Міністерство освіти і науки України
Донецький національний університет імені Василя Стуса
Київський національний університет будівництва і архітектури
Державний торговельно-економічний університет
Черкаський державний технологічний університет
Львівський національний університет імені Івана Франка
Тернопільський національний технічний університет імені Івана Пулюя

ISSN: 2788-4465

МАТЕРІАЛИ IV ВСЕУКРАЇНСЬКОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ СТУДЕНТІВ, АСПІРАНТІВ ТА МОЛОДИХ ВЧЕНИХ

8 грудня 2023 року

Матеріали надруковані в авторській редакції.
Достовірність поданої інформації лежить на авторах публікацій

УДК 004(063)(083)

Рекомендовано до друку Вченою радою
факультету інформаційних і прикладних технологій
Донецького національного університету імені Василя Стуса
(протокол № 10 від 16.04.2024)

Організаційний комітет конференції:

Голова:

ПРЯМУХІНА Наталія Валентинівна, доктор економічних наук, професор, в. о. декана факультету інформаційних і прикладних технологій ДонНУ імені Василя Стуса.

Заступники голови:

ЗЕЛІНСЬКА Оксана Владиславівна, кандидат технічних наук, доцент, в. о. завідувача кафедри інформаційних технологій ДонНУ імені Василя Стуса.

Відповідальний секретар оргкомітету:

БАБАКОВ Роман Маркович, доктор технічних наук, доцент, професор кафедри інформаційних технологій ДонНУ імені Василя Стуса.

Члени комітету:

НІКОЛЮК Петро Карпович, доктор фізико-математичних наук, професор, професор кафедри інформаційних технологій ДонНУ імені Василя Стуса;

ШТОВБА Сергій Дмитрович, доктор технічних наук, професор, професор кафедри інформаційних технологій ДонНУ імені Василя Стуса;

СІЧКО Тетяна Василівна, кандидат технічних наук, доцент, доцент кафедри інформаційних технологій ДонНУ імені Василя Стуса;

ПОТАПОВА Надія Анатоліївна, кандидат економічних наук, доцент, доцент кафедри інформаційних технологій ДонНУ імені Василя Стуса;

АНТОНОВ Юрій Сергійович, кандидат фізико-математичних наук, доцент, доцент кафедри інформаційних технологій ДонНУ імені Василя Стуса;

КОМАРОВ Василь Федорович, кандидат технічних наук, старший викладач кафедри інформаційних технологій ДонНУ імені Василя Стуса;

ФРИЗ Ірина Василівна, кандидат фізико-математичних наук, старший викладач кафедри інформаційних технологій ДонНУ імені Василя Стуса;

РИМАР Павло Володимирович, старший викладач кафедри інформаційних технологій ДонНУ імені Василя Стуса;

ХМЕЛІВСЬКИЙ Юрій Сергійович, асистент кафедри інформаційних технологій;

СЕНИК Іван Олександрович, асистент кафедри інформаційних технологій;

ГОРЯШИН Антон Сергійович, асистент кафедри інформаційних технологій;

ГОНЧАР Віталій Миколайович, асистент кафедри інформаційних технологій.

Матеріали IV Всеукраїнської науково-практичної конференції «Комп'ютерні технології обробки даних». Вінниця: ДонНУ імені Василя Стуса, 2023. 296 с.

Збірник містить матеріали IV Всеукраїнської науково-практичної конференції «Комп'ютерні технології обробки даних». Тематика збірника окреслює актуальні проблеми, які стосуються питань методів обробки і аналізу даних, інтелектуальних систем та мереж, експертних, рекомендаційних систем та систем підтримки прийняття рішень, баз даних і знань (Big Data), прикладних аспектів обробки даних в інформаційних системах. Матеріали учасників конференції адресовано фахівцям та усім, хто цікавиться сучасним станом вивчення комп'ютерних технологій обробки даних.

УДК 004(063)(083)

ЗМІСТ

СЕКЦІЯ 1. МЕТОДИ ОБРОБКИ І АНАЛІЗУ ДАНИХ

Shulhin O., Shtovba S. COMPARATIVE ANALYSIS OF MODERN ONLINE TOOLS FOR OBJECT RECOGNITION IN IMAGES 12

Бобошко В. В., Штовба С. Д. МЕТРИКА СХОЖОСТІ НАВЧАЛЬНИХ ДИСЦИПЛІН ЗА ЇХ ВНЕСКОМ У КОМПЕТЕНТНОСТІ.....15

Пустовойтенко В. В., Потапова Н. А. МЕТОДИ ТА ТЕХНОЛОГІЇ ОБРОБКИ ДАНИХ ВЕТЕРИНАРНИХ КЛІНІК.....19

Бездушний В. О., Штовба С. Д. ВИЯВЛЕННЯ ПРИХОВАНОЇ ЛАЙКИ В ТЕКСТОВИХ ПОВІДОМЛЕННЯХ ЗА АНАЛІЗОМ ВІЗУАЛЬНО ПОДІБНИХ СИМВОЛІВ 21

Громович О. С., Римар П. В. ДОСЛІДЖЕННЯ МЕТОДІВ ЗБОРУ, ОБРОБКИ ТА АНАЛІЗУ ДАНИХ З МОБІЛЬНИХ СЕНСОРІВ 25

Зінченко Б. В., Римар П. В. ОПТИМІЗАЦІЯ БАЗИ ДАНИХ ЗА ДОПОМОГОЮ ВИКОРИСТАННЯ COLUMNSTORE-ІНДЕКСІВ 28

Колосова К. К., Огороднік М. О., Ніколюк П. К. ПОРІВНЯННЯ РІЗНИХ МЕТОДІВ ЗНАХОДЖЕННЯ ЗНАЧЕННЯ КОРЕЛЯЦІЇ МІЖ НАБОРАМИ ЗМІННИХ.....30

Крохмалюк В. В., Потапова Н. А. ВИКОРИСТАННЯ ІНСТРУМЕНТІВ ДЛЯ ЗБОРУ ДАНИХ У ДОСЛІДНИЦЬКІЙ ДІЯЛЬНОСТІ ЯК ЕФЕКТИВНИЙ МЕТОД ОТРИМАННЯ ІНФОРМАЦІЇ.....33

Бевз Д. М., Римар П. В. ЕФЕКТИВНІСТЬ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ НА ПРИКЛАДІ ЗАДАЧІ КЛАСИФІКАЦІЇ ТЕКСТІВ 35

Луцков М. П., Римар П. В. МЕТОДИ ТЕХНІЧНОЇ ОПТИМІЗАЦІЇ САЙТІВ.....38

Афанасьєва Д. С., Вєтров О. С. РЕАЛІЗАЦІЯ БІНАРНОГО ДЕРЕВА МОВОЮ ПРОГРАМУВАННЯ PYTHON З ВИКОРИСТАННЯМ РЕКУРСІЇ.....41

Афанасьєва Д. С., Фриз І. В. ЗАСТОСУВАННЯ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ ТА ЇХ СИСТЕМ У СФЕРІ КОМП'ЮТЕРНИХ НАУК.....44

Бєжин Є. В., Січко Т. В. ІНТЕГРОВАНІ СИСТЕМИ УПРАВЛІННЯ В ПРОМИСЛОВОСТІ: АНАЛІЗ ЕФЕКТИВНОСТІ ТА ВПРОВАДЖЕННЯ ОПТИМАЛЬНИХ СТРАТЕГІЙ.....47

Бєжин Є. В., Січко Т. В. ОПТИМІЗАЦІЯ ВИРОБНИЧИХ ПРОЦЕСІВ В УМОВАХ НЕСТАБІЛЬНОСТІ ВИРОБНИЧОГО СЕРЕДОВИЩА: ВИКОРИСТАННЯ АДАПТИВНИХ МЕТОДІВ.....49

Бєжин Є. В., Хмелівський Ю. С. СТАТИСТИЧНЕ НАВЧАННЯ В УМОВАХ НЕВИЗНАЧЕНОСТІ: МОДЕЛЮВАННЯ ТА УПРАВЛІННЯ НЕВИЗНАЧЕНІСТЮ У ПРОГНОСТИЧНИХ МОДЕЛЯХ.....51

Бондарєв О. О., Горін В. В., Ніколюк П. К. ЗВ'ЯЗОК МАТЕМАТИЧНОЇ КРИПТОГРАФІЇ ТА ОПТИМАЛЬНОГО КОДУВАННЯ.....53

Бурківський О. С., Зелінська О. В. СУЧАСНІ ФРЕЙМВОРКИ МАШИННОГО НАВЧАННЯ.....56

Бурківський О. С., Потапова Н. А. ЗАСТОСУВАННЯ ДИНАМІЧНОГО ПРОГРАМУВАННЯ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ПРО НАЙДОВШУ ПОСЛІДОВНІСТЬ.....58

Грибаніна А. О., Потапова Н. А. АНАЛІЗ АЛГОРИТМІВ БІНАРНОГО ТА ЛІНІЙНОГО ПОШУКУ.....61

Гуцуляк Д. В., Потапова Н. А. АНАЛІЗ ЕФЕКТИВНОСТІ АЛГОРИТМІВ ПОШУКУ.....63

Диркач Х. М., Потапова Н А. СОРТУВАННЯ ТА ЙОГО ВПЛИВ НА АРХІТЕКТУРУ ПАМ'ЯТІ.....65

Діброва І. С., Вєтров О. С. СОРТУВАННЯ МАСИВІВ МЕТОДОМ БУЛЬБАШКИ.....68

Зимич А. П., Фриз І. В. ОГЛЯД МАТЕМАТИЧНИХ БІБЛІОТЕК PYTHON.....72

Ілик В. В., Потапова Н. А. СТЕКИ ТА АЛГОРИТМИ З ВИКОРИСТАННЯМ СТЕКІВ.....76

Костенко Р. О., Січко Т. В. СИСТЕМНИЙ АНАЛІЗ ТА МОДЕЛЮВАННЯ ЕКОНОМІЧНИХ КРИЗ.....78

Левченко М. Р., Вєтров О. С. РОЗГЛЯД ЗАСТОСУВАННЯ АЛГОРИТМІВ НА ГРАФАХ ДЛЯ ВИРІШЕННЯ ПРИКЛАДНИХ ЗАДАЧ.....81

Малінін А. О., Волонтир Л. О. ВИСОКОТЕХНОЛОГІЧНІ ІНСТРУМЕНТИ У ВЕБДИЗАЙНІ ТА ВЕБПРОГРАМУВАННІ ЯК ДВИГУНИ ІННОВАЦІЙ У ЦИФРОВОМУ ПРОСТОРІ.....85

Матіяш М. В., Потапова Н. А. СТРУКТУРИ ДАНИХ ТА ЇХ КЛАСИФІКАЦІЯ.....87

Мельник В. Р., Січко Т. В. ВИКОРИСТАННЯ МЕТОДІВ ОПТИМІЗАЦІЇ ДЛЯ ОБРОБКИ ТА АНАЛІЗУ ВЕЛИКИХ ДАНИХ.....88

Менделюк К. В., Потапова Н. А. ОЦІНКА СКЛАДНОСТІ АЛГОРИТМІВ.....90

Морозюк А. А., Січко Т. В. ВПЛИВ ЕКОНОМІЧНИХ ФАКТОРІВ НА ПОСТАНОВКУ ТА РОЗВ'ЯЗАННЯ ОПТИМІЗАЦІЙНИХ ЗАДАЧ 92

Поліщук В. С., Потапова Н. А. АЛГОРИТМ ПОШУКУ У ГЛИБИНУ 95

Поліщук О. С., Потапова Н. А. СУТНІСТЬ І СКЛАДНИКИ РЕКУРЕНТНИХ АЛГОРИТМІВ.....98

Попова К. О., Потапова Н. А. ВИКОРИСТАННЯ РЕКУРСИВНИХ ВІДНОШЕНЬ В АЛГОРИТМІЗАЦІЇ.....100

Семен О. Д., Ніколюк П. К. ВИКОРИСТАННЯ АЛГОРИТМУ A-STAR ДЛЯ ОПТИМІЗАЦІЇ ПОСТАЧАННЯ БОЄПРИПАСІВ.....103

Семен О. Д., Січко Т. В. АНАЛІЗ ДАНИХ У ПРИЙНЯТТІ РІШЕНЬ У СУЧАСНОМУ БІЗНЕСІ.....105

Семенюк А. М., Ніколюк П. К. ОПТИМІЗАЦІЯ ВИРОБНИЦТВА РЕАБІЛІТАЦІЙНИХ ЗАСОБІВ З ВИКОРИСТАННЯМ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ 107

Скороход О. М., Потапова Н. А. ГРАФИ ТА ПІДХОДИ ДО АЛГОРИТМІЧНИХ РІШЕНЬ НА ГРАФАХ.....111

Труханська В. О., Хмелівський Ю. С. ОГЛЯД НАПРЯМІВ ПРИХОВАНОЇ ПЕРЕДАЧІ ДАНИХ У ВІДЕОФАЙЛАХ 113

Труханська В. О., Хмелівський Ю. С. ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА МОВ ОБРОБКИ ДАНИХ.....115

Юстименко Є. А., Труханська В. О., Ніколюк П. К. МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ МІСЬКИХ ТРАНСПОРТНИХ СИСТЕМ.....117

Щербина Д. С., Ніколюк П. К. ЗНАХОДЖЕННЯ ОПТИМАЛЬНОГО МАРШРУТУ ПОСТАЧАННЯ ВІЙСЬКОВОЇ ПРОВІЗІЇ ЗА ДОПОМОГОЮ АЛГОРИТМУ ДЕЙКСТРИ.....119

Щербина Д. С., Січко Т. В. ПОНЯТТЯ ПРО ГЕНЕТИЧНІ АЛГОРИТМИ ТА ЇХ ЗАСТОСУВАННЯ В ЗАДАЧАХ ОПТИМІЗАЦІЇ 122

Юстименко Є. А., Хмелівський Ю. С. МЕТОДИ ОПТИМІЗАЦІЇ АЛГОРИТМІВ У ПРОГРАМУВАННІ 124

СЕКЦІЯ 2. ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ ТА МЕРЕЖІ

Вербіцький Р. Р., Гарматій Н. М. ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ ЗМІНИ КУРСУ АНГЛІЙСЬКОГО ФУНТА СТЕРЛІНГІВ ЩОДО ГРИВНІ МЕТОДОМ «БШ».....128

Кадикова А. А., Шостак І. В. ПЕРСПЕКТИВИ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДО РОЗВ'ЯЗАННЯ ЗАВДАНЬ АНАЛІЗУ РОЗМОВ ПОЛІЦЕЙСЬКИМИ СТРУКТУРАМИ 130

Богач Т. О., Потапова Н. А. ОСОБЛИВОСТІ ВИКОРИСТАННЯ АЛГОРИТМІВ ОПТИМІЗАЦІЇ В ЗАДАЧАХ ШТУЧНОГО ІНТЕЛЕКТУ 133

Журовський Я. О., Січко Т. В. МАШИННЕ НАВЧАННЯ В УПРАВЛІННІ ТРАНСПОРТНИМИ ПОТОКАМИ.....135

Журовський Я. О., Січко Т. В. ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ДІЯЛЬНОСТІ ОРГАНІЗАЦІЙ 138

Костенко Р. О., Хмелівський Ю. С. ОПТИМІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ ЗАПАСАМИ В РОЗДРІБНІЙ ТОРГІВЛІ 141

Куцмай В. Я., Волонтир Л. О. ОПТИМІЗАЦІЯ ПРОЦЕСІВ УПРАВЛІННЯ ПРОЄКТАМИ В ІТ-СФЕРІ.....144

Михайляк М. О., Січко Т. В. ВИКОРИСТАННЯ DJANGO REST FRAMEWORK ДЛЯ СТВОРЕННЯ API: ПЕРЕВАГИ ТА МОЖЛИВОСТІ 147

Поліщук А. М, Січко Т. В. ОПТИМІЗАЦІЯ МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ МЕРЕЖ.....149

Поліщук А. М, Хмелівський Ю. С. ОПТИМІЗАЦІЯ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ПОКРАЩЕННЯ ЇХНЬОЇ ПРОДУКТИВНОСТІ.....152

Рудь О. С., Січко Т. В. РОЛЬ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ У ВИРІШЕННІ ЕКОЛОГІЧНИХ ПРОБЛЕМ 156

Семенюк А. М., Січко Т. В. ОПТИМІЗАЦІЯ РОЗМІЩЕННЯ СОНЯЧНИХ ПАНЕЛЕЙ.....159

Слободянюк С. С., Січко Т. В. ВИКОРИСТАННЯ ШІ В ОПТИМІЗАЦІЇ ОБСЛУГОВУВАННЯ КЛІЄНТІВ.....162

Цегольник В. В., Ніколюк П. К. ПОБУДОВА МОДЕЛІ НА ОСНОВІ ТЕОРІЇ ГРАФІВ ДЛЯ ОПТИМІЗАЦІЇ ВІЙСЬКОВОЇ ЛОГІСТИКИ 164

СЕКЦІЯ 3. ЕКСПЕРТНІ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ ТА СИСТЕМИ ПІДРИМКИ ПРИЙНЯТТЯ

Ліваковський В. К., Січко Т. В. ВПЛИВ ШТУЧНОГО ІНТЕЛЕКТУ НА БІЗНЕС-ПРОЦЕСИ ТА ПРИЙНЯТТЯ РІШЕНЬ 169

Ліваковський В. К., Січко Т. В. ВДОСКОНАЛЕННЯ БІЗНЕС-ПРОЦЕСІВ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ 172

Стукан А. О., Хмелівський Ю. С. ВИКОРИСТАННЯ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ДЛЯ АНАЛІЗУ ПОПИТУ ТА РЕКОМЕНДАЦІЙ В E-COMMERCE ТА ІНШИХ СФЕРАХ 175

Стукан А. О., Хмелівський Ю. С. ПОБУДОВА РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ НА ОСНОВІ ШТУЧНХ НЕЙРОННИХ МЕРЕЖ.....176

Таран Н. М., Зелінська О. В. РЕКОМЕНДАЦІЙНА СИСТЕМА ПЛАНУВАННЯ РЕКЛАМНОЇ КАМПАНІЇ НА ОСНОВІ АНАЛІЗУ ДАНИХ ВЕБТРАФІКУ ТА ПОВЕДІНКИ КОРИСТУВАЧІВ 178

СЕКЦІЯ 4. БАЗИ ДАНИХ І ЗНАНЬ. BIG DATA

Алексюк В. В., Комаров В. Ф. MONGODB ЯК ЕФЕКТИВНИЙ ЗАСІБ У ЗАДАЧАХ ОБРОБКИ ТА АНАЛІЗУ ДАНИХ 181

Підруцький Д. А., Січко Т. В. BIG DATA В ОБРОБЦІ І АНАЛІЗІ ДАНИХ З РІЗНИХ СФЕР.....183

Сніжинський М. В., Січко Т. В. СИСТЕМА ОБРОБКИ ТА АНАЛІЗУ ДАНИХ У МЕДИЧНІЙ СФЕРІ НА ОСНОВІ ТЕХНОЛОГІЙ BIG DATA 187

Юстименко Є. А., Труханська В. О., Хмелівський Ю. С. ВИКОРИСТАННЯ БАЗ ДАНИХ В ІНФОРМАЦІЙНИХ СИСТЕМАХ.....189

Шафорост В. В., Корнієнко К. К., Хмель А. Є., Комаров В. Ф. МОДЕЛЮВАННЯ ВЕЛИКИХ НАБОРІВ ДАНИХ 192

Ватаманеску С. К., Потапова Н. А. ОПТИМІЗАЦІЯ АЛГОРИТМІВ ДЛЯ ВЕЛИКИХ ОБСЯГІВ ДАНИХ.....195

Калько Д. Р., Гончар В. М. ЗАСТОСУВАННЯ ГРАФОВОЇ БАЗИ ДАНИХ У СОЦІАЛЬНИХ МЕРЕЖАХ.....198

Козачок А. О., Вєтров О. С. СТИСНЕННЯ ДАНИХ ТА ЇХ ВПЛИВ НА ШВИДКОДІЮ АЛГОРИТМІВ ОБРОБКИ 201

Коновалюк І. Л., Зелінська О. В. ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ БАЗ ДАНИХ 203

Лаптєва М. А., Гончар В. М. РІЗНИЦЯ МІЖ SQL І NOSQL: ПЕРЕВАГИ ТА ОБМЕЖЕННЯ РЕЛЯЦІЙ НИХ БАЗ ДАНИХ 206

Маруняк А. О., Гончар В. М. КЛАСИФІКАЦІЯ ТА ЗАСТОСУВАННЯ КЛЮЧІВ У РЕЛЯЦІЙНИХ БАЗАХ ДАНИХ 209

Назаренко М. С., Зелінська О. В. ОПТИМІЗАЦІЯ БАЗ ДАНИХ ДЛЯ АНАЛІЗУ СОЦІАЛЬНИХ МЕРЕЖ 211

Поліщук В. С., Зелінська О. В. АДМІНІСТРАТОР БАЗ ДАНИХ 214

Поліщук Д. О., Січко Т. В. МЕТОДИ ОПТИМІЗАЦІЇ ВЕЛИКИХ ДАНИХ 216

Проценко А. С., Гончар В. М., СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ REDIS: ПЕРЕВАГИ ТА НЕДОЛІКИ.....218

Шевцов М. В., Гончар В. М. ВИКОРИСТАННЯ КРИПТОГРАФІЇ ДЛЯ ЗАХИСТУ ДАНИХ У БАЗІ ДАНИХ.....221

СЕКЦІЯ 5. ПРИКЛАДНІ АСПЕКТИ ОБРОБКИ ДАНИХ В ІНФОРМАЦІЙНИХ СИСТЕМАХ

Гарматій Н. М., Сербін В. С. ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ В ЕКОНОМІЧНИХ СИСТЕМАХ МАСОВОГО ОБСЛУГОВУВАННЯ 225

Бойко У. В. ВИЯВЛЕННЯ ФАКТОРІВ, ЩО ВПЛИВАЮТЬ НА РЕЙТИНГ І КАСОВІ ЗБОРИ ФІЛЬМУ МЕТОДАМИ DATA SCIENCE 228

Бушменьов В. Є., Потапова Н. А. КЛАСИФІКАЦІЯ ФОТОГРАФІЙ ЗА ОЗНАКОЮ МЕДИЧНОЇ МАСКИ З ВИКОРИСТАННЯМ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ 231

Векленко С. Ю., Потапова Н. А. АНАЛІЗ ДАНИХ ПІДЧАС ОЦІНЮВАННЯ ТЕНДЕНЦІЙ ІНВЕСТИЦІЙНИХ ПРОЦЕСІВ У ЗАКЛАДАХ МЕДИЧНОЇ ГАЛУЗІ.....234

Демашкевич А. В., Антонов Ю. С. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРИХОВУВАННЯ ПОВІДОМЛЕННЯ В ЦИФРОВИХ ЗОБРАЖЕННЯХ ЗА ДОПОМОГОЮ МЕТОДУ LSB 236

Колосова К. К., Потапова Н. А. РОЛЬ SEO У ПРОСУВАННІ ТА ПІДВИЩЕННІ ВІДВІДУВАНОСТІ ВЕБСАЙТІВ ІТ-КОМПАНІЙ 240

Огороднік М. О., Потапова Н. А. РОЛЬ INFLUENCE-МАРКЕТИНГУ У СФЕРІ УПРАВЛІННЯ ІТ-ПРОЄКТАМИ.....242

Римар П. В., Сеник І. О. ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ MAPLE ПІД ЧАС ПОБУДОВИ ГРАФІКІВ ФУНКЦІЙ 244

Чайковський П. А., Потапова Н. А. ВИКЛИКИ ДИСТАНЦІЙНОГО УПРАВЛІННЯ ІТ-ПРОЄКТАМИ 246

Ярош О. Л., Бабаков Р. М. КРОСПЛАТФОРМОВИЙ API ДЛЯ СИСТЕМИ РОЗПОДІЛУ ЗАВДАНЬ ТА АНАЛІЗУ ПРОДУКТИВНОСТІ ВИКОНАВЦІВ 249

Гуменюк К. В., Сеник І. О. ВАРІАНТИ ВИКОРИСТАННЯ NLP 251

Дорофєєв Є. О., Вєтров О. С. АЛГОРИМ ЗЛИТТЯ В МОВІ ПРОГРАМУВАННЯ PYTHON 254

Короленко С. О., Волонтир Л. О. СУЧАСНІ РЕАЛІЇ ТА ЦІЛІ РОЗШИРЕННЯ ШТУЧНОГО ІНТЕЛЕКТУ 257

Мигун Р. С., Горяшин А. С. РЕАЛІЗАЦІЯ БЛОКЧЕЙН-ТЕХНОЛОГІЙ З ВИКОРИСТАННЯМ PYTHON ДЛЯ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ ТА ЦІЛІСНОСТІ ДАНИХ.....259

Мисак М. П., Бабаков Р. М. ОБРОБКА ДАНИХ У МОБІЛЬНІЙ ІНФОРМАЦІЙНО-ДОВІДНИКОВІЙ СИСТЕМІ ДЛЯ СТУДЕНТІВ ФАКУЛЬТЕТУ 264

Михайляк М. О., Ніколюк П. К. МОДЕЛЮВАННЯ ВПЛИВУ ФАКТОРІВ ПРИРОДОКОРИСТУВАННЯ НА ЕКОЛОГІЮ.....266

Підруцький Д. А., Хмелівський Ю. С. АНАЛІЗ ПЕРСПЕКТИВ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ 270

Рудь О. С., Хмелівський Ю. С. РОЛЬ СТАТИСТИЧНОГО НАВЧАННЯ У ВДОСКОНАЛЕННІ МЕДИЧНИХ ЗАСОБІВ ТА ТЕХНОЛОГІЙ 272

Сіклічук А. С., Січко Т. В. ТЕСТУВАННЯ API ЯК ІНСТРУМЕНТ ВИЯВЛЕННЯ ПОМИЛОК У РОБОТІ КЛІЄНТ-СЕРВЕРНИХ СИСТЕМ 275

Юстименко Є. А., Труханська В. О., Ніколюк П. К. ПАТЕРНИ ПРОЄКТУВАННЯ В ПРОГРАМУВАННІ.....279

Цегольник В. В., Хмелівський Ю. С. ОПТИМІЗАЦІЯ ВИКОРИСТАННЯ СОНЯЧНОЇ ЕНЕРГІЇ ЗА ДОПОМОГОЮ МЕТОДІВ ОПТИМІЗАЦІЇ.....281

Чемес В. С., Вєтров О. С. ПОРІВНЯННЯ АЛГОРИТМІВ ПОШУКУ НАЙКОРОТШОГО ШЛЯХУ 284

Юстименко Є. А., Волонтир Л. О. МОЖЛИВОСТІ ВИКОРИСТАННЯ РАНГОВОЇ КОРЕЛЯЦІЇ В РЕАЛЬНОМУ ЖИТТІ.....287

Юстименко Є. А., Сеник І. О. МОДЕЛІ НА ОСНОВІ ТЕОРІЇ ГРАФІВ В ЛОГІСТИЦІ.....289

Яценко В. В., Вєтров О. С. ОГЛЯД АЛГОРИТМІВ ЧИТАННЯ ТА ЗАПИСУ В РЕЛЯЦІЙНИХ БАЗАХ ДАНИХ ТА СПОСОБИ ЇХ ОПТИМІЗАЦІЇ.....291

СЕКЦІЯ 1
МЕТОДИ ОБРОБКИ І АНАЛІЗУ ДАНИХ

UDC 004.932.72
Shulhin O., graduate student in Computer Science,
Shtovba S., professor at the Department of Information Technologies

COMPARATIVE ANALYSIS OF MODERN ONLINE TOOLS FOR OBJECT RECOGNITION IN IMAGES

Vasyl' Stus Donetsk National University, Vinnytsia, Ukraine

There are many practical tasks that require object recognition in images. Various approaches are used to solve this task, typically involving neural networks [1]. These networks have been pre-trained on a specific set of reference images, with corresponding objects assigned to specific classes.

In this work, an analysis of 4 online tools providing functionality for object recognition in images has been conducted. The tools are as follows: Amazon Rekognition, Google Cloud Vision, Microsoft Azure AI Vision Studio, and Imagga. To investigate and assess their performance, the graphical interface of each tool's website was used. It is worth adding that these services allow users to interact with them both through their graphical interfaces and via API.

To analyze the quality of object recognition, 8 images were selected. These images cover a wide range of classes and exhibit diversity. The types of images can be described by the following list: street photo, interior, exterior, still life, animals, and sports (Figure 1).

Figure 1. The testing images

The recognition results of the «still life» image type for each tool are shown in Figure 2.

To calculate the metric for evaluating recognition quality, we will apply the following approach [2]. For each tool and each image, we will limit the size of identified objects to the top 20 in descending order of the system's confidence in the accuracy of the prediction. Confidence is represented as a percentage value from 0 to 100. We will analyze the recognition quality as follows: for a correct prediction we add the confidence value. And for an incorrect prediction we subtract it. The resulting sum is divided by the number of recognized objects. The final assessment will be the sum of the tool's results across all images divided by the number of images:

$$MeanQuality = \frac{1}{8}\sum_{i=1}^{8}\frac{1}{L_i}\sum_{j=1}^{L_i} P_{ij} \cdot sign(P_{ij}), \quad (1)$$

where L_i denotes the number of recognized objects for the *i*-th image;

P_{ij} denotes the confidence value of the tool for the *j*-th object of the *i*-th image;

$sign(P_{ij})$ is a function that takes a value of 1 in the case of correct recognition and –1 in the case of wrong recognition.

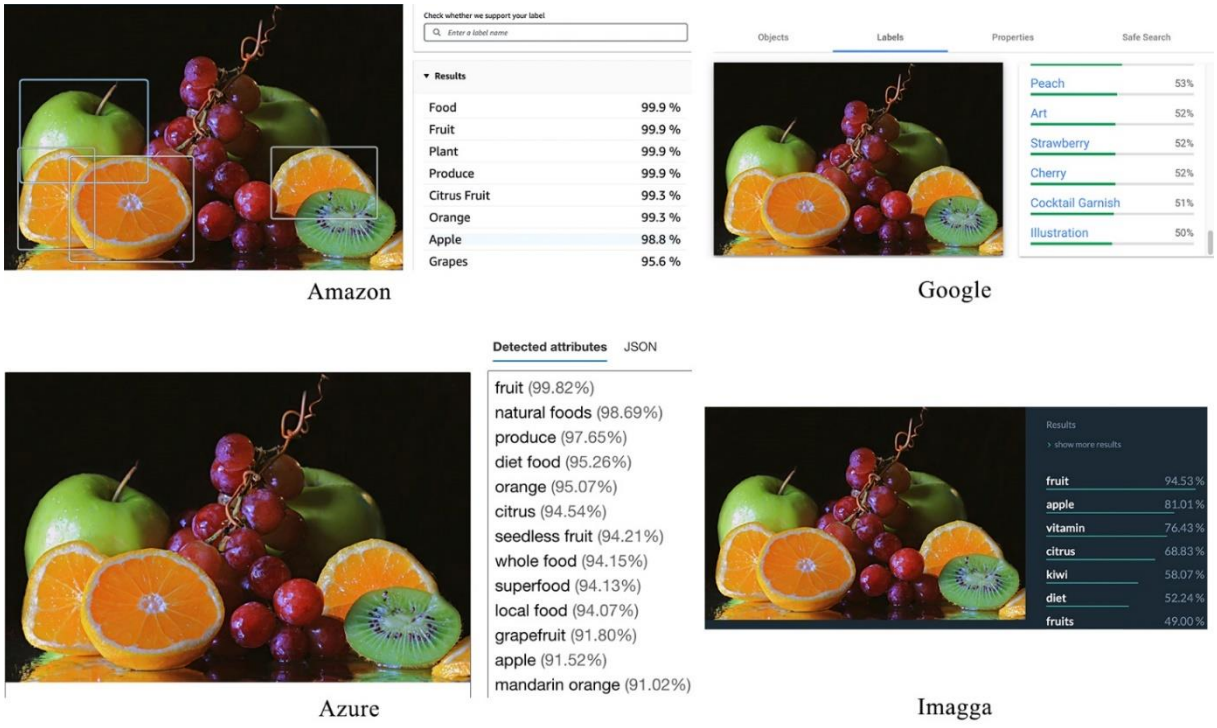


Figure 2. Recognition results by online tools

The calculated metric is presented in Table 1. The higher the metric value, the higher the tool's rating.

Amazon and Azure are considered the best. Still life turned out to be the most challenging for all, except Imagga (we don't see any meaningful drop). However, Imagga makes significant errors for images like Street photo 1 and Animals, which are easy for other tools. This may indicate that Imagga has been trained using different

methods and/or datasets. Such characteristics of Imagga show that it could be considered as a candidate for inclusion in ensemble models. However, this hypothesis needs further verification.

Table 1 – Results of the recognition quality analysis

Image	Amazon	Google	Azure	Imagga
Street photo 1	88,46	43,65	67,78	8,40
Street photo 2	33,43	72,35	63,02	19,35
Interior 1	55,46	9,8	66,45	28,04
Interior 2	67,29	8,7	85,56	31,83
Exterior	54,27	64,25	49,33	23,54
Sports	34,61	52,05	76,46	22,94
Animals	55,28	44,9	70,03	12,70
Still life	18,86	-5,6	27,80	26,54
Mean Quality	50,96	36,26	63,30	21,67

In addition, let us visualize the correctness of the detection for each tool based on all images. By x-axis we show the sum of confidence value for correct detections, by y-axis – the sum of confidence value for false detections. Each point on the chart represents the result for one image (Figure 3).

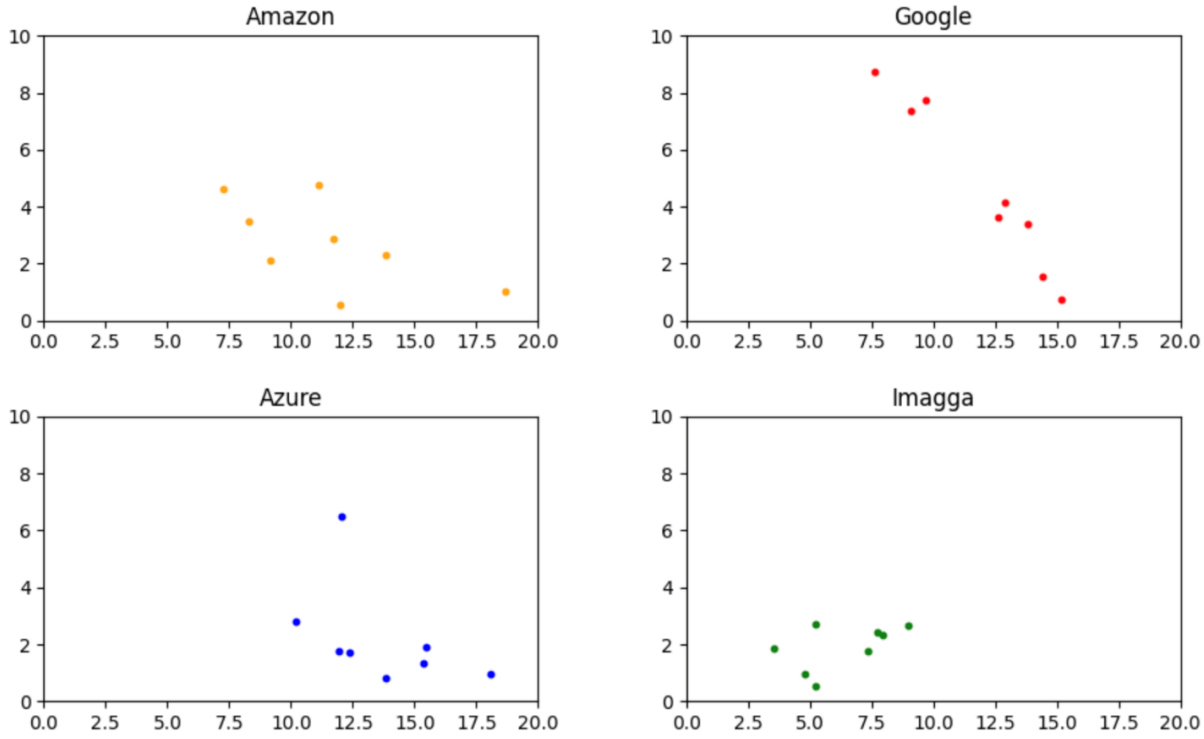


Figure 3. Visualization of the detection correctness for all tools

As we can see, all tools except Google have pretty tight scattering. This indicates that the quality of the detection does not vary significantly from one image to another.

In contrast Google scattering is linear which means that the quality of the detection depends on the image and is not as stable as observed with other tools. Azure scattering is located in the lower right corner which represents its highest ranking in current analysis.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM,* 60(6), 84–90.

2. Rachev, S. T., Stoyanov, S., Fabozzi, F. J. (2008). Advanced stochastic models, risk assessment, and portfolio optimization: The ideal risk, uncertainty, and performance measures. Optimization (p. 382).

УДК 681.3.06

Бобошко В. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Штовба С. Д., д-р техн. наук, професор, професор кафедри інформаційних технологій

МЕТРИКА СХОЖОСТІ НАВЧАЛЬНИХ ДИСЦИПЛІН ЗА ЇХ ВНЕСКОМ У КОМПЕТЕНТНОСТІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Після прийняття в 2015 р. нового закону «Про вищу освіту» освітня діяльність в університетах та в інших закладах вищої освіти розпочала формуватися навколо освітніх програм. Освітня програма – це документ, що визначає змістовний та процедурний складники навчання студентів та інших здобувачів вищої освіти. Освітні програми розроблюють за компетентісним підходом з урахуванням вимог освітніх стандартів за спеціальністю. Освітні стандарти є доволі загальними – вони містять перелік обов'язкових компетентностей та програмних результатів навчання. Натомість освітні програми, окрім компетентностей та програмних результатів навчання, містять і перелік освітніх дисциплін, обсяг кредитів та матриці впливу освітніх компонентів на програмні результати навчання й на компетентності.

Метою дослідження є розробка метрики для оцінювання схожості двох дисциплін з освітніх програм з однієї і тієї ж спеціальності. Метрика принципово відрізняється від традиційного суб'єктивного порівняння змісту дисциплін освітніми експертами. Метрика не враховує змістовне наповнення дисциплін, а розраховує схожість на основі внеску дисципліни у формування компетент-

ностей або програмних результатів навчання. Отже, метрика дає змогу у деякий статистичний спосіб ідентифікувати схожі дисципліни освітніх програм. Вона є аналогом *word2vec*, але працює у просторі «дисципліни – програмні результати навчання» та «дисципліни – компетентості». Цим вона відрізнятиметься від експертного підходу.

Початковими даними оберемо освітні програми за деякою спеціальністю. Оцінювання проведемо за освітніми програмами, які НАЗЯВО вважає якісними за критерієм 2 «Структура та зміст освітньої програми». За цим критерієм освітні програми мають бути оцінені агентством на рівень *A* або *B*.

Для опису дисципліни в просторі компетентностей вважатимемо, що кредити дисципліни рівномірно розподілено на усі компетентності, у формуванні яких вона задіяна. Наприклад, якщо деяка дисципліна обсягом у 6 кредитів задіяна у формуванні двох компетентностей, тоді вважаємо, що на кожну компетентність припадає 3 кредити. Аналогічно можна описати дисципліни в просторі програмних результатів навчання.

Схожість дисциплін X та Y визначимо за модифікованою метрикою Чекановського [1]:

$$Sim(X,Y) = \frac{sum(X \cap Y)}{sum(X) + sum(Y) - sum(X \cap Y)},$$

де перетин реалізується операцією мінімуму.

Наведемо такий приклад розрахунку схожості двох дисциплін. Нехай дисципліна X формує першу, другу та п’яту компетентності, а дисципліна Y – першу, другу, третю та четверту компетентності. Обсяг дисципліни X – 6 кредитів, а дисципліни Y – 4 кредити. Тоді у просторі із п’яти компетентностей дисципліни описуються так: $X = (2,\ 2,\ 0, 0, 2)$ та $Y = (1,\ 1,\ 1,\ 1, 0)$. Їх перетин дорівнює: $X \cap Y = (1, 1, 0, 0, 0)$. За цих даних коефіцієнт схожості становить:

$$Sim(X,Y) = \frac{2}{6 + 4 - 2} = 0.25.$$

За запропонованим підходом проаналізуємо акредитовані освітні програми за спеціальністю 122 Комп’ютерні науки магістерського рівня. Розглядатимемо лише освітні програми, які розроблені після затвердження освітнього стандарту за цією спеціальністю. Розрахунки здійснимо за компетентностями та програмними результатами навчання, які є у стандарті.

У базі даних НАЗЯВО нами виявлено 17 таких освітніх програм з кількістю освітніх компонентів від 10 до 22 (рис. 1).

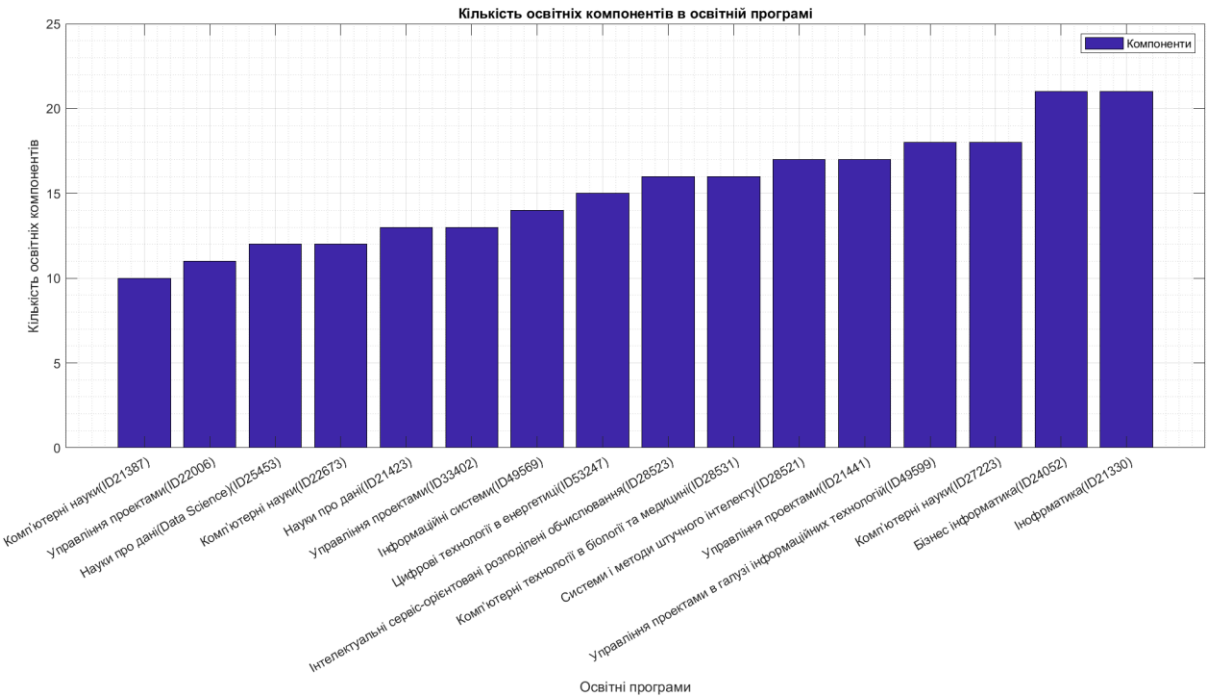


Рисунок 1. Кількість освітніх компонентів в аналізованих освітніх програмах

Із множини освітніх компонентів відберемо навчальні дисципліни, опишемо кожну з них у просторі стандартних програмних результатів навчання та просторі стандартних компетентностей, і розрахуємо схожості дисциплін. Таких пар дисциплін виявилося понад 9 тисяч. Рангові розподіли коефіцієнтів схожості дисциплін наведено в напівлогарифмічному форматі на рис. 2–3. Розподіли подібні до паретівських. У просторі компетентностей ідентифіковано 38 пар ідентичних дисциплін з коефіцієнтом схожості 1, а у просторі програмних результатів навчання таких пар ідентичних дисциплін виявилося 27. Частина з цих пар – це абсолютно однакові дисципліни, які викладаються в одному і тому ж університеті, але на різних освітніх програмах (рис. 4).

З рис. 4 видно, що є пари дисциплін, які за назвами доволі далекі одна від одної, але відповідно до освітніх програм формують ідентичні множини компетентностей з однаковим ресурсом кредитів. Наприклад, пара дисциплін *Фінансова аналітика* та *Методи соціальних досліджень* з освітньої програми «Бізнес інформатика» Київського національного університету імені Тараса Шевченка, пара дисциплін *Професійна та корпоративна етика* і *Психологія комунікації в галузі інформаційних технологій* з освітньої програми «Інформатика» Київського національного університету імені Тараса Шевченка, пара дисциплін *Комп'ютерна лінгвістика* та *Математична теорія ігор* з освітньої програми «Комп'ютерні науки» Національного університету «Києво-Могилянська академія». З чим це пов'язано – з помилками компетентністного проєктування освітньої програми, з занадто широкими компетентностями в освітньому стан-

дарті чи з не завжди коректним припущенням, що дисципліни вносять однакову частку кредитів в усі задіяні компетентності – потрібно додатково аналізувати.

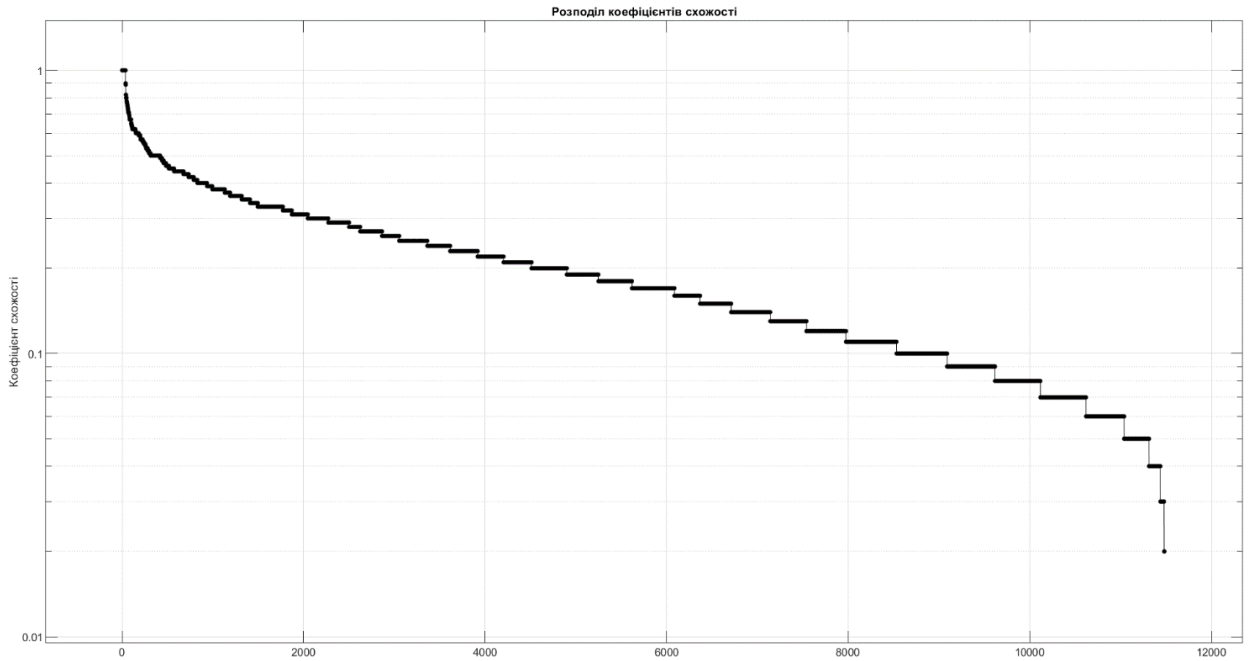


Рисунок 2. Розподіл схожості пар дисциплін за їх внеском у компетентності

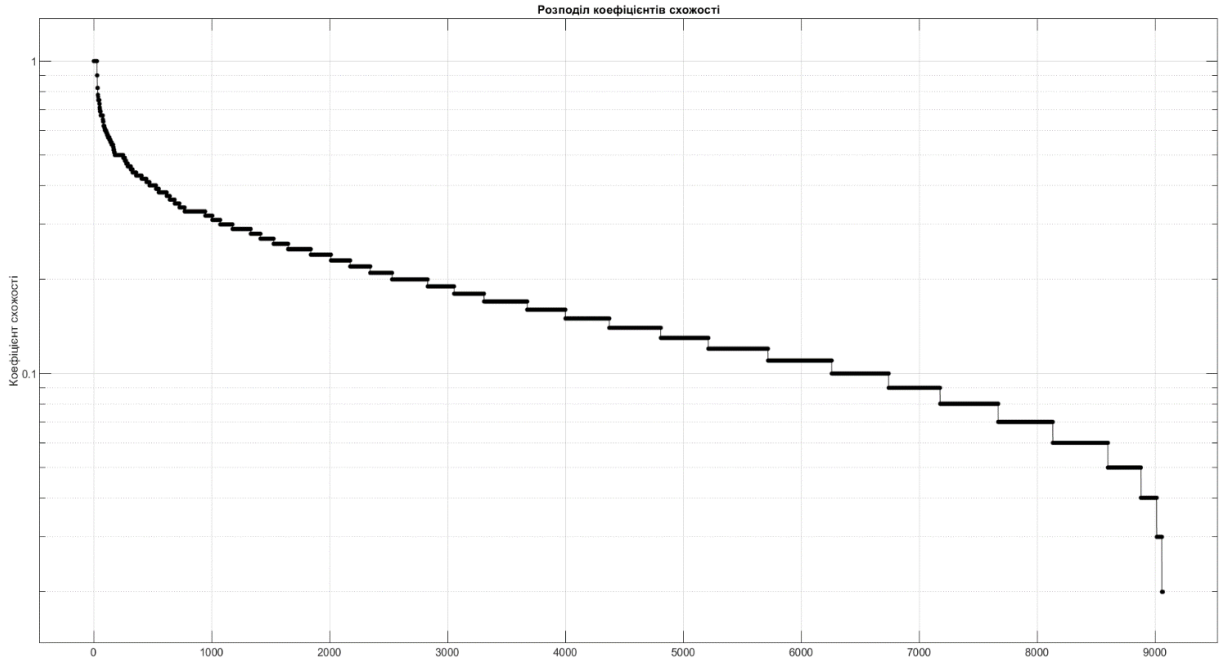


Рисунок 3. Розподіл схожості пар дисциплін за їх внеском у програмні результати навчання

	Discipline1	Discipline2
1	Фінансова математика (Бізнес інформатика КНУ)	Методи соціальних досліджень (Бізнес інформатика КНУ)
2	Професійна та корпоративна етика (Інформатика КНУ)	Психологія комунікації в галузі інформаційних технологій (Інформатика КНУ)
3	Комп'ютерна лінгвістика (Комп'ютерні науки КМА)	Математична теорія ігор (Комп'ютерні науки КМА)
4	Інтелектуальна власність та патентознавство (Інтелектуальні сервіс-орієнтовані ро…	Інтелектуальна власність та патентознавство (Комп'ютерні технології в біології та …
5	Інтелектуальна власність та патентознавство (Інтелектуальні сервіс-орієнтовані ро…	Інтелектуальна власність та патентознавство (Цифрові технології в енергетиці КПІ)
6	Інтелектуальна власність та патентознавство (Інтелектуальні сервіс-орієнтовані ро…	Інтелектуальна власність та патентознавство (Системи і методи штучного інтелект…
7	Сталий інноваційний розвиток (Інтелектуальні сервіс-орієнтовані розподілені обчи…	Сталий інноваційний розвито (Комп'ютерні технології в біології та медицині КПІ)
8	Сталий інноваційний розвиток (Інтелектуальні сервіс-орієнтовані розподілені обчи…	Сталий інноваційний розвиток (Цифрові технології в енергетиці КПІ)
9	Сталий інноваційний розвиток (Інтелектуальні сервіс-орієнтовані розподілені обчи…	Сталий інноваційний розвиток (Системи і методи штучного інтелекту КПІ)
10	Практичний курс іноземної мови для ділової комунікації (Інтелектуальні сервіс-оріє…	Практичний курс іноземної мови для ділової комунікації (Комп'ютерні технології в …
11	Практичний курс іноземної мови для ділової комунікації (Інтелектуальні сервіс-оріє…	Практичний курс іноземної мови для ділової комунікації (Цифрові технології в ене…
12	Практичний курс іноземної мови для ділової комунікації (Інтелектуальні сервіс-оріє…	Практичний курс іноземної мови для ділової комунікації (Системи і методи штучно…
13	Розробка стартап-проєктів (Інтелектуальні сервіс-орієнтовані розподілені обчислю…	Розробка стартап-проєктів (Цифрові технології в енергетиці КПІ)
14	Розробка стартап-проєктів (Інтелектуальні сервіс-орієнтовані розподілені обчислю…	Розробка стартап-проєктів (Системи і методи штучного інтелекту КПІ)
15	Педагогіка вищої школи (Інтелектуальні сервіс-орієнтовані розподілені обчислюва…	Педагогіка вищої школи (Комп'ютерні технології в біології та медицині КПІ)
16	Педагогіка вищої школи (Інтелектуальні сервіс-орієнтовані розподілені обчислюва…	Педагогіка вищої школи (Цифрові технології в енергетиці КПІ)
17	Педагогіка вищої школи (Інтелектуальні сервіс-орієнтовані розподілені обчислюва…	Педагогіка вищої школи (Системи і методи штучного інтелекту КПІ)
18	Обробка надвеликих масивів даних (Інтелектуальні сервіс-орієнтовані розподілені …	Оброблення надвеликих масивів даних (Комп'ютерні технології в біології та медиц…
19	Обробка надвеликих масивів даних (Інтелектуальні сервіс-орієнтовані розподілені …	Обробка надвеликих масивів даних (Цифрові технології в енергетиці КПІ)
20	Обробка надвеликих масивів даних (Інтелектуальні сервіс-орієнтовані розподілені …	Обробка надвеликих масивів даних (Системи і методи штучного інтелекту КПІ)
21	Інтелектуальна власність та патентознавство (Комп'ютерні технології в біології та …	Інтелектуальна власність та патентознавство (Цифрові технології в енергетиці КПІ)
22	Інтелектуальна власність та патентознавство (Комп'ютерні технології в біології та …	Інтелектуальна власність та патентознавство (Системи і методи штучного інтелект…
23	Сталий інноваційний розвито (Комп'ютерні технології в біології та медицині КПІ)	Сталий інноваційний розвиток (Цифрові технології в енергетиці КПІ)
24	Сталий інноваційний розвито (Комп'ютерні технології в біології та медицині КПІ)	Сталий інноваційний розвиток (Системи і методи штучного інтелекту КПІ)
25	Практичний курс іноземної мови для ділової комунікації (Комп'ютерні технології в б…	Практичний курс іноземної мови для ділової комунікації (Цифрові технології в ене…
26	Практичний курс іноземної мови для ділової комунікації (Комп'ютерні технології в б…	Практичний курс іноземної мови для ділової комунікації (Системи і методи штучно…
27	Педагогіка вищої школи (Комп'ютерні технології в біології та медицині КПІ)	Педагогіка вищої школи (Цифрові технології в енергетиці КПІ)
28	Педагогіка вищої школи (Комп'ютерні технології в біології та медицині КПІ)	Педагогіка вищої школи (Системи і методи штучного інтелекту КПІ)
29	Оброблення надвеликих масивів даних (Комп'ютерні технології в біології та медиц…	Обробка надвеликих масивів даних (Цифрові технології в енергетиці КПІ)
30	Оброблення надвеликих масивів даних (Комп'ютерні технології в біології та медиц…	Обробка надвеликих масивів даних (Системи і методи штучного інтелекту КПІ)
31	Інтелектуальна власність та патентознавство (Цифрові технології в енергетиці КПІ)	Інтелектуальна власність та патентознавство (Системи і методи штучного інтелект…
32	Сталий інноваційний розвиток (Цифрові технології в енергетиці КПІ)	Сталий інноваційний розвиток (Системи і методи штучного інтелекту КПІ)
33	Практичний курс іноземної мови для ділової комунікації (Цифрові технології в енер…	Практичний курс іноземної мови для ділової комунікації (Системи і методи штучно…
34	Розробка стартап-проектів (Цифрові технології в енергетиці КПІ)	Розробка стартап-проєктів (Системи і методи штучного інтелекту КПІ)
35	Педагогіка вищої школи (Цифрові технології в енергетиці КПІ)	Педагогіка вищої школи (Системи і методи штучного інтелекту КПІ)
36	Обробка надвеликих масивів даних (Цифрові технології в енергетиці КПІ)	Обробка надвеликих масивів даних (Системи і методи штучного інтелекту КПІ)
37	Візуалізація графічної та геометричної інформації (Цифрові технології в енергетиці…	Методи синтезу віртуальної реальності (Цифрові технології в енергетиці КПІ)
38	Математичне моделювання в IT проєктах (Управління проектами КНУ)	Математичне моделювання в IT проєктах (Управління проектами ID33402 КНУ)

Рисунок 3. Пари дисципліни, що формують ідентичний набір компетентностей

Список використаних джерел

1. Штовба С. Д., Петричко М. В. Тематичне моделювання науковців на основі їх інтересів у Google Scholar: *System research and information technologies.* 2021. № 2. С. 113–129.

УДК 004.6+005.7

Пустовойтенко В. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки ОС Магістр,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

МЕТОДИ ТА ТЕХНОЛОГІЇ ОБРОБКИ ДАНИХ ВЕТЕРИНАРНИХ КЛІНІК

Донецький національний університет імені Василя Стуса, м. Вінниця

Ветеринарні клініки, подібно до багатьох інших сфер, відчувають вплив розвитку сучасних технологій на свою роботу. Обробка даних та автоматизація

процесів обслуговування клієнтів стає невід'ємною частиною ветеринарної медицини. Можна виділити методи та технології, які сьогодні найбільш використовуються для поліпшення якості обслуговування ветеринарних клінік:

1. Електронні медичні картки та їх роль у покращенні медичного обслуговування.

2. Системи онлайн-запису та планування візитів.

3. Автоматизація фінансового обліку та оплати послуг.

4. Використання технологій Інтернету речей (IoT) для моніторингу стану тварин.

5. Системи нагадувань та спілкування з клієнтами.

6. Віддалений доступ до інформації через онлайн-портали.

7. Використання технологій штучного інтелекту (AI) та машинного навчання (ML).

Електронні медичні картки та їх роль у покращенні медичного обслуговування. Електронні медичні картки є основним складником систем обробки даних ветеринарних клінік. Створення електронної бази даних дає змогу зберігати і відстежувати історію лікування тварин, легко доступну для ветеринарів. Це забезпечує швидший доступ до важливої інформації та сприяє більш ефективній діагностиці та лікуванню.

Системи онлайн-запису та планування візитів. Запровадження онлайн-систем запису дає змогу клієнтам легко здійснювати запис на прийом через інтернет. Це не тільки зменшує очікування у черзі, але й полегшує процес планування робочого дня ветеринарної клініки. Такі системи допомагають автоматично нагадувати клієнтам про наближення часу їхнього візиту.

Автоматизація фінансового обліку та оплати послуг. Використання різноманітних методів оплати, як-от онлайн-платежі та мобільні додатки, спрощує процес оплати послуг. Автоматизовані системи фінансового обліку дають змогу ветеринарним клінікам ефективно вести облік прибутку та витрат, забезпечуючи фінансову стабільність.

Використання технологій Інтернету речей (IoT) для моніторингу стану тварин. Впровадження технологій Інтернету речей допомагає ветеринаріям моніторити стан тварин у реальному часі. Використання датчиків для вимірювання показників здоров'я, як-от температура тіла та пульс, дає змогу вчасно виявляти проблеми та швидко реагувати на них. Це сприяє покращенню діагностики та лікування тварин.

Системи нагадувань та спілкування з клієнтами. Нагадування про прийоми, вакцинації та інші процедури через SMS-повідомлення й електронні листи допомагають уникнути забутих візитів та покращують рівень клієнтської дисципліни. Системи спілкування дають змогу ветеринарним клінікам підтримувати постійний зв'язок з клієнтами, відповідати на їхні запитання та вирішувати можливі непорозуміння.

Віддалений доступ до інформації через онлайн-портали. Створення онлайн-порталів для клієнтів, де вони можуть переглядати медичну інформацію про своїх тварин, отримувати результати аналізів та звертатися до лікаря віддалено, забезпечує зручність і доступність інформації для власників тварин.

Використання технологій штучного інтелекту (AI) та машинного навчання (ML) у ветеринарії. Не менш важливим є використання технологій штучного інтелекту й машинного навчання, яке може значно поліпшити діагностику та лікування тварин. Алгоритми машинного навчання можуть аналізувати великі обсяги клінічних даних та допомагати ветеринаріям у точній діагностиці різних захворювань.

Отже, сучасні методи й технології обробки даних ветеринарних клінік відіграють важливу роль у поліпшенні обслуговування клієнтів та наданні якісної ветеринарної допомоги. Електронні системи, IoT, технології AI та інші інноваційні рішення сприяють оптимізації робочих процесів та покращенню якості надання ветеринарних послуг.

Список використаних джерел

1. Mitchell R. Web Scraping with Python: Collecting More Data from the Modern Web. 2nd Edition. O'Reilly, 2018. 290 p.

2. Heydt M. Python Web Scraping Cookbook: Over 90 proven recipes to get you scraping with Python, micro services, Docker and AWS. Packt Publishing, 2018. 364 p.

3. Beautiful Soup Documentation. URL: https://www.crummy.com/software/ BeautifulSoup/ bs4/doc/

УДК 004.912

Бездушний В. О., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Штовба С. Д., д-р техн. наук, професор, професор кафедри інформаційних технологій

ВИЯВЛЕННЯ ПРИХОВАНОЇ ЛАЙКИ В ТЕКСТОВИХ ПОВІДОМЛЕННЯХ ЗА АНАЛІЗОМ ВІЗУАЛЬНО ПОДІБНИХ СИМВОЛІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Приховані лайливі слова – це слова, що зазнали певних замін символів, але під час їх читання пересічний користувач легко розуміє їх лайливий сенс. Такі заміни роблять навмисно, щоб обійти автоматичні фільтри повідомлень у чатах, коментарях тощо. Для прикладу можна навести таку просту заміну

fool на *f00l*. У початковому слові замінено літери *oo* на нулі – *00*. Слова залишилися візуально дуже схожими, тому і *fool*, і *f00l* сприймається як лайливі з тотожними значеннями. Автоматичні фільтри, які базуються на посимвольному порівнянні слів, слово *fool* ідентифікують як лайливе, а слово *f00l* – ні, бо воно відсутнє у словниках образливої лексики.

Нами пропонується підхід до виявлення прихованих лайливих слів на основі аналізу візуально схожих символів. Якщо деяке слово вдається заміною одного чи кількох символів на візуально схожі символи перетворити на лайливе, тоді вважаємо, що приховане лайливе слово виявлено. Щоб реалізувати цю ідею, потрібно мати списки візуально подібних символів. Нами знайдено кілька матриць сплутувань для деякого набору літер англійського алфавіту, зокрема [1–4]. Ми агрегували матриці з цих робіт та доповнили їх відсутніми коефіцієнтами сплутувань для букв верхнього й нижнього регістрів та коефіцієнтами сплутувань цифр з буквами. Після цього виділили перелік найбільш схожих пар символів. Ці пари схожих символів і будемо аналізувати для виявлення спотворених лайливих слів. Фрагмент переліку найбільш схожих пар символів наведено нижче:

1 → l, i, I;
4 → a, A;
7 → t, T;
0 → o, u, O, U;
5 → S;
8 → B;
9 → g, q, R;
c → e;
L → I;
z → s;
G → O, C.

Також ми сформували словник базових лайливих слів, щоб було на що спиратись під час пошуку прихованої лайки. У базовий словник увійшло 169 слів.

Аналіз тексту відбувається у 3 етапи. Спочатку аналізоване речення розбивається на токени – окремі слова. Далі кожен токен направляється на грубу перевірку на схожість з словами із базового словника за відстанню Левенштейна. Якщо кількість елементарних перетворень аналізованого слова у базове лайливе слово не перевищує порогове значення, тоді запускається процедура тонкої перевірки. Тонка перевірка здійснюється, щоб встановити, чи можна перетворити аналізоване слово на лайливе заміною візуально схожих символів. Тонка перевірка здійснюється за бектрекінг-алгоритмом, який синтезує нові слова, враховуючи кожен символ із вхідного слова та потенційні заміни цих

символів на візуально схожі. Якщо синтез виявляється успішним, тоді аналізоване слово вважається приховано лайливим, і алгоритм повертає відповідне слово із базового словника.

Проілюструємо роботу алгоритму на такому реченні: *The ztubid fuekJng infant bItcb ass fock crawled into my ass the stupid fucking dense grass*. Це речення містить 8 лайливих слів, 4 із яких словникові і 4 – спотворені. Результат аналізу цього речення за запропонованим алгоритмом наведено на рис. 1. Із нього видно, що виявлено усі 8 лайливих слів. Лайливі слова зі словника було виявлено відразу без запуску бектрекінг-алгоритму. Тривалість аналізу такого речення у розробленому вебзастосунку становить 127 мілісекунд.

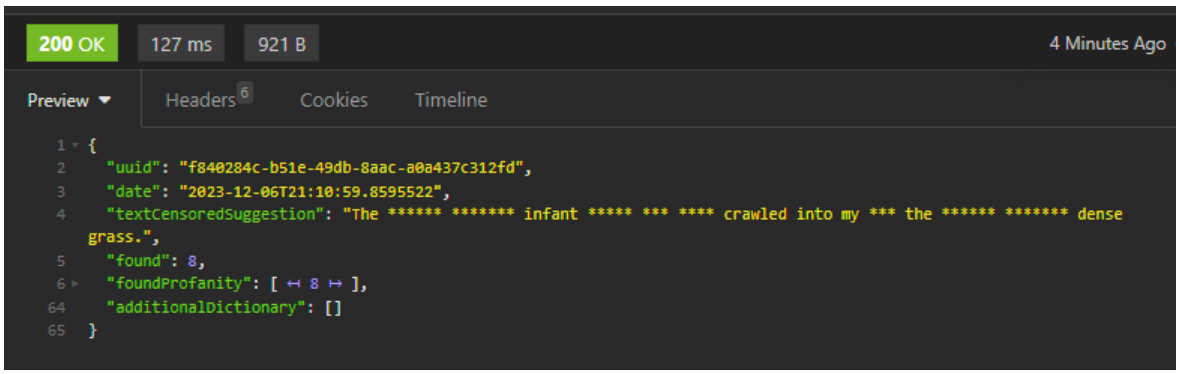


Рисунок 1. Результат аналізу тестового речення

Для порівняння з конкурентами згенеровано 3 такі тестові речення, що містять 6, 8 та 8 лайливих слів:

(1) *A fuck1ng man out a focklnq blue sweater fat ass sat at zh1tt1ng the st0p1d fuckinb desk*;

(2) *The ztubid fuekJng infant bItcb ass fock crawled into my ass the stupid fucking dense grass*;

(3) *Dude tried to fuekinq impress her ugly bitch face and fuekinb failed because the stubJd fucking c00n of is an Idjot.*

Результати тестування (табл. 1) свідчать, що запропонований алгоритм значно переважає конкурентів.

Таблиця 1 – Порівняння з конкурентами

Сервіс	**Кількість виявлених лайливих слів**			**Рівень виявлення лайливих слів**
	речення (1)	**речення (2)**	**речення (3)**	
Readable.com	4	4	3	50 %
Sightengine	3	5	2	45,5 %
Webpurify	3	4	3	45,5 %
PurgoMalum	3	3	3	41 %
Запропонований	5	6	7	82 %

Запропонований алгоритм протестовано на розміченій базі коментарів з *Kaggle Toxic Comment Classification Challenge* (табл. 2). Перевірка здійснювалася для коротких коментарів довжиною до 500 символів. Із таблиці видно, що запропонований алгоритм рідко коли виявляє лайливе слово в нейтральних коментарях – коментарях третьої групи. Образливі та погрозливі коментарі – коментарі з другої групи у 75,6 % випадків містять лайливе слово, яке виявляється запропонованим алгоритмом. Більшість токсичних коментарів першої групи також містить правильні або спотворені лайливі слова. Отже, виявлені за запропонованим алгоритмом приховані лайливі слова можуть розглядатися як додатковий інформативний атрибут для створення моделей виявлення токсичного контенту в соціальних мережах. Принциповою відмінністю запропонованого алгоритму є те, що він базується на аналітичному підході, а не на індуктивному. Тому він здатен виявляти не лише відомі спотворені лайливі слова, але і нові, які будуть створені на основі нових варіантів заміни символів.

Таблиця 2 – Результати тестування на датасеті Kaggle Toxic Comment

Група	**Тип коментаря**	**Кількість коментарів**	**Виявлено лайливе слово**	**Не виявлено лайливого слова**
1	Toxic	9 812	5 975 61 %	3 837 39 %
	Severe toxic			
2	Obscene	6 822	5 156 75,6 %	1 666 24,4 %
	Threat			
	Insult			
	Identity hate			
3	Neutral	50 030	1 604 3,2 %	48 426 96,8 %

Список використаних джерел

1. Loomis, J. M. (1982). Analysis of tactile and visual confusion matrices. *Perception & Psychophysics, 31,* 41–52.

2. Geyer, L. H. (1977). Recognition and confusion of the lowercase alphabet. *Perception & Psychophysics, 22,* 487–490.

3. Townsend, J. T. (1971). Theoretical analysis of an alphabetic confusion matrix. *Perception & Psychophysics, 9,* 40–50.

4. Dunn-Rankin, P., Leton, D. A., Shelton, V. F. (1968). Congruency factors related to visual confusion of English letters. *Perceptual and Motor Skills, 26(2),* 659–666.

УДК 004.93:621.396(043.2)

Громович О. С., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки,

Римар П. В., старший викладач кафедри інформаційних технологій

ДОСЛІДЖЕННЯ МЕТОДІВ ЗБОРУ, ОБРОБКИ ТА АНАЛІЗУ ДАНИХ ІЗ МОБІЛЬНИХ СЕНСОРІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Мобільні сенсори перетворили нашу взаємодію з технологіями на невербальний обмін інформацією завдяки їх потужності та непомітності. Ці сенсори постійно забезпечують нам потік інформації про наші дії і навколишній світ [1]. Сучасні смартфони та планшети потребують їх, щоб реагувати на дотик, ідентифікувати наші рухи та відтворювати звук. Їх функції не обмежуються цим, і разом з розвитком технологій їх можливості розширюються.

Швидкий розвиток мобільних технологій та збільшення кількості смартфонів призвели до значного зростання обсягу та доступності даних, зібраних з допомогою мобільних сенсорів. Цей феномен підтримується ключовими аргументами:

1. Мобільні смартфони як міні-лабораторії: смартфони перетворилися з комунікаційних засобів у повноцінні міні-лабораторії для збору різноманітної інформації завдяки різним сенсорам, які вимірюють рух, звук, світло, температуру та інші параметри.

2. Зростання кількості смартфонів і необхідність удосконалення методів збору та аналізу даних: збільшення користувачів смартфонів створює потреби у вдосконаленні методів збору та аналізу даних, оскільки користувачі не лише споживають, а й створюють значні обсяги інформації.

3. Широкий спектр застосувань мобільних сенсорів: вони застосовуються в різних галузях, як-от медицина, транспорт, спорт, наука та інші, розширюючи їх функціональність від моніторингу здоров'я до покращення систем навігації та досліджень у науці.

Насамперед мета цього дослідження полягає в систематичному аналізі інформації про методи збору, обробки та аналізу даних з мобільних сенсорів. Цей аналіз має на меті визначити поточний стан справ у цій галузі та ідентифікувати можливість покращення й оптимізації наявних методів. А також розробку нових підходів, методів чи алгоритмів, які можуть бути використані для збору, обробки й аналізу даних з мобільних сенсорів з більшою ефективністю, точністю та надійністю.

Науковий внесок. Дослідження може сприяти розвитку наукових знань у галузі обробки сигналів, інформатики та інших наукових дисциплін. Усі вдос-

коналення методів та технологій додадуться до загального фонду знань і можуть бути використані в подальших дослідженнях.

Практична користь. В результаті дослідження можуть бути розроблені практичні інструменти та рекомендації для використання даних з мобільних сенсорів у практичних задачах. Це може допомогти покращити якість життя людей, розвивати нові технології та підвищувати конкурентоспроможність підприємств і організацій.

Це дослідження є важливим не лише з технічного, але і з наукового та практичного погляду. Воно спрямоване на вирішення конкретних проблем та відкриття нових можливостей у використанні даних з мобільних сенсорів, що має великий потенціал для покращення сучасного інформаційного середовища.

З метою виявлення можливих проблем і областей для покращення було проведено аналіз методів збору даних з мобільних сенсорів, інструментів обробки даних та їх аналізу. Також здійснено докладний ретельний аналіз приватності й безпеки процесів збору, обробки та аналізу даних. Ці аспекти становлять основу для подальших досліджень та розробки нових методів і технологій у сфері аналізу даних з мобільних пристроїв.

Однією з основних проблем є висока затрата ресурсів [2]. Деякі алгоритми обробки та аналізу даних можуть бути високообчислювальними і вимагати значних обчислювальних ресурсів, що може бути проблематичним для мобільних пристроїв з обмеженими ресурсами.

Іншою важливою проблемою є недостатня точність. Деякі сенсори можуть мати обмежену роздільну здатність, що впливає на точність і якість зібраних даних.

Конфіденційність і безпека також є важливими питаннями [3]. Відсутність сильних механізмів автентифікації та авторизації може призвести до несанкціонованого доступу до даних або систем обробки даних.

Ці відкриття підкреслили необхідність розробки нових методів і технологій, які можуть вирішити наявні проблеми та вдосконалити процеси збору, обробки й аналізу даних з мобільних сенсорів. Особливу увагу варто приділити розробці більш ефективних алгоритмів для обробки та аналізу даних, які здатні зменшити обчислювальні витрати й підвищити точність результатів, а також поліпшенню механізмів безпеки для захисту даних від несанкціонованого доступу та можливих зловживань.

Аналізуючи роботи інших дослідників, можна запропонувати декілька наступних вдосконалень [4–5].

По-перше, можна розробити алгоритми для динамічного управління частотою збору даних залежно від контексту користувача та стану системи.

По-друге, важливо розробити методи для ефективного управління приватністю та безпекою даних під час збору й передачі даних до хмарних сервісів.

По-третє, можна розглянути можливість використання машинного навчання й інших методів аналізу даних безпосередньо на мобільних пристроях для виявлення цінних закономірностей та отримання нової корисної інформації без необхідності передачі великих обсягів даних до хмари.

Нові методи збору даних з мобільних сенсорів можуть відкрити нові можливості для розробки інноваційних мобільних додатків та сервісів, які можуть точніше відповідати вимогам користувачів і забезпечувати високий рівень безпеки та приватності даних.

Пропонується покращення методів збору даних про місцезнаходження на основі акселерометра та історії рухів користувача. Основна мета цього дослідження – зменшення енерговитрат системи позиціонування під час забезпечення достатньо точної інформації про місцезнаходження.

Головна ідея дослідження полягає в розробці та вдосконаленні методу визначення місцезнаходження в міських умовах, зменшуючи залежність від GPS і зберігаючи енергію. Запропонований підхід використовує акселерометр для вимірювання активності користувача та історії рухів для оцінки швидкості. Ключовим елементом є просторово-часова таблиця, що утримує інформацію про історію переміщень.

Поведінка користувача в певних точках і часах часто є систематичною, що дає змогу ефективно визначати рух, використовуючи GPS лише за необхідності. Це сприяє збереженню енергії та оптимізації роботи додатка. Досліджено різні аспекти збору й обробки даних з мобільних сенсорів, зокрема проблеми енерговитрат та обчислювальних обмежень.

У роботі розглянуто сучасні методи збору даних та їх обмеження, а також наведено нові підходи та алгоритми для оптимізації цих процесів. Досліджено ефективність динамічного збору даних для оптимізації ресурсів мобільних пристроїв у розробці додатка на основі сучасних технологій.

