МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Кваліфікаційна наукова
праця на правах рукопису

КОРОТКОВ СЕРГІЙ СТАНІСЛАВОВИЧ

УДК 004.896

ДИСЕРТАЦІЯ

МЕТОДИКА ПОБУДОВИ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ТРАНСПОРТНОЮ ІНФРАСТРУКТУРОЮ МІСТА НА БАЗІ ТЕОРІЇ S-ГІПЕРМЕРЕЖ

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

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

Науковий керівник: Лащевська Наталія Олександрівна, кандидат технічних наук, доцент

Київ – 2024

АНОТАЦІЯ

Коротков С. С. . Методика побудови системи управління транспортною інфраструктурою міста на базі теорії S-гіпермереж. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 123 – Комп'ютерна інженерія (Галузь знань – 12 Інформаційні технології). – Державний університет інформаційно-комунікаційних технологій Міністерства освіти і науки України, Київ, 2024.

Транспорт має життєво важливе значення для ефективного економічного розвитку та є ключем до забезпечення соціального добробуту населення, забезпечуючи повсякденну мобільність людей та необхідний розподіл товарів. Адекватна інфраструктура є основною попередньою умовою для ефективності транспортних систем. Урбанізація та автомобілізація – основні причини, через які не можуть бути задоволені вимоги міської мобільності.

Підвищення рівня автомобілізації у мегаполісах рідко супроводжувалися відповідним оновленням дорожньої мережі, і ці збільшення, ймовірно, продовжаться й надалі, що ще більше посилить проблему. Реальна цифра "активних" автомобілів в місті Київ, за даними мерії, як мінімум 1,13 млн. одиниць.

Затори на дорогах не дешеві. За оцінками INRIX, такі витрати американських водіїв становлять близько 305 млрд. доларів за рік або 1445 доларів на одного водія. Ці витрати надходять прямо і опосередковано з джерел, таких як втрачений час, паливо та продуктивність.

Отже, математичне моделювання транспортних систем міста із застосуванням теорії S-гіпермерож є **завданням актуальним**.

Метою роботи є підвищення ефективності управління транспортними потоками на основі дослідження та розробки інформаційних технологій для комплексного математичного моделювання транспортних систем міста із застосуванням теорії S-гіпермереж.

Для досягнення поставленої мети необхідно розв’язати наступні **задачі.**

1. Провести аналіз сучасних комп’ютерних технологій проектування та управління складними транспортними мережами.

2. Розробити модель інформаційної системи керування транспортними потоками для підвищення ефективності функціонування транспортної мережі міста.

3. Дослідити методи побудови комп’ютерних мереж для розгортання систем відоспостереження за транспортними потоками міста.

4. Підвищити ефективність функціонування інформаційних систем керування транспортними потоками.

5. Провести обчислювальний експеримент на основі моделювання для перевірки ефективності результатів дослідження.

У дисертації проведено аналіз інфраструктури міста, її математичне моделювання у вигляді S-гіпермережі для основних видів транспорту.

У дослідженнях транспортних потоків застосовуються ідеї, методи та алгоритми нелінійної динаміки та можливостей керування. Їх доцільність обґрунтована наявністю у транспортному потоці стійких та нестійких режимів руху, втрат стійкості при зміні умов руху, нелінійних зворотних зв'язків, необхідності у великій кількості змінних для адекватного опису системи.

Розглянуто актуальність моделювання транспортних потоків, суть основних існуючих моделей та їх можливості.

Виявлено та показано, що теорія S-гіпермереж застосовна для аналізу та синтезу багатьох систем мережевої структури. У тому числі для завдань аналізу міжмережевих структурних взаємодій складних систем транспортної інфраструктури.

Дано математичне формулювання маршрутів та метрик у S-гіпермережах, що дозволяє обчислити віддаленість та відстані за допомогою відомих методів на спеціально побудованих графах, орграфах, гіперграфах та ультраграфах.

Проведено аналіз методів моделювання та управління рухом транспорту через перехрестя. За результатами порівняння автоматичних контролерів, що

базуються на нечіткій логіці, показано виграш за часом проходження автомобілями перехрестя порівняно із звичайними методами, у яких час горіння сигналів світлофора фіксувався.

Розглянуто та доведено можливості визначення пропускної спроможності вулично-дорожньої мережі за допомогою мереж Петрі.

Розглянуто моделювання перехрестя та блоку перехресть з використанням нечіткої логіки в керуванні світлофором. Результати порівняння автоматичних контролерів, що базуються на нечіткій логіці, показали виграш у часі проходження автомобілями перехрестя порівняно із звичайними методами.

Розглянуто способи керування транспортними потоками на світлофорних об'єктах за допомогою розгінних світлофорів. На основі проведеного аналізу виявлено позитивні особливості застосування розгінних світлофорів.

Проведено аналіз основних типів розв'язок. Побудовано математичну модель для кожного типу розв'язок з використанням з використанням теорії S-гіпермереж.

Запропоновано алгоритм побудови оптимальної транспортної розв'язки, при якому суттєво зменшується час проходження даного вузла автотранспортом та знижуються витрати.

Розглянуті особливості алгоритму Форда-Фалкерсона та процедура розстановки позначок для задачі максимального потоку.

Досліджено вплив одностороннього руху на величину транспортного потоку, надано опис рівнів дорожньо-транспортної мережі.

Розроблена концепція імітаційної моделі транспортних потоків та управління транспортними системами на основі теорії S-гіпермереж.

У межах імітаційної моделі проведено моделювання потоків на ділянці дорожньо-транспортної мережі та виявлено, взагалі, позитивний вплив впровадження одностороннього руху.

Розглянуті приклади розрахунку пропускної здатності ділянки дорожньо-транспортної мережі.

Був розроблений алгоритм усунення заторів шляхом зміни режимів роботи світлофорів, що дозволило усунути додаткові затори на бічних дорогах, які

призводили до цього маршруту.

Алгоритм маршрутизації був розроблений, і складена таблиця маршрутизації.

Сформульовано принцип оптимальності. Математична задача маршрутизації зводиться до знаходження найкоротшого шляху в неорієнтованому графі. Були розглянуті різні алгоритми розв'язання задач маршрутизації з використанням теорії S-гіпермережі.

Основні наукові та практичні результати роботи полягають в наступному:

1 Розроблено модель інформаційної системи, наукова новизна якої полягає в тому, що вона грунтується на керуванні транспортними потоками на базі теорії S-гіпермереж, що дозволила підвищити ефективність функціонування транспортної мережі міста.

2 Удосконалено методику побудови комп’ютерної мережі, яка на відміну від існуючих дозволяє мінімізувати кількість точок розміщення відеокамер на заданій території за рахунок застосування наближених алгоритмів.

3 Розроблено методику управління функціонування інформаційних систем керування транспортними потоками наукова новизна якої полягає в тому, що вона грунтується на нечіткій логіці та дозволяє покращити управління інформаційною системою транспортної мережі, за рахунок зменшення часу проходження потоку на перехресті.

Доведена правильність алгоритму FreeRoad. Отримані алгоритми були реалізовані на мові програмування JavaScript. Оскільки алгоритм FreeRoad аналізує лише транспортну мережу на проходження потоку і виявлення заторів, то у випадку, якщо потік не може пройти, для отримання результату, що відображає реальний стан доріг, при вирішенні пов'язаних задач, цей модуль повинен використовуватися в ітеративній взаємодії з іншими.

Ключові слова: нечітка логіка, S-гіпермережі, задачі маршрутизації, мережа, об’єкт інформаційної системи, принцип оптимальності, нелінійна динаміка, комп’ютерне моделювання, теорія графів, алгоритм, інформаційні технології, датчики, інформаційні системи, програмне забезпечення, комп’ютерні мережі.

ANNOTATION

Korotkov S. S. Methodology for developing a system for managing the transport infrastructure of a place based on the theory of S-hyperinterference. – Qualified scientific work as a manuscript.

Dissertation for the degree of Doctor of Philosophy for specialty 123 – Computer Engineering (Galuz knowledge – 12 Information Technologies). – National University of Information and Communication Technologies of the Ministry of Education and Science of Ukraine, Kiev, 2024.

Transport is vitally important for effective economic development and is the key to ensuring the social well-being of the population, ensuring the daily mobility of people and the necessary distribution of goods. Adequate infrastructure is a fundamental factor in the efficiency of transport systems. Urbanization and motorization are the main reasons why people cannot be satisfied with global mobility.

Increases in the level of motoring in megacities have rarely been accompanied by corresponding updates to road surfaces, and the increase will likely continue in the future, further exacerbating the problem. The real number of “active” cars in the city of Kiev, according to the measure, is at least 1.13 million units.

Traffic congestion is not cheap. According to INRIX estimates, such expenditures of American waters amount to approximately 305 billion dollars per river or 1445 dollars per water. What you spend goes directly and indirectly with the elements, such as spending an hour, burning and productivity.

Therefore, mathematical modeling of the city's transport systems using the theory of S-hypergrids is an urgent task.

The aim of the work is to improve the efficiency of traffic flow management based on the research and development of information technologies for complex mathematical modeling of the city's transport systems using the theory of S-hypernetworks.

To achieve the goal, it is necessary to unleash the upcoming tasks.

1. To conduct an analysis of modern computer technologies for designing and managing complex transport networks.

2. Develop a model of the information system for managing traffic flows to improve the efficiency of the city's transport network.

3. Investigate methods of building computer networks for deploying monitoring systems for city traffic flows.

4. Increase the efficiency of the information systems for managing traffic flows.

5. Conduct a computational experiment based on simulation to verify the effectiveness of the research results.

The dissertation analyzes the city's infrastructure, its mathematical modeling in the form of an S-hypernetwork for the main types of transport.

The ideas, methods and algorithms of nonlinear dynamics and control capabilities are applied in the research of traffic flows. Their expediency is justified by the presence of stable and unstable traffic modes in the traffic flow, loss of stability when traffic conditions change, non-linear feedback, and the need for a large number of variables for an adequate description of the system.

The relevance of traffic flow modeling, the essence of the main existing models and their capabilities are considered.

It is revealed and shown that the theory of S-hypernetworks is applicable to the analysis and synthesis of many network structure systems. Including for tasks of analyzing inter-network structural interactions of complex transport infrastructure systems.

The mathematical formulation of routes and metrics in S-hypernetworks is given, which allows calculating remoteness and distances using known methods on specially constructed graphs, digraphs, hypergraphs, and ultragraphs.

An analysis of the methods of modeling and managing traffic through the intersection was carried out. According to the results of the comparison of automatic controllers based on fuzzy logic, the gain in the time of crossing of the intersection by cars is shown compared to conventional methods, in which the burning time of the traffic lights was fixed.

The possibility of determining the bandwidth of UDS using Petri nets is considered and proven.

Modeling of an intersection and a block of intersections using fuzzy logic in traffic light control is considered. The results of the comparison of automatic controllers based on fuzzy logic showed a gain in the time of crossing of the intersection by cars compared to conventional methods.

Ways of controlling traffic flows at traffic lights with the help of speeding traffic lights are considered. On the basis of the conducted analysis, positive features of the use of speeding traffic lights were revealed.

An analysis of the main types of solutions was carried out. A mathematical model was built for each type of solutions using the theory of S-hypernetworks.

An algorithm for the construction of an optimal transport junction is proposed, which significantly reduces the time it takes to pass a given node by motor vehicle and reduces costs.

The features of the Ford-Falkerson algorithm and the procedure for placing marks for the maximum flow problem are considered.

The influence of one-way traffic on the amount of traffic flow is studied, and a description of the levels of the road and transport network is provided.

The concept of the simulation model of transport flows and management of transport systems based on the theory of S-hypernetworks has been developed.

Within the limits of the simulation model, simulation of flows on the section of the road and transport network was carried out and, in general, the positive impact of the introduction of one-way traffic was revealed.

Considered examples of calculating the throughput of a section of the road and transport network.

An algorithm was developed to eliminate traffic jams by changing the operation modes of the traffic lights, which made it possible to eliminate additional traffic jams on the side roads that led to this route.

The routing algorithm was developed, and the routing table was compiled.

The principle of optimality is formulated. The mathematical problem of routing is reduced to finding the shortest path in undirected graph. Various algorithms for solving routing problems using the S-hypernetwork theory were considered.

The main scientific and practical results of the work are as follows:

1 An information system model was developed, the scientific novelty of which is that it is based on the management of traffic flows based on the theory of S-hypernetworks, which made it possible to increase the efficiency of the city's transport network.

2 The method of building a computer network has been improved, which, unlike the existing ones, allows to minimize the number of video camera placement points on a given territory due to the use of approximate algorithms.

3 A methodology for managing the functioning of information systems for managing traffic flows has been developed, the scientific novelty of which is that it is based on fuzzy logic and allows to improve the management of the information system of the transport network, due to the reduction of the flow time at the intersection.

The correctness of the FreeRoad algorithm has been proven. The resulting algorithms were implemented in the JavaScript programming language. Since the FreeRoad algorithm analyzes only the transport network for the passage of the flow and the detection of traffic jams, in the event that the flow cannot pass, in order to obtain a result that reflects the real state of the roads, when solving related problems, this module should be used in iterative interaction with other .

Keywords: fuzzy logic, S-hypernetworks, routing problems, network, information system object, optimality principle, nonlinear dynamics, computer modeling, graph theory, algorithm, information technology, sensors, information systems, software, computer networks.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. С.С. Коротков “Архітектурно-математичне моделювання систем мережної структури з використання теорії S-Гіпермереж”, Телекомунікаційні та інформаційні технології, № 4, с. 64–72, 2023.
2. С.С. Коротков “Імітаційне моделювання керування транспортним потоком із застосуванням S-Гіпермережі”, Звя’зок, №1, с. 31–36, 2024.
3. С.С. Коротков, В.О. Сосновий, О. М. Ткаченко, А. В. Лемешко І. А. Бученко “Проблема маршрутизації для мереж міського транспорту”, Зв’язок, № 4, с. 32–36, 2021.
4. С.С. Коротков, А. О. Барабаш “Проблема семантичних протиріч у великих обсягах даних”, Зв'язок, № 3, с. 31–33, 2022.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. О.М. Ткаченко, Я.І. Торошанко, А.В. Лемешко, В.О. Сосновий, С.С. Коротков. Комп’ютерні мережі: контроль та прогнозування перевантажень. Навчальний посібник. – Київ: ДУТ, 2021. – 77 с.
2. Korofin Denys, Tverdokhleb Arsenii, Lemeshko Andrii Antonenko Artem, Korotkov Serhii, Hunko Oleksandra Stepanchuk Vadym, Sosnovyy Vladyslav, Kovalenko Anna Brovenko Tetiana, Tolok Galina, Tonkykh Oleksii Artem Gorkun, Yushchenko Arsen, Balvak Andrii. "EQUIPMENT OF HOTEL AND RESTAURANT COMPLEXES WITH NETWORK EQUIPMENT". Монографія. Participants of the International Scientific symposium
3. С.С. Коротков, В.С. Засадюк Алгоритм зменшення транзитних потоків транспортної мережі у заданому напрямку. - К.: Міжнародний науково-технічний університет імені академіка Юрія Бугая, “IT Synergy” - 2022 - №2 с. 55-61.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ТС – транспортна система;

ДТМ - дорожньо-транспортна мережа;

РС – розгінний світлофор;

КС – керуючий світлофор;

ПС – пішохідний світлофор;

А, В – напрямки руху на перехресті;

Lр – відстань між стоп – лініями розгінного та керуючого світлофорів;

Lт– відстань від стоп – лінії світлофора до місця перехрестя транспортних потоків.

ВДМ – вулично-дорожня мережа

ВСТУП

Актуальність теми. Транспорт має життєво важливе значення для ефективного економічного розвитку та є ключем до забезпечення соціального добробуту населення, забезпечуючи повсякденну мобільність людей та необхідний розподіл товарів. Адекватна інфраструктура є основною попередньою умовою для ефективності транспортних систем. Урбанізація та автомобілізація – основні причини, через які не можуть бути задоволені вимоги міської мобільності.

Підвищення рівня автомобілізації у мегаполісах рідко супроводжувалися відповідним оновленням дорожньої мережі, і ці збільшення, ймовірно, продовжитсья й надалі, що ще більше посилить проблему. Реальна цифра “активних” автомобілів в місті Київ, за даними мерії, як мінімум 1,13 млн. одиниць.

Затори на дорогах не дешеві. За оцінками INRIX, такі витрати американських водіїв становлять близько 305 млрд. доларів за рік або 1445 доларів на одного водія. Ці витрати надходять прямо і побічно з джерел, таких як втрачений час, паливо та продуктивність.

Вирішення данної проблеми було відобрежено в багатьох науково-дослідних роботах та публікаціях починаючи з кінця 20-го століття. В публікації Karl E. Stoffers, був запропоновани підхід до планування роботи світлофора. В публікації Yuan Gao, Yunchao Qu, було запропонован алгоритм глибокого навчання для реалізації прогнозування часу в дорозі на основі даних DSRC та даних RTMS. В публікації Berthold K. P. Horn, Liang Wang, затухаючої хвилі, що регулює рух транспорту при двосторонньому контролі.

В даній дисертації для подолання цих та інших проблем використовується математичне моделювання транспортних систем міста на базі теорії S-гіпермереж одним із завданням якого є завдання маршрутизації транспорту на міських транспортних мережах, що є суміжною областю з маршрутизацією комп'ютерного трафіку в мережах інформаційних. Головні відмінності полягають у тому, що в першому випадку як пакет розглядається транспортний засіб, а також існують правила дорожнього руху, що обмежують пересування таких пакетів. Таким чином,

завдання маршрутизації транспорту полягає у тій же проблемі знаходження найкоротшого шляху між двома вузлами.

Отже, математичне моделювання транспортних систем міста із застосуванням теорії S-гіпермереж є **завданням актуальним**.

Зв'язок роботи з науковими програмами, планами, темами. Дослідження, результати яких наведені у дисертаційній роботі, попередньо проводились за тематикою науково-дослідних робіт (НДР) кафедри комп'ютерної інженерії Державного університету інформаційно-комунікаційних технологій при виконанні НДР "Розробка моделі оптимізації транспортної мережі за допомогою нейромережевого аналізу" (№18/2100; Др. № 0114U002402).

Мета і задачі дослідження. Метою роботи є підвищення ефективності управління транспортними потоками на основі дослідження та розробки інформаційних технологій для комплексного математичного моделювання транспортних систем міста із застосуванням теорії S-гіпермереж.

Для досягнення поставленої мети необхідно розв'язати наступні **задачі.**

1. Провести аналіз сучасних комп'ютерних технологій проектування та управління складними транспортними мережами.
2. Розробити модель інформаційної системи керування транспортними потоками для підвищення ефективності функціонування транспортної мережі міста.
3. Дослідити методи побудови комп'ютерних мереж для розгортання систем відоспостереження за транспортними потоками міста.
4. Підвищити ефективність функціонування інформаційних систем керування транспортними потоками.
5. Провести обчислювальний експеримент на основі моделювання для перевірки ефективності результатів дослідження.

Об'єктом досліджень є процес управління інформаційною системою транспортної мережі міста.

Предметом досліджень є методи керування транспортними потоками, які базуються на теорії S-гіпермереж.

Методами досліджень, що були запропоновані в дисертаційній роботі є методи керування транспортними потоками, теорії графів, методи системного аналізу та власне сама теорія S-гіпермереж.

Наукова новизна отриманих результатів дисертаційної роботи.

1 Розроблено модель інформаційної системи, наукова новизна якої полягає в тому, що вона грунтується на керуванні транспортними потоками на базі теорії S-гіпермереж, що дозволила підвищити ефективність функціонування транспортної мережі міста.

2 Удосконалено методику побудови комп'ютерної мережі, яка на відміну від існуючих дозволяє мінімізувати кількість точок розміщення відеокамер на заданій території за рахунок застосування наближених алгоритмів.

3 Розроблено методику управління функціонування інформаційних систем керування транспортними потоками наукова новизна якої полягає в тому, що вона грунтується на нечіткій логіці та дозволяє покращити управління інформаційною системою транспортної мережі, за рахунок зменшення часу проходження потоку на перехресті.

Обгрунтованість і достовірність наукових положень та висновків зумовлюється коректністю здійснених у дисертаційній роботі аналітичних досліджень, які базуються на фундаментальних наукових теоріях. Ефективність запропонованої теорії S-гіпермереж та її переваги підтверджені результатами моделювання на ЕОМ.

Наукове значення роботи.

Виявлено та показано, що теорія S-гіпермереж застосовна для аналізу та синтезу багатьох систем мережевої структури, у тому числі для завдань аналізуміжмережевих структурних взаємодій складних систем різної природи. Сформульовано та доведено теорему планарності гіпермережі.

Результати дисертаційної роботи включені у матеріали НДР кафедри комп'ютерної інженерії Державного університету інформаційно-комунікаційних технологій. Матеріали досліджень були впроваджені в навчальний процес ННІІТ в курс дисципліни: " Апаратні та програмні засоби комп'ютерної інженерії".

Особистий внесок здобувача.

Наукові положення та результати дисертації отримані автором самостійно. Особисто автором здійснена розробка загальної концепції дисертації, вибір об'єктів, визначено мету та задачі роботи, обрано та обґрунтовано методи досліджень. В дисертаційній роботі узагальнено результати досліджень, виконаних автором самостійно і опублікованих в співавторстві в роботах [1-4]. У опублікованих у співавторстві роботах автором дисертації зроблено такий особистий внесок. В [1] Розглянуто роботу системи керування маршрутизацією транспорту, проведено огляд теоретичної маршрутизації для інформаційних мереж. Дано опис алгоритмів пошуку найкоротших шляхів на графах. Запропоновано концепцію керування маршрутизацією транспортних засобів для міських мереж. В [2] Розглянуто проблему усунення надмірності семантично близької текстової інформації на основі латентно-семантичного аналізу та одного з алгоритмів нечіткого висновку. Дано опис латентно-семантичного аналізу як способу

виявлення семантичної близькості документів. Сформульовано варіант правил нечіткого висновку щодо розв'язання завдання усунення надмірності семантично близької інформації. Запропоновано оцінити ступінь впливу модуля усунення протиріч на оперативність функціонування інформаційних систем. В [3] Розглянуто основні поняття теорії S-гіпермереж. Показано, що мовою цієї теорії можна описати багато систем мережевої структури з точністю достатньою для вирішення поставленого завдання. В [4] Розглянуто проблему впливу на пропускну здатність мережі запровадження одностороннього руху на певних ділянках вулиць. Запропоновано можливість застосування імітаційного моделювання для визначення правил керування потоками машин у мегаполісі. В основу побудови імітаційної моделі покладено нестаціонарну s-гіпермережу, яка дає можливість цілком адекватно відобразити потоки машин вулицями міста.

Апробація результатів дисертації.

Основні положення і результати дисертації, практичні висновки і рекомендації, які одержані в ході роботи, апробовані та оприлюднені в ході: XII Науково-технічна конференція студентів та молодих вчених «Сучасні

інфокомунікаційні технології», (Київ, ДУТ, 21.05. 2021), Науково-практична конференція «Проблеми комп'ютерної інженерії», (Київ, ДУТ, 02.12. 2020).

Крім того, основні положення і результати дисертації, практичні висновки і рекомендації також апробовано на міжкафедральному семінарі Навчально-наукового інституту інформаційних технологій Державного університету інформаційно комунікаційних технологій.

Публікації.

Основні наукові положення та результати дисертаційного дослідження опубліковано в 4 наукових працях у періодичних виданнях України включених до "Переліку наукових фахових видань України".

Структура і обсяг роботи.

Дисертаційна робота складається з анотації, змісту, переліку умовних скорочень вступу, чотирьох розділів, загальних висновків, списку використаних джерел має 118 сторінки основного тексту, 64 рисунки та 17 таблиць. Список використаних джерел містить 100 найменувань і займає 10 сторінок. Загальний обсяг дисертаційної роботи – 141 сторінка.

РОЗДІЛ 1 СТАН І ПРОБЛЕМИ УПРАВЛІННЯ ТРАНСПОРТНОЮ ІНФРАСТРУКТУРОЮ МІСТА

1.1 Транспортна інфраструктура міста

Транспортна інфраструктура міста як єдина ієрархічна нестаціонарна система складається з елементів зовнішнього та всередині міського транспорту, що взаємодіють між собою. Особливу увагу при проектуванні транспортної системи слід приділяти рівню автоматизації транспортних операцій, який вибирають, виходячи з економічних міркувань. Необхідно враховувати, що з незначних капітальних вкладеннь у транспортну систему вивільняється значна частина допоміжних робочих. Транспортна інфраструктура є інтегральною взаємопов'язаною системою мереж різного виду що належать різним власникам і функціонує внаслідок постійного вдосконалення управління з боку муніципальних та державних органів влади.

Основними елементами транспортної системи (ТС) міста є перехрестя. Розрізняють такі елементи перехресть[7]:

– з рівнозначними дорогами та з різними дорогами;

– планарні, К – рівневі та розподілені;

– з пішохідними переходами (на землі, надземні, підземні) та без пішохідних переходів;

– регульовані та нерегульовані;

– розмічені та нерозмічені;

– однорідні з транспортом та неоднорідні.

Транспортні одиниці різняться на автомобільні та електричні. Автомобільними транспортними одиницями є: автобуси, мікроавтобуси, індивідуальні автомобілі, вантажний транспорт, службовий, швидкісний (з сиреною та мигалкою), спеціальний транспорт. Електричними транспортними одиницями є трамваї, тролейбуси, монорейковий надземний, електропоїзди на естакаді, метрополітен, фунікулер. Елементи транспортної інфраструктури

включають: вулично-дорожню мережу; позавуличну транспортну мережу (наземну, надземну та підземну); мережі зовнішнього (міжміського) транспорту, прокладені через міські планувальні структури; споруди з обслуговування транспортного господарства (парки та депо, вантажні термінали чи станції, енергетичне господарство, вокзали)[16].

Засобами управління рухом на перехресті є: світлофори; інформаційні щити та електронні табло; дорожні знаки та покажчики; дорожня розмітка; смуги розгону та гальмування при перехресті; правила дорожнього руху; індивідуальні інформаційні та навігаційні системи.

Для міської вулиці дорога включає поперечний Перехрестя між зовнішніми межами тротуарів, а також проїзди, рейки (трамваю, метро, електропоїзди та ін.), естакади, мости тощо. Проїжджа частина - елемент дороги або дорожньої споруди, призначений для руху безрейкових транспортних засобів. Під смугою руху розуміють подовжню ділянку проїжджої частини, шириною достатньої для руху автомобілів в один ряд. Кінцевими вузлами смуг є: ділянки проїжджих частин між перехрестями, розгалуженнями та іншими вузлами дорожньої мережі: рейкові колії, лінії руху на дорогах, лінії руху з контактними проводами для тролейбусів, монорейки[1].

Існують такі модельні рівні дорожньо-транспортної мережі(ДТМ)[26]:

а) Локуси або локальні ділянки мережі на смугах руху відповідних місць займаним транспортними одиницями у певний інтервал часу.

б) Маршрути руху транспорту – шлях проходження об'єкта, що враховує напрям руху щодо географічних орієнтирів або координат, із зазначенням початкової, кінцевої та проміжних точок, у разі їх наявності.

в) Графи алелів. Графи алелів мають безліч вершин відповідних Перехрестя стоп-ліній зі смугами руху на перехрестях, розгалуженнях та інших вузлах дорожньої мережі. Ребрам зіставляються ділянки смуг на проїжджій частині або алелі. Алель є безперервною смугою руху між вузлами, а зупинка необхідна з правил дорожнього руху.

г) Граф локусів. Граф локусів є найбільшим за кількістю вершин і ребер серед вторинних мереж S-гіпермережі транспортної системи міста. Для побудови графа алелі розбиваються на ділянки, у кожній з яких може бути трохи більше однієї транспортної одиниці. Вершинами є: можливі зупинки транспорту чи кордону між локусами. Дві вершини будуть суміжні, якщо транспортну одиницю переміщено за один такт у процесі руху.

Основними завданнями моделювання транспортних потоків всередині міста є:

– математичні моделі потоків основних транспортних одиниць усередині міста;

– завдання аналізу транспортних потоків;

– завдання розподілу різних міських транспортних підсистем;

– класифікація, моделювання та оптимізація транспортних розв'язок.

Існують такі моделі та завдання пересування по місту пасажирів та товарів:

– математичні моделі пасажиропотоків на міському транспорті;

– завдання аналізу та оптимізації маршрутів для пасажирів на міському особистому транспорті;

– математичні моделі вантажопотоків та їх оптимізація;

– завдання організації транспортних потоків для забезпечення переїздів.

Завданнями керування транспортними системами є:

– завдання управління потоками транспортних засобів за допомогою розстановки розміток і розв'язок;

– завдання та методи управління потоками за допомогою світлофорів та іншими сигнальними засобами;

– завдання оптимального управління міськими транспортними системами та алгоритми маршрутизації транспортних одиниць;

– моніторинг руху транспорту та пішоходів.

Основними математичними моделями для завдань розміщення пунктів обслуговування транспортних засобів та населення є:

– завдання розміщення бензо-заправок та СТО;

– оптимальне розміщення гаражів та інших місць зберігання транспортних одиниць;

– завдання оптимізації розміщення зупинок, тимчасових стоянок транспортних одиниць;

– розміщення місць обслуговування населення з використанням міського транспорту та інші завдання.

1.2 Моделювання транспортної мережі у вигляді S-гіпермережі

Математична модель повинна поєднувати по можливості всі фактори, що впливають на вигляд та функціонування транспортної системи. Спочатку наведемо визначення простої гіпермережі, в якій елементи у вигляді вузла однотипні, а лінійні елементи (гілки, ребра) мають різну природу.

Шістка, що складається з трьох множин і трьох відображень S=(X,V,R;P,F,W)[36] називається гіпермережею, якщо

$\forall v \in V \mid P(v) \mid = 2$,

$\forall r \in R \mid W(r) \mid = 2$,

$\forall r \in R$ множина $F(r) \subseteq V$ становить маршрут у графі PS = (X, V).

Первинна PS і вторинна WS мережі гіпермережі S є графами, а F відображає ребра WS=(X,R) маршрути графа PS=(X,V). Множина F(r) є маршрутом, тому відображення F єдиним чином визначає відображення W. Кінцеві вершини маршруту F(r) є одночасно кінцями ребра r, тобто гіпермережу S можна задати п'ятіркою (X,V,R;P,F).

У гіпермережі виду S=(Y,V,R) вузол $y \subset Y$ замінюється на граф виду $y = \{x_j^i, E_j\}$ - граф вузла у структурованої гіпермережі, де x_j^i - j-я вершина вторинної мережі WS_i, відображена у вузол у структурованої гіпермережі $SA = (Y, V, G(X_j, R_j))$. Таким чином, на відміну від гіпермереж, вершини вторинних мереж поміщаються у вузли первинної мережі незалежно один від одного, без

обмеження характеру відображення, тобто можливі варіанти відображення декількох вершин однієї вторинної мережі в один вузол первинної мережі.

На, рис 1.1, представлена S-гіпермережа з трьома видами транспортних систем (метро, автобус, трамвай)[46].

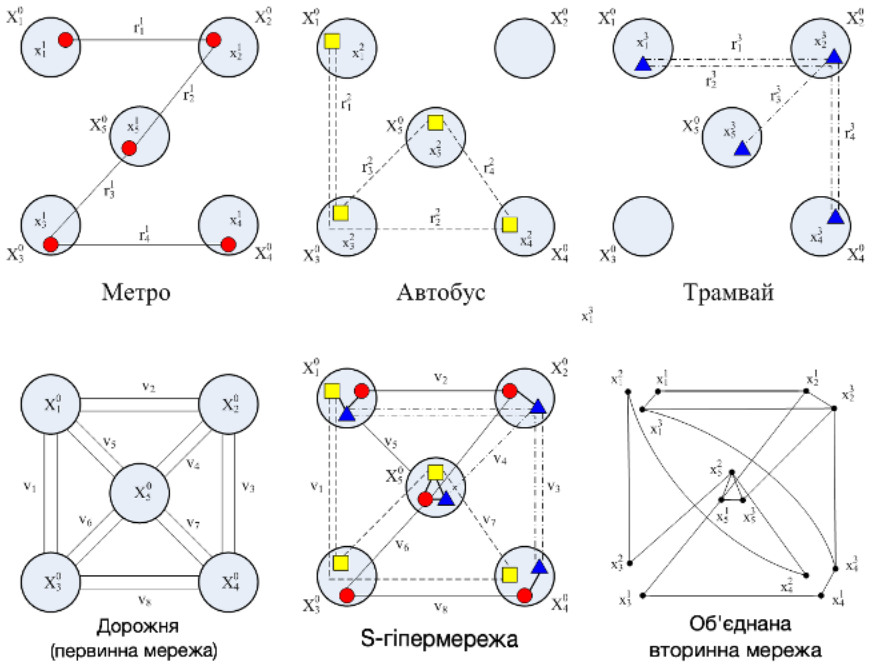


Рисунок 1.1- Приклад об'єднання (підсумовування) всіх вторинних мереж гіпермережі H=(G0(G1,G2,G3))

Дамо формальне визначення S – гіпермережі. Нехай надано множину графів (гіперграфів) $G_0 = (X^0, V), G_1 = (X^1, U^1), \ldots, G_0 = (X^k, U^k)$ і кореневе дерево T0=(Z,R), де $Z = z_0, z_1, \ldots, z_k, R = r_1, \ldots, r_k$ визначальне вкладення графів Gj в Gi(i<j) аналогічно до вкладень, що визначаються в гіпермережах за тим лише винятком, що вершини x_k^i і x_l^j графів Gi та Gj не тотожні, а інцидентні. Очевидно, що одній і тій самій вершині x_k^iможе бути інцидентних кілька вершин $X_k^j = \left\{x_{k_1}^{j_1}, x_{k_2}^{j_2}, \ldots, x_{k_l}^{j_l}\right\}$ із l графів $\{G_{j_s}\}, s = 1, \ldots l$.[56] На множину вершин X_k^j визначається $L^j = (X_k^j, E)$. Вершини $x_{k_j}^{j_i}$ і $x_{k_s}^{j_s}$ суміжні в Lj, якщо відповідні графи G_{j_i} і G_{j_s}у вершині x_k^i мають деякий системоутворюючий зв'язок $l(x^{j_i}, x^{j_s})$. В іншому випадку ці вершини не пов'язані. Також як у гіпермережах ребру $u_i^j \in G_j$ у графі і

Gj зіставляється ланцюг або деяка зв'язкова частина між відповідними вершинами *Gi*. Тут слід зазначити, що системоутворюючі зв'язки типу {l(x,y)}, можуть мати різну природу і, як правило, суттєво залежать від часу. У деяких випадках, наприклад, у системі транспортних мереж різного типу (метро, автобус, трамвай), такими зв'язками у транспортних мульти вузлах будуть тротуарні лінії (піші переходи). У цьому випадку, має сенс розглядати об'єднання всіх вторинних мереж. Однак для деяких завдань є сенс розглядати суму всіх графів гіпермережі H, включаючи первинну мережу *PS,* тобто *G = G0 + G1 + ... + Gn + {Lj}*. На рис 1.2 наведено приклад об'єднання всіх графів, що входять до S-гіпермережі *H[2]*.

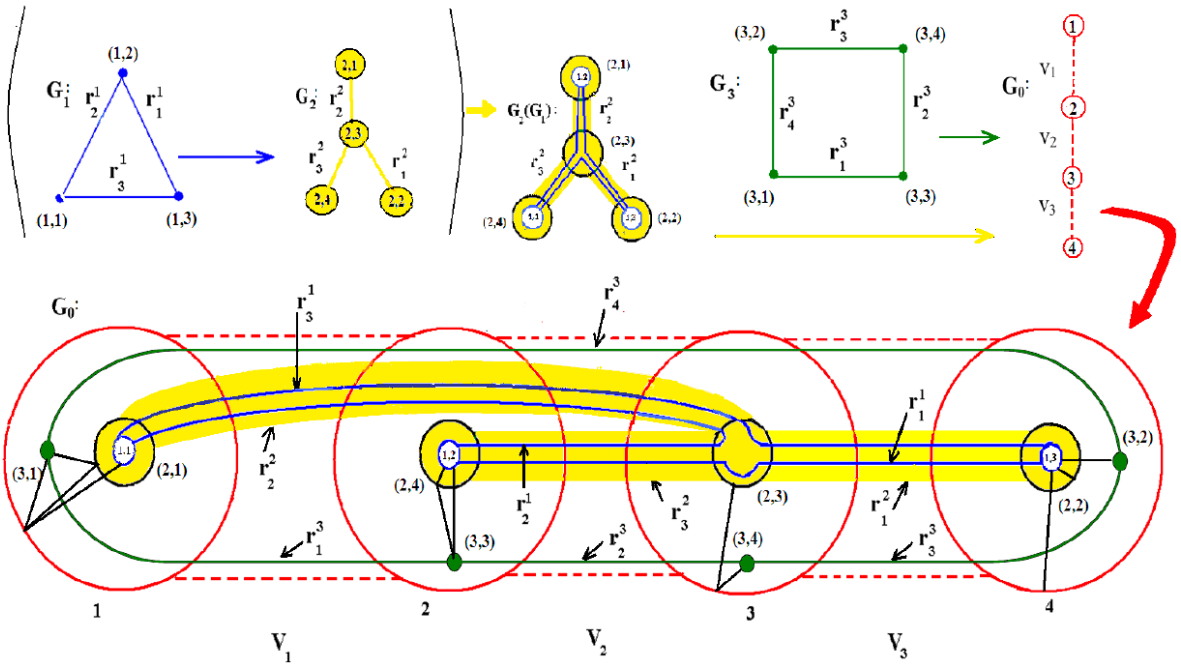


Рисунок 1.2 – Приклад гімермережі

Визначену таким чином S - гіпермережу за допомогою наведених нижче матриць можна встановити з точністю до ізоморфізму і навіть з точністю до нумерації вершин та ребер.

