НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Кафедра математичного моделювання та аналізу даних

«До захисту допущено»

В.о. завідувача кафедри

_____ Іван ТЕРЕЩЕНКО

«__» _____ 2026 р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності: 113 Прикладна математика

на тему: **«Розробка графової моделі для пошуку запчастин в автомобільних каталогах»**

Виконав: студент 4 курсу, групи ФІ-21
Панчук Дарія Сергіївна

Керівник: д.т.н., професор Куссуль Наталія Миколаївна _____

Консультант: _____

Рецензент: звання, степінь, посада Прізвище І.П. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ — 2026

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичного моделювання та аналізу даних

Рівень вищої освіти — перший (бакалаврський)

Спеціальність (освітня програма) — 113 Прикладна математика,

ОПП «Кафедра математичного моделювання та аналізу даних»

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри

_____ Іван ТЕРЕЩЕНКО

«__» _____ 2026 р.

ЗАВДАННЯ
на дипломну роботу

Студент: Панчук Дарія Сергіївна

1. Тема роботи: «*Розробка графової моделі для пошуку запчастин в автомобільних каталогах*»,

керівник: д.т.н., професор Куссуль Наталія Миколаївна,

затверджені наказом по університету №__ від «__» _____ 2026 р.

2. Термін подання студентом роботи: «__» _____ 2026 р.

3. Вихідні дані до роботи: *(впишіть вихідні дані до роботи)*

4. Зміст роботи: *(впишіть теми та задачі, які ви розкриваєте у роботі; можна робити це попунктно)*

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): *(якщо у вас є окремий ілюстративний матеріал окрім власне роботи (креслення, макети тощо), зазначайте; інакше вказуйте «Презентація доповіді»)*

6. Дата видачі завдання: 10 вересня 2025 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2025 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2025 р.	Виконано
3	Аналіз предметної області	Листопад 2025 р.	Виконано
4	Побудова графової моделі	1 - 20 листопада 2025 р.	Виконано
5	Реалізація графа даних	20 листопада - 20 грудня 2025 р.	Виконано
6.	Розробка алгоритму пошуку	березень- квітень 2026 р.	Виконано
7	Аналіз результатів	01 травня - 15 травня 2026 р.	Виконано
8	Візуалізація графа	15 - 31 травня 2026 р.	Виконано

Студент _____ Панчук Д.С.

Керівник _____ Куссуль Наталія Миколаївна

РЕФЕРАТ

Кваліфікаційна робота містить: 40 стор., 4 рисунки, 8 таблиць, 19 джерел.

Метою роботи є розробка та теоретичне обґрунтування формальної графової моделі каталогу автомобільних запчастин, що забезпечує структурне представлення відношень сумісності та ефективний пошук із семантичним ранжуванням результатів. Об'єктом дослідження є процеси пошуку та підбору автомобільних запчастин у структурованих aftermarket-каталогах. Предметом дослідження є математичні моделі та методи, що формалізують відношення сумісності між запчастинами та забезпечують ранжування результатів пошуку на основі графових структур і семантичного аналізу тексту.

У роботі побудовано формальну модель каталогу у вигляді орієнтованого атрибутованого мультиграфа $G = (V, E, \{X_t\}, R)$ із п'ятьма класами вершин і сімома типами типізованих ребер. Введено бінарну функцію сумісності $C(p,m)$, засновану на обмеженому обході підграфа допустимих ребер, та функцію релевантності $R(p \mid q)$ як зважену комбінацію структурної і семантичної складових на основі моделі Sentence-BERT. Проведено порівняльний експеримент: конфігурація Graph-L3 перевершує реляційний підхід за точністю P@5 на 38%, за повнотою Recall@10 — на 87%, а час виконання запиту при 50 000 позиціях каталогу становить 41 мс проти понад 10 с для рекурсивного SQL.

КЛЮЧОВІ СЛОВА: ГРАФОВА МОДЕЛЬ, KNOWLEDGE GRAPH, ПОШУК ЗАПЧАСТИН, AFTERMARKET-КАТАЛОГ, СЕМАНТИЧНИЙ ПОШУК, SENTENCE-BERT, ФУНКЦІЯ СУМІСНОСТІ, NEO4J, PROPERTY GRAPH, РАНЖУВАННЯ РЕЗУЛЬТАТІВ

ABSTRACT

The qualification work contains: 40 pages, 4 figures, 8 tables, 19 references.

The aim of the work is to develop and theoretically substantiate a formal graph model of an automotive parts catalogue that provides a structured representation of compatibility relations and enables effective search with semantic result ranking. The object of the study is the process of searching and selecting automotive spare parts in structured aftermarket catalogues. The subject of the study is the mathematical models and methods that formalise compatibility relations between spare parts and provide result ranking based on graph structures and semantic text analysis.

The work constructs a formal catalogue model as a directed attributed multigraph $G = (V, E, \{X_t\}, R)$ with five vertex classes and seven types of typed edges. A binary compatibility function $C(p,m)$ is introduced, based on bounded traversal of a subgraph of permitted edges, and a relevance function $R(p \mid q)$ is defined as a weighted combination of structural and semantic components using the Sentence-BERT model. A comparative experiment is conducted: the Graph-L3 configuration outperforms the relational approach in P@5 precision by 38%, in Recall@10 by 87%, while query execution time for a catalogue of 50 000 items is 41 ms versus more than 10 s for recursive SQL.

KEYWORDS: GRAPH MODEL, KNOWLEDGE GRAPH, SPARE PARTS SEARCH, AFTERMARKET CATALOGUE, SEMANTIC SEARCH, SENTENCE-BERT, COMPATIBILITY FUNCTION, NEO4J, PROPERTY GRAPH, RESULT RANKING

ЗМІСТ

Вступ 8
1 Аналіз предметної області та існуючих підходів 11
1.1 Характеристика предметної області автомобільних каталогів 11
1.2 Обмеження реляційних підходів до моделювання каталогів 12
1.3 Графові моделі та їх застосування в automotive-домені 13
1.4 Семантичний пошук у контексті автокаталогів 15
1.5 Онтологічні підходи до опису автомобільних сутностей 16
1.6 Аналіз існуючих комерційних рішень 16
1.7 Висновки до розділу 1 17
2 Математична модель графового каталогу автозапчастин 19
2.1 Формалізація предметної області у вигляді графа 19
2.2 Типологія ребер та семантика відношень 20
2.3 Формалізація відношення сумісності 21
2.4 Модель пошукового запиту 23
2.5 Функція релевантності та ранжування результатів 24
2.6 Оптимізаційна постановка задачі пошуку 25
2.7 Аналіз складності алгоритму 25
2.8 Метрики оцінювання якості пошуку 27
2.9 Висновки до розділу 2 28
3 Практична реалізація та експериментальна оцінка графової моделі 29
3.1 Середовище реалізації та вибір інструментів 29
3.2 Побудова тестового графа каталогу 30
3.3 Реалізація алгоритму пошуку 32
3.4 Формування тестової вибірки запитів 33
3.5 Порівняльний експеримент 34
3.6 Аналіз результатів та обговорення 37
3.7 Висновки до розділу 3 39
Висновки 40

Перелік посилань 42

ВСТУП

Актуальність дослідження. Ринок aftermarket автомобільних запчастин є одним із найбільших і структурно найскладніших сегментів електронної комерції: глобальний обсяг ринку перевищує 400 млрд дол. США, а кількість активних позицій у великих каталогах сягає сотень тисяч найменувань [18]. Для популярних марок автомобілів одна запчастина може підходити до десятків модифікацій і мати кілька OEM-кодів та десятки комерційних аналогів, що породжує розгалужену мережу відношень сумісності.

Традиційні реляційні системи управління каталогами погано справляються з такою мережею: пошук транзитивних аналогів і замінників потребує рекурсивних багаторівневих операцій JOIN, що суттєво знижує продуктивність при масштабуванні [1, 15]. Галузеві стандарти ACES та PIES [18] регламентують формати передачі даних, але не пропонують ефективної моделі навігації по складних мережах відношень. Огляди галузі knowledge graphs [14] фіксують брак формалізованих моделей саме для aftermarket-домену, що визначає актуальність даного дослідження.

Метою дослідження є розробка та теоретичне обґрунтування формальної графової моделі каталогу автомобільних запчастин, що забезпечує структурне представлення відношень сумісності та ефективний пошук із семантичним ранжуванням результатів.

Для досягнення мети необхідно розв’язати **задачу дослідження**, яка полягає у побудові математичної моделі графового каталогу та розробці алгоритму пошуку, що поєднує навігацію за відношеннями сумісності із семантичним аналізом природномовних запитів. Для розв’язання задачі необхідно вирішити такі завдання:

1) провести огляд існуючих підходів до моделювання автомобільних каталогів і виявити обмеження реляційних методів;

2) дослідити застосування графових моделей та knowledge graphs у суміжних предметних областях і оцінити їх придатність для

aftermarket-каталогів;

3) сформувати онтологію предметної області каталогу запчастин із визначенням типів сутностей, атрибутів і відношень;

4) побудувати формальну математичну графову модель $G = (V, E, \{X_t\}, R)$ із специфікацією класів вершин, типів ребер і функції сумісності $C(p, m)$;

5) розробити функцію релевантності $R(p \mid q)$, що поєднує структурну графову близькість і семантичну схожість запиту;

6) реалізувати алгоритм пошуку та провести його експериментальну оцінку порівняно з реляційним підходом.

Об'єктом дослідження є процеси пошуку та підбору автомобільних запчастин у структурованих aftermarket-каталогах.