Список використаних джерел

1. Android Developers. Sensors overview. URL: https://developer.android. com/guide/topics/sensors/sensors_overview

2. Khan, N., Khusro, S., Ali, S., Ahmad, J. (2016). Sensors are Power Hungry: An Investigation of Smartphone Sensors Impact on Battery Power from Lifelogging Perspective. URL: https://www.researchgate.net/publication/317494054_Sensors_are_Power_Hungry_An_Investigation_of_Smartphone_Sensors_Impact_on_Battery_Power_from_Lifelogging_Perspective

3. Choenni, S., Bargh, M. S., Roepan, C., Meijer, R. (2015). Privacy and security in data collection by citizens. URL: https://www.researchgate.net/publication/282646818_Privacy_and_security_in_data_collection_by_citizens

4. Xian, L., Bi, S., Quan, Z., Wang, H. (2021). Online Cognitive Data Sensing and Processing Optimization in Energy-harvesting Edge Computing Systems. URL: https://arxiv.org/pdf/2106.14113.pdf

5. Capponi, A., Fiandrino, C., Kliazovich, D., Bouvry, P., Giordano, S. (2017). A Cost-Effective Distributed Framework for Data Collection in Cloud-Based Mobile Crowd Sensing Architectures. DOI: 10.1109/TSUSC.2017.2666043/.

УДК 004.657:004.33(043.2)

Зінченко Б. В., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки,

Римар П. В., старший викладач кафедри інформаційних технологій

ОПТИМІЗАЦІЯ БАЗИ ДАНИХ ІЗ ДОПОМОГОЮ ВИКОРИСТАННЯ COLUMNSTORE-ІНДЕКСІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі обробка великої кількості даних у реальному часі стала ключовим аспектом багатьох бізнес-процесів. Оптимізація баз даних стає необхідністю для забезпечення швидкодії та ефективності систем обробки інформації. У цьому контексті використання Columnstore-індексів виявляється перспективним напрямом для оптимізації роботи баз даних [1].

У процесі дослідження буде проведений порівняльний аналіз продуктивності баз даних із використанням Columnstore-індексів та традиційні реляційні індекси. В роботі розглядаються конкретні сценарії, – як-от вибірка даних за певними умовами, сортування, агрегація та з'єднання таблиць. Для ілюстрації порівнянь виконаємо кілька запитів на обидві системи та проаналізуємо час виконання. Одним із методів цього дослідження є ретельний аналіз та порівняння Columnstore-індексів із традиційними реляційними базами даних із погляду ефективності, швидкодії та витрат ресурсів. Розглянемо види оптимізації [2].

Оптимізація запитів: цей вид оптимізації спрямований на поліпшення виконання SQL-запитів, містить у собі використання індексів, оптимізацію структури запиту, аналіз та розробку планів виконання.

Оптимізація структури бази даних: орієнтована на вдосконалення самої структури бази даних. Це може містити нормалізацію таблиць, вибір оптимальних типів даних, розподіл даних між таблицями для зменшення дублювання інформації.

Оптимізація індексів: забезпечує ефективність використання індексів для швидкодії вибірки даних, містить у собі створення правильних індексів, їх видалення або переструктуризацію для підвищення продуктивності.

Оптимізація запитів із використанням кешування: зменшення навантаження на базу даних шляхом зберігання результатів, часто використовуваних запитів у кеші. Це допомагає швидше давати відповіді на однакові або схожі запити.

Оптимізація конфігурації бази даних: встановлення оптимальних параметрів та налаштувань бази даних, як-от розмір буферного пулу, параметри кешування, налаштування паралельного виконання запитів та ін.

Оптимізація завдань та процесів: вдосконалення та оптимізація розкладу й виконання завдань, як-от індексація, резервне копіювання, планування оптимізованих завдань обслуговування бази даних.

Вибірка даних за певними умовами. Розглянемо випадок, коли компанія має базу даних клієнтів і хоче витягти інформацію про клієнтів, які здійснили покупки на суму більше 1 000 доларів протягом останнього місяця. Виконаємо цей запит на обидві системи та порівняємо час виконання [3].

```
SELECT CustomerName, PurchaseAmount
FROM Customers
WHERE PurchaseAmount > 1000 AND PurchaseDate >= DATEADD(MONTH, -1, GETDATE());
```

Рисунок 1. Програмний код запиту

Після виконання запитів було порівняно час виконання на обох системах. Як і очікувалось, запит на базі даних із Columnstore-індексами виконається швидше через їх здатність ефективно обробляти великі об’єми даних [4, 5].

На підставі проведеного дослідження можна визначити конкретні області, де використання Columnstore-індексів демонструє значні покращення продуктивності. Наприклад, у великих таблицях, де велика кількість колонок, та в операціях агрегації чи аналітичних запитах, де необхідно опрацювати великий обсяг даних, Columnstore-індекси можуть значно прискорити вибірку та обробку інформації.

Список використаних джерел

1. Farber L., West M., (2015). Columnstore index: SQL Server’s answer to a columnar storage format. Microsoft White Paper.

2. Graefe, G. (2016). Data Management for General Data Analytics. *ACM SIGMOD Record, 45(4),* 27–32.

3. Baklarz E., Zikopoulos P., (2015). Big Data and Analytics: Infonomics and the Business of Managing Data. IBM Press.

4. Kandiraju G., Shegalov G., Bilke A., (2013). SQL Server Columnstore Index. Apress.

5. Nikolic B., Ivanovic M., Miloševic D. (2017). A Comparative Study of Columnar Storage and Indexing Techniques in Relational Database Management Systems. *Journal of Computer Science and Technology,* 17(4), 407–416.

УДК 004.62

Колосова К. К., здобувачка 1 курсу спеціальності 122 Комп'ютерні науки,
Огороднік М. О., здобувачка 1 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ПОРІВНЯННЯ РІЗНИХ МЕТОДІВ ЗНАХОДЖЕННЯ ЗНАЧЕННЯ КОРЕЛЯЦІЇ МІЖ НАБОРАМИ ЗМІННИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

В основі багатьох кількісних досліджень лежить встановлення зв'язку між даними, а саме знаходження кореляції для розуміння, як пов'язані дві чи більше змінних. Дослідники зауважують, що кореляція не вказує на причинно-наслідкові зв'язки, особливо, коли цей критерій застосовується до перехресних даних. Припущення про вимірювання елементів тільки однією факторною величиною призводить до можливого завищення величини кореляції між змінними. Це обмеження викликало появу методів кореляції другого покоління, заснованих на моделюванні структурних рівнянь, як-от підтверджувальний факторний аналіз та дослідницьке моделювання структурних рівнянь. Точна оцінка кореляції між змінними має вирішальне значення для достовірності результатів дослідження. Коли дослідники отримують точні результати кореляції, це гарантує, що зв'язки між змінними представлені якомога ближче до реальності. Ця точність важлива, оскільки вона дає змогу дослідникам робити обґрунтовані та значимі висновки на основі своїх даних.

Якщо результати кореляції неточні або необ'єктивні, це може призвести до неправильної інтерпретації зв'язків між змінними. Наприклад, якщо кореляція завищена, це може свідчити про сильніший зв'язок між змінними, ніж існує насправді, що є результатом неправильних висновків і потенційно помилкових втручань, заснованих на помилкових доказах. У кількісних дослідженнях у сферах освіти та психології дослідники використовують різні методи для оцінки кореляції між змінними. Деякі приклади цих методів містять двофакторну кореляцію, підтверджувальний факторний аналіз та моделювання на основі дослідницьких структурних рівнянь.

Двофакторна кореляція передбачає обчислення коефіцієнта кореляції та є основним методом, який використовується для оцінки зв'язку між двома змінними (рис. 1). Цей показник вимірює силу та напрям лінійного зв'язку між наборами змінних. Він обчислює коефіцієнт кореляції, зазвичай коефіцієнт кореляції Пірсона, який вимірює силу та напрям лінійного зв'язку між двома змінними. Двофакторна кореляція передбачає, що змінні неперервні та нормально розподілені [1].

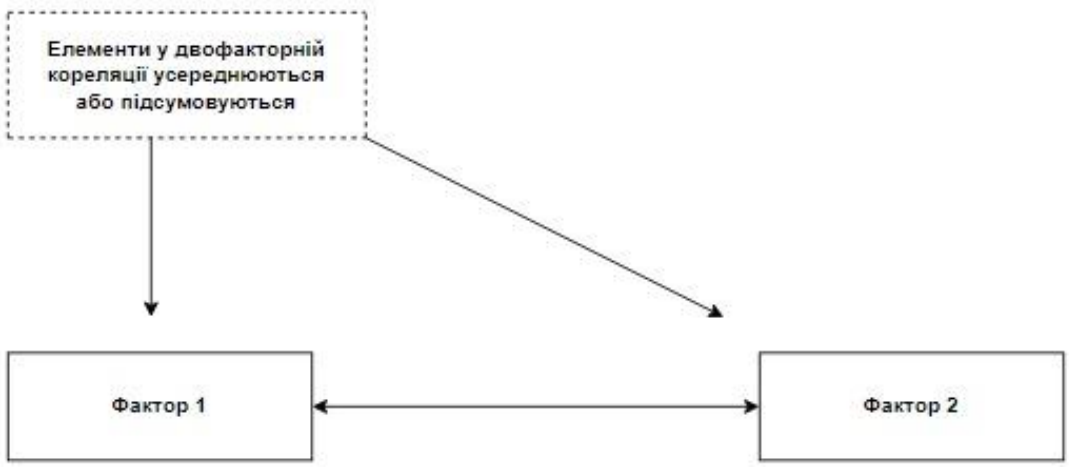


Рисунок 1. Візуальне представлення кореляції у методі двовимірної кореляції

Підтверджувальний факторний аналіз – це метод моделювання структурних рівнянь, що досліджує зв'язок між спостережуваними змінними та прихованими факторами (рис. 2). Його можна використовувати для оцінки кореляції між латентними змінними шляхом оцінки факторних навантажень і коваріацій. Цей статистичний метод використовується для перевірки моделі вимірювання й оцінки зв'язків між спостережуваними змінними та латентними конструктами. Він часто використовується в моделюванні структурними рівняннями [2].

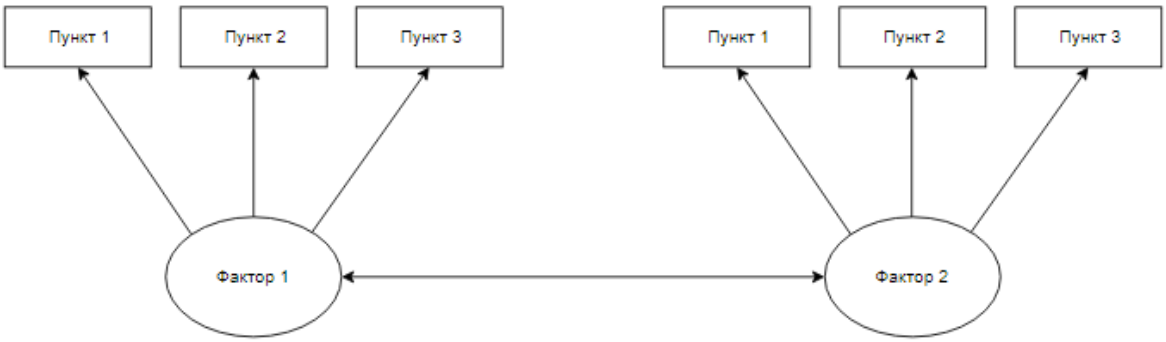


Рисунок 2. Візуальне представлення кореляції у методі підтверджувального факторного аналізу

Моделювання на основі дослідницьких структурних рівнянь – це інший тип моделювання на основі структурних рівнянь, що поєднує елементи дослідницького факторного аналізу та підтверджувального факторного аналізу (рис. 3). Цей метод дає змогу досліджувати як спостережувані, так і латентні змінні та може забезпечити більш точні оцінки кореляції, враховуючи складність і багатовимірність конструкцій. На відміну від підтверджувального факторного аналізу, метод моделювання на основі дослідницьких структурних рівнянь не припускає, що спостережувані змінні пов'язані виключно з їх гіпотетичними латентними конструктами. Це допускає можливість перехресних завантажень, коли спостережувані змінні можуть завантажуватися на кілька прихованих конструкцій. Цей метод забезпечує більш гнучку та точну оцінку факторних кореляцій, порівняно з підтверджувальним факторним аналізом [3].

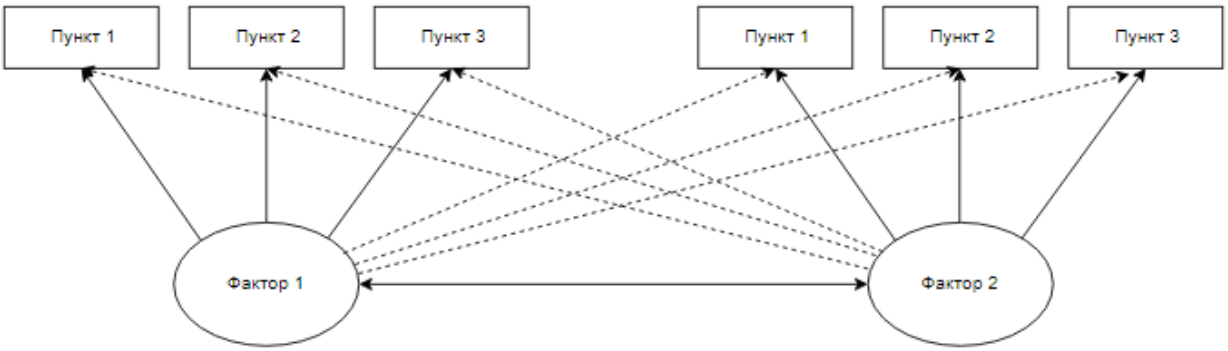


Рисунок 3. Візуальне представлення кореляції у методі моделювання на основі дослідницьких структурних рівнянь

Описані методи знаходження кореляції відрізняються своїми припущеннями та підходами до оцінки кореляції між змінними. Двовимірна кореляція є простим і зрозумілим методом, водночас підтверджувальний факторний аналіз та метод моделювання на основі дослідницьких структурних рівнянь забезпечують більш складні методи аналізу складних зв'язків між спостережуваними змінними і латентними конструктами.

Проаналізувавши методи знаходження кореляції у цій роботі, можна зробити висновки про те, що різні методи аналізу можуть давати різні результати та інтерпретації під час оцінки кореляції між змінними в кількісних дослідженнях. Можна виокремити обмеження методу двовимірної кореляції, що покладається на усереднення або підсумовування балів окремих елементів для представлення конструкцій. Цей метод може призвести до завищених кореляцій і перешкодити точній оцінці зв'язку між змінними. У дослідженнях рекомендують використовувати методи підтверджуючого факторного аналізу і дослідницького моделювання структурних рівнянь, які враховують багатовимірність конструкцій і допомагають точніше оцінювати кореляції між змінними. Загалом не можна

заперечувати важливість вибору відповідних методів для оцінки кореляції та необхідність використання більш точних методів, як-от методи підтверджуючого факторного аналізу і дослідницького моделювання структурних рівнянь.

Список використаних джерел

1. Hair, J. F., Hult, T., Ringle, C. M., Sarstedt, M. (2022). A Primer on Partial Least Squares Structural Equation Modeling (PLS-SEM), 3rd edn. New York, NA: Sage.

2. Morin, S., Arens, K., Tran, A., Caci, H. (2016). Exploring sources of construct-relevant multidimensionality in psychiatric measurement: a tutorial and illustration using the composite scale of morningness. *Int. J. Methods Psychiatr. Res*. 25, 277–288. DOI: 10.1002/mpr.1485.

3. Alamer, A., Marsh, H. (2022). Exploratory structural equation modeling in second language research: an applied example using the dualistic model of passion. *Stud. Second Lang. Acquis, 1–24.* DOI: 10.1017/S0272263121000863.

УДК 004.6+005.7

Крохмалюк В. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки ОС «Магістр»,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ВИКОРИСТАННЯ ІНСТРУМЕНТІВ ДЛЯ ЗБОРУ ДАНИХ У ДОСЛІДНИЦЬКІЙ ДІЯЛЬНОСТІ ЯК ЕФЕКТИВНИЙ МЕТОД ОТРИМАННЯ ІНФОРМАЦІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

Парсинг – це процес аналізу вхідних даних з метою виділення з них значимих елементів та перетворення їх у структурований формат для подальшої обробки. У програмуванні парсинг часто використовується для обробки текстових даних, як-от дані у форматі CSV, JSON або XML, а також для обробки коду мов програмування [2].

Парсинг може виконуватись із допомогою спеціальних програм, які називають парсерами. Парсер зчитує вхідні дані, аналізує їх за заданою граматикою та перетворює їх у структурований формат. Із допомогою парсера можна виділити з великого обсягу текстової інформації лише ту, що потрібна для подальшої обробки. У веброзробці парсинг часто використовується для збору даних із вебсторінок. Основна його ідея полягає в автоматичному зборі даних із вебсторінок за допомогою скриптів [1].

Парсинг може бути використаний для збору інформації про товари, послуги, ціни, новини тощо. Отже, парсинг – це процес аналізу вхідних даних з метою їх

обробки та перетворення у структурований формат [2]. У програмуванні парсинг часто використовується для обробки текстових даних, збору даних із вебсторінок. Для виконання парсингу використовуються спеціальні бібліотеки та інструменти, які дають змогу зчитувати та аналізувати дані з різних джерел, як-от файлові системи, бази даних, вебсторінки, електронні документи та інші джерела.

Розглянемо принцип роботи на прикладі парсера, який збирає дані з вебсторінок. Парсер аналізує HTML-код сторінки та виконує процес розбору (парсингу) на основі заздалегідь встановлених правил. Основний принцип роботи парсера полягає в тому, що він сканує HTML-код сторінки та визначає різні типи тегів, як-от заголовки, текстові блоки, посилання тощо. Після того, як парсер визначив теги та їх вміст, він може зібрати ці дані та структурувати їх у зручний формат для подальшого використання. Наприклад, якщо парсеру потрібно зібрати інформацію зі списку товарів на вебсторінці, то він буде шукати тег <ul> та його елементи <li>. Після знаходження цих тегів парсер може отримати дані про кожен товар, зокрема його назву, опис, зображення та ціну [3].

Одним із найпоширеніших методів парсингу є парсинг із використанням регулярних виразів. Регулярні вирази дають змогу шукати та витягувати певні частини тексту на основі певного шаблону. Цей метод дуже ефективний для простих парсерів, але не дуже надійний для складних структур сторінок, оскільки може не враховувати всі варіації розмітки HTML [1].

Існує також інший метод парсингу – парсинг з використанням бібліотеки для обробки HTML-коду. Цей метод допомагає парсеру більш точно визначати структуру сторінки, зокрема, з допомогою визначення CSS-селекторів. Завдяки цьому методу парсер може точніше зібрати інформацію з вебсторінки та зменшити ймовірність помилок під час парсингу.

Інструменти, з допомогою яких найчастіше створюються парсери, залежать від використовуваної мови програмування та типу даних, які потрібно парсити. Найбільш популярними інструментами для створення парсерів є:

1. Beautiful Soup: це бібліотека Python, призначена для парсингу HTML- та XML-даних. Beautiful Soup дає змогу зчитувати дані з вебсторінок та обробляти їх, знаходячи необхідні теги й атрибути [3].

2. lxml: це бібліотека Python для обробки XML- та HTML-даних, яка має високу швидкодію. lxml дає змогу використовувати XPath-запити для пошуку необхідних даних.

3. Scrapy: це фреймворк Python для парсингу вебсторінок. Scrapy дає змогу створювати скрапери (вебпавуки), які можуть збирати дані з кількох вебсторінок одночасно, виконувати асинхронні запити та оброблювати результати.

4. Selenium: це інструмент для автоматизації браузерів, який дає змогу створювати скрипти для взаємодії з вебсторінками. Selenium може бути використаний для автоматичного заповнення форм, натискання кнопок та збору даних із динамічних вебсторінок.

Парсер є корисним інструментом для науковців, які працюють із великими об’ємами даних. Завдяки парсингу вебсторінок вони можуть отримувати необхідну інформацію з різних джерел, аналізувати її та використовувати для своїх наукових досліджень.

Основна перевага використання парсера полягає в тому, що він може значно зменшити витрати часу та зусиль на пошук, обробку й аналіз даних. До того ж використання парсера допомагає отримати більш точну та повну інформацію, ніж під час ручного збору даних.

Із допомогою парсера науковці можуть зібрати та проаналізувати великий об’єм даних, що дасть змогу зробити більш точні висновки й отримати нові знання в різних галузях науки. До того ж використання парсера допомагає автоматизувати процес збору даних, що зменшує ймовірність помилок і забезпечує швидкий та ефективний доступ до необхідної інформації.

Отже, користування парсером може допомогти науковцям ефективніше й точніше збирати та аналізувати дані, що дає змогу робити більш досконалі дослідження та отримувати нові знання в різних галузях науки.

Список використаних джерел

1. Mitchell R. Web Scraping with Python: Collecting More Data from the Modern Web 2nd Edition, 2018. 306 p.

2. Heydt M. Python Web Scraping Cookbook: Over 90 proven recipes to get you scraping with Python, micro services, Docker and AWS, 2018. 364 p.

3. Beautiful Soup Documentation. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

УДК 004.85:004.93(043.2)

Бевз Д. М., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп’ютерні науки,

Римар П. В., старший викладач кафедри інформаційних технологій

ЕФЕКТИВНІСТЬ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ НА ПРИКЛАДІ ЗАДАЧІ КЛАСИФІКАЦІЇ ТЕКСТІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі обсяги текстових даних зростають із неймовірною швидкістю, що створює потребу в ефективних методах їх обробки та аналізу. Особливо актуальним є завдання класифікації текстів, яке має широке застосування в різних сферах, від виявлення спаму до аналізу настроїв.

Мета дослідження полягає у вивченні та порівнянні ефективності різних алгоритмів машинного навчання для задачі класифікації текстів. Для цього були обрані три алгоритми: наївний класифікатор Баєса, логістична регресія та дерево ухвалення рішень [1, 2].

Методологія дослідження містить збір і підготовку датасету, реалізацію вибраних алгоритмів, їх навчання та тестування. Особлива увага була приділена аналізу точності, швидкості й ефективності кожного з алгоритмів.

Види та класифікації текстів

Перед початком обробки текстів необхідно спочатку класифікувати їх за різними критеріями. Одним із таких критеріїв є тип тексту або його призначення. Розрізняють такі види текстів:

- інформативні тексти: новини, статті, наукові дослідження тощо;
- літературні тексти: романи, вірші, оповідання;
- технічні тексти: технічні описи, інструкції, технічні звіти;
- рекламні тексти: оголошення, рекламні брошури, рекламні статті;
- соціальні медіатексти: повідомлення в соціальних мережах, коментарі, пости.

Крім типу тексту, текстові дані можна класифікувати за тематикою, мовою, жанром тощо. Це допомагає виконувати більш специфічні завдання обробки та аналізу.

Для тестування алгоритму потрібні дані, на яких можна його навчити. Такі дані було взято з сайта kaggle [3]. Задача класифікації текстів вирішувалась на прикладі датасету із текстами зі статтями BBC [4]. Датасет містить 2 225 текстів, розділених на 5 категорій (рис. 1).

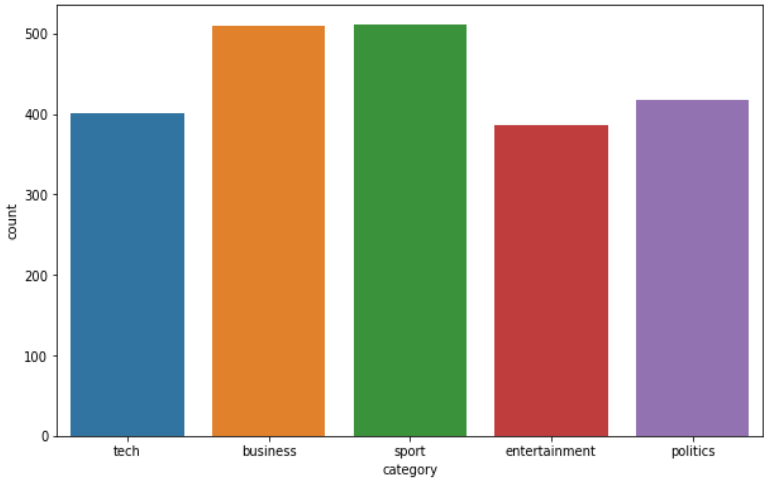


Рисунок 1. Категорії поділу тексту в датасеті

Після запуску алгоритму в Colab Notebooks потрібно завантажити датасет та подивитись інформацію про нього. Наступним кроком треба отримати гістограму, в якій показано розподіл текстів за групами. Далі потрібно створити

прогностичну модель із допомогою Word2Vec у вигляді векторного простору. Після цього запустити функцію класифікації зі створеним вектором, яка розділить текст на дві групи: для навчання і для тестування. З цього результату перший вектор відповідатиме за кількість текстів із датасету, що виділені для навчання, другий за кількість текстів для перевірки. Отриманий результат відображено на рис. 2.

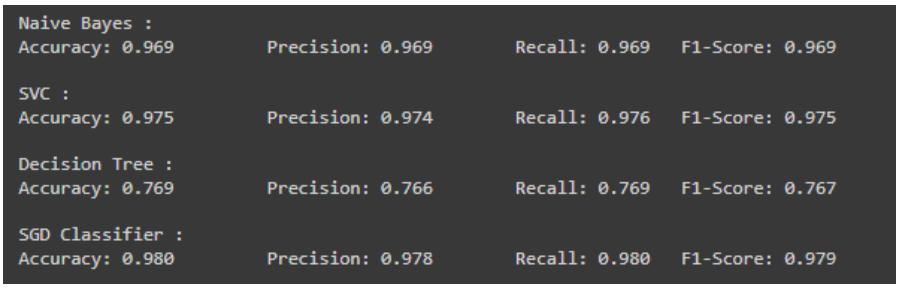


Рисунок 2. Результат навчання та тестування

Після цього можна аналізувати отримані результати роботи алгоритму на основі датасету. Після запуску алгоритму отримано такий результат:

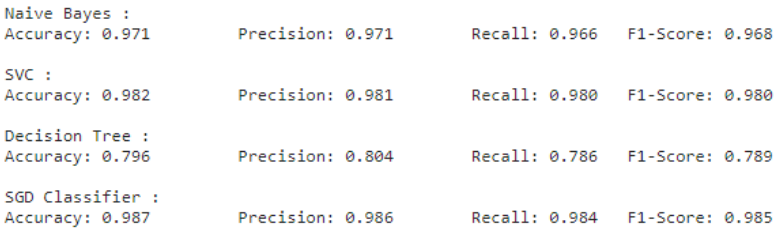


Рисунок 3. Результат роботи алгоритму

У цьому результаті є дані про кожен виконаний алгоритм: Accuracy, Precision, Recall та F1-Score.

На підставі проведених досліджень можна дійти висновку, що для цього датасету доцільно використовувати алгоритм логістичної регресії. Отже, користувачі системи можуть самі завантажувати датасети та оцінювати моделі машинного навчання для обрання найбільш ефективної моделі.

Результати дослідження показали, що кожен з алгоритмів має свої переваги й недоліки залежно від специфіки задачі та характеристик датасету. Наївний класифікатор Байєса виявився ефективним у випадках із великою кількістю класів, логістична регресія показала високу точність у бінарній класифікації, а дерево ухвалення рішень було корисним для інтерпретації результатів.

Дослідження підкреслюють важливість вибору відповідного алгоритму машинного навчання для конкретної задачі класифікації текстів. Результати

цього дослідження можуть бути корисними для фахівців у сфері обробки природної мови та машинного навчання.

Список використаних джерел

1. Palanivinayagam A., El-Bayeh C. Z., Damaševičius R. A Review of Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review. *Algorithms 2023.* № 16(5). 236 p.

2. Naive Bayes Classifier Explained: Applications and Practice Problems of Naive Bayes Classifier: вебсайт. URL: https://www.analyticsvidhya.com

3. Kaggle: вебсайт. URL: https://www.kaggle.com

4. BBC articles fulltext and category: вебсайт. URL: https://www.kaggle.com

УДК 004.4’416:004.738.1(043.2)

Луцков М. П., здобувач вищої освіти 2 курсу ОС «Магістр» спеціальності 122 Комп’ютерні науки,

Римар П. В., старший викладач кафедри інформаційних технологій

МЕТОДИ ТЕХНІЧНОЇ ОПТИМІЗАЦІЇ САЙТІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному інтернет-середовищі важко переоцінити роль вебсайтів, які є ключовим елементом взаємодії між користувачами та інформацією. Зростання конкуренції в електронному просторі підкреслює важливість належної функціональності та продуктивності вебресурсів. Оптимізація сайта виконує важливу роль у покращенні його продуктивності, швидкості завантаження та загальної ефективності.

Методи технічної оптимізації сайта є необхідним інструментарієм для забезпечення ефективної роботи вебресурсів в умовах усе вищих технологічних вимог та різноманітних платформ користувачів. Вони охоплюють широкий спектр заходів, спрямованих на поліпшення архітектури вебсайтів, їх швидкості реакції та адаптивності до різних пристроїв [1].

Ця робота спрямована на вивчення та аналіз методів технічної оптимізації сайтів з метою розкриття їх впливу на користувачів та бізнес-середовище. Вона також розглядає сучасні тенденції в розробці вебресурсів та використання новітніх технологій для досягнення максимального рівня оптимізації.

Вивчення цієї теми дає змогу розглядати важливі аспекти взаємодії між технічними показниками сайта та його користувачами. Отже, вивчення методів

технічної оптимізації сайта має велике значення для розуміння та вдосконалення вебтехнологій, що відіграють ключову роль у сучасному цифровому світі.

Оптимізація швидкості завантаження вебсайта є важливим складником для покращення користувацького досвіду та взаємодії з відвідувачами. Цей процес передбачає впровадження різних технічних стратегій, які спрямовані на зниження часу, необхідного для повного завантаження вебсторінок.

Ефективне використання кешу для збереження статичних ресурсів, як-от CSS, JavaScript і зображення, дає змогу уникнути повторного завантаження тих самих файлів під час кожного нового відвідування, забезпечуючи в таким спосіб більш швидке завантаження сторінок.

Мініфікація та компресія ресурсів, як-от CSS та JavaScript, а також стискання зображень без втрати якості, допомагають зменшити обсяг файлів і прискорити завантаження вебсторінок.

Використання CDN (Content Delivery Network) дає змогу розміщувати копії статичних ресурсів на серверах, що розташовані ближче до користувачів, що так само допомагає зменшити час передачі даних і прискорює завантаження сторінок.

Асинхронне завантаження ресурсів, як-от JavaScript та CSS, сприяє уникненню блокування завантаження сторінки, допомагаючи користувачам швидше бачити контент, навіть коли деякі елементи ще завантажуються.

Оптимізація зображень через їх стиснення та використання оптимальних форматів сприяє подальшому зменшенню їх розміру, що впливає на загальний час завантаження [2].

Отже, враховуючи ці технічні стратегії, можливо досягти оптимальної швидкості завантаження вебсайта, що є ключовим аспектом для покращення взаємодії з користувачами та підвищення його ефективності в цифровому середовищі.

Оптимізація структури та архітектури вебсайта є ключовим етапом у створенні ефективного та орієнтованого на користувача середовища. Цей процес включає в себе різноманітні технічні стратегії, спрямовані на поліпшення організації контенту та забезпечення легкого доступу користувачів до необхідної інформації.

SEO-оптимізація відіграє важливу роль у визначенні індексації сайта пошуковими системами. Використання правильних HTML-тегів та оптимізованих URL-адрес допомагає поліпшити ранжування та забезпечити більшу видимість у пошукових результатах.

Мобільна оптимізація передбачає адаптивний дизайн для оптимального відображення контенту на різних пристроях, забезпечуючи в такий спосіб зручність використання для мобільних користувачів.

Ефективне використання технічних ресурсів включає оптимізовані файли CSS та JavaScript, а також кодування і стиснення для зниження обсягу файлів, що впливає на швидкість завантаження [3].

Навігаційна структура відіграє ключову роль у забезпеченні легкості знаходження інформації для користувачів. Зручна система навігації та логічна ієрархія контенту сприяють покращенню взаємодії з вебсайтом.

Аналіз та вдосконалення з допомогою інструментів вебаналітики та A/B-тестування допомагають виявляти слабкі сторони та здійснювати постійні покращення для оптимізації результатів і відповіді на потреби аудиторії.

Оптимізація структури й архітектури визначається бажанням надати ефективне та логічно побудоване середовище для користувачів, що сприяє вдосконаленню користувацького досвіду й досягненню успіху в онлайн-середовищі.

Як висновок можна зазначити, що оптимізація вебсайта – це складний та багатогранний процес, що об’єднує в собі різноманітні технічні стратегії для досягнення важливих цілей. Оптимізація швидкості завантаження, структури та архітектури сайта є важливими аспектами для покращення користувацького досвіду та досягнення високої ефективності в онлайн-середовищі.

Шлях до успішної оптимізації швидкості завантаження передбачає використання різних стратегій, як-от кешування, мініфікація та компресія ресурсів, використання CDN та інші технічні прийоми, які спрямовані на зменшення часу завантаження сторінок.

Оптимізація структури та архітектури вебсайта передбачає SEO-оптимізацію, мобільну оптимізацію, ефективне використання технічних ресурсів, створення зручної навігаційної системи і постійний аналіз для вдосконалення.

Важливо зазначити, що успішна оптимізація вимагає комплексного підходу та постійного вдосконалення відповідно до змін в інтернет-середовищі та потреб користувачів. Оптимізація вебсайта – це постійний процес, спрямований на забезпечення високої продуктивності, зручності використання та відповідності сучасним вимогам онлайн-середовища.

Список використаних джерел

1. Види методів оптимізації сайтів. URL: https://webtune.com.ua/statti/internet-marketing/vydy-poshukovoyi-optymizacziyi-vnutrishnya-ta-zovnishnya/

2. Poper T. (2021). Improve WebSite Perfomance, 55–65.

3. Marroti G. (2023). Hight Perfomance Design, 123–130.

УДК 004.43

Афанасьєва Д. С., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

РЕАЛІЗАЦІЯ БІНАРНОГО ДЕРЕВА МОВОЮ ПРОГРАМУВАННЯ PYTHON З ВИКОРИСТАННЯМ РЕКУРСІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі програмування бінарні дерева є важливими структурами даних, які знаходять широке застосування у різних областях. Особливо важливим є їх використання у великих проєктах. Одним із варіантів побудови бінарного дерева є застосування рекурсії (принцип у програмуванні, коли функція у своїй реалізації викликає сама себе), яка дає змогу ефективно опрацьовувати та зберігати дані. Така реалізація матиме багато переваг, а саме: простота і зрозумілість, ефективність у використанні пам'яті та застосування у різних типах алгоритмів. У представленій роботі буде розглянуто реалізацію бінарного дерева мовою програмування Python із використанням підходу, що базується на рекурсії.

Бінарне дерево – певна абстрактна структура, яка складається з вершин (вузлів) та ребер, які пов'язують ці вершини між собою. Вузли дерева можуть мати лише по два нащадки кожен, а саме: лівого та правого [1]. Ребра ж поєднують між собою вершини-родичі.

Основні елементи бінарних дерев [2]:

1. Кореневий елемент (root) – перший вузол дерева, від якого відходять усі інші елементи.

2. Внутрішні вузли (internal nodes) – це ті вузли, які мають хоча б одного нащадка.

3. Листя (leaves) – вузли, які не мають жодного нащадка.

4. Лівий та правий нащадки – два варіанти нащадків певного вузла.

Бінарні дерева активно використовуються в алгоритмах пошуку даних, адже вони можуть бути побудовані так, що отримувати доступ до даних буде доволі швидко. Також їх застосування можливе в алгоритмах сортування, де використовується бінарне дерево пошуку, використання якого допомагає реалізовувати великі об'єми даних.

Розглянемо, як саме можна реалізувати бінарне дерево мовою програмування Python завдяки застосуванню рекурсії (рис. 1).

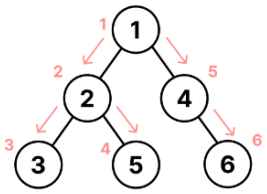


Рисунок 1. Приклад бінарного дерева для реалізації [3]

У цьому прикладі обхід бінарного дерева буде реалізованим за принципом прямого обходу, коли першим виводиться кореневий вузол, далі рекурсивний обхід лівого піддерева й рекурсивний обхід правого піддерева. Спочатку необхідно реалізувати клас Node, який представляє вузол бінарного дерева. У конструкторі __init__ вузол ініціалізується з ключем value та двома дочірніми вузлами (left_node та right_node), які на початку встановлені в None (рис. 2).

```
class Node:
    def __init__(self, value):
        self.left_node = None
        self.right_node = None
        self.data = value
```

Рисунок 2. Створення класу Node [4]

Додаємо функцію insert_node, що вставляє новий вузол зі значенням value в бінарне дерево. Якщо вузол node є порожнім, то створюється новий вузол зі значенням value. Якщо value менше, ніж значенню поточного вузла, то функція рекурсивно викликається для вузла лівого піддерева. У випадку, якщо value більше або рівне значенню поточного вузла, функція викликається для вузла правого піддерева. У будь-якому випадку функція повертає оновлений вузол node (рис. 3).

```
def insert_node(node, value):
    if node is None:
        return Node(value)
    if value < node.data:
        if node.left_node is None:
            node.left_node = Node(value)
        else:
            insert_node(node.left_node, value)
    else:
        if node.right_node is None:
            node.right_node = Node(value)
        else:
            insert_node(node.right_node, value)
    return node
```

Рисунок 3. Функція додавання вузлів до дерева [4]

Далі необхідно створити функцію для побудови бінарного дерева з використанням масиву вузлів nodes. Якщо список nodes порожній, функція повертає None. Інакше, вона створює кореневий вузол з першого значення списку, а потім викликає функцію insert_node для додавання всіх інших значень у дерево (рис. 4).

```
def build_binary_tree(nodes):
    if not nodes:
        return None
    root = Node(nodes[0])
    for val in nodes[1:]:
        insert_node(root, val)
    return root
```

Рисунок 4. Побудова бінарного дерева

Фінальним етапом є реалізація функції preorder_traversal для виведення значення кожного вузла в порядку прямого обходу. Вибудоване бінарне дерево створюється з допомогою списку nodes, а потім виконується прямий обхід від кореневого вузла tree_root, виводячи значення вузлів (рис. 5).

Після запуску програми отримуємо результат та перевіряємо правильність виконання програми (рис. 6).

```
def preorder_traversal(node):
    if node:
        print(node.data, end=" ")
        preorder_traversal(node.left_node)
        preorder_traversal(node.right_node)


nodes = [1, 2, 4, 5, 3, 6]
tree_root = build_binary_tree(nodes)
print("Прямий обхід дерева:")
preorder_traversal(tree_root)
```

Рисунок 5. Прямий обхід дерева та виведення результатів [4]

```
Прямий обхід дерева:
1 2 4 3 5 6
Process finished with exit code 0
```

Рисунок 6. Результат виконання програми

Отже, було розглянуто питання реалізації бінарного дерева мовою програмування Python з використанням рекурсивного підходу. Така реалізація дасть змогу легко створювати та опрацьовувати бінарні дерева. Використання рекурсивних методів спрощує реалізацію операцій над деревом та надає чітку

структуру для подальших досліджень у галузі алгоритмів обробки даних для широкого застосування у різних областях інформатики та програмування.

Список використаних джерел

1. Binary Tree. URL: https://www.programiz.com/dsa/binary-tree
2. Binary Trees. URL: https://www.andrew.cmu.edu/course/15-121/lectures/Trees/trees.html
3. More recursion examples. URL: https://programming-23.mooc.fi/part-11/4-more-recursion-examples
4. Binary Tree Implementation and Visualization in Python. URL: https://levelup.gitconnected.com/binary-tree-implementation-and-visualization-in-python-2f4782887ca2

УДК 517.2

Афанасьєва Д. С., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Фриз І. В., канд. фіз.-мат. наук, старший викладач кафедри інформаційних технологій

ЗАСТОСУВАННЯ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ ТА ЇХ СИСТЕМ У СФЕРІ КОМП'ЮТЕРНИХ НАУК

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному інформаційному суспільстві використання диференціальних рівнянь та їх систем у різноманітних сферах стає все більш актуальним та розповсюдженим. Диференціальні рівняння виступають не лише як математична теорія, але й як потужний інструмент для аналізу, моделювання та розв'язування різноманітних задач.

У процесі розв'язування багатьох задач із різних предметних областей часто виникає ситуація, коли важко чи навіть неможливо встановити функціональну залежність між шуканими величинами та вхідними змінними. Проте може існувати інший тип зв'язку, який пов'язує незалежні змінні, функції та їх похідні, тобто приходимо до диференціальних рівнянь. Нагадаємо, що диференціальним рівнянням n-го порядку називається рівняння виду:

$$F\left(x, y, y', y'', \ldots y^{(n)}\right) = 0,$$

де x, y – змінні,

$y', y'', y^{(n)}$ – похідні відповідних порядків [1].

У випадку, коли необхідно описати поведінку складної системи, яка змінюється в часі, використовуються системи диференціальних рівнянь. Вони являють собою математичну модель, яка описує залежність між декількома функціями та їх похідними. Нормальною системою диференціальних рівнянь є сукупність рівнянь виду [1]:

$$\begin{cases} \frac{dy_1}{dx} = f_1(x, y_1, \dots y_n) \\ \dots \\ \frac{dy_n}{dx} = f_n(x, y_1, \dots y_n) \end{cases}.$$

Розглянемо деякі приклади застосування диференціальних рівнянь у комп'ютерних науках. Варто зазначити, що вони є важливими у нейронних мережах. Деякі екосистеми, як-от Julia для наукового машинного навчання, дають можливість вбудовувати моделі диференціальних рівнянь у нейронні мережі [2]. Це відкриває можливості для розширення класичних моделей та включення великої кількості знань. Для навчання нейронних мереж використовуються різні алгоритми, і одним із найпоширеніших є метод зворотного поширення помилки. Цей метод використовує диференціальні рівняння з похідними частинними для розрахунку градієнта функції втрат, який визначає, наскільки добре мережа виконує завдання. Градієнт використовується для зміни параметрів мережі з метою мінімізації втрат. Наприклад, якщо маємо нейронну мережу, що навчається розпізнавати числа з певного набору даних, використовується метод зворотного поширення помилки. Мережа робить прогноз для кожного зображення, обчислює функцію втрат та використовує градієнт для корекції параметрів, змінюючи їх так, щоб зменшити втрати.

Іншою сферою застосування диференціальних рівнянь є системи автоматичного управління, які є комплексом технічних та програмних засобів, що використовуються для підтримки бажаного стану або параметрів системи шляхом автоматичного впливу на її роботу. Лінійні системи автоматичного управління використовують диференціальні рівняння для моделювання та аналізу різноманітних процесів управління, як описано у [3]. Нехай маємо об'єкт управління, який математично може бути виражено лінійними диференціальними рівняннями з визначеним вектором стану $x(t)$ й сигналом управління $u(t)$ виду:

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t),$$

де A і B – коефіцієнти динаміки системи у вигляді матриць.

Тоді лінійний регулятор подається у вигляді формули:

$$u(t) = -Kx(t),$$

де K – матриця керування.

Диференціальні рівняння використовуються також у відеоіграх. Їх застосування базується на побудові фізичних моделей, що описано у [4], зокрема для моделювання руху тіл, взаємодії об'єктів, симуляції різних фізичних ефектів, що дає змогу створювати реалістичні фізичні сценарії у віртуальному середовищі. Рух об'єктів, як-от тканини чи рідини, може бути описаний системою часткових диференціальних рівнянь. Наприклад, для симуляції динаміки тканин використовуються рівняння, що враховують сили натягу, гравітації та взаємодії між частинками.

Варто також зазначити застосування диференціальних рівнянь у комп'ютерному зоровому сприйнятті для завдань, як-от виявлення контурів та сегментація зображень. Градієнт зображення може бути обчислений із допомогою диференціальних рівнянь із частинними похідними, і ця інформація використовується для ідентифікації країв на зображенні [5].

Отже, диференціальні рівняння та їх системи є потужним інструментом для аналізу, моделювання та розв'язування різноманітних задач у сфері комп'ютерних наук. У майбутньому можна очікувати на подальше збільшення використання диференціальних рівнянь та їх систем у різних сферах життя. Це пов'язано з розвитком комп'ютерних технологій, які допомагають ефективно розв'язувати диференціальні рівняння чисельними методами.

Список використаних джерел

1. Гой Т. П., Махней О. В. Диференціальні рівняння: навч. посіб. Івано-Франківськ: Сімик, 2012. 352 с.
2. Differential Equations as a Neural Network Layers. URL: https://towardsdatascience.com/differential-equations-as-a-neural-network-layer-ac3092632255 (дата звернення: 03.12.2022).
3. Differential coding in networked controlled linear systems / C. Canudas de Wit, F. Rubio, J. Fornes, F. Gomez Estern. *American Control Conference, IEEE ACC'06.* Jun 2006, France. P. 4177–4182. DOI: 10.1109/ACC.2006.1657374.
4. Fluid Simulation for Video Games (part 1). URL: https://venturebeat.com/pc-gaming/fluid-simulation-for-video-games-part-1/ (дата звернення: 03.12.2022).
5. Lin Z., Zhang W., Tang X. Learning Partial Differential Equations for Computer Vision. *Microsoft technical report #MSR-TR-2008-189*. August, 2008. URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2008/08/MSR-TR-2008-189.pdf

УДК 621.39

Бєжин Є. В., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ІНТЕГРОВАНІ СИСТЕМИ УПРАВЛІННЯ В ПРОМИСЛОВОСТІ: АНАЛІЗ ЕФЕКТИВНОСТІ ТА ВПРОВАДЖЕННЯ ОПТИМАЛЬНИХ СТРАТЕГІЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

Інтегровані системи управління (ІСУ) є важливим елементом сучасного промислового виробництва. Вони дають змогу підприємствам підвищити ефективність своїх операцій, зменшити витрати, покращити якість і гнучкість, а також підвищити безпеку.

ІСУ – це комп'ютерні системи, які об'єднують різні процеси та операції в межах організації. Вони допомагають збирати, обробляти та аналізувати дані з різних джерел, щоб забезпечити більш ефективне управління та прийняття рішень.

У роботі розглядаються потенційні переваги та стратегії впровадження інтегрованих систем управління (ІСУ) в промисловості. ІСУ призначені для покращення ефективності виробничого процесу. Однак для досягнення цієї мети необхідно провести аналіз ефективності вже наявних систем та ідентифікувати слабкі місця. Важливо враховувати потреби конкретної промислової галузі та вибрати такі ІСУ, які найкраще відповідають вимогам [1].

Наприклад, ІСУ можуть допомогти підприємствам:

- зменшити витрати, знижуючи використання ресурсів, скорочуючи простої та підвищуючи продуктивність;
- підвищити якість продуктів та послуг, контролюючи процеси та забезпечуючи дотримання вимог;
- бути більш гнучкими у відповідь на зміни ринку, даючи змогу швидко адаптуватися до нових умов;
- підвищити безпеку своїх операцій, забезпечуючи контроль і моніторинг небезпечних умов.

Аналіз ефективності інтегрованих систем управління (ІСУ) є важливим етапом для визначення того, наскільки добре система відповідає бізнес-потребам та чи досягнуті поставлені цілі. Однак щоб ІСУ були ефективними, їх необхідно правильно оцінювати та аналізувати. Для проведення такого аналізу важливо врахувати низку ключових аспектів.

ІСУ повинні сприяти покращенню ефективності операцій підприємства. Це можна виміряти з допомогою таких показників:

- продуктивність праці;
- витрати;
- якість.

Для проведення аналізу ефективності ІСУ необхідно зібрати дані за цими показниками до та після впровадження ІСУ. Після того, як дані будуть зібрані, їх необхідно проаналізувати, щоб визначити, чи відбулися позитивні зміни після впровадження ІСУ. Якщо зміни позитивні, то можна зробити висновок про те, що ІСУ ефективна [2].

Однак важливо враховувати, що ефективність ІСУ може змінюватися з часом. Це пов'язано з тим, що потреби та цілі підприємства можуть змінюватися, а також те, що ІСУ може розвиватися та вдосконалюватися.

Впровадження оптимальних стратегій для інтегрованих систем управління (ІСУ) в промисловості є критичним етапом, який вимагає уважного підходу та дотримання кількох ключових принципів. Для досягнення максимального потенціалу ІСУ важливо розробити та впровадити оптимальні стратегії. Нижче подано кілька кроків, які можуть бути передбачені стратегією впровадження ІСУ:

- гнучкість і масштабованість: забезпечте гнучкість системи, щоб вона могла адаптуватися до змін у бізнес-середовищі. Масштабованість є також важливою, особливо якщо підприємство планує зростання в майбутньому;
- навчання та підтримка персоналу: забезпечте достатнє навчання для персоналу, щоб вони могли ефективно використовувати нову систему. До того ж забезпечте механізми підтримки та вирішення проблем після впровадження;
- аналіз потреб та визначення мети: проведіть докладний аналіз потреб вашого підприємства і визначте конкретні цілі, яких ви хочете досягти впровадженням ІСУ. Це може передбачати оптимізацію виробничих процесів, підвищення ефективності чи покращення моніторингу.

Інтегровані системи управління (ІСУ) є потужним інструментом, який може допомогти підприємствам у промисловості підвищити свою ефективність і конкурентоспроможність. Вони можуть забезпечити підприємствам конкурентні переваги в ефективності операцій, зниженні витрат, якості, гнучкості і безпеці, але виробники повинні постійно переглядати та модернізувати свої ІСУ, щоб залишатися конкурентоспроможними та відповідати вимогам швидкозмінюваного ринку.

Список використаних джерел

1. Snee, A. Integrated Management Systems: A Practical Guide. ASQ Quality Press, 2016.

2. Bryson, M. Implementing Integrated Management Systems: A Step-by-Step Guide. Quality Progress, vol. 41, № 1, 2008, 45–50.

3. International Organization for Standardization (2023). *ISO.org. Benefits of Integrated Management Systems.* URL: www.iso.org/isoiec-45001-benefits.html

4. Семенюк, О. А., Кирилащук, Т. Г., Січко, Т. В. (2021). Прикладні аспекти обробки даних в інформаційних системах. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця. С. 212–213.

УДК 658.012.3

Бєжин Є. В., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ ВИРОБНИЧИХ ПРОЦЕСІВ В УМОВАХ НЕСТАБІЛЬНОСТІ ВИРОБНИЧОГО СЕРЕДОВИЩА: ВИКОРИСТАННЯ АДАПТИВНИХ МЕТОДІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сучасне виробниче середовище характеризується високою динамікою та нестабільністю. Швидкі технологічні зміни, зростання конкуренції, зміни попиту та вимог споживачів, а також непередбачувані зовнішні фактори створюють значні виклики для підприємств. У таких умовах ефективне управління виробничими процесами є ключовим фактором успіху.

Оптимізація виробничих процесів в умовах нестабільності вимагає застосування адаптивних методів. Адаптовані методи дають змогу підприємствам швидко реагувати на зміни у виробничому середовищі, підтримувати ефективність виробництва та конкурентоспроможність.

Агільність як ключовий елемент адаптивних методів. Агільність є ключовим елементом адаптивних методів, оскільки вона допомагає підприємствам швидко адаптуватися до змін у виробничому середовищі. Агільні методи засновані на принципах ітеративності, інкрементальної розробки та постійного зворотного зв'язку. Ці принципи дають змогу підприємствам:

- швидко адаптуватися до змін вимог;
- реагувати на непередбачувані події;
- інноваційно вирішувати проблеми.

Одним із найпоширеніших агільних методів є методологія Scrum. Scrum заснований на принципах ітеративності та інкрементальної розробки. У межах Scrum-процесу розробка продукту здійснюється в ітераціях, які називаються Scrum-спринтами. Тривалість спринта зазвичай становить 1–4 тижні [1]. На початку кожного спринта команда розробників проводить sprint planning meeting, на якому визначає цілі спринта та планує роботу. Упродовж спринта команда

працює над реалізацією цих цілей. Наприкінці спринта команда проводить sprint review, на якому демонструє результати роботи клієнтам або користувачам. Наприкінці спринта команда проводить sprint retrospective, на якій аналізує роботу спринта та визначає напрями для вдосконалення.

Автоматизація та контейнеризація для стабільності виробничого середовища. Автоматизація та контейнеризація можуть використовуватися разом для підвищення стабільності виробничого середовища. Наприклад, автоматизовані процеси тестування та розгортання можуть бути використані для забезпечення належної роботи програмного забезпечення у виробничому середовищі. Контейнери можна використовувати для створення стандартизованого середовища для розгортання програмного забезпечення, що допомагає зменшити кількість помилок [2]. Налагодження процесу безперервної інтеграції / безперервного розгортання (CI/CD) дає змогу швидко розгортати зміни і знижує ризики; використання технологій контейнеризації, як-от Docker, забезпечує стабільне робоче середовище, допомагаючи розгортати зміни в ізольованих контейнерах перед розгортанням у виробничому середовищі. Автоматизація зменшує кількість людських помилок та підвищує швидкість і ефективність роботи. Контейнери ізольовані від інших компонентів середовища, що забезпечує стабільне виробниче середовище.

Постійне вдосконалення за допомогою моніторингу та зворотного зв'язку. Важливим елементом адаптивності є постійний моніторинг і зворотний зв'язок. Використання метрик продуктивності, автоматизованих тестів та аналізу даних допомагає оперативно виявляти проблеми та вдосконалювати процеси. Зворотний зв'язок від користувачів та учасників проєкту є важливим компонентом управління змінами та покращення якості продукту. Моніторинг дає змогу підприємствам виявляти проблеми та можливості для вдосконалення й оцінювати ефективність виробничих процесів. Зворотний зв'язок може бути як позитивним, так і негативним, але він завжди цінний, оскільки допомагає підприємствам розуміти, як їх виробничі процеси сприймаються іншими, оцінювати результати впроваджених змін і вдосконалювати стратегію та тактику управління виробничими процесами [3].

Найбільш ефективними адаптивні методи є в умовах високої мінливості виробничого середовища. У таких умовах традиційні методи оптимізації, які базуються на статичному аналізі, можуть виявитися неефективними. Адаптивні методи дають змогу постійно адаптувати виробництво до змін у виробничому середовищі, що забезпечує його оптимальність.

Прикладом застосування адаптивних методів оптимізації виробничих процесів є використання методів управління запасами в умовах коливань попиту. У таких умовах підприємству необхідно постійно контролювати рівень запасів і своєчасно їх поповнювати, щоб уникнути дефіциту або перевиробництва про-

дукції. Адаптивні методи управління запасами дають змогу підприємствам оперативно реагувати на зміни попиту, забезпечуючи водночас оптимальний рівень запасів.

Список використаних джерел

1. Оптимізація виробничих процесів / під ред. В. М. Горєлова. Київ, 2017.

2. Горєлова В. М., Козарєва В. В. Адаптивні методи оптимізації виробничих процесів. Київ, 2022.

3. Кірієнко В. М., Бойко В. В. Адаптивні методи оптимізації виробничих процесів. *Вісник Національного університету «Львівська політехніка».* Львів, 2009.

4. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів СО «Бакалавр» денної та заочної форм навчання спеціальностей 122 «Комп'ютерні науки», 113 «Прикладна математика». Вінниця: ДонНУ імені Василя Стуса. 2020.

УДК 519.2

Бєжин Є. В., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

СТАТИСТИЧНЕ НАВЧАННЯ В УМОВАХ НЕВИЗНАЧЕНОСТІ: МОДЕЛЮВАННЯ ТА УПРАВЛІННЯ НЕВИЗНАЧЕНІСТЮ У ПРОГНОСТИЧНИХ МОДЕЛЯХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У реальному світі багато даних є невизначеними. Це може бути через наявність шуму, невизначеності параметрів або зміни в середовищі. Статистичне навчання в умовах невизначеності – це галузь досліджень, яка займається розробкою методів навчання моделей, які можуть працювати з невизначеними даними.

Статистичне навчання – це галузь машинного навчання, яка використовує статистичні методи для навчання моделей з даних. Ці моделі можна використовувати для прогнозування майбутніх результатів, класифікації об'єктів або виявлення аномалій [1].

Моделювання невизначеності – це перший крок у статистичному навчанні в умовах невизначеності. Воно допомагає зрозуміти природу невизначеності даних і розробити методи для її врахування.

Існує два основні підходи до моделювання невизначеності:

➢ Підсумкові методи моделюють невизначеність даних, використовуючи статистичні методи, як-от середнє значення, медіана, модальний або ймовірнісний розподіл. Ці методи прості у використанні та можуть бути ефективними для зменшення шуму в даних. Однак вони не завжди можуть адекватно враховувати невизначеність даних, особливо якщо дані є нелінійними або мають складну структуру.

➢ Методи баєсівської статистики використовують баєсівську теорію для моделювання невизначеності даних. Баєсівська теорія дає змогу враховувати невизначеність параметрів моделі та змін у середовищі. Ці методи більш складні, ніж підсумкові методи, але вони можуть бути більш точними для прогнозування в умовах невизначеності [2].

Управління невизначеністю необхідно розробити після того, як невизначеність даних була змодельована. Це дає змогу оцінити рівень невизначеності прогнозів і зробити більш обґрунтовані рішення [3].

Існує два основні підходи до управління невизначеністю:

➢ Методи зворотного поширення невизначеності використовують баєсівську теорію для обчислення апостеріорної ймовірності прогнозів. Це допомагає оцінити рівень невизначеності кожного прогнозу. Наприклад, метод зворотного поширення невизначеності в регресії дає змогу обчислити апостеріорну ймовірність прогнозованого значення показника.

➢ Методи прийняття рішень в умовах невизначеності використовують теорію прийняття рішень для вибору найкращого прогнозу, беручи до уваги рівень невизначеності. Наприклад, метод максимального очікування використовують для вибору прогнозу, який має максимальну очікувану вигоду.

Приклади статистичного навчання в умовах невизначеності

Статистичне навчання в умовах невизначеності використовується в різноманітних галузях, як-от:

➢ у медицині – для прогнозування ризику захворювань, ефективності лікування та інших медичних результатів. У цьому випадку невизначеність може бути викликана шумом у даних, неповною інформацією про пацієнта або іншими факторами. Методи: використання класифікаційних алгоритмів машинного навчання для аналізу медичних даних та класифікації пацієнтів за наявністю хвороби; врахування експертної думки лікарів для покращення точності моделей. Приклади методів: метод опорних векторів (SVM) для класифікації, нейронні мережі для аналізу зображень, баєсівські моделі для інтеграції експертних знань.

➢ у фінансах – для прогнозування цін на акції, валютних курсів та інших фінансових показників. У цьому випадку невизначеність може бути викликана шумом у даних, змінами у ринкових умовах або іншими факторами. Методи: використання часових рядів та регресійних моделей для прогнозування цін

активів; методи ансамблю для об'єднання прогнозів різних моделей та зменшення невизначеності. Приклади методів: ARIMA для аналізу часових рядів, баєсівські мережі для моделювання невизначеності у фінансових даних.

У роботі розглянуто актуальні аспекти використання статистичного навчання в умовах невизначеності та запропоновано практичні підходи до моделювання та управління невизначеністю у прогностичних моделях. Результати досліджень можуть бути використані для покращення точності та достовірності прогнозів у різних галузях застосування статистичного навчання. Підкреслено важливість розвитку адаптивних алгоритмів та використання баєсівського підходу для ефективного управління невизначеністю.

Список використаних джерел

1. Murphy, K. P. Machine Learning: A Probabilistic Perspective. MIT Press. 2021.

2. Машинне навчання. URL: https://uk.wikipedia.org/wiki/%D0% 9C%D0% B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B5_%D0%BD%D0%B0%D0%B2%D1%87%D0%B0%D0%BD%D0%BD%D1%8F (дата звернення: 02.12.2023).

3. Barber D. Bayesian reasoning and machine learning.Cambridge University Press. 2012.

УДК 004.056

Бондарєв О. О., Горін В. В., здобувачі вищої освіти 1 курсу спеціальності 125 Кібербезпека та захист інформації,

Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ЗВ'ЯЗОК МАТЕМАТИЧНОЇ КРИПТОГРАФІЇ ТА ОПТИМАЛЬНОГО КОДУВАННЯ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Математична криптографія та оптимальне кодування є двома ключовими галузями інформаційних технологій, які спільно використовують математичні концепції для забезпечення безпеки й ефективності обміну інформацією. Ми розглянемо основні принципи цих двох галузей та їх взаємозв'язок. Математична криптографія є наукою, яка вивчає методи захисту інформації від несанкціонованого доступу. Основними завданнями криптографії є конфіденційність, цілісність та автентифікація інформації. Для досягнення цих цілей використовуються різні математичні алгоритми, як-от шифрування, підписи та інші методи [1].

Актуальність. Класифікація криптографічних алгоритмів (КА):

➢ Тайнопис. Відправник і одержувач роблять над повідомленням перетворення, відомі лише їм двом. Стороннім особам невідомий алгоритм шифрування. Деякі фахівці вважають, що тайнопис не є криптографією взагалі.

➢ КА з ключем. Алгоритм впливу на передані дані відомий усім стороннім особам, але він залежить від деякого параметра – «ключа», яким володіють лише відправник і одержувач.

➢ Симетричні КА. Для зашифровки і розшифровки повідомлення використовується один і той же блок інформації (ключ).

➢ Асиметричні КА. Це такий алгоритм, який для зашифровки повідомлення використовує один («відкритий») ключ, відомий усім охочим, а для розшифровки – інший («закритий») ключ, який існує тільки в одержувача.

Залежно від кількості ключів, які застосовуються у конкретному алгоритмі:

➢ Безключові КА – не використовують в обчисленнях жодних ключів.

➢ Одноключові КА – працюють з одним додатковим ключовим параметром (якимсь таємним ключем).

➢ Двоключові КА – на різних стадіях роботи в них застосовуються два ключові параметри: секретний та відкритий ключі.

Залежно від характеру впливів, що виробляються над даними, алгоритми підрозділяються на:

➢ Перестановочні – блоки інформації (байти, біти, більші одиниці) не змінюються самі по собі, але змінюється їх порядок проходження, що робить інформацію недоступною сторонньому спостерігачеві.

➢ Підстановочні – самі блоки інформації змінюються за законами криптоалгоритму. Переважна більшість сучасних алгоритмів належить цій групі.

Залежно від розміру блоку інформації криптоалгоритми поділяються на:

➢ Потокові шифри – одиницею кодування є один біт. Результат кодування не залежить від попереднього вхідного потоку. Схема застосовується в системах передачі потоків інформації, тобто в тих випадках, коли передача інформації починається і закінчується в довільні моменти часу і може випадково перериватися. Найбільш поширеними представниками потокових шифрів є скремблери.

➢ Блочні шифри – одиницею кодування є блок із декількох байтів (діапазон кодування знаходиться в межах 4–32 байтів).

Результат кодування залежить від усіх вихідних байтів цього блоку. Схема, що застосовується під час пакетної передачі інформації та кодування файлів, представлена на рис. 1 [2].

Алгоритми шифрування базуються на математичних операціях, які ускладнюють процес розшифрування без наявності відповідного ключа. Криптографічні протоколи, як-от RSA, ECC та інші, використовують абстракції алгебричних

структур, як-от групи та поля. Оптимальне кодування є галуззю теорії інформації, яка займається розробкою кодів для передачі та збереження інформації з мінімальною кількістю біт. Це означає, що коди мають ефективно використовувати ресурси та забезпечувати найменший можливий обсяг інформації. Основним поняттям в оптимальному кодуванні є ентропія, яка визначає середню кількість біт, необхідних для кодування символа [3]. Коди, які наближаються до цієї ентропії, вважаються оптимальними. Обидві галузі використовують математичні концепції для досягнення своїх цілей. У криптографії використовуються алгебричні структури для створення надійних систем шифрування та підпису. Водночас оптимальне кодування використовує теорію інформації та ймовірності для створення ефективних кодів. Одним із прикладів взаємодії цих галузей є застосування кодів для захисту переданих криптографічно зашифрованих повідомлень. Використання оптимальних кодів дає змогу ефективно передавати інформацію за мінімальних витрат ресурсів. Математична криптографія та оптимальне кодування є ключовими галузями, що забезпечують безпеку й ефективність обміну інформацією в інформаційному суспільстві. Їх взаємозв'язок виявляється у використанні спільних математичних концепцій для досягнення відповідних цілей, створюючи у такий спосіб основу для розвитку сучасних технологій безпеки та збереження інформації [4].

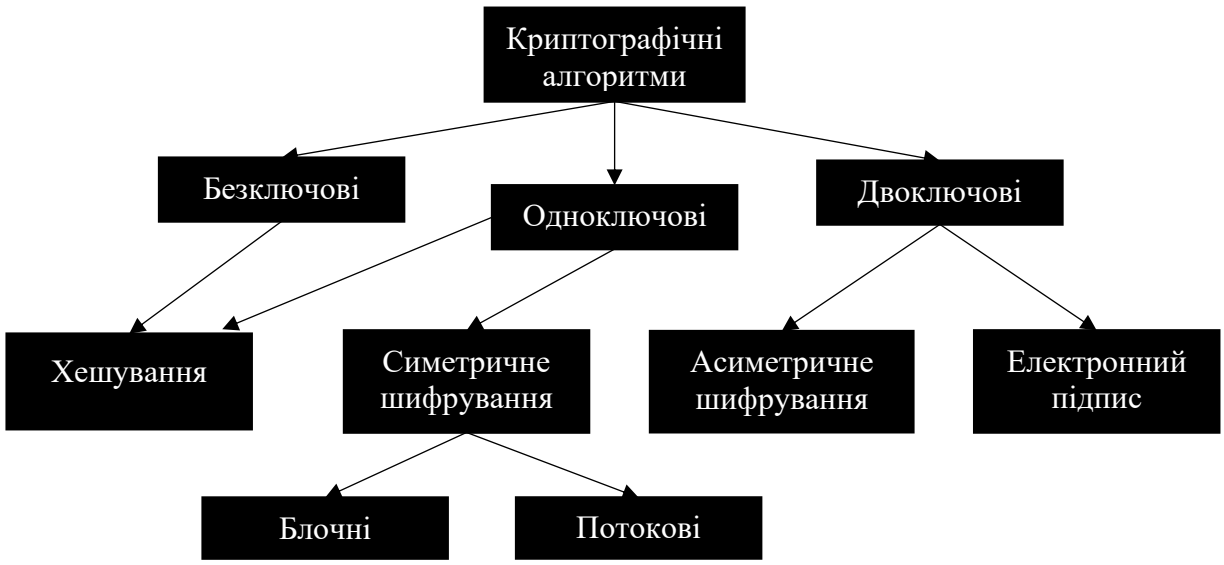


Рисунок 1. Структура криптографічних алгоритмів

Висновки. Обидві галузі спільно сприяють забезпеченню ефективної та безпечної передачі інформації, застосовуючи математичні концепції й алгоритми. Математична криптографія забезпечує захист, використовуючи складні математичні методи, тоді як оптимальне кодування максимізує ефективність використання ресурсів для ефективної обробки та передачі даних.

Список використаних джерел

1. Математичні основи криптографії: URL: https://cutt.ly/SwYzQnGY (дата звернення: 13.11.2023).
2. Wikipedia. URL: https://cutt.ly/IwYzQ2mz (дата звернення: 13.11.2023).
3. Оптимальне кодування. URL: https://cutt.ly/gwYzWSOm (дата звернення: 13.11.2023).
4. Математична криптографія та оптимальне кодування. URL: https://cutt.ly/gwqzMSDa (дата звернення: 13.11.2023).

УДК 004.8

Бурківський О. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Зелінська О. В., канд. техн. наук, доцент, в. о. завідувача кафедри інформаційних технологій

СУЧАСНІ ФРЕЙМВОРКИ МАШИННОГО НАВЧАННЯ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сучасний розвиток технологій у галузі машинного навчання визначається більшою мірою використанням потужних фреймворків. Ці інструменти не лише прискорюють розробку моделей, а й надають розробникам та дослідникам гнучкість і можливості для ефективного впровадження інтелектуальних систем. У публікації ми розглянемо кілька ключових сучасних фреймворків машинного навчання.

TensorFlow є високопродуктивним відкритим вихідним фреймворком для машинного та глибокого навчання, розробленим і підтримуваним Google. Запущений у 2015 р., він завоював широку популярність завдяки своїй гнучкості, ефективності та розширеною підтримкою глибокого навчання.

Особливості TensorFlow:

1. TensorFlow дає змогу розробникам вибирати між статичною та динамічною обчислювальною графікою. Це забезпечує гнучкість під час вибору оптимального підходу до розробки моделей.

2. Вона також має великий набір заздалегідь вбудованих шарів для різних видів мереж, а також оптимізаторів для тренування моделей із різною складністю.

3. Підтримка роботи на різноманітних апаратних платформах дає змогу ефективно використовувати ресурси для тренування великих моделей.

4. TensorFlow має активну та велику спільноту користувачів та розробників. Це забезпечує доступ до різноманітних ресурсів, зокрема документації, навчальних матеріалів та багатофункціональних модулів.

5. TensorFlow широко використовується у наукових дослідженнях, оскільки надає зручний інтерфейс для реалізації та тестування нових ідей у галузі машинного навчання. Багато компаній використовують TensorFlow для розробки та впровадження своїх продуктів, оскільки він забезпечує потужність і масштабованість [1, 2].

Рисунок 1. Логотип фреймворка «TensorFlow»

Scikit-learn є високорівневою бібліотекою для машинного навчання, розробленою на мові програмування Python. Вона надає простий і ефективний інтерфейс для роботи з різноманітними алгоритмами машинного навчання, що робить її популярною серед дослідників, студентів та інженерів з області даних.

Рисунок 2. Логотип фреймворка «Scikit-learn»

Особливості Scikit-learn:

1. Scikit-learn ставить перед собою мету забезпечити чіткий та однорідний інтерфейс для різних методів машинного навчання, що допомагає швидко розробляти й експериментувати з моделями.

2. Бібліотека передбачає широкий набір алгоритмів для класифікації, регресії, кластеризації, вимірювання важливості ознак та багато інших завдань машинного навчання.

3. Scikit-learn надає інструменти для оптимізації параметрів моделей, крос-валідації та оцінки їх продуктивності. Це допомагає забезпечити надійні та оптимальні результати.

4. Scikit-learn є популярним інструментом для вивчення основ машинного навчання завдяки своїй простоті та інтуїтивному інтерфейсу. Широкий набір

методів дає змогу використовувати бібліотеку для аналізу даних та прогнозування різноманітних явищ. Їх легко інтегрувати з іншими бібліотеками Python, як-от Matplotlib та Seaborn, що допомагає створювати візуалізації результатів та аналізу даних [3].

Сучасні фреймворки машинного навчання відкривають нові можливості для розробників та дослідників, даючи змогу їм ефективно впроваджувати та вдосконалювати моделі. Вибір конкретного фреймворка залежить від завдань, типів моделей, а також особистих вподобань. Загальна тенденція полягає в тому, що розвиток машинного навчання значною мірою зумовлений появою потужних та гнучких фреймворків, які продовжують визначати та трансформувати цю захоплюючу галузь.

Список використаних джерел

1. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. URL: https://www.tensorflow.org/about/bib (дата звернення: 06.12.2023).

2. Introduction to TensorFlow. URL: https://www.tensorflow.org/learn (дата звернення: 06.12.2023).

3. Scikit-learn: machine learning in Python – scikit-learn 1.3.2 documentation. URL: https://scikit-learn.org/stable/index.html (дата звернення: 06.12.2023).

УДК 004.021

Бурківський О. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ЗАСТОСУВАННЯ ДИНАМІЧНОГО ПРОГРАМУВАННЯ ДЛЯ ВИРІШЕННЯ ЗАДАЧІ ПРО НАЙДОВШУ ПОСЛІДОВНІСТЬ

Донецький національний університет імені Василя Стуса, м. Вінниця

Задача про найдовшу послідовність є однією з класичних задач алгоритмізації, яка виникає в різних контекстах, як-от обробка рядків, генетика, аналіз текстів тощо. Через це використання методів динамічного програмування може суттєво полегшити розв'язання проблеми та забезпечити ефективний алгоритм для пошуку найдовшої послідовності.

Динамічне програмування – це ефективний метод розв'язання складних задач, який має бути застосований у різних галузях, зокрема алгоритмізації, оптимізації та штучному інтелекті. Основні ідеї динамічного програмування включають:

1. Розбиття задачі на підзадачі. Динамічне програмування передбачає розбиття складної задачі на простіші підзадачі, що сприяє полегшенню розв'язання задачі, шляхом зменшення та більшої керованості.

2. Зберігання та використання попередніх результатів. Ключовим елементом динамічного програмування є збереження результатів розв'язку підзадач для подальшого використання, що дає змогу уникнути зайвих повторних обчислень та підвищити ефективність алгоритму.

3. Визначення рекурентних відносин. Динамічне програмування використовує рекурентні відносини для опису взаємозв'язків між задачами різного розміру. Це означає, що розв'язання більшої задачі може бути ефективно побудоване на основі розв'язань менших підзадач.

4. Визначення базових випадків. Для кожної задачі, що розв'язується динамічним програмуванням, визначаються базові випадки – найпростіші випадки, які можна вирішити без подальших рекурсивних викликів. Базові випадки відіграють ключову роль у рекурсивному розв'язанні задачі.

5. Побудова оптимального розв'язку знизу вгору. Динамічне програмування може використовуватися для побудови оптимального розв'язку знизу вгору, тобто розв'язання менших підзадач та поступове сходження до більшої задачі. Цей підхід дає змогу ефективно обчислити розв'язок для всіх можливих комбінацій підзадач.

6. Ітераційний підхід і таблиці для оптимізації. Динамічне програмування може бути реалізоване як ітераційний алгоритм, який використовує таблиці для збереження результатів обчислень підзадач. Це забезпечує оптимізацію в плані часу та пам'яті.

Задача знаходження найдовшої зростаючої послідовності (LIS) виникає в різних областях, як-от аналіз даних, оптимізація та комп'ютерна наука. Мета полягає в тому, щоб знайти найбільшу послідовність чисел у масиві, де кожне число більше від попереднього. Приклад коду на мові програмування C# для задачі знаходження найдовшої зростаючої послідовності (LIS) з допомогою динамічного програмування (рис. 1).

Постановка задачі: існує масив чисел arr довжиною n. Необхідно визначити найдовшу послідовність індексів i_1, i_2, …..i_k, де $0 <= i_1 < i_2 <... < i_k < n$, таку, що $arr[i_1] < arr[i_2] <... < arr[i_k]$. Довжина цієї найдовшої зростаючої підпослідовності є ключовим показником ефективності розв'язання задачі. Приклад виконання програми наведено на рис. 2.

