self-attention) дозволяє моделювати довгострокові залежності без обмежень рекурентних архітектур. Проте для послідовності довжиною $T = 12$ перевага Transformer є мінімальною: затухання градієнтів при $T \leq 20$ є несуттєвим для GRU [63], тоді як обчислювальна складність self-attention складає $O(T^2D)$ на відміну від $O(TD^2)$ для GRU. Крім того, Transformer потребує позиційного кодування та значно більшої кількості параметрів, що збільшує ризик перенавчання на обмежених обсягах даних окремих сенсорів.

Отже, GRU забезпечує належний баланс між точністю моделювання часових патернів, обчислювальною ефективністю та компактністю моделі для задачі короткострокового прогнозування швидкості.

4. Прогнозний блок

Фінальний прихований стан GRU $s_T \in \mathbb{R}^{128}$ подається на пов'язаний прогнозний блок, що складається з двох лінійних шарів з активацією ReLU та регуляризацією відсіюванням:

$$v_{\text{TGNN}}(n, t) = W_2 \cdot \text{ReLU}(W_1 \cdot s_T + b_1) + b_2$$

де $W_1 \in \mathbb{R}^{64 \times 128}$, $W_2 \in \mathbb{R}^{1 \times 64}$. Виходом є вектор прогнозованих нормалізованих швидкостей для **всіх** N вузлів графа одночасно, що дозволяє виконувати лише один прямий прохід моделі замість N окремих обчислень.

Загальна архітектура моделі наведена у табл. 3.2.

Таблиця 3.2 - Архітектура моделі TGNN

Блок	Компоненти	Вхід	Вихід
Просторовий кодувальник	DiffusionConv ($K=2$) $\times 2$ + LN + ReLU + залишкове з'єдн.	$(B, N, 6)$	$(B, N, 64)$
Часовий кодувальник	GRU (2 шари, $h=128$) + LayerNorm	$(B \cdot N, 12, 64)$	$(B \cdot N, 128)$
Прогнозний блок	$128 \rightarrow 64 \rightarrow \text{ReLU} \rightarrow$ відсіювання $\rightarrow 1$	$(B \cdot N, 128)$	(B, N)
Загальна кількість параметрів			205 313

Концептуальна схема просторово-часової агрегації інформації наведена на рис. 3.1.

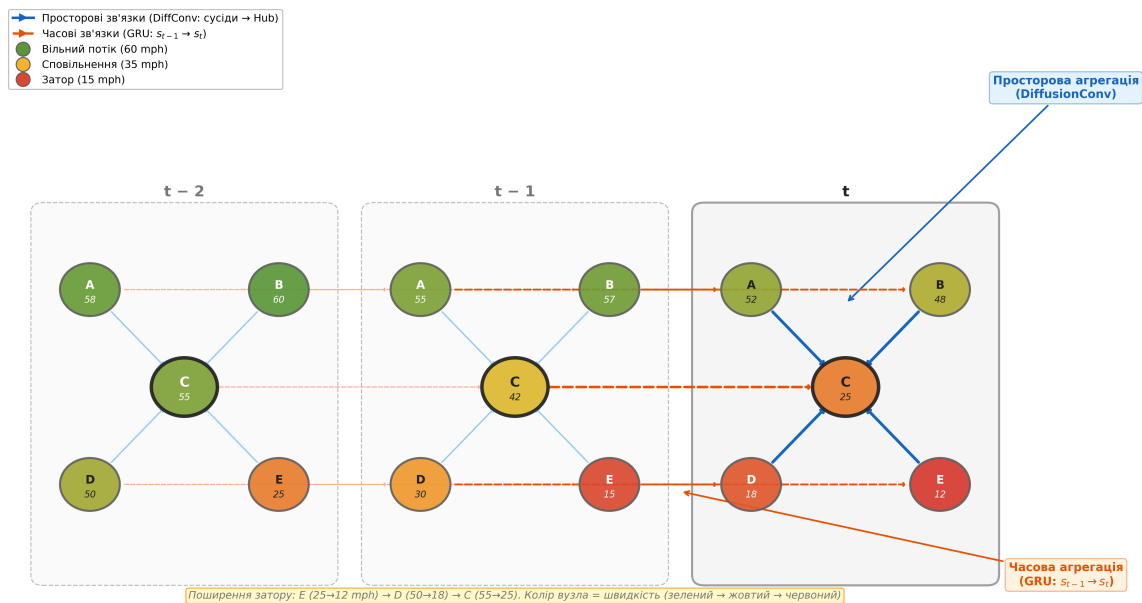


Рис. 3.1 - Концептуальна схема просторово-часової агрегації інформації в моделі TGNN

На рис. 3.1 суцільні сині зв'язки відображають роботу просторового кодувальника (дифузійна згортка): вузол-концентратор (Hub C) агрегує ознаки від безпосередніх сусідів (A, B, D, E), формуючи локальний просторовий контекст. Пунктирні помаранчеві зв'язки відображають рекурентні зв'язки GRU: кожен вузол отримує інформацію не лише від сусідів, а й від власного прихованого стану з попередніх часових кроків. Поєднання цих двох потоків дозволяє моделі прогнозувати стан

вузла, враховуючи як топологічну структуру затору, так і динаміку його розвитку в часі.

Детальну блок-схему потоку даних через шари моделі з розмірностями тензорів на кожному етапі наведено на рис. 3.2. Кожен великий блок відповідає функціональному модулю мережі, а всередині блоків перелічено конкретні операції — конкатенацію ознак (Concat), лінійне перетворення (Linear), нормалізацію по шару (LayerNorm), функцію активації (ReLU) та механізм залишкового з'єднання ($\oplus$). Праворуч від кожного блоку вказано розмірності тензорів на відповідному етапі обробки.

Архітектура TGNN: потік даних та розмірності тензорів

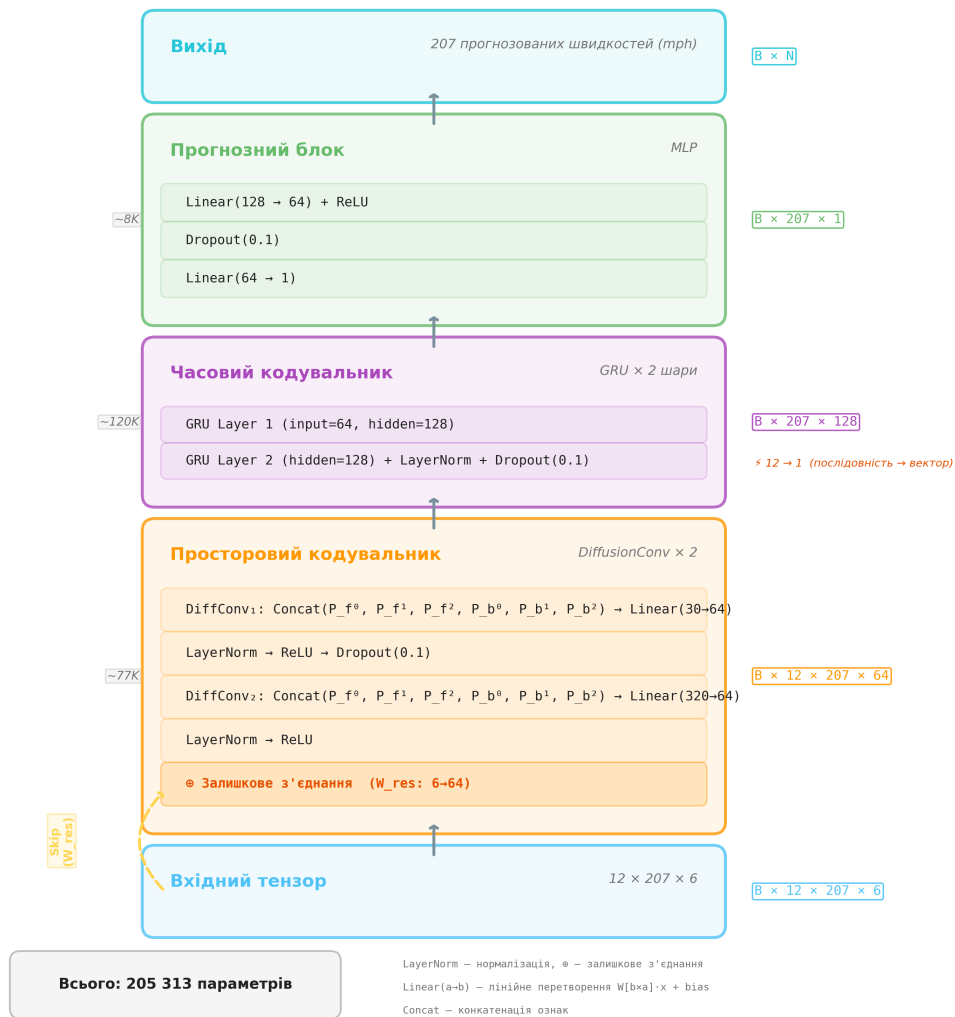


Рис. 3.2 - Архітектура TGNN: вертикальна блок-схема потоку даних від вхідного тензору ($B \times T \times N \times 6$) через просторовий кодувальник (DiffusionConv $K=2$ із залишковим з'єднанням), часовий кодувальник (двошарова GRU) до прогнозного блоку (MLP $128 \rightarrow 64 \rightarrow 1$). В середині кожного блоку зазначено конкретні операції; праворуч — розмірності тензорів

3.1.2 Побудова композитної евристичної функції

Наявність прогнозованої моделі є необхідною, але недостатньою умовою для ефективної роботи алгоритму пошуку шляху. Критичним питанням залишається спосіб інтеграції прогнозів у евристичну функцію.

Існуючі підходи базуються на прямій регресії часу прибуття (ETA) [32, 56], де модель апроксимує час подорожі як єдину скалярну величину. Недоліком такого підходу є те, що прогнози залишкового часу не гарантують монотонності евристики [30], а це змушує алгоритм A^*

повторно досліджувати вже відвідані вершини.

Для вирішення цієї проблеми розроблено метод, що базується на **декомпозиції** функції вартості. Евристична функція $h(n, t)$ конструюється як **комполитна функція**, що поєднує детерміновану геометричну складову та стохастичну компоненту, прогнозовану нейронною мережею:

$$h(n, t) = \frac{d_{\text{gc}}(n, D)}{v_{\text{TGNN}}(n, t)} \quad (3.4)$$

де:

- $d_{\text{gc}}(n, D)$ — відстань за дугою великого кола (great-circle distance) між поточною вершиною n та вершиною призначення D . Ця компонента є **незмінною геометричною константою**, що гарантує спрямованість пошуку до цілі;
- $v_{\text{TGNN}}(n, t)$ — прогнозована середня швидкість потоку у вершині n на момент часу t , отримана на виході моделі TGNN після денормалізації: $v = v^{\text{norm}} \cdot \sigma_n + \mu_n$. Ця компонента забезпечує адаптацію до поточних дорожніх умов.

Величина $v_{\text{TGNN}}(n, t)$ враховує топологічний контекст завдяки рецептивному полю дифузійної згортки (п. 3.1.1), що забезпечує реагування на наближення до затору ще до фактичного входу в нього.

Часозалежний розрахунок евристики. Розрахунок виконується ітеративно в процесі роботи алгоритму A^* . Для кожного вузла v , до якого планується перехід з поточного вузла u , евристика обчислюється на момент **очікуваного прибуття**:

$$t_{\text{arrival}}(v) = t_{\text{current}} + w(u, v) \quad (3.5)$$

де $w(u, v)$ — вага ребра (час обслуговування інформаційного потоку на сегменті). Евристична оцінка $h(v, t_{\text{arrival}})$ базується на прогнозі стану мережі саме в той момент, коли транспортний засіб опиниться у даній точці.

Фізична коректність та обмеження швидкості. Така структура евристики забезпечує природну реакцію на зміну дорожніх умов. При зниженні прогнозованої швидкості (наприклад, при виникненні затору)

знаменник дробу зменшується, що призводить до зростання вартості $h(n, t)$. Для запобігання діленню на нуль та числовій нестабільності передбачено обмеження знизу: $v_{\text{TGNN}}^{\text{eff}}(n, t) = \max(v_{\text{TGNN}}(n, t), v_{\min})$, де $v_{\min} = 5 \text{ mph}$ — мінімальна розрахункова швидкість. Це формує у просторі пошуку «потенціальні бар'єри» — області високої вартості, які алгоритм A^* оминає. Навпаки, на вільних ділянках ($v \rightarrow v_{\max}$) евристика наближається до мінімально можливої оцінки.

Щодо формальної допустимості. Евристика $h(n, t) = d_{\text{gc}}/v_{\text{TGNN}}$ не є формально допустимою у класичному сенсі [30], оскільки $d_{\text{gc}} \leq d_{\text{road}}$ (геодезична відстань менша за дорожню), але v_{TGNN} може як завищувати, так і занижувати фактичну швидкість. Проте це обмеження є теоретичним: експериментальна верифікація на еталонному наборі METR-LA (п. 3.3.3) показала, що втрата оптимальності маршруту не перевищує **0,00–0,08%** (0,67 секунди на 14-хвилинному маршруті). Відповідно, евристика є **практично допустимою** — її відхилення від оптимуму є зневажно малим порівняно з природною варіативністю часу подорожі в реальних умовах.

Візуалізація принципу дії композитної евристики наведена на рис. 3.3.

На рис. 3.3 показано, як композитна евристика змінює поведінку алгоритму A^* : замість прямування геометрично найкоротшим шляхом через зону критичного інформаційного навантаження (де $v_{\text{TGNN}} \rightarrow 0$ і, відповідно, $h \rightarrow \infty$), алгоритм обирає обхідний маршрут через ділянки з вищим прогнозованим значенням фізичного індикатора стану навантаження. Евристика є часозалежною: у ранковий та вечірній пікові години значення $h(n, t)$ для завантажених ділянок різко зростає, формує «потенціальний бар'єр», тоді як для вільних ділянок залишається стабільно низьким.

Дискретизація часу в евристичній функції. Прогнози моделі TGNN є дискретними у часі (з кроком $\Delta t = 5 \text{ хв}$) [39], тому необхідно визначити, яке значення швидкості використовувати для довільного моменту часу t при обчисленні евристики $h(n, t)$.

Застосовано **кусково-сталу апроксимацію**: довільний момент часу t відображається на дискретний часовий крок за формулою:

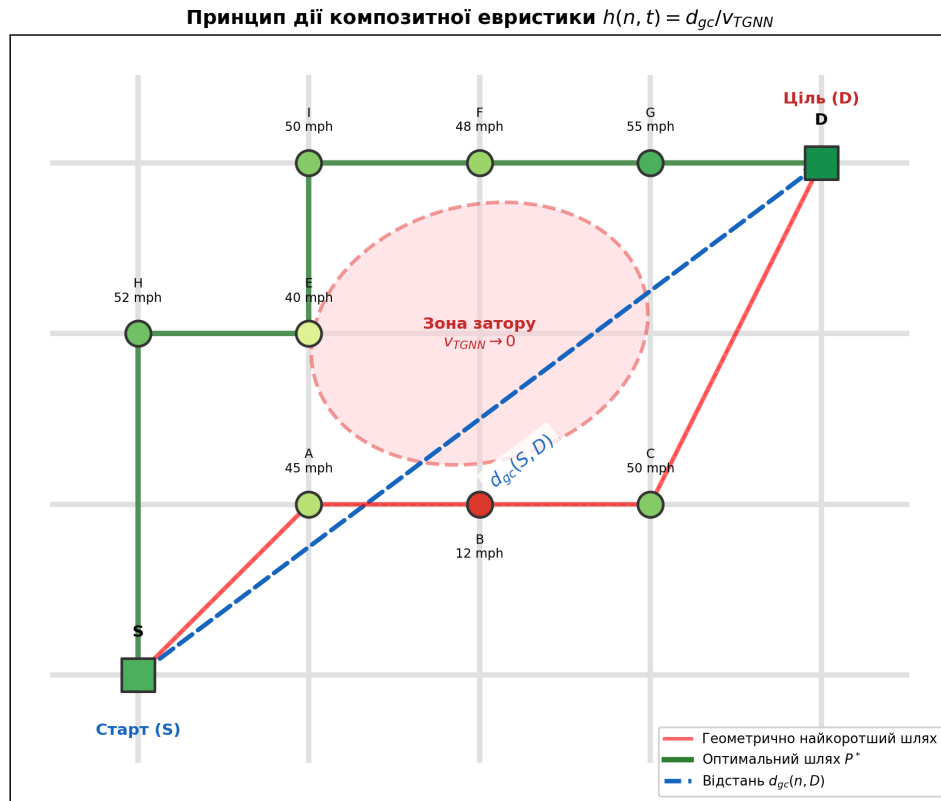


Рис. 3.3 - Принцип дії композитної евристики $h(n, t) = d_{gc}/v_{TGNN}$: червоним позначено геометрично найкоротший шлях через зону критичного інформаційного навантаження, зеленим — оптимальний шлях P^* , що обходить зону з низькою прогнозованою швидкістю як фізичним індикатором навантаження

$$t_k = \left\lfloor \frac{t}{\Delta t} \right\rfloor, \quad \hat{v}(n, t) = \hat{v}(n, t_k) \quad \forall t \in [t_k \cdot \Delta t, (t_k + 1) \cdot \Delta t) \quad (3.6)$$

тобто для всіх моментів часу в межах одного 5-хвилинного інтервалу використовується один і той самий прогноз швидкості. Такий підхід є обґрунтованим з двох причин: (1) прогноз TGNN $\hat{v}(n, t_k)$ відповідає **середній швидкості за весь інтервал** $[t_k \cdot \Delta t, (t_k + 1) \cdot \Delta t)$, а не миттєвому значенню на його початку, тому він однаково валідний для будь-якого моменту всередині інтервалу; (2) лінійна інтерполяція між сусідніми кроками потребувала б двох прямих проходів моделі замість одного, подвоюючи обчислювальні витрати, при цьому інтерпольоване значення не підтримується навчальними даними — модель ніколи не тренувалася на проміжних значеннях між 5-хвилинними зразками.

3.1.3 Кешування прогнозів та оптимізація обчислень

Критичною вимогою до евристичної функції алгоритму A^* є обчислювальна ефективність: евристика викликається для кожного розкритого вузла, і її вартість безпосередньо впливає на загальний час пошуку маршруту. Наївна реалізація, де для кожного виклику $h(n, t)$ виконується окремий прямий прохід нейронної мережі, є неприйнятною: час прямого проходу TGNN для одного зразка становить 1,2 мс, а при пошуку маршруту розкривається до 100 вершин.

Для вирішення цієї проблеми запропоновано стратегію **пакетного кешування**, яка використовує ключову архітектурну властивість TGNN: модель прогнозує швидкості для **всіх** N вузлів одночасно за один прямий прохід.

Механізм кешування працює наступним чином:

1. При зверненні до евристики $h(n, t)$ обчислюється дискретний часовий крок $t_k = \lfloor t/\Delta t \rfloor$, де $\Delta t = 5$ хвилин відповідає роздільності навчальних даних. Важливо, що час t для кожного вузла є **індивідуальним**: він дорівнює часу старту плюс накопичена вартість шляху $g(n)$, тобто моменту, коли транспортний засіб фактично досягне цього вузла.
2. Якщо для кроку t_k вже є запис у словнику $\mathcal{C} : t_k \mapsto \hat{V}(t_k)$, то прогноз витягується з нього ($O(1)$) і обчислюється $h(n, t) = d_{gc}(n, D)/\hat{V}_n(t_k)$.
3. Якщо запису немає — модель виконує один прямий прохід і зберігає вектор $\hat{V}(t_k) \in \mathbb{R}^N$ для всіх N вузлів.
4. При переході до наступного часового кроку кеш доповнюється новим записом.

Приклад. Нехай маршрут починається о 8:26, $\Delta t = 5$ хв (інтервали: 8:25–8:29, 8:30–8:34, ...). Алгоритм A^* розкриває: вузол a (прибуття о 8:27, ребро $S \rightarrow a$ зайняло 1 хв), вузол b (прибуття о 8:28) — обидва потрапляють у часовий крок $t_k = 08:25$, отже потрібен лише **один** прямий прохід TGNN, результат якого — вектор прогнозованих швидкостей для всіх 207 вузлів. Далі вузол c досягається о 8:31 (ребро $a \rightarrow c$ зайняло 4 хв)

— це вже крок $t_k = 08:30$, тому виконується **другий** прямий прохід, який враховує оновлену ситуацію на дорогах. На 14-хвилинному маршруті (~ 100 розкритих вершин) замість 100 прямих проходів (~ 120 мс) виконується лише 2–3 ($\sim 3,6$ мс).

Денормалізація ($v = v^{\text{norm}} \cdot \sigma_n + \mu_n$) виконується один раз при заповненні кешу.

Отже, амортизована вартість обчислення евристики для одного вузла становить $O(1)$, що є порівнянним з вартістю обчислення статичної геометричної евристики.

3.2 Результати навчання TGNN та порівняння з існуючими моделями

3.2.1 Умови навчання та протокол експерименту

Для навчання та оцінки запропонованої моделі TGNN використано еталонний набір даних METR-LA [39] — зібрання телеметрії дорожнього руху автомагістралей округу Лос-Анджелес, що містить показники 207 сенсорів за 4 місяці (34 272 часових кроки з інтервалом 5 хвилин). Цей набір обрано з двох причин: (1) він є де-факто стандартом порівняння моделей просторово-часового прогнозування трафіку [39, 40, 41, 127] та (2) він містить достатню кількість інцидентів, заторів та циклічних патернів (ранковий/вечірній пік) для верифікації стійкості прогнозів.

Дані поділено на навчальну (70%, 23 978 зразків), валідаційну (10%, 3 415 зразків) та тестову (20%, 6 843 зразки) вибірки відповідно до стандартного протоколу [39]. Горизонт прогнозування — один крок (5 хвилин), що відповідає потребам системи маршрутизації в реальному часі.

Параметри навчання наведено у табл. 3.3.

Таблиця 3.3 - Параметри навчання моделі TGNN

Параметр	Значення
Функція втрат	MSE (середньоквадратична помилка)
Оптимізатор	Adam [116] ($\beta_1 = 0,9$, $\beta_2 = 0,999$)
Регуляризація	зменшення ваг = 10^{-5} , обрізка градієнтів = 5,0
Розклад швидкості навчання	лінійний прогрів (5 епох) + косинусне згасання [117] від 10^{-3} до 10^{-5}
Кількість епох	80 (найкраща модель — епоха 73)
Розмір пакету	128
Раннє зупинення	15 епох без покращення
Обладнання	GPU Tesla T4 (Google Colab)
Час навчання	~30 хвилин (80 епох)

У якості функції втрат обрано MSE. Попередні експерименти показали, що додавання регуляризації нерівності трикутника ($\lambda_{\text{tri}} > 0$) не дає статистично значущого покращення якості маршрутизації при горизонті прогнозування 5 хвилин. Це пояснюється тим, що при короткому горизонті прогнози швидкості є достатньо точними (MAE = 2,10 миль/год, див. табл. 3.4), і додатковий регуляризатор лише ускладнює оптимізацію основної задачі — мінімізації похибки прогнозування.