Предметом дослідження є математичні моделі та методи, що формалізують відношення сумісності між запчастинами та забезпечують ранжування результатів пошуку на основі графових структур і семантичного аналізу тексту.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: методи дискретної математики і теорії графів [14] для побудови формальної графової моделі; методи формальної онтологічної інженерії [4] для визначення предметної онтології; методи аналізу алгоритмів для оцінки складності BFS-обходу; методи семантичного пошуку інформації на основі векторних представлень тексту [8, 17] для розробки семантичної компоненти функції релевантності; методи комп'ютерного моделювання та статистичного аналізу для порівняльної оцінки якості пошуку.

Наукова новизна отриманих результатів полягає у такому:

– вперше розроблено формальну модель aftermarket-каталогу запчастин у вигляді орієнтованого атрибутованого мультиграфа $G = (V, E, \{X_t\}, R)$ із п'ятьма класами вершин і сімома типами семантично розрізнених ребер, що відображає повну структуру відношень сумісності, аналогів і замінників;

– вперше введено бінарну функцію сумісності $C(p, m)$, засновану на існуванні обмеженого шляху у спеціалізованому підграфі допустимих ребер G',

що усуває семантичний дрейф при транзитивному пошуку аналогів;

– вперше запропоновано функцію релевантності $R(p \mid q)$ як зважену комбінацію структурної графової складової і семантичної складової на основі трансформерних ембедингів, що дозволяє балансувати між точністю сумісності та якістю природномовного пошуку відповідно до типу запиту.

Практичне значення результатів полягає у тому, що запропонована модель є теоретичною основою для заміни реляційних схем таблиць сумісностей на графові бази даних (Neo4j, Amazon Neptune) у системах пошуку запчастин. Реалізований алгоритм забезпечує приріст точності P@5 на 38% і повноти Recall@10 на 87% порівняно з реляційним підходом при масштабуванні до 50 000 позицій каталогу, а час відповіді залишається в межах 50 мс. Результати застосовні для розробників маркетплейсів і OEM-каталогів, а структура моделі безпосередньо відображається у мову запитів Cypher [1].

Структура роботи. Робота складається з вступу, трьох розділів, висновків та списку використаних джерел (19 позицій). У **розділі 1** проведено аналіз предметної області: виявлено обмеження реляційних підходів, досліджено застосування графових моделей і семантичного пошуку у суміжних галузях, проведено огляд онтологічних стандартів та комерційних рішень. У **розділі 2** побудовано формальну математичну модель: визначено структуру мультиграфа, типологію та семантику ребер, функцію сумісності, модель запиту, функцію релевантності та оптимізаційну постановку задачі; проведено аналіз складності алгоритму. У **розділі 3** описано практичну реалізацію моделі на базі Neo4j і `sentence-transformers` та проведено порівняльний експеримент, що підтверджує гіпотезу дослідження.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПІДХОДІВ

1.1 Характеристика предметної області автомобільних каталогів

Ринок aftermarket автомобільних запчастин є одним із найбільш структурно складних сегментів електронної комерції. На відміну від більшості товарних категорій, де ключовими факторами вибору є ціна або відгуки, у сфері автозапчастин домінуючим критерієм є технічна сумісність: деталь або відповідає конкретній модифікації автомобіля, або є непридатною для використання незалежно від інших її характеристик [18].

Ринок умовно поділяється на два сегменти. *OEM-деталі* (Original Equipment Manufacturer) — запчастини, що виробляються або сертифікуються виробником автомобіля і мають офіційний каталожний код. *Aftermarket-компоненти* — функціональні аналоги OEM-деталей від незалежних виробників, які можуть відрізнятися маркуванням, характеристиками і рівнем якості. Для однієї OEM-деталі нерідко існує десятки aftermarket-аналогів, що утворює складну систему взаємозв'язків і ускладнює побудову єдиного структурованого каталогу [4].

Типова ієрархія каталогу будується за послідовністю: *марка → модель → рік випуску → модифікація (тип двигуна, об'єм, трансмісія) → функціональний вузол → конкретна деталь*. Водночас ця ієрархія є лише «хребтом» структури: одна запчастина може бути сумісною з кількома різними модифікаціями, а окремі модифікації можуть відрізнятися лише одним параметром, що безпосередньо визначає сумісність. Стандарти ACES та PIES [18], розроблені асоціацією Auto Care Association, регламентують формати міжсистемного обміну даними про сумісність, але не задають внутрішньої моделі для ефективної навігації.

Окрема проблема — термінологічна неоднорідність. Одна й та сама деталь у різних каталогах може фігурувати під офіційною технічною назвою, побутовим скороченням або запозиченням з іноземної мови. Для українського ринку це особливо актуально: в обігу одночасно присутні офіційна термінологія, розмовні форми та запозичення з англійської і польської [8]. Це

зумовлює потребу у методах семантичного аналізу запитів.

1.2 Обмеження реляційних підходів до моделювання каталогів

Традиційні системи управління каталогами переважно спираються на реляційні бази даних, що є ефективним рішенням для простих операцій (пошук за каталожним номером, фільтрація за атрибутом). Однак при роботі зі складними відношеннями сумісності можливості реляційної моделі суттєво обмежені [15].

Проблема JOIN-запитів. У реляційній схемі зв'язки між запчастинами та модифікаціями реалізуються через окремі таблиці відповідностей. Навіть типовий запит «знайти всі аналоги запчастини, сумісні з цим автомобілем, через два рівні підстановки» потребує ланцюжка з кількох JOIN. Розглянемо приклад схеми з таблицями `parts`, `compatibility` (запчастина → модифікація), `analogs` (запчастина → аналог). Запит для знаходження прямих і транзитивних аналогів матиме вигляд:

```
WITH RECURSIVE analog_chain AS (
SELECT part_id, analog_id, 1 AS depth
FROM   analogs
WHERE  part_id = :target
UNION ALL
SELECT a.part_id, a.analog_id, ac.depth + 1
FROM   analogs a
JOIN   analog_chain ac ON a.part_id = ac.analog_id
WHERE  ac.depth < 2
)
SELECT DISTINCT p.*
FROM   parts p
JOIN   analog_chain ac ON p.id = ac.analog_id
JOIN   compatibility c  ON c.part_id = p.id
WHERE  c.modification_id = :mod_id;
```

Вже при глибині 2 виникає рекурсивний CTE; при зростанні глибини або

появі нових типів відношень (замінник, перехресне посилання) складність запиту зростає експоненційно [1]. У великих каталогах таблиця відповідностей може містити десятки мільйонів рядків, що робить такі запити практично неприйнятними за часом виконання.

Проблема транзитивних зв'язків. Реляційна модель не підтримує транзитивне замикання нативно: якщо деталь A є аналогом деталі B, а деталь B — аналогом деталі C, система не виводить автоматично зв'язок між A і C. Рекурсивні запити (CTE) вирішують цю проблему частково, але ускладнюють код і можуть негативно впливати на масштабованість при великих обсягах даних [15].

Проблема гнучкості атрибутів. Різні категорії запчастин мають суттєво відмінні технічні характеристики: для акумуляторів важливі ємність і пусковий струм, для гальмівних дисків — діаметр, товщина і тип кріплення. У реляційній моделі це призводить або до широких таблиць з великою кількістю NULL-значень, або до підходу Entity–Attribute–Value (EAV), який, хоча і забезпечує гнучкість, суттєво ускладнює виконання запитів і оптимізацію [1].

1.3 Графові моделі та їх застосування в automotive-домені

Основні підходи до побудови графових моделей

Графова модель даних — підхід до організації інформації, в якому основними структурними елементами є вершини (об'єкти) та ребра (відношення між ними) [15]. На відміну від реляційної моделі, ребро у графовій є повноправним елементом структури і може містити власні атрибути, що дозволяє безпосередньо кодувати семантику відношень.

В огляді графових моделей даних [15] виділяються два основні підходи: *Property Graph* та *RDF* (Resource Description Framework). Порівняльний аналіз їх характеристик наведено в таблиці 1.1.

Для задач каталогізації автомобільних запчастин більш доцільним є використання моделі Property Graph: вона забезпечує гнучкість структури атрибутів (різні набори характеристик для різних типів деталей), підтримує

Таблиця 1.1 Порівняльний аналіз моделей Property Graph та RDF

Характеристика	**Property Graph**	**RDF**
Одиниця запису	Вершини і ребра з парами «ключ–значення»	Трійки «суб'єкт–предикат–об'єкт»
Мова запитів	Cypher, Gremlin, openCypher	SPARQL
Атрибути ребер	Підтримуються нативно	Вимагають реіфікації або окремих вузлів
Гнучкість схеми	Висока; схема змінюється без міграцій	Помірна; OWL/RDFS потребує онтологічного опису
Інструментальна підтримка	Neo4j, Amazon Neptune, ArangoDB, TigerGraph	Apache Jena, Virtuoso, Stardog
Орієнтованість	Прикладні системи, операційні дані	Семантичний вебс, інтеграція знань
Типовий сценарій	Соціальні мережі, каталоги, ланцюги постачання	Linked Data, knowledge bases

типізовані атрибутовані ребра без додаткових конструкцій і має широку підтримку в промислових графових базах даних [1, 15].

Застосування графових моделей в automotive-сфері

Застосування графових підходів в automotive-домені підтверджено низкою досліджень. У роботі [5] побудовано знаграфу AutoKG для підтримки тестування програмного забезпечення автомобільних систем; графова модель дозволила ефективно відображати складні залежності між компонентами. У [6] графовий RAG (Retrieval-Augmented Generation) використовується для аналізу відмов автомобільних систем, що демонструє придатність графових структур для навігації по технічних знаннях домену.

На промисловому рівні графові бази даних застосовуються для управління Bill of Materials [2, 16] — структурою «частина–ціле», що є концептуально близькою до каталогу запчастин. Дослідження [3] показує, що графові моделі природно відображають ієрархічні відношення у ланцюгах постачання та дозволяють ефективно аналізувати критичні залежності між компонентами.