Тобто якщо вхідний масив чисел {10, 22, 9, 33, 21, 50, 41, 60, 80}, то найбільша зростаюча послідовність у цьому масиві – {10, 22, 33, 50, 60, 80}.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LongestIncreasingSubsequence
{
    class Program
    {
        static int LongestIncreasingSubsequenceLength(int[] arr, int n)
        {
            int[] lis = new int[n];
            int maxLisLength = 1;

            for (int i = 0; i < n; i++)
            {
                lis[i] = 1;

                for (int j = 0; j < i; j++)
                {
                    if (arr[i] > arr[j] && lis[i] < lis[j] + 1)
                    {
                        lis[i] = lis[j] + 1;
                    }
                }

                if (maxLisLength < lis[i])
                {
                    maxLisLength = lis[i];
                }
            }

            return maxLisLength;
        }

        static void Main()
        {
            int[] sequence = { 10, 22, 9, 33, 21, 50, 41, 60, 80 };
            int n = sequence.Length;

            int result = LongestIncreasingSubsequenceLength(sequence, n);
            Console.OutputEncoding = System.Text.Encoding.UTF8;
            Console.WriteLine($"Довжина найдовшої зростаючої підпослідовності становить {result}");
        }
    }
}
```

Рисунок 1. Приклад коду

```
Довжина найдовшої зростаючої підпослідовності становить 6
```

Рисунок 2. Приклад виконання

Отже, знаходження найдовшої зростаючої послідовності з допомогою динамічного програмування використовується для оптимізації торгових стратегій, аналізу даних, а також у задачах, пов’язаних із послідовністю подій чи елементів у великих наборах даних.

Список використаних джерел

1. Задачі динамічного пошуку. URL: http://www.tsatu.edu.ua/kn/wp-content/uploads/sites/16/zadachi-dynamichnoho-prohramuvannja.pdf (дата звернення: 29.11.2023).

2. Застосування динамічного програмування. URL: https://ua5.org/osnprog/1907-dynamichne-programuvannya.html (дата звернення: 29.11.2023).

УДК 004.021

Грибаніна А. О., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

АНАЛІЗ АЛГОРИТМІВ БІНАРНОГО ТА ЛІНІЙНОГО ПОШУКУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Алгоритми пошуку призначені для пошуку конкретного значення в наборі даних. Якщо значення наявні в наборі, то пошук вважається успішним, і процес пошуку дає місце розташування цього значення в наборі даних. Інакше кажучи, якщо значення відсутнє у наборі, процес пошуку відображає відповідне повідомлення, і у цьому випадку пошук називається невдалим. Для пошуку ми використовуємо ключ елементу даних.

Пошук є однією з найчастіше використовуваних операцій у програмуванні. Взагалі різноманітних видів пошуку дуже багато і вони розрізняються між собою способом організації даних. Але у кожному алгоритмі пошуку є свої переваги й недоліки. Загальний алгоритм пошуку можна описати так:

1. Обчислення елементу, що часто передбачає отримання значення елементу, ключа елементу та ін.
2. Порівняння елементу з еталоном або порівняння двох елементів.
3. Перебір елементів множини, тобто проходження по елементах масиву.

Найпростіший алгоритм для пошуку інформації – це лінійний (послідовний) алгоритм. Він полягає в тому, що переглядаються один за одним усі елементи списку і відшукується їх відповідність для заданого елементу-ключа. Якщо такий елемент буде знайдено, то програма повертає його адресу (індекс елементу), якщо ні, то повертається нульове значення [1].

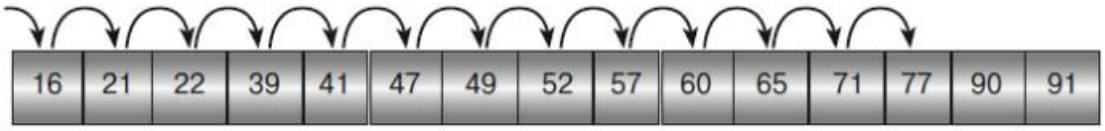


Рисунок 1. Приклад лінійного пошуку [1]

На відміну від бінарного пошуку, лінійний може працювати як з масивами, так і зі зв'язними списками, де можна перестрибнути з однієї частини в іншу. Списки можуть бути і несортованими, але з відсортованими алгоритм економить значну частину часу: він зупиняється, дійшовши до елементу зі значенням, що перевищує цільове, і не веде подальший пошук того, чого явно не існує. Перевагами такого пошуку є простота його реалізації, він не потребує додат-

кового об’єму пам’яті або додаткової роботи з функціями. Це дає змогу працювати вже під час отримання даних.

Бінарний пошук – це алгоритм пошуку, який використовується в відсортованому масиві шляхом багаторазового ділення інтервалу навпіл. Бінарний пошук застосовується для відсортованих множин. Пошук починається зі знаходження центрального елементу заданого масиву. Значення цього елементу порівнюється зі значенням елементу-ключа. Якщо потрібне значення-ключ збігається з центральним, то пошук завершено. Якщо воно або більше, або менше, то пошук переміщується ліворуч або праворуч від центрального елементу і відбувається в цьому масиві.

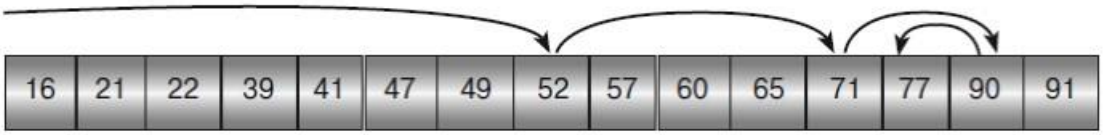


Рисунок 2. Приклад бінарного пошуку числа 77 [1]

Цей алгоритм пошуку є швидшим, порівняно з послідовним пошуком. Проте він може використовуватись лише на впорядкованій множині даних. Його доцільно застосовувати, якщо елементи масиву впорядковано, тому цей алгоритм пошуку не використовується для знаходження максимального або мінімального елементу, оскільки у відсортованому масиві ці елементи містяться на початку та в кінці масиву відповідно, залежно від того, як відсортований масив, за зростанням чи спаданням. Тому, хоча бінарний пошук ефективніший за лінійний, його не завжди можна застосувати [2].

Отже, сьогодні алгоритми пошуку застосовуються у багатьох сферах, адже їх розвиток набув величезних масштабів. Не можна точно сказати, що один алгоритм буде ефективніший від іншого, поки перед нами не стоїть умова задачі, яку потрібно вирішити. Для різних типів прикладних задач, для різних масивів та сфер даних підходять різні алгоритми пошуку. Найважливіше вміти використовувати знання про них.

Список використаних джерел

1. Креневич А. П. Алгоритми і структури даних: підручник. Київ: ВПЦ Київський Університет, 2021. 200 с. URL: https://www.mechmat.univ.kiev.ua/wp-content/uploads/2021/09/ pidruchnyk-alhorytmy-i-struktury-danykh.pdf.

2. Освітній портал iua5. URL: https://ua5.org/osnprog/418-algoritmi-poshuku.html

3. Грудзинський Ю. Є. Алгоритми та структури даних: навч. посіб. Київ: НТУУ КПІ ім. Ігоря Сікорського, 2022. 215 с. URL: https://ela.kpi.ua/bitstream/123456789/56538/1/Alhorytmy_ta_struktury%20danykh_Navch_posib.pdf

УДК 004.021

Гуцуляк Д. В., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

АНАЛІЗ ЕФЕКТИВНОСТІ АЛГОРИТМІВ ПОШУКУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Алгоритми та структури даних у сучасному світі відіграють ключову роль у розвитку програмного забезпечення. Багато науковців приділяють увагу історії створення самих алгоритмів пошуку. Якби люди зберігали сотні записів, щоб мати легкий доступ до будь-якого з них пізніше, то для ефективного пошуку цих записів їх необхідно було б зберігати та індексувати. Ця проблема була успадкована, коли перші дослідники почали переміщувати записи на комп'ютери. Галузь пошукових алгоритмів добре вивчена, має довгу історію, яка почалася в математиці і швидко розширилася після винаходу комп'ютерів. Це почалося скромно з простими алгоритмами, як-от лінійний і двійковий пошук, але вони швидко зростали і незабаром їх стало більше [1].

Насамперед розглянемо основні алгоритми пошуку, як-от лінійний пошук, лінійний пошук з бар'єром, бінарний пошук. Кожен із цих алгоритмів має свої переваги та недоліки залежно від контексту застосування та обсягу даних. Ідея лінійного пошуку – проглядати почергово елементи масиву, доки не буде знайдено шуканий елемент або не буде досягнуто кінця масиву. Псевдокод (рис. 1):

```
i=0,
поки (i < N і a[i] ! = Key),
{ i=i+1},
якщо i=N,
то елемент відсутній,
інакше елемент має номер і
```

Рисунок 1. Псевдокод лінійного пошуку

Лінійний пошук з бар'єром. Ідея – як і у попередньому випадку, елементи проглядаються по черзі, але для зменшення кількості порівнянь після останнього елементу додається елемент із ключем, що дорівнює шуканому значенню. Псевдокод наведено на рис. 2.

Бінарний пошук. Ідея – пошук, що реалізується в упорядкованому масиві. Шукане значення треба порівняти з ключем середнього елементу, внаслідок

цього порівняння визначити, у якій половині масиву знаходиться шуканий ключ, і знову застосувати ту саму процедуру до половини масиву. Процес припиняється, коли буде знайдено елемент або коли «довжина» таблиці стане менше 1 [2]. Псевдокод (рис. 3).

```
a[N] = Key; i = 0,
поки (a[i] ! = Key),
{ i = i+1},
якщо i = N,
то елемент відсутній,
інакше елемент має номер i
```

Рисунок 2. Псевдокод лінійного пошуку з бар'єром

```
L = 0; R = N–1,
поки L < R,
{
m =(L+R) / 2,
якщо a[m] < Key,
то L = m+1,
інакше R = m,
}
якщо a[R] = Key,
то елемент має номер R,
інакше елемент відсутній
```

Рисунок 3. Псевдокод бінарного пошуку

Із цього огляду можем зробити висновок, що лінійний пошук простий, але неефективний для великих обсягів даних; лінійний пошук із бар'єром зменшує кількість порівнянь, а бінарний пошук ефективний лише для відсортованих даних, проте швидко знаходить шукане значення у відсортованому масиві.

Одним із поширених алгоритмів пошуку є пошук на основі хеш-таблиць. Хеш-таблиці, забезпечуючи швидкий доступ за ключем, використовуються для реалізації асоціативних масивів [3].

Порівняння ефективності різних алгоритмів пошуку у контексті різних структур даних розкриває ключові особливості кожного методу залежно від обсягу та властивостей даних. Лінійний пошук простий, але має обмежену ефективність для великих наборів даних. Лінійний пошук із бар'єром зменшує кількість порівнянь, проте не є універсальним рішенням для обробки великих обсягів даних.

Водночас бінарний пошук ефективний у відсортованих наборах, проте його ефективність залежить від ступеня відсортованості даних. Під час порівняння з хеш-таблицями, які забезпечують швидкий доступ за ключем, виявлено, що вони ефективні для асоціативних масивів та операцій з великими обсягами даних. Такий аналіз підкреслює важливість вибору конкретного алгоритму у поєднанні зі структурою даних, зокрема враховуючи обсяг і характеристики даних для оптимального використання в конкретних умовах [4].

Отже, вибір алгоритму пошуку має критичне значення для оптимального використання ресурсів та швидкості обробки інформації. Аналіз лінійного пошуку, лінійного пошуку з бар'єром та бінарного пошуку в контексті різних структур даних підкреслює їх переваги та обмеження. Для великих обсягів даних бінарний пошук впритул до відсортованих масивів є ефективним, тоді як хеш-таблиці виявляються корисними для швидкого доступу за ключем.

Список використаних джерел

1. Tudor Octavian Pocola. The evolution of search algorithms over time. Delft University of Technology. URL: https://filelist.tudelft.nl/Websections/Honours%20Exhibition/Scientific%20 Writing/The_evolution_of_search_algorithms_over_time.pdf

2. Пошукові алгоритми. Алгоритми і структури даних: вебсайт. URL: http://elcat.pnpu.edu.ua/docs/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B8%20%D1%96%20%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B8%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85/lab2_search.html

3. Поняття структур даних, масив, список, словник, стек, черга, хеш-таблиця. Структури даних: вебсайт. URL: https://ua5.org/struktury-danyh/1621-ponyattya-struktur-danyh-masyv-spysok-slovnyk-stek-cherga-hesh-tablyczya.html

4. Introduction to Algorithms. 3rd Edition – H. Thomas, E. Charles, L. Ronald, C. Stein. *MIT PRESS*, 2009. 1292 p.

УДК 004.07

Диркач Х. М., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

СОРТУВАННЯ ТА ЙОГО ВПЛИВ НА АРХІТЕКТУРУ ПАМ'ЯТІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Алгоритм сортування визначається як алгоритм, що розміщує елементи якоїсь однотипної послідовності даних (масив, список, файл) у певному по-

рядку, який може бути або числовим порядком, або лексикографічним порядком, або будь-яким іншим заданим користувачем порядком.

Мета сортування – полегшити подальший пошук, оновлення, виключення, включення елементів у структуру даних. На відсортованих даних легше визначити, чи є пропущені елементи, чи всі елементи перевірені, легше знайти загальні елементи двох однотипних структур, злити їх воєдино. Сортування є важливим засобом для прискорення роботи практично будь-якого алгоритму, в якому потрібно часте звертання до певних елементів структури даних.

Ефективність алгоритму сортування може залежати від низки факторів:

- кількості сортованих елементів;
- діапазону і розподілу значень сортованих елементів;
- ступеня початкової відсортованості елементів;
- характеристик алгоритму (складність, вимоги до пам'яті тощо);
- місць розміщення елементів (в оперативній пам'яті (масив) або на диску (файл)).

Основними вимогами до алгоритмів сортування, як і до будь-яких алгоритмів, є вимоги до пам'яті і часу виконання. Існує безліч класичних алгоритмів сортування. Деякі з них не надто швидкі, проте мають невеликі вимоги до пам'яті, інші, навпаки, – дуже швидкі, проте потребують значних ресурсів пам'яті. Розглянемо кілька основних класичних алгоритмів сортування, що використовуються для сортування послідовностей чисел.

Бульбашкове сортування (також використовується термін сортування обміном, англ. *Bubble sort*) є найпростішим з алгоритмічного погляду, алгоритмом сортування. Його ідея полягає у тому, що здійснюється кілька проходів по списку, під час кожного з яких порівнюють пари сусідніх елементів. Якщо елементи стоять неправильно, вони міняються місцями. Кожен прохід по списку ставить наступне найбільше значення на його правильну позицію.

Алгоритм сортування вибором дуже подібний до бульбашкового сортування, проте є дещо оптимальнішим, оскільки за кожен прохід по списку відбувається лише одна операція перестановки елементів. Його ідея полягає у тому, що на кожному кроці відбувається лінійний пошук найбільшого елементу серед невідсортованої частини списку, який переставляється на відповідну позицію (крайню праву / ліву позицію невідсортованої частини списку).

Під час сортування вставкою, кожен наступний елемент вставляється у потрібну позицію так, щоб підсписок лишався відсортованим. Спочатку вважаємо, що підсписок з одного елементу (що знаходиться на нульовій позиції) відсортований. Далі кожен наступний елемент на кожному проході вставляється у відповідну позицію. Під час цього, може виникнути ситуація, коли для вставки необхідно зсунути частину елементів списку.

Сортування злиттям працює за таким принципом: невідсортовану вхідну множину довільно розбивають на підмножини. Потім ці підмножини сортують окремо цим самим методом і об'єднують в одну множину. У цьому методі діє відомий принцип «розділяй і володарюй».

Алгоритм швидкого сортування використовується для того, щоб отримати ті ж переваги у швидкості, що і сортування злиттям, не використовуючи під час цього додатку пам'ять. Однією з особливостей алгоритму є те, що він використовує значно меншу кількість порівнянь і перестановок елементів. Основна ідея полягає у виборі «опорного» елементу з масиву та розбитті масиву на два підмасиви: один, що містить елементи, менші за опорний, інший – більші.

Сортування впливає на архітектуру пам'яті через вимоги до часу та простору. Бульбашкове сортування та сортування вибором хоча й прості в реалізації, є менш ефективними на великих обсягах даних. Сортування вставкою підходить для відсортованих або малих масивів. Сортування злиттям та швидке сортування дають швидші результати, але вимагають більше ресурсів. Швидке сортування, з використанням стратегії розбиття на підмасиви з допомогою опорного елементу, характеризується швидкістю та меншими вимогами до пам'яті, тому є оптимальним у багатьох випадках.

Під час вибору алгоритму важливо брати до уваги конкретні умови використання та характеристики даних, щоб оптимізувати роботу програми і забезпечити оптимальне використання архітектури пам'яті.

Список використаних джерел

1. Коротєєва Т. О. Алгоритми та структури даних: навч. посіб., Львів: Видавництво Львівської політехніки, 2014. 280 с.

2. Креневич А. П. Алгоритми і структури даних: підручник. Київ: ВПЦ Київський Університет, 2021. 200 с.

3. Грудзинський Ю. Є. Алгоритми та структури даних. Київ: КПІ ім. Ігоря Сікорського. 215 с. URL: https://ela.kpi.ua/bitstream/123456789/56538/1/Alhorytmy_ta_struktury%20danykh_Navch_posib.pdf

УДК 004.414:004.43:519.683:004.9

Діброва І. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Вєтров О. С., страший викладач кафедри прикладної математики та кібербезпеки

СОРТУВАННЯ МАСИВІВ МЕТОДОМ БУЛЬБАШКИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сортування масиву означає розташування всіх елементів масиву в певному порядку. Сортування даних робить пошук у масивах ефективнішим як для людини, так і для комп'ютера. Сортування є фундаментальним поняттям в інформатиці та програмуванні. Цей процес організовує елементи в певному порядку, щоб полегшити їх подальше використання та аналіз. Ефективні методи сортування є важливим фактором у вирішенні різноманітних завдань, від оптимізації пошуку до підготовки даних до подальшої обробки.

У мовах програмування важливо мати доступ до ефективних алгоритмів сортування, щоб ефективно обробляти дані. Багато мов програмування надають стандартні бібліотеки сортування, які використовують різні алгоритми, як-то швидке сортування, сортування злиттям, і в нашому випадку, – сортування бульбашками [1].

Бульбашковий метод – це простий алгоритм сортування, який працює шляхом послідовного порівняння та обміну місцями сусідніх елементів, доки весь масив не буде вирівняно. Це алгоритм, який порівнює два сусідні елементи і міняє їх місцями, поки вони не опиняться в потрібному порядку. Подібно до того, як бульбашки у воді піднімаються на поверхню, кожен елемент масиву переміщується в кінець кожної ітерації. Хоча він не є найефективнішим для великих обсягів даних, він відображає основні принципи сортування і може бути використаний для розуміння базових концепцій сортування.

Загалом роботу алгоритму можна поділити на дві основні ітерації. Перша ітерація – порівняння та обмін. Друга – ітерації, що залишилися [2].

Порівняння розпочнеться з першого елементу, який необхідно порівняти з другим елементом. Якщо перший елемент більший за другий, треба поміняти їх місцями. Потім так само другий і третій елементи. Якщо вони розташовані в неправильному порядку, їх також треба поміняти місцями. Цей процес повторюється до останнього елементу.

Другий етап – цей самий процес триває для решти ітерацій. Після кожної ітерації найбільший елемент серед несортованих елементів розміщується в кінці.

У кожній ітерації порівняння відбувається до останнього невідсортованого елементу.

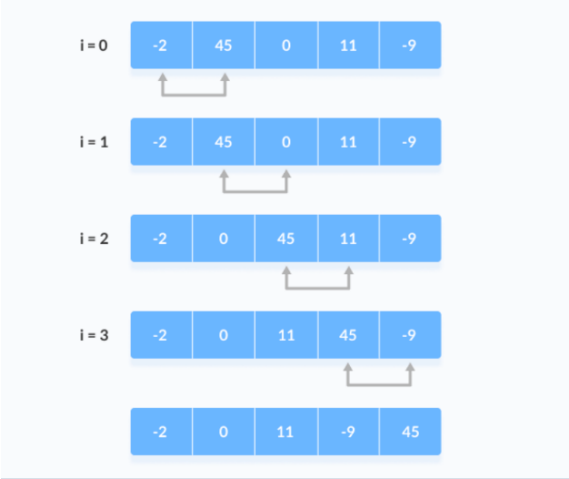


Рисунок 1. Перша ітерація. Порівняння суміжних елементів

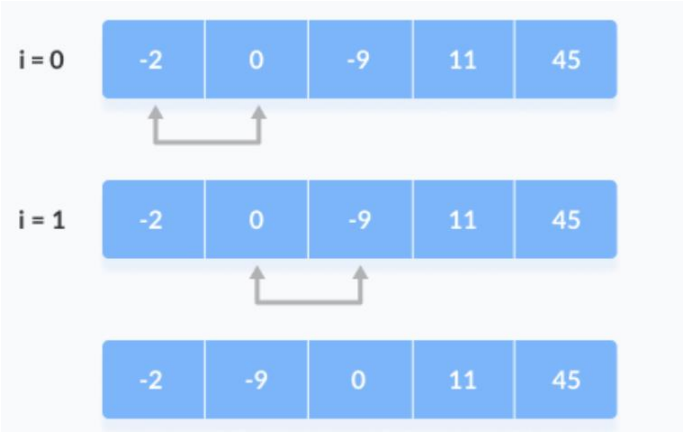


Рисунок 2. Порівняння сусідніх елементів

Масив можна вважати відсортованим, коли всі невідсортовані елементи розміщено на їх правильних позиціях.

Рисунок 3. Відсортований масив

На мові програмування Python лістинг програми для сортування масиву матиме такий вигляд:

```
# Сортування бульбашкою на мові Python
```

```
def bubbleSort(array):

  # Цикл для доступу до кожного елементу масиву
  for i in range(len(array)):

    # Цикл для порівняння елементів масиву
    for j in range(0, len(array) - i - 1):

      # Порівняння двох сусідніх елементів
      # Змініть > на <, щоб сортувати за спаданням
      if array[j] > array[j + 1]:

        # Обмін елементів, якщо вони не в потрібному порядку
        temp = array[j]
        array[j] = array[j + 1]
        array[j+1] = temp

data = [-2, 45, 0, 11, -9]

bubbleSort(data)

print('Відсортований масив за зростанням:')
print(data)
```

Ще одним важливим завданням є оптимізація алгоритму та пришвидшення роботи програми. У наведеному вище алгоритмі всі порівняння виконуються, навіть якщо масив уже відсортовано. Це збільшує час виконання.

Для вирішення цієї проблеми можна ввести додаткову змінну swapped. Якщо елементи міняються місцями, то значення swapped набуває значення true. В іншому випадку – значення False. Після ітерації значення swapped дорівнює false, якщо перестановки не відбулося. Це означає, що елементи вже відсортовано, і подальші ітерації не потрібні. Це зменшує час виконання і допомагає оптимізувати бульбашкове сортування [3].

```
# Оптимізоване сортування бульбашкою на мові Python

def bubbleSort(array):

  # цикл для проходження крізь кожен елемент масиву
  for i in range(len(array)):

    # відстежуємо, чи відбувся обмін
    swapped = False

    # цикл для порівняння елементів масиву
    for j in range(0, len(array) - i - 1):
```

```
        # порівнюємо два сусідні елементи
        # замініть > на <, щоб сортувати за спаданням
        if array[j] > array[j + 1]:

          # обмін відбувається, якщо елементи
          # не в потрібному порядку
          temp = array[j]
          array[j] = array[j+1]
          array[j+1] = temp

          swapped = True

      # якщо обмін не відбувся, це означає, що масив уже впорядкований
      # і немає потреби у подальших порівняннях
      if not swapped:
        break

data = [-2, 45, 0, 11, -9]

bubbleSort(data)

print('Відсортований масив за зростанням:')
print(data)
```

Підсумовуючи, варто зазначити, що природа сортування масивів є важливим етапом у програмуванні та інформатиці. Спрощуючи пошук та аналіз даних, сортування забезпечує ефективне використання та обробку інформації як для користувачів, так і для комп'ютерів. Алгоритми сортування є важливим аспектом вирішення різноманітних завдань, розширюючи можливості від оптимізації пошуку до підготовки даних до подальшої обробки. Хоча метод бульбашок не завжди є оптимальним для великих обсягів даних, він є гарним вступом до базових понять сортування і може бути використаний для навчання основам алгоритму.

Список використаних джерел

1. Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К. (2019). Введення в алгоритми. Київ: К.І.С.

2. Sedgewick, R., Wayne, K. (2011). Algorithms, 4th edition. Addison-Wesley Professional.

3. Mühlegg, S., Scherer, S., Floreano, D. (2020). Path Planning and Control of UAVs using Model Predictive Control and Reinforcement Learning. Robotics and Autonomous Systems, 128, 103–628.

УДК 004.4

Зимич А. П., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Фриз І. В., канд. фіз.-мат. наук, старший викладач кафедри інформаційних технологій

ОГЛЯД МАТЕМАТИЧНИХ БІБЛІОТЕК PYTHON

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. У світі наук про дані та їх дослідження мова програмування Python займає одну із центральних позицій, надаючи різноманітні математичні бібліотеки для обробки, аналізу та візуалізації даних. Ці інструменти стали невід'ємною частиною роботи дослідників, аналітиків та розробників, надаючи їм зручні засоби для вирішення практичних завдань.

Опис бібліотек. Бібліотека NumPy (скорочено від Numerical Python) створена для ефективної роботи з багатовимірними масивами числових даних завдяки зусиллям Тревіса Оліфанта. Вона стала основою для наукових обчислень та аналізу даних у мові програмування Python. Особливостями NumPy [1, 2] є:

- NumPy надає ефективні структури даних для зберігання *n*-вимірних масивів даних та здійснення операцій над числовими даними;
- бібліотека містить великий набір математичних функцій, починаючи від простих операцій (наприклад, синус, косинус) і закінчуючи складними алгоритмами лінійної алгебри;
- може інтегруватися з кодом на низькорівневих мовах, як-от C та Fortran, що сприяє оптимізації продуктивності;
- завдяки оптимізованим операціям та підтримці багатопотоковості NumPy забезпечує високу продуктивність під час обробки великих обсягів даних;
- NumPy є проєктом з відкритим кодом, що дає змогу користувачам адаптувати та розширювати його під свої потреби.

Бібліотека SciPy є відкритою бібліотекою для наукових обчислень у Python. Розроблялася Тревісом Оліфантом для розширення функціональності NumPy. SciPy надає високорівневі алгоритми для оптимізації, інтеграції, обробки сигналів, статистики та багатьох інших галузей, ставши ключовим компонентом екосистеми Python для наукових та інженерних розрахунків. Відмінності з NumPy та її особливості [3]:

- SciPy надає широкий набір функцій для чисельного інтегрування, розв'язування диференціальних рівнянь, оптимізації та інших математичних і наукових завдань;
- SciPy надає інструментарій алгоритмічної реалізації методів оптимізації, зокрема методи мінімізації функцій та глобальної оптимізації, що робить її

корисною для вирішення задач машинного навчання та інших оптимізаційних задач;

- бібліотека забезпечує засоби інтеграції з кодом, написаним низькорівневими мовами, що розширює її можливості та покращує продуктивність;
- бібліотека надає широкий спектр інструментарію для обробки та аналізу інформації на основі статистичних методів, зокрема тести для перевірки гіпотез, дисперсійний аналіз тощо;
- SciPy взаємодіє з NumPy, Matplotlib та іншими бібліотеками, створюючи сукупну екосистему для наукових обчислень та аналізу даних у Python;
- як і NumPy, SciPy є проєктом з відкритим кодом, що сприяє активній участі спільноти та її постійному розвитку.

SciPy розширює можливості NumPy, а деякі часті дії в ній реалізовані як окремі функції, тому бібліотека спрощує роботу зі складними завданнями з використанням просунутої математики. Для низки простіших завдань вона надмірна, буде достатньо NumPy чи інших бібліотек.

Matplotlib написана і підтримувалася в основному Джоном Хантером та призначена для створення візуалізацій на мові програмування Python. Згенеровані у різних форматах зображення можуть бути використані в інтерактивних графіках, наукових публікаціях, графічному інтерфейсі користувача, вебзастосунках тощо [4]. Основними можливостями Matplotlib є робота з різними видами графічного подання даних:

- графіки;
- діаграми розсіювання;
- стовпчасті діаграми та гістограми;
- кругові діаграми;
- діаграми стовбур-листя;
- контурні графіки;
- поля градієнтів;
- спектральні діаграми.

Приклади використання бібліотек Python. Далі розглянемо прості приклади застосування описаних бібліотек до розв'язування певних задач [5].

1. *Реалізація методу найменших квадратів для лінійних функцій.* Код, наведений у Лістингу 1, створює набір випадкових даних (*x*, *y*), де *y* залежить від *x* лінійно з додаванням випадкового шуму. Потім він використовує метод найменших квадратів (least-squares) для знаходження коефіцієнтів лінійної регресії (alpha), які найкраще відповідають цим даним. Нарешті код будує графік вихідних даних та лінії регресії (рис. 1).

```
import numpy as np
import matplotlib.pyplot as plt
```

```
NUM_POINTS = 100
x = np.random.rand(NUM_POINTS)
y = 2.5 * x + 0.5 * np.random.randn(NUM_POINTS)
A = np.vstack([x, np.ones(len(x))]).T
alpha = np.linalg.lstsq(A, y, rcond=None)[0]

print("(alpha):", *alpha)

plt.scatter(x, y, label='Inital data')
plt.plot(x, alpha[0] * x + alpha[1], 'r', label='Regression')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid(True)
plt.show()
```

Лістинг 1. Код програми наближення за методом найменших квадратів

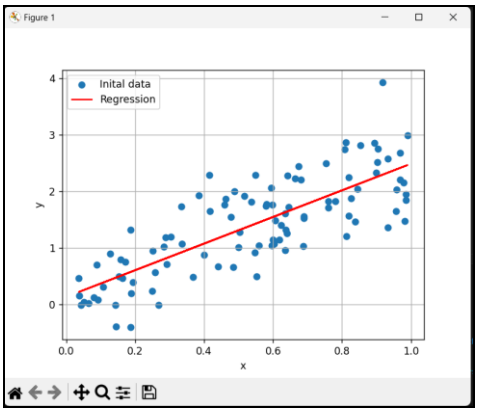


Рисунок 1. Графічне зображення результату програми

2. *Приклад програми на знаходження екстремумів функції багатьох змінних.* Програма з Лістингу 2 оптимізує функцію двох змінних методом Nelder-Mead. Код знаходить мінімум цієї функції, починаючи з початкового припущення [1, 1], а потім виводиться у вигляді тривимірного графіка (рис. 2).

```
import numpy as np
from scipy.optimize import minimize
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def multivariable_function(x):
    return x[0]**2 + x[1]**2 + 15

initial_guess = [1, 1]
```

```
    result = minimize(multivariable_function, initial_guess, method='Nelder-Mead')

    minima_location = result.x
    minima_value = result.fun

    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')

    x = np.linspace(-2, 2, 100)
    y = np.linspace(-2, 2, 100)
    X, Y = np.meshgrid(x, y)
    Z = multivariable_function([X, Y])

    ax.plot_surface(X, Y, Z, alpha=0.5, cmap='viridis', edgecolors='w', linewidth=0.5,
label='Function')
    ax.scatter(*minima_location, minima_value, color='red', marker='o', s=100,
label='Minimum')
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')

    ax.legend()

    ax.text2D(0, -0.05, f'Minimum Location: ({minima_location[0]:.4f};
{minima_location[1]:.4f})', transform=ax.transAxes)
    ax.text2D(0, -0.1, f'Minimum Value: {minima_value:.4f}', transform=ax.transAxes)

    plt.show()
```

Лістинг 2. Код програми для знаходження екстремумів функції

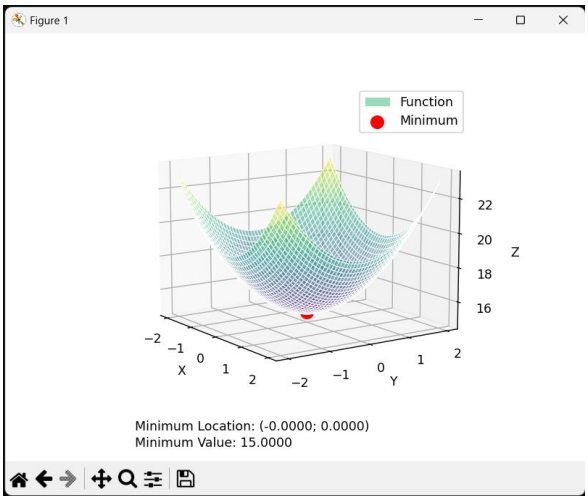


Рисунок 2. Графічне зображення результату програми

Висновки. NumPy виступає як основна бібліотека для обробки числових даних, надаючи ефективні структури для зберігання та операції над багатовимірними масивами. SciPy, яка базується на NumPy, доповнює його високорівневими алгоритмами для різних математичних та наукових задач, як-от оптимізація, обробка сигналів, статистика тощо. Matplotlib надає інструменти для візуалізації даних у різних форматах, що робить його потужним інструментом для графічного подання результатів аналізу.

Список використаних джерел

1. NumPy. *Wikipedia*. URL: https://en.wikipedia.org/wiki/NumPy (дата звернення: 26.11.2023).

2. NumPy: the absolute basics for beginners. NumPy v1.26 Manual. URL: https://numpy.org/doc/stable/user/absolute_beginners.html (дата звернення: 26.11.2023).

3. SciPy. *Wikipedia*. URL: https://en.wikipedia.org/wiki/SciPy (дата звернення: 26.11.2023).

4. Matplotib. *Wikipedia*. URL: https://en.wikipedia.org/wiki/Matplotlib (дата звернення: 26.11.2023).

5. Kong Q., Siauw T., Bayen A. Python Programming And Numerical Methods: A Guide for Engineers and Scientists. URL: https://pythonnumericalmethods. berkeley.edu/notebooks/Index.html (дата звернення: 26.11.2023).

УДК 004.021

Ілик В. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

СТЕКИ ТА АЛГОРИТМИ З ВИКОРИСТАННЯМ СТЕКІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Стеки та алгоритми з використанням стеків є ключовими компонентами у галузі інформатики та програмування. Стек, який працює за принципом Last In, First Out (LIFO), являє собою важливий інструмент для зберігання та управління даними. З розвитком хмарних технологій та збільшенням обчислювальних завдань ця тематика стає ще більш актуальною.

В обличчі складних завдань у сфері обчислень та зростання обсягів даних розробники стикаються з потребою оптимізації алгоритмів та раціонального використання ресурсів. Стеки відіграють важливу роль у вирішенні цих завдань, сприяючи підвищенню продуктивності та швидкості виконання програм.

Останні дослідження в галузі стеків та алгоритмів зосереджуються на поліпшенні швидкодії, зменшенні витрат пам’яті та розширенні можливостей застосування стеків. Нові методи оптимізації та адаптації стеків до специфічних вимог різноманітних завдань досліджуються для досягнення оптимальних результатів [1].

Метою цієї роботи є систематизація знань з області стеків та алгоритмів, їх використання в сучасному програмуванні та дослідження можливостей покращення ефективності застосування стеків.

1. Провести аналіз наявних стеків та алгоритмів [2].

2. Розглянути застосування стеків у великих обчислювальних завданнях та хмарних технологіях.

3. Розробити нові методи оптимізації та адаптації стеків до специфічних вимог.

У процесі реалізації поставлених завдань важливо розглянути наявні стеки та алгоритми, щоб визначити їх ефективність та придатність для вирішення різноманітних задач. Згідно з аналізом Cormen et al. [2] виявлено, що оптимізація стеків може відігравати ключову роль у поліпшенні продуктивності програм. Це особливо актуально в сучасних умовах, коли завдання стають все більш складними, а обсяги даних швидко зростають.

Важливим кроком у дослідженні стеків є їх застосування у великих обчислювальних завданнях та хмарних технологіях. Із допомогою стеків можна ефективно розподіляти завдання та оптимізувати використання ресурсів, що є ключовим у високопродуктивних обчисленнях [3].

У процесі дослідження були розроблені нові методи оптимізації та адаптації стеків до специфічних вимог. Ці методи спрямовані на досягнення оптимальних результатів у різноманітних сценаріях використання. Водночас важливо враховувати не лише швидкодію, але й витрати пам’яті, оскільки ефективне використання ресурсів стає все більш актуальним у світі зростаючих обчислювальних завдань.

Отримані результати підтверджують, що оптимізація та вдосконалення стеків може сильно вплинути на продуктивність програм і ефективність використання ресурсів. Застосування стеків у хмарних технологіях дає змогу підвищувати гнучкість та масштабованість систем, а нові методи оптимізації стеків відкривають шляхи до розв’язання різноманітних завдань.

Висновок дослідження вказує на те, що вивчення та оптимізація стеків і алгоритмів, зокрема їх адаптація до конкретних вимог, є ключовими факторами для подальшого розвитку сучасних програмних систем. Результати дослідження свідчать про великий потенціал вдосконалення стеків у вирішенні сучасних обчислювальних задач.

Список використаних джерел

1. Stephens R. Essential Algorithms: A Practical Approach to Computer Algorithms Using. John Wiley Sons, 2013. 624 p.
2. Korman T. H. Algorithms. Unlocked. The MIT Press, Massachusetts Institute of Technology, 2013. 207 p.
3. Sedgewick R., Wayne K. Algorithms. Addison-Wesley Professional, 2020. 956 p.

УДК 330.3

Костенко Р. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

СИСТЕМНИЙ АНАЛІЗ ТА МОДЕЛЮВАННЯ ЕКОНОМІЧНИХ КРИЗ

Донецький національний університет імені Василя Стуса, м. Вінниця

Економічні кризи, які виникають періодично у глобальній та регіональній економіці, завжди були предметом серйозної уваги для дослідників, аналітиків та урядових органів. Вони можуть мати катастрофічні наслідки для суспільства, зокрема втрату робочих місць, падіння доходів, фінансові кризи та соціальні нестабільності. У цьому контексті системний аналіз та моделювання економічних криз стають дуже важливими інструментами для розуміння, передбачення й управління цими складними явищами. Цей підхід полягає у створенні математичних та інших моделей, які дають змогу аналізувати причини, динаміку і наслідки економічних криз.

Серед основних методів моделювання економічних криз важливе місце займають математичні моделі. Вони можуть використовувати рівняння та математичні співвідношення для опису різних аспектів економічної системи. Вони можуть бути лінійними або нелінійними, дискретними або неперервними і дають змогу аналізувати взаємозв'язки між різними змінними та параметрами, які впливають на економічну динаміку. Один із прикладів математичних моделей передбачає використання фінансових індексів, як-от індекс S&P 500, для передбачення фінансових криз. Дослідники створюють моделі, які аналізують попередні коливання індексу та інших факторів, наприклад, ставки нафти чи інфляція. Звернемося до факту: у 2008 р. модель, яка враховувала попередні зміни індексу S&P 500 та показники нерухомості, передбачила наближення фінансової кризи.

До того ж у моделюванні економічних криз застосовуються агентні моделі. У цих моделях економічна система розглядається як сукупність окремих «агентів» (наприклад, підприємств, споживачів), кожен з яких має власні правила та стратегії взаємодії. У сучасному світі агентні моделі стають все популярнішим інструментом для аналізу економічних криз. Наприклад, під час пандемії COVID-19 агентні моделі були використані для передбачення поширення вірусу та його впливу на економіку. Ці моделі враховують індивідуальну поведінку й міжособистісні взаємодії, що дає змогу краще розуміти, як швидкість передачі вірусу впливає на рішення людей та влади.

Однією з ключових функцій моделювання економічних криз є аналіз причин, динаміки та наслідків кризових ситуацій. Моделі допомагають ідентифікувати основні фактори та причини, які призводять до виникнення економічних криз. Це може передбачати фінансові ризики, макроекономічні зміни, політичні рішення, агрегатний попит та інші чинники. У 2007–2008 рр. фінансова криза, відома як «Глобальна фінансова криза», призвела до суттєвих втрат на фінансових ринках та спричинила світову рецесію. Використання математичних моделей допомогло дослідникам виявити, як поширення кризи міжнародно взаємодіяло з надмірною ризикованістю у фінансових інститутах.

Для кращого розуміння ролі системного аналізу й моделювання економічних криз розглянемо конкретні приклади, які ілюструють їх застосування та важливість:

- світова фінансова криза 2008 р. є одним із найбільш відомих прикладів, коли системний аналіз виявився надзвичайно важливим. Вона почалася з фінансових ринків, але швидко поширилася на реальний сектор економіки. Аналітики використовували системні моделі для розуміння взаємозв'язків між банками, страховими компаніями та іншими фінансовими установами. Цей аналіз допоміг усвідомити, як зміни в одному сегменті можуть мати каскадний ефект на весь фінансовий світ. Внаслідок цього були розроблені стратегії для стабілізації фінансової системи, зокрема ін'єкції капіталу в банки та зміни регулювань;
- пандемія COVID-19 стала глобальною кризою зі складними економічними наслідками. Системний аналіз використовується для моделювання поширення вірусу та його впливу на різні сектори економіки. Аналітики враховують взаємодію між галузями, як-от туризм, авіація, ресторани та інші, й розробляють стратегії для підтримки вразливих секторів та відновлення економіки;
- у фінансовому секторі банки проводять стрес-тести, використовуючи системний аналіз, для оцінки своєї стійкості до різних сценаріїв кризи. Наприклад, банки можуть аналізувати вплив спаду цін на нерухомість, зростання безробіття та інші фактори на їх фінансовий стан. Це допомагає банкам готуватися до можливих кризових ситуацій та розробляти запасні плани;

➢ міжнародні організації та уряди використовують системний аналіз для оцінки глобальних ризиків, як-от: зміни клімату, епідемії, геополітичні конфлікти та інші. Наприклад, аналіз системних зв'язків між країнами та регіонами допомагає прогнозувати можливі наслідки глобальних подій і розробляти стратегії для зменшення ризиків;

➢ азіатська фінансова криза 1997 р. є прикладом, як системний аналіз допоміг зрозуміти поширення кризи між країнами та фінансовими ринками. Дослідники використовували системні моделі для аналізу зв'язків між економіками Азійського регіону та іноземними інвесторами, розкриваючи ризики й можливі шляхи подолання кризи;

➢ у 1970-х рр. світ зіштовхнувся з глобальною енергетичною кризою, коли ціни на нафту раптово зросли, а це призвело до значного впливу на світову економіку. Аналітики використовували системні підходи для дослідження взаємозв'язків між постачальниками, споживачами й геополітичними факторами, що впливають на ціни на нафту;

➢ фінансова криза в Японії 1990-х рр. стала прикладом, як системний аналіз використовувався для аналізу внутрішніх факторів, що призвели до довготривалої економічної стагнації. Дослідники аналізували взаємодію між банками, корпораціями та урядом Японії, щоб зрозуміти причини та шляхи виходу з кризи;

➢ велика депресія 1929 р. є історичним прикладом економічної кризи, який відтворюється у вченні та аналізі фінансових ринків. Системний аналіз використовується для розгляду взаємодій між банками, компаніями і державними органами, що вплинули на поширення кризи та її тривалість;

➢ у 2010-х рр. деякі країни Єврозони стали свідками кризи державного боргу. Системний аналіз допоміг аналізувати взаємозв'язки між економіками країн Єврозони та їх фінансовими інституціями. Це дало змогу розробити стратегії для подолання кризи та встановити механізми фінансової стабільності.

Ці приклади ілюструють, як системний аналіз може бути використаний для розуміння та управління економічними кризами в різних історичних і сучасних контекстах. Аналіз взаємодій між різними елементами економіки допомагає приймати обґрунтовані рішення та розробляти стратегії для мінімізації наслідків кризових ситуацій.

Отже, стає очевидним, що системний аналіз та моделювання є необхідними інструментами для вивчення й управління економічними кризами. Розуміння системних аспектів економіки сприяє більш ефективному передбаченню та реагуванню на кризові ситуації, допомагаючи цим зменшити їх негативний вплив на суспільство та підприємства.

Список використаних джерел

1. Гладка О. М., Карпович І. М., Сінчук А. М. Моделі економічної динаміки: посібник. Рівне: Університетська книга, 2019. 158 с. URL: https://core.ac.uk/download/pdf/297135502.pdf (дата звернення: 27.11.2018).

2. Клебанова Т. С., Дубровина Н. А., Полякова О. Ю. Моделювання економічної динаміки: посібник. Харків: Університетська книга, 2005. 244 с.

3. Січко Т. В., Рибак І. І. Системний підхід до аналізу організаційних структур. *Вісник Хмельницького національного університету. Технічні науки*. 2020, № 4. С. 70–74.

УДК 004.021

Левченко М. Р., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,

Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

РОЗГЛЯД ЗАСТОСУВАННЯ АЛГОРИТМІВ НА ГРАФАХ ДЛЯ ВИРІШЕННЯ ПРИКЛАДНИХ ЗАДАЧ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному інформаційному суспільстві, де величезний обсяг даних неперервно зростає, а взаємодії та зв'язки між об'єктами стають складнішими, застосування алгоритмів на графах є надзвичайно актуальним. Графи є відмінним математичним інструментом для моделювання та аналізу різноманітних систем, де об'єкти взаємодіють і мають складні структури.

З погляду комп'ютерних наук і дискретної математики графи є абстрактним методом представлення типів відносин, як-от дороги, що з'єднують міста, та інших видів мереж. Графи складаються з ребер та вершин. Вершина – це точка на графі, а ребро – це те, що з'єднує дві точки на графі [1].

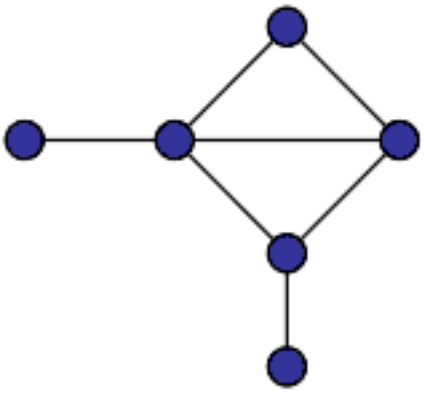


Рисунок 1. Загальний вигляд графу

Неорієнтований граф. У неорієнтованому графі ребра не мають напрямку. Це означає, що вони представляють просто зв'язок між двома вершинами, і цей зв'язок є взаємним. Якщо вершини A і B з'єднані ребром у неорієнтованому графі, то можна сказати, що A пов'язано з B, і навпаки. Формально ребро (A, B) еквівалентне ребру (B, A).

Орієнтований граф. У орієнтованому графі ребра мають напрямок. Кожне ребро вказує на одну з вершин як на вихідну (початкову) і на іншу як на вхідну (кінцеву). Такий граф може ілюструвати однобічні або однонапрямлені взаємозв'язки між об'єктами. Наприклад, якщо ребро (A, B) існує в орієнтованому графі, це не означає автоматично наявність ребра (B, A).

Пошук в ширину (BFS) – це алгоритм обходу або пошуку в графі, який починає з визначеної вершини (часто називається «початковою вершиною») і поступово відвідує всі сусідні вершини на поточному рівні перед переходом на наступний рівень.

У незваженому графі, де кожне ребро має однакову вагу (або вага не враховується взагалі), BFS знаходить найкоротший шлях між двома вершинами, тобто шлях, який має мінімальну кількість ребер. Алгоритм гарантує знаходження найкоротшого шляху, оскільки він відвідує вершини в порядку збільшення відстані від початкової вершини [3].

Алгоритм працює за $O(n + m)$, де n – кількість вершин, m – кількість ребер.

Основна ідея полягає в тому, що вершини додаються в чергу для обробки в порядку їх відстані від початкової вершини, і обробка продовжується доти, поки черга не стане порожньою.

Алгоритм пошуку в ширину (англ. breadth-first research, BFS) використовують також для:

1) вебсерфінгу;
2) пошуку друзів у соціальних мережах;
3) доступу до мережі Internet;
4) обрахунку кон'юнкції;
5) перевірки моделей у теорії автоматів та ін.

Загалом алгоритм пошуку в ширину розглядає простий зв'язний граф G = <V, E>. Задачею алгоритму є побудова маршруту від початкової і до всіх вершин графу, обхід графу [2].

Задача. Є лабіринт у вигляді сітки, де кожна клітина може бути або вільною, або перешкодою. Кожна вільна клітина – це вершина графу, а сусідні вільні клітини – це ребра, які їх з'єднують. Потрібно знайти найкоротший шлях від початкової до кінцевої точки лабіринту, обходячи перешкоди.

У цьому коді використовується BFS (пошук в ширину) для знаходження найкоротшого шляху у лабіринті. Лабіринт представлений у вигляді двовимірного списку (maze), де 0 відповідає вільній клітині, а 1 – перешкоді.

```
maze = [
    [0, 1, 0, 0, 0],
    [0, 1, 0, 1, 0],
    [0, 0, 0, 1, 0],
    [1, 1, 1, 1, 0],
    [0, 0, 0, 0, 0]
]
```

Рисунок 2. Двовимірний масив

Спочатку створюємо двовимірний список visited, який використовується для відстеження відвіданих клітин у лабіринті. Кожен елемент цього списку визначається значенням False, яке вказує на те, що відповідна клітина ще не була відвідана. Після цього визначаємо напрямки руху в лабіринті, представлені координатами (dx, dy) для руху праворуч, вниз, ліворуч та вгору. Ці напрямки використовуються для визначення сусідніх клітин під час обходу лабіринту.

```
def shortest_path_in_maze(maze, start, end):
    rows, cols = len(maze), len(maze[0])
    visited = [[False] * cols for _ in range(rows)]
    directions = [(0, 1), (1, 0), (0, -1), (-1, 0)]
```

Рисунок 3. Двовимірний масив та напрямок руху в лабіринті

Далі створюємо чергу queue з початковою клітиною та відстанню 0 і позначаємо початкову клітину як відвідану.

```
queue = deque([(start[0], start[1], 0)])
visited[start[0]][start[1]] = True
```

Рисунок 4. Черга з початковою клітиною

Поки черга не порожня: виймаємо клітину з черги, перевіряємо, чи досягли кінцевої точки, додаємо сусідні клітини до черги і позначаємо їх як відвідані.

```
while queue:
    x, y, distance = queue.popleft()

    if (x, y) == end:
        return distance

    for dx, dy in directions:
        new_x, new_y = x + dx, y + dy

        if is_valid_move(new_x, new_y):
            queue.append((new_x, new_y, distance + 1))
            visited[new_x][new_y] = True
```

Рисунок 5. Реалізація пошуку в ширину

Повертаємо довжину найкоротшого шляху або – 1, якщо досягти кінцевої точки неможливо.

```
result = shortest_path_in_maze(maze, start_point, end_point)
print("Найкоротший шлях:", result)
```

Рисунок 6. Виведення результату

```
Найкоротший шлях: 12

Process finished with exit code 0
```

Рисунок 7. Результат програми

Підсумовуючи, варто зазначити, що використання алгоритмів на графах для вирішення прикладних завдань виявляється дуже ефективним і практичним напрямом. Розглянуті алгоритми, як-от пошук найкоротшого шляху в лабіринті з допомогою BFS, можуть бути застосовані в різноманітних областях, зокрема в оптимізації маршрутів кур'єрських служб у місті. Застосування теорії графів та алгоритмів на графах дає змогу розв'язувати складні завдання шляхом моделювання взаємозв'язків між об'єктами. Цей підхід виявляється важливим і у сучасному світі, де оптимізація та аналіз мережевих структур стають ключовими вимогами для ефективного вирішення прикладних задач.

Список використаних джерел

1. Medium. Графи: основи, алгоритми, теорії. URL: https://medium.com/nuances-of-programming/графы-основы-теории-алгоритмы-поиска-b93672f597472

2. Теорія графів. URL: https://ela.kpi.ua/bitstream/123456789/35854/1/Teoriia_hrafiv.pdf

3. Алгоритми оптимізації на графах. URL: https://ami.lnu.edu.ua/wp-content/uploads/2014/02/DO_CH2_Alhorytmy-optymizatsii-na-hrafakh.pdf

УДК 004.77

Малінін А. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Волонтир Л. О., канд. техн. наук, доцент доцент кафедри інформаційних технологій

ВИСОКОТЕХНОЛОГІЧНІ ІНСТРУМЕНТИ У ВЕБДИЗАЙНІ ТА ВЕБПРОГРАМУВАННІ ЯК ДВИГУНИ ІННОВАЦІЙ У ЦИФРОВОМУ ПРОСТОРІ

Донецький національний університет імені Василя Стуса, м. Вінниця

У стрімкому розвитку цифрового світу високотехнологічні інструменти вебдизайну та програмування відіграють важливу роль у формуванні та прискоренні інноваційних аспектів інтернету. Метою цього дослідження є детальний аналіз важливих аспектів впливу та ролі інструментів у сучасній веброзробці, а також того, як ці інструменти діють як творчі каталізатори та інформують про технологічний прогрес й інноваційні підходи [1]. Основні принципи цієї ініціативи визначають високотехнологічні інструменти не лише як інструменти для вирішення завдань, а й як творчі інструменти, які сприяють і стимулюють інноваційний розвиток. Важливо підкреслити, що ці інструменти не тільки автоматизують рутинні завдання, але й слугують джерелом нових ідей, розширюють горизонти і збагачують технічні та творчі сфери веброзробки. Одним із важливих моментів цього дослідження є визначення впливу високотехнологічних інструментів на розвиток вебпростору. Вони не тільки відповідають сучасним вимогам, а й визначають тенденції розвитку. Високотехнологічні рішення у сфері вебдизайну дають змогу створювати інтерфейси, які є не тільки естетично привабливими, але й функціонально багатими, що відображають сучасні тенденції та враховують очікування користувачів. Ще один важливий аспект, який необхідно розглянути більш детально, – це вплив високотехнологічних інструментів на безпеку та стандартизацію веброзробки. Їх внесок у забезпечення кібербезпеки та створення загальних систем стандартів відіграє важливу роль у формуванні довіри користувачів до цифрових платформ [2].

Також розглядається вплив високотехнологічних інструментів на розвиток електронної комерції та динаміку взаємодії бізнесу з клієнтами. Сучасні веб-

рішення дають змогу не тільки створювати зручні інтернет-магазини, а й реалізовувати інноваційні підходи до персоналізації пропозицій для кожного користувача [3]. Аналіз купівельної поведінки та використання алгоритмів штучного інтелекту та машинного навчання дають змогу нам створювати унікальні пропозиції, які максимально відповідають індивідуальним потребам кожного клієнта. Також необхідно враховувати важливість віртуальної реальності (VR) та інтерактивних технологій у веброзробці. Використання VR у вебдизайні дають змогу користувачам взаємодіяти зі вмістом у просторі, відкриваючи нові можливості для творчих вебпроєктів, як-от віртуальні музеї, екскурсії та вебсеріали. Це створює потужний інструмент для створення унікальних і привабливих вебсередовищ, які забезпечують унікальний досвід для кожного відвідувача. До того ж важливо враховувати вплив високотехнологічних інструментів на сферу вебаналітики [4].

Використовуючи інструменти штучного інтелекту для аналізу великих обсягів даних, можна не лише збирати інформацію про відвідувачів, але й ефективно інтерпретувати цю інформацію. Це дає можливість точніше прогнозувати тенденції, реагувати на зміни в реальному часі та покращувати свою маркетингову стратегію. Отже, високотехнологічні інструменти вебдизайну та програмування виявилися справжніми рушійними силами інновацій, формуючи не лише сучасну веброзробку, але й її майбутній напрям. Його роль як інструменту технологічного прогресу неможливо переоцінити, і він продовжує визначати нові горизонти можливостей у цифровому просторі [5].

Список використаних джерел

1. Інструменти веб-дизайну. URL: https://lemon.school/blog/instrumenty-veb-dyzajnu (дата звернення: 03.12.2023) (дата звернення: 03.12.2023).

2. Вебдизайн. Інструменти і технології. URL: https://naurok.com.ua/urok-6-veb-dizayn-instrumenti-i-tehnologi-129685.html (дата звернення: 03.12.2023).

3. Незамінні інструменти веб-дизайнера. URL: https://itea.ua/news_itea/nezaminni-instrumenti-veb-dizaynera/ (дата звернення: 03.12.2023).

4. Інноваційні технології навчання: навч. посіб. для студ. вищих технічних навчальних закладів / кол. авторів; відп. ред. Х. Ш. Бахтіярова; наук. ред. А. В. Арістова; упорядн. словника С. В. Волобуєва. Київ: НТУ, 2017. 172 с.

5. Вебдизайн. URL: https://ru.wikipedia.org/wiki/Веб-дизайн (дата звернення: 03.12.2023).

УДК 004.6

Матіяш М. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

СТРУКТУРИ ДАНИХ ТА ЇХ КЛАСИФІКАЦІЯ

Донецький національний університет імені Василя Стуса, м. Вінниця

У світі програмування та обробки інформації структури даних виступають основним складником, що визначає ефективність та організацію даних. Розуміння основних типів структур даних та їх класифікації є важливим кроком для будь-якого, хто бажає глибше розібратися в програмуванні та аналізі даних.

Структури даних – це не просто концепція організації інформації, це надзвичайно потужний інструмент, що визначає шлях, яким дані будуть зберігатися, оброблятися й використовуватися в програмах та системах. Вони забезпечують можливість ефективного доступу до даних та виконання різноманітних операцій з ними. Дослідження класифікації структур даних дасть змогу краще розібратися в їхньому розмаїтті та ролі в програмуванні. Класифікація структур даних виконується за декількома ознаками:

1. За способом представлення: фізична та логічна. Поняття «фізична структура даних» має відношення до способу фізичного представлення даних у пам'яті машини і називається ще структурою збереження, внутрішньою структурою або структурою пам'яті. Логічна чи абстрактна структура – це розгляд структури даних без врахування її представлення в машинній пам'яті.

2. За складністю: прості й інтегровані. Прості (базові, примітивні) структури – це такі, які не можуть бути розподілені на складники. Інтегровані (структуровані, композитні, складні) – такі структури даних, складниками яких є інші структури даних – прості чи так само інтегровані.

3. За наявністю зв'язків між елементами даних: незв'язні та зв'язні. Незв'язні структури характеризуються відсутністю зв'язків між елементами структури, зв'язні – наявністю такого зв'язку. Прикладами незв'язних структур є вектори, масиви, рядки, стеки, черги; приклади зв'язних структур – зв'язні списки.

4. За мінливістю: статичні, напівстатичні, динамічні. Статичні – до цієї групи відносять масиви, множини, записи, таблиці. Напівстатичні – це стеки, черги, деки, дерева. Динамічні – лінійні та розгалужені зв'язні списки, графи, дерева.

5. За характером упорядкованості елементів у структурі: лінійні та нелінійні. Лінійні структури залежно від характеру взаємного розташування елементів у пам'яті поділяють на структури з послідовним розподілом елементів у пам'яті (вектори, рядки, масиви, стеки, черги) і структури з довільним зв'язним роз-

поділом елементів у пам’яті (однозв’язні і двозв’язні лінійні списки). Нелінійні структури – багатозв’язні списки, дерева, графи.

6. За видом пам’яті, використовуваної для збереження даних: структури даних для оперативної і зовнішньої пам’яті. Структури даних для оперативної пам’яті – це дані, розміщені в статичній і динамічній пам’яті комп’ютера. Всі вищенаведені структури даних – це структури для оперативної пам’яті. Структури даних для зовнішньої пам’яті називають файловими структурами чи файлами. Прикладами файлових структур є послідовні файли; файли; організовані розділами; B-дерева.

Отже, дослідження структур даних та їх класифікація відкрили широкий спектр інструментів для організації та управління даними в програмуванні.

Список використаних джерел

1. Креневич А. П. Алгоритми і структури даних: підручник. Київ: ВПЦ Київський Університет, 2021. 200 с.

2. Лінійний однозв’язний список. URL: https://erudyt.net/navchalni-predmety/informatika/prohramuvannya/linijnyj-odnozvyaznyj-spysok.html (дата звернення: 03.02.2024 р.)

3. Двозв’язні списки. URL: https://krypton.com.ua/rozdil-2-struktury-danyh/dvozvyazkovi-spysky/ (дата зверненя: 03.02.2024 р.)

4. Korman T. H. Algorithms. Unlocked. The MIT Press, Massachusetts Institute of Technology, 2013. 207 p.

УДК 519.86

Мельник В. Р., здобувач 3 курсу спеціальності 122 Комп’ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВИКОРИСТАННЯ МЕТОДІВ ОПТИМІЗАЦІЇ ДЛЯ ОБРОБКИ ТА АНАЛІЗУ ВЕЛИКИХ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Великі дані (Big Data) є одним із найважливіших трендів сучасного світу. Великі обсяги даних генеруються в різних сферах діяльності, як-от бізнес, наука, уряд та суспільство. Обробка й аналіз великих даних є складною задачею, оскільки вимагає значних обчислювальних ресурсів [1].

Методи оптимізації є потужним інструментом для обробки й аналізу великих даних. Вони дають змогу вирішувати різні задачі: розподіл даних на збері-

гання, вибір алгоритму машинного навчання, прогнозування та оптимізація бізнес-процесів.

Актуальність використання методів оптимізації для обробки та аналізу великих даних обумовлена такими факторами:

- зростання обсягів даних. Обсяг даних, які генеруються, постійно зростає. Це викликано розвитком технологій: інтернету, штучного інтелекту та аналітики даних;
- усе більша складність даних. Дані, які генеруються в сучасному світі, є різноманітними та складними. Вони можуть містити текстові дані, дані з датчиків, дані з соціальних мереж та ін.;
- затребуваність аналітики великих даних. Організації все більше потребують аналітики великих даних для прийняття обґрунтованих рішень, покращення ефективності бізнес-процесів та отримання нових insights.

Методи оптимізації допомагають організаціям отримувати максимальну користь від великих даних. Вони дають змогу:

- мінімізувати витрати. Методи оптимізації можуть бути використані для мінімізації витрат на зберігання, обробку та аналіз даних;
- підвищити ефективність. Методи оптимізації можуть бути використані для підвищення ефективності бізнес-процесів, як-от логістика, виробництво та маркетинг;
- приймати більш обґрунтовані рішення. Методи оптимізації можуть бути використані для прийняття більш обґрунтованих рішень на основі даних;
- отримувати нові insights. Методи оптимізації можуть бути використані для отримання нових insights, які можуть бути використані для покращення бізнесу.

Приклади використання методів оптимізації для обробки та аналізу великих даних:

- компанія Amazon використовує методи оптимізації для розподілу даних на зберігання у своїх дата-центрах. Методи оптимізації дають змогу Amazon мінімізувати витрати на зберігання та час доступу до даних;
- компанія Netflix використовує методи оптимізації для вибору алгоритму машинного навчання для прогнозування попиту на фільми та телешоу. Методи оптимізації дають змогу Netflix забезпечувати найкраще можливе відеоспоживання для своїх користувачів [2].

Враховуючи щораз більшу актуальність великих даних, можна прогнозувати, що використання методів оптимізації для обробки та аналізу великих даних буде продовжувати зростати в майбутньому.

Список використаних джерел

1. Big data. URL: https://www.it.ua/knowledge-base/technology-innovation/big-data-bolshie-dannye

2. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів ОС «Бакалавр» денної та заочної форм навчання спеціальностей 122 Комп'ютерні науки, 113 Прикладна математика. Вінниця: ДонНУ імені Василя Стуса. 2020, 104 с.

УДК 004.021

Менделюк К. В., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ОЦІНКА СКЛАДНОСТІ АЛГОРИТМІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Алгоритм – це послідовність точно визначених інструкцій, призначених для виконання конкретного завдання. Оцінка алгоритмів містить вивчення двох ключових аспектів: ефективності та складності. Оцінка ефективності та складності алгоритмів стає важливим етапом у процесі розробки, спрямованим на забезпечення оптимального використання ресурсів та досягнення високої продуктивності.

Ефективність алгоритмів визначається часом виконання та обсягом використаних ресурсів. Часова ефективність оцінює швидкість роботи алгоритму, а просторова ефективність – його використання пам'яті. Аналізуючи ці параметри, розробники можуть підбирати оптимальні алгоритми для конкретних завдань.

Складність алгоритму визначається кількістю операцій, які він виконує, та взаємозв'язком між ними. Часова та просторова складність взаємопов'язані і вимагають балансу для досягнення оптимальності. Оцінка складності алгоритмів допомагає уникнути виникнення зайвих витрат ресурсів та забезпечити оптимальну продуктивність. Основними чинниками, що впливають на складність і ефективність алгоритмів, є:

1. Оцінка часової та просторової складності. Одним з основних складників оцінки алгоритмів є їх часова та просторова складність. Часова складність визначає, як швидко алгоритм вирішує завдання під час збільшення розміру вхідних даних, тоді як просторова складність вказує на обсяг пам'яті, який

алгоритм використовує. Ефективний алгоритм повинен забезпечувати оптимальні значення обох цих параметрів.

2. Вибір оптимальних структур даних. Для досягнення високої ефективності важливо вибирати оптимальні структури даних. Наприклад, використання хеш-таблиць чи бінарних дерев може значно поліпшити часову та просторову ефективність. Однак вибір оптимальних структур даних пов'язаний із завданням та його особливостями.

3. Покращення алгоритмів. Оцінка ефективності алгоритмів найчастіше включає їх покращення. Це може бути досягнуто з допомогою оптимізації коду, використання паралельних обчислень чи впровадження нових підходів. Процес пошуку оптимального рішення може займати певний час, проте він є критичним для досягнення максимальної продуктивності.

4. Оцінка ефективності алгоритмів має велике значення у сучасних технологіях, як-от штучний інтелект, обробка великих даних та Інтернет речей. У цих галузях найважливішою є не тільки швидкість роботи алгоритмів, але й їх можливості адаптації до змінних умов та обсягів даних.

5. Через стрімкий розвиток технологій та збільшення обсягів даних, виникають нові виклики в оцінці ефективності алгоритмів. Алгоритми мають бути адаптовані до обробки великих потоків даних у режимі реального часу, що ставить перед розробниками завдання пошуку оптимальних рішень для високопродуктивних систем.

6. Забезпечення стійкості та безпеки алгоритмів. Оцінка ефективності алгоритмів також пов'язана із їх стійкістю та безпекою. У сучасному світі, де зростає кількість кіберзагроз, важливо враховувати не лише продуктивність, але й стійкість алгоритмів до атак, а також забезпечити конфіденційність оброблюваних даних.

7. Машинне навчання та оцінка якості моделей. У контексті машинного навчання, оцінка ефективності стосується не лише алгоритмів, але і моделей. Налаштування параметрів, вибір оптимальних функцій витрат, оцінка точності та генералізації стають ключовими аспектами, які визначають успіх алгоритмів машинного навчання.

8. Перспективи розвитку оцінки ефективності. З огляду на швидкі зміни в інформаційному суспільстві, перспективи розвитку оцінки ефективності алгоритмів передбачають автоматизацію процесів вибору та оптимізації, використання технік метааналізу й розширення використання штучного інтелекту для самоналаштування алгоритмів у реальному часі.

Оцінка ефективності та складності алгоритмів є необхідним етапом у розробці програмного забезпечення, який визначає його успішність і конкурентоспроможність. У сучасному світі, де швидкість та точність вирішення завдань

стають критичними, розробники мають постійно вдосконалювати свої підходи до оцінки та оптимізації алгоритмів для відповіді на виклики інформаційного століття.

Список використаних джерел

1. Cormen T., Leiserson C., Rivest R., Stein C. Introduction to Algorithms. The MIT Press, 2009. 1312 p.

2. Седжвік Р., Уейн К. Алгорити на Java. Addison-Wesley, 2012. 956 с.

3. Гудрич М., Тамассіа Р. Алгоритми та структури даних. Wiley, 2013. 672 с.

УДК 519.86

Морозюк А. А., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВПЛИВ ЕКОНОМІЧНИХ ФАКТОРІВ НА ПОСТАНОВКУ ТА РОЗВ'ЯЗАННЯ ОПТИМІЗАЦІЙНИХ ЗАДАЧ

Донецький національний університет імені Василя Стуса, м. Вінниця

Оптимізаційні задачі стали необхідним складником ефективного управління та прийняття рішень у різних галузях економіки. В сучасному світі, коли ресурси обмежені, а завдання стають усе більш складними, математичні задачі оптимізації дають змогу знаходити оптимальні рішення для різноманітних завдань та проблем. Роль таких задач в економіці важко переоцінити. Вони є чудовим інструментом для планування виробничих процесів, оптимізації логістичних мереж, управління фінансами, ресурсами та багатьма іншими аспектами бізнесу й господарювання. Задачі оптимізації допомагають компаніям та організаціям максимізувати прибуток, мінімізувати витрати, раціонально використовувати ресурси та ефективно вирішувати стратегічні завдання [1].

Вивчення впливу економічних змін на параметри та умови оптимізаційних задач є ключовим етапом аналізу в галузі методів оптимізації. Цей аспект дає змогу розглядати оптимізаційні задачі як динамічні системи, що реагують на зміни в економічному середовищі. Аналізується вплив таких факторів: інфляція, політичні та правові чинники, демографічні тенденції, ринкова конкуренція тощо, на параметри цільової функції й обмеження оптимізаційних моделей. Це сприяє збереженню актуальності та високої ефективності оптимізаційних рішень в умовах змінного економічного середовища. Вивчення впливу економічних модифікацій передбачає аналіз чутливості оптимальних рішень до зміни

параметрів, як-от цінові коливання, податкова політика чи трансформації у витратах на ресурси. Цей підхід також передбачає врахування економічних ризиків. Аналізуються можливі варіації економічних умов та розробляються стратегії оптимізації, які беруть до уваги ймовірність виникнення різних економічних сценаріїв. Такий спосіб вивчення впливу змін робить оптимізаційні моделі більш гнучкими та адаптивними до реальних умов бізнесу й економіки, підвищуючи їх реалістичність та ефективність у практиці.

Класифікація оптимізаційних задач в економіці базується на різноманітних аспектах і враховує специфіку економічних процесів. Основні напрями класифікації [2]:

1. Оптимізація виробництва: максимізація прибутку за обмежених ресурсів, мінімізація витрат на виробництво певної кількості товарів чи послуг.

2. Оптимізація розподілу запасів: максимізація використання обмежених матеріалів, забезпечення ефективного розподілу робочої сили, капіталу та інших ресурсів.

3. Оптимізація фінансових рішень: максимізація доходів та мінімізація ризиків в інвестиційних та фінансових операціях, оптимізація портфеля інвестицій для досягнення заданих фінансових цілей.

4. Оптимізація ланцюга постачання: мінімізація витрат та часу доставки, максимізація ефективності.

5. Оптимізація цінової політики: встановлення оптимальних цін на товари та послуги для максимізації прибутку, адаптація цінової політики до змін у внутрішньому й зовнішньому середовищі.

Після класифікації оптимізаційних задач важливим аспектом стає їх взаємодія з економічними моделями. Ці категорії не лише визначають основні напрями оптимального рішення, але й взаємодіють між собою, створюючи комплексні моделі для ефективного розв’язання реальних завдань та оптимізації управлінських рішень. Основні напрями взаємодії можна розглядати на прикладі вказаних категорій [3]:

1. Оптимізація виробництва:

1) економічний аспект: максимізація прибутку та мінімізація витрат виробництва;

2) оптимізаційна модель: розробка стратегій для оптимального використання ресурсів та максимізації виробничої ефективності.

2. Оптимізація розподілу запасів:

1) економічний аспект: максимізація використання обмежених матеріалів та забезпечення ефективного розподілу ресурсів;

2) оптимізаційна модель: розробка оптимальних сценаріїв розподілу робочої сили, капіталу та інших ресурсів для досягнення ефективності.

3. Оптимізація фінансових рішень:

1) економічний аспект: максимізація доходів та мінімізація ризиків в інвестиційних і фінансових операціях;

2) оптимізаційна модель: розробка стратегій оптимального управління ресурсами та інвестиціями для досягнення фінансових цілей.

4. Оптимізація ланцюга постачання:

1) економічний аспект: мінімізація витрат та оптимізація управління ланцюгом постачання;

2) оптимізаційна модель: розробка оптимальних стратегій для забезпечення ефективності та зниження часу доставки.

5. Оптимізація цінової політики:

1) економічний аспект: встановлення оптимальних цін для максимізації прибутку;

2) оптимізаційна модель: розробка алгоритмів для адаптації цінової політики до змін у внутрішньому та зовнішньому середовищі.

Ці категорії взаємодіють між собою, формуючи комплексні моделі, які сприяють вирішенню реальних економічних завдань та оптимізації управлінських рішень.

Отже, у сфері сучасної економіки врахування економічних факторів у формулюванні та розв'язанні оптимізаційних задач є ключовим елементом для досягнення успіху і стійкості в управлінні різноманітними процесами. Здатність враховувати подібні аспекти у математичних моделях оптимізації відкриває широкі можливості для ефективного управління ресурсами, оптимізації витрат та досягнення стратегічних цілей. Здатність враховувати економічні фактори у процесі постановки та розв'язання оптимізаційних задач стає ключовою у ситуаціях невизначеності й динамічного бізнес-середовища.

На практиці це дослідження підкреслює необхідність для підприємств і організацій ретельно аналізувати економічні фактори під час розробки та вирішення оптимізаційних завдань, що дасть змогу їм ефективно пристосовуватися до змін в економічному середовищі й досягати оптимальних результатів в умовах постійної нестабільності та конкуренції.

Список використаних джерел

1. Методи оптимізації та моделі в економіці: вебсайт. URL: https://cutt.ly/OwYEmZLh (дата звернення: 10.11.2023).

2. Вітлінський В. В., Терещенко Т. О., Савіна С. С. Економіко-математичні методи та моделі: оптимізація: навч. посіб. Київ: КНЕУ, 2016. 303 с.

3. Казарєзов А. Я., Ципліцька О. О. Економіко-математичне моделювання: навч. посіб. Миколаїв: Вид-во ЧДУ ім. Петра Могили, 2009. 248 с.

4. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів СО «Бакалавр» денної та заочної форм навчання спеціальностей 122 «Комп'ютерні науки», 113 «Прикладна математика». Вінниця: ДонНУ імені Василя Стуса. 2020, 104 с.

УДК 004.021

Поліщук В. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

АЛГОРИТМ ПОШУКУ У ГЛИБИНУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Алгоритм пошуку в глибину є одним з найважливіших обчислень для графів і дерев. Він використовує методологію «глибокого» погляду, щоб почати з дослідження якомога більшої відстані в глибину графу, а потім повернутися назад. Найчастіше для актуалізації обчислень використовується рекурсивний підхід або стек.

Пошук у глибину (DFS) – це алгоритм для обходу або пошуку структур даних дерева або графу. Алгоритм починається з кореневого вузла і досліджує, наскільки це можливо, уздовж кожної гілки перед зворотним відстеженням. Додаткова пам'ять (зазвичай стек) потрібна для відстеження вузлів, знайдених на цей момент уздовж певної гілки. Це допоможе вам відстежити графік. Варіант пошуку в глибину досліджувався як стратегія вирішення лабіринту французьким математиком Шарлем-П'єром Тремо в XIX ст.

Часовий і просторовий аналіз DFS різниться залежно від області його застосування. У цьому налаштуванні часові та просторові межі такі ж, як і для пошуку в ширину, і вибір того, який із цих двох алгоритмів використовувати, залежить більше від різниці в порядку згенерованих вершин, ніж від складності. Це залежить від характеристик. Для програм DFS, пов'язаних із певним доменом, як-от пошук рішень із допомогою штучного інтелекту або вебсканування, граф, який потрібно пройти, часто занадто великий, щоб бути повністю доступним, або нескінченний (з можливістю того, що DFS ніколи не завершиться). У таких випадках пошук здійснюватиметься лише на обмеженій глибині. Через обмежені ресурси, як-от пам'ять і дисковий простір, структури даних зазвичай не використовуються для відстеження набору всіх раніше відвіданих вершин. Під час виконання пошуку на обмеженій глибині час залишається лінійним із

розширеною кількістю вершин і ребер хоча деякі вершини можуть шукатися кілька разів, тому це число може змінюватися на графі. Проте складність простору цього варіанта DFS пропорційна лише межі глибини, тому вона набагато менша, ніж простір, необхідний для пошуку на таку саму глибину з допомогою пошуку в ширину. У таких програмах DFS також підходить евристичний метод для вибору ймовірних гілок. Якщо відповідні обмеження глибини невідомі заздалегідь, ітеративний пошук у глибину неодноразово застосовує DFS зі зростаючими наборами обмежень. У режимі аналізу штучного інтелекту, коли коефіцієнт розгалуження більший за 1, кількість вузлів на рівень геометрично збільшується, тому ітераційне поглиблення зменшує час виконання до постійного значення, порівняно зі знанням точного обмеження глибини. Ви також можете використовувати DFS для збору зразків вузлів графу. Однак неповна DFS, як і неповна BFS, спрямована на вузли вищого порядку.

Пошук у глибину також можна використовувати для лінійного впорядкування вершин графу або дерева. Це можна зробити чотирма можливими способами. Попереднє впорядкування – це список вершин у тому порядку, в якому до них вперше звернувся алгоритм пошуку в глибину. Це компактний і природний спосіб описати процес пошуку, як це було зроблено в першій частині цієї статті. Попереднє впорядкування дерева виразів – це вираз у польській нотації. Заднє впорядкування – це список вершин у порядку, в якому алгоритм востаннє відвідував вершини. Задній порядок дерева виразів – це вираз у зворотному польському записі. Зворотне попереднє впорядкування – це зворотне попередньому впорядкуванню. Це означає, що список вершин знаходиться у порядку, зворотному порядку першого відвідування. Зворотне попереднє впорядкування – це не те ж саме, що і пост-впорядкування. Зворотний пост-впорядкування – це зворотне пост-впорядкування, тобто список вершин у зворотному порядку до попереднього відвідування. Зворотне пост-впорядкування не тотожне попередньому впорядкуванню. Для бінарних дерев також існує впорядкування і зворотне впорядкування.

Для реалізації алгоритму знадобиться відзначати, в яких вершинах був дослідник, а в яких – ні. Позначку робитимемо в списку Visited, де Visited [i] = True для відвіданих вершин, і Visited [i] = False для невідвіданих. Позначка «про відвідування вершини» ставиться під час входу в цю вершину. Алгоритм обходу в глибину оформимо у вигляді рекурсивної процедури DFS, де start – номер вершини, з якої запускається обхід.

У цьому алгоритмі *n* – число вершин у графі, вершини нумеруються числами від 1 до *n*, зберігає матрицю суміжності. Для запуску алгоритму, наприклад, для вершини з номером start необхідно викликати DFS. Після цього виклику всі вершини, доступні із start, будуть позначені в списку Visited.