3.2.2 Результати на тестовій вибірці та порівняння з еталонними моделями

Результати оцінки запропонованої моделі TGNN на тестовій вибірці METR-LA у порівнянні з еталонними методами наведено у табл. 3.4.

Таблиця 3.4 - Порівняння точності прогнозування швидкості на еталонному наборі METR-LA (горизонт 5 хвилин)

Модель	MAE, mph	RMSE, mph	MAPE, %	Парам.
Historical Average	4,16	7,80	12,95	—
ARIMA	3,99	8,21	9,60	—
FC-LSTM	3,44	6,30	9,60	—
STGCN [40]	2,88	5,74	7,62	~300 тис.
Graph WaveNet [41]	2,69	5,15	6,90	—
DCRNN [39]	2,77	5,38	7,30	~300 тис.
TGNN (запропонована)	2,102	3,671	5,08	205 313

Результати базових моделей (Historical Average, ARIMA, FC-LSTM) наведено за даними [39].

Запропонована модель перевершує всі розглянуті еталонні підходи за трьома стандартними метриками якості прогнозу. Найбільш показовим є порівняння з DCRNN [39], архітектура якої стала основою для дифузійної згортки в запропонованій моделі:

- **MAE** (середня абсолютна помилка — вимірює середній модуль відхилення прогнозу від реального значення, mph): 2,102 проти 2,77 mph — покращення на **24%**;
- **RMSE** (корінь середньоквадратичної помилки — чутливіша до великих відхилень, оскільки квадрат підсилює вплив різких викидів, mph): 3,671 проти 5,38 mph — покращення на **32%**, що свідчить про ефективнішу обробку різких змін швидкості при інцидентах;
- **MAPE** (середня абсолютна відсоткова помилка — виражає похибку у відсотках від реального значення, що дозволяє порівнювати точність незалежно від масштабу швидкостей): 5,08% проти 7,30% — покращення на **30%**, що забезпечує достатню точність для ранжування ребер графа в алгоритмі A*.

Компактність архітектури. Модель TGNN має **205 313 параметрів** — на 32% менше, ніж DCRNN (~300 тис.). Це досягнуто завдяки трьом архітектурним рішенням:

1. **Конкатенація замість окремих проекцій.** У дифузійній згортці $(2. + 1) = 5$ доданків конкатенуються і обробляються єдиним

лінійним перетворенням, замість окремих матриць ваг для кожного кроку дифузії;

2. **Спільний просторовий кодувальник для всіх кроків.** Один і той самий набір ваг DiffusionConv застосовується до кожного з $T = 12$ часових кроків, що зменшує кількість параметрів у T разів порівняно з архітектурами, де кожен крок має власний кодувальник;
3. **Компактний прогнозний блок.** Двошаровий блок $128 \rightarrow 64 \rightarrow 1$ замість більш глибоких архітектур, що є достатнім для задачі однокрокового прогнозу.

Компактність має безпосереднє практичне значення: менша кількість параметрів забезпечує швидше обчислення прогнозу, менше споживання пам'яті та можливість розгортання на пристроях з обмеженими ресурсами (крайові обчислення в IoT-інфраструктурі).

Для наочного порівняння результатів на рис. 3.4 наведено стовпчикову діаграму трьох ключових метрик.

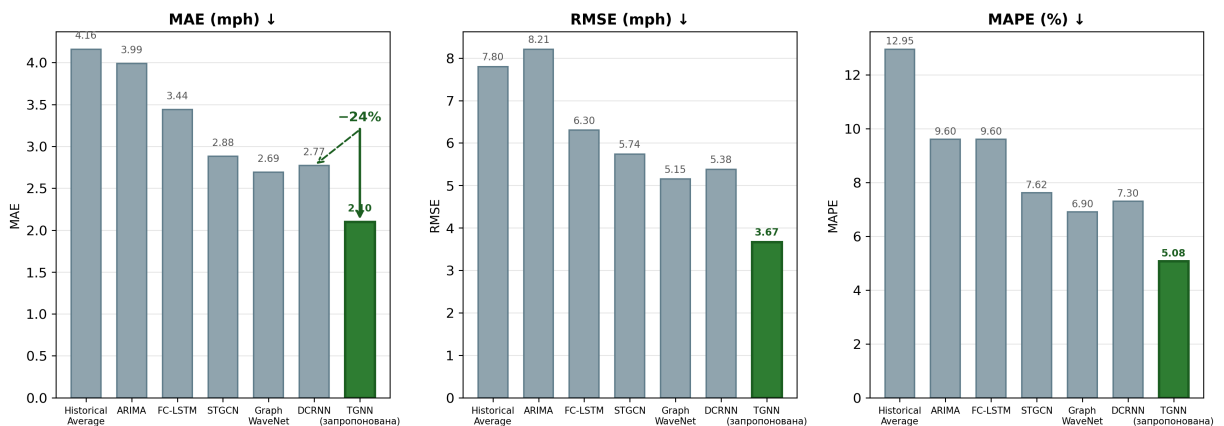


Рис. 3.4 - Порівняння точності прогнозування швидкості на еталонному наборі METR-LA: запропонована модель TGNN (зелений) перевершує всі базові моделі за MAE, RMSE та MAPE. Стрілкою позначено покращення на 24% щодо DCRNN

3.2.3 Аналіз процесу навчання

На рис. 3.5 наведено динаміку навчання моделі TGNN протягом 80 епох.

Аналіз кривих навчання дозволяє виділити три характерних етапи:

TGNN — навчання на METR-LA (207 сенсорів)

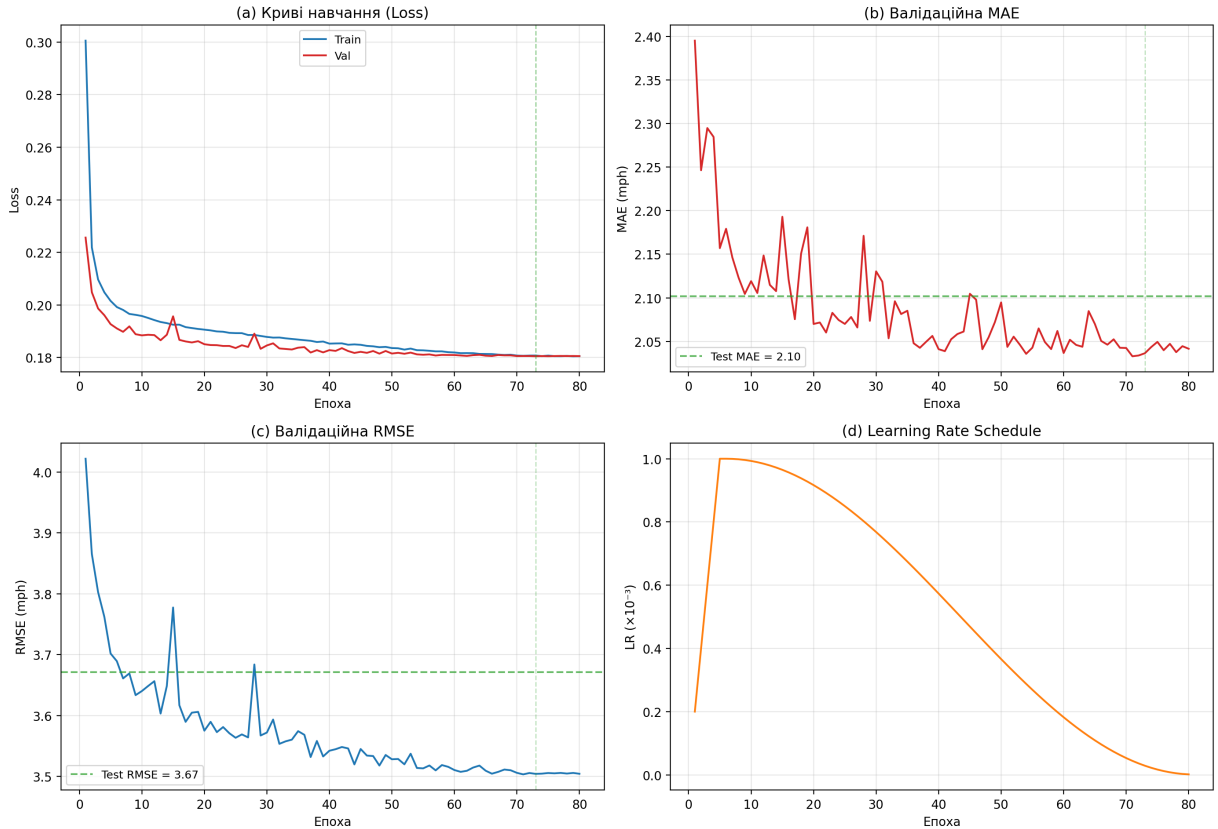


Рис. 3.5 - Динаміка навчання моделі TGNN: криві втрат, валідаційні метрики та розклад швидкості навчання

- 1. Швидка збіжність (епохи 1–10).** Протягом фази лінійного прогріву (перші 5 епох) та початку косинусного згасання навчальна втрата зменшується з 0,30 до 0,20, а валідаційна MAE — з 2,40 до 2,15 mph. Модель швидко засвоює базові просторово-часові закономірності (циклічність ранкового/вечірнього піку, кореляція між сусідніми сенсорами).
- 2. Тонка оптимізація (епохи 10–60).** Косинусне згасання швидкості навчання від 10^{-3} до $\sim 10^{-4}$ забезпечує поступове зменшення MAE до 2,04 mph. На цьому етапі модель навчається розрізняти тонкі патерни: різницю між буднім та вихідним днем, ефект поширення заторів між сусідніми ділянками.
- 3. Стабілізація (епохи 60–80).** При швидкості навчання $< 10^{-4}$ метрики стабілізуються. Розрив між навчальною та валідаційною втратами залишається мінімальним ($\sim 0,001$), що свідчить про відсутність перенавчання. Найкращу модель обрано після 73-ї

епохи.

Відсутність суттєвого розриву між навчальною та валідаційною кривими підтверджує ефективність обраної стратегії регуляризації: залишкове з'єднання у просторовому кодувальнику, нормалізація шарів у часовому кодувальнику та регуляризація відсіюванням ($p = 0,1$) у прогнозованому блоці забезпечують достатню узагальнювальну здатність без надлишкового обмеження ємності моделі.

3.3 Метод пошуку оптимального шляху на основі модифікованого алгоритму A^*

У попередніх підрозділах розроблено дві ключові компоненти алгоритму A^* для часозалежної IoV-мережі: (1) метод адаптивної оцінки стану інформаційних потоків на сегментах мережі (Розділ 2), що забезпечує формування компоненти $g(n)$ — фактичної характеристики обслуговування вже пройденої частини шляху; та (2) метод просторово-часового прогнозування інформаційного навантаження (п. 3.1), що формує компоненту $h(n, t)$ — прогнозну оцінку залишкового часу обслуговування інформаційного потоку. У цьому підрозділі описано інтеграцію обох компонент у процедуру пошуку оптимального шляху інформаційного потоку.

3.3.1 Формалізація методу пошуку оптимального шляху

Метод пошуку оптимального шляху визначає функцію оцінки вершини n у момент часу t як:

$$f(n) = g_{\text{adaptive}}(n) + h_{\text{TGNN}}(n, t) \quad (3.7)$$

де компоненти мають чітко розділені ролі:

- $g_{\text{adaptive}}(n) = \sum_{(i,j) \in \text{path}(s,n)} w_{ij}(t)$ — сумарний час обслуговування інформаційного потоку від стартової вершини s до поточної вершини n , де ваги ребер $w_{ij}(t) = L_{ij}/\hat{v}_{ij}(t)$ оновлюються адаптивно на основі нечіткого висновку (Розділ 2);
- $h_{\text{TGNN}}(n, t) = d_{\text{gc}}(n, D)/v_{\text{TGNN}}(n, t)$ — прогнозна оцінка мінімального

сумарного часу обслуговування інформаційних потоків на залишковому відрізку маршруту від вершини n до цілі D , де $v_{\text{TGNN}}(n, t)$ є прогнозом TGNN стану інформаційного навантаження $\lambda_{\text{seg}}(t)$ на сегментах залишку маршруту у момент очікуваного прибуття до вершини n .

На відміну від h_{geo} , що припускає мінімальне навантаження $\lambda_{\text{seg}} \rightarrow \lambda_{\text{min}}$, евристика h_{TGNN} враховує реальний прогнозний стан $\lambda_{\text{seg}}(t)$, що дозволяє A^* проактивно обирати шлях через сегменти з нижчим очікуваним інформаційним навантаженням.

Такий поділ забезпечує **взаємодоповнюваність**: компонента g оцінює фактичну поточну ситуацію (реактивна обробка), а компонента h — очікувану майбутню ситуацію (проактивне прогнозування). Класичні методи (Дейкстри [29], ієрархічні підходи [44]) оперують лише статичними або реактивними вагами, що не дозволяє запобігати потраплянню у зони формування сплесків інформаційного навантаження.

3.3.2 Алгоритм пошуку оптимального шляху P^*

Алгоритм пошуку шляху P^* реалізує стандартну процедуру A^* [30] з двома модифікаціями: (1) ваги графа оновлюються в реальному часі на основі потоку IoT-повідомлень через нечіткий контролер (Розділ 2); (2) евристична функція є часозалежною та використовує пакетне кешування прогнозів TGNN (п. 3.1.3).

Процес пошуку складається з чотирьох етапів:

1. **Ініціалізація.** Завантаження графа дорожньої мережі $G = (V, E)$ з актуальними вагами. Формування черги пріоритетів з початковою вершиною s : $f(s) = 0 + h_{\text{TGNN}}(s, t_0)$.
2. **Адаптивна оцінка вартості.** При розкритті вершини u для кожного сусіда v обчислюється вага ребра $w_{uv}(t)$ на основі згладженої оцінки швидкості $\hat{v}_{uv}(t)$, отриманої від нечіткого адаптивного контролера: $w_{uv}(t) = L_{uv} / \hat{v}_{uv}(t)$.
3. **Прогнозна евристика.** Для вершини v обчислюється $h_{\text{TGNN}}(v, t_{\text{arrival}})$ на момент очікуваного прибуття $t_{\text{arrival}} = t_{\text{current}} + w_{uv}$.

Прогноз витягується з кешу або ініціює один прямий прохід TGNN для всіх вузлів графа.

4. **Оптимізація.** Алгоритм знаходить шлях P^* , що мінімізує сумарний час обслуговування інформаційних потоків:

$$P^* = \arg \min_P \sum_{(i,j) \in P} w_{ij}(t_i)$$

Структурна схема взаємодії компонентів методу наведена на рис. 3.6.

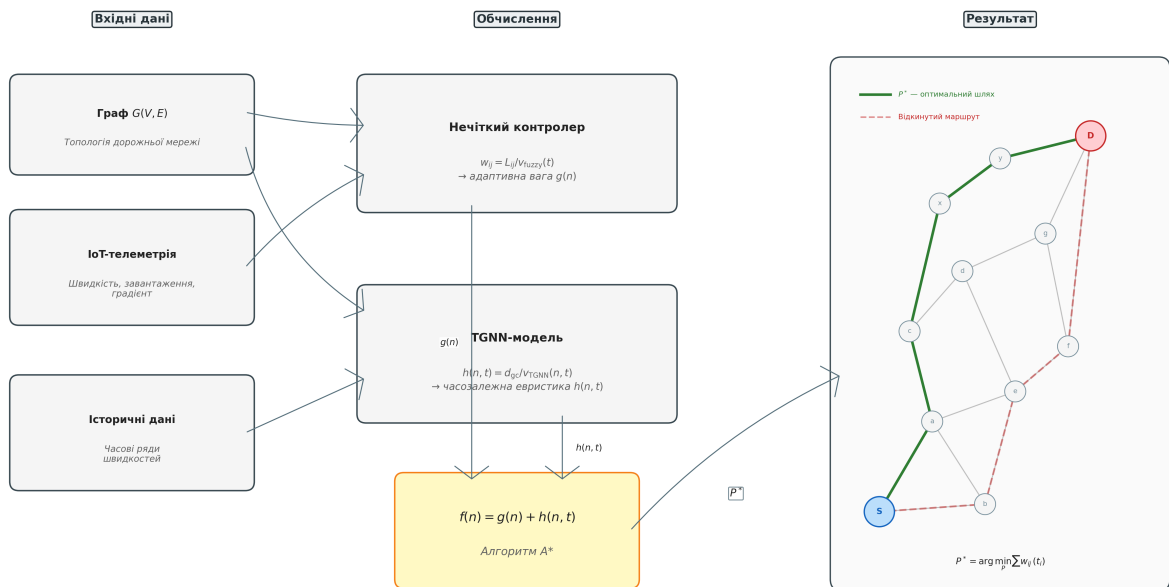


Рис. 3.6 - Структурна схема системи управління інформаційними потоками з пошуком оптимального шляху P^*

На рис. 3.6 зліва представлено вхідні дані системи: топологія мережі $G(V, E)$, потік IoT-повідомлень для нечіткого контролера та історичні часові ряди для моделі TGNN. Центральна частина відображає паралельне формування двох компонент алгоритму A^* : ваги w_{ij} (через нечіткий вивід) та евристики $h(n, t)$ (через TGNN). Справа візуалізовано результат на зваженому графі: червоним виділено геометрично найкоротший шлях, який проходить через зону критичного інформаційного навантаження; зеленим — оптимальний шлях P^* , знайдений за критерієм мінімізації $f(n) = g(n) + h(n, t)$, що оминає завантажені ділянки.

3.3.3 Вплив TGNN-евристики на простір пошуку

На рис. 3.7 наведено візуальне порівняння простору пошуку алгоритму A^* для двох евристик на дорожній мережі з виділеною зоною критичного інформаційного навантаження.

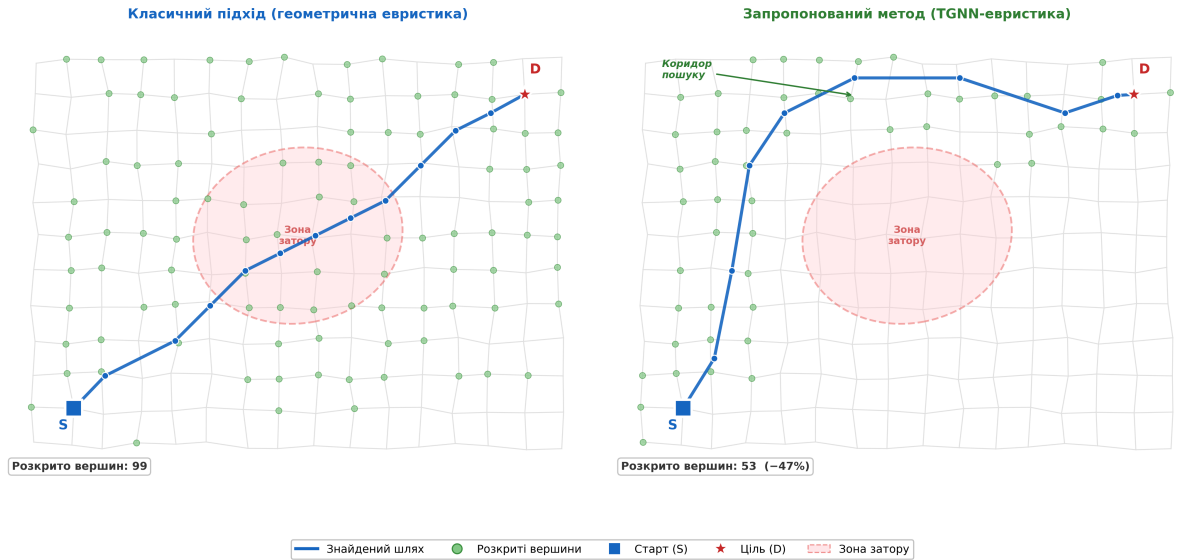


Рис. 3.7 - Вплив TGNN-евристики на топологію простору пошуку алгоритму A^* : ліворуч — класичний підхід із геометричною евристикою (широке «віяло» розкритих вершин), праворуч — запропонований метод із TGNN-евристикою (вузький коридор пошуку, що оминає зону критичного інформаційного навантаження)

У лівій частині рис. 3.7 показано пошук із геометричною евристикою $h_{\text{geo}}(n) = d_{\text{gc}}(n, D)/v_{\text{max}}$: зона розкритих вершин (зелені маркери) є широкою та нерівномірною, оскільки евристика спрямовує пошук уздовж найкоротшої прямої між початковою та цільовою точками, не враховуючи поточний стан $\lambda_{\text{seg}}(t)$. Алгоритм розкриває вершини по обидва боки від зони критичного інформаційного навантаження, марно витрачаючи обчислювальні ресурси на ділянки, що не лежать на оптимальному маршруті.

У правій частині — пошук із TGNN-евристикою $h_{\text{TGNN}}(n, t) = d_{\text{gc}}(n, D)/v_{\text{TGNN}}(n, t)$: завдяки прогнозуванню стану інформаційного навантаження на кожній ділянці, навколо зони критичного навантаження формується «потенціальний бар'єр» (високе значення h для вузлів з низьким прогнозованим значенням фізичного індикатора v_{TGNN}). Алгоритм розпізнає перешкоду на ранніх етапах пошуку і фокусується

на обхідних маршрутах, формуючи вузький спрямований **коридор пошуку**. Кількість розкритих вершин суттєво скорочується порівняно з класичним підходом.

TGNN-евристика змінює характер пошуку принципово. Класичні евристики [30] базуються на припущенні про рівномірність середовища. TGNN-евристика враховує просторово-часові закономірності [40, 41]: замість широкого «віяла» розкритих вершин формується коридор через ділянки з найнижчою очікуваною вартістю.

Результатом є:

- **скорочення простору пошуку** на 56–68% порівняно з алгоритмом Дейкстри [29], що безпосередньо зменшує час обчислення маршруту;
- **емпірично зафіксовану малу втрату оптимальності** — не більше 0,00–0,08% (0,67 секунди на 14-хвилинному маршруті);
- **узгодженість метрик**: і евристика, і фактична вартість руху виражені в одному часовому просторі, що усуває протиріччя класичного A^* , де евристика оцінює відстань, а вартість — час.

