НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
«Інститут прикладного системного аналізу»
(повна назва інституту/факультету)

Кафедра системного проектування
(повна назва кафедри)

«На правах рукопису»
УДК _004.852

«До захисту допущено»

Завідувач кафедри
_____ _А.І. Петренко
(підпис) (ініціали, прізвище)
“___”_____20__ р.

Магістерська дисертація

зі спеціальності (спеціалізації)__122 – Комп'ютерні науки та інформаційні технології (спеціалізація – Інформаційні системи та технології проектування)
(код і назва спеціальності)

на тему: Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей

Виконав: студент _6_ курсу, групи _____ДА-61м
(шифр групи)

_____Кравчук Євгеній Сергійович _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник _ст.викл., к.т.н. Сергеєв-Горчинський О.О. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____Розроблення стартап-проекту _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2018 року

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»

Інститут/факультет ННК «Інститут прикладного системного аналізу»
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-професійною (освітньо-науковою) програмою

Спеціальність (спеціалізація) 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І. Петренко
(підпис) (ініціали, прізвище)
«___»_____20__ р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Кравчуку Євгенію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема дисертації «Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей»
науковий керівник дисертації Сергеєв-Горчинський О.О., ст.викл, к.т.н.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «_27_»___03__ 2018 р. №_1028-с_____

2. Строк подання студентом дисертації _____

3. Об'єкт дослідження __Онтології_____

4. Предмет дослідження _Алгоритми автоматичної побудови онтологій, що можна застосовувати у різних доменних областях

5. Перелік завдань, які потрібно розробити провести огляд існуючих методів та технологій побудови онтологій навчальних дисциплін, розробити програму з реалізацією алгоритму семантичного графу знань та продемонструвати її роботу на приклаі, описати перспективи розвитку роботи, оформити роботу на основі отриманих результатів

6. Орієнтовний перелік публікацій Кравчук Є.С., Сергеєв-Горчинський О.О. Методи оцінки семантичної орієнтації тексту / Є.С. Кравчук // System Analysis And Information Technology, May 21-24, 2018, Kyiv, Ukraine, - С. 70-71

7. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Реалізація стартап-проекту			

8. Дата видачі завдання _01.02.2018_

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	Отримання завдання	01.02.2018	
2	Збір інформації та аналіз літератури	15.02.2018	
3	Проведення огляду існуючих методів побудови онтологій	28.02.2018	
4	Вибір стеку технологій та корпусу даних	13.04.2018	
5	Реалізація семантичного графу знань	25.04.2018	
6	Оформлення дипломної роботи	30.04.2018	
7	Отримання допуску до захисту та подача роботи в ДЕК	10.05.2018	

Студент _____ ___Кравчук Є.С.____
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ _Сергеєв-Горчинський О.О_
(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено наукового керівника

РЕФЕРАТ МАГІСТЕРСЬКОЇ ДИСЕРТАЦІЇ

виконаної на тему: Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей

студентом: Кравчуком Євгенієм Сергійовичем

Робота виконана на 96 сторінках, містить 25 ілюстрації, 23 таблиць. При підготовці використовувалась література з 32 джерел.

Актуальність теми

В наш час з ростом кількості інформації дуже гостро постає проблема у структуризації накопичених даних задля отримання корисної інформації з них, зокрема у сфері навчання. Стрімкий ріст ринку масових онлайн курсів та поява нових дисциплін та напрямів навчання робить вищезгадану проблему актуальною і у даній сфері.

Існуючі засоби опису онтологій навчальних дисциплін та курсів вже не можуть повною мірою відповідати потребам по швидкості створення, підтримці в актуальному стані та обслуговуванню.

Як результат, тема набуває особливої актуальності у контексті опису навчальних дисциплін зокрема, та представлення доменних знань вцілому.

Мета та задачі дослідження

Метою даної роботи є дослідження існуючих підходів до опису та побудови онтологій, зокрема, у домені навчальних дисциплін та розробка власного сервісу, що дозволяє автоматично будувати представлення доменних знань з корпусу документів з попередньо зазначеною схемою.

Рішення поставлених завдань та досягнуті результати

Для вирішення поставлених завдань, у роботі пропонується модель семантичного графу знань, практичну цінність котрого представлено програмною реалізацією сервісу, ядро котрого реалізовує математичну модель семантичного графу знань. Для взаємодії з графом було додано REST API. Результати роботи представлені рядом схем, графіків, знімків екрану, що показують процес створення, розгортання та роботи сервісу, включаючи опис та пояснення.

Об’єкт досліджень

Онтології навчальних дисциплін, методи автоматичної побудови онтологій довільних доменних областей.

Предмет досліджень

Підходи до створення масштабованих додатків для автоматичної побудови представлення доменних знань та математичні моделі. що описують дані підходи.

Методи досліджень

Для вирішення проблеми в даній роботі використовуються методи аналізу і синтезу, системного аналізу, порівняння, логічного узагальнення результатів.

Наукова новизна

Наукова новизна роботи полягає у створенні моделі, що дозволяє в автоматичному режимі будувати семантичний граф знань з подальшою реалізацію пропонованих ідей та аналізом результатів.

Практичне значення одержаних результатів

Отримані результати відкривають нові можливості для бізнесу та інших сфер діяльності, шляхом аналізу та систематизації накопичених даних.

Апробації результатів дисертації

Результати досліджень оприлюднені на міжнародній конференції «System Analysis And Information Technology», травень 2018 року.

Публікації

Кравчук Є.С., Сергеєв-Горчинський О.О. Методи оцінки семантичної орієнтації тексту // System Analysis And Information Technology, May 21-24, 2018, Kyiv, Ukraine, - С. 70-71

Ключові слова

Онтологія, семантичні бази даних, граф знань, data mining, семантичний веб, семантичний пошук.

ABSTRACT ON MASTER'S THESIS

on topic: Ontological models of educational disciplines and application of Data Mining for assessment of their content interaction

student: Yevgeny S. Kravchuk

Work carried out on 96 pages containing 25 figures, 23 tables. The paper was written with references to 32 different sources.

Topicality

Nowadays, due to increasing amount of information, the problem of structuring the accumulated data in order to obtain useful information from it becomes very acute, especially in education field. The rapid growth in the market of massive online courses and the emergence of new disciplines and directions of training makes the above-mentioned problem relevant in this area as well.

Existing tools for describing ontologies of academic disciplines and courses can no longer fully meet the needs for speed of creation, maintenance and service.

As a result, the topic becomes especially relevant in the context of the description of academic disciplines in particular, and the presentation of domain knowledge in general.

Purpose

The purpose of this work is to study existing approaches to the description and construction of ontologies, in particular, in the domain of educational disciplines and the development of our own service, which allows will allow to build the representation of domain knowledge from the body of documents with the previously specified scheme automatically.

Solution

In order to solve the problems, a model of the semantic graph of knowledge is proposed, the practical value of which is represented by the software implementation of the service, the core of which implements the mathematical model of the semantic graph of knowledge. REST API was added to interact with the graph. The results of the work are presented in a series of diagrams, charts, screenshots showing the process of creating, deploying and operating the service, including a description and explanation.

Object of research

Ontology of educational disciplines, methods of automatic construction of ontologies of arbitrary domain areas.

Subject of research

Approaches to creating scalable applications for automatic building of the domain knowledge representation and mathematical models describing these approaches.

Research methods

To solve the problem were used methods of analysis and synthesis, system analysis, comparison, logical generalization of results in this paper .

Scientific novelty

The scientific novelty of the work consists in the creation of a model that allows to build a semantic graph of knowledge with the further implementation of the proposed ideas and analysis of the results in the automatic mode.

The practical value of the results

The obtained results open new opportunities for business and other fields of activity, by analyzing and systematizing the accumulated data.

Research results approbation

Research results presented at the international conference "System Analysis and Information Technologies", 2018.

Publications

Kravchuk Y.S., Sergeev-Gorchynsky O.O. Methods of estimating the semantic orientation of the text // System Analysis and Information Technology, May 21-24, 2018, Kyiv, Ukraine, pp. 70-71.

Keywords

Ontology, semantic databases, graph of knowledge, data mining, semantic web, semantic search.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – application programming interface, прикладний програмний інтерфейс

SKG – Semantic Knowledge Graph, семантичний граф знань

REST – Representational State Transfer, передача представлення стану

PGMHD - Probabilistic Graphical Model for Massive Hierarchical Data, імовірнісна графова модель для масивних ієрархічних даних

ВСТУП

Стрімкий розвиток онлайн навчання та масових онлайн-курсів призвели до появи величезної кількості непов’язаних між собою даних. Наприклад, тільки проект Coursera, котрий налічує приблизно тисячу курсів. Також гігантами у цій сфері є EdX, Udemy, Udacity та ряд інших, зокрема OpenCourseware, - освітній ресурс Масачусетського Технологічного Інституту.

В Україні прикладом проекту масових відкритих онлайн-курсів є Prometheus, котрий може скласти гідну конкуренцію закордонним проектам та має виключну якість матеріалу.

З огляду на стрімкий зріст автоматизації навчальних процесів, виникає потреба у зборі, агрегації та пов’язування навчальних дисциплін та матеріалів в одну систему. Розробка методики, що охоплює ці процеси дозволить створювати розподілені системи дистанційного навчання і використовувати різні учбові матеріали.

На сьогоднішній день для дистанційного навчання використовуються курси, котрі містять у собі інформаційний документ з переліком основних понять та предметів і посилання на літературу. Основною проблемою розвитку дистанційного навчання є «механічність» розробки освітніх курсів, так як створення освітнього курсу, як правило, відбувається переносом матеріалу лекції вручну у програмний додаток. В даний час проблемою є ефективний опис та організація існуючого контенту для полегшення обміну, повторного використання та модифікації.

Семантичні мережі вирішують цю проблему. Семантична мережа - це область досліджень штучного інтелекту, що має за мету створення метаданих. Основний виклик семантичного вебу - представити саме значення контенту, зрозуміле кожному користувачу.

Внесок даної роботи полягає у дослідженні можливості застосування онтологій для представлення доменної області навчальних дисциплін та курсів. По-друге, у

роботі досліджено інноваційні сучасні методи автоматичної побудови графу знань у межах будь-якої доменної області. По-третє, в ході порівняльного аналізу був обраний та реалізований метод SKG для автогенерації семантичного графу знань.

1 ОНТОЛОГІЇ ТА СЕМАНТИЧНИЙ ВЕБ

Онтологія - термін, запозичений з філософії, який означає науку, що описує форми буття і те, як вони відносяться між собою. Іншими словами онтологія - це явний опис концептуалізації [1]. Веб-онтологія може включати опис класів, їх властивостей, а також індивідів класів. Онтології грають критично важливу роль у організації обробки знань на базі веб, їх спільного використання та обміну ними між додатками. Онтології в загальному вигляді визначаються як спільно використовувані формальні концептуалізації конкретних предметних областей, вони дають загальне уявлення про теми, інформацію про які можуть обмінюватися люди і додатки.

1.1 Формальний опис поняття онтології

Онтологія – це специфікація концептуалізації, формалізоване представлення основних понять та зв'язків між ними [2]. Компонентами онтології є:

множина понять предметної області та їх атрибутів;

множина відношень, або асоціацій між виділеними поняттями;

множина аксіом і правил виведення, заданих на виділених поняттях.

Формальну модель онтології можна представити формулою:

$$O = <C, Inst, R, I>$$

де: C— множина концептів (класів) предметної області; Inst — множина екземплярів; R — множина відношень між ними; I — множина правил інтерпретації, що визначають семантику концептів.

Концепт предметної області C складається з наступних компонент і описується формулою:

$$C = <Name, SupC, Icc>$$

де: Name – найменування концепту; SupC – множина концептів, від яких унаслідуваний даний концепт; Icc ⊂ I – підмножина всіх правил інтерпретації, що визначають семантику даного концепту.

Екземпляр концепту має наступну структуру, описану формулою:

Inst = <Name, CS, Rel>

де: Name – найменування екземпляра; CS – множина концептів, до якої належить екземпляр; Rel ⊂ R – підмножина відношень, в котрих присутній екземпляр. Відношення між концептами та їх екземплярами R задаються наступним чином, формулою:

R = <Name, Domain, Range>

де: Name – найменування відношення; Domain – область визначення; Range – область значень.

1.2 Алгоритм побудови онтології предметної області

Побудова онтології буде відбуватися за наступним алгоритмом [3]:

- Визначення області і масштабу онтології.
- Розгляд варіантів повторного використання існуючих онтологій.
- Перелік важливих термінів онтології.
- Визначення класів та їх ієрархії.
- Визначення властивостей класів та створення їх екземплярів.

Для визначення області та масштабу онтології треба відповісти на наступні питання:

1. Яку область буде охоплювати онтологія?
2. Для чого вона використовується?
3. На які типи питань повинна давати відповіді інформація в онтології?
4. Хто буде використовувати та підтримувати онтологію?

Далі буде проведено огляд специфікацій та стандартів онтологій, котрі широко використовуються в світі для опису доменних моделей навчальних курсів, матеріалів, установ тощо.

1.3 Мови опису онтологій навчальних дисциплін

Відношення між семантичними поняттями зазвичай визначаються за правилами логічного виводу. Консорціумом W3C (World Wide Web Consortium) розроблені два основних стандарти для представлення онтологій, а саме: RDF Schema (RDFS) і мова OWL (Web Ontology Language). RDFS і OWL засновані на RDF (Resource Description Framework), стандарті на основі XML, який дозволяє визначити трійки (триплети). Трійки визначають відношення між довільними суб'єктом і об'єктом за допомогою так званого предикату. В RDF і OWL суб'єкти, об'єкти і предикати визначаються через однакові ідентифікатори ресурсів URI (Uniform Resource Identifier), або відповідно багатомовні ідентифікатори ресурсів (Iris) [4]. Таким чином, стандарт RDF забезпечує універсальний і гнучкий механізм для представлення знань. Далі кожен з форматів описується більш детально.

1.3.1 Мова опису онтологій RDF

Для зберігання даних в Semantic Web була розроблена модель RDF (Resource Description Framework). RDF - це мова загального призначення для представлення інформації в Інтернеті. RDF представляє твердження про ресурси у вигляді, придатному для машинної обробки та Data Mining [5]. Ресурсом в RDF може бути будь-яка сутність, як інформативна, так і неінформативна. RDF розроблений для представлення інформації гнучким способом з мінімумом обмежень та може використовуватися в ізольованих додатках, де спеціально розроблені формати можуть бути більш виправдані.

Зокрема, підтримуються детально описані поняття відношення наслідку, які надають базис для визначення надійних правил логічного висновку в даних.

1.3.2 Мова опису онтологій OWL

OWL (Ontology Web Language) призначена для запису семантики предметних областей у вигляді онтології. OWL був створений у відповідь на потребу в стандартизації способів подання знань в веб.

OWL розроблений для додатків, які не просто надають інформацію користувачеві, але й виконують маніпуляції над даними. OWL розширює і доповнює технології представлення семантичних даних, таких як XML, RDF і RDF Schema. В основі мови - представлення дійсності в моделі даних «об'єкт - властивість» [8]. OWL придатний для опису не тільки веб-сторінок, але й будь-яких об'єктів дійсності. Кожному елементу опису в цій мові (у тому числі властивості, що зв'язує об'єкти) ставиться в відповідність URI. OWL дозволяє описати значення термінів в словниках та відношення між ними. В відмінності від RDF, OWL дозволяє явно описати властивості та класи: наслідування класів, метахарактеристики, потужність зв'язків та еквівалентність. Існує три діалекти мови OWL, що розрізняються за складністю та рівнем розпізнавання можливостей: OWL Lite, OWL DL, OWL. Розробники онтологій, які використовують OWL, повинні вирішити, який з діалектів краще підходить для їх завдань.

Отримавши статус рекомендації W3C, мова OWL стала активно використовуватися в програмних продуктах та дослідницьких проектах.

1.3.3 Мова опису онтологій SPARQL

Для пошуку і виведення знань з баз знань використовується мова SPARQL. SPARQL розшифровується як SPARQL Protocol and RDF Query Language і є мовою запитів до даних, представленим по моделі RDF, а також протоколом для передачі цих запитів і відповідей на них. SPARQL використовується для подання запитів до різноманітних джерел даних, незалежно від того, зберігаються ці дані безпосередньо в RDF або подаються у вигляді RDF за допомогою проміжного програмного забезпечення [11].

Результати запитів SPARQL можуть бути представлені результуючими наборами онтологій або RDF-графами.

```
SELECT LO
FROM
{los:LO}
{los:contents}
{ums:topic};
{dms:Concept};
{ums:Role};
WHERE
(Concept=”SQL” and Role=”description”)
or
(Concept=”Relational algebra” and Role=”illustration”)
```

Лістинг 1 - Приклад запиту Learning Object мовою SPARQL.

1.4 Світові стандарти опису онтологій навчальних дисциплін

У даному розділі будуть описані специфікації та стандарти онтологій, котрі широко використовуються в світі для опису доменних моделей навчальних курсів, матеріалів, установ тощо.

1.4.1 Онтологія Learning Object Metadata Ontology (LOM)

Онтологія та словник, що використовується для представлення IEEE LOM – стандарт метаданих для освітнього контенту. Це своєрідний «міст» для зв'язування навчальних метаданих з Linked Open Data. Візуалізація графу даної онтології представлена на рисунку 1.1, далі, на рисунку 1.2 проілюстровано список її класів. У даній онтології створено маппінг елементів IEEE LOM з RDF, заснованого на принципах Linked Data [8].

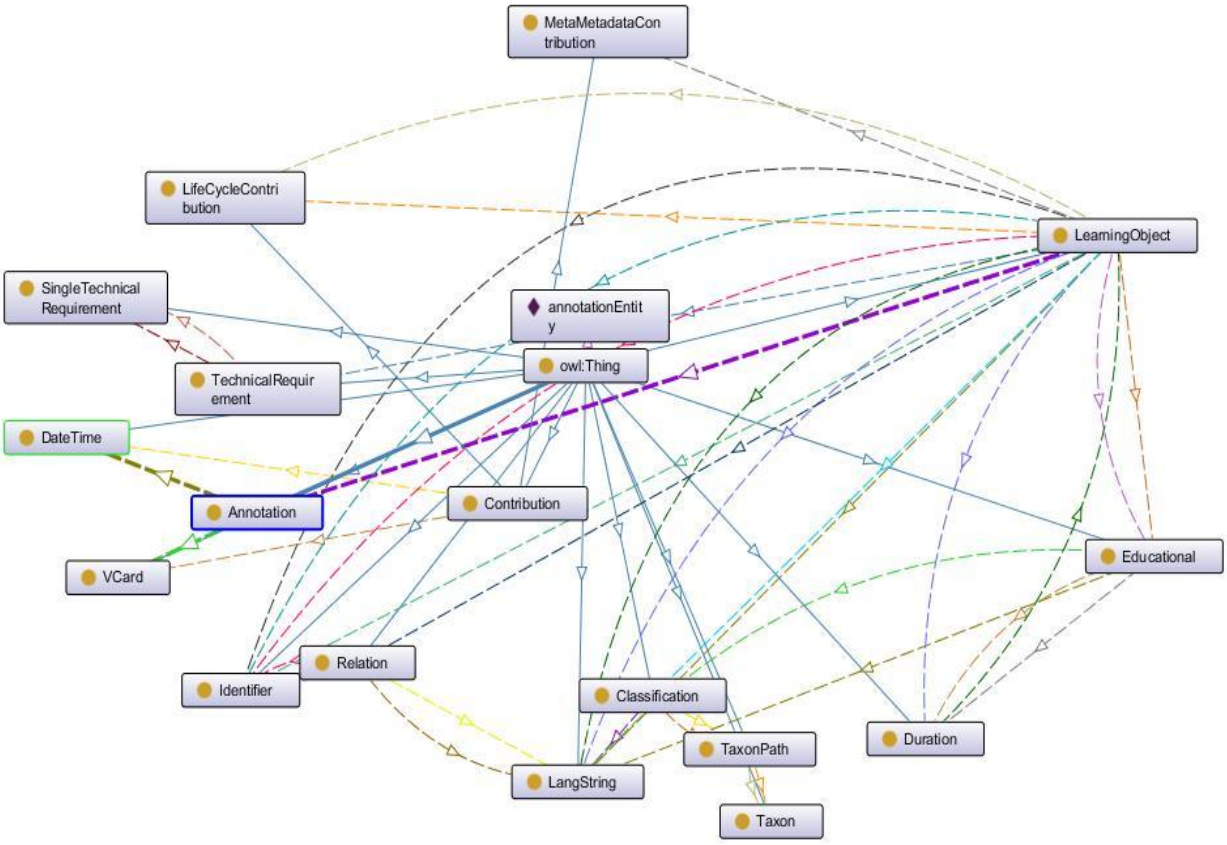


Рисунок 1.1 – Візуалізація графу онтології LOM додатком Protégé

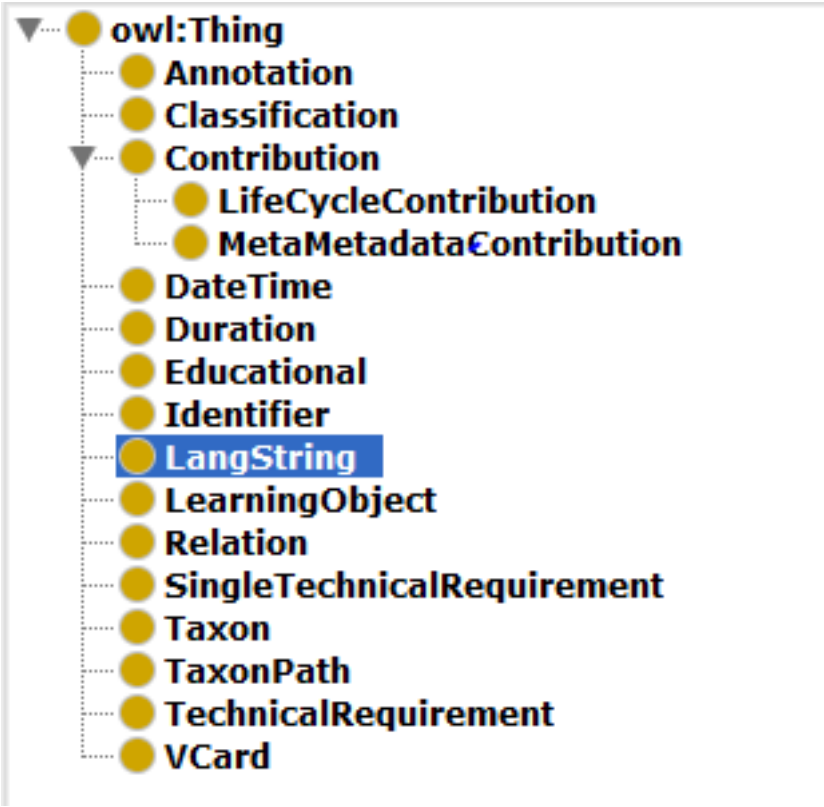


Рисунок 1.2 – Ієрархія класів онтології LOM

1.4.2 Онтологія Academic Institution Internal Structure Ontology (AIISO)

AIISO надає класи та властивості для опису внутрішньої структури академічної установи, граф представлений на рисунку 1.3 [6]. Дана онтологія надає класи, показані на рисунку 1.4, та властивості для опису курсів, модулів, практичних та теоретичних учбових матеріалів.