```
procedure DFS(start:integer);
        var i: integer;
begin
                Visited[start]:=true;
                for i:=1 to n do
                        if (a[start, i]=1) and (Visited[i]=false)
                                then DFS(i)
        end;
```

Рисунок 1. Код алгоритму пошуку в глибину

Візуалізація алгоритму пошуку в глибину наведена на рис. 2.

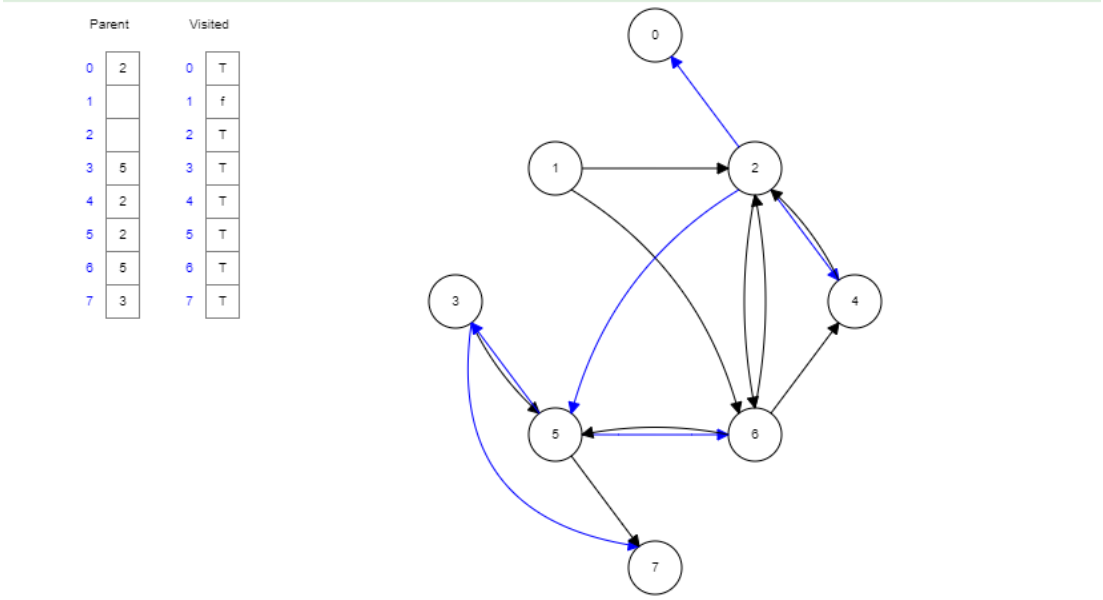


Рисунок 2. Реалізація алгоритму пошуку в глибину

Глибинний пошук у графі дає змогу систематично досліджувати структуру графу або дерева, проникаючи вглиб та перевіряючи всі можливі гілки. Це допомагає виявляти компоненти зв'язності, цикли, шляхи та інші характеристики графів. Важливо пам'ятати, що глибинний пошук не гарантує знаходження найкоротших шляхів у невагомих графах, і у деяких випадках може призвести до перевірки одних і тих самих вузлів кілька разів. Однак він є ефективним інструментом для визначення властивостей графів та дерев.

Список використаних джерел

1. Algoua. URL: https://algoua.com/algorithms/graphs/dfs/ (дата звернення: 28.11.2023).

2. Tilda. URL: https://grafi.tilda.ws/vglibiny (дата звернення: 28.11.2023).

3. Mathros. URL: https://www.mathros.net.ua/obhid-grafa-v-glybynu.html (дата звернення: 28.11.2023).

4. Biz. URL: http://www.ni.biz.ua/5/5_15/5_159663 (дата звернення: 28.11.2023).

УДК 004.021

Поліщук О. С., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

СУТНІСТЬ І СКЛАДНИКИ РЕКУРЕНТНИХ АЛГОРИТМІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Рекурсія є фундаментальним поняттям у комп'ютерних науках, що використовується для вирішення різноманітних завдань. Вона базується на ідеї функції, що викликає саму себе, і відіграє важливу роль у розробці алгоритмів та програм.

Рекурсія – процес повторення чого-небудь самоподібним способом. Наприклад, вкладені віддзеркалення, утворені двома точно паралельними одне одному дзеркалами, є однією з форм нескінченної рекурсії. Цей термін має більш спеціальні значення в різних галузях знань – від лінгвістики до логіки. Кількість вкладених викликів функції або процедури називається глибиною рекурсії.

Перевага рекурсивного визначення об'єкта полягає в тому, що таке вичерпне визначення теоретично здатне описувати нескінченно велику кількість об'єктів. Із допомогою рекурсивної програми можливо описати нескінченне обчислення, причому без явних повторень частин програми [1].

Рекурсивну програму завжди можна перетворити в нерекурсивну (ітеративну, з використанням циклів), яка виконує ті ж обчислення. І навпаки, використовуючи рекурсію, будь-яке обчислення, що припускає використання циклів, можна реалізувати, не застосовуючи цикли. Рекурентне співвідношення є рекурсивною функцією з цілочисельними значеннями. Значення будь-якої такої функції можна визначити, обчислюючи усі її значення, починаючи з найменшого, використовуючи на кожному кроці раніше обраховані значення для підрахунку поточного значення. Рекурентні вирази використовуються, зокрема, для визначення складності рекурсивних обчислень [2].

Властивості рекурентних алгоритмів передбачають [2]:

1. Самовиклик (Self-calling). Рекурсія полягає у виклику функції або алгоритму самим собою, що покладено в основну ідею рекурсивного підходу.

2. Базовий випадок (Base Case). Кожен рекурсивний алгоритм повинен мати базовий випадок, коли виклик функції просто повертає результат без нового рекурсивного виклику. Базовий випадок є умовою завершення рекурсії.

3. Повторюваність (Repetition). Рекурсивний алгоритм повинен робити прогрес у кожному новому виклику, зменшуючи або збільшуючи розмір задачі, щоб спростити вирішення.

4. Взаємодія зі стеком викликів (Call Stack Interaction). Кожен рекурсивний виклик додає новий елемент у стек викликів, і виклик функції вирішується у зворотному порядку, коли досягається базовий випадок або обробляється кожен рекурсивний виклик.

5. Зменшення задачі (Task Reduction). У кожному рекурсивному виклику задача повинна зменшуватися або наближатися до базового випадку, щоб рекурсія була ефективною.

6. Можливість використання для деревоподібних структур даних. Рекурсія особливо корисна під час роботи з деревоподібними або рекурсивними структурами даних, як-от бінарні дерева, графи тощо.

7. Потенційне переповнення стека. Рекурсивний підхід може призвести до переповнення стека, особливо за великої глибини рекурсії. Оптимізація, як-от хвостова рекурсія, може допомогти уникнути цього.

8. Читабельність коду. Рекурсивні алгоритми можуть бути більш читабельними та компактними, особливо коли вони добре підходять для розв’язання конкретних завдань.

9. Можливість заміни ітераційних конструкцій. Рекурсія може бути використана замість ітераційних конструкцій, але варто враховувати витрати на рекурсивні виклики та потенційне переповнення стека.

Оптимізація та нові підходи до використання рекурсії можуть привести до розробки більш ефективних та швидших алгоритмів. Застосування рекурсії в обробці та аналізі великих обсягів даних дає змогу проводити розробки нових алгоритмів для швидкого та ефективного вирішення завдань, пов’язаних з великими об’ємами інформації.

Список використаних джерел

1. Завіша В. В. Алгоритми і структури даних. URL: https://e-tk.lntu.edu.ua/pluginfile.php/20064/mod_resource/content/0/ Тема%209.%20Рекурсія.pdf

2. Завіша В. В. Алгоритмічні стратегії. URL: https://e-tk.lntu.edu.ua/pluginfile.php/20063/mod_resource/content/0/ Тема%2010.%20Алгоритмічні%20стратегії.pdf

УДК 004.2

Попова К. О., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ВИКОРИСТАННЯ РЕКУРСИВНИХ ВІДНОШЕНЬ В АЛГОРИТМІЗАЦІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

Рекурсивною називається програма, яка звертається сама до себе. Особливістю рекурсивної програми є наявність умови завершення, оскільки вона не може викликати себе до нескінченності. З огляду на це, програма, яка містить рекурсію, повинна мати як мінімум два шляхи виконання, один з яких передбачає рекурсивний виклик, а другий – виконання програми без рекурсивного виклику.

Для реалізації цієї умови будь-яка рекурсивна процедура повинна містити базис і крок рекурсії.

Базис рекурсії – це пропозиція, що визначає якусь початкову ситуацію або ситуацію, яка відбувається у момент припинення. Зазвичай в цьому реченні записується якийсь найпростіший випадок, під час якого відповідь виходить відразу навіть без використання рекурсії.

Крок рекурсії – це правило, в тілі якого обов'язково міститься в якості підцілі, виклик обумовленого предиката. Параметри повинні змінюватися на кожному кроці так, щоб внаслідок цього спрацював або базис рекурсії, або умова виходу з рекурсії, розміщена в самому правилі.

У загальному вигляді правило, що реалізує крок рекурсії, буде виглядати так:

<ім'я правила рекурсії>:-
<список предикатів>, <предикат умови виходу>,
<список предикатів>, <ім'я правила рекурсії>,
<список предикатів>.

Реукурсивне відношення для обрахунку чисел факторіалу має вигляд:

Fact (1,1):- !. / * Умова припинення рекурсії * /

Fact (N, F):- N1 = N-1, fact (N1, F1), / * F1 дорівнює факторіалу числа, на одиницю меншого вихідного числа * /

F = F1 * N. / * Факторіал числа дорівнює добутку F1 на саме число *

Використання рекурсії пов'язане з обчисленням чисел Фібоначчі, які можна визначити так: перше і друге число дорівнюють одиниці, а кожне наступне число є сумою двох попередніх. Першим базисом є твердження, що перше число Фібоначчі дорівнює одиниці. Другий базис – аналогічне твердження про друге число Фібоначчі. Крок рекурсії: для обчислення числа Фібоначчі з номером N спочатку потрібно обчислити і додати числа Фібоначчі з номерами N-1 і N-2. Записати ці міркування можна так:

Fib (1,1): - !. / * Перше число Фібоначчі дорівнює одиниці * /
Fib (2,1): - !. / * Друге число Фібоначчі дорівнює одиниці * /
Fib (N, F): -
N1 = N-1, fib (N1, F1), / * F1 це N-1-е число Фібоначчі * /
N2 = N-2, fib (N2, F2), / * F2 це N-2-е число Фібоначчі * /
F = F1 + F2. / * N-е число Фібоначчі дорівнює сумі N-1-го і N-2-го чисел Фібоначчі *

Рекурентне співвідношення – це рівняння або нерівність, яка описує функцію із використанням самої себе, але тільки з меншими аргументами.

Під час розгляду рекурентних співвідношень зазвичай вводяться два спрощення. По-перше, приймається, що час роботи алгоритму $T(n)$ визначається лише для цілочислових n, оскільки в більшості алгоритмів кількість вхідних елементів виражається цілим числом. По-друге, оскільки час роботи алгоритму із вхідними даними фіксованого розміру виражається константою, то в рекурентних співвідношеннях для достатньо малих n зазвичай справедлива тотожність $T(n) = \Theta(1)$. Тому для зручності граничні умови рекурентних співвідношень зазвичай відкидаються, і приймається, що для малих n час роботи алгоритму є $T(n)$ константою. Методи вирішення рекурентних співвідношень:

1. Метод підстановки, який застосовується для вирішення рекурентних співвідношень, складається з двох етапів:

- робиться припущення про вигляд розв'язку;
- за допомогою методу математичної індукції визначаються константи та доводиться, що рішення правильне.

Метод підстановки можна використовувати для визначення або верхньої, або нижньої межі рекурентного співвідношення. В якості прикладу розглянемо верхню границю рекурентного співвідношення:

$$T(n) = 2T([n/2]) + n.$$

Розв'язок має вигляд $T(n) = O(n \times lgn)$. Метод полягає в доведенні того, що за придатного вибору константи $c>0$ виконується нерівність $T(n) \le c \times n\ lgn$.

2. Метод дерев рекурсії – прямий шлях до вдалого припущення. В дереві рекурсії кожний вузол представляє час, необхідний для виконання окремо взятої підзадачі, яка розв'язується за одного з багатьох рекурсивних викликів функції. Далі значення часу роботи окремих етапів підсумовується в межах кожного рівня, а потім – за всіма рівнями дерева, внаслідок чого отримуємо повний час роботи алгоритму.

Дерева рекурсії краще за все підходять для того, щоб допомогти зробити припущення про вигляд розв'язку, який потім перевіряється методом підстановок. Водночас у припущенні часто допускається наявність невеликих неточностей, оскільки воно потім все одно перевіряється. Якщо ж побудова дерева рекурсії та підсумування час роботи за всіма складниками відбувається ретельно, то саме дерево рекурсії може стати джерелом доведення коректності розв'язку. (рис. 1).

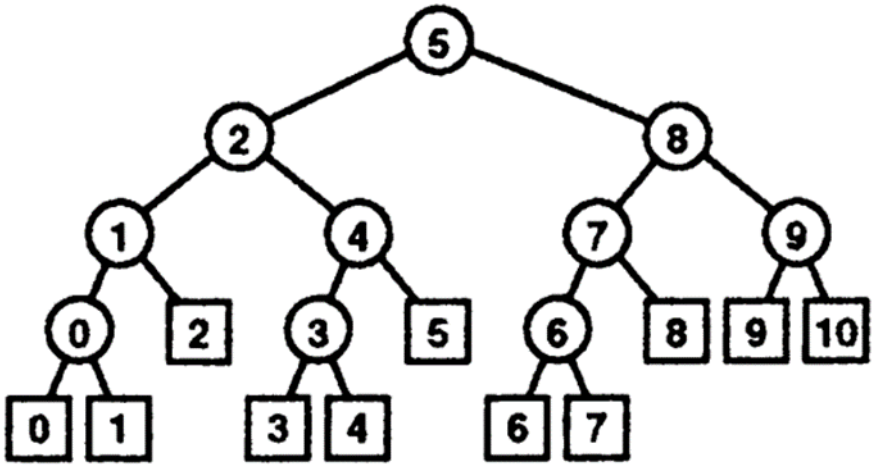


Рисунок 1. Дерево рекурсії

Отже, розглянуто поняття рекурсії та рекурсивних відношень. В алгоритмізації рекурсія є одною з найпотужніших технологій реалізації алгоритмів за принципом «розділяй та володарюй».

Список використаних джерел

1. Introduction to Algorithms / H. C. Thomas, E. L. Charles, L. R. Ronald, Clifford S. 3rd Edition. MIT PRESS, 2009. 1292 p.

2. Креневич А. Алгоритми та структури даних. Київ: ВПЦ Київський Університет, 2018. 172 с.

3. Мелешко Є. В., Якименко М. С., Поліщук Л. І. Алгоритми та структури даних. Кропивницький: Видавець – Лисенко В. Ф., 2019. 156 с.

УДК 519.8

Семен О. Д., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ВИКОРИСТАННЯ АЛГОРИТМУ A-STAR ДЛЯ ОПТИМІЗАЦІЇ ПОСТАЧАННЯ БОЄПРИПАСІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Питання логістики відіграє дуже важливу роль у сучасному світі. Логістика – це не лише планування маршрутів, це комплексний підхід до управління потоками, ресурсами та процесами, спрямований на забезпечення оптимального переміщення матеріальних цінностей чи інформації від початкової точки до кінцевого пункту призначення [1].

Логістика важлива у багатьох галузях. Якщо розглядати військову сферу, логістика – це один з основних аспектів підтримки боєздатності армії. Швидкість, точність, раціональність – три найважливіші пункти для успішності доставки. Для забезпечення цих умов існують різні способи прокладання маршрутів із баз постачання до місця призначення.

Теорія графів – розділ математики, який займається вивченням графів та використовується для моделювання парних відношень між певними об'єктами [2]. Теорія графів може використовуватися для розв'язання логістичних задач, оскільки вона дає змогу враховувати складність шляху під час пошуку найкоротшого маршруту. Існує велика кількість алгоритмів для пошуку найкоротшого шляху, наприклад: Дейкстри, Беллмана-Форда, Флойда-Уоршелла, Джонсона та інші. Їх різниця полягає в тому, для яких графів їх можна застосовувати.

Розглянемо практичність використання теорії графів та алгоритмів пошуку найкоротшого шляху на прикладі постачання боєприпасів. Для того, щоб прокласти найкращий маршрут, потрібно враховувати низку факторів:

1) безпечність маршруту, оскільки боєприпаси – це дуже вразлива ціль, знищення якої може понести за собою великі втрати;

2) якість доріг має значення для швидкості доставки, зношування техніки, цілісності вантажу;

3) швидкість доріг: варто враховувати тип дороги, тому що по шосе довшу відстань можна проїхати швидше, ніж коротшу по ґрунтовій дорозі;

4) непомітність: під час переміщення військових важливо враховувати фактор непомітності, оскільки ворожа розвідка може довідатися про плани та спробувати стати на заваді.

Окрім цих чотирьох, можна враховувати ще багато інших факторів. Для перелічених вище факторів створюється шкала оцінювання, на якій найменше значення – найсприятливіші умови, найбільше – несприятливі умови.

Наступним кроком потрібно створити граф. Процес створення графу може бути складним і зазвичай складається з таких етапів:

1. Визначення вузлів графу: це можуть бути населені пункти, точки інтересу або будь-які місця, між якими потрібно знайти шлях.

2. Створення ребер графу: визначення шляхів між цими вузлами. Це можуть бути вулиці, дороги, транспортні лінії або будь-які з'єднання між вузлами, через які може пересуватися техніка.

3. Визначення факторів для обчислення ваги ребер: вага ребра графу відображає складність проходження цього шляху, тому під час її обчислення варто враховувати відстань, час, вартість, ступінь безпеки, тип дороги чи будь-який інший фактор, який впливає на вибір найкращого шляху.

4. Обчислення ваг ребер: вага кожного ребра може обчислюватися різними способами, наприклад, з допомогою простого додавання оцінок або створення коефіцієнтів. Просте додавання оцінок являє собою сумування величин, що відображають відстань, час, вартість тощо. Внаслідок цього отримане число буде вагою ребра. Створення коефіцієнтів означає використання ваги, яка складається зі співвідношення різних факторів (наприклад, врахування відстані та трафіка з допомогою коефіцієнтів).

5. Спрощення або ускладнення моделі: враховуючи обсяг даних та реалістичність, буде доцільно використовувати спрощені моделі графів, або навпаки, додавати додаткові атрибути до ребер для більшої точності.

Після того, як граф буде створено, можна застосовувати алгоритм A-star. Цей алгоритм використовують у багатьох галузях (штучний інтелект, комп'ютерні ігри, робототехніка, навігація та інші). Серед переваг цього алгоритму можна виділити такі [3]:

1. Ефективність. Алгоритм A* є оптимізованим методом, який використовує менше ресурсів, порівняно з іншими алгоритмами, як-от пошук у ширину чи пошук у глибину.

2. Гарантована оптимальність. Якщо вартість переміщення між вузлами (або вага ребер) у графі є додатною і не перевищує деяку межу, A* забезпечує знаходження найкоротшого шляху.

3. Можливість використання різних функцій евристики. A* дає змогу використовувати різні функції оцінки відстані між поточним вузлом та цільовим, що допомагає оптимізувати пошук для конкретних умов задачі.

Сукупність цих переваг дає змогу вважати його хорошим вибором для пошуку маршруту з пункту А в пункт Б. Правильно створивши граф та застосувавши алгоритм A*, можна оптимізувати постачання боєприпасів на фронт, підставляючи актуальні дані в граф.

Отже, теорія графів є дуже потужним інструментом у сфері логістики в різних галузях, зокрема й у військовій. Використовуючи її разом із алгоритмом пошуку найкоротшого шляху A*, можна оптимізувати постачання боєприпасів і не тільки. Наприклад, її можна використовувати для розгортання військ, налагодження мереж зв'язку, планування та координації операцій. Загалом використання теорії графів разом із різними алгоритмами пошуку найкоротших шляхів допомагає оптимізувати різні процеси та забезпечити кращу обороноздатність і ефективність військ.

Список використаних джерел

1. Логістика – наука про організацію і вдосконалення матеріалопотоків. URL: https://mk.nmu.org.ua/ua/source/Logisticl1.pdf (дата звернення: 01.12.2023).

2. Теорія графів. URL: https://ela.kpi.ua/bitstream/123456789/35854/1/Teoriia_hrafiv.pdf (дата звернення: 01.12.2023).

3. A* search algorithm. URL: https://en.wikipedia.org/wiki/A*_search_algorithm (дата звернення: 01.12.2023).

УДК 004.89

Семен О. Д., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

АНАЛІЗ ДАНИХ У ПРИЙНЯТТІ РІШЕНЬ У СУЧАСНОМУ БІЗНЕСІ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасних реаліях темпи життя дуже швидко зростають, нещодавні потреби людства втрачають актуальність, і з'являється багато нових потреб. Тенденції ведення бізнесу також змінюються в ногу з часом. Успішний сучасний бізнес відрізняється своєю креативністю та гнучкістю, оскільки потрібно якомога швидше відгукуватись на збільшення та зменшення попиту на ринку. Правильні рішення можуть принести значний прибуток, неправильні – великі збитки.

Аналіз даних – це процес перетворення та моделювання даних з метою виявлення корисної інформації [1]. Цей процес містить в собі багато методів, спрямованих на виявлення закономірностей, які можна використовувати для різних цілей.

Аналіз даних є невід'ємною частиною сучасного бізнесу, який дає змогу підприємствам отримувати цінну інформацію, краще розуміти клієнтів, прогно-

зувати продажі та приймати обґрунтовані рішення [2]. Простіше кажучи, він допомагає сформувати надійне підґрунтя для тих чи інших рішень.

Як було зазначено раніше, попит на ринку швидко змінюється, тому потрібно вчасно реагувати на ці зміни, щоб бізнес працював успішно. Для цього необхідно обробляти велику кількість даних, які можуть бути структурованими, неструктурованими, потоковими. Big Data – це термін, яким визначають великі обсяги даних, які характеризуються великою об'ємністю, різноманітністю та швидкістю надходження. Для їх обробки потрібний особливий підхід та інструменти. Фреймворк Apache Hadoop спрощує обробку даних завдяки розбиттю великого завдання на маленькі, а також створює можливість розподілити обробку на кластері комп'ютерів. Apache Spark – рішення, з допомогою якого можна швидко обробити та проаналізувати великі набори даних. Apache Kafka – платформа, що забезпечує надходження та аналіз потоків даних у реальному часі. Також використовують NoSQL бази даних, які дають змогу зберігати та обробляти неструктуровані дані, наприклад, тексти, фотографії, відео.

В сучасному бізнесі велика кількість бізнес-процесів, проблеми в яких можуть перешкоджати успішній роботі бізнесу. Аналіз даних відіграє ключову роль у оптимізації бізнес-процесів, даючи змогу підвищувати їх ефективність та зменшувати витрати. Аналіз даних є корисним у таких задачах:

1. Ідентифікація слабких місць: аналіз даних допомагає оптимізувати та вдосконалити бізнес-процеси, виявляючи слабкі місця, де компанія зазнає зайвих витрат.

2. Оптимізація цінової стратегії: аналіз даних дає змогу визначити оптимальні цінові точки на продукцію чи послуги, що може позитивно впливати на обсяги продажів та прибуток.

3. Покращення обслуговування клієнтів: із допомогою аналізу даних можна розуміти потреби та побажання клієнтів, що допоможе адаптуватися під потреби ринку та збільшити прибуток. Чим краще у клієнтів ставлення до компанії, тим більше зворотних відгуків та перспектив для розвитку.

4. Вдосконалення ланцюга постачань: використовуючи аналіз даних, можна оптимізувати ланцюг постачань, покращуючи взаємодію з постачальниками, обираючи ефективніші логістичні рішення.

5. Автоматизація процесів: використовуючи аналіз даних, можна знайти процеси які повторюються регулярно та мало змінюються. Такі процеси можна автоматизувати, що посприяє ефективнішому використанню ресурсів.

Ефективне управління ресурсами в сучасному бізнесі є одним із ключових факторів для досягнення успіху та сталого розвитку компаній [3]. Визначення, які ресурси найбільше впливають на прибуток, а які є надлишковими, стає завданням критичного значення для оптимізації фінансових витрат та максимізації

ефективності. На основі цього аналізу можна розробити ресурсну стратегію, яка не лише забезпечить економію витрат, а й покращить ефективність бізнесу загалом. Такий підхід допомагає компаніям зосередити зусилля на найбільш важливих напрямах, підвищуючи конкурентоспроможність та стійкість до змін.

Отже, важливість методів аналізу даних у прийнятті рішень для сучасного бізнесу неможливо переоцінити. Використовуючи методи аналізу даних, можна оптимізувати роботу бізнесу та суттєво збільшити прибутки. Також, за такого підходу можна покращити популярність бренду серед клієнтів завдяки підвищенню рівня послуг. Загалом сукупність усіх розглянутих факторів призведе до стабільного зростання бізнесу, оскільки мінливість зовнішніх умов не матиме такого великого впливу.

Список використаних джерел

1. Аналіз даних. URL: https://library.sumdu.edu.ua/uk/doslidnyku/prohramne-zabezpechennia/analiz-danykh-ta-vizualizatsiia/instrumenty-dlia-analizu-danykh.html (дата звернення: 15.11.2023).

2. Важливість аналізу даних у прийнятті бізнес-рішень. URL: https://whitesales.ua/uk/blog/cekreti-uspihu-vazhlivist-analizu-danih-u-priynyatti-biznes-rishen (дата звернення: 15.11.2023).

3. Оцінювання ефективності використання ресурсів у підприємницькій діяльності. URL: http://srd.pgasa.dp.ua:8080/handle/123456789/2013 (дата звернення: 15.11.2023).

4. Алексюк В. В., Січко Т. В. Дослідження особливостей програмного забезпечення для аналізу даних. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2022. С. 249–251.

УДК 004.94+519.876.5

Семенюк А. М., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ОПТИМІЗАЦІЯ ВИРОБНИЦТВА РЕАБІЛІТАЦІЙНИХ ЗАСОБІВ ІЗ ВИКОРИСТАННЯМ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

Донецький національний університет імені Василя Стуса, м. Вінниця

Оскільки розвиток комп'ютерних технологій, їх впровадження в медицину і охорону здоров'я потребує здійснення аналізу захворюваності, стану пацієнта під час реабілітації, постлікувального стану, подальшого визначення заходів

зі зменшення ризиків епідемій, спалахів хвороб, необхідності засобів реабілітації та лікування, то моделювання в медицині набуває все більш важливого значення, особливо в умовах сьогодення.

Методи моделювання можуть бути використані в різних сферах медицини, особливо у сферах проєктування, управління та постачання, де основними є процеси ухвалення ефективних рішень на основі отримуваної інформації. Головною метою цих заходів є отримання, аналіз, представлення та подальше використання даних. Під час моделювання можна використовувати для аналізу великі обсяги даних із різних джерел, як-от статистичні бази даних закладів охорони здоров'я, виробничих потужностей спеціалізованих підприємств, наукова література та результати випробування різних методів і пристроїв для лікування та реабілітації.

Комп'ютерне моделювання будь-яких процесів базується на трьох основних поняттях:

- факти;
- правила;
- керуюча структура.

Факти – це запас знань про об'єкт моделювання. Ці знання об'єднуються в окремі групи за певними ознаками. Такий підхід дає змогу не запам'ятовувати характеристики кожного факту, а лише параметри групи, до якої він належить.

Правила дають змогу винайти відмінність між класами. Цей процес доволі складний, більшою мірою інтуїтивний, але під час обмеження моделювання «вузькими» напрямами можливий до вирішення.

Керуюча структура – механізм використання правил. У ній визначається послідовність використання правил і групи фактів, до яких їх потрібно використати.

Збір та аналіз інформації, даних первинного, діагностичного та ін. оглядів здоров'я населення; моделювання та представлення результату в графічному вигляді з можливістю зміни часових меж та різних чинників середовища дасть змогу з великою ймовірністю отримати результат розвитку подій.

Використання пакету моделювання AnyLogic дає змогу представити початкові дані для аналізу та вирішувати безліч управлінських завдань у сфері охорони здоров'я: від планування заходів запобігання епідеміям до оптимізації планування використання приміщень у медичному закладі та організації роботи медичного закладу і розробки стратегії виведення на ринок нових засобів реабілітації (ЗР) та їх використання. Цей пакет допомагає сформувати потрібну керуючу модель відповідно до наявних правил та фактів.

У цій роботі розглядається можливість моделювання необхідності засобів реабілітації для осіб з інвалідністю.

Організаційні процеси можна зобразити у вигляді інтерактивної моделі (рис. 1), яка дасть змогу проаналізувати необхідність замовлення предметів для відновлення стану, можливості підприємств із їх випуску, оптимального використання ресурсів, виявити ризики та слабкі місця. Зрештою, отримати наочне уявлення про напрям розвитку і дійти оптимальних рішень.

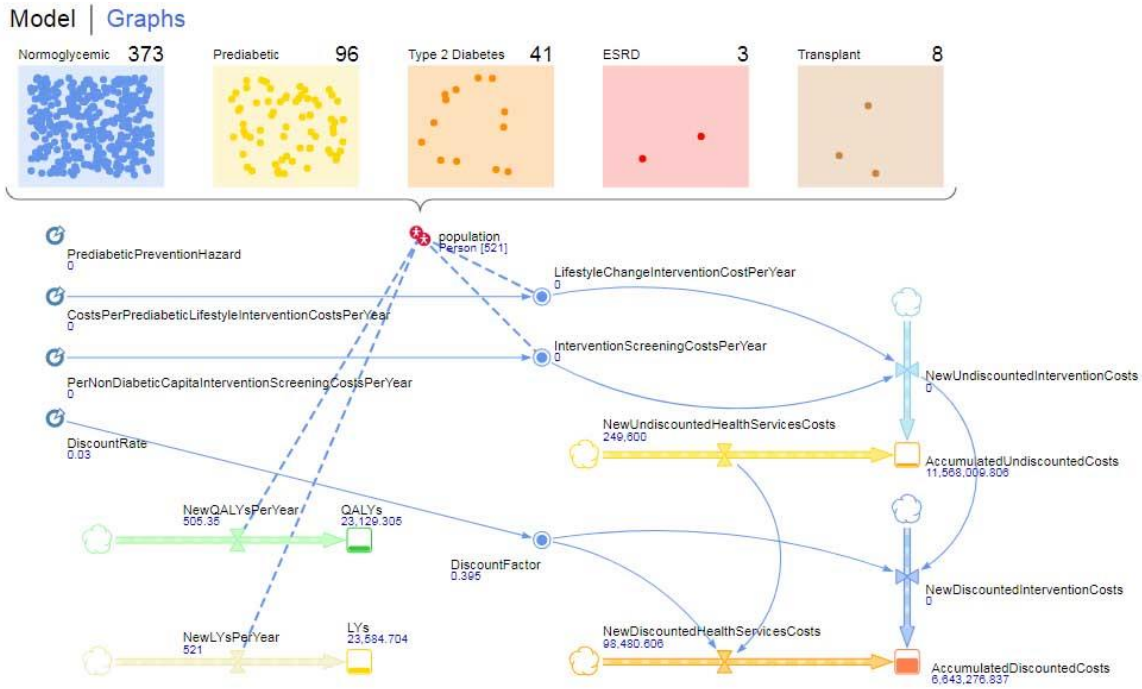


Рисунок 1. Приклад моделювання необхідності ЗР залежно від типу захворювання

Для моделювання зміни ресурсів (основних фондів і трудових ресурсів) агентів-підприємств будемо використовувати методи системної динаміки (з використанням діаграми станів – Statechart), а для моделювання можливих станів підприємств – методи агентного моделювання (рис. 2).

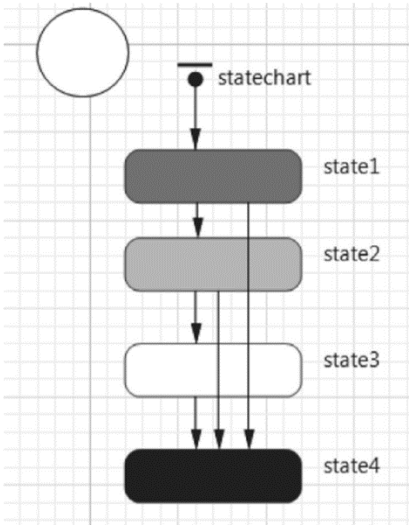


Рисунок 2. Приклад агентного моделювання виробництва ЗР

Statechart – це найдосконаліша конструкція для опису поведінки, що керується визначеними подіями та залежить від часових проміжків. Вона дає змогу графічно задавати поведінку об'єкта з використанням блоків діаграм. Такий підхід допомагає за необхідності визначити більш складну поведінку об'єкта.

Для створення прогнозуючої моделі використовують інформацію з медичних карток пацієнта і порівнюють її з поточними обсягами виробництва, щоб запропонувати відповідні зміни.

Новітні технології дали змогу персоналізувати підхід до відновлення і профілактики з використанням пристроїв та засобів для реабілітації здоров'я, які допомагають приймати обґрунтовані рішення щодо необхідності у ЗР.

Під час моделювання можливе аналізування великих обсягів даних про пацієнтів для виявлення закономірностей, кореляцій і взаємозв'язків між різними змінними, як-от демографічна інформація, історія хвороби та історія лікування. Ця інформація стане у нагоді під час розробки індивідуальних планів лікування, її можна використовувати для розробки нових типів та видів відновлювальних та допоміжних засобів, прогнозуючи, які з них будуть найефективнішими та найменш токсичними. Використання пакету моделювання на основі статистичних даних може покращити дієвість і навіть дизайн таких засобів – зовнішній і конструктивний.

Імітаційні моделі здатні відобразити динаміку систем надання послуг із реабілітації, що дасть змогу оцінити їх ефективність. Це полегшить розуміння специфіки проблеми та сприятиме тісній співпраці між замовниками (закладами охорони здоров'я, особами з інвалідністю), виробниками та відділами соціального захисту населення. Неперевершені можливості імітаційного моделювання та візуалізація забезпечують надійні та позбавлені ризиків нововведення.

Список використаних джерел

1. Ніколюк П. К. Моделювання систем: навч. посіб. для здобувачів вищої освіти спеціальності 122 Комп'ютерні науки. Вінниця: ДонНУ імені Василя Стуса, 2023. 275 с.

2. Комп'ютерне моделювання систем та процесів. Методи обчислень. Частина 1: навч. посіб. / Р. Н. Квєтний, І. В. Богач, О. Р. Бойко, О. Ю. Софина, О. М. Шушура; за заг. ред. Р. Н. Квєтного. Вінниця: ВНТУ, 2012. 193 с.

3. Основні показники медико-соціальної реабілітації осіб з інвалідністю в Україні за 2022 р.: Аналітико-інформаційний довідник / В. І. Шевчук, Р. Я. Перепелична, Н. М. Беляєва, Л. О. Сторожук, І. В. Куриленко, Л. Г. Семененко, М. В. Семенюк, А. М. Семенюк. Вінниця: ФОП Данилюк В. Г., 2023. 112 с.

УДК 004.2

Скороход О. М., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ГРАФИ ТА ПІДХОДИ ДО АЛГОРИТМІЧНИХ РІШЕНЬ НА ГРАФАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Графи являють собою структуру даних, із допомогою якої можна ідентифікувати моделі зв'язків. Практичне застосування графів можливе, коли задача представлена у вигляді з'єднань вершин та відстаней між ними, тобто підходить для опису маршрутів, мереж тощо. Алгоритми на графах призначені для моделювання зв'язків між різними об'єктами. До основних алгоритмів на графах належать: пошук у глибину, пошук у ширину, пошук найкоротшого шляху.

1. Пошук у глибину (DFS) – це алгоритм обходу графу, що використовує стратегію глибокого спуску для систематичного вивчення його структури. Починаючи з обраної вершини, алгоритм рекурсивно відвідує всі сусідні вершини, доки не буде досліджено весь граф. Процедура формалізується з допомогою вибору початкової вершини, визначення її відвіданою, застосування рекурсії для сусідніх невідвіданих вершин.

Цей метод дає змогу алгоритму систематично досліджувати граф, виявляючи його структуру та взаємозв'язки. DFS ефективно використовується для виявлення компонент зв'язності, циклів та аналізу характеристик графу у різних галузях, як-от комп'ютерні науки, біоінформатика та аналіз соціальних мереж [3].

2. Пошук у ширину (BFS – Breadth-First Search): алгоритм пошуку в ширину (BFS) є фундаментальним методом обходу графів та розв'язання задач найкоротших шляхів та аналізу структури графів. Запропонований Леонардом Ейлером у XVIII ст. і систематизований Гарі Каспером у 1950 р., цей алгоритм є ефективним та універсальним. Алгоритм будується на таких складниках: ініціалізація початкової вершини, створення черги, пошаровий обхід вершин, виявлення найкоротших шляхів, точка завершення під час відвідування усіх вершин.

Алгоритм BFS застосовується у транспортній логістиці для оптимізації маршрутів, в аналізі соціальних мереж для виявлення взаємодій, а також у теорії комп'ютерних мереж для ефективного обходу та маршрутизації даних [2].

3. Алгоритм Дейкстри, спрямований на ефективне знаходження найкоротших шляхів у графі з невід'ємними вагами ребер, розроблений Едсгером Дейкстрою у 1959 р. Цей алгоритм визначає оптимальний шлях від однієї фіксованої стартової вершини до всіх інших вершин графу. Принцип роботи ґрунтується на

таких складниках: ініціалізація відстані від стартової вершини до всіх інших, реалізація жадібного вибору вершини з найменшою відстанню від стартової з ознакою відвіданої, оновлення відстаней шляхом перевірки можливості скорочення відстані, повторення процесу перевірки.

Алгоритм Дейкстри широко використовується в телекомунікаціях для пошуку найкоротших шляхів у комп'ютерних мережах, у транспортному моделюванні для оптимізації маршрутів, у системах навігації та геоінформаційних системах, а також для управління ресурсами мережі в інформаційних системах [1].

4. Алгоритм Беллмана–Форда для розв'язання задачі пошуку найкоротших шляхів у напівзважених або зважених графах з урахуванням можливості від'ємних ваг ребер, розроблений американськими вченими Р. Беллманом і Л. Фордом у 1956 р. Алгоритм знаходить широке застосування в галузях оптимізації мереж, транспортних систем і аналізу великих даних. Алгоритм будується на виконанні таких процесів: ініціалізації початкової вершини з нульовою вартістю та інших невизначених вершин, проведення ітеративного оновлення вартості вершин під час порівняння їх із вартістю поточного найкоротшого шляху та вагою ребра, виявлення циклів з від'ємною сумою ваг.

Цей алгоритм застосовується в мережевому плануванні, транспортній логістиці та управлінні ресурсами, де важливо враховувати від'ємні ваги або цикли з від'ємною сумою ваг. Це ефективний інструмент для розв'язання задач оптимізації в умовах невизначеності.

Отже, сутність алгоритмів на графах зводиться до визначення ознак проходження вершин та зв'язків між ними, що дає змогу ефективно використовувати ці алгоритми під час опису мереж різного призначення, зокрема комунікаційних та транспортних.

Список використаних джерел

1. Sedgewick R., Wayne K. Algorithms. Addison-Wesley Professional, 2011. 976 p.

2. Bhargava A. Grokking Algorithms. An Illustrated Guide for Programmers and Other Curious People. Manning Publications Co, 2016. 256 p.

3. Aho A., Hopscroft J., Ullman J. Data Structures and Algorithms. Pearson, 1983. 448 p.

4. Креневич А. П. Алгоритми і структури даних: підручник. Київ: ВПЦ Київський Університет, 2021. 200 с. URL: https://www.mechmat.univ.kiev.ua/wp-content/uploads/2021/09/ pidruchnyk-alhorytmy-i-struktury-danykh.pdf

УДК 004.056

Труханська В. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ОГЛЯД НАПРЯМІВ ПРИХОВАНОЇ ПЕРЕДАЧІ ДАНИХ У ВІДЕОФАЙЛАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі, коли все більше інформації передається в цифровому вигляді, проблема захисту інформації від стороннього втручання є доволі поширеною. Електронне підслуховування, шахрайство, комп'ютерні віруси та інші електронні загрози можуть призвести до серйозних наслідків, як-от втрата конфіденційності або фінансові втрати. Дослідження методів цифрової стеганографії – це актуальне завдання, яке сприятиме покращенню захисту інформації в цифровому світі.

Існують два основні напрями розв'язання задачі прихованої передачі даних: криптографія та стеганографія. Метою криптографії є обмеження доступу до інформації шляхом її шифрування. На відміну від криптографії, стеганографія дає змогу приховати сам факт наявності прихованих даних. Бурхливий розвиток інформаційних технологій в останні роки дав суттєвий поштовх для появи і покращення методів комп'ютерної стеганографії. З'явились нові варіанти застосування – приховані повідомлення вбудовують у графічні, аудіо- та відеоматеріали, текстові файли та навіть у файли програм [1]. Комп'ютерна стеганографія базується на двох принципах: файли, що містять зображення чи звукові матеріали, можуть бути певною мірою змінені без втрати функціональності, і органи чуття людини не здатні відрізнити незначні зміни в кольорі або якості звуку. Другий принцип можна успішно використовувати з огляду на надлишковість деяких сучасних аудіо- та графічних форматів: наприклад, зміна найменш значимих бітів 24-бітного зображення, які відповідають за колір конкретного пікселя, не призводить до явних змін у зображенні [2].

Сьогодні через збільшення об'ємів інформації та збільшення пропускної здатності каналів зв'язку все більшу актуальність має питання приховування інформації у відеопослідовностях. Передача цифрового відео в останні роки є типовою подією і не викликає підозр. Наприклад, сервіс YouTube нараховує сотні мільйонів відеофайлів, причому один і той же відеоматеріал зустрічається в різних форматах. Велика кількість відеофайлів розміщується в P2P-мережах. Були розглянуті деякі особливості використання форматів відеофайлів для приховування інформації. Незважаючи на те, що існує велика кількість відеоформатів, на практиці для приховування інформації використовуються формати

MPEG-2 і MPEG-4. Розглянемо три способи вбудовування інформації в файли формату MPEG-2: вбудовування на рівні коефіцієнтів, на рівні бітової площини і завдяки енергетичній різниці між коефіцієнтами [5].

У першому способі біти приховуваної інформації вбудовуються в коефіцієнти дискретного косинусного перетворення (ДКП). Головною проблемою модифікації коефіцієнтів ДКП в стисненому потоці відео є накопичення зміщень та помилок. Спотворення, викликані зміною коефіцієнтів ДКП, можуть поширюватися в часовій і в просторовій областях, тому для компенсації спотворень додають спеціальний сигнал. Через обмеження бітової швидкості, під час додавання даних змінюються лише 10–20 % коефіцієнтів ДКП. Під час використання цього методу приховувана інформація зберігається під час фільтрування, зашумлення адитивним шумом і дискретизації [3].

Другий метод відрізняється високою пропускною здатністю і невеликою обчислювальною складністю. Але є й істотний недолік: інформація вбудована так, що може бути легко видалена. Під час повторного накладенні послідовності біт якість відео погіршиться незначно, а прихована інформація буде знищена.

В основі останнього методу лежить диференціальне вбудовування енергії (ДВЕ). Цей метод може бути застосований не лише до відеоданих MPEG, але і до інших алгоритмів стиснення відео. Інформація вбудовується шляхом видалення декількох коефіцієнтів ДКП, і це має свої переваги. Алгоритм ДВЕ вносить у відео менше спотворень, ніж метод вбудовування інформації на рівні бітової площини. Для видалення прихованої інформації потрібне проведення більш складних обчислювальних операцій, ніж вбудовування нової довільної бітової послідовності [4].

Стеганографія – перспективний метод захисту даних, що ґрунтується на маскуванні зберігання та передачі інформації у масиві даних контейнера. Для досягнення мети розглянуто низку прийомів, із яких найбільш ефективним є застосування надлишковості медіафайлів. У процесі дослідження було розглянуто особливості приховування інформації у відеофайлах, здійснено порівняння наявних алгоритмів комп'ютерної відеостеганографії.

Список використаних джерел

1. Стеганографія: навч. посіб. / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. Харків: Вид. ХНЕУ, 2011. 232 с.

2. Комп'ютерна стеганографія: навч. посіб. / В. О. Хорошко, Ю. Є. Яремчук, В. В. Карпінець. Вінниця: ВНТУ, 2017. 155 с.

3. Що таке стеганографія? URL: https://uk.theastrologypage.com/steganography (дата звернення: 24.11.2023).

4. Digital Watermarking and Steganography / I. Cox, M. Miller, J. Bloom, J. Fridrich, T. Kalker. London: Elsevier, 2008. 593 p.

5. Wayner P. Disappearing Cryptography: Information Hiding: Steganography and Watermarking. London: Elsever, 2009. 440 p.

УДК 004.43

Труханська В. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА МОВ ОБРОБКИ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі дані стають все більш важливими. Вони використовуються в різних сферах діяльності, від бізнесу до науки. Для аналізу їх даних використовуються різні методи, зокрема статистичний аналіз та машинне навчання, а для реалізації цих методів – мови програмування. У наш час існує велика кількість мов програмування, які дають змогу систематизувати послідовність операцій, що здійснюються з даними, насамперед у комп'ютері, для отримання нової інформації шляхом обчислень, перегляду і уточнення наявної інформації. Найпопулярнішими мовами є Python та R.

Python – це універсальна високорівнева мова програмування, яка широко використовується в різних галузях, а також орієнтована на підвищення продуктивності розробника і полегшення процесу читання коду. Python має простий та інтуїтивно зрозумілий синтаксис, що робить його відносно легким для вивчення. Водночас стандартна бібліотека має достатній обсяг корисних функцій [1].

Мова R спеціально розроблена для статистичного аналізу та машинного навчання і використовується для аналізу даних та складання прогнозів. R має широкий спектр вбудованих функцій, бібліотек та пакетів для вирішення різних завдань. Для статистики, управління даними та їх візуалізації існують стандартні функції та прогресивні алгоритми машинного навчання. До того ж R має велику спільноту користувачів, які створюють пакети для різних завдань [2].

Отже, спершу необхідно зрозуміти, в яких випадках краще застосовувати R, а в яких Python. R зазвичай застосовується в тих випадках, коли для аналізу даних потрібні виділені обчислювальні потужності або окремі сервери. R добре підходить для дослідницької роботи, зручна та практична за будь-якого варіанта аналізу даних, оскільки в мові R існує безліч пакетів, а також готові тести, які забезпечують потрібний інструментарій для швидкого старту. R навіть може бути корисною під час роботи з великими даними [3].

Python може знадобитись у випадках, коли завдання, які пов'язані з аналізом даних, вмонтовуються в роботу вебдодатків, або якщо статистичний код

потрібно ввести в робочу базу даних. Python, який є повнофункціональною мовою програмування, добре підходить для реалізації алгоритмів з їх подальшим практичним використанням. Ще нещодавно пакети для аналізу даних на Python перебували на початковому етапі розробки, що становило певну проблему, але останніми роками ситуація значно покращилася [3].

Щоб дані були виразнішими та зрозумілішими, їх можна візуалізувати для інформативнішого і зручнішого їх аналізу. Хоча в Python і є зручні бібліотеки для візуалізації, наприклад, Seaborn, Bokeh і Pygal, вибір може бути надмірно великий [4]. До того ж порівняно з R, візуалізація на Python влаштована набагато складніше, а її результати іноді не надто наочні. Мова R створена саме для візуалізацій. Існують стандартні пакети для візуалізації, як-от: ggplot2, ggvis, googleVis і rCharts [3]. Зважаючи на все, можна зробити висновок, що мова R краще візуалізує дані. У ній результати краще відображати графічно, оскільки вони більш наочні у більшості проєктів. Мова R надає масштабний і якісний контент.

Python та R – це потужні мови програмування, які можна використовувати для аналізу даних. Python є універсальною мовою, яка добре підходить для різних задач, зокрема аналізу даних, водночас R спеціально розроблена для статистичного аналізу, має широкий спектр вбудованих функцій. Вважаємо, R краще підходить для дослідницької роботи та зручної візуалізації даних, однак Python легший для вивчення, ніж мова R, тому вибір між Python та R залежить від конкретних потреб користувача.

Список використаних джерел

1. Python у науці про дані: Як використовувати Python для аналізу даних та машинного навчання. URL: https://buki.com.ua/blogs/python-u-nauci-pro-dani-iak-vikoristovuvati-python-dlia-analizu-danix-ta-masinnogo-navcannia/ (дата звернення: 02.12.2023).

2. Rahlf T. Data Visualisation with R. Springer International Publishing, New York, 2017.

3. Python Vs R: Know The Difference. URL: https://www.interviewbit.com/blog/python-vs-r/ (дата звернення: 03.12.2023).

4. R vs. Python: 12 Key Comparisons.URL: https://www.spiceworks.com/tech/devops/articles/r-vs-python/ (дата звернення: 03.12.2023).

УДК 004.42+519.1+004.94

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Труханська В. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ МІСЬКИХ ТРАНСПОРТНИХ СИСТЕМ

Донецький національний університет імені Василя Стуса, м. Вінниця

У світі спостерігається стрімка урбанізація, що призводить до збільшення чисельності міського населення та активного розвитку міст. Це явище ставить перед владою та управлінцями великі виклики у забезпеченні ефективності та безпеки транспортних систем.

За останні десятиліття транспортні системи міст стали предметом різноманітних досліджень через збільшення кількості заторів (рис. 1), недостатню мобільність і незручності для мешканців.

Рисунок 1. Типовий варіант міського транспортного колапсу

Ця проблема є особливо актуальною в контексті збільшення попиту на транспортні послуги та збільшення кількості транспортних засобів на дорогах, що призводить до загострення проблем руху та збільшення викидів забруднюючих речовин [1].

Забезпечення оптимальної транспортної інфраструктури для великих міст є ключовим завданням для забезпечення не лише комфорту громадян, але й збереження екологічної стабільності та зменшення забруднення навколишнього

середовища. Розглянемо основну методологію моделювання та оптимізації транспортної системи міста.

Збір та аналіз даних:

➢ Статистичні дані про транспортну систему: збір і аналіз наявних даних про транспортні потоки, рух транспортних засобів, обсяги перевезень у різні періоди, зокрема пікові години та пікові дні.

➢ Географічні дані: зібрання географічних даних для аналізу інфраструктури міста, зокрема дороги, зупинки громадського транспорту, розташування промислових зон тощо.

Моделювання транспортної системи:

➢ Математичні моделі трафіка: розроблення моделей, які враховують рух транспорту, можливі затори, час очікування та оптимальні маршрути.

➢ Геопросторові аналізи: використання географічних інформаційних систем для визначення оптимальних транспортних маршрутів та виявлення найбільш навантажених ділянок.

Аналіз моделі на ефективність транспортної системи міста:

➢ Локалізація заторів – ідентифікація місць і часу заторів, аналіз перевантажених ділянок доріг та систем громадського транспорту.

Розробка стратегій оптимізації:

➢ Інфраструктурні зміни: пропозиції щодо вдосконалення наявної інфраструктури, будівництва нових доріг, пішохідних та велосипедних доріжок, розвитку громадського транспорту.

➢ Впровадження технологій: розгляд можливостей впровадження технологічних інновацій, як-от системи керування рухом, використання сучасних інформаційних технологій для оптимізації маршрутів тощо.

Оцінка та прогнозування результатів:

➢ Оцінка ефективності стратегій: аналіз і прогнозування ефективності запропонованих стратегій на основі математичних моделей та симуляцій.

➢ Прогнозування змін: оцінка очікуваних змін у транспортній системі міста після впровадження пропозицій та стратегій оптимізації [2].

У роботі представлено метод оптимізації сучасної транспортної інфраструктури у містах, який включає виявлені ключові проблеми та фактори, що впливають на їх ефективність. Ці проблеми охоплюють широкий спектр аспектів, від точок заторів до недостатнього розвитку громадського транспорту та неефективного використання наявної інфраструктури.

Список використаних джерел

1. Стеценко І. В. Моделювання систем: навч. посіб. М-во освіти і науки України, Черкас. держ. технол. ун-т. Черкаси: ЧДТУ, 2010. 399 с.

2. Transport system. URL: https://www.sciencedirect.com/topics/earth-and-planetary-sciences/transport-system (дата звернення: 01.12.2023).

УДК 004.021

Щербина Д. С., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ЗНАХОДЖЕННЯ ОПТИМАЛЬНОГО МАРШРУТУ ПОСТАЧАННЯ ВІЙСЬКОВОЇ ПРОВІЗІЇ ЗА ДОПОМОГОЮ АЛГОРИТМУ ДЕЙКСТРИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Будь-яке постачання ресурсів залежить від того, наскільки добре створені логістичні маршрути, адже від цього залежить, як швидко та надійно вони прибудуть до місця призначення. Особлива ця тема є важливою, коли йдеться про постачання військової провізії, тому що від цього залежить життя та здоров'я солдатів. Тема оптимізації маршрутів постачання військової провізії стала дуже актуальною у наш час, коли почалася повномасштабна війна з боку Росії.

Одним із рішень вирішення цієї проблеми є застосування алгоритму Дейкстри під час розроблення маршруту. Суть цього алгоритму полягає у тому, щоб знайти шлях у зваженому графі, коли загальна сума ваги ребер графу буде мінімальною. Саме у такий спосіб можна буде знайти оптимальні маршрути для постачання військової провізії.

Алгоритм Дейкстри має такі стадії:

- ініціалізація – це стадія, на якій встановлюється відстань до початкового вузла, яка дорівнює нулю, а до інших – нескінченності, також під час ініціалізації створюється список точок, які ще не були відвідані;
- вибір початкового вузла – це стадія, на якій обирається вузол, який є відправним (початковим) у роботі алгоритму;
- оновлення найкоротших відстаней – це стадія, на якій алгоритм оновлює відстані до сусідніх невідвіданих вузлів та оновлює її за умови, якщо вона є найкоротшою;
- обрання наступного вузла – стадія, на якій алгоритм обирає наступний вузол, який не є відвіданим та у нього найменша вага;
- повторення – ця стадія алгоритму є найдовшою, оскільки вона відбувається до того моменту, поки не буде знайдений найменший маршрут, який задовольняє умову задачі [1].

Як можемо побачити, важливим елементом алгоритму Дейкстри у пошуку оптимального маршруту є вага ребра графу. Оскільки нашою метою є знайти не просто оптимальний маршрут поставки звичайної провізії, а саме військової провізії, то для позначення ваги ребер графу треба буде застосовувати декілька значень. Така можливість стає доступною, якщо використовувати зважений мультиграф [2].

Зважений мультиграф – це зважений граф, у якому вага ребра складається з декількох компонентів. У нашому випадку їх наявність обумовлена тим, що, окрім **відстані,** є ще різні фактори, які ускладнюють поставку військової провізії. Ними є:

- небезпека: потрібно розрахувати, наскільки маршрут є небезпечним, адже по ньому можуть відкрити вогонь або місцевість може бути замінована;
- стан дороги: через масові руйнування якість дороги значно знизилася – з'явилися різні ями та перешкоди, тому потрібно обрати не лише короткий, а й легкий у просуванні маршрут;
- терміновість: можна сказати, що це є найважливішим фактором, але він не буде використовуватися в кожній задачі. Оскільки фактор терміновості буде застосовуватися, коли припаси потрібно доставити не в одне місце, а в декілька, то потрібно обрати, куди саме потрібно доставити допомогу першочергово.

Після визначення усіх компонентів, які становлять вагу графу, потрібно створити формулу, яка буде обчислювати цю вагу. Ця формула потрібна, оскільки важливість компонентів різна, і неможливо просто зіставити їх значення без додавання коефіцієнтів у формулу [3].

Вважаємо, що коефіцієнти мають бути пропорційні тому, наскільки є важливим той чи інший компонент ваги ребра під час вибору оптимального маршруту. Оскільки компонент відстані є сталим і не зможе змінитися через деякий час, він повинен мати найменший коефіцієнт – 0.01. Далі йде стан дороги, оскільки його іноді можна нівелювати підбором вдалого транспорту, але наявний фактор зміни через певний проміжок часу, тому цей компонент повинен мати коефіцієнт – 0.1. Небезпека є основним фактором, від якого залежить життя людей, які везуть ресурси, а також компонентом з найбільшою мінливістю, оскільки він може змінюватися щохвилини. Звідси випливає, що цей компонент повинен мати найбільший коефіцієнт – 1. Останній компонент ваги ребра, який буде застосовуватися не так часто, – це терміновість. Оскільки вона дуже сильно впливає на вибір маршруту, то в неї має бути коефіцієнт 0.5. Зазвичай коефіцієнт не може бути більший, ніж у небезпеки, оскільки незважаючи на терміновість, не можна нехтувати життям людей. Тому формула ваги ребра (1) буде такою:

$$\omega = 0.1 \cdot x_1 + 0.1 \cdot x_2 + 1 \cdot x_3 + 0.5 \cdot x_4, \quad (1)$$

де ω – вага ребра;

x_1 – відстань;

x_2 – стан дороги;

x_3 – небезпека;

x_4 – терміновість.

Але для того, щоб ця формула нормально функціонувала, потрібно вказати, в який спосіб задавати змінні. Причиною цього є те, що деякі змінні задаються реальними числами, а деякі за певною шкалою:

- відстань – оскільки це реальна величина, то вона записується такою, як вона є;
- стан дороги – оцінюється за шкалою від 1 до 10, де 1 – гарний стан дороги, а 10 – поганий;
- небезпека – оцінюється за шкалою від 1 до 10, де 1 – це повна безпека на дорозі, а 10 – велика небезпека (беруться до уваги тільки фактори небезпеки, які створені ворогом);
- терміновість – оцінюється за шкалою від 1 до 10, де 1 – надзвичайно терміново, а 10 – не терміново (для задач, де цей показник не потрібний, то його не використовують або, інакше кажучи, терміновість = 0).

Підсумовуючи, зазначимо, що саме за таким принципом можна знаходити оптимальні маршрути для постачання військової провізії, оскільки тут враховуються різні фактори, що впливають на швидкість подолання заданої відстані. Звичайно, за потреби можна збільшити кількість компонентів у вазі ребра графу, що допоможуть ще ефективніше встановлювати потрібні нам маршрути. Також потрібно вказати те, що цей принцип можна застосовувати не тільки в простих програмах для знаходження оптимального маршруту, а також визначати оптимальні маршрути в режимі реального часу, якщо використовувати різні технології. Саме так можна максимально розкрити потенціал алгоритму Дейкстри.

Список використаних джерел

1. Алгоритм Дейкстри. URL: https://ua5.org/algorithm/1970-algorytm-dejkstry.html

2. Дослідження операцій. Частина 2. Алгоритми оптимізації на графах / М. Я. Бартіш, І. М. Дудзяний. Львів: Видавничий центр ЛНУ імені Івана Франка, 2007.120 с.

3. Слюсар В., Громлюк К. Удосконалений метод Дейкстри для визначення найкоротших маршрутів між вузлами зв’язку у системі військового зв’язку. *Сучасні інформаційні технології у сфері безпеки та оборони.* 2023. № 46. С. 5–12.

УДК 004.9

Щербина Д. С., студент 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ПОНЯТТЯ ПРО ГЕНЕТИЧНІ АЛГОРИТМИ ТА ЇХ ЗАСТОСУВАННЯ В ЗАДАЧАХ ОПТИМІЗАЦІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

Генетичні алгоритми (ГА) – це клас оптимізаційних методів, які базуються на принципах еволюції та природного відбору. Їх застосовують у різних галузях, як-от інженерія, економіка, біологія та ін. Через тісний зв'язок із еволюцією для опису принципів ГА використовуються біологічні терміни [1]:

- особина – потенційний розв'язок задачі;
- популяція – деякий набір особин;
- нащадок – краща версія однієї з батьківських особин;
- хромосома – масив закодованої інформації про особину;
- ген – елемент масиву [1].

Переваги ГА дають їм змогу бути актуальними у сучасному світі. Ці переваги можна розділити на дві частини, а саме: перша – це те, що генетичні алгоритми можуть швидко та ефективно обробляти великі обсяги даних, що є перевагою, оскільки кількість інформації щороку збільшується. Другою є те, що ГА мають високий рівень адаптивності, що дає змогу пристосовуватися до нових умов без необхідності застосування змін. Також великий рівень адаптивності показує те, що ці алгоритми можна ефективно застосовувати в різних сферах, що додає їм ще декілька балів переваг.

Ключовою частиною кожного алгоритму є його оператори. У ГА такими є генетичні оператори. Вони визначають, як інформація передається від одного покоління особин до іншого, роблячи в такий спосіб генетичні алгоритми подібними до природного відбору та еволюції. Основними генетичними операторами є селекція, кросинговер та мутація.

Використання оператора селекції дає змогу обирати декілька найкращих особин на кожній ітерації алгоритму. Процес відбору особин відбувається різними способами, розглянемо декілька найпопулярніших [2]:

- елітний відбір: обираються найкращі особини з допомогою зрівняння зі значеннями цільової функції;
- на основі рулетки: кожній особині виділяється ділянка на рулетці, яка пропорційна її значенню цільової функції;

➢ турнірна селекція: вибирається або випадкова, або задана кількість особин, які відповідно до типу турніру змагаються за перемогу в ньому. Цей метод став основним, оскільки ймовірність виграшу найсильнішої особини близька до 1;

➢ за заданою шкалою: вибір особин проводиться за заданими критеріями селекції.

Наступним оператором є кросинговер. Він використовується для створення нової популяції з допомогою схрещення батьківських особин. Схрещення відбувається шляхом поділу в певних точках, кількість та вибір залежить від виду кросинговеру: рівномірний, одно- та двоточковий.

Останнім генетичним оператором є мутація. Він використовується рідко, але відіграє важливу роль, оскільки з допомогою того, що мутація вносить випадкові зміни в ген однієї або декількох хромосом, збільшується різновид популяції, що призводить до її розширення. Також кількість мутацій не може перевищувати 10 %, оскільки це може призвести до суттєвої різниці між генами батьків та нащадків [2].

Як можемо побачити, існує багато способів застосувати ГА, що свідчить про їх гарну адаптивність. Спочатку ГА у більшості випадків застосовувалися для створення штучного інтелекту, але чим більше їх вивчали, тим більше їх використовували в різних оптимізаційних задачах [3]:

➢ екстремальні задачі: пошук мінімуму та максимуму став набагато простішим завдяки використанню ГА;

➢ вирішення задач на графах: із допомогою ГА можна добирати та комбінувати різні варіації параметрів для вирішення задач на графах;

➢ навчання та налаштування штучної нейронної мережі: ГА допомагають вдало працювати з нейронними мережами на основі тваринних нейронних мереж.

Серед великої кількості різних оптимізаційних задач, які можуть вирішуватися з допомогою генетичних алгоритмів, останні часто використовують в різних екстремальних задачах з елементами комбінаторики, оскільки ГА можуть добре оптимізувати пошук оптимальних рішень, коли є велика кількість можливих комбінацій. Найчастіше це є такі класичні задачі:

➢ задача комівояжера: знаходження найкоротшого маршруту для мінімізації витрат часу;

➢ задача рюкзака: обрати предмети за заданою вартістю та вагою для того, щоб мінімізувати загальну вартість;

➢ задача розміщення: знаходження оптимального порядку виконання певного завдання для того, щоб максимізувати ефективність або мінімізувати витрату часу;

➢ задача розкладу: знаходження оптимального розкладу виконання певних завдань для максимізації ефективності за умови урахування всіх обмежень і цільових функцій.

Підсумовуючи, можна дійти висновку, що генетичні алгоритми використовуються для розв'язання багатьох оптимізаційних задач, оскільки можуть добре пристосовуватися до різних умов, водночас не втрачаючи ні якість, ні швидкість прийняття рішень. Саме через це ГА і далі будуть актуальними, незважаючи на стрімкий розвиток технологій.

Список використаних джерел

1. Мороз О. Г. Аналіз застосування генетичних алгоритмів в задачах глобальної оптимізації. *Control systems and computers.* 2018. № 2. С. 68–79.

2. Гриник Р. О. Застосування генетичного алгоритму для вирішення задач криптоаналізу. *Інформаційно-комунікаційні технології в сучасній освіті: досвід, проблеми, перспективи*: матеріали IV Міжнародної науково-практичної конференції. 21–22 жовт. 2015 р.: зб. наук. пр. Ч. 1. Львів, Вид-во ЛДУ БЖД, 2015. С. 168–170.

3. Мельник А. М., Босько В. В., Резніченко В. А. Дослідження сучасних методів роботи та напрямків застосування генетичних алгоритмів. *Інформаційні технології в економіці, медицині та освіті*: матеріали VI Міжнародної науково-практичної конференції (м. Кропивницький, 20–21 квітня 2023 р.). Кропивницький, 2023. С. 75–76.

4. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів СО «Бакалавр» денної та заочної форм навчання спеціальностей 122 «Комп'ютерні науки», 113 «Прикладна математика». Вінниця: ДонНУ імені Василя Стуса. 2020, 104 с.

УДК 004.4:004.021

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

МЕТОДИ ОПТИМІЗАЦІЇ АЛГОРИТМІВ У ПРОГРАМУВАННІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Зі стрімким розвитком інформаційних технологій у світі майже кожна сфера нашого життя стала напряму пов'язана з простими або складними інформаційними системами, які виконують ту чи іншу дію. Зокрема, інформаційні системи побудовані з використанням різних алгоритмів, які і підтримують роботу таких систем. Оптимізація алгоритмів у програмуванні відіграє ключову роль

у покращенні ефективності програм та забезпеченні швидкодії і низької обчислювальної складності. Алгоритми застосовуються в різних сферах, від програмування та обчислювальної техніки до математики, науки про дані, логістики, телекомунікацій та інших галузей.

Алгоритм – набір дій та процедур, які потрібні для виконання певного завдання. Ці дії мають певну послідовність, деякі з них можуть повторюватись певну кількість разів для досягнення завдання цього алгоритму [1].

Основна мета алгоритму – розробити такий набір кроків, який буде чітко визначений та може бути виконаний механічно без будь-яких уявлень про специфічні обставини. Він може бути використаний для розв'язання певної задачі у багатьох різних умовах.

Головною характеристикою алгоритму для його оптимізації є складність алгоритму. Складність алгоритму – міра, яка визначає, наскільки затратний алгоритм під час виконання всіх дій алгоритму.

Розглянемо основні аспекти складності алгоритму:

- часова складність – вимірює кількість операцій або час, необхідний для виконання алгоритму у відношенні до розміру вхідних даних. Часова складність вказує, як швидко збільшується час виконання алгоритму під час збільшення об'єму вхідних даних;
- просторова складність – вимірює кількість пам'яті, яка використовується алгоритмом у відношенні до розміру вхідних даних. Просторова складність вказує на об'єм пам'яті, який алгоритм потребує для своєї роботи;
- «O» (O-нотація) – використовується для оцінки верхньої межі часової або просторової складності алгоритму у відношенні до розміру вхідних даних. У термінах O-нотації O(f(n)) визначає, як змінюється часова або просторова складність алгоритму зі збільшенням розміру вхідних даних (n).

Розглянемо основні методи оптимізації алгоритмів у програмуванні.

- Аналіз алгоритму – один із основних методів оптимізації алгоритмів. Оцінка часової та просторової складності дає змогу визначити найбільш обтяжливі частини коду. Змінивши ці частини коду або вилучивши непотрібні дії, можна значно знизити складність алгоритмів.
- Використання ефективних структур даних є ключовим для оптимізації алгоритмів. Застосування оптимальних масивів, списків, черг або дерев дає змогу значно зменшити час доступу та операцій із даними.
- Оптимізація циклів та рекурсії. Уникнення зайвих ітерацій у циклах, використання оптимізованих ітераційних конструкцій (як-от індексація замість пошуку) допомагає покращити швидкість виконання. Також оптимізація рекурсивних викликів (наприклад, застосування хвостової рекурсії) зменшує використання пам'яті та збільшує ефективність алгоритму.

➢ Паралельна обробка та асинхронність дає змогу використовувати ресурси більш ефективно. Використання багатопотоковості або асинхронних функцій може значно зменшити час виконання завдань.

➢ Оптимізація самого алгоритму часто є ключем до покращення продуктивності. Від вибору правильного алгоритму для конкретної задачі до оптимізації його роботи важливою стає глибока розробка самого алгоритму.

Зважаючи на постійний розвиток технологій та зростання обсягів даних, оптимізація алгоритмів стає надзвичайно важливою для розробників програмного забезпечення. Прагнення до швидкодії, оптимального використання ресурсів та забезпечення мінімальних обчислювальних витрат визначає успіх багатьох сучасних програм.

Застосування методів оптимізації, що були розглянуті, дає змогу покращити ефективність програм та алгоритмів. Використання ефективних структур даних, оптимізація циклів і рекурсії, а також паралельна обробка дають змогу створювати програмне забезпечення, яке працює швидко та ефективно за будь-яких умов.

Список використаних джерел

1. Алгоритм. URL: https://ua5.org/osnprog/187-algoritm.html (дата звернення: 12.11.2023).

2. Algorithms optimization. URL: https://d2l.ai/chapter_optimization/ (дата звернення: 13.11.2023).

3. Оптимізація алгоритмів. URL: https://dou.ua/forums/topic/39967/ (дата звернення: 12.11.2023).

СЕКЦІЯ 2
ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ ТА МЕРЕЖІ

УДК 519.2

Вербіцький Р. Р., студент групи СНм-61,
Гарматій Н. М., канд. екон. наук, доцент кафедри економічної кібернетики

ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ ЗМІНИ КУРСУ АНГЛІЙСЬКОГО ФУНТА СТЕРЛІНГІВ ЩОДО ГРИВНІ МЕТОДОМ «БШ»

Тернопільський національний технічний університет
імені Івана Пулюя, м. Тернопіль

Імітаційне моделювання та комп'ютерний експеримент є методами, що дають змогу не лише моделювати вхідні потоки, але й вивчати різні аспекти систем, як-от тривалість обслуговування замовлень, цінові показники обслуговування, а в кінцевому підсумку функціонування систем масового обслуговування (СМО) загалом. Відрізняючи імітаційне моделювання від комп'ютерного експерименту, варто зазначити, що комп'ютерний експеримент є більш загальним терміном, який вказує на використання комп'ютерних технологій для вивчення різних явищ. Натомість імітаційне моделювання є конкретним методом дослідження, який базується на аналізі системи з допомогою її імітатора [1].

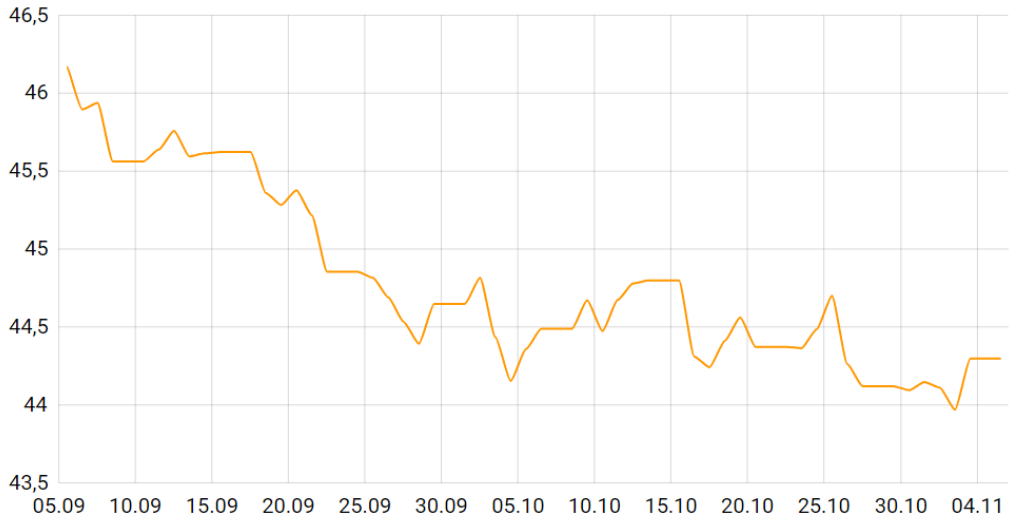


Рисунок 1. Курс англійського фунта стерлінгів до гривні за період 4 кварталу 2023 (5 вересня – 5 листопада)

Білий шум з дискретним аргументом, або дискретний білий шум, – це послідовність випадкових величин, де будь-які дві величини є незалежними. Термін «базовий білий шум» застосовується до послідовності випадкових величин з рівномірним розподілом на інтервалі від 0 до 1. Основні алгоритми моделювання базового білого шуму передбачають метод лишків, ідея якого полягає в

утворенні базової послідовності з допомогою рекурентного співвідношення з допоміжної послідовності цілих чисел.

Для визначення курсу англійського фунта стерлінгів до гривні за четвертий квартал 2023 р. (в період із 5 вересня до 5 листопада) буде використано метод моделювання, а конкретно – метод лишків [2].

Перший етап передбачає проведення імітаційного моделювання базового білого шуму та нормування отриманих даних для визначеного курсу валют. Важливо враховувати, що моделювання проводиться в розрізі днів. Це означає, що для отримання результатів за четвертий квартал 2023 р. (з 5 вересня до 5 листопада) ми отримаємо вибірку, яка містить 91 значення.

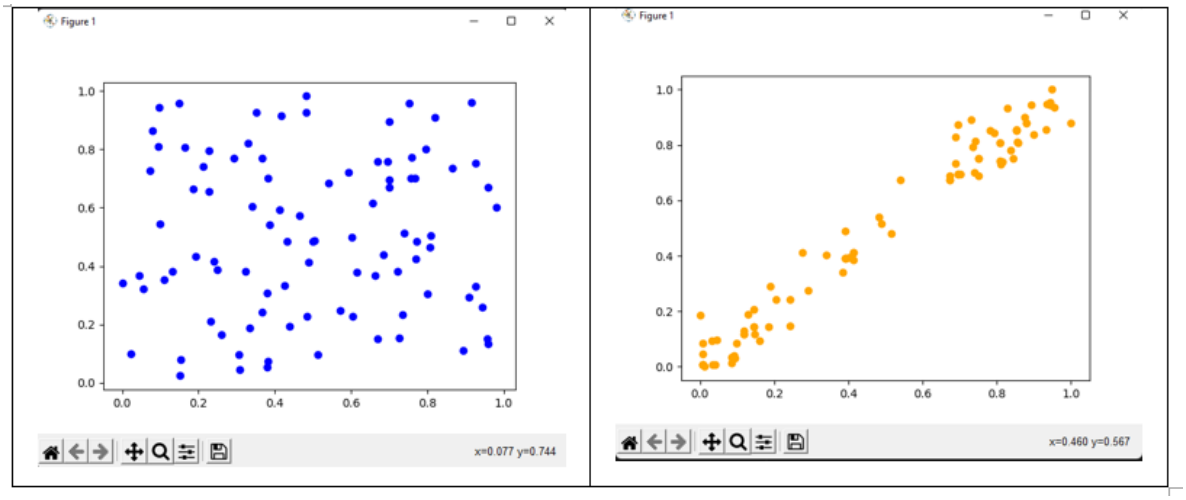


Рисунок 2. Графік 91 значення ББШ та 91 значення вибірки

Отже, після отримання вибірки даних ми можемо провести аналіз і визначити значення математичного сподівання та середнього квадратичного відхилення як для базового білого шуму, так і для нормованої вибірки. Для цього також потрібно вибрати об’єм n базового білого шуму. Для порівняння ми виберемо значення об’єму вибірки 100, 1 000 та 10 000.

Таблиця 1 – Значення математичного сподівання та середнього квадратичного відхилення для ББШ і нормованої вибірки

Об’єм	**Математичне сподівання**	**Середнє квадратичне відхилення**
100	0,49661	0,27687
1 000	0,50559	0,28495
10 000	0,50420	0,28777

Отримані дані показують, що математичне сподівання наближується до 0,504, водночас середнє квадратичне відхилення наближується до 0,287, тобто до стандартних значень ББШ.

У межах прогнозів щодо майбутнього фунта стерлінгів на міжнародному валютному ринку очікується, що валюта збереже високий рівень стабільності. Цей висновок ґрунтується на комплексному аналізі експертами різноманітних аспектів, які впливають на вартість і курс фунта.

Економічні показники Великої Британії, як-от рівень економічного зростання, зайнятість та інфляція, вважаються ключовими факторами, які сприятимуть стабільності фунта стерлінгів. Прогнозується, що позитивні динаміки у цих сферах сприятимуть збереженню високої довіри та попиту на фунт серед міжнародних інвесторів.

До того ж враховуючи геополітичний контекст, стабільність у внутрішній політиці та міжнародних відносинах також визначатиме поведінку фунта. Відсутність значимих глобальних турбуленцій та конфліктів сприятиме утриманню валюти на рівні високої стабільності.

Такий прогноз ґрунтується на ретельному аналізі фундаментальних чинників, що впливають на економіку та міжнародний фінансовий ринок, і передбачає позитивний сценарій для фунта стерлінгів у майбутньому.

Список використаних джерел

1. Гарматій Н. М., Мартиняк І. О., Ціх Г. В. Класичні та сучасні моделі економіки: навч. посіб. Вид-во: ФОП Паляниця В. А., 2023. 300 с.

2. Звіт НБУ щодо курсу англійського фунту стерлінгів до гривні за 4 квартал 2023. URL: https://bank.gov.ua/ua/markets/exchangeratechart?cn%5B%5D=BGN&startDate=01.07.2023&endDate=30.09.2023

УДК 004.8

Кадикова А. А., здобувач 4 курсу спеціальності 121 Інженерія програмного забезпечення,

Шостак І. В., д-р техн. наук, професор, професор кафедри інженерії програмного забезпечення

ПЕРСПЕКТИВИ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДО РОЗВ'ЯЗАННЯ ЗАВДАНЬ АНАЛІЗУ РОЗМОВ ПОЛІЦЕЙСЬКИМИ СТРУКТУРАМИ

Національний аерокосмічний університет
імені М. Є. Жуковського «Харківський авіаційний інститут», м. Харків

Стрімкий розвиток технологій у XXI ст. приніс багато змін у сучасне суспільство. Новітні технології змінили спосіб спілкування та взаємодії суспільства.

Тепер без зайвих труднощів можна спілкуватися з людьми, незалежно від їхньої локації, надсилати повідомлення за лічені секунди і так само швидко інформувати своїх друзів щодо найменших змін у своєму житті. Але попри всі досягнення та позитивний внесок у нашу здатність спілкуватися також виникла цифрова криміналістика [1]. Ніколи злочинні угрупування не могли так легко контактувати між собою, як зараз. Сьогодні всього за кілька кліків потенційні злочинці можуть знайти потрібні контакти, домовитись про час та місце зустрічі, придбати нелегальні товари.

На превеликий жаль, цифрова криміналістика не встигла так стрімко розвинутися разом із новітніми технологіями. У США цифрова криміналістика зіткнулась із проблемою недостатньої стандартизації – станом на 2017 р. в усіх американських правоохоронних органах досі не було визначених стандартів щодо відповідної моделі «розслідування», підготовки цифрової криміналістичної експертизи та її методів [2, 3].

Через застарілість своїх засобів поліція не може спрогнозувати виникнення злочинних дій заздалегідь, і хоча в Україні існує кіберполіція з 2009 р. [4], нею все ще використовуються далеко не всі доступні інструменти. Навіть можна зазначити, що за цим пунктом Україна відстає від багатьох країн [5].

Тож з огляду на вищенаведене перспективи застосування штучного інтелекту до розв'язання завдань аналізу розмов поліцейськими структурами є високі як в Україні, так і загалом у світі.

Для опису процесів такого використання розроблено модель у нотації BPMN, яка розкриває логіку функціонування автоматизованих процесів (рис. 1).

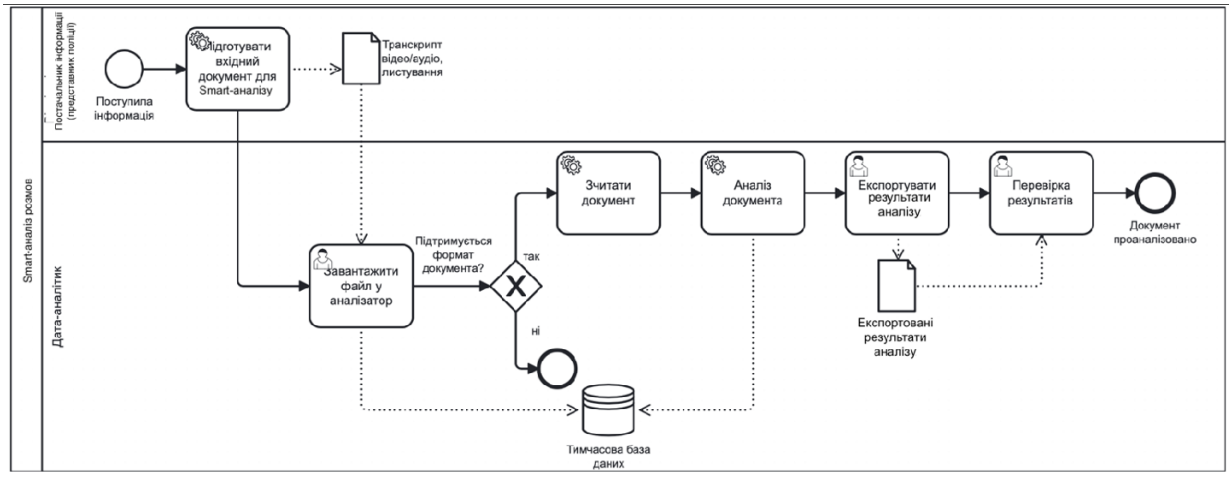


Рисунок 1. Розроблена процесна модель у нотації BPMN

Наведена процесна модель передбачає отримання вторинних вхідних даних із відділу поліції. Це зумовлено тим, що подібна система буде інтегруватись у наявну інформаційну систему поліцейських структур.

Для відображення взаємозв'язків між об'єктами і класами побудована діаграма класів для реалізації майбутньої системи Smart-аналізу розмов із допомогою framework Django. Для розробки може бути обраний framework Django, який надаватиме два користувацькі інтерфейси: один призначений суто для користувачів, а інший – суто для адміністрування. Їх роботу забезпечуватимуть ключові компоненти framework Django (рис. 2).

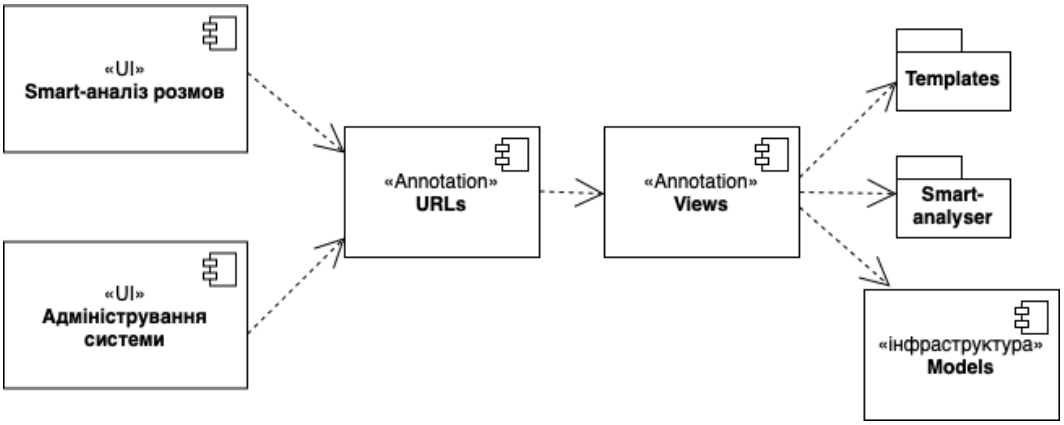


Рисунок 2. Діаграма класів для системи Smart-аналізу розмов

Компонент URLs буде визначати, які конкретні представлення (views) та функції будуть необхідні для обробки запитів користувацьких інтерфейсів, після чого вони передаватимуть необхідну інформацію компоненту представлень, які оброблятимуть логіку відображення даних та відповідатимуть на запити користувача (рис. 3).

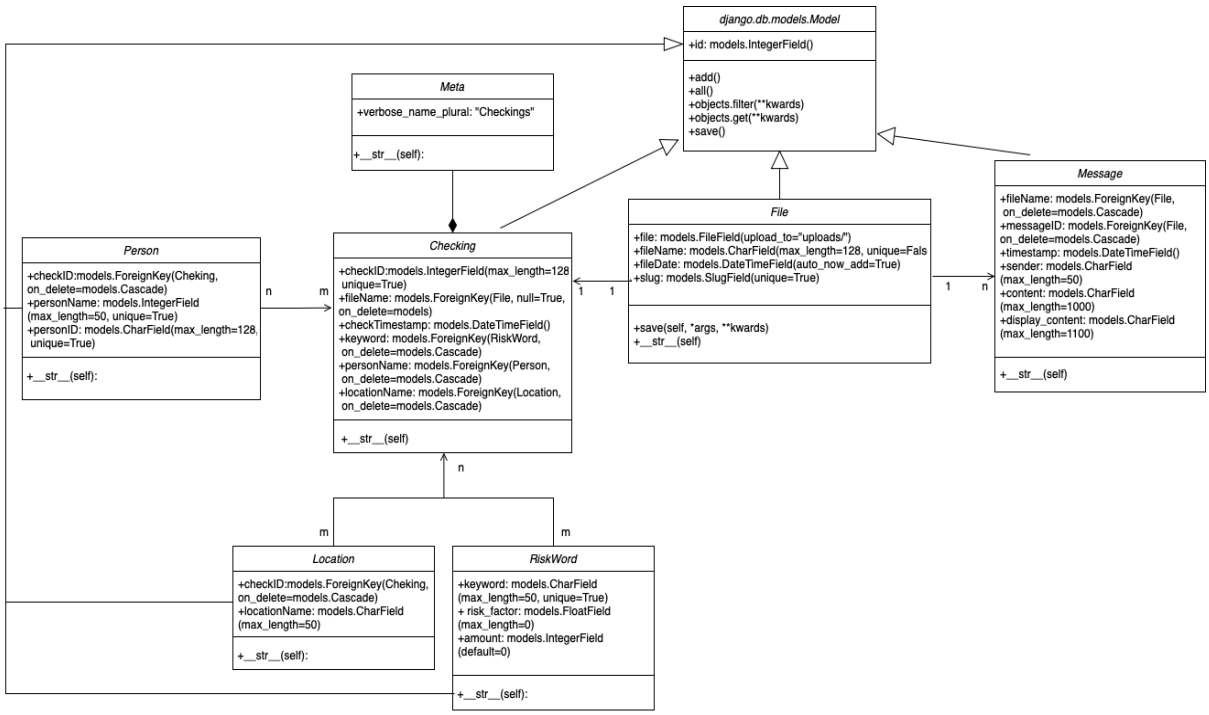


Рисунок 3. Діаграма компонентів для системи Smart-аналізу розмов

Компонент представлень передаватиме виконання роботи за необхідності до компоненту шаблонів («Templates»), який визначатиме вигляд сторінок, компонента «Smart-analyser» загальної функціональності проєкту та компонента «Models», що представлятиме розроблені класи в структурі бази даних.

Список використаних джерел

1. Tymoshenko, Y. P., Kozachenko, O. I., Kyslenko, D. P., Horodetska, M. S., Chubata, M. V., Barhan, S. S. (2022). Latest technologies in criminal investigation (testing of foreign practices in Ukraine). *Amazonia Investiga, 11(51)*, 149–160.

2. Keeling, D. G., Losavio, M. (2017). Public Security & Digital Forensics in the United States: The Continued Need for Expanded Digital Systems for Security. *Journal of Digital Forensics, Security and Law*, 9, 51.

3. Aleksandrov, D., Okhrimenko, I., Drozd, O. (2017). Psychological adaptation of Ukrainian National Police officers for law enforcement activities. *Science and Education,* 11, Ukraine. DOI: 10.24195/2414- 4665-2017-11-4.

4. Kryvolapchuk, V., Kulyk, O., Barko, V., Kalynovskyi, B., Kosiak, N. (2020). Attitude of young people to the criminality problem in Ukrainian postmodern society, *Postmodern Openings,* 11(1Supl1), Romania. DOI: 10.18662/po/11.1sup1/125.

5. Fedorenko, O., Dotsenko, V., Okhrimenko, I., Radchenko, K., Gorbenko, D. (2020). Coping behavior of criminal police officers at different stages of professional activity, *BRAIN. Broad Research in Artificial Intelligence and Neuroscience*, 11(2), Romania. DOI: 10.18662/brain/11.2/78.

УДК 004.21:004.8

Богач Т. О., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ОСОБЛИВОСТІ ВИКОРИСТАННЯ АЛГОРИТМІВ ОПТИМІЗАЦІЇ В ЗАДАЧАХ ШТУЧНОГО ІНТЕЛЕКТУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Штучний інтелект (ШІ) та машинне навчання (МН) зазнали значного розвитку, визначаючи нові перспективи для автоматизації та вдосконалення великої кількості завдань. Важливим складником цих технологій є алгоритми оптимізації, які визначають ефективність та точність моделей.

Оптимізація – ключовий етап у розв'язанні задач машинного навчання. Її мета – знайти найкращі параметри моделі, які мінімізують функцію втрат або

максимізують точність прогнозування. Для досягнення цієї мети використовуються різні алгоритми оптимізації. Водночас виникає потреба в постійному вдосконаленні й адаптації алгоритмів оптимізації до нових завдань та обмежень. Класичними алгоритмами оптимізації є:

1. Градієнтний спуск: один із найпоширеніших методів оптимізації. Він використовує похідні функції втрат для оновлення параметрів моделі. Градієнтний спуск може бути варіантом стохастичного чи мініпакетного підходу.

2. Метод Ньютона: цей метод використовує другі похідні функції втрат для знаходження оптимальних параметрів. Він збільшує швидкість збіжності, але може бути обчислювально витратним.

3. Метод симплексу (Nelder-Mead): використовується для нелінійної оптимізації. Він працює з обмеженими областями параметрів і знаходить локальний мінімум.

Більшість задач оптимізації в середовищі використання штучного інтелекту використовують нейронні мережі. Нейронні мережі – це обчислювальні структури, які обробляють вхідні сигнали за принципом процесів, що відбуваються в нейронах живих істот. Саме через це нейронні мережі часто порівнюють із мозком людини. Кожна ланка в мережі може обробляти інформацію паралельно з іншими, що допомагає значно прискорити цей процес і уникнути великої кількості помилок.

Порівняльний аналіз алгоритмів дав змогу виявити їх особливості щодо використання.

1. Градієнтний спуск (Gradient Descent):

➢ опис: градієнтний спуск є одним із найпоширеніших методів оптимізації. Він використовує градієнт функції втрат для оновлення параметрів моделі в напрямі, що мінімізує функцію втрат;

➢ використання: ідеально підходить для навчання моделей машинного навчання, зокрема нейронних мереж;

➢ рекомендація: використовуйте градієнтний спуск для навчання моделей, особливо за великої кількості даних.

2. Метод симплексу (Nelder-Mead):

➢ опис: цей метод є ітераційним алгоритмом оптимізації без похідних. Він використовує геометричні перетворення для знаходження мінімуму функції;

➢ використання: підходить для нелінійних задач оптимізації;

➢ рекомендація: використовуйте метод симплексу, коли немає можливості обчислення похідних.

3. Трансформери (Transformers):

➢ опис: трансформери – це архітектура нейронних мереж, яка стала популярною в області обробки природних мов та комп'ютерного бачення. Вони використовують механізм самоуваги для ефективної обробки послідовностей;

➢ використання: добре підходять для завдань NLP та синтезу зображень;

➢ рекомендація: вивчайте та використовуйте трансформери для послідовнісних завдань.

4. Генеративні змагальні мережі (GAN):

➢ опис: GAN – це архітектура, яка використовує дві нейронні мережі – генератор і дискримінатор – для генерації нових даних, як-от зображення, текст чи звук;

➢ використання: для генерації реалістичних даних;

➢ рекомендація: вивчайте GAN для завдань синтезу даних [3].

Можна зробити висновок, що дослідження алгоритмів оптимізації для ШІ та МН є важливим напрямом у розвитку цих технологій. Вони не лише визначають ефективність моделей, але і створюють можливості для нових досягнень у галузі штучного інтелекту та машинного навчання. Це поле відкриває нові перспективи для автоматизації та розуміння складних завдань, забезпечуючи стабільний розвиток цих інноваційних технологій.

Список використаних джерел

1. 10 найкращих алгоритмів машинного навчання. URL: https://www.unite.ai/uk/десять-найкращих-алгоритмів-машинного-навчання/

2. Машинне навчання: як штучний інтелект вчиться і розвивається. URL: https://bizmag.com.ua/mashynne-navchannya/

3. Що таке GAN-генеративно-змагальні нейронні мережі і як їх застосувати для генерації зображень. URL: https://evergreens.com.ua/ua/articles/gan.html

УДК 004.89

Журовський Я. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

МАШИННЕ НАВЧАННЯ В УПРАВЛІННІ ТРАНСПОРТНИМИ ПОТОКАМИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Зважаючи на постійне збільшення населення та розвиток інфраструктури, міста стикаються зі складнощами управління транспортними потоками. Використання машинного навчання стає ключовим інструментом у вирішенні цих викликів, сприяючи оптимізації руху транспорту та покращенню мобільності у містах. Сучасні технології машинного навчання дають змогу аналізувати транспортні потоки та ефективно передбачати їх рух, використовуючи величезні обсяги даних.

Моделі машинного навчання, зокрема нейронні мережі, відкривають можливості для прогнозування та управління транспортними потоками. Їх використання допомагає ефективно керувати рухом, враховуючи різноманітні фактори, як-от: погодні умови чи попередні перевезення.

Однією з ключових переваг є здатність машинного навчання адаптуватися до непередбачуваних сценаріїв, умов руху, які змінюються, та реагувати на реальний час, що дає змогу виконувати точні та стратегічно обґрунтовані рішення.

Машинне навчання використовується для створення систем, які можуть аналізувати великі обсяги даних та прогнозувати зміни у транспортних потоках. Це дає змогу не лише прогнозувати затори, а й запропонувати оптимальні маршрути, що сприяє зменшенню часу подорожі та зниженню викидів в атмосферу.

Один із алгоритмів, який може бути використаний для підвищення ефективності трафіка та маршрутизації, – це алгоритм «Random Forest» (випадковий ліс). Він використовується для прогнозування трафіка, аналізуючи велику кількість факторів, як-от: час доби, день тижня, погодні умови, події на дорозі тощо. Цей алгоритм працює шляхом об'єднання рішень багатьох дерев прийняття рішень. Він будує багато дерев, кожне з яких вирішує проблему передбачення трафіка на основі різних факторів. Усі ці дерева потім об'єднуються, щоб отримати остаточне передбачення трафіка. Такий підхід дає змогу адаптуватися до змін у потоці даних та враховувати різні впливи на трафік для підтримки оптимальної маршрутизації.

Алгоритми машинного навчання впроваджуються у системи керування світлофорами, що дає змогу їм реагувати на зміни в потоці транспорту в реальному часі. Це сприяє оптимізації часу очікування на світлофорі та зниження кількості заторів.

Один із прикладів алгоритму машинного навчання, який може бути використаний для адаптивного керування світлофорами, – це Reinforcement Learning (навчання з підкріпленням). Такий алгоритм дає змогу системі навчатися, спостерігаючи результати своїх дій і отримуючи відповідні винагороди або покарання за кожен крок. Система керування світлофорами, яка використовує Reinforcement Learning, може навчатися оптимальним стратегіям управління світлофорами на перехрестях. Наприклад, вона може навчатися змінювати тривалість зеленого світла залежно від інтенсивності транспортного потоку, який вимагає переходу. Алгоритм буде спостерігати результати своїх дій, які полягають у зменшенні заторів і часу очікування на світлофорі. Принцип роботи полягає у використанні навчального процесу, де система отримує відповідь на свої дії, тобто як вони впливають на трафік на перехрестях. Це допомагає системі постійно покращувати свої стратегії, оптимізуючи роботу світлофорів для забезпечення більш ефективного руху.

З допомогою машинного навчання створюються моделі передбачення руху транспорту, які допомагають аналізувати й ідентифікувати області з підвище-

ним ризиком для запобігання аварій та надзвичайних ситуацій. У цій ситуації також допоможе алгоритм випадкового лісу.

Машинне навчання у системах відеоспостереження дорожнього руху здатне вчасно виявляти небезпеку на дорозі, розпізнавати порушення правил та миттєво реагувати для попередження негативних ситуацій. Один із прикладів алгоритму машинного навчання, що забезпечує безпеку та регулювання порушень на дорогах, – це алгоритм розпізнавання образів на основі глибокого навчання, – Convolutional Neural Networks (CNN, згорткові нейронні мережі). CNN застосовується для аналізу відеопотоків із відеокамер, встановлених на дорогах. Він здатен вчасно виявляти різні небезпечні ситуації, як-от аварії, перешкоди на дорозі, порушення правил, водіїв, які не дотримуються дистанції, пішоходів, що переходять на червоне світло та інші небезпечні дії. Принцип роботи CNN полягає в тому, що він навчається розпізнавати патерни та особливості на зображеннях. Він використовується для обробки відеопотоку, виділяє ключові об'єкти (наприклад, транспортні засоби, пішоходів, світлофори) та розпізнає ризикові сценарії або порушення правил. Після розпізнавання небезпечної ситуації система може миттєво реагувати, наприклад, надсилаючи повідомлення на пульт диспетчера, активуючи сигнали попередження на дорозі або автоматично сповіщаючи відповідні служби про негайну потребу втручання.

Розвиток автономних транспортних систем залежить від машинного навчання для створення більш безпечного та надійного середовища для автономних автомобілів у міських умовах. Один із прикладів алгоритму машинного навчання, що використовується для розвитку автономних транспортних систем, – це алгоритм reinforcement learning (навчання з підкріпленням). Reinforcement learning використовується для тренування систем управління автономними автомобілями у міських умовах. Під час тренування автомобіль навчається приймати рішення в реальному часі, коригуючи свої дії на основі отриманих даних та взаємодії з оточенням. Принцип роботи полягає в тому, що автомобіль отримує вхідні дані від сенсорів (камери, радари, лідари тощо) про дорожнє середовище. Reinforcement learning допомагає автомобілю вчитися приймати рішення на основі цих даних та максимізувати визначені критерії, як-от безпека, швидкість, комфортність маршруту. Після тренування автомобіль може самостійно керувати рухом у міських умовах, враховуючи різноманітні фактори середовища і приймаючи оптимальні рішення для безпечної та ефективної подорожі.

Машинне навчання завдяки своїй здатності до постійного вдосконалення та адаптації до нових умов визначає майбутнє управління транспортними системами у містах, допомагаючи знижувати затори, покращувати безпеку дорожнього руху та забезпечувати більш ефективну і стійку мобільність у мегаполісах.

Отже, з машинним навчанням міста отримують надзвичайний інструмент управління транспортними потоками. Від прогнозування трафіка до керування світлофорами і розвитку автономних систем – його вплив охоплює всі аспекти

мобільності. Використання цих алгоритмів революціонізує рух транспорту, робить його ефективнішим і безпечнішим у мегаполісах. Розвиток цієї технології відкриває нові перспективи для міст, сприяючи покращенню інфраструктури та забезпеченню зручності для мешканців.

Список використаних джерел

1. Штучний інтелект. Теорія і застій. Луганськ: Вид-во СНУ ім. С. Даль, 2006. 242 с.

2. Шодрон Л., Мейл Н. 1-й Одерський логічний формальний аналіз концепцій: від логічного програмування до теорії зв'язування електронних артикулів в комп'ютерних та інформаційних науках. 1998. Том 3. № 13. URL: http://www.ep. Liu.se/ea/cis/1998/013/ (дата звернення: 09.04.2023).

3. Honda розробляє технологію автономного керування автомобілем зі штучним інтелектом. *Укрінформ – актуальні новини України та світу*. URL: https://www.ukrinform.ua/rubric-technology/3606901-honda-rozroblae-tehnologiu-avtonomnogo-keruvanna-avtomobilem-zi-stucnim-intelektom.html (дата звернення: 09.04.2023).

4. Algorithmen zur formalen Begriffsanalyse / Б. Гантер, Р. Вілле, К. Е. Вольф *Beitrage zur Begriffsanalyse. Mannheim Wien Zurich, BIWissenschaftsverlag*. Berlin, 1987. P. 241–254.

5. Січко Т. В., Смоктій К. В., Ткачук А. О. Прикладні аспекти розрахунку структурно-топологічних характеристик систем. *Системи та технології*. 2019. № 1(57). С. 141–153.

УДК 004.8

Журовський Я. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ДІЯЛЬНОСТІ ОРГАНІЗАЦІЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

Технологічний прогрес постійно впроваджує способи, які компанії застосовують у своїй діяльності для покращення ефективності та спрощення робочого процесу.

Використання хмарних технологій не лише дає змогу зберігати великі обсяги даних, а й стимулює зміну парадигми управління: замість локальних серверів чи баз даних компанії переходять до використання хмарних платформ,

що дає можливість працювати з даними в реальному часі та з будь-якої точки світу. Наприклад, компанія, яка використовує хмарні технології для управління своєю клієнтською базою, може забезпечити своїм працівникам можливість доступу до цих даних через хмарні рішення. Незалежно від їхнього місцезнаходження – чи це відділення у різних куточках світу, чи робочі зони поза офісом – співробітники матимуть можливість вносити зміни, переглядати актуальну інформацію та спілкуватися, що сприяє швидкому та координованому прийняттю рішень на основі актуальних даних. До того ж хмарні технології допомагають компаніям легко масштабувати свої системи. Наприклад, якщо бізнес розширюється та потребує більшої потужності обробки даних або збільшення місця для зберігання, він може легко розширити свої можливості через хмарні ресурси без необхідності встановлення нового обладнання чи інфраструктури.

Інтернет речей (IoT) поширюється в багатьох галузях, від виробництва до роздрібної торгівлі, даючи змогу підприємствам автоматизувати багато процесів. Наприклад, сенсори виробничого устаткування можуть надсилати дані про стан обладнання в реальному часі, що допомагає прогнозувати поломки та планувати технічне обслуговування, яке змінює підходи до управління обладнанням. IoT перетворює способи, як підприємства використовують дані для автоматизації та покращення ефективності. Наприклад, у виробництві сенсори, що вбудовані в обладнання, можуть постійно контролювати його стан. Це допомагає виявляти незвичайні або неправильні показники, які можуть попереджати майбутні поломки. З допомогою аналізу цих даних компанії можуть розробляти стратегії планування технічного обслуговування, запобігаючи непередбаченим зупинкам обладнання та зменшуючи витрати на ремонт. У роздрібній торгівлі IoT використовується для відстеження запасів та організації поставок. Сенсори можуть автоматично надсилати дані про рівень запасів, що дає змогу підприємствам точно прогнозувати потреби та уникати нестачі товарів на полицях. Це допомагає оптимізувати ланцюжок постачання та забезпечує покупців постійною наявністю товарів, що може покращити їхнє враження від обслуговування.

Штучний інтелект (ШІ), зокрема системи машинного навчання, забезпечують підприємствам можливість аналізувати великі обсяги даних для генерації прогнозів, рекомендацій та вирішення складних завдань. Наприклад, у сфері маркетингу ШІ використовується для персоналізації реклами та прогнозування попиту, що змінює підходи до стратегічного планування та комунікації з клієнтами. Наприклад, системи машинного навчання можуть аналізувати купівельні звички клієнтів та їхню інтернет-поведінку, щоб створювати персоналізовані рекламні кампанії. Це означає, що кожен клієнт може бачити рекламу, яка відповідає його індивідуальним інтересам та потребам, що підвищує ефективність реклами та ймовірність конверсії у продажі.

До того ж штучний інтелект використовується для прогнозування попиту на товари чи послуги. Наприклад, аналізуючи дані про купівлі, демографічні характеристики та тренди споживання, системи машинного навчання можуть передбачати, які товари будуть популярними в майбутньому. Це допомагає підприємствам ефективно планувати свої запаси, уникати нестачі чи надмірної кількості товарів та пристосовувати виробництво до очікуваного попиту.

Блокчейн – технологія, яка використовується для безпечного та надійного зберігання даних, може застосовуватись у фінансових операціях, ланцюжку постачання та навіть управлінні правами власності, що перевертає традиційні підходи до фінансів та логістики. Наприклад, у фінансовому секторі блокчейн може використовуватись для створення безпечних і швидких транзакцій без посередників. Це означає, що переказ коштів може відбуватись миттєво та безпечно, оскільки дані про транзакції зберігаються у блоках, що підтверджуються мережею користувачів, забезпечуючи високий рівень безпеки та надійності. У ланцюжку постачання блокчейн може слугувати як система, що відстежує походження продуктів від виробника до споживача. Кожен етап виробництва та постачання може бути зафіксований у блоках, що забезпечує прозорість і недоторканність інформації про товари, що може бути особливо корисним у сферах, де важлива точність і контроль якості, наприклад, у харчовій промисловості. Також були досліджені деякі ресурси, які спрощують діяльність компаній:

- LinkedIn для бізнесу: LinkedIn надає можливості не лише для знаходження талановитих фахівців, а й для встановлення бізнес-контактів, просування продуктів та послуг, а також для вивчення нових тенденцій у вашій галузі;
- Google Analytics: цей інструмент дає змогу відстежувати та аналізувати поведінку користувачів на вебсайті, що допомагає виробляти більш обґрунтовані стратегії маркетингу і розвитку продуктів;
- Trello або Asana для управління проєктами: ці платформи дають змогу створювати завдання, призначати їх та відстежувати прогрес проєктів, полегшуючи співпрацю та координацію команди;
- Canva або Adobe Spark для дизайну: якщо потрібно швидко створити графічний контент для маркетингу чи соцмереж, ці інструменти забезпечать широкі можливості для створення професійного дизайну;
- Slack або Microsoft Teams для комунікації: ці платформи надають зручні засоби комунікації та співпраці в команді, об’єднуючи обговорення, обмін файлами та спільну роботу над проєктами в одному місці.

Отже, сучасні технології мають великий вплив на організації, перетворюючи їх функції та способи управління. Вони сприяють переходу від традиційних моделей до більш гнучких, реагуючих та інноваційних форм організації роботи підприємства чи компанії. Усі ці технології допомагають організаціям бути

більш адаптивними, швидше реагувати на зміни та ефективніше використовувати ресурси, що є ключовим для досягнення конкурентної переваги в сучасному бізнес-середовищі.