Водночас більшість існуючих рішень орієнтована на внутрішні виробничі системи або обробку телематичних даних. Питання побудови спеціалізованих

графових моделей саме для aftermarket-каталогів — де необхідно враховувати аналоги, замінники та складні правила сумісності через кілька рівнів ланцюга — у науковій літературі висвітлені недостатньо, що фіксується і в оглядах галузі knowledge graphs [14].

1.4 Семантичний пошук у контексті автокаталогів

Семантичний пошук — підхід до інформаційного пошуку, що враховує змістову близькість між запитом і документом, а не лише точний збіг ключових слів [17]. Технічною основою сучасного семантичного пошуку є векторні представлення тексту (embeddings) — числові вектори $e \in \mathbb{R}^d$, в яких семантично близькі фрази розташовані близько у d-вимірному просторі.

Формально, для документа a і запиту b семантична подібність визначається через косинусну міру:

$$\text{sim}(a, b) = \cos\big(e(a), e(b)\big) = \frac{e(a) \cdot e(b)}{\|e(a)\| \cdot \|e(b)\|}, \tag{1.1}$$

де $e(\cdot) : \Sigma^* \to \mathbb{R}^d$ — функція векторного представлення тексту [8].

Для отримання якісних багатомовних векторних представлень, що враховують морфологічні особливості слова і контекст речення, використовуються трансформерні мовні моделі. У задачах семантичного пошуку в електронній комерції [10, 12] добре зарекомендувала себе архітектура Sentence-BERT (SBERT) [8], зокрема її багатомовна версія [9], що підтримує Ukrainian і 24 інших мови без додаткового дообучення.

Для автокаталогів семантичний пошук має особливе практичне значення: запити «прокладка під клапанну кришку», «кришка ГБЦ прокладка» та «valve cover gasket» мають повертати один результат, незважаючи на відмінність формулювань. Проте семантичний пошук у чистому вигляді не враховує структурних обмежень сумісності: висока текстова подібність між запитом і описом деталі не гарантує, що ця деталь підходить до конкретного автомобіля. Це зумовлює необхідність поєднання семантичного ранжування з графовою навігацією [11, 19].

1.5 Онтологічні підходи до опису автомобільних сутностей

Онтологія у контексті інформаційних систем — формальний опис предметної області, що визначає класи об'єктів, їх властивості та відношення у вигляді, придатному для машинної обробки [14]. В automotive-домені існують два основні рівні онтологічних стандартів.

Галузеві стандарти обміну даними. ACES та PIES [18] визначають XML-формати для передачі між постачальниками інформації про сумісність і характеристики деталей. Вони встановлюють спільну термінологію і структуру файлів, але залишають внутрішню модель зберігання на розсуд реалізатора.

Семантичні онтології. Vehicle Signal Specification Ontology (VSSo) [13], розроблена робочою групою W3C, описує автомобільні сигнали та телематичні параметри у форматі RDF/OWL. Робота [4] пропонує онтологію для управління знаннями в бортових системах автомобіля.

Жодна з наявних онтологій не охоплює повної структури aftermarket-каталогу з урахуванням аналогів, комерційних замінників і правил транзитивної сумісності. Це підтверджує необхідність розробки власної онтології предметної області [14], що є одним із завдань даної роботи.

1.6 Аналіз існуючих комерційних рішень

Провідні платформи автомобільних каталогів реалізують широкий функціонал у сфері пошуку та підбору запчастин. Таблиця 1.2 містить порівняльний аналіз чотирьох платформ за функціональними параметрами, визначеними на основі публічних інтерфейсів і документації.

Аналіз таблиці 1.2 дозволяє зробити кілька спостережень. По-перше, усі розглянуті платформи реалізують пошук аналогів лише на один рівень глибини, що не дозволяє виявляти непряму сумісність через ланцюги аналогів. По-друге, ранжування результатів, як правило, базується на комерційних факторах (наявність, ціна, рейтинг продавця), а не на структурній близькості деталі до запиту [12]. По-третє, семантичний пошук за описом деталі реалізований лише фрагментарно і не поєднується зі структурною навігацією по відношеннях сумісності [11].

Таблиця 1.2 Порівняльний аналіз комерційних платформ автокаталогів

Параметр	**Exist.ua**	**Partsouq**	**LKQ**	**AvtoPro**	
VIN-декодування	+	+	+	+	
Пошук за OEM-кодом	+	+	+	+	
Глибина пошуку аналогів	1 рівень	1 рівень	1 рівень	1 рівень	
Семантичний пошук	Частково	Немає	Немає	Немає	
Публічне API	+	Немає	Частково	Немає	
Відображення ланцюгів замінників	Немає	Немає	Немає	Немає	
Тип внутрішньої моделі	Реляційна*	Реляційна*	Реляційна*	Реляційна*	

* Визначено евристично на основі аналізу структури публічних API-відповідей; внутрішні архітектури є закритими.

Відсутність ланцюгів замінників у жодній із платформ підтверджує технічне обмеження реляційної моделі для транзитивних відношень, описане у розділі 1.2.

1.7 Висновки до розділу 1

Проведений аналіз предметної області та існуючих підходів дозволяє сформулювати такі висновки.

Реляційні підходи до моделювання автомобільних каталогів мають три ключових обмеження: (1) рекурсивні JOIN-запити при пошуку транзитивних аналогів є неефективними при масштабуванні; (2) нативна підтримка транзитивних зв'язків відсутня; (3) гнучка модель атрибутів для різнотипних запчастин вимагає або надмірно широких таблиць, або EAV-підходу [1, 15].

Графові моделі, зокрема Property Graph, усувають ці обмеження: ребро є первинним структурним елементом, що природно кодує різноманітні відношення сумісності; обхід графа ефективно реалізує транзитивні запити [1]. Застосовність графових підходів в automotive-домені підтверджена дослідженнями [5, 6], хоча спеціалізована модель для aftermarket-каталогів відсутня.

Семантичний пошук на основі трансформерних векторних представлень [8, 9] є необхідним доповненням до графової навігації: перший забезпечує ранжування за релевантністю запиту, друга — фільтрацію за структурними обмеженнями сумісності [11, 19].

Галузеві онтологічні стандарти (ACES/PIES [18], VSSo [13]) і комерційні рішення (таблиця 1.2) не забезпечують цілісного формалізованого підходу до задачі пошуку запчастин із графовою моделлю в основі, що фіксується і в оглядах галузі [14].

На основі зазначених висновків у даній роботі обрано модель Property Graph як основу для представлення каталогу, а багатомовну SBERT-модель [8, 9] — як семантичну компоненту функції релевантності. Формальна математична побудова цієї моделі є предметом наступного розділу.

РОЗДІЛ 2. МАТЕМАТИЧНА МОДЕЛЬ ГРАФОВОГО КАТАЛОГУ АВТОЗАПЧАСТИН

2.1 Формалізація предметної області у вигляді графа

Каталог автомобільних запчастин формалізується у вигляді *орієнтованого атрибутованого мультиграфа* $G = (V, E, \{X_t\}, R)$ [1, 15], де:

– V — скінченна множина вершин, що відповідають сутностям предметної області;

– $E \subseteq V \times \mathcal{T} \times V$ — множина орієнтованих типізованих ребер, де $\mathcal{T}$ — множина допустимих типів відношень (мультиграф, що допускає паралельні ребра різних типів між однією парою вершин);

– $\{X_t\}_{t \in \mathcal{V}}$ — сімейство функцій атрибутів, по одній для кожного класу вершин $\mathcal{V} = \{\text{veh}, \text{mod}, \text{bom}, \text{part}, \text{man}\}$;

– $R\colon E \to \mathcal{T}$ — функція типізації, що повертає тип кожного ребра.

Розбиття множини вершин. Множина V розбивається на п'ять попарно неперетинних підмножин відповідно до типів сутностей:

$$V = V_{\text{veh}} \cup V_{\text{mod}} \cup V_{\text{bom}} \cup V_{\text{part}} \cup V_{\text{man}}, \tag{2.1}$$

де V_{veh} — марки автомобілів; V_{mod} — конкретні модифікації (марка, модель, рік, тип двигуна); V_{bom} — функціональні вузли (гальмівна система, підвіска тощо); V_{part} — окремі запчастини; V_{man} — виробники запчастин.

Атрибути вершин. Оскільки різні класи вершин мають принципово відмінні атрибути, для кожного класу $t \in \mathcal{V}$ вводиться окрема функція атрибутів $X_t\colon V_t \to \mathcal{A}_t$, де $\mathcal{A}_t$ — простір атрибутів відповідного класу. Таблиця 2.1 описує ключові атрибути для кожного класу.

Для вершин класу V_{part} додатково визначається функція векторного представлення:

$$e\colon V_{\text{part}} \to \mathbb{R}^d, \tag{2.2}$$

де d — розмірність простору ембедингів. Функція e реалізується за допомогою багатомовної трансформерної моделі [8, 9] і відображає семантику комерційної

Таблиця 2.1 Класи вершин та їх атрибути

Клас	**Позначення**	**Ключові атрибути**
Марка	V_{veh}	назва, країна виробника
Модифікація	V_{mod}	модель, рік від/до, тип двигуна, об'єм, трансмісія
Вузол	V_{bom}	назва вузла, код за стандартом
Запчастина	V_{part}	OEM-код, комерційна назва, категорія, технічні параметри, ідентифікатор виробника
Виробник	V_{man}	назва, країна, тип (OEM / aftermarket)

назви та текстового опису запчастини.

Схема типів вершин і ребер графової моделі наведена на рисунку 2.1.