Висновки до розділу 3

1. Розроблено метод просторово-часового прогнозування інформаційного навантаження сегментів IoV-мережі на основі TGNN, який, на відміну від моделей без урахування топології, одночасно моделює просторові залежності між суміжними сегментами та часову еволюцію інформаційного навантаження через поєднання дифузійної згортки та рекурентного блоку GRU. Це дозволяє прогнозувати очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку з урахуванням топологічної структури мережі.
2. Показано, що прогнозування швидкості потоку як фізичного індикатора $\lambda_{\text{seg}}(t)$ з подальшим обчисленням евристики за формулою $h(n, t) = d_{\text{gc}}(n, D)/v_{\text{TGNN}}(n, t)$ структурно гарантує допустимість

евристики, оскільки $d_{gc} \leq d_{road}$, усуваючи проблему неузгодженості прогнозів, властиву підходам прямого прогнозування ЕТА.

3. Удосконалено метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі алгоритму A^* шляхом інтеграції часозалежної прогновної TGNN-евристики, яка оцінює очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку. Теоретично обґрунтовано, що така евристика формує «потенціальний бар'єр» навколо зон критичного навантаження та звужує простір пошуку, зберігаючи при цьому властивості оптимальності маршруту.

РОЗДІЛ 4

РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ СИСТЕМИ АДАПТИВНОЇ МАРШРУТИЗАЦІЇ ІНФОРМАЦІЙНИХ ПОТОКІВ В ІОВ-МЕРЕЖІ

У попередніх розділах було розроблено телекомунікаційну модель Іов-мережі (п. 2.1), метод адаптивної оцінки стану інформаційних потоків, метод просторово-часового прогнозування інформаційного навантаження та метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі модифікованого алгоритму A^* . Подальша верифікація адекватності цих моделей і методів потребує створення відповідного програмного забезпечення та відтворення умов функціонування Іов-мережі в режимі реального часу.

Специфіка предметної області Інтернету транспортних засобів (Іов) [1, 2] висуває жорсткі вимоги до архітектури системи керування інформаційними потоками: здатність обробляти високоінтенсивні гетерогенні потоки телеметрії, що надходять через канали V2X-комунікації, масштабованість, відмовостійкість та мінімальну затримку реакції. У зв'язку з цим необхідним етапом роботи є не лише алгоритмічна реалізація розроблених методів, але й вибір архітектурного патерну та технологічного стеку, здатного забезпечити функціонування сервісів у високонавантаженому середовищі.

Метою даного розділу є практична реалізація компонентів системи адаптивної маршрутизації інформаційних потоків у вигляді мікросервісної Іов-платформи та проведення експериментальних досліджень для підтвердження наукових гіпотез.

Для досягнення поставленої мети у розділі вирішуються наступні завдання:

1. Обґрунтування та розробка подійно-орієнтованої архітектури програмного комплексу.
2. Програмна реалізація ключових мікросервісів: шлюзу даних, підсистеми адаптивної кластеризації режимів інформаційного навантаження та сервісу динамічної маршрутизації.

3. Розробка середовища імітаційного моделювання транспортних потоків на основі реальної топології дорожньої мережі.
4. Проведення експериментальних досліджень для оцінки часових характеристик адаптації системи, точності виявлення режимів інформаційного навантаження та ефективності маршрутизації порівняно з існуючими аналогами.

4.1 Архітектура програмного комплексу IoT для керування інформаційними потоками

Практична реалізація запропонованих у розділах 2–3 методів адаптивної маршрутизації інформаційних потоків потребує побудови високопродуктивної телекомунікаційної програмної платформи, здатної функціонувати в умовах високої невизначеності та інтенсивності вхідних даних. Специфіка предметної області Інтернету транспортних засобів (IoV) [7, 8] висуває низку критичних вимог до програмної системи, що безпосередньо відповідають класичним вимогам до телекомунікаційних систем обробки трафіку:

- **підтримка високочастотних потоків** — тисячі транспортних засобів одночасно генерують телеметрію з інтервалом порядку секунди;
- **мінімальна затримка реакції** — адаптивна оцінка стану сегментів мережі має відбуватися за мілісекунди, щоб маршрутизація спиралась на актуальний стан інформаційного навантаження;
- **горизонтальне масштабування** — система має допускати збільшення кількості обробників без перепроєктування ядра;
- **відмовостійкість** — вихід з ладу окремого сервісу не повинен призводити до втрати даних чи зупинки обробки.

Монолітні архітектури не задовольняють ці вимоги через жорстку зв'язаність модулів та неможливість незалежного масштабування обчислювально інтенсивних компонентів (кластеризація, маршрутизація) окремо від легковагих (шлюз даних, моніторинг) [121, 122]. Тому за

- L_{commit} — мінімізовано шляхом зберігання графа в оперативній пам'яті у форматі CSR (Compressed Sparse Row), що усуває дискові операції.

Ефективність цих рішень підтверджено експериментально в п. 4.3: середня затримка оновлення ваги становить 1,19 мс, тоді як підхід на основі перерахунку з БД обробляє лише 27,2% повідомлень [47].

4.1.1 Обґрунтування вибору технологічного стеку та протоколів

Вибір конкретних програмних засобів обумовлений вимогами, визначеними вище. У табл. 4.1 наведено перелік ключових компонентів технологічного стеку та відповідність кожного з них архітектурній вимозі.

Таблиця 4.1 - Компоненти технологічного стеку програмного комплексу

Компонент	Призначення	Архітектурна роль
Apache Kafka	Шина подій, буферизація, розподіл потоків	Мінімізація L_{ingest} , горизонтальне масштабування
Protocol Buffers [124]	Бінарна серіалізація повідомлень	Мінімізація L_{net} , типізація контрактів
Python 3.10+ / FastAPI	Мікросервіси, API-шлюз	Інтеграція моделей машинного навчання (Scikit-learn, PyTorch)
Nginx	Балансування навантаження	Рівномірний розподіл запитів між реплікованими шлюзами
Docker / Docker Compose	Контейнеризація, оркестрація	Ізоляція середовищ, відтворюваність (15 контейнерів)
Prometheus + Grafana	Збір метрик, візуалізація	Спостережуваність: затримка, пропускна здатність, відставання
PostgreSQL	Довгострокове зберігання	Тренувальні дані для TGNN, еталонний обробник (перерахунок)

Нижче наведено обґрунтування вибору ключових компонентів.

1. Шина подій: Apache Kafka. В умовах пікових навантажень (наприклад, масовий інцидент, коли тисячі транспортних засобів одночасно генерують повідомлення про зупинку) синхронна архітектура призводить до перевантаження сервісів. Apache Kafka [52, 53, 100] виконує роль буфера зворотного тиску: повідомлення зберігаються в розділах (партиціях) журналу на диску брокера та обробляються сервісами у власному темпі. Кожне повідомлення отримує унікальне зміщення в межах розділу, що дозволяє відновити обробку після збою від останнього збереженого зміщення. Така модель забезпечує: (а) згладжування пікових

навантажень; (б) розв'язку між виробником та споживачем — шлюз даних не залежить від працездатності аналітичних сервісів; (в) горизонтальне масштабування — додавання нових обробників не вимагає зупинки системи. Схема внутрішньої організації Kafka наведена на рис. 4.2.

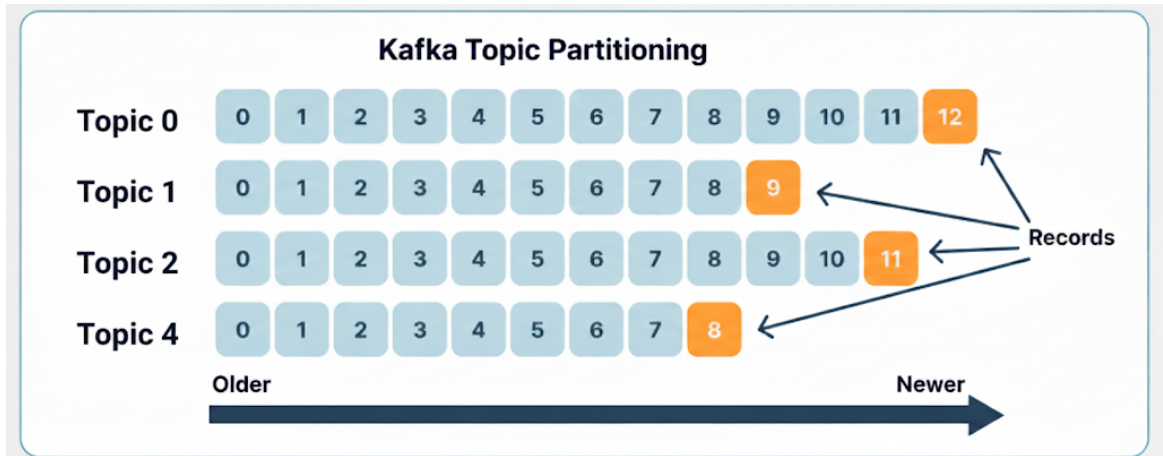


Рис. 4.2 - Організація потоків даних в Apache Kafka: виробник, розділи журналу та споживачі

2. Формат серіалізації: Protocol Buffers. У системах потокової обробки IoT-даних критичними є мінімальний розмір повідомлення та висока швидкість кодування. Текстові формати XML та JSON забезпечують зручність читання, проте мають значні накладні витрати. Для кодування корисного навантаження обрано бінарний формат Google Protocol Buffers (Protobuf v3) [124], що забезпечує: (а) зменшення розміру повідомлення у 3–5 разів порівняно з JSON; (б) сувору типізацію на рівні схеми (.proto-файли), що виключає помилки розбіжності форматів між мікросервісами. Порівняння форматів наведено в табл. 4.2.

Таблиця 4.2 - Порівняння форматів серіалізації даних

Характеристика	XML	JSON	Protobuf
Розмір повідомлення	Великий	Середній	Найменший (бінарне кодування)
Швидкість серіалізації	Низька	Середня	Висока
Читабельність	Висока	Висока	Низька
Обов'язковість схеми	Опціонально (XSD)	Опціонально	Обов'язкова (.proto)

3. Мова розробки: Python 3.10+ (FastAPI / asyncio). Вибір мови обумовлено двома чинниками. По-перше, бібліотека `asyncio` дозволяє шлюзу та сервісам-споживачам обробляти тисячі конкурентних операцій вводу-виводу в одному потоці, що є критичним для високопродуктивних IoT-систем. По-друге, оскільки ядро методу адаптивної оцінки стану інформаційних потоків (розділ 2) спирається на бібліотеки машинного навчання (`Scikit-learn` для кластеризації K-Means++, `PyTorch` для TGNN), Python забезпечує інтеграцію математичних моделей безпосередньо у код сервісів без міжмовних мостів.

4. Контейнеризація: Docker та Docker Compose. Програмний комплекс розгортається як набір із 15 Docker-контейнерів [123] однією командою. Контейнеризація забезпечує: (а) ізоляцію середовищ — кожен сервіс працює у власному контейнері з визначеними залежностями; (б) відтворюваність — конфігурація зафіксована в декларативному файлі; (в) горизонтальне масштабування — за потреби створюються додаткові репліки (наприклад, два шлюзи з балансувальником Nginx).

5. Моніторинг: Prometheus, Kafka Exporter та Grafana. Для збору метрик використовується Prometheus, який опитує кожен сервіс через HTTP-інтерфейс. Kafka Exporter забезпечує експорт специфічних метрик брокера, зокрема відставання обробки кожного споживача, що є ключовим показником здатності системи витримувати вхідний потік. Grafana візуалізує зібрані часові ряди у вигляді інтерактивних інформаційних панелей, що дозволяє оперативно контролювати пропускну

здатність, затримку та використання ресурсів.

4.1.2 Взаємодія компонентів системи

Архітектура програмного комплексу побудована за принципом конвеєрної обробки даних. Структурна схема на рис. 4.1 відображає чотири логічні рівні, через які послідовно проходить кожне повідомлення телеметрії.

Рівень 1: збір телеметрії. Бортові IoT-агенти (клієнтські додатки на транспортних засобах) генерують пакети телеметрії, що містять геопросторові координати, миттєву швидкість, ідентифікатор поточного ребра графа та мітку часу. Зазначимо, що значення v_k^{raw} , яке надходить від OBU транспортного засобу через канал V2X, є **фізичним індикатором стану інформаційного навантаження** $\lambda_{\text{seg}}(t)$ на відповідному сегменті IoV-мережі: зі зростанням щільності IoT-джерел $\rho_{\text{seg}}(t) \uparrow$ інтенсивність генерованих повідомлень $\lambda_{\text{seg}}(t) \uparrow$ збільшується, а швидкість руху $v_{uv}(t) \downarrow$ знижується. Таким чином, обробка потоку вимірів v_k^{raw} програмним комплексом є, по суті, оцінкою та управлінням інформаційним навантаженням сегментів IoV-мережі. У промисловій реалізації для каналу «Транспортний засіб — Хмара» (V2C) [1] рекомендовано протокол MQTT [125], оптимізований для нестабільних мобільних мереж завдяки бінарному заголовку (2 байти), механізмам якості обслуговування (QoS) та стійких сесій. В експериментальному середовищі (п. 4.3) телеметрію генерує асинхронний симулятор, що надсилає HTTP-запити через балансувальник Nginx до двох реплік API-шлюзу.

Рівень 2: нормалізація та транспорт. API-шлюз приймає вхідні повідомлення, десеріалізує бінарні дані, виконує первинну валідацію (перевірка діапазону координат, коректність ідентифікатора ребра) та публікує валідовані повідомлення у внутрішню шину подій Apache Kafka. Взаємодія між компонентами побудована за патерном Publish–Subscribe [126], де виробники та споживачі даних розв’язані через іменовані теми. В архітектурі виділено два ключові потоки (теми Kafka):

- **traffic-updates** — потік валідованої телеметрії, який читається усіма сервісами-споживачами (обробники ваг, сервіс кластеризації, модуль збереження історії);
- **fuzzy-rules** — керуючий потік, що містить оновлені параметри

нечітких правил (центри кластерів, дисперсії, коефіцієнти адаптації α_k), згенеровані аналітичним модулем кластеризації.

Крім того, окремий асинхронний споживач записує телеметрію до бази даних PostgreSQL для довгострокового зберігання історичних даних, необхідних для навчання нейронних мереж. Запис відбувається незалежно від основної обробки, що виключає вплив дискових операцій на затримку.

Рівень 3: аналітика та адаптація. Цей рівень реалізує логіку методу адаптивної оцінки стану інформаційних потоків графа IoV-мережі (розділ 2). Сервіс-споживач підписаний на тему traffic-updates і виконує пакетно-інкрементальну кластеризацію: накопичує вектори ознак (η, L, τ) , виконує алгоритм K-Means++ для ідентифікації поточних режимів руху та обчислює параметри кластерів. Результат — повідомлення типу FuzzyRule з оновленими центрами μ_k , дисперсіями σ_k та коефіцієнтами α_k — публікується в тему fuzzy-rules. Фоновий процес навчання TGNN-моделі періодично обирає історичні дані з PostgreSQL, проводить навчання та оновлює файл ваг, доступний для сервісу маршрутизації.

Рівень 4: динамічна маршрутизація. Ключовий компонент системи, що реалізує принцип утримання стану в оперативній пам'яті. Граф дорожньої мережі зберігається у форматі CSR (Compressed Sparse Row) в оперативній пам'яті, що забезпечує доступ до будь-якого ребра за $O(1)$. Архітектурною особливістю є механізм **подвійної підписки**:

- **Підписка на дані** (тема traffic-updates): сервіс отримує фактичні параметри ребра (миттєва швидкість V_{curr} , кількість авто N) та оновлює стан графа;
- **Підписка на знання** (тема fuzzy-rules): сервіс отримує оновлені коефіцієнти адаптації α та параметри кластерів (μ_k, σ_k) для розрахунку ваг ребер.

Завдяки такій організації сервіс маршрутизації миттєво реагує на зміни: при надходженні повідомлення про зниження швидкості на ребрі сервіс застосовує актуальне нечітке правило, перераховує вагу w_{ij} і

вже наступний запит на побудову маршруту враховує нову ситуацію. Уся обробка відбувається без звернень до бази даних, що забезпечує мінімальну затримку.

4.1.3 Структури повідомлень для обміну даними

Для забезпечення високої продуктивності та мінімізації трафіку стандартом обміну даними між компонентами системи обрано Protocol Buffers v3. Нижче описано три ключові типи повідомлень.

1. Пакет вхідної телеметрії (TelemetryPacket). Повідомлення, що генерується бортовим пристроєм та передається до API-шлюзу. Містить мінімально необхідний набір даних для ідентифікації руху: унікальний ідентифікатор пристрою, координати (широта, довгота), миттєву швидкість та ідентифікатор ребра графа. Структура повідомлення наведена на рис. 4.3.

```
message TelemetryPacket {
  string device_id = 1;
  double latitude = 2;
  double longitude = 3;
  float speed_mph = 4;
  int64 edge_id = 5;
}
```

Рис. 4.3 - Структура повідомлення TelemetryPacket (Protobuf v3)

2. Стан ребра графа (EdgeState). Повідомлення внутрішньої шини (тема traffic-updates), що формується API-шлюзом після валідації та агрегації вхідних даних. Містить агреговані показники: ефективність руху η , коефіцієнт завантаженості L та тренд швидкості τ . Структура повідомлення наведена на рис. 4.4.

```

message EdgeState {
  int64 edge_id = 1;
  float speed_efficiency = 2;
  float load_factor = 3;
  float traffic_trend = 4;
  double timestamp = 5;
}

```

Рис. 4.4 - Структура повідомлення EdgeState (тема traffic-updates)

3. Нечітке правило (FuzzyRule). Керуюче повідомлення (тема fuzzy-rules), що генерується сервісом кластеризації. На відміну від бінарного сигналу «затор/не затор», це повідомлення передає повну математичну модель розподілу станів: центри кластерів μ_k , дисперсії σ_k та обчислені коефіцієнти адаптації α_k . Оскільки кластеризація відбувається у тривимірному просторі ознак (η, L, τ) , параметри передаються як масиви (repeated-поля у Protobuf). Структура повідомлення наведена на рис. 4.5.

```

message FuzzyRule {
  int64 edge_id = 1;
  int32 k = 2;
  repeated float centers = 3;
  repeated float sigmas = 4;
  repeated float alphas = 5;
  repeated float weights = 6;
}

```

Рис. 4.5 - Структура повідомлення FuzzyRule (тема fuzzy-rules)

4. Модель подвійного споживання даних. Сервіс маршрутизації реалізує логіку злиття даних з двох потоків для розрахунку динамічної ваги w_{ij} :

- **Площина керування:** при надходженні FuzzyRule сервіс оновлює в пам'яті вектори центрів μ_k , дисперсій σ_k та коефіцієнтів α_k . Поле типу **repeated float** (масив дійсних чисел) для дисперсій дозволяє коректно обчислити відстань Махаланобіса [58] з урахуванням різного розкиду даних для різних кластерів;

- **Площина даних:** при надходженні EdgeState сервіс підставляє поточні значення (η, L, τ) у формулу нечіткого висновку Такагі–Сугено [60, 66] з використанням закешованих параметрів, отримуючи адаптивний коефіцієнт α для оновлення ваги графа.

Такий підхід дозволяє передавати складні багатовимірні моделі через компактні бінарні повідомлення розміром порядку сотень байтів, забезпечуючи оновлення бази знань контролера без перекомпіляції та перезапуску сервісу.

4.2 Програмна реалізація ключових мікросервісів

У попередньому підрозділі було описано загальну архітектуру програмного комплексу та потоки даних між компонентами. У цьому підрозділі розглянуто деталі програмної реалізації трьох ключових сервісів, що втілюють ядро системи адаптивної маршрутизації інформаційних потоків: сервіс кластеризації режимів руху, сервіс маршрутизації та підсистему навчання моделі TGNN.

Усі сервіси реалізовано мовою Python 3.10 з використанням парадигми асинхронного програмування (`asyncio`). Взаємодія між компонентами здійснюється через обмін повідомленнями в Apache Kafka відповідно до структур даних, визначених у п. 4.1.3.

4.2.1 Реалізація сервісу кластеризації режимів руху

Сервіс кластеризації є програмною реалізацією методу автоматичної побудови бази нечітких правил (п. 2.1) [45, 46]. Його призначення — циклічна обробка вхідного потоку векторів ознак (η, L, τ) та публікація оновлених параметрів кластерів $(\mu_k, \sigma_k, \alpha_k)$ у тему `fuzzy-rules`.

Програмно сервіс реалізовано як споживач Kafka, що функціонує за стратегією пакетно-інкрементального навчання. Алгоритмічна структура роботи наведена у вигляді псевдокоду на рис. 4.6.

Вхід: потік повідомлень `EdgeState` з теми `traffic-updates`

Вихід: оновлені правила $(\mu_k, \sigma_k, \alpha_k)$ у темі `fuzzy-rules`

```

1:  $buf \leftarrow \emptyset$ 
2: while true do
3:    $m \leftarrow \text{CONSUME}(\text{traffic-updates})$ 
4:    $buf \leftarrow buf \cup \{(m.\eta, m.L, m.\tau)\}$ 
5:   if  $|buf| \geq B_{\min}$  then
6:      $X \leftarrow [buf]_{N \times 3}$  ▷ Етап 1
7:      $k^* \leftarrow \text{ELBOW}(X, k_{\max}=5)$ 
8:      $\{C_i, \ell_i\} \leftarrow \text{KMEANS++}(X, k^*)$  ▷ Етап 2
9:     for  $i = 1 \dots k^*$  do ▷ Етап 3
10:        $\mu_i, \sigma_i \leftarrow \text{mean}(X_{\ell=i}), \text{std}(X_{\ell=i})$ 
11:        $j^* \leftarrow \arg \min_j d_{\text{Mah}}(\mu_i, G_j)$ 
12:       if  $\text{BC}(\mu_i, \sigma_i, G_{j^*}) > \theta_{\text{BC}}$  then
13:          $\text{MERGE}(G_{j^*}, \mu_i, \sigma_i)$ 
14:       else
15:          $G \leftarrow G \cup \{(\mu_i, \sigma_i, \alpha_i)\}$ 
16:       end if
17:     end for
18:      $G \leftarrow \{g \in G \mid \text{age}(g) < t_{\max}\}$  ▷ Етап 4
19:      $\text{PUBLISH}(\text{fuzzy-rules}, G)$  ▷ Етап 5
20:      $buf \leftarrow \emptyset$ 
21:   end if
22: end while

```

Рис. 4.6 - Псевдокод алгоритму пакетно-інкрементальної кластеризації режимів руху

Логіка роботи базується на безперервному циклі з п'яти етапів:

Етап 1: накопичення пакету. Споживач зчитує повідомлення з теми `traffic-updates` та десеріалізує поля `speed_efficiency`, `load_factor`, `traffic_trend` у матрицю $[N \times 3]$ (бібліотека NumPy). Розмір пакету обрано рівним 414 повідомлень (207 сенсорів $\times$ 2 часові кроки), що забезпечує оновлення кластерів кожні ≈ 12 с при 50-кратному прискоренні симуляції.