Враховуючи, що через один (пару) полюс можуть проходити кілька ребер вторинних мереж, можна запропонувати дану властивість використовувати для транспортного моделювання. Фрагмент S – гіпермережі[95] для транспортних моделей показано на рис 1.3.

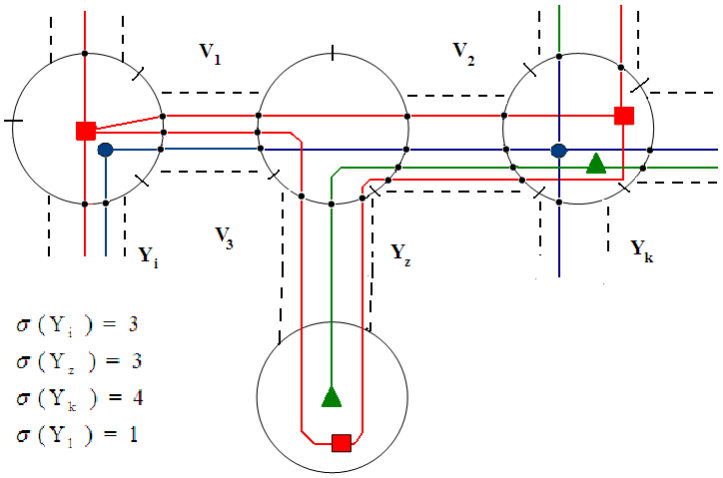


Рисунок 1.3 - Фрагмент S - гіпермережі для транспортних моделей

Вузли *Yi* та *Yk* відповідають транзитно-кінцевим вузлам, а *Yz* цілком транзитний вузол, а вузол *Yl*. – кінцевий σ(Y) – ступінь вершини *Y* у графі *PS*.[66]

Відношення між вторинною мережею з первинною, а також між вторинними мережами, що мають характер типу симбіотичних зв'язків. Інакше кажучи, вони можуть бути корисними, нейтральними чи шкідливими. Дослідження таких взаємовідносин дозволить аналізувати системи різних взаємопов'язаних мереж на предмет їх стійкості, розвитку та

інших характеристик. У таблиці 1.1 наведено таку класифікацію симбіотичних зв'язків .

Таблиця1.1 - Класифікація симбіотичних зв'язків

Типи симбіозу S-гіпермереж PS, WS		Первинна мережа S-гіпермережі, PS = (Y, V)		
		Корисна(К)	Нейтральна(Н)	Шкідлива(Ш)
Вторинна мережа S-гіпермережі, WS=(X,R)	К	Мутуалізм Mutualism	Коменсалізм Commensalism	Паразитизм Parasitism
	Н	Коменсалізм Commensalism	Нейтралізм Neutralism	Аменсалізм Amensalism
	Ш	Паразитизм Parasitism	Аменсалізм Amensalism	Конкуренція Competition

Для кращого розуміння цих взаємовідносин та з урахуванням відображення елементів гіпермережі на поверхні та різних взаємозв'язків інцидентним гілкам ребер вторинних мереж розглянемо чотири таблиці з прикладами різних симбіотичних зв'язків та з урахуванням наступних відображень ребер у гілках первинної мережі:

– Ребра вторинної мережі WS не відображаються в ребра первинної - екс-відображення $WS \overset{\text{екс}}{\rightarrow} PS$.[76]

– Ребра вторинної мережі $WS = (X^1,R)$ йдуть поруч із гілками первинної - пара-відображення $WS \overset{p}{\rightarrow} PS$.

– РебраWS розташовуються на «пласких» гілках первинної мережі – екто-відображення $WS \overset{\text{ек}}{\rightarrow} PS$.

– Ребра вторинних мереж розташовуються всередині гілок первинної мережі – ендо-відображення $WS \overset{\text{ен}}{\rightarrow} PS$.

Наведені приклади досить переконливо показують важливість даних зв'язків для дослідження різних та важливих характеристик комплексів, що складаються з різнотипних систем мережевої структури. Приклади екс-симбіозу, пара-симбіозу, екто-симбіозу та ендо-симбіозу в S-гіпермережі $H=(WS,PS)$ для різних СЗ показані у таблицях 1.2-1.5[3].

Таблиця 1.2 - Приклади екс-симбіозу WS і PS в S-гіпермережі H=(WS,PS) для різних СЗ

Екс-симбіоз		Первинна мережа S-гіпермережі, PS=(Y,V)		
		К	Н	Ш
Вторинна мережа S-гіпермережі - WS=(X,R)	К	Команди одного клубу(Мутуалізм-ес/пп)	Еліта та суспільство (Комменсалізм - ес/пн)	Фінансова піраміда (Паразитизм - ес/пв)
	Н	Соціальне обслуговування населення	Незалежні клуби по інтересам (Нейтралізм - ес/нн)	Інфекційні хворі та суспільство (Аменсалізм - ес/нв)

		(Комменсалізм - ес/нп)		
	Ш	Система жертва – хижак (Паразитизм - ес/вп)	Істоти в агресивному середовищі (Аменсалізм -ес/вн)	Суперники (Конкуренція - ес/вв)

Таблиця 1.3 - Приклади пара-симбіозу WS і PS в S-гіпермережі *H=(WS,PS)* для різних СЗ

Пара-симбіоз		Первинна мережа S-гіпермережі-RS=(Y,V)		
		К	Н	Ш
Вторинна мережа S-гіпермережі - WS=(X,R)	К	Композиційні матеріали(пс/пп)	Лінії електропередач (ЛЕП) та шосе (пс/пн)	Заняття телефонних пар у кабелі передачі даних (пс/пв)
	Н	Система освітлення дороги та шосе (пс/нп)	Лінії та інженерні мережі в одному колекторі (пс/нн)	ЛЕП та низькочастотні канали в кабелі тональної частоти (пс/нв)
	Ш	Зимове гаряче та холодне водопостачання в одному колекторі (пс/вп)	Тротуари та проїжджа частина (пс/вн)	Трамвайні шляхи та дорога з проїжджою частиною (пс/вв)

Таблиця 1.4 - Приклади екто-симбіозу WS і PS S-гіпермережах *H=(WS,PS)* для різних СЗ

Екто-симбіоз		Первинна мережа S-гіпермережі-RS=(Y,V)		
		К	Н	Ш
Вторинна мережа S-гіпермережі - WS=(X,R)	К	Асфальтове покриття бетонних доріг (ек.с/пп)	Маршрути громадського транспорту на дорогах (ек.с/пп)	Потоки вантажного транспорту дорогахек.с/пп)
	Н	Спеціальний транспорт під час ремонту доріг (ек.с/пп)	Смуги руху на проїжджій частині (ек.с/пп)	Ожеледиця на дорогах (ек.с/пп)
	Ш	Захисні покриття доріг, труб тощо. (ек.с/пп)	Потоки транспорту на розбитих дорогах (ек.с/пп)	Рейковий транспорт на автодорогах (ек.с/пп)

Таблиця 1.5 - Приклади ендо-симбіозу WS і PS в S-гіпермережах H=(WS,PS) для різних СЗ

Ендо-симбіоз		Первинна мережа S-гіпермережі - PS=(Y,V)		
		К	Н	Ш
Вторинна мережа S-гіпермережі - WS=(X,R)	К	Мережі електрозв'язку міста (ен.с/пп)	Потік пасажирів у метро (ен.с/пн)	Потоки мін. води у сталевих трубах (ен.с/пв)
	Н	Потік агресивної рідини і в трубах (ен.с/нп)	Кабелі зв'язку в тунелях метро (ен.с/нн)	Зливовий потік у кабельній каналізації (ен.с/нв)
	Ш	Система живлення в автотранспорті (ен.с/вп)	Зледеніння ЛЕП (ен.с/вн)	Віртуальний комп'ютер (ПК в ПК) (ен.с/вв)

Наведені приклади систем мережевої структури показують, що їхня взаємодія може також описуватися симбіотичними зв'язками.

Системні топології показані у таблицях 1.6-1.8[86].

Таблиця 1.6 - Системна типологія первинних та вторинних мереж

Класифікаційні ознаки	Характеристики первинної та вторинних мереж					
Тип математичного об'єкта	Граф		Гіперграф		Гіпермережа	
Орієнтація	Орієнтований		Змішаний		Неорієнтований	
Складність	Тривіальний	Порожній	Дерево	Планарний	Повний	Довільний
Однорідність	Однорідний		Неоднорідний			
Вагові характеристики	Зважений		Незважений			
Атрибути	Детермінована		Випадкова			
Стабільність	Стаціонарна		Нестаціонарна			

Таблиця 1.7 – Системна типологія елементів

Класифікаційні ознаки	Характеристики елементів		
Типы элементов	Вузли (вершини)	Гілки (ребра)	Полюси (напівзв'язок)
Вагові характеристики	Зважений	Незважений	
Орієнтація	Орієнтований	Неорієнтований	
Час існування	Постійно	За розкладом	Випадкове
Подання	Абстрактне	Геометричне	
Число полюсів	Усі полюси	Вільні полюси	

Таблиця1.8 - Системна типологія зв'язків

Класифікаційні ознаки	Характеристики зв'язків			
Спосіб з'єднання	Злиття	Інцидентність	Слабка інцидентність	Суміжність
Орієнтація	Орієнтований		Неорієнтований	
Час існування зв'язку	Постійно	За розкладом	Випадкове	
Надійність з'єднання	З резервуванням		Без резервування	
Зв'язок точкових елементів	Абстрактна (функціональна)		Фізична (геометрична)	
Зв'язок елементів	Екс-вкладення	Пара-вкладення	Екто-вкладення	Ендо-вкладення

Топологічні типології показані у таблицях 1.9-1.12.

Таблиця 1.9 - Топологічна типологія S-гіпермереж

Класифікаційні ознаки	Топологічні характеристики S-гіпермереж		
Розмірність простору	Одновимірне	Двовимірне	Тривимірне
Рід поверхні первинної мережі	Плоска	К-міста	
Число рівнів в ієрархії	Звичайна гіпермережа (два шари)	К-рівнів (шарів)	
Число мереж на К-му рівні	0 рівень – первинна мережа	N(K) число мереж на рівні К	
Розмірність	Кінцева	Нескінченна	
Тип укладання елементів вторинних мереж у первинній	Не укладаються	Укладаються поряд з іншими	Вкладаються на елементі	Вкладаються всередині елемента

Таблиця 1.10 – Топологічна типологія вторинних мереж

Класифікаційні ознаки	Топологічні характеристики вторинних мереж			
Тип математичного об'єкта	Граф	Гіперграф	Ультраграф	Гіпермережа
Наявність циклів	Циклічний		Ациклічний	
Орієнтація	Орієнтований		Неорієнтований	
Рід графа	Плоский		К-го роду	
Зв'язок графів вторинних мереж	Пов'язана (мережа пов'язана з іншими вторинними мережами)		Вільна (мережа пов'язана лише з первинною мережею)	

Таблиця 1.11 – Топологічна типологія елементів

Класифікаційні ознаки	Топологічні характеристики елементів			
Типи елементів	Вузол (вершина)	Ветвь (ребро)	Полюс (напівзв'язок)	
Розмірність	Без розміру	Лінійний	Плоский	Об'ємний
Число полюсів	Без полюсів		К-полюсний	
Представлення	Абстрактне		Геометричне	

Таблиця 1.12 – Топологічна типологія зв'язків

Класифікаційні ознаки	Топологічні характеристики зв'язків		
Орієнтація полюсів	Неорієнтовані	Орієнтовані	
Число полюсів в елементах	Неоднорідні	Однорідні	
Числотипів полюсів	Однотипні	Різнотипні	
Зайнятість полюсів	Зайняті	Вільні	
Тип зв'язку полюсів	Злиття	Інцидентність	Напівінцидентність

1.3 Розроба моделі інформаційної системи

Потрібно здійснити проходження потоку M з пункту S в пункт T за час, що не перевищує Time. Якщо цього зробити неможливо через те, що на дорогах затори, то виявити ці місця та передати на вихід. Швидкість усіх машин усереднена.

Для вирішення задачі використовуються такі дані[85]:

Дорожня мережа міста – дороги, перехрестя (звичайні), перехрестя зі світлофорами, кільця.

Параметри дороги:

а) Pmax(i, j)∈R+ – максимальна пропускна здатність дуги (i, j).

б) Preal(i, j)∈R+ – реальний усереднений у періоді часу Time потік.

в) zerotime(i, j)∈R+– час проходження дуги, якщо у ній немає жодної машини.

3) δ∈(0,1) – світлофорний параметр (для кожного перехрестя свій), позначає, яку частину часу горить світло (відповідно інше світло горить 1-δ часу).

4) Параметри кільця:

а) Circle_number∈Z+ – число вхідних (вихідних) з кільця доріг.

б) Circlemax∈R+ – максимальна пропускна здатність кільця.

в) Circlereal.1, …, Circlereal.Circle_number∈R+ – реальні, усереднені за період часу Time, потоки між 1-ою вихідною дорогою та другою, між другою та третьою, …., між Circle_number та 1-ою.

г) Circletime.1, …, Circletime.Circle_number∈R+ – за аналогією з zerotime(i, j) iCirclereal.

5) Поточні дані:

а) S – точка надсилання потоку.

б) T – точка прийняття потоку.

в) M∈Z+– величина потоку (скільки машин подається до точки S).

г) Time∈R+– час, який не повинен перевищити потік S в T.

В якості моделі використовується орієнтований, зважений мультиграф. Кожна дорога – це дуга графа, кожне перехрестя без світлофора – вершина. Перехрестя зі

світлофором і кільце - це складніші конструкції, які, тим не менш, зводяться до перших двох (дорога, перехрестя без світлофора) таким чином:

Спочатку перехрестя без світлофора – звичайна вершина (за винятком того, що їй приписано параметр δ), яка перетворюється на повний граф *K4*, як показано на рис. 1.4.

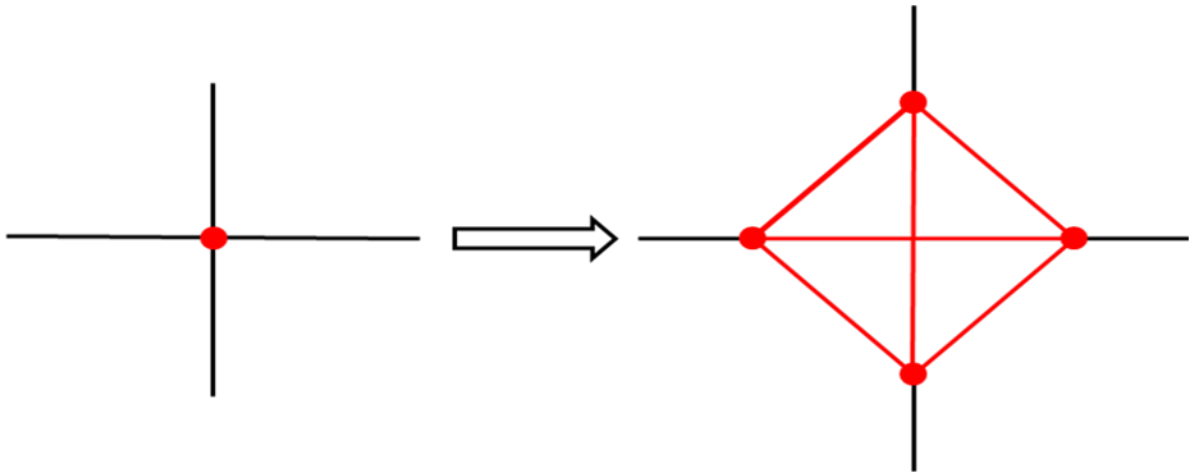


Рисунок 1.4- Перетворення перехрестя зі світлофором

Пропускна здатність *Pmax(i, j)* кожної з цих шести дуг дорівнює мінімуму пропускної здатності інцидентних основних дуг (не червоного кольору). Оскільки світлофор горить δ часу в один бік *i* (1-δ)[22] в інший, то, відповідно, реальна пропускна здатність буде δ**Pmax* та (1-δ)**Pmax*. Пояснення до перетворення перехрестя зі світлофором показано на рис 1.5.

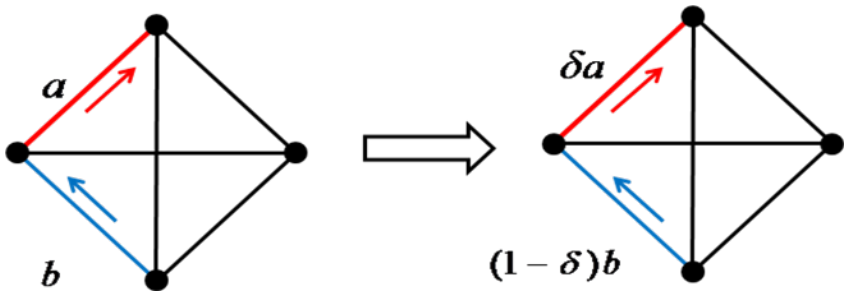


Рисунок 1.5 - Пояснення до перетворення перехрестя зі світлофором

Узагальнений потік *Preal(i, j)* кожної дуги дорівнює 0 (вважатимемо, що машини проходять перехрестя миттєво)[75]. Відповідно *zerotime(i,j)* з попереднього припущення також дорівнює 0.

Нехай дано кільце. Тоді перетворимо його під нашу модель на рис 1.6.

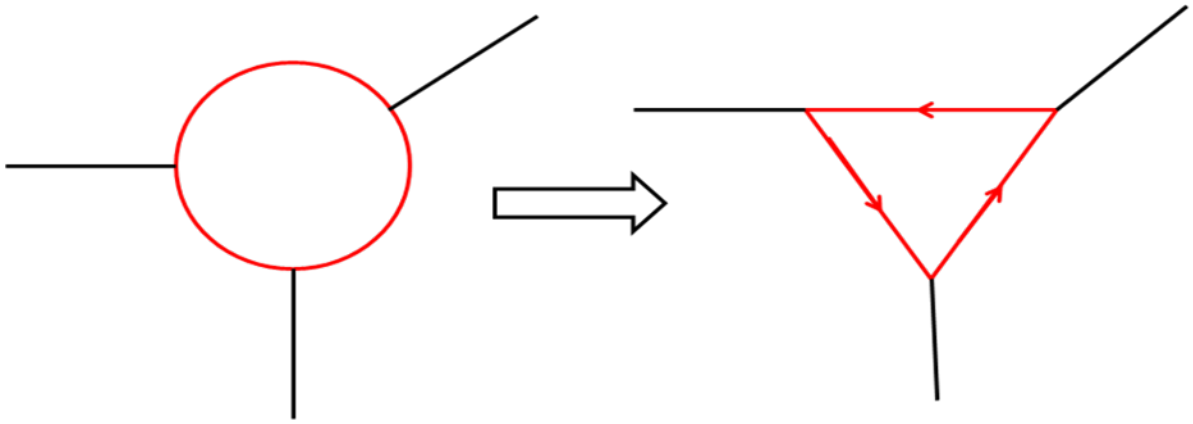


Рисунок 1.6 - Перетворення кільця

В результаті отримаємо структуру в рамках найпростіших вершин і дуг. В якості *Pmax(i, j)* беремо *Circlemax*. В якості *Preal(i,i+1)* і *zerotime(i,i+1)* беремо *Circlereal.i* та *Circletime.i*, де розглядається шлях між *i-ою* та *i+1-ою* дорогами.

Для того, щоб аналізувати рух, нам ще буде потрібна спеціальна функція, яка визначатиме час, витрачений для проїзду одиниці потоку по даній дузі.

Вводиться функція $h(i, j) = h(Pmax(i, j), Preal(i, j), zerotime(i, j)) \in R^+$ – час проходження одиниці потоку цієї ділянки (дуги). Вона отримує на вхід пропускну здатність дуги, усереднений потік і час, який би знадобився одиниці потоку, щоб проїхати цією дугою, якби вона була порожня (не було б інших машин). Ця функція має більш емпіричний характер і може задаватися багатьма способами. При реалізації алгоритму використовується така формула:

$$h(i,j) = (1 + \frac{Preal(i,j)}{Pmax(i,j)} * zerotime(i,j)[96]$$

Для здійснення всіх фаз комплексної обробки інформації при вирішенні поставленої та інших завдань моделювання та управління транспортною інфраструктурою міста може бути теорія S-гіпермереж, що дозволяє розробити

сучасне методологічне та інструментальне забезпечення та технологію обробки інформації на її основі.

1.4 Висновки до розділу 1

Проведено аналіз сучасних комп’ютерних технологій проектування та управління складними транспортними мережами.

У дослідженнях транспортних потоків застосовуються ідеї, методи та алгоритми нелінійної динаміки та можливостей керування. Їх доцільність обґрунтована наявністю у транспортному потоці стійких та нестійких режимів руху, втрат стійкості при зміні умов руху, нелінійних зворотних зв'язків, необхідності у великій кількості змінних для адекватного опису системи.

Розглянуто актуальність моделювання транспортних потоків, суть основних існуючих моделей та їх можливості.

Виявлено та показано, що теорія S-гіпермереж застосовна для аналізу та синтезу багатьох систем мережевої структури. У тому числі для завдань аналізу міжмережевих структурних взаємодій складних систем транспортної інфраструктури.

Дано математичне формулювання маршрутів та метрик у S-гіпермережах, що дозволяє обчислити віддаленість та відстані за допомогою відомих методів на спеціально побудованих графах, орграфах, гіперграфах та ультраграфах.

На основі проведеного аналізу сформульовано мету дослідження та завдання для реалізації поставленої мети.

РОЗДІЛ 2 МЕТОДИ ДОСЛІДЖЕННЯ ТА РІШЕННЯ ЗАВДАНЬ ОПТИМІЗАЦІЇ РУХУ МІСЬКИМ ТРАНСПОРТОМ

2.1 Дослідження методів моделювання та управління рухом транспорту через перехрестя

Управління транспортною інфраструктурою великих міст із застосуванням технологій інтелектуальних транспортних систем дедалі активніше використовується у світовій практиці організації дорожнього руху. Оптимальне регулювання руху транспортних потоків через перехрестя є важливим стратегічним завданням у міському плануванні. Природним способом керування рухом є регулювання роботи світлофора таким чином, щоб черги автомобілів, що утворюються, були якнайменше, і щоб автомобілі, що стоять в черзі, якнайшвидше залишали перехрестя[8].

Проаналізовано основні характеристики транспортних потоків з точки зору утворення заторів. Наводиться визначення затора на регульованих Перехрестя: затор - це стан транспортного потоку, коли тривалість затримки транспортних засобів на перехрестях становить більшу тривалість одного циклу світлофора.

Проаналізовано вирішення проблем стану транспортних потоків на регульованих перехрестях. Існуюча методика розрахунку затримок транспортних засобів на регульованих перехрестях, розроблена М.С. Фішельсон, в сучасних умовах має недоліки, оскільки розраховує затримки лише від одноразової зупинки транспортних засобів та не враховує існування заторів.

2.2 Управління перехрестям у регулярних містах за допомогою мереж Петрі

«Регулярним» містом називається місто, яке має симетрію, принаймні приблизно. Для таких симетричних міст будується мережа Петрі, що описує

синхронізацію між усіма світлофорами міста, заснована на віртуальній циркуляції автомобілів на даній швидкості та простого опису потоків цих автомобілів.

Для оптимізації проведення часу у системі автомобілем потрібно розробляти "зелені хвилі". Якщо правильно вибрати тривалість циклу, то можна розробити чотири системи сумісних зелених хвиль, такі, що автомобіль може проїхати між двома точками міста з запропонованою швидкістю, зустрівши максимум одне червоне світло[15]. Цей результат є дійсним лише тоді, коли немає насичення, коли потоки на всіх вулицях менші, ніж віртуальний потік машин. Припущення про геометричні закономірності у місті менш строгі. Це може бути досягнуто часто шляхом адаптації швидкості на кожній частині вулиці таким чином, що час, необхідний для проходження кожного блоку, залишався б однаковим.

2.3 Макс-Плюс моделювання мереж Петрі

Пояснимо спосіб розрахунку пропускної спроможності досить загального класу мереж Петрі, які можуть бути інтерпретовані в термінах стохастичного контролю.

Безперервна мережа Петрі[18] визначається:

$$N = (P, Q, M, p, m, \tau),$$

де:

P - кінцева множина елементів, які називаються місцями;

Q - являє собою кінцеву множину елементів, які називаються переходами;

$M \in (\mathbb{R})^{P\times Q \cup P\times Q}$ є множниками дуг, що Mpq(відповідно Mqp)[65] означає число дуг від переходу q до місця p (відповідно від місця p до переходу q);

$p: Q \times P \to \mathbb{R}^+$ визначена рівнянням:

$$\sum\nolimits_{q\in p^{int}} \rho_{qp} = 1, \forall p \in P$$

є правилом маршрутизації, яке дає пропорцію рідин, що йдуть від місця p до переходу q по відношенню до кількості рідини, що приходить на місце p;

$m \in (\mathbb{R}^+)^p$- початкові маркування, а саме: mp це кількість рідини p в початковий час;

$t \in (\mathbb{R}^+)^p$- є часом затримки, тобто часом, який молекули рідини мають провести на місці перед від'їздом.

Мережа Петрі показана на рис 2.1.[17]

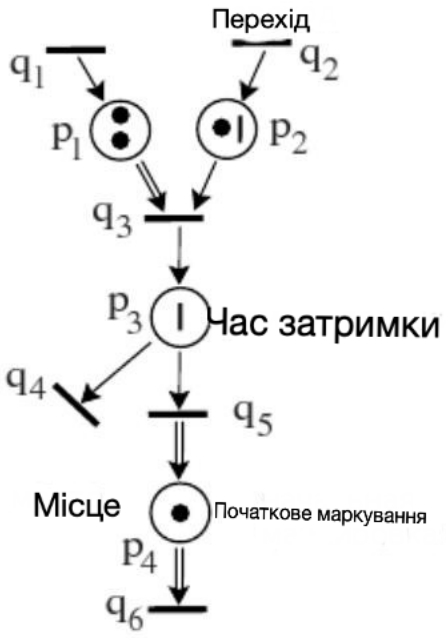


Рисунок 2.1 - Мережа Петрі

Динаміка системи визначається переходами. Перехід відбувається відразу, якщо є достатня кількість щільність потоку, доступного у всіх місцях p $\in q^{in}$ вгорі переходу q (під "доступним" ми маємо на увазі, що рідина провела мінімальний час стоячи на місці). Загальна кількість проведеного q на місці p під час t є $Z_q(t)M_{qp}$(де сума спрацьовувань q позначається Zq). Загальний обсяг щільності потоку, виробленого під час t у $p \in q^{out}$, є $M_{pq}Z_q(t)$[11]. Процес спрацювань йде як найшвидше. Загальний об'єм рідини, який вступив на місце p під час t позначається Zp(t). Визначаємо[27]:

$$\mu_{pq} = M_{pq}, \mu_{qp} = M_{qp}^{-1}, \tilde{\mu}_{qp} = \mu_{qp}\rho_{qp},$$

динаміка системи повністю визначається так:

$$Z_q(t) = \min_{p \in q^{in}} \tilde{\mu}_{qp} Z_p\big(t - \tau_p\big),$$

$$Z_p(t) = m_p + \sum_{q \in p^{in}} \mu_{pq} Z_q(t).$$

Позбавляючись змінної Zp, отримуємо рівняння динамічного програмування, що визначає Zq[47] :

$$Z_q(t) = \min_{p \in q} \left[\tilde{\mu}_{qp} m_p + \sum\nolimits_{q' \in p^{in}} \mu_{pq'} Z_{q'}(t - \tau_p) \right] \quad (2.1)$$

Це рівняння може бути трактоване як рівняння динамічного програмування завдання стохастичного управління з дисконтованою вартістю. За певних умов, описаних у наступній теоремі, ця задача (2.1) є недисконтованою.

Теорема 2.1. Якщо існує $\upsilon \in (\mathbb{R}^+)^Q$:

$\sum_{q \in p^{out}} \upsilon_q \, M_{qp} = \sum_{q' \in p^{in}} M_{pq'} \, \upsilon_{q'}, \forall p \in P,$ (2.2)

Рівняння (2.1) має інтерпретацію недисконтованого стохастичного управління з функцією Беллмана:

$$W_q = \frac{z_q}{\upsilon_q}.$$

Зокрема, умова теореми виконується, коли:

$$\sum\nolimits_{q \in p^{out}} M_{qp} = \sum\nolimits_{\tilde{q} \in p^{in}} M_{p\tilde{q}}, \forall p \in P,$$

тобто, коли для всіх місць існує рівна кількість дуг, що входять і виходять з місця із загальним правилом маршрутизації (тобто $\rho_{qp} = 1/|p^{out}|$, де |A| кінцевої множини A означає його потужність). У цьому випадку ми маємо ν =1(1 – вектор з координатами, рівними 1).

Використовуючи це зауваження, ми можемо визначити пропускну здатність цих мереж Петрі, які є єдиними живими та стабільними мережами Петрі (інші або вибухнуть, або вмирають через кінцевий час).

Теорема2.2. Позначаючи[57]

$$P^p_{qq'} = \nu_q^{-1} \widetilde{\mu_{qp}} \mu_{pq'} \upsilon_{q'}, \nu_q^p = m_p \widetilde{\mu_{qp}}$$

пропускна здатність $\lambda = \frac{\lim_{t \to \infty} Z_q(t)}{t}$ сильно пов'язаної мережі Петрі, задовольняє (2.2), існує, є незалежною від q і є рішенням рівняння динамічного програмування:

$$w_q = \min_{p_\epsilon q^{in}} \left(v^p - \lambda\tau_p + P^p w_q\right), \forall q \in Q.$$

На основі цієї теореми можна обчислити пропускну здатність таких мереж Петрі за допомогою алгоритму Говарда, складність якого експериментально майже лінійна по відношенню до дуг в мережі Петрі.

2.3 Моделювання синхронізації світлофора за допомогою мережі Петрі

Спочатку необхідно зробити припущення про те, що автомобілі є віртуальними, тобто реальні автомобілі не зобов'язані рухатися як віртуальні. Віртуальні автомобілі корисні для дослідження ідеального координування. Потоки реальних машин не можуть бути більшими, ніж потоки віртуальних.

Моделювання перехрестя:

Мережа Петрі[4], пов'язана з перехрестям, наведена на рис 2.2. Позначимо через $x_0(t)$ та $x_1(t)$ загальну кількість зелених фаз, що сталися на кожному з двох світлофорах до моменту t.

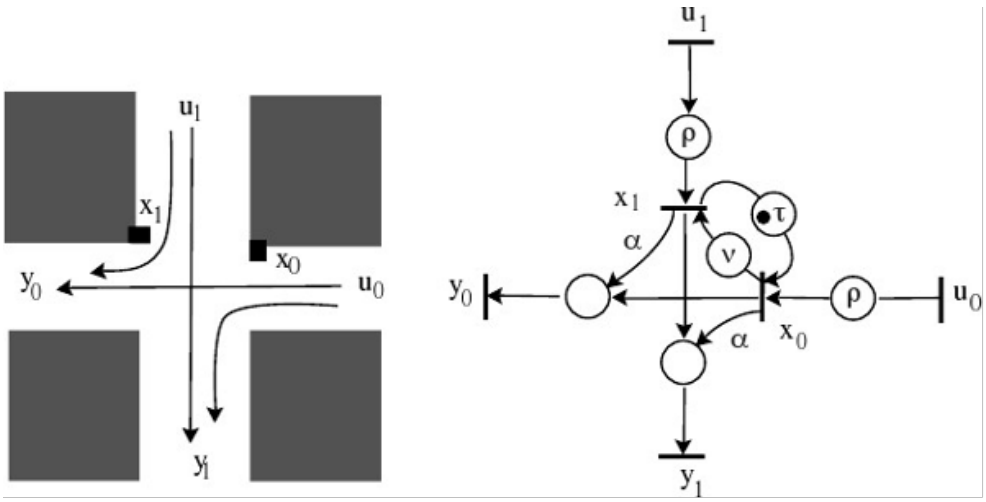


Рисунок 2.2 - Мережа Петрі на перехресті

Тривалості зеленої фази двох світлофорів позначаються відповідно τ і ν. Ми припускаємо, що кількість автомобілів, які можуть Перехрестяати перехрестя, пропорційно до довжини відповідної фази зеленого з коефіцієнтом, який ми вибираємо рівним одиниці. Ми припускаємо, що на кожному перехресті частка

транспортних засобів, що дорівнює, повертає тільки в один бік. Ми позначимо через $u_0(t)$ і $u_1(t)$ загальну кількість легкових автомобілів, що прибули на перехрестя до часу t, $y_0(t)$ і $y_1(t)$ загальна кількість автомобілів, які покинули перехрестя до часу t.

Співвідношення між входами u та виходами y є рівнянням стохастичного динамічного програмування, де функція Беллмана є x:

$x = a \otimes x \oplus b \otimes u$, y = cx , (2.3)

де

$$a = \begin{bmatrix} \varepsilon & \gamma\delta^{\tau} \\ \delta^{\nu} & \varepsilon \end{bmatrix}, b = \begin{bmatrix} \frac{\delta^{\rho}}{\nu} & \varepsilon \\ \varepsilon & \frac{\delta^{\rho}}{\tau} \end{bmatrix}, c = \begin{bmatrix} (1-\alpha)\nu & \alpha\tau \\ \alpha\tau & (1-\alpha)\tau \end{bmatrix},$$[67]

де $\oplus$ позначає minplus додавання матриць , $\otimes$ - minplus множення матриць (заміна складання на мінімум і множення на додавання у звичайному матричному множенні, $\varepsilon = \infty, e = 0$), δ- це одиничний зсув у часі $(\delta\nu(t) = \nu(t) - 1)$ і γ є одиничним зрушенням у нумерації $\left(\gamma\nu(t) = 1 + \nu(t)\right)$.

За допомогою цих позначень у першому рівнянні (2.3)[55] отримуємо: $x_0(t) = min\{1 + x_1(t-\tau), u(t-p)/\nu\}, y_0(t) = (1-\alpha)\nu x_0(t) + \alpha\tau x_1(t)$

Важливо звернути увагу, що добуток матриць y = cx є стандартним. Очевидно ці рівняння динамічного програмування (2.3), не лінійні ні в minplus алгебрі, ні в стандартній. Використання minplus матричного множення є лише зручним та компактним способом написання векторних рівнянь.

Моделювання блоку перехресть:

Розглянемо регулярне місто, зображено на рис 2.5., що складається з розділених вулицями квадратів (з протилежним напрямом руху для послідовних вулиць).

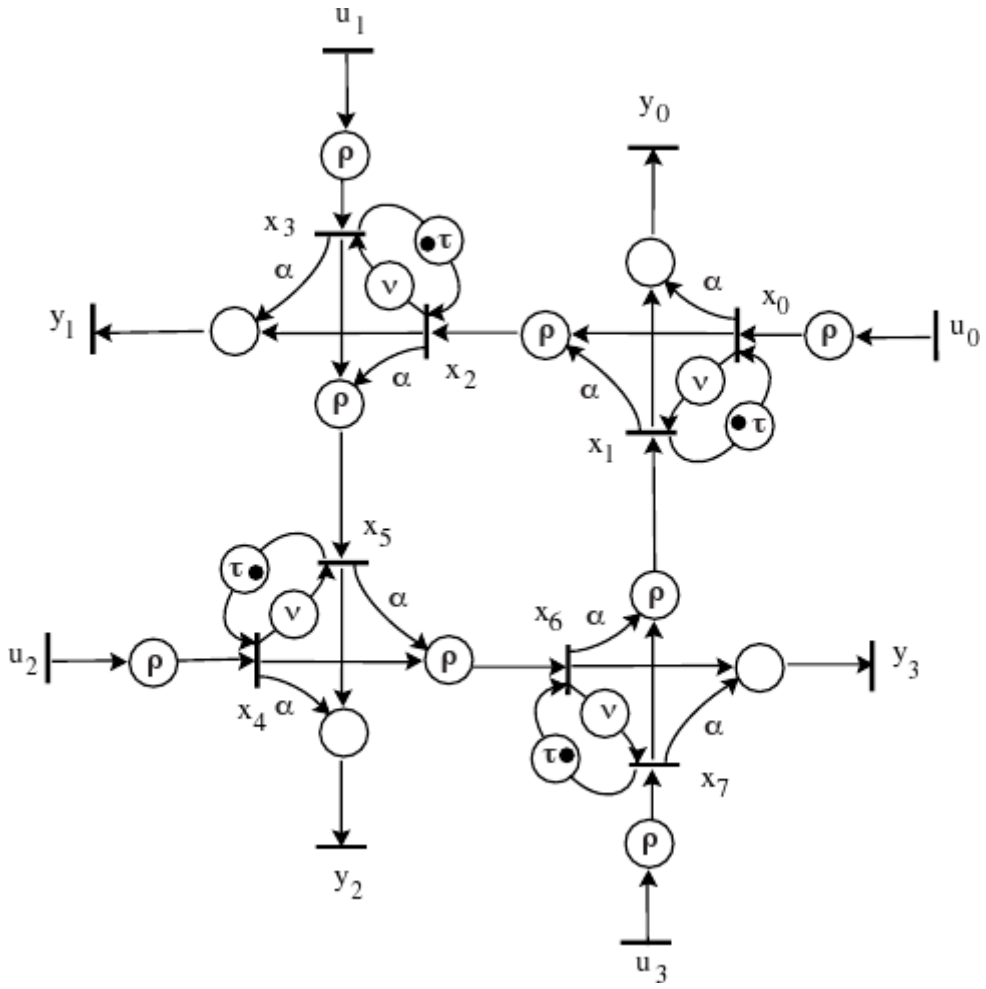


Рисунок 2.3 - Мережа Петрі для блоку з чотирьох перехресть

Для визначення динаміки цієї системи було б корисно спочатку визначити динаміку блоку, що складається з чотирьох перехресть.