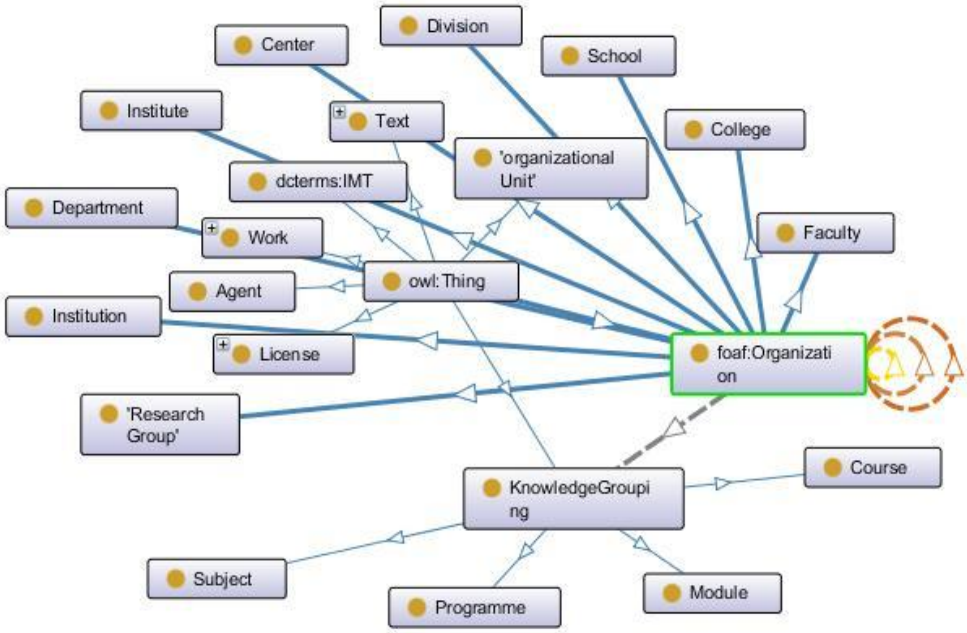


Рисунок 1.3 – Візуалізація графу онтології AIISO додатком Protégé

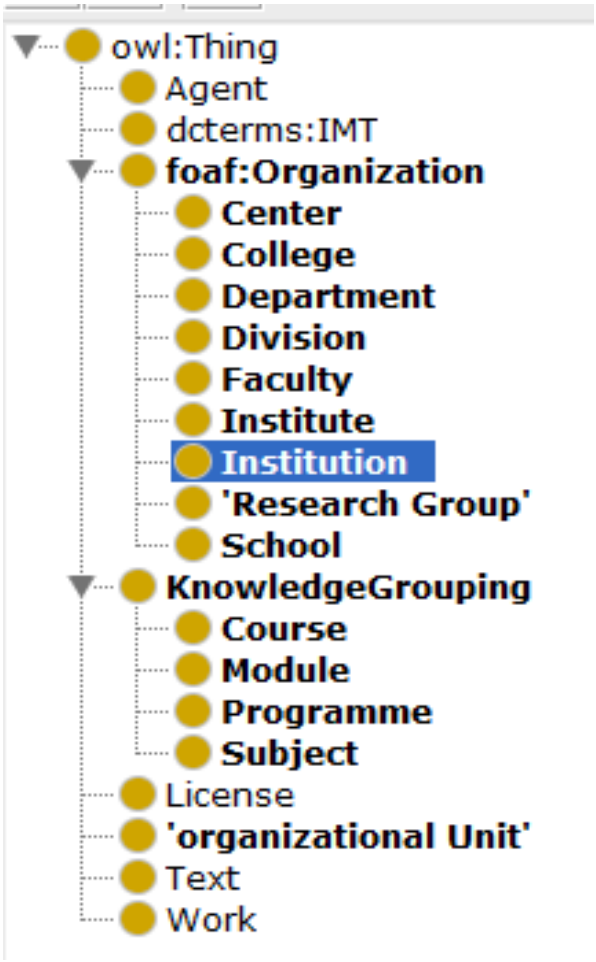


Рисунок 1.4 – Ієрархія класів онтології AIISO

1.4.3 Онтологія The Bibliographic Ontology

Бібліографічна онтологія (BIBO) описує бібліографічну інформацію на семантичній павутині в форматі RDF. Дана онтологія може використовуватися як антологія цитування, так і онтологія класифікації документів чи просто як спосіб опису будь-якого документа в RDF.

Ця специфікація служить документом простору імен «Бібліографічна онтологія». По суті, вона описує онтологію (класи та властивості) і терміни, з яких вона складається, щоб семантичні веб-додатки могли використовувати ці терміни у різних форматах та додатках, сумісних з RDF.

1.4.4 Онтологія LSC

Linked Science Core Vocabulary Specification – зв’язаний основний науковий словник призначений для опису наукових ресурсів, включаючи елементи дослідження, їх контекст та взаємозв’язок. Таким чином, LSC є прикладом «будівельних блоків» для опису зв’язків між дисциплінами зрозумілим для комп’ютера чином. LSC фокусується на простих властивостях, котрі можуть бути використані для опису змісту дослідницької роботи, тобто для співставлення досліджень, гіпотез, експериментів, даних та публікацій разом. LSC визначає тільки основні терміни для науки. Більш конкретні терміни, необхідні для різних наукових спільнот, можуть бути представлені як розширення LSC.

1.4.5 Онтологія LRMI

Learning Resource Metadata Initiative (LRMI) – ініціатива, що має за мету зробити легшим пошук освітніх матеріалів через пошукові та спеціалізовані сервіси викладачів та учнів. Підходом, що лежить в основі LRMI, є розширення онтології schema.org таким чином, щоб можна було виражати важливі освітні характеристики та відношення [7].

1.4.6 Онтологія TEACH

Teaching Core Vocabulary Specification (TEACH) – легкий словник основ викладання, котрий містить терміни, що дають змогу викладачам пов’язувати свої курси разом. Основний навчальний словник заснований на практичних вимогах, пред’явлених до проведення семінарів та опису курсів, як зв’язаних даних [9].

1.4.7. SemUNIT

Проект SemUnit був ініційований французькими вищими навчальними закладами. Мета проекту - отримати переваги Semantic Web та пов'язаних даних для покращення електронного навчання для широкого кола французьких вищих навчальних закладів. В рамках проекту було розроблено онтології OWL з

урахуванням семантики елементів LOM, а також архітектуру репозиторію для зберігання онтологій [10].

1.5 Висновки

У даному розділі було описано базові поняття онтологій у контексті семантичного веб та розкрито відповідну термінологію, описано алгоритм створення онтологій та проведено пошук та детальний огляд існуючих стандартів і специфікацій, присвячених формалізації процесів створення, зберігання та опису онтологій у домені навчальних дисциплін та пов’язаних з ними сферах. Результатом є розуміння того, що тема семантичного вебу та використання онтологій, як основного апарату представлення доменних знань є інноваційною та актуальною, що активно розвивається.

Наступний розділ має на меті огляд та аналіз методів побудови онтологій і представлення сильних і слабких сторін кожного з методів.

2 АНАЛІЗ ВЕЛИКИХ ДАНИХ ПРИ ПОБУДОВІ СЕМАНТИЧНИХ БАЗ ЗНАНЬ

У епоху аналітики великих даних, пошукові та рекомендаційні системи стали основними механізмами, за допомогою яких користувачі знаходять і виявляють корисну інформацію. Таким чином, є важливим, щоб ці системи обробки даних могли надавати цільові, релевантні результати, які повністю відповідають намірам користувача.

Пошукові та рекомендаційні двигуни рідко можуть конкурувати якщо вони не використовують моделі, що містять у собі детальну інформацію про типи заданих питань і, більш важливо, типи відповідей. Одним з найбільш типових шляхів представлення доменної області для наочної демонстрації цих ідей є використання онтологій - комбінацій таксономій, що містять попередньо відомі сутності, їх властивості та взаємозв'яки. Такі онтології можуть бути інтегровані у пошуковий додаток для того, щоб покращити його спроможність надавати кінцевому користувачу саме ті дані, які він очікує отримати. Наприклад, якщо хтось шукає термін "сервер" у домені інформаційних технологій, цей термін має дуже різні поняття (комп'ютерний сервер), аніж у ресторані (той, хто серверує, офіціант) і, якщо хтось використовую двигун для пошуку роботи, то запит може насправді представляти інше значення, в залежності від контексту користувача [12].

Онтології зазвичай будуються вручну експертами, що робить їх створення, підтримку та оновлення витратним рішенням. Щоб побороти це, системи навчання онтологій, котрі намагаються автоматизовано навчитися зв'язкам з доменної області та створити проекцію на онтологію, стають більш розповсюдженими.

Далі будуть описані методи та алгоритми, які використовують потужність великих аналітичних даних і розподілених обчислень для автоматичного створення

мовно-незалежних семантичних баз знань. Такі семантичні бази знань дозволяють значно покращити узгодженість між запитами та документами, і, як наслідок, видавати набагато більш релевантні результати для будь-якого запиту на пошук або рекомендацію. Буде розглянуто деякі основні технології, що дозволяють побудувати таку систему (Apache Lucene / Solr та Apache Hadoop), і деякі практичні деталі того, як така семантична пошукова система була побудована та використовується в реальній реалізації [13].

2.1 Пошукові двигуни

Пошукові системи є одним із найпоширеніших способів взаємодії людей з цифровою інформацією, і вони можуть отримати набагато більше користі від інтеграції з семантичними базами знань, які покращують загальну спроможність пошукової системи точно інтерпретувати та відповідати на запити. Основні структури пошукової системи, як буде видно пізніше, також ідеально підходять для автогенерації і моделювання тих самих семантичних баз знань. В області пошуку інформації пошукові системи є інструментом вибору для здійснення adhoc запитів вільної текстової інформації (як правило, ключових слів) у великій кількості існуючих документів (до трильйонів документів), одночасно класифікуючи та сортуючи результати за їх релевантністю до вхідного запиту. Найчастіше пошукові машини виконують всю цю роботу за мілісекунди або не більше кількох секунд.

Ця можливість шукати будь-яку комбінацію ключових слів у трильйонах документів і оцінювати релевантність всіх результатів запиту за часом відповіді порядку секунди вимагає вузькоспеціалізовних структур даних та підходів до моделювання даних, що працюють паралельно в розподіленій системі . Головним з них є інвертований індекс, відокремлення та реплікація даних, денормалізована модель даних, а також розподілена модель агрегації та оцінки.

2.2. Інвертований індекс

Інвертований індекс - це механізм швидкого пошуку за ключовими словами. Хоча основна реалізація інвертованого індексу може бути дуже складною задля оптимізації швидкості пошуку та максимальної компресії даних, щоб збільшити можливу кількість даних у пам'яті, основна ідея є дуже простою. Для створення інвертованого індексу може бути корисним (хоча і не обов'язково) спочатку створити прямий індекс, який співставляє документ та список термінів, що є у ньому. Це корисно для пошуку по документу, щоб побачити, які слова він містить, але це менш корисно, якщо ви намагаєтеся знайти, які документи відповідають певному набору ключових слів, тому що доведеться проходитися в циклі по списку слів для кожного документа для визначення чи міститься будь-яке з запитуваних ключових слів в документі.

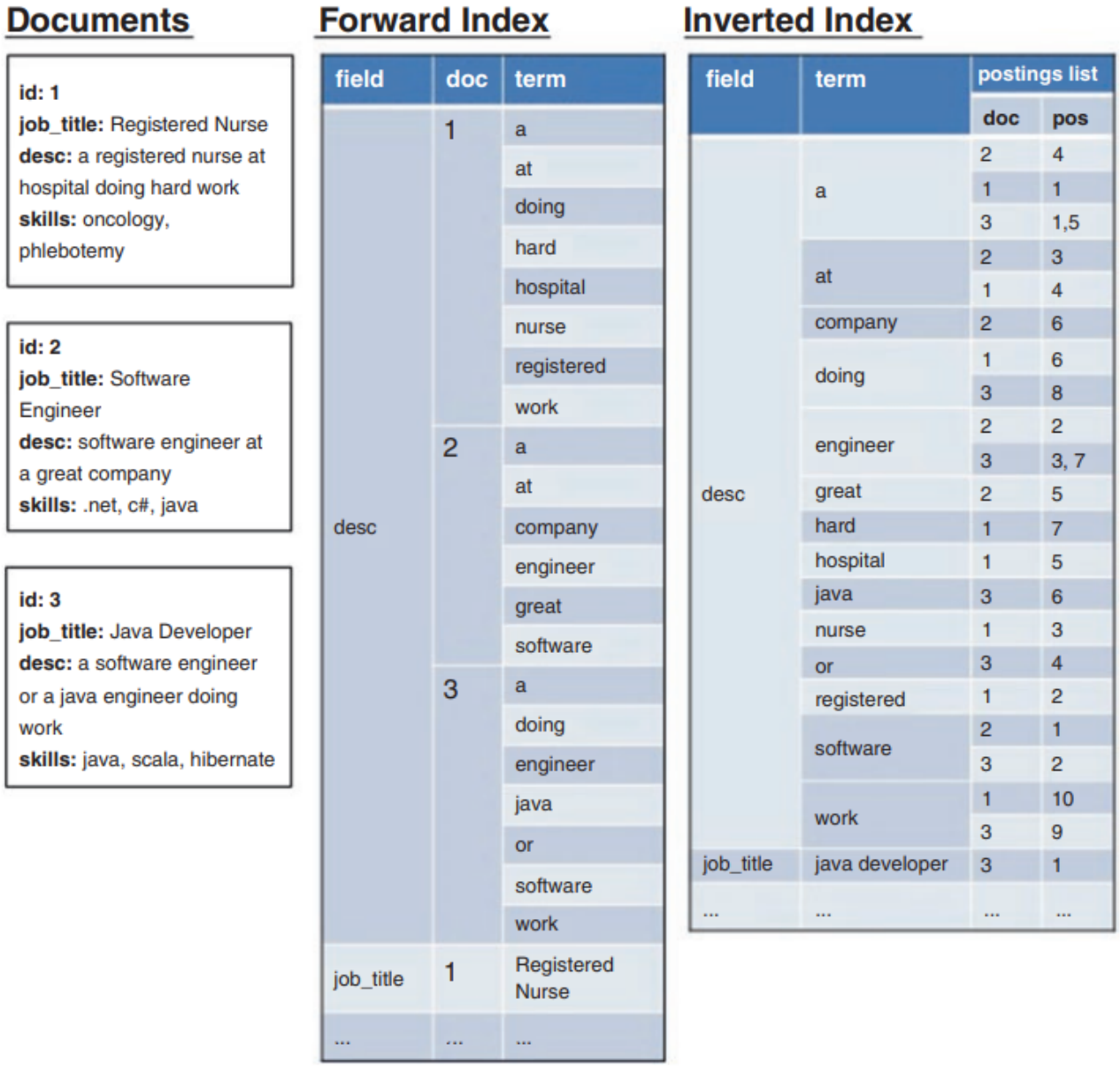
Замість цього пошукові системи покладаються на інвертовану версію цього індексу, де кожне ключове слово відповідає набору документів, які містять його, що робить складність пошуку *O(log n)* для будь-якого ключового слова. Приклад того, як набір документів буде представлений як у прямому індексі, так і в інвертованому індексі, можна знайти на рис. 1.

Єдине, що пропущено у спрощеному варіанті індексу, це те, що додаються ще і позиції терміну у документі, таким чином ці позиції (часто разом з метаданими) зберігаються разом з ідентифікатором документа.

Кожного разу, коли запит виконується по інвертованому індексу, пошук здійснюється за інвертованим індексом для кожного терміну в запиті. Також можуть бути використані операції на множинах, що відповідають кожному терміну, для швидкого вирішення будь-яких складних логічних запитів (наприклад, медсестра І лікарня, Java АБО Scala). Фразові запити (наприклад, "коричнева лисиця") можна знайти, використовуючи позиції в posting-списку, щоб відфільтрувати документи, де всі терміни з’являються в послідовному порядку.

2.3 Шардінг даних

Однією з додаткових корисних властивостей інвертованого індексу є те, що зі збільшенням кількості індексованих документів зростає ймовірність використання термінів документів, вже існуючих в індексі. Це означає, що для великих наборів даних кількість термінів в індексі буде стабільною, а кількість документів може продовжувати збільшуватися, оскільки кількість термінів в основному залежить від розподілу термінів в основній мові(мовах) документів.



Documents

id: 1
job_title: Registered Nurse
desc: a registered nurse at hospital doing hard work
skills: oncology, phlebotemy

id: 2
job_title: Software Engineer
desc: software engineer at a great company
skills: .net, c#, java

id: 3
job_title: Java Developer
desc: a software engineer or a java engineer doing work
skills: java, scala, hibernate

Forward Index

field	doc	term
desc	1	a
		at
		doing
		hard
		hospital
		nurse
		registered
		work
	2	a
		at
		company
		engineer
		great
		software
	3	a
		doing
		engineer
		java
		or
		software
		work
job_title	1	Registered Nurse
...	...	...

Inverted Index

field	term	postings list	
		doc	pos
desc	a	2	4
		1	1
		3	1,5
	at	2	3
		1	4
	company	2	6
	doing	1	6
		3	8
	engineer	2	2
		3	3, 7
	great	2	5
	hard	1	7
	hospital	1	5
	java	3	6
	nurse	1	3
	or	3	4
	registered	1	2
	software	2	1
		3	2
	work	1	10
		3	9
job_title	java developer	3	1
...	...	...	...

Рисунок 2.1 - Співставлення документів у прямий та інвертований індекси

Це дає можливість легко розділити інвертований індекс на декілька підіндексів і потім розподіляти запити до кожного з них паралельно і просто агрегувати отримані документи. Таке розбиття індексу часто називається шардінгом (відокремленням) індексу.

Паралельний пошук і агрегація через шарди можна здійснювати на кількох мережевих комп'ютерах, що дозволяє пошуковим системам здійснювати пошук у мільярдах або навіть трильйонах документів за кілька секунд.

2.4 Реплікація даних

Так само, як шардінг, реплікація даних дозволяє збільшити швидкість запитів по великій кількості документів і масштабуватися на декілька серверів, часто пошуковий механізм має потребу в обробці великої кількості запитів за раз. Коли перевищено можливості обробки вхідних запитів одного вузла з шардом, використовуються репліки (копії) цього вузла з шардом, щоб забезпечити можливість балансування навантаження великої кількості пошукових запитів.

Однією додатковою перевагою реплік є те, що вони можуть бути використані для забезпечення відмовостійкості в пошуковому кластері. Оскільки час від часу сервери будуть виходити з ладу, якщо на окремому сервері існує щонайменше одна копія копії кожного шарду, то кластер пошуку може продовжувати успішно відповідати на запити без втрати даних.