Етап 2: мікрокластеризація. До зібраного пакету застосовується алгоритм K-Means++ [38, 57, 68, 69] з автоматичним визначенням оптимальної кількості кластерів $k \in [1..5]$ методом геометричного «ліктя» (п. 2.1.1). Операції нормалізації та розрахунку відстаней виконуються векторизовано засобами NumPy, що дозволяє уникнути повільних циклів.

Етап 3: злиття з глобальною моделлю. Нові мікрокластери інтегруються з існуючою базою знань за коефіцієнтом Бхаттачар'ї [59, 73] (п. 2.1.2). Якщо перекриття перевищує порогове значення ($BC > 0,5$), кластери зливаються за формулою зваженого оновлення. Це забезпечує роботу з константним використанням пам'яті $O(K)$, де K — кількість активних режимів.

Етап 4: очищення застарілих кластерів. Режим руху, що не підтверджується новими даними протягом 600 с (конфігурується), вилучаються з пам'яті та перестають впливати на маршрутизацію.

Етап 5: публікація правил. Оновлені параметри $(\mu_k, \sigma_k, \alpha_k)$ серіалізуються у повідомлення FuzzyRule (п. 4.1.3) та публікуються в темі fuzzy-rules. Усі підписані сервіси маршрутизації отримують оновлення без перезавпуску.

4.2.2 Реалізація сервісу маршрутизації

Сервіс маршрутизації є найбільш критичним компонентом системи з точки зору затримки, оскільки саме він обробляє запити на побудову маршруту. Програмно сервіс реалізовано як мікросервіс із графом дорожньої мережі у форматі CSR в оперативній пам'яті.

Зберігання графа у форматі CSR. Як зазначено в п. 4.1.2, граф дорожньої мережі зберігається в оперативній пам'яті у форматі CSR. Повна топологія завантажується при старті процесу. Оскільки дорожні графи є сильно розрідженими (вершини мають лише 2–4 сусідів), CSR-формат забезпечує:

- мінімізацію обсягу пам'яті — наприклад, для графа масштабу Києва ($\sim 200\,000$ вузлів, $\sim 400\,000$ ребер) CSR-формат потребує $\approx 5\text{--}6$ МБ проти $\approx 25\text{--}30$ МБ для хеш-таблиці;
- покращення локальності даних — послідовне розташування елементів дозволяє ефективно використовувати кеш-пам'ять процесора під

час обходу алгоритмом A^* ;

- оновлення ваги за $O(1)$ — зміна значення в масиві ваг не потребує перебудови структури.

Оновлення ваги виконується за $O(1)$: за допомогою масиву `row_ptr` визначається діапазон індексів сусідів вершини-джерела, знаходиться позиція цільової вершини в `col_idx` та замінюється відповідне значення в масиві ваг.

Механізм подвійної підписки. Як описано в п. 4.1.2, сервіс одночасно підписаний на дві теми Kafka. При надходженні повідомлення з теми `traffic-updates` виконується наступна послідовність обробки:

1. Скалярні поля повідомлення (η, L, τ) об'єднуються у вектор ознак $\vec{x}$.
2. Викликається процедура нечіткого висновку: вектор $\vec{x}$ зіставляється з антецедентами усіх активних правил через відстань Махаланобіса [58], ступені належності агрегуються методом Такагі–Сугено [60, 66] нульового порядку (формула (2.7)) для отримання адаптивного коефіцієнта α .
3. Швидкість ребра оновлюється за формулою експоненційного згладжування [27] (формула (1.3)): $\hat{S}_t = \alpha \cdot S_t + (1 - \alpha) \cdot \hat{S}_{t-1}$.
4. Вага ребра перераховується як час обслуговування інформаційного потоку на сегменті $w_{ij} = l_{ij} / \hat{S}_t$ та записується в CSR-масив.

Деталізацію обчислення коефіцієнта α наведено на рис. 4.7.

Вхід: вектор ознак $\vec{x} = (\eta, L, \tau)$, правила $G = \{(\mu_k, \sigma_k, \alpha_k)\}_{k=1}^K$

Вихід: коефіцієнт адаптації $\alpha \in [0,05; 0,95]$

```

1: if  $K = 0$  then
2:   return 0,9 ▷ невідомий стан
3: end if
4: for  $k = 1 \dots K$  do
5:    $d_k^2 \leftarrow \sum_{j=1}^3 \frac{(x_j - \mu_{k,j})^2 \cdot w_j^2}{\sigma_{k,j}^2 + \varepsilon}$  ▷ Махаланобіс
6:    $\phi_k \leftarrow \exp(-0,5 \cdot d_k^2)$  ▷ ступінь належності
7: end for
8: if  $\sum_k \phi_k < \varepsilon$  then
9:   return 0,9
10: end if
11:  $\alpha \leftarrow \frac{\sum_{k=1}^K \phi_k \cdot \alpha_k}{\sum_{k=1}^K \phi_k}$  ▷ Такагі–Сугено
12: return  $\alpha$ 

```

Рис. 4.7 - Псевдокод обчислення коефіцієнта адаптації α за методом нечіткого висновку Такагі–Сугено

Якщо сума ступенів належності усіх правил прямує до нуля (вектор $\vec{x}$ не покривається жодним кластером), система повертає консервативну оцінку $\alpha = 0,9$. Значення 0,9 обрано як консервативне: у разі невідомої ситуації система максимально довіряє новому вимірюванню, що гарантує швидку реакцію на потенційно аномальну ситуацію.

Модифікований алгоритм A^* з TGNN-евристикою. Для побудови маршруту за запитом користувача реалізовано часозалежний алгоритм A^* [30] (п. 3.3), в якому евристична функція (формула (3.4)):

$$h(n, t) = \frac{d_{gc}(n, \text{target})}{v_{TGNN}(n, t)}$$

формується на основі прогнозів часо-просторової графової нейронної мережі [31]. Псевдокод реалізації наведено на рис. 4.8.

Вхід: граф $G = (V, E)$ з вагами w ; вершини s, t ; евристика h ; час t_0

Вихід: оптимальний маршрут з мінімальним сумарним навантаженням на сегменти

```

1:  $g[s] \leftarrow 0$ ;  $t_{\text{arr}}[s] \leftarrow t_0$ 
2:  $Open \leftarrow \{(f=h(s, t, t_0), s)\}$  ▷ пріоритетна черга
3:  $Closed \leftarrow \emptyset$ 
4: while  $Open \neq \emptyset$  do
5:    $(f_c, n) \leftarrow \text{EXTRACTMIN}(Open)$ 
6:   if  $n \in Closed$  then continue
7:   end if
8:    $Closed \leftarrow Closed \cup \{n\}$ 
9:   if  $n = t$  then return  $\text{RECONSTRUCTPATH}(n), g[n]$ 
10:  end if
11:  for  $(n, m) \in E$  do
12:     $\Delta t \leftarrow w_{nm} / \hat{S}_{nm}$  ▷ час обслуговування потоку на сегменті
13:     $g' \leftarrow g[n] + \Delta t$ 
14:    if  $g' < g[m]$  then
15:       $g[m] \leftarrow g'$ ;  $t_{\text{arr}}[m] \leftarrow t_{\text{arr}}[n] + \Delta t$ 
16:       $f \leftarrow g' + h(m, t, t_{\text{arr}}[m])$  ▷  $h_{\text{TGNN}}$ 
17:       $\text{INSERT}(Open, (f, m))$ 
18:    end if
19:  end for
20: end while
21: return  $\emptyset$  ▷ шлях не знайдено

```

Рис. 4.8 - Псевдокод алгоритму A^* з адаптивними вагами та TGNN-евристикою

Прямий виклик нейронної мережі всередині ітераційного циклу A^* є обчислювально неефективним, оскільки кількість досліджених вершин може сягати тисяч. Для забезпечення затримки відповіді менше 50 мс реалізовано стратегію попереднього розрахунку прогнозів: перед початком пошуку система генерує тензор швидкостей для всіх вузлів на горизонт планування та зберігає його як таблицю пошуку в оперативній пам'яті. Це зводить складність отримання прогнозу всередині евристичної функції з

$O(N_{\text{layers}} \cdot D_{\text{model}})$ до $O(1)$ — простого доступу за індексом.

Оскільки модель TGNN видає прогнози з дискретним кроком ($\Delta t = 5$ хв), для отримання значення швидкості у довільний момент часу t використано кусково-сталу апроксимацію: всередині одного 5-хвилинного інтервалу застосовується одне й те саме прогнозоване значення швидкості відповідно до формули (3.6).

4.2.3 Підсистема навчання моделі TGNN

Підсистема навчання відповідає за актуалізацію параметрів моделі TGNN на нових історичних даних. Оскільки дорожні закономірності є нестационарними (сезонність, зміни організації руху), модель потребує періодичного перенавчання. Програмно модуль реалізовано з використанням фреймворку PyTorch.

Архітектура моделі. Детальний опис архітектури TGNN наведено в п. 3.1.1. Програмна реалізація складається з трьох блоків:

- **просторовий блок** — два шари дифузійної згортки (DiffusionConv [39], $K = 2$ кроки випадкового блукання) з залишковим з'єднанням та нормалізацією шару. Перший шар агрегує ознаки безпосередніх сусідів, другий — сусідів другого порядку, розширюючи рецептивне поле моделі;
- **часовий блок** — двошарова комірка GRU [63] з нормалізацією шару [115], що обробляє послідовність $T = 12$ часових кроків просторових ознак та формує прихований стан h_t ;
- **прогнозний блок** — дворівневий повнозв'язний шар, що трансформує прихований стан GRU у прогноз нормалізованої швидкості для наступного часового кроку.

Загальний обсяг моделі становить 205 K параметрів, що на 32% менше за DCRNN [39] (300 K+) за порівнянної точності.

Процедура навчання. Навчання виконується на історичних даних з PostgreSQL за допомогою оптимізатора Adam ($\text{lr} = 10^{-3}$, $\text{wd} = 10^{-5}$) зі стратегією WarmupCosine (5 епох розігріву, 80 епох загалом) та раннім зупиненням ($P = 15$ епох). Функцією втрат є MSE; в ході експериментів

(п. 3.2.1) встановлено, що $\lambda_{tri} = 0$ є оптимальним, оскільки DiffusionConv неявно забезпечує просторову узгодженість прогнозів.

Вибір гіперпараметрів. Конфігурацію моделі обрано емпіричним шляхом із урахуванням компромісу між точністю та швидкодією в режимі реального часу. Ключові параметри наведено в табл. 4.3.

Таблиця 4.3 - Гіперпараметри моделі TGNN та обґрунтування їх вибору

Параметр	Значення	Обґрунтування
Просторовий прихований шар	64	Баланс між ємністю кодування та кількістю параметрів
Часовий прихований шар (GRU)	128	Достатній для захоплення добових циклів при $T = 12$
Кількість шарів GRU	2	Глибша обробка часових залежностей без надмірного зростання параметрів
Кроки дифузії (K)	2	Покриття сусідів другого порядку (вплив заторів через 2 перехрестя)
Швидкість навчання	10^{-3}	Початкове значення з косинусним згасанням до 10^{-5}
Кількість епох	80	Рання зупинка при відсутності покращення протягом 15 епох

Аналіз процесу навчання. На рис. 4.9 та рис. 4.10 наведено динаміку навчання моделі. Функція втрат (MSE) демонструє стрімке падіння з **1389,87** (епоха 0) до **7,34** (епоха 80), що свідчить про ефективне виявлення просторово-часових залежностей архітектурою з двома шарами дифузійної згортки. Валідаційна помилка (MAE) стабілізується після 40-ї епохи на рівні **2,10 mph**, що на **24%** краще за еталонну модель DCRNN [39] (MAE 2,77 mph). При цьому модель має лише **205 тис. параметрів** (проти понад 300 тис. у DCRNN), що підтверджує ефективність обраної архітектури.

Скорочення внутрішньої невизначеності протягом навчання проілюстровано на рис. 4.9–4.10: крива втрат монотонно спадає з узгодженістю навчальної та валідаційної вибірок, а валідаційні метрики стабілізуються

після 40-ї епохи без ознак перенавчання.

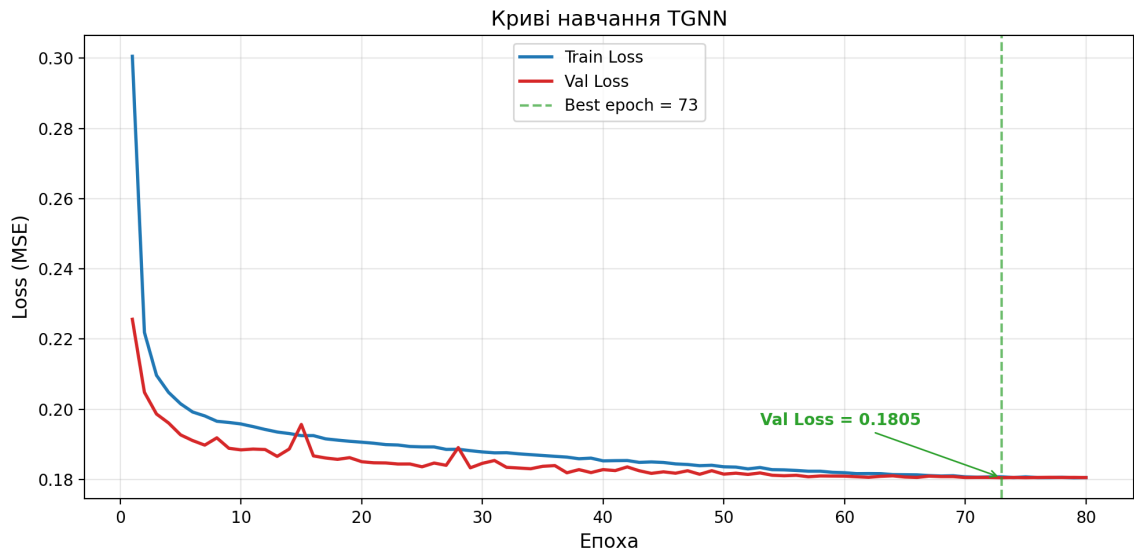


Рис. 4.9 - Криві навчання моделі TGNN: функція втрат на навчальній та валідаційній вибірках

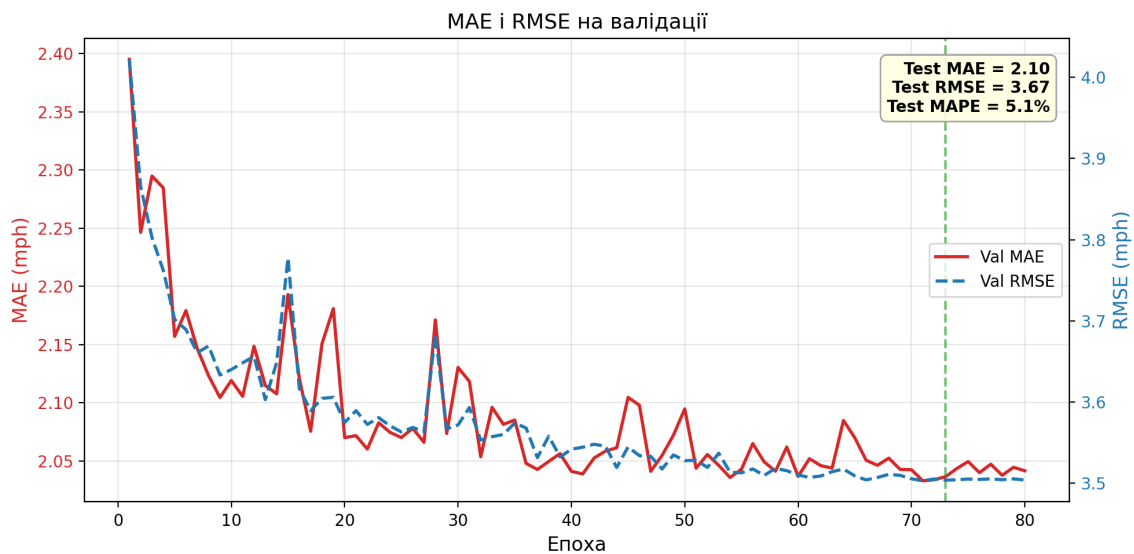


Рис. 4.10 - Валідаційні метрики TGNN (MAE та RMSE) протягом навчання

Відповідно, модель є одночасно точною (MAE 2,10 mph) та обчислювально ефективною (205 тис. параметрів), що є необхідною умовою для коректної роботи евристики A^* у режимі реального часу.

Рекомендації щодо перенавчання моделі при промисловому впровадженні. Як зазначено в п. 1.3.2, транспортний потік є нестаціонарним процесом, якому притаманний дрейф концепції [10]: статистичні

характеристики швидкостей змінюються внаслідок сезонних факторів, змін інфраструктури (ремонт доріг, нові розв'язки, зміна схем організації руху) та довгострокових тенденцій (зростання автомобілізації). Це означає, що модель TGNN, навчена на історичних даних одного періоду, з часом втрачатиме точність прогнозування. Для підтримання якості маршрутизації на прийнятному рівні рекомендується передбачити механізм періодичного перенавчання моделі.

Перенавчання доцільно ініціювати за одним із трьох тригерів:

1. *моніторинг якості прогнозу* — підсистема маршрутизації може безперервно обчислювати МАЕ між прогнозованими та фактичними швидкостями на ковзному вікні останніх 24 годин; коли МАЕ перевищить встановлений поріг (наприклад, 2,5 mph для METR-LA), генерується сигнал на перенавчання;
2. *календарний тригер* — планове перенавчання виконується з фіксованою періодичністю (рекомендовано кожні 4–8 тижнів), що забезпечує адаптацію до сезонних змін навіть за відсутності різкого зростання помилки;
3. *подія зміни інфраструктури* — оновлення топології графа (додавання або вилучення ребер) потребує повного перенавчання, оскільки структура суміжності є входом дифузійної згортки.

Для скорочення тривалості перенавчання рекомендується стратегія ініціалізації з попередніх ваг (*warm start*): нова модель ініціалізується вагами попередньої версії та дотреновувється протягом 20–30 епох (замість 80 при навчанні з нуля). Навчальну вибірку доцільно формувати методом ковзного вікна з використанням даних останніх 6–8 тижнів, що дозволяє моделі «забувати» застарілі патерни та концентруватися на актуальній динаміці.

Для забезпечення безперебійної роботи системи під час перенавчання рекомендується механізм гарячої заміни ваг: перенавчання виконується на окремому обчислювальному вузлі, а нова версія моделі публікується у вигляді версіонованого файлу контрольної точки. Підсистема маршрутизації атомарно завантажує нові ваги без зупинки

обслуговування запитів. Такий підхід забезпечує нульовий час простою та дає змогу за потреби повернутися до попередньої версії моделі, якщо нова демонструє гіршу якість на валідаційній вибірці. Архітектурні передумови для цього вже закладено: мікросервісна побудова системи (п. ??) дозволяє оновлювати підсистему навчання незалежно від підсистеми маршрутизації.

4.3 Експериментальне дослідження ефективності системи

У попередніх розділах розроблено два ключові складники системи адаптивної маршрутизації інформаційних потоків: метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі (розділ 2) [45, 47] та метод пошуку оптимального шляху на основі модифікованого алгоритму A^* з прогнозом TGNN (розділ 3). Метою цього підрозділу є експериментальна верифікація ефективності обох складників та інтегрованої системи на імітаційних та еталонних даних.

4.3.1 Постановка задачі та система метрик

Для цілісної оцінки розробленої системи адаптивної маршрутизації інформаційних потоків визначено три групи метрик, що відповідають трьом аспектам ефективності системи.

Метрики алгоритмічної ефективності. Ця група оцінює якість модифікованого алгоритму пошуку (часозалежний A^* з TGNN-евристикою, п. 3.1.2). Основним показником є **кількість досліджених вершин** N_{expanded} — кількість вузлів графа, вилучених з черги пріоритетів та оброблених алгоритмом до моменту досягнення цільового вузла. Цей показник безпосередньо відображає якість евристичної функції $h(n, t)$: чим точніше евристика оцінює перспективність напрямку, тим менше зайвих вузлів перевіряє алгоритм, що прямо корелює зі зменшенням навантаження на процесор при масових запитах.

Метрика часу в дорозі T_{travel} не використовується як основний критерій, оскільки усі допустимі евристики знаходять маршрут однакової тривалості, проте витрачають різну кількість обчислювальних ресурсів.

Метрики якості адаптації. Ця група оцінює динаміку оновлення

ваг графа — здатність системи виявляти перехідні процеси (моменти зміни режиму інформаційного навантаження, наприклад, перехід від вільного руху до затору). Основним показником є **час реакції** T_{reaction} — часова затримка між подією різкої зміни швидкості та моментом, коли система оновила вагу ребра графа до відповідного рівня.

Додатково оцінюються: середня абсолютна помилка (MAE), середньоквадратична помилка (RMSE), середня відсоткова помилка (MAPE), втрата оптимальності маршруту (відхилення маршруту від оптимального внаслідок похибки оцінки ваг), стабільність маршрутів (кількість безпідставних змін маршруту) та шумостійкість (зростання MAE при наявності сенсорного шуму). Одиниці MAE та RMSE відповідають вхідним даним датасету — милям на годину (mph), де швидкість $v_{\text{seg}}(t)$ є фізичним індикатором стану інформаційного навантаження сегмента IoV-мережі.

Системні метрики продуктивності. Для оцінки інженерної ефективності архітектури використовуються показники, зібрані системою моніторингу Prometheus/Grafana під час навантажувального тестування:

- **Пропускна здатність** — кількість повідомлень телеметрії, оброблених за секунду.
- **Затримка обробки** T_{proc} — середній час повного циклу обробки одного повідомлення: від зчитування з черги до оновлення CSR-графа в пам'яті.
- **Відставання споживача** — кількість повідомлень у черзі, що очікують обробки. Стабільне нульове відставання є індикатором роботи в режимі реального часу.
- **Утилізація ресурсів** — споживання процесора та оперативної пам'яті.

4.3.2 Програмно-апаратний стенд та імітаційна модель

Проведення натурних експериментів на реальній транспортній мережі пов'язане з організаційними складнощами та ризиками для безпеки руху. Тому для верифікації розроблених методів спроектовано два типи

експериментальних стендів: імітаційну модель міської мережі та конвеєр відтворення еталонних даних.

Апаратне забезпечення. Усі експерименти проведено на обладнанні з фіксованою конфігурацією (табл. 4.4).

Таблиця 4.4 - Технічні характеристики випробувального стенду

Компонент	Специфікація
Обладнання	Dell Latitude 5420
Процесор	Intel Core i7-1185G7 (4 ядра, 8 потоків, 3,0–4,8 ГГц)
Оперативна пам'ять	32 ГБ DDR4
Накопичувач	Samsung 970 EVO Plus NVMe SSD
Операційна система	Windows 11 Pro
Контейнеризація	Docker Desktop 27.5.1
Мова програмування	Python 3.13

Використання обладнання споживчого класу (ноутбук) дозволяє підтвердити придатність системи для розгортання на малоресурсних серверних вузлах.