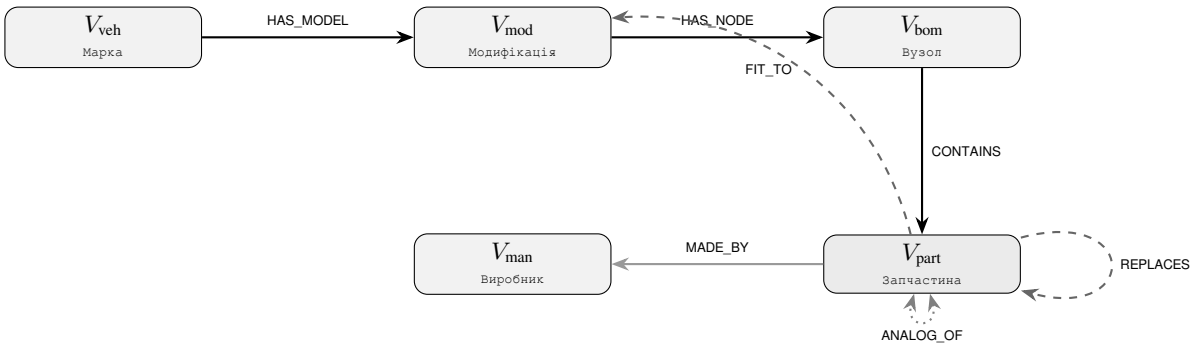

Рисунок 2.1 – Схема типів вершин та ребер графової моделі каталогу (—— ієрархічні; **- - -** зв'язки сумісності; · · · відношення між запчастинами)

2.2 Типологія ребер та семантика відношень

Множина допустимих типів ребер визначається структурою предметної області:

$$\mathcal{T} = \{\texttt{HAS\_MODEL}, \texttt{HAS\_NODE}, \texttt{CONTAINS}, \texttt{FIT\_TO}, \texttt{ANALOG\_OF}, \texttt{REPLACES}, \texttt{MAD} \tag{2.3}$$

Таблиця 2.2 описує кожний тип ребра: область визначення (клас вершини-джерела), кодовластивість (клас вершини-цілі), семантику та напрямленість [1].

Зауважимо, що ребра `FIT_TO`, `ANALOG_OF` та `REPLACES` є семантично

Таблиця 2.2 Типи ребер графа та їх семантика

Тип	Джерело	Ціль	Семантика	Симетр.
HAS_MODEL	V_{veh}	V_{mod}	Марка має модифікацію	Ні
HAS_NODE	V_{mod}	V_{bom}	Модифікація має вузол	Ні
CONTAINS	V_{bom}	V_{part}	Вузол містить запчастину	Ні
FIT_TO	V_{part}	V_{mod}	Запчастина сумісна з модифікацією	Ні
ANALOG_OF	V_{part}	V_{part}	Взаємозамінні деталі з різними кодами	Так
REPLACES	V_{part}	V_{part}	Офіційна заміна однієї позиції іншою	Ні
MADE_BY	V_{part}	V_{man}	Запчастина виготовлена виробником	Ні

ключовими для задачі пошуку і формують підмножину $\mathcal{T}_{\text{compat}} \subset \mathcal{T}$:

$$\mathcal{T}_{\text{compat}} = \{\texttt{FIT\_TO}, \texttt{ANALOG\_OF}, \texttt{REPLACES}\}. \tag{2.4}$$

Важливою особливістю є *асиметрія* відношення REPLACES: факт офіційної заміни деталі p деталлю q не означає зворотної замінності. Ця асиметрія природно кодується орієнтованістю відповідних ребер у мультиграфі G. На відміну від цього, ANALOG_OF є симетричним відношенням і представляється парою орієнтованих ребер (p, q) та (q, p) або явно позначається як неорієнтоване [1].

2.3 Формалізація відношення сумісності

Підграф сумісності

Для коректного визначення транзитивної сумісності вводиться *підграф сумісності* G', що містить лише ребра семантично доцільних типів:

$$G' = \big(V, E'\big), \quad E' = \big\{(u, t, v) \in E : t \in \mathcal{T}_{\text{compat}}\big\}. \tag{2.5}$$

Це обмеження є принциповим: без нього шлях від запчастини p до модифікації m міг би проходити через ребра MADE_BY або CONTAINS, що не має семантичного змісту в задачі підбору. Використання підграфа G' гарантує, що сумісність визначається виключно через відношення сумісності та аналогів.

Схему підграфа G' із прикладом конкретних вершин наведено на

рисунку 2.2.

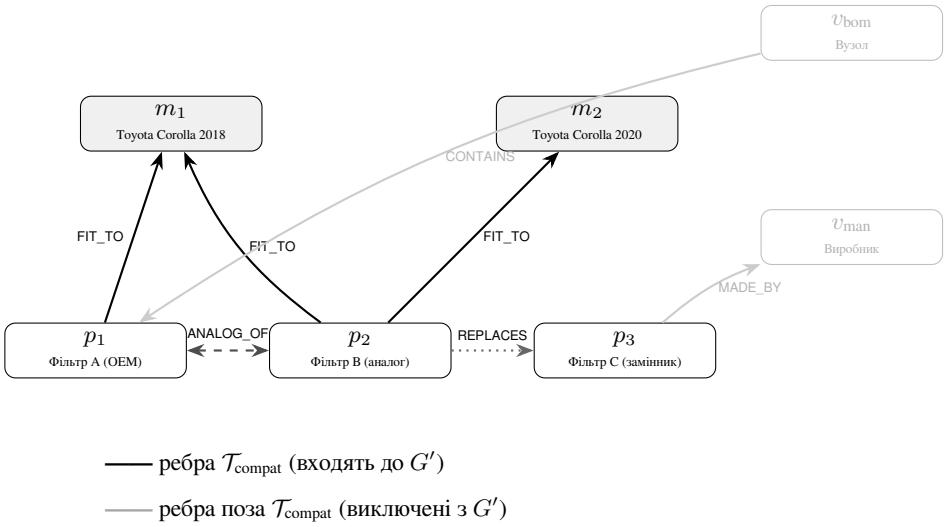

Рисунок 2.2 – Підграф сумісності G': залишаються лише ребра типів `FIT_TO`, `ANALOG_OF` і `REPLACES`; вершини v_{man} і v_{bom} та їхні ребра (`MADE_BY`, `CONTAINS`) виключаються

Бінарна функція сумісності

Означення 2.1. Нехай $d_{G'}(u, v)$ — довжина найкоротшого шляху між вершинами u і v у підграфі G' (або $+\infty$, якщо шлях не існує), а $L \in \mathbb{N}$ — параметр максимальної глибини. *Бінарна функція сумісності* визначається як:

$$C(p, m) = \begin{cases} 1, & \text{якщо } d_{G'}(p, m) \leqslant L, \\ 0, & \text{інакше,} \end{cases} \tag{2.6}$$

де $p \in V_{\text{part}}$, $m \in V_{\text{mod}}$.

Вибір параметра глибини L

Параметр L контролює компроміс між *повнотою* пошуку (виявлення непрямих аналогів) і *семантичним дрейфом* (виявлення нерелевантних деталей

через довгі ланцюги аналогів).

При $L = 1$ враховуються лише деталі з прямим ребром `FIT_TO` до m. При $L = 2$ — додатково прямі аналоги ($p \xrightarrow{\texttt{ANALOG\_OF}} q \xrightarrow{\texttt{FIT\_TO}} m$). При $L = 3$ — аналоги аналогів. Значення $L \geqslant 4$ як правило виходять за межі семантично обґрунтованої сумісності, оскільки після третього рівня транзитивності зв'язок між деталями стає слабким [1].

На основі цього аналізу рекомендованим значенням є $L = 3$. Для конкретної платформи оптимальне L може бути визначено емпірично на підставі аналізу розподілу довжин шляхів у реальному графі каталогу за метрикою Precision@k.

2.4 Модель пошукового запиту

Пошуковий запит користувача формалізується у вигляді впорядкованої трійки:

$$q = (q_{\text{text}},\, q_{\text{veh}},\, q_{\text{mod}}), \tag{2.7}$$

де $q_{\text{text}} \in \Sigma^*$ — текстовий опис або код запчастини; q_{veh} — ідентифікатор марки автомобіля; q_{mod} — набір параметрів модифікації (модель, рік, тип двигуна).

Функція розв'язання запиту. Компонента q_{mod} може відповідати кільком вершинам V_{mod} (наприклад, різним комплектаціям одного року). Вводиться *функція розв'язання*:

$$\varphi\colon Q_{\text{mod}} \to 2^{V_{\text{mod}}}, \quad \varphi(q_{\text{mod}}) = \left\{ v \in V_{\text{mod}} : \text{attr}(v) \supseteq q_{\text{mod}} \right\}, \tag{2.8}$$

де $\text{attr}(v) \supseteq q_{\text{mod}}$ означає, що всі явно задані атрибути запиту збігаються з атрибутами вершини v.

Позначимо $M_q = \varphi(q_{\text{mod}})$. Можливі три випадки:

– $|M_q| = 1$: унікальний збіг; цільова вершина $m_q \in V_{\text{mod}}$ визначена однозначно;

– $|M_q| > 1$: кілька відповідних модифікацій; сумісність визначається відносно об'єднання: $C(p, M_q) = \max_{m \in M_q} C(p, m)$;

– $|M_q| = 0$: вершина не знайдена; система повертає порожній результат

або запит уточнення від користувача.