2.5 Денормалізована модель даних

Можливість шардінгу інвертованого індексу, створення реплік цих шардів та розподілених запитів та індексування через кластер серверів дає змогу пошуковим системам масштабуватися практично в будь-якому напрямку (зменшити час відгуку, збільшити об’єм даних, збільшити кількість запитів). Треба дотримуватися критичного правила моделювання даних, щоб забезпечити це розпаралелювання, однак, дотримуючись правил використання денормалізованої моделі даних. У

традиційних системах керування реляційними базами даних (RDBMS) кращою практикою є нормалізація таблиць і поєднання шляхом зовнішніх ключів як зв'язків між кількома таблицями, щоб запобігти появі збиткових даних та неконсистентності бази даних. Хоча це добре працює в теорії, немає можливості легко відокремити індекс у шард, оскільки вимога join-у по окремих індексах означає, що треба мати ці індекси повністю на кожному сервері, щоб можна було ефективно виконувати об'єднання. Хоча деякі сучасні пошукові системи (наприклад, Apache Solr) підтримують функцію join-у, вона повинна використовуватися дуже обережно, щоб зберегти масштабованість пошукового кластера, а також консистентність даних.

2.6 Розподілена агрегація та оцінювання

Однією з найважливіших особливостей пошукової системи є можливість оцінити релевантність кожного документа до відповідного запиту та повертати всі відповідні документи в відсортованому порядку. Цей сортований порядок зазвичай є розрахунком релевантності, але сортування також може базуватися на значенні будь-якого іншого поля або функції.

Для того, щоб розподілена пошукова система була настільки ефективною, наскільки це можливо, вона повинна максимально збільшити роботу, яка проводиться паралельно на кожному шарді, в той же час мінімізувати кількість мережевих запитів та обсяг переданих даних.

Хоча кожна пошукова система підраховує результати релевантності різним способом, більшість використовує статистичні дані, одержані від структури інвертованого індексу. Розрахунки, що використовують значення tf-idf (частота терміну * частота інвертованого документу), такі як популярний алгоритм підрахунку BM25, враховують кількість разів, коли кожен термін у запиті з'являється у кожному документі (tf), помножену на те, наскільки важливим є це слово у запиті (idf). TF - це відношення числа входжень обраного слова до загальної кількості слів документа,

IDF - інверсія частоти, з якою слово зустрічається в документах колекції. Використання IDF зменшує вагу широковживаних слів.

Таким чином, для того, щоб повернути остаточний список результатів рейтингу по релевантності, принаймні один запит потрібно розподілити паралельно кожному окремому індексу, кожен шард повинен потім самостійно шукати набір документів, що відповідають кожному ключовому слову, і знаходити відповідні перетини множин, основані на запиті, отримані документи потім повинні бути відсортовані, використовуючи оцінку релевантності, розраховану з статистики шарду, доступної в інвертованому індексі. Потім, набір результатів, достатньо великий для того, щоб відповідати запитуваній кількості документів, повинен бути повернутий до агрегувального вузла кластера. Тоді агрегувальний вузол просто потребує повторного сортування повернутих документів з кожного з розподілених шардів, і повернення конкретної кількості документів кінцевому користувачеві. У більшості пошукових систем існує безліч додаткових функцій, таких як аналітика, підсвічування та виправлення орфографії, які можуть потребувати включення додаткових кроків до цього робочого процесу, але, по суті, всі вони працюють паралельним способом через шарди, щоб зробити можливим виконання запитів по мільярдам або трильйонам документів.

Останньою особливо важливою характеристикою пошукових систем є те, що, оскільки користувачі роблять запити і отримують результати, взаємодіють з цими результатами. Користувачі натискають, пропускають або створюють додаткові запити для виправлення орфографії чи додають нові пов'язані ключові слова, щоб побачити, чи можна отримати кращий результат. У подальших розділах буде описано виконання аналізу великих даних журналу таких користувацьких виправлень та уточнень як ключову техніку для автоматичного створення семантичних баз знань.

2.7 Рекомендаційні системи

Рекомендаційні системи автоматизують процес виявлення інтересів користувача шляхом застосування технік дата майнінгу для прогнозування об'єктів, що представляють інтерес для окремих користувачів, а потім пропонують те, що має відповідати його потребам [14, 15]. Протягом багатьох років методи та застосування рекомендаційних систем розвивалися як в академічному середовищі, так і в індустрії (електронна комерція / електронний шопінг, електронна бібліотека, електронне навчання, електронний туризм тощо) через експоненційне збільшення обсягу даних. Рекомендаційні системи можуть бути поділені на три основні категорії: контентні, спільна фільтрація та гібридні технології [15, 16]. Контентні є найбільш чутливими до розуміння текстового змісту, оскільки ці системи засновані на матчінгу елементів/користувачів на основі подібності між їх текстовим описом [17]. Таким чином, наявність семантичної бази знань має вирішальне значення для підвищення ефективності контентних рекомендаційних систем [20].

Хоча рекомендаційні системи часто будуються як самостійні системи, які можуть матчити контент відповідно до інтересів користувачів, вони дуже перетинаються з функціональними можливостями пошукової системи. Прийнято думати, що пошукові системи, як правило, приймають запит і повертають результати, що відповідають цьому запиту, але пошукові системи також можуть використовувати інформацію про користувачів та їхні уподобання для персоналізації результатів пошуку. Аналогічним чином, у рекомендаційних системах дуже корисно мати можливість коригувати матчінг рекомендацій у режимі реального часу та мати можливість виконувати матчінг на основі довільного змісту та властивостей, що є завданням, яке надзвичайно добре виконують пошукові системи.

2.8 Семантичне виявлення

Побудова семантичних баз знань традиційно зосереджена на використанні онтологій/таксономій, які вручну створюються та підтримуються, або використовують кластеризацію та зменшення розмірності для виявлення семантичних зв'язків між термінами даного корпусу. Побудова онтологій/таксономій вручну не масштабується, важко підтримується і є дуже витратною роботою. З іншого боку, зменшені розмірності схильні до шуму, і не дуже зрозумілі людині. Натомість можна спиратися на журнали пошуку, які є гарним джерелом для виявлення семантичних зв'язків між фразами. У цьому розділі описано, як використовувати розподілену аналітику великих даних для майнінгу пошукових журналів, щоб виявити семантичні зв'язки між ключовими фразами, незалежно від мови. Конкретна реалізація даної техніки буде представлена в контексті пошукової системи для працевлаштування на англійській мові, але ця техніка є одночасно доменно- та мовно-незалежною.

2.9 Опис проблеми

Щоб краще зрозуміти проблему, подумаємо про різні значення слова *архітектор* в контексті архітектури будівлі та архітектора програмного забезпечення. Якщо хтось вводить слово *архітектор* у формі пошуку, пошукова система на основі ключових слів поверне змішаний набір документів, деякі з них стосуються архітектури програмного забезпечення, проте інші стосуються архітекторів будівель. Такі змішані результати збивають користувача, який майже напевно шукає лише одне з двох значень. Навіть якщо користувач шукає архітектора будівель, типовий пошуковий двигун трансформує запит у булевий запит *будівельний І ахітектор* як незалежні терміни, що може спричинити вибір документів, у котрих розповідається про когось, хто *архітектор програмного забезпечення*, котрий *будує* додатки. Розробка більш розумних пошукових систем для подолання таких проблем - це те, що буде обговорено в іншому розділі. Маючи доступ до історії пошуку тисяч або

мільйонів користувачів, можна виявити взаємозв'язок між пошуковими фразами та найпоширенішим значенням кожного терміна. Такі семантичні знання можуть потім бути використані для кращого розуміння намірів користувача.

2.10 Семантична схожість

Семантична схожість є мірою схожості сенсу двох термінів [21]. Два основних підходи, використовувані для обчислення семантичної подібності, - це семантичні мережі (підхід, оснований на знаннях), а також обчислення зв'язності термінів у великому корпусі тексту (структурний підхід) [22, 23]. Основні методи - поточкова взаємна інформація (Point-wise Mutual Information, PMI) та латентний семантичний аналіз (Latent Semantic Analysis, LSA).

Дослідження показують, що PMI зазвичай перевершує LSA на видобувних синонімах в Інтернеті [26]. Ще одна цікава методика виявлення семантичних зв'язків між словами запропоновані дослідниками Google. Дві нові моделі, запропоновані компанією Google, наступні [27]:

1. Continuous Bag-of-Words model (CBOW)

2. Continuous Skip-gram model (SG)

Ці моделі використовують глибинні нейронні мережі для вивчення словесних векторів. Проте, ці дві моделі не підходять для використання у нашому випадку через кілька обмежень. По-перше, це відсутність контексту в нашому наборі даних, який складається з запитів, які зазвичай містять лише 1-3 ключові слова. CBOW і SG не працюють добре без контексту, що робить наш випадок складним. Інше обмеження полягає в тому, що ці моделі найкраще підходять для юні-грам або одиночних токенів, а не фраз, тоді як користувачами частіше вводяться саме фрази. Наприклад, "*Java Developer*" слід розглядати як одну фразу, коли ми знаходимо інші семантично пов'язані фрази. У експерименті виявлено високоякісні семантичні відношення, використовуючи набір даних з 1,6 мільярдів записів журналів пошуку (складається з

ключових слів, які використовуються для пошуку завдань на careerbuilder.com). Для цього було використано імовірнісні графові моделі для масивних ієрархічних даних (Probabilistic Graphical Model for Massive Hierarchical Data, PGMHD), які були впроваджені через відомі розподілені обчислювальні системи Apache Hadoop [28].

2.11 Імовірнісна оцінка семантичної схожості з використанням PGMHD

PGMHD потребує збору пошукових термінів, введених користувачами для проведення пошуку, а також класифікації кожного користувача. Спосіб подання цих даних для розрахунку оцінки смислової схожості, заснований на імовірнісних показниках, полягає в тому, щоб помістити класи, до яких належать користувачі, у верхньому шарі моделі, розмістити пошукові терміни в нижньому шарі моделі, а потім з'єднати їх ребрами, які вказують, скільки користувачів з даного класу у верхньому шарі шукали певний термін у нижньому шарі. У таблиці 1 показані вхідні дані, а на рис.2 показано подання вихідних даних в PGMHD.

Таблиця 2.1 - Вхідні дані PGDMH для системи Hadoop

Користувач1	Java Розробник	Java, Java Розробник, C#, Software Engineer
Користувач2	Медсестра	Медсестра, Ліцензована медсестра, Охорона здоров'я
Користувач3	.NET Розробник	C#, ASP, VB, Software Engineer, SE
Користувач4	Java Розробник	Java, JEE, Struts, Software Engineer, SE
Користувач5	Охорона здоров'я	Охорона здоров'я, HealthCare

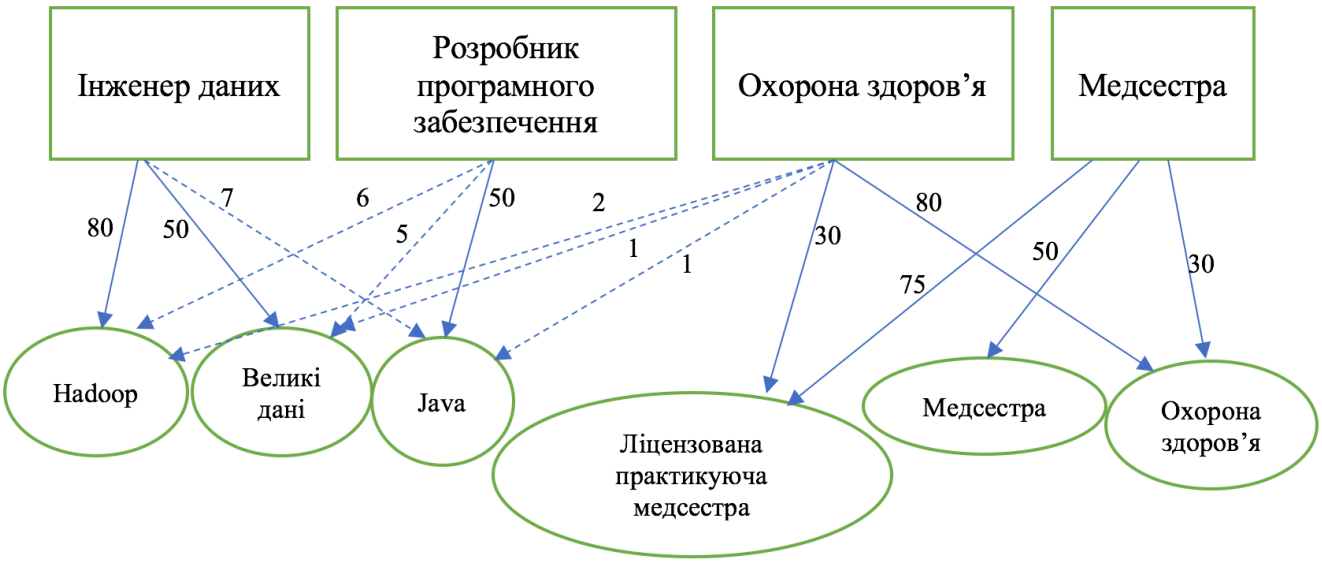


Рисунок 2.2 - Використання PGDMH для представлення журналів пошуку працевлаштування

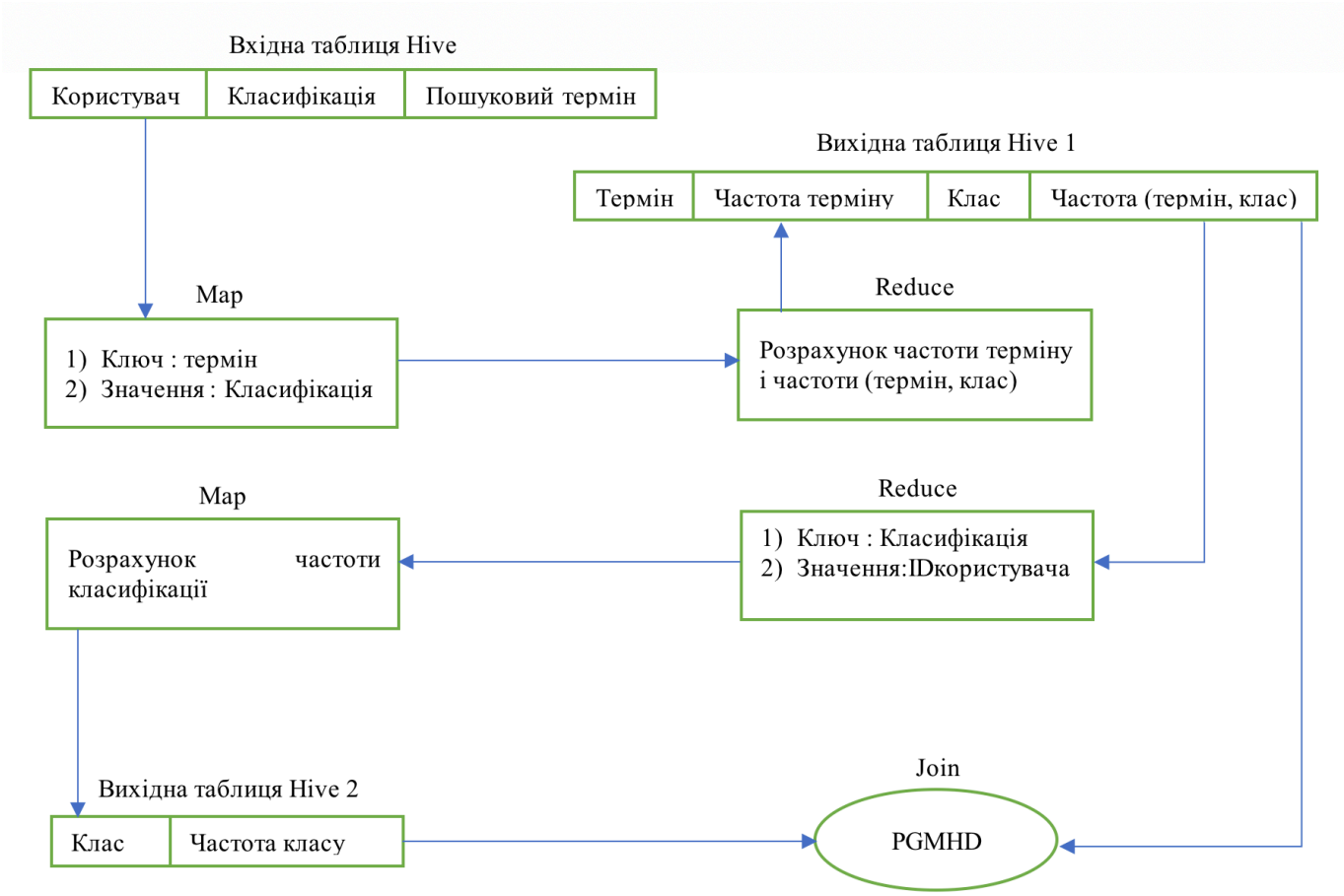


Рисунок 2.3 - Реалізація PGDMH як Map/Reduce з використанням Apache Hadoop

2.11.1 Визначення неоднозначності сенсу слів

Ми можемо використовувати виявлені семантично пов'язані терміни, щоб покращити розуміння запитів. Зробити це можна шляхом розширення запиту, шляхом включення семантично пов'язаних термінів, які допоможуть пошуковій системі отримати більш релевантні результати, оскільки наявність запиту та/або кількість його семантично пов'язаних термінів у документі буде підвищуватись. Наприклад, запит "*великі дані*" може бути розширений до "*великі дані*" АБО *hadoop* АБО *spark* АБО *hive*. Як і слід було очікувати, результати розширеного запиту, як правило, будуть більш релевантними та всеосяжними.

Однак, як передбачалося, ця методика не буде працювати при роботі з термінами, які можуть представляти істотно різні значення (неоднозначні терміни). Неоднозначний термін - це термін, який відноситься до більш ніж одного значення, залежно від контексту. Наприклад, термін *java* може посилатися на мову програмування Java, або на тип кави під назвою java, або на острів в Індонезії, що називається Java. Оскільки користувач, який виконує пошуковий запит, найімовірніше, буде шукати тільки одне значення терміна, важливо, щоб ми могли ідентифікувати можливі сенси слова. Щоб виявити ці неоднозначні терміни, ми знову використовуємо PGMHD, де ми розраховуємо оцінку класифікації для кожного терміну з його базовими класами як потенційні класи. Якщо рейтинг класифікації вище, ніж певне порогове значення для більш ніж одного базового класу, ми вважаємо, що цей термін може бути неоднозначним. Ідея цієї технології полягає в тому, що кожен батьківський клас у PGMHD представляє групу користувачів з різних классифікацій, тому, коли термін може бути класифікований з високим показником в більш ніж одному класі, це означає, що він широко використовувався користувачами з обох класів. Крім того, якщо сукупність інших термінів, що використовуються разом із терміном, значно відрізняється між різними класами, це також означає, що термін відноситься до двох або більше різних концепцій. Методика виявлення неоднозначних термінів пояснюється нижче:

Нехай:

C: = {C1, ..., Cn} множина різних класів робочих посад (Java Developer, Nurse, Accountant, etc);

S: = {t1, ..., tN} множина різних пошукових термінів, введених користувачами, коли вони робили пошук (N - кількість різних термінів);

F(*Cj, s*) кількість (частота), коли користувач з класу *Cj* $\in C$ шукав ключове слово *s* $\in S$.

- Для зменшення шуму будуть розглядатися тільки частоти з хоча б 100 різними пошуками $f(c,s) \geq 100$.

Тоді, визначимо *O(c):* кількість разів, коли користувач з класу *c* шукав ключове слово

$$O(c): \sum_{s \in \mathcal{S}} f(c,s) \quad c \in \mathcal{C};$$

T(s): кількість пошуків ключового слова *tj*:

$$T(s): \sum_{c \in \mathcal{C}} f(c,s) \quad s \in \mathcal{S};$$

T: загальна кількість пошуку ключового слова:

$$T: \sum_{c,s} f(c,s) = \sum_{c \in \mathcal{C}} O(c) = \sum_{s \in \mathcal{S}} T(s).$$

Для кожного $c \in$ C і $s \in$ S при випадкових змінних C, S, що представляють клас роботи та пошуковий термін одного запиту користувача, відповідно, можна оцінити їх PMI за формулою:

$$\frac{\mathbb{P}(C=c.S=s)}{\mathbb{P}(C=c)\mathbb{P}(S=s)}=\frac{\mathbb{P}(C=c|S=s)}{\mathbb{P}(C=c)}$$

тоді,

$$\mathrm{pmi}(c,s):\log\frac{f(c,s)}{T(s)}\frac{T}{O(c)}\quad c\in\mathcal{C},s\in\mathcal{S}.$$

Нормалізована версія оригінальної оцінки PMI задана формулою:

$$\tilde{p}(c,s):\frac{\mathrm{pmi}(c,s)}{-\log\frac{f(c,s)}{T}}=\frac{\log T+\log f(c,s)-\log\left[O(c)T(s)\right]}{\log T-\log f(c,s)}$$

$$=-1+\frac{2\log T-\log O(c)-\log T(s)}{\log T-\log f(c,s)}\in[-1,1]c\in\mathcal{C},s\in\mathcal{S}.$$

Ця нормалізована версія оригінального PMI може бути застосована для генерації оцінки неоднозначності для визначення, чи має термін розглядатися як неоднозначний.