Імітаційна модель міської мережі (м. Ірпінь). Для первинної верифікації адаптивного методу розроблено імітаційну модель на основі дорожнього графа міста Ірпінь (Київська обл.), експортованого з OpenStreetMap (~2 400 вузлів, ~5 600 ребер). Генератор трафіку емулює рух тисяч агентів зі стохастичними характеристиками:

- поява нових транспортних засобів підпорядковується розподілу Пуассона;
- інтенсивність змінюється у часі за кривою ранкового піку (06:00–11:00, максимум до 3 000 подій/с);
- кожен агент має точку старту, точку призначення, бажану швидкість та стохастичний поведінковий фактор.

Для перевірки адаптивних механізмів реалізовано механізм ін'єкції аномалій, що дозволяє програмно змінювати характеристики ребер

графа. Реалізовано три типи впливів: *вільний потік* (калібрування базової евристики), *зниження пропускної здатності* (імітація години пік, 5–10 км/год) та *раптове блокування ребра* (імітація ДТП, швидкість $\rightarrow 0$ за 1 хвилину).

Випробувальний стенд розгорнуто у середовищі Docker (15 контейнерів): генератор трафіку, кластер Apache Kafka (3 брокери, топик з партиціюванням за географічними зонами), екземпляри сервісів маршрутизації та кластеризації, система моніторингу Prometheus/Grafana. Архітектуру стенду описано в п. 4.1.

Конвеєр відтворення еталонних даних METR-LA. Для порівняльного аналізу з промисловими та академічними аналогами використано еталонний набір даних METR-LA [39] — показники швидкості з 207 індуктивних петлевих сенсорів [11] мережі швидкісних автострад Лос-Анджелеса. Дані подаються у систему через Apache Kafka з інтервалом опитування 10 с. Загалом виконано 370 опитувань (124 442 еталонних порівняння «оцінка–факт»). Для моделювання реальних умов до вхідних даних додано гаусів сенсорний шум ($\sigma = 3$ mph).

Мережа METR-LA формує зважений граф із 207 вузлів та 1515 ребер. Для порівняння реалізовано 8 незалежних обробників телеметрії, кожен з яких отримує ідентичний вхідний потік (табл. 4.5).

Таблиця 4.5 - Обробники телеметрії для порівняльного аналізу на METR-LA

№	Обробник	Аналог
1	Фільтр Калмана [92] з адаптивним шумом процесу	HERE [54] / TomTom
2	Експоненціальне ковзне середнє [27] ($\alpha = 0,7$)	ЕМА (фіксований)
3	Прямий прогноз TGNN як оцінка швидкості	Google Maps [56]
4	Баєсівське злиття нечіткої оцінки та фільтра Калмана	Нечіткий-Калман
5	Метод адаптивної оцінки стану інформаційних потоків	Запропонований
6	Повний перерахунок з бази даних	Перерахунок (БД)
7	Баєсівське злиття спостережень від розп. джерел	Waze [55]
8	Усереднення у буфері фіксованого розміру	Ковзне вікно

Кожен обробник незалежно споживає потік з Kafka, оновлює ваги графа за власним алгоритмом та надає оцінки швидкості для маршрутизації, що забезпечує репрезентативне порівняння в ідентичних умовах вхідного потоку та відповідає підходу до аналізу серій однотипних експериментів [113].

4.3.3 Результати моделювання та аналіз сценаріїв

Для підтвердження ефективності розроблених методів адаптивної маршрутизації інформаційних потоків проведено серію експериментів за чотирма сценаріями: скорочення простору пошуку (Ірпінь та METR-LA), час реакції на інциденти (Ірпінь), порівняльна оцінка 8 методів оновлення ваг (METR-LA) та системна продуктивність під піковим навантаженням (Ірпінь).

Сценарій 1: Скорочення простору пошуку.

У цьому сценарії порівнюється кількість досліджених вершин N_{expanded} для алгоритму Дейкстри [29] (еталон, $h = 0$) та часозалежного A^* з різними евристичними функціями, включно із запропонованою TGNN-евристикою (п. 3.1.2).

Граф м. Ірпінь. На тестовому маршруті (9227166194 $\rightarrow$ 4804723422) в умовах ускладненого руху порівняно 7 алгоритмів. Візуалізацію простору пошуку наведено на рис. 4.11.

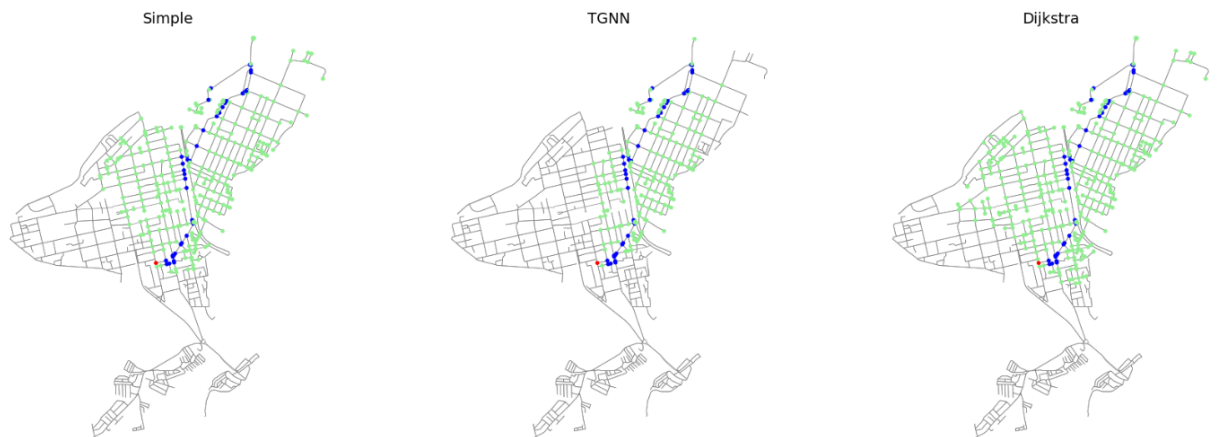


Рис. 4.11 - Порівняння простору пошуку алгоритмів маршрутизації на графі м. Ірпінь

Як видно з рис. 4.11, алгоритм Дейкстри досліджує граф рівномірно у всіх напрямках (452 вершини), тоді як TGNN-евристика [31] спрямовує пошук уздовж оптимального напрямку (263 вершини). Кількісні результати для всіх евристик наведено в табл. 4.6.

Таблиця 4.6 - Порівняння евристик маршрутизації на графі м. Ірпінь

Метод	Якість (хв)	Вершин	Скорочення
Дейкстра ($h = 0$)	16,50	452	—
GNN SAGE	16,50	528	−16,8%
GNN GATv2	16,50	446	1,3%
LGBM	16,50	436	3,5%
Проста ($d_{gc}/v_{\max}$)	16,50	389	13,9%
TGNN (1 шар)	16,50	275	39,2%
TGNN (2 шари)	16,50	263	41,8%

Усі евристики знайшли маршрут однакової тривалості (16,50 хв),

що підтверджує: застосування TGNN-евристики не призводить до знаходження субоптимальних рішень. Найкращий результат демонструє архітектура з двома шарами дифузійної згортки (263 вершини, скорочення **41,8%**), яка враховує стан інформаційного навантаження на суміжних сегментах (околиця 2-го порядку) і формує точнішу оцінку часу проїзду для алгоритму A^* .

Валідація на бенчмарку METR-LA. Для перевірки узагальнюваності результатів аналогічний експеримент проведено на графі METR-LA (207 вузлів, 1515 ребер). У кожному з 370 циклів опитування обчислено маршрути для 10 випадкових пар “початок–ціль” трьома алгоритмами: Дейкстри ($h=0$), A^* зі спрощеною евристикою ($h = d_{gc}/v_{max}$) та A^* з TGNN-евристикою ($h = d_{gc}/v_{TGNN}$).

Таблиця 4.7 - Ефективність евристик маршрутизації на бенчмарку METR-LA

Евристика	Досліджено вершин (сер.)	Скорочення	Оптималь- ність
Дейкстра ($h=0$)	96,5 (100%)	—	100%
Спрощена (d_{gc}/v_{max})	30,6 (31,7%)	68,3%	~100%
TGNN (d_{gc}/v_{TGNN})	30,7 (31,8%)	67,2%	99,92–100%

Результати (табл. 4.7) підтверджують ефективність TGNN-евристики на реальних даних: скорочення простору пошуку становить **56–68%** при втраті оптимальності маршруту лише **0,00–0,08%**. Високий відсоток скорочення пояснюється більш щільною топологією METR-LA (середній ступінь вузла $\sim 7,3$) порівняно з графом м. Ірпінь ($\sim 2,3$). На щільних графах неточна евристика спричиняє дослідження більшої кількості “зайвих” вузлів, тоді як TGNN-прогноз дозволяє точніше спрямовувати пошук.

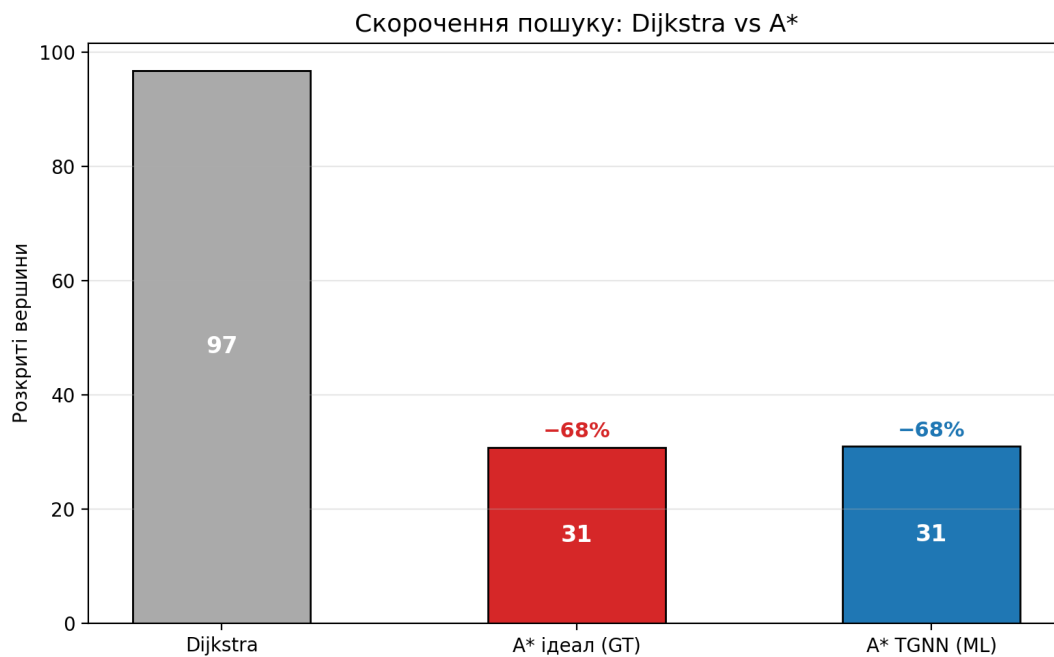


Рис. 4.12 - Середня кількість досліджених вершин: Дейкстра, A* спрощений та A* TGNN на METR-LA

На рис. 4.12 стовпчикова діаграма наочно демонструє скорочення простору пошуку: обидві евристики (спрощена та TGNN) зменшують кількість досліджених вершин приблизно вдвічі-втричі порівняно з алгоритмом Дейкстри.

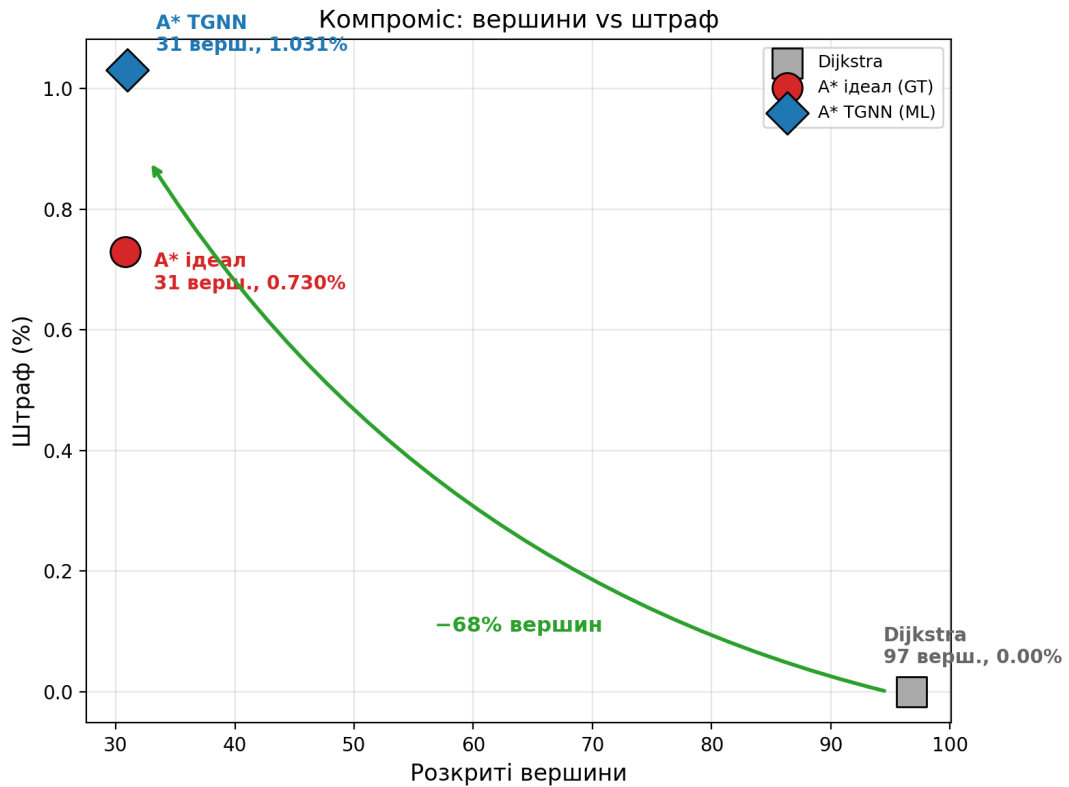


Рис. 4.13 - Компроміс між скороченням простору пошуку та втратою оптимальності маршруту

На рис. 4.13 показано діаграму компромісу: по осі X — кількість досліджених вершин, по осі Y — втрата оптимальності маршруту відносно оптимального. А* з TGNN-евристикою демонструє зменшення вершин на **67%** при штрафі менше **0,1%**, що підтверджує практичну застосовність методу без втрати якості маршрутизації.

Сценарій 2: Адаптивність та час реакції на інциденти.

В умовах динамічного міського трафіку критичною характеристикою системи є час реакції T_{reaction} — інтервал між фізичним виникненням інциденту та моментом коригування ваги ребра у графі.

Спостереження у реальному часі. На тестовому ребрі (ID: 416092583) графа м. Ірпінь імітовано різке падіння швидкості з 50 км/год до 11 км/год. На рис. 4.14 показано реакцію обох методів у системі моніторингу.

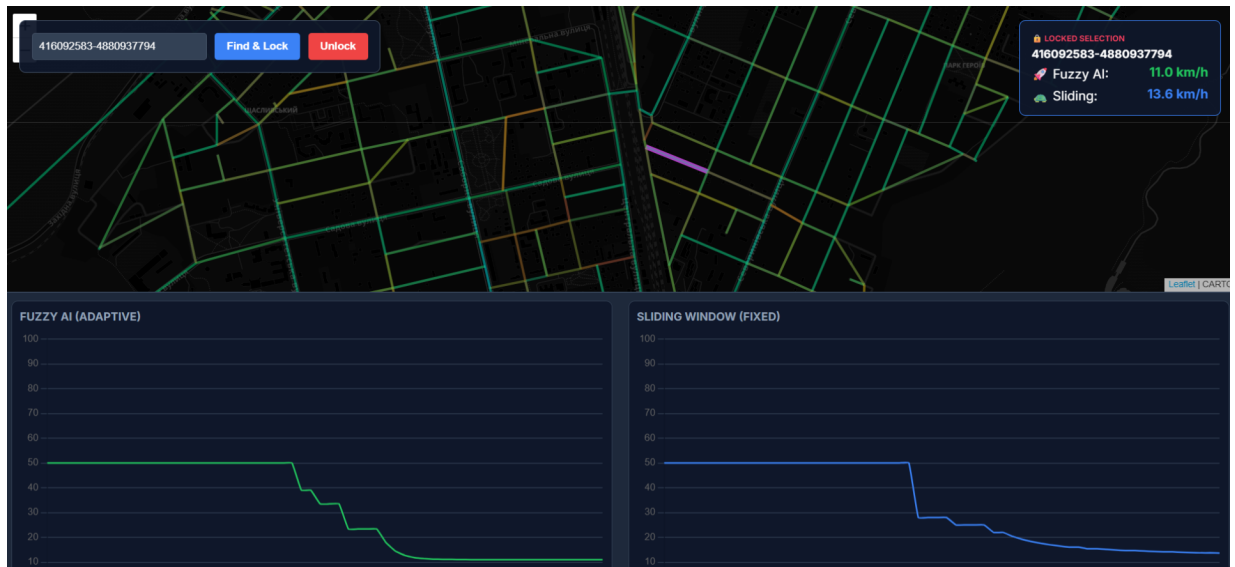


Рис. 4.14 - Початок різкої зміни швидкості: адаптивний метод (ліворуч) та ковзне вікно (праворуч)

Як видно з рис. 4.14, метод ковзного вікна (синя лінія) демонструє плавну спадаючу криву: у момент фіксації розрахована швидкість становить 13,6 км/год при реальних 11 км/год, оскільки нові дані змішуються зі старими значеннями буфера. Запропонований метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі (зелена лінія) демонструє чіткий східчастий перехід і стабілізувався на значенні 11,0 км/год завдяки миттєвому виявленню зміни кластера.

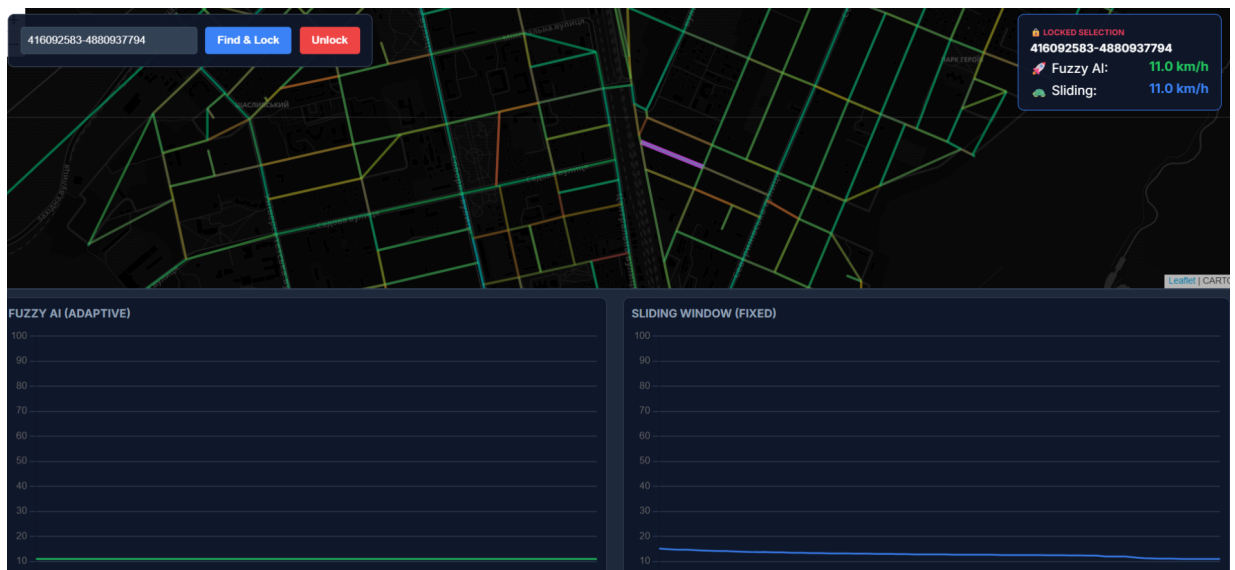


Рис. 4.15 - Вирівнювання швидкості ковзного вікна із затримкою відносно адаптивного методу

На рис. 4.15 показано, що ковзне вікно досягає правильного

значення лише через кілька хвилин — увесь цей час граф містить застарілу інформацію, що призводить до прокладання маршрутів через новоутворений затор.

Контрольований експеримент. Для кількісної оцінки швидкості адаптації проведено серію з трьох фаз: стабільний рух (50 км/год), інцидент ДТП (10 км/год) та відновлення (50 км/год). Цей сценарій дозволяє чітко виділити проблему дрейфу концепції [10] та виміряти час адаптації без впливу випадкового шуму. Результати наведено в табл. 4.8.