Динаміка блоку визначається мережею Петрі, наведеною на рис 2.3.

Відповідні рівняння[45]:

$$\chi_i = a \otimes \chi_i \oplus b \otimes \pi^i \otimes c\chi_{i-1} \oplus b \otimes \pi^{i-1} \otimes E \otimes u_i, y_i = E' \otimes \pi^i \otimes c\chi_i, i = 0, 1, 2, 3,$$

де

$$\chi_i = \begin{bmatrix} x_{2i} \\ x_{2i+1} \end{bmatrix}, \pi^1 = \pi^3 = \begin{bmatrix} e & \epsilon \\ \epsilon & \epsilon \end{bmatrix}, \pi^0 = \pi^2 = \begin{bmatrix} \epsilon & \epsilon \\ \epsilon & e \end{bmatrix}, E = \begin{bmatrix} e \\ e \end{bmatrix},$$

та розрахунок для індексу і був зроблений за модулем 4.

Це система з 8 станами, 4 входами та 4 виходами системи, що можна формально записати, як:

$$x = A \otimes x \oplus B \otimes u, \; y = Cx,$$

де А – це 8 х 8 нелінійний оператор:

$$A = \begin{bmatrix} a & \epsilon & \epsilon & b\pi^0 c \\ b\pi^1 c & a & \epsilon & \epsilon \\ \epsilon & b\pi^0 c & a & \epsilon \\ \epsilon & \epsilon & b\pi^1 c & a \end{bmatrix}.$$

Наявність стаціонарного режиму одночасно гарантує існування недисконтованої інтерпретації стохастичного контролю цього рівняння динамічного програмування[33]. Для цього достатньо, щоб

$$(1-\alpha)\nu + \alpha\tau = \nu, (1-\alpha)\tau + \alpha\tau = \tau$$

де поклали $\nu = \tau$

Справді, $\alpha, \nu\, i\, \tau$ можуть залежати від перехрестя, і в даному випадку достатньою умовою стає:

$$\alpha_{q-1}\tau_{q-1} + \left(1-\alpha_{q-2}\right)\nu_{q-2} = \nu_q, q \text{ парне},$$

$$\alpha_{q-3}\nu_{q-3} + \left(1-\alpha_{q-2}\right)\tau_{q-2} = \tau_q, q \text{ непарне}.$$

Моделювання регулярного міста:

Регулярне місто, рис 2.4, складається із блоків, які ми можемо нумерувати парою (I, J) де I це координата "захід-схід" (W-E) блоку і J координата "південь-північ" (S-N)[77].

Тоді у динаміці повного міста можна написати:

$$x_{IJ} = Ax_{IJ} \oplus \mathcal{A}_0 x_{I+1,J} \oplus \mathcal{A}_1 x_{I,J+1} \oplus \mathcal{A}_2 x_{I-1,J} \oplus \mathcal{A}_3 x_{I,J-1},$$

де

$$\mathcal{A}_0 = \begin{bmatrix} \epsilon & b\pi^1 c & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \end{bmatrix}, \quad \mathcal{A}_1 = \begin{bmatrix} \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & b\pi^0 c & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \end{bmatrix},$$

$$\mathcal{A}_2 = \begin{bmatrix} \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & b\pi^1 c \\ \epsilon & \epsilon & \epsilon & \epsilon \end{bmatrix}, \quad \mathcal{A}_4 = \begin{bmatrix} \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ \epsilon & \epsilon & \epsilon & \epsilon \\ b\pi^0 c & \epsilon & \epsilon & \epsilon \end{bmatrix}.$$

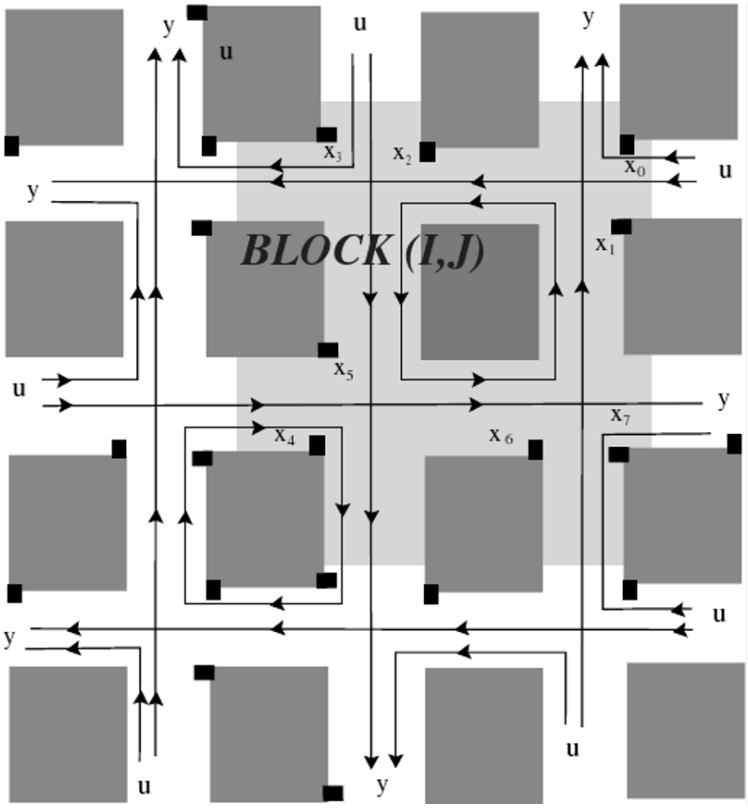


Рисунок 2.4 - Регулярне місто

Як було показано вище, попередній тип системи міг бути вирішений ефективно. Але для цієї системи доведеться оптимізувати деякі параметри. Наприклад, ми повинні провести початкове позначення у місцях, що відповідають вулиці, час горіння зеленого та червоного світла. Коли немає насичення, можна підібрати досить хороші параметри та отримати систему зелених хвиль, яка дозволяє проїзд між двома точками у місті з максимум одним червоним світлом.

Мінуси цієї моделі очевидні:

Вона адекватна лише в симетричних містах, при невеликої щільності потоку. Крім того, потрібно вручну регулювати безліч параметрів. Що робить її поки що невідповідною до регулювання перехрестя в загальному випадку[24].

2.3 Використання нечіткої логіки в керуванні світлофором

У системах, заснованих на нечіткій логіці, є контролер, який керує часом горіння світлофора для того, щоб забезпечити рух трафіку з мінімальною

затримкою. У цих системах також використовуються датчики, розміщені на перехрестях, сигналізують контролеру про прибуття транспорту, а також про покидання транспортом перехрестя. Робота світлофорів ґрунтується на кількох принципах. Етап, коли трафік деяким набором смуг залишає перехрестя. На рис. 2.5[25] показано чотири типові фази для перехрестя. Завершення всіх фаз називається циклом. Довжина циклу визначається довжиною фаз циклу. Зазвичай довжина циклу наперед визначена і є фіксованою величиною. Однак, у динамічних методах довжина циклу може змінюватись залежно від стану руху. Колір світлофора показує початок чи кінець фази. Зелене світло позначає активну фазу, жовте світло вказує на закінчення фази і червоний колір означає завершення поточної фази і початок наступної фази. Зазвичай фази змінюються одна одною в кожному циклі.

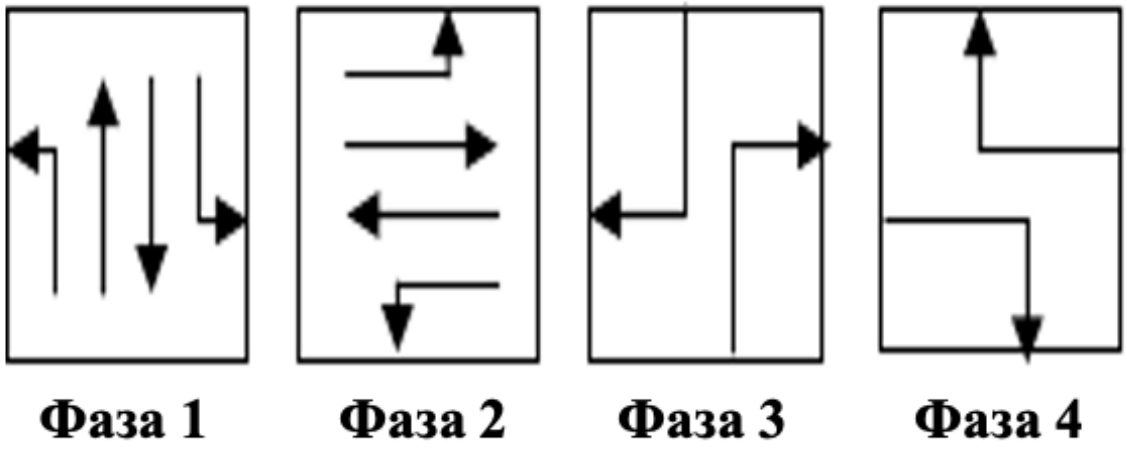


Рисунок 2.5 - Фази ізольованого перехрестя із чотирма дорогами

Диспетчер Нечіткої Логіки дивиться на стан при інших фазах світлофора. Нечітка логіка дозволяє використання підходу подібного до людського, щоб прийняти рішення, які система неможе прийняти, що базується на кількісті даних. Наприклад, "ЯКЩО БАГАТО автомобілів прибуло, І МАЛО ЗЕЛЕНОГО СВІТЛА, то ПРОДОВЖИТИ ЗЕЛЕНИЙ" - лінгвістичний підхід, що використовується в нечіткій логіці[87].

2.4 Модель нечіткої логіки для ізольованого перехрестя

Розглянемо ізольоване перехрестя з чотирма під'їзними шляхами, як показано на рис 2.6. Кожен підхід має 3 смуги вхідних та 3 вихідних. У цій моделі чотири фази[6]:

– автомобілі рухаються з півночі на південь (прямо), з півночі на схід (поворот), з півдня на північ (прямо) та з півдня на захід (поворот);

– із заходу на схід (прямо), із заходу на північ (поворот), зі сходу на захід (прямо) та зі сходу на південь (поворот);

– з півночі на захід (поворот) та з півдня на схід (поворот);

– із заходу на південь (поворот) та зі сходу на північ (поворот).

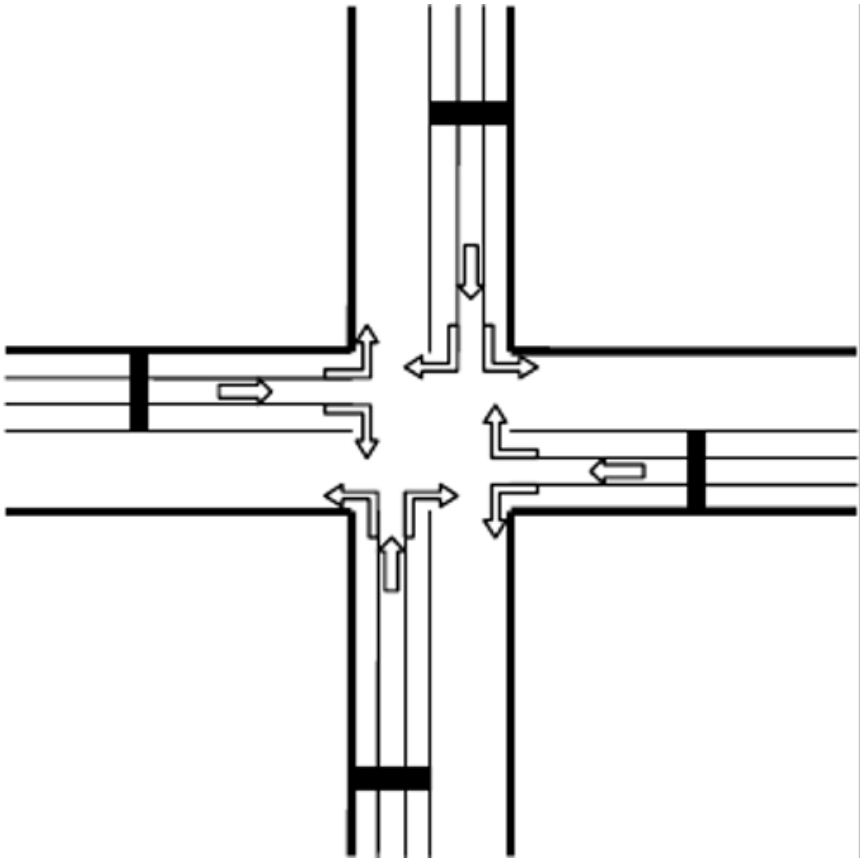


Рисунок 2.6 - Ізольоване перехрестя

В якості вхідного вектора до диспетчера нечіткої логіки використовується чотири параметри. Перший – довжина черги на червоній фазі (RQL)[97], яка є

залишком транспортних засобів, починаючи з останнього зеленого сигналу плюс прибуття протягом поточного червоного сигналу:

$RQL = Q_{gr} + \sum Vr(i)$ (2.4)

де Q_{gr}- число транспортних засобів, які не залишили перехрестя за фазу зеленого сигналу, і $\sum Vr(i)$ - сума транспортних засобів, що прибули, в і-ту секунду червоної фази.

Другий параметр (RGT) - відношення часу, горіння зеленого сигналу, що залишився, до загальної тривалості горіння зеленого сигналу.

$RGT = \frac{q(t)}{T_g}$, (2.5)[9]

Де T_g - повний час для зеленої фази і g(t) - час, що залишився в момент t від початку зеленої фази.

Третій параметр - прибуття транспортних засобів протягом горіння зеленого сигналу(AG), де $\sum Vg(i)$ - число транспортних засобів, що прибули в і-ю зеленої фази.

$AG = \sum Vg(i)$. (2.6)

Нарешті, четвертий критичний параметр цієї моделі – середній час розвантаження зеленого світла (AGDT). Він вказує, чи транспортні засоби залишаються у межах нормального потоку чи ні.

$AGDT(n) = \alpha(CDT) + (1 - \alpha)AGDT(n - 1)$, (2.7)

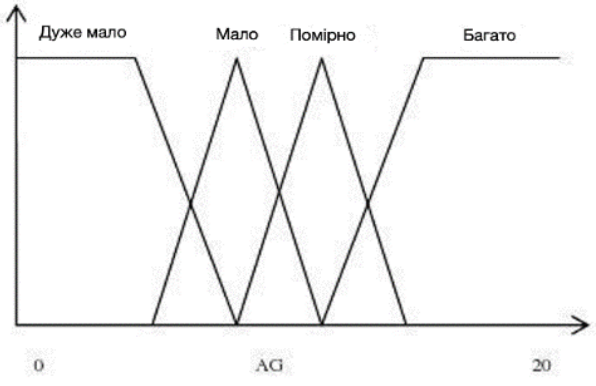
де CDT - поточний розвантажувальний час залишення перехрестя автомобілями, AGDT(n-1) – попередній час, що вирахували.

Затримка = момент залишення перехрестя транспортним засобом – момент прибуття на перехрестя (2.8)

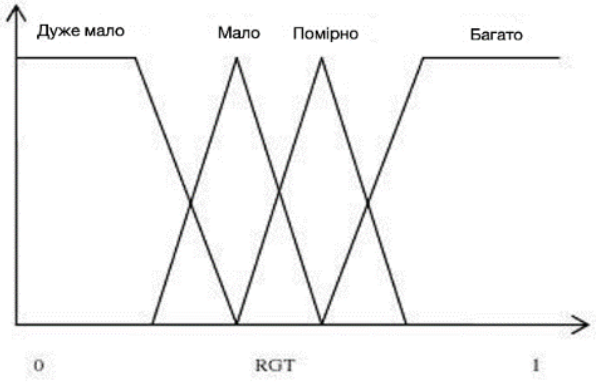
Нечіткі правила нормального диспетчера. Диспетчер бере три вхідні параметри. Перші два – AG та RGT[38], які беруться з поточної зеленої фази. Третій параметр – RQL, який береться з дорожніх смуг на червоній фазі. Він порівнює ці параметри, щоб вирішити, чи розширити чи зменшити час горіння зеленого сигналу поточної фази. Вихідними даними є ступінь зміни часу горіння зеленого сигналу. Нечіткі набори показуються рисунку 2.7 (b, c).

Якщо черга на червоній фазі є довгою та прибуття автомобілів на зеленій фазі невелике, то логічно поточну зелену фазу потрібно закінчити таким чином, щоб диспетчер може перейти на наступну фазу. Наприклад, нечітке правило може бути заявлене подібно до цього:

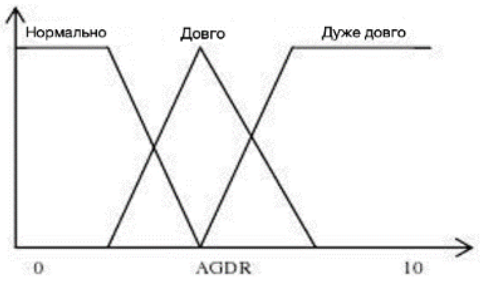
Якщо RQL багато та AG дуже мало і RGT багато, то Розрахунок часу Зменшено[48].



(b) AG



(c) RGT



(d) AGDT

Рисунок 2.7 - Нечіткі набори

Рис 2.8 показує нечіткі набори вибору тривалості сигналу.

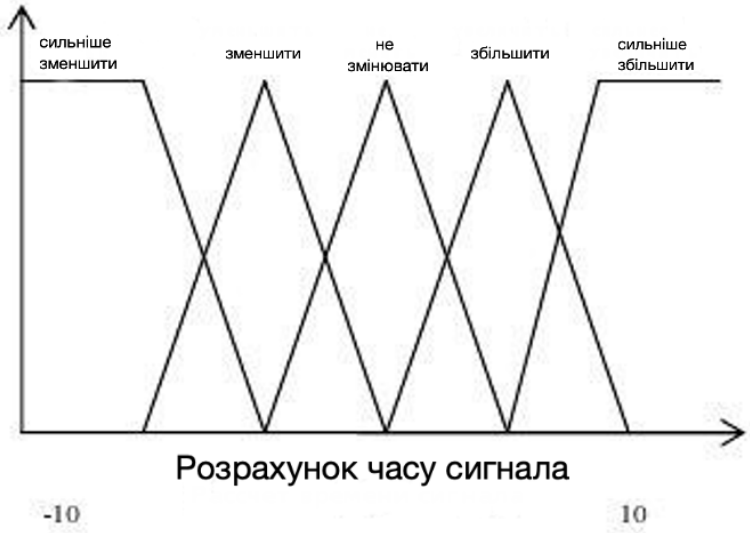


Рисунок 2.8 - Нечіткий розрахунок часу сигналу

Нечіткі правила анормального диспетчера. Цей диспетчер подібний до нормального, але він має додатковий параметр AGDT. AGDT вказує, скільки часу в середньому автомобілю потрібно залишити перехрестя за зелену фазу. Якщо AGDT зростає, то є можливість, що час горіння зеленого сигналу буде збільшено. Наприклад, нечітке правило для цього модуля може бути заявлено так: якщо RQL мало і AG помірно і RGT багато і AGDT дуже довгий, то розрахунок часу зменшено.

Рис 2.7(d) ілюструє нечіткі множини для AGDT.[94]

Ця система може бути розширена на групу перехресть. Для цього диспетчери додатково проводять синхронізацію зі своїми сусідами.

Результати порівняння автоматичних контролерів, заснованих на нечіткій логіці, показали виграш у часі проходження автомобілями перехрестя порівняно із звичайними методами, у яких час горіння сигналів світлофора фіксувався. Однак для реалізації такої моделі в місті необхідно зазнати солідних грошових збитків на встановлення контролерів, датчиків та їх з'єднання в мережу. Однак це не є перешкодою для комп'ютерного моделювання.

2.5 Завдання управління світлофорами на перехрестях будь-якої конфігурації

Спосіб відноситься до управління рухом транспортних потоків і може бути використаний у різних транспортних системах. Технічний результат полягає у підвищенні пропускної спроможності у місцях Перехрестя потоків (у тому числі транспорт – пішоходи), обладнаних світлофорами. Спосіб полягає в тому, що перед світлофорами, що здійснюють регулювання транспортних потоків, встановлюють так звані розгінні світлофори - РС. За допомогою РС транспортний потік розганяється і стоп - лінію світлофора, що дозволяє виїзд на перехрестя, проходить вже зі значною швидкістю. Це дозволяє збільшити обсяг транспортного потоку через світлофорні об’єкти.

Для досягнення зазначеного технічного результату додамо до управляючого світлофора (УС)[58] додатково розгінний світлофор (РС), який встановимо на відстані Lр перед ним. На дозвільний сигнал РС транспортний засіб починає прискорюватися і коли він долає відстань Lр, через час Тр подається сигнал на УС. Транспортний засіб зі швидкістю Перехрестяає стоп – лінію УС та здійснює подальший рух. У свою чергу, забороняючий сигнал на УС подається через Тз після його надходження на РС. При цьому Тз< Тр. Тому за однакової тривалості перехідних сигналів на РС і УС час дозволяючого сигналу РС більше, ніж УС. Для того, щоб при необхідності транспортний засіб міг запобігти зіткненню з транспортним засобом, який своєчасно не звільнив смугу руху або не здійснив наїзд на пішохода, він повинен мати можливість загальмувати. З огляду на це УС має бути розташований з відривом Lт від перехрестя чи пішохідного переходу для світлофорного пішохідного переходу поза перехрестям. Величини Lр, Lт, Тз і Тр визначаються для кожного транспортного потоку. На їхню величину впливає склад транспортного потоку чи неможливість зробити величини Lр, Lт більше якоїсь величини через малу відстань між сусідніми світлофорними об'єктами чи через інші причини. Отримані шляхом моделювання значення: Lр = Lт = 8м,Тр = 3,5 сек.,

Тз= 0,6 сек. Звідки видно, що зелене світло на РС горить більше, ніж на УС на Тр-Тз = 3,5 сек. - 0,6 сек. = 2,9 сек.

Lp = Lт з міркування того, що транспортний засіб повинен мати можливість загальмувати перед перехрестям, якщо при Перехрестяі стоп – лінії УС водій транспортного засобу бачить, що перехрестя зайняте, а ділянка розгону зазвичай довша за гальмівну колію. Тз трохи більше інтервалу часу, протягом якого транспортний засіб подолає відстань від стоп – лінії РС до стоп – лінії УС. Це зроблено для того, щоб транспортний засіб, що проскочив РС, Перехрестяав стоп – лінії УС «у більш ранній стадії перед загорянням червоного» і встигало перетнути перехрестя.

Об'єктна модель розгінного світлофора показано на рисунку 2.9.

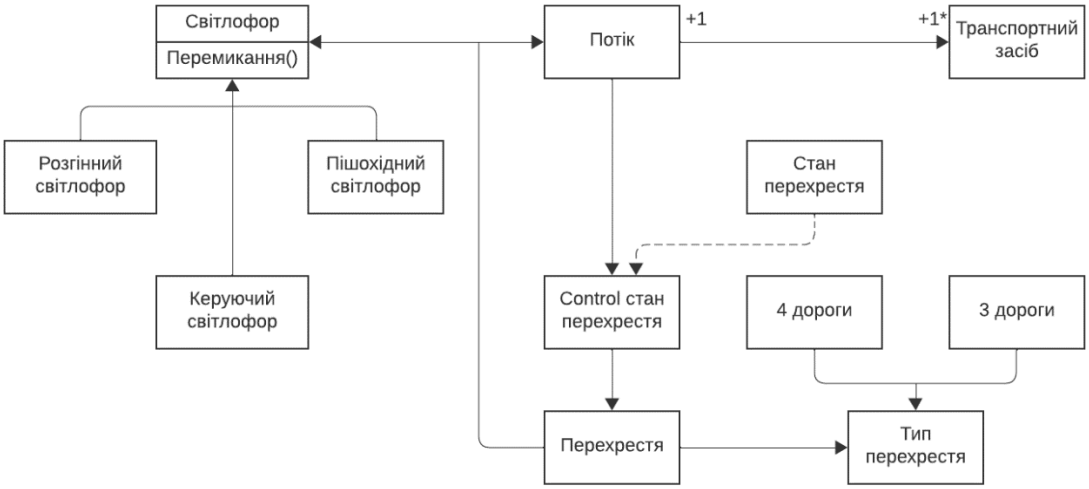


Рисунок 2.9 - Об'єктна модель розгінного світлофора

На рисунку 2.10 представлена структура світлофорного циклу з двома фазами з одним проміжним тактом у кожній фазі, що пояснює принцип роботи запропонованого способу управління пар світлофорів РС та КС для рівнозначного перехрестя.

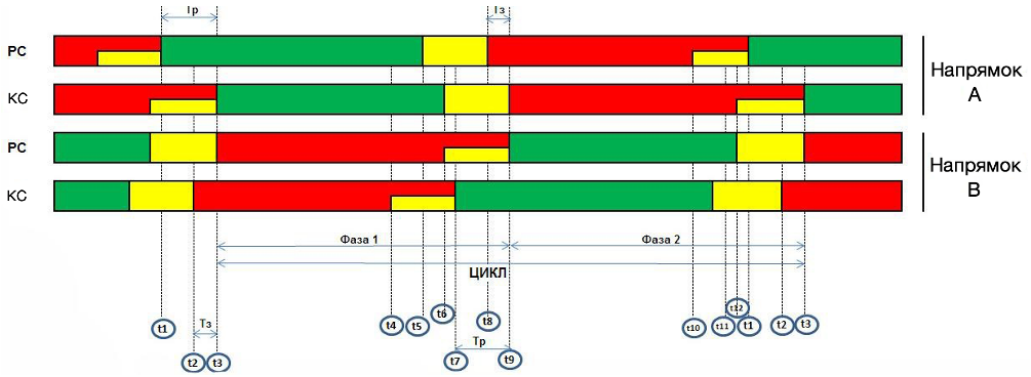


Рисунок 2.10 - Структура світлофорного циклу

На рисунку 2.10 прийняті такі позначення:

t1, t2, t3,… t12 [68]моменти часу, між якими сигнали РС і КС постійні, Тр – час розгону транспортного засобу перед Перехрестяом ним стоп – лінії світлофора, що управляє, Тз – час затримки надходження забороняючого сигналу на керуючий світлофор після його надходження на розгінний світлофор.

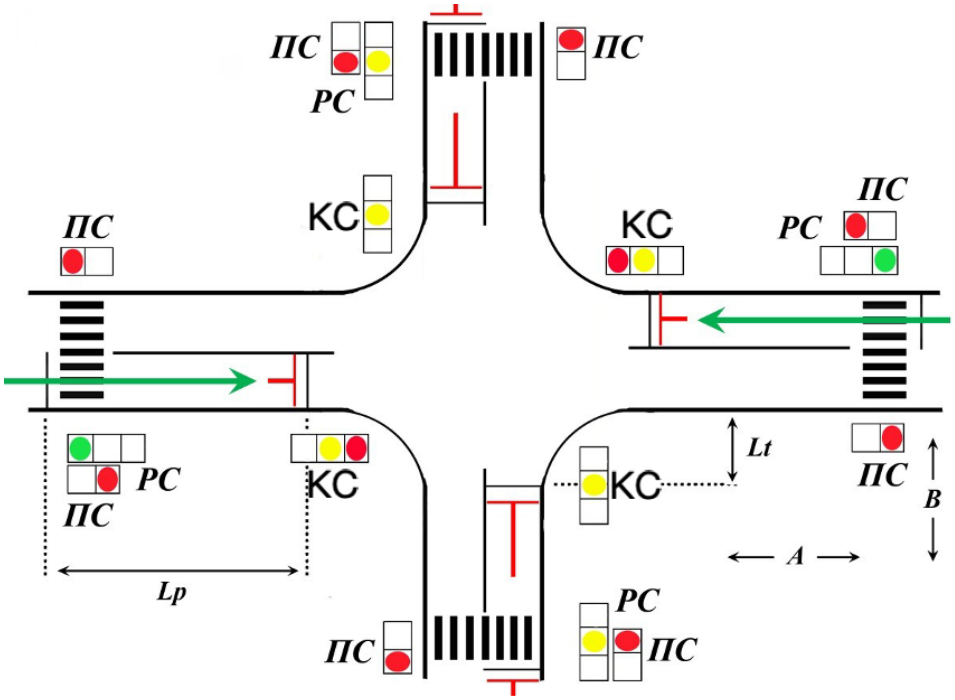


Рисунок 2.11- Рівнозначне перехрестя з встановленими на ньому світлофорами

На рис 2.11 зобоажено рівнозначне перехрестя з встановленими на ньому світлофорами та прийняті такі позначення:

– РС – розгінний світлофор,

– КС – керуючий світлофор,

– ПС – пішохідний світлофор,

– А, В – напрямки руху на перехресті,

– Lр – відстань між стоп – лініями розгінного та керуючого світлофорів,

– Lт– відстань від стоп – лінії світлофора до місця Перехрестя транспортних потоків.

У наступних рисунках зображено перехрестя в інтервалах часу: t1 – t2,t2 – t3,t3 – t4, … t12 – t1.

Наведені рисунки пояснюють принцип роботи запропонованого способу керування транспортними потоками на світлофорних об'єктах за допомогою розгінних світлофорів (Рисунки 2.12-2.23).

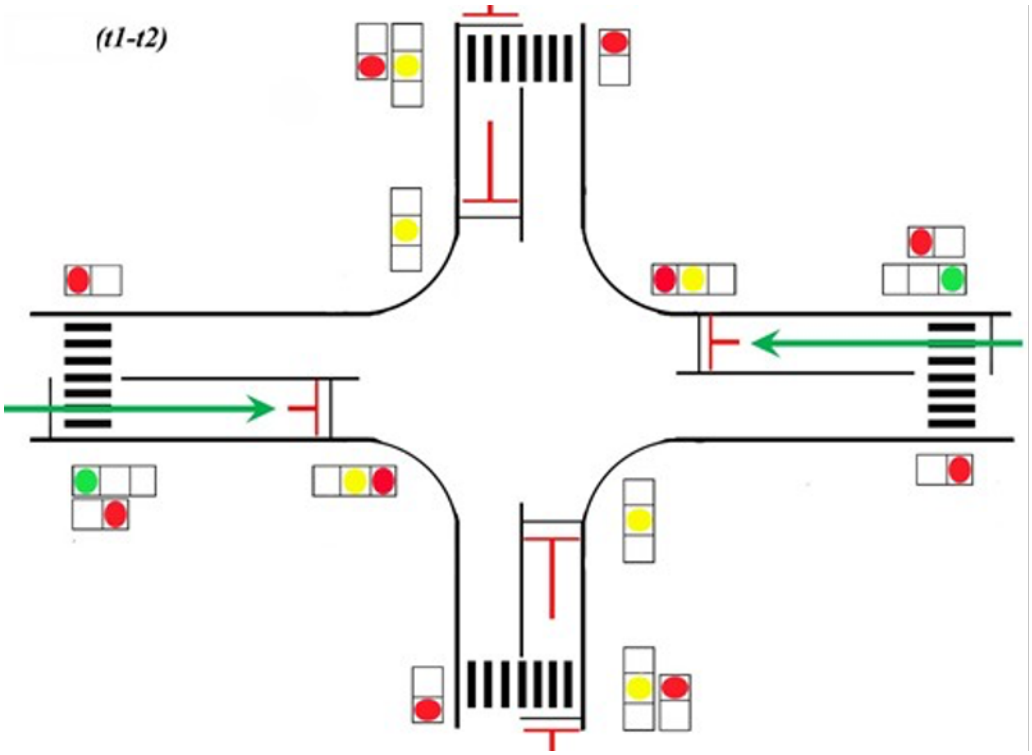


Рисунок 2.12 - В інтервалі часу t1–t2

В інтервалі часу t1–t2[78] транспортні засоби потоків А пришвидшуються. На РС напрямів А спалахнув зелений. Пішохідний перехід скрізь заборонено.

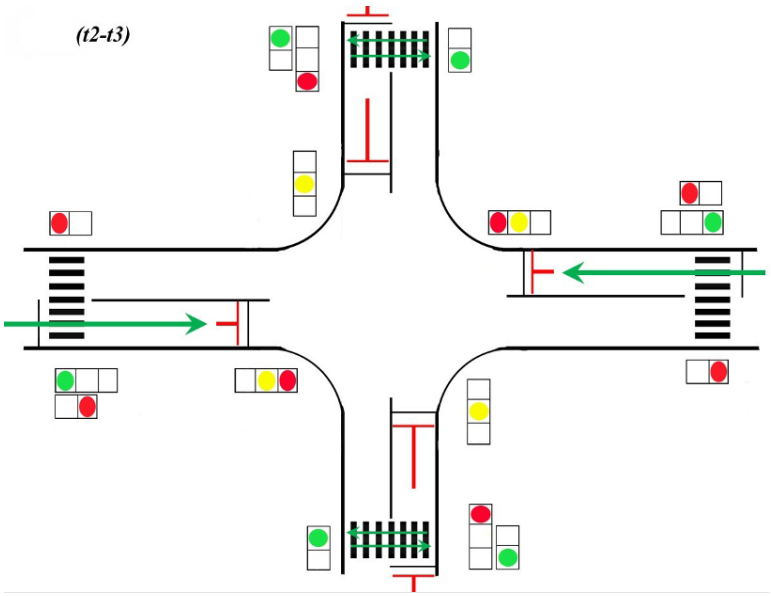


Рисунок 2.13 - В інтервалі часу t2–t3

В інтервалі часу t2-t3 транспортні засоби потоків А прискорюються.

На КС напрямів А червоний – жовтий. Пішоходи переходять проїжджу частину В.

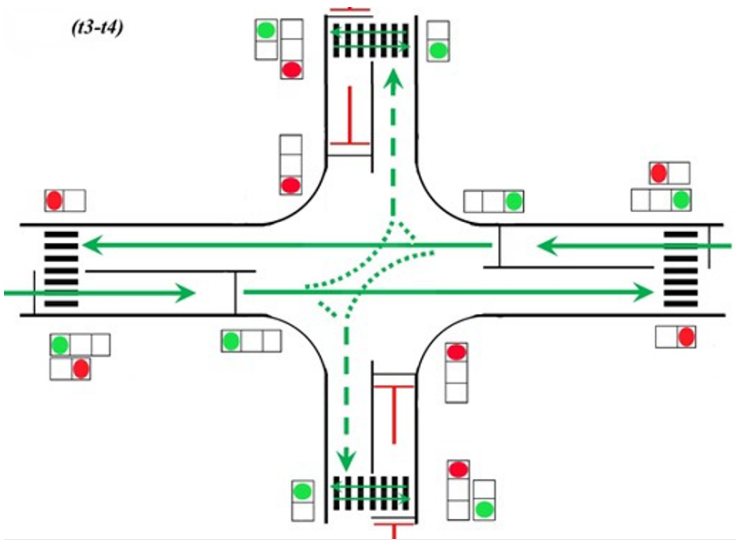


Рисунок 2.14 - В інтервалі часу t3–t4

В інтервалі часу t3– t4 транспортні засоби потоків А Перехрестяають стоп – лінію КС напрямків А та виїжджають на перехрестя, Перехрестяають його, здійснюють повороти ліворуч та праворуч. Повороти здійснювати легше, ніж звичайному перехресті оскільки є значні відстані від транспортних потоків А до

пішохідних переходів на потоках В. На цих ділянках перехрестя транспортні засоби можуть очікувати на можливість Перехрестяання пішохідних переходів. На КС напрямів А зелений.

Пішоходи переходять проїжджу частину В.

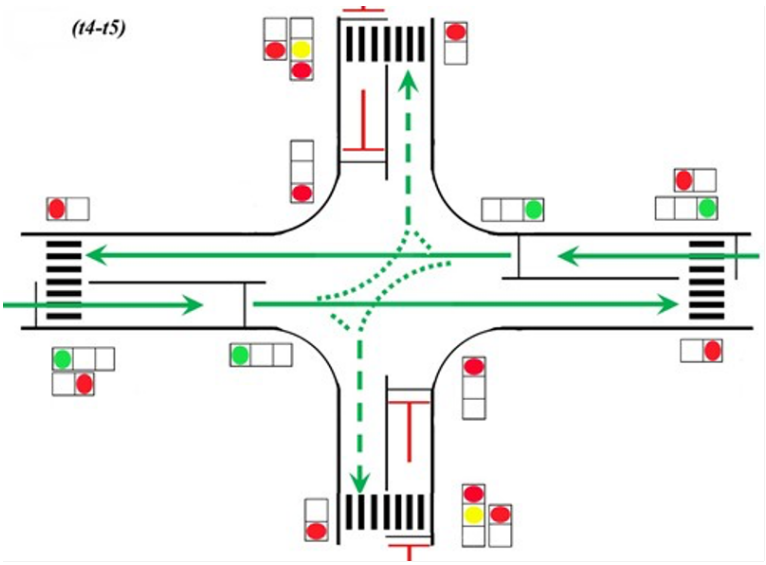


Рисунок 2.15 - В інтервалі часу t4–t5

В інтервалі часу t4–t5 транспортні засоби потоків – А виїжджають на перехрестя, Перехрестяають його, здійснюють повороти ліворуч та праворуч. Поворотам пішоходи не заважають. Пішохідний перехід скрізь заборонено.