Оцінка неоднозначності

Для кожного пошукового ключового слова $s \in S$, визначимо наступну *оцінку неоднозначності* $A\alpha(s)$ як

$$A_\alpha(s):\left|\{i:\tilde{p}(c,s)>0\}\right|,$$

і скажемо, що пошукове слово *tj* є *кандидатом* на те, щоб бути неоднозначним, якщо A*j*(α) > 1. Потім, можна визначити множину термінів, що є кандидатами на неоднозначність *CA* як

$$CA = \left\{ t_j : A_j(\alpha) > 1, j = 1,\ldots,N \right\}.$$

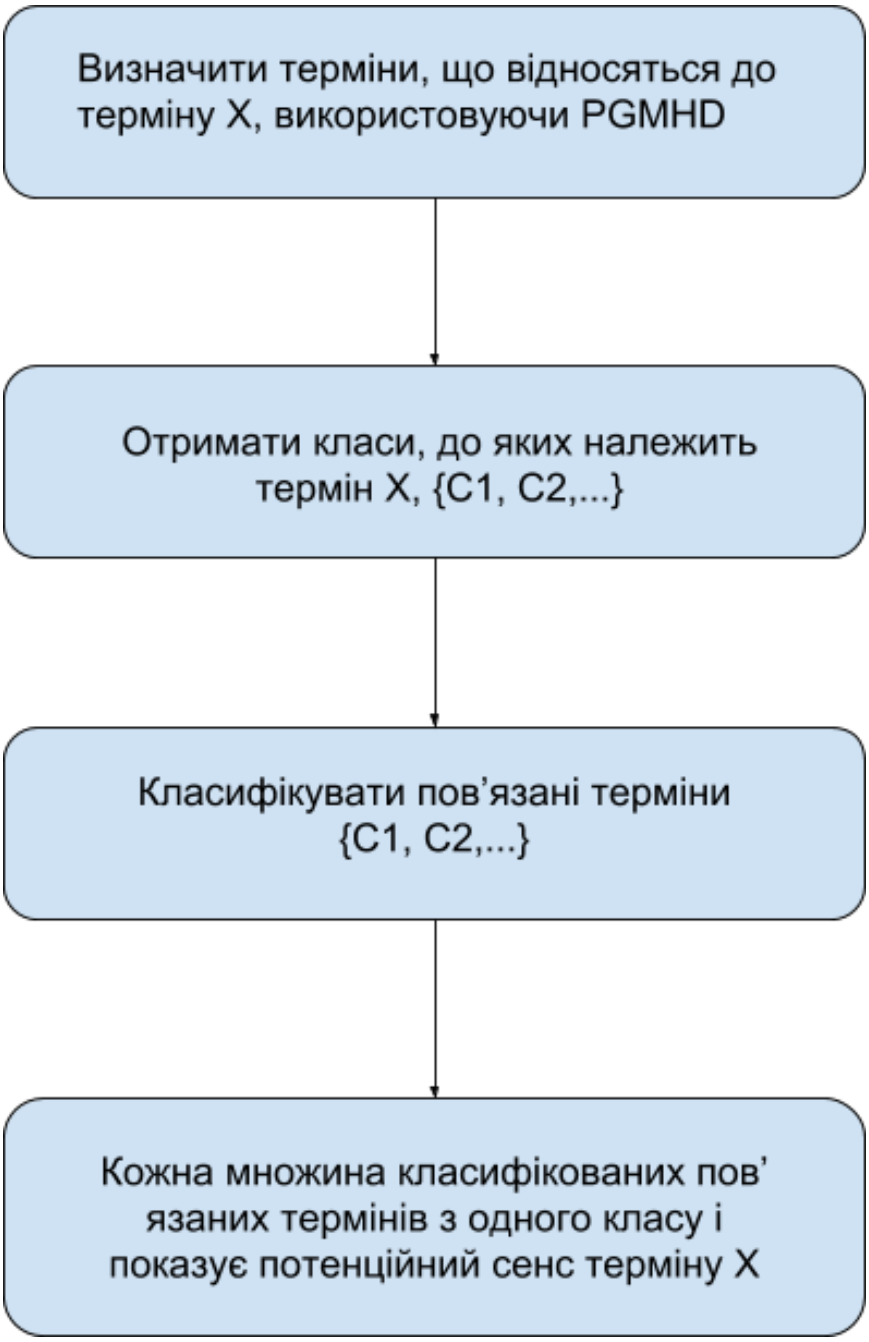


Рисунок 2.4 - Система для вирішення проблеми неоднозначності за допомогою PGMHD

2.11.2 Вирішення проблеми неоднозначності сенсу слова

Після виявлення неоднозначності термінів наступна задача постає в тому, як вирішити цю неоднозначність. Вирішення невизначеності означає визначення можливих значень неоднозначного терміну. У запропонованій системі використовується семантично пов'язані терміни, які виявлено, використовуючи раніше описаний модуль семантичного виявлення. Кожна група цих семантично пов'язаних термінів являє собою можливе значення первинного терміну з урахуванням контексту, в якому терміни були використані. Наприклад, неоднозначний термін *driver* має семантично пов'язані терміни *transportation, truck driver, software, embedded system,* та *CDL*. Класифікуючи ці терміни за допомогою класів користувачів, які згадували їх в журналах пошуку, ми в кінцевому підсумку класифікуємо їх у дві групи: "*transportation, truck driver*, *CDL*" та "*software*, *embedded system*".

Зрозуміло, що кожна з цих груп семантично пов'язаних термінів являє собою окремий можливий сенс слова “*driver*”, при чому перша група представляє сенс “*transportation*”, а друга - сенс “*computer device driver*”.

На рисунку 2.4 показана методологія для вирішення неоднозначності. Оскільки ми вже створили PGMHD для виявлення неоднозначних термінів, ми можемо використовувати ту саму модель, щоб знайти семантично пов'язані терміни для будь-якого заданого терміна, що належить до одного класу. Для цього ми обчислимо оцінку подібності, засновану на імовірнісних показниках, між даним терміном X та терміном Y, якщо вони обидва поділяють одні і ті ж базові клас(-и) наступним чином:

Нехай для рівня $i \in \{2,...,m\}$ є ідентично розподілені випадкові змінні X, Y $\in L2 \times \cdots \times Lm$. Визначимо ймовірнісну оцінку схожості CO між двома незалежними наслідниками одного рівня Xij, $Yig \in Li$ шляхом обчислення умовної спільної ймовірності:

$$\mathrm{CO}\left(X_{ij},Y_{ig}\right):P\left(X_{ij},|Y_{ig},|\mathrm{pa}\left(X_{ij}\right)\bigcap\mathrm{pa}\left(Y_{ig}\right)\right)$$

як

$$\mathrm{CO}\left(X_{ij},Y_{ig}\right)=\prod_{C_k^{'}\in\mathrm{pa}\left(X_{ij}\right)\bigcap\mathrm{pa}\left(Y_{ig}\right)}P\left(X_{ij}|C_k^{'}\right)P\left(Y_{ig}|C_k^{'}\right),$$

де

$$P\left(X_{ij}|C_k^{'}\right)=\frac{P\left(C_k^{'},X_{ij}\right)}{P\left(C_k^{'}\right)}$$

для кожного

$$\left(X_{ij},C_k^{'}\right)\in L_{i-1}\times L_i.$$

З визначеним $out\left(C_k^{'}\right)$, як загальна кількість входжень $out\left(C_k^{'}\right)$ і $f\left(C_k^{'},X_{ij}\right)$, як частоту входжень $out\left(C_k^{'}\right)$ разом з $f\left(C_k^{'},X_{ij}\right)$, можна оцінити спільні ймовірності $P\left(X_{ij},C_k^{'}\right)$ і $\hat{p}\left(X_{ij},C_k^{'}\right)$, визначені як

$$\hat{p}\left(X_{ij},C_k^{'}\right):\frac{f\left(C_k^{'},X_{ij}\right)}{out\left(C_k^{'}\right)}$$

Як наслідок, можна оцінити кореляцію між *Xij*, *Yig* шляхом визначення

$$\mathrm{CO}\left(X_{ij},Y_{ig}\right)=\prod_{C_k^{'}\in\mathrm{pa}\left(X_{ij}\right)\bigcap\mathrm{pa}\left(Y_{ig}\right)}P\left(X_{ij}|C_k^{'}\right)P\left(Y_{ig}|C_k^{'}\right)$$

імовірнісної оцінки схожості .

Після того, як список пов'язаних термінів буде згенерований за допомогою PGMHD, класифікуємо їх у класи (оскільки термін неоднозначний, вони повинні належати до декількох класів), до якого належить неоднозначний термін. Ця фаза класифікації відповідних термінів реалізується за допомогою PGMHD наступним чином:

Для випажкової змінної на рівні $i \in \{2,...m\}$, а саме $Xij \in Li$, де Xij - j-та випадкова змінна на рівні i, визначимо *оцінку класифікації* $Cl\left(C_k^{'} | X_{ij}\right)$ для Xij з його найближчим базовим класом $C_k^{'} \in L_{i-1}$. Він використовується для оцінки умовної ймовірності $P\left(C_k^{'} | X_{ij}\right)$. Нотація $C_k^{'} \in L_{i-1}$ використана для позначення базового класу, і, коли він на рівні 1, він буде представляти клас Cj. Нехай, $f\left(C_k^{'}, X_{ij}\right)$ - частота спільної появи $C_k^{'} \in L_{i-1}$ та $f\left(C_k^{'}, X_{ij}\right)$

$$\mathrm{Cl}\left(C_k^{'} | X_{ij}\right) : \frac{f\left(C_k^{'}, X_{ij}\right)}{in\left(X_{ij}\right)}$$

Класифікаційна оцінка - це співвідношення частоти спільної появи $C_k^{'} \in L_{i-1}$ та $f\left(C_k^{'}, X_{ij}\right)$, поділене на загальну кількість появ $f\left(C_k^{'}, X_{ij}\right)$. Загальна кількість появ $f\left(C_k^{'}, X_{ij}\right)$ розрахована шляхом сумування частот спільних появ $f\left(C_k^{'}, X_{ij}\right)$ з його батьківськими класами будь-якого рівня.

$$in\left(X_{ij}\right) : \sum_{C \in pa\left(X_{ij}\right)} f\left(C, X_{ij}\right), \qquad \forall X_{ij} \in V$$

Група семантично пов'язаних термінів, які класифікуються під тим самим батьківським класом, формує можливий сенс неоднозначного терміну. Використовуючи цю техніку, ми не обмежені фіксованою кількістю можливих значень: для деяких термінів передбачено два можливих значення, деякі отримують три можливі значення та інше.

2.12 Semantic Knowledge Graph

На додачу до майнінгу журналів запитів для автоматичного створення семантичних баз знань також можна використовувати взаємозв'язок між словами і фразами, закодованими як в тексті, так і в структурованому контенті множини документів.

У цьому увага буде фокусуватися на використанні великих даних з застосуванням масштабованих розподілених алгоритмів, мета - використовувати систему, яка може автоматично генерувати представлення області знань у вигляді графу, шляхом обробки вмісту даних, що представляють доменну область. Після того, як цей граф буде побудований, можна обійти його, щоб отримати взаємозв'язки між кожним з ключових слів, фраз, сутностей та інших лінгвістичних варіацій, представлених у вхідних даних. Ця модель називається семантичним графом знань, а реалізація з відкритим вихідним кодом також є загальнодоступною.

Інші системи навчання онтології зазвичай намагаються витягти певні сутності з колекції і створити заздалегідь сформований граф взяємозв’язків між сутностями. Це, на жаль, призводить до значної втрати інформації у випадках, коли значення терміна або фрази змінюється в залежності від його лінгвістичного контексту. Одна з цілей підходу Semantic Knowledge Graph полягає в тому, щоб повністю передбачити всі смислові взаємозв'язки, що містяться в текстовій збірці документів.

Щоб дійсно зрозуміти значення цієї мети, розглянемо, як сенс слів може варіюватися в залежності від контексту, в якому вони знайдені. Слова *архітектор* та *інженер* добре відомі, але, коли вони зустрічаються всередині фаз, таких як

архітектор програмного забезпечення або інженер-електрик, вони мають набагато більш обмежену інтерпретацію. Хоча передбачуване значення слів і фраз в різних контекстах матиме подібні властивості, семантичний граф знань здатний моделювати ці подібності, водночас зберігаючи кожне з контекстно-залежних значень. Таким чином, семантичний граф знань дозволяє краще представити всю галузь в компактному вигляді.

2.12.1 Структура моделі

Дано неорієнтований граф $G = (V,E)$ з V та $E \subset V \times V$ що позначають набори вузлів та ребер, відповідно. Дано наступні визначення:

$D = \{d_1, d_2, ..., d_m\}$ це набір документів, який представляє собою колекцію, яку ми будемо використовувати для визначення та оцінки семантичних відношень в графі семантичних знань.

$X = \{x_1, x_2, ..., x_k\}$ являє собою набір всіх елементів, які зберігаються в D. Ці елементи можуть бути термінами, фразами або навіть будь-якими довільними лінгвістичними конструкціями, які можна знайти в межах D.

$d_i = \{x|x \in X\}$ де кожен документ $d \in D$ це набір елементів.

$T = \{t_1, t_2, ..., t_n\}$ де t_i це тег, який ідентифікує тип сутності для конкретного елемента. Приклади тегів можуть включати ключове слово, місце розташування, школу, компанію, особу тощо.

З урахуванням цих визначень, набір вузлів V в графі визначається як $V = \{v1, v2, ..., vn\}$, де vi представляє елемент $xi \in X$ який позначено тегом $tj \in T$, у той час як $Dvi = \{d \mid xi \in d, d \in D\}$ - це набір документів, що містять елемент xi з відповідним тегом tj. Потім ми визначимо eij як ребро між (vi, vj) за функцією $ч$, що представляє кожне ребро з набором документів, що містять обидва елементи xi і та xj, кожен з відповідними тегами. Нарешті, визначаємо функцію $g(eij, vk) = \{d: d \in f(eij) \cap Dvk\}$ що містить загальний набір документів між $f(eij)$ та Dk на кожному ребрі ejk.

2.12.2 Матеріалізація вузлів та ребер

Модель SKG відрізняється від більшості традиційних графових структур, так як використовує шар неорієнтованості між будь-якими двома вузлами і ребром, яке їх з'єднує. Зокрема, замість двох вузлів v_i та v_j будучи безпосередньо пов'язаними один з одним через явне ребро e_{ij}, вузли замість них з'єднуються через документи, такі, що ребру e_{ij} між вузлом v_i та v_j, тобто *матеріалізуються*, коли $|f(e_{ij})| > 0$.

Для того, щоб перейти від вихідного вузла v_i на інший вузол v_j, треба мати індекс пошуку (інвертований індекс) що встановлює відповідність вузла v_i до базового набору документів, а також інший індекс пошуку (прямий індекс) який вміє ставити ці документи у відповідність до будь-якого іншого вузла v_j, з яким ці документи також пов’язані. Ця комбінація інвертованого індексу і прямого індексу дозволяє відображати всі терміни або комбінації термінів як вузли в графі, дозволяючи обхід між будь-якими двома вузлами через набір спільно використовуваних документів між ними, як показано на рисунку 2.5.

Оскільки ребра побудовані на перетині множин документів, до яких прив'язані обидва вузла, це означає, що нове ребро також може бути згенеровано «на льоту» між будь-якою довільною комбінацією інших вузлів. Будемо називати динамічну генерацію ребер *матеріалізацією ребер*. Крім того, оскільки обидва вузла і ребра повністю засновані на заданих перетинах документів, це означає, що також можна динамічно матеріалізувати нові вузли на основі довільних комбінацій інших вузлів, як показано на рисунку 2.6.

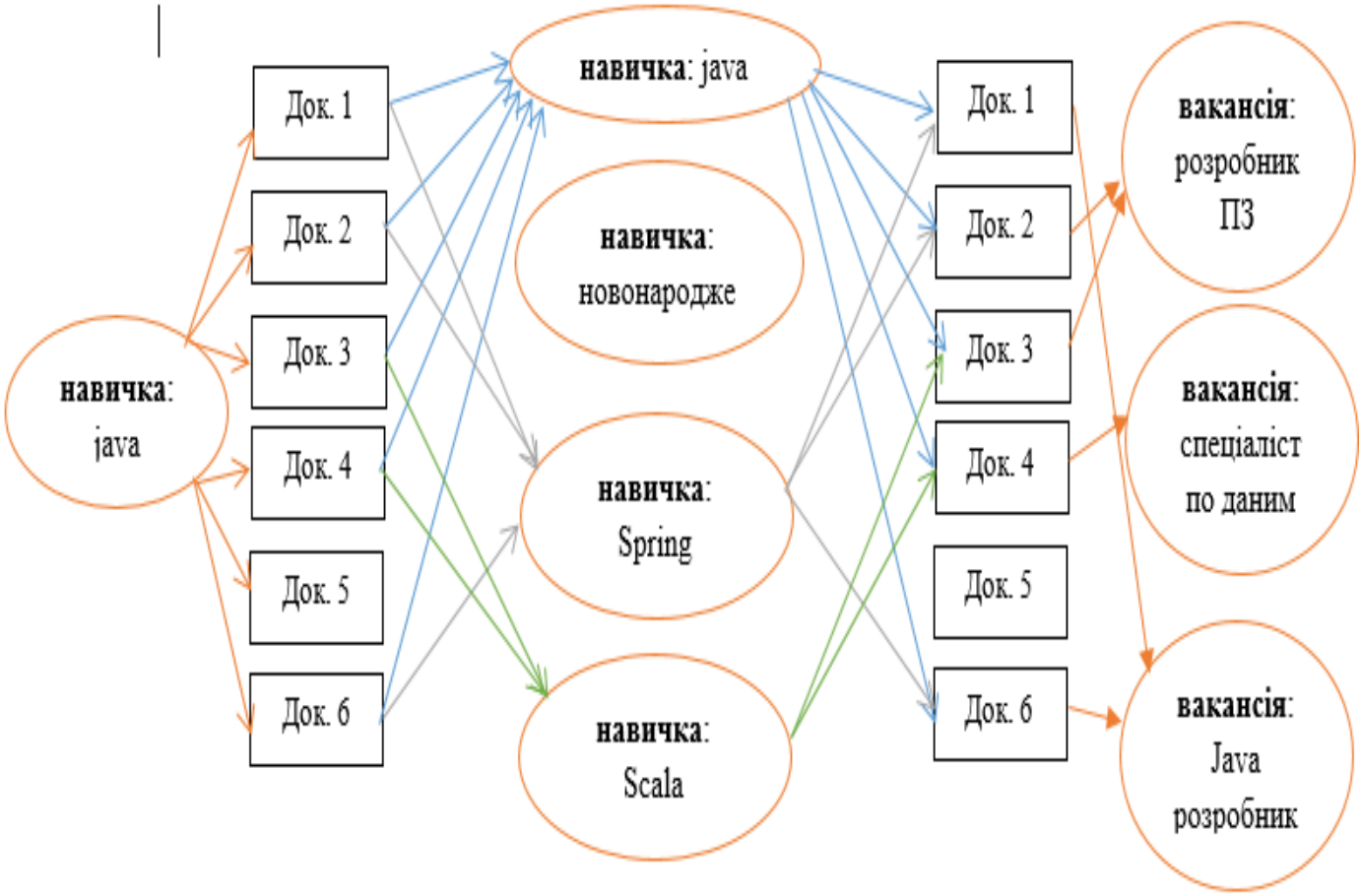


Рисунок 2.5 - Матеріалізація ребер з використанням документів. Ребра існують між документами, які містять спільні терміни. Масштаби ребер обчислюються на льоту, за допомогою функції, яка використовує статистичний розподіл документів, що розділяються між вузлами

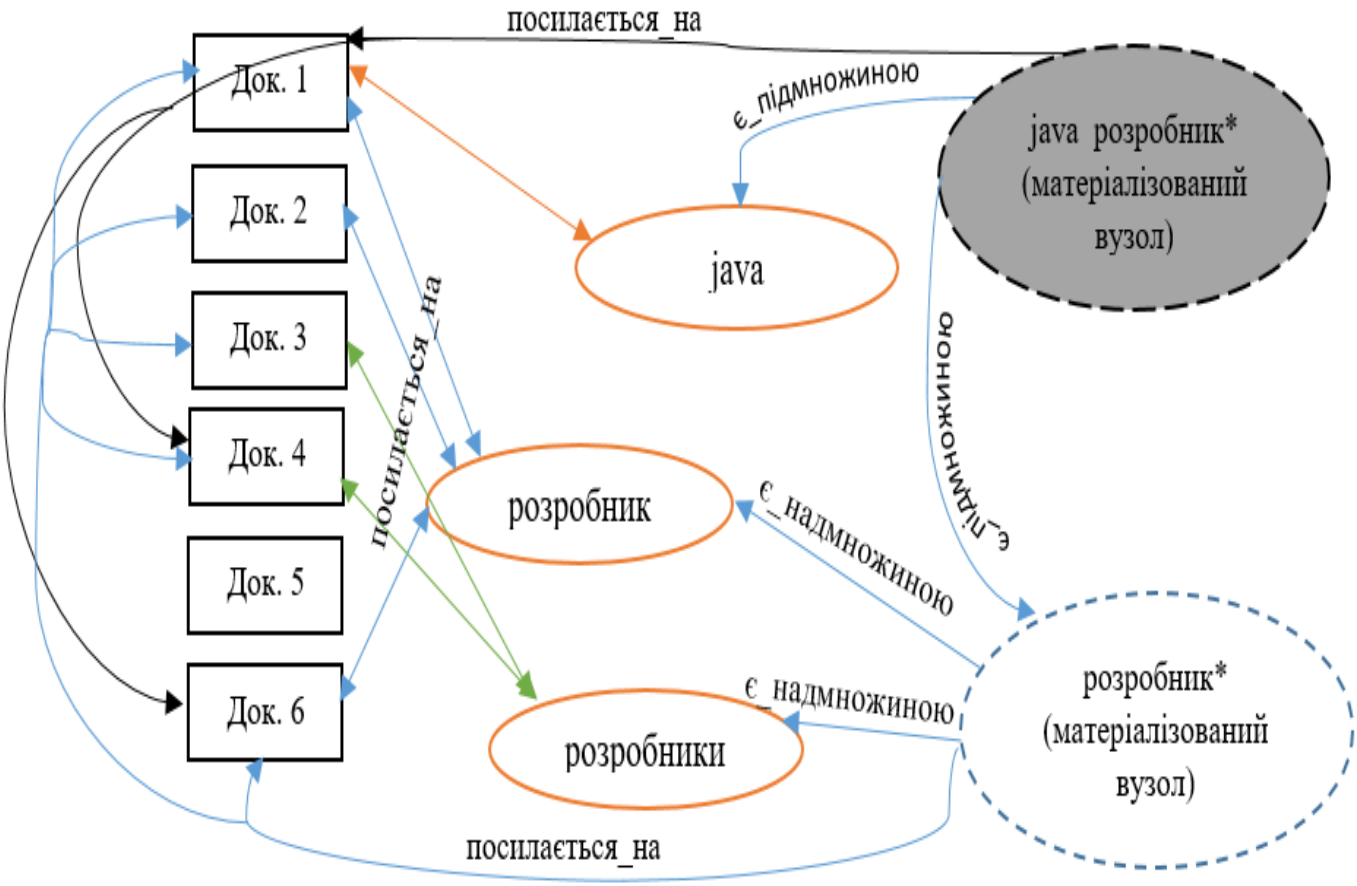


Рисунок 2.6 - Динамічне створення нових вузлів. Нові вузли можуть бути сформовані динамічно з будь-якої довільної комбінації інших вузлів, слів, фраз або будь-якої іншиої лінгвістичної конструкції

Оскільки обидва вузла і ребра можуть бути реалізовані «на льоту», це не тільки дозволяє нам створювати вузли, що представляють довільно складні комбінації існуючих термінів, але також розкладати довільно складні об'єкти і відношення на їх складові частини.