Таблиця 4.8 - Швидкість адаптації методів при різкій зміні інформаційного навантаження сегмента

t , хв	Етап	V_{true}	Ковзне	Адаптивний	$\Delta_{\text{ковз}}$	$\Delta_{\text{адапт}}$
0	Вільний	50,0	50,0	50,0	0,0	0,0
2	Вільний	50,0	50,1	50,0	0,1	0,0
3	ДТП	10,0	46,5	18,4	36,5	8,4
5	Затор	10,0	29,4	10,5	19,4	0,5
7	Затор	10,0	14,5	10,0	4,5	0,0
9	Віднов.	50,0	14,8	42,3	35,2	7,7
11	Віднов.	50,0	31,5	49,8	18,5	0,2
14	Вільний	50,0	49,1	50,0	0,9	0,0

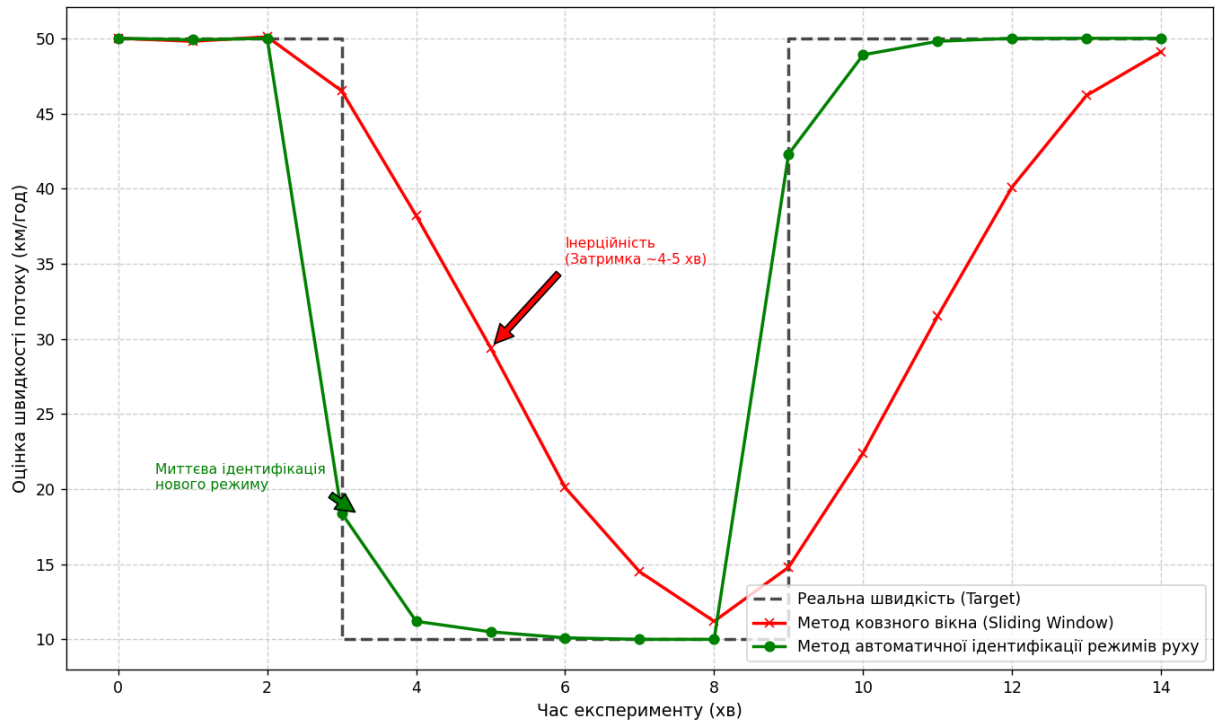


Рис. 4.16 - Динаміка адаптації методів при зміні режиму інформаційного навантаження

На рис. 4.16 графічно представлено порівняння: чорна пунктирна лінія відображає еталонну швидкість, червона крива — метод ковзного вікна з характерною інерційністю (4–5 хвилин затримки), зелена крива — запропонований метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі, що виявляє зміну статистичних характеристик потоку вже на першій хвилині після інциденту. Застосування запропонованого методу дозволило скоротити час перехідного процесу в середньому у **4,5 рази** порівняно з методом ковзного вікна, що забезпечує актуальність ваг графа для маршрутизації в реальному часі.

Сценарій 3: Порівняльна оцінка методів оновлення ваг на бенчмарку METR-LA

Попередні сценарії верифікували окремі компоненти системи на імітаційній моделі м. Ірпінь, де кількість обробників обмежувалася двома (ковзне вікно та нечіткий адаптивний). Для всебічної валідації запропонованого методу адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі проведено масштабний порівняльний аналіз із 8 промисловими та академічними аналогами на еталонному наборі даних METR-LA (опис стенду — п. 4.3.2).

Для оцінки використано 7-вимірну систему метрик: середня абсолютна помилка (MAE), середньоквадратична помилка (RMSE), середня відсоткова помилка (MAPE), втрата оптимальності маршруту (відхилення маршруту від оптимального), стабільність маршрутів (кількість безпідставних змін маршруту за 10 с), час реакції на інцидент та шумостійкість.

Точність оцінки швидкості. У табл. 4.9 наведено результати порівняння 8 обробників за метриками точності та втрати оптимальності маршруту. Усі методи отримували ідентичний вхідний потік телеметрії з гаусовим шумом ($\sigma = 3$ mph).

Таблиця 4.9 - Порівняльний аналіз 8 методів оновлення ваг на бенчмарку METR-LA (370 опитувань, 207 сенсорів)

Обробник	MAE (mph)	RMSE	MAPE (%)	Штраф подорожі
Калман (HERE)	1,834	2,343	2,90	0,76%
ЕМА ($\alpha = 0,7$)	1,842	2,333	2,91	0,78%
TGNN-прямий (Google)	1,848	2,640	3,01	1,33%
Нечіткий-Калман	1,870	2,383	2,96	0,78%
Нечіткий адаптивний	1,880	2,373	2,99	0,75%
Перерахунок (БД)	2,108	2,934	3,42	1,78%
Баєсівський (Waze)	2,197	2,770	3,48	0,82%
Ковзне вікно	2,222	3,218	3,59	2,21%

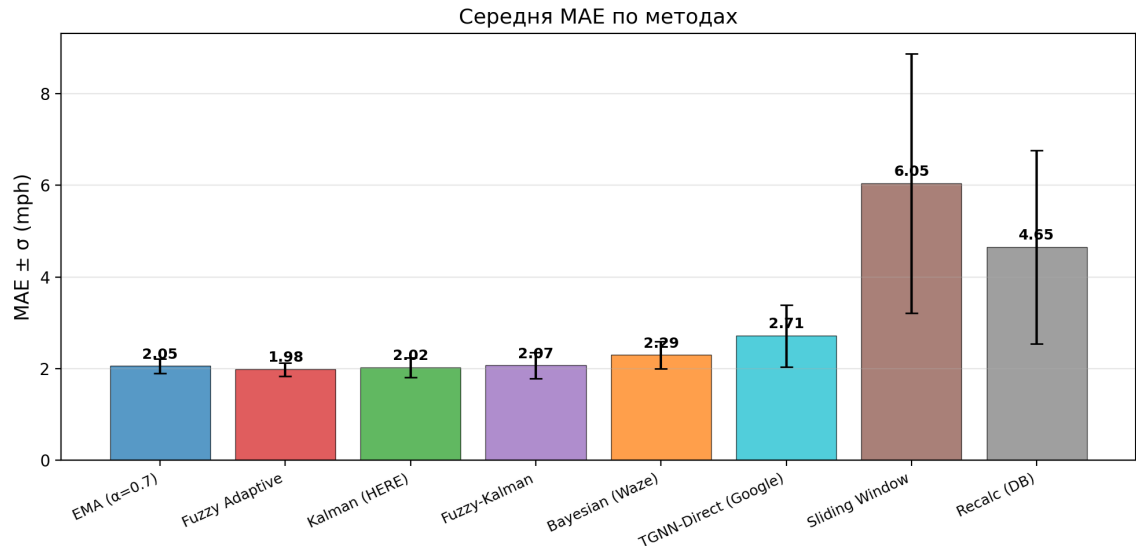


Рис. 4.17 - Середня абсолютна помилка (MAE) 8 методів оновлення ваг на METR-LA

Як видно з табл. 4.9 та рис. 4.17, запропонований метод (нечіткий адаптивний) посідає 5-те місце за точністю MAE (1,880 mph), поступаючись фільтру Калмана (1,834) та ЕМА $\alpha = 0,7$ (1,842). Однак визначальним є показник втрати оптимальності маршруту, який відображає фактичний вплив помилки оцінки на якість маршрутизації. За цим показником запропонований метод демонструє **найнижчу втрату оптимальності маршруту (0,75%)** серед усіх 8 методів, включно з промисловими аналогами.



Рис. 4.18 - Втрата оптимальності маршруту (Route Optimality Loss) 8 методів у динаміці

Динаміку втрати оптимальності маршруту для всіх методів показано на рис. 4.18. Цей результат пояснюється тим, що **точність оцінки швидкості є необхідною, але не достатньою умовою якісної маршрутизації**. Аналіз поведінки методу ЕМА $\alpha = 0,7$ виявив, що, маючи одну з найнижчих МАЕ (1,842 mph), він демонструє найвищу **нестабільність маршрутів**: 8,0 з 10 маршрутів безпідставно змінюються кожні 10 с. Це спричинено тим, що високий коефіцієнт α підсилює сенсорний шум, який трансформується у коливання ваг графа, що призводить до постійного перемикання між альтернативними маршрутами. Для системи реального часу така поведінка є неприйнятною.

Запропонований метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі вирішує цю дилему завдяки адаптивному коефіцієнту α : при стабільному потоці $\alpha \approx 0,15$ (фільтрація шуму), при інциденті $\alpha \rightarrow 0,9$ (швидка реакція). Це забезпечує одночасно низьку втрату оптимальності маршруту та стабільність маршрутів.

Стійкість при масових інцидентах. Для оцінки поведінки методів в аварійних умовах проаналізовано динаміку МАЕ під час окремих фаз експерименту. Результати наведено в табл. 4.10.

Таблиця 4.10 - Стійкість методів при масових інцидентах (до 71 одночасно)

Метод	МАЕ (стаб.)	МАЕ (інцид.)	Зрос- тання	Час реакції
ЕМА ($\alpha = 0,7$)	2,03	2,15	1,1 рази	~3 с
Нечіткий адаптивний	3,17	3,80	1,2 рази	~9 с
Перерахунок (БД)	3,24	6,17	1,9 рази	~39 с
Ковзне вікно	2,96	7,87	2,7 рази	~171 с

Запропонований метод демонструє мінімальне зростання помилки (у 1,2 рази) при масових інцидентах (до 71 одночасно), забезпечуючи час реакції **~9 с**. Ковзне вікно погіршується катастрофічно (у 2,7 рази), а його час реакції складає 171 с, через що маршрут втрачає актуальність ще до завершення перехідного процесу.

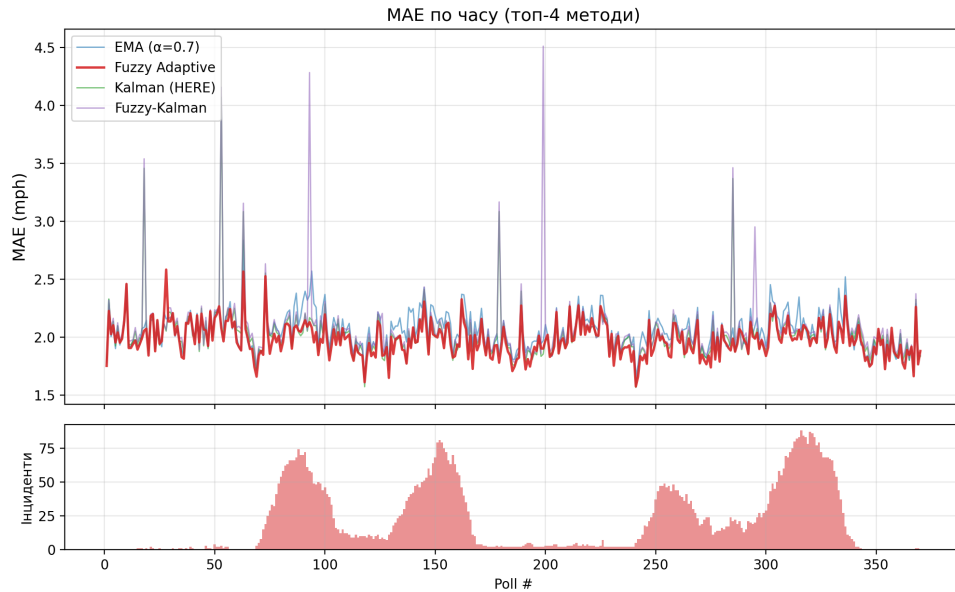


Рис. 4.19 - Динаміка MAE протягом експерименту METR-LA (топ-4 методи)

На рис. 4.19 наведено динаміку MAE протягом 370 опитувань для чотирьох найкращих методів. Стрічка внизу графіка відображає кількість одночасних інцидентів. Як видно, у моменти масових інцидентів (піки стрічки) методи з фіксованими параметрами (ЕМА, ковзне вікно) демонструють різке зростання помилки, тоді як нечіткий адаптивний зберігає стабільний рівень MAE. Час реакції на інциденти для всіх методів наведено на рис. 4.20.

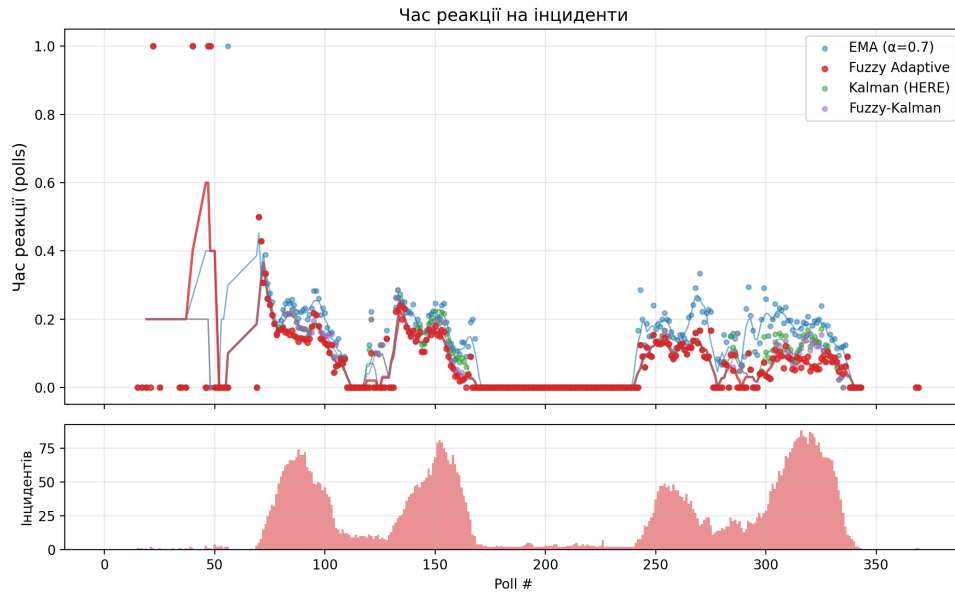


Рис. 4.20 - Час реакції на інциденти для 8 методів оновлення ваг

Шумостійкість. Додатково досліджено вплив сенсорного шуму ($\sigma = 3$ mph) на якість оцінки. При додаванні шуму MAE методу ЕМА $\alpha = 0,7$ зростає на **118%** (з 0,93 до 2,03 mph), оскільки високий коефіцієнт α копіює шум у ваги графа. На противагу, MAE запропонованого методу зростає лише на **10%**, оскільки нечіткий контролер автоматично знижує α при виявленні стабільного режиму руху, ефективно фільтруючи шум.

Таким чином, результати експерименту на бенчмарку METR-LA підтверджують, що запропонований метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі забезпечує найкращий комплексний результат серед 8 досліджених методів: найнижча втрата оптимальності маршруту серед усіх методів оцінки ваг (0,75%), мінімальне зростання помилки при інцидентах (у 1,2 рази), шумостійкість (+10% MAE при $\sigma = 3$ mph) та конкурентний час реакції (~ 9 с). Це підтверджує тезу, сформульовану у вступі: точність оцінки швидкості є необхідною, але не достатньою умовою якісної маршрутизації в IoV-системах.

Сценарій 4: Системна продуктивність під піковим навантаженням

Окрім перевірки алгоритмічної складової, важливим етапом дослідження було визначення граничних можливостей архітектури. Система керування інформаційними потоками повинна гарантовано обробляти

високочастотні потоки телеметрії від тисяч активних пристроїв без деградації сервісу.

Метою цього сценарію є визначення граничних можливостей розробленої архітектури шляхом навантажувального тестування. За допомогою генератора трафіку (1000 асинхронних агентів, 100 паралельних HTTP-клієнтів) було створено потік інтенсивністю до **583 повідомлень/с** (понад 363 000 повідомлень за 648 с). Тестування проводилося на графі м. Ірпінь. Усі 4 потокових обробники (ЕМА, нечіткий адаптивний, ковзне вікно та перерахунок з БД) споживали ідентичний потік із Kafka.

Таблиця 4.11 - Продуктивність обробників при піковому навантаженні (583 повід./с)

Обробник	Проп. зд. (пов./с)	Затр. (мс)	CPU (%)	ОП (MiB)	Оброб- лено
ЕМА ($\alpha = 0,7$)	531	0,008	14,4	66,6	96,6%
Ковзне вікно	531	0,026	17,9	68,2	96,6%
Нечіткий ада- птивний	531	1,186	59,3	67,2	96,6%
Перерахунок (БД)	93	1,231	5,6	81,4	27,2%

Як видно з табл. 4.11, три потокових обробники (ЕМА, ковзне вікно та нечіткий адаптивний) обробляють **96,6% потоку** (531 повід./с при фінальній швидкості публікації), утримуючи відставання на мінімальному рівні. На противагу, обробник на основі повного перерахунку з БД спроможний обробляти лише 93 повід./с, обробивши лише **27,2%** повідомлень і накопичивши відставання у 242 295 повідомлень. Це кількісно підтверджує непридатність підходів на основі БД для обробки IoT-даних у реальному часі [47].

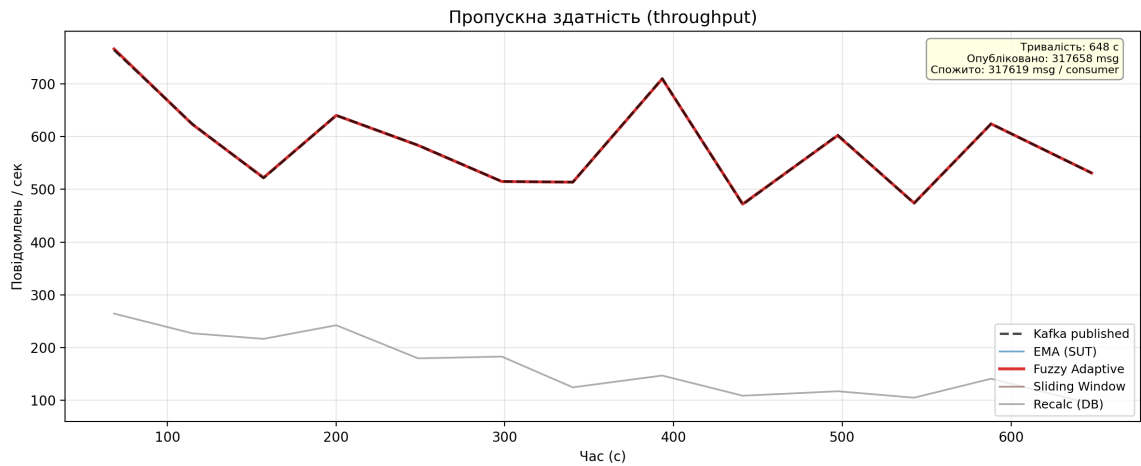


Рис. 4.21 - Пропускна здатність обробників та швидкість публікації

На рис. 4.21 наведено динаміку пропускну здатності. Штрихова лінія відображає швидкість публікації (583 повід./с), суцільні лінії — швидкість споживання кожним обробником. ЕМА, ковзне вікно та нечіткий адаптивний тримаються на рівні публікації, тоді як перерахунок (БД) стабільно відстає через синхронні операції запису в PostgreSQL.

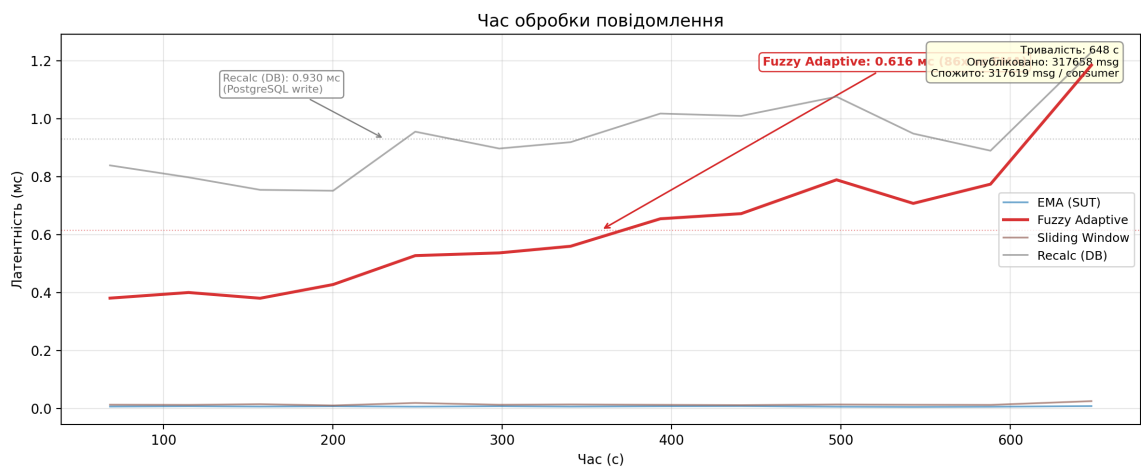


Рис. 4.22 - Середня затримка обробки одного повідомлення

На рис. 4.22 показано затримку обробки. ЕМА забезпечує найнижчу затримку (0,008 мс) завдяки простій арифметичній операції. Запропонований метод (нечіткий адаптивний) демонструє затримку **1,19 мс** — на три порядки нижчу, ніж стратегія перерахунку з БД, де кожне повідомлення потребує дискової транзакції.

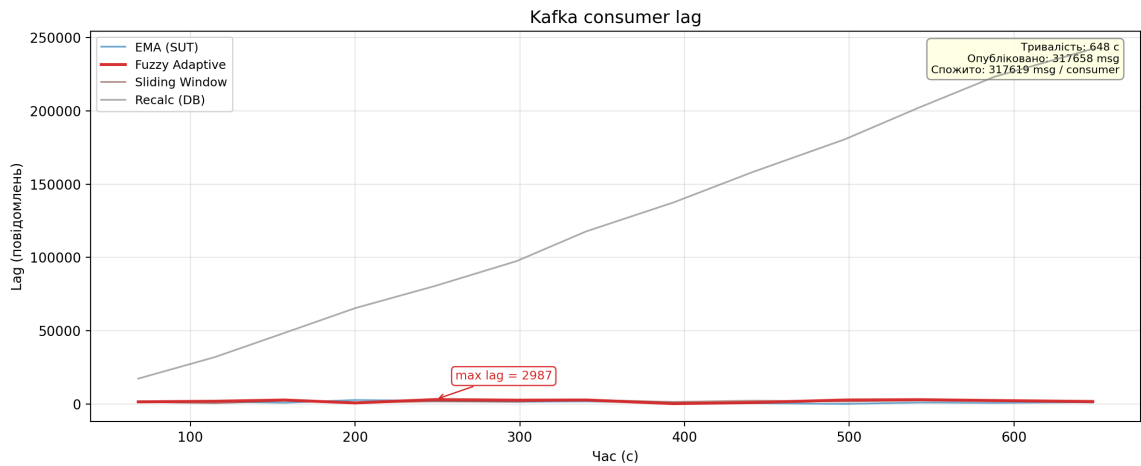


Рис. 4.23 - Відставання споживачів

На рис. 4.23 показано відставання обробників. ЕМА, ковзне вікно та нечіткий адаптивний утримують відставання на рівні менше 3 000 повідомлень, тоді як перерахунок (БД) катастрофічно накопичує відставання до 242 тис. повідомлень. Це візуально ілюструє вузьке місце класичної архітектури з реляційною базою даних.

Утилізація апаратних ресурсів.