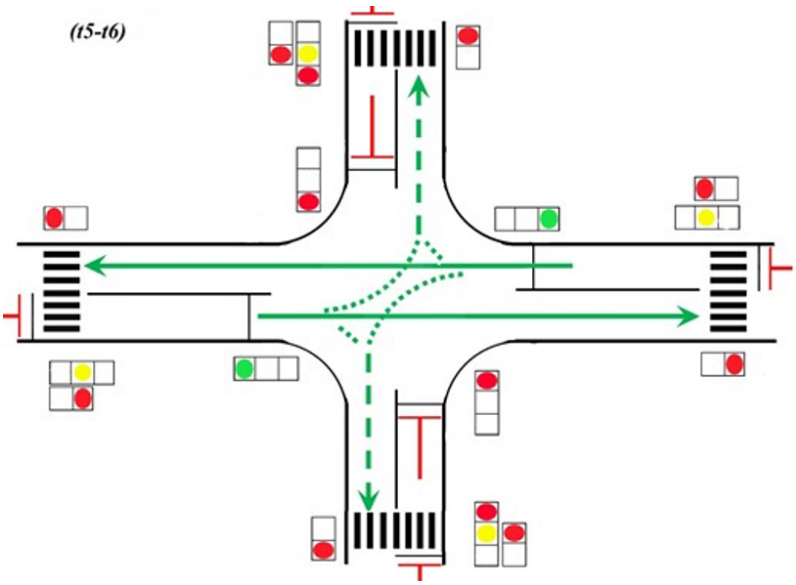


Рисунок 2.16 - В інтервалі часу t5–t6

В інтервалі часу t5-t6 транспортні засоби потоків А виїжджають на перехрестя, Перехрестяають його, здійснюють повороти ліворуч і праворуч. Поворотам пішоходи не заважають. На РС потоків А жовтий. Пішохідний перехід скрізь заборонено.

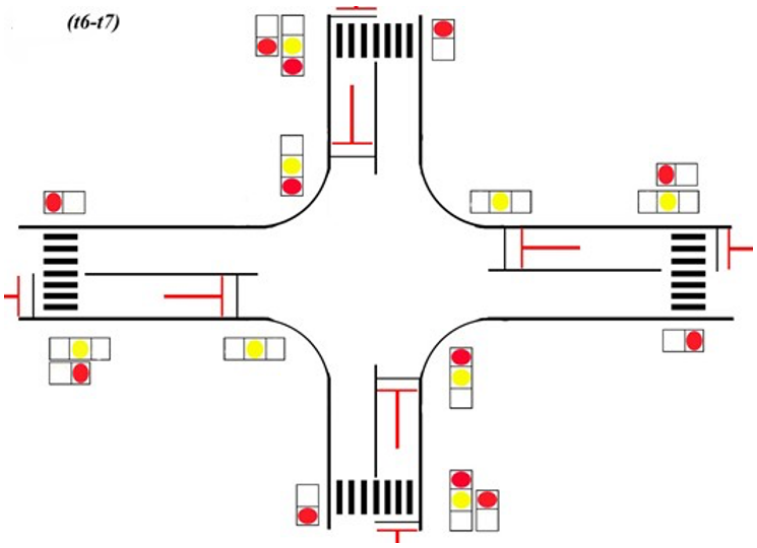


Рисунок 2.17 - В інтервалі часу t6–t7

В інтервалі часу t6–t7 на всіх РС та КС напрямів А жовтий. На всіх РС та КС напрямків В червоний – жовтий. Транспортні засоби звільняють перехрестя. Пішохідний перехід скрізь заборонено.

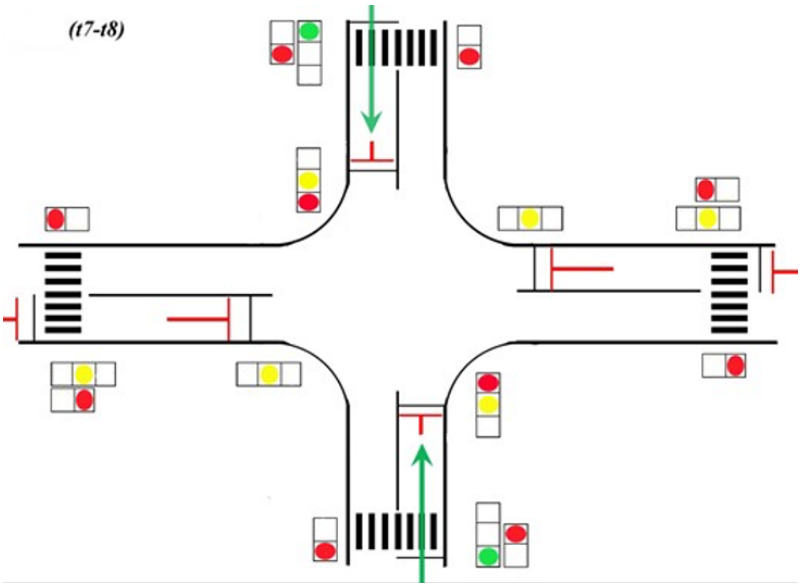


Рисунок 2.18 - В інтервалі часу t7–t8

В інтервалі часу t7–t8 транспортні засоби потоків В прискорюються. На РС напрямів зелений. Пішохідний перехід скрізь заборонено.

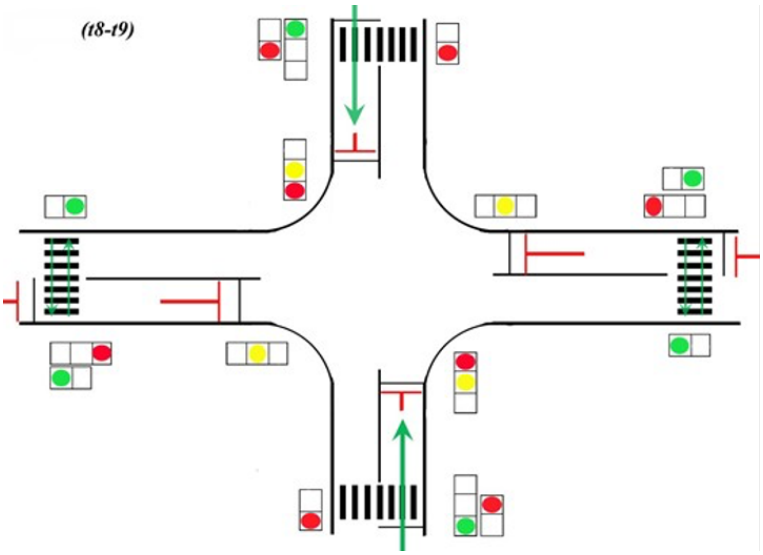


Рисунок 2.19 - В інтервалі часу t8–t9

В інтервалі часу t8–t9 транспортні засоби потоків прискорюються. На КС напрямів В червоний – жовтий. Пішоходи переходять проїжджу частину потоків А.

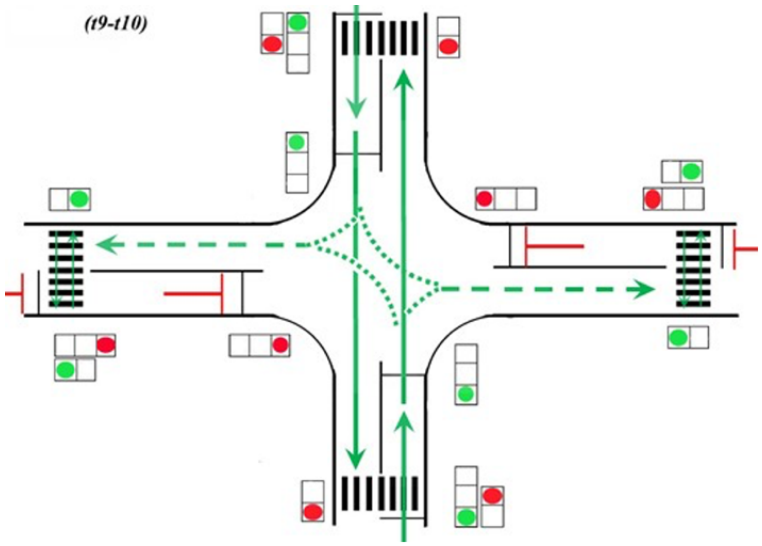


Рисунок 2.20 - В інтервалі часу t9–t10

В інтервалі часу t9–t10 транспортні засоби потоків В Перехрестяають стоп – лінію КС напрямків В і виїжджають на перехрестя, Перехрестяають його і

здійснюють повороти ліворуч і праворуч. Повороти здійснювати легше, ніж звичайному перехресті оскільки є значні відстані від транспортних потоків В до пішохідних переходів на потоках А. На цих ділянках перехрестя транспортні засоби можуть очікувати можливості перетнути пішохідні переходи. На КС напрямів В зелений.

Пішоходи переходять проїжджу частину потоків А.

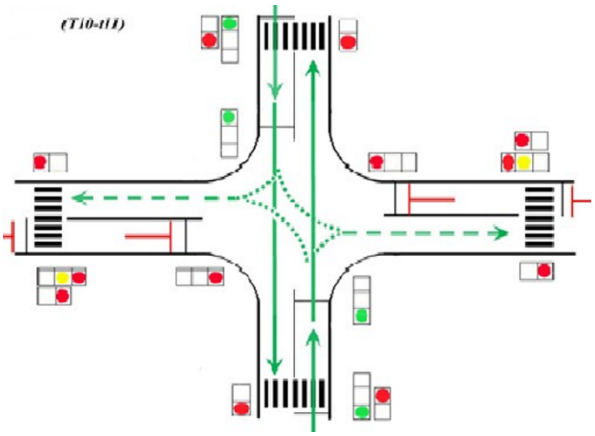


Рисунок 2.21 - В інтервалі часу t10–t11

В інтервалі часу t10–t11 транспортні засоби потоків В виїжджають на перехрестя, Перехрестяають його, здійснюють повороти ліворуч і праворуч. Поворотам пішоходи не заважають.

Пішохідний перехід скрізь заборонено.

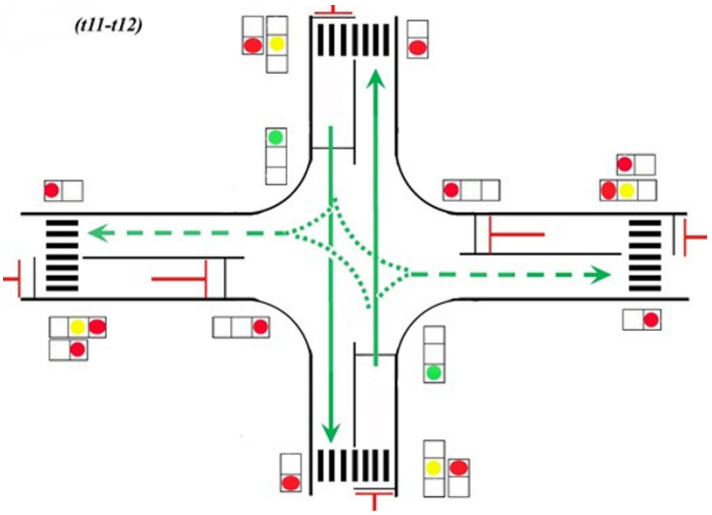


Рисунок 2.22 - В інтервалі часу t11–t12

В інтервалі часу t11–t12 транспортні засоби потоків В виїжджають на перехрестя, Перехрестяають його, здійснюють повороти ліворуч і праворуч. Поворотам пішоходи не заважають. На РС потоків В жовтий.

Пішохідний перехід скрізь заборонено

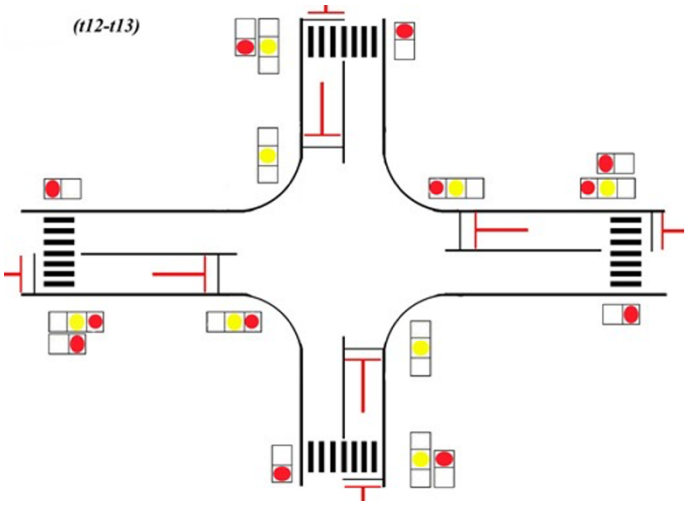


Рисунок 2.23 - В інтервалі часу t12–t13

В інтервалі часу t12 – t13 на всіх РС та КС напрямків В жовтий. На всіх РС та КС напрямків А червоно-жовтий сигнали.

Пішохідний перехід скрізь заборонено.

Варто зазначити, що:

Розгінні світлофори[88] можуть застосовуватися перед усіма керуючими світлофорами на світлофорному об'єкті, або їх частиною;

При застосуванні розгінних світлофорів транспортні засоби та пішоходи керуються існуючими правилами дорожнього руху.

Позитивні особливості застосування розгінних світлофорів:

а) Спосіб управління транспортними потоками на світлофорних об'єктах за допомогою розгінних світлофорів, що полягає в розгоні транспортного потоку перед проходженням його на дозвільний сигнал світлофора, що управляє, підвищує середню швидкість транспортного потоку, що проходить через керуючий світлофор

і як наслідок цього більш висока пропускна здатність світлофора, перед яким стоїть розгінний світлофор перед звичайним світлофором.

б) Пішохідні переходи на перехрестях, обладнаних розгінними світлофорами більш віддалені від місць Перехрестя транспортних потоків, що дозволяє акумулювати на цих ділянках перехрестя транспортні засоби, що повертають ліворуч і праворуч, і цим створювати менше перешкод іншим учасникам руху на перехресті, що збільшує пропускання.

в) Частина часу Перехрестя пішохідних переходів транспортними потоками, що повертають ліворуч і праворуч на перехрестях, обладнаних розгінними світлофорами, відбувається, коли рух пішоходів через пішохідні переходи заборонено, що дозволяє створювати менше перешкод іншим учасникам руху на перехресті і збільшує пропускну здатність. Це реалізується без збільшення світлофорного циклу.

г) Для збільшення пропускної спроможності існуючих перехресть не потрібно розширення проїжджої частини.

д) Пропонований спосіб керування транспортними потоками забезпечує підвищення безпеки дорожнього руху за рахунок високої ефективності.

2.6 Завдання аналізу та оптимізації транспортних розв'язок та розміщення міських зупинок

Київ – найбільший мегаполіс України. Природно, що він, як і інші великі міста розвинених країн, має необхідність вирішувати проблему дорожніх розв'язок. Сьогодні при проектуванні автомобільних доріг перевагу віддають сучасним технологіям та методам виробництва досліджень, заснованим насамперед на використанні високопродуктивних методів збору інформації про місцевість[98]:

– використання ГІС - технологій при пошуках автомобільних доріг та споруд на них;

– методам наземної та аерокосмічної цифрової фотограмметрії;

– системам супутникової навігації «GPS»;

– методам електронної тахеометрії,

– наземного лазерного сканування місцевості та геофізичних методів інженерно-геологічних вишукувань.

Транспортна розв'язка – комплекс дорожніх споруд (мостів, тунелів, доріг), призначеніх для мінімізації Перехрестяів транспортних потоків і, як наслідок, збільшення пропускної спроможності доріг. Переважно під транспортними розв'язками розуміються транспортні Перехрестя на різних рівнях, але термін використовується і для спеціальних випадків транспортних Перехрестяів в одному рівні. На сьогоднішній день при будівництві використовуються новітні сучасні технології під час будівництва автотранспортних розв'язок для покращення якості та безпеки розв'язок.

У нашому місті найчастіше використовують такі прилади, як тахеометри, а також електронні рулетки та системи GPS. Під час будівництва розв'язок використовуються новітні програми ГІС, такі як Credomix і AutoCAD. Ці програми призначені для вирішення завдань різних видів та складнощів при проектуванні будівництва ВДМ[28].

Спочатку визначимо типи розв'язок. Транспортні Перехрестя в різних рівнях за накресленням їх у плані поділяються на такі групи[84]:

а) конюшино подібні;

б) кільцеві;

в) петлеподібні;

г) складні перехрестя з відокремленими лівоповоротними з'їздами;

д) лінійні, ромбоподібні та комбіновані перехрестя в різних рівнях з поєднанням елементів різних видів перехресть, переважно таких, як конюшинні листи, лівоповоротні відокремлені з'їзди, петлі та ділянки перебудов.

Найбільшого поширення набули транспортні перехрестя у двох рівнях, які поділяються на такі типи:

а) повний конюшинний лист;

б) повний конюшинний лист з об'їздом навколо прилеглих кварталів;

в) неповний конюшинний лист;

г) сплющений конюшинний лист;

д) з п'ятьма шляхопроводами типу «хрест»;

е) з віднесеними лівими поворотами;

ж) з регульованим рухом по другорядному напрямку і т.д.

Кожен із наведених типів перехресть має свої недоліки та переваги. Визначення доцільності застосування того чи іншого перехрестя у різних рівнях у конкретних умовах потребує докладного вивчення всіх перерахованих типів.

Транспортне перехрестя у двох рівнях – “повний конюшинний лист”[32]:

Транспортне перехрестя типу “повний конюшинний лист” за своїм обрисом у плані нагадує контур листа конюшини. До переваг даної розв'язки відноситься те, що для її пристрою потрібно не так багато місця, порівняно з іншими видами багаторівневих розв'язок. Воно забезпечує безперервний рух транспорту по всіх напрямках при перехресті між собою двох магістралей (рисунок 2.24).

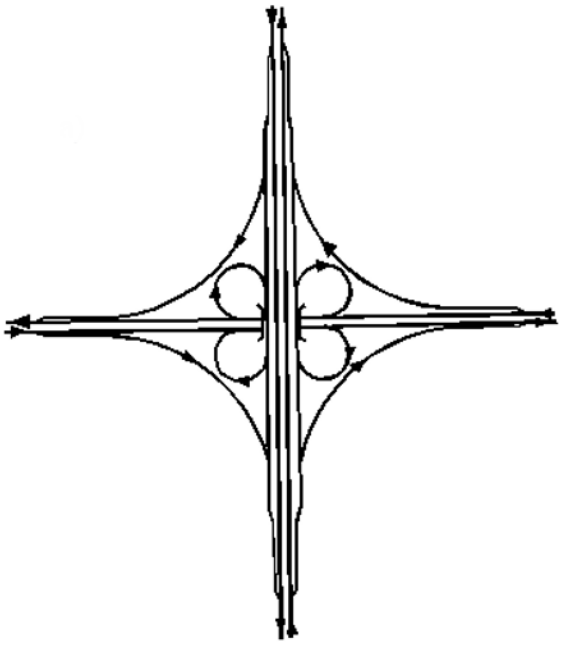


Рисунок 2.24 - Перехрестя типу "повний конюшинний лист"

Принцип його проектування полягає у забезпеченні такої організації руху, при якій прямі напрямки руху, що перехрестя перетинаються між собою, здійснюються у двох рівнях, а лівоповоротні напрямки замінюються поворотами вправо по спеціальних чотирьох з'їздах.

Перехрестя у двох рівнях – “неповний конюшинний лист”

При виборі схеми неповного конюшинного листа важливо враховувати пріоритетні напрямки з'їздів. Особливість такого перехрестя полягає в тому, що деякі потоки здійснюють виключно праві повороти, а деякі – ліві в районі примикання в одному рівні. Якщо пріоритетні напрямки збігаються з правоповоротними потоками, розв'язка добре працюватиме.

Перехрестя у двох рівнях типу “сплющений конюшинний лист”[35]:

У міських умовах для зменшення займаної території застосовується так званий сплющений конюшинний лист з лівоповоротними з'їздами, описаними мінімальними радіусами і розташованими паралельно рампам тунелю або естакади, рис 2.25. Спорудження такого перехрестя можливе при ширині основної магістралі безперервного руху не менше 80 – 100 м і пов'язане із труднощами у пропуску лівоповоротних потоків. Перевага полягає в тому, що при відносно малій площі, що займає перехрестя, забезпечується безперервність руху по всіх напрямках[30].

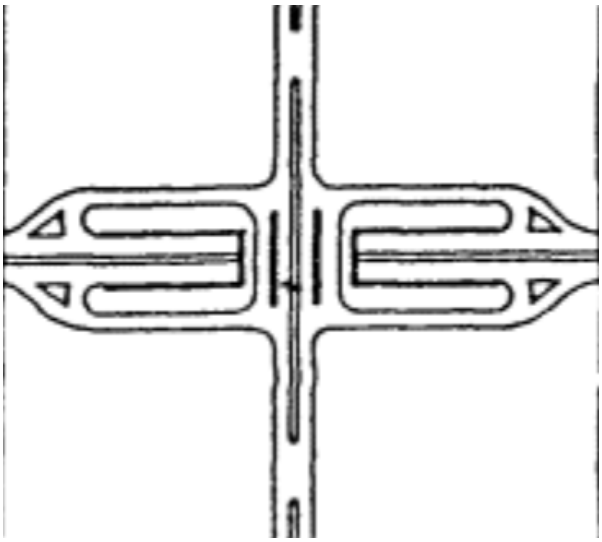


Рисунок 2.25 - Перехрестя за типом "сплющений конюшинний лист"

Покращене перетхрестя у двох рівнях на кшталт “розподільне кільце з п'ятьма шляхопроводами”:

Перехрестя "розподільне кільце з п'ятьма шляхопроводами" можна застосувати при схрещуванні двох рівнозначних магістралей з великими наскрізними потоками транспорту для повної розв'язки руху по всіх напрямках безпосередньо на площі перехрестя (рисунок 2.26)[43].

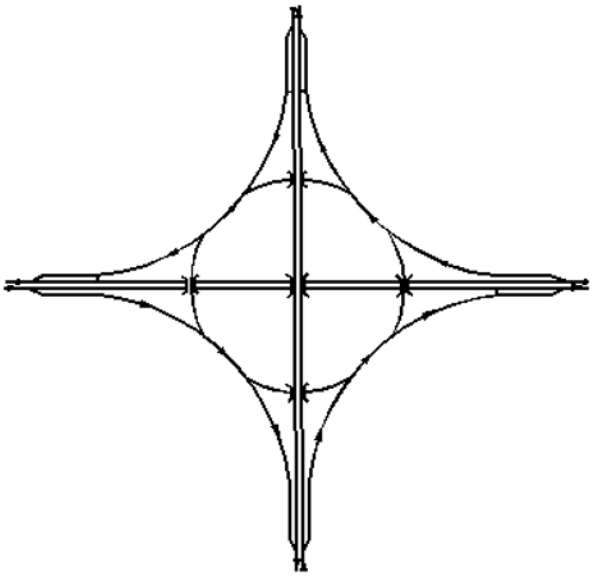


Рисунок 2.26 - Схема "розподільного кільця" з п'ятьма шляхопроводами

Перехрестя у двох рівнях на примиканні магістралей на кшталт “труба”[44]:

Перехрестя за типом "труба" застосовується на примиканні однієї магістралі до іншої при Т-подібних або У-подібних перехрестях. У міських умовах розв'язка «труба» так само, як «конюшинний лист», зберігає схему руху, але змінює планування: мала ширина вулиць притискає з'їзди до основної транспортної споруди. Радіуси лівоповоротних з'їздів зменшуються до 10-15м.

Перехрестя у двох рівнях на кшталт “подвійна петля”:

Цей тип перехрестя можна споруджувати при схрещенні магістральної вулиці, що вимагає великої пропускної спроможності з вулицею другорядного значення. Тут наскрізні потоки транспорту на магістральній вулиці прямують прямою, а на другорядному проїзді по розгалуженій кривій – через два шляхопроводи одностороннього руху (рисунок 2.27). Весь поворотний рух (у тому числі й праві повороти) пропускається через шляхопроводи[20].

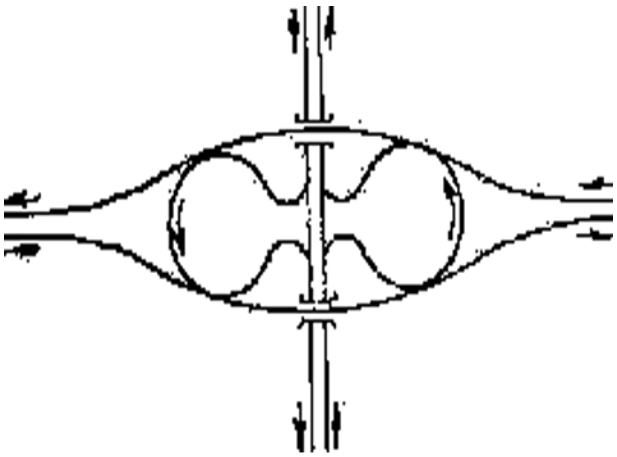


Рисунок 2.27 - Схема перехрестя типу "подвійна петля"

При будівництві мосту Кавацу-Нанадару[37] у Японії інженери використовували розв'язку типу подвійна петля. Він знаходиться в місті Кавацу. Ця подвійна спіраль дозволяє автомобілям підніматися і спускатися на 45 м, причому вони начебто підвішені між двома сторонами гори. У діаметрі петля дорівнює 80 м, а вся похила частина складає 1,1 км завдовжки. Міст на шосе 414 між Токіо та півостровом Іцу був завершений у 1982 році і став місцевою пам'яткою.

Перехрестя у двох рівнях із п'ятьма шляхопроводами типу «хрест»

Перевага даного перехрестя полягає у відсутності зайвих перехідних кривих, у тому, що всі повороти йдуть прямо, без відхилень, виключаючи перепробіги при лівоповоротному русі (характерні для перехрестя "конюшинного" типу). Довжина шляху кожного лівого повороту на 500-600 м коротше, ніж у "конюшинному листі". В даному випадку краще організується і рух, оскільки машини, що згортають ліворуч, дотримуються лівої сторони, праворуч - правої сторони, а прямий наскрізний потік йде посередині.

При виборі типу транспортної розв'язки на різних рівнях необхідно, передусім, враховувати інтенсивність і характер руху на вузлі. Інтенсивність руху на вузлі характеризується розмірами транспортних потоків, що проходять через вузол у певний час. У транспортних розрахунках приймаються максимальні розміри руху в усіх напрямках за годину[100].

При виборі типу перехрестя на різних рівнях поряд з переліченими вище факторами, насамперед, слід враховувати вартість його будівництва та подальшої

експлуатації. Проектування перехресть на різних рівнях має враховувати також зручність реконструкції та стадійності будівництва зі зростанням розмірів руху[10].

Потрібно запропонувати математичну модель транспортної розв'язки з використанням теорії S-гіпермереж та залежно від отриманого результату побудувати конструкцію розв'язки з наступними властивостями (рис. 2.29)[23]:

- Потік йшов безперервно. Під безперервністю мається на увазі таке, що потік автомобілів не зупиняється при виїзді чи в'їзді на іншу дорогу, тобто не повинні створюватися затримки під час руху автомобілів;

- Конструкція була оптимальною. Під оптимальністю розуміється таке: побудована споруда не містила б зайвих (не використовуваних) доріг.

Початкові дані (рис 2.28)[13]:

- місце, де необхідно побудувати транспортну розв'язку і дороги, які проходять це місце, тобто первинна мережа PS з її пропускною здатністю дороги Dk, де k-кількість доріг.

- матриця потоків даних доріг, тобто вторинна мережа WS (зважені орієнтовані ребра Pij).

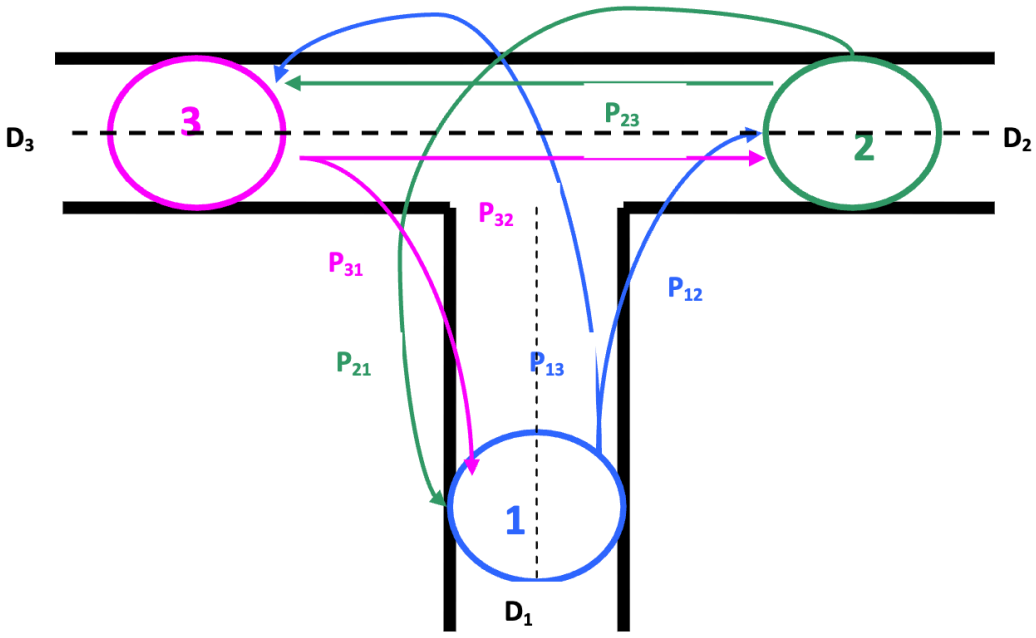


Рисунок 2.28 – Початкові дані

2.7 Алгоритм побудови транспортних розв'язок з використанням теорії S-гіпермереж

Алгоритм побудови транспортних розв'язок з використанням теорії S-гіпермереж виглядає наступним чином:

1) обирається "головна" дорога;
2) по матриці потоків обирається придатна розв'язка;
3) здійснюється пошук додаткових можливостей .

Крок 1. "Головна" дорога – це дорога, у якої потік максимальний; якщо потоки рівнозначні, можна взяти за головну одну з них і позначити її “1”. Перед під'їздами до перехрестя поставити смугу уповільнення, щоб автотранспорт зміг змінити напрям на необхідну смугу.

Крок 2. Є матриця потоків на перехресті Pij. За даною матрицею пропонується вибрати необхідну розв'язку наступним чином[91]:

а) для 4 доріг маємо:

Якщо потоки Pij задовольняють наступній системі (2.4.1), тоді вибираємо розв'язку типу "повний конюшинний лист" або "сплющений конюшинний лист"[12].

$$\begin{cases} P_{i,i+1} + P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+3} > \frac{1}{4} D_i \\ P_{i,i+1} > \frac{1}{4} D_i \end{cases} \qquad (2.4.1)$$

де $\forall i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо потоки Pij задовольняють наступну систему (2.4.2), тоді вибираємо розв'язку типу " неповний конюшинний лист " з чотирма односторонніми проїздами[42].

$$\begin{cases} P_{i,i+1} + P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+1} + P_{i,i+3} > \frac{1}{4} D_i \\ P_{i,i+3} < \frac{1}{4} D_i \end{cases} \qquad (2.4.2)$$

де $\forall i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо потоки Pij задовольняють наступній системі (2.4.3), тоді вибираємо розв'язку типу "неповний конюшинний лист" з двома з'їздами та проїздами двостороннього руху, розташованими в сусідніх чвертях перехрестя[52].

$$\begin{cases} P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+1} = 0 \\ P_{i+1,i} + P_{i+1,i+2} + P_{i+1,i+3} < D_i \\ P_{i,i+3} > \frac{1}{4} D_i \\ P_{i+2,i+3} + P_{i+2,i} < D_i \\ P_{i+2,i+3} = 0 \\ P_{i+3,i+2} = 0 \\ P_{i+3,i} = 0 \end{cases} \tag{2.4.3}$$

де $\exists i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо потоки Pij задовольняють наступній системі (2.4.4), тоді вибираємо розв'язку типу "неповний конюшинний лист" з двома з'їздами та проїздами двостороннього руху, розташованими у навхрест лежачих чвертях перехрестя[62].

$$\begin{cases} P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+1} = 0 \\ P_{i+1,i} + P_{i+1,i+2} + P_{i+1,i+3} < D_i \\ P_{i+1,i} < \frac{1}{4} D_i \\ P_{i+2,i} + P_{i+2,i+1} < D_i \\ P_{i+3,i} + P_{i+3,i+1} + P_{i+3,i+2} < D_i \\ P_{i+3,i+2} < \frac{1}{4} D_i \end{cases} \tag{2.4.4}$$

де $\exists\, i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо потоки Pij задовольняють наступній системі (2.4.5), тоді вибираємо розв'язку типу "неповний конюшинний лист" з трьома з'їздами та проїздами двостороннього руху, розташованими у трьох чвертях перехрестя[81].

$$\begin{cases} P_{i,i+1} + P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+3} > \frac{1}{4} D_i \\ P_{i+1,i} + P_{i+1,i+2} + P_{i+1,i+3} < D_i \\ P_{i+1,i} > \frac{1}{4} D_i \\ P_{i+2,i} + P_{i+2,i+1} + P_{i+2,i+3} < D_i \\ P_{i+2,i+1} < \frac{1}{4} D_i \\ P_{i+3,i+1} + P_{i+3,i+2} < D_i \\ P_{i+3,i} = 0 \end{cases} \quad (2.4.5)$$

де $\exists i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо потоки Pij задовольняють наступній системі (2.4.6), тоді вибираємо розв'язку типу "розподільне кільце з п'ятьма шляхопроводами"[72].

$$\begin{cases} P_{i,i+1} + P_{i,i+2} + P_{i,i+3} < D_i \\ P_{i,i+1} + P_{i,i+3} + P_{i,i} + P_{i+2,i+1} + P_{i+2,i+2} + P_{i+3,i+3} + P_{i+3,i+2} < max\{D_i\} \end{cases}$$

(2.4.6)

де $\forall i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

б) для 3 доріг маємо:

Якщо потоки Pij задовольняють наступній системі (2.4.7), тоді вибираємо розв'язку типу Т-подібного примикання[82].

$$\begin{cases} P_{i,i+1} + P_{i,i+2} < D_i \\ P_{i,i+2} > \frac{1}{4} D_i \end{cases} \quad (2.4.7)$$

де $\forall i = \overline{\mathbf{1,4}}$ та $i + \mathbf{4} = i$.

Якщо до (2.4.7) додати ще $\exists$ j $P_{j,j} \neq 0$ (2.4.8), тоді можна обрати розв'язку напівклеверного типу примикання або кільцевого типу примикання[92].

Крок 3. Як уже зазначалося, при розбудові розв'язки потрібно враховувати територію, прилеглу до перехрестя, наприклад, архітектурні споруди тощо. Багато розв'язків займають значно більшу територію, ніж територія перехрестя. Але ми показали, що для даного перехрестя необхідна така розв'язка, згідно з кроком 2.

Пропонуються наступні методи або модернізації для скорочення території займаної розв'язкою, при цьому потік автотранспорту йтиме безперервно:

а) для повороту праворуч створити смугу розгону.

Якщо прилегла територія до перехрестя не дозволяє побудувати поворот праворуч, можна здійснити поворот за допомогою смуги розгону, тобто. під'їжджаючи до перехрестя, повернути на смугу розгону (рисунок 2.29). Смуга розгону необхідна у тому, щоб автотранспорт, який здійснив поворот, зміг вписатися у транспортний потік[53].

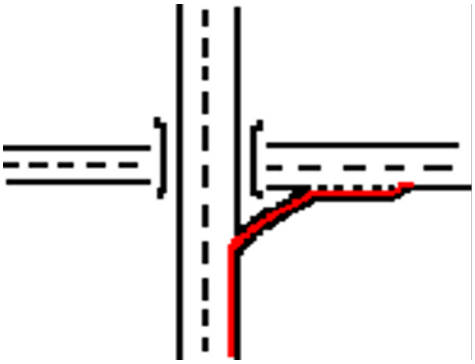


Рисунок 2.29 - Поворот праворуч за допомогою лінії розгону

б) поворот ліворуч здійснити на наступному перехресті.

Якщо прилегла територія до перехрестя не дозволяє побудувати поворот ліворуч, можна здійснити поворот за допомогою найближчого перехрестя праворуч або зверху, на якому можна буде розвернутися, рис 2.30

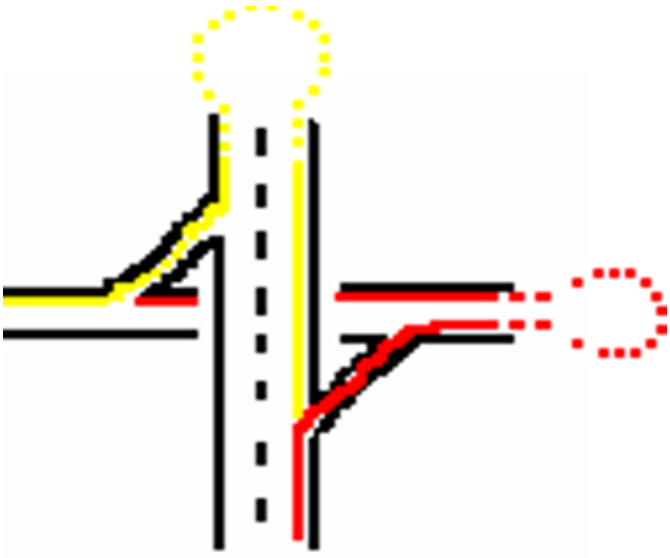


Рисунок 2.30 - Поворот ліворуч за допомогою іншого перехрестя

Можна вибрати це перехрестя, якщо виконується така нерівність:

$$t_{left} \geq \frac{2 \cdot \min\{L_1, L_2\}}{v}$$

де: tleft - середній час для здійснення повороту наліво;

L1 – відстань для ближнього правого перехрестя;

L2 – відстань для ближнього верхнього перехрестя;

v – середня швидкість руху.