Таким чином, семантичний граф знань забезпечує як без втрат, так і з високим ступенем стиснення, представлення всіх можливих лінгвістичних конструкцій, знайдених в колекції документів, а також кожне потенційне ребро, яке могло б пов'язувати всі можливі матеріалізовані вузли з іншими вузлами.

2.12.3 Виявлення семантичних відношень

Однією з ключових можливостей семантичного графу знань є її здатність виявляти приховані відношення між вузлами. Щоб виявити зв'язок між вузлом з певним тегом (ім'ям поля) t_k з іншим елементом x_i з певним тегом t_j, ми спочатку опитуємо елемент x_i інвертоваого індексу і призначаємо його множину документів на вузол v_i, що відповідає набору документів D_{vi}. Щоб потім знайти вузли-кандидати, які треба обійти, переглядаємо прямий індекс для тега t_k, назвемо цей набір документів $D_{tk} = \{d \mid x \in d, x: t_k\}$. Потім ми визначаємо $V_{vi, tk} = \mathrm{C}v_j \mid x_j \in d, d \in D_{tk} \cap D_{vi}\}$, де vj є вузлом, де зберігається об'єкт x_j, і далі визначаємо $V_{vi, tk}$, як набір вузлів, що зберігають елементи з ребром до x_i типу t_k (рис. 2.7). Потім ми застосовуємо $\forall v_j \in V_{vi, tk}$, функцію *relatedness(Vi, Vj)* для оцінки семантичного взаємозв'язку між v_i и v_j. Цей показник зв'язнаності, що буде описаний в наступному розділі, дозволяє ранжувати кожне з ребер між вузлами, щоб вибрати m найбільш пов'язаних вузлів. Також визначимо поріг t, і будемо приймати тільки відношення $relatedness(v_i, v_j) > t$. Вищеописана операція може відбуватися рекурсивно, щоб перейти на кілька рівнів відношень, як показано на рисунку 2.8.

Ваги розраховуються на основі всього пройденого шляху, хоча можна альтернативно розрахувати ваги, не обумовлені шляхом у графі, і використовувати тільки кожну окрему пару безпосередньо пов'язаних вузлів.

2.12.4 Оцінка семантичних відношень

Однією з найпотужніших особливостей графу семантичних знань (SKG) є його здатність оцінювати ребра між вузлами в графі, ґрунтуючись на силі семантичної схожості між сутностями, представленими цими вузлами. Якщо невідомо, як фраза *physician's assistant* відноситься до *doctor* або навіть до фрази *truck driver*, можна використати SKG для оцінки сили семантичного відношення між всіма цими термінами. Для оцінки семантичної схожості між елементами *xi* і *xj*, матеріалізуємо вихідний вузол *vi* (що представляє набір документів, що містять *xi*), і вузол призначення *vj* (що представляє набір документів, що містять *xj*).

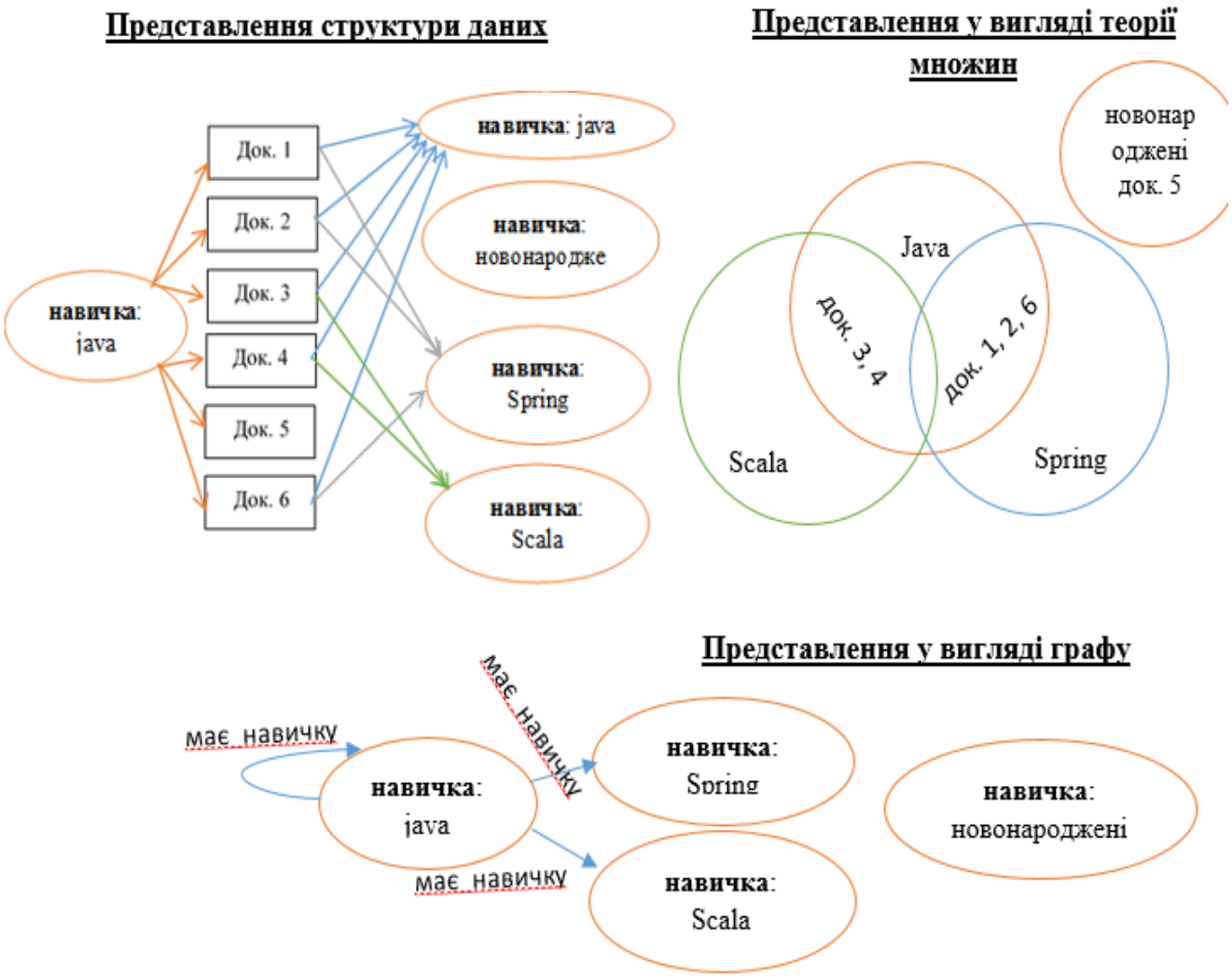


Рисунок 2.7 - Три види обходу. Представлення структури даних являє собою посилання терміна на документ і на терміни в структурах даних. У контексті теорії множин показані відношення між кожним терміном після того, як були встановлені посилання, а представлення у вигляді графу відображає абстрактне уявлення в семантиці, що виникла при взаємодії з SKG

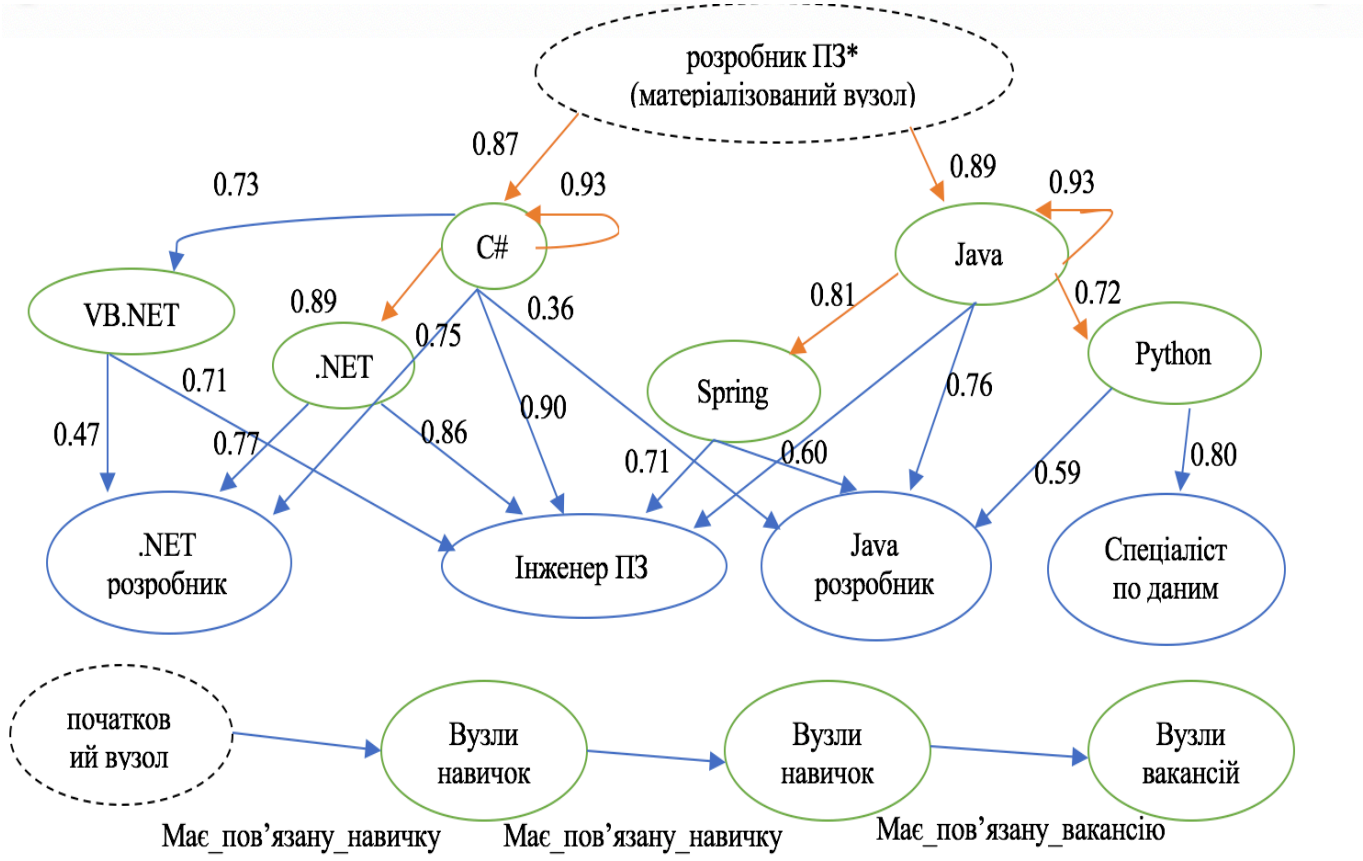


Рисунок 2.8 - Обхід графа. Цей приклад обходить граф починаючи з матеріализованого вузла (*software developer**), через всі пов'язані з навичками ребра (*has-related-skill)*, потім по кожному вузлу через їхні пов'язані ребра (*has-related-skill)*, і, нарешті, з цих вузлів до кожного (*related-job-title)* вузла

Найпростішим прикладом оцінки семантичних зв'язків є порівняння двох безпосередньо з'єднаних вузлів, які ми називатимемо v_i і v_j. Для цього спочатку робиться запит до інвертованого індексу для елемента x_i, який позначено тегом *tj*, і цей запит повертає назад *Dvi*. Потім виконується аналогічний запит для *xj*, який позначається тегом *tk* і повертає *Dvj*.

Ребро *eij* існує між v_i і v_j, коли $f(e_{ij}) 6 = \varphi$. D_{vi} - множина документів переднього плану (*fore-ground*) D_{FG} і, відповідно, називаємо $D_{BG} \subseteq D$ множиною документів заднього плану (*background*). Техніка оцінки базується на гіпотезі, що x_i більш семантично-пов'язаний з x_j, коли відносна частота входження x_j у множину

документів перенього плану D_{FG} є більшою, ніж відносна частота входження x_j у множину документів заднього плану D_{BG} . Як міра схожості дял цієї гіпотези використовується оцінка z:

$$z\left(v_i, v_j\right) = \frac{y - n * p}{\sqrt{n * p(1-p)}}$$

де $n = |D_{FG}|$ - розмір множини документів переднього плану, $y = |f(e_{ij})|$ - кількість термінів x_j з тегом t_k і $p = \frac{\left|D_{v_j}\right|}{D_{BG}}$ - ймовірність побачити термін x_j з тегом t_k у множині документів заднього плану.

Тим не менш, часто потрібно перейти на більш ніж один рівень глибини у графі, щоб оцінити відношення між більш ніж двома вузлами. Наприклад, якщо вирішено зробити перехід від *java* до *developer* до *architect*, то вага ребра між *developer* і *architect* буде мати більший сенс, так як це обумовлено попереднім переходом від *java* до *developer*.

Семантичний граф знань дозволяє зберегти контекст з будь-яких n попередніх вузлів разом з шляхом $P = v_1, v_2, ..., v_n$, з кожним вузлом, що зберігає елемент x_i з тегом t_i. Для підрахунку $z(v_i, v_j)$ між двома будь-якими вузлами, слід також опрацьовувати оцінку вузла по повному шляху P, для цього треба внести деякі зміни в оціночну функцію:

$$D_{FG} = \begin{cases} f\left(e_{ij}\right) & \text{if } n = 3 \\ \left\{ \bigcap_{i=1, j=i+1, k=j+1}^{n-3} g\left(e_{ij}, D_{v_k}\right) \right\} & \text{if } n > 3 \end{cases}$$

де $y = |D_{FG} \cap D_{vm}|$. Застосуємо до оцінки z нормалізацію сигмоїдою, щоб оцінки вміщувалися у відрізок $[-1,1]$. Тепер будемо називати нормалізовану оцінку між вузлами *оцінкою відношення* (*relatedness score*), де 1 означає абсолютно повне відношення, 0 - відсутність відношення, а -1 - абсолютно негативне відношення.

Важливо відзначити, що, оскільки вагові коефіцієнти ребер обчислюються під час обходу (ребра матеріалізуються), можна легко замінити функцію підрахунку очок на іншу, коли це необхідно. Більш проста, але, як правило, менш значуща альтернативна функція *scoring* - це загальна кількість документів, котрі перетинаються, що зазвичай використовується для підрахунку очок в більшості графових баз даних. Також можливе підключення більш складних функцій підрахунку, які використовують статистику, доступну в інвертованому індексі і прямому індексі.

2.12.5 Характеристики масштабування

Семантичний граф знань, створений на основі інвертованого індексу і прямого індексу, принципово розділяє ті ж принципи масшатбування, що лежать в основі розподіленої пошукової системи.

Як описано раніше, як інвертований, так і прямий індекси добре масштабуються по горизонталі до трильйонів документів, розподілених по декількох серверах. Незважаючи на те, що між термінами на кожному шарі інвертованого і прямого індексу буде значний збіг, кількість термінів росте логарифмічно, оскільки кожен додатковий документ з меншою імовірністю, додасть нові терміни до індексу. Документи, навпаки, завжди розділені між серверами, так що всі операції можуть виконуватися паралельно з підмножиною документів на кожному шарді.

Для багатовимірних обходів графу (наприклад, прохід від навичок до назв робочих місць, а потім до галузей) необхідно, щоб для кожного вкладеного рівня відбувалася додатковий процес агрегації. Наприклад, якщо запущено прохід по графу по двом шардам і шард 1 повертає вузли *a*, *b*, *c*, а шард 2 повертає вузли *a*, *c*, *d*, тоді

необхідно відправити інший уточнюючий запит на шард 1, щоб отримати його статистику для раніше відсутнього вузла *d* і один запит на шард 2, щоб отримати його статистику для раніше відсутнього вузла *b*.

Складність такого уточнення лінійно залежить від кількості вкладених рівнів, і рідко бувають випадки, для яких потрібно багато вкладених рівнів обходу. З огляду на ці характеристики масштабування, семантичний граф знань може бути легко побудований і запущений широкомасштабно.

2.13 Висновки

У даному розділі було розглянуто методи та інструменти, доступні для побудови та використання семантичних баз знань. Ці методи включають видобуток масивних обсягів журналів запитів, що використовують імовірнісні графові моделі масових ієрархічних даних (PGMHD) в кластері Hadoop, щоб знайти цікаві терміни та фрази, а також семантично пов'язані терміни та фрази, які можуть бути використані для розширення поняття. Також було описано метод виявлення та зменшення неоднозначності сенсу термінів і фраз, які знаходяться в журналах запитів.

Далі було розглянуто модель, яка називається Semantic Knowledge Graph, яка використовує співвідношення між словами та фразами в рамках документів, щоб автоматично генерувати граф співвідношення між цими фразами. Цей граф можна обійти, виявляючи та оцінюючи міцність зв’язків між будь-якими суб'єктами, що містяться в ньому, виходячи виключно з вмісту в документах пошукової системи.

Такі компоненти самі по собі є корисними інструментами, а їх композиція може утворити потужний "цільовий додаток", який вміє індексувати вміст у пошукову систему, а потім використовувати автоматизовані семантичні бази знань для аналізу та інтерпретації вхідних запитів або документів. Ці методики були успішно застосовані в одній з найбільших систем для пошуку роботи в світі і в остаточному підсумку було підвищено релевантність пошукової системи (за даними NDCG) на 59-76% [32].

Таке значне поліпшення релевантності результатів пошуку є свідченням результатів, які можуть бути досягнуті за допомогою використання розподілених великих аналітичних даних для автоматизації створення семантичних баз знань та їх застосування для підвищення релевантності інформаційно-пошукової системи.

3 ЗАСТОСУВАННЯ СЕМАНТИЧНОГО ГРАФУ ЗНАНЬ ДЛЯ ПОБУДОВИ ОНТОЛГІЇ НАВЧАЛЬНИХ ДИСЦИПЛІН

Даний розділ має на меті показати процес вибору технологій для реалізації проекту, процес створення та наповнення системи попередньо підготовленними даними визначеної структури. Для створення системи обрано модель семантичного графу знань, математична модель котрого описана у попередньому розділі.

3.1 Вибір стеку технологій

Для реалізації SKG буде використано Apache Lucene/Solr для потрібних йому структур даних. Використані структури даних включають інвертований індекс для пошуку попередньо існуючих вузлів, логіку перетину множин документів, потрібну для матеріалізації нових вузлів та ребер з інвертованого індексу термін-документ і прямий індекс документ-термін, потрібний лдя обходу матеріалізованих між вузлами ребер.