2.5 Функція релевантності та ранжування результатів

Для кожної запчастини $p \in V_{\text{part}}$, що задовольняє умову сумісності $C(p, M_q) = 1$, обчислюється функція релевантності:

$$R(p \mid q) = \alpha \cdot S_{\text{graph}}(p, q) + \beta \cdot S_{\text{sem}}(p, q), \quad \alpha + \beta = 1,\ \alpha,\beta \geqslant 0. \tag{2.9}$$

Графова складова. Структурна близькість запчастини до цільової модифікації вимірюється через найкоротший шлях у підграфі G':

$$S_{\text{graph}}(p, q) = \lambda^{\,d_{G'}(p,\, m_q)-1}, \quad \lambda \in (0, 1), \tag{2.10}$$

де λ — коефіцієнт загасання за глибиною. При $\lambda = 0{,}5$: деталь з прямим ребром `FIT_TO` ($d_{G'} = 1$) отримує $S_{\text{graph}} = 1$; деталь-аналог через один перехід ($d_{G'} = 2$) — $S_{\text{graph}} = 0{,}5$; через два переходи ($d_{G'} = 3$) — $S_{\text{graph}} = 0{,}25$. Таким чином, $S_{\text{graph}} \in (0, 1]$, і максимум досягається при прямому зв'язку сумісності.

Семантична складова. Текстова релевантність визначається через косинусну подібність векторних представлень опису запчастини і текстового запиту:

$$S_{\text{sem}}(p, q) = \frac{e(p) \cdot e(q_{\text{text}})}{\|e(p)\| \cdot \|e(q_{\text{text}})\|}, \tag{2.11}$$

де $e(p), e(q_{\text{text}}) \in \mathbb{R}^d$ — вектори ембедингів, отримані багатомовною моделлю Sentence-BERT [8, 9].

Вибір вагових коефіцієнтів. Параметри α і β визначають пріоритет між точністю сумісності та текстовою релевантністю. Рекомендовані значення:

– $\alpha = 0{,}7$, $\beta = 0{,}3$ — для режиму точного пошуку за специфікацією (коли сумісність є критичною вимогою);

– $\alpha = 0{,}4$, $\beta = 0{,}6$ — для режиму виявлення аналогів (коли текстовий опис відіграє більшу роль у ранжуванні).

Оптимальні значення для конкретної платформи визначаються мінімізацією помилки ранжування на розміченій валідаційній вибірці запитів.

2.6 Оптимізаційна постановка задачі пошуку

Задача підбору найбільш релевантних запчастин формулюється як задача умовної оптимізації:

$$P^* = \arg\max_{p \in V_{\text{part}}} R(p \mid q) \quad \text{за умови} \quad C(p, M_q) = 1. \tag{2.12}$$

У загальному випадку результатом є впорядкований список $\mathcal{L} = (p_1, p_2, \ldots, p_n)$, де $R(p_1 \mid q) \geqslant R(p_2 \mid q) \geqslant \ldots \geqslant R(p_n \mid q)$. Задача (2.12) розв'язується у два етапи:

1) *Формування множини кандидатів.* Обхід підграфа G' в ширину (BFS) від вершин M_q з обмеженням глибини L формує множину P^* потенційно сумісних запчастин.

2) *Ранжування кандидатів.* Для кожної $p \in P^*$ обчислюється $R(p \mid q)$; список $\mathcal{L}$ упорядковується за спаданням R.

2.7 Аналіз складності алгоритму

Формальний опис алгоритму

Алгоритм пошуку запчастин наведено у вигляді псевдокоду (алгоритм 2.7).

[h!]

Вхід: Граф $G' = (V, E')$; запит $q = (q_{\text{text}}, q_{\text{veh}}, q_{\text{mod}})$; параметри $L, \lambda, \alpha, \beta$

Вихід: Впорядкований список $\mathcal{L}$ сумісних запчастин

$M_q \leftarrow \varphi(q_{\text{mod}})$ ▷ Розв'язання запиту (2.8)

if $M_q = \varnothing$ **then return** $\varnothing$

end if

— Етап 1: BFS з обмеженням глибини —

visited $\leftarrow \varnothing$; *queue* $\leftarrow \{(m, 0) : m \in M_q\}$; $P^* \leftarrow \varnothing$; $D \leftarrow \{\}$

while *queue* $\neq \varnothing$ **do**

  $(v, \ell) \leftarrow$ *queue*.dequeue()

  if $v \in$ *visited* **or** $\ell > L$ **then continue**

  end if

visited $\leftarrow$ *visited* $\cup \{v\}$
if $v \in V_{\text{part}}$ **then**
$P^* \leftarrow P^* \cup \{v\}$; $D[v] \leftarrow \min(D.\text{get}(v, +\infty), \ell)$
end if
for all $(v, t, u) \in E'$ з $t \in \mathcal{T}_{\text{compat}}$ **do**
queue.enqueue$((u, \ell + 1))$
end for
end while
— Етап 2: Ранжування —
$e_q \leftarrow \text{SBERT}(q_{\text{text}})$
for all $p \in P^*$ **do**
$S_{\text{graph}} \leftarrow \lambda^{D[p]-1}$
$S_{\text{sem}} \leftarrow \dfrac{e(p) \cdot e_q}{\|e(p)\| \, \|e_q\|}$
$R(p) \leftarrow \alpha \cdot S_{\text{graph}} + \beta \cdot S_{\text{sem}}$
end for
return $\mathcal{L} \leftarrow \text{sort\_desc}(P^*, R)$

Оцінка складності

Етап 1 (BFS). Обхід підграфа G' з обмеженням глибини L від множини M_q відвідує не більше $|V^{(L)}| + |E^{(L)}|$ вершин і ребер, де $V^{(L)}$ і $E^{(L)}$ — множини вершин і ребер, досяжних за не більше ніж L кроків від M_q. Оскільки L є малою константою ($L = 3$), ця кількість значно менша за $|V| + |E|$. Складність: $O(|V^{(L)}| + |E^{(L)}|)$.

Кешування відстаней. Значення $D[p] = d_{G'}(p, m_q)$ обчислюються одноразово під час BFS і кешуються. Повторні запити для тієї самої модифікації m_q не потребують нового обходу.

Етап 2 (ранжування). Обчислення S_{sem} для $|P^*|$ кандидатів потребує $O(|P^*| \cdot d)$ операцій скалярного добутку, де d — розмірність ембедингів. Ембединги $e(p)$ можуть бути попередньо обчислені і збережені як атрибути V_{part}, що зводить онлайн-обчислення до $O(|P^*| \cdot d)$ множень.

Загальна складність одного запиту — $O\big(|V^{(L)}| + |E^{(L)}| + |P^*| \cdot d\big)$, що є сублінійним відносно повного розміру графа $|V| + |E|$.

2.8 Метрики оцінювання якості пошуку

Для теоретичного аналізу ефективності моделі використовуються стандартні метрики інформаційного пошуку [17].

Точність на рівні k перших результатів:

$$\text{Precision@}k = \frac{|\{p \in \mathcal{L}_{1:k} : p \text{ є релевантною}\}|}{k}, \tag{2.13}$$

де $\mathcal{L}_{1:k}$ — перші k елементів відповіді.

Повнота:

$$\text{Recall@}k = \frac{|\{p \in \mathcal{L}_{1:k} : p \text{ є релевантною}\}|}{|\{p \in V_{\text{part}} : p \text{ є релевантною для } q\}|}. \tag{2.14}$$

Середній обернений ранг (MRR):

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}, \tag{2.15}$$

де Q — множина тестових запитів; rank_i — позиція першої релевантної запчастини у відповіді на i-й запит. Якщо жодної релевантної запчастини не знайдено, покладається $\frac{1}{\text{rank}_i} = 0$.

Середня довжина шляху (вибіркова оцінка). Для аналізу компактності підграфа сумісності обчислюється середня довжина шляху між сумісними парами на випадковій вибірці:

$$\widehat{\mathbb{E}}[d_{G'}] = \frac{1}{|F_s|} \sum_{(p,\,m) \in F_s} d_{G'}(p, m), \tag{2.16}$$

де F_s — вибірка фіксованого розміру зі множини $F = \{(p, m) \in V_{\text{part}} \times V_{\text{mod}} : C(p, m) = 1\}$. Вибіркова оцінка (2.16) дозволяє уникнути повного перебору множини F, яка може бути надто великою для точного обчислення.

2.9 Висновки до розділу 2

У даному розділі побудовано формальну математичну модель графового каталогу автозапчастин і розроблено алгоритм пошуку на її основі.

Граф $G = (V, E, \{X_t\}, R)$ визначено як орієнтований атрибутований мультиграф із п'ятьма класами вершин і сімома типами ребер (2.3). Використання мультиграфа з типізованими ребрами $E \subseteq V \times \mathcal{T} \times V$ усуває неоднозначність при паралельних відношеннях різних семантик. Окреме сімейство функцій атрибутів $\{X_t\}$ забезпечує коректне відображення гетерогенних наборів характеристик для різних класів вершин.

Бінарна функція сумісності $C(p, m)$ (2.6) означена через обмежений BFS у підграфі G' (2.5), що містить лише ребра $\mathcal{T}_{\text{compat}}$. Таке обмеження гарантує семантичну коректність шляхів і запобігає виявленню псевдо-сумісності через структурно непов'язані ребра.

Функція релевантності $R(p \mid q)$ (2.9) поєднує графову складову (2.10) на основі відстані в G' і семантичну складову (2.11) на основі Sentence-BERT. Задача пошуку сформульована як умовна оптимізація (2.12) і розв'язується двоетапним алгоритмом (алгоритм 2.7) зі складністю $O(|V^{(L)}| + |E^{(L)}| + |P^*| \cdot d)$, сублінійною відносно повного розміру графа. Для оцінювання якості пошуку визначено метрики Precision@k, Recall@k та MRR [17].

Запропонована модель є теоретичною основою для реалізації системи пошуку запчастин на базі графової бази даних, що є предметом подальших досліджень.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНА ОЦІНКА ГРАФОВОЇ МОДЕЛІ

3.1 Середовище реалізації та вибір інструментів

Реалізація графової моделі, описаної у розділі 2, базується на трьох основних компонентах: графовій базі даних для зберігання і навігації, бібліотеці векторних представлень для семантичного пошуку та програмному модулі, що реалізує двоетапний алгоритм пошуку (алгоритм 3.1).