Список використаних джерел

1. Вакалюк Т. А., Рантюк І. І. Організаційні структури в ІТ компаніях. Тези ІІ Всеукраїнської науково-технічної конференції *«Комп'ютерні технології: інновації, проблеми, рішення».* 2019. URL: https://conf.ztu.edu.ua/wpcontent/uploads/2019/12/148.pdf (дата звернення: 01.09.2020).

2. Георгіаді Н. Г., Вільгуцька Р. Б. Організаційна структура управління як складова системи менеджменту підприємства. *Менеджмент та підприємництво в Україні: етапи становлення і проблеми розвитку*. 2012. URL: http://ena.lp.edu.ua/bitstream/ntb/23165/1/6-33-40.pdf (дата звернення: 01.09.2020).

3. Катренко А. В., Магац Д. С. Імовірнісні та імітаційні моделі планування та управління в мультипроєктному середовищі. *Інформаційні системи та мережі*. 2014. URL: http://science.lpnu.ua/sisn/all-volumes-and-issues/volume-805-2014/imovirnisni-taimitaciyni-modeli-planuvannya-ta (дата звернення: 01.09.2020).

4. Кравченко М. О., Малишевська А. О. Особливості створення Agile-команд для підвищення інноваційної активності промислових підприємств України, Збірник тез доповідей міжнародної науково-практичної конференції *«Бізнес, інновації, менеджмент: проблеми та перспективи».* 2020. URL: http://confmanagement.kpi.ua/proc/article/view/201224/201335 (дата звернення: 01.09.2020).

5. Алексюк В. В., Січко Т. В. Дослідження особливостей програмного забезпечення для аналізу даних. *Комп'ютерні технології обробки даних: матеріали* всеукр. наук.-практ. конф., м. Вінниця, 2022. С. 249–251.

УДК 330.3

Костенко Р. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ ЗАПАСАМИ В РОЗДРІБНІЙ ТОРГІВЛІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Управління запасами в роздрібній торгівлі займає центральне місце у досягненні успіху в цій сфері. Важливість ефективного управління запасами виявляється на практиці через низку фактів та реальних прикладів із бізнесу.

В цій роботі ми розглянемо, як застосовуються методи оптимізації, зокрема модель ABC-аналізу та метод Марковіца, для покращення управління запасами в роздрібній торговельній мережі.

Розглянемо два основні методи оптимізації запасів, які знаходять своє використання в роздрібній торгівлі та мають значимий вплив на бізнес.

➢ Модель ABC-аналізу визначає товари на основі їх важливості та прибутковості. Вона класифікує товари на три категорії: «A», «B» і «C», відповідно до їх важливості. Наприклад, товари класу «A» – це найважливіші та найприбутковіші, і їх запаси повинні бути керовані уважно. Товари класу «C» – менш важливі та прибуткові, і їх запаси можуть бути менш строгими. Прикладом застосування цього методу є роздрібна мережа ресторанів McDonald's, яка використовує ABC-аналіз для оптимізації асортименту та запасів продуктів.

➢ Метод Марковіца використовується для оптимізації інвестиційного портфеля, але він також знаходить своє застосування у керуванні запасами. Цей метод допомагає визначити оптимальний баланс між різними видами запасів та постачальниками, зокрема як розподіляти ресурси між різними товарами або категоріями товарів. Прикладом використання методу Марковіца є компанія Apple, яка застосовує цей підхід для оптимізації запасів компонентів для виробництва своєї продукції.

Саме ці два методи допомагають досягнути більшої ефективності та прибутковості, а також забезпечують належний рівень обслуговування клієнтів.

Аналіз роздрібної торговельної мережі передбачає докладне дослідження поточного стану управління запасами та вивчення конкретних ситуацій, які підтверджують важливість цього процесу. Розгляньмо деякі аспекти цього аналізу, спираючись на реальні приклади та факти:

➢ Структура запасів. Розглянемо, які товари або групи товарів переважають у запасах. Наприклад, у роздрібній мережі електроніки смартфони та планшети можуть становити більшу частину запасів, оскільки це популярні товари з високим оборотом.

➢ Оберт товарів. Оцінимо, як швидко товари реалізуються. Наприклад, якщо велика мережа гіпермаркетів помітила, що товари для дітей швидко закінчуються під час шкільних канікул, це вказує на необхідність більш активного управління цими запасами в цей період.

➢ Витрати на складське утримання. Розглянемо витрати на утримання складських приміщень та персоналу. Наприклад, компанія IKEA ретельно оптимізує розмір та розташування своїх складів для зменшення витрат на утримання.

➢ Ланцюг постачання. Дослідимо ланцюг постачання, щоб визначити можливість покращення. Наприклад, компанія Unilever впровадила систему «Zero Waste to Landfill» та вдало оптимізує постачання своїх товарів.

➢ Сезонність і попит. Аналізувати сезонність та патерни попиту критично важливо. Наприклад, у сфері роздрібного продажу одягу сезонні колекції можуть вимагати більших запасів перед початком сезону.

Отже, аналіз роздрібної торговельної мережі передбачає детальне дослідження різних аспектів управління запасами та використання реальних прикладів і фактів для підтвердження важливості цього процесу.

На цьому етапі ми впровадимо методи оптимізації, як-от модель ABC-аналізу та метод Марковіца, для активного покращення управління запасами в роздрібній торговельній мережі. Розглянемо конкретні приклади і факти, що демонструють успішне використання цих методів у практиці:

➢ Модель ABC-аналізу. Прикладом застосування цього методу є компанія Walmart, яка використовує його для класифікації своїх товарів. Вони визначають товари «А», які становлять велику частину обороту та прибутковості, і ретельно керують ними. Товари «С», менш важливі та прибуткові, мають менше уваги. Це дає змогу Walmart оптимізувати запаси та витрати.

➢ Метод Марковіца. Amazon – приклад компанії, яка успішно використовує метод Марковіца для оптимізації запасів і ланцюга постачання. Завдяки аналізу ризиків та кореляцій між різними активами. Amazon знаходить оптимальні співвідношення між різними видами запасів та постачальниками, що дає змогу їм забезпечувати більш швидку та ефективну поставку товарів своїм клієнтам.

➢ Аналіз результатів. Коли методи оптимізації впроваджуються, результати можуть бути вражаючими. Наприклад, після впровадження моделі ABC-аналізу компанія Procter & Gamble знизила свої запаси на 30 % та зекономила мільйони доларів на утриманні запасів.

➢ Порівняння зі станом до оптимізації. Порівняльний аналіз стану запасів та результатів після впровадження оптимізації допомагає визначити ефективність застосованих методів. Наприклад, після застосування методу Марковіца компанія Ford досягла покращення в ефективності ланцюга постачання та зменшенні ризиків.

Застосування методів оптимізації управління запасами дає змогу підприємствам досягати значних переваг у вигляді ефективнішого управління запасами, зменшення витрат та поліпшення якості обслуговування клієнтів. Реальні приклади успішного впровадження цих методів свідчать про їх важливість та ефективність.

Отже, результати свідчать про те, що оптимізація управління запасами в роздрібній торговельній мережі є важливим фактором успіху бізнесу. Вона дає змогу знижувати витрати, збільшувати прибутковість та забезпечувати належне обслуговування клієнтів, що робить цей процес незамінним для сучасних компаній.

Список використаних джерел

1. Оптимізація управління запасами в багаторівневому ланцюгу постачань. URL: https://abmcloud.com/uk/optimizatsiya-upravlinnya-zapasami-v-bagatorivnevomu-lantsyugu/ (дата звернення: 30.11.2023).

2. Nahmias S. Analysis of production and operations, 6ª edn. McGraw W-Hill / InterAmerican Editors, S. A. de C. V., Mexico. 2014. P. 205.

УДК 004.01:005

Куцмай В. Я., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Волонтир Л. О., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ ПРОЦЕСІВ УПРАВЛІННЯ ПРОЄКТАМИ В ІТ-СФЕРІ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі, коли технології стрімко розвиваються, ІТ-проєкти стають ключовим елементом успішного функціонування підприємств. Швидкі темпи змін і висока конкуренція вимагають від компаній ефективних методів управління проєктами для забезпечення високої якості продукції та своєчасного впровадження інновацій. Оптимізація процесів управління проєктами в ІТ-сфері стає ключовим елементом успіху, сприяючи підвищенню продуктивності, зниженню ризиків та вдосконаленню комунікацій.

Розглянемо основні методи оптимізації запасів, які знаходять своє використання в роздрібній торгівлі та мають значимий вплив на бізнес.

1. Використання методів Agile в управлінні проєктами

Використання методів Agile в управлінні проєктами в ІТ-сфері є однією з найбільш перспективних і ефективних стратегій для досягнення успішності та гнучкості в реалізації проєктів. Agile – це філософія та набір практик, які спрямовані на реалізацію ітераційних та інкрементальних змін у процесах розробки, зокрема в ІТ-проєктах. Використання методів Agile в управлінні проєктами в ІТ-сфері має декілька ключових аспектів, які сприяють оптимізації процесів.

Методи Agile сприяють гнучкому та адаптивному підходу до управління проєктами, що особливо важливо в умовах швидкозмінюваного ІТ-середовища. Класичні методи управління проєктами, які базуються на жорсткому плануванні та фіксації вимог, можуть виявитися неефективними у змінних умовах ринку та технологічного прогресу. Agile дає змогу командам більш ефективно реагувати на зміни та швидко адаптуватися до нових вимог.

Аналізувати сезонність та патерни попиту є критично важливо. Наприклад, у сфері роздрібного продажу одягу сезонні колекції можуть вимагати більших запасів перед початком сезону.

2. Впровадження інструментів управління завданнями та звітністю

Інструменти управління завданнями надають можливість точно розподіляти завдання між членами команди. Наприклад, платформи, як-от Jira чи Asana, дають змогу визначити конкретні завдання, призначити їх відповідальним співробітникам та встановити терміни виконання. Це не лише забезпечує чіткість щодо обов'язків, але й дає змогу ефективно відслідковувати прогрес кожного завдання.

Інструменти управління завданнями також відіграють ключову роль у створенні прозорості в процесах проєкту. Вони дають змогу генерувати різноманітні звіти щодо прогресу, витрат, термінів виконання та інших параметрів. Це дає змогу керівництву та всім учасникам проєкту отримувати необхідну інформацію в реальному часі, що сприяє швидкому виявленню можливих проблем та прийняттю своєчасних рішень.

Вони створюють сприятливе середовище для покращення комунікації між членами команди, допомагають обмінюватися інформацією, коментувати завдання, вирішувати питання та обговорювати стратегії в межах конкретних завдань. Це сприяє збільшенню взаєморозуміння та покращенню співпраці, що є критичним для успішності IT-проєктів.

3. Вдосконалення процесів тестування та якості програмного забезпечення

Використання автоматизованих тестів допомагає прискорити процес тестування та забезпечити його повторюваність. Автоматизовані тести можуть бути запущені автоматично під час кожної зміни коду, що дає змогу виявляти помилки раніше та зменшує час, потрібний для тестування. Це особливо важливо в умовах Agile-розробки, коли вимоги можуть змінюватися на ранніх етапах проєкту.

Впровадження CI / CD-процесів допомагає автоматизувати процеси інтеграції коду та впровадження змін у продукт. Це сприяє швидкому виявленню конфліктів та помилок, а також забезпечує можливість швидкої доставки нових функцій або виправлень користувачам.

Використання інструментів моніторингу та профілювання допомагає виявляти проблеми з продуктивністю та оптимізувати шляхи в коді. Це допомагає забезпечити ефективність програмного забезпечення та уникнути проблем, які можуть виникнути в продукційному середовищі [1].

4. Ефективне управління змінами та ризиками

Перший крок в ефективному управлінні ризиками – це ідентифікація та аналіз можливих небезпек. Команда проєкту повинна активно займатися пошу-

ком потенційних загроз та визначенням їх впливу на проєкт. Це може бути здійснено шляхом проведення SWOT-аналізу (Strengths, Weaknesses, Opportunities, Threats) та використання методів, як-от аналіз Fishbone (Ishikawa), або дерево визначення ризиків. Чітка ідентифікація допомагає побачити зміни та ризики заздалегідь, щоб вони не впливали негативно на виконання проєкту.

На основі ідентифікованих ризиків необхідно розробити план управління ризиками. Цей план містить стратегії зниження, прийняття, перенесення чи уникнення ризиків. Для кожного ризику треба визначити відповідальних за управління та контроль ризиками, а також визначити етапи реагування в разі виникнення негативних сценаріїв. Чітко сформований план допомагає команді ефективно реагувати на ризики та впроваджувати відповідні заходи ще до їх виникнення [2].

Зміни в процесі роботи нерідко необхідні для вдосконалення проєкту чи адаптації до нових вимог, однак важливо ефективно управляти цими змінами. Використання стандартів управління змінами, як-от ITIL (Information Technology Infrastructure Library) або CMDB (Configuration Management Database), допомагає контролювати та документувати всі зміни, визначати їх вплив на проєкт і забезпечувати комунікацію між усіма сторонами.

Ефективне управління змінами та ризиками потребує чіткої комунікації. Важливо відкрито обговорювати будь-які потенційні ризики з учасниками проєкту та стейкхолдерами. Застосування методів, як-от регулярні збори, статус-звіти та комунікація через спільні платформи, сприяє розумінню всіх сторін щодо поточного стану проєкту та змін, що можуть виникнути.

Список використаних джерел

1. Sutherland J. Scrum. Learn to do twice as much in less time. Crown Business, New York City, U.S. 2014. P. 280.

2. Anderson D. J. Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press, 2010. P. 261.

УДК 004.42

Михайляк М. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВИКОРИСТАННЯ DJANGO REST FRAMEWORK ДЛЯ СТВОРЕННЯ API: ПЕРЕВАГИ ТА МОЖЛИВОСТІ

Донецький національний університет імені Василя Стуса, м. Вінниця

В епоху стрімкого розвитку технологій та постійного прагнення до ефективності в сфері веброзробки використання потужних інструментів стає ключовим елементом успіху. Один із таких інструментів, що здобуває все більше популярності у спільноті розробників, – Django REST Framework (DRF). Вбудований в фреймворк Django, цей інструмент дає змогу легко створювати вебсервери та API, сприяючи швидкій розробці та принципам чистого та практичного дизайну.

Django – це високорівневий вебфреймворк, написаний мовою програмування Python, який сприяє швидкій розробці та чистому, практичному дизайну вебдодатків. Фреймворк використовує архітектурний шаблон Model-View-Controller (MVC) і підтримує принципи DRY (Don't Repeat Yourself) і CoC (Convention over Configuration).

Django REST Framework (DRF) – це розширення для фреймворку вебдодатків Django, яке надає потужні інструменти для створення вебсерверів та API. DRF допомагає легко створювати API, використовуючи стандарти RESTful, і надає широкий вибір можливостей для ефективної розробки вебдодатків.

DRF продовжує вирізнятися своєю гнучкістю, широким функціоналом і зручним інструментарієм для розробки високоефективних вебсервісів та API. З погляду тенденцій розробки програмного забезпечення акцент на мікросервісній архітектурі та розподілених системах підкреслює значимість добре розроблених і зручних інструментів для створення API, зокрема DRF. Актуальність теми підтверджується постійним попитом на фахівців, які володіють навичками роботи з Django REST Framework, і високим рівнем зацікавленості спільноти веброзробників у вдосконаленні та розширенні можливостей DRF для розвитку сучасних та ефективних вебдодатків [1].

Django REST Framework допомагає створювати API, використовуючи простий та ефективний підхід. Загалом у нього є багато переваг та можливостей, наприклад [2]:

- зручна робота з Django: DRF інтегрується плавно з фреймворком Django, розширюючи його можливості для швидкого створення API;
- автоматична документація: DRF автоматично генерує документацію для API, що спрощує розробку, тестування і розгортання;

➢ легкість серіалізації та десеріалізації: має потужний інструментарій для перетворення об’єктів Django в формати даних, як-от JSON, і навпаки;

➢ гнучка автентифікація та авторизація: надає готові рішення для автентифікації та авторизації, як-от токени, OAuth, базова автентифікація;

➢ розширюваність та кастомізація: легко розширюється та кастомізується з допомогою власних переглядів, серіалізаторів та інших компонентів;

➢ використання Django ORM: дає змогу використовувати Django ORM для взаємодії з базою даних, що робить роботу з даними простішою та зручною;

➢ вбудована підтримка переглядів: містить великий набір вбудованих переглядів для різноманітних сценаріїв використання, що прискорює розробку;

➢ відмінна підтримка автентифікації користувача: надає можливості роботи зі стандартними моделями користувача Django та автоматичну обробку сесій;

➢ маршрутизація: вбудована система маршрутизації для визначення, які URL відповідають різним переглядам;

➢ фільтрація і пошук: підтримка фільтрації та пошуку для легкості отримання конкретних наборів даних;

➢ робота з файлами і зображеннями: можливість обробляти завантаження файлів і роботи зображень;

➢ підтримка форматів даних: підтримка різних форматів введення / виведення даних, як-от JSON, XML, YAML;

➢ вбудовані засоби валідації: засоби для валідації та перевірки правильності даних у запитах.

Загалом DRF робить створення API простим та ефективним, забезпечуючи готові рішення для багатьох ключових аспектів розробки. Стабільність Django та спільноти навколо нього неймовірно зросла з моменту його першого випуску [3]. Офіційна документація та навчальні посібники фреймворку є одними з найкращих у своєму роді. І з кожною новою версією Django продовжує додавати нові можливості. Отже, використання Django REST Framework не лише допомагає швидко створювати високоякісні API, але і стає ключовим фактором у вдалих та масштабованих веброзробках, де ефективність і надійність важливі як ніколи.

Список використаних джерел

1. Фреймворки, які полегшують веб розробку на Python: вебсайт. URL: https://avivi.academy/blogs/python-framework/ (дата звернення: 26.11.2023).

2. Вкладені API за допомогою Django REST Framework: вебсайт. URL: https://devzone.org.ua/post/vkladeni-api-za-dopomogoiu-django-rest-framework (дата звернення: 26.11.2023).

3. Плюси і мінуси Django: вебсайт. URL: https://blog.ukrnames.com/vebmaster/plyusi-i-minusi-django (дата звернення: 26.11.2023).

4. Павлов Д. Л., Січко Т. В. Принцип роботи Web API та його застосування. *Прикладні інформаційні технології:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2023. С. 166–168.

УДК 004.021:656.02

Поліщук А. М., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ МЕРЕЖ

Донецький національний університет імені Василя Стуса, м. Вінниця

Транспортна мережа відіграє важливу роль у переміщенні товарів, послуг та людей. Незалежно від розмірів та складності мережі, оптимізація маршрутизації стає ключовим завданням, спрямованим на покращення ефективності та зменшення витрат. Ця проблема особливо актуальна в умовах зростаючого руху, коли міста та транспортні мережі стикаються з викликами, як-от затори, забруднення довкілля та пальне.

Дослідження та оптимізація маршрутизації стають необхідністю для подолання цих проблем. Точне та науково обґрунтоване планування маршрутів транспортних засобів може значно полегшити пересування та скоротити час подорожі. Оптимізована маршрутизація також може допомогти зменшити витрати на пальне, що важливо для економії ресурсів та збереження навколишнього середовища.

Отже, оптимізація маршрутизації у транспортних мережах є актуальною та важливою темою з практичним значенням і великим потенціалом для досліджень у галузі операційного дослідження та інженерії транспорту. У цьому контексті обговорюється важливість оптимізації маршрутизації та підходи до її вирішення.

Останні дослідження у сфері оптимізації маршрутизації вказують на зростання інтересу до використання штучного інтелекту та машинного навчання. Зокрема, використання цих методів дає змогу враховувати історичні дані про трафік, прогнозувати потік трафіка та враховувати різноманітні фактори, як-от погодні умови та події на дорогах. Це веде до підвищення точності та ефективності процесу маршрутизації.

Також варто зазначити зростання ролі геоінформаційних систем (ГІС) у цьому контексті. Використання ГІС дає змогу не лише визначати оптимальні маршрути, а й враховувати географічні обмеження, як-от особливості маршруту

та обмеження швидкості, що важливо для точного планування маршрутів у різноманітних умовах.

У цій галузі продовжуються дослідження, спрямовані на оптимізацію різних алгоритмів. Наприклад, дослідження адаптивних підходів до алгоритмів Дейкстри та A*, що допомагають їм більш ефективно працювати у складних транспортних мережах із багатьма пунктами призначення. Загалом останні дослідження свідчать про постійний розвиток та удосконалення методів оптимізації маршрутизації відповідно до сучасних викликів та технологічних можливостей.

Робота присвячена дослідженню оптимальних стратегій маршрутизації у транспортних мережах з метою зменшення часових витрат та зниження економічних ресурсів. Дослідження буде базуватися на використанні сучасних методів оптимізації та аналізу даних для покращення транспортних мереж і підвищення їх ефективності. Результати дослідження можуть бути використані для покращення логістики, зниження екологічного впливу та зменшення витрат у транспортній галузі.

Основними завданнями дослідження є аналіз наявних систем маршрутизації, моделювання транспортних мереж, розробка алгоритмів оптимізації, валідація алгоритмів, оцінка ефективності, аналіз впровадження у практиці, висновки.

Дослідження розглядає методи оптимізації маршрутизації з використанням теорії графів. Ця концепція передбачає представлення транспортної мережі у вигляді графу, де вузлами є пункти призначення, а ребрами – шляхи між ними. Для пошуку оптимальних маршрутів використовуються алгоритми Дейкстри та A*. Хоча вони досить точні та ефективні у багатьох випадках, вони можуть мати обмеження у складних транспортних мережах із багатьма пунктами призначення.

Існує багато різних підходів і алгоритмів, які використовуються для отримання оптимальних маршрутів у системі руху. Деякі з основних методів, які будуть розглянуті, включають:

1) алгоритм Дейкстри, заснований на пошуку найкоротшого шляху в графі;

2) алгоритм A*, що поєднує метод Дейкстри з евристичним методом;

3) генетичні алгоритми використовують моделювання природного відбору та еволюції, щоб знайти оптимальний шлях;

4) методи на основі штучного інтелекту, як-от машинне навчання та нейронні мережі, враховують багато факторів і змінних.

Кожен із цих методів має переваги та недоліки, і вибір залежить від конкретних вимог та обставин завдання маршрутизації. Алгоритм Дейкстри працює просто та швидко для малих графів, але неефективний для великих графів.

Алгоритм A* використовує евристику для спрощення пошуку та може бути надзвичайно ефективним, але неправильний вибір евристики може вплинути

на результати. Генетичні алгоритми можуть враховувати багато чинників, але вимагають обчислювальних ресурсів та налаштування параметрів. Методи на основі штучного інтелекту можуть враховувати змінні умови та велику кількість даних, але потребують обчислювальних ресурсів та даних. Вибір методу маршрутизації повинен враховувати конкретні вимоги щодо швидкості, точності та обчислювальної потужності, а також особливості мережі та потреби користувачів. Дослідники й інженери повинні ретельно аналізувати всі переваги та недоліки кожного методу для вибору найкращого рішення в конкретному сценарії маршрутизації.

Після дослідження та тестування ефективності різних методів оптимізації маршрутів у мережах трафіка було зроблено такі висновки:

1. Порівняння методів. У процесі тестування та поглибленого аналізу були реалізовані різні методи оптимізації маршрутів, зокрема Дейкстри, A* та генетичні алгоритми. Оцінено їх переваги та недоліки, пов'язані з конкретними завданнями транспортної мережі.

2. Ефективність маршрутизації. Використання різних методів оптимізації маршруту значно підвищило його ефективність, порівняно з традиційними методами. Оптимізовані маршрути часто скорочують час у дорозі та витрати на паливо.

3. Застосування в реальних умовах. Отримані результати перевірені на реальних мережах трафіка та логістичних системах. Розгортання оптимізованих маршрутів у реальних умовах допомогло скоротити час доставки, зменшити витрати на паливо та підвищити загальну продуктивність.

4. Інтеграційні моделі. У межах дослідження були розроблені моделі для інтеграції оптимізованих маршрутів із наявними системами управління логістикою та транспортними підсистемами. Це сприяло успішній реалізації вдосконалених маршрутів. Дослідження підтвердили, що використання розроблених методів оптимізації маршрутів може значно підвищити ефективність і продуктивність транспортних мереж, приводячи до скорочення часу та витрат у дорозі, а також сприяє оптимізації логістичних процесів. Отримані результати мають велике значення, коли точність і швидкість маршрутизації є вирішальними для успіху операцій і зменшення негативного впливу на навколишнє середовище.

Дослідження показують, що оптимізація маршрутів у транспортних мережах є важливим завданням, яке може покращити транспортну інфраструктуру, скоротити час у дорозі та заощадити паливо.

Розглянуті методи, а саме алгоритми на основі графів, машинне навчання та генетичні алгоритми, можна успішно використовувати у різних сферах – від громадського транспорту до логістики. Дослідження підтвердило ефективність цих методів, які допомагають скорочувати час та витрати під час планування маршрутів. Однак є питання, які потребують подальших досліджень – адапта-

ція методів до змінних умов і співпраця з географічними інформаційними системами. Загалом оптимізація маршрутизації у транспортних мережах є актуальною і важливою задачею з потенціалом покращити якість життя та зменшити витрати в різних галузях. Розглянуті методи можуть бути використані для вирішення реальних проблем у сферах транспорту і логістики.

Список використаних джерел

1. Гарін, Д. В. (2018). Оптимізація маршрутизації в логістичних системах. Київ: Видавництво «Логістика».

2. Петренко, О. І., Коваленко, В. М. (2019). Застосування методів оптимізації у транспортних системах. Дніпро: Дніпропетровський національний університет ім. О. Гончара.

3. Коцюба, В. П. (2020). Геоінформаційні системи у транспорті. Львів: Видавництво Львівської політехніки.

4. Степаненко, О. В., Мельник, І. М. (2017). Оптимізація маршрутизації в автомобільних транспортних системах. Харків: Видавництво Харківського національного університету ім. Каразіна.

5. Січко, Т. В., Смоктій, К. В., Ткачук, А. О. Прикладні аспекти розрахунку структурно-топологічних характеристик систем. Системи та технології. 2019. № 1(57), 2019. С. 141–153.

УДК 004.89

Поліщук А. М., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ПОКРАЩЕННЯ ЇХНЬОЇ ПРОДУКТИВНОСТІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Останніми роками глибокі нейронні мережі (ГНМ) завоювали значну популярність завдяки своїм вражаючим досягненням у багатьох сферах, зокрема комп'ютерне зорове сприйняття та обробка природних мов. Однак разом зі зростанням їх застосування виникають проблеми, пов'язані з високим рівнем складності та великою кількістю параметрів, що можуть призводити до значних обчислювальних витрат. Глибокі нейронні мережі, особливо глибокі згорткові та рекурентні мережі, мають велику кількість шарів, які потрібно оптимізувати під час тренування. Для цього необхідно на великих обсягах даних ефективне навчання моделі, що може призводити до перетренування на невеликих набо-

рах даних. Також велика кількість параметрів ускладнює роботу з моделями і може вимагати великих обчислювальних ресурсів для їх тренування та використання.

Отже, виникає потреба в пошуку ефективних методів оптимізації для покращення продуктивності глибоких нейронних мереж. Деякі з напрямів досліджень передбачають оптимізацію функцій витрат, використання регуляризації для уникнення перетренування, а також розробку нових архітектур та методів оптимізації, які дають змогу ефективно використовувати обмежені обчислювальні ресурси. Одним із ключових завдань є зменшення кількості параметрів без втрати продуктивності моделі, що може передбачати застосування методів стиснення моделей, які дають змогу зберігати лише найважливіші параметри, або розробку нових методів регуляризації для автоматичного видалення зайвих параметрів. Усі ці напрями досліджень спрямовані на те, щоб зробити глибокі нейронні мережі більш доступними та ефективними у різних застосуваннях, зменшуючи обчислювальні витрати та покращуючи їх здатність до роботи з обмеженими обсягами даних.

Оптимізація глибоких нейронних мереж є актуальною проблемою в сучасній науці та технологіях. Вдосконалення продуктивності ГНМ може відкрити нові можливості в області штучного інтелекту, а також допоможе ефективно використовувати їх у реальних умовах [1].

Останні дослідження у сфері оптимізації ГНМ сфокусовані на різноманітних аспектах, як-от алгоритми оптимізації, регуляризація, архітектурна оптимізація та інші. Низка робіт вказує на ефективність методів оптимізації градієнтів, як-от Adam та RMSProp, водночас як інші дослідження розглядають використання еволюційних алгоритмів для оптимізації архітектур нейронних мереж.

Метою є вдосконалення продуктивності глибоких нейронних мереж шляхом аналізу та оптимізації різних аспектів їх структури та навчання.

Вплив різних методів оптимізації та регуляризації на результати нейронних мереж є критично важливим аспектом у галузі глибокого навчання.

1. Стандартний стохастичний градієнтний спуск (SGD). Це базовий метод оптимізації, який оновлює масу моделі в напрямі, протилежному градієнту функції втрати. Однак цей метод може застрягати в локальних мінімумах та мати повільний збіг.

2. Метод Адама. Цей метод використовує експоненційно зважене середнє градієнта та квадрата градієнта. Він може адаптувати швидкість навчання для кожного параметра окремо, що робить його ефективним для швидкого та стабільного тренування [2].

3. RMSprop. Цей метод також адаптує швидкість навчання для кожного параметра, але використовує експоненційно згладжений середній квадрат градієнта. Це допомагає вирішити проблему швидкого величезного зростання градієнта.

Використання різних методів оптимізації допомагає досягти оптимального балансу між швидкістю збігу та уникненням проблем, як-от розходження чи застрягання в локальних мінімумах.

Методи регуляризації:

1. Dropout. Цей метод регуляризації полягає у випадковому вимиканні деяких нейронів під час тренування. Це допомагає уникнути перенавчання;

2. L1- та L2-регуляризація. Ці методи вводять штраф на величину мас, змушуючи модель використовувати менше параметрів. Це допомагає уникнути перенавчання;

3. Batch Normalization. Цей метод вставляє шар нормалізації між шарами мережі. Він допомагає стабілізувати процес тренування та прискорює збіг моделі.

Експерименти показали, що комбінація різних методів регуляризації може призводити до ще кращих результатів. Важливо враховувати специфіку завдання та розмір набору даних під час вибору підходів.

Вплив комбінації: ефективність нейронної мережі визначається не тільки окремими методами, але й їх комбінацією. Наприклад, використання алгоритму Адама разом із dropout може покращити здатність моделі уникати перенавчання та підвищити її робастність.

Також важливо враховувати взаємодію параметрів, як-от швидкість навчання та розмір пакету для навчання. Наприклад, занадто велика швидкість навчання може призвести до розходження, тоді як занадто мала може призвести до повільного збігу.

Архітектура мережі:

1. Глибина мережі: використання глибоких архітектур може допомогти у вирішенні складних завдань та отриманні високої точності. Проте важливо уникати перенавчання, особливо за обмежених ресурсів для тренування.

2. Кількість нейронів в шарах: оптимальний розмір шарів може залежати від специфіки завдання. Велика кількість нейронів може призвести до перенавчання, тоді як занадто мала – до недонавчання.

3. Функції активації: вибір функцій активації, як-от ReLU, Sigmoid або Tanh, впливає на здатність моделі до навчання та уникнення проблем, як-от зникаючий градієнт.

Параметри навчання:

1. Швидкість навчання: оптимальна швидкість навчання допомагає досягти збіжності моделі. Проте важливо уникати занадто великих значень, які можуть призвести до розходження.

2. Розмір пакету для навчання: великий розмір пакету може прискорити тренування, але під час цьому може виникнути проблема втрати різноманітності даних. Малий розмір пакету може призвести до непостійності оновлень ваг.

Функція втрати:

Вибір оптимальної функції втрати: залежно від завдання вибір відповідної функції втрати може бути критичним. Наприклад, для класифікації може використовуватися категоріальна крос-ентропія, а для регресії – середньоквадратична помилка.

Регуляризація:

Використання dropout та L1 / L2 регуляризації: ці методи можуть допомагати уникнути перенавчання та покращити загальну робастність моделі.

Експерименти показали, що оптимальна комбінація параметрів залежить від конкретного завдання та набору даних. Постійне налаштування та вдосконалення цих параметрів під час тренування допомагає досягти кращих результатів та покращити ефективність нейронних мереж.

Використання функцій активації, як-от ReLU або Leaky ReLU, може сприяти швидшому збігу під час тренування. Також треба зазначити, що велике значення має правильний вибір функції втрат для кожного конкретного завдання. Використання відповідної функції втрати може значно поліпшити результати моделі та допомогти уникнути проблем перенавчання.

Загальним висновком до виконаних досліджень є те, що успіх у навчанні нейронних мереж залежить від балансу між різними методами оптимізації та регуляризації, вибором правильних параметрів і архітектурою мережі. Також важливо постійно адаптувати ці параметри під час тренування, особливо під час роботи з великими та складними даними. Ці дослідження роблять важливий внесок у сферу штучного інтелекту, допомагаючи розуміти, як різні методи можуть взаємодіяти та впливати на результати моделей.

Список використаних джерел

1. Гудфелоу, Я., Бенжіо, І., Курвіль, А. Глибоке навчання. MIT Press, (2016). 652 с.

2. Кінгма, Д. Адам: Метод стохастичної оптимізації. Препринт arXiv arXiv. Вип. 1412, п. 6980. (2014). С. 1–15.

3. Срівастава, Н., Гінтон, Г., Крижевський, А., Суцкевер, Я., Салахутдінов, Р. (2014). Dropout: Простий спосіб запобігти перенавчанню нейронних мереж. Journal of Machine Learning Research, 15, С. 1929–1958.

УДК 004.9

Рудь О. С., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

РОЛЬ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ У ВИРІШЕННІ ЕКОЛОГІЧНИХ ПРОБЛЕМ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сьогодні екологічні проблеми стають не лише актуальними, але й невідкладними завданнями, що потребують ретельного вивчення та негайних заходів для їх вирішення. Загострення змін клімату, виснаження природних ресурсів та інші екологічні виклики стають суттєвими загрозами для екосистем та життя на планеті. У цьому контексті обговорення ролі інформаційних технологій у вирішенні екологічних питань є важливим і необхідним завданням, оскільки саме вони можуть виступити каталізатором для розробки ефективних стратегій боротьби з цією проблемою.

Актуальність обраної теми надзвичайно висока, зважаючи на зростання негативного впливу людської діяльності на навколишнє середовище. Сучасна екологічна криза та забруднення природних ресурсів вимагають ефективних рішень, а інформаційні технології є інструментом для моніторингу, аналізу та управління екологічними процесами. Використання ІТ у цій сфері є доволі важливим складником для досягнення сталого розвитку та збереження природного середовища для майбутніх поколінь.

ІТ відкривають перед нами можливості нагляду та контролю за станом довкілля. Завдяки передовим сенсорам, дронам і супутникам ми можемо отримувати детальні дані про забруднення повітря, води та ґрунту. Це дає змогу точно визначити джерела забруднення та вчасно реагувати на екологічні загрози.

Електронні системи моніторингу забруднення працюють у режимі реального часу, надаючи оперативну інформацію про перевищення норм забруднення. Це створює можливість вжити швидких та ефективних заходів для захисту довкілля від негативного впливу людської діяльності [1].

Інформаційні технології також революціонізують управління відходами. Електронні системи відслідковують потоки відходів та оптимізують їх переробку. Це сприяє зменшенню кількості відходів, що потрапляють на смітники, та створенню ефективних систем утилізації.

Використання екологічно чистих технологій, як-от відновлювані джерела енергії та енергоефективні рішення, стає можливим завдяки розвитку ІТ. Це не лише зменшує викиди забруднюючих речовин, але й сприяє збереженню природних ресурсів.

Можливості ІТ допомагають виявляти та реагувати на екологічні проблеми, й до того ж сприяють зміні свідомості суспільства щодо важливості збереження природи для майбутніх поколінь.

Приклад: системи моніторингу забруднення повітря у мегаполісах. Мегаполіси зазнають серйозних проблем із забрудненням повітря, що має негативний вплив на здоров'я населення та екосистеми. Системи моніторингу за допомогою інформаційних технологій можуть виявляти джерела забруднення, надавати реальний час для реагування та сприяти прийняттю ефективних заходів для зменшення впливу на довкілля [2].

Реалізація: задача моніторингу забруднення повітря в мегаполісі полягає в тому, щоб забезпечити точний аналіз та контроль якості повітря в режимі реального часу. Для цього використовуються різноманітні інформаційні технології та сучасні засоби збору даних.

Розглянемо детальніше функціональності такої системи:

1. Датчики забруднення:

➢ розміщення: система використовує мережу датчиків, розміщених у різних частинах міста, зокрема центральні райони, промислові зони та житлові райони;

➢ типи датчиків: датчики вимірюють різні параметри: рівні оксидів азоту, сірки, вуглеводні та інші шкідливі речовини.

2. Супутникові засоби:

➢ оптичні супутники: використання оптичних супутників для отримання високоякісних зображень та аналізу областей з високим рівнем забруднення повітря;

➢ теплові карти: застосування теплових карт для визначення точок найбільшого тепловиділення, що може вказувати на витоки забруднюючих речовин.

3. Збір та передача даних: безпровідний зв'язок – використання технологій безпровідного зв'язку для передачі даних у режимі реального часу, що дає змогу оперативно реагувати на зміни у якості повітря.

4. Прогностичні моделі: застосування математичних моделей для прогнозування розвитку забруднення повітря на певний період часу на основі історичних даних та даних від супутників.

5. Географічна інформаційна система (ГІС):

➢ карти забруднення: використання ГІС для створення карт, які візуалізують рівні забруднення повітря в різних частинах міста;

➢ аналіз розташування: визначення розташування джерел забруднення та його вплив на конкретні райони.

6. Вебпортал та мобільні додатки:

➢ реально-часовий доступ: створення вебпорталу та мобільних додатків, які надають громадянам реальний час доступу до даних про якість повітря у їх локації;

➢ системи сповіщень: можливість надсилання автоматичних сповіщень на мобільні пристрої громадян у разі перевищення допустимих норм забруднення.

7. Аналіз та звітність:

➢ статистичний аналіз: проведення статистичного аналізу даних для виявлення трендів та аномалій у забрудненні повітря;

➢ звітність: генерація регулярних звітів для владних органів та громадськості, які містять інформацію про якість повітря та ефективність заходів для її покращення.

Ці функціональності спільно допомагають створити інтегровану систему моніторингу, яка забезпечує ефективний контроль за якістю повітря та дає змогу приймати оперативні рішення для зменшення негативного впливу на здоров'я та довкілля [3].

У висновку треба зазначити, що інформаційні технології є важливим рушієм у вирішенні сучасних екологічних проблем. Їх роль у моніторингу та контролі за станом довкілля дає змогу ефективно реагувати на екологічні загрози. Можливості систем моніторингу забруднення повітря, які забезпечують сучасні сенсорні технології, відкривають шлях до оперативного виявлення та вирішення проблемних ситуацій. ІТ стають не лише інструментом збору даних, але й фактором у формуванні науково-інформаційної бази для прийняття обґрунтованих рішень у галузі екології. Загалом інформаційні технології стають надійним союзником у збереженні природи та формуванні екологічно відповідального суспільства.

Список використаних джерел

1. Інформаційні технології у розв'язанні екологічних проблем: вебсайт. URL: https://referatss.com.ua/work/informacijni-tehnologii-u-rozv-jazanni-ekologichnih-problem/ (дата звернення: 01.12.2023).

2. Використання ІТ у екології: вебсайт. URL: https://moodle.znu.edu.ua/pluginfile.php/984836/mod_resource/content/1/Лекція%206.pdf (дата звернення: 02.12.2023).

3. Левків С. П. Застосування інформаційних технологій на уроках екології: *Вісник Житомирського державного університету. Педагогічні науки.* Житомир, 2014. Вип. 4(76). С. 127–133.

УДК 519.86

Семенюк А. М., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ РОЗМІЩЕННЯ СОНЯЧНИХ ПАНЕЛЕЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

Для зменшення негативного впливу людини на природу під час вироблення електроенергії використовують відновлювальні ресурси. Позитивний ефект від їх залучення можливий лише завдяки підвищенню коефіцієнта генерації потужностей таких альтернативних джерел енергії (АДЕ) – як за кількістю, так і за ефективністю.

Особливо актуальним це питання стає під час використання сонячних станцій (СЕС). У цьому випадку існує багато факторів, які впливають на збільшення вихідної потужності. Дослідження цих факторів, а також оптимізація операцій вироблення електроенергії, режимів експлуатації СЕС із використанням методів моделювання, дасть змогу отримати приріст генерації – збільшити ККД станції.

Розглядувана в цій роботі система керування положення сонячних панелей (СКП СП) дає можливість проводити на підставі статистичної інформації про положення сонця залежно від дати та часу доби автоматичний розрахунок орієнтації світлоприймачів як у горизонтальній, так і у вертикальній площині, тобто позиціонування їх відносно сонця для отримання максимального ефекту. Для обчислення зміни положення панелей використовується програмний засіб і оновлюванні статистичні кутові координати СП.

Сонячна енергія, яка є головним джерелом «сировини» для отримання електроенергії СЕС – це невичерпне безкоштовне джерело, тому використання СП є перспективним напрямом. Але, як уже зазначалось, своя «ложка дьогтю» тут теж присутня – це пошук оптимального розміщення СП під час побудови сонячних електростанцій.

Під час цього треба врахувати:

- сонце – не статичний елемент, воно рухається як в азимутальних координатах, так і по висоті (кут нахилу сонячних променів відносно Землі, рис. 1);
- широту місця установки;
- нахил до горизонту;
- хмарність неба;
- часткове затінення панелей;
- туман, дощ;
- сніг та його налипання на панелі та ін.

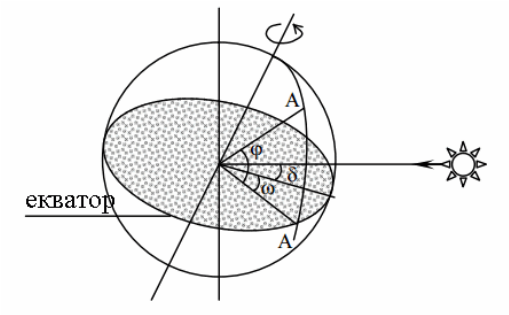


Рисунок 1. Основні параметри руху Сонця

Необхідно розраховувати положення СП в оптимальному розміщені відносно Сонця:

$$cos(i) = \cos(\delta)\cos(\varphi)\cos(\omega) + \sin(\delta)\sin(\varphi), \quad (1)$$

де d – кут схилення Сонця;

j – широта місця установки;

w – годинний кут.

Найчастіше ця дія зводиться до розрахунку середньорічного (у кращому випадку середньосезонного) кута сонячних променів і фіксації панелей у перпендикулярному положенні відносно цих променів (рис. 2).

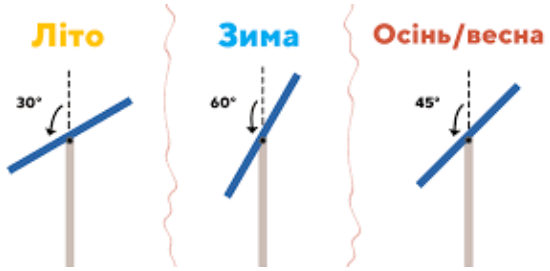


Рисунок 2. Оптимальний «сезонний» кут розміщення СП

Такий підхід є нераціональним, тому пропонується метод постійного моніторингу координат Сонця. На підставі аналізу результатів визначається перспективне положення точки в області пошуку. Ці значення будуть використані для позиціонування панелі в базове положення, в якому проводиться більш «тонке», прецизійне налаштування.

Для цього змінюємо положення фотомодулів на фіксовані кути за двома осями, з урахуванням показників допоміжних фотодатчиків актуаторів, і проводимо розрахунок, використовуючи інтерполяційний поліном Лагранжа:

$$P_n(t) = \sum_{i=0}^{n} L_i(t) y_i, \quad (2)$$

де y_i – значення кута нахилу панелей у вузлах інтерполяції.

Похибку параболічної інтерполяції (оптимальне зміщення) можна оцінити з допомогою залишкового члена ряду, який можна записати так:

$$f(t) - P_n(t) = \frac{f^{(n+1)}(\varepsilon)}{(n+1)!}(t - t_0)(t - t_1) \dots (t - t_n), \quad (3)$$

де e – точка, що належить інтервалу, на якому розміщені вузли інтерполяції.

Якщо точка знаходиться всередині області пошуку, то процес розв'язання оптимізаційної задачі завершується, тобто точка являє собою наближене рішення цієї задачі. Для кожної зі змінних ця помилка визначається виразом:

$$\Delta x_i = (x_{i\,min}, x_{i\,max})/N, \quad (4)$$

де $x_{(i\,min)}$, $x_{(i\,max)}$ – від'ємне та додатне (зазвичай різні) відхилення положення СП від оптимального;

N – кількість вузлів ґратки фотоелемента.

Комп'ютерне моделювання етапу переміщення дає змогу використовувати СП найбільш оптимально і економічно раціонально. Застосування актуаторів з декількома додатковими фотодатчиками освітленості одночасно з коригуванням оптимального положення панелі дає можливість слідувати за Сонцем і отримувати максимум сонячної енергії протягом дня і року.

Завдяки накопиченню інформації про зміну положення панелі цій СКП доступна функція самонавчання, що дає змогу коригувати положення СП залежно від низки факторів: спрацювання механізму, похибки тощо.

СКП СП, яка розглянута в цій роботі, допоможе підвищити загальну потужність генерації СЕС завдяки оптимальному розміщенню фотоприймачів відносно Сонця. Накопичені статистичні дані дадуть змогу розраховувати положення СП на початку світлового дня («стартове») залежно від руху Сонця відносно Землі відповідно до періоду року та географічного розміщення пристрою. За недостатньої освітленості, у випадку негоди, чи темного періоду доби відбувається переведення СП в черговий режим. Після його закінчення панелі позиціонуються згідно з попередньо накопиченою і обробленою статистичною інформацією на поточний період.

Список літературних джерел

1. Ляшенко Б. М., Кривонос О. М., Вакалюк Т. А. Методи обчислень: навч.-метод. посіб. для студентів фізико-математичного факультету. Житомир: Вид-во: ЖДУ, 2014. 228 с.

2. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів СО «Бакалавр» денної та заочної форм навчання спеціальностей

122 «Комп'ютерні науки», 113 «Прикладна математика. Вінниця: ДонНУ імені Василя Стуса. 2020, 104 с.

3. Семенюк А. М. Розрахунок кута повороту та нахилу площини сонячних панелей. *Наукові дослідження молоді з проблем європейської інтеграції:* збірник тез доповідей XII Міжнародної науково-практичної конференції молодих учених та студентів. Харків: ХНУ імені В. Н. Каразіна, 2023. С. 299–301.

УДК 004.8

Слободянюк С. С., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВИКОРИСТАННЯ ШІ В ОПТИМІЗАЦІЇ ОБСЛУГОВУВАННЯ КЛІЄНТІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Штучний інтелект (ШІ), розглядається як ключовий компонент у вдосконаленні взаємодії між підприємствами та їх клієнтами, сприяючи покращенню ефективності та задоволеності обох сторін.

1. Автоматизація обробки запитань та замовлень. ШІ дає змогу автоматизувати обробку запитань та замовлень через використання чат-ботів та віртуальних асистентів. Це допомагає підприємствам оперативно відповідати на запитання клієнтів та виконувати замовлення без затримок, підвищуючи швидкість обслуговування.

2. Персоналізація обслуговування. Інтелектуальні алгоритми ШІ аналізують дані про покупки, взаємодію з платформою й інші параметри, щоб створити персоналізовані рекомендації та пропозиції для кожного клієнта. Це допомагає підприємствам підтримувати тісний зв'язок зі своїми клієнтами та підвищувати рівень їх задоволеності.

3. Прогнозування та аналіз потреб. ШІ забезпечує можливість прогнозування змін у попиті та аналізу потреб клієнтів на основі великої кількості даних. Це допомагає підприємствам адаптувати свої стратегії обслуговування, уникати переповнень та недостач товарів, а також пропонувати точно те, що потрібно клієнтам у конкретний момент.

4. Забезпечення якості обслуговування 24/7. Інтелектуальні системи можуть працювати некеровано та обслуговувати клієнтів у будь-який час доби. Це створює можливість для підприємств забезпечувати високий рівень обслуговування в будь-який момент, що є особливо важливо, коли багато клієнтів перебувають у різних часових поясах.

5. Захист від шахрайства та надання безпечних послуг. Інтелектуальні системи можуть використовувати алгоритми машинного навчання для виявлення шахрайства та захисту від кібератак. Це робить обслуговування більш безпечним для клієнтів та підприємств.

Приклад: віртуальний асистент для обробки клієнтських запитань.

Припустимо, що компанія виробляє та продає продукти через свій інтернет-магазин. Щоб полегшити процес взаємодії з клієнтами, вони впроваджують віртуального асистента на основі штучного інтелекту. Коли клієнти заходять на вебсайт чи використовують мобільний додаток компанії, вони можуть взаємодіяти з віртуальним асистентом для вирішення своїх питань та проблем.

Запитання клієнта: «Які варіанти доставки доступні для мого регіону?»

Відповідь віртуального асистента: віртуальний асистент аналізує текст запитання, використовуючи природну мову та розпізнавання мови, і надає конкретну відповідь на запитання клієнта.

Персоналізація: віртуальний асистент може враховувати історію покупок та попередні взаємодії клієнта для надання персоналізованої інформації, наприклад, даючи змогу клієнту знати, чи є у нього знижки на доставку, або пропонуючи оптимальний варіант.

Миттєва відповідь: система штучного інтелекту допомагає надавати миттєві відповіді, покращуючи враження клієнта та зменшуючи час очікування.

Розширення функціональності: віртуальний асистент може бути розширений для виконання більш складних завдань, як-от оформлення замовлень, відстеження статусу доставки або навіть рекомендації товарів на основі історії купівель та попередніх взаємодій.

Отже, використання штучного інтелекту в обслуговуванні клієнтів є ключовим фактором у вдосконаленні бізнес-процесів та підвищенні конкурентоспроможності. Це не тільки сприяє автоматизації, але і створює нові можливості для покращення взаємодії з клієнтами, що призводить до збільшення задоволеності та лояльності клієнтів.

Список використаних джерел

1. TS2. URL: https://ts2.space/uk/ші-в-обслуговуванні-клієнтів/#gsc.tab=0 (дата звернення: 25.11.2023).

2. Гуменюк К. В., Січко Т. В. Основи машинного навчання. Комп’ютерні технології обробки даних: матеріали всеукр. наук.-практ. конф., м. Вінниця, 2022. С. 142–144.

3. Андрійченко К. А., Січко Т. В. Аналіз даних як складова інформаційних технологій. *Комп’ютерні технології обробки даних:* матеріали Всеукр. наук.-практ. конф., м. Вінниця, 2020. С. 8–10.

УДК 519.17

Цегольник В. В., здобувач 3 курсу ОС «Бакалавр», спеціальність 122 Комп'ютерні науки,

Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ПОБУДОВА МОДЕЛІ НА ОСНОВІ ТЕОРІЇ ГРАФІВ ДЛЯ ОПТИМІЗАЦІЇ ВІЙСЬКОВОЇ ЛОГІСТИКИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Теорія графів – це розділ математики, який вивчає властивості графів – математичних структур, що складаються з вершин і ребер, які з'єднують ці вершини.

Моделі на основі теорії графів – це моделі, які використовують графи для представлення досліджуваної системи. Ці моделі можуть бути використані для різних цілей, як-от аналіз системи, прогнозування її поведінки або оптимізація її роботи. Вони мають низку переваг перед іншими типами моделей, оскільки є потужним інструментом для аналізу складних систем, і вони можуть бути використані для вирішення широкого спектру завдань.

В умовах повномасштабної війни рівень логістики має вирішальне значення для успіху Збройних Сил України. Від того, наскільки ефективно будуть доставлені зброя, боєприпаси, продовольство та інше необхідне майно, залежить, чи зможе Україна дати відсіч російському вторгненню. Логістика включає в себе планування, організацію та управління рухом матеріалів, людей і інформації. В умовах війни це завдання значно ускладнюється. Російські війська активно обстрілюють транспортні шляхи, що ускладнює доставку вантажів. До того ж необхідно враховувати безпеку людей, які займаються логістикою. Маючи безліч понівечених доріг різного рівня складності та небезпеки, набагато краще було б мати заздалегідь обраховані шляхи із можливістю зміни стану змінних за короткі проміжки часу [1].

Розглянемо реальну ситуацію. Волонтерам необхідно привезти визначену кількість їжі та медикаментів усім найближчим підрозділам в найкоротшим шляхом у найкоротший час. Кожна вершина має певну кількість з'єднань, кожна з яких має уже обраховану вагу, що включає в себе значення складності та небезпеки. Цей підхід допоможе нам створити універсальну модель, яка може застосовуватись у будь-якій сфері, де необхідно щось оптимізувати, необов'язково військовій [2].

Постановка задачі така: представимо зважений ненапрямлений граф із заданими параметрами та створимо програму для знаходження мінімального кістяка за алгоритмом Пріма, водночас у програмі виводитимемо зображення

графу у графічному вікні перед кожною зупинкою. Таке рішення буде раціональним, якщо в майбутньому на основі цієї моделі дані будуть оновлювати в режимі реального часу. Зрозуміло, що кількість вершин буде обмеженою для зручності сприйняття в невеликому графічному вікні. Під час обходу графу також побудуємо дерево його кістяка.

Мінімальний кістяк графу – це підмножина ребер графу, яка з'єднує всі вершини графу і має мінімальну вагу. Алгоритм Пріма – це жадібний алгоритм, який працює шляхом побудови дерева з фіксованою вершиною. На кожному кроці алгоритму вибирається ребро з найменшою вагою, яке з'єднує дерево з вершиною, яка не входить у нього. Це ребро додається до дерева, і вершина, до якої воно приєднується, включається в дерево. Алгоритм продовжує працювати, поки всі вершини графу не будуть включені в дерево. На практиці саме обраний алгоритм гарантуватиме, що побудований ним граф є мінімальним деревом, оскільки на кожному кроці він вибирає ребро з найменшою вагою, яке з'єднує дерево з вершиною, яка не входить у нього [3]. Це означає, що сума ваг ребер у цьому дереві є мінімальною. Алгоритм є лінійним відносно до кількості ребер у графі, що робить його ефективним для великих графів.

Для автоматичної побудови складного графу, використаємо правила побудови, запропоновані штучним інтелектом [4]:

$$A = mulmr((1.0 – 0.01 –7 \cdot 0{,}005 – 0.05) \cdot T).$$

Матриця ваг W формується так:

1) Wt = roundm((randm(n, n) · 100) ** A), де roundm – це функція, що округлює кожен елемент матриці до найближчого цілого числа, символ «**» – поелементне множення;

2) одержується матриця B, у якій bij = 0, якщо wij = 0, bij = 1, якщо wij > 0, bij ∈ B, wij ∈ Wt;

3) одержується матриця C, у якій cij = 1, якщо bij != bji, та cij = 0 в іншому випадку;

4) одержується матриця D, у якій dij = 1, якщо bij = bji = 1, та dij = 0 в інших випадках;

5) Wt = (C + (D ** Tr)) ** Wt, де Tr – верхній трикутник матриці одиниць (без головної діагоналі), + це поелементна сума матриць;

6) одержується матриця ваг W шляхом симетризування матриці Wt.

Використовуючи процедурну мову C, напишемо програму (рис. 1).

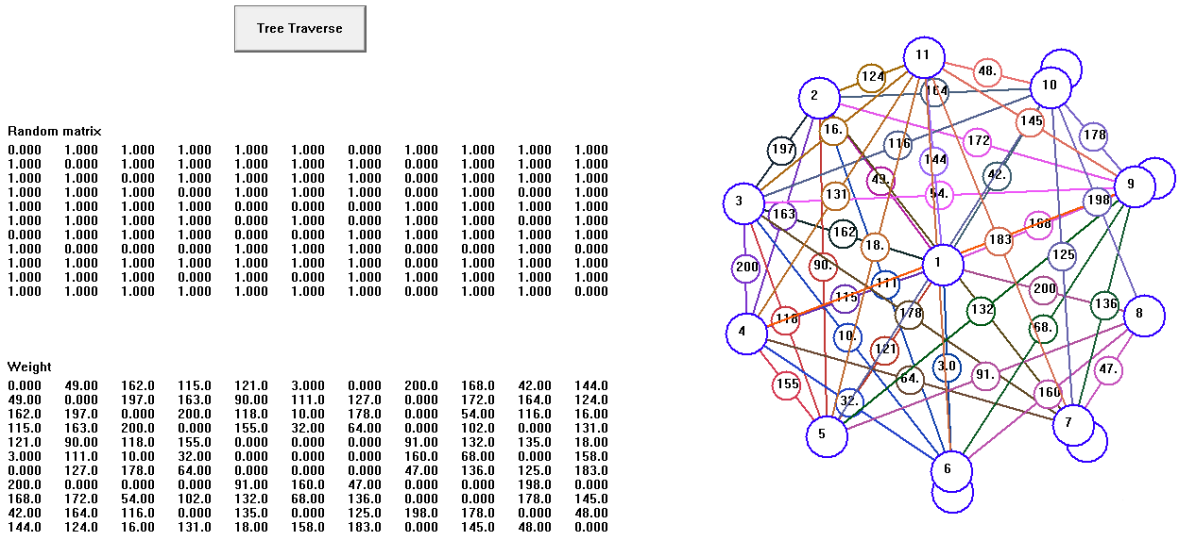


Рисунок 1. Стартовий вигляд моделі

Для зручності орієнтації в малюнку зв'язки позначаються різними кольорами, а зв'язки вершин самих до себе є лише побічним ефектом рандому і жодної цінності для досліджуваної задачі не несуть. Під час натискання кнопки «Обхід дерева», «Вершина», «Шлях, обраний алгоритмомом» та «Вага шляху» будуть підсвічуватись іншим кольором, що допоможе обрати найоптимальніший шлях (рис. 2).

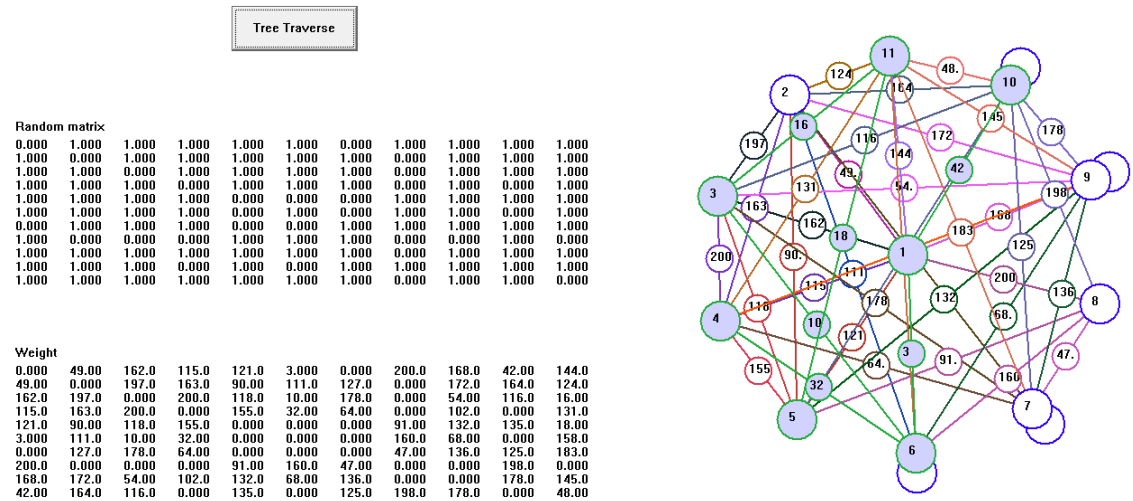


Рисунок 2. Приклад роботи моделі

Отже, розгляд витрат у логістичних системах є одним із найважливіших і найзатребуваніших питань. Становлення логістики насамперед обумовлюється прагненням до скорочення різної класифікації витрат, які пов'язані з товарорухом і утворенням результативних логістичних ланцюгів для підвищення ефективності військових операцій та зменшення ризику затримок у доставці

матеріальних ресурсів. Внаслідок виконання поставленої задачі була отримана модель, представлена графічним способом, що досліджує найкоротший шлях між усіма вершинами графу, базуючись на складності зв’язків, і може бути сміливо застосована у будь-якій сфері оптимізації шляху, насамперед військовій.

Список використаних джерел

1. Contemporary challenges in military logistics support. URL: https://securityanddefence.pl/Contemporary-challenges-in-military-logistics-support,103332,0,2.html

2. Data science in battle: Applying graph theory to the Ukraine war. URL: https://medium.com/intuitionmath/data-science-in-battle-applying-graph-theory-to-the-ukraine-war-6cda499c86fc

3. Prim’s Algorithm for Minimum Spanning Tree (MST) URL: https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/

4. C Program to Create a Random Graph Using Random Edge Generation. URL: https://www.sanfoundry.com/c-program-generate-random-undirected-graph-given-number-edges/

СЕКЦІЯ 3
ЕКСПЕРТНІ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ ТА СИСТЕМИ ПІДРИМКИ ПРИЙНЯТТЯ

УДК 004.8

Ліваковський В. К., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВПЛИВ ШТУЧНОГО ІНТЕЛЕКТУ НА БІЗНЕС-ПРОЦЕСИ ТА ПРИЙНЯТТЯ РІШЕНЬ

Донецький національний університет імені Василя Стуса, м. Вінниця

Останніми роками штучний інтелект набув значного поширення у багатьох сферах людської діяльності, зокрема й в управлінні бізнес-процесами. Потужність і можливості штучного інтелекту привернули увагу дослідників та практиків, що призвело до використання штучного інтелекту в системах управління бізнес-процесами і виявлення сильних та слабких сторін цього підходу. Результати використання ШІ в системах управління бізнес-процесами сприяють кращому розумінню цього підходу і можуть слугувати основою для подальшого розвитку та застосування ШІ в сучасних практиках управління бізнесом.

Штучний інтелект став невід'ємною частиною сучасного суспільства і має великий потенціал для використання в системах управління бізнес-процесами. Він може аналізувати великі обсяги даних, виявляти складні залежності, робити прогнози та приймати рішення на основі об'єктивних алгоритмів, наприклад, коли користувач переглядає або купує товари в інтернет-магазині, система штучного інтелекту може враховувати ці дані для створення індивідуального профілю користувача. На основі цього профілю алгоритми можуть прогнозувати, які товари можуть бути цікавими користувачу в майбутньому. Такий підхід допомагає підвищити ефективність продажів, зменшити час, який користувач витрачає на пошук товарів і створює позитивний досвід купівель. Усе це робиться завдяки аналізу великих обсягів даних та прийняттю об'єктивних рішень алгоритмами штучного інтелекту.

Однією з переваг використання штучного інтелекту є автоматизація рутинних і повторюваних завдань управління бізнес-процесами, наприклад, коли компанія отримує велику кількість резюме від потенційних працівників, система штучного інтелекту може використовуватися для автоматичного аналізу цих документів. Алгоритми обробки природної мови можуть ідентифікувати ключові навички, досвід та освіту кандидатів. Цей процес автоматизує рутинні етапи відбору, зменшує час, який витрачається на перегляд великої кількості документів, і допомагає кадровим спеціалістам зосередитися на більш стратегічних аспектах, як-от співбесіди та взаємодія зі співробітниками. Отже,

штучний інтелект допомагає підвищити ефективність управління бізнес-процесами шляхом автоматизації рутинних завдань. Це звільняє людські ресурси від монотонної роботи і спрямовує їх на вирішення складних завдань, які потребують креативного мислення та стратегічного підходу.

Штучний інтелект має переваги під час використання, в системах управління бізнес-процесами. Він може автоматично виконувати завдання з обробки даних, моніторингу процесів, прогнозування попиту та оптимізації робочих графіків. Успішним прикладом застосування ШІ в управлінні бізнес-процесами є використання систем розпізнавання образів і машинного навчання для автоматичного контролю якості продукції. Ці системи можуть розпізнавати дефекти продукції з високою точністю і швидкістю, зменшуючи у такий спосіб кількість бракованої продукції й підвищуючи її якість. До того ж використання додатків зі штучним інтелектом знижує витрати, покращує процес прийняття рішень і стимулює інновації; з огляду на свої можливості та обмеження штучний інтелект є потужним інструментом для досягнення успіху в управлінні бізнес-процесами.

Незважаючи на численні переваги, використання штучного інтелекту в системах управління бізнес-процесами має певні недоліки та обмеження. Справді, ефективність системи штучного інтелекту напряму залежить від якості та репрезентативності вхідних даних. Моделі машинного навчання, які використовуються в ШІ, навчаються на основі наданих їм наданих. Якщо ці дані неправильні, несистематичні або не репрезентують реальний світ, результати можуть бути неточними та недостовірними.

Наприклад, якщо система вчиться розпізнавати об’єкти на фотографіях і отримує обмежений набір зображень лише одного типу або недостатньо різноманітний, її здатність розпізнавання може бути обмеженою та неадекватною у різних сценаріях. Відсутність або низька якість даних може призвести до недостовірних результатів і помилкових процесів прийняття рішень. Це може виникнути, коли алгоритми намагаються знайти залежності або вирішувати завдання на неповних, неоднорідних або змінюваних даних. Важливо визнати, що якість і репрезентативність даних – ключові фактори для успішного впровадження та роботи системи штучного інтелекту.

Однак використання ШІ в системах управління бізнес-процесами може значно підвищити ефективність роботи компанії. Зокрема, використання ШІ може підвищити продуктивність завдяки автоматизації рутинних завдань і швидкого аналізу великих обсягів даних. Наприклад, в інтернет-магазині система ШІ може автоматизувати обробку замовлень, визначаючи оптимальний шлях для виконання замовлення та прогнозуючи час доставки на основі аналізу великих обсягів даних. Це дає змогу підвищити продуктивність шляхом автомати-

зації рутинних завдань і вдосконалення внутрішніх процесів, забезпечуючи оперативне та ефективне обслуговування клієнтів. Співробітники можуть зосередитися на стратегічних завданнях і важливих процесах прийняття рішень. До того ж ШІ може допомогти компаніям скоротити витрати. Завдяки автоматизації бізнес-процесів та оптимізації планування ресурсів можна зменшити витрати на робочу силу, скоротити час виконання завдань і зменшити ймовірність помилок. ШІ може аналізувати багатофакторні дані та враховувати складні залежності, що дає змогу приймати об'єктивні та оптимальні рішення.

Штучний інтелект має великий потенціал для виконання важливих наукових і практичних завдань у сфері управління бізнес-процесами. Наприклад, алгоритми машинного та глибокого навчання можуть бути використані для виявлення складних закономірностей у даних та формулювання нових стратегій управління. Штучний інтелект також можна використовувати для прогнозування ринкових тенденцій, аналізу конкурентного середовища та здійснення стратегічного планування.

Отже, штучний інтелект має великий потенціал у системах управління бізнес-процесами. Він уможливлює автоматизацію, підвищує ефективність, зменшує витрати та покращує якість прийняття рішень. Однак варто враховувати недоліки та обмеження, пов'язані з наявністю якісних даних і системною інтеграцією: використання ШІ вимагає ретельного планування та аналізу, але він може стати потужним інструментом для успішного управління бізнес-процесами.

Водночас ризики безконтрольного чи неетичного застосування технологій штучного інтелекту становлять цілком реальні небезпеки для людства. А саме:

- втрата робочих місць людьми через автоматизацію рутинних повторюваних операцій;
- порушення приватності ледь не до повної її руйнації;
- Deepfakes (синтез слів «глибинне навчання» та «підробка» – методика синтезу зображення чи аудіоряду мовлення людини, яка базується на штучному інтелекті; вона використовується для поєднання і накладення наявних зображень та відео на вихідні зображення або відеоролики).

Однак, знаючи про способи створення й навчання сучасних систем штучного інтелекту, неважко пересвідчитися у тому, що ці ризики можуть стати реальними не через власне самі системи, а через «втаємничених невідомих» – «деміургів», тобто знову ж таки людей.

Список використаних джерел

1. Гужва В. М. Інформаційні системи і технології на підприємствах: навч. посіб. Київ: КНЕУ, 2001. 300 с.

2. Association I. R. M. Cognitive Analytics: Concepts, Methodologies, Tools, and Applications. IGI Global, 2020. 196 с.

3. Brynjolfsson E., Mcafee A. Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies. Norton & Company Limited, W. W., 2016. 336 с.

4. Intelligence A. Introducing the Neo Revolutionary Thought User Interface (TUI). Independently Published, 2017. 134 с.

5. Семенюк О. А., Кирилащук Т. Г., Січко Т. В. Прикладні аспекти обробки даних в інформаційних системах. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2021. С. 212–213.

УДК 004.8

Ліваковський В. К., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

ВДОСКОНАЛЕННЯ БІЗНЕС-ПРОЦЕСІВ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Останні десятиліття свідчать про швидкі та радикальні зміни в бізнес-середовищі, викликані впровадженням штучного інтелекту (ШІ). ШІ виступає не лише як інструмент автоматизації, але й як ключовий компонент для оптимізації та вдосконалення бізнес-процесів. Багато досліджень присвячені вивченню методів оптимізації та вдосконалення бізнес-процесів, базованих на принципах та можливостях ШІ.

Безліч компаній уже впровадили технології штучного інтелекту у свої бізнес-процеси й отримали значні вигоди від цього. Наприклад, компанія Amazon застосовує системи ШІ для рекомендації продуктів покупцям на основі їхніх уподобань і попередніх купівель. Google використовує штучний інтелект для поліпшення пошукових результатів і персоналізації рекомендацій. Facebook застосовує AI для аналізу вмісту та виявлення порушень правил платформи, а також для створення автоматичних субтитрів і розпізнавання облич. Це лише кілька прикладів компаній, які успішно впровадили технології штучного інтелекту і отримали значні переваги на ринку.

Хоча впровадження штучного інтелекту в бізнесі обіцяє безліч переваг, воно також пов'язане з деякими викликами і перешкодами. Однією з основних

проблем є необхідність наявності якісних даних для навчання систем ШІ. Без достатнього і якісного обсягу даних системи штучного інтелекту не можуть давати точні та достовірні результати. Для подолання цієї проблеми необхідно інвестувати у збір і обробку даних. Ще одним викликом є необхідність розроблення етичних стандартів і нормативів у галузі штучного інтелекту. Системи ШІ можуть мати доступ до великої кількості персональних даних користувачів, що порушує питання про конфіденційність та захист особистої інформації. Також постає питання про відповідальність за помилки та рішення, які ухвалює ШІ. Розробка та впровадження відповідних правових та етичних меж є важливим аспектом успішного використання штучного інтелекту в бізнесі.

Методи оптимізації та їх взаємодія зі штучним інтелектом є важливим аспектом наукових та практичних досліджень у сучасній інформаційній епосі. Оптимізація як математична галузь спрямована на знаходження оптимальних рішень в умовах обмежень і часто використовується для покращення різноманітних бізнес-процесів та прийняття ефективних стратегій, наприклад, у компанії з електронною комерцією оптимізація може бути використана для управління запасами товарів на складі. Штучний інтелект аналізує історію продажів, попиту та інші фактори. Оптимізаційні методи враховують обмеження, як-от місце на складі, терміни доставки та бюджет, і автоматично визначають оптимальні обсяги товарів для замовлення.

Штучний інтелект вносить новий рівень інтелектуальності в оптимізаційні завдання. Використання алгоритмів машинного навчання, нейронних мереж, генетичних алгоритмів та інших методів ШІ дає змогу автоматизувати та покращувати процеси оптимізації. Наприклад, у сфері енергетики штучний інтелект може використовуватися для оптимізації виробництва електроенергії відновлювальними джерелами. Алгоритми машинного навчання можуть аналізувати дані про погодні умови, попит на електроенергію та характеристики обладнання. Система може використовувати ці дані для оптимізації режимів роботи вітро- та сонячних електростанцій, враховуючи змінність погодних умов і пікові години споживання електроенергії. Алгоритми нейронних мереж можуть прогнозувати попит і допомагати в управлінні ресурсами для максимізації виробництва зеленої енергії та зниження витрат. Такий підхід допомагає ефективно використовувати відновлювальні джерела енергії, оптимізуючи їх продуктивність у реальному часі.

Принципово важливою є роль ШІ в навчанні систем адаптивності, які можуть адекватно реагувати на зміни у вхідних даних та умовах завдань оптимізації. Моделі машинного навчання можуть аналізувати та узагальнювати великі обсяги даних, роблячи оптимальні рішення більш точними та адаптованими. Наприклад, у сфері фінансів ШІ може відіграти ключову роль в оптимізації інвестиційного портфеля. Система може використовувати моделі машинного

навчання для аналізу ринкових тенденцій, економічних показників та інших факторів. Навчання системи адаптивності може включати в себе автоматичне виявлення та коригування стратегій інвестування в реальному часі відповідно до змін на ринку. Наприклад, якщо зміни в економічних умовах вказують на зростання ризику в конкретному секторі, система може динамічно перерозподілити інвестиції для зменшення можливих втрат.

Підхід, що базується на ШІ, може мати значимі переваги в ситуаціях, де традиційні методи досягають своїх обмежень, особливо в умовах невизначеності та динамічних змін. Інтеграція ШІ із задачами оптимізації розширює область застосування та підвищує рівень ефективності у вирішенні реальних проблем. Наприклад, у компанії виникає несподіваний підвищений попит на певний продукт через велике замовлення або зміни в ринкових умовах. Традиційні методи управління ланцюгом постачання можуть бути недостатньо ефективними у вирішенні цієї ситуації. ШІ може використовувати алгоритми машинного навчання для аналізу рядів даних, зокрема попиту наявності на складі та часу постачання. Система може динамічно перерозподілити ресурси, прогнозуючи і адаптуючись до змін у реальному часі. Нейронні мережі можуть допомагати у прогнозуванні подій та управлінні запасами, забезпечуючи оптимальне реагування на непередбачувані обставини. Цей підхід допомагає оптимізаційній системі ефективно вирішувати завдання в умовах невизначеності і динамічних змін, що в перспективі підвищує продуктивність і знижує ризики.

Отже, впровадження штучного інтелекту в бізнес-середовище не тільки автоматизує процеси, але також стає ключовим компонентом для оптимізації та їх вдосконалення. Однак існують виклики, як-от необхідність якісних даних і вирішення етичних питань. Методи оптимізації, які взаємодіють зі штучним інтелектом, вирішують завдання ефективно та розширюють область їх застосування. Отже, ШІ та оптимізація є важливим аспектом сучасного бізнес-середовища, забезпечуючи підвищення продуктивності та вирішення реальних проблем.

Список використаних джерел

1. Юрчак О. В. Індустрія 4.0 – що це таке та навіщо це Україні. URL: https://appau.org.ua/publications/industriya-4-0-shho-tse-take-ta-navishho-tse-ukrayini/

2. Metasonic. URL: http://bpm.blogic20.ru/metasonic

3. Schwab K. The Fourth Industrial Revolution: what it means, how to respond Retrieved. URL: https://www.weforum.org/agenda/2016/01/the-fourth-industrial-revolution-what-itmeans-and-how-to-respond/

4. Семенюк О. А., Кирилащук Т. Г., Січко Т. В. Прикладні аспекти обробки даних в інформаційних системах. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2021. С. 212–213.

5. Січко Т. В., Нескородєва Т. В. Методичні вказівки щодо виконання лабораторних робіт з дисципліни «Методи оптимізації та дослідження операцій» для студентів СО «Бакалавр» денної та заочної форм навчання спеціальностей 122 «Комп'ютерні науки», 113 «Прикладна математика». Вінниця: ДонНУ імені Василя Стуса. 2020, 104 с.

УДК 004.8

Стукан А. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ВИКОРИСТАННЯ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ДЛЯ АНАЛІЗУ ПОПИТУ ТА РЕКОМЕНДАЦІЙ В E-COMMERCE ТА ІНШИХ СФЕРАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Використання інтелектуальних систем у сферах e-commerce та інших галузях стає все більш важливим для аналізу попиту та надання рекомендацій користувачам. Штучні нейронні мережі, а також інші алгоритми машинного навчання допомагають створювати складні моделі, які можуть аналізувати великі обсяги даних та робити передбачення з високою точністю.

Аспекти використання інтелектуальних систем:

1. Аналіз попиту в e-commerce. Розробка інтелектуальних систем, які аналізують купівлі, перегляди, дії користувачів для визначення попиту на товари та послуги. Використання штучних нейронних мереж для прогнозування та моделювання змін попиту в часі.

2. Рекомендації користувачам. Розробка персоналізованих систем рекомендацій, що базуються на історії купівель, вподобань та поведінки користувачів. Алгоритми машинного навчання використовуються для надання точних рекомендацій товарів або послуг, що підходять конкретним користувачам.

3. Оптимізація інвентарю. Використання інтелектуальних систем для прогнозування попиту на товари та планування запасів. Аналіз даних допомагає управляти інвентарем більш ефективно та уникати недостачі чи перевищення запасів.

4. Розуміння поведінки користувачів. Використання аналітики даних і моделей машинного навчання для розуміння та прогнозування поведінки користувачів. Це допомагає усунути бар'єри для купівлі та підвищити залучення клієнтів.

Перспективні напрями розвитку інтелектуальних систем:

1. Застосування методів комп'ютерного зору на основі нейронних мереж для аналізу та класифікації зображень товарів чи користувачів. Це дає змогу отримувати додаткову інформацію про об'єкти.

2. Використання чат-ботів і діалогових систем на основі штучного інтелекту для взаємодії з користувачами. Такі системи здатні надавати консультації, відповідати на запитання, робити персональні рекомендації.

3. Застосування технологій обробки природної мови для аналізу відгуків, чатів, дописів користувачів з метою кращого розуміння їхніх потреб та уподобань.

Висновки. Використання інтелектуальних систем у сферах e-commerce та інших галузях створює можливості для аналізу великих обсягів даних та надання персоналізованих рекомендацій користувачам. Впровадження таких систем дає змогу підвищити точність прогнозів попиту, поліпшити взаємодію з користувачами та ефективно управляти інвентарем, забезпечуючи більш задовільний досвід для клієнтів.

Список використаних джерел

1. Технічна документація від компанії-розробника моделей штучного інтелекту H2O.ai. URL: https://docs.h2o.ai/

2. Відкриті онлайн-курси «Машинне навчання» від Стенфордського університету. URL: https://www.coursera.org/learn/machine-learning

УДК 004.8

Стукан А. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ПОБУДОВА РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ НА ОСНОВІ ШТУЧНХ НЕЙРОННИХ МЕРЕЖ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Рекомендаційні системи на основі штучних нейронних мереж є ключовим елементом сучасних інформаційних технологій, сприяючи зручності та персоналізації взаємодії користувачів з різноманітними об'єктами. Одним із найефективніших підходів до створення таких систем є використання штучних нейронних мереж (ШНМ). Нейронні мережі здатні адаптуватися до складних зв'язків у даних та роблять можливим використання різноманітних моделей для аналізу і прогнозування інтересів користувачів.

Приклади застосування рекомендаційних систем на основі ШНМ:

1. Netflix. Netflix використовує рекомендаційну систему на основі штучних нейронних мереж, яка аналізує історію перегляду користувачів та забезпечує персоналізовані рекомендації фільмів та серіалів.

2. Amazon. Рекомендаційна система Amazon використовує ШНМ для аналізу купівель та інших дій користувачів, щоб надавати рекомендації стосовно товарів на основі інтересів користувачів.

3. YouTube. YouTube використовує нейронні мережі для рекомендацій відео, аналізуючи перегляди та дії користувачів для рекомендацій відповідно до їх вподобань.

Покращення ефективності рекомендаційних систем на основі ШНМ:

1. Змішані моделі. Комбінація CF та Content-Based методів може покращити точність рекомендацій, використовуючи гібридні підходи.

2. Вдосконалені архітектури. Використання складніших архітектур, як-от Transformer, дає змогу враховувати довгострокові залежності та покращує точність прогнозування.

3. Колаборативне навчання. Поєднання декількох моделей із допомогою методів колаборативного навчання сприяє покращенню якості рекомендацій.

4. Зважені фактори та увага. Використання механізмів уваги та оптимізація функцій втрати підвищує точність рекомендаційних систем.

5. Оптимізація гіперпараметрів. Ефективне налаштування гіперпараметрів допомагає підвищити ефективність рекомендаційних систем на основі ШНМ.

Висновки. Ці підходи сприяють покращенню точності та зручності рекомендаційних систем на основі штучних нейронних мереж, забезпечуючи більш персоналізовані і задовільні досвіди для користувачів.

Список використаних джерел

1. Блог компанії Netflix про їх рекомендаційну систему. URL: https://netflixtechblog.com/netflix-recommendations-beyond-the-5-stars-part-1-55838468f429

2. Онлайн-курс «Рекомендаційні системи» на Coursera. URL: https://www.coursera.org/courseraplus/special/insite?redirectedfromexpired

3. Стаття Вікіпедії про рекомендаційні системи. URL: https://uk.wikipedia.org/wiki/Рекомендаційна_система

УДК 004.89

Таран Н. М., здобувач 2 курсу ОС «Магістр»,
Зелінська О. В., канд. техн. наук, доцент, в. о. завідувача кафедри інформаційних технологій

РЕКОМЕНДАЦІЙНА СИСТЕМА ПЛАНУВАННЯ РЕКЛАМНОЇ КАМПАНІЇ НА ОСНОВІ АНАЛІЗУ ДАНИХ ВЕБТРАФІКУ ТА ПОВЕДІНКИ КОРИСТУВАЧІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Роль системи рекомендацій у плануванні та виконанні рекламних кампаній на основі поведінки користувачів є надзвичайно важливою і сприяє досягненню максимального впливу та ефективності в рекламній сфері. Система рекомендацій аналізує великі обсяги даних про поведінку користувачів, щоб зрозуміти їхні потреби, інтереси, попередні взаємодії та передбачити їхні майбутні вчинки.

Однією з ключових переваг цієї системи є її здатність до персоналізації рекламних повідомлень. Вони враховують унікальні характеристики кожного користувача, як-от його зацікавленості, демографічні дані, попередні купівлі та взаємодії з рекламою, щоб зробити рекомендації, які найкраще відповідають їх потребам. Це допомагає збільшити рівень залучення та зацікавленості користувачів, що підвищує конверсію та ефективність рекламної кампанії.

Покращення взаємодії з користувачами шляхом надання індивідуальних та контекстуальних рекомендацій реклами відіграє ключову роль у підвищенні ефективності рекламних кампаній і створенні більш особистого досвіду для користувачів. Переваги такого підходу:

1. Підвищення релевантності. Надання індивідуальних рекомендацій дає змогу адаптувати рекламу до унікальних потреб та інтересів кожного користувача. Замість загальних повідомлень контент реклами стає більш релевантним, що збільшує ймовірність привернення уваги та зацікавленості користувача.

2. Персоналізація повідомлень. Індивідуальні рекомендації дають змогу адаптувати контент рекламного повідомлення до унікальних характеристик кожного користувача, як-от його розташування, демографічні дані, попередні взаємодії тощо. Це створює враження персоналізованості та підвищує ймовірність позитивної реакції на рекламу.

3. Забезпечення контекстуальної релевантності. Контекстуальні рекомендації враховують контекст, у якому переглядається реклама, як-от тип контенту, місце, час тощо. Це допомагає добирати рекламні повідомлення, які більш точно відповідають ситуації та потребам користувача. Наприклад, реклама товарів

для водного спорту може відображатися на сторінках про пляжний відпочинок або в літній період.

4. Покращення користувацького досвіду. Індивідуальні та контекстуальні рекомендації допомагають зробити взаємодію з рекламою більш приємною та цікавою для користувачів. Вони отримують контент, який є релевантним та цікавим для них, що підвищує ймовірність переходу до дії, наприклад, купівлі або реєстрації.

5. Підвищення ефективності рекламної кампанії. Внаслідок надання індивідуальних та контекстуальних рекомендацій реклами збільшується ефективність рекламної кампанії. Користувачі, які бачать релевантну рекламу, з більшою ймовірністю відреагують на неї та перейдуть до бажаної дії, що допомагає досягти поставлених маркетингових цілей.

Впровадження систем рекомендацій реклами на основі поведінки користувачів може стикатися з деякими викликами та перешкодами, як-от: захист приватності, недостача якісних даних, складність алгоритмів та технічних рішень, обробка великого обсягу даних. Незважаючи на ці виклики, система рекомендацій реклами може принести значні переваги для бізнесу і користувачів, забезпечуючи більш персоналізований та релевантний досвід взаємодії з рекламою.

Список використаних джерел

1. The effect of personalized and behavior based advertising on the consumers. URL: https://www.researchgate.net/publication/346018188_THE_EFFECT_OF_PERSONALIZED_AND_BEHAVIOR_BASED_ADVERTISING_ON_THE_CONSUMERS (дата звернення: 22.05.2023).

2. When is personalized advertising crossing personal boundaries? How type of information, data sharing, and personalized pricing influence consumer perceptions of personalized advertising. URL: https://www.sciencedirect.com/science/article/pii/S2451958821000920 (дата звернення: 22.05.2023).

3. How to Personalize Content With Behavioral Targeting. URL: https://www.clearvoice.com/resources/10-ways-to-put-the-power-of-behavioral-targeting-to-work/ (дата звернення: 22.05.2023).

СЕКЦІЯ 4
БАЗИ ДАНИХ І ЗНАНЬ. BIG DATA

УДК 004.62:004.65:004.75

Алексюк В. В., здобувач 4 курсу спеціальності 122 Комп'ютерні науки,
Комаров В. Ф., канд. техн. наук, старший викладач кафедри інформаційних технологій

MONGODB ЯК ЕФЕКТИВНИЙ ЗАСІБ У ЗАДАЧАХ ОБРОБКИ ТА АНАЛІЗУ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У міру того, як компанії намагаються подолати все більший обсяг і різноманітність інформаційних потоків, потреба у гнучких, масштабованих і швидких системах баз даних (БД) стає все більш очевидною. У динамічному середовищі науки про дані (Data Science), де попит на ефективні рішення для управління даними та аналітики продовжує невпинно зростати, MongoDB, що пропонує універсальну платформу та знаходиться на старті вирішення означених проблем, стає ключовим гравцем, який змінює загальний підхід до великих наборів даних (Big Data) та їх використання.

Існує два основні типи баз даних: реляційні (SQL) та нереляційні (NoSQL). Реляційні бази даних зберігають дані у стовпцях і рядках. Microsoft SQL Server, Oracle і Sybase використовують реляційні системи керування базами даних (СКБД). На відміну від них, бази NoSQL зберігають неструктуровані дані без схем у численних колекціях і вузлах. Вони не потребують фіксованих схем таблиць, масштабуються горизонтально, але підтримують обмежену кількість запитів на об'єднання. MongoDB є представником таких БД.

MongoDB була створена в 2009 р. [1] як відкрита, масштабована, надійна і безкоштовна база даних NoSQL з відкритим вихідним кодом, зарекомендувала себе як універсальна, гнучка база даних, і сьогодні використовується як внутрішнє сховище даних багатьох відомих компаній та організацій, як-от Forbes, Facebook, Google, IBM, X (Twitter) та інші.

Актуальність MongoDB в контексті Data Science складно переоцінити. Її пристосованість до роботи з неструктурованими та напівструктурованими даними в поєднанні з можливістю безперешкодної інтеграції з популярними мовами програмування позиціонує її як потужний інструмент в арсеналі фахівців.

MongoDB має такі ключові особливості [2]:

- **загальне призначення** – MongoDB може обслуговувати різноманітні набори даних та різні цілі в межах однієї бази даних;
- **гнучка модель** – MongoDB використовує гнучку, безсхемну модель документів, що дає змогу зберігати різнорідні типи даних в одній колекції;
- **масштабованість** – MongoDB розроблена для горизонтального масштабування, що робить її добре придатною для задач із швидко зростаючими обся-

гами даних, розподіляючи дані між декількома серверами; MongoDB може забезпечувати безперебійну масштабованість для пристосування до зростаючих навантажень;

➢ **мова запитів** – MongoDB допомагає користувачам виконувати складні запити, агрегації та індексування завдяки потужній та виразній мові запитів, що полегшує ефективний пошук та аналіз даних і має вирішальне значення для задач науки про дані;

➢ **JSON** – для бази даних використовується той самий протокол, що широко використовується для зв'язку між інтерфейсом і API.

Завдяки гнучкій моделі документів, що підтримує різні типи даних, масштабованості та можливостям мови запитів MongoDB є динамічним та універсальним рішенням для використання в задачах Data Science, яке дає такі можливості [3]:

➢ **Зберігання та обробка даних.** MongoDB слугує основним сховищем для зберігання різноманітних даних, зокрема, неструктурованих або напівструктурованих даних, які часто зустрічаються в Data Science.

➢ **Робота з великими обсягами даних.** MongoDB масштабується горизонтально, легко обробляючи великі обсяги даних. Така масштабованість має вирішальне значення в сучасних задачах обробки даних, обсяги яких постійно збільшуються.

➢ **Аналітика та агрегація даних.** Використовуючи мову запитів та фреймворк агрегації MongoDB, складні аналітичні операції та агрегацію даних можна виконувати безпосередньо в базі даних, що суттєво спрощує процес підготовки даних до аналізу.

➢ **Геопросторовий аналіз.** Завдяки вбудованій підтримці геопросторових запитів та індексації MongoDB є ефективним інструментом для аналізу й обробки географічних даних.

➢ **Зберігання та використання результатів моделей машинного навчання.** MongoDB може слугувати сховищем для вхідних даних та результатів моделей машинного навчання, даючи змогу зберігати й керувати моделями в єдиному середовищі.

➢ **Інтеграція з інструментами Data Science.** MongoDB легко інтегрується з різними популярними інструментами Data Science та мовами програмування, як-от Python або R, забезпечуючи безперебійний робочий процес та обмін даними між середовищами.

Останній пункт варто розібрати детальніше. Python, що є популярною мовою програмування у спільноті фахівців Data Science, легко інтегрується з MongoDB за допомогою бібліотеки PyMongo [4]. Бібліотека надає інтерфейс Python для взаємодії з MongoDB, який дає змогу швидко отримати доступ для

виконання маніпуляцій із даними, їх пошуку і аналізу, коли на встановлення з'єднання з базою даних потрібно лише кілька рядків коду.

Такий зв'язок Python та MongoDB стає все популярнішим, оскільки обидва інструменти є вкрай потужними в області розробки та обробки даних. Python як мова програмування володіє зручним синтаксисом, що робить його обраною мовою для великої кількості розробників та аналітиків. MongoDB як БД типу NoSQL пропонує гнучку модель документів, що ідеально взаємодіє з динамічним та розширюваним характером Python. Це допомагає розробникам та аналітикам з легкістю працювати з неструктурованими даними, а також швидко й ефективно взаємодіяти з базою даних. Швидкість і гнучкість є ключовими факторами поєднання Python та MongoDB.

Підсумовуючи вкажемо, що інтеграція MongoDB у Data Science виходить за межі традиційного зберігання даних, слугуючи рушійною силою для інновацій. Окрім своєї фундаментальної ролі в управлінні даними, MongoDB надає переваги у вирішенні безпосередньо задач обробки та аналізу даних, поєднання великих наборів даних і машинного навчання.

Список використаних джерел

1. Why Use MongoDB: What It Is and What Are the Benefits. URL: https://www.simplilearn.com/tutorials/mongodb-tutorial/what-is-mongodb

2. Growing Significance of MongoDB in Data Science Field. URL: https://www.edureka.co/blog/the-growing-significance-of-mongodb-in-data-science-field/

3. Introduction to MongoDB for Data Science. URL: https://www.knowledgehut.com/blog/data-science/mongodb-for-data-science

4. Exploring MongoDB for Data Science and Python. URL: https://www.linkedin.com/pulse/exploring-mongodb-data-science-python-aritra-pain-3rd-year-/

УДК 004.04

Підруцький Д. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

BIG DATA В ОБРОБЦІ І АНАЛІЗІ ДАНИХ З РІЗНИХ СФЕР

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі, охопленому хвилею технологічного розвитку, важко уявити галузь, яка залишається поза впливом великих обсягів даних, або так

званих Big Data. З допомогою високотехнологічних інструментів і методів обробки даних Big Data стає ключовим елементом у різних сферах – від бізнесу та медицини до науки і громадського управління.

Метою цього дослідження є ретельний аналіз та систематизація можливостей використання технологій Big Data в різних сферах. Зокрема, дослідження спрямоване на визначення самих технологій, того, як саме їх можливо застосовувати, їх методів пов'язаних із даними, а також можливих ризиків під час застосування цих технологій.

Для досягнення визначеної мети поставлено такі завдання:

- надати визначення технологій Big Data;
- визначити основні методи аналізу Big Data;
- визначити ключові аспекти використання технологій Big Data в різних сферах.

Big Data (укр. – Великі дані) – це великий масив структурованої та неструктурованої інформації, а також інструменти, підходи, методи обробки і зберігання даних. Важливість великих даних залежить не тільки від їх кількості, а й від того, як компанія їх інтерпретує та використовує. Через об'єм та різноманітність даних обробляти їх традиційним програмним забезпеченням неможливо [1].