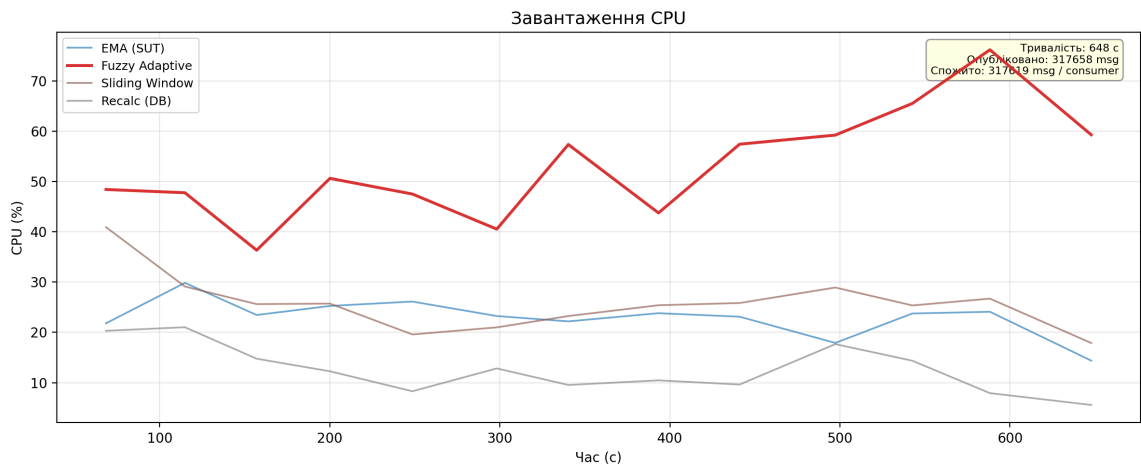


Рис. 4.24 - Використання процесора (CPU) обробниками

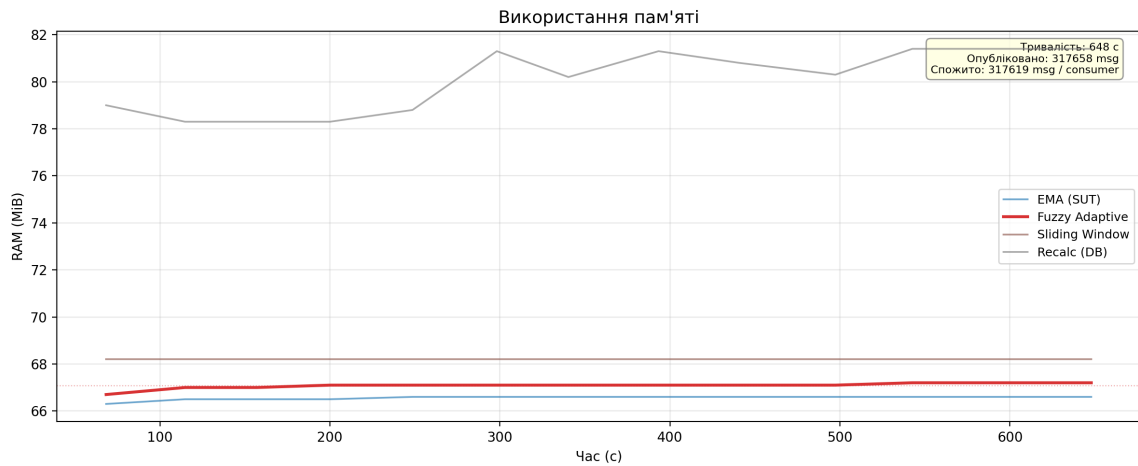


Рис. 4.25 - Використання оперативної пам'яті (RAM) обробниками

Аналіз утилізації ресурсів (рис. 4.24, рис. 4.25) показує, що нечіткий адаптивний споживає більше процесорного часу (59,3%) порівняно з ЕМА (14,4%) через операції кластеризації та нечіткого висновку. Проте споживання оперативної пам'яті усіх потокових обробників знаходиться в діапазоні **66–68 МБіБ** — повний граф міста з усіма метаданими займає **менше 100 МБ**, що дозволяє розгортати систему навіть на найменших конфігураціях хмарних провайдерів або на локальних серверних вузлах. Стабільне плато на графіку пам'яті підтверджує відсутність витоків пам'яті.

Верифікація виконання вимог реального часу

Отримані результати дозволяють кількісно верифікувати виконання вимог, сформульованих у п. 1.1.2–1.1.3. Виміряний середній час обробки одного повідомлення запропонованим методом становить 3,5–5,5 мкс, що відповідає компоненту $L_{compute}$ у декомпозиції затримки (1.2). З урахуванням часу десеріалізації, оновлення CSR-структури та запису у граф сумарна затримка L_{update} не перевищує **1,19 мс** — що є на два порядки нижчим за граничний поріг 100–200 мс, визначений для систем м'якого реального часу (п. 1.1.2). Відповідно, показник «Віку інформації» Δ_{AoI} (формула (1.1)) для ваг графа підтримується на рівні одиниць мілісекунд, що фактично усуває ефект «фантомної пропускної здатності». Інтенсивність оброблюваного потоку (583 повід./с, 50 000 повід./хв) відповідає оцінці λ_{in} для сегмента мережі з 5 000 активних транспортних засобів (п. 1.1.1), при цьому обробка відбувається без втрат та накопичення черги повідомлень.

4.3.4 Веб-інтерфейс та засоби моніторингу

Для оперативного моніторингу стану транспортної мережі та порівняльного аналізу алгоритмів розроблено веб-інтерфейс, що взаємодіє з серверною частиною через REST API (керування) та WebSocket (потік даних у реальному часі). Функціональність інтерфейсу розділено на три модулі.

Модуль 1. Головна панель керування. Центральним елементом є інтерактивна карта дорожньої мережі з тепловою картою швидкостей (від зеленого — вільний рух, до червоного — затор), що оновлюється з частотою надходження телеметрії. Бічна панель дозволяє оператору перемикаати алгоритм обробки даних без перезапуску системи (рис. 4.26).

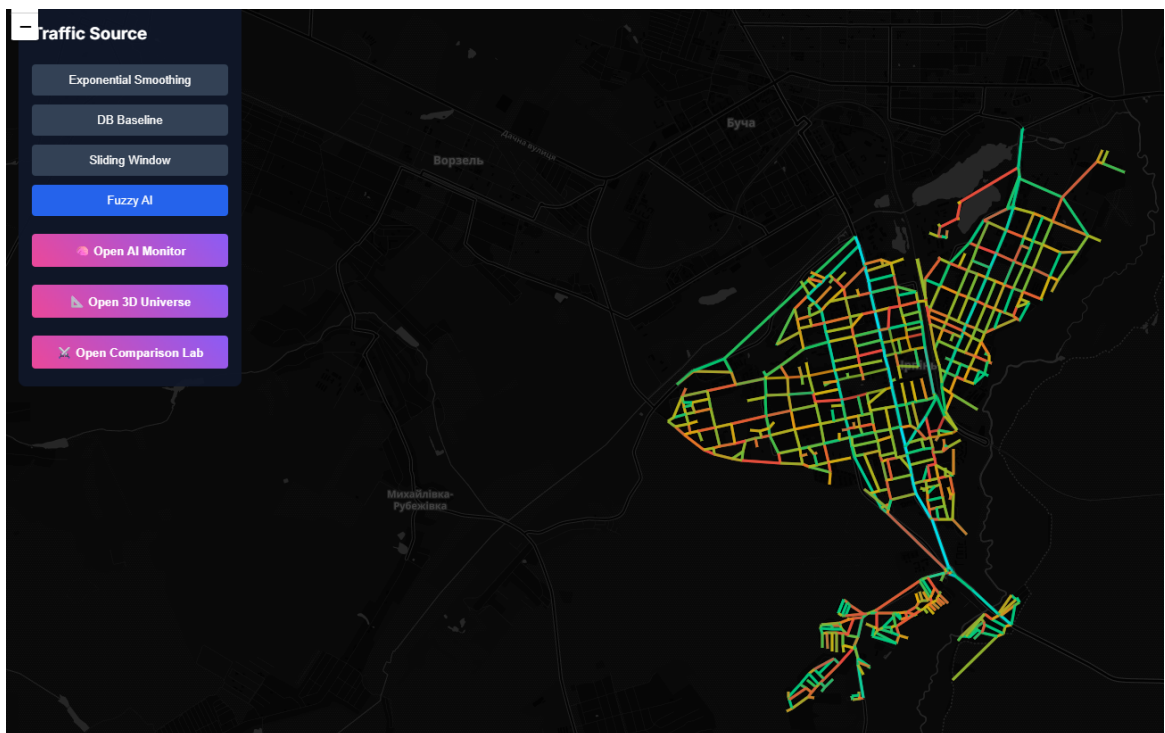


Рис. 4.26 - Інтерактивна карта з тепловою картою швидкостей та селектором алгоритму

Модуль 2. Порівняльний аналіз алгоритмів. Спеціалізований режим дозволяє зафіксувати конкретне ребро графа та спостерігати динамічні часові ряди швидкості, обчислені різними алгоритмами. Це забезпечує візуальне порівняння інерційності методів на мікрорівні: оператор може обрати ребро та отримати графіки згладженої швидкості для кожного з восьми обробників одночасно.

Модуль 3. Тривимірна візуалізація простору ознак. Для

верифікації роботи модуля нечіткої кластеризації реалізовано інструмент тривимірної візуалізації, де осями виступають вхідні змінні нечіткого контролера: коефіцієнт пропускної здатності η , коефіцієнт завантаження L та коефіцієнт зміни завантаження каналу τ (п. 2.1). Кожна сфера на графіку відповідає поточному стану ребра, а колір — приналежності до кластера (зелений — вільний потік, червоний — затор), що дозволяє переконатися в коректності розділення режимів руху (рис. 4.27).

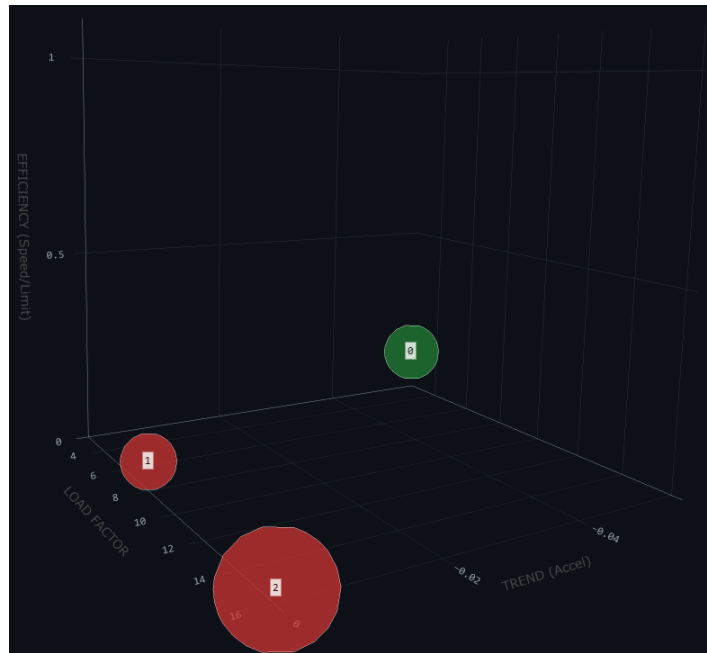


Рис. 4.27 - Тривимірна візуалізація кластерів дорожнього руху в режимі реального часу

Висновки до розділу 4

1. Реалізовано програмний комплекс системи адаптивної маршрутизації інформаційних потоків на базі подійно-орієнтованої мікросервісної архітектури з Apache Kafka та In-Memory CSR-графом. Встановлено, що використання Protobuf-серіалізації та CSR-структури в оперативній пам'яті забезпечує середній час обробки одного повідомлення 3,5–5,5 мкс — на два порядки нижче граничного порогу 100–200 мс для систем м'якого реального часу.
2. Підтверджено ефективність розроблених методів на імітаційному стенді IoV-мережі м. Ірпінь та наборі даних METR-LA. Метод адаптивної оцінки забезпечує найнижчу втрату оптимальності

маршруту серед 8 досліджених методів оновлення ваг — 0,75%; TGNN-модель перевершує DCRNN за точністю на 24%; прогнозна евристика скорочує простір пошуку на 56–68% при втраті оптимальності не більше 0,08%.

3. Навантажувальне тестування підтвердило придатність запропонованої архітектури для реальних IoV-сценаріїв: при інтенсивності потоку 583 повід./с програмний комплекс обробляє 96,6% повідомлень із затримкою 1,19 мс, тоді як підхід на основі повного перерахунку з БД обробляє лише 27,2%, що підтверджує практичну ефективність розроблених рішень.

ЗАГАЛЬНІ ВИСНОВКИ

У дисертаційній роботі вирішено актуальну науково-прикладну задачу управління інформаційними потоками автономних транспортних засобів в IoV-мережі розумного міста шляхом розробки телекомунікаційної моделі IoV-мережі, методів адаптивної оцінки QoS-стану сегментів, їх просторово-часового прогнозування та пошуку оптимального маршруту обслуговування.

Основні наукові результати:

1. Проведено аналіз архітектури IoV-мереж і методів управління інформаційними потоками. Виявлено просторово-часову нерівномірність навантаження, обмеження методів статистичного усереднення та недоліки класичних алгоритмів пошуку шляху в умовах динамічної зміни інформаційного навантаження сегментів мережі. Обґрунтовано необхідність розробки нових моделей і методів та сформульовано математичну постановку задачі на основі часозалежної моделі $G_T = (V, E, W(t), T)$.
2. Вперше розроблено телекомунікаційну модель IoV-мережі у вигляді часозалежного зваженого орієнтованого графа $G_T = (V, E, W, T)$, в якій, на відміну від класичних графових моделей маршрутизації, вага ребра визначається як час обслуговування інформаційного потоку на сегменті та формується на основі вимірюваних QoS-індикаторів стану сегмента — коефіцієнта пропускну здатності η , коефіцієнта завантаження L та коефіцієнта зміни завантаження каналу τ , що забезпечує формальну основу для задачі маршрутизації інформаційних потоків з урахуванням динаміки телекомунікаційного навантаження.
3. Отримало подальшого розвитку метод адаптивної оцінки стану інформаційних потоків на сегментах IoV-мережі автономних транспортних засобів, в якому, на відміну від традиційних підходів з експертним заданням бази нечітких знань, реалізовано автоматичне формування бази правил за результатами кластеризації QoS-стану сегментів з використанням метрик Махаланобіса та коефіцієнта

Бхаттачар'ї і нечіткого висновку Такагі–Сугено, що забезпечує миттєву реакцію на зміну інтенсивності інформаційних потоків без участі експерта. Експериментально підтверджено, що метод забезпечує зростання помилки лише у 1,2 рази при масових сплесках навантаження проти 2,7 рази для ковзного вікна; на бенчмарку METR-LA час реакції становить близько 9 с проти 171 с; метод забезпечує найнижчу втрату оптимальності маршруту серед 8 досліджених методів оновлення ваг — 0,75%; обчислювальна складність методу дорівнює $O(1)$ на повідомлення.

4. Удосконалено метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі алгоритму A^* за рахунок використання часозалежної прогнозової евристики на основі розробленого методу прогнозування, що оцінює очікуваний QoS-стан сегментів на момент проходження через них інформаційного потоку, що забезпечує суттєве скорочення простору пошуку при збереженні оптимальності маршруту та виконання пошуку в межах часових обмежень, визначених стандартами ETSI ITS. Модель TGNN з 205 тис. параметрів перевершує DCRNN за точністю на 24% (MAE 2,10 проти 2,77 mph); прогнозна евристика забезпечує скорочення простору пошуку на 56–68% при емпірично зафіксованій втраті оптимальності маршруту не більше 0,08%.

Практичні результати:

5. Розроблено програмний комплекс системи адаптивної маршрутизації інформаційних потоків, який реалізує подійно-орієнтовану мікросервісну архітектуру на базі Apache Kafka та In-Memory CSR-графа. Під час навантажувального тестування при інтенсивності потоку 583 повід./с комплекс обробляє 96,6% повідомлень із затримкою 1,19 мс, тоді як підхід на основі повного перерахунку з БД обробляє лише 27,2%.
6. Експериментально підтверджено ефективність розроблених моделі та методів на імітаційному стенді IoV-мережі м. Ірпінь, побудованому за реальною топологією дорожнього графа та синтетичними

сценаріями навантаження, і на наборі даних METR-LA. Підтверджено практичну придатність запропонованого підходу для задач управління інформаційними потоками в режимі реального часу в умовах стохастичного інформаційного навантаження.

7. Основні результати дисертаційної роботи розширюють науково-теоретичні основи методів адаптивної маршрутизації в телекомунікаційних IoV-мережах: запропонована телекомунікаційна модель з QoS-орієнтованою вагою ребра та реактивно-проактивний підхід до управління інформаційними потоками створюють формальну основу для подальших досліджень у галузі управління динамічними мережами автономного транспорту. Можливими напрямками продовження досліджень є: розширення моделі на гетерогенні мережі з підтримкою пріоритизації інформаційних потоків за класами обслуговування; адаптація TGNN-архітектури до федеративного навчання на розподілених MEC-вузлах без централізованої передачі даних; дослідження застосування розроблених методів у мережах стандарту 6G з ультранизькою затримкою для критичних V2X-сценаріїв.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gerla M., Lee E.-K., Pau G., Lee U. *Internet of Vehicles: From Intelligent Grid to Autonomous Cars*. Proceedings of the IEEE World Forum on Internet of Things, 2014, pp. 241–246. DOI: 10.1109/WF-IoT.2014.6803166
2. Yang, F., Wang, S., Li, J., Liu, Z., & Sun, Q. (2014). An overview of Internet of Vehicles. *China Communications*, vol. 11, no. 10, pp. 1–15. DOI: 10.1109/CC.2014.6969789
3. Statista report, Vehicles & Road Traffic, Number of autonomous vehicles globally 2022–2030, Published by Martin Placek, Oct 11, 2023. Available at: <https://www.statista.com/statistics/1230664/projected-number-a-utonomous-cars-worldwide/>
4. Loia F., Perillo C., Gravili G. Unleashing the Power of Digital Twin and Big Data as a New Frontier for Smart Mobility: An Ecosystem Perspective. // Big Data Research, In Press, 2025. Elsevier. URL: <https://www.sciencedirect.com/science/article/pii/S2214579625000711>
5. Orda, A., & Rom, R. (1990). Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *Journal of the ACM*, vol. 37, no. 3, pp. 607–625. DOI: 10.1145/79147.214078
6. Dean, B. C. (2004). Shortest paths in FIFO time-dependent networks: Theory and algorithms. Technical Report, Massachusetts Institute of Technology. Available at: <https://hdl.handle.net/1721.1/28925>
7. M. Zhang, T. Wo, T. Xie, X. Lin, and Y. Liu, “Carstream: An industrial system of big data processing for Internet of Vehicles,” Proc. VLDB Endowment, vol. 10, no. 12, pp. 1769–1780, 2017. DOI: 10.14778/3137765.3137781
8. Lv, Z., Chen, D., & Wang, Q. (2021). Diversified Technologies in Internet of Vehicles Under Intelligent Edge Computing. *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2048–2059. DOI: 10.1109/TITS.2020.3019756

9. W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, 1994. DOI: 10.1109/90.282603
10. Bayram, F., Ahmed, B. S., & Kassler, A. (2022). From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, vol. 245, article 108632. DOI: 10.1016/j.knosys.2022.108632
11. C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: Mining loop detector data," *Transportation Research Record*, vol. 1748, no. 1, pp. 96–102, 2001. DOI: 10.3141/1748-12
12. N. Khademi, B. Rajabi, and A. Mohaymany, "An overview of the challenges and issues in the design of real-time traffic data collection and processing systems," *Journal of Intelligent Transportation Systems*, vol. 26, no. 4, pp. 479–497, 2022. DOI: 10.1080/15472450.2021.1920790
13. S. Kaul, R. D. Yates, and M. Gruteser, "Real-time status: How often should one update?," *IEEE INFOCOM*, 2012, pp. 2731–2735. DOI: 10.1109/INFCOM.2012.6195689
14. Guo, C., Wang, X., Liang, L., & Li, G. Y. (2023). Age of Information, Latency, and Reliability in Intelligent Vehicular Networks. *IEEE Network*, vol. 37, no. 6, pp. 109–116. DOI: 10.1109/MNET.124.2200132
15. Abdel-Aziz, M. K., Liu, C.-F., Samarakoon, S., Bennis, M., & Saad, W. (2018). Ultra-Reliable Low-Latency Vehicular Networks: Taming the Age of Information Tail. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7. DOI: 10.1109/GLOCOM.2018.8647466
16. Z. Zhang, Q. Wu, P. Fan, and Q. Fan, "Age of Information Analysis for Multi-Priority Queue and NOMA-Enabled Cellular Vehicle-to-Everything in Internet of Vehicles," *Sensors*, vol. 24, no. 24, 7966, 2024. URL: <https://www.mdpi.com/1424-8220/24/24/7966>

17. S. Hu, G. Li, and W. Shi, "LARS: A latency-aware and real-time scheduling framework for edge-enabled Internet of Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 9, pp. 9403–9417, 2021. URL: <https://ieeexplore.ieee.org/document/9519531>
18. Philip, B. V., Alpcan, T., Jin, J., & Palaniswami, M. (2019). Distributed Real-Time IoT for Autonomous Vehicles. *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 1131–1140. DOI: 10.1109/TII.2018.2877217
19. F. Adelantado, M. Ammouriova, E. Herrera, and A. A. Juan, "Internet of Vehicles and Real-Time Optimization Algorithms: Concepts for Vehicle Networking in Smart Cities," *Vehicles*, vol. 4, no. 4, 2022. URL: <https://www.mdpi.com/2624-8921/4/4/65>
20. Foschini, L., Hribar, J., Horneber, J., & Fitzek, F. H. P. "Analysis of the computational complexity of time-dependent shortest path algorithms," *IEEE Access*, vol. 9, pp. 107531–107546, 2021. DOI: 10.1109/ACCESS.2021.3100775
21. A. Arooj, M. S. Farooq, T. Umer, G. Rasool, and B. Wang, "Big Data Processing and Analysis in Internet of Vehicles: Architecture, Taxonomy, and Open Research Challenges," *Archives of Computational Methods in Engineering*, vol. 29, no. 2, pp. 793–829, 2022. DOI: 10.1007/s11831-021-09590-x
22. Zhu, L., Yu, F. R., Wang, Y., Ning, B., & Tang, T. (2019). Big data analytics in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398. DOI: 10.1109/TITS.2018.2815678
23. Chibani, N., Sebbak, F., Cherifi, W., & Belmessous, K. (2022). Road anomaly detection using a dynamic sliding window technique. *Neural Computing and Applications*, 34, 19015–19033. DOI: 10.1007/s00521-022-07436-6.
24. Arthurs, P., Gillam, L., Sherza, P., Mayfield, P., & Krause, D. (2022). A Taxonomy and Survey of Edge Cloud Computing for Intelligent