Так як Apache Solr є веб сервером, його буде використано як фреймворк для побудови RESTful API поверх реалізації графу, для обробки HTTP запитів, використовуватиметься Jetty.

Опис логіки та вихідного коду. Процес розгортання

Проект написано мовою Java. Для пакування проекту у JAR-файл використано зборщик проектів Apache Maven. На рисунку 3.1 показано список вищезгаданих залежностей.

```
<groupId>com.ykravchuk</groupId>
<artifactId>semantic-knowledge-graph</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>jar</packaging>


<name>semantic_knowledge_graph</name>

<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <solr.version>6.5.1</solr.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.apache.solr</groupId>
        <artifactId>solr-test-framework</artifactId>
        <version>${solr.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.lucene</groupId>
        <artifactId>lucene-test-framework</artifactId>
        <version>${solr.version}</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.solr</groupId>
        <artifactId>solr-core</artifactId>
        <version>${solr.version}</version>
```

Рисунок 3.1 - Фрагмент списку залежностей системи

Реалізацію моделі даних, відповідну описаному SKG, представлено на рисунку 3.2 у вигляді UML-діаграми з найважливішими компонентами шару моделі даних.

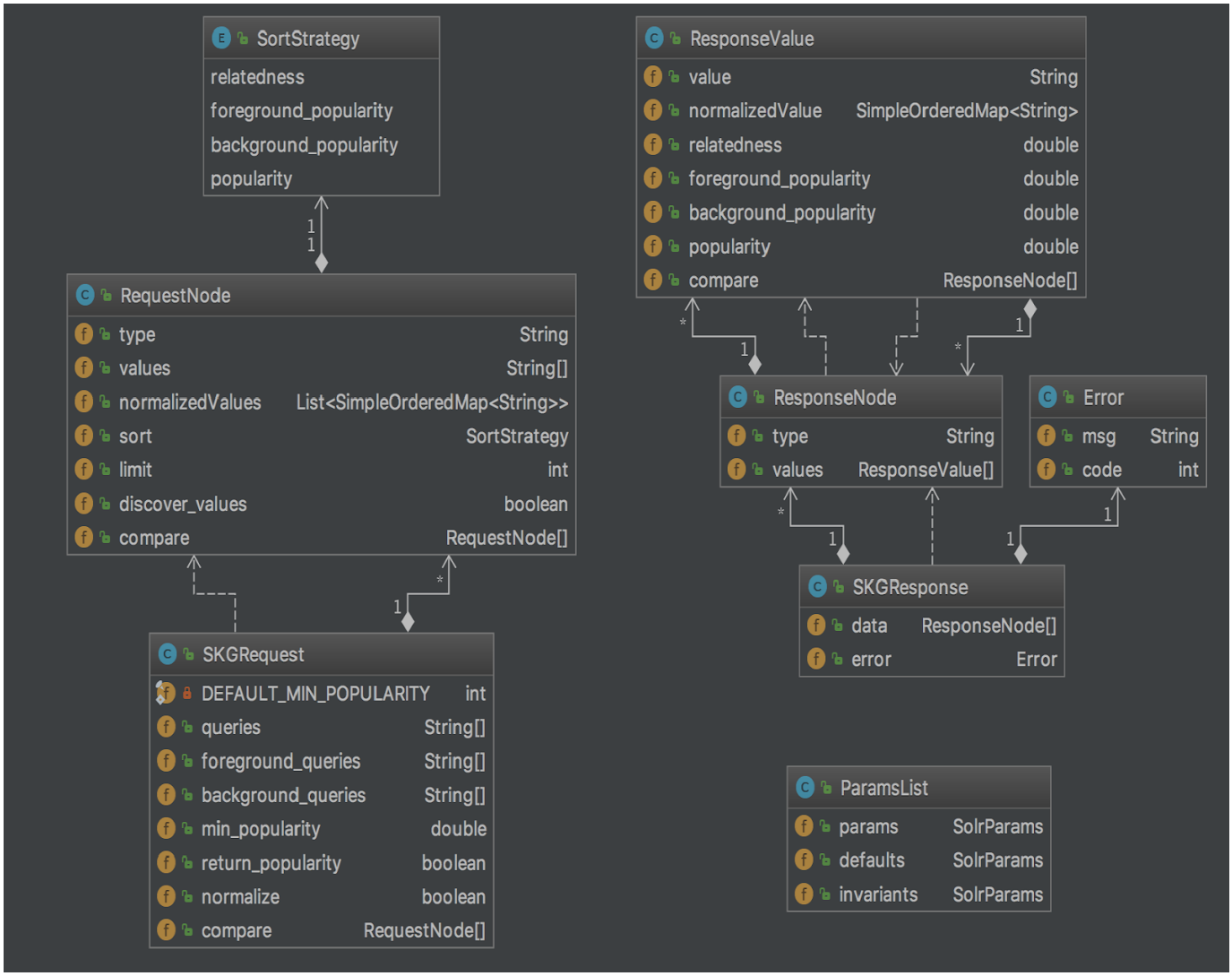


Рисунок 3.2 - Фрагмент списку залежностей системи

Наступним кроком було написання модулю оцінки значення *relatedness-score*, котра залежить від результату *z*-функції. Фрагменти коду представлені рисунком 3.3 та рисунком 3.4.

```
private void relatednessScore(ResponseNode response,
                              int foregroundOccurrences, int backgroundOccurrences) {

    IntStream.range(0, response.values.length)
            .forEach(k -> response.values[k].relatedness =
                    BinomialStrategy.zFunction(foregroundOccurrences, backgroundOccurrences,
                            response.values[k].foreground_popularity,
                            response.values[k].background_popularity));
}
```

Рисунок 3.3 - Код методу розрахунку *relatedness-score*

```
private void relatednessScore(ResponseNode response,
                              int foregroundOccurrences, int backgroundOccurrences) {

    IntStream.range(0, response.values.length)
            .forEach(k -> response.values[k].relatedness =
                    BinomialStrategy.zFunction(foregroundOccurrences, backgroundOccurrences,
                            response.values[k].foreground_popularity,
                            response.values[k].background_popularity));
}
```

Рисунок 3.4 - Код методу розрахунку *z*-функції

Після написання коду для обробки запитів до REST API, був розгорнутий та зконфігурований сервер Apache Solr, процес розгортування котрого буде описано далі. На рисунку 3.5 показано результат успішної зборки проекту. Для наочності опущено фрагменти результатів тестування та логування компонентів, що не стосуються інстанціювання Apache Solr/Lucene та Jetty.

Рисунок 3.5 - Процес зборки проекту і тестовий запуск Apache Solr

Тепер Apache Solr доступний на порту 8983. Після логіну до його системи управління, видно, що розгортування пройшло успішно.

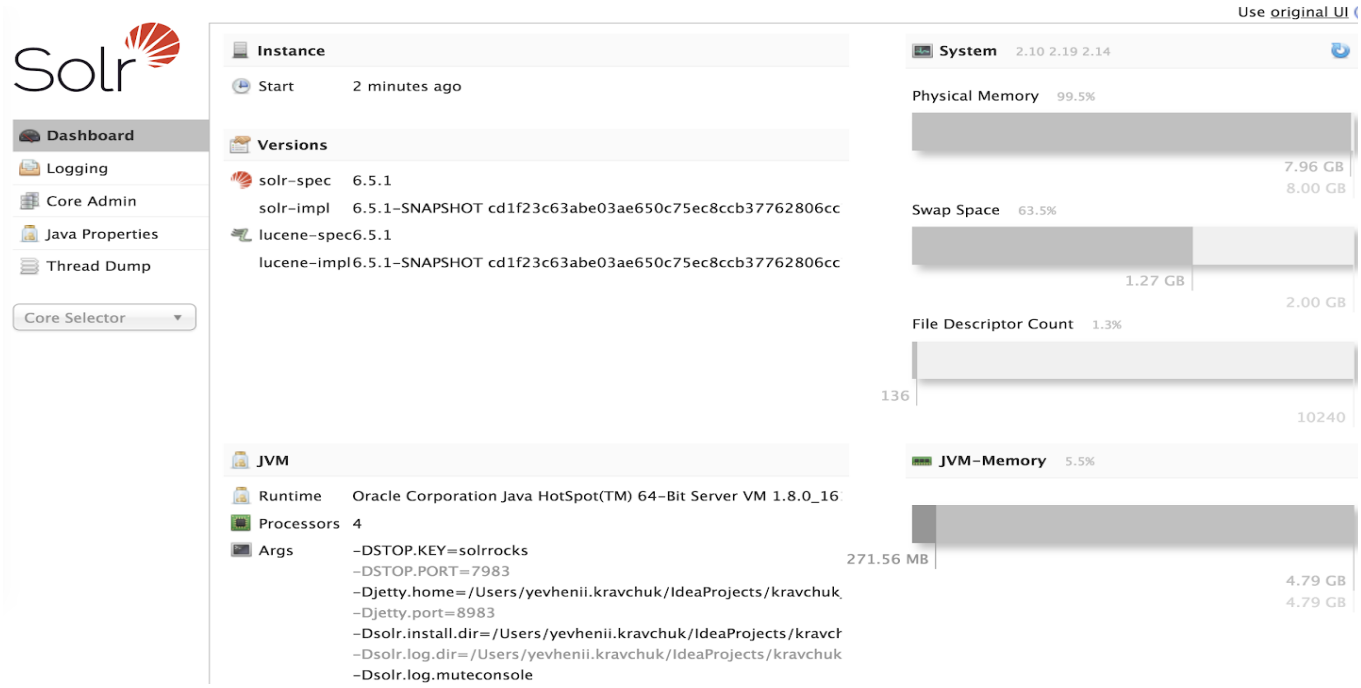


Рисунок 3.6 - Панель моніторингу Apache Solr

Для демонстрації роботи була обрана вибірка даних з ресурсу для спеціалістів з data science та data mining, Kaggle. Процес пошуку вибірки даних показано на рисунку 3.7. Вибірка містить у собі очищені дані і є достатньо актуальною у часі, має формат CSV, тобто, у документі є визначена схема даних. До того ж вона достатньо повна, так як містить потрібні поля - назву курсу та короткий опис.

Після скачування даних, треба вказати двигуну Apache Solr маппінг схеми, вказавши метадані про поля, які поля будуть індексовані, і які поля являються ключовими.

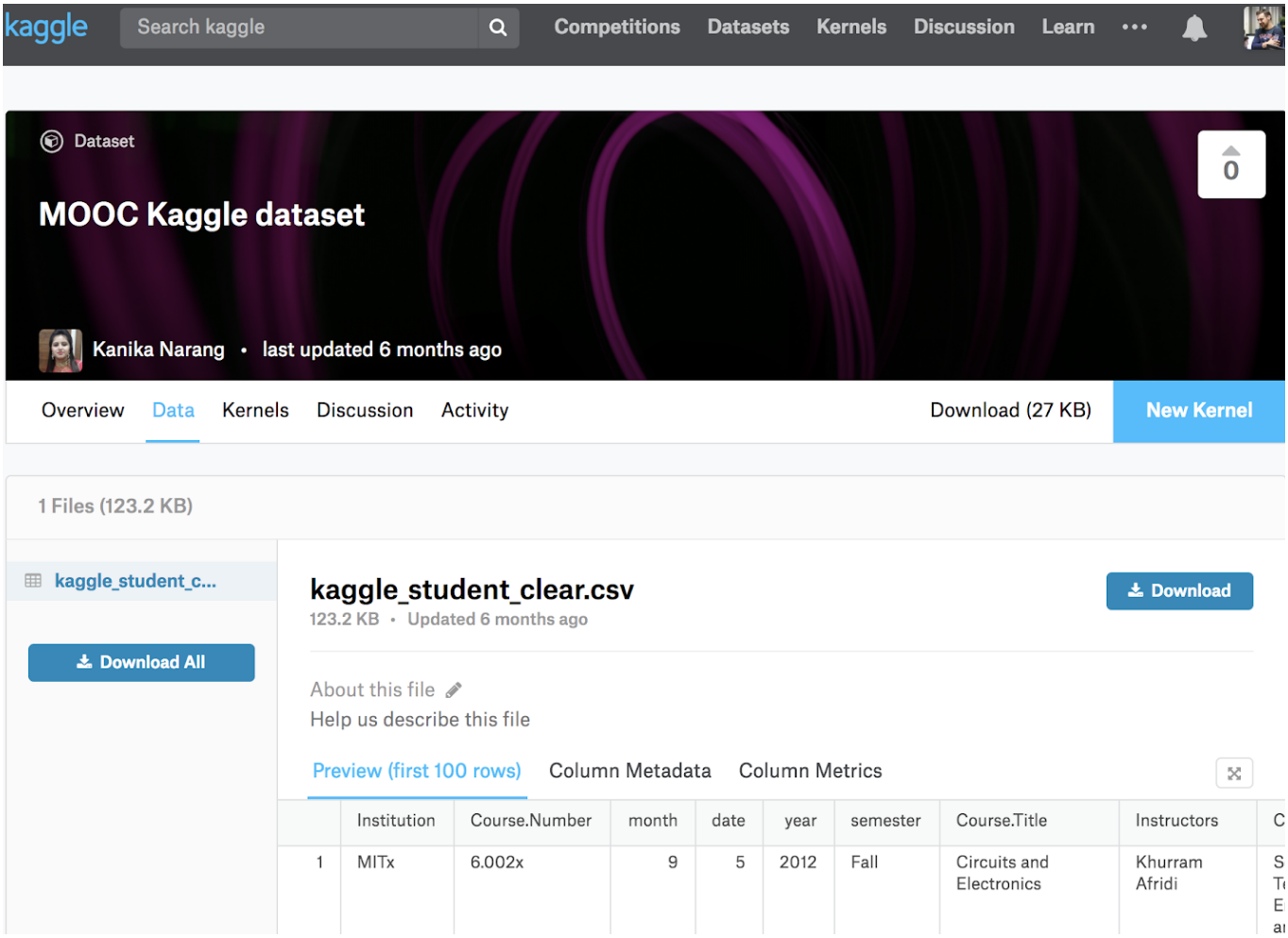


Рисунок 3.7 - Очищена вибірка даних курсів з MIT та Гарвардського університету на ресурсі Kaggle

Повна структура даних показана на рисунку 3.8. Не дивлячись на те, що деякі поля не є репрезентативними та значущими для нашої онтології, включимо їх до індексу.

```
id
Institution
Course.Number
month
date
year
semester
Course.Title
Instructors
Course.Subject
Year
Honor.Code.Certificates
Participants..Course.Content.Accessed.
Audited....50..Course.Content.Accessed.
Certified
X..Audited
X..Certified
X..Certified.of...50..Course.Content.Accessed
X..Played.Video
X..Posted.in.Forum
X..Grade.Higher.Than.Zero
Total.Course.Hours..Thousands.
Median.Hours.for.Certification
Median.Age
X..Male
X..Female
X..Bachelor.s.Degree.or.Higher
drop_outs
```

Рисунок 3.8 - Список полів у CSV-файлі вибірки даних

Маппінг даних для Apache Solr описаний у файлі schema.xml, найбільш значущі фрагменти маппінгу показані на рисунку 3.9.

```
<uniqueKey>id</uniqueKey>

<fields>
    <field name="id" type="long" indexed="true" stored="true" required="false" multiValued="false"/>
    <field name="Institution" type="string" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Course.Number" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="month" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="date" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="year" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="semester" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Course.Title" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Instructors" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Course.Subject" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Year" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="Honor.Code.Certificates" type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Participants..Course.Content.Accessed." type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Audited....50..Course.Content.Accessed." type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Certified" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Audited" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Certified" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Certified.of...50..Course.Content.Accessed" type="text" indexed="true" stored="false"
           required="false" multiValued="true"/>
    <field name="X..Played.Video" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Posted.in.Forum" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Grade.Higher.Than.Zero" type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Total.Course.Hours..Thousands." type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Median.Hours.for.Certification" type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="Median.Age" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Male" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Female" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="X..Bachelor.s.Degree.or.Higher" type="text" indexed="true" stored="false" required="false"
           multiValued="true"/>
    <field name="drop_outs" type="text" indexed="true" stored="false" required="false" multiValued="true"/>
    <field name="_version_" type="long" indexed="true" stored="true"/>
</fields>
```

Рисунок 3.9 - Фрагмент маппінгу структури даних для Apache Solr

Після проходження даного етапу і перезапуску сервера, можна починати завантаження даних до Apache Solr.

Для цього використано утиліту Insomnia, котра дозволяє зручним чином працювати з HTTP-запитами. Конфігурація POST-запиту показана на рисунку 3.10. Фрагмент тіла запиту відповідає раніше зазначеній структурі.

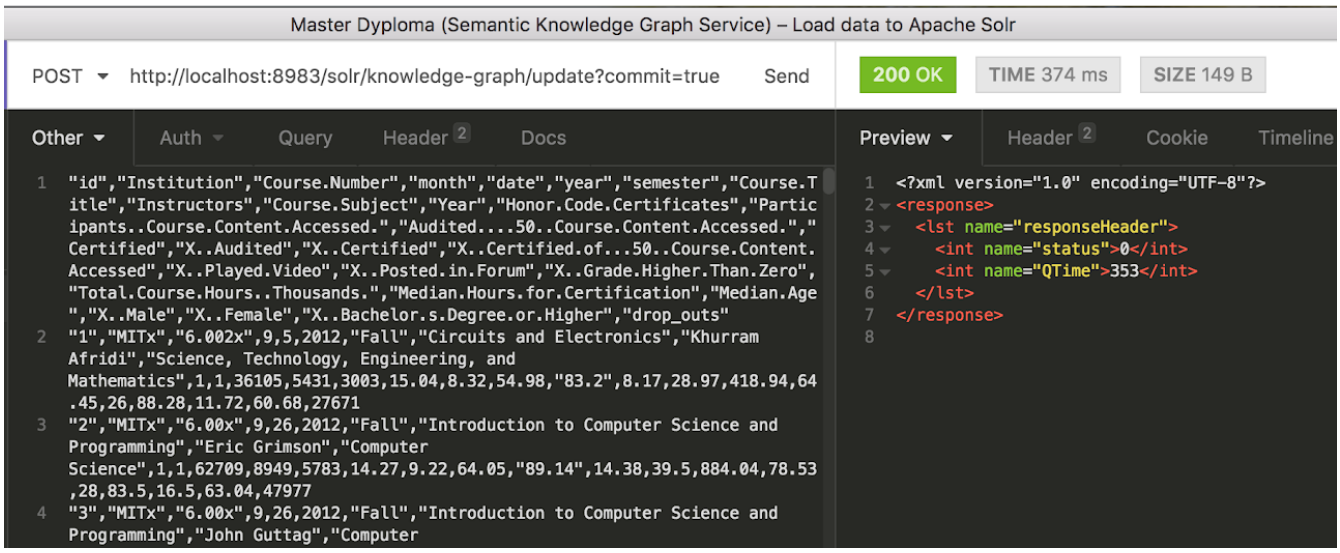


Рисунок 3.10 - Запит на завантаження даних до Apache Solr

У правій частині рисунку 3.10 видно відповідь, котра свідчить про успішне завантаження корпусу даних.

Для демонстрації роботи надсилається запит на пошук даних. Тіло запиту і результат показані на рисунку 3.11. У запиті відбувається пошук даних, котрі містять заголовок *“Introduction to”* для пошуку найрелевантніших базових курсів і, в той же час, результат порівнюється по співпадінню теми *“Subject”* і вказано те, що найбільше нас цікавлять теми *“Science”* та *“Technology”*.