Дані в галузі Big Data класифікуються згідно з «П'ятьма V», які визначають п'ять ключових характеристик [2] (рис. 1):

Рисунок 1. Основні риси Big Data

Перша характеристика «Об'єм (Volume)» – являє собою величезний обсяг накопиченої бази даних, який практично неможливо обробити та зберегти за допомогою традиційних методів. Рис. 1 демонструє, що цей обсяг інформації вимагає альтернативного підходу та використання спеціалізованих інструментів.

Друга характеристика «Швидкість (Velocity)» – визначає темпи накопичення та обробки даних, які постійно збільшуються. Сучасний попит на технології обробки даних у режимі реального часу позначається таким самим темпом зростання.

Третя характеристика «Різноманітність (Variety)» – позначає можливість одночасної обробки як структурованої, так і неструктурованої інформації. Структурована інформація, яка становить лише 5 % від загального обсягу, містить дані, які можна класифікувати, наприклад, інформацію з банківської бази даних. Неструктурована інформація, що становить решту 80 %, охоплює різноманітні дані, як-от фотографії, відео та інші, наприклад, із соціальних мереж.

Необхідність перевірки достовірності даних виражена у четвертій характеристиці – «Достовірність (Veracity)». Якість отриманої інформації може значно відрізнятися, що впливає на точність подальшого аналізу.

Остання, п'ята характеристика «Змінюваність (Variability)» – відображає невідповідність інформації, що може ускладнювати й іноді значно заважати процесам обробки та управління даними. Початок форми.

Тепер перейдемо до методів аналізу великих даних [3], оскільки великі обсяги інформації у самому своєму виді не мають значення для людини, необхідно провести їх аналіз для досягнення конкретної мети. Для обробки цих великих обсягів даних використовуються різноманітні інструменти, перелік яких постійно оновлюється. Серед них виділяють такі техніки та методики:

- класифікація (classification): використовується для передбачення поведінки споживачів у певному сегменті ринку;
- кластерний аналіз (cluster analysis): застосовується для класифікації об'єктів за групами, виявляючи їх спільні ознаки;
- масовий аналіз (crowdsourcing): використовується для збору інформації з різноманітних джерел;
- розробка даних (data mining): застосовується для виявлення раніше невідомих та корисних відомостей, які можуть бути використані для прийняття рішень у різних галузях;
- машинне навчання (machine learning): передбачає створення нейронних мереж, які самонавчаються та здатні якісно і швидко обробляти інформацію;
- обробка сигналів (signal processing): використовується для розпізнавання сигналів на тлі шуму та їх подальшого аналізу;
- навчання без нагляду (unsupervised learning): використовується для виявлення прихованих функціональних взаємозв'язків у даних;
- візуалізація (visualization): використовується для презентування результатів аналізу у вигляді діаграм і анімації.

Розглянемо сфери прямого застосування Big Data [4].

1. У медицині: шляхом аналізу обширної кількості медичних записів та обробки медичних знімків можна досягти точнішого й раніше поставленого діагнозу, глибшого розуміння природи різноманітних захворювань та розробки нових методів лікування і ліків.

2. У галузі сільського господарства відбувається справжня революція Big Data, що допомагає оптимізувати використання ресурсів для максимального збільшення врожаїв за мінімального впливу на екосистему та зниження вартості вирощених продуктів завдяки раціональному використанню обладнання і добрив.

3. У науці: завдяки аналізу великої кількості даних, отриманих від телескопів, НАСА може визначати хімічний склад атмосфер планет, віддалених на світлові роки, та робити припущення щодо їх придатності для життя.

4. Також є можливим прогнозування надзвичайних ситуацій: з використанням даних від численних сенсорів та супутників науковці можуть передбачати можливість землетрусів чи природних катаклізмів, а також моделювати поведінку людей під час надзвичайних ситуацій, що підвищує шанси на виживання.

5. Для запобігання злочинам: за допомогою нових технологій та аналізу обширних даних можливо автоматично виявляти фінансові махінації і відмивання грошей.

6. У торгівлі: торгові мережі використовують дані для випуску нових продуктів та впроваджують глобальні маркетингові кампанії для окремих сегментів покупців.

Хоча використання Big Data неперебрно розширює можливості аналізу, це із собою це також несе певні ризики в конфіденційності користувачів, оскільки збір великої кількості особистих даних може порушити приватність і дати корпораціям можливість маніпулювати власниками цих даних. Недостатня захищеність даних може призвести до витоку інформації. Також висока залежність від технологій може призвести до труднощів у випадку системних збоїв і створює загрозу стійкості та доступності даних.

Отже, використання технологій Big Data визначається як важливий та перспективний напрям у різних галузях. Необхідність ретельного врахування етичних аспектів та забезпечення надійного захисту конфіденційності даних є невід'ємною умовою для досягнення максимального ефекту від застосування цих технологій. Одночасно це дослідження висвітлює необхідність усвідомлення ризиків, пов'язаних із використанням Big Data, зокрема порушення приватності, можливості маніпуляцій та недостатньої захищеності даних від кіберзагроз. Спрямоване на розширення розуміння можливостей та викликів, це дослідження стимулює подальший розвиток і ефективне впровадження технологій Big Data у різноманітних сферах, враховуючи всі аспекти їх використання.

Список використаних джерел

1. Що таке Big Data?: вебсайт. URL: https://hub.kyivstar.ua/articles/shho-take-big-data (дата звернення: 21.11.2023).

2. Що таке Big Data і як це працює: вебсайт. URL: https://qagroup.com.ua/publications/shcho-take-big-data-i-iak-tce-pratciuie/ (дата звернення: 21.11.2023).

3. Технології Big Data: ключові характеристики, особливості та переваги: вебсайт. URL: https://cutt.ly/YwU8lX0m (дата звернення: 21.11.2023).

4. Що таке Big Data: все що вам слід знати про великі дані?: вебсайт. URL: https://cutt.ly/gwU8uYRS (дата звернення: 21.11.2023).

5. Степанюк О. С., Січко Т. В. Особливості використання реляційних та нереляційних баз даних в Big Data. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2020. С. 103–106.

6. Ткачук Н. О., Січко Т. В. Застосування Від Data у бізнесі. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2022. С. 224–226.

УДК 004.67

Сніжинський М. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки СО «Магістр»,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

СИСТЕМА ОБРОБКИ ТА АНАЛІЗУ ДАНИХ У МЕДИЧНІЙ СФЕРІ НА ОСНОВІ ТЕХНОЛОГІЙ BIG DATA

Донецький національний університет імені Василя Стуса, м. Вінниця

Останніми роками відбулось значне збільшення обсягів даних у сфері охорони здоров'я завдяки впровадженню цифрових технологій, як-от: електронні медичні картки, медична візуалізація та геноміка. Ці дані включають інформацію про пацієнтів, їхні медичні історії, діагностичні результати та лікування. Донедавна обробка даних у сфері охорони здоров'я відбувалася з великими труднощами через їх обсяг та різноманітність. Використання технологій великих даних, як-от машинне навчання, штучний інтелект і розподілені обчислення, відкриває багатообіцяючі перспективи. Впровадження масштабованої та захищеної інфраструктури разом із передовими алгоритмами уможливлює ефективну обробку даних, розпізнавання образів, прогностичне моделювання та аналіз результатів. До того ж включення інтероперабельних систем і стандартизованих протоколів сприяє безперешкодному обміну даними і співпраці між медичними установами. Зміна парадигми охорони здоров'я на основі даних зумовлює необхідність створення комплексної системи обробки й аналізу великих обсягів медичних даних із використанням можливостей технологій великих даних.

Усе більший обсяг і різноманітність медичних даних, що включають клінічні записи, дані візуалізації, омічні дані, дані з натільних пристроїв тощо, становлять великий виклик з погляду інтеграції, зберігання та аналізу даних. Гетерогенна природа цих наборів даних впливає на безперешкодне вилучення інформації, обмежуючи потенціал для трансформаційних ідей і відкриття нових знань. Для вирішення цих проблем пропонується використовувати технології великих даних як фундаментальне рішення. Впровадження надійних конвеєрів обробки даних, застосування алгоритмів машинного навчання та використання масштабованої інфраструктури уможливлюють ефективну агрегацію, попередню обробку та аналіз даних. До того ж застосування предиктивного моделювання і систем підтримки прийняття рішень на основі даних надає лікарям практичну інформацію для персоналізованого догляду за пацієнтами та стратегій лікування.

Основні компоненти, які можуть бути частиною системи обробки та аналізу даних у медичній сфері на основі технологій Big Data:

1. Модуль первинної обробки даних. Цей модуль є API та призначений для збору та обробки інформації у вигляд, зрозумілий системі. Неструктуровані дані часто містять шум, невідповідності, скорочення, орфографічні помилки та несуттєву інформацію, що впливає на точність і надійність аналізу. Очищення, попередня обробка та нормалізація цих даних зі збереженням важливого клінічного контексту є важливим процесом.

2. Модуль для аналізу даних та пошуку асоціативних правил. Цей компонент призначений для видобутку потрібних даних із баз даних та їх аналізу за певними наборами параметрів. Для цього можна використати алгоритм Apriori. Алгоритм Apriori є одним із популярних алгоритмів, який використовується для створення булевих асоціативних правил і видобутку наборів елементів, що часто зустрічаються. Він працює у циклічному ієрархічному порядку пошуку, де важливою властивістю є те, що всі дочірні набори елементів, які не є порожніми, повинні бути також частими.

3. Модуль для збереження даних. Розробка надійного конвеєра збору даних, здатного обробляти безперервні потоки даних із різних джерел, має важливе значення. Цей конвеєр забезпечує безперебійний і надійний потік даних до системи баз даних. Він може включати процеси ETL (Extract, Transform, Load), потокове передавання даних у реальному часі та процедури нормалізації даних для підтримки якості й узгодженості даних. Модуль для збереження даних у системі, орієнтованій на обробку та аналіз медичних даних, повинен мати пріоритети масштабованості, безпеки, сумісності з різними форматами даних і безперешкодної інтеграції з аналітичними інструментами, дотримуючись водночас нормативних стандартів, щоб забезпечити безпечне та ефективне управління великими й різноманітними наборами медичних даних.

4. Системи контролю доступу та автентифікації. Управління доступом користувачів до конфіденційних медичних даних: з допомогою надійних механіз-

мів автентифікації. Шифрування та маскування даних: захист конфіденційних даних з допомогою методів шифрування та маскування. Відповідність нормативним вимогам та інструменти регулювання: забезпечення відповідності нормативним вимогам охорони здоров'я (HIPAA, GDPR тощо), які стосуються обробки та конфіденційності даних.

5. Системи управління робочими процесами. Оркестрування складних потоків даних і автоматизація рутинних завдань. Автоматизація завдань обробки даних, аналізу та обслуговування системи.

Вирішуючи проблеми, пов'язані з управлінням великими і різноманітними медичними даними, використовуючи передові технології, забезпечуючи безпеку даних і сприяючи співпраці, запропонована система має на меті змінити ландшафт охорони здоров'я, що в кінцевому підсумку призведе до поліпшення результатів лікування пацієнтів і прогресу в медичній науці та практиці.

Список використаних джерел

1. The role of big data in medicine. URL: https://www.mckinsey.com/industries/life-sciences/our-insights/the-role-of-big-data-in-medicine

2. 24 Examples Of Big Data Analytics In Healthcare That Can Save People. URL: https://www.datapine.com/blog/big-data-examples-in-healthcare/

3. A Review of the Role and Challenges of Big Data in Healthcare Informatics and Analytics. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9536942/

4. Top 10 Challenges of Big Data Analytics in Healthcare. URL: https://healthitanalytics.com/news/top-10-challenges-of-big-data-analytics-in-healthcare

5. Степанюк О. С., Січко Т. В. Особливості використання реляційних та нереляційних баз даних в Big Data. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2020. С. 103–106.

УДК 004.65

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Труханська В. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ВИКОРИСТАННЯ БАЗ ДАНИХ В ІНФОРМАЦІЙНИХ СИСТЕМАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Одним із поширених видів програмного забезпечення, призначеного для обробки та зберігання інформації, є інформаційні системи. Вони відрізняються

функціональними можливостями, підтримкою багатокористувацького режиму, а також ціною та іншими параметрами. Ключовим складником будь-якої інформаційної системи є система управління базою даних та сама база даних.

Під базою даних розуміється набір інформації, який систематично зберігається у впорядкованому форматі на зовнішніх або внутрішніх носіях. Організація даних відбувається відповідно до конкретних правил, що визначаються моделлю даних. Зазвичай у базі даних міститься інформація, яка стосується певної предметної галузі і представлена в різних форматах даних.

Під терміном СУБД розуміється комплекс програмних і мовних засобів, спрямованих на управління даними в базі даних, створення та обслуговування бази даних, забезпечення взаємодії з програмами та обробку службової інформації. До того ж СУБД дають змогу отримувати різноманітні дані з бази даних та подавати їх у зручній для нас формі.

Незважаючи на все більшу популярність об'єктно-орієнтованих та NoSQL-баз даних, реляційні бази даних (наприклад, Oracle, MS SQL Server, MySQL) залишаються найбільш поширеними. У реляційних базах даних інформація організована у вигляді одного чи декількох взаємозв'язаних двовимірних таблиць із фіксованою кількістю стовпців та змінною кількістю записів, де зберігаються значення.

Головне завдання СУБД полягає у керуванні та обробці бази даних, де зберігається інформація користувача та системна інформація. Важливо зазначити, що можливості інформаційної системи напряму залежать від структури даних, що зберігаються в базі даних. Вибір конкретної СУБД та бази даних для інформаційної системи обумовлюється її функціональними можливостями, підтримкою багатокористувацького режиму, кількістю користувачів та їх рівнем інформаційної підготовки.

Локальні системи управління базами даних (СУБД) переважно використовуються для невеликих інформаційних систем, які функціонують та розташовані на одному комп'ютері або в єдиній внутрішній мережі. Для корпоративних систем або інформаційних систем підприємств найбільш поширені розподілені бази даних та системи управління базами даних, як-от MySQL, Oracle, MS SQL Server. Ці системи забезпечують багатокористувацький режим роботи з даними та службовою інформацією.

Розподілені системи управління базами даних (СУБД) зазвичай функціонують за допомогою технології «клієнт-сервер». У таких інформаційних системах програма-клієнт генерує текстові запити, створені мовою SQL, та відсилає їх на сервер для отримання необхідної інформації. Сервер обробляє ці запити і повертає лише необхідні дані у вигляді табличної або службової інформації. За необхідності змін у базі даних відбувається відправка відповідних запитів до сервера. У такий спосіб обмін даними через мережу ґрунтується переважно

на текстових запитах, які мають відносно невеликий обсяг, сприяючи ефективному використанню мережевих ресурсів. Це відомо як дволанкова архітектура технології «клієнт-сервер».

У межах архітектури «клієнт-сервер» для розвантаження робочих станцій клієнтів та зниження навантаження на мережу використовується триланкова архітектура. В цій схемі, окрім клієнтської частини та сервера бази даних, застосовується також проміжний сервер додатків. Із боку клієнта виконуються лише інтерфейсні дії, водночас уся логіка обробки інформації утримується на сервері додатків. У підсумку, найпопулярнішими базами даних, які застосовують для розробки інформаційних систем, є розподілені реляційні бази даних, які побудовані на технології «клієнт-сервер».

Використання баз даних стало ключовим компонентом у забезпеченні ефективної роботи та зберігання великих обсягів інформації. Воно допомагає оптимізувати процеси обробки, зберігання та аналізу інформації, сприяє забезпеченню доступу до неї у відповідний час. Застосування новітніх технологій у розробці баз даних дає змогу підвищити швидкість обробки даних та забезпечити їх безпеку.

Розвиток баз даних відкриває безліч можливостей у багатьох сферах – від бізнесу до наукових досліджень. Їх значення стає вирішальним у світі, де обмін та аналіз даних є основними аспектами. Оптимізація та розвиток нових технологій баз даних забезпечує не лише збереження інформації, а й сприяє її безпеці, доступності та обробці.

Список використаних джерел

1. СУБД. URL: https://highload.today/uk/subd-yaki-buvayut-yak-vibrati/ (дата звернення: 12.11.2023).

2. Реляційна база даних.URL: https://ua5.org/database/189-reljacjjna-baza-danikh.html (дата звернення: 13.11.2023).

3. Databases in information technologies. URL: https://www.researchgate.net/publication/325721258_The_Application_of_Database_Technology_in_Information_Society_and_its_Existing_Problems (дата звернення: 12.11.2023).

УДК 004.62:004.65:004.75

Шафорост В. В., Корнієнко К. К., Хмель А. Є., здобувачі 4 курсу спеціальності 122 Комп'ютерні науки,

Комаров В. Ф., канд. техн. наук, старший викладач кафедри інформаційних технологій

МОДЕЛЮВАННЯ ВЕЛИКИХ НАБОРІВ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

В епоху цифрових технологій та інтернету ми спостерігаємо значне збільшення обсягу та доступності різнобічної інформації. Вона надходить з багатьох джерел, як-от комп'ютери, мобільні пристрої, датчики чи соціальні медіа, які створюють структуровані, напівструктуровані або неструктуровані дані безперервно. Обсяг даних, із якими ми стикаємося, зріс до терабайтів або петабайтів і продовжує збільшуватися далі, а типи даних, які генеруються програмами, стають різноманітнішими, ніж раніше. Внаслідок цього традиційні реляційні бази даних стикаються з проблемами збору, зберігання, пошуку, обміну, аналізу та візуалізації даних. Багато IT-компаній намагаються розв'язувати означені задачі, використовуючи бази даних (БД) типу NoSQL, як-от Cassandra або HBase, та розподілені обчислювальні системи, як-от Hadoop [1]. Бази даних NoSQL зазвичай є нереляційними, розподіленими, горизонтально масштабованими та вільними від схем.

Моделювання даних – це процедура створення концептуального представлення об'єктів даних та їх асоціацій один з одним. Процес моделювання даних зазвичай включає кілька етапів, зокрема збір вимог, концептуальне проєктування, логічне проєктування, фізичне проєктування та впровадження. На кожному етапі розробники моделей даних співпрацюють з усіма учасниками процесу, щоб визначити вимоги до даних, визначити сутності та атрибути, встановити зв'язки між об'єктами даних і створити модель, яка точно відображає дані та може використовуватися програмою, розробниками, адміністраторами БД та іншими учасниками [2].

Традиційне моделювання даних зосереджене на вирішенні складних зв'язків між даними, що підтримують схему, однак воно не стосується нереляційних баз даних без схем. Старі способи моделювання даних більше не застосовуються – потрібна інша методологія керування великими даними.

Моделювання великих наборів даних містить дві термінології, а саме «Великі дані» (Big Data) та «Моделювання даних» (Data Modeling). Термін «великі дані» використовується для опису великих, складних і швидко збільшуваних

наборів даних, які мають численні, автономні та незалежні джерела[3]. Точний аналіз таких даних може мати значні переваги, оскільки він дає змогу розпізнавати закономірності та зв'язки в наборах даних [4].

Моделювання даних зазвичай виконується з допомогою кількох рівнів концептуалізації, зокрема: концептуального, логічного і фізичного.

Під час створення концептуального рівня визначається, що необхідно включити в конфігурацію моделі для опису та координації комерційних принципів. Він зосереджений насамперед на бізнес-записах, функціональності та комунікації. За його розробку насамперед відповідають архітектори даних і бізнес-стейкхолдери. Концептуальна модель даних використовується для визначення сфери застосування методу. Це інструмент для організації, визначення та візуалізації ідей компанії. Метою розробки обчислювальних моделей даних є розробка нових сутностей, зв'язків і атрибутів. Архітектори даних і зацікавлені сторони часто створюють обчислювальні моделі даних.

Логічна модель даних пояснює розташування структур даних і їх зв'язки. Вона допомагає включити додаткові дані в компоненти концептуальної моделі даних і закладає основу для побудови фізичної моделі. Ця модель дає нам змогу перевіряти та оновлювати інформацію про використання зв'язків, які були встановлені раніше. Логічна модель даних налаштовується і створюється окремо від системи керування базами даних (СКБД). Вона визначає дані, необхідні для проєкту, але взаємодіє з іншими логічними моделями даних відповідно до складності проєкту.

Фізична модель пояснює, як модель даних реалізується в базі даних, як використовувати систему керування базою даних для виконання моделі даних. Вона описує процес з погляду таблиць, операцій створення, читання, оновлення і вилучення (CRUD), індексів, розділення тощо. Фізичний рівень передбачає створення конкретних деталей про те, як дані будуть зберігатися, зокрема типи даних, індекси та іншу технічну інформацію.

У такий спосіб моделювання даних допомагає підвищити узгодженість іменування, правил та безпеки. Це покращує аналітику даних. Акцент робиться на необхідності наявності та організації даних незалежно від способу їх застосування [5].

Проте моделювання великих даних – це не лише про розуміння складних даних, але й про те, як ці дані можуть бути використані для підвищення ефективності та продуктивності роботи бази даних. Використовуючи різні моделі даних, ми можемо виявити закономірності, тренди та зв'язки, які можуть бути приховані в сировинних даних. Щоб використовувати великі дані повною мірою, ми повинні моделювати їх із допомогою різних типів моделей даних, як-от ієрархічна модель, мережева модель тощо.

Наприклад, ієрархічна модель використовує деревоподібну структуру для організації даних. Кожен запис у такій моделі має один корінь, який з'єднує всі дані. Цей корінь розширюється, як гілка, з'єднуючи батьківські вузли з відповідними дочірніми вузлами, причому кожен дочірній вузол має лише однин батьківський вузол. У цій моделі дані організовано в реляційну систему з кореляцією один-до-багатьох, що дає змогу легко знаходити і маніпулювати певними типами даних. Якщо йдеться про однорідні документи, вони впорядковані особливим способом – це фізичний порядок, у якому зберігається інформація. Метод можна застосувати до різних взаємозв'язків моделі в реальному часі.

Ієрархічна модель може бути обмеженою, коли потрібно представити більш складні відносини між даними, які не можуть бути чітко виражені в ієрархічній структурі. Незважаючи на це, ієрархічна модель даних залишається важливим інструментом для організації та обробки великих обсягів даних.

Мережева модель даних є гнучкою альтернативою ієрархічній моделі. Вона дає змогу кожному дочірньому вузлу мати кілька батьківських вузлів, у такий спосіб створюючи багатофакторну мережеву структуру, яка допомагає показати більш складні зв'язки між об'єктами. Наприклад, у базі даних компанії співробітники можуть мати різні ролі в різних проєктах, а мережева модель даних дасть змогу кожному співробітнику бути пов'язаним із різними проєктами в різних ролях.

Окрім зазначених моделей треба також згадати й інші [2]: модель ER (Entity-Relationship), реляційну, об'єктно-орієнтовану та об'єктно-реляційну моделі.

Незважаючи на численні переваги, моделювання даних у контексті великих наборів даних має обмеження та проблеми. Моделі даних можуть бути негнучкими, що ускладнює адаптацію до мінливих вимог або структур даних. Моделі даних можуть бути складними та складними для розуміння, через що може бути важко вносити дані чи ефективно взаємодіяти з ними. Моделювання даних може бути трудомістким процесом, особливо для великих або складних наборів даних.

Отже, складність реляційної бази даних обмежує масштабованість зберігання даних, проте дає змогу легко запитувати дані через механізм SQL. БД типу NoSQL мають протилежні властивості – необмежену масштабованість із більш обмеженими можливостями запитів. А великі дані потребують одночасно легкої масштабованості та можливості легко надсилати запити на дані.

У майбутньому для великих наборів даних будуть з'являтись і поширюватись нові гібридні системи, що поєднуватимуть атрибути обох підходів, втім розробка моделей та моделювання даних залишаться трудомістким, але критично важливим процесом у розробці програмного забезпечення або систем баз даних.

Список використаних джерел

1. Data Modeling for Big Data. URL: https://citeseerx.ist.psu.edu/ document? repid=rep1&type=pdf&doi=6779310266a5f15a6e85fec3cbb3f37170a40c16#page=77

2. What is Data Modelling? Overview, Basic Concepts, and Types in Detail. URL: https://www.simplilearn.com/what-is-data-modeling-article

3. Gil D., Song I.-Y. Modeling and Management of Big Data: Challenges and opportunities. *Future Generation Computer Systems, Elsevier BV*. 2016. № 63. P. 96–99. DOI: 10.1016/j.future.2015.07.019.

4. Ribeiro A., Silva A., da Silva A. R. Data Modeling and Data Analytics: A Survey from a Big Data Perspective. *Journal of Software Engineering and Applications, Scientific Research Publishing, Inc.* 2015. № 08. P. 617–634. DOI: 10.4236/jsea.2015.812058.

5. Big Data Modeling. URL: https://hkrtrainings.com/big-data-modeling

УДК 004.021:004.67

Ватаманеску С. К., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ АЛГОРИТМІВ ДЛЯ ВЕЛИКИХ ОБСЯГІВ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному інформаційному суспільстві обробка та аналіз великих обсягів даних стали важливим завданням у багатьох галузях, як-от бізнес, наука та медицина. Збільшення обсягів даних виникає через використання технологій, як-от Інтернет речей та соціальні мережі. Однак це також призводить до викликів для алгоритмів обробки, що потребують постійної оптимізації.

Дослідження оптимізації алгоритмів для великих обсягів даних вимагає огляду принципів роботи з такими обсягами інформації. Застосування паралельного програмування та розподілених обчислень разом з ефективним використанням ресурсів визначає успішну обробку даних у реальному часі.

В опрацьованій літературі виділено ключові принципи, як-от паралельне програмування та розподілені обчислення для оптимальної обробки великих обсягів даних. Розподілені системи зберігання даних, як-от Apache Hadoop та Apache Spark, а також техніки індексації та кешування, використовуються для покращення швидкодії обробки даних.

Дослідницька спільнота активно досліджує різні підходи до оптимізації алгоритмів. Розв'язки включають використання розподіленої системи зберігання

даних та технік індексації. Аналіз свідчить про необхідність розуміння впливу різних типів даних на ефективність алгоритмів. Збільшення обсягів даних створює проблеми, що ставлять під сумнів ефективність традиційних алгоритмів. Зростання часу обробки даних та нестабільність алгоритмів під час роботи з великим обсягом даних є ключовими аспектами. Також виникає потреба у масштабованості алгоритмів та їх оптимізації для розподілених систем. Збільшення обсягів даних породжує виклики для традиційних алгоритмів, що може призвести до затримок і втрат точності. Вимоги до масштабованості є викликами для розподілених систем, вимагаючи ефективного управління ресурсами та забезпечення консистентності даних.

Для оптимізації алгоритмів важливе використання паралельного програмування та розподілених обчислень. Кешування, індексація та техніки сортування даних відіграють ключову роль у забезпеченні швидкодії й ефективності обробки великих обсягів інформації [1].

У розподілених системах важливо вирішувати проблеми консистентності даних та ефективно використовувати мережеві ресурси. Паралельне програмування, розподілене обчислення та оптимізація завдань відіграють вирішальну роль у досягненні масштабованості алгоритмів. Кешування допомагає уникнути повторних обчислень, а індексація та сортування сприяють швидкому доступу до даних. Компресія даних зменшує їх об’єм, полегшуючи обробку та передачу.

Інтенсивне вивчення питань оптимізації алгоритмів для великих обсягів даних передбачає вирішення проблем часу обробки, стабільності та масштабованості. Активні дослідження включають використання розподілених систем, паралельного програмування та ефективних стратегій управління ресурсами. Оптимізація алгоритмів суттєво впливає на швидкість обробки даних у різних інформаційних системах. Ця тема досліджує конкретні застосування оптимізованих алгоритмів та їх вплив на інформаційні системи.

Швидкодія додатків та сервісів. Оптимізація алгоритмів покращує швидкість додатків та онлайн-сервісів. Наприклад, оптимізовані алгоритми пошуку допомагають отримувати дані швидше, покращуючи відгук системи. Ефективність обробки великих обсягів даних: у галузях, де важлива обробка великих обсягів даних (фінанси, медицина, аналітика), оптимізовані алгоритми прискорюють аналіз та забезпечують ефективність роботи.

Підвищення продуктивності баз даних. Оптимізовані алгоритми зменшують час виконання запитів та операцій, поліпшуючи доступ до даних та продуктивність баз даних.

Оптимізація обчислень у хмарних сервісах. У хмарних обчисленнях оптимізовані алгоритми максимально використовують обчислювальні потужності, економлячи ресурси. У сферах обробки великих даних та машинного навчання оптимізовані алгоритми вирішальні. Розглянемо їх конкретні застосування та вплив [2]:

1. Аналіз великих обсягів даних. Оптимізовані алгоритми допомагають ефективно аналізувати та витягати цінні інсайти, що корисно у фінансах, маркетингу та наукових дослідженнях.

2. Оптимізація моделей машинного навчання. У машинному навчанні швидкі та ефективні алгоритми прискорюють тренування та класифікацію даних.

3. Реальний час та обробка стрімінгових даних. Оптимізовані алгоритми дають змогу ефективно обробляти стрімінгові дані в режимі реального часу.

4. Вдосконалення рекомендаційних систем. У електронній комерції і сервісах стримінгу оптимізовані алгоритми забезпечують точні та швидкі рекомендації.

У розвитку квантових обчислень розглядаються такі напрями [3]:

1. Розробка ефективних квантових алгоритмів оптимізації: створення алгоритмів, які використовують переваги квантових обчислень для оптимізації завдань.

2. Використання квантових машин для великих обсягів даних: у дослідженнях можливостей квантових обчислень для швидкої обробки та аналізу великих обсягів даних.

3. Розширення квантових алгоритмів у сферах машинного навчання: вивчення та розвиток квантових алгоритмів для задач машинного навчання.

4. Створення квантових алгоритмів забезпечення безпеки дослідження використання квантових алгоритмів у криптографії для забезпечення безпеки інформації.

У сфері розподілених та паралельних алгоритмів для великих обсягів даних актуальні такі напрями [4]:

1. Розробка розподілених алгоритмів обробки стрімінгових даних: створення алгоритмів для ефективної обробки стрімінгових даних у розподіленому середовищі.

2. Паралельна обробка графових структур: розробка алгоритмів для паралельної обробки та аналізу графових структур.

3. Оптимізація алгоритмів для масштабованих баз даних: вивчення методів оптимізації алгоритмів для масштабованих баз даних.

4. Створення адаптивних розподілених алгоритмів: розробка методів, які змінюють стратегію залежно від обсягу даних та умов обчислень.

У сучасному інформаційному суспільстві великі обсяги даних – норма. Розвиток та оптимізація алгоритмів стають визначальними для ефективності обчислень та швидкості обробки інформації. Застосування новітніх алгоритмів може покращити ефективність у сферах, де обробка та аналіз даних є ключовими компонентами успіху.

Подальші дослідження в цій галузі будуть визначати стандарти та відкривати можливості для розвитку інноваційних технологій і застосувань, що впливають на різні сфери нашого суспільства.

Список використаних джерел

1. Вступ до алгоритмів / Т. Кормен, Ч. Лейзерсон, Р. Рівест, К. Штайн; пер. з англ. Київ: К.І.С., 2019. 1288 с.
2. Mayer-Schönberger V., Cukier K. Big Data: A Revolution That Will Transform How We Live, Work, and Think Paperback. Harper Business, 2014. 272 p.
3. Rieffel E. G., Polak W. H. Quantum Computing. MIT Press Ltd, 2014. 392 p.
4. Leopold C. Parallel and Distributed Computing: A Survey of Models, Paradigms and Approaches. Wiley-Interscience, 2020. 272 p.

УДК 004.65

Калько Д. Р., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

ЗАСТОСУВАННЯ ГРАФОВОЇ БАЗИ ДАНИХ У СОЦІАЛЬНИХ МЕРЕЖАХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Соціальні мережі Facebook, Twitter і LinkedIn стали невід'ємною частиною нашого життя. Ці мережі дають змогу нам спілкуватися з друзями, родиною, колегами, брендами, знаменитостями тощо. За своєю суттю соціальні мережі можна моделювати як графіки, де користувачі представлені як вузли, а їхні зв'язки – як ребра [1]. Ця графоподібна структура робить бази даних графів природними для зберігання та аналізу даних соціальних мереж. Соціальні мережі також використовуються компаніями для зв'язку з клієнтами, просування продуктів і послуг, отримання інформації про поведінку клієнтів.

Актуальність. Соціальні мережі набувають все більшої популярності та містять величезні обсяги даних про зв'язки між користувачами, і традиційні реляційні бази даних уже не дуже ефективні для моделювання та аналізу таких складних зв'язків [2]. Графові бази даних краще підходять для представлення та обробки даних соціальних мереж через їх здатність працювати зі складними зв'язками як з графом. Це дає змогу покращити функціонування та аналітику соціальних мереж із допомогою графових баз даних.

Аналіз останніх досліджень. Деякі останні дослідження показали переваги використання графових баз даних для соціальних мереж порівняно з реляційними базами даних. Проте ще не до кінця вивчені особливості їх використання для конкретних випадків, як-от побудова стрічок новин, рекомендаційні системи тощо [3].

Мета дослідження – проаналізувати методи застосування графових баз даних для моделювання та аналізу даних соціальних мереж, на прикладах конкретних задач, як-от побудова стрічок новин, рекомендаційні системи тощо.

Постановка завдання. Завдання під час виконання цієї роботи – дослідити підходи до представлення даних соціальних мереж у вигляді графів; розглянути, як за допомогою запитів до графових баз даних можна реалізувати різні функції соціальних мереж; проаналізувати особливості використання графових баз даних, порівняно з реляційними, на прикладах конкретних задач; сформулювати переваги графового підходу та рекомендації щодо вибору графової чи реляційної бази даних для певного випадку.

Соціальний граф представляє зв'язки між користувачами в мережі. Кожен користувач моделюється як вузол, а його зв'язки з іншими користувачами є ребрами [1]. У графовій базі даних ми можемо зберігати облікові записи користувачів як вузли з такими властивостями: ім'я, місцезнаходження, інтереси тощо. Дружні зв'язки моделюються як межі між вузлами користувачів. Членство в групах також може бути представлено шляхом підключення вузлів користувача до вузлів групи. Це дає змогу відобразити всю соціальну структуру у схемі бази даних. Моделюючи соціальні дані у вигляді графіка, бази даних графів допомагають ефективно обходити зв'язки [2]. Ми можемо легко отримати список друзів користувача, груп, до яких він належить, спільних друзів іншого користувача тощо з допомогою простого обходу графу. З допомогою графових запитів можна знайти зв'язки та закономірності в соціальному графі, щоб рекомендувати друзів, ідентифікувати впливових людей і спільноти тощо.

Основною функцією соціальних додатків є стрічка новин, де користувачі бачать оновлення від друзів і груп. Моделювання каналів як потоків активності, приєднаних до вузлів користувачів дає змогу ефективно створювати персоналізовані канали [3]. Коли користувач виконує певну дію, наприклад, публікує оновлення, коментує публікацію тощо, це може зберігатися в його потоці активності. Щоб створити стрічку новин користувача, ми просто переглядаємо зв'язки його друзів і збираємо оновлення з їхніх потоків.

Структура графіка дає змогу легко застосовувати фільтри, наприклад, спочатку показувати дописи від близьких друзів або приховувати оновлення від знайомих. Канали також можна персоналізувати на основі інтересів, відстежуючи теми, пов'язані з публікаціями [3]. Завдяки графовій базі даних, що керує каналами та зв'язками, генерувати релевантні налаштовані канали новин у масштабі просто.

Ідентифікація спільнот важлива для аналітики та реклами в соціальних мережах. Алгоритми графів можна запускати на графі соціальної мережі, щоб виявити щільні кластери, які представляють спільноти [2]. Це можуть бути кола

друзів, групи випускників коледжу, шанувальники музики тощо. Розуміння цих спільнот може допомогти націлити оголошення чи рекомендації.

Наприклад, кілька користувачів можуть утворити взаємопов'язаний кластер навколо спільного інтересу – фотографії. Цю спільноту можна виявити з допомогою алгоритма, підрахунку трикутників. Після визначення цільове просування фотопродуктів або послуг може бути спрямоване на цю групу. Бази даних графів забезпечують власну підтримку для запуску таких складних алгоритмів графів безпосередньо в наборі даних соціальної мережі [3].

Використання графових баз даних для соціальних мереж має кілька переваг:

1. Ефективне зберігання та запити зв'язків: бази даних Graph можуть ефективно зберігати та запитувати складні зв'язки між користувачами [1]. Це пояснюється тим, що вони зберігають зв'язки безпосередньо як ребра на графах, а не нормалізують їх у таблиці, як у реляційній базі даних.

2. Оновлення в реальному часі: можуть підтримувати оновлення графів у реальному часі, що важливо для соціальних мереж, де користувачі постійно взаємодіють один з одним [3].

3. Масштабованість: бази даних можна масштабувати для роботи з великими соціальними мережами з мільйонами або навіть мільярдами вузлів і ребер [2].

4. Гнучкість: вони гнучкі та можуть використовуватися для моделювання різноманітних програм соціальних мереж [3].

Висновки. Отже, бази даних графів забезпечують гнучкий, масштабований спосіб моделювання даних соціальних мереж на основі зв'язків. Їх можливості, орієнтовані на графи, забезпечують основні функції: персоналізовані стрічки новин, рекомендації друзів і виявлення спільноти. Оскільки соціальні мережі стають все більшими та складнішими, бази даних графів залишатимуться ідеальною платформою для зберігання та аналізу даних соціальних графів. Графічні бази даних все ще є відносно новою технологією, але вони можуть революціонізувати спосіб створення та управління соціальними мережами. У майбутньому ми можемо очікувати, що бази даних графів будуть використовуватися для забезпечення навіть більш складних і витончених програм соціальних мереж.

Список використаних джерел

1. Робінсон І., Веббер Дж., Ейфрем Е. Графові бази даних. 2-е вид. O'Reilly Media. 2016. 256 с.

2. Графова база даних. URL: https://cutt.ly/dw5yaWIQ

3. MS SQL Server: реалізація підтримки графової моделі даних. URL: https://dou.ua/forums/topic/31431/

УДК 004.627

Козачок А. О., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

СТИСНЕННЯ ДАНИХ ТА ЇХ ВПЛИВ НА ШВИДКОДІЮ АЛГОРИТМІВ ОБРОБКИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Зі збільшенням обсягів інформації, що обробляється в сучасному суспільстві, виникає потреба в оптимізації ефективного управління та обробки даних. Важливим аспектом цього завдання є методи стиснення даних, спрямовані на зменшення обсягу інформації без втрати її значимості.

Методи стиснення даних стали невід'ємною частиною розвитку цілої низки технологій – від зберігання і передачі даних до обчислень у реальному часі. Ці методи не лише економлять простір і ресурси, але й суттєво впливають на швидкість роботи алгоритмів, що є важливим параметром у сучасних обчислювальних середовищах [1].

Алгоритми стиснення зазвичай використовуються для зменшення розміру файла без видалення інформації. Це може збільшити їх ентропію та зробити файли більш випадковими, оскільки всі можливі байти стають більш загальними. Алгоритми стиснення також можуть бути корисними, коли вони використовуються для створення імітації через запуск функцій стиснення у зворотному порядку. Стиснення даних становить велику цікавість для всіх, хто хоче приховати дані з чотирьох причин: менше даних легше обробляти; стислі дані зазвичай більші; реверсивне стиснення може імітувати дані. Сьогодні використовується низка методів стиснення даних. За останні кілька років ця сфера надзвичайно розширилася через велику економічну цінність таких алгоритмів. Одними з найбільш популярних методів є ймовірнісні, словникові методи, кодування довжини серії, хвильові методи, фрактальні методи та адаптивні схеми стиснення. Усі ці схеми стиснення корисні в окремих доменах [2].

Поширені методи стиснення даних видаляють і замінюють повторювані елементи даних та символи. Елементи даних і символи видаляються та замінюються, щоб зменшити розмір даних. Дублюючі елементи (тобто надмірність), замінюючи ці послідовності коротшими значеннями способом заміни повторюваних елементів (тобто надлишкових) на коротші значення. Стиснення ненадлишкових даних (тобто дані без надлишковості, наприклад, випадкові сигнали або шум) не можна стиснути. Стиснення зашифрованої інформації також зазвичай неможливе.

Існує два типи стиснення даних.

1. Стиснення без втрат: оригінальні дані можна відновити без спотворень.

2. Стиснення зі втратами: стиснення під час якого дані можуть бути відновлені з невеликими (іноді значними) спотвореннями. Ці методи можна застосовувати тільки для таких типів даних, для яких втрата частини вмісту не приводить до суттєвого спотворення інформації. До таких типів даних належать відео- та аудіо-, графічні дані.

Існує багато різних практичних методів стиснення без втрати інформації, які, як правило, мають різну ефективність для різних типів даних та різних обсягів. Однак в основі цих методів лежать три теоретичних алгоритми [3]:

- RLE (Run Length Encoding) алгоритм;
- групи LZW алгоритми;
- Хафмана алгоритми.

Алгоритм RLE (Run Length Encoding) – один з найстаріших і найпростіших алгоритмів стиснення графіків. Він розбиває зображення на байтовий ланцюжок уздовж растрових ліній. Простіше кажучи, алгоритм замінює однакову послідовність символів на відповідні пари «число-значення». Наприклад, рядок «AAAABBBBGGG» перетворюється на «4A4B3G». Основне застосування цього алгоритму – для зображень з невеликою кількістю кольорів. Це стосується, наприклад, спеціальних або наукових зображень. Однак він не набув широкого поширення через свою простоту, швидкість і низьку ефективність, незважаючи на дуже низьке використання пам'яті під час операцій архівування та розпакування.

Алгоритм LZW названо на честь його розробників Лампела, Зів і Велча; на відміну від RLE, цей алгоритм базується на тому самому ланцюжку байтів. Алгоритм стиснення можна описати так. Спочатку послідовно зчитуються всі символи з вхідних даних і перевіряється таблиця рядків (яка буде згенерована) на предмет існування такого рядка. Якщо відповідний рядок знайдено, процес зчитування продовжується (символічно), якщо такого рядка не існує, до таблиці додається 13-й рядок і процес повторюється. Основними цільовими зображеннями цього алгоритму є 8-розрядні зображення. Варто зазначити високу гнучкість цього алгоритму – у поширених архіваторах використовується декілька його варіантів.

Алгоритм Хаффмана вважається одним із класичних алгоритмів, відомих давно (з 60-х рр. XX ст.). Він базується виключно на частоті появи одного і того ж байта на зображенні. Алгоритм присвоює коротші бітові ланцюжки символам, які найчастіше зустрічаються у вхідному потоці. І навпаки, довші ланцюжки призначаються символам, які зустрічаються доволі рідко. Алгоритм вимагає двох проходів над вихідним зображенням. Варто зазначити, що алгоритм Хаффмана не використовується як окремий алгоритм. Насправді він є частиною більш склад-

ної схеми алгоритму багатоступеневого стиснення. Прикладом цього є формат даних TIFF, де алгоритм Хаффмана є лише одним із декількох кроків [4].

Отже, ефективне управління та обробка даних є важливим питанням у сучасному суспільстві, де різноманітної інформації багато і її обсяг постійно зростає. Технологія стиснення даних є важливим інструментом для оптимізації обробки великих масивів інформації та зменшення обсягу інформації без втрати її важливості. Було розглянуто кілька методів стиснення, зокрема RLE, групи KWE та алгоритм Хаффмана, кожен з яких має свої переваги та обмеження залежно від застосування.

Список використаних джерел

1. Salomon D. Data Compression: The Complete Reference. Springer London. 2007. DOI: 10.1007/978-1-84628-603-2.

2. Compression Algorithm. URL: https://www.sciencedirect.com/topics/ computer-science/compression-algorithm (дата звернення: 20.10.2023).

3. Основи інформаційних технологій. URL: https://informatics6.webnode.com.ua/zanyattya-10/ (дата звернення: 20.10.2023).

4. Олашин О. О. Інтелектуальна система покращення алгоритмів стиску зображення. URL: https://ela.kpi.ua/server/api/core/bitstreams/d2665c09-73e8-4319-8dc6-82bca5d989c8/content (дата звернення: 20.10.2023).

УДК 004.056.5:004.65

Коновалюк І. Л., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Зелінська О. В., канд. техн. наук, доцент, в. о. завідувача кафедри інформаційних технологій

ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ БАЗ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Кожен бізнес, корпорація чи державна установа має інформаційну базу, яка включає дані про клієнтів, нормативні акти, продукти та фінансову звітність. Великі об'єми інформації зазвичай містять конфіденційні, корпоративні та особисті дані. Несанкціонований доступ до цих даних може призвести до серйозних наслідків у фінансовій і репутаційній сферах.

Існують дві ключові причини, які зумовлюють необхідність для приватних компаній та державних установ вкладати значні ресурси у захист баз даних.

По-перше, це кіберзлочинність. Зловмисники постійно удосконалюють свої методи, використовують нові види програм-вимагачів та безфайлові методи

проникнення, що ставить під загрозу конфіденційність інформації. Тільки у 2019 р. було розкрито понад 9 мільярдів облікових записів із персональною інформацією. Спільно з розвитком злочинних технологій розробляються рішення для захисту конфіденційної інформації, як-от налаштування конфігурації брандмауера для обмеження доступу до підозрілого трафіка та впровадження процедур у разі порушення безпеки [1].

По-друге, існує проблема відповідності. Міжнародне законодавство щодо захисту персональної інформації стає дедалі суворішим, і відповідальність за дотримання конфіденційності даних покладається на організації. Нормативні вимоги можуть значно відрізнятися залежно від галузі та типу інформаційних активів. Забезпечення конкурентоспроможності вимагає від українських компаній великих витрат на захист баз даних [2].

Аспект безпеки даних є важливою частиною загальної стратегії захисту, яка включає в себе виявлення загроз, оцінку ризиків та їх зменшення для захисту конфіденційної інформації. Важливо відрізняти захист даних від безпеки баз даних, оскільки вони обидва вимагають активних та пасивних заходів відповідно. Захист баз даних включає в себе різноманітні методи, програмні засоби, процеси та технології, спрямовані на запобігання несанкціонованого доступу, модифікацій, випадкового розкриття та інших загроз. Політика конфіденційності стосується обробки та управління конфіденційною інформацією, зокрема особистою інформацією, даними кредитних карт та ін. конфіденційними даними.

Фундаментальна тріада, відома як конфіденційність, цілісність і доступність (CIA), є ключовими аспектами. Конфіденційність базується на принципі найменших привілеїв для запобігання несанкціонованого доступу. Цілісність спрямована на захист від неправомірного видалення чи зміни даних, а використання цифрових підписів є одним із методів гарантування цілісності. Доступність є ключовим елементом, який вимагає правильної роботи контролю, систем і програм для забезпечення доступності послуг та інформаційних систем за потреби [3].

Шифрування або криптографічний захист бази даних є одним з найбільш ефективних методів забезпечення безпеки БД. Алгоритм шифрування перетворює інформацію в незрозумілі символи з допомогою математичного процесу. Саме тоді, як інші інструменти безпеки захищають систему від вторгнень або атак, шифрування є фундаментальною формою, яка стосується безпеки самих даних. Навіть у разі злому системи інформація буде доступна для читання тільки авторизованим користувачам, які мають ключі шифрування.

Процес захисту бази даних неможливий без управління паролями, що має визначальне значення для підтримки безпеки. За цією стороною стратегії безпеки зазвичай стежать співробітники IT-підрозділу. Практика безпеки БД також

включає управління привілеями. Організації можуть зробити безліч різних кроків для управління паролями, наприклад, використовувати актуальні методи дво- або багатофакторної автентифікації, надавати користувачам обмежений час для введення облікових даних.

Розробка та забезпечення систем зберігання корпоративних даних є складним завданням, вирішення якого вимагає знаходження балансу між продуктивністю, доступністю і вартістю (рис. 1).

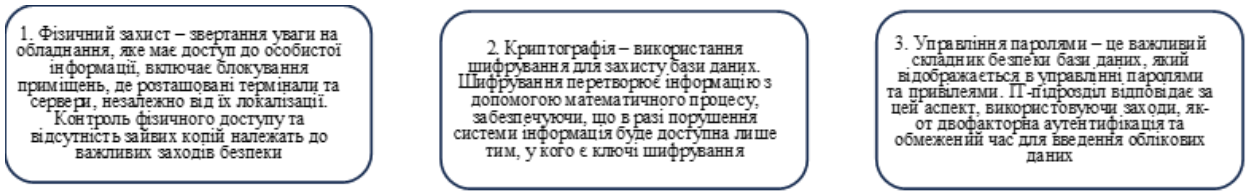


Рисунок 1. Системи зберігання корпоративних даних [4]

Можна зробити висновок, що використання лише одного методу не може гарантувати повного збереження даних. Тому для підвищення рівня безпеки інформації в базі даних рекомендується впровадження комплексних заходів. Розвиток у сфері безпеки баз даних є дуже важливим і потребує постійного вдосконалення. Проблема стала особливо актуальною під час російсько-української війни, коли для забезпечення державної безпеки були вжиті максимальні заходи, як-от закриття повного доступу до всіх державних реєстрів шляхом відокремлення їх від глобальної мережі.

Список використаних джерел

1. Касянчук Н. В., Ткачук Л. М. Захист інформації в базах даних. 2019. URL: https://conferences.vntu.edu.ua/index.php/all-fm/all-fm-2019/paper/download/7001/5715.

2. Системи управління базами даних. URL: http://rodak.if.ua/komptech/lection4.htm. (дата звернення: 28.11.2023).

3. Шифрування та захист баз даних. URL: https://iitd.com.ua/shifruvannj a-ta-zahistbaz-danih/ (дата звернення: 28.11.2023).

4. Як створити надійний пароль – рекомендації спеціалістів ESET. URL: https://eset.ua/ua/news/view/649/nadezhnyy-parol-sposoby-sozdaniya-parolya-otspetsialistov-eset

УДК 004.657

Лаптєва М. А., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

РІЗНИЦЯ МІЖ SQL І NOSQL: ПЕРЕВАГИ ТА ОБМЕЖЕННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Технологія баз даних значно розвинулася за останні десятиліття. З'явилися дві основні моделі баз даних: реляційна модель, яка використовує мову структурованих запитів (SQL), і нереляційні, або бази даних NoSQL. Обидві моделі мають різні архітектури, які надають переваги й обмеження для певних випадків зберігання та пошуку даних. Розглянемо ключові відмінності між системами баз даних SQL та NoSQL, а також висвітлимо сильні і слабкі сторони, притаманні реляційним системам керування базами даних (СКБД).

Бази даних SQL базуються на реляційній моделі, розробленій Е. Ф. Коддом у 1970 р. [1]. Дані структуруються в таблиці, що містять рядки і стовпці, з визначеними зв'язками між ними. Таблиці мають заздалегідь визначену схему, яка забезпечує сувору цілісність і стандартизацію даних. Сервери SQL надають розширені можливості управління даними, зокрема транзакції ACID (Atomicity, Consistency, Isolation, Durability), підтримку складних запитів та аналітики, а також власні програмні інтерфейси. Це забезпечує цілісність даних, але може не вистачати гнучкості та масштабованості NoSQL.

На противагу цьому, NoSQL охоплює різноманітні нетабличні та розподілені архітектури баз даних, які з'явилися на початку 2000-х рр. [2]. Найпоширеніші типи включають сховища ключ-значення, документи, графіки та сховища з широкими стовпцями. Гнучкі схеми, масштабованість між товарними серверами, висока доступність і відмовостійкість – основні переваги NoSQL. Структури даних і можливості запитів сильно відрізняються залежно від конкретної реалізації NoSQL.

Наприклад, база даних SQL моделює замовлення, використовуючи нормалізовані таблиці клієнтів, замовлень і продуктів, пов'язані ключами. Об'єднання збирають записи для транзакцій, а база даних NoSQL зберігає замовлення як окремі документи, проіндексовані за ідентифікаторами, без використання об'єднань [3]. Це полегшує ітеративне кодування, але не має жорстких схем.

Переваги реляційних баз даних:

1. Стандартизація формату даних та взаємозв'язків через фіксовані схеми таблиць сприяє цілісності, узгодженості та організації даних. Принципи нормалізації бази даних мінімізують надмірність даних.

2. Підтримка ACID-транзакцій дає змогу надійно обробляти операції створення, читання, оновлення та видалення в декількох таблицях і записах.

3. Розширена функціональність запитів та оптимізація продуктивності індексування полегшують створення складних аналітичних звітів і сценаріїв агрегації. SQL залишається основною мовою для запитів до баз даних.

4. Корпоративні можливості, як-от безпека на основі ролей, інструменти резервного копіювання / відновлення та конфігурації реплікації, забезпечують відмовостійкість і механізми контролю доступу, що є критично важливими для бізнес-додатків.

5. Фреймворки об'єктно-реляційного відображення (ORM) спрощують розробку додатків, абстрагуючи SQL-запити з допомогою інтуїтивно зрозумілих API на мовах C# та Java.

Недоліки реляційних баз даних:

1. Зміни схеми можуть містити дорогі процедури міграції даних у міру розвитку додатків.

2. Вертикальне масштабування завдяки збільшенню обчислювальної потужності одного сервера має обмеження, що обмежує економічно ефективну масштабованість СКБД.

3. Інтенсивні SQL-запити між великими таблицями можуть погано виконуватися, що впливає на чутливі до затримок додатки.

4. Технології розподіленої кластеризації SQL додають складності в розподілених транзакціях, налагодженні та балансуванні навантаження.

Розберемо на прикладі Exel реляційні бази даних. Системи RDBMS чудово справляються з цими поширеними сценаріями використання:

- структуровані бізнес-дані зі складними взаємозв'язками, що вимагають гарантій ACID;
- інтенсивні аналітичні додатки зі спеціальними потребами в запитах;
- застарілі середовища з наявними кодовими базами SQL та наборами навичок;
- системи, де цілісність даних є критично важливою, наприклад, фінансові записи або медичні дані.

NoSQL бази даних Excel:

- швидке прототипування та ітерації на гнучких моделях даних;
- надзвичайно інтенсивні завдання з простими моделями доступу до даних;
- схеми, що постійно розвиваються, наприклад, керування профілями користувачів у соціальних додатках;
- вимоги до читання / запису з низькою затримкою у високопродуктивних транзакційних додатках;
- масово розподілені архітектури з доступом до терабайт даних.

Фундаментальні конструкції даних у SQL та NoSQL суттєво відрізняються. SQL структурує дані у попередньо визначені таблиці, рядки та стовпці. NoSQL охоплює сховища документів, що організовують дані у гнучкі JSON-документи, сховища широких стовпців, що зберігають дані у стовпцях, а не в рядках, та бази даних графів, що моделюють сутності й зв'язки у вигляді вузлів / ребер мережі. Ці дуже різні парадигми даних підходять для різних архітектур додатків і моделей доступу.

Бази даних SQL дають змогу розробникам писати запити на потужній мові структурованих запитів для вставки, оновлення, видалення та пошуку записів. SQL пропонує об'єднання, агрегати, підзапити та інші конструкції для складного аналізу. Бази даних NoSQL зазвичай підтримують прості CRUD-операції та пошук із допомогою API, а не декларативних мов. Деякі винятки, як-от MongoDB, мають декларативні інтерфейси на основі JSON. Оптимізація запитів також суттєво відрізняється залежно від навігації між реляційними та нереляційними сховищами.

Понад 40 років реляційні бази даних забезпечують основу для надійного управління даними в незліченних критично важливих робочих навантаженнях. Проте повсюдне поширення неструктурованих великих даних та нові архітектури додатків вимагають додаткових нереляційних систем, оптимізованих для гнучкості, масштабованості і продуктивності. Розуміння переваг та обмежень, притаманних як SQL, так і NoSQL, допомагає приймати виважені архітектурні рішення щодо баз даних з урахуванням моделей даних, шаблонів доступу, бізнес-вимог та технічних обмежень.

Список використаних джерел

1. Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, 13, 377–387.

2. Han, J., Haihong, E., Le, G., Du, J. (2011). Survey on NoSQL Database. 6th International Conference on Pervasive Computing and Applications, Port Elizabeth, 26–28 October 2011, 363–366.

3. Barata, M., Bernardino, J., Furtado, P. An overview of decision support benchmarks: TPC-DS, TPC-H and SSB. Rocha, A., Correia, A. M., Costanzo, S., Reis, L. P. (eds.). *New Contributions in Information Systems and Technologies.* Cham: Springer International Publishing, 2015. P. 619–628.

УДК 004.654

Маруняк А. О., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

КЛАСИФІКАЦІЯ ТА ЗАСТОСУВАННЯ КЛЮЧІВ У РЕЛЯЦІЙНИХ БАЗАХ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Реляційні бази даних є однією з основних систем управління даними, що використовуються для ефективного зберігання та доступу до інформації. Їх фундаментальна структура базується на таблицях, що містять рядки (записи) і стовпці (атрибути), а між таблицями створюються зв'язки. Ключі відіграють життєво важливу роль у реляційних базах даних, унікально ідентифікуючи записи та забезпечуючи зв'язки між пов'язаними таблицями. Існує кілька типів ключів зі спеціалізованими функціями, зокрема первинні ключі, зовнішні ключі, складені ключі та інші. Правильне призначення ключів дає змогу підтримувати цілісність даних, уникати проблем надмірності та дублювання, а також підвищувати продуктивність запитів, тому їх класифікація і правильне застосування є важливими для оптимальної побудови схеми бази даних і адміністрування системи. Далі розглянемо деякі види ключів.

Первинний ключ (PK) [1] – це поле або набір полів зі значеннями, унікальними для всієї таблиці. Значення ключа можна використовувати для посилання на цілі записи, оскільки кожен запис має своє значення для ключа. Кожна таблиця може мати лише один первинний ключ. Додаткові функції первинного ключа такі:

- точний пошук даних завдяки унікальному визначенню записів;
- вище забезпечення цілісності даних.

Зовнішній ключ (FK) [2] – це стовпець або комбінація стовпців, які використовуються для встановлення і забезпечення зв'язку між даними у двох таблицях, щоб контролювати дані, які можуть зберігатися в таблиці зовнішнього ключа. Посилання на зовнішній ключ створює зв'язок між двома таблицями, коли на стовпець або стовпці, які містять значення первинного ключа для однієї таблиці, посилається стовпець або стовпці іншої таблиці. Основні ролі зовнішнього ключа:

- підтримує узгодженість, обмежуючи вставку недійсних даних;
- запобігає появі порожніх записів, вимагаючи посилань на батьківські таблиці;
- оптимізує з'єднання таблиць на основі визначених індексів посилань;
- каскадне оновлення та видалення даних у взаємопов'язаних таблицях.

Складений ключ (Compound key) [3] – це ключ, який складається з 2 або більше атрибутів, які однозначно ідентифікують входження сутності. Кожен атрибут, що входить до складу складеного ключа, сам по собі є простим ключем.

Прикладом складеного ключа може бути сутність, що представляє модулі, які кожен студент відвідує в університеті. Сутність має ідентифікатор студента (studentId) та код модуля (moduleCode) як первинні ключі. Кожен з атрибутів, які складають первинний ключ, є простими ключами, оскільки кожен з них представляє унікальне посилання під час ідентифікації студента в одному екземплярі і модуля в іншому. Переваги складеного ключа:

- необхідні у таблицях зв'язку «багато-до-багатьох» між двома первинними таблицями;
- уникає ускладнення дизайну завдяки додаванню одиничних ключів з автоматичною нумерацією;
- зберігає основні функції первинного ключа;
- простіше, ніж схема розбиття на додаткові таблиці.

Ключ-кандидат (CK) [4] у SQL є підмножиною суперключів і потребує більше необхідних атрибутів, які не є важливими для унікальної ідентифікації кортежів. З цієї причини ключ-кандидат можна також назвати мінімальним суперключем. Ключі-кандидати в SQL вибираються з суперключів, і один із цих ключів-кандидатів надалі стає первинним ключем. Адміністратор бази даних робить вибір первинного ключа відповідно до частоти запитів. Ключ кандидат забезпечує:

- мінімальність для унікальності та ефективності;
- вибір первинного ключа.

Суперключ (SK) [5] – це один ключ або група з декількох ключів, які можуть однозначно ідентифікувати кортежі в таблиці. Суперключ може містити декілька атрибутів, які не можуть ідентифікувати кортежі в таблиці самостійно, але коли вони згруповані з певними ключами, вони можуть ідентифікувати кортежі однозначно.

Отже, ключі забезпечують належний зв'язок між даними таблиць у реляційних базах даних. Їх класифікація та використання підтримують критично важливі принципи продуктивності та цілісності. Фундаментальні знання про ключі допомагають уникнути неправильного вибору дизайну, що негативно впливає на бази даних. Розрізнення первинних, зовнішніх, складених ключів і ключів-кандидатів дає розуміння того, як задовольнити потреби схеми з допомогою оптимального застосування.

Список використаних джерел

1. Add or change a table's primary key in Access. URL: https://support.microsoft.com/en-us/office/add-or-change-a-table-s-primary-key-in-access-07b4a84b-0063-4d56-8b00-65f2975e4379?ui=en-us&rs=en-us&ad=us

2. Primary and Foreign Key Constraints. URL: https://learn.microsoft.com/en-us/sql/relational-databases/tables/primary-and-foreign-key-constraints?view=sql-server-ver16

3. Compound key. URL: http://wiki.gis.com/wiki/index.php/Compound_key

4. Candidate key in SQL. URL: https://datatrained.com/post/candidate-key-in-sql/

5. Different Keys in SQL (Primary Key, Candidate Key, Foreign Key). URL: https://www.analyticsvidhya.com/blog/2020/07/difference-between-sql-keys-primary-key-super-key-candidate-key-foreign-key/

УДК 004.65’416:316.472.4(043.2)

Назаренко М. С., здобувачка 2 курсу спеціальності 122 Комп’ютерні науки,

Зелінська О. В., канд. техн. наук, доцент, в. о. завідувача кафедри інформаційних технологій

ОПТИМІЗАЦІЯ БАЗ ДАНИХ ДЛЯ АНАЛІЗУ СОЦІАЛЬНИХ МЕРЕЖ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Зі збільшенням кількості соціальних мереж стає все більш важливим всебічний аналіз цих мереж для розуміння різних явищ. Аналіз соціальних мереж (SNA) – це одне з таких досліджень, яке допомагає проаналізувати структуру мереж. SNA може виконувати складні аналізи, як-от прогнозування майбутніх тенденцій та виявлення важливих комунікаційних патернів. Однак обробка даних SNA може бути ресурсномісткою, і для забезпечення ефективності важливо оптимізувати структури баз даних. З огляду на це наше дослідження зосереджується на оптимізації баз даних для забезпечення швидкості та ефективності аналізу соціальних мереж.

Актуальність. Актуальність цієї теми визначається необхідністю вирішення проблем, пов’язаних зі зростанням обсягів інформації у соціальних мережах. Аналіз соціальних мереж має величезне значення для розуміння поведінки користувачів, прогнозування трендів та прийняття управлінських рішень у реальному часі.

Дослідження з питань оптимізації баз даних має таке визначальне значення через кілька аспектів. По-перше, оптимізовані бази даних забезпечують швидкий доступ та оброблення великого обсягу інформації, що є ключовим для вчасного реагування на зміни в соціальних мережах. По-друге, вони дають змогу покращити точність і правдивість аналізу, сприяючи більш точним прогнозам і трендам.

Отже, розглянемо методи оптимізації баз даних для SNA.

Індексація

Створення правильних індексів для таблиць у базі даних є, мабуть, однією з найкращих технік налаштування продуктивності [1]. Без індексів весь доступ до даних у базі повинен здійснюватися шляхом сканування всіх доступних рядків, що дуже нераціонально, особливо для великих таблиць.

Цей процес передбачає створення структур даних, які дають змогу швидко шукати та отримувати певну інформацію. Для аналізу соціальних мереж індексування відіграє важливу роль у ефективному пошуку відносин і зв'язків між окремими особами чи організаціями. Завдяки створенню відповідних індексів СУБД може швидко отримувати доступ і проходити через мережу, забезпечуючи швидший аналіз.

Паралельна обробка

Аналіз великомасштабних соціальних мереж часто вимагає значних обчислювальних ресурсів. Методи паралельної обробки передбачають розподіл робочого навантаження між кількома процесорами або машинами, у такий спосіб скорочуючи час аналізу. Завдяки використанню можливості паралельної обробки, задачі SNA, які раніше вимагали багато часу, тепер можуть виконуватися за менший проміжок часу.

Попередня обробка даних

Попередня обробка даних перед зберіганням і аналізом є ще одним менш важливим аспектом оптимізації. Цей процес передбачає їх очищення, фільтрування нерелевантної інформації та агрегування точок даних для зменшення розміру і складності даних. Виконуючи поставлені завдання, база інформації стає більш оптимізованою, що призводить до швидшого аналізу та підвищення ефективності зберігання. До того ж шляхом видалення нерелевантних даних можна підвищити якість результатів аналізу.

Графові бази даних

Під час роботи з взаємопов'язаними даними, як-от соціальні мережі, традиційні реляційні бази даних можуть не забезпечувати оптимальної продуктивності.

Графові бази даних – це тип баз даних, які зберігають дані як вузли та ребра, що представляють сутність та зв'язки відповідно. Вони дають змогу моделювати й запитувати дані природним та інтуїтивно зрозумілим способом, використовуючи теорію графів і алгоритми [2]. Такий тип БД особливо підходить для аналізу соціальних мереж, оскільки вони може обробляти великий обсяг різноманітних даних соціальних мереж, а також забезпечують швидкий і гнучкий аналіз властивостей мережі, як-от центральність, кластеризація або найкоротші шляхи.

Щоб використовувати бази даних графів для SNA, потрібно визначити схему мережі, тобто типи та атрибути вузлів і ребер мережі (наприклад, можуть бути

вузли для користувачів, публікацій, коментарів і груп, а також межі для вподобайок, підписок та згадок).

Графові БД пропонують власну підтримку для обходу та запиту взаємопов'язаних даних, що робить їх зручними для аналізу соціальних мереж. Їх ефективні механізми обходу графів і вбудовані алгоритми значно підвищують швидкість і продуктивність завдань аналізу соціальних мереж.

Прикладом використання цих БД для SNA є Facebook [3]. Це є один із найбільших сайтів соціальних мереж, і його функція Graph Search дає змогу користувачам шукати вміст на платформі. Щоб створити Graph Search, інженерам Facebook потрібно було оптимізувати свою базу даних для обробки величезної кількості даних, створених їх користувачами.

Graph Search від Facebook використовує Social Graph – структуру бази даних, яка зберігає зв'язки між користувачами. Соціальний графік містить дані про користувачів, публікації, місця і події та ін. Інженери Facebook оптимізували записування Social Graph на диск і читання в пам'яті, щоб мінімізувати перевантаження системи. Щоб зменшити надмірність даних, вони використовували такі методи матеріалізованого представлення та попереднього завантаження даних у розділи. Оптимізація бази даних Social Graph допомогла пошуковій системі Facebook підтримувати десятки мільйонів щоденних запитів.

Висновки. Отже, ефективний аналіз соціальних мереж для отримання важливої інформації вимагає ретельної оптимізації баз даних, які зберігають і обробляють величезну кількість взаємопов'язаних даних. Завдяки використанню вищеназваних методів оптимізація бази даних може значно підвищити швидкість, масштабованість і результативність аналізу соціальних мереж.

Оскільки розмір і складність соціальних мереж продовжують зростати, існує постійна потреба в інноваційних підходах до оптимізації баз даних для SNA. Подальші дослідження та розробки в цій галузі сприятимуть кращій продуктивності та аналізу в реальному часі.

Список використаних джерел

1. Techniques for optimizing databases. URL: https://www.dbta.com/Columns/DBA-Corner/Techniques-for-Optimizing-Databases-154027.aspx (дата звернення: 07.07.2022)

2. How Graph databases can boost social network analysis. URL: http://surl.li/nwidk

3. Facebook Graph Search. URL: https://en.ryte.com/wiki/Facebook_Graph_Search

УДК 004.73.53

Поліщук В. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Зелінська О. В., канд. техн. наук, доцент, в. о. завідувача кафедри інформаційних технологій

АДМІНІСТРАТОР БАЗ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Основна роль адміністратора бази даних (DBA) полягає в тому, щоб зробити дані компанії доступними для користувачів. DBA повинен тісно співпрацювати з розробниками, щоб забезпечити ефективне використання бази даних додатками, а також із системними адміністраторами, щоб забезпечити належне обслуговування та ефективне використання фізичних ресурсів.

DBA – це абревіатура від англ. *database administrator* (адміністратор баз даних). Це група експертів, які проєктують, створюють, експлуатують і підтримують бази даних, переважно в компаніях. Їх роль полягає в тому, щоб зробити значний внесок у прибуток компанії шляхом належного управління середовищем баз даних, яке є незамінним для систем компанії.

Актуальність теми адміністратора бази даних зростає щороку. Оскільки підприємства все більше покладаються на дані для прийняття рішень і підвищення ефективності, роль DBA стає все більш важливою. У цифровій економіці дані є цінним активом для кожної компанії. Адміністратори баз даних відіграють ключову роль у захисті та ефективному використанні цих даних [1].

Основними ролями DBA є:

1. Проєктування та створення баз даних. DBA проєктують бази даних, необхідні для роботи компанії і відповідають за створення баз даних на етапі розробки системи. Вони також можуть переробляти бази даних і додавати нову функціональність, коли змінюються системні вимоги.

2. Експлуатація та обслуговування баз даних. Бази даних є важливими активами, які керують важливою інформацією компанії. Тому DBA зосереджуються на експлуатації та обслуговуванні баз даних. Вони забезпечують безпеку даних, виконуючи резервне копіювання та відновлення, перевірку цілісності даних і моніторинг.

3. Вирішення проблем та усунення несправностей. Якщо з базою даних виникає проблема, DBA реагує на неї. Щоб вирішити проблему, він аналізує файли журналів і файли трасування, щоб визначити причину збою. Потім вживаються відповідні заходи для якнайшвидшого відновлення після збою.

Навички, необхідні для DBA, включають:

1. Знання SQL. DBA використовують SQL для маніпулювання базами даних. Отже, знання SQL є необхідним – не тільки базовий синтаксис SQL, але й знання з налаштування та оптимізації.

2. Знання роботи в мережі. Доступ до баз даних здійснюється через мережі. Тому бажано, щоб DBA мали базові знання про роботу в мережі: вони повинні розуміти IP-адресацію, маршрутизацію та віртуальні приватні мережі (VPN).

3. Знання про безпеку. Оскільки бази даних керують важливою інформацією компанії, знання безпеки є дуже важливими; DBA відповідають за налаштування дозволів на доступ до бази даних, шифрування інформації в базі даних та налаштування брандмауерів.

4. Навички вирішення проблем. DBA повинні вміти швидко і правильно вирішувати проблеми, тому навички вирішення проблем є дуже важливими. Вони повинні розвивати навички аналізу та усунення несправностей, а також спокійно реагувати на виникнення несправностей. Більшість адміністраторів баз даних працюють у фірмах, що займаються розробкою комп'ютерних систем та наданням супутніх послуг, як-от інтернет-провайдери та фірми, що займаються обробкою даних. Інші DBA працюють у фірмах з великими базами даних, як-от страхові компанії та банки, які зберігають великі обсяги персональних і фінансових даних своїх клієнтів. Деякі DBA керують базами даних для роздрібних компаній, які відстежують інформацію про кредитні картки своїх покупців та інформацію про доставку, а інші працюють у медичних компаніях і ведуть медичні записи пацієнтів.

Адміністратори баз даних повинні спланувати логічну структуру зберігання бази даних, загальний дизайн бази даних і стратегію резервного копіювання бази [2]. Важливо спланувати, як логічна структура зберігання бази даних вплине на продуктивність системи та різні операції з управління базою даних. Наприклад, перед створенням табличного простору бази даних потрібно знати кількість файлів даних, які становлять табличний простір, тип інформації, яка буде зберігатися в кожному табличному просторі, і дискові накопичувачі, на яких будуть фізично зберігатися файли даних. Плануючи загальну логічну структуру зберігання бази даних, враховуємо вплив, який ця структура матиме, коли база даних буде фактично створена і запущена.

Цей етап планування має вирішальне значення в середовищі розподіленої бази даних. Фізичне розташування даних, до яких часто звертаються, має значний вплив на продуктивність програми [3].

DBA є невід'ємною частиною компанії і відповідають за експлуатацію та обслуговування баз даних, що вимагає знання мови SQL, базових знань про роботу в мережі, знань про безпеку та навичок вирішення проблем. Отже, адміністратори баз даних керують конкретними базами даних і визначають дозволи

для окремих користувачів. Адміністратори архівують дані та несуть відповідальність за коректну роботу системи, використовуючи технічну і програмну документацію цієї бази та функціональні можливості засобів, наданих її розробниками (виробниками) для забезпечення безперебійної роботи апаратного та програмного забезпечення. Контролюйте використання інформації бази даних авторизованими користувачами в дозволених межах.

Список використаних джерел

1. Nure. URL: https://cn.nure.ua/administrator-bazi-danih/ (дата звернення: 05.12.2023).

2. Klona.ua. URL: https://klona.ua/uk/blog/artificial-intelligence-uk/bazy-danyh-ponyattya-subd (дата звернення: 05.12.2023).

3. Освіта.ua. URL: https://osvita.ua/proforientation/profession/71517/ (дата звернення: 05.12.2023).

УДК 004.67:005

Поліщук Д. О., здобувач 3 курс спеціальності 122 Комп'ютерні науки,

Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

МЕТОДИ ОПТИМІЗАЦІЇ ВЕЛИКИХ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сьогодні існує проблема надмірного накопичення даних у надзвичайно великих розмірах, що зумовлено їх щоденним продукуванням, тож для її вирішення було створено поняття Big Data. У загальному плані Big Data, вони ж великі дані, є структурованими і неструктурованих даними величезних обсягів і такої ж значної різноманітності, ефективно оброблених масштабованими програмними інструментами. Якщо підсумувати, то це інформація, що не піддається обробці класичними способами через її величезний об'єм [1].

Big Data, згідно з правилом «П'яти V», ділиться на п'ять ключових характеристик [2].

Згідно з рис. 1, першою рисою є Volume – об'єм накопиченої бази даних, який охоплює настільки великий обсяг інформації, що його практично нереально обробляти та зберігати традиційними способами.

Другою рисою є Velocity – швидкість та темпи накопичення і обробки даних постійно збільшуються.

Наступною є різноманітність Variety, яка позначає можливості одночасно обробляти неструктуровану та структуровану інформацію.

Рисунок 1. Ключові риси Big Data

Також потрібно перевіряти дані на справжність, тому далі йде Veracity – виокремлення достовірних даних.

І протилежність минулому Variability – невідповідність інформації, яка ускладнює та іноді сильно заважає процесам обробки й управлінням інформації.

Розглянемо, як відбувається оптимізація великих даних, адже вона є важливим аспектом ефективної обробки та аналізу великих обсягів інформації. Нижче подано деякі методи оптимізації, які можна використовувати у великих даних.

1. Розподілений обчислювальний підхід: використання розподільних систем обчислень та обробки даних, як-от Apache Hadoop або Apache Spark. Це дає змогу паралельно обробляти великі обсяги даних на різних вузлах кластера.

2. Компресія даних: використання ефективних методів компресії даних, щоб зменшити їх обсяг для зберігання та передачі. Це допомагає знизити вимоги до простору і підвищити швидкість передачі даних.

3. Індексація: створення індексів для швидшого доступу до конкретних даних. Індексація допомагає зменшити час пошуку та фільтрації даних.

4. Кешування: використання систем кешування для збереження результатів попередніх операцій та уникнення повторних обчислень.

5. Оптимізація запитів: під час проєктування запитів до баз даних дотримуйтеся найкращих практик і використовуйте оптимізовані SQL-запити для ефективного вибору необхідної інформації.

6. Розподілене зберігання: використання розподіленого зберігання даних за допомогою бази даних NoSQL для забезпечення масштабованості та швидкодії.

7. Паралельні алгоритми – використання паралельних алгоритмів для обчислень та обробки даних. Це покращить продуктивність завдяки використання багатьох обчислювальних ресурсів одночасно.

8. Оптимізація мережевої взаємодії: мінімізація навантаження на мережу шляхом передачі лише необхідних даних та використання оптимізованих протоколів передачі.

Ці методи можна використовувати окремо чи в комбінації для досягнення оптимальної продуктивності під час роботи з великими обсягами даних [3, 4, 5].

Існує достатня кількість методів оптимізації великих даних, головне – це вміти правильно їх поєднати. Адже саме правильно підібрані методи оптимізації для потоку Big Data дають можливість передавати, зберігати та користуватись великими даними.

Список використаних джерел

1. Wikipedia. URL: https://en.wikipedia.org/wiki/Big_data. (дата звернення: 27.11.2023).

2. Gagroup. URL: https://qagroup.com.ua/publications/shcho-take-big-data-i-iak-tce-pratciuie/ (дата звернення: 27.11.2023).

3. Aiconference. URL: https://aiconference.com.ua/uk/news/tehnologii-big-data-klyuchevie-harakteristiki-osobennosti-i-preimushchestva-97883 (дата звернення: 27.11.2023).

4. Степанюк О. С., Січко Т. В. Особливості використання реляційних та нереляційних баз даних в Big Data. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2020. С. 103–106.

5. Ткачук Н. О., Січко Т. В. Застосування Big Data у бізнесі. *Комп'ютерні технології обробки даних:* матеріали всеукр. наук.-практ. конф., м. Вінниця, 2022. С. 224–226.

УДК 004.658.2

Проценко А. С., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ REDIS: ПЕРЕВАГИ ТА НЕДОЛІКИ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі, де обробка великого обсягу даних є ключовим аспектом для багатьох галузей, системи управління базами даних стають невід'ємною частиною розробки програмного забезпечення. Однією з таких систем є Redis, яка привертає увагу своєю унікальною архітектурою та функціональністю.

Далі розглянемо основну характеристику, переваги й недоліки цієї СУБД. Також намагатимемося зрозуміти, у яких сценаріях використання Redis може бути найефективнішим і чи відповідає вона вимогам різноманітних завдань, які постають перед розробниками сучасного програмного забезпечення.

Redis (Remote Dictionary Server) – відкрите програмне забезпечення, основна особливість якого полягає в збереженні даних у формі пар ключ-значення, які зберігаються в оперативній пам'яті, з можливістю забезпечувати довговічність зберігання на бажання користувача. Ця СУБД з відкритим початковим кодом та написана на ANSI C.

Переваги:

➢ **Висока швидкість** – одна з ключових переваг Redis, що досягається завдяки використанню оперативної пам'яті для зберігання даних. Якщо в довговічності даних немає потреби, Redis може пропонувати значно кращі результати, порівняно з СУБД, які зберігають кожну зміну на диск перед завершенням транзакції. Також Redis може ефективно виконувати операції читання та запису без помітної різниці у швидкості. До прикладу, під час тестування на сервері з CPU Xeon X3320 2,5 ГГц вдалося забезпечити 110 000 операцій запису і 81 000 операцій читання за секунду. Це робить Redis відмінним вибором для завдань, де вимагається максимальна продуктивність і миттєвий доступ до даних, як-от системи кешування та інші сценарії реального часу.

➢ **Модель даних у Redis** – це асоціативний масив, у якому ключі відображаються в значення. Основна відмінність між Redis та іншими базами такого типу в тому, що значення словника не обмежені рядковими типами. На додачу до рядків підтримуються такі абстрактні типи даних: списки рядків, множини рядків (невпорядкований набір неповторюваних елементів), впорядковані множини рядків (набори неповторюваних елементів впорядкованих за пов'язаним значенням з рухомою комою), хеші, в яких ключі і значення є рядками. Тип значення визначає, які операції можна виконувати з цим значенням. У Redis доступні високорівневі атомарні операції з боку, як-от перетин, об'єднання та різниця між множинами та списками.

➢ **Потужні механізми реплікації та розподіленості** визначають Redis як ефективний інструмент для масштабування й забезпечення надійності. Застосовуючи модель «master-slave» для реплікації, Redis дає змогу створювати дублікати даних з будь-якого сервера Redis довільну кількість разів. Цей підхід особливо корисний для масштабування читання (але не запису) чи надлишковості даних. Версія Redis 3.0 та пізніші включають у себе Redis Cluster, що розширює можливості системи для створення розподілених сховищ. Він дає змогу автоматично розподіляти дані між численними вузлами, сприяючи створенню відмовостійких конфігурацій, де дані дублюються на різних вузлах. Це гарантує неперервну доступність навіть у випадку відмови одного з вузлів, не призводячи до зупинки системи загалом [1].

➢ **Легкість конфігурації та швидкість інсталяції.** Процес встановлення Redis вражає простотою, даючи змогу швидко розпочати роботу з цією СУБД.

До того ж Redis відома своєю легкістю налаштування, що робить її привабливою для розробників, які цінують ефективність та мінімальний час витрачений на налагодження.

Недоліки:

➢ **Обмеженість об'ємом пам'яті** є суттєвим недоліком Redis. Оскільки вона зберігає всі дані в оперативній пам'яті, об'єм доступної пам'яті може бути обмеженим. Це обмеження призводить до ситуації, коли Redis стає менш придатним для роботи, де зберігання всієї інформації в оперативній пам'яті стає витратним та неефективним. Такий аспект може потребувати ретельного планування та оптимізації використання ресурсів для досягнення найкращої продуктивності системи.

➢ **Обмежена здатність виконувати складні запити.** Орієнтований на швидкодію та простоту операцій із даними, Redis може виявитися менш ефективним під час обробки запитів, які вимагають складних операцій чи аналізу даних за складними критеріями. Це обмеження може становити проблему для проєктів, де важлива гнучкість і розширена функціональність обробки даних. Під час вибору інструментів розробники повинні враховувати цю особливість Redis і вибирати його для завдань, де простота та швидкодія важливіші, або доповнювати його іншими рішеннями у випадках, коли потрібно виконати складні операції.

➢ **Втрата даних** у разі вимкнення системи. Навіть з урахуванням наявності механізму реплікації, який дає змогу створювати копії даних на різних серверах, існує ризик втрати інформації у випадку, якщо не всі репліки успішно синхронізовані перед вимкненням системи. Це може виникнути у випадку непередбачуваних помилок або затримок у процесі реплікації. Цю потенційну вразливість важливо враховувати під час проєктування системи та розробки стратегій резервного копіювання для запобігання втрати даних у випадку аварійних ситуацій або вимушеного вимкнення системи.

➢ **Обмежені можливості безпеки** можуть призвести до потенційних проблем під час встановлення відкритого доступу до сервера. Redis не завжди надає розширені засоби для безпеки даних, тому важливо усвідомлювати це і вживати відповідні заходи, як-от використання автентифікації для захисту інформації. У великих або критичних проєктах треба уважно розглядати додаткові заходи безпеки та налаштування системи для мінімізації можливих ризиків [2, 3].

Отже, Redis є потужною системою управління базами даних з унікальною архітектурою та низкою переваг, які визначають її популярність серед розробників. Висока швидкість завдяки використанню оперативної пам'яті, гнучка модель даних, легкість конфігурації, а також механізми реплікації та розподіленості роблять Redis ідеальним вибором для сценаріїв, де важлива продуктивність і надійність. Однак існують певні недоліки, як-от обмеженість об'ємом

пам’яті, виконання складних запитів, ризик втрати даних під час вимкнення системи. Ці обмеження важливо враховувати під час вибору Redis для конкретного проєкту та вживати відповідних заходів для оптимізації та захисту від потенційних проблем.

Список використаних джерел

1. Redis. URL: https://en.wikipedia.org/wiki/Redis (дата звернення: 20.11.2023).

2. Redis на практичних прикладах. URL: https://devzone.org.ua/post/redis-na-praktichnikh-prikladakh (дата звернення: 22.11.2023).

3. Introduction to Redis. URL: https://redis.io/docs/about/ (дата звернення: 22.11.2023).

УДК 004.65:004.056.5

Шевцов М. В., здобувач 2 курсу спеціальності 122 Комп’ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

ВИКОРИСТАННЯ КРИПТОГРАФІЇ ДЛЯ ЗАХИСТУ ДАНИХ У БАЗІ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі, коли інформація є найціннішим ресурсом, забезпечення безпеки даних стає критично важливим завданням для багатьох організацій. Паролі користувача, цифрові підписи та інша конфіденційна інформація може бути злита в мережу за будь-яких обставин – протиправних дій зловмисників чи через просту несправність сервера. Тому зберігати цю інформацію в базах даних треба у зашифрованому форматі. До того ж необхідно, щоб із шифру не можна було отримати ключ у тому вигляді, в якому його вводив користувач. Саме інтеграція криптографічних методів та хеш-функцій стає ключовим аспектом забезпечення конфіденційності та цілісності даних.

Найвідомішими криптографічними алгоритмами для захисту інформації, які використовуються зокрема і в базах даних, є:

- AES (Advanced Encryption Standard) – це симетричний блоковий метод, який використовує ключ для шифрування та розшифрування даних в блоках фіксованого розміру. Він забезпечує доволі високий рівень безпеки і широко використовується для шифрування даних у реляційних базах даних [1].
- RSA (Rivest–Shamir–Adleman) – це асиметричний алгоритм, який використовує пару ключів для шифрування та розшифрування даних. Публічний ключ

використовується для шифрування, а приватний – для дешифрування. Він відіграє важливу роль у захисті даних під час їх передачі та обміну [2].

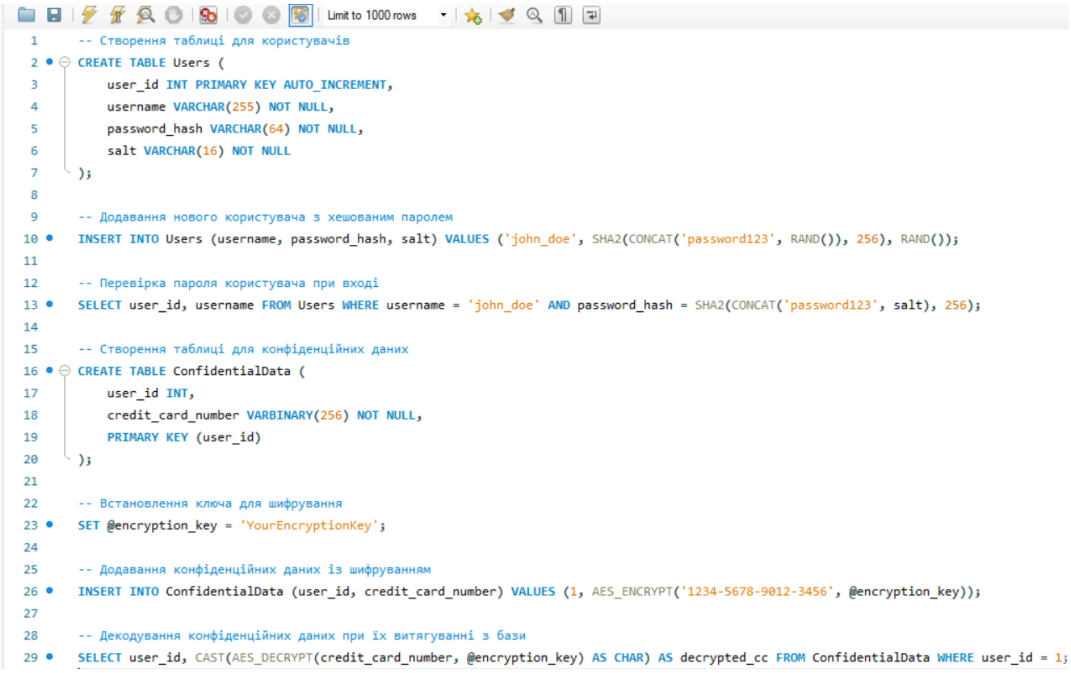
➢ SHA-256 (Secure Hash Algorithm 256-bit) – це хеш-функція, яка є однією з сімейства алгоритмів SHA. Вони відрізняються розміром вхідного хешу, стійкістю до колізій та швидкістю обчислення. Саме SHA-256 використовується для створення фіксованого 256-бітного хеш-коду зі вхідних даних. Вона застосовується для перевірки цілісності даних та генерації контрольних сум.

➢ MD5 (Message Digest Algorithm 5) – це хеш-функція, яка обчислює контрольну суму вхідного повідомлення. Результат – фіксована 128-бітна хеш-сума [2]. Хоча MD5 була широко використовуваною раніше, зараз вона не є безпечною і відома своєю вразливістю до колізій. Проте вона все ще використовується для простих завдань, як-от генерація хеш-сум файлів.

Для забезпечення більшого рівня безпеки існують концепції генерації одноразових ключів та використання солів. Генератори одноразових ключів – це алгоритми генерації унікальних ключів для кожної транзакції або сесії, які надають додатковий рівень безпеки, оскільки ключі використовуються лише один раз і не можуть бути використані для наступних атак. Один із таких алгоритмів – це TOTP (Time-Based One-Time Password), який базується на поточному часі та секретному ключі. Концепція солів – випадкових рядків, які додаються до паролів користувачів перед хешуванням, – підвищує безпеку, оскільки два однакові паролі будуть мати різні хеш-значення через випадковий символ, який додається до кожного пароля. Солі важливі для захисту бази даних від атак методом перебору (brute force) та атак методом таблиць радужних хешів (rainbow tables) [3].

Спробуємо розібратись на практиці. Розглянемо приклад шифрування конфіденційних даних (наприклад, номера кредитної картки).

Тут ми створюємо таблицю Users, яка містить інформацію про користувачів, як-от ідентифікатор (user_id), ім’я користувача (username), хеш пароля (password_hash) і випадковий рядок (salt). Код вставляє нового користувача в таблицю Users зі згенерованим хешем пароля і випадковим рядком солі для безпеки пароля. Під час входу програма виконує запит до таблиці Users, де перевіряє, чи співпадає хеш введеного пароля користувача з тим, що зберігається в базі даних. Таблиця ConfidentialData призначена для зберігання конфіденційних даних, як-от номери кредитних карток. Для цієї таблиці використовується VARBINARY для зберігання зашифрованих даних. Щоб забезпечити безпеку конфіденційних даних, встановлюється ключ шифрування (@encryption_key), який використовується для шифрування та розшифрування даних. Код вставляє новий запис у таблицю ConfidentialData, де номер кредитної картки зашифрований з використанням ключа шифрування. Під час витягування конфіденційних даних із таблиці ConfidentialData код розшифровує зашифрований номер кредитної карти за допомогою ключа шифрування.