Зауваження:

а) Використовувалося таке твердження:

Якщо число поворотного транспорту перевищує чверть від усього потоку, то для безперервного руху без світлофорного регулювання необхідно побудувати спеціальну поворотну дорогу.

б) Алгоритм працює для k = 3,4. При k>4 маємо[63]:

Якщо всі потоки будуть істотними, то буде розв'язка величезних розмірів і не придатна до міських умов. І для побудови розв'язки пропонується таке:

а) "розподільне кільце" з k вхідними дорогами, рис 2.31;

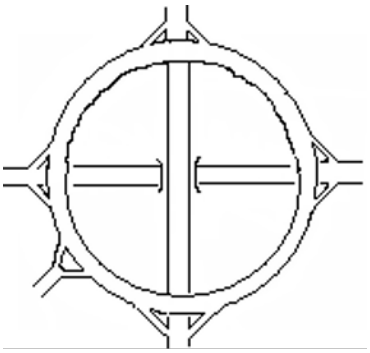


Рисунок 2.31 -" розподільне кільце" з 5 дорогами

б) розв'язка зі світлофорним регулюванням.

Розглянемо тепер переваги споруди "розподільне кільце":

а) Істотно зменшується час проходження даного вузла автотранспортом.

Час проходження вузла при розв'язці по всіх напрямках істотно швидше, ніж час проходження вузла при світлофорному регулюванні за рахунок того, що потік починає рух тільки зеленим сигналом.

б) Суттєво знижуються витрати.

При світлофорному регулюванні на перехресті при насиченому потоці неминуче виникають затори. За підрахунками експертів кожен автомобіль, що застряг у столичному заторі, коштує державі близько 150 гривні на годину. Стояння в заторі означає зростання витрат бензину, додаткову амортизацію транспорту, перешкоди у робочому графіку людей, які застрягли на дорозі. Моделювання з використанням з використанням теорії S-гіпермереж при побудові розв'язки показує, що на цьому перехресті ці показники суттєво знизяться.

2.7 Висновки до розділу 2

Розроблена модель інформаційної системи керування транспортними потоками для підвищення ефективності функціонування транспортної мережі міста.

За результатами порівняння автоматичних контролерів, що базуються на нечіткій логіці, показали виграш за часом проходження автомобілями перехрестя порівняно із звичайними методами, у яких час горіння сигналів світлофора фіксувався.

Розглянуто та доведено можливості визначення пропускної спроможності ВДМ за допомогою мереж Петрі.

Розглянуто моделювання перехрестя та блоку перехресть з використанням нечіткої логіки в керуванні світлофором. Результати порівняння автоматичних контролерів, що базуються на непарній логіці, показали виграш у часі проходження автомобілями перехрестя порівняно із звичайними методами.

Розглянуто способи керування транспортними потоками на світлофорних об'єктах за допомогою розгінних світлофорів. На основі проведеного аналізу виявлено позитивні особливості застосування розгінних світлофорів.

Проведено аналіз основних типів розв'язок. Побудовано математичну модель для кожного типу розв'язок з використанням з використанням теорії S-гіпермебреж.

Запропоновано алгоритм побудови оптимальної транспортної розв'язки, при якому суттєво зменшується час проходження даного вузла автотранспортом та знижуються витрати.

РОЗДІЛ 3 МЕТОДИ ТА РІШЕННЯ ЗАВДАНЬ УПРАВЛІННЯМ ПОТОКАМИ ТРАНСПОРТУ В МЕГАПОЛІСІ З ВИКОРИСТАННЯМ ТЕОРІЇ S-ГІПЕРМЕРЕЖІ

3.1 Завдання знаходження максимального потоку та методи вирішення

Для того, щоб раціонально організувати рух транспортних потоків необхідно: оцінити максимальний потік у мережі, знайти найбільш ефективний розподіл потоку, виявити вузькі місця та своєчасно їх ліквідувати. Поруч із цими завданнями необхідно оцінити сумарні витрати часу транспортних засобів за її русі з початкового пункту до кінцевий[19].

Проведено аналіз низки літературних джерел за знаходженням максимального потоку, розглянуто аналітичні моделі дослідження операцій та теорії автоматичного управління. Також проаналізовано застосування імітаційного моделювання на дослідження динаміки транспортних потоків регіону. На основі даного аналізу була виявлена актуальність розробки алгоритму суперпозиції потоків відповідно до теореми Форда-Фалкерсона, що дозволяє визначити інтегральний максимальний потік для ділянки регіональної мережі.

3.2 Розв'язання задачі знаходження максимального потоку у транспортній мережі з використанням алгоритму Форда-Фалкерсона

Нехай G = (N, A) - орієнтована мережа з одним джерелом $s \in N$ і одним стоком $t \in N$, і нехай дуги $(i, j) \in A$ мають обмежену пропускну здатність. Завдання про максимальний потік полягає в пошуку таких потоків по дугах, що належать множині A, що результуючий потік, який протікає з джерела s в стік t, є максимальним. Передбачається, що в джерело може надходити необмежений потік, для кожного проміжного вузла мережі виконується умова збереження потоку, а пропускна здатність U_{ij} кожної дуги являє собою кінцеву верхню межу потоку f_{ij} по

цій дузі.

Завдання про максимальний потік може бути сформульоване у вигляді наступної задачі[29]:

максимізувати $\sum_{i \neq n} f_{in}$ (3.1)

при умові, що $\sum_j f_{ij} - \sum_{ji} f_j = 0, i \neq 1, i \neq n$, (3.2)

$0 \leq f_{ij} \leq f_{ij}, (i,j) \in A$. (3.3)

Для його вирішення можна скористатися звичайним симплексним методом. Однак існує більш ефективна процедура пошуку розв'язання цієї задачі. Алгоритм починає роботу з деякого допустимого рішення. Потім виконується процедура розміщення позначок, розроблена Фордом і Фалкерсоном, за допомогою якої визначається інший допустимий потік більшої величини. У цьому алгоритмі вузли розглядаються як проміжні пункти передачі потоку, а дуги – як розподільні канали. Для формального опису алгоритму необхідно ввести два основні поняття – позначки та аугментальні шляхи потоку[74].

Позначка вузла використовується для вказівки як величини потоку, так і джерела потоку, що викликає зміну поточної величини потоку по дузі, що з'єднує джерело з вузлом. Якщо q_i одиниць потоку надсилається з вузла *i* у вузол j і викликає збільшення потоку цієї дуги, то вузол j позначається з вузла *i* символом + q_j.

У цьому випадку вузлу j приписується позначка [+ q_j, *i*]. Аналогічно якщо посилка q_j одиниць потоку викликає зменшення потоку по дузі, то вузол j позначається із вузла *i* символом - q_j. У цьому випадку вузлу j приписується позначка [- q_j, *i*].

Поточний потік із вузла *i* у вузол j збільшується, коли q_j одиниць додаткового потоку посилається у вузол j по орієнтованій дузі (i, j) у напрямку, що збігається з її орієнтацією. У цьому випадку дуга (i, j) називається *прямою.[39]*

Поточний потік з і до j зменшується, коли q_j одиниць потоку посилається у вузол j за орієнтованою дугою (i, j) у напрямку, протилежному її орієнтації. У цьому випадку дуга (i, j) називається *зворотньою.*

Якщо вузол j позначається з вузла і та дуга (i, j) пряма, то потік по цій дузі

збільшується і величина, відповідна невикористаної пропускної здатності дуги, що залишилася, повинна бути належним чином скоригована. Цю величину називають залишковою пропускною здатнісю дуги. Якщо деякому вузлу приписується позначка і навіть використовується пряма гілка, вона може мати лише позитивну «залишкову пропускну здатність». Крім того, вузол може бути помічений з вузла і тільки після того, як вузлу і приписана позначка.

Аугментальний шлях потоку з s в t визначається як зв'язкова послідовність прямих і зворотних дуг, по яких з s в t можна послати кілька одиниць потоку. Потік по кожній прямій дузі збільшується, не перевищуючи її пропускної здатності, а потік по кожній зворотній дузі зменшується, залишаючись при цьому позитивним.

Аугментальний шлях потоку використовується для вибору способу зміни потоку, при якому потік у вузлі t збільшується і при цьому для кожного внутрішнього вузла мережі не буде порушена умова збереження потоку. Алгоритм Форда-Фалкерсона має низку обмежень, які на практиці не виконуються[49]:

По-перше, пропускні здатності гілок мережі мають бути цілими невід'ємними детермінованими числами. У реальній мережі пропускна здатність не є постійною і залежить від таких імовірнісних факторів, як завантаженість, стан дороги, параметри зовнішнього середовища. Завантаженість на різних ділянках дороги буває різною і залежить від наявності внутрішніх транспортних потоків на цій ділянці, які можуть розглядатися як перешкоди при пересуванні транспортної одиниці з початкового пункту мережі до кінцевого. Стан дороги визначається її зношеністю, умовами експлуатації, впливом погодних умов. Параметри зовнішнього середовища змінюються залежно від пори року та доби. Як правило, у транспортній мережі ці фактори є взаємопов'язаними.

По-друге, виконання алгоритму Форда-Фалкерсона пропонується починати з деякого початкового потоку, що вже існує в мережі, в той час як вибір початкового потоку не змінює величину максимального потоку, а змінює розподіл максимального потоку в мережі.

По-третє, завдання пошуку максимального потоку вирішується для єдиного потоку через мережу, тобто, у самій мережі має бути лише один вхід – вузол

мережі, через який транспортні засоби потрапляють у мережу, і лише один вихід – вузол мережі, через який вони виходять із мережі.

Виходячи з вищевикладеного, як вихід із становища можна вдатися до імітаційного моделювання транспортних потоків у мережі, що дозволяє врахувати велику кількість зовнішніх факторів і якомога ближче прив'язати програмну модель до її реального уявлення. Метод дозволяє визначити максимальний потік та його розподіл по гілках мережі, що має найкращу ефективність серед інших варіантів розподілу потоків у мережі, пропускні здатності гілок якої змінюються у часі випадковим чином.

Далі розглядається завдання знаходження інтегрального максимального потоку в мережі, що має задану кількість входів і виходів, що визначають деякий напрямок руху транспорту в мережі. Пропонується метод визначення максимального потоку регіональної мережі для заданого напрямку, що знімає обмеження алгоритму Форда-Фалкерсона.

Для знаходження інтегрального максимального потоку транспортної мережі в заданому напрямку (ZV) для кожного часового інтервалу складаються матриці величин максимальних потоків та ефективності цих потоків, елементами яких є значення максимальних потоків та ефективностей потоків по кожному поєднанню входу та виходу. Матрицю величин потоків позначимо $\varphi = \left\|\varphi_{ij}^{\max}\right\|, i = 1, m,\ j = 1, n.$ Матрицю ефективностей потоків позначимо $\Phi = \left\|\varphi_{ij}\right\|, i = 1, m,\ j = 1, n.$ Отримані матриці нормуються по максимальному елементу[73]:

$$\varphi^* = \left\|\varphi_{ij}^*\right\| = \left\|\frac{\varphi_{ij}}{\max\limits_{ij}\varphi_{ij}}\right\|, \Phi^* = \left\|\Phi_{ij}^*\right\| = \left\|\frac{\Phi_{ij}}{\max\limits_{ij}\Phi_{ij}}\right\|, i = \overline{1, m}, j = \overline{1, n}\ .(3.4)$$

В результаті всі елементи матриць задовольняють нерівності $0 < \varphi_{ij}^* < 1, 0 < \Phi_{ij}^* < 1$. В прямокутній системі координат відмічаємо точки. Причому через те, що елементи матриць номіновані за максимальним елементом, всі точки знаходитимуться в межах одиничного квадрата, лівий нижній кут якого поєднаний із початком координат.[59]

Усі точки одиничного квадрата порівнюються за допомогою деякої метрики. Складається список потоків, в якому всі елементи ранжуються від "найгіршого" за ефективністю до "найкращого" відповідно до даних одиничного квадрата. Одночасно складається матриця досяжності $D = \|d_{ij}\|, i = \overline{1,m}, j = \overline{1,n}$, де $d_{ij} = 1$, якщо є потік з і-того входу в j-й вихід, та d_{ij}= 0 якщо немає потоку з і-того входу в j-й вихід.

Список проглядається, починаючи з потоку, найгіршого за ефективністю. Поточний потік виключається зі списку, якщо його виняток не залишає жодної початкової вершини без вихідного потоку і не залишає жодної кінцевої вершини без вхідного потоку. При виключенні зі списку потоку, що йде з і-того входу в і-тий вихід, модифікується матриця досяжності D, в якій на перехресті і-того рядка та і-того стовпця одиниця замінюється нулем.

Таким чином, поточний потік з і-того входу в j-тий вихід виключається зі списку в тому випадку, якщо після його виключення та модифікації матриці досяжності D в її і-тому рядку буде хоча б одна одиниця та в j-тому стовпці також буде хоча б одна одиниця. Якщо ж виняток потоку веде до того що, у матриці досяжності і-тий стовпець чи j-та рядок складатиметься з нулів, то потік зі списку потоків не виключається, й у списку переходимо до наступного потоку.

В результаті відбраковування потоків отримуємо безліч найбільш ефективних потоків $ZV^{\epsilon} = \{Z_i, V_j\}$, таких, що кожен із входів пов'язаний транзитним потоком принаймні з одним виходом. Тобто кожен із входів транспортної мережі має, принаймні, один вихідний з нього потік, а кожен з виходів мережі має хоча б один потік, що входить до нього. Для множини таких потоків ZV^{ϵ} застосовується принцип суперпозиції, коли відповідні їм матриці розподілу потоків $X^{ij} = \|x_{kl}^{ij}\|$ сумуються, утворюючи матрицю інтегрального транзитного потоку за вибраним напрямом. Причому, завдання суперпозиції потоків вирішується таким чином, щоб в мережі могли одночасно існувати всі потоки, що залишилися з множини ZV^{ϵ}.

З цією метою складається загальна кількість дуг мережі DN={<d_{ij},n>}. Елементами цієї множини є пари <d_{ij},n, що складаються з покажчика насиченої

дуги d_{ij} з і-того вузла в j-тий, і числа *n*, що показує, наскільки інтегральний потік цієї дуги перевищує її пропускну здатність. З множини насичених дуг *DN* вибирається елемент, у якого величина *n1* максимальна. Цей елемент списку $<d_{ij},n$ описує дугу, на яку величина *n1* сумарного потоку, побудованого з транзитних потоків, що залишилися, найбільше перевищує пропускну здатність дуги. Тому в кожному з транзитних потоків множини DN, де зустрічається ця обрана дуга, необхідно зменшити потік в n1 раз для того, щоб після виконання суперпозиції потоків, що залишилися, дуга d_{ij} виявилася насиченою.[69]

Зменшення проводиться для кожного потоку з множини $\{Z_iV_j\}$ по всіх шляхах, які насичували дугу в ході виконання алгоритму Форда-Фалкерсона, пропорційно їх вкладу в насичення дуги. Після цього безліч *DN* перебудовується через те, що було зроблено зменшення кожного з транзитних потоків, що спричинило зміну величин n для гілок мережі, які були задіяні при зменшенні потоку.

Описані дії виконуються до тих пір, поки в множині DN є хоча б один елемент, у якого n більше нуля. Як тільки у всіх елементів множини DN величини n стануть негативними або рівними нулю, процес зменшення транзитних потоків закінчується і як рішення задачі знаходиться результуючий інтегральний потік. При цьому потоки розподілені таким чином, що після застосування до них принципу суперпозиції величини сумарного результуючого потоку на дугах не перевищують їх пропускних здібностей.

В результаті виконання алгоритму суперпозиції потоків відповідно до теореми Форда-Фалкерсона величина потоку на будь-якому розрізі мережі буде максимальною, а сумарний потік буде складатися зі зменшених транзитних потоків.

3.3 Визначення інтегрального максимального потоку для ділянки регіональної мережі

Для ілюстрації роботи алгоритму знаходження інтегрального потоку

розглянемо ділянку регіональної транспортної мережі, подану у вигляді графа на рис. 3.1.

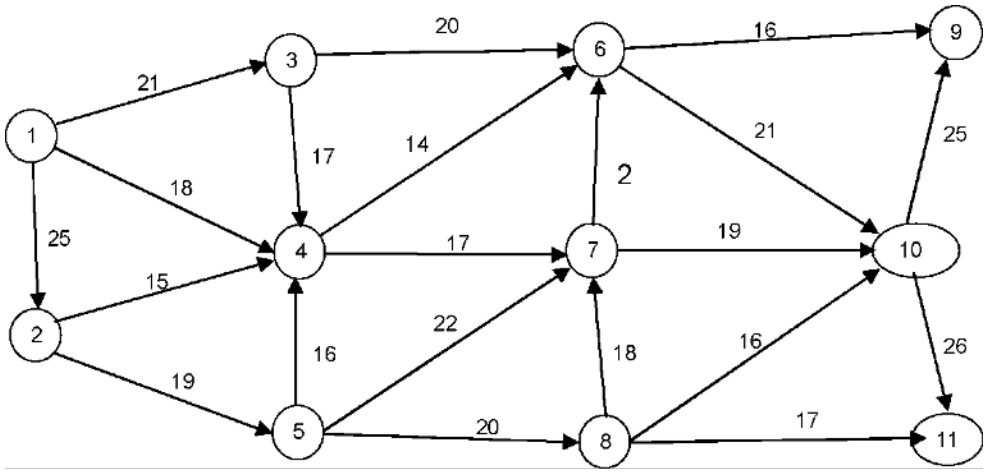


Рисунок 3.1 - Графічне подання ділянки регіональної транспортної мережі

Безліч входів у мережі представлено вершинами 1 і 2. Безліч виходів задається вершинами 9 і 11. У мережі розглядаються такі транзитні потоки: (1,…,9), (1,…,11), (2,…,9) и (2,…,11)[40].

Для кожного з вказаних транзитів вирішується задача про максимальний потік окремо, в результаті отримаємо чотири матриці ефективностей ($\varphi_{1\_9}^{max}$, $\varphi_{1\_11}^{max}, \varphi_{2\_9}^{max}, \varphi_{2\_11}^{max}$) , а також відповідні розподіли величин потоків по гілках мережі ($X^{1\_9}, X^{1\_11}, X^{2\_9}, X^{2\_11}$).

Для формування списку найбільш ефективних потоків відкидаємо найменш ефективні потоки таким чином, щоб не залишити жоден вхід без хоча б одного потоку, що виходить, і жоден вихід хоча б без одного вхідного потоку. Для прикладу найбільш ефективними потоками виявилися потоки 1→9 з величиною потоку 41 одиниця, 1→11 з величиною потоку 43 одиниці і 2→9 з величиною потоку 34 одиниці.

Підсумовуючи матриці максимальних потоків $\|X^{1\_9}\|, \|X^{1\_11}\|$, і $\|X^{2\_9}\|$ отримуємо матрицю інтегрального транзитного потоку$\|\Sigma X\|$. Шляхом

поелементного віднімання матриці $\|\Sigma X\|$ з матриці пропускних здатностей мережі$\|C\|$ отримуємо матрицю$\|C - \Sigma X\|$, яка має такий вигляд[50]:

$$\left\|C - \sum X\right\| = \begin{pmatrix} 0 & 5 & -19 & -6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -3 & -17 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 17 & 0 & -20 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -15 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16 & 0 & 0 & 3 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -16 & -18 & 0 \\ 0 & 0 & 0 & 0 & 0 & 19 & 0 & 0 & 0 & -11 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 18 & 0 & 0 & 16 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -18 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Від'ємні значення елементів матриці свідчать про недостатню пропускну здатність відповідних гілок мережі у разі руху через мережу одночасно всіх залишених потоків. Вибираємо гілку, сумарний потік де найбільш перевищує її пропускну здатність. Для прикладу ця гілка (3,6) – найменший елемент матриці $\|C - \Sigma X\|$[83].

Вибираємо всі шляхи, які насичували гілку (3,6) у ході розв'язання задачі про максимальний потік для кожного з транзитних напрямків та величини Δ, на які збільшувався потік цими шляхами.

Для транзитного напрямку 1→9 це будуть шляхи:

(1,3) → (3,6) → (6,9), Δ =16;

- (1,3) → (3,6) → (6,10) → (10,9), Δ=4.

Для транзитного напрямку 1 →11:

- (1,3) → (3,6) → (6,10) → (10,11), Δ =20.

В транзитному напрямку 2→9 жоден із шляхів у ході вирішення завдання про максимальний потік не проходив через вершину (3,6). Для того, щоб сумарний

потік зміг пройти через гілку (3,6), зменшуємо потоки транзитних напрямів обраними шляхами так, щоб весь сумарний потік зменшився на 20 одиниць, причому по кожному шляху потік зменшується пропорційно величині Δ. Для транзитного напрямку 1→9 потік шляхом 1 зменшуємо на 8 одиниць, а потік шляхом 2 зменшуємо на 2 одиниці. Для транзитного напрямку 1→11 потік шляхом 1 зменшуємо на 10 одиниць. З цією метою віднімаємо від елементів матриць $\|X^{1\_9}\|, \|X^{1\_11}\|$ відповідних вузлам шляхів величини, на які зменшується потік шляхом.

Далі перераховуються значення елементів матриці $\|C - \Sigma X\|$. В результаті одержуємо нову матрицю, у якої елемент (3,6) дорівнює 0, так як потік через гілку мережі (3,6) було зменшено.

Процес зменшення аналізованих потоків повторюється до того часу, $\|C - \Sigma X\|$ поки в матриці залишатимуться негативні елементи. Коли всі елементи матриці $\|C - \Sigma X\|$ виявляться невід'ємними, це означатиме, що пропускних здібностей гілок мережі достатньо для того, щоб усі зменшені транзитні потоки змогли існувати в мережі одночасно. У цьому випадку алгоритм зменшення транзитних потоків закінчується і розв'язанням задачі про максимальний потік буде сумарна матриця $\|\Sigma X\|$.

Для графа, що розглядається, шляхом послідовних зменшень була отримана наступна матриця $\|\Sigma X\|$79:

$$\left\|\sum X\right\| = \begin{pmatrix} 0 & 12 & 20 & 17 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 17 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 14 & 13 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 8 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 19 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 23 & 0 & 16 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Величина потоку після зменшення по транзиту 1→9 склала 24 одиниці, по транзиту 1→11 – 25 одиниць, по транзиту 2→9 – 15 одиниць. Сумарна величина трьох потоків склала 64 одиниці[60].

Як видно з прикладу, сумарна величина інтегрального максимального потоку більша, ніж будь-яка величина максимального потоку для потоків які розглядаються, знайдена для випадку одного потоку в мережі, що свідчить про найбільш повне використання ресурсів мережі.

3.4 Дослідження впливу одностороннього руху на величину транспортного потоку

Основний показник завантаженості вулиць – середня швидкість руху транспортних засобів, яка з роками різко знижується. Причому ця проблема стала торкатися не лише великих міст, а й дрібніших. Зростання скупчення транспортних засобів на дорожньо-транспортній мережі міста не тільки підвищує витрати за рахунок втрати часу, але також збільшує ймовірність пригод і негативно впливає на навколишнє середовище та якість життя людей.

Керування транспортними потоками дорожньо-транспортних мереж тісно пов'язане з їхньою пропускною спроможністю. Для найефективнішої роботи

мережі необхідно забезпечити можливість проходження крізь неї максимального потоку.

Аналіз літератури показав, що з побудови моделей дорожньо-транспортної мережі застосовують теорію графів чи гіперграфів. Однак у побудованих таким чином моделей є недоліки, наприклад, сильна перевантаженість у вузлах (перехрестя, розв'язки тощо). При додаванні в модель потоків різного сорту граф перетворюється на абсолютно інший об'єкт, з яким часом незрозуміло як працювати[21].

Гіпермережі, на відміну від графів, дозволяють адекватно описувати системи мережевої структури пошарової ієрархії.

Так як транспортна інфраструктура мегаполісу є складною ієрархічною нестаціонарною системою мережевої структури, обрана теорія дозволяє близько до реальності моделювати транспортні мережі та вирішувати ряд завдань, пов'язаних з ними, таких як: моделювання транспортних потоків дорожньо-транспортної мережі міста; керування транспортними системами; оптимальне розміщення пунктів обслуговування та ін.

Розглянемо ділянку дорожньо-транспортної мережі з двостороннім рухом (одна смуга руху на один бік, інша — на протилежний). В цьому разі на перехресті виникає так звана проблема повороту ліворуч[70].

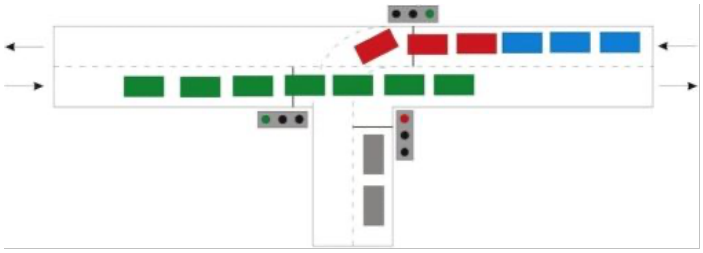


Рисунок 3.2. Проблема лівого повороту

На, рис. 3.2, червоні машини за правилами дорожнього руху змушені чекати, доки проїдуть зелені, і тільки після цього вони можуть повернути ліворуч. Однак блакитним машинам потрібно рухатися прямо, але вони очікують, доки проїдуть

червоні. Внаслідок цього виникає затор та знижується пропускна здатність даної ділянки.

Введення одностороннього руху (обидві смуги руху в один бік) може повністю або частково вирішити цю проблему, розвантажити ділянку дороги та збільшити пропускну спроможність, а як наслідок, і величину потоку[80].

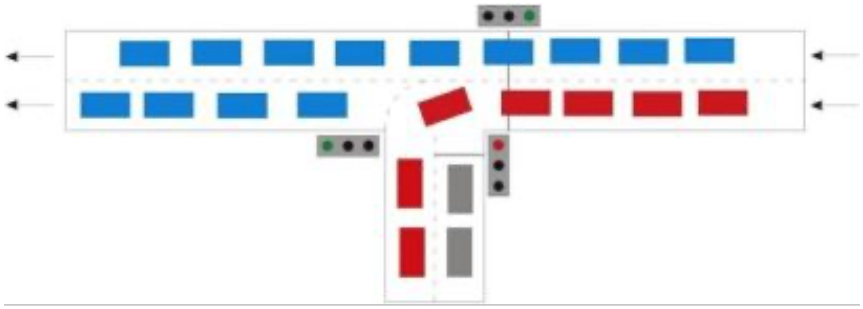


Рисунок 3.3. Односторонній рух по головній дорозі

На, рис. 3.3, видно, що введення одностороннього руху по всій горизонтальній проїжджій частині повністю вирішує проблему лівого повороту[93].

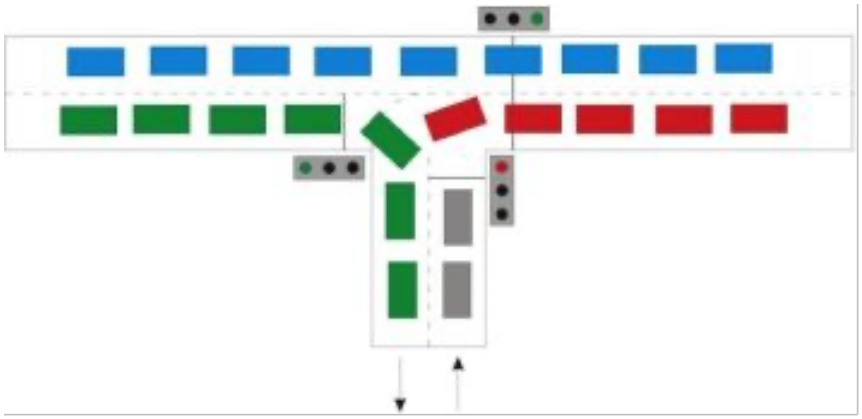


Рисунок 3.4. Односторонній рух праворуч до перехрестя

Введення одностороннього руху праворуч від перехрестя, рис. 3.4, вирішує проблему частково: дозволяє блакитним машинам безперешкодно проїжджати перехрестя, але червоним машинам, як і раніше, необхідно чекати, поки проїдуть зелені.

Тут слід зауважити, що теорія нестаціонарних s-гіпермереж дозволяє врахувати практично всі дорожні колізії, пов'язані зі структурою мережі; потоками машин, залежними від часу доби та/або днів тижня; системою керування транспортними потоками (світлофори, інверсні смуги, дорожні знаки як постійні, так і змінювані залежно від ситуації на дорогах); розміткою доріг тощо.

Локуси – це ділянки мережі, які займають транспортні одиниці. Кількість локусів, що містяться на один аллель, обчислюється за формулою[90]:

$$S = \frac{L_a}{L_l}$$

де S – кількість локусів, L_a – довжина алелю, L_l – довжина локусу. Однак довжина локусу не може бути занадто маленькою, наприклад, якщо L_l дорівнює довжині автомобіля. За такого розкладу забезпечити безпечну швидкість руху (відмінну від нуля) неможливо.

Передбачається, що для безпечного руху ділянкою дороги необхідно дотримуватися дистанції між транспортними засобами. Причому дистанція залежить від швидкості автомобіля, крім місць, позначених знаком, її визначальним.

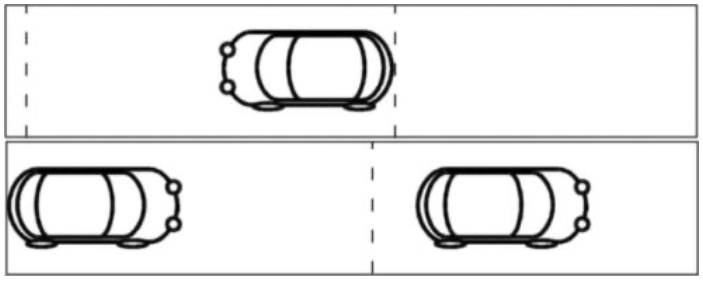


Рисунок 3.5. Безпечна дистанція в залежності від швидкості руху

Орієнтовно безпечною дистанцією між транспортними засобами, що рухаються в транспортному потоці з однаковою швидкістю, прийнято вважати відстань у метрах, що дорівнює третині величини швидкості (при швидкості 60 км/год безпечна дистанція – 20 метрів). Отже, у формулі довжина локусу L_l

дорівнює - $\frac{v}{3}$, де v – швидкість транспортного засобу. У цьому випадку отримаємо ситуацію, подану на, рис. 3.5.

Локус активізується і починає працювати тоді, як у ньому з'являється транспортна одиниця (автомобіль). Варіант його роботи залежить від того, якого типу автомобіль у нього потрапив, залежно від цього обирається „програма" роботи. Наприклад, локуси можуть працювати за розкладом або випадково: з ймовірністю 0,01 автомобіль поїде назад, 0,02 — зупиниться, 0 97 поїде вперед, що дозволяє розглядати різнотипні потоки та їх поведінку[21].

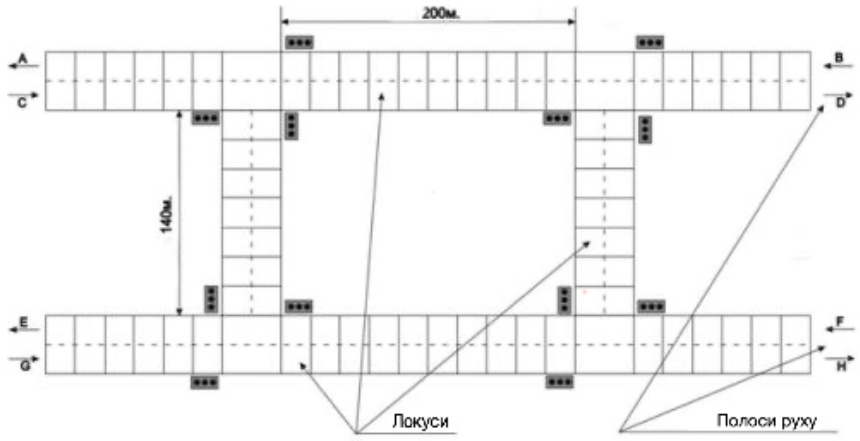


Рисунок 3.6. Приклад ділянки дорожньо-транспортної мережі

Розглянемо ділянку дорожньо-транспортної мережі, зображену на рис. 3.6.

Нехай із C їдуть машини, чергуючись: 5 - в D, 5 - в E; з F їдуть машини, чергуючись: 5 - в E, 5 - в D; з B їдуть в A, але на кожні 7 машин припадають 3, які їдуть у H; із G їдуть в H, але на кожні 7 машин припадають 3, які їдуть до A. Швидкість машин становить 60 км/год. Світлофори працюють із періодичністю 20 секунд (20 секунд горить зелений, 20 — червоний), за цей час по одній смузі перехрестя встигають подолати 15 машин[31].

Завдання: порахувати пропускну спроможність даної ділянки.

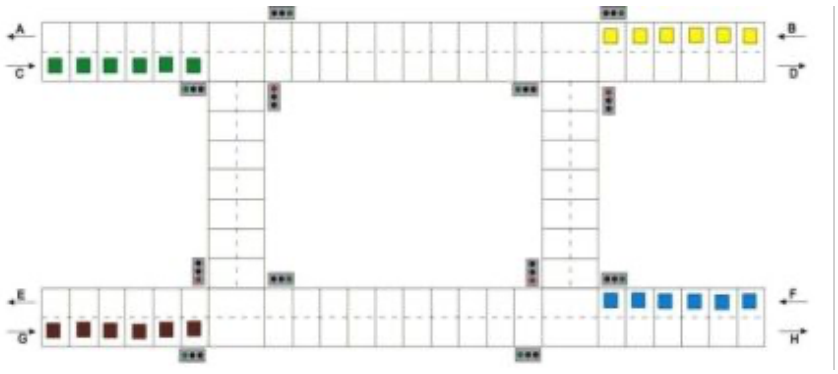


Рисунок 3.7. Початок руху на цій ділянці

Розглянемо потактово, що відбувається з потоками на цій ділянці. На, рис. 3.7, показано стан у момент часу t = 0с.

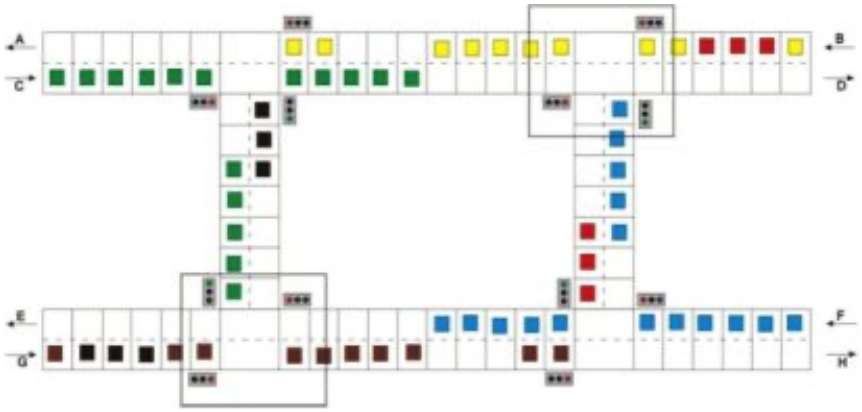


Рисунок 3.8. Проблема повороту ліворуч

На, рис. 3.8, показано стан у момент часу t = 20с.

Також на, рис. 3.8, виділено дві ділянки, на яких при наступному такті світлофора виникає проблема лівого повороту, коли червоні та чорні машини, чекаючи на поворот, будуть заважати проїхати жовтим і коричневим відповідно.

Моделюючи кілька тактів світлофора, отримуємо, що пропускна здатність даної ділянки дорожньо-транспортної мережі дорівнює ≈ 2600 машин на годину в один бік.

Розглянемо аналогічне завдання, тепер не буде розкладу, скільки машин куди їде. З E машини їдуть з ймовірністю 0,33, інші їдуть в D; з F в D машини їдуть із

ймовірністю 0,33, інші їдуть у E; із B в A машини їдуть з ймовірністю 0,8, інші їдуть в H; з G в H машини їдуть із ймовірністю 0,8, інші їдуть в A. Швидкість машин становить 60 км/год. Світлофори працюють із періодичністю 20 секунд (20 секунд горить зелений, 20 – червоний), за цей час по одній смузі перехрестя встигають подолати 15 машин.

Завдання: порахувати пропускну здатність даної ділянки.