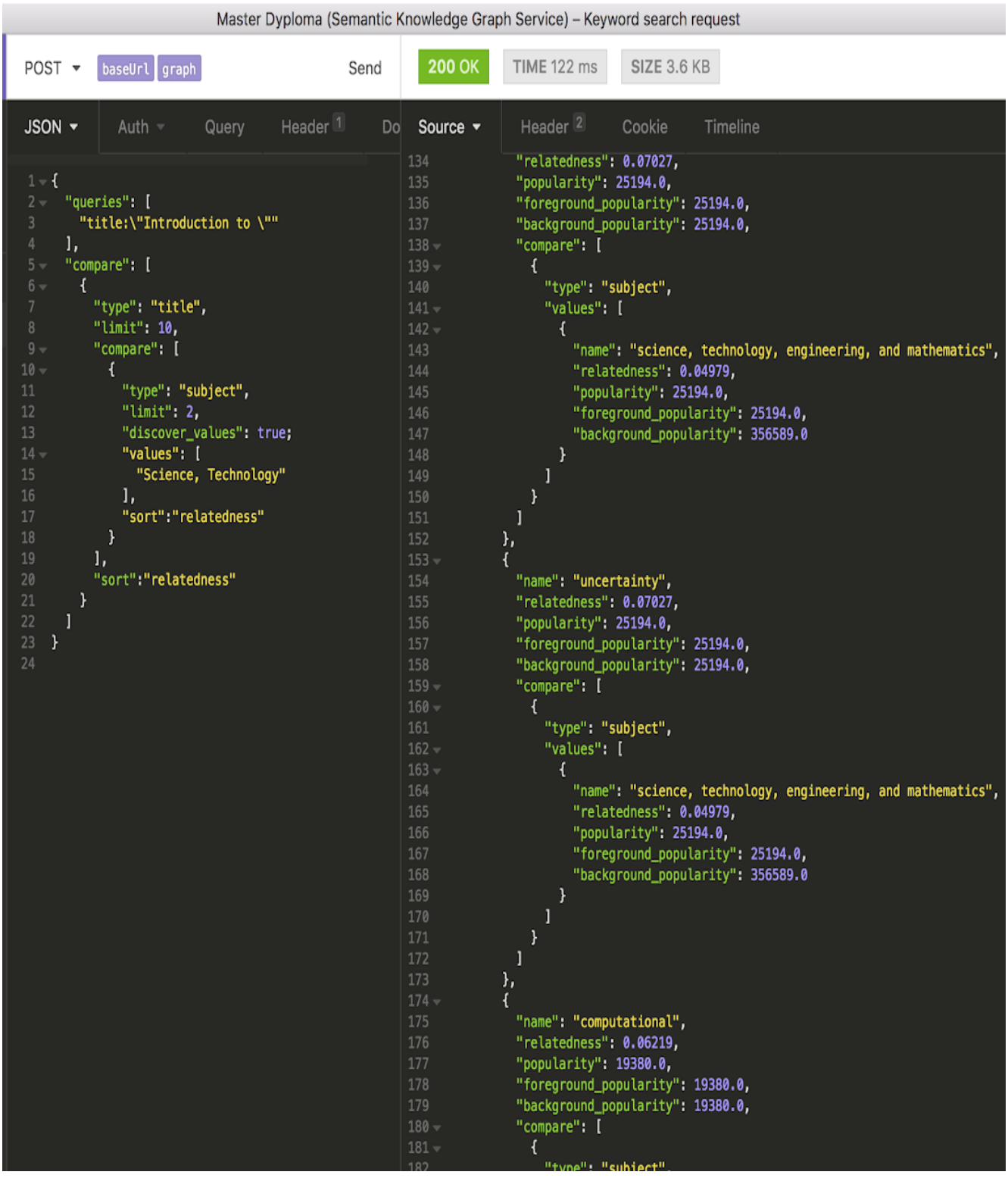


Рисунок 3.11 - Приклад комплексного запиту

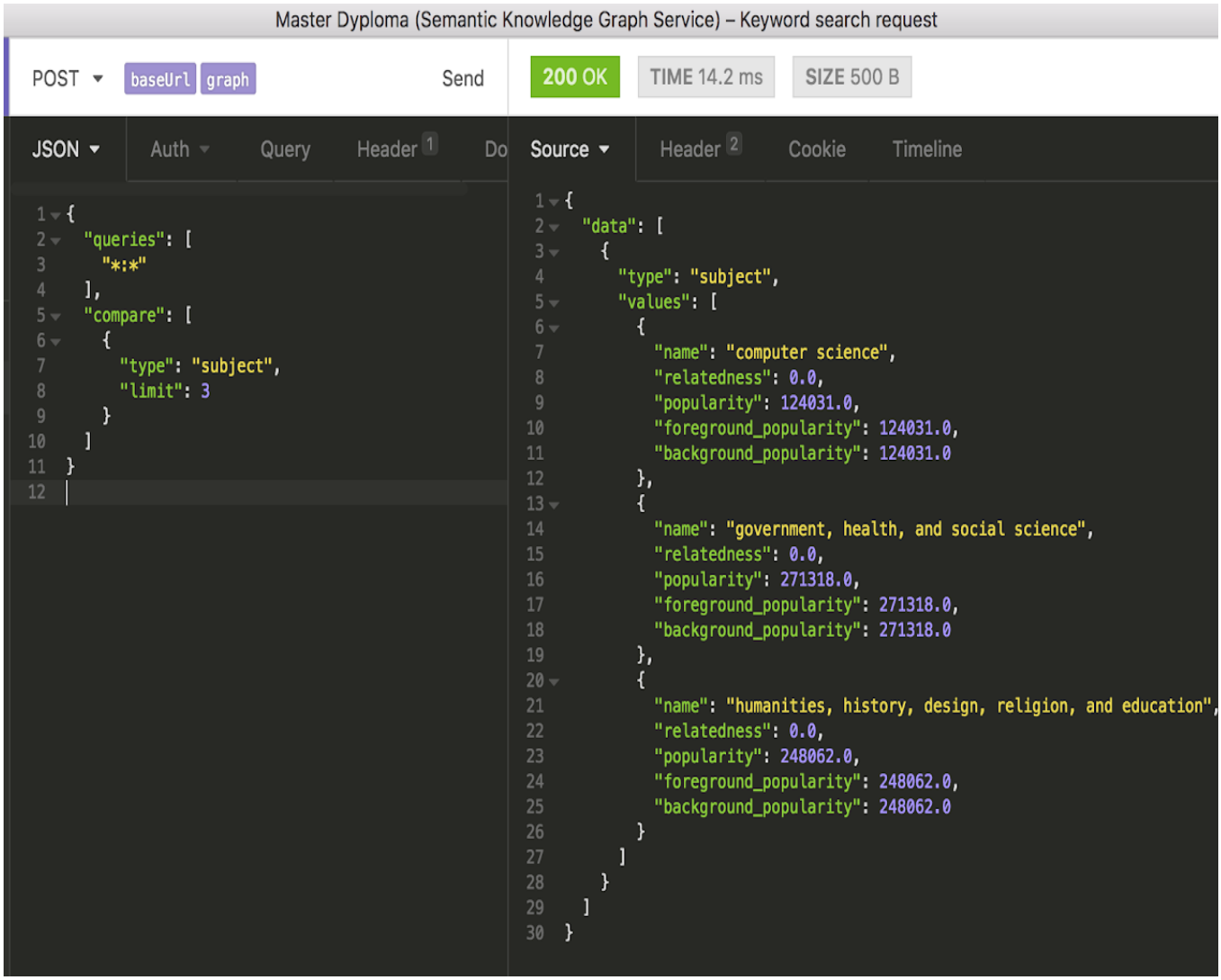


Рисунок 3.12 - Приклад пошуку найпопулярніших тем у корпусі

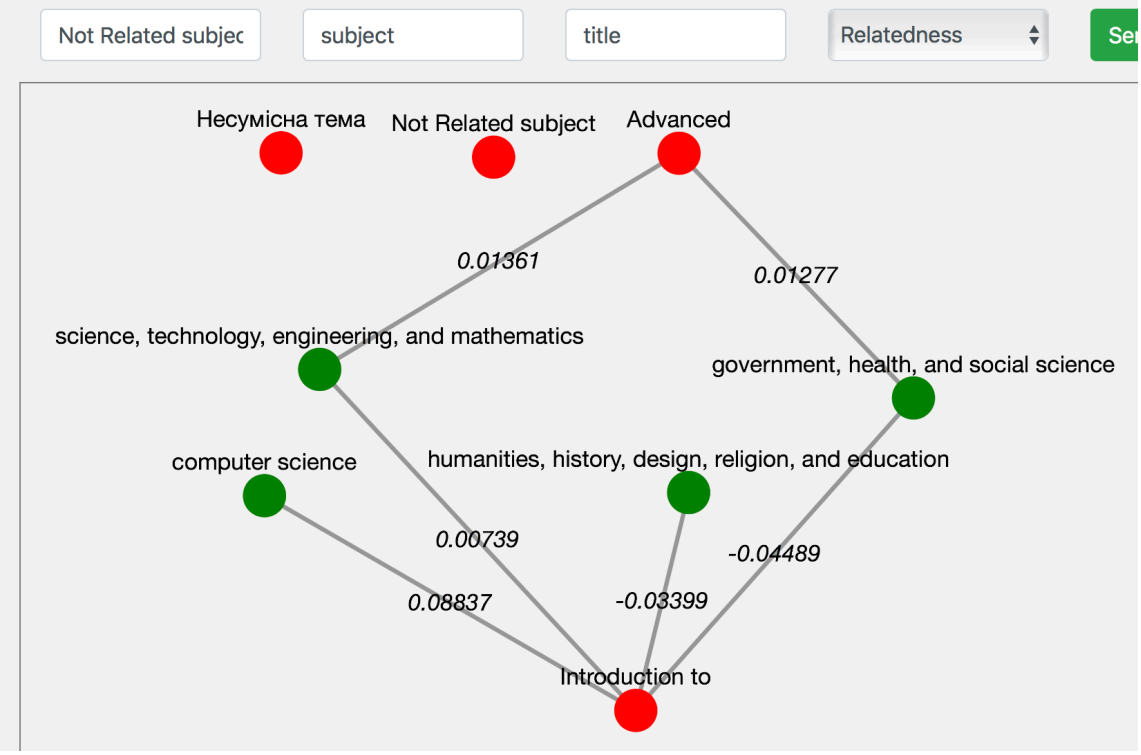


Рисунок 3.13 – Приклад побудови графу через користувацький інтерфейс

Проаналізувавши кількість входжень кожної теми у вибірку даних, видно, що навіть на невеликому наборі даних, семантичний граф знань показує результати, котрі відповідають дійсності.

3.2 Висновки

В ході виконання даного етапу роботи було проведено дослідження щодо вибору інструментарію та набору технологій. В результаті було обрано Apache Solr та Apache Lucene. Це пов'язано з тим, що дана система є де-факто стандартом світовій практиці побудови інфраструктури для пошуку по текстовим даним та містить функціонал, що надає можливість будувати високоефективні індексні структури даних.

Після однозначного вибору технологій була проведена процедура конфігурації додатку, що зображено на рисунках 3.1 та 3.9, в результаті чого було отримано

можливість завантажити набір даних з професійного ресурсу для спеціалістів в сфері науки про дані.

Наступним завданням було виконати завантаження даних до двигуна Apache Solr, що показано на рисунку 3.10. В результаті виконаних дій була реалізована математична модель семантичного графу знань, працездатність котрої було доведено шляхом надсилання запитів до REST API семантичного графу.

Після виконання тестування були отримані результати, що відображають дані, відсортовані в порядку спаду семантичного відношення до пошукового запиту користувача, що проілюстровано на рисунку 3.11 та рисунку 3.12.

4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ “ОНТОЛОГІЧНІ МОДЕЛІ НАВЧАЛЬНИХ ДИСЦИПЛІН ТА ЗАСТОСУВАННЯ DATA MINING ДЛЯ ОЦІНКИ ВЗАЄМОДІЇ ЗМІСТУ МОДЕЛЕЙ”

Розділ має на меті проведення маркетингового аналізу стартап проекту “Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей” задля визначення принципової можливості його ринкового впровадження та можливих напрямів реалізації цього впровадження.

Метою розділу є формування інноваційного мислення, підприємницького духу та формування здатностей щодо оцінювання ринкових перспектив і можливостей комерціалізації основних науково-технічних розробок, сформованих у попередній частині магістерської дисертації у вигляді розроблення концепції стартап-проекту “База знань як сервіс” в умовах висококонкурентної ринкової економіки глобалізаційних процесів.

4.1 Опис стартап-проекту

Опис проекту “Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей ” наведено у Таблиці 4.1.

Таблиця 4.1. Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Створення та розгортання семантичного графу знань у хмарі з наданням користувачеві доступу читання та запису через REST API. Надання SPARQL-ендпойнту для читання та оновлення	1. Використання для виконання аналізу даних, що зберігаються у вигляді корпусів документів.	REST API надає можливість використовувати систему як сервіс, що значно спрощує доступ до сховища. Знаходження транзитивних відношень, класів та об’єктів онтологій може спростити

даних у сховищі триплетів, можливості завантажувати цілі онтології, знаходити транзитивні відношення, класи та об'єкти онтологій.		подальший аналіз даних.
	2. Використання сховища триплетів у інших додатках як спосіб збереження сильно пов'язаних доменних даних.	Можливість виконання SPARQL запитів дає можливість гнучкого доступу до даних та їх оновлення. Сервіс легко інтегрувати у інші системи завдяки REST API

Отже, проект " Онтологічні моделі навчальних дисциплін та застосування Data Mining для оцінки взаємодії змісту моделей" може бути використаним як інструментом для деякого аналізу даних, так і прошарком постійного збереження сильно пов'язаних доменних даних у інфраструктурі інших додатків завдяки можливості використання даної системи як сервісу через REST API та реалізованому SPARQL-ендпойнту.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*				*W (слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Конкурент 1*	*Конкурент2*	*Конкурент 3*			
1.	Форма виконання	Веб-сервіс	Програма	Веб-додаток	Програма			+
2.	Собівартість	Низька	Низька	Висока	Висока			+
3.	Кросплатформність	Так	Ні	Так	Так			+
4.	Наявність SPARQL-ендпойнту для	Так	Так	Так	Так		+	

	зчитування даних							
5.	Наявність SPARQL-ендпойнту для оновлення даних	Так	Ні	Ні	Так			+
6.	Застосування логічного виведення	Так	Ні	Так	Так			+
7.	Горизонтальне масштабування	Ні	Так	Ні	Так	+		

Сильними сторонами проекту є форма виконання у вигляді веб-сервісу, низька собівартість, кросплатформеність, Наявність SPARQL-ендпойнту для оновлення даних, застосування логічного виведення. Слабкою стороною є відсутність можливості горизонтального масштабування, нейтральною - наявність SPARQL-ендпойнту для зчитування даних. Отож, система є конкурентоспроможною.

4.2 Технологічний аудит ідеї проекту

В межах даного підрозділу необхідно провести аудит технології, за допомогою якої можна реалізувати ідею проекту (технології створення товару).

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1.	Створення сховища триплетів	Apache Jena TDB	Наявна	Безкоштовна, доступна
		Dydra	Наявна	Частково безкоштовна
2.	Створення REST API для доступу до бази	Spring Boot, Maven	Наявні	Безкоштовна, доступна

	знань	Amazon API Gateway, AWS Lambda	Наявна	Платні
3.	Хмарне розгортання додатку	Heroku	Наявна	Безкоштовна, доступна
		AWS EC2, S3	Наявна	Платні

Обрані технології реалізації ідеї проекту: Apache Jena TDB через повну безкоштовність фреймворку та наявність докладної документації, наявність досвіду роботи розробників з даною технологією; Spring Boot, Maven через простоту використання, безкоштовність та можливість розгортання додатків на основі таких технологій у хмарі; Heroku для розгортання у хмарі через безкоштовність.

4.3. Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1.	Кількість головних гравців, од	5
2.	Загальний обсяг продаж, грн/ум.од	8000 грн./ум.од
3.	Динаміка ринку (якісна оцінка)	Зростає
4.	Наявність обмежень для входу (вказати характер обмежень)	Немає

5.	Специфічні вимоги до стандартизації та сертифікації	Немає
6.	Середня норма рентабельності в галузі (або по ринку), %	R = (3000000 * 100) / (1000000 * 12) = 25%

Отже, було проаналізовано наявність попиту, обсяг, динаміку розвитку ринку. Обмеження для входу на ринок відсутні, динаміка ринку зростає, галузь є рентабельною.

Надалі визначаються потенційні групи клієнтів, їх характеристики, та формується орієнтовний перелік вимог до товару для кожної групи (табл. 4.5).

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товау*
1.	Необхідне програмне забезпечення(REST API) для доступу до бази знань як до сервісу	Потенційними цільовими групами є дослідницькі центри, університети та компанії, специфіка роботи яких потребує обробки сильно пов’язаних даних	Цільова група займається дослідженнями або має обробляти дані у вигляді онтологій та RDF-графів	Рішення повинне бути придатним до інтеграції в інші більш складні системи, мати SPARQL-endpoint, бути здатним надавати перелік класів, об’єктів і транзитивних зв’язків у сховищі триплетів бути розгорнутим у хмарі

Згідно проведеної характеристики потенційних клієнтів стартап-проекту випливає, що на ринку є затребуваним програмне забезпечення(REST API) для доступу до бази знань як до сервісу і потенційними цільовими групами є дослідницькі центри, університети та компанії, специфіка роботи яких потребує обробки сильно пов’язаних даних.

Після визначення потенційних груп клієнтів проводиться аналіз ринкового середовища: складаються таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (табл. № 4.6-4.7). Фактори в таблиці подавати в порядку зменшення значущості.

Таблиця 4.6 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1.	Конкуренція	Вихід на ринок великої компанії	1. Вихід з ринку 2. Запропонувати великій компанії поглинути себе 3. Передбачити додаткові переваги власного ПЗ для того, щоб повідомити про них саме після виходу міжнародної компанії на ринок
2.	Зміна потреб користувачів	Користувачам необхідне програмне забезпечення з іншим функціоналом	1. Передбачити можливість додавання нового функціоналу до створюваного ПЗ
3.	Зміна тарифів провайдера хмарного розгортання на платні	Необхідність оплати послуг провайдера хмари	1. Пошук іншого безкоштовного провайдера 2. Пошук інвестицій для оплати існуючого провайдера
4.	Надходження на ринок альтернативних продуктів	Перехід користувачів нашого товару на інший продукт	Впровадження нового функціоналу, якого немає у конкурентів
5.	Уповільнення росту ринку	Скорочення користувачів продуктів, що тільки виходять на ринок	Інвестиції у впровадження ефективної реклами продукту

Отже, було проаналізовано фактори загроз ринкового впровадження проекту, серед яких: конкуренція, уповільнення росту ринку, зміна потреб користувачів, зміна

тарифів провайдера хмарного розгортання на платні та надходження на ринок альтернативних продуктів. Було також запропоновано можливі реакції компанії.

Таблиця 4.7 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1.	Стрімкий ріст попиту на інструменти обробки даних у вигляді онтологій та RDF-графів	Наявність попиту на інструменти для обробки даних у вигляді онтологій та RDF-графів	Змога запропонувати продукт більшій кількості потенційних користувачів
2.	Поява нових різонерів	Надання нового функціоналу для надання результатів роботи нового логічного виведення	Розробка нового функціоналу у вигляді нового HTTP запиту для надання користувачам результатів логічного виведення
3.	Стрімке зростання росту ринку	Компаніям, що тільки виходять на ринок, буде простіше отримати клієнтів	Змога запропонувати продукт більшій кількості потенційних користувачів
4.	Обслуговування додаткових груп споживачів	Поява нових потенційних груп споживачів	Змога розширити продукт для подальшого впровадження у нові галузі
5.	Розширення асортименту можливих послуг	Поява нового функціоналу, що привабить нових користувачів	Розробка нового функціоналу, що є потребою певної групи користувачів

У Таблиці 4.7 наведено фактори можливостей ринкового впровадження проекту, серед яких: стрімкий ріст попиту на інструменти обробки даних у вигляді онтологій та RDF-графів, поява нових різонерів, стрімке зростання росту ринку, обслуговування додаткових груп споживачів, розширення асортименту можливих послуг; було також запропоновано можливі реакції компанії.

Надалі проводиться аналіз пропозиції: визначаються загальні риси конкуренції на ринку (табл. 4.8).

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Вказати тип конкуренції - досконала	Існує 3 фірми-конкурентки на ринку	Врахувати ціни конкурентних компаній на початкових етапах створення бізнесу, реклама (вказати на конкретні переваги перед конкурентами)
2. За рівнем конкурентної боротьби - міжнародний	Одна з компаній – з іншої країни, дві – з України	Додати можливість вибору мови ПЗ, щоб легше було у майбутньому вийти на міжнародний ринок
3. За галузевою ознакою - внутрішньогалузева	Конкуренти мають ПЗ, яке використовується лише всередині даної галузі	Створити основу ПЗ таким чином, щоб можна було легко переробити дане ПЗ для використання у інших галузях
4. Конкуренція за видами товарів: - товарно-видова	Види товарів є однаковими, а саме – програмне забезпечення	Створити ПЗ, враховуючи недоліки конкурентів
5. За характером конкурентних переваг - нецінова	Вдосконалення технології створення ПЗ, щоб собівартість була нижчою	Використання менш дорогих технологій для розробки, ніж використовують конкуренти
6. За інтенсивністю - не марочна	Бренди відсутні	-

У Таблиці 4.8 наведено ступеневий аналіз конкуренції на ринку, де було визначено особливості конкурентного середовища та їх вплив а діяльність підприємства. Однією з найбільш важливих дій компанії для досягнення

конкурентоспроможності є необхідність створити основу ПЗ таким чином, щоб можна було легко переробити дане ПЗ для використання у інших галузях

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі (табл. 4.9).

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

Складов і аналізу	*Прямі конкуренти в галузі*	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
	Навести перелік прямих конкурентів	*Визначити бар'єри входження в ринок*	*Визначити фактори сили постачальників*	*Визначити фактори сили споживачі в*	*Фактори загроз з боку замінників*
Висновки:	Існує 3 конкуренти на ринку. Найбільш схожим за виконанням є конкурент 2, так як його рішення також представлене у вигляді веб-додатку	Так, можливості для входу на ринок є, бо наше рішення має SPARQL-ендпойнт для оновлення даних та можливість логічного виведення	Постачальники відсутні.	Важливим для користувача є швидкість роботи ПЗ	Товари-замінники можуть використат и більш дешеву технологію створення ПЗ та зменшити собівартість товару

Було здійснено аналіз конкуренції в галузі за М. Портером, в результаті чого було визначено, що існує 3 конкуренти на ринку. Найбільш схожим за виконанням є конкурент 2, так як його рішення також представлене у вигляді веб-додатку, але можливості для входу на ринок є, бо наше рішення має SPARQL-ендпойнт для оновлення даних та можливість логічного виведення.