```sql
-- Створення таблиці для користувачів
CREATE TABLE Users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(255) NOT NULL,
    password_hash VARCHAR(64) NOT NULL,
    salt VARCHAR(16) NOT NULL
);

-- Додавання нового користувача з хешованим паролем
INSERT INTO Users (username, password_hash, salt) VALUES ('john_doe', SHA2(CONCAT('password123', RAND()), 256), RAND());

-- Перевірка пароля користувача при вході
SELECT user_id, username FROM Users WHERE username = 'john_doe' AND password_hash = SHA2(CONCAT('password123', salt), 256);

-- Створення таблиці для конфіденційних даних
CREATE TABLE ConfidentialData (
    user_id INT,
    credit_card_number VARBINARY(256) NOT NULL,
    PRIMARY KEY (user_id)
);

-- Встановлення ключа для шифрування
SET @encryption_key = 'YourEncryptionKey';

-- Додавання конфіденційних даних із шифруванням
INSERT INTO ConfidentialData (user_id, credit_card_number) VALUES (1, AES_ENCRYPT('1234-5678-9012-3456', @encryption_key));

-- Декодування конфіденційних даних при їх витягуванні з бази
SELECT user_id, CAST(AES_DECRYPT(credit_card_number, @encryption_key) AS CHAR) AS decrypted_cc FROM ConfidentialData WHERE user_id = 1;
```

Рисунок 1. Приклад бази даних

Ми використали функцію SHA-2 з сімейства SHA для створення хешу паролів, додавали сіль (salt) для підвищення безпеки, і використали функції AES_ENCRYPT() та AES_DECRYPT(), які працюють за згаданим раніше алгоритмом AES, для шифровки та дешифровки конфіденційних даних у MySQL.

Отже, інтеграція криптографічних методів та хеш-функцій у бази даних є основним фактором забезпечення безпеки інформації. Криптографічні методи дають змогу захистити дані від несанкціонованого доступу та перегляду, а хеш-функції забезпечують надійне шифрування й перевірку цілісності даних. Інтеграція цих методів дає змогу створити надійну систему зберігання та обробки даних, що важливо для багатьох сфер діяльності.

Список використаних джерел

1. Database Encryption in SQL Server 2008 Enterprise Edition. URL: https://download.microsoft.com/download/8/b/2/8b22991f-3f2f-4cea-b2ba-55c190841145/tdeandefsbitlocker.docx

2. Schneier B. Applied Cryptography: Protocols, Algorithms, and Source Code in C. Wiley, 1996. 784 c.

3. Давлетханов М. Концепция одноразовых паролей в построении системы аутентификации Byte, 2006. Vol. 7–8(95).

СЕКЦІЯ 5
ПРИКЛАДНІ АСПЕКТИ ОБРОБКИ ДАНИХ В ІНФОРМАЦІЙНИХ СИСТЕМАХ

УДК 339, 004.8
Гарматій Н. М., канд. екон. наук,
Сербін В. С., студент групи СТм-61

ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ В ЕКОНОМІЧНИХ СИСТЕМАХ МАСОВОГО ОБСЛУГОВУВАННЯ

Тернопільський національний технічний університет імені Івана Пулюя, м. Тернопіль

Нейронні мережі – це математичні моделі, що імітують роботу біологічних нейронів, здатні навчатися на основі даних, адаптуватися до змінних умов і вирішувати складні задачі. Ця технологія має переваги над класичними методами аналізу даних, оскільки має можливість працювати з неповними даними, автоматизувати аналіз, досягати високої точності результатів, а також відтворювати складні нелінійні залежності між різними індикаторами розвитку.

Нейронні мережі широко застосовуються в економічних системах масового обслуговування, як-от торгівля, транспорт, зв'язок тощо з метою прогнозування, класифікації, моделювання та оптимізації процесів обслуговування великої кількості заявок, що надходять у випадковий спосіб; дають змогу аналізувати великі обсяги історичних і поточних даних, виявляти закономірності і тенденції, а також генерувати точні і своєчасні прогнози для підтримки прийняття рішень, планування, контролю та оцінки ефективності систем [1].

У торгівлі нейронні мережі використовуються для прогнозування попиту на товари та послуги, оптимального формування асортименту, ціноутворення, рекомендаційних систем, управління складом, логістики, маркетингу, аналізу споживчої поведінки, сегментації ринку, виявлення шахрайства та аномалій тощо. Наприклад, система Amazon використовує нейронні мережі для аналізу історії купівель, переглядів, відгуків, пошукових запитів та інших даних про користувачів, щоб пропонувати їм товари, які вони можуть купити, а також для визначення оптимальних цін, знижок, рекламних кампаній тощо [2]. Також Amazon використовує нейронні мережі для управління своїми складами, де роботи, що працюють під керівництвом штучного інтелекту, забезпечують ефективне складування, сортування та доставку товарів, скорочуючи до 30 % витрат часу на ці дії, порівняно з виконанням завдань людиною [3]. Пристрої у процесі виконання завдань під управлінням нейронних мереж на складі Amazon зображені на рисунку 1.

Принцип роботи такої системи полягає в тому, що вона навчається на великій кількості даних, використовуючи різні типи нейронних мереж, як-от згорткові, рекурентні, генеративно-змагальні та інші, і видає рекомендації, які максимізують задоволення клієнтів та прибутки компанії [2].

Рисунок 1. Пристрої під управлінням нейронних мереж Amazon

У транспорті нейронні мережі можуть використовуватися для прогнозування трафіка, оптимальної маршрутизації, управління рухом, розкладу рейсів, розподілу місць, бронювання квитків, безпеки, навігації, автономного водіння, аналізу аварійності та екологічності тощо. Наприклад, компанія UPS використовує нейронні мережі для розробки своєї системи ORION (On-Road Integrated Optimization and Navigation), яка допомагає водіям вибирати найкращі маршрути для доставки вантажів, враховуючи різні фактори, як-от трафік, погода, час доставки, відстань та інше. За даними компанії, ця система дає змогу заощадити приблизно 100 мільйонів галонів палива, зменшити викиди вуглекислого газу на 100 тисяч тонн і зекономити 50 мільйонів годин робочого часу на рік.

Іншим прикладом застосування нейронних мереж в економічних системах масового обслуговування є система Google Maps, яка використовує нейронні мережі для аналізу даних із супутників, датчиків, смартфонів, камер та інших джерел, щоб показувати користувачам актуальну інформацію про стан доріг, пропонувати найкращі маршрути, враховуючи різні фактори, як-от час, відстань, погода, затори, аварії, ремонти тощо. Такий підхід допомагає скоротити витрати часу та матеріальних ресурсів на транспортування товарів на 20 %.

Принцип роботи такої системи, аналогічно системам у торгівлі, полягає у тому, що вона навчається на великій кількості даних, використовуючи різні типи нейронних мереж, як-от згорткові, рекурентні, генеративно-змагальні та інші, і видає рекомендації, які мінімізують час та витрати подорожі й підвищують комфорт і безпеку користувачів [4]. Приклад виконання процесу побудови просторових даних трафіка району аеропорту міста Ріад (Саудівська Аравія) подано на рисунку 2.

В економічній сфері зв’язку нейронні мережі можуть використовуватися для прогнозування навантаження на мережу, оптимального розподілу ресурсів, покращення якості передачі даних, компресії та шифрування інформації, управління якістю обслуговування, тарифікації, розпізнавання мови та зображень, синтезу мови та зображень, кодування та декодування сигналів, захисту від атак і перешкод тощо. Наприклад, система Skype використовує нейронні мережі для забезпечення високоякісної та надійної передачі голосу та відео між користу-

вачами, а також для надання додаткових функцій, як-от переклад мови, розпізнавання обличь та інші [5].

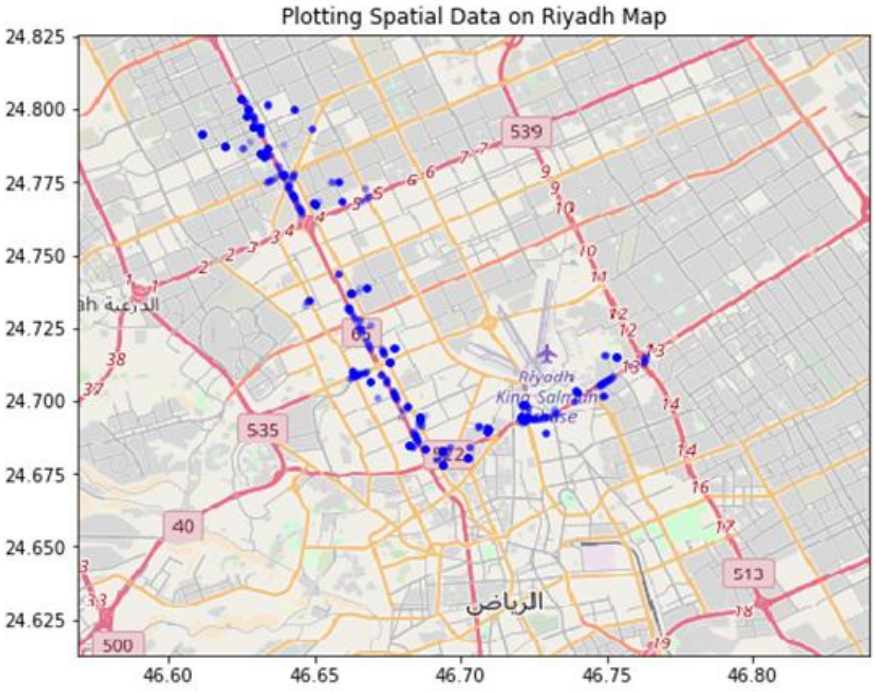


Рисунок 2. Графік потоку трафіка

Компанія Google використовує нейронні мережі для своєї системи Google Translate, яка здатна перекладати текст та мову між більш ніж 100 мовами, використовуючи глибоке навчання та нейронні мережі [6]. Також Google використовує нейронні мережі для своєї системи Google Duplex, яка може вести розмову з людьми по телефону, використовуючи природну мову, щоб виконувати завдання, як-от бронювання столиків у ресторанах, запис на стрижку, нагадування про події та інші.

Висновки. Нейронні мережі – це потужний інструмент, здатний налагодити та прискорити економічні процеси, оптимізувати системи різних рівнів і підвищити прибутки, однак для використання ця технологія потребує великої кількості початкових даних та часу для навчання, а впровадження може виявитись більш складним залежно від програмної платформи, потужностей комп'ютерів та обраного типу нейронної мережі.

Список використаних джерел

1. Василенко О. О., Бурлєєв О. Л., Іваненко Р. М. Ефективність використання штучних нейронних мереж в економіці. URL: http://surl.li/nbluh

2. Гарматій Н. М. Класичні та сучасні моделі економіки: навч. посіб. / Н. М. Гарматій, І. О. Мартиняк, Г. В. Ціх. Вид-во: ФОП Паляниця В. А., 2023. 300 с.

3. Чайка П. Нейронні мережі: пояснення простою і зрозумілою мовою. URL: https://www.poznavayka.org/uk/nauka-i-tehnika-2/neyronni-merezhi-yih-zastosuvannya-robota/

4. Кетеринич Л. О. Прикладне застосування нейронних мереж. URL: http://csc.knu.ua/media/study/master-degree/courses-2020-202x/application-of-neural-networks.pdf

5. Кравченко В. Що таке нейронні мережі та як вони працюють. URL: https://livingfo.com/shcho-take-nejronni-merezhi-ta-iak-vony-pratsiuiut/

6. Добровська Л. М., Добровська І. А. Теорія та практика нейронних мереж. URL: https://ela.kpi.ua/bitstream/123456789/49841/1/Teoriia_praktyka_neironnykh_merezh.pd

7. FutureNow. Що таке нейронна мережа: простими словами. URL: https://futurenow.com.ua/shho-take-nejronna-merezha-prostymy-slovamy/

УДК 004.67

Бойко У. В., здобувач 2 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки

ВИЯВЛЕННЯ ФАКТОРІВ, ЩО ВПЛИВАЮТЬ НА РЕЙТИНГ І КАСОВІ ЗБОРИ ФІЛЬМУ МЕТОДАМИ DATA SCIENCE

Донецький національний університет імені Василя Стуса, м. Вінниця

Сучасна кіноіндустрія постійно шукає ключі до розуміння успіху фільмів. Ця робота фокусується на використанні методів Data Science для виявлення факторів, що впливають на рейтинг і касові збори фільмів.

У цьому досліджені було проведено аналіз набору даних «*tmdb_movies.csv*». Цей датасет містить широкий спектр інформації про різні фільми, що дає змогу провести глибокий аналіз та визначити показники, які впливають на касовий і глядацький успіх фільмів.

Дослідження базується на застосуванні інструментів програмування R [1], середовища RStudio [2] та пакету Tidyverse для детального аналізу даних про кінофільми. Робота включає вибірку даних, їх підготовку, очищення та аналіз, забезпечуючи об'єктивність і точність результатів.

На рисунку 1 зображено кількісні зведення за кожною змінною в очищеному наборі даних.

Робота включає всебічний статистичний аналіз даних про кінофільми, зокрема описову статистику, кореляційний аналіз та візуалізацію.

На рисунку 2 зображена кореляційна матриця, на якій можна побачити зв'язки між змінними.