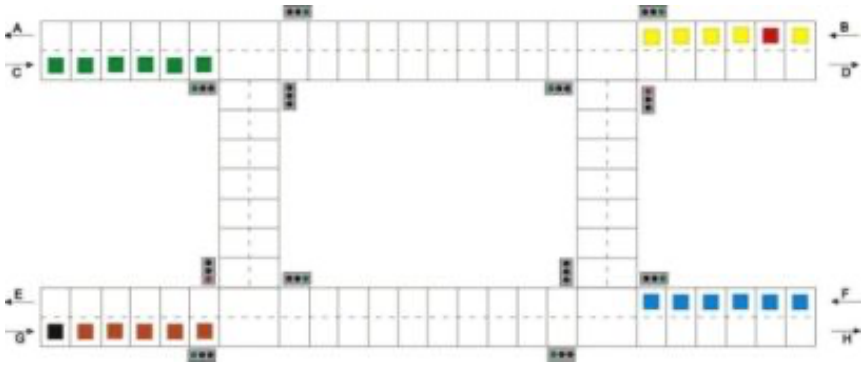


Рисунок 3.9. Початок руху транспортних одиниць ділянкою

Аналогічно минулому випадку, розглянемо потактово, що відбувається на ділянці. На, рис.3.9, показано стан у момент часу t=0c.

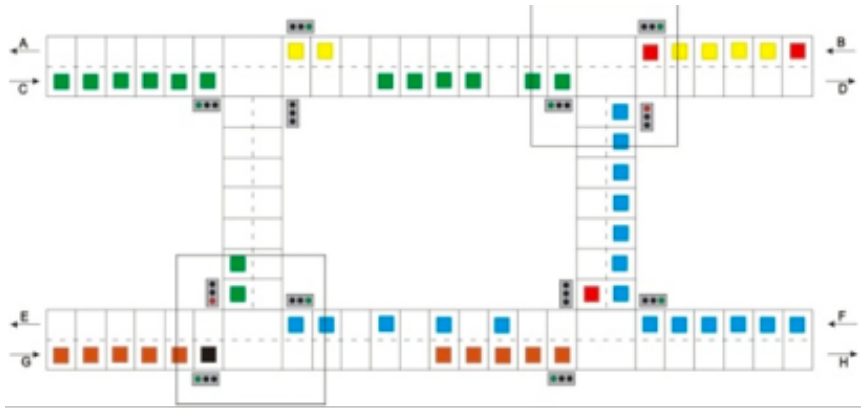


Рисунок 3.10. Колізії, що виникають при повороті ліворуч

На, рис. 3.10, показано стан у момент часу t=60c. Виділені ділянки вказують на місця виникнення проблеми лівого повороту. Так, наприклад, за час з 40 до 60 секунд по самій верхній смузі перехрестя залишили всього 3 машини (1 червона та 2 жовті)[41].

Моделюючи кілька тактів світлофора, отримуємо, що пропускна здатність даної ділянки дорожньо-транспортної мережі дорівнює ≈ 2220 машин на годину в один бік.

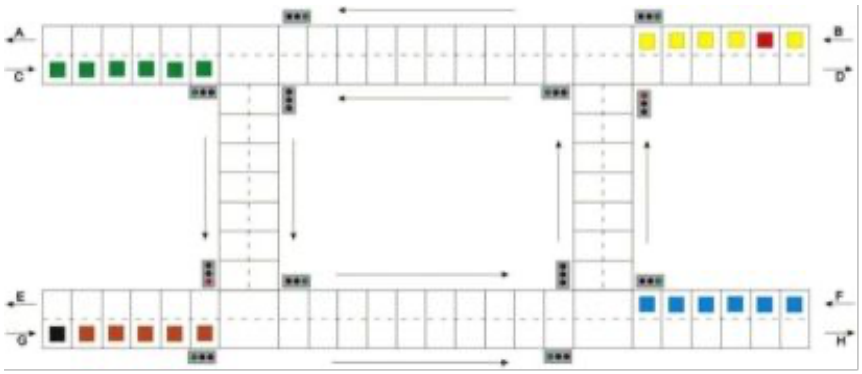


Рисунок 3.11. Стан під час t=0

Тепер розглянемо ситуацію, коли двосторонній рух замінили на односторонній, як це показано на, рис. 3.11.

Вихідні дані залишаються незмінними за винятком того, що з машини їдуть в E, а не в H; з G машини їдуть у D, а не в A.

Завдання: порахувати пропускну здатність даної ділянки.

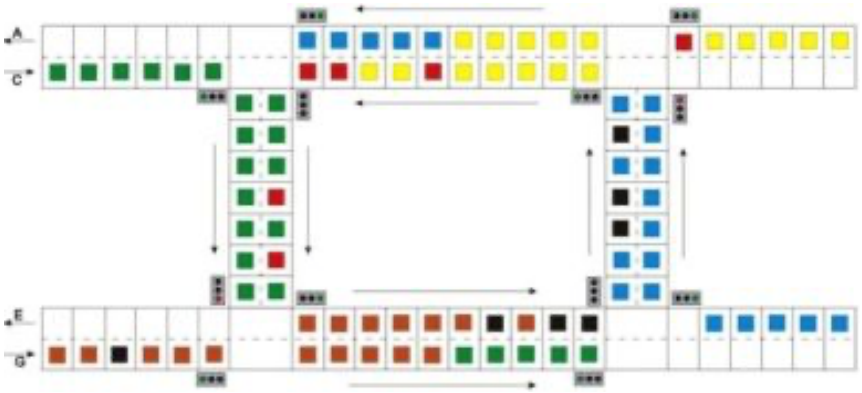


Рисунок 3.12. Приклад невірної орієнтації рокадних ділянок вулиць

На, рис. 3.12, показано стан у момент часу t=60с. В цьому разі затори виникають скрізь. Відбувається це через те, що надто великий потік машин надходить на вертикальні ділянки, що мають невелику місткість[51].

Моделюючи кілька тактів світлофора, отримуємо, що пропускна здатність даної ділянки дорожньо-транспортної мережі дорівнює ≈ 1860 машин на годину в один бік.

Рисунок 3.13. Стан в момент часу t=0

Тепер розглянемо ситуацію, коли двосторонній рух замінили на односторонній, як показано на, рис. 3.13. Тоді умова завдання перепишеться наступним чином: машини їдуть із B в A, але на кожні 15 машин зустрічаються 5, які їдуть в D; машини їдуть із C в D, але на кожні 15 машин зустрічаються 5, які їдуть в A[61].

Моделюючи кілька тактів світлофора, отримуємо, що пропускна здатність даної ділянки дорожньо-транспортної мережі дорівнює ≈ 3600 машин на годину в один бік.

Введення одностороннього руху не тільки вирішує проблему повороту ліворуч, а може як збільшити, так і зменшити пропускну здатність ділянки дорожньо-транспортної мережі залежно від структури потоків даної території міста. Таким чином, імітаційна модель дозволить обчислити оптимальну орієнтацію руху вулицями міста.

Також, варто зазначити, що при побудові моделі варто враховувати не тільки містобудівні фактори дорожньо-транспортної мережі, поведінку потоків та характеристики видів транспортних засобів, їх складових. Розроблено концепцію імітаційної моделі транспортних потоків та керуючих транспортних систем. У рамках імітаційної моделі проведено моделювання потоків на ділянці дорожньо-

транспортної мережі та виявлено загалом позитивний вплив впровадження одностороннього руху[71].

3.5 Завдання усунення заторів на маршруті під час керування транспортними потоками на мережі міських доріг

З кожним роком машин стає дедалі більше. Відповідно, зростає і кількість заторів. Ця проблема набуває характеру епідемії – всі великі міста світу стикаються з нею тією чи іншою мірою. Вражають цифри статистики – наприклад, в Києві вважається нормальним, якщо час проїзду від роботи до будинку складає близько години-півтори. Це лише в один бік. Неважко порахувати, скільки часу йде на тиждень/місяць/рік під час простою. Це втрата часу, як окремої людини, так економіки країни загалом. Крім цього, можна відзначити, і «порожню» витрату бензину, а отже, і зайве забруднення навколишнього середовища.

Управління дорожнім рухом доволі нетривіальне завдання. З одного боку, керування транспортним потоком, що складається з технічних засобів (автомобілів), з іншого – управляти доводиться живими людьми, кожен із яких має на свою мету.

Повне вирішення проблеми виникнення пробок може бути лише комплексним, що враховує не лише безпосереднє регулювання на перехрестях, але дещо більше. Для наочних прикладів звернемося до закордонного досвіду:

– Можна реалізувати почерговий рух залежно від якихось властивостей автомобілів. Наприклад, у багатьох країнах Євросоюзу (таких, як Греція) реалізовано такий механізм: по парних днях дозволено проїзд автомобілів з парними номерами, і непарними з непарними номерами.

– Створення у місті зон із платним проїздом. Наприклад тут можна навести Лондон, де вже близько 10 років діє тарифікація машин для проїзду через центр міста, надаючи тим часом пільги інвалідам і таксі. Стосовно Українських реалій для деяких великих міст (таких, як Київ та Харків), її ввести в принципі можна. І хоча за найоптимістичнішими прогнозами, це знизить транспортний потік не більше,

ніж на третину, це може стати додатковим джерелом фінансування дорожнього будівництва – нових розв'язок, ремонту доріг тощо.

– Вирішення проблеми з паркуваннями – нерідко можна спостерігати таку картину – машини припарковані на узбіччі (а найчастіше, і не в один ряд), що звичайно створює проблеми руху.

– Стосовно наших реалій, варто додати і обов'язкове підвищення культури водіння, без якої це так і залишиться «рекомендаціями», які більшість ігноруватимуть.

Нехай V непорожня безліч вершин, і $E \subseteq VxV$ – множина ребер, з умовою симетричності: пара $(u,w) \in E$ тоді і лише тоді, коли $(w,u) \in E$. Тоді сукупність G=(V, E) називається *неорієнтованим графом* .

Орграф – неорієнтований граф без умови симетричності. Вершини графа зазвичай називають вузлами, а дуги – орієнтованими ребрами.

Мультиграф – граф, у якому допускається більше одного ребра між вершинами.

Шляхом в орграфі називається послідовність вершин $v_1,..,v_n$(*) для яких існують дуги $v_i \rightarrow v_{i+1}$. *Довжиною шляху* (*) є n-1. Шлях називається *простим*, якщо в ньому всі вершини, за винятком, можливо, початкової і кінцевої, різні. *Цикл* – простий шлях довжини більший за 1. *Дерево* – це ациклічний граф. Існує кілька способів обходу всіх вузлів дерева (обхід дерева рівнозначний упорядкування за якимось правилом вузлів). Виділимо найбільш поширені:

Якщо дерево T – порожнє, то в список обходу заноситься порожній запис

Якщо дерево T складається з одного вузла, то в список обходу заноситься ця вершина.

Далі, нехай T дерево з коренем n та під деревами T_1, …, T_k

При *прямому обході* спочатку відвідується корінь n, потім усі вузли піддерева T_1, потім усі вузли піддерева T_2 і т.д. Останніми відвідуються вузли піддерева T_k

При симетричному обході спочатку відвідуються у симетричному порядку всі вузли піддерева T_1, потім корінь n, потім послідовно у симетричному порядку всі вузли піддерев T_2, …, T_k

Під час обходу у зворотньому порядку спочатку відвідуються у зворотному порядку всі вузли поддерева T_1, потім T_2, …, T_k, останнім відвідується корінь n.

Наприклад, у місті, для виділеного маршруту S-T ($S=X_1$, …, $X_k=T$), потрібно усунути всі затори шляхом зміни режимів роботи світлофорів, не створюючи додаткових заторів на бічних до цього маршруту дорогах[5].

Зупинимося на формулюванні трохи докладніше. По-перше, під затором ми розумітимемо перевищення вхідного потоку на перехресті над його пропускною здатністю. Крім того, звернемо увагу на другу частину постановки - про "не створення додаткових заторів на бічних напрямках", без неї завдання вирішувалося б тривіально – достатньо було б встановити зелене світло на всіх перехрестях маршруту S-T.

Вхідні дані:

а) місто зі світлофорами

б) S-T маршрут

в) PR – безліч пробок маршруту

г) K – глибина аналізу бічних доріг

д) I – число ітерацій алгоритму

е) T – максимальний час очікування на перехресті Розпишемо вхідні дані докладніше.

Місто є орієнтованим мультиграфом M(U, V), де в якості вершин U ми розглядаємо перехрестя, а як ребра V – можливі шляхи проїзду від одного перехрестя до іншого. Тобто, ребро $(X_i;X_j)\in M$ тоді і тільки тоді, коли дозволено рух від перехрестя X_i до перехрестя X_j.

Для кожного ребра задаються ваги[14]:

$M_{xj,\ Xj+1}$ відповідає максимальної (фізичної) пропускної спроможності вулиці за одиницю часу t у напрямку від X_j до X_{j+1}.

Приклад частини міста з трьох перехресть (X_1, X_2, X_3), із заданими максимальними пропускними здатностями вулиць. $Rx_{j,Xj+1}$ відповідає реальним даним про кількість машин за одиницю часу t у напрямку від X_j до X_{j+1}

Перейдемо до перехресть.

Розглянемо загальний випадок Перехрестя двох доріг (рис. 3.14).

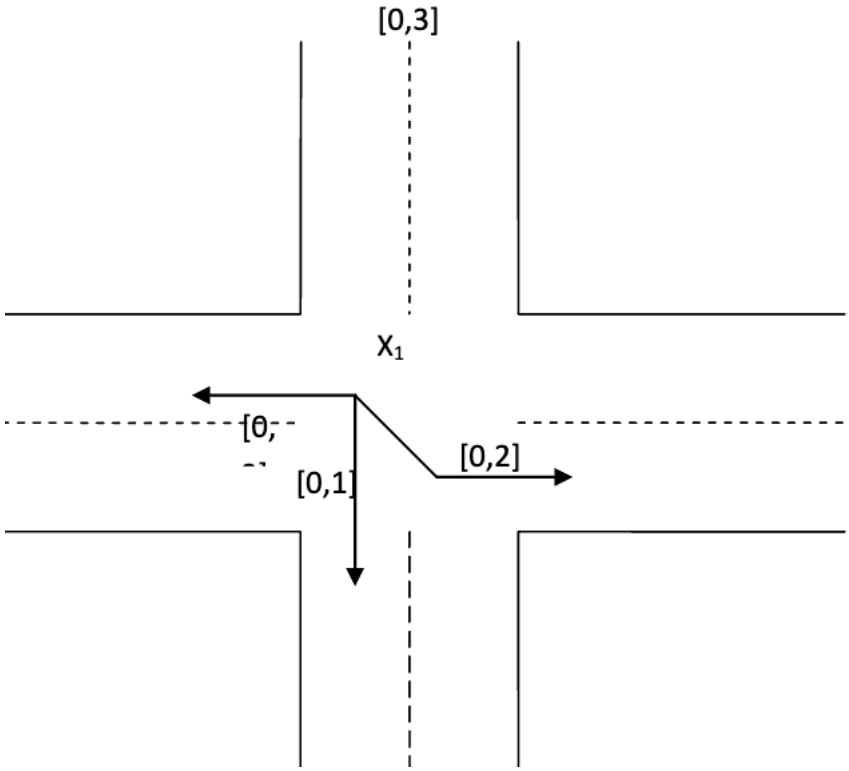


Рисунок 3.14 - Загальний випадок Перехрестя двох доріг

Наприклад розглянемо якийсь один напрямок. Нехай на перехресті можна рухатися у напрямках, зазначених малюнку I, II і III. Позначимо потоки, що проходять перехрестя під час горіння зеленого за P_i. Тоді час горіння зеленого та червоного для кожного з напрямків позначимо G_i та R_i одиниць часу t відповідно. Звідси, обчислимо «середню пропускну здатність» перехрестя[34]:

$$S_i = \left[\frac{P_i}{G_i + R_i}\right], \text{для } i = I, II, III$$

Для сумісності з позначеннями максимальної та реальної пропускної здатності вулиць, приймемо таке позначення для перехресть: $J_{X_j,X_n}^{X_{j+1}}$, де X_{j+1} – поточне перехрестя, X_j – «попереднє», а X_n – «наступне» перехрестя, та J літера, що позначає максимальну(M), або реальну(R) пропускну здатність. Тепер, переходячи від перехрестя до графа, отримуємо таку картинку 3.15.

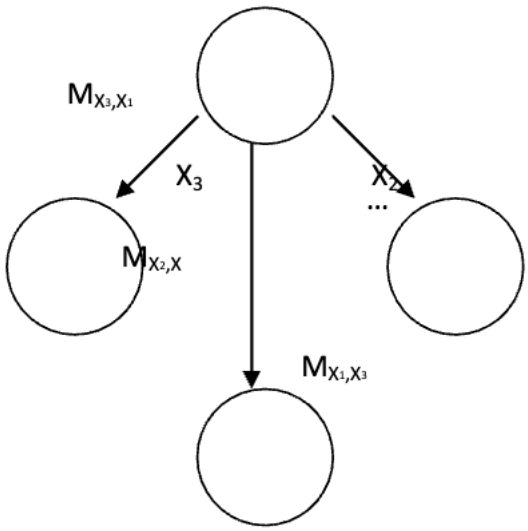


Рисунок 3.15 - Граф

Аналогічно визначаємо максимальну (фізичну) пропускну здатність напрямків[54].

Інші вхідні дані:

- S-T маршрут – послідовність вершин $X_1 \ldots X_k$, де існує шлях довжини 1 з X_i до X_{i+1}

PR – підмножина вершин маршруту, на перехрестях яких є затори

- Глибина аналізу K – висота дерева бічних доріг (потрібна для алгоритму)

- I – число ітерацій алгоритму

Алгоритм

I. Початкові дані алгоритму:

- J – "внутрішній" ітератор, що зберігає поточну вершину, в якій ми зараз знаходимося, спочатку $J=X_{k-1}$. Умовимося:

$J+1 = X_{i+1}$, якщо $J=X_i$ при $k-1 \geq i \geq 1$

$J-1 = X_{i-1}$, якщо $J=X_i$ при $k-1 \geq i \geq 1$

- COUNT – поточна кількість заторів. Спочатку COUNT=|PR|

- ITERATOR – "глобальний" ітератор, що зберігає номер поточної ітерації алгоритму, спочатку дорівнює 1.

- C – величина поточної різниці потоку, потрібна для передачі змін величини потоку між сусідніми вершинами, спочатку дорівнює 0

2. Хід алгоритму[64]:

Випадок 1. $J=X_{k-1}$ тобто ми робимо перший крок нашого алгоритму.

Крок 1. Зменшуємо бічні для нашого маршруту потоки на цьому перехресті

Крок 2. Перераховуємо режим роботи світлофора таким чином, щоб вихідний потік був найбільшим, не створюючи при цьому заторів на бічних дорогах і щоб він не перевищував $M_{J, J+1}$.

Позначимо отримані бічні потоки P_I та P_{II}, а максимальну пропускну здатність M_I та M_{II} відповідно. Тоді $M_{max}=\max\{M_I, M_{II}\}$, $P_{max}=\max\{P_I, P_{II}\}$. Тепер вирахуємо новий режим світлофора (для бічних доріг):

$$G = \frac{P_{max}T}{M_{max}-P_{max}}$$

R=T

У випадку $M_{max}=P_{max}$, пропускаємо це перехрестя (тобто не змінюємо режим роботи). Крім цього, може вийти, що G>T, у цьому випадку:

$$G = \frac{T(M_{max} - P_{max})}{P_{max}}$$

R=G,

Де R – час горіння червоного, а G – зеленого світла.

Крок 3. До C заносимо різницю між реальним вхідним потоком і тим потоком, який тепер може прийняти перехрестя, не створюючи затору, $C\leq 0$. Зауважимо, що у вершині J був затор, ми від нього позбулися, тобто у цьому випадку COUNT=COUNT-1

Крок 4. Переходимо до наступної вершини (J=J-1)

Випадок 2. Тепер нехай ми стоїмо у $J=X_i$ і J не збігається з вершиною X_{k-1}

Крок 1. Зменшуємо бічні потоки

Крок 2. Змінюємо режим роботи світлофора таким чином, щоб вихідний потік був $N_p=MIN\{M_{J, J+1}; R_{J, J+1} + C\}$.

Для цього:

Перераховуємо світлофор за допомогою формул з попереднього випадку.

Далі $R_{\text{ГЛ}}^{\text{СВ}} = \frac{R_{J-1,J}-N_pT'}{N_p}$, де $T' = min\left\{T, \frac{T(M_{max}-P_{max})}{P_{max}} \frac{R_{J-1,J}}{2N_p}\right\}$ $G_{\text{ГЛ}}^{\text{СВ}} = \text{T}'$

Крок 3. Далі, знову обчислюємо різницю між реальним вхідним потоком і вихідним, що вийшов, і заносимо в C. При цьому, якщо в поточній вершині був затор, то зменшуємо COUNT на 1

Примітка. Якщо перехрестя нерегульоване, то пропускаємо крок зміни режиму роботи світлофора і переходимо до наступної вершини.

III Мінімізація бічних потоків

Нехай ми стоїмо у вершині J. У загальному випадку, наш потік формується із трьох потоків, позначених як I, II, III (рисунок 3.16).

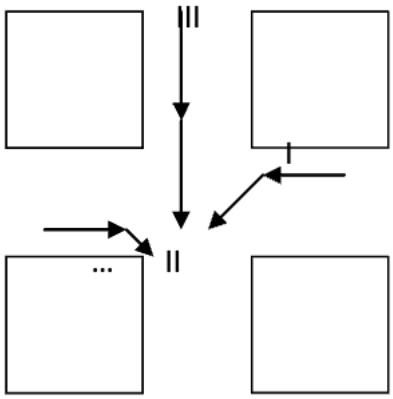


Рисунок 3.16 - Потоки

З них до бокових потоків відносяться лише I та II. Ці підвипадки однакові, тому розглянемо тільки один, I. Зауважимо, що як потік, I так само формується (загалом) з трьох: 1, 2, 3, і т.д. Будуватимемо дерево з цих потоків таким чином: у ролі кореня виступає потік I. Припишемо цій вершині вагу, що дорівнює величині цього потоку.

Тепер нехай у нас є дерево T, якщо його висота менша за K, то для кожного листа цього дерева задамо нащадків, що формують кожен лист, як потоки. Продовжуватимемо доти, доки висота дерева не стане рівною K.

Позначимо це дерево $Tree_J^I$.

Тепер наша мета – обійти дерево, зменшуючи ваги вузлів, щоб зменшити сумарний вплив бокового потоку (у нашому випадку, I) на маршрут S-T. Але зауважимо, що окремі вузли дерева можуть брати участь і під час аналізу сусідніх

перехресть. Крім того, ваги якоїсь частини вершин ми поміняти не зможемо, оскільки вони вже були перераховані попередніми алгоритмами ітераціями. Щоб встановити, які вершини підлягають зміні, а які ні, задамо розмальовку дерева за такими правилами:

Для початку задамо кольори – Червоний, Чорний та Синій. Червоний колір означатиме, що вершина підлягає перерахунку при обході, Чорний, що не підлягає, а Синій – що вершина вже була перерахована попередніми ітераціями (рис. 3.17).

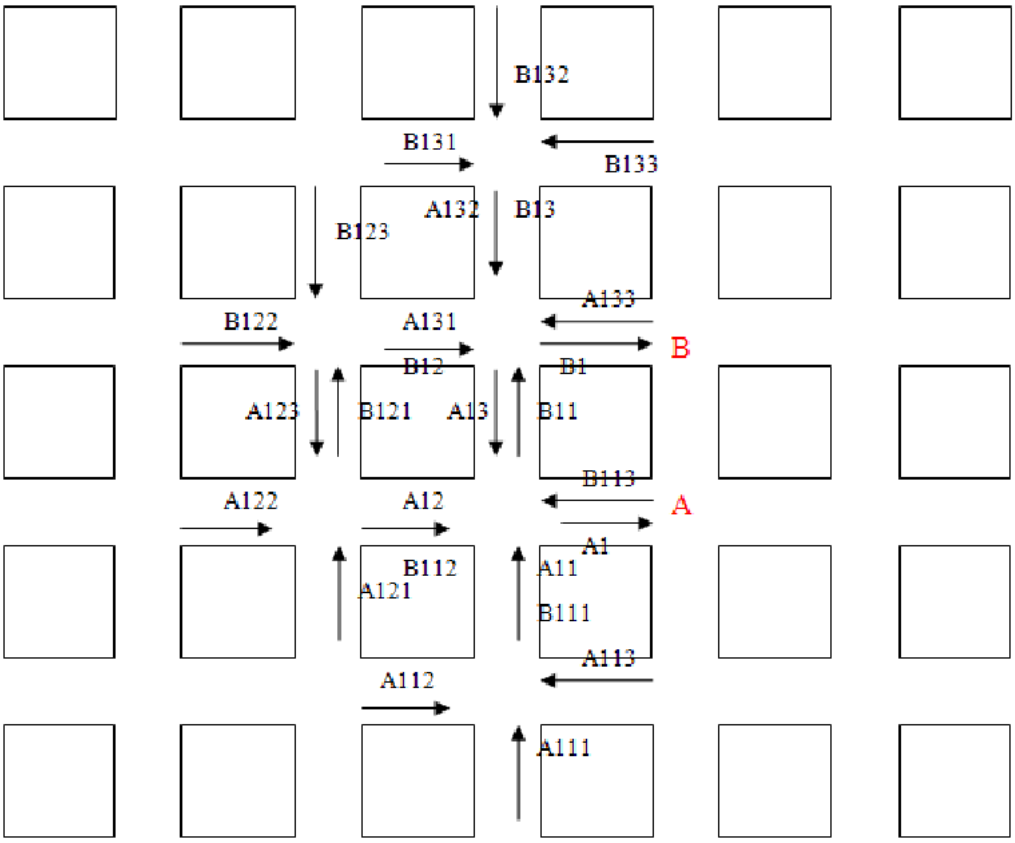


Рисунок 3.17 - Початковий варіант

Почнемо забарвлення дерева з кореня. Корінь завжди буде червоним. Зробимо зауваження – чим більше шлях від кореня до вершини, тим далі вона знаходиться від аналізованого перехрестя. Доречно зауважити, що в загальному випадку це означає, що така вершина бере участь у формуванні бокового потоку в меншій мірі, ніж вершина того ж дерева з меншим шляхом. Розглянемо два сусідні перехрестя. З малюнка видно, що частина елементів належать відразу дереву А і

дереву B (як, наприклад, A131/B12), тому кожна з цих вершин буде перерахована як мінімум два рази. Враховуючи зауваження, звідси можна вивести ще одне правило: при збігу вершин дерев сусідніх перехресть вершина з меншим номером фарбується в червоний колір, а з більшим – у синій. Крім того, фарбуватимемо вершину в чорний колір, якщо вона потрапляє на вже перераховані значення нашого маршруту. Виходячи з цього, зауважимо, що якщо в дереві T є піддерево з Синім/Чорним коренем, то і все піддерево має бути пофарбоване в чорний колір (рисунок 3.18).

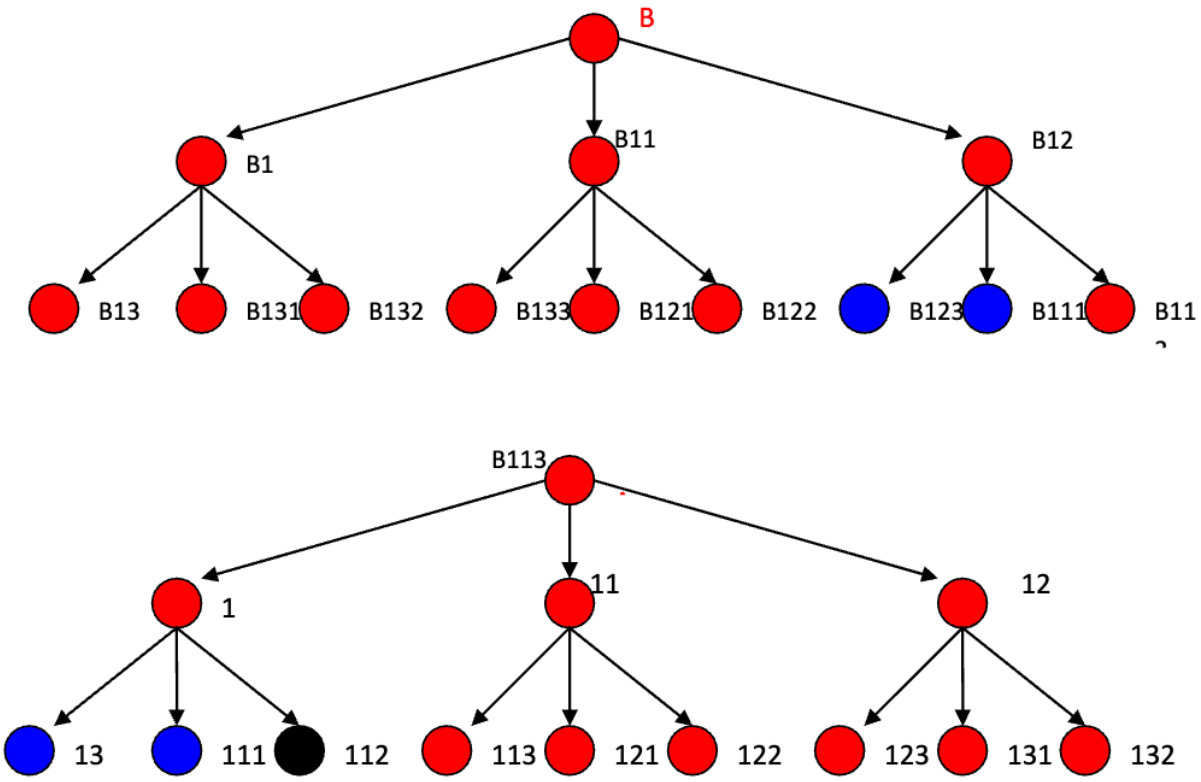


Рисунок 3.18 - Приклад розмальовки дерев для двох сусідніх перехресть при k=3

Далі, задамо функцію f, яка для кожного перехрестя видає допустиму зміну потоку – f(g, p, ITERATOR), де g – поточна відстань від нашого перехрестя, p – потік на цьому перехресті та ITERATOR – номер ітерації алгоритму.

Примітка – у загальному випадку, ця функція емпірична, оскільки ситуація на кожному перехресті різна і заздалегідь вказати який саме вона матиме

вигляд практично неможливо. Понад те, доречно цю функцію розглядати як своєрідного Оракула. Зупинимося трохи докладніше. Під Оракулом розуміється певний зовнішній механізм на ситуацію – наприклад, Суперкомп'ютер, який вираховує дані з урахуванням статистики. Далі, роблячи зворотній обхід дерева, застосовуємо до вершин функцію f.

Умови завершення

Одна ітерація нашого алгоритму може завершитись з двох причин:

У якийсь час C≤0 і COUNT=0.[89] У цьому випадку можна сказати, що наш алгоритм впорався із завданням, ми повертаємо оброблений маршрут

C>0 І ми у вершині S, тобто. виправити ситуацію повністю нам не вдалося. У цьому випадку ми збільшуємо ITERATOR на 1 і запускаємо наступну ітерацію, посилюючи умови. Тут знову ж таки можливі два варіанти –

а) або ми потрапимо у випадок 1 (тобто вирішимо задачу) після якогось числа ітерацій,

б) або в якийсь час ми перевищимо вхідні дані на число ітерацій. У цьому випадку ми повертаємо, що ситуація не можна вирішити в нашій моделі.

Результати показані у вигляді таблиці (таблиця 3.1).

Таблица 3.1 - Таблиця результів

Число ітерацій	Вид емпіричної функції	Глибина аналізу	Результат
2	F=числу ітерацій	2	Рішення знайдено
1	F=числу ітерацій	3	Рішення знайдено
5	F=1	2	Рішення не знайдено
1	F=1	3	Рішення знайдено
1	F=висоті піддерева для кожної вершини	2	Рішення знайдено
1	F=висоті піддерева для кожної вершини	3	Рішення знайдено

3.6 Завдання маршрутизації транспортними потоками міста із застосуванням теорії S-гіпермережі

Завдання маршрутизації транспорту на міських транспортних мережах дуже схоже на маршрутизацію комп'ютерного трафіку в мережах інформаційних. Головні відмінності полягають у тому, що в першому випадку як пакет розглядається транспортний засіб, а також існують правила дорожнього руху, що обмежують пересування таких пакетів. Таким чином, завдання маршрутизації транспорту полягає у тій же проблемі знаходження найкоротшого шляху між двома вузлами.

Маршрутизація може бути представлена у такому вигляді. Нехай дано спрямований зважений граф G=(V,E), в якому кожен вузол з множини V являє собою пристрій, що обробляє та передає дані, а кожне ребро з множини є лінією зв'язку. Основним завданням алгоритмів маршрутизації є передача даних із вузла джерела у вузол приймач, максимізуючи при цьому продуктивність мережі. Тут мається на увазі, наприклад, передача максимальної кількості пакетів за мінімальний час.

Алгоритми маршрутизації повинні виконувати такі функції:

- збір, організація та розподіл інформації про створений користувачем трафік та стан мережі;

- використання зібраної інформації для створення відповідних маршрутів, що максимізують продуктивність об'єктів

- направлення трафіку користувача за вибраним маршрутом. Спосіб реалізації описаних трьох функцій сильно залежить від технології передачі та комутації пакетів, покладеної в основу мережі, і від особливості інших взаємодіючих рівнів додатків. Надсилання трафіку користувача може відбуватися з використанням двох базових операцій мережі: комутація каналів та комутація пакетів (які також пов'язані з поняттями орієнтований та неорієнтований на з'єднання). При комутації каналів на стадії встановлення з'єднання шукаються та резервуються ресурси мережі, які згодом будуть надані кожній новій сесії. У цьому випадку всі пакети

даних, що належать до однієї і тієї ж сесії, будуть направлені по тому самому шляху. Від маршрутизаторів потрібне зберігання інформації про активну сесію. При комутації пакетів немає стадії резервування, інформація про стан не зберігається на маршрутизаторах, і пакети даних можуть надсилатися різними шляхами. У кожному проміжному вузлі приймається самостійне рішення про вибір вихідної лінії, якою буде відправлено пакет даних у вузол приймач.

Загальним параметром всім видів алгоритмів маршрутизації є таблиця маршрутизації. Таблиця маршрутизації розміщується в кожному вузлі мережі, і містить всю інформацію про неї. Ця інформація, у свою чергу, використовується маршрутизаторами для створення маршрутів відправлення пакетів даних. Тип інформації, що міститься в маршрутних таблицях, залежить лише від алгоритму маршрутизації.

Алгоритми маршрутизації можна класифікувати так:

- централізовані та розподілені;

- Статичні та адаптивні.

У централізованих алгоритмах головний керуючий пристрій відповідає за оновлення таблиць маршрутизації всіх вузлів та/або приймає кожне рішення про маршрутизацію. Централізовані алгоритми можуть бути використані тільки в окремих випадках і для малих мереж. Загалом, затримки, необхідні для збору інформації про стан мережі та для трансляції запиту на оновлення даних, роблять такі алгоритми незастосовними на практиці. Більше того, централізовані системи не є стійкими до відмови. У статичних системах маршрутизації шлях, який проходить пакет, визначається тільки на основі його джерела та приймача, без розгляду поточного стану мережі. Цей шлях зазвичай вибирається як найкоротший щодо обраного вартісного критерію, і може бути змінений лише за рахунок пошкоджених ліній зв'язку або вузлів.

Адаптивні маршрути, в принципі, більш привабливі, оскільки вони можуть адаптувати спосіб маршрутизації до тимчасових та просторових змін трафіку. Як недолік такого підходу виділяють те, що занадто часті зміни в мережі можуть спричинити коливання у вибраних шляхах. Ця обставина, у свою чергу, може

призвести до створення циклічних шляхів, а також великих відхилень у виконанні алгоритму. До того ж адаптивна маршрутизація може призвести до суперечливих ситуацій, які можуть виникнути при виході з ладу вузлів, ліній зв'язку або зміни локальної топології. Проте, всі ці проблеми стійкості найбільш характерні для мереж неорієнтованих на з'єднання.

Інший спосіб класифікації алгоритмів маршрутизації має вигляд:

- Мінімальна маршрутизація та немінімальна маршрутизація;

-Оптимальна маршрутизація та маршрутизація, що визначає найкоротший шлях.

Мінімальні маршрути дозволяють пакетам вибирати тільки шляхи з мінімальною вартістю, тоді як немінімальні алгоритми дозволяють робити вибір між усіма доступними шляхами, використовуючи деякі евристичні стратегії.

Оптимальна маршрутизація має доступ до всієї мережі та всіх її об'єктів для оптимізації функції всіх окремих потоків ліній зв'язку (зазвичай ця функція є сумою вартості шляхів, призначених відповідно до середніх затримок пакетів).

Маршрутизація, що визначає найкоротші шляхи, об'єктивно визначає найкоротший шлях (мінімальну вартість) між двома вузлами. Враховуючи зміст у таблицях маршрутизації різної інформації, алгоритми знаходження найкоротших шляхів можуть бути поділені на два класи: дистанційно-векторні та алгоритми стану зв'язку.

Оптимальна маршрутизація є статичною та вимагає знання всіх характеристик трафіку. Алгоритми знаходження найкоротших шляхів гнучкіші, вони не вимагають апріорних знань про моделі трафіку, і, на даний момент, найбільш поширені серед алгоритмів маршрутизації

3.7 Опис та аналіз методів маршрутизації

Перш ніж розпочати розгляд конкретних алгоритмів маршрутизації, сформулюємо принцип оптимальності. Цей принцип стверджує, що й вузол комутації (ВК) J перебуває в оптимальному шляху між ВК I і K, то оптимальний

маршрут між J і K належить цьому оптимальному шляху. Це так, оскільки існування між J і K оптимального маршруту відмінного від частини маршруту між I і K суперечив би твердженню про оптимальність маршруту між I і K. Справа в тому, що якщо розглянути (Рисунок 3.19) маршрут від I до K як від I до J (назвемо його S1) і від J до (назвемо його S2), то якщо між J і K є маршрут краще, ніж S2, наприклад S3, то маршрут S1S2 не може бути кращим. Взявши конкатенацію маршрутів S1S3 ми отримаємо кращий маршрут, ніж маршрут S1S2.