За результатами аналізу таблиці робиться висновок щодо принципової можливості роботи на ринку з огляду на конкурентну ситуацію. Також робиться висновок щодо характеристик (сильних сторін), які повинен мати проект, щоб бути

конкурентоспроможним на ринку. Другий висновок враховується при формулюванні переліку факторів конкурентоспроможності у п. 4.6) На основі аналізу конкуренції, проведеного в п. 3.5 (табл. 4.9), а також із урахуванням характеристик ідеї проекту (табл. 4.2), вимог споживачів до товару (табл. 4.5) та факторів маркетингового середовища (табл. № 4.6-4.7) визначається та обґрунтовується перелік факторів конкурентоспроможності. Аналіз оформлюється за табл. 4.10

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1.	Наявність SPARQL-endpoint для зчитування та оновлення даних	Дозволяє користувачам здійснювати гнучкий доступ до бази знань
2.	Можливість завантажувати цілі онтології	Висока швидкість заповнення бази знань
3.	Наявність REST API	Дозволяє інтегрувати сервіс у складні системи завдяки універсальному API
4.	Хмарне розгортання	Дозволяє звертатись до бази знань як до сервісу
5.	Горизонтальне масштабування	Можливість гнучкого масштабування за допомогою додавання апаратних компонентів

У таблиці 4.7 наведено обґрунтування факторів конкурентоспроможності, серед яких: наявність SPARQL-endpoint для зчитування та оновлення даних, можливість завантажувати цілі онтології, наявність REST API та хмарне розгортання. Було також наведено обґрунтування цих факторів.

За визначеними факторами конкурентоспроможності (табл. 4.10) проводиться аналіз сильних та слабких сторін стартап-проекту (табл. 4.11).

У наступній таблиці наведено проведення аналізу сильних та слабких сторін стартап-проекту, факторами конкурентоспроможності виступили такі: наявність SPARQL-endpoint для зчитування та оновлення даних, можливість завантажувати цілі онтології, наявність REST API, хмарне розгортання, горизонтальне масштабування.

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін проекту

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з нашим підприємством*						
			3	2	1	0	1	2	3
1.	Наявність SPARQL-endpoint для зчитування та оновлення даних	20				+			
2.	Можливість завантажувати цілі онтології	20			+				
3.	Наявність REST API	15			+				
4.	Хмарне розгортання	15		+					
5.	Горизонтальне масштабування	10					+		

Отже, серед сильних сторін проекту можна виділити наступні: наявність SPARQL-endpoint для зчитування та оновлення даних, можливість завантажувати цілі онтології, наявність REST API, можливість хмарного розгортання. Серед слабких сторін можна виділити відсутність можливості горизонтального масштабування.

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення. Наприклад: зниження доходів потенційних споживачів – фактор загрози, на основі якого можна зробити прогноз щодо посилення значущості

цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

У наступній таблиці буде проілюстровано SWOT-аналіз стартап-проекту, тобто його слабкі та сильні сторони, можливості та загрози виходу на ринок.

Таблиця 4.12. SWOT- аналіз стартап-проекту

Сильні сторони: наявність SPARQL-endpoint для зчитування та оновлення даних, можливість завантажувати цілі онтології, наявність REST API, можливість хмарного розгортання	Слабкі сторони: можливість зміни тарифів провайдером хмарного розгортання на платні, відсутність можливості горизонтального масштабування
Можливості: стрімкий ріст попиту на інструменти обробки даних у вигляді онтологій та RDF-графів, можливість впровадження нових різонерів, стрімке зростання росту ринку, обслуговування додаткових груп споживачів,розширення асортименту можливих послуг	Загрози: конкуренція, зміна потреб користувачів, зміна тарифів провайдера хмарного розгортання на платні, надходження на ринок альтернативних продуктів, уповільнення росту ринку

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів.

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1.	Створення додатку з використанням фреймворків Apache Jena, Spring Boot	90%	3 місяці
2.	Створення	35%	8 місяців

	програми на основі без використання будь- яких фреймворків для обробки даних		

З означених альтернатив обирається та, для якої: а) отримання ресурсів є більш простим та ймовірним; б) строки реалізації – більш стислими. Тому обираємо альтернативу (створення додатку з використанням фреймворків Apache Jena, Spring Boot).

4.4 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів (табл. 14).

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів.

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовни й попит в межах цільової групи (сегменту)*	*Інтенсивніст ь конкуренції в сегменті*	*Прост ота входу у сегмент*
1.	Дослідницькі центри	Спрощення роботи з високо пов’язаними даними	Великий	Існує 3 конкуренти, які надають схожі, але більш вузькі і дорогі рішення.	Наявність REST API, SPARQL-ендпойнту, логічного виведення
2.	Підприємства	Спрощення роботи з високо пов’язаними даними	Великий		Можливість інтеграції в уже існуючі системи завдяки REST API, зручне хмарне розгортання,

					наявність SPARQL-ендпойнту
Які цільові групи обрано: обираємо підприємства та дослідницькі центри					

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку. Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку. За М. Портером, існують три базові стратегії розвитку, що відрізняються за ступенем охоплення цільового ринку та типом конкурентної переваги, що має бути реалізована на ринку (за витратами або визначними якостями товару).

Отже, проілюструвати базову стратегію розвитку можна у виляді таблиці 4.15

Таблиця 4.15 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспромо жні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку*
1.	Створення веб-сервісу, використовуючи Spring Boot, Apache Jena TDB	Ринкове позиціонування	Можливість інтеграції в уже існуючі системи завдяки REST API, зручне хмарне розгортання, наявність SPARQL-ендпойнту, логічного виведення	Диференціація

Було обрано таку альтернативу розвитку проекту: створення веб-сервісу, використовуючи Spring Boot, Apache Jena TDB, адже завдяки цим технологіям можна досягнути ключових конкурентоспроможних позицій кінцевого продукту.

Таблиця 4.16 - Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки*
1.	Ні	Так	Буде, а саме: основною задачею є розробка ПЗ з використанням сховища триплетів(конкуренти 1, 2, 3), форма виконання - веб-сервіс (конкурент 2)	Зайняття конкурентної ніші

Отже, було визначено базову стратегію конкурентної поведінки як зайняття конкурентної ніші.

Визначимо стратегію позиціонування у Таблиці 4.17, що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект

Таблиця 4.17 - Визначення стратегії позиціонування

№ п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап- проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
1.	Наявність універсального API, зручне хмарне розгортання, наявність SPARQL-ендпойнту, логічного виведення	Диференціація	Можливість інтеграції в уже існуючі системи завдяки REST API, зручне хмарне розгортання, наявність SPARQL-ендпойнту, логічного виведення	Інтеграція, хмарне розгортання, SPARQL-ендпойнт

Отже, було вибрано такі асоціації, які мають сформувати комплексну позицію власного проекту: інтеграція (адже завдяки REST API сервіс просто інтегрувати у існуючі системи), хмарне розгортання (оскільки Spring Boot додатки легко розгортаються в хмарах), SPARQL-ендпойнт (у системі є повноцінний SPARQL-ендпойнт).

4.5 Розробка маркетингової програми

Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у табл. 4.18 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.18 - Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)*
1.	Наявність універсального API	Додаток реалізований у вигляді RESTful сервісу, що надає відповіді у вигляді JSON, що дає змогу користувачам звертатись до сервісу за допомогою стандартних HTTP POST та GET запитів	Перевага в універсальності на можливості інтегрувати сервіс у існуючі системи.
2.	Можливість зручного хмарного розгортання	Можливість розгорнути додаток всюди, де є функціонал розгортання Spring Boot додатків, а така можливість є у більшості хмарних провайдерів	Користувачі мають змогу працювати з системою віддалено у хмарі
3.	Наявність SPARQL-ендпойнту,	Можливість виконання будь-яких SPARQL-запитів	Підтримка стандарту SPARQL 1.1
4.	Наявність логічного виведення	Виведення транзитивних відношень у онтологіях	Можливість виведення транзитивних відношень у онтологіях

Отже бачимо, що проект має ключові переваги перед конкурентами, які повністю відповідають потребам цільової аудиторії. Додаток реалізований у вигляді RESTful сервісу, що надає відповіді у вигляді JSON, що дає змогу користувачам звертатись до сервісу за допомогою стандартних HTTP POST та GET запитів, а це є досить універсальним способом для подальшої інтеграції сервісу в інші системи.

Далі у Таблиці 4.19 проілюстрована трирівнева маркетингова модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання.

Таблиця 4.19 - Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Веб-сервіс, що надає доступ до сховища триплетів за допомогою HTTP запитів, дозволяє працювати зі SPARQL-ендпойнтом та надає можливості логічного виведення та хмарного розгортання</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Наявність універсального API
2. Можливість зручного хмарного розгортання
3. Наявність SPARQL-ендпойнту,
4. Наявність логічного виведення</td><td>1.Нм
2.Нм
3.Нм
4.Нм</td><td>1.Технологічна
2.Технологічна
3.Технологічна
4.Технологічна</td></tr>
<tr><td colspan="3">Якість: згідно до стандарту ISO 4444 буде проведено тестування</td></tr>
<tr><td colspan="3">Маркування відсутнє</td></tr>
<tr><td colspan="3">Моя компанія: “Semantic knowledge”</td></tr>
<tr><td rowspan="2">III. Товар із підкріплюнням</td><td colspan="3">1-місячна пробна безкоштовна версія</td></tr>
<tr><td colspan="3">Постійна підтримка для користувачів</td></tr>
</table>

За рахунок чого потенційний товар буде захищено від копіювання: патент

Було описано три рівні моделі товару, з чого можна зробити висновок, що основні властивості товару у реальному виконанні є нематеріальними та технологічними. Також було надано сутність та складові товару у задумці та товару з підкріпленням.

Після формування маркетингової моделі товару слід особливо відмітити – чим саме проект буде захищено від копіювання. У даному випадку найбільш вірогідним гарантом буде патент.

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (табл. 4.20). Аналіз проводиться експертним методом.

Таблиця 4.20 - Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники, грн.*	*Рівень цін на товари-аналоги, грн.*	*Рівень доходів цільової групи споживачів, грн.*	*Верхня та нижня межі встановлення ціни на товар/послугу, грн.*
1.	45000	38000	150000	35000-40000

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (табл. 4.21).

Таблиця 4.21 - Формування системи збуту

№ п/п	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*

1.	Придбання підписки та оплата щомісячних внесків для продовження ліцензії	Продаж	0(напряму), 1(через одного посередника)	Власна та через посередників

Отже, система приноситиме прибуток завдяки щомісячним внескам для подовження ліцензії та придбанням підписок, продаж будк виконуватись напряму або через одного посередника.

Останньою складовою маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (табл. 4.22).

Таблиця 4.22 - Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуютьс я цільові клієнти*	*Ключові позиції, обрані для позиціону вання*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
1.	Придбання ліцензії на користування в мережі Інтернет, щомісячне її продовження, користування сервісом у хмарі або ж на власних серверах.	Інтернет	Інтеграція, хмарне розгортання, SPARQL-ендпойнт	Показати переваги сервісу, у тому числі і перед конкурентами	Демо-ролик із використання, рекламні оголошення на популярних сайтах.

Отже, в Таблиці 4.22 наведено концепцію маркетингових комунікацій, було визначено, що придбання ліцензії на користування буде здійснюватись в мережі Інтернет, необхідним буде щомісячне її продовження, користування сервісом можливе у хмарі або ж на власних серверах.

4.6 Висновки

Згідно до проведених досліджень існує можливість ринкової комерціалізації проекту. Також, варто відмітити, що існують перспективи впровадження з огляду на потенційні групи клієнтів, бар‘єри входження не є високими, а проект має дві значні переваги перед конкурентами. Для успішного виконання проекту необхідно реалізувати програму із використанням засобів Apache Jena, Spring Boot. Для успішного виходу на ринок у продукту повинні бути наступні характеристики:

- наявність універсального API
- можливість зручного хмарного розгортання
- наявність SPARQL-ендпойнту
- наявність логічного виведення

В рамках даного дослідження були розраховані основні фінансово-економічні показники проекту, а також проведений менеджмент потенційних ризиків. Проаналізувавши отримані результати, можна зробити висновок, що подальша імплементація є доцільною.

Було визначено такі сильні сторони: наявність SPARQL-endpoint для зчитування та оновлення даних, можливість завантажувати цілі онтології, наявність REST API, можливість хмарного розгортання. Серед слабких сторінможна виділити можливість зміни тарифів провайдером хмарного розгортання на платні, відсутність можливості горизонтального масштабування.

Можливості для виходу на ринок включають стрімкий ріст попиту на інструменти обробки даних у вигляді онтологій та RDF-графів, можливість впровадження нових різонерів, стрімке зростання росту ринку, обслуговування додаткових груп споживачів, розширення асортименту можливих послуг. Наявні такі фактори загроз: конкуренція, зміна потреб користувачів, зміна тарифів провайдера хмарного розгортання на платні, надходження на ринок альтернативних продуктів, уповільнення росту ринку.

ВИСНОВКИ

В ході роботи було розгорнуто систему побудови семантичного графу знань, котра дозволяє автоматизовано будувати графи знань на основі корпусу документів, що завантажені у систему.

Під час виконання роботи було описано поняття онтології у контексті комп'ютерних наук та семантичного веб, розглянуто різні існуючі онтолгогії навчальних дисциплін, що використовуються у світі та виявлено слабкі сторони ручного створення та підтримки онтології.

Наступним кроком було проведено аналіз методів та підходів, що автоматизують процес побудови онтологій і розглянуто математичні моделі цих методів, їх обмеження та проблематику, котра постає при вирішенні подібних задач як з теоретичної сторони, так і у контексті практичних аспектів побудови програмних додатків, що реалізують такі моделі.

Працездатність математичної моделі семантичного графу знань було доведено шляхом створення, розгортання та тестування прототипу системи, створеної на основі попередньо обраного сучасного набору технологій, котрий широко використовується у світовій практиці.

В якості основи було обрано Apache Solr, що є найбільш придатною для виконання даного роду досліджень. Функціональність даної технології дозволяє виконувати різного роду операції над масивами даних, зокрема, для побудови ефективних індексних структур даних.

Результати тестування доводять придатність даної математичної моделі для обробки великих масивів даних та автоматичного створення графу знань на основі цих даних. Це підтверджують відповіді, отримані в ході виконання тестових запитів.

Наступним кроком у розвитку даної роботи є, в першу чергу, вдосконалення стратегії завантаження даних шляхом реалізації можливості індексації наборів даних

без попередньо визначеної схеми. Окрім цього важливим аспектом є створення користувацького інтерфейсу для зручного використання реалізованого функціоналу.

Дана реалізація може надати можливість отримання корисної для бізнесу інформації при обробці великих об’ємів даних, що не мають визначеної структури та очевидних взаємозв’язків. Окрім того, система має перспективи розвитку, котрі були описані вище.

ПЕРЕЛІК ПОСИЛАНЬ

1. Грибова В.В., Клещеев А.С. Онтологическая парадигма программирования. // конференція OSTIS-2012 (Open Semantic Technologies for Intelligent Systems), 2012. – с. 213-220.

2. Піднебесна Г.А., Концепція використання онтологій для конструювання засобів індуктивного моделювання. // Індуктивне моделювання складних систем, випуск 5, 2013. - с. 248-255

3. Gruber T. Towards principles for the design of Ontologies used for knowledge sharing // International Journal of Human-Computer Studies. – 1995. – № 43(5/6). – С. 907-928.

4. Петренко І.А, Петренко О.О. Автоматизовані методи пошуку і відкриття необхідних сервісів. // Системні дослідження та інформаційні технології, № 4, 2015.- С.

5. Resource Description Framework (RDF) - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: https://www.w3.org/RDF/ (дата звернення: 12.02.2018). - Назва з екрана.

6. Academic Institution Internal Structure Ontology (AIISO) - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: http://vocab.org/aiiso/schema# (дата звернення: 21.02.2018). - Назва з екрана.

7. Learning Resource Metadata Initiative (LRMI) - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: https://wiki.creativecommons.org/wiki/LRMI (дата звернення: 23.02.2018). - Назва з екрана.

8. Web Ontology Language (OWL) - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: https://www.w3.org/OWL/ (дата звернення: 13.02.2018). - Назва з екрана.

9. Teaching Core Vocabulary Specification - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: http://linkedscience.org/teach/ns/#sec-reference (дата звернення: 26.02.2018). - Назва з екрана.

10. Ontologies informatiques at SemUnit - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: http://semunt.supelec.fr/pubby/page/ressource/unit/_2166_Ontologies_informatiques (дата звернення: 27.02.2018). - Назва з екрана.

11. SPARQL Query Language for RDF - [Електронний ресурс] : [Веб-сайт]. - Електроні дані. - Режим доступу: https://www.w3.org/TR/rdf-sparql-query/ (дата звернення: 14.02.2018). - Назва з екрана.

12. R. Navigli and P. Velardi, “Learning domain ontologies from document warehouses and dedicated web sites,” Computational Linguistics, vol. 30, no. 2, 2004.

13. T. Grainger and T. Potter, Solr in Action. Manning Publications Co, 2014.

14. J. Bobadilla, F. Ortega, A. Hernando, and A. Gutierrez,´ “Recommender systems survey,” Knowledge-Based Systems, vol. 46, pp. 109–132, 2013.

15. J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, “Recommender system application developments: a survey,” Decision Support Systems, vol. 74, pp. 12–32, 2015.

16. C. C. Aggarwal, “Content-based recommender systems,” in Recommender Systems, pp. 139–166, Springer, 2016.

17. M. J. Pazzani and D. Billsus, “Content-based recommendation systems,” in The adaptive web, pp. 325–341, Springer, 2007.

18. X. Su and T. M. Khoshgoftaar, “A survey of collaborative filtering techniques,” Advances in artificial intelligence, vol. 2009, p. 4, 2009.

19. R. Burke, “Hybrid recommender systems: Survey and experiments,” User modeling and useradapted interaction, vol. 12, no. 4, pp. 331–370, 2002.

20. M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, “Semantics-aware content based recommender systems,” in Recommender Systems Handbook, pp. 119–159, Springer, 2015.

21. S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain, “Semantic measures for the comparison of units of language, concepts or entities from text and knowledge base analysis,” arXiv preprint arXiv:1310.1285, 2013.

22. R. Mihalcea, C. Corley, and C. Strapparava, “Corpus-based and knowledge-based measures of text semantic similarity,” in AAAI, vol. 6, pp. 775–780, 2006.

A. Budanitsky and G. Hirst, “Semantic distance in wordnet: An experimental, applicationoriented evaluation of five measures,” in Workshop on WordNet and Other Lexical Resources, vol. 2, 2001.

23. G. Bouma, “Normalized (pointwise) mutual information in collocation extraction,” in Proceedings of the Biennial GSCL Conference, pp. 31–40, 2009.

24. S. T. Dumais, “Latent semantic analysis,” Annual review of information science and technology, vol. 38, no. 1, pp. 188–230, 2004.

25. P. D. Turney, “Mining the web for synonyms: PMI-IR versus lsa on toefl,” in Proceedings of the 12th European Conference on Machine Learning, EMCL ‘01, (London, UK, UK), pp. 491–502, Springer-Verlag, 2001.

26. T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” arXiv preprint arXiv:1301.3781, 2013.

27. K. AlJadda, M. Korayem, C. Ortiz, T. Grainger, J. A. Miller, and W. S. York, “Pgmhd: A scalable probabilistic graphical model for massive hierarchical data problems,” in Big Data (Big Data), 2014 IEEE International Conference on, pp. 55–60, IEEE, 2014.

28. K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The hadoop distributed file system,” in Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on, pp. 1–10, IEEE, 2010.

29. J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” Communications of the ACM, vol. 51, no. 1, pp. 107–113, 2008.

A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, and R. Murthy, “Hive: a warehousing solution over a map-reduce framework,” Proceedings of the VLDB Endowment, vol. 2, no. 2, pp. 1626–1629, 2009.

30. T. Grainger, K. AlJadda, M. Korayem, and A. Smith, “The semantic knowledge graph: A compact, auto-generated model for real-time traversal and ranking of any relationship within a domain,” in IEEE 3rd International Conference on Data Science and Advanced Analytics, IEEE, 2016.

31. K. AlJadda, M. Korayem, T. Grainger, and C. Russell, “Crowdsourced query augmentation through semantic discovery of domain-specific jargon,” in IEEE International Conference on Big Data (Big Data 2014), pp. 808–815, IEEE, 2014.

32. M. Korayem, C. Ortiz, K. AlJadda, and T. Grainger, “Query sense disambiguation leveraging large scale user behavioral data,” in IEEE International Conference on Big Data (Big Data 2015), pp. 1230–1237, IEEE, 2015.