```
     budget              revenue              original_title        runtime
 Min.   :     7000   Min.   :    13308   Length:3801        Min.   : 26.0
 1st Qu.: 10000000   1st Qu.:  14460000   Class :character   1st Qu.: 96.0
 Median : 25000000   Median :  46236000   Mode  :character   Median :106.0
 Mean   : 37609526   Mean   : 109155433                      Mean   :109.4
 3rd Qu.: 50000000   3rd Qu.: 126216940                      3rd Qu.:119.0
 Max.   :425000000   Max.   :2781505847                      Max.   :338.0
    genres             vote_count      vote_average     release_year
 Length:3801        Min.   :  10.0   Min.   :2.200   Min.   :1960
 Class :character   1st Qu.:  73.0   1st Qu.:5.700   1st Qu.:1995
 Mode  :character   Median : 207.0   Median :6.200   Median :2004
                    Mean   : 533.8   Mean   :6.174   Mean   :2001
                    3rd Qu.: 584.0   3rd Qu.:6.700   3rd Qu.:2010
                    Max.   :9767.0   Max.   :8.400   Max.   :2015
```

Рисунок 1. Кількісні зведення за кожною змінною

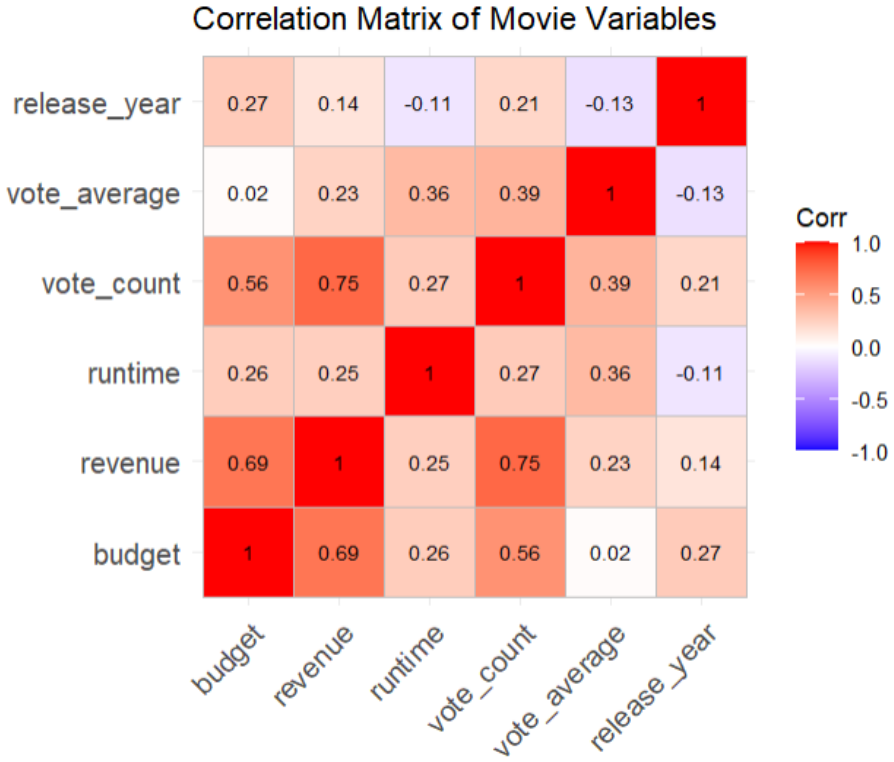


Рисунок 2. Кореляційна матриця

Використано багатофакторну лінійну регресію для дослідження впливу різних змінних, як-от бюджет, рік випуску, кількість голосів оцінки, тривалість і жанр, на рейтинг і касові збори фільмів. Для зручності дослідження впливу жанрів їх було розбито на окремі бінарні змінні. Виявлено значимі фактори, які впливають на успіх фільму.

Регресійна модель для передбачення касових зборів показала, що ключові фактори, як-от бюджет, тривалість фільму і кількість голосів, мають міцний позитивний зв’язок з касовими зборами. Певні жанри, як-от анімація та пригоди, мають статистично значимий позитивний вплив на касовий успіх.

Регресійна модель для передбачення рейтингу показала, що фільми з більшою кількістю голосів мають тенденцію до вищих оцінок, що може відображати схильність популярних фільмів мати вищий рейтинг. Тривалість фільму також має позитивний вплив на рейтинг. Це може бути пов'язано з тим, що довші фільми часто є більш глибокими або детальними в розкритті сюжету і персонажів. Серед жанрів анімація, документальні фільми, драми мають позитивний вплив, тоді як жанри, як-от наукова фантастика, пригоди, комедія, трилери, фентезі і жахи вказують на негативний вплив на середню оцінку.

У майбутніх дослідженнях можна додатково дослідити вплив акторського складу, репутації режисера, сезонності випуску фільму та ін. на його рейтинг та комерційний успіх. На основі проведеного аналізу можна зробити певні рекомендації:

1. Продюсерам та інвесторам треба ретельно планувати бюджет, зважаючи на його вплив на потенційні збори та рейтинг.

2. Важливе інвестування в маркетинг для збільшення кількості голосів та популярності фільму, оскільки це має великий вплив на досліджувані показники.

3. Треба забезпечити оптимальну тривалість фільму, яка забезпечуватиме глибоку розповідь, але не буде втомлювати аудиторію.

Розробники фільмів могли б зосередитися на жанрах, які історично показують високі рейтинги та касові збори.

Список використаних джерел

1. R Programming Language – Introduction. *GeeksforGeeks*. 2021. URL: https://www.geeksforgeeks.org/r-programming-language-introduction/

2. What Is RStudio? A Beginner's Guide. *CareerFoundry*. 2023. URL: https://careerfoundry.com/en/blog/data-analytics/what-is-rstudio/

3. Wickham H. Ggplot2: Elegant Graphics for Data Analysis. *New York: Springer International Publishing*, 2016. 260 с.

4. TMDb. *Вікіпедія*. 2023. URL: https://uk.wikipedia.org/wiki/TMDb

5. Hiller W. What Is Data Cleaning and Why Does It Matter? *CareerFoundry*. 2023. URL: https://careerfoundry.com/en/blog/data-analytics/what-is-data-cleaning/

УДК 004.82:004:85

Бушменьов В. Є., здобувач 2 курсу спеціальності 122 Комп'ютерні науки ОС «Магістр»,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

КЛАСИФІКАЦІЯ ФОТОГРАФІЙ ЗА ОЗНАКОЮ МЕДИЧНОЇ МАСКИ З ВИКОРИСТАННЯМ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Донецький національний університет імені Василя Стуса, м. Вінниця

Штучні нейронні мережі – це комп'ютерні системи, розроблені за зразком біологічних нейронних мереж, які зустрічаються в мозку людини і тварин. Ці системи працюють за допомогою імітованих нейронів. Штучні нейрони як окремо прості та базові елементи можуть об'єднуватися у складні мережі для вирішення складних завдань. Навчання цих мереж полягає у поступовому активуванні та налаштуванні з'єднань між нейронами, що включає налаштування інтенсивності зв'язків з використанням зворотного зв'язку для виправлення помилок і підвищення ефективності. Сучасне використання цієї технології охоплює широкий спектр наукових і технічних галузей [1].

Нейронні мережі використовуються для розпізнавання образів, аналізу відео, автоматичного виявлення та класифікації об'єктів. У медичній галузі згорткові мережі використовуються для аналізу рентгенівських знімків, МРТ, ЦТ для виявлення патологій, як-от рак, зміни, пов'язані з Альцгеймером, та інші захворювання [2].

Вхідний набір даних для моделі складається із 4 категорій фотографій: «Without mask», «With mask», «Without person», «Without mask not trivial». Внаслідок анотування зображень категорії поділяються в зібраному датасеті так, як зображено на рисунку 1.

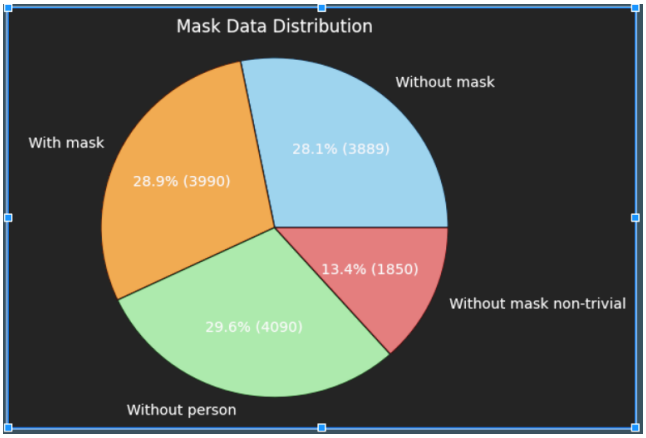


Рисунок 1. Розподіл вхідного потоку даних нейромережі

Для аналізу даних були використані модифіковані базові нейронні мережі з Python-бібліотеки tensorflow.keras. Додаються нові шари до моделі, які будуть відповідальні за вирішення задачі. Ці нові шари будуть навчатися під час процесу файн-тюнінгу. Модель тренується на своїх даних, використовуючи комбінацію заморожених шарів та нових доданих шарів [3]. Процес тренування вимагає підлаштування гіперпараметрів та регуляризації для досягнення найкращих результатів. Модель набрала доволі хороші результати на тестовій виборці та показала чудову швидкість передбачення [4]. Найбільш ефективною у вирішенні цієї задачі виявилася нейронна мережа MobileNet з активаційною функцією ReLU6.

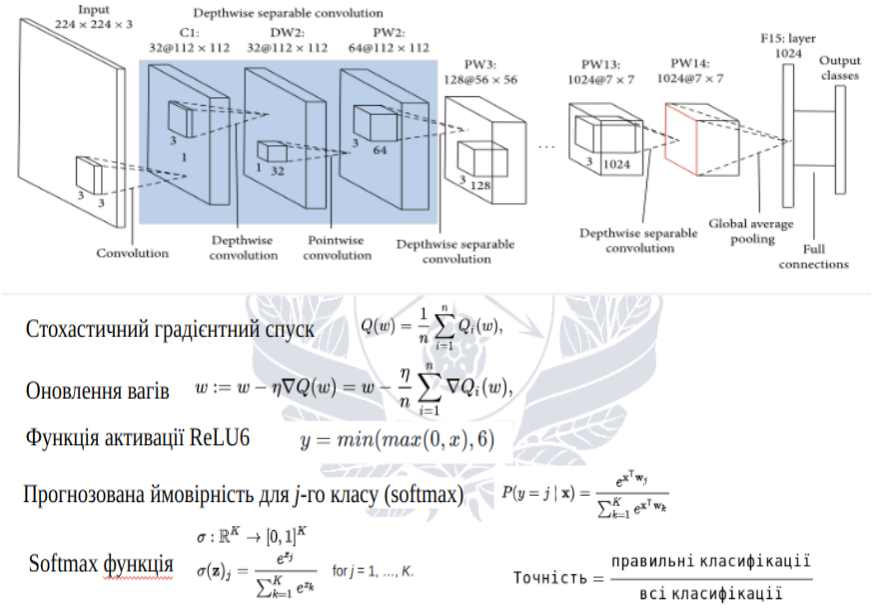
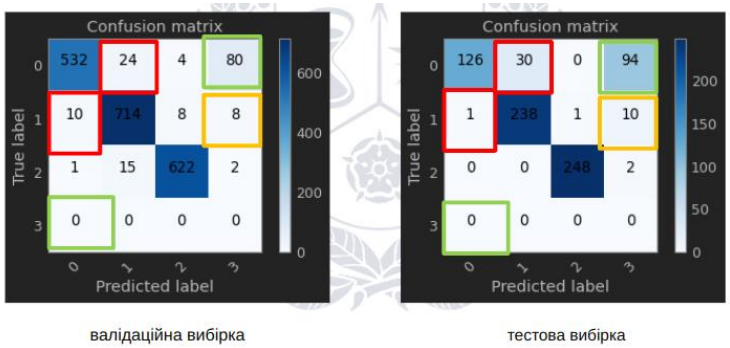


Рисунок 2. Нейронна мережа MobileNet

На рисунку 3 зображено отримані матриці невідповідностей внаслідок тренування моделі. Для моделі: 0 – «Without mask», 1 – «In mask», 2 – «Without person», 3 – «Without person not trivial».



валідаційна вибірка тестова вибірка

Рисунок 3. Матриці невідповідності

Графік зростання точності відносно епох на тренувальній та валідаційній виборці зображено на рисунку 4.

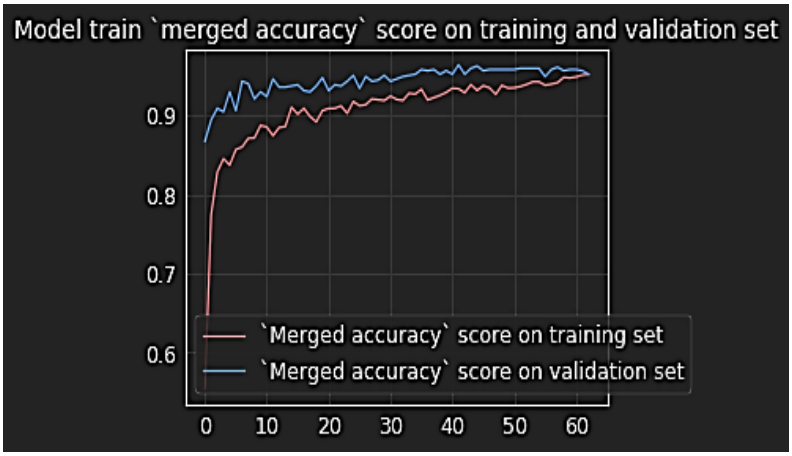


Рисунок 4. Графік зростання точності відносно епох на тренувальній та валідаційній вибірці

Внаслідок цього ми отримали модель, яка добре сприймає будь-який тип фотографій, окрім тих, де людина закриває собі рукою лице поверх маски. Модель набрала хороші результати на тестовій вибірці та показала чудову швидкість передбачення [4, 5].

Отже, реалізовано програмний код, який модифікує та візуалізує результати функціонування нейромережі. Створена модель модифікованої нейронної мережі має точність 94,3 %.

Список використаних джерел

1. Нейронні мережі від теорії до практики. URL: https://www.mql5.com/ru/articles/497

2. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining and Prediction. Springer New York, 2009. 745 p.

3. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. O'Really Media, 2017. 547 p.

4. Provost F., Fawcett T. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking. O'Really Media, 2013. 413 p.

5. Bruce P., Bruce A., Gedeck P. Practical statistics for Data Scientists. O'Really Media, 2020. 360 p.

УДК 004.9+005.7

Векленко С. Ю., здобувач 2 курсу спеціальності 122 Комп'ютерні науки СО «Магістр»,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

АНАЛІЗ ДАНИХ ПІД ЧАС ОЦІНЮВАННЯ ТЕНДЕНЦІЙ ІНВЕСТИЦІЙНИХ ПРОЦЕСІВ У ЗАКЛАДАХ МЕДИЧНОЇ ГАЛУЗІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Розвиток сучасних економічних процесів неможливий без інвестицій у галузі та окремі проєкти. Сьогодні одним із перспективних напрямів інвестування є медична галузь. Усе більша потреба в обслуговуванні хворих робить необхідним запровадження сучасних інформаційних технологій, як основи підтримки роботи з базами даних пацієнтів, їх супроводу та аналізу даних. Систематизація підходу роботи з інформацією дає змогу не тільки підвищувати рівень обслуговування пацієнтів, а й проводити контроль за використанням фінансових надходжень, направлених на підтримку цього процесу. Через це, одним із актуальних питань є запровадження в інформаційних системах закладів охорони здоров'я додаткових модулів аналізу даних інвестиційних надходжень.

Сьогодні інвестиції закладів охорони здоров'я направляються як із державних джерел фінансування, так і за рахунок коштів приватних інвесторів. За даними Міністерства фінансів України у 2022 р. видатки на охорону здоров'я становили 215,3 млрд грн, у тому числі видатки держбюджету разом із трансфертами – 187,2 млрд грн. Одним з основних напрямів залишається реалізація програм медичних гарантій, на яку у 2022 р. було спрямовано 146,3 млрд грн. Зокрема, розподіл коштів складав [1]:

- витрати на оплату медичних послуг за програмою медичних гарантій 143,9 млрд грн;
- на реімбурсацію (відшкодування вартості ліків) для лікування серцево-судинних захворювань, цукрового діабету, хронічних хвороб дихальних шляхів, розладів психіки та поведінки 2,4 млрд грн. Водночас лікарські засоби з повним або частковим відшкодуванням їх вартості отримали понад 3,8 млн осіб.

Треба зазначити, що за рахунок закордонних фінансових вкладень у межах фінансування програми медичних гарантій було виплачено заробітну плату понад 500 тисячам працівникам медичних закладів. Зважаючи на вагомі обсяги фінансових потоків постає питання щодо оцінювання тенденцій та ефективності їх залучення за різними інвестиційними проєктами.

Інформаційний процес у сфері охорони здоров'я визначається складною мережею компонентів, які взаємодіють для забезпечення потрібної інформації для ефективного управління. Одним із ключових етапів є збір даних, який може відбуватися в різних точках системи охорони здоров'я, зокрема і в медичних установах, лабораторіях та ін. Передача даних може бути як локальною, на рівні конкретного підрозділу, так і глобальною, коли інформація передається між різними рівнями системи.

Реалізація процесу аналізу та обробки даних, зокрема інвестиційних вкладень, може бути здійснена через електронну систему охорони здоров'я (E-health), яка розуміє під собою використання інформаційно-комунікаційних технологій для поліпшення рівня охорони здоров'я, зокрема і спосіб мислення й організації процесів у системі охорони здоров'я та пов'язаних сферах (науці, освіті, дослідницькій діяльності). E-health є напрямом, який включає в себе не лише інформаційно-телекомунікаційні системи, але й такі компоненти: органи управління, нормативно-правова база, стандарти і контроль відповідності, кадрові ресурси, інфраструктура, стратегія та модель залучення інвестицій [2].

Сутність обробки та аналізу даних має конкретні характеристики з огляду на особливість направлених фінансових потоків у систему. Так, для державних установ, що за своєю суттю є неприбутковими організаціями, отриманий ефект від фінансування буде оцінюватись повнотою використаних коштів у межах закладеного планового бюджету. Для приватних організацій ідеться про реалізацію інвестиційний проєктів з отриманою часткою прибутку для їх подальшого утримання та функціонування як бізнесу загалом. Тобто під час характеристики особливостей аналізу даних у медичних закладах доцільним є використання принципів розумного інвестування [3], базовим із яких є правильність оцінювання діяльності організації (компанії) в межах проєктованих робіт.

Отже, аналіз даних інвестиційних процесів є одним з основних складників інформаційних технологій, що запроваджуються в сучасних умовах в організаціях медичної галузі, оскільки на нього покладаються основні задачі оцінювання сподівань інвесторів щодо ефективності витрачених коштів.

Список використаних джерел

1. Офіційний сайт Міністерства фінансів України. URL: https://www.mof.gov.ua/uk/news/minfin_u_2022_rotsi_vidatki_na_okhoronu_zdorovia_stanovili_2153_mlrd_griven-3908

2. Радзішевська Є. Б., Висоцька О. В. Інформаційні технології в медицині. *E-health.* Харків: ХНМУ, 2019. 72 с.

3. Buffett W. E. The Essays of Warren Buffett: Lessons for Corporate America. The Cunningham Group & Carolina Academic Press, 2015. 313 p.

УДК 004.056.55

Демашкевич А. В., здобувач 2 курсу спеціальності 113 Прикладна математика,

Антонов Ю. С., канд. фіз.-мат. наук, доцент, доцент кафедри інформаційних технологій

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ПРИХОВУВАННЯ ПОВІДОМЛЕННЯ В ЦИФРОВИХ ЗОБРАЖЕННЯХ ЗА ДОПОМОГОЮ МЕТОДУ LSB

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасну цифрову епоху важливого значення набуває питання захисту інформації від несанкціонованого використання та розповсюдження, коли потрібен цифровий аналог водяних знаків. У деяких ситуаціях необхідно передавати зашифровані повідомлення відкритими каналами зв'язку, приховуючи їх від інших, або навпаки, потрібно мати можливість виявляти такі повідомлення. Через це виникає необхідність вдосконалення засобів приховування повідомлень. Серед наявних засобів перевагу має стеганографія, оскільки інформація може міститися у файлах мультимедіа, які не сприймаються як носії тексту. В Україні методи стеганографії для захисту конфіденційності можуть використовувати і військові, і цивільні особи.

Стеганографією називають сукупність методів та засобів їх реалізації, які базуються на різних принципах і дають змогу приховувати сам факт існування секретної інформації в тому чи іншому середовищі [4]. Методи комп'ютерної стеганографії використовують комп'ютерну техніку та програмне забезпечення для приховування інформації в потоках оцифрованих сигналів. В. О. Денисюк комп'ютерною стеганографією називає фактичне приховування одного файла в іншому [1]. Метою стеганографії є захист інформації від несанкціонованого використання шляхом вбудовування секретних повідомлень в інші дані, відомі як контейнери [2]. Це забезпечує неможливість візуального або технологічного доступу до повідомлень [3].

Аналіз сучасних стеганографічних програм продемонстрував, що вони лише частково відповідають набору вимог. Були виявлені наступні недоліки: використання одного контейнера для приховування одного повідомлення, неможливість приховати великий об'єм інформації без видимих аномалій у контейнері, розробка програм лише на базі одного методу стеганографії без додаткових заходів захисту. Загалом серед тенденцій у комп'ютерній стеганографії можна виділити розповсюджене використання статичних цифрових зображень в якості контейнерів, нестачу надійних програмних стеганографічних засобів у вітчизняному

просторі. Розробка програмного забезпечення на основі стеганографічного методу для приховування повідомлень є досить актуальним завданням.

Перевагою обраного методу LSB (Least Significant Bit) є універсальність та простота, оскільки його можна використовувати в програмах з різними типами контейнерів, як-от цифрові зображення, аудіофайли та відеофайли. Він є поширеним серед методів заміни в просторовій множині. Сутність цього методу полягає в заміні найменш значимих бітів у контейнері (зображення, аудіо- або відеозапису) на біти приховуваного повідомлення. Різниця між порожнім і заповненим контейнерами непомітна для органів чуття людини [1]. Зважаючи на те, що кожна точка в RGB-зображенні кодується кількома байтами, кожен байт визначає інтенсивність червоного (Red), зеленого (Green) і синього (Blue) кольору. Сукупність інтенсивностей кольору в кожному з 3-х каналів визначає відтінок пікселя. Найменш значимі розряди меншою мірою впливають на підсумкове зображення. З цього можна зробити висновок, що заміна одного або двох молодших, найменш значимих бітів, на інші довільні біти мало спотворить відтінок пікселя (рис. 1).

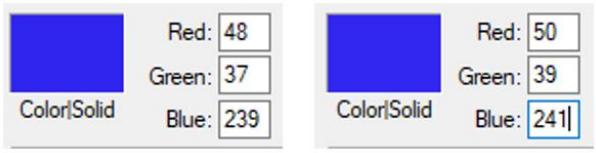


Рисунок 1. Порівняння двох відтінків

Розглянемо ключовий алгоритм програмного засобу, а саме приховування повідомлення в цифрових зображеннях. Колірна компонента кожного пікселя описується одним байтом, модифікації підлягає останній біт. Алгоритм розподілу повідомлення між контейнерами базується на пропорційному підході. У разі використання в одному наборі кількох зображень, різних за розміром, у менший контейнер програма розподілить меншу кількість інформації, відповідно в більший – більший обсяг інформації. Формула (1) описує знаходження коефіцієнта пропорційності, формула (2) визначає довжину підрядка для контейнера:

$$k = \frac{\sum_{i=1}^{n} l_i}{l_t}; \quad (1)$$

$$l_{t_i} = \frac{l_i}{k}, \quad (2)$$

де n – кількість контейнерів; l_i – максимально допустима довжина повідомлення i-му контейнері; l_t – довжина вихідного повідомлення; l_{t_i} – довжина частини повідомлення в i-му контейнері, k – коефіцієнт.

У заповненому контейнері 15 пікселів першого рядка містять системну інформацію, а саме: наявність прихованого повідомлення, порядковий номер частини повідомлення, загальну кількість частин вихідного повідомлення, довжину прихованого повідомлення в контейнері. Максимальний обсяг повідомлення для одного контейнера розраховується за кількістю його пікселів.

Розглянемо алгоритм приховування повідомлень у контейнери (рис. 2). Основними кроками є вибір та завантаження пустих контейнерів та повідомлення. Додатковим рівнем захисту є наявність ключа шифрування. Використовуючи стандартну бібліотеку для криптографічних операцій System.Security.Cryptography, вихідний текст шифрується.

Рисунок 2. Блок-схема приховування повідомлення

Процес вилучення повідомлення будується в аналогічний спосіб. Спочатку проводиться завантаження контейнерів, далі вводиться ключ шифрування, частини повідомлення з різних контейнерів об’єднуються, текст дешифрується, отримуються дані з контейнерів. Якщо під час перевірки контейнерів програма

не виявила жодного заповненого з наявних, тобто завантажені пусті зображення, користувач отримує повідомлення про відсутність прихованої інформації.

На основі викладеного алгоритму розроблено програмне забезпечення на мові програмування C# (рис. 3). У процесі проєктування використана модель багатошарової архітектури з урахуванням особливостей впровадження та масштабування у майбутньому. У програмі під час вибору контейнерів бажано використовувати зображення форматів PNG та BMP.

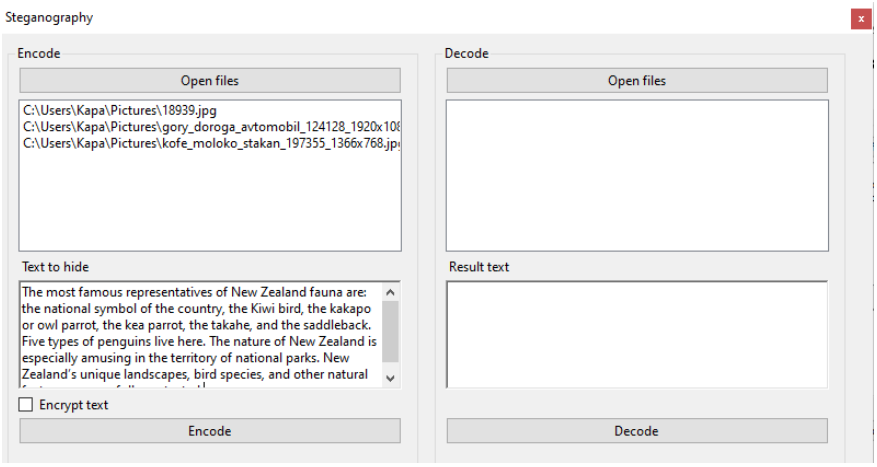


Рисунок 3. Головне вікно програми

На рис. 4 показані контейнери для зрівняння, ліворуч – незаповнений контейнер, праворуч – заповнений. Як бачимо, заповнений контейнер візуально майже не відрізняється від пустого.

Рисунок 4. Порівняння пустого та заповненого контейнерів

Отже, проблему оптимального приховування повідомлень у серії зображень за допомогою методу LSB можна вважати вирішеною.

Список використаних джерел

1. Хорошко В. О., Яремчук Ю. Є., Карпінець В. В. Комп’ютерна стеганографія: навч. посіб. Вінниця: ВНТУ, 2017. 155 с.

2. Денисюк В. О. Стеганографічний алгоритм захисту даних з використанням файлів зображень. *Ефективна економіка.* 2017. № 5. URL: http://www.economy.nayka.com.ua/?op=1&z=5584 (дата звернення: 30.10.2023).

3. Кошкіна Н. В. Стегоаналіз цифрових зображень із застосуванням контрольного вкраплення. *Захист інформації і безпека інформаційних систем:* матеріали III міжнародної науково-технічної конференції. Львів, 2014. С. 98–100.

4. Римар П. В., Крохмалюк В. В. Атаки на стеганосистеми. Криптографічні атаки. *Матеріали наукової конференції професорсько-викладацького складу, наукових працівників і здобувачів наукового ступеня за підсумками науково-дослідної роботи за період 2019–2020 рр.* (квітень–травень 2021 р.). Вінниця: ДонНУ імені Василя Стуса, 2021. С. 344–346.

УДК 004.9+005.5

Колосова К. К., здобувач 1 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

РОЛЬ SEO У ПРОСУВАННІ ТА ПІДВИЩЕННІ ВІДВІДУВАНОСТІ ВЕБСАЙТІВ ІТ-КОМПАНІЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

SEO-просування – це сукупність методів оптимізації вебсайта з метою підвищення його видимості в пошукових системах. Це означає, що відповідні запити користувачів, які стосуються сайта, будуть відображатися на вищих позиціях у результатах пошуку. Оптимізація вебсайта включає внутрішні та зовнішні елементи, які впливають на його рейтинг у пошукових системах.

Просування пошукових систем (SEO) дає змогу збільшити кількість відвідувачів сайта, залучити більше потенційних клієнтів і підвищити конверсію. Оскільки він підвищує вплив бренду, збільшує продажі та покращує конкурентоспроможність на ринку, це важливий інструмент для будь-якого бізнесу [1].

Основні аспекти SEO для ІТ-компаній включають технічну оптимізацію вебсайта (швидкість завантаження, мобільна сумісність, правильна структура URL), вибір правильних ключових слів та створення відповідного контенту, побудову якісних посилань з авторитетних джерел, а також постійний аналіз і вдосконалення стратегій SEO на основі вебаналітики та даних про користувачів. Ці аспекти є важливими для підвищення видимості в пошукових системах та привертання цільового трафіка на вебсайт ІТ-компанії.

Розглянемо більш детально основні аспекти SEO для IT-компаній:

➢ Технічна оптимізація вебсайта. Технічна оптимізація включає ряд заходів, спрямованих на покращення технічних аспектів вебсайта з метою підвищення його ефективності у пошукових системах. Це охоплює аспекти швидкості завантаження сторінок, мобільної сумісності (респонсивного дизайну), правильної структуру URL-адрес, оптимізації метатегів (заголовків, описів) і використання правильних HTML-тегів.

➢ Ключові слова та контент. Ключові слова є фундаментальним елементом стратегії SEO. Це терміни чи фрази, які користувачі вводять у пошукових системах для знаходження інформації. IT-компанії повинні ретельно досліджувати та обирати ключові слова, які найкраще відображають їх діяльність і послуги. Контент, що містить ці ключові слова у заголовках, метатегах, тексті сторінок, зображеннях та відеоматеріалах, сприяє підвищенню рейтингу в пошукових системах.

➢ Зовнішня оптимізація та посилання. Зовнішня оптимізація SEO включає побудову посилань (backlinking) з авторитетних джерел, які посилюють авторитет вашого вебсайта в очах пошукових систем. До того ж співпраця з іншими вебсайтами та блогами для поширення вмісту і створення посилань також допомагає підвищити видимість вашого сайта в пошукових системах.

➢ Аналітика та вдосконалення. Одним із ключових складників SEO є постійний аналіз та вдосконалення стратегій. Використання інструментів веб-аналітики для вимірювання ефективності стратегій SEO дає змогу зрозуміти, що працює, а що потребує покращення. Регулярне оновлення та оптимізація контенту, а також аналіз даних про трафік і поведінку користувачів є важливими етапами в постійному удосконаленні стратегій SEO для підвищення видимості в пошукових системах.

Ці аспекти є важливими для будь-якої IT-компанії, оскільки правильна реалізація SEO-стратегій дає змогу підвищити видимість бренду, залучити більше цільового трафіку та підвищити конверсію на вебсайті [2].

Для успіху IT-компаній стратегії пошукової оптимізації надзвичайно важливі. По-перше, їх мета полягає в тому, щоб підвищити видимість у пошукових системах. Це означає, що цільова аудиторія привертає увагу за допомогою оптимізації контенту та ключових слів. Це збільшує кількість людей, які відвідують вебсайт, і потенційних клієнтів. По-друге, SEO-техніки спрямовані на покращення досвіду користувача. Сайт, оптимізований для мобільних пристроїв, має швидке завантаження та зручну навігацію, що приваблює користувачів і позитивно впливає на їх досвід і сприйняття бренду.

Отже, добре підібрана стратегія оптимізації пошукових систем збільшує кількість переглядів. Це означає, що більше людей, які заходять на вебсайт

через пошукові системи, роблять бажані речі, як-от купівля товарів, реєстрація чи запитання про послуги. Використання ефективних SEO-технік може значно вплинути на успіх IT-компаній, надаючи їм високий рівень конкурентоспроможності та впевненість у своєму місці на ринку [3].

Список використаних джерел

1. SEO просування: оптимізація та розкрутка сайту. URL: https://webdevandseo.com/seo-promotion-optimization-and-promotion-of-the-site/

2. Базове SEO для вебсайтів. URL: https://www.whitepress.com/ua/baza-znan/166/bazove-seo-dlia-vebsaitiv

3. The Importance of SEO Analysis: Unveiling Opportunities and Weaknesses. URL: https://aicontentfy.com/en/blog/importance-of-seo-analysis-unveiling-opportunities-and-weaknesses

УДК 004:659.4

Огороднік М. О., здобувач 1 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

РОЛЬ INFLUENCE-МАРКЕТИНГУ У СФЕРІ УПРАВЛІННЯ IT-ПРОЄКТАМИ

Донецький національний університет імені Василя Стуса, м. Вінниця

У світі стрімкого інформаційного розвитку управління IT-проєктами IT-маркетинг є невід'ємною частиною повного циклу розробки будь-якого інформаційного продукту. Одним із важливих складників цього процесу є сфера маркетингу, що виконує багато функцій: стратегічне планування та позиціонування, комунікацію та спілкування із зацікавленими особами, привертання та залучення користувачів продуктів, аналіз ринку та конкурентів, супровід та підтримку продукту на ринку. Оскільки маркетинг допомагає не тільки гарантувати успішне впровадження, а й створити високий рівень зацікавленості та підтримки з боку користувачів, включення плану маркетингу у процес управління IT-проєктами стає важливою умовою досягнення успіху.

Виділяють різні типи напрямів IT-маркетингу в залежності від очікуваних результатів. Одним із ефективних напрямів є influence-маркетинг, який є частиною brand-маркетингу. Brand-маркетинг відповідає за побудову іміджу та репутації бренду чи продукту, підвищення обізнаності про продукт і роботу

над довірою та лояльністю користувачів. Influence-маркетинг є окремим напрямом brand-маркетингу, що працює власне у соціальних медіа, метою якого є побудова партнерських відносин між брендом та інфлюенсером.

Інфлюенсери – це люди, які завдяки своїй популярності, авторитету, досвіду, знання та репутації можуть впливати на думку і поведінку суспільства. Також їх називають лідерами думок, оскільки їхня сила полягає в здатності впливати на свою аудиторію, роблячи її потенційно важливими фігурами для брендів і маркетологів. До інфлюенсерів відносять блогерів, зірок, артистів, авторів контенту, експертів у різноманітних галузях тощо.

Незадовго до технологічного прогресу телебачення було фактично єдиним ЗМІ, до якого мали доступ споживачі, і це було однією з основних платформ, які використовували для реклами масам. Із появою інтернету та популярністю платформ соціальних медіа споживачі мають необмежену свободу та вибір вмісту, який вони бажають переглядати. Тому постає проблема складності охоплення цільової аудиторії. Реклама через інфлюенсерів дає змогу брендам просуватись через когось, із ким взаємодіє та якому довіряє суспільство [1].

Індустрія інфлюенсерського маркетингу стабільно зростає та стала популярною за останнє десятиліття. Для підтвердження цього факту можна знайти відповідну інформацію на рис. 1 зі звіту сервісу HubSpot «State of Influencer Marketing 2023: Benchmark Report»: ця індустрія виросла на 21,1 млрд доларів США, що є значним збільшенням (на 29 %), у порівняно з 16,4 млрд доларів у попередньому році [2].

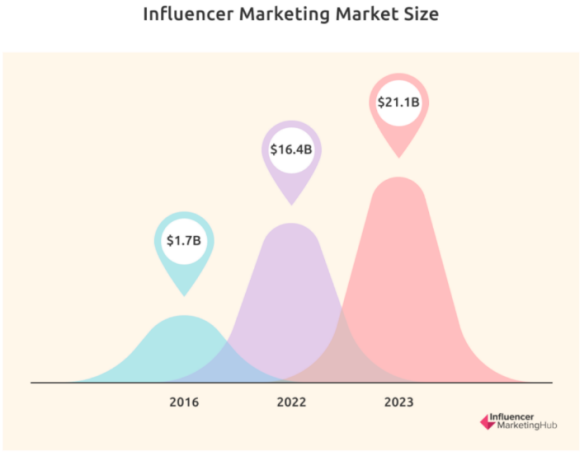


Рисунок 1. Розмір ринку influence-маркетингу за останні роки у доларах США

Оскільки лідери думок можуть мати різну кількість аудиторії в соціальних мережах, було створено класифікацію інфлюенсерів: наноінфлюенсери; імікронфлюенсери; макроінфлюенсери; мегаінфлюенсери.

Згідно зі звітом сервісу HubSpot «State of Influencer Marketing 2023: Benchmark Report», компанії у 2022 р. працювали з інфлюенсерами, що мали різні розміри

аудиторії, але найкращий успіх у партнерських відносинах 38 % компаній мали з макроінфлюенсерами, 33 % – з мікроінфлюенсерами, 18 % з мегаінфлюенсерами та 11 % з наноінфлюенсерами [1].

Під час роботи у сфері influence-маркетингу виділяють переваги: довготривалий ефект у процесі співпраці, швидка ефективність, креативність і оригінальність. До недоліків можна віднести інколи занадто високі ціни співпраць, невідомість щодо аудиторії інфлюенсера та зміни в алгоритмах соціальних мереж, що можуть впливати на результати співпраць [3].

Отже, можна зробити висновки, що незважаючи на деякі недоліки, influence-маркетинг залишається важливим інструментом управління ІТ-проєктами.

Список використаних джерел

1. The State of Marketing. Marketing Trends in 2023, from AI to Z. URL: https://www.hubspot.com/hubfs/2023%20State%20of%20Marketing%20Report.pdf (дата звернення: 01.12.2023).

2. The State of Influencer Marketing 2023: Benchmark Report. URL: https://influencermarketinghub.com/influencer-marketing-benchmark-report/ (дата звернення: 01.12.2023).

3. What is Influencer Marketing? The Ultimate Guide for 2024. URL: https://influencermarketinghub.com/influencer-marketing/ (дата звернення: 01.12.2023).

УДК 004.42:517.5

Римар П. В., старший викладач кафедри інформаційних технологій,
Сеник І. О., асистент кафедри інформаційних технологій

ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ MAPLE ПІД ЧАС ПОБУДОВИ ГРАФІКІВ ФУНКЦІЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

Maple – це потужне математичне програмне забезпечення, розроблене для вирішення різноманітних завдань у галузі математики та інженерії. Заснована на потужному символьному обчисленні, ця система дає змогу користувачам виконувати різноманітні обчислення, алгебраїчні та числові операції, а також вивчати та візуалізувати математичні концепції. Maple пропонує багато функцій для аналізу, маніпуляції та вивчення математичних об’єктів. Спеціалізовані інструменти дають змогу будувати графіки функцій, розв’язувати рівняння та

системи рівнянь, знаходити похідні та інтеграли, а також виконувати чисельні обчислення. Це програмне забезпечення використовується в навчанні, дослідженнях та професійній практиці, де важливою є потреба в точних та ефективних обчисленнях. Використання Maple дає змогу швидко отримувати результати, вдосконалювати та візуалізувати математичні концепції, що сприяє зрозумінню і розв'язанню складних математичних задач [1].

Використання комп'ютерного програмного забезпечення, зокрема Maple, значно полегшує та покращує процес вивчення математики. Використання графічних можливостей Maple допомагає створювати візуальні представлення математичних об'єктів та концепцій. Це сприяє студентами кращому розумінню графічних властивостей функцій, геометричних формул та інших математичних аспектів. Maple надає зручний інтерфейс для розв'язання різноманітних математичних задач. Maple володіє потужними алгоритмами для роботи зі складними математичними конструкціями. Використання цього програмного забезпечення дає змогу швидко та ефективно виконувати завдання, які можуть виявитися складними без використання комп'ютерної підтримки.

Процес побудови графіків функцій у середовищі Maple інтуїтивний та зручний [2]. Нижче подано загальний опис цього процесу:

1. Введення функції. Користувач може ввести математичну функцію або вираз у вікні Maple, використовуючи стандартний математичний синтаксис. Наприклад: plot([cos(5*t), sin(3*t), t=0..2*Pi]).

2. Виклик графічного інтерфейсу. Після введення функції користувач викликає графічний інтерфейс для побудови графіків. Це може бути зроблено через меню або за допомогою команди.

3. Вибір параметрів графіка. Користувач може налаштувати різні параметри графіка, діапазон значень аргументу, колір лінії, товщина лінії та ін.

4. Побудова графіка. Після введення функції та вибору параметрів, Maple автоматично будує графік функції у графічному вікні. Графік може бути побудований для одного або кількох аргументів, залежно від потреби [3].

5. Відображення та аналіз графіка. Користувач може взаємодіяти з отриманим графіком, масштабувати його, переміщувати, виводити координати точок, а також використовувати інші інструменти для аналізу графіка.

6. Збереження та експорт. Після виконання користувач може зберегти графік у вигляді зображення або експортувати його у різноманітні формати для використання в інших документах чи презентаціях.

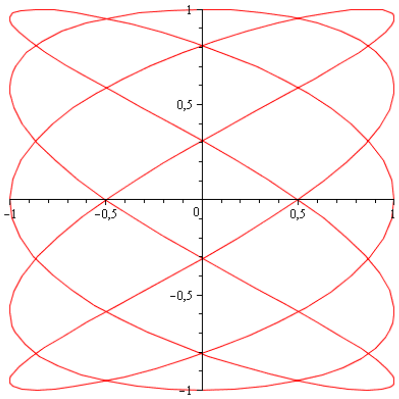


Рисунок 1. Графік побудованої в Maple функції, яка задана у параметричному вигляді

Процес побудови графіків у Maple не лише дає змогу візуалізувати функції, але й допомагає вивчати їх властивості та здійснювати аналіз за допомогою різних інструментів, що робить вивчення математики більш доступним та ефективним.

Список використаних джерел

1. Графіка в Mapple. URL: https://jak.koshachek.com/articles/grafika-v-maple-studopedija.html

2. Mapple використання функцій та побудова графіків. URL: https://bondarenko.dn.ua/maple-issledovanie-funktsij-i-postroenie-grafikov/

3. Побудова графіків в Mapple. URL: https://jak.bono.odessa.ua/articles/pobudova-grafikiv-v-paketi-maple.php

УДК 004.01:005

Чайковський П. А., здобувач 1 курсу ОС «Магістр» спеціальності 122 Комп'ютерні науки,

Потапова Н. А., канд. екон. наук, доцент, доцент кафедри інформаційних технологій

ВИКЛИКИ ДИСТАНЦІЙНОГО УПРАВЛІННЯ ІТ-ПРОЄКТАМИ

Донецький національний університет імені Василя Стуса, м. Вінниця

Поява дистанційної роботи, прискорена пандемією COVID-19, трансформувала ландшафт управління проєктами в секторі ІТ. Хоча цей зсув пропонує переваги, як-от зниження витрат і доступ до глобального таланту, він вводить унікальні виклики, з якими керівникам проєктів потрібно вміло маневрувати [1]. Основні виклики управління дистанційною роботою наведено на рис. 1.

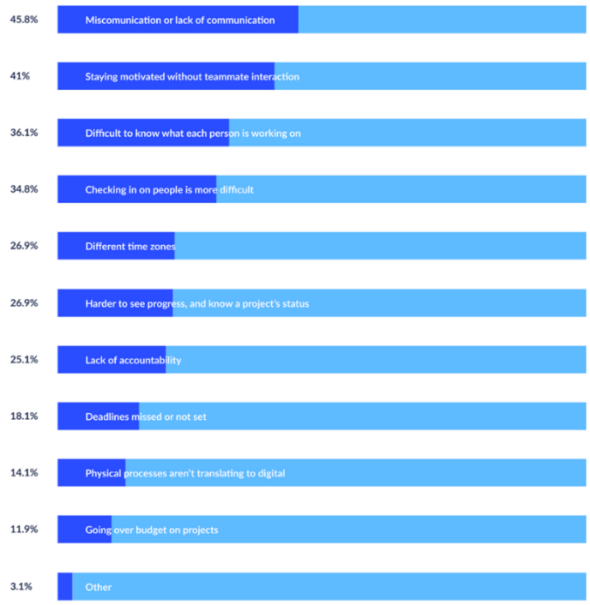


Рисунок 1. Основні виклики управління дистанційної роботи [2]

Одним із найсуттєвіших викликів у дистанційному управлінні ІТ-проєктами є потенційне непорозуміння. Без нюансів особистого спілкування можуть виникати непорозуміння, що призводить до зміни обсягу або конфліктів. Цей виклик посилюється в ІТ-проєктах, де точні технічні вимоги та чітке спілкування є необхідними для успіху. До того ж неможливість фізичної присутності може спричинити відчуженість членів команди, впливаючи на моральний дух та продуктивність. Керівникам проєктів необхідно ефективно використовувати інструменти спілкування та сприяти культурі відкритого діалогу для зменшення цих ризиків.

Координація у різних часових зонах становить логістичний виклик. Це може призвести до затримок, оскільки члени команди чекають на внески від колег з інших часових зон, потенційно призводячи до пропущених крайніх термінів. Ефективні керівники проєктів протидіють цьому, плануючи спільний робочий час, де це можливо, та встановлюючи чіткі очікування щодо часу відповіді [3].

Дистанційна робота значною мірою покладається на технології, що робить технічні проблеми критично важливими. Різна якість інтернету та доступ до необхідних інструментів можуть порушити робочі процеси. До того ж забезпечення безпеки даних стає складнішим у децентралізованому середовищі, де члени команди отримують доступ до конфіденційної інформації з різних місць. Керівникам проєктів потрібно забезпечити надійну ІТ-підтримку та дотримуватися строгих протоколів безпеки даних.

Знаходження балансу між моніторингом продуктивності та автономією є делікатним завданням у дистанційних умовах. Надмірний контроль може призвести до зниження морального духу, тоді як недостатній контроль може

вплинути на прогрес проєкту. Керівникам проєктів треба встановлювати чіткі метрики продуктивності та довіряти своїй команді, бути готовими надавати підтримку за потреби [3].

Адаптація нових членів команди на відстані не передбачає безпосередності та ефективності особистого навчання. Формування згуртованості команди також є викликом без регулярних особистих взаємодій. Для вирішення цього керівникам проєктів треба розробляти всебічні віртуальні програми адаптації та сприяти віртуальним командоутворюючим заходам [3].

Відсутність чітких цілей може призвести до безцільних проєктів. У дистанційних умовах, де прямий нагляд обмежений, це стає більш вираженим. Керівникам проєктів треба забезпечити, щоб цілі та завдання були чітко визначені і передані всім членам команди [4].

Дистанційне управління проєктами часто вимагає додаткових ресурсів для засобів спілкування та співпраці. Обмеження бюджету можуть перешкоджати придбанню цих необхідних ресурсів. До того ж недостатнє управління ризиками може призвести до провалу проєктів, особливо в дистанційних умовах, де передбачення та зменшення ризиків є складнішими. Ефективні керівники проєктів повинні відстоювати достатні бюджети та розробляти міцні стратегії управління ризиками.

Забезпечення володіння членів команди необхідними навичками, є життєво важливим. У дистанційних умовах виявлення та усунення прогалин у навичках може бути складним. Так само важливо підтримувати відчуття відповідальності, оскільки дистанційна робота іноді може призводити до розподілу відповідальності, і допомогти у цьому може регулярне оцінювання навичок та чітке визначення ролей і обов’язків.

Отже, дистанційне управління ІТ-проєктами включає в себе низку викликів, що вимагають обдуманих і проактивних стратегій. Вирішуючи питання, пов’язані з комунікацією, різницею в часових зонах, технічними викликами та динамікою команд, керівники проєктів можуть успішно маневрувати в цьому складному ландшафті. Оскільки ІТ-індустрія продовжує охоплювати дистанційну роботу, розробка ефективних практик дистанційного управління проєктами буде ключовою для забезпечення успіху проєктів.

Список використаних джерел

1. Sebastien Tang – Remote Work. URL: https://www.linkedin.com/posts/sebastien-tang-salesforce_think-remote-project-management-is-a-walk-activity-7094331569217155072-ayA2 (дата звернення: 24.11.2023).

2. State of Remote Management. URL: https://hubstaff.com/tasks/state-of-remote-project-management (дата звернення: 24.11.2023).

3. Nimble Network – Project Management Strategies. URL: https://www.nimblework.com/blog/project-management-strategies-remote-first/ (дата звернення: 24.11.2023).

4. Outsource Access – Remote project management Challenges. URL: https://www.linkedin.com/pulse/how-address-10-remote-project-management-challenges-outsourceaccess-a6bwc?trk=public_post_feed-article-content (дата звернення: 24.11.2023).

УДК 004.67:004.42

Ярош О. Л., здобувач 2 курсу спеціальності 122 Комп'ютерні науки ОС «Магістр»,

Бабаков Р. М., д-р техн. наук, доцент, професор кафедри інформаційних технологій

КРОСПЛАТФОРМОВИЙ API ДЛЯ СИСТЕМИ РОЗПОДІЛУ ЗАВДАНЬ ТА АНАЛІЗУ ПРОДУКТИВНОСТІ ВИКОНАВЦІВ

Донецький національний університет імені Василя Стуса, м. Вінниця

Кросплатформовий API для системи розподілу завдань та аналізу продуктивності може бути корисним для розробників або команд, які прагнуть створити власні додатки, або інтегрувати функціональність розподілу завдань та аналізу продуктивності виконавців у свої проєкти. Він надає зручний спосіб взаємодії з системою без необхідності використання графічного інтерфейсу користувача.

Через масові переміщення громадян через війну в Україні такий додаток стає ще більш актуальним. Війна і переміщення можуть суттєво вплинути на організацію та ефективність роботи команд, особливо якщо члени команди знаходяться в різних географічних регіонах або працюють на віддалених робочих місцях.

API дає змогу забезпечити комунікацію та співпрацю між виконавцями, незалежно від їх місцезнаходження. Це допоможе ефективно розподіляти завдання між членами команди, забезпечувати контроль та відстеження їх виконання, незалежно від того, де знаходяться виконавці. Особливо у випадку переміщених громадян, які можуть працювати у нових умовах та середовищах, такий API допоможе забезпечити стабільність та організацію роботи. Він дасть змогу керувати завданнями, встановлювати пріоритети, розподіляти ресурси та контролювати виконання завдань у режимі реального часу.

До того ж аналіз продуктивності виконавців стає важливим у разі переміщень і змін у робочих умовах. Він дасть змогу оцінити ефективність і результативність виконавців, незалежно від їх географічного місцезнаходження. Це допоможе виявити проблемні моменти, покращити процеси та забезпечити оптимальне використання ресурсів.

Основні компоненти, які можуть бути включені у кросплатформовий API для системи розподілу завдань та аналізу продуктивності виконавців, такі:

1. Модуль розподілу завдань. Цей модуль відповідає за прийом та розподіл завдань між виконавцями. Він може включати функції створення завдань, призначення виконавців, контролю стану виконання завдань та сповіщення про зміни.

2. Модуль аналізу продуктивності. Цей модуль збирає та аналізує дані про продуктивність виконавців. Він може включати функціональність збору статистики, вимірювання часу виконання завдань, оцінки ефективності та генерації звітів.

3. Модуль автентифікації та авторизації. Цей модуль забезпечує механізми перевірки прав доступу та ідентифікації користувачів API. Він дає змогу обмежувати доступ до функцій API лише авторизованим користувачам та контролювати їхні дії.

4. Модуль комунікації. Цей модуль відповідає за комунікацію між додатками та системою розподілу завдань. Він може підтримувати різні протоколи передачі даних, як-от HTTP, WebSocket або інші, залежно від вимог системи.

5. Документація та приклади використання. Важливим аспектом кросплатформового API є наявність документації, яка описує доступні функції, параметри та способи використання API. Також можуть бути надані приклади коду для допомоги розробникам у розумінні та використанні API.

Переваги використання кросплатформового API для системи розподілу завдань та аналізу продуктивності виконавців включають:

- універсальність (кросплатформовий API може бути використаний на різних платформах та мовах програмування, що робить його більш доступним для розробників з різних середовищ);
- легкість інтеграції (за допомогою API розробники можуть швидко інтегрувати функціональність розподілу завдань та аналізу продуктивності виконавців у свої додатки без необхідності розробки всієї системи з нуля);
- розширюваність (кросплатформовий API може бути розширений та модифікований для врахування специфічних потреб користувачів і додатків);
- централізоване керування (використання API дає змогу централізовано керувати розподілом завдань і аналізом продуктивності виконавців, що полегшує адміністрування та моніторинг цих процесів).

Однак під час розробки кросплатформового API необхідно враховувати такі фактори:

➢ безпека (API повинен мати механізми захисту від несанкціонованого доступу та зловживання);

➢ масштабованість (API повинен бути спроєктований так, щоб витримувати великі навантаження та забезпечувати ефективну роботу системи);

➢ сумісність (API повинен бути сумісним із різними версіями програмного забезпечення і здатним легко адаптуватися до змін у системі розподілу завдань та аналізу продуктивності виконавців).

Список використаних джерел

1. The Future of Remote Working the good, the bad and the ugly. URL: https://luminalearning.com/the-future-of-remote-working-the-good-the-bad-and-the-ugly/

2. Organizations are embracing shifts to remote work, presenting opportunities for tech companies. URL: https://www.businessinsider.com/work-from-home-presents-opportunity-for-tech-providers-2020-5

3. What is REST. URL: https://restfulapi.net/

4. Project Management System: Definition & Features. URL: https://www.proofhub.com/articles/project-management-system

5. Clean Architecture. URL: https://github.com/jasontaylordev/CleanArchitecture

УДК 004.934

Гуменюк К. В., здобувачка 4 курсу спеціальності 122 Комп'ютерні науки,
Сеник І. О., асистент кафедри інформаційних технологій

ВАРІАНТИ ВИКОРИСТАННЯ NLP

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі штучний інтелект (ШІ) і технології обробки природної мови (NLP) [1] виступають у ролі ключових факторів розвитку інновацій та автоматизації. NLP відкриває безліч можливостей для вдосконалення комунікації та оптимізації бізнес-процесів [2]. Існує безліч сценаріїв, де ці технології вже знаходять своє застосування та принесли суттєві покращення. Інновації розпізнавання мови та розуміння контексту дають змогу створювати продукти, які взаємодіють із користувачами більш інтуїтивно та ефективно.

Обробка природної мови (NLP) належить до галузі комп'ютерних наук, а точніше, до галузі штучного інтелекту (ШІ), яка займається наданням комп'ютерам здатності розуміти текст і усне мовлення так само, як це роблять люди.

NLP поєднує комп'ютерну лінгвістику – моделювання людської мови на основі правил – зі статистичними методами, машинним навчанням (МН) і моделями глибокого навчання [3]. Разом ці технології дають змогу комп'ютерам обробляти людську мову у вигляді тексту або голосових даних і розуміти її повне значення, із мотивами того, хто говорить чи пише.

Обробка природної мови є рушійною силою машинного інтелекту в багатьох сучасних сферах застосування. Розглянемо декілька варіантів використання NLP. Одним зі способів є виявлення спаму. Ви можете не вважати розпізнання спаму завданням NLP, але більшість технологій використовують можливості NLP для розпізнавання текстів, щоб сканувати електронні листи на наявність ознак, які часто вказують на спам або фішинг. Такими ознаками можуть бути зловживання фінансовими термінами, погрози, недоречна наполегливість, неправильне написання назв компаній та багато іншого. Виявлення спаму – одне з небагатьох завдань NLP.

Ще одним варіантом є машинний переклад. Google Translate – приклад широко доступної технології NLP у практичному застосуванні. Насправді ефективний машинний переклад передбачає більше, ніж просто заміну слів однієї мови на слова іншої. Ефективний переклад повинен точно передавати зміст і відтінок вхідної мови і перекладати його з тим самим змістом і бажаним ефектом. Сучасні інструменти машинного перекладу успішно працюють над цією задачею, покращуючи якість перекладу та забезпечуючи більш правильне відтворення сенсу і стилістичних особливостей вихідного тексту. Результативність машинного перекладу можна визначити не лише за точністю передачі інформації, але і за здатністю зберегти відтінок емоцій та індивідуальний стиль автора.

Віртуальні агенти, як-от Siri від Apple та Alexa від Amazon, використовують розпізнавання мови для ідентифікації голосових команд і формування природної мови, для реагування на запити відповідними діями чи корисними коментарями. Ця техніка базується на глибокому навчанні та обробці природної мови (NLP), де моделі вивчають синтаксичні і семантичні структури мови для кращого розуміння користувацьких запитів. Чат-боти демонструють таку ж саму техніку, реагуючи на введені текстові повідомлення. Найліпші серед них використовують алгоритми машинного навчання, які не лише адаптуються до конкретного користувача, але й аналізують та узагальнюють широкий спектр лінгвістичних концепцій.

Ще одним зі способів є узагальнення тексту, що використовує методи NLP для обробки величезних обсягів цифрового тексту і створення підсумків та коротких описів для дослідницьких баз даних або для завантажених користувачів, які не мають часу на читання повного тексту. Найкращі програми для узагальнення тексту використовують методи семантичного аналізу та генерації при-

родної мови (NLG) [4], щоб додати до результатів узагальнення корисну інформацію та висновки.

Аналіз думок у соціальних мережах із використанням технології обробки природної мови (NLP) став необхідним інструментом для багатьох компаній у сучасному бізнесі. Ця технологія дає змогу вивчати та розуміти мову, яку користувачі використовують у своїх постах, відгуках та рецензіях, щоб виявити їхні ставлення, емоції та вподобання. Завдяки аналізу думок у соціальних мережах за допомогою NLP компанії мають можливість отримувати цінні інсайти щодо сприйняття своїх продуктів, акцій та подій. Розкриття реальних відгуків та реакцій користувачів допомагає підприємствам адаптувати свої стратегії розвитку, виправляти недоліки та покращувати якість своїх товарів та послуг.

Технологія обробки природної мови знаходить широке застосування в різних сферах. NLP вдосконалює комунікацію, оптимізує бізнес-процеси та сприяє автоматизації завдань, що раніше вимагали значних людських ресурсів. Завдяки постійному розвитку та вдосконаленню обробка природної мови продовжує відкривати нові можливості для технологічного прогресу. Використання NLP стає необхідністю для компаній, які прагнуть залишатися конкурентоспроможними та адаптуватися до бізнес-середовища, що швидко змінюється.

Список використаних джерел

1. Jurafsky D., Martin J. Speech and Language Processing: An Introduction to Natural Language Processing, Pearson, 2008. С. 265–337.

2. Eisenstein J. Introduction to Natural Language Processing, MIT Press, 2019. С. 48–59.

3. Goldberg Y. Neural Network Methods in Natural Language, 2017.

4. Processing, Morgan and Claypool Publishers, 2017. С. 2–6.

5. Areces C., Koller, A., Striegnitz K. Referring expressions as formulas of description logic, Association for Computational Linguistics, 2008. URL: https://www.researchgate.net/publication/228370565_Referring_Expressions_as_Formulas_of_Description_Logic

УДК 004.4(2)-004.42

Дорофєєв Є. О., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,
Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

АЛГОРИМ ЗЛИТТЯ В МОВІ ПРОГРАМУВАННЯ PYTHON

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Сортування злиттям у Python є одним з ефективних алгоритмів сортування, який базується на принципі «розділяй і володарюй». Цей алгоритм розбиває вхідний масив на менші частини, сортує їх окремо, а потім об'єднує в єдиний відсортований масив. Основні етапи алгоритму включають розділення, сортування та злиття, що гарантує ефективність і стабільність сортування.

Актуальність. Алгоритм сортування злиттям знайшов широке застосування в сучасному програмуванні, оскільки він вирішує проблему ефективного сортування великих обсягів даних. У сучасному світі, де обробка великих масивів даних стала нормою, ефективні алгоритми сортування є критичними для оптимальної роботи програм та додатків. Алгоритм Merge Sort вирізняється своєю асимптотичною оптимальністю з часовою складністю O(n log n), що робить його хорошим вибором для великих обсяг даних, порівняно з іншими алгоритмами сортування. Подальша оптимізація алгоритму, зокрема використання сортування вставкою для невеликих частин масиву та уникання зайвих операцій копіювання під час злиття, може ще більше підвищити його продуктивність у реальних випадках використання. Розгляд алгоритму та його оптимізацій здебільшого важливий для розробників, що працюють із великими обсягами даних і прагнуть досягти оптимальної продуктивності своїх програм.

Основні кроки алгоритму Merge Sort:

Розділення (Divide):

- вхідний масив розділяється на дві половини, знаходячи середину масиву (middle);
- рекурсивно застосовується алгоритм Merge Sort до кожної половини. Цей крок повторюється, поки розмір кожної половини не стає 1 або менше (базовий випадок).

***Сортування (Conquer)*:**

- кожна половина масиву сортується рекурсивно за допомогою алгоритму Merge Sort.

Злиття (Merge):

- два відсортованих підмасиви (ліва та права половини) об'єднуються в один відсортований масив;

➢ створюється новий порожній масив (result), який буде містити об’єднані елементи;

➢ ітеративно порівнюються елементи лівого та правого підмасивів;

➢ елемент, який має менше значення, додається до результату, а потім відповідний підмасив оновлюється, видаляючи доданий елемент;

➢ якщо один із підмасивів стає порожнім, залишок іншого підмасиву додається до результату;

➢ процес триває, поки обидва підмасиви стають порожніми;

➢ отриманий відсортований масив повертається.

Повторне застосування цих трьох кроків приведе до повного відсортування вхідного масиву.

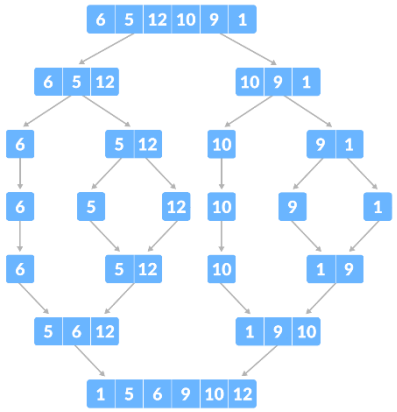


Рисунок 1. Приклад сортування злиттям

Швидкість алгоритму є асимптотично оптимальною, але її можна пришвидшити у константну кількість разів. Оптимізація впорядкування невеликих частин масиву – невеликі частини масиву (наприклад, кількістю менше 50) впорядковувати сортуванням вставкою. Оптимізація кількості копіювань елементів – під злиття двох упорядкованих масивів в один кожен елемент копіюється двічі (спочатку у тимчасовий масив, а потім знову у початковий). Кількість копіювань можна зменшити удвічі, якщо по черзі використовувати для об’єднання початковий і тимчасовий масиви.

Переваги алгоритму сортування злиттям:

➢ стабільність (алгоритм сортування злиттям є стабільним, що означає, що порядок рівнозначних елементів не змінюється. Це важливо в тих випадках, коли необхідно зберігати порядок елементів, які мають однакові ключі);

➢ асимптотична ефективність (у найгіршому, середньому та кращому випадках часова складність сортування злиттям становить $O(n \log n)$, де n – кількість елементів у масиві. Це робить його ефективним для великих об’ємів даних);

➢ послідовна реалізація (сортування злиттям може легко адаптуватися для паралельної обробки, що робить його практичним для використання в сучасних багатоядерних системах);

➢ невибагливість до пам'яті (хоча алгоритм вимагає додаткового простору для збереження тимчасових підмасивів під час злиття, його можна реалізувати так, щоб цей додатковий простір був константним (наприклад, за допомогою ітеративної версії замість рекурсивної).

Недоліки алгоритму сортування злиттям:

➢ неоптимальний для невеликих масивів (для невеликих масивів або вже частково відсортованих даних інші алгоритми, як-то сортування вставкою чи швидке сортування, можуть працювати швидше, оскільки витрачається менше часу на розділення та злиття);

➢ не змінює порядок рівнозначних елементів (якщо важливо змінювати порядок рівнозначних елементів для певних вимог, то сортування злиттям може не бути найкращим вибором);

➢ споживання часу на рекурсію (рекурсивному варіанті алгоритму може бути деякий витрати часу на виклик функцій та збереження додаткового контексту, особливо за великих об'ємах даних.)

Висновки. Сортування злиттям у Python є потужним і ефективним алгоритмом, заснованим на принципі «розділяй і володарюй». У статті ми розглянули основні етапи алгоритму: розділення, сортування та злиття, і висвітлили його ключові переваги.

Актуальність сортування злиттям визначається його ефективністю у роботі з великими обсягами даних, що є актуальним у сучасному програмуванні. Захист порядку рівнозначних елементів та асимптотична оптимальність часової складності роблять його привабливим вибором для розробників, які працюють із великими об'ємами даних та прагнуть досягти оптимальної продуктивності своїх програм.

Оптимізація алгоритму, а саме використання сортування вставкою для невеликих частин масиву та уникання зайвих операцій копіювання під час злиття, посилюють його продуктивність.

У галузі сортування й оптимізації алгоритмів сортування злиттям – є потужним інструментом, який відповідає високим вимогам ефективності та стабільності під час обробки великих обсягів даних.

Список використаних джерел

1. Introduction to Algorithms, 3-тє видання / Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. MIT Press. 2009.

2. Merge Sort. URL: https://www.geeksforgeeks.org/merge-sort/ (дата звернення: 13.11.2023).

УДК 004.8

Короленко С. О., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Волонтир Л. О., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

СУЧАСНІ РЕАЛІЇ ТА ЦІЛІ РОЗШИРЕННЯ ШТУЧНОГО ІНТЕЛЕКТУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Штучний інтелект – це не лише концепція, яка пронизує наше повсякденне життя, а й ключовий елемент сучасної технологічної революції. Технології швидко розвиваються з часом, і штучний інтелект лежить в основі цього технологічного вибуху. Від автоматизації виробництва до персональних помічників на наших смартфонах нас оточують інтелектуальні системи.

Актуальність цієї теми полягає в тому, як ці технології впливають на наше повсякденне життя і як люди можуть використовувати їх для досягнення нових цілей. Сьогодні людство має неймовірні успіхи у сфері штучного інтелекту. Від розпізнавання облич до безпілотних автомобілів відкриття в цій галузі прискорюються щодня. Дослідження показують, що штучний інтелект може революціонізувати охорону здоров'я, допомагаючи діагностувати та лікувати хвороби. Мета роботи – висвітлити сучасні реалії світу штучного інтелекту та визначити цілі його подальшого розширення; розуміти, як технології штучного інтелекту можуть справді принести користь суспільству, і визначити проблеми для вдосконалення.

Експансія штучного інтелекту відбувається в різних напрямах, особливо в медичній сфері. Згідно з «Journal of Medical Research», системи штучного інтелекту вже показали більш точну ідентифікацію захворювань на рентгенівських зображеннях, відкриваючи нові можливості для точної та швидкої діагностики. Не менш вражаючим є використання штучного інтелекту в економічній сфері. Згідно з «Forbes», компанії, які використовують аналітику на основі штучного інтелекту, можуть передбачати мінливі умови ринку та адаптуватися до них, що дає їм конкурентну перевагу. Високотехнологічні алгоритми визначають тенденції та закономірності, щоб допомогти підприємствам приймати більш обґрунтовані рішення.

Використання штучного інтелекту в безпілотних автомобілях. Згідно з дослідженнями, системи штучного інтелекту в безпілотних автомобілях не тільки підвищать рівень безпеки, але й зможуть реагувати на транспортний потік і стан доріг та приймати оптимальні рішення, забезпечувати ефективність роботи.

Роль штучного інтелекту в розробці нових ліків. Дані показують, що використання штучного інтелекту у фармацевтичній промисловості може значно

прискорити процес відкриття нових ліків і точніше визначати їх можливі побічні ефекти.

Ефективні системи управління енергією з використанням штучного інтелекту. Відповідно до дослідження, опублікованого в журналі «Sustainable Energy, Grids and Networks», використання систем штучного інтелекту в енергетичному секторі може покращити оптимізацію виробництва та розподілу енергії. Це сприятиме створенню більшого обсягу енергії, одержувати стійкі та ефективні енергетичні системи.

Вплив штучного інтелекту освіту. Сучасні системи штучного інтелекту дають змогу персоналізувати процес навчання та адаптувати матеріал під потреби окремих студентів. Наукові дослідження в цій галузі свідчать про підвищення якості освіти та інтересу до навчання. Якщо поєднати цей аспект з аналізом досліджень, то можна побачити, що штучний інтелект просувається не тільки у сферах медицини та бізнесу, але й значно сприяє формуванню майбутньої еліти суспільства. Такі інноваційні застосування штучного інтелекту демонструють його різноманітний вплив на різні сфери та підкреслюють важливість продовження досліджень і впровадження цієї технології в різні сфери людської діяльності.

Динамічний розвиток штучного інтелекту відкриває багато можливостей для покращення різних аспектів життя. Важливо зазначити, що зі збільшенням можливостей AI виникають етичні проблеми, пов’язані з конфіденційністю та безпекою. Тому наша місія полягає в тому, щоб забезпечити баланс між використанням цієї потужної технології та захистом основних прав і цінностей суспільства. Використання штучного інтелекту в усіх сферах життя не тільки відкриває нові горизонти можливостей, але й накладає на нас відповідальність за створення сталого та гуманного майбутнього.

Отже, штучний інтелект є не лише інструментом для досягнення конкретних цілей, а й важливою частиною формування суспільства загалом. Подальший розвиток цієї технології захоплює, але вимагає обережності та уваги на кожному кроці, тому використання потенціалу штучного інтелекту вимагає не лише технологічних зусиль, але й моральної відповідальності перед суспільством і майбутніми поколіннями. Зокрема, щодо досліджень здоров’я населення інтеграція штучного інтелекту дає змогу ідентифікувати шкідливі фактори та прогнозувати ризики захворювань, відкриваючи можливості для ефективних профілактичних заходів.

Насамкінець підкреслимо, що штучний інтелект – це не просто технологічна інновація, а ключ до розвитку суспільства загалом. Ми зобов’язані використовувати цей інструмент для досягнення значного прогресу та забезпечення процвітання для всіх.

Список використаних джерел

1. Історія штучного інтелекту. URL: https://uk.wikipedia.org/wiki/Історія_штучного_інтелекту

2. Перспективи використання штучного інтелекту у сфері медицини. URL: https://ena.lpnu.ua:8443/server/api/core/bitstreams/3e2bd0d2-cd07-4896-936d-771be7934e94/content

3. Sustainable Energy, Grids and Networks. URL: https://www.sciencedirect.com/journal/sustainable-energy-grids-and-networks

4. Вчитель + штучний інтелект – це майбутнє освіти. URL: https://intboard.ua/pres-sluzhba/blog/vchytel-shtuchnyy-intelekt-tse-maybutnye-osvity/

УДК 004.43:004.056.5

Мигун Р. С., здобувач 1 курсу спеціальності 122 Комп'ютерні науки,
Горяшин А. С., асистент кафедри інформаційних технологій

РЕАЛІЗАЦІЯ БЛОКЧЕЙН-ТЕХНОЛОГІЙ З ВИКОРИСТАННЯМ PYTHON ДЛЯ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ ТА ЦІЛІСНОСТІ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Блокчейн, або децентралізований цифровий реєстр – це особливий вид бази даних, який підтримується численними комп'ютерами, розміщеними в усьому світі. Дані блокчейну організовані в блоки, які розташовані в хронологічному порядку і захищені криптографією [1].

Ідея блокчейн-технології була описана ще в 1991 р., коли вчені-дослідники С. Хабер та В. Скотт Сторнетта запропонували обчислювально-практичне рішення для тимчасового маркування цифрових документів, щоб їх не можна було змінити або підробити. Система використовувала криптографічно захищений ланцюг блоків для зберігання документів із позначками часу, і у 1992 р. в конструкцію були додані дерева Меркла, що зробило її ефективнішою та дало змогу збирати кілька документів в один блок. Однак ця технологія не використовувалася, а патент втратив чинність у 2004 р., за чотири роки до появи Bitcoin [2].

Основні переваги блокчейну, які можна виділити:

Децентралізація – Блокчейн розподілений між багатьма вузлами, тому він більш стійкий до атак і витоку даних.

Прозорість – транзакції у блокчейні видні всім учасникам, що спрощує відстеження і перевірку транзакцій, а також забезпечує їх точність.

Незмінність – блокчейн-транзакції незмінні, тому їх можна перевірити та відстежити. Це відрізняється від традиційних систем, де транзакції можна змінити або видалити.

Ефективність – блокчейн-транзакції швидші, бо їх не потрібно перевіряти посередникам.

Система «без довіри» – блочейн-транзакції публічні та надійні, бо їх перевіряють і підтверджують усі учасники мережі, а не один посередник.

Алгоритм консенсусу – це принципи роботи блокчейну, які забезпечують безпеку мережі та узгодженість даних між вузлами. Це необхідно для підтримки цілісності та безпеки блокчейну. Завдяки цим принципам система не потребує адміністраторів та центральних сховищ. Алгоритми консенсусу підтверджують правильність інформації у кожному блоці системи.

Розробники пропонували безліч алгоритмів консенсусу, але найпопулярнішими стали ті, основу яких лежить:

Доказ роботи – Proof-of-Work або PoW;

Підтвердження частки – Proof-of-Stake або PoS [3].

Реальні приклади блокчейну

Відповідаючи на запитання «Які існують приклади блокчейну?», розглянемо реальні випадки впровадження технології блокчейн у різних галузях.

Фінансовий сектор: Ripple – це платіжна система, яка дає змогу здійснювати міжнародні перекази з низькими комісіями та миттєвими операціями.

Охорона здоров'я: MediChain – це платформа для зберігання медичних записів. Забезпечує безпеку та доступність даних для лікарів та пацієнтів.

Логістика: IBM Food Trust – це система, що пропонує прозорий та ефективний контроль відстеження продуктів харчування вздовж ланцюга постачання.

Що таке розподілена база даних? Найпростіший приклад пристрою блокчейн-сховища – величезна таблиця даних, яка мільйони разів дублюється на різних пристроях мережі. А ще в такій системі кожну хвилину оновлюються дані. Інформація, яка зберігається в ланцюжку блоків, – загальнодоступна. Вона постійно звіряється на основі тисяч дублів. У такого формату є безліч істотних переваг. Дані не містяться в якомусь одному місці, вони публічні й доступні. Немає централізованого сховища, яке можна було б зламати або пошкодити [4].

Давайте спробуємо розгорнути свій блокчейн на мові програмування python. Цей код зроблений для ознайомлення, на більш відповідальних проєктах розробка блокчейну робиться не за 1 місяць [5].

```
# -*- coding: utf-8 -*-

from hashlib import sha256
import json
```

```
from time import time

class Block:

    def __init__(self, timestamp=None, data=None):
        self.timestamp = timestamp or time()
        # У this.data повинна зберігатися інформація, на кшталт відомостей про транзакції.
        self.data = [] if data is None else data
        self.prevHash = None # Хеш минулого блока
        self.nonce = 0
        self.hash = self.getHash()

    def getHash(self):

        hash = sha256()
        hash.update(str(self.prevHash).encode('utf-8'))
        hash.update(str(self.timestamp).encode('utf-8'))
        hash.update(str(self.data).encode('utf-8'))
        hash.update(str(self.nonce).encode('utf-8'))
        return hash.hexdigest()

    def mine(self, difficulty):
        # Тут запускається цикл, що працює до тих пір, поки хеш не буде починатися з рядка
        # 0...000 довжини <difficulty>.
        while self.hash[:difficulty] != '0' * difficulty:
            print("process")
            # Інкрементуємо nonce, що дає змогу отримати абсолютно новий хеш.
            self.nonce += 1
            # Перераховуємо хеш блоку з урахуванням нового значення nonce.
            self.hash = self.getHash()

class Blockchain:

    def __init__(self):
        # У цій властивості будуть утримуватися всі блоки.
        self.chain = [Block(str(int(time())))]
        self.difficulty = 1
        self.blockTime = 30000

    def getLastBlock(self):
        return self.chain[len(self.chain) - 1]

    def addBlock(self, block):
            # Оскільки ми додаємо новий блок, prevHash буде хешем попереднього останнього блоку.
        block.prevHash = self.getLastBlock().hash
        # Оскільки тепер в prevHash є значення, ми повинні перерахувати хеш блоку.
```

```
            block.hash = block.getHash()
            block.mine(self.difficulty)
            self.chain.append(block)

                self.difficulty += (-1, 1)[int(time()) - int(self.getLastBlock().timestamp) < self.blockTime]

        def isValid(self):
            # Перед перебором ланцюжка блоків необхідно встановити i в 1, оскільки до первинного блоку жодних блоків немає. В результаті ми починаємо з другого блоку.
            for i in range(1, len(self.chain)):
                currentBlock = self.chain[i]
                prevBlock = self.chain[i - 1]

                # Перевірка
                    if (currentBlock.hash != currentBlock.getHash() or prevBlock.hash != currentBlock.prevHash):
                        return False

            return True

        def __repr__(self):
                return json.dumps([{'data': item.data, 'timestamp': item.timestamp, 'nonce': item.nonce, 'hash': item.hash, 'prevHash': item.prevHash} for item in self.chain], indent=4)
```

Тепер після того, як у нас є каркас блокчейну, ми можемо в ньому зберігати різну інформацію, наприклад:

```
JeChain = Blockchain()

# Додамо новий блок
JeChain.addBlock(Block(str(int(time())), ({"from": "John", "to": "Bob", "amount": 1001})))

# Вивід оновленого блокчейна
print(JeChain)
```

Після запуску коду ми зможемо побачити приблизно ось таку інформацію:

```
[
    {
        "data": [],
        "timestamp": "1700350466",
        "nonce": 0,
        "hash":
"60f3cef058fab68497d70e80df4c22970d5919e2787a1d53f6619929c6a60e13",
        "prevHash": null
    },
```

```
        {
          "data": {
            "from": "John",
            "to": "Bob",
            "amount": 1001
          },
          "timestamp": "1700350466",
          "nonce": 16,
          "hash":
"0c5ddae74fbdb8203add0b95cb784006776550fed7acef8022d445f96d5f50f7",
          "prevHash":
"60f3cef058fab68497d70e80df4c22970d5919e2787a1d53f6619929c6a60e13"
        }
      ]
```


Висновки. Отже, блокчейн – це децентралізована база даних, яка використовує криптографію для забезпечення безпеки та цілісності даних. Це робить його привабливим для широкого спектра застосувань, як-от фінансові послуги, охорона здоров'я та логістика. Блокчейн-транзакції незмінні, що робить їх прозорими та надійними. Це може допомогти зменшити ризик шахрайства та зловживань. Алгоритми консенсусу забезпечують безпеку мережі блокчейну та узгодженість даних між вузлами. Це робить блокчейн більш стійким до атак і витоку даних. Загалом блокчейн є потужною технологією, яка має потенціал змінити спосіб, яким ми зберігаємо та обмінюємося даними.

Список використаних джерел

1. Що таке блокчейн і як він працює? URL: https://academy.binance.com/uk/articles/what-is-blockchain-and-how-does-it-work
2. Історія блокчейну. URL: https://academy.binance.com/uk/articles/history-of-blockchain
3. Що таке блокчейн? Пояснюємо простими словами. URL: https://blog.whitebit.com/ uk/what-is-blockchain-technology/
4. Blockchain. URL: https://astwellsoft.com/uk/blog/blockchain.html

УДК 004.62

Мисак М. П., здобувач 2 курсу спеціальності 122 Комп'ютерні науки ОС «Магістр»,

Бабаков Р. М., д-р техн. наук, доцент, професор кафедри інформаційних технологій

ОБРОБКА ДАНИХ У МОБІЛЬНІЙ ІНФОРМАЦІЙНО-ДОВІДНИКОВІЙ СИСТЕМІ ДЛЯ СТУДЕНТІВ ФАКУЛЬТЕТУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сучасний світ характеризується стрімким розвитком інформаційних технологій. Через це зростає роль мобільних інформаційно-довідникових систем (МІДС), які дають змогу користувачам отримувати доступ до інформації в будь-який час і в будь-якому місці.

Одним із важливих аспектів створення МІДС є обробка даних. Обробка даних включає в себе такі завдання: збір, зберігання, аналіз і представлення даних.

У роботі розглянуто прикладні аспекти обробки даних у МІДС для студентів факультету. Для реалізації МІДС використано технологічний стек Flutter, Dart і Firebase.

Актуальність дослідження обумовлена такими факторами:

- усе більша популярність мобільних пристроїв, зокрема смартфонів;
- потреба студентів у доступі до актуальної інформації про факультет;
- можливості технологічного стеку Flutter, Dart і Firebase для реалізації МІДС.

У літературі існує значна кількість досліджень, присвячених обробці даних у МІДС. У цих дослідженнях розглядаються такі аспекти [1]:

- методи збору даних;
- методи зберігання даних;
- методи аналізу даних;
- методи представлення даних.

У цій роботі описаний процес створення частини обробки даних мобільної інформаційно-довідкової системи для студентів факультету на основі даних, які забезпечують доступ до актуальної інформації про навчання на факультеті.

Зібрані