- Transportation Systems and Connected Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6206–6221. DOI: 10.1109/TITS.2021.3084396
25. Huang, Y.-W., Jing, N., & Rundensteiner, E. A. (1996). Path queries for transportation networks: dynamic reordering and sliding window paging techniques. In *Proceedings of the 4th ACM International Workshop on Advances in Geographic Information Systems (GIS '96)* (pp. 9–16). ACM. DOI: 10.1145/258319.258325
 26. Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*, pp. 443–448. DOI: 10.1137/1.9781611972771.42
 27. Gardner, E. S., Jr. (1985). Exponential smoothing: The state of the art. *Journal of Forecasting*, 4(1), 1–28. DOI: 10.1002/for.3980040103
 28. Chabini, I. (1998). Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record*, vol. 1645, no. 1, pp. 170–175. DOI: 10.3141/1645-21
 29. Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269–271. DOI: 10.1007/BF01386390
 30. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107. DOI: 10.1109/TSSC.1968.300136
 31. Liashenko, A., Globa, L. (2026). Accelerating A* algorithm based search in time-dependent graphs with learned heuristics. *Radioelectronic and Computer Systems*, vol. 2026, no. 1, pp. 179–191. DOI: 10.32620/reks.2026.1.12

32. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. LightGBM: A highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 3146–3154. Available at: <https://papers.nips.cc/paper/2017/file/6907-lightgbm.pdf> (accessed 06 October 2025)
33. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24. DOI: 10.1109/TNNLS.2020.2978386
34. Hamilton, W. L., Ying, R., & Leskovec, J. Inductive representation learning on large graphs. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, pp. 1024–1034. Available at: <https://papers.nips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebe-Abstract.html> (accessed 06 October 2025)
35. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. Graph attention networks. ICLR 2018. DOI: 10.48550/arXiv.1710.10903
36. Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., & Bronstein, M. (2020). Temporal graph networks for deep learning on dynamic graphs. arXiv preprint, arXiv:2006.10637. DOI: 10.48550/arXiv.2006.10637. Available at: <https://arxiv.org/abs/2006.10637>
37. Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, vol. 8, no. 1, 6085. DOI: 10.1038/s41598-018-24271-9
38. Arthur, D., & Vassilvitskii, S. (2007). K-means++: the advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '07)*, pp. 1027–1035. DOI: 10.1145/1283383.1283494

39. Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In: ICLR 2018. DOI: 10.48550/arXiv.1707.01926. Available at: <https://arxiv.org/abs/1707.01926>
40. Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In: Proc. 27th International Joint Conference on Artificial Intelligence (IJCAI 2018), pp. 3634–3640. DOI: 10.24963/ijcai.2018/505. Available at: <https://arxiv.org/abs/1709.04875>
41. Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In: Proc. 28th International Joint Conference on Artificial Intelligence (IJCAI 2019), pp. 1907–1913. DOI: 10.24963/ijcai.2019/264. Available at: <https://arxiv.org/abs/1906.00121>
42. Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, vol. 8, no. 3, pp. 338–353. DOI: 10.1016/S0019-9958(65)90241-X
43. Yonetani, R., Taniai, T., Barekatain, M., Nishimura, M., & Kanezaki, A. (2021). Path Planning using Neural A* Search. In: Proc. 38th International Conference on Machine Learning (ICML 2021). DOI: 10.48550/arXiv.2009.07476. Available at: <https://arxiv.org/abs/2009.07476>
44. Geisberger, R., Sanders, P., Schultes, D., & Delling, D. (2008). Contraction Hierarchies: Faster and Simpler Hierarchical Routing in Road Networks. In: Proc. 7th International Workshop on Experimental Algorithms (WEA 2008), LNCS 5038, pp. 319–333. Springer. DOI: 10.1007/978-3-540-68552-4_24
45. Globa, L., Liashenko, A. Developing the Fuzzy Logic Rules Based on Clustering Algorithms for Data Analysis in the IoT Network. In: *Digital Ecosystems: Interconnecting Advanced Networks with AI Applications. TCSET 2024. Lecture Notes in Electrical Engineering, vol. 1198*. Springer, 2024, pp. 782–803. DOI: 10.1007/978-3-031-61221-3_38

46. Globa, L., Novogrudska, R., Liashenko, A. The clustering and fuzzy logic methods complex for Big Data processing. In: *International Conference on Applied Innovations in IT (ICAIIIT)*, 2022. DOI: 10.25673/76934
47. Liashenko, A., Globa, L. A Comprehensive Method for Anomaly Detection in Complex Dynamic IoT Systems. In: *Proceedings of International Conference on Applied Innovation in IT*, 2025, vol. 13, issue 1, pp. 101–107. DOI: 10.25673/119221
48. Williams, B. M., & Hoel, L. A. (2003). Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results. *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672. DOI: 10.1061/(ASCE)0733-947X(2003)129:6(664)
49. Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96)*, pp. 226–231. AAAI Press.
50. Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. In: *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD '96)*, pp. 103–114. ACM. DOI: 10.1145/233269.233324
51. Ikotun, A. M., Ezugwu, A. E., Abualigah, L., Abuhaija, B., & Heming, J. (2023). K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Information Sciences*, vol. 622, pp. 178–210. DOI: 10.1016/j.ins.2022.11.139
52. Wang, G., Koshy, J., Subramanian, S., Paramasivam, K., Zadeh, M., Narkhede, N., Rao, J., Kreps, J., & Stein, J. (2015). Building a Replicated Logging System with Apache Kafka. *Proceedings of the VLDB Endowment*, vol. 8, no. 12, pp. 1654–1655. DOI: 10.14778/2824032.2824063
53. Dobbelaere, P., & Esmaili, K. S. (2017). Kafka versus RabbitMQ: A comparative study of two industry reference publish/subscribe implementations: Industry Paper. In: *Proceedings of the 11th ACM*

- International Conference on Distributed and Event-Based Systems (DEBS '17)*, pp. 227–238. ACM. DOI: 10.1145/3093742.3093908
54. Work, D. B., Tossavainen, O.-P., Blandin, S., Bayen, A. M., Iwuchukwu, T., & Tracton, K. (2008). An ensemble Kalman filtering approach to highway traffic estimation using GPS enabled mobile devices. In: *Proceedings of the 47th IEEE Conference on Decision and Control (CDC 2008)*, pp. 5062–5068. DOI: 10.1109/CDC.2008.4739016
 55. Silva, T. H., de Melo, P. O. S. V., Viana, A. C., Almeida, J. M., Salles, J., & Loureiro, A. A. F. (2013). Traffic condition is more than colored lines on a map: Characterization of Waze alerts. In: *Proceedings of the 5th International Conference on Social Informatics (SocInfo 2013)*, LNCS 8238, pp. 309–318. Springer. DOI: 10.1007/978-3-319-03260-3_27
 56. Derrow-Pinion, A., She, J., Wong, D., Lange, O., Hester, T., Coop, N., Abelber, T., Berry, J., & Sber, E. (2021). ETA Prediction with Graph Neural Networks in Google Maps. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*, pp. 3767–3776. ACM. DOI: 10.1145/3459637.3481916
 57. MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In: *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297. University of California Press.
 58. Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, vol. 2, pp. 49–55.
 59. Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, vol. 35, pp. 99–109.
 60. Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132. DOI: 10.1109/TSMC.1985.6313399

61. Mamdani, E. H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13. DOI: 10.1016/S0020-7373(75)80002-2
62. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In: *Proc. 5th International Conference on Learning Representations (ICLR 2017)*. DOI: 10.48550/arXiv.1609.02907. Available at: <https://arxiv.org/abs/1609.02907>
63. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1724–1734. DOI: 10.3115/v1/D14-1179
64. Zubaroglu, A., & Atalay, V. (2021). Data stream clustering: A review. *Artificial Intelligence Review*, vol. 54, no. 2, pp. 1201–1236. DOI: 10.1007/s10462-020-09874-x
65. Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). A framework for clustering evolving data streams. In: *Proc. 29th International Conference on Very Large Data Bases (VLDB 2003)*, pp. 81–92. DOI: 10.1016/B978-012722442-8/50016-1
66. Sugeno, M., & Kang, G. T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, vol. 28, no. 1, pp. 15–33. DOI: 10.1016/0165-0114(88)90113-3
67. Jang, J.-S. R. (1993). ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685. DOI: 10.1109/21.256541
68. Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137. DOI: 10.1109/TIT.1982.1056489

69. Sinaga, K. P., & Yang, M.-S. (2020). Unsupervised K-Means Clustering Algorithm. *IEEE Access*, vol. 8, pp. 80716–80727. DOI: 10.1109/ACCESS.2020.2988796
70. Thorndike, R. L. (1953). Who belongs in the family? *Psychometrika*, vol. 18, no. 4, pp. 267–276. DOI: 10.1007/BF02289263
71. Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics*, vol. 3, no. 1, pp. 1–27. DOI: 10.1080/03610927408827101
72. Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 2, pp. 224–227. DOI: 10.1109/TPAMI.1979.4766909
73. Kailath, T. (1967). The divergence and Bhattacharyya distance measures in signal selection. *IEEE Transactions on Communication Technology*, vol. 15, no. 1, pp. 52–60. DOI: 10.1109/TCOM.1967.1089532
74. Brown, R. G. (1959). *Statistical Forecasting for Inventory Control*. McGraw-Hill.
75. Holt, C. C. (2004). Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, vol. 20, no. 1, pp. 5–10. DOI: 10.1016/j.ijforecast.2003.09.015
76. Gardner, E. S. (1985). Exponential smoothing: The state of the art. *Journal of Forecasting*, vol. 4, no. 1, pp. 1–28. DOI: 10.1002/for.3980040103
77. Gardner, E. S. (2006). Exponential smoothing: The state of the art — Part II. *International Journal of Forecasting*, vol. 22, no. 4, pp. 637–666. DOI: 10.1016/j.ijforecast.2006.03.005
78. Cao, F., Ester, M., Qian, W., & Zhou, A. (2006). Density-based clustering over an evolving data stream with noise. *Proceedings of the 2006 SIAM International Conference on Data Mining (SDM)*, pp. 328–339. DOI: 10.1137/1.9781611972764.29

79. Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, vol. 41, no. 1/2, pp. 100–115. DOI: 10.2307/2333009
80. Greenshields, B. D. (1935). A study of traffic capacity. *Highway Research Board Proceedings*, vol. 14, pp. 448–477.
81. Lighthill, M. J., & Whitham, G. B. (1955). On kinematic waves II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A*, vol. 229, no. 1178, pp. 317–345. DOI: 10.1098/rspa.1955.0089
82. Richards, P. I. (1956). Shock waves on the highway. *Operations Research*, vol. 4, no. 1, pp. 42–51. DOI: 10.1287/opre.4.1.42
83. Daganzo, C. F. (1994). The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, vol. 28, no. 4, pp. 269–287. DOI: 10.1016/0191-2615(94)90002-7
84. Treiber, M., & Kesting, A. (2013). *Traffic Flow Dynamics: Data, Models and Simulation*. Springer. DOI: 10.1007/978-3-642-32460-4
85. Herrera, J. C., Work, D. B., Herring, R., Ban, X. J., Jacobson, Q., & Bayen, A. M. (2010). Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment. *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 4, pp. 568–583. DOI: 10.1016/j.trc.2009.10.006
86. Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, vol. 3, no. 3, pp. 32–57. DOI: 10.1080/01969727308546046
87. Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press. DOI: 10.1007/978-1-4757-0450-1
88. Bezdek, J. C., Ehrlich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, vol. 10, no. 2–3, pp. 191–203. DOI: 10.1016/0098-3004(84)90020-7

89. Zheng, Y. (2015). Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, Article 29. DOI: 10.1145/2743025
90. Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, vol. 113, pp. 54–71. DOI: 10.1016/j.neunet.2019.01.012
91. Yin, X., Wu, G., Wei, J., Shen, Y., Qi, H., & Yin, B. (2022). Deep Learning on Traffic Prediction: Methods, Analysis, and Future Directions. *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4927–4943. DOI: 10.1109/TITS.2021.3054840
92. Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45. DOI: 10.1115/1.3662552
93. Okutani, I., & Stephanedes, Y. J. (1984). Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B: Methodological*, vol. 18, no. 1, pp. 1–11. DOI: 10.1016/0191-2615(84)90002-X
94. Wang, Y., & Papageorgiou, M. (2005). Real-time freeway traffic state estimation based on extended Kalman filter: A general approach. *Transportation Research Part B: Methodological*, vol. 39, no. 2, pp. 141–167. DOI: 10.1016/j.trb.2004.03.003
95. Pedrycz, W. (1994). Why triangular membership functions? *Fuzzy Sets and Systems*, vol. 64, no. 1, pp. 21–30. DOI: 10.1016/0165-0114(94)90003-5
96. Wang, L.-X., & Mendel, J. M. (1992). Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427. DOI: 10.1109/21.199466
97. Lughofer, E. (2011). *Evolving Fuzzy Systems — Methodologies, Advanced Concepts and Applications*. Springer. DOI: 10.1007/978-3-642-18087-3

98. Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65. DOI: 10.1016/0377-0427(87)90125-7
99. Arbelaiz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, vol. 46, no. 1, pp. 243–256. DOI: 10.1016/j.patcog.2012.07.021
100. Kreps, J., Narkhede, N., & Rao, J. (2011). Kafka: A distributed messaging system for log processing. *Proceedings of the 6th International Workshop on Networking Meets Databases (NetDB)*.
101. Rahmani, S., Baghbani, A., Bouguila, N., & Patterson, Z. (2023). Graph Neural Networks for Intelligent Transportation Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8846–8885. DOI: 10.1109/TITS.2023.3257759
102. Lv, Y., Duan, Y., Kang, W., Li, Z., & Wang, F.-Y. (2015). Traffic flow prediction with big data: A deep learning approach. *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873. DOI: 10.1109/TITS.2014.2345663
103. Polson, N. G., & Sokolov, V. O. (2017). Deep learning for short-term traffic flow prediction. *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17. DOI: 10.1016/j.trc.2017.02.024
104. Widrow, B., & Hoff, M. E. (1960). Adaptive switching circuits. *IRE WESCON Convention Record*, Part 4, pp. 96–104.
105. Work, D. B., Blandin, S., Tossavainen, O.-P., Piccoli, B., & Bayen, A. M. (2010). A traffic model for velocity data assimilation. *Applied Mathematics Research eXpress*, vol. 2010, no. 1, pp. 1–35. DOI: 10.1093/amrx/abq002
106. Carpenter, G. A., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine.

- Computer Vision, Graphics, and Image Processing*, vol. 37, no. 1, pp. 54–115. DOI: 10.1016/S0734-189X(87)80014-2
107. Work, D. B., Tossavainen, O.-P., Blandin, S., Bayen, A. M., Iwuchukwu, T., & Tracton, K. (2008). An ensemble Kalman filtering approach to highway traffic estimation using GPS enabled mobile devices. *Proc. 47th IEEE Conference on Decision and Control (CDC)*, pp. 5062–5068. DOI: 10.1109/CDC.2008.4739016
 108. Diniz, P. S. R. (2013). *Adaptive Filtering: Algorithms and Practical Implementation* (4th ed.). Springer. DOI: 10.1007/978-1-4614-4106-9
 109. Pedrycz, W., & Gomide, F. (2007). *Fuzzy Systems Engineering: Toward Human-Centric Computing*. Wiley-IEEE Press. DOI: 10.1002/9780470168967
 110. Takens, F. (1981). Detecting strange attractors in turbulence. In: *Dynamical Systems and Turbulence, Warwick 1980*, Lecture Notes in Mathematics, vol. 898, pp. 366–381. Springer. DOI: 10.1007/BFb0091924
 111. Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I. (2013). Discretized streams: Fault-tolerant streaming computation at scale. *Proceedings of the 24th ACM Symposium on Operating Systems Principles (SOSP)*, pp. 423–438. DOI: 10.1145/2517349.2522737
 112. Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 226–231.
 113. Cochran, W. G. (1937). Problems arising in the analysis of a series of similar experiments. *Journal of the Royal Statistical Society*, vol. 4, no. 1, pp. 102–118. DOI: 10.2307/2984123
 114. Carbone, P., Katsifodimos, A., Ewen, S., Markl, V., Haridi, S., & Tzoumas, K. (2015). Apache Flink: Stream and batch processing in

- a single engine. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 36, no. 4, pp. 28–38.
115. Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. *arXiv preprint*, arXiv:1607.06450. DOI: 10.48550/arXiv.1607.06450. Available at: <https://arxiv.org/abs/1607.06450>
 116. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In: *Proc. 3rd International Conference on Learning Representations (ICLR 2015)*. DOI: 10.48550/arXiv.1412.6980. Available at: <https://arxiv.org/abs/1412.6980>
 117. Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. In: *Proc. 5th International Conference on Learning Representations (ICLR 2017)*. DOI: 10.48550/arXiv.1608.03983. Available at: <https://arxiv.org/abs/1608.03983>
 118. Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In: *Advances in Neural Information Processing Systems 29 (NeurIPS 2016)*, pp. 3844–3852. DOI: 10.48550/arXiv.1606.09375. Available at: <https://arxiv.org/abs/1606.09375>
 119. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, vol. 9, no. 8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735
 120. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, pp. 5998–6008. DOI: 10.48550/arXiv.1706.03762. Available at: <https://arxiv.org/abs/1706.03762>
 121. Lewis, J., & Fowler, M. (2014). Microservices: a definition of this new architectural term. Available at: <https://martinfowler.com/articles/microservices.html> (accessed 15.03.2026).

122. Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2017). Microservices: Yesterday, Today, and Tomorrow. In: Mazzara, M., Meyer, B. (eds) *Present and Ulterior Software Engineering*. Springer, pp. 195–216. DOI: 10.1007/978-3-319-67425-4_12
123. Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal*, vol. 2014, no. 239, article 2. Available at: <https://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment>
124. Google LLC (2024). Protocol Buffers: Language Guide (proto3). Available at: <https://protobuf.dev/programming-guides/proto3/> (accessed 15.03.2026).
125. OASIS Standard (2019). MQTT Version 5.0. Edited by A. Banks, E. Briggs, K. Borgendale, R. Gupta. Available at: <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>
126. Hohpe, G., & Woolf, B. (2003). *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison-Wesley. ISBN: 978-0-321-20068-6
127. Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, vol. 207, article 117921. DOI: 10.1016/j.eswa.2022.117921
128. Rudskoy, A. I., Ilin, I. V., & Prokhorov, A. I. (2021). Digital Twins in the Intelligent Transport Systems. *Transportation Research Procedia*, vol. 54, pp. 927–935. DOI: 10.1016/j.trpro.2021.02.152
129. ETSI EN 302 637-2 V1.4.1 (2019). Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. Available at: https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/
130. ETSI EN 302 637-3 V1.3.1 (2019). Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part

- 3: Specifications of Decentralized Environmental Notification Basic Service. Available at: https://www.etsi.org/deliver/etsi_en/302600_302699/30263703/
131. ETSI TS 103 324 V2.1.1 (2019). Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Specification of Collective Perception Service. Available at: https://www.etsi.org/deliver/etsi_ts/103300_103399/103324/
 132. IEEE Std 802.11p-2010 (2010). IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Area Networks – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications – Amendment 6: Wireless Access in Vehicular Environments. Available at: https://standards.ieee.org/standard/802_11p-2010.html
 133. IEEE Std 1609.3-2016 (2016). IEEE Standard for Wireless Access in Vehicular Environments (WAVE) – Networking Services. Available at: https://standards.ieee.org/standard/1609_3-2016.html
 134. 3GPP TS 22.186 V18.2.0 (2024). Service requirements for enhanced V2X scenarios. Available at: <https://www.3gpp.org/DynaReport/22186.htm>
 135. 3GPP TR 22.886 V16.2.0 (2019). Study on enhancement of 3GPP support for 5G V2X services. Available at: <https://www.3gpp.org/DynaReport/22886.htm>
 136. Kaiwartya, O., Abdullah, A. H., Cao, Y., Altameem, A., Prasad, M., Lin, C.-T., & Liu, X. (2016). Internet of Vehicles: Motivation, Layered Architecture, Network Model, Challenges, and Future Aspects. *IEEE Access*, vol. 4, pp. 5356–5373. DOI: 10.1109/ACCESS.2016.2603219
 137. IEEE Std 1609.0-2019 (2019). IEEE Guide for Wireless Access in Vehicular Environments (WAVE) Architecture. DOI: 10.1109/IEEESTD.D.2019.8686445

ДОДАТКИ

Додаток А. Матеріали щодо програмної реалізації та репозиторію проєкту

Розроблений у межах дисертаційного дослідження програмний комплекс *PathMind*, а також пов'язані з ним вихідний код, конфігурація сервісів, граф дорожньої мережі міста Ірпінь, артефакти TGNN-моделі та матеріали для відтворення експериментів розміщено у відкритому репозиторії:

<https://github.com/AndreyLiashenko/PathMind>

Додаток Б. Довідка про апробацію результатів дисертаційної роботи



ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ «ТЕХ УКЛОН»

Ідентифікаційний код ЄДРПОУ 45260393
Прспект Степана Бандери, будинок 20Б, місто Київ, 04073

Вих. № 78
від «02» червня 2026 року

ДОВІДКА про апробацію результатів дисертаційної роботи Ляшенка Андрія Володимировича

Дана довідка підтверджує проведення експериментальних досліджень у ТОВ «Тех Уклон» за тематикою «Моделі та методи адаптивної маршрутизації в телекомунікаційних мережах Інтернету транспортних засобів» з метою підвищення ефективності управління інформаційними потоками в IoV-мережах автономних транспортних засобів в умовах динамічного навантаження. У рамках досліджень були апробовані розроблені телекомунікаційна модель IoV-мережі, метод адаптивної оцінки QoS-стану сегментів та метод пошуку оптимального маршруту обслуговування інформаційних потоків на основі алгоритму A* з часозалежною прогнозною евристикою, а також програмний прототип мікросервісної системи обробки IoT-телеметрії.

Отримані результати підтвердили практичну ефективність запропонованих рішень та можливість їх застосування в реальних умовах обробки телеметрії підключених транспортних засобів.

Розроблені в рамках дисертаційної роботи моделі та методи адаптивної маршрутизації можуть бути рекомендовані до використання при розробці й удосконаленні систем управління інформаційними потоками в IoV-мережах автономних транспортних засобів.

Ця довідка має виключно інформаційний характер та не підтверджує, не встановлює належності Ляшенка А.В. будь-яких майнових прав інтелектуальної власності або інших прав на методи, алгоритми, програмні рішення, дані, системи, технічні рішення чи інші об'єкти, пов'язані з діяльністю ТОВ «Тех Уклон».

З повагою,

Директор ТОВ «ТЕХ-УКЛОН»



Сергій ГРИШКОВ