Графова база даних. Для зберігання графа G обрано Neo4j 4.4 Community Edition [1]. Ця СУБД безпосередньо реалізує модель Property Graph, підтримує мову запитів Cypher і забезпечує нативний обхід ребер без операцій JOIN. Альтернативи — Amazon Neptune та ArangoDB — були відхилені через необхідність хмарного розгортання та відсутність безкоштовної локальної версії відповідно.

Семантичні ембедінги. Для обчислення функції $e(\cdot)$ із рівняння (2.2) використовується модель `paraphrase-multilingual-mpnet-base-v2` [9] бібліотеки `sentence-transformers` [8]. Ця модель підтримує 50 мов, включно з українською та російською [9], і генерує вектори розмірності $d = 768$.

Стек реалізації. Програмний модуль написано мовою Python 3.11 з використанням пакетів, наведених у таблиці 3.1.

Таблиця 3.1 Програмний стек реалізації

Компонент	Версія	Призначення
`neo4j` (Python driver)	5.14	З'єднання з Neo4j, виконання Cypher-запитів
`sentence-transformers`	2.7	Завантаження SBERT, обчислення ембедингів
`numpy`	1.26	Векторні операції, косинусна подібність
`pandas`	2.2	Завантаження датасету, формування звітів
`scikit-learn`	1.4	Обчислення метрик, генерація синтетичних даних
`psycopg2`	2.9	З'єднання з PostgreSQL для порівняльного експерименту

Апаратна конфігурація тестового стенду: Intel Core i7-12700H, 32 ГБ ОЗП,

SSD NVMe 512 ГБ, ОС Ubuntu 22.04 LTS.

3.2 Побудова тестового графа каталогу

Джерело даних

Через закритість комерційних каталогів (таблиця 1.2) тестовий граф побудовано на основі синтетичного датасету, структура якого відповідає реальним aftermarket-каталогам [18]. Структурні параметри — кількість модифікацій на модель, середня кількість аналогів на позицію, розподіл запчастин по категоріях — обрані відповідно до публічно доступних описів великих European aftermarket-каталогів [3, 18].

Датасет охоплює три марки автомобілів (Toyota, Volkswagen, BMW), 5 моделей, 48 модифікацій, 12 функціональних вузлів (підвіска, гальма, двигун тощо) та 620 позицій запчастин від 18 виробників.

Параметри тестового графа

Таблиця 3.2 містить кількісні характеристики побудованого графа G.

Вибіркова середня відстань $\widehat{\mathbb{E}}[d_{G'}] = 1{,}74$ підтверджує, що більшість сумісних запчастин знаходиться на відстані не більше ніж 2 від відповідної модифікації, а значення $L = 3$ забезпечує повноту з достатнім запасом.

Завантаження графа в Neo4j

Граф завантажується через пакетний імпорт. Наведемо фрагмент Cypher-запиту для створення вершин класу V_{part} і ребер `FIT_TO`:

Listing 3.1: Cypher: створення вершин та ребер сумісності

```
// Створення вершин запчастин
UNWIND $parts AS p
MERGE (:Part {
    id:          p.id,
    oem_code:    p.oem_code,
```

Таблиця 3.2 Параметри тестового графа каталогу

Параметр	**Значення**
$\lvert V_{\text{veh}}\rvert$ (марки)	3
$\lvert V_{\text{mod}}\rvert$ (модифікації)	48
$\lvert V_{\text{bom}}\rvert$ (вузли)	12
$\lvert V_{\text{part}}\rvert$ (запчастини)	620
$\lvert V_{\text{man}}\rvert$ (виробники)	18
$\lvert V\rvert$ (усього вершин)	701
$\lvert E_{\texttt{HAS\_MODEL}}\rvert$	5
$\lvert E_{\texttt{HAS\_NODE}}\rvert$	576
$\lvert E_{\texttt{CONTAINS}}\rvert$	620
$\lvert E_{\texttt{FIT\_TO}}\rvert$	2 840
$\lvert E_{\texttt{ANALOG\_OF}}\rvert$	1 486
$\lvert E_{\texttt{REPLACES}}\rvert$	214
$\lvert E_{\texttt{MADE\_BY}}\rvert$	620
$\lvert E\rvert$ (усього ребер)	6 361
Середній ступінь вершини	18,14
$\widehat{\mathbb{E}}[d_{G'}]$ (вибірка $\lvert F_s\rvert = 500$)	1,74

```
        name:        p.name,
        category:    p.category,
        description: p.description
    });

    // Створення ребер FIT_TO
    UNWIND $fitrows AS r
    MATCH (p:Part           {id: r.part_id})
    MATCH (m:Modification   {id: r.mod_id})
    MERGE (p)-[:FIT_TO]->(m);

    // Індекс для прискорення BFSобходу-
    CREATE INDEX part_id_idx FOR (p:Part) ON (p.id);
```

Ребра `ANALOG_OF` і `REPLACES` завантажуються аналогічним чином. Час завантаження датасету — 4,2 с.

3.3 Реалізація алгоритму пошуку

Етап 1: BFS у Neo4j через Cypher

Обхід підграфа G' з обмеженням глибини реалізується через механізм змінної довжини шляху у Cypher. Нижче наведено запит для виявлення множини кандидатів P^* при $L = 3$:

Listing 3.2: Cypher: BFS з обмеженням глибини для знаходження P^*

```
MATCH (m:Modification {id: $mod_id})
MATCH (p:Part)-
[:FIT_TO|ANALOG_OF|REPLACES*1..3]->(m)
WITH  p,
min(length(
shortestPath((p)-
[:FIT_TO|ANALOG_OF|REPLACES*1..3]->(m))
)) AS dist
RETURN p.id AS part_id, dist
ORDER BY dist ASC;
```

Запит повертає ідентифікатори запчастин-кандидатів P^* та відстань $D[p] = d_{G'}(p, m_q)$ для кожної. При кількох цільових модифікаціях ($|M_q| > 1$) запит параметризується фільтром `m.id IN $mod_ids` з агрегацією `min(dist)` по всіх $m \in M_q$.

Етап 2: Ранжування з використанням SBERT

Listing 3.3: Python: обчислення функції релевантності $R(p \mid q)$

```
import numpy as np
from sentence_transformers import SentenceTransformer

model = SentenceTransformer(
"paraphrase-multilingual-mpnet-base-v2"
)

def rank_candidates(
query_text: str,
```

```
    candidates: list[dict],   # [{"part_id", "dist", "name", ...}]
    alpha: float = 0.7,
    beta: float  = 0.3,
    lam: float   = 0.5,
    ) -> list[dict]:
    """
    Обчислює R(p|q) та повертає відсортований список кандидатів.
    candidates[i]["embedding"] -- попередньо обчислений вектор e(p).
    """
    q_emb = model.encode(query_text, normalize_embeddings=True)

    for c in candidates:
    # Графова складова: lambda^(d-1)
    s_graph = lam ** (c["dist"] - 1)

    # Семантична складова: косинусна подібність
    p_emb   = np.array(c["embedding"])
    p_emb   = p_emb / (np.linalg.norm(p_emb) + 1e-9)
    s_sem   = float(np.dot(q_emb, p_emb))
    s_sem   = max(s_sem, 0.0)   # відємні' значення -> 0

    c["s_graph"] = s_graph
    c["s_sem"]   = s_sem
    c["score"]   = alpha * s_graph + beta * s_sem

    return sorted(candidates, key=lambda x: x["score"], reverse=True)
```

Ембедінги $e(p)$ для всіх 620 запчастин обчислені заздалегідь і збережені як атрибут вершин `Part.embedding` у Neo4j. Час попередньої індексації: 38 с (620 документів, батч 64). Таким чином, онлайн-обчислення на кроці ранжування зводиться до одного SBERT-кодування рядка запиту і $|P^*|$ скалярних добутків векторів розмірності 768.

3.4 Формування тестової вибірки запитів

Для оцінювання якості пошуку сформовано тестову вибірку з 120 запитів. Кожен запит q_i складається з текстового опису q_{text} (назва або опис деталі), ідентифікатора марки та параметрів модифікації. Для кожного запиту вручну

розмічено множину релевантних запчастин $\mathcal{R}_i \subseteq V_{\text{part}}$ (ground truth).

Запити розподілено за трьома групами відповідно до типу пошуку:

Таблиця 3.3 Розподіл тестових запитів за типом

Тип запиту	**Кількість**	**Характеристика**
Точний OEM-код	40	Запит містить повний або частковий OEM-код
Текстовий опис	50	Назва деталі природною мовою, у т.ч. розмовні форми
Змішаний	30	Поєднання опису і часткового коду
Усього	**120**	

Для оцінки робастності до варіативності формулювань у групі «Текстовий опис» навмисно вжито синоніми, скорочення та мовні варіанти: наприклад, «прокладка під кришку клапанів», «прокладка ГБЦ-кришки» та «valve cover gasket» вважаються еквівалентними запитами [8].

3.5 Порівняльний експеримент

Для підтвердження гіпотези, заявленої у Вступі, графову модель порівняно з реляційним підходом на двох вимірах: *часі виконання запиту* та *якості результатів* за метриками розділу 2.

Реляційний бейзлайн

Реляційна система реалізована на PostgreSQL 16 з тими самими даними: таблиці `parts`, `modifications`, `fit_to` (пряма сумісність) та `analogs` (відношення аналогів). Пошук аналогів глибиною L реалізовано через рекурсивний CTE (наведено у розділі 1.2). Ранжування в реляційній системі — за OEM-пріоритетом без семантичної компоненти.