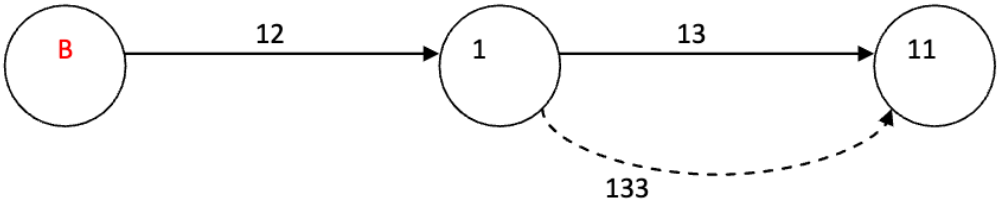


Рисунок 3.19 – Маршрут

Наслідком із принципу оптимальності є твердження, що всі маршрути до заданої точки мережі утворюють дерево з коренем у цій точці. Це дерево називається деревом заходу.

Оскільки дерево заходу - це дерево, там немає циклів так, що кожен пакет буде доставлений за кінцеве число стрибків. Насправді все може бути складніше. Маршрутизатори можуть виходити з ладу і навпаки з'являтися нові, канали можуть виходити з ладу, різні маршрутизатори можуть дізнаватися про ці зміни в різний час і т.д. і т.п.

Маршрутизація найкоротшим шляхом

Ідея цього алгоритму полягає у побудові графа транспортного середовища, де вершини – маршрутизатори, а дуги – лінії зв'язку. Алгоритм знаходить для будь-якої пари маршрутизаторів, а точніше абонентів, підключених до цих маршрутизаторів, найкоротший маршрут у цьому графі.

Відстань можна вимірювати у стрибках, а можна у кілометрах. Можливі і інші заходи. Наприклад, дуги графа можуть бути розмічені вагами, величина яких дорівнює середній затримці для пакетів у відповідному каналі. У графі з такою

розміткою найкоротший шлях - найшвидший шлях, хоча він не обов'язково має мінімальну кількість стрибків або кілометрів. У загальному випадку ваги на дугах можуть бути функціями від відстані, пропускної здатності каналу, середнього трафіку, вартості передачі, середньої довжини черги в буфері та інших факторів. Змінюючи вагову функцію, алгоритм обчислюватиме найкоротший шлях у сенсі різних заходів.

Відомо кілька алгоритмів обчислення найкоротшого шляху у графі. Один із них запропонував Дейкстра (Dijksra). Ідею цього алгоритму можна описати так. Усі вершини у графі позначаються відстанню (воно зазначено у дужках) до вихідної вершини вздовж найкращого, відомого шляху. Спочатку ніяких шляхів не відомо і всі вершини позначені нескінченністю. У міру роботи алгоритму та знаходження шляхів, мітки можуть змінюватися. Мітки можуть бути двох видів або пробними або постійними. Спочатку всі мітки пробні. Коли виявляється, що мітка представляє найкоротший шлях до вихідної вершини, вона перетворюється на постійну і більше не змінюється.

Маршрутизація лавиною

Іншим прикладом статичного алгоритму може бути алгоритм переповнення. У цьому алгоритмі, кожен пакет, що надходить, відправляють по всіх наявних лініях, за винятком тієї, по якій він надійшов. Зрозуміло, що якщо нічим не обмежити кількість повторно генерованих пакетів, їх кількість може зростати не обмежено. Для цього в заголовку кожного пакета, що спочатку генерується, встановлюється лічильник стрибків. При кожному пересиланні цей лічильник зменшується на одиницю. Коли він досягає нуля, пакет скидається і далі не надсилається. Як початкове значення лічильника вибирають найгірший випадок, наприклад, діаметр транспортної підмережі.

Іншим прийомом, що обмежує зростання пакетів, що дублюються, є відстеження на кожному маршрутизаторі тих пакетів, які через нього одного разу проходили. Такі пакети скидаються та більше не пересилаються. І тому кожен маршрутизатор, отримуючи пакет безпосередньо від абонентської машини, позначає його належним числом. У свою чергу, кожен маршрутизатор веде список

номерів, згенерованих іншим маршрутизатором. Якщо пакет, що надійшов, вже є в списку, то цей пакет скидається. Для запобігання безмежного росту списку вводять обмежувальну константу k. Вважається, що всі номери, починаючи з k і далі, вже зустрічалися.

Маршрутизація на основі потоку

Алгоритми, які були розглянуті досі, брали до уваги лише топологію транспортного середовища та ніяк не враховували її завантаження. Хоча, наприклад, у тому випадку, коли найкоротший маршрут перевантажений, очевидно, краще скористатися нехай довшим, але менш завантаженим маршрутом. Тут ми розглянемо статичний алгоритм маршрутизації на основі потоку, який враховує як топологію, так і завантаження транспортної підмережі.

У деяких мережах трафік між кожною парою вузлів відомий заздалегідь і відносно стабільний. Наприклад, взаємодія мережі торгуючих організацій зі складом. Час подання звітів, розмір та форма звітів відомі заздалегідь. У цих умовах, знаючи пропускну здатність каналів, можна за допомогою теорії масового обслуговування обчислити середню затримку пакета в каналі. Тоді не важко побудувати алгоритм, що обчислює шлях із мінімальною затримкою пакета між двома вузлами.[99]

Для реалізації цієї ідеї потрібно знати заздалегідь про кожне транспортне середовище:

- топологію,
- матрицю трафіку Fij,
- матрицю пропускних здатностей Cij,
- алгоритм маршрутизації.

Маючи ці дані, будується алгоритм обчислення найкоротшого шляху з погляду вагів на дугах графа. Якщо трафік змінитися з будь-якої причини, достатньо перерахувати таблицю, не змінюючи алгоритму.

Маршрутизація по вектору відстані

Усі сучасні транспортні підмережі використовують динамічну маршрутизацію, а не статичну. Один із найбільш популярних алгоритмів –

маршрутизація по вектору відстані. Цей алгоритм побудований на ідеях алгоритмів Беллмана-Форда та Форда-Фолкерсона. Він спочатку використовувався в мережі ARPA і використовується до цього дня під назвою алгоритму RIP (метод рельєфів). Він використовується у мережах Novell, AppleTalk, Cisco маршрутизаторах.

В основі його лежить ідея, що кожен маршрутизатор в транспортній підмережі має таблицю відстаней до кожного маршрутизатора в підмережі. Періодично маршрутизатор обмінюється такою інформацією зі своїми сусідами та оновлює інформацію у своїй таблиці. Кожен елемент таблиці і двох полів: перше - номер лінії, якою треба відправляти пакети, щоб досягти потрібного місця, друге - величина затримки до призначення. Ця величина затримки може бути виміряна в різних одиницях: стрибках, мілісекундах, довжині черги на лінії і т.д.

Кожні Т секунд маршрутизатор надсилає своїм сусідам свій вектор затримок до всіх маршрутизаторів у підмережі. У свою чергу він отримує такі ж вектори від своїх сусідів. Крім того, він постійно заміряє затримки до своїх сусідів. Тому, маючи вектор відстаней від сусідів і знаючи відстань до сусідів, маршрутизатор завжди може обчислити найкоротший маршрут.

Алгоритм маршрутизації по вектору відстані теоретично працює добре, але має один недолік: він дуже повільно сходить до правильного значення. Інформація про появу хорошого маршруту в підмережі поширюється більш-менш швидко, а ось дані про втрату, руйнування якогось маршруту поширюються не так швидко.

Маршрутизація по стану лінії

Алгоритм маршрутизації вектором відстаней використовувався в мережі ARPANET до 1979 року, після чого він був замінений. Тому було дві основні причини. Перша, оскільки основною мірою затримки була довжина черги, пропускна спроможність каналу ніяк не враховувалася. Поки переважна більшість каналів мала пропускну спроможність 56Кб/с, це було не страшно. Однак, коли з'явилися канали на 230 Кб/с та 1.5 Мб/с, це стало недоліком. Друга проблема – повільна збіжність алгоритму за змін. З цих причин було створено новий алгоритм маршрутизації за станом ліній.

Основна ідея побудови цього алгоритму проста і складається з п'яти основних

кроків:

а) Визначити своїх сусідів та їх мережеві адреси;

б) Виміряти затримку чи оцінити втрати до кожного сусіда;

в) Сформувати пакет, де вказані всі дані, отримані на кроці 2;

г) Надіслати цей пакет усім іншим маршрутизаторам;

д) Обчислити найкоротший маршрут для кожного маршрутизатора.

Топологія та всі затримки дійсно оцінюються експериментально та повідомляються всім вузлам. Після цього можна використати, наприклад, алгоритм Дейкстри для обчислення найкоротшого маршруту.

3.8 Розв'язання задачі маршрутизації транспорту із застосуванням теорії S-гіпермережі

В даний час із зростанням населення великих міст значно збільшилася кількість автомобілів на дорогах. Завантаженість міських автомагістралей посилилася, також неминуче зросла ймовірність аварій, з вини якої рух сповільнюється або взагалі зупиняється. Разом з тим людина, природно, хоче дістатися куди-небудь якомога швидше. Або вона просто їздить з роботи додому і назад і не хоче витрачати свій час на «стояння» в пробці, або це кур'єр, або працівник екстреної служби, для якого кожна хвилина на рахунку.

Одним з виходів може стати система управління маршрутизацією транспорту, що володіє найсвіжішою інформацією про стан доріг і здатна обчислити найкоротший шлях через все місто. Таким чином, автомобілісти зможуть уникнути пробок і досягти місця призначення в найкоротші терміни. Розглянемо ідею роботи системи керування маршрутизацією транспорту (рис. 3.20).

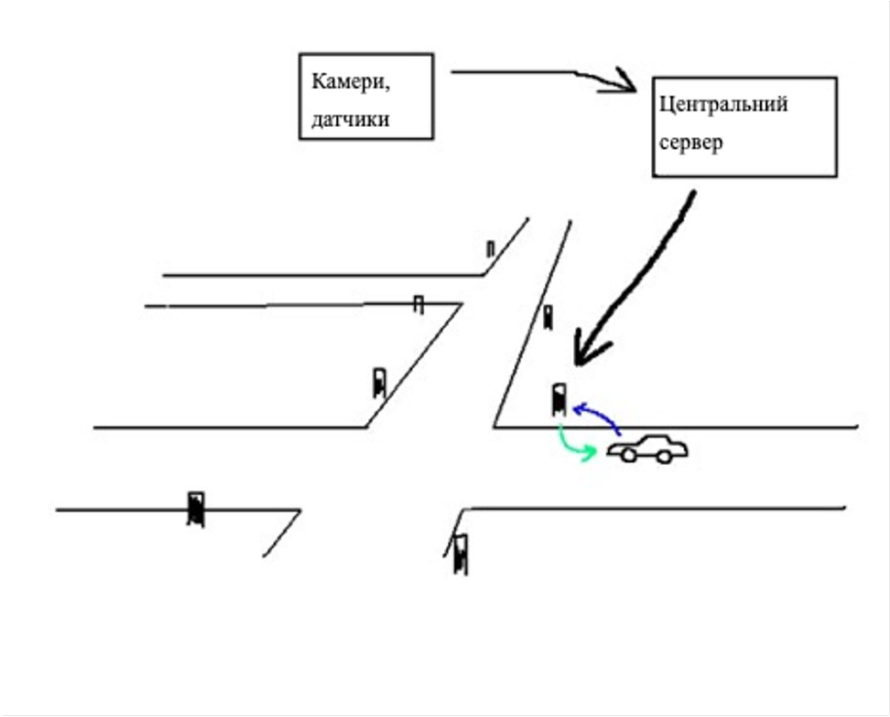


Рисунок 3.20 -Робота системи управління маршрутизацією транспорту

На під'їзді до перехресть встановлені спеціальні пристрої прийому-передачі, пов'язані з центральним сервером. На центральний сервер періодично надходить інформація з камер/датчиків, розставлених на дорогах, які стежать за станом дорожньої мережі. Ці камери здатні заміряти швидкість руху потоку по ділянках дороги. Таким чином, центральний сервер дізнається про зміну умов руху на якійсь ділянці дороги і відправляє відповідну інформацію на пристрої прийому-передачі. Автомобіль обладнаний спеціальною міткою. При під'їзді до перехрестя, який є фактично вузлом комутації, пристрій прийому-передачі зчитує інформацію про автомобіль. Ця інформація може бути зашифрована у вигляді <номер автомобіля, точка призначення>. Потім автомобіль отримує направлення до наступного вузла і продовжує рух.

Як вже було сказано вище, ми будемо розглядати централізований адаптивний алгоритм управління маршрутизацією, так як це дозволяє знизити витрати, оскільки немає необхідності оснащувати пристрої прийому-передачі процесорами великої потужності. Також необхідно враховувати затримки при обміні інформацією між пристроєм прийому-передачі і центральним сервером. Однак швидкість руху автомобілів набагато менше швидкості поширення електронна

сигналу, так що неприємностей можна уникнути, розставивши пристрої прийому-передачі відповідним чином.

Математина постановка задачі маршрутизації

Математично задача маршрутизації взагалі і транспорту зокрема зводиться до знаходження найкоротшого шляху в (не) орієнтованому графі.

Міська транспортна мережа — $G = (N,A)$ — зважений орієнтований граф, де N — безліч вершин (перехресть) A — безліч дуг (ділянок доріг, що з'єднують дві вершини кожна).

Функція $c : A \to R+$ визначає вартість кожної дуги $(i,j) \in A$. Метрикою в нашому випадку є час проходження пакета по дузі.

Таким чином, цільова функція має такий вигляд: (1)

$$min \sum_{(i,j)\in A} c_{ij}x_{ij}\,, x_{ij} \in \{0,1\}$$

$$\sum_{(j,i)\in BS(i)} x_{ji} - \sum_{(i,j)\in FS(i)} x_{ij} = \begin{cases} -n+1, & i=r \\ 1, & i\in N\setminus\{r\}; \end{cases}$$

$BS(i)$ *і* $FS(i)$ — це відповідні зірки вихідних і вхідних шляхів в вершину i, тобто:

$BS(i) = \{(u,v) \in A \mid v = i\}$,

$FS(i) = \{(u,v) \in A \mid u = i\}$;

Таким чином, вирішується целочисленне завдання булевого лінійного програмування - мінімізація сумарного часу шляху від вузла r до вузла i.

Існують різні алгоритми рішення подібного типу завдань.

Метод Дейкстри (Dijkstra).

Алгоритм Дейкстри - це класичний алгоритм для пошуку найкоротших шляхів від однієї вершини графа до всіх інших. Нехай $G=(V,E)$ — зважений орієнтований граф з невід'ємними вагами дуг, $s \in V$ – джерело, t — сток, і $l(v,w)$ довжина дуги $(v,w) \in E$. Тоді алгоритм виглядає наступним чином:

Алгоритм 1

1. Нехай U порожня множина, і потенціали $p(v)=+\infty$ для кожної вершини $v \in V$, крім $p(s)=0$.

2. Додаємо до U вершину v_0, що має мінімальний потенціал в $V\backslash U$. Якщо $t \in U$, стоп.

3. Для кожної вершини $v \in V$ такої, що $(v_0,v) \in E$, якщо $p(v_0)+l(v_0,v)<p(v)$, покласти $p(v):=p(v_0)+l(v_0,v)$ і рухаємось до вершини $v(v:=v_0)$.

4. Перейти на шаг 2.

Мінімальний час роботи алгоритму є $O(|E|+|V|log(|V|)$, що використовує метод Дейкстри для пошуку найкоротшого шляху на графі між двома вершинами.

Алгоритм А *.

А * («А-зірочка») - це розширення методу Дейкстри, евристичний алгоритм для пошуку найкоротшого шляху між двома вершинами. Він використовує евристичне наближення для довжини найкоротшого шляху від кожної вершини до вершини-призначенням. Нехай $h(u,v)$ — оцінювач довжини найкоротшого шляху між u та v. І нехай t — точка призначення. Тоді алгоритм діє за такою схемою:

Алгоритм 2

1. Нехай U пуста множина, і потенціали $p(v)=+\infty$ для кожної вершини $v \in V$, крім $p(s)=0$.

2. Додаємо к U вершину v_0, для якої $p(v)+h(v_0,t)$ мінімально в $V\backslash U$. Якщо $v_0=t$, стоп.

3. Для кожної вершини $v \in V$, такої, що $(v_0,v) \in E$, якщо $p(v_0)+l(v_0,v)<p(v)$, покласти $p(v):=p(v_0)+l(v_0,v)$, переміщуємося в вершину $v(v:=v_0)$ і видаляємо v із V, якщо $v \in V$.

4. Перейти на шаг 2.

Якщо $h(v,t)$ задовольняє обмеженню, $h(v,t)$ що є нижня границя $h^*(v,t)$, то отриманий путь обов'язково буде оптимальним найкоротшим шляхом.

$\forall v \in V\ h(v,t) \leq h^*(v,t)$ (3)

Слід зазначити, що якщо $h(v,t) \leq h^*(v,t)\ \ \forall v \in V$, алгоритм А* відразу знайде тільки дуги найкоротшого шляху від джерела до стоку. Більш того, видалення вершини із U на кроці 3 може бути виключено з алгоритму, якщо наближення задовольняє наступному обмеження, має назву монотонним обмеженням:

$\forall(u,v) \in E\ l(u,v) + h(v,t) \geq h(u,t)$ (4)

В цьому випадку оцінювач називається здійсненним для двоїстої задачі (dual feasible estimator). Наприклад, для евклидово відстані на дорожньої мережі - оцінювач для двоїстої задачі виконаємо, тобто двоїста задача здійсненна. Очевидно, що $h^*(v,t)$ також задовольняє верхньому обмеженню. Слід зазначити, що число знайдених вершин в цьому випадку завжди не перевищує числа знайдених вершин методом Дейкстри. Фактично метод Дейкстри - це метод А*, у якого $h(v,t)=0$ для всех вершин.

Двонаправлений метод (The bidirectional Method)

Двонаправлений метод також використовується для пошуку найкоротшого шляху між двома вершинами графа. Він не вимагає евристичної оцінки, але може зменшити число знайдених вершин в більшості випадків. У цьому алгоритмі пошук ведеться не тільки від джерела до стоку, а й у зворотній бік. Алгоритм працює наступним чином:

Алгоритм 3

1. Нехай U і W — порожні множини, і нехай потенціали $p_s(v)$ і $p_t(v)$ є рівними $+\infty$ для кожної вершини $v \in V$, за винятком $p_s(s)=0$ і $p_t(t)=0$.

2. Додаємо к U вершину v_0, яка має найменший потенціал $p_s(v)$ в $V \backslash W$. Якщо $v_0 \in U$, перейти на крок 7.

3. Для кожної вершини $v \in V$, такої, що $(v0,v) \in E$, якщо $p_s(v_0)+l(v_0,v)<p_s(v)$, припустимо $p_s(v):=p_s(v_0)+l(v_0,v)$ і розглядаємо попередню s вершину V в якості v_0.

4. Додаємо к W вершину v_0, яка має найменший потенціал $p_t(v)$ в $V \backslash W$. Якщо $v_0 \in W$, перейти на крок 7.

5. Для кожної вершини $v \in V$, такої, що $(v,v_0) \in E$, якщо $p_t(v_0)+l(v,v_0)<p_t(v)$, припустимо $p_t(v):=p_t(v_0)+l(v,v_0)$ і розглянемо попередню t вершину V в якості v_0.

6. Перейти на шаг 2.

7. Знайти дугу $(u_0,w_0) \in E$, яка має найменше значення $p_s(u)+l(u,w)+p_t(w)$. Найкоротший шлях від s до t складається з найкоротшого шляху від s до u_0, дуги (u_0,w_0) і найкоротшого шляху від w_0 до t.

Пропонується розбити задачу управління маршрутизацією транспорту на дві частини: статичну і динамічну.

Якщо розглядати міську транспортну мережу у вигляді графа, то з'являється проблема, пов'язана з великою розмірністю системи. Полегшити обчислювальну задачу може розбиття міської мережі на менші сегменти, пов'язані між собою через єдині вузли. Тобто такі вузли, які не можна минути при русі з однієї точки міста в іншу. При такій розбивці найкоротші шляхи не зміняться, тому що тут буде працювати принцип оптимальності.

Далі, при первісному прокладанні маршруту пропонується скористатися статистичними даними. Це середні швидкості транспортного потоку на всіх ділянках доріг в залежності від дня тижня, часу доби, можливо погоди. За цими даними обчислюються ваги ребер графа, і найкоротший маршрут обчислюється за допомогою будь-якого з алгоритмів пошуку найкоротшого шляху на графі.

У динамічної частини передбачається власне управління маршрутизацією транспортних засобів центральним сервером в режимі реального часу. При цьому пропонується використовувати аналог методу рельєфів для інформаційних мереж.

Метод рельєфів - це динамічний децентралізований адаптивний алгоритм. Його зручність полягає в тому, що в цьому випадку пакету дається напрямок руху до наступного вузла, що дуже зручно в разі маршрутизації автомобільного транспорту. Таким чином, водієві транспортного засобу буде дана рекомендація по вибору подальшого напрямку. І при під'їзді наступного автомобіля не буде потрібно ніяких нових обчислень, за умови збереження тих же параметрів пропускної здатності доріг.

3.9 Завдання про розміщення мінімальної кількості відеокамер на заданій транспортній мережі

Автомобільні затори одна з найнагальніших проблем сучасних мегаполісів, перенасичених автотранспортом. Ускладнений рух став однією з невід'ємних рис життя великого міста. Тому практично в кожному промислово розвиненому місті

цікавить впровадження інтелектуальних транспортних систем (ІТС). Такі системи призначаються для збільшення пропускної здатності вулиць та магістралей, підвищення швидкості сполучення та безпеки руху транспорту, скорочення затримок на перехрестях, зниження витрати паливно-мастильних матеріалів, оздоровлення екологічної обстановки. Основа всіх інтелектуальних транспортних систем: безперервний збір інформації про завантаженість доріг, швидкість потоків, аварії та умови для руху машин, яка далі обробляється і зручно доводиться до відома водіїв або використовується для регулювання руху.

Відеокамери – одні з найпростіших, звичних та дієвих засобів, що використовуються в моніторингу. Перевага відеокамер перед іншими системами стеження в тому, що можна безпосередньо побачити все, що знаходиться в огляді камери. Сучасне спеціалізоване програмне забезпечення обробки зображень від відеокамер дозволяє вимірювати параметри потоку, що рухається (наприклад, швидкість руху автомобілів, щільність потоку руху), визначати державні реєстраційні знаки автомобілів, виявляти порушення правил дорожнього руху. Надалі ця інформація може бути використана також у різних цілях. З розвитком цифрової техніки та здешевленням відеокамер з'явилася можливість організовувати відеоспостереження (з подальшим розвитком ІТС) у містах із застосуванням мінімальної кількості камер за умови найповнішого спостереження за міською транспортною мережею. Ця робота передбачає реалізацію саме цього підходу.

Для побудови адекватної моделі потрібна деяка початкова інформація про проблемні ділянки у місті, швидкісні обмеження, середня завантаженість окремих ділянок залежно від часу доби, пори року. Крім цього, потрібна карта міста та знання відстані від однієї точки до іншої.

На першому етапі територія міста візуально ділиться на райони області з густою дорожньої мережею, з'єднані з іншими районами магістральними дорогами. Розміщення камер в одному районі не залежить від розміщення в іншому. Таким чином, завдання дещо спрощується, ділиться на кілька підзадач меншої розмірності.

За тим же принципом можна розбити район на мікрорайони. Наступним кроком є визначення доріг у районі, на яких зосереджено основну масу транспортного потоку, а також аварійних ділянок. З спостереження слід виключити дороги, які не дають суттєвого вкладу в потік. До них належать дороги з малим завантаженням, у яких виникнення аварійних ситуацій не змінює характеру руху, внутрішні дороги житлових зон. Не слід виключати ділянки, що безпосередньо примикають до магістралей.

Наступним завданням буде визначення на одержаній території місць можливого розташування камер. Зручно розташовувати їх так, щоб у колі огляду знаходилося якнайбільше ділянок, які потрібно простежувати. Такими точками є передусім перехрестя, повороти, дахи або кути будівель, висотні споруди. Після того, як основні точки розміщення визначені, потрібно розділити всю необхідну територію на багато ділянок, що переглядаються кожною камерою, і при необхідності додати або прибрати камери.

Розглянемо можливі ситуації.

1. Часткове простеження

Така ситуація характерна для довгих проміжків між перехрестями камери частково покривають ділянку. Якщо потрібно простежувати ділянку по всій довжині, потрібно додати точки розміщення камер залежно від довжини ділянки. Найпростішим варіантом у такому разі є розміщення через відстань, як показано на рис 3.21., що дорівнює двом радіусам огляду камери (або одному радіусу, якщо камера не може здійснювати кругове обертання).

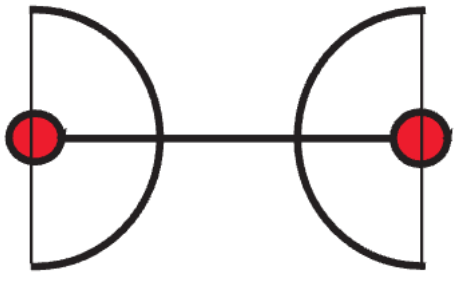


Рисунок 3.21 - Часкове простеження

Якщо ця ділянка не є аварійною або не вимагає простежування по всій довжині, то двох камер буде достатньо, так як про аварійну ситуацію всередині проміжку можна судити за другорядними ознаками (наприклад, порівнюючи щільність потоку на кінцях проміжку).

На довгих неаварійних ділянках магістралей, до яких примикають тільки другорядні дороги, також не потрібно повне простеження і камери зручно розміщувати або на поворотах, або через певну відстань, яка може бути набагато більшою, ніж радіус огляду камери.

2. Часткове перекриття

Одна і та ж ділянка може різною мірою простежуватися двома і більше камерами. При незначному перекритті (незначність перекриття може визначатися, наприклад, у відсотковому співвідношенні або в одиницях довжини і її конкретне значення залежить від ділянки, що розглядається) можна розділити ділянку, що покривається, на потрібну кількість підділянок і вказати для кожної камери території які їй належить спостерігати. Якщо перекриття істотне, можна вважати, що обидві камери простежують ділянку повністю. Приклади таких ситуацій показано на рис. 3.22.

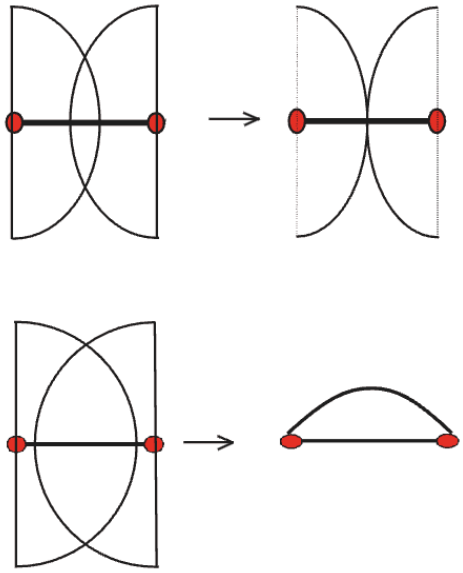


Рисунок 3.22 - Незначне і значне перекриття

3. Закруглені ділянки

Для закруглених ділянок дороги застосовується або осциляція, або є точка

всередині закруглення, з якою безперешкодно простежується весь поворот (рисунок 3.5.3).

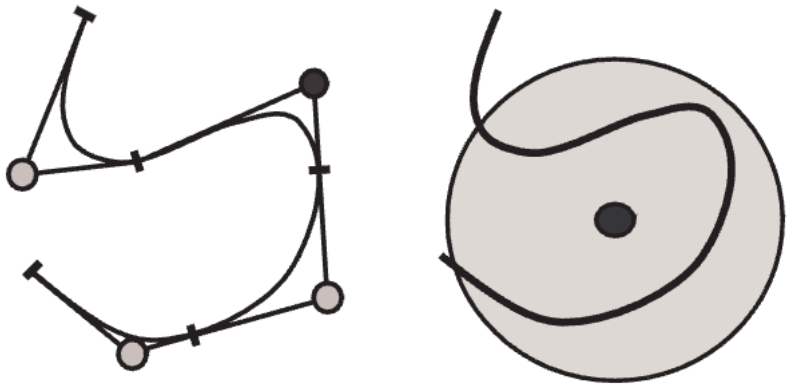


Рисунок 3.23 - Закруглення

Також з безлічі можливих точок розміщення потрібно виділити ті, в яких камери повинні знаходитися обов'язково. Такі точки можуть знаходитися, наприклад, на мостах, тунелях, на особливо аварійних ділянках.

Після проведеного аналізу завдання формалізується так.

Нехай $I=\{1,...m\}$ - множина усіх можливих місць розміщення камер;

$J=\{1,...n\}$ - множина усіх ділянок які спостерігаються;

$$a_{ij} = \begin{cases} 1, \text{якщо камера і контролює ділянку } j \\ 0, \text{в іншому випадку} \end{cases}$$

Змінні завдання:

$$x_i = \begin{cases} 1, \text{якщо на ділянці і встановлюється камера} \\ 0, \text{в іншому випадку} \end{cases}$$

Тоді математична модель розв'язуваної задачі набуває вигляду $\min \sum_{i \in I} x_i$, при обмеженнях $\sum_{i \in l} a_{ij} x_i \geq 1, j \in J; x_i \in \{0,1\}, i \in I$

Це завдання відоме в цілісному програмуванні як завдання про мінімальне покриття (ЗМП).

Для вирішення ЗМП розроблено велику кількість як точних, так і наближених методів рішення. Всі існуючі точні алгоритми пов'язані з перебором на деякій кінцевій множині, що задається вихідними даними. За обсягом цього перебору як

функції кількості вихідних даних точні алгоритми рішення ЗМП відносяться до експоненційних, що унеможливлює їх застосування для задач з розмірністю матриці більше 100. Для практичного вирішення реальних завдань використовуються, як правило, наближені алгоритми, серед яких провідне місце належить, так званим "жадібним" алгоритмам. Ідея алгоритмів цього типу пов'язані з послідовною побудовою допустимих варіантів. На кожному алгоритмічному етапі робиться спроба досягнення деякої локальної допоміжної мети (мінімізація числа невиконаних обмежень, оптимізація вихідної цільової функції та ін) за допомогою параметрів, що вибираються на даному етапі. Локальний підхід такого типу не передбачає спроб оцінки ні попередніх, ні наступних кроків алгоритму.

У даній задачі можна очікувати, що результати попереднього аналізу дозволять так сконструювати вхідні дані, що завдання можна буде вирішувати за допомогою точних алгоритмів за прийнятний час. Використовуємо для вирішення задачі адитивний алгоритм Балаша.

Адитивный алгоритм Балаша

Алгоритм представлений в, є методом неявного перебору, у загальному випадку призначений для вирішення наступного завдання: $Z(x) = \sum_{j=1}^{n} c_j x_j \to \min\limits_{x \in B^n}$, $\sum_{j=1}^{n} a_{ij} x_j \geq b_i, i = 1, \dots, m$

Рішенням назвем множину $\left\{x_i : x_j = 1, j \in N_1; x_j = 0, j \in N | N_1, N = \{1, \dots, n\}\right\}$.

Рішення є допустимим, якщо виконуються нерівності $\sum_{j=1}^{n} a_{ij} x_j \geq b_i$

Припустимо, що виконані нерівності $0 \leq c_1 \leq c_2 \leq \dots \leq c_n$

Задача має 2^n різних рішень. Розіб'ємо усю множину рішень на (n +1) підмножин з номерами k = 0,1,...,n таким чином, що k–е підмножина містить всі рішення з k змінними, рівними одиниці, і n–k змінними, рівними нулю.

Звідси:

- при $k = 0$, підмножина рішень склдається із єдиного рішення x = 0:
- при $k = 1$, підмножина рішень включає C_n^1 рішень, в яких $x_i = 1; x_j = 0$, $j \neq i$, $i = 1,...,n$;
- k -е підмножина складається із C_n^k рішень.

Потім всі рішення упорядковуються за допомогою діаграми, де кожна вершина, яка містить список N_1, представляє рішення, в якому змінні з індексами із N_1 рівні одиниці, а решта нулю.

Якщо в діаграмі існує шлях із вершини u у вершину v, то вершина u називається попередньою для вершины v, а вершина v называется наступною за u. Таким чином, рішення часково упорядковані.

Для перевірки допустимості рішень будемо перевіряти умову $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$.

Алгоритм розпочинає роботу з вершини x=0, потім переглядає наступні за нею вершини. Якщо яка-небудь вершина відповідає допустимому рішенню, то не потрібно переглядати наступні за нею вершини, вони виключаються

На основі розглянутого алгоритму Балаша розроблено 3 варіанти: приблизний алгоритм С1, алгоритм Climb - при проходженні діаграми знизу і алгоритм Descend - при проходженні діаграми зверху.

1) Наближений алгоритм С1[65]

Алгоритм використовує теоретико-множинне формулювання задачі.

Множини S_i будуються для нашої задачі наступним чином: для кожного номера i множина S_i представляє собою набір номерів тих участків, які прослідковуються і-ю камерою.

Нехай Sub містить включені в покриття підмножини, Un*cov* містить ще непокриті елементи.

Алгоритм С1 складається з наступного:

1. Встановлюємо $Sub = \emptyset, Uncov = \bigcup_{S \epsilon F} S; N = |F|; Set(i) = S_i, 1 \leq i \leq N;$
2. Якщо Uncov = $\emptyset$, алгоритм завершує роботу і повертає Sub як рішення;
3. Обираємо j ≤ N таке, що |Set(j)| максимальне;
4. Вважаємо *Sub = Sub*$\cup${*Sj*}*, Uncov = Uncov - Set(j), Set(i) = Set(i) - Set(j), 1 ≤ i ≤ N;*
5. повернемося на крок 2.

Особливості реалізації: з вхідної матриці формуються множини S_i, кожна

така множина зберігається у вигляді списку, є лічильник кількості елементів для кожного S_i.

Якщо лічильник рівний нулю, множина виключається з розгляду. Програма завершує свою роботу, коли всі лічильники стають рівні нулю.

2)Алгоритм Climb[93]

Враховуючи структуру результатів, отриманих після попереднього аналізу, можна припускати, що рішення буде знайдено швидше, якщо діаграму розглядати знизу вгору. При цьому достатньо знаходити перше за порядком рішення на рівні і рухатися вище, оскільки всі інші можливі рішення на даному рівні будуть не кращими за перше знайдене. Алгоритм зупиниться, коли на рівні не буде допустимих рішень, і оптимальним буде рішення, отримане на попередньому рівні. Даний підхід реалізовано в програмі Climb.

Особливості реалізації: на кожному рівні програма послідовно генерує перестановки (вектор довжини depth з номерами камер) і одночасно перевіряє нерівності $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$. Для того, щоб перевірити ці нерівності, не потрібно обчислювати суму; достатньо порівняти рядок матриці і перестановку. Якщо в рядку є хоча б одиниця на місці, визначеному номерами в перестановці, то нерівність для даного рядка виконується. Після знаходження першого допустимого рішення перевірка на поточному рівні завершується, отримане рішення запам'ятовується, і відбувається перехід на рівень вище.

Результат програми C1 зручно використовувати в Down. Розмір отриманого вектора рішення, зменшений на 1, слід вказати як початковий параметр depth. Програма знайде рішення на рівнях вище, і воно буде кращим, або не знайде, тоді отриманий вектор буде оптимальним.

3)Descend[22]

У цій задачі зручно розглядати вершини в іншому порядку, послідовно за рівнем у діаграмі, тобто розглядати рішення з однаковою кількістю індексів у списку N_1. У цьому випадку перше допустиме рішення буде одним із оптимальних. Даний метод реалізовано в програмі Descend.

Використано аналогічний метод, як у Climb, із відмінністю, що рівні діаграми

проходяться зверху. Також в програмі можна використовувати результат C1: оскільки він мало відрізняється від точного рішення, розмір вектора точного рішення повинен бути не менше цілої частини 1+lnN, тому в якості початкового значення для depth слід використовувати саме цей вираз.

Цей алгоритм краще працює на мало розріджених матрицях.

4. Обчислення та результати

Для тестування був проведений приблизний аналіз двох мікрорайонів і створені вхідні матриці (Таблиця 3.2). Середня кількість точок розташування камер складає від 20 до 40. При N>40 ефективно працює тільки приблизний алгоритм, оскільки в точному може знадобитися повний огляд середнього рівня (в якому CN=2>1010 елементів).

Таблиця 3.2. Вхідні матриці двох мікрорайнів районів

Вхідна матриця мікрорайону A	Вхідна матриця мікрорайону B
110000000000010000000000	11000000000000000000000000000000
110000000000010000000000	01100000000000000000000000000000
100000000000001000000000	01000000000000000010000000000000
100000000000000000010000100	00000000000000000011000000000000
010000000000100000000000	00000000000000000010000000000010
001000000000010000000000	10000000000000000000000000000010
001100000000100000000000	10000000000000000000000000000011
000100000000100100000000100	00000000000000001100000000000000
000100000000100100000000100	00000000000000011000000000000000
000100000000101100000000100	00000000000000110000000000000000
000100000000001100000000100	00000000000011100000000000000000
000010000000001000000000000	00000000000111000000000000000000
000010000000000000000100000	00000000000110000000000000000000
000010000000000000001000000	00000000011100000000000000000000
000001000000000000011000000	00000000011000000000000000000000
100000000000000000010000100	00000000110000000000000000000000

000001100000000000110000000	00000001100000000000000000000000
000001000000000010000000000	00000011000000000000000000000000
000001100000000000110000000	00000110000000000000000000000000
000100000000000000100000000	00001100000000000000000000000000
000001101000000000110000000	00011000000000000000000000000000
000000100000100000000000000	00110000000000000000000000000000
001000000000000001000000000	00100000000000000000100000000000
000000101000000001100000000	00000000000000000000110000000000
000000101000000001000000000	00000000000000000000011000000000
000000001000000000000000001	00000000000000000000001100000000
000000100000000000000000010	00000001000000000000000100000000
000010000000000000001001000	00000000010000000000000010000000
000000000100000000000100000	00000000000000000000000011000000
000000000011000000000000000	00000000000000000000000001100000
000000000010000000000000001	00000000000000000000000000110000
000000000011000000000000000	00000000000000000000000000011000
000000000001000000000000010	00000000000000000000000000001100
000000000011000000000000000	01000000000000000001000000011000
000000000100000000000100000	00000000000000000001000000000100
	00000000000001000000000010000000
	00000000000000000100000000110000
	00000000000000001100000000000001

Вхідними даними для всіх програм є булева матриця A(i,j) розміром N⊗M, яку формується наступним чином: A(i,j)=1, якщо камера j відстежує відрізок i, в іншому випадку A(i,j)=0. Розміщення матриці в текстовому файлі відбувається рядково, де рядок може містити або не містити пробіли між елементами. Після введення матриці повинен відбутися перехід на новий рядок.

При побудові матриці стовбці можна розмістити в такому порядку, що перші k стовбців будуть формувати прийнятне рішення, що прискорить роботу програм

(завдяки особливостям генерації перестановок). Для цього достатньо, щоб в кожному рядку матриці було хоча б одна одиниця.

У матриці не повинно бути нульових рядків та стовпців. Нульовий рядок вказує на те, що якийсь відрізок не прослідковується, і задача не має рішення; нульовий стовпець є ознакою того, що деяка камера не прослідковує жодного відрізка і, отже, не потрібна в покритті.

Якщо в матриці є рядок, в якому одиничний елемент входить тільки один раз, це означає, що даний відрізок прослідковується тільки однією камерою, і ця камера обов'язково буде включена в покриття. Отже, можна видалити рядок і стовпець з матриці, на Перехрестяі яких знаходиться одиниця, тим самим зменшивши розмірність (в подальшому цю камеру потрібно буде додати до отриманого покриття). Крім того, всі інші відрізки, які прослідковуються цією камерою, слід видалити з наборів відрізків інших камер (видаляються рядки, які містять одиничні елементи даної камери).

Програми Descend та Climb вважають розмірність вхідної матриці (параметри N та M).

Приклад обчислень наведено в таблиці 3.3.

Таблиця 3.3. Тривалість обчислень програмами різних алгоритмів

Матриця	Алгоритм	Час работы
A	C1	менше 1 секунди
A	Climb	1 секунда
A	Descend	2 секунди
A	Climb при depth=10	1 секунда
A	Descend при depth=2	2 секунди
B	C1	менше 1 секунди
B	Climb	5 хвилин 52 сек

B	Descend	11 хвилин 23 сек
B	Climb при depth=15	5 хвилин 31 сек
B	Descend при depth=3	11 хвилин 23 сек

У цьому розділі описані заходи, проведення яких дозволить ефективно розподілити існуючу транспортну систему міста так, щоб в кожній отриманій частині можна було організувати систему відеоспостереження з мінімальною кількістю камер і переглядом всіх необхідних ділянок, використовуючи точні алгоритми для задачі покриття.

Отримано необхідний комплекс заходів і програм, які реалізують приблизні та адаптовані точні алгоритми для даної задачі. Цю роботу можна використовувати для організації будь-яких систем відеоспостереження.

3.10 Практична реалізація результатів дослідження та розробка програми

Представлений алгоритм на вхідних даних буде перевіряти можливість пропуску даного потоку між пунктами (від вершини S до вершини T) протягом часу, який не перевищує заданий. На виході алгоритм або повідомлятиме, що потік пройшов, або, якщо це неможливо, буде визначати шляхи та конкретні перехрестя, на яких виникають затори, заважаючи успішному проходженню заданого потоку протягом вказаного часу.

Алгоритм FreeRoad

1-крок. Використовуємо алгоритм Mstream, в якому:

а) в якості початкової і кінцевої точок беремо вершини S і T,

б) в якості c_{ij} беремо Pmax(i, j),

в) в якості cost(i,j) беремо h(Pmax(i,j), Preal(i,j), zerotime(i,j)).

г) в якості M беремо Time.

2-крок. Перетворимо отриманий у попередньому кроці список в:

Memory.вартість = round(Time - Memory.вартість) + 1. Потім отримуємо F як суму всіх Memory.вартість.

3-крок. Отриманий на другому кроці максимальний потік F порівнюємо з M:

а) Якщо M <= F, то алгоритм завершує роботу з виведенням "Успіх".

б) Якщо M > F, то переходимо до 4-го кроку.

4-крок. Беремо перший шлях зі списку Memory.шлях і розпочинаємо йти по ньому.

а) беремо першу дугу і переходимо до пункту б).

б) Порівнюємо Preal(i,j) поточної дуги з сумою Pmax(j,k) всіх вихідних з вершини дуг, в яку входить поточна. Якщо ця сума більше, ніж Preal(i,j) a<b+c+d, то переходимо до наступної дуги, якщо така існує, і повторюємо пункт б). Якщо такої дуги не існує, то переходимо до пункту г). Якщо сума менше, ніж Preal(i,j), то переходимо до пункту в).

в) Якщо сума виявилася менше, то в список FreeRoad (у рядок з номером, аналогічним номеру запису поточного шляху в списку Memory.шлях) вносимо номер вершини, в яку входила поточна дуга.

г) Якщо наступна дуга в шляху не існує, то переходимо до кроку 5.

5-крок. Беремо наступний шлях і переходимо до кроку 4а. Якщо такого шляху немає, то переходимо до 6-го кроку.

6-крок.Якщо FreeRoad порожній, то алгоритм завершує свою роботу з виходом "Некоректні вхідні дані". Якщо FreeRoad містить записи, то алгоритм також завершує роботу і на виході надає два списки: Memory.шлях – зберігає шляхи, і FreeRoad – зберігає затори.

Отримані алгоритми були реалізовані на мові JavaScript. Програма аналізує транспортну мережу на проходження потоку через неї (рисунок 3.24) та виявляє затори, пропонуючи коротший варіант маршруту (рисунок 3.25). У випадку, якщо потік не може пройти, для отримання результату, який відображає реальний стан доріг при вирішенні пов'язаних задач, цей модуль повинен використовуватися в ітеративному зв'язку з іншими (усунення заторів, побудова розв'язок – в цьому

випадку потрібно, щоб модуль, який будує розв'язки, перед подачею модулю, запропонованому в даній роботі, перетворював розв'язок в формат, який розуміє модель – звичайні дуги, вершини та ваги на них).

Пункт А	Пункт Б	Час
вул. Михайлівська	вул. Хрещатик	6
вул. Прорізна	вул. Хрещатик	15
вул. Хрещатик	бул. Тараса Шевченка	13
бул. Тараса Шевченка	вул. Володимирська	7
бул. Тараса Шевченка	вул. Евгена Чекаленко	10
вул. Евгена Чекаленко	вул. Богдана Хмельницького	23
вул. Евгена Чекаленко	вул. Прорізна	6
вул. Володимирська	вул. Богдана Хмельницького	5
вул. Володимирська	вул. Прорізна	28
вул. Володимирська	Володимирський проїзд	30
Володимирський проїзд	вул. Михайлівська	23
бул. Тараса Шевченка	вул. Леонтовича	17
вул. Леонтовича	вул. Богдана Хмельницького	5
вул. Богдана Хмельницького	вул. Лисенка	7
вул. Лисенка	вул. Ярославіва вал	20
вул. Ярославіва вал	вул. Прорізна	5
вул. Ярославіва вал	вул. Володимирська	3
вул. Михайлівська	вул. Велика Житомирська	10
вул. Велика Житомирська	вул. Ярославіва вал	15

Додати Видалити Редагувати Закрити

Рисунок 3.24 – Вікно «Час шляху»

Вноситься час, що витрачається на кожному перехресті. Згодом дані використовуються для побудови суміжної матриці.

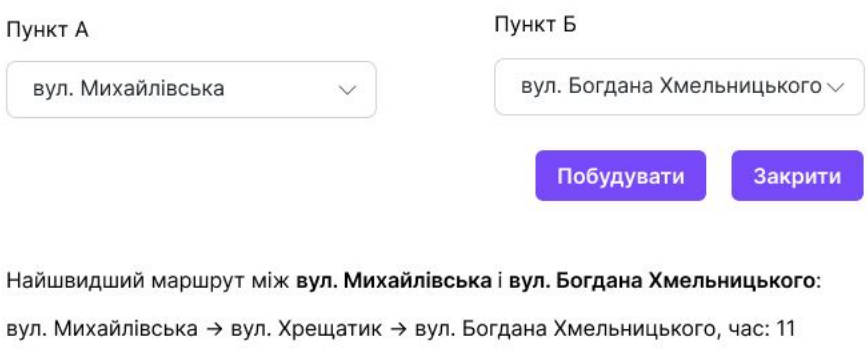


Рисунок 3.25 - Побудова короткого маршруту

Доказ коректності алгоритму

Алгоритм FreeRoad є коректним, його часова складність - $O((n^2 + m^2)mC)$, де n - кількість вершин графа (V, E), m - кількість дуг, $C = \max\{Pmax_{ij}\}$.

В алгоритмі спочатку шукається потік, який в принципі можна направити з пункту S в пункт T за час, що не перевищує Time. Це обумовлено тим, що F є максимальним потоком за одиницю часу, то, щоб зрозуміти, скільки машин встигнуть пройти з S в T протягом часу Time, треба взяти Time мінус час, витрачений на елементарний шлях, плюс ще одиницю (яка встигла дістатися до T в останню мить), і після знаходження його починаємо перевірку. Якщо вхідний потік не перевищив максимальний, то вважаємо, що він встигає успішно пройти з S в T за час Time. Якщо перевищує, то тоді ми припускаємо, що на шляху нашого потоку зустрічаються затори і, починаючи з четвертого кроку, ми їх починаємо шукати. Для цього беремо послідовно шляхи зі списку Memory.шлях, який був сформований ще при виконанні алгоритму Mstream, і для кожного з цих шляхів проводимо наступні обчислення: Йдемо по ньому і перевіряємо, щоб реальний (усереднений) потік, що входить в вершину, весь зміг пройти через неї, тобто пропускні здатності всіх вихідних дуг йому це дозволяли. Якщо, пройшовши по всіх шляхах, ми не зможемо знайти заторів – FreeRoad буде порожній, то тоді, швидше за все, вхідні дані (потік M або час Time) були введені некоректно. Якщо ж такі знайдені, то тоді ми запам'ятовуємо за допомогою TrafficWay, на якому шляху і на якому вузлі цей затор виник. Після чого на вихід передаємо ці два списки.

Обчислимо тимчасову складність даного алгоритму за кроками.

Виконання алгоритму Mstream дає тимчасову складність $O((n^2+m)mC)$.

Зміна Memory.вартості та її подальше сумування займає не більше, ніж O(mC) операцій. Потім, коли починаємо перевіряти всі шляхи (не більше, ніж mC), і на кожному шляху порівнюємо кожну дугу (не більше, ніж m) з сумою вихідних (не більше m), отримуємо $O(m^3C)$. В кінці отримуємо $O(n^2mC+m^2C+mC+m^3C) = O((n^2m+m^3+m+m^2)C) = O((n^2+m^2)mC)$.

3.11 Висновки до розділу 3

Підвищено ефективність функціонування інформаційних систем керування транспортними потоками.

Розглянуті особливості алгоритму Форда-Фалкерсона та процедура розстановки позначок для задачі максимального потоку.

Досліджено вплив одностороннього руху на величину транспортного потоку, надано опис рівнів дорожньо-транспортної мережі.

Розроблена концепція імітаційної моделі транспортних потоків та управління транспортними системами на основі теорії S-гіпермереж.

Проведен обчислювальний експеримент на основі моделювання для перевірки ефективності результатів дослідження.

Розглянуті приклади розрахунку пропускної здатності ділянки дорожньо-транспортної мережі.

Був розроблений алгоритм усунення заторів шляхом зміни режимів роботи світлофорів, що дозволило усунути додаткові затори на бічних дорогах, які призводили до цього маршруту.

Сформульовано принцип оптимальності. Математична задача маршрутизації зводиться до знаходження найкоротшого шляху в неорієнтованому графі. Були розглянуті різні алгоритми розв'язання задач маршрутизації з використанням теорії S-гіпермережі.

Був досліджен метод побудови комп’ютерних мереж для розгортання систем відоспостереження за транспортними потоками міста.

В результаті проведених досліджень були розроблені:

– Був розроблений алгоритм для визначення найкоротшого шляху в орієнтованому мультиграфі.

– Був розроблений алгоритм Mstream для знаходження максимального потоку, вартість якого не перевищує задану.

– Був розроблений алгоритм FreeRoad для аналізу транспортної мережі та виявлення на ній вузьких місць.

Доведена правильність алгоритму FreeRoad. Отримані алгоритми були реалізовані на мові програмування JavaScript. Оскільки алгоритм FreeRoad аналізує лише транспортну мережу на проходження потоку і виявлення заторів, то у випадку, якщо потік не може пройти, для отримання результату, що відображає реальний стан доріг, при вирішенні пов'язаних задач, цей модуль повинен використовуватися в ітеративній взаємодії з іншими.

ВИСНОВКИ

У дисертаційній роботі вирішена актуальна наукове завдання підвищення ефективності управління транспортними потоками на основі дослідження та розробки інформаційних технологій для комплексного математичного моделювання транспортних систем міста із застосуванням теорії S-гіпермереж.

Основні наукові та практичні результати роботи полягають в наступному:

1 Розроблено модель інформаційної системи, наукова новизна якої полягає в тому, що вона грунтується на керуванні транспортними потоками на базі теорії S-гіпермереж, що дозволила підвищити ефективність функціонування транспортної мережі міста.

2 Удосконалено методику побудови комп'ютерної мережі, яка на відміну від існуючих дозволяє мінімізувати кількість точок розміщення відеокамер на заданій території за рахунок застосування наближених алгоритмів.

3 Розроблено методику управління функціонування інформаційних систем керування транспортними потоками наукова новизна якої полягає в тому, що вона грунтується на нечіткій логіці та дозволяє покращити управління інформаційною системою транспортної мережі, за рахунок зменшення часу проходження потоку на перехресті.

4. Результати дослідження можуть бути використані в побудові дорожньо-транспорнї мережі. науковими та проектними організаціями. У ході подальших досліджень планується розробка методів, що дозволяють підвищити точність зняття характеристик обладнання за рахунок використання попередньо оптимізованих експериментальниї шаблонів для конкретних виробників.

5. Достовірність результатів забезпечується коректністю використання математичних положень термінів та прийомів щодо доведень і обґрунтувань тверджень та підтверджується результатами наведених експериментальних досліджень. Таким чином, наукове завдання в дисертації виконано, мета досягнута.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. С.С. Коротков, В.О. Сосновий, О.М. Ткаченко, А.В. Лемешко І.А. Бученко “Проблема маршрутизації для мереж міського транспорту”, Зв’язок, №4, с. 32–36, 2021.

2. С.С. Коротков, А.О. Барабаш “Проблема семантичних протиріч у великих обсягах даних”, Зв'язок, №3, с. 31–33, 2022.

3. С.С. Коротков “Архітектурно-математичне моделювання систем мережної структури з використання теорії S-Гіпермереж”, Телекомунікаційні та інформаційні технології, № 4, с. 64–72, 2023.

4. С.С. Коротков “Імітаційне моделювання керування транспортним потоком із застосуванням S-Гіпермережі”, Звя’зок, №1, с. 31–36, 2024.

5. О.М. Ткаченко, Я.І. Торошанко, А.В. Лемешко, В.О. Сосновий, С.С. Коротков. Комп’ютерні мережі: контроль та прогнозування перевантажень. Навчальний посібник. – Київ: ДУТ, 2021. – 77 с.

6. orofin Denys, Tverdokhleb Arsenii, Lemeshko Andrii Antonenko Artem, Korotkov Serhii, Hunko Oleksandra Stepanchuk Vadym, Sosnovyy Vladyslav, Kovalenko Anna Brovenko Tetiana, Tolok Galina, Tonkykh Oleksii Artem Gorkun, Yushchenko Arsen, Balvak Andrii. "EQUIPMENT OF HOTEL AND RESTAURANT COMPLEXES WITH NETWORK EQUIPMENT". Монографія. Participants of the International Scientific symposium

7. С.С. Коротков Проблема семантичних протиріч у великих обсягах даних. - К.: ДУТ, “Зв’язок” - 2023 - №3

8. С.С. Коротков, В.С. Засадюк Алгоритм зменшення транзитних потоків транспортної мережі у заданому напрямку. - К.: Міжнародний науково-технічний університет імені академіка Юрія Бугая, “IT Synergy” - 2022 - №2

9. Schadschneider A., Schreckenberg M. Cellular automata for traffic.

10. Turky A.M. The Use of Genetic Algorithm for Traffic Light and Pedestrian Crossing Control // IJCSNS International Journal of Computer Science and Network Security. -2009.-Vol.9, №2. -P.88-96.

11. Madhavan Nair B., Cai J. A fuzzy Logic Controller for Isolated Signalized Intersection with Traffic Abnormality Considered //Proc. 2007 IEEE Intelligent Vehicles Symposium Istanbul. -2007. -P.1229-1233.

12. Hartman D. Head leading algorithm for urban traffic modeling //Proc. 16th European Simulation Symposium György Lipovszki, IstvánMolnár © SCS Press. – 2004. -P.10-17.

13. Mancinelli E. On Traffic Light Control of Regular Towns //Rapport de recherché. Institut national de rechercheeninformatique et enautomatique. -2001. -P.12-18.

14. Gonzalez H. Adaptive Fastest Path Computation on a Road Network: A Traffic Mining Approach //VLDB 07.-2007. -P.794-805.

15. Jee-Hyong LEE. Traffic Control of Intersection Group Based on Fuzzy Logic //Proc. 6th International Fuzzy Systems Association World Congress. -1995. -P. 465-468.

16. Benjaafar S. Cellular Automata for Traffic Flow Modeling // Final report, Center for Transportation Studies University of Minnesota, 1997. – P.28-36.

17. Shivaram Subramanian. Routing algorithms for dynamic, intelligent transportation networks // Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1997. – P.175-183.

18. .Elsken T., Metzen J.H., Hutter F. Multi-objective Architecture Search for CNNs. 2018.

19. Domhan T., Springenberg J.T., Hutter F. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. International Joint Conferences on Artificial Intelligence. 2015.

20. Bergstra J. Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems. 2011.

21. Bengio Y. Gradient-based optimization of hyperparameters. Neural Computation. 2000.

22. Snoek J., Adams R. Practical bayesian optimization of machine learning algorithms. Advances in Neural Information Processing Systems. 2012.

23. Li L., Talwalkar A. Random Search and Reproducibility for Neural Architecture Search. 2019.

24. Bergstra J., Bengio Y. Random search for hyper-parameter optimization. Journal of Machine Learning Research. 2012. No.13. P. 281–305.

25. Cai H., Chen T., Zhang W. Efficient Architecture Search by Network Transformation. AAAI. 2017. No.18.

26. Elsken T., Metzen J., Hutter F. Neural Architecture Search: A Survey. Journal of Machine Learning Research. 2019. No.20. P. 1–21.

27. Лящинський П. Б., Лящинський П.Б. Автоматизований синтез структур згорткових нейронних мереж. Problèmes et perspectives d'introduction de la recherche scientifique innovante. 2019. Вип. 2. С. 104–105.

28. Автомобільні мережі - Головна. URL: https://disted.edu.vn.ua/courses/learn/397 (10.04.2022).

29. Software-defined networking: the new road for Networks [Електронний ресурс] / Електронний репозитарій ДВНЗ "УжНУ". // – Режим доступу: https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/31527/1 (10.04.2014 р.)

30. Wikiwand - Матриця інцидентності. Wikiwand. URL: https://www.wikiwand.com/uk/Матриця_інцидентності#google_vignette(15.01.2022).

31. Кривець Т.О., Прудкой Ю.І., Городенська В.Я. Математична модель постійних інформаційно-обчислювальних мереж: Наукові праці УДУХТ, Київ, 2001.

32. Кривец Т.А., DYNAMICAL SYSTEM MODELLING AND STABILITY INVESTIGATION, Kyiv, 2003.

33. Jafari S., Jafari S.M, Ebrahimi M., Kijpatanasilp I., Assatarakul K., A decade overview and prospect of spray drying encapsulation of bioac-tives from fruit products:

Characterization, food application and in vitro gastrointesti-nal digestion, Food Hydrocolloids, V. 134, 2023, 108068, https://doi.org/10.1016/j.foodhyd.2022.108068.

34. Sloan E. Top 10 functional food trends. Food Technology. 2018. V.72, Is.4

35. Nair A., Chattopadhyay D., Saha B. New Look to Phytomedicine: Plant-derived immunomodulators. USA: Academic Press: Cambridge. 2019. P. 435-499.

36. Korhonen H. Technology options for new nutritional concepts. Int J of Dairy Technology. 2002. V. 55, Is.2. P. 79-88. https://doi.org/10.1046/j.1471-0307.2002.00050.x

37. Coates PM., Blackman MR et al. Encyclopedia of dietary supplements. Solomon P. Wasser Shiitake (Lentinula edodes). 2004. P. 653-664.

38. Daniele da Silva Bastos, Maria do Pilar Gonçalves, Cristina Tristão de Andrade, Kátia Gomes de Lima Araújo, Maria Helena Miguez da Rocha Leão, Microencapsulation of cashew apple (Anacardium occidentale, L.) juice using a new chitosan–commercial bovine whey protein isolate system in spray drying. Food and Bioproducts Processing. 2012. V. 90, Is.4. P. 683-692. https://doi.org/10.1016/j.fbp.2012.04.005

39. T. A. G. Langrish, Advances in spray drying. Proceedings of the 2nd Asian-Oceania Drying Conference ADC'01 Batu Feringhi, Pulau Pinang, Malaysia, 20-22 August 2001. pp. 393-418.

40. Kudre T., Arun S., Mujumdar A. Advanced Drying Technologies. 2001. 472 p.

41. Masters, K. Spray Drying in Practice. Spray Dry Consult Int. 3rd ed. 2002. 464p.

42. Bandyopadhyay S. (2023) Decision Support System Tools and Techniques. CRC Press, 394 p.

43. Bek-Pedersen E., Lind M., Asheim B. (2019). AI Based Real-Time Decision Making. Abu Dhabi International Petroleum Exhibition & Conference: 22nd International Conference (Abu Dhabi, UAE, November 11th–14th, 2019).

44. Bench-Capon T. J. M. (2014) Knowledge Representation: An Approach to Artificial Intelligence. Academic Press, 236 p.

45. Blokdyk G. (2020) Semantic Network A Complete Guide. 5STARCooks.

46. Brachman R. J., Levesque H. J. (2004) Knowledge Representation and Reasoning. Morgan Kaufmann Publishers, 416 p.

47. Gulmamedov R. G. (2011) Metod postroeniya strategii v sistemah situatsionnogo upravleniya [The method of constructing a strategy in situational management systems]. Information and control systems, vol. 6, pp. 36–39.

48. Kovalenko I. I., Shved A. V., Antipova K. O. (2018) Modeli podannia ta vyvedennia znan u systemakh sytuatsiinoho upravlinnia [Models of presenting and extracting of knowledge in situational control systems]. Mykolaiv: Ilion, 91 p.

49. Larichev O. I., Petrovskiy A. B. (1987), Sistemy podderzhki vybora resheniy: sovremennoye sostoyaniye i perspektivy razvitiya [Decision support systems: current state and development prospects]. Results of science and technology, vol. 21.

50. Levesque H. J., Lakemeyer G. (2001) The Logic of Knowledge Bases. Mit Press, 282 p

51. Madarász L., Andoga R., Fozo L., Lazar T. (2009). Situational Control, Modeling and Diagnostics of Large Scale Systems. In: Rudas I. J., Fodor J., Kacprzyk J. (Eds.) Towards Intelligent Engineering and Information Technology. Studies in Computational Intelligence, vol. 243, pp 153–164.

52. Morozov A. A. (2016) Situatsionnyie tsentryi. Ponyatiya i opredeleniya [Situation centers. Concepts and definitions]. Mathematical Machines and Systems, vol. 1, pp. 48–54.

53. Power D. (2002) Decision Support Systems: Concepts and Resources for Managers. Praeger, 272 p.

54. Aasim, M., Baloch, F. S., Nadeem, M. A., Bakhsh, A., Sameeullah, M., & Day, S. (2018). Fenugreek (Trigonella foenum-graecum L.): An Underutilized Edible Plant of Modern World. Global Perspectives on Underutilized Crops, 381–408. https://doi.org/10.1007/978-3-319-77776-4_12

55. Ahmad, A., Alghamdi, S. S., Mahmood K., & Afzal, M. (2016). Fenugreek a multipurpose crop: Potentialities and improvements. Saudi Journal of Biological Sciences. 23(2), 300–310. https://doi.org/10.1016/j.sjbs.2015.09.015

56. Beyzi, E., & Gürbüz, B. (2020). Influence of sowing date and humic acid on fenugreek (Trigonella foenum-graecum L.). Journal of Applied Research on Medicinal and Aromatic Plants.//doi.org/10.1016/j.jarmap.2019.100234

57. Bhutia, P. H., & Sharang, A. B. (2018.) Influence of dates of sowing and irrigation scheduling on phenology, growth and yield dynamics of fenugreek (Trigonella foenum greacum L.). Legume Research-An International Journal. 41(2), 275–280. http://dx.doi.org/10.18805/LR-3432

58. Bienkowski, T., Zuk-Golaszewska, K., Kurowski, T., & Golaszewski, J. (2016). Agrotechnical indicators for Trigonella foenum-gracum L. production in the environmental conditions of northeastern Europe. Turkish Journal of Field Crops. 21(1), 16–28. https://doi.org/10.17557/tjfc.37573

59. Новини - Вживані авто - Авто 24. Новини: останні автоновини сьогодні на порталі auto.24tv.ua. URL: https://auto.24tv.ua/tag/vzhyvani_avto_tag130 (10.02.2023).

60. Scheduling of traffic lights-a new approach [Електронний ресурс] / Pergamon Press – 1968 // – Режим доступу: https://www.sciencedirect.com/sdfe/pdf/download/eid/1-s2.0-0041164768900166/first-page-pdf. (22.03.2014 p.).

61. Horn B. K. P., Wang L. Wave Equation of Suppressed Traffic Flow Instabilities. IEEE Transactions on Intelligent Transportation Systems. 2018. Vol. 19, no. 9. P. 2955–2964. URL: https://doi.org/10.1109/tits.2017.2767595 (22.05.2022).

62. Pratt W.K. Digital Image Processing/W.K.Pratt//J.Wile&Sons. – 1991,v.1,2

63. Jan J. Diskretni Metody Zpracovani biosignalu. SNTL Praga-VUT Brno, 1976.

64. Banks S. Signal Processing, Image Processing and Pattern Recognition/ Prentice Hill Int. (UK) Ltd., 1990.

65. Beauchamp K. Yuen C. Digital Methods for Signal Analysis. G. Allen & Unwin Ltd. London, 1979.

66. Bellanger M. Digital Processing of Signals, Theory and Practice. J. Wiley & Sons NY, 1990.

67. Gold B., Rader C.M Prentice Hill Int. (UK) Ltd., 1990Processing of Signals. McGraw-Hill, 1969.

68. Jain A.K. Fundamentals of Digital Image Processing. Prentice Hill Int. (UK) Ltd., 1989.

69. Kuc R. Introduction to Digital Signal Processing. McGraw-Hill Int. Ed. 1988.

70. Oppenheim A.V., Schafer C.M. Digital Signal Processing. Prentice Hill, Ebglewood Cliffs, N. Jersey, 1975.

71. Solomon C.J., Breckon T.P. Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab. Wiley-Blackwell, 2010. doi:10.1002/9780470689776. ISBN 0470844736.

72. Fisher R., Dawson-Howe K, Fitzgibbon A., Robertson C., Trucco E. Dictionary of Computer Vision and Image Processing. John Wiley, 2005). ISBN 978-0-470-01526-1.

73. Gonzalez R. C., Woods R. E., Eddins S. L. Digital Image Processing using MATLAB. Pearson Education. 2004. ISBN 978-81-7758-898-9.

74. Morris T. Computer Vision and Image Processing. Palgrave Macmillan, 2004. ISBN 978-0-333-99451-1.

75. Jähne B. Digital Image Processing. Springer, 2002. ISBN 978-3-540-67754-3.

76. Sonka M., Hlavac V., Boyle Roger Image Processing, Analysis, and Machine Vision. PWS Publishing, 1999. ISBN 978-0-534-95393-5.

77. Rosenfeld A., Picture Processing by Computer New York: Academic Press, 1969

78. Yang Y. Y. et al. Implementation of network traffic monitor system with SDN / Y. Y. Yang, C. T. Yang, S. T. Chen, W. H. Cheng, F. C. Jiang, //2015 IEEE 39th annual computer software and applications conference. - IEEE, 2015. - T. 3. - C. 631¬634.

79. Wong, M. D. Testing Compilers for Programmable Switches Through Switch Hardware Simulation / M. D. Wong, A. Varma, A. Sivaraman //arXiv preprint arXiv:2005.02310. - 2020.

80. Wang, S. Y. EstiNet openflow network simulator and emulator / S. Y. Wang, C. L. Chou, C. M. Yang //IEEE Communications Magazine. - 2013. - Т. 51. - №. 9. - С. 110-117.

81. Wang, G. Programming Your Network at Run-time for Big Data Applications / G. Wang, T.S. Eugene Ng, A. Shaikh // Hot Topics in Software Defined Networking (HotSDN). — Helsinki, Finland, 2012.

82. Wang S. Y. Comparison of SDN OpenFlow network simulator and emulators: EstiNet vs. Mininet //2014 IEEE Symposium on Computers and Communications (ISCC). - IEEE, 2014. - С. 1-6.

83. NetSim Network Simulator [Электронный ресурс] / Boson Holdings. - 2020.- Режим доступа: https://www.boson.com/netsim-cisco-network-simulator.

84. NetSim User Manual [Электронный ресурс]. -2019. -Режим доступа: https://www.tetcos.com/downloads/v11/NetSim_User_Manual.pdf

85. Salih, M. A. OpenFlow 1.3 extension for OMNeT++ / M. A. Salih, J. Cosmas, Y. Zhang //2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and

86. Secure Computing; Pervasive Intelligence and Computing. - IEEE, 2015. - С. 1632-1637.

87. Sato T. et al. Abstract model of sdn architectures enabling comprehensive performance comparisons //2015 11th International Conference on Network and Service Management (CNSM). - IEEE, 2015. - С. 99-107.

88. Shi X. et al. PABO: Mitigating congestion via packet bounce in data center networks //Computer Communications. - 2019. - Т. 140. - С. 1-14.

89. Tang Y. et al. Automatic belief network modeling via policy inference for SDN fault localization //Journal of Internet Services and Applications. - 2016. - Т. 7. - №. 1. - С. 1.

90. Tarasov V. et al. Analysis of intervals between traffic packets on the Software Defined Networks depending on the TCP window size //2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T). - IEEE, 2016. - С. 15-17.

91. Teixeira, R. PacketScope: Monitoring the Packet Lifecycle Inside a Switch / R. Teixeira, R. Harrison, A. Gupta, J. Rexford //Proceedings of the Symposium on SDN Research. - 2020. - С. 76-82.

92. Ushakov, Y. A. Service-oriented routing in distributed self-organizing software-defined networks / Y. A. Ushakov, M. V. Ushakova, A. E. Shukhman, P. N. Polezhaev, L. V. Legashev //2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT). - IEEE, 2018. - С. 1-5.

93. Kumar, D. Analysis of Impact of Network Topologies on Network Performance in SDN / D. Kumar, M. Sood //International Conference on Innovative Computing and Communications. - Springer, Singapore, 2020. - С. 357-369.

94. Kriege, J.,\ Simulating stochastic processes with OMNeT++ / J. Kriege, P. Buchholz //SimuTools. - 2011. - С. 367-374.

95. Kabir M. H. et al. Detail comparison of network simulators //International Journal of Scientific & Engineering Research. - 2014. - Т. 5. - №. 10. - С. 203-218.

96. Ivey J. et al. Comparing a scalable SDN simulation framework built on ns-3 and DCE with existing SDN simulators and emulators //Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation. - 2016. - С. 153-164.

97. Bolodurina, I. Method of Obtaining Generalized Statical Characteristics of Network Equipment for Creation of Simulation Models / Bolodurina I., Ushakova M., 146 Ushakov Y. // 2020 International Russian Automation Conference (RusAutoCon). - IEEE, 2020. - С. 947-952.

98. Banjar, A. Analysing the performance of the OpenFlow standard for software-defined networking using the OMNeT++ network simulator/ A. Banjar, P. Pupatwibul, R. Braun, B. Moulton //2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE). - IEEE, 2014. - С. 31-37.

99. Steinbach T. et al. An extension of the OMNeT++ INET framework for simulating real-time ethernet with high accuracy / T. Steinbach, H.D. Kenfack, F. Korf, T.C. Schmidt //Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques. - 2011. - C. 375-382.

100. System and Technologies of Digital Television: manual for graduate students. [Text]/ V.A. Loshakov, V.V . Popovsky , S.O. Saburova, I.S. Shostko, M.Y. Oshepkov, K.O. Popovskaya, L.I. Melnikova. Under the general editorship of Professor V.A. Loshakov – Kh: Company SMIT", 2019. – 416 p.

ДОДАТОК 1

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. С.С. Коротков *“Архітектурно-математичне моделювання систем мережної структури з використання теорії S-Гіпермереж”*, Телекомунікаційні та інформаційні технології, № 4, с. 64–72, 2023.
2. С.С. Коротков *“Імітаційне моделювання керування транспортним потоком із застосуванням S-Гіпермережі”,* Звя’зок, №1, с. 31–36, 2024.
3. С.С. Коротков, В.О. Сосновий, О. М. Ткаченко, А. В. Лемешко І. А. Бученко *“Проблема маршрутизації для мереж міського транспорту”*, Зв’язок, № 4, с. 32–36, 2021.
4. С.С. Коротков, А. О. Барабаш *“Проблема семантичних протиріч у великих обсягах даних”*, Зв'язок, № 3, с. 31–33, 2022.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. О.М. Ткаченко, Я.І. Торошанко, А.В. Лемешко, В.О. Сосновий, С.С. Коротков. Комп’ютерні мережі: контроль та прогнозування перевантажень. Навчальний посібник. – Київ: ДУТ, 2021. – 77 с.
2. Korofin Denys, Tverdokhleb Arsenii, Lemeshko Andrii Antonenko Artem, Korotkov Serhii, Hunko Oleksandra Stepanchuk Vadym, Sosnovyy Vladyslav, Kovalenko Anna Brovenko Tetiana, Tolok Galina, Tonkykh Oleksii Artem Gorkun, Yushchenko Arsen, Balvak Andrii. "EQUIPMENT OF HOTEL AND RESTAURANT COMPLEXES WITH NETWORK EQUIPMENT". Монографія. Participants of the International Scientific symposium
3. С.С. Коротков, В.С. Засадюк Алгоритм зменшення транзитних потоків транспортної мережі у заданому напрямку. - К.: Міжнародний науково-технічний університет імені академіка Юрія Бугая, “IT Synergy” - 2022 - №2 с. 55-61.

ДОДАТОК 2

„ЗАТВЕРДЖУЮ"
Перший проректор ДУІКТ,
доктор технічних наук, професор
_____ Олександр КОРЧЕНКО
_____ 2024р.

АКТ

використання у навчальному процесі Навчально-наукового інституту інформаційних технологій результатів дисертаційної роботи викладача кафедри комп'ютерної інженерії Державного університету інформаційно-комунікаційних технологій Короткова Сергія Станіславовича на тему: «Методика побудови інформаційної системи управління транспортною інфраструктурою міста на базі теорії S-гіпермереж» на здобуття наукового ступеня доктора філософії за спеціальністю 123 – Комп'ютерна інженерія

Комісія у складі:

голова – директор Наукового центру, кандидат технічних наук, професор Дробик О.В.; члени комісії – завідувачка кафедри штучного інтелекту, доктор технічних наук, професор Зінченко О.В.; завідувачка кафедри комп'ютерної інженерії, кандидат технічних наук, доцент Лащевська Н.О.

розглянули дисертаційну роботу Короткова Сергія Станіславовича на тему: «Методика побудови інформаційної системи управління транспортною інфраструктурою міста на базі теорії S-гіпермереж» та публікації автора за матеріалами дисертаційної роботи. Результати впроваджено в начальний процес Навчально-наукового інституту інформаційних технологій, а саме:

1. Модель інформаційної системи, наукова новизна якої полягає в тому, що вона грунтується на керуванні транспортними потоками на базі теорії S-гіпермереж, що дозволила підвищити ефективність функціонування транспортної мережі міста.