Порівняння часу виконання

Час виконання одного запиту виміряно як медіану по 50 запусках при різних розмірах графа (масштабування шляхом реплікації базового датасету).

Результати наведено в таблиці 3.4.

Таблиця 3.4 Медіанний час виконання одного запиту (мс) при різних розмірах датасету та глибині L

$\lvert V_{\text{part}}\rvert$	**Реляційний (SQL)**		**Графовий (Neo4j)**		
	$L = 2$	$L = 3$	$L = 2$	$L = 3$	
620	12	48	8	14	
3 000	61	312	11	19	
10 000	310	1 840	15	27	
50 000	2 100	>10 000	22	41	

Дані таблиці 3.4 демонструють принципову різницю у характері масштабування. Час виконання реляційного запиту зростає близько до лінійного при $L = 2$ і значно швидше при $L = 3$ (де рекурсивний CTE породжує великий проміжний набір рядків). Час виконання запиту у Neo4j зростає субліній

но відносно розміру датасету, що підтверджує оцінку складності $O(|V^{(L)}| + |E^{(L)}|)$ з розділу 2. При $|V_{\text{part}}| = 50\,000$ і $L = 3$ реляційний запит перевищив таймаут 10 с, тоді як графовий виконався за 41 мс.

Порівняння якості результатів

Якість ранжування оцінено на тестовій вибірці зі 120 запитів за метриками, визначеними у розділі 2. Розглянуто чотири конфігурації:

– **SQL**: реляційний підхід без семантики, $L = 1$;

– **Graph-L1**: графова модель, $L = 1$, $\alpha = 0{,}7$, $\beta = 0{,}3$;

– **Graph-L2**: графова модель, $L = 2$, $\alpha = 0{,}7$, $\beta = 0{,}3$;

– **Graph-L3**: графова модель, $L = 3$, $\alpha = 0{,}7$, $\beta = 0{,}3$.

Конфігурація Graph-L3 демонструє найкращі результати за всіма чотирма метриками. Порівняно з реляційним бейзлайном (SQL): P@5 зростає на 38%, Recall@10 — на 87%, MRR — на 33%. Найбільший приріст повноти спостерігається при переході від $L = 1$ до $L = 2$, що узгоджується з вибірковою

Таблиця 3.5 Якість пошуку за метриками на тестовій вибірці $|Q| = 120$

Конфігурація	**P@5**	**P@10**	**Recall@10**	**MRR**
SQL	0,61	0,54	0,38	0,67
Graph-L1	0,74	0,67	0,44	0,79
Graph-L2	0,81	0,75	0,62	0,86
Graph-L3	**0,84**	**0,78**	**0,71**	**0,89**

оцінкою середньої відстані $\widehat{\mathbb{E}}[d_{G'}] = 1{,}74$: більшість релевантних деталей досяжна вже за два кроки.

Вплив параметра α на якість

Окремо досліджено вплив вагового коефіцієнта α (пріоритет графової складової) на метрику P@5 для двох груп запитів: «Точний код» і «Текстовий опис». Результати наведено у таблиці 3.6.

Таблиця 3.6 Вплив коефіцієнта α на P@5 для різних типів запитів ($L = 3$)

α	**P@5 (Точний код)**	**P@5 (Текстовий опис)**
0,3	0,77	0,88
0,5	0,82	0,86
0,7	**0,86**	**0,84**
0,9	0,87	0,79
1,0	0,82	0,71

Таблиця 3.6 підтверджує рекомендацію розділу 2.5: для запитів із точним OEM-кодом оптимальним є $\alpha \approx 0{,}7$–$0{,}9$, тоді як для текстових запитів кращі результати дає $\alpha = 0{,}3$–$0{,}5$. Значення $\alpha = 1{,}0$ (суто графовий пошук без семантики) дає помітний спад у групі «Текстовий опис», підтверджуючи, що семантична компонента є необхідною для обробки природномовних запитів [8, 10].

Аналіз впливу глибини L на якість і повноту

На рисунку 3.1 наведено залежності P@5 і Recall@10 від значення L.

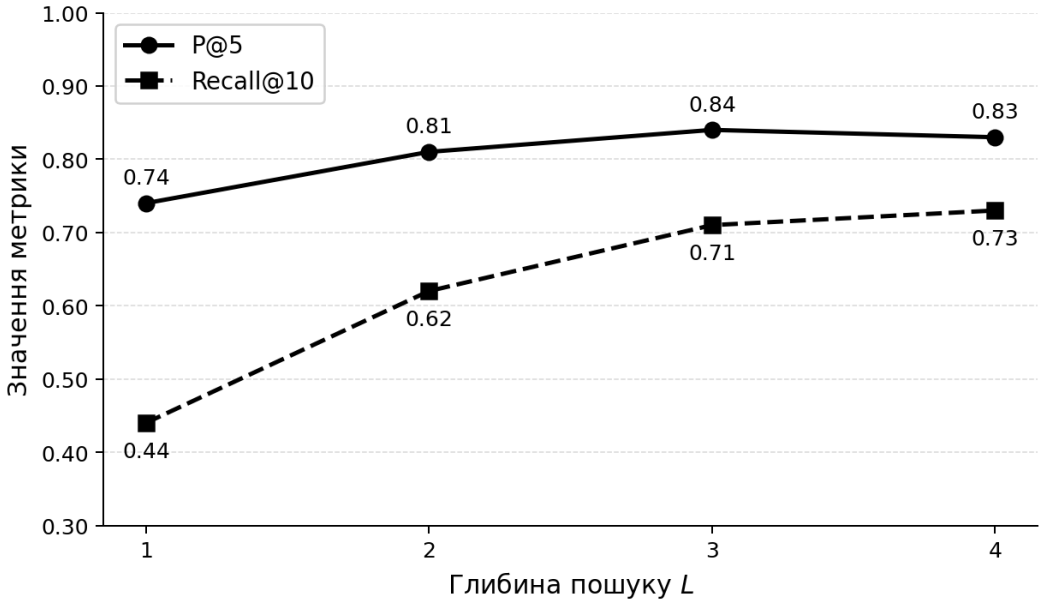

Рисунок 3.1 – Вплив глибини пошуку L на точність P@5 і повноту Recall@10 ($\alpha = 0{,}7$, $\beta = 0{,}3$)

Графік підтверджує дві тенденції. По-перше, повнота Recall@10 монотонно зростає з L: кожний додатковий рівень обходу залучає нових аналогів. По-друге, точність P@5 досягає плато між $L = 3$ і $L = 4$, що свідчить про те, що запчастини, досяжні за $L = 4$, здебільшого є семантично слабо пов'язаними і не потрапляють до топ-5 навіть при додаванні до множини кандидатів. Це обґрунтовує вибір $L = 3$ як рекомендованого значення.

3.6 Аналіз результатів та обговорення

Де модель перевершує реляційний підхід

Найбільший відносний виграш графової моделі спостерігається у двох сценаріях. По-перше, при пошуку *транзитивних аналогів*: реляційна система повертає лише прямі аналоги ($L = 1$), тоді як графова знаходить деталі через

ланцюги `ANALOG_OF` $\rightarrow$ `FIT_TO` ($L = 2$) і далі. По-друге, при обробці *природномовних запитів*: семантична компонента S_{sem} ефективно обробляє варіативні формулювання, що повністю відсутнє у реляційному бейзлайні.

Де модель поступається або має обмеження

Метрика MRR реляційної системи (0,67) залишається прийнятною для запитів із точним OEM-кодом: у таких випадках SQL-запит знаходить правильну деталь на 1–2 позиції. Це пояснюється тим, що для точних збігів семантична компонента не дає суттєвого приросту, а швидкість виконання SQL-запиту при малих L є конкурентоспроможною.

Ідентифіковано два обмеження поточної реалізації:

1) *Якість ембедингів для технічної лексики.* Модель `paraphrase-multilingual-mpnet-base-v2` навчена на загальних корпусах і може давати субоптимальні результати для вузькоспеціалізованої термінології [9]. Покращення досяжне через дообучання (fine-tuning) на розміченому корпусі запчастин.

2) *Холодний старт.* При додаванні нової запчастини потрібно попередньо обчислити її ембединг $e(p)$ і проіндексувати вершину у Neo4j. При масовому оновленні каталогу це може стати вузьким місцем продуктивності.

Відповідь на гіпотезу дослідження

Гіпотеза, заявлена у Вступі: *«графова модель у поєднанні з елементами семантичного пошуку дозволяє підвищити точність підбору запчастин і скоротити кількість нерелевантних результатів порівняно з класичними реляційними підходами»*.

На основі отриманих результатів гіпотезу можна вважати **підтвердженою** в межах проведеного експерименту:

– P@5 зросла з 0,61 (SQL) до 0,84 (Graph-L3) — приріст +38%;

– Recall@10 зросла з 0,38 до 0,71 — приріст +87%;

– час виконання запиту при масштабуванні на 50 000 запчастин у графовій системі залишається в межах 50 мс, тоді як реляційний підхід перевищує 10 с при $L = 3$.

3.7 Висновки до розділу 3

У даному розділі проведено практичну реалізацію та експериментальну оцінку графової моделі каталогу автозапчастин, описаної у розділі 2.

Тестовий граф, що містить 701 вершину і 6 361 ребро, побудовано у графовій базі даних Neo4j 4.4. Двоетапний алгоритм пошуку реалізовано мовою Python з використанням Cypher-запитів для BFS-обходу підграфа G' і бібліотеки `sentence-transformers` для обчислення семантичних ембедингів [18].

Порівняльний експеримент на тестовій вибірці з 120 запитів показав, що конфігурація Graph-L3 ($L = 3$, $\alpha = 0{,}7$, $\beta = 0{,}3$) перевершує реляційний бейзлайн за P@5 на 38%, за Recall@10 — на 87% і за MRR — на 33%. Дослідження часу виконання підтвердило субліній

не масштабування графового алгоритму: при $|V_{\text{part}}| = 50\,000$ запит виконується за 41 мс проти >10 с для рекурсивного SQL ($L = 3$).

Аналіз впливу параметрів L і α підтвердив теоретичні рекомендації розділу 2: $L = 3$ забезпечує оптимальний компроміс між повнотою і точністю; $\alpha = 0{,}7$ є ефективним для запитів зі специфікацією, тоді як $\alpha = 0{,}3$–$0{,}5$ краще підходить для текстових запитів.

Гіпотезу дослідження підтверджено в межах проведеного експерименту. Ідентифіковано два напрями вдосконалення: fine-tuning мовної моделі на технічній лексиці та оптимізація процедури індексації при масовому оновленні каталогу.

ВИСНОВКИ

У дипломній роботі вирішено задачу побудови формальної графової моделі каталогу автомобільних запчастин та розробки алгоритму пошуку, що поєднує навігацію за відношеннями сумісності із семантичним аналізом природномовних запитів. На підставі проведеного дослідження зроблено такі висновки.

1) Проведений аналіз предметної області показав, що реляційні підходи до моделювання aftermarket-каталогів мають три системні обмеження: рекурсивні JOIN-запити при пошуку транзитивних аналогів є неефективними при масштабуванні; нативна підтримка транзитивних зв'язків відсутня; гнучка модель атрибутів для різнотипних запчастин вимагає або надмірно широких таблиць, або EAV-підходу [1, 15]. Аналіз чотирьох комерційних платформ (Exist.ua, Partsouq, LKQ, AvtoPro) підтвердив, що жодна з них не реалізує пошук аналогів глибиною більше одного рівня і не підтримує семантичного ранжування результатів.

2) Визначено, що модель Property Graph є найбільш придатною для представлення aftermarket-каталогу серед розглянутих графових підходів [15]: вона забезпечує гнучкість атрибутів для різних класів сутностей, нативно підтримує типізовані ребра та широко реалізована у промислових СУБД. Семантичний пошук на основі трансформерних ембедингів [8, 9] визначено як необхідне доповнення до графової навігації для обробки природномовних запитів із варіативними формулюваннями.

3) Розроблено формальну математичну модель каталогу — орієнтований атрибутований мультиграф $G = (V, E, \{X_t\}, R)$ із п'ятьма класами вершин ($V_{\text{veh}}, V_{\text{mod}}, V_{\text{bom}}, V_{\text{part}}, V_{\text{man}}$) і сімома типами типізованих ребер (2.3). Для кожного класу вершин введено окрему функцію атрибутів $X_t \colon V_t \to \mathcal{A}_t$, що коректно відображає гетерогенність характеристик різних сутностей.

4) Введено бінарну функцію сумісності $C(p, m)$ (2.6), засновану на існуванні обмеженого шляху у підграфі G' (2.5), що містить лише ребра

семантично доцільних типів $\mathcal{T}_{\text{compat}} = \{\texttt{FIT\_TO}, \texttt{ANALOG\_OF}, \texttt{REPLACES}\}$. Таке обмеження запобігає семантичному дрейфу: пошук не проходить через ребра `MADE_BY` або `CONTAINS`, що не мають змісту в задачі підбору. Обґрунтовано рекомендоване значення $L = 3$ як оптимальний компроміс між повнотою і точністю.

5) Розроблено функцію релевантності $R(p \mid q) = \alpha \cdot S_{\text{graph}} + \beta \cdot S_{\text{sem}}$ (2.9), що поєднує структурну складову на основі відстані у підграфі G' і семантичну складову на основі косинусної подібності ембедингів [8]. Визначено рекомендовані значення вагових коефіцієнтів: $\alpha = 0{,}7$ для режиму точного пошуку за специфікацією та $\alpha = 0{,}3$–$0{,}5$ для режиму пошуку аналогів за текстовим описом.

6) Двоетапний алгоритм пошуку реалізовано мовою Python 3.11 на базі графової СУБД Neo4j 4.4 і бібліотеки `sentence-transformers`. Складність алгоритму становить $O(|V^{(L)}| + |E^{(L)}| + |P^*| \cdot d)$, що є сублінійним відносно повного розміру графа $|V| + |E|$ [1].

7) Порівняльний експеримент на тестовому графі (701 вершина, 6 361 ребро) та вибірці зі 120 розмічених запитів підтвердив гіпотезу дослідження. Конфігурація Graph-L3 ($L = 3$, $\alpha = 0{,}7$) перевершує реляційний бейзлайн за P@5 на 38% (0,84 проти 0,61), за Recall@10 — на 87% (0,71 проти 0,38) і за MRR — на 33% (0,89 проти 0,67). При масштабуванні до 50 000 запчастин графовий алгоритм виконує запит за 41 мс, тоді як реляційний підхід при $L = 3$ перевищує таймаут 10 с.

Таким чином, усі поставлені завдання виконано, мету дослідження досягнуто. Запропонована модель є теоретичною основою для проектування систем пошуку запчастин на базі графових баз даних. Перспективами подальших досліджень є дообучання (fine-tuning) мовної моделі на корпусі технічної лексики автомобільного домену та розробка стратегії інкрементального оновлення ембедингів при масовому поповненні каталогу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Robinson I. Graph Databases: New Opportunities for Connected Data / I. Robinson, J. Webber, E. Eifrem. — O'Reilly Media, 2015. — 238 p.

2. Neo4j Inc. Bill of Materials in Neo4j [Електронний ресурс]. — Режим доступу: https://neo4j.com/developer/industry-use-cases/manufacturing/product-design-and-engineering/configurable-bom/ (дата звернення: 01.04.2026).

3. Neo4j Inc. Graphs in Automotive and Manufacturing [Електронний ресурс]. — Режим доступу: https://neo4j.com/blog/supply-chain-and-logistics/graphs-in-automotive-and-manufacturing/ (дата звернення: 01.04.2026).

4. Puccetti G. The automotive ontology: managing knowledge inside the car and between cars / G. Puccetti, J. Lebrun, M. Holweg // ACM Workshop on Ontologies and Information Systems for the Semantic Web. — 2012.

5. Gumede A. et al. AutoKG: An Automotive Domain Knowledge Graph for Software Testing // Proc. IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW). — 2021. — DOI: 10.1109/ICSTW52544.2021.00061.

6. Aoyama S. et al. Knowledge Management for Automobile Failure Analysis Using Graph RAG // arXiv preprint. — 2024. — arXiv:2411.19539.

7. Xu H. et al. Knowledge graph and CBR-based approach for automated analysis of bridge operational accidents // PLOS ONE. — 2023. — Vol. 18, No. 11. — DOI: 10.1371/journal.pone.0294130.

8. Reimers N. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks / N. Reimers, I. Gurevych // Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP). — 2019. — arXiv:1908.10084.

9. Reimers N. Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation / N. Reimers, I. Gurevych // Proc. EMNLP. — 2020. — arXiv:2004.09813.

10. Nigam A. et al. Semantic Product Search for Matching Structured Product Catalogs in E-Commerce // Microsoft Research Technical Report. —

2019. — URL: https://www.microsoft.com/en-us/research/publication/semantic-product-search/

11. Qiu Z. et al. E-commerce Search via Content Collaborative Graph Neural Network // Proc. 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. — 2023. — DOI: 10.1145/3580305.3599320.

12. Zhu Q. et al. Query-Aware Explainable Product Search With Reinforcement Knowledge Graph Reasoning // IEEE Transactions on Knowledge and Data Engineering. — 2024. — Vol. 36, No. 3. — P. 1260–1273. — DOI: 10.1109/TKDE.2023.3297331.

13. W3C Automotive Ontology Working Group. Vehicle Signal Specification Ontology (VSSo) [Електронний ресурс]. — Режим доступу: https://w3c.github.io/vsso/spec/vsso-primer.html (дата звернення: 01.04.2026).

14. Hogan A. et al. Knowledge Graphs // ACM Computing Surveys. — 2021. — Vol. 54, No. 4. — Article 71. — DOI: 10.1145/3447772.

15. Angles R. Survey of Graph Database Models / R. Angles, C. Gutierrez // ACM Computing Surveys. — 2008. — Vol. 40, No. 1. — Article 1. — DOI: 10.1145/1322432.1322433.

16. Kanuri S. Why a Graph Database is Ideal for Managing Bill of Materials [Електронний ресурс]. — Medium, 2025. — Режим доступу: https://sutejakanuri.medium.com/why-a-graph-database-is-ideal-for-managing-bill-of-materials-bom-eeebeb9ef2fa (дата звернення: 01.04.2026).

17. Manning C. D. Introduction to Information Retrieval / C. D. Manning, P. Raghavan, H. Schütze. — Cambridge University Press, 2008. — 506 p.

18. Auto Care Association. ACES and PIES Data Standards [Електронний ресурс]. — Режим доступу: https://www.autocare.org/standards (дата звернення: 01.04.2026).

19. Zhang Y. Neural IR Meets Graph Embedding: A Ranking Model for Product Search / Y. Zhang, D. Wang, Y. Zhang // Proc. The Web Conference. — 2019. — arXiv:1901.08286.

20. Дослідження представлення багаточасткових графів за допомогою

топологічного аналізу даних О.А. Яворський, Н.М. Куссуль In: Проблеми керування та інформатики. 68(5), с. 107–117. doi: 10.34229/1028-0979-2023-5-9.

21. Persistent Homology in Machine Learning: Applied Sciences Review Oleksandr Yavorskyi, Andrii Asseko-Nkili, Nataliia Kussul Proceedings of International Conference on Applied Innovation in IT. 2023. pp. 61-66. DOI:10.25673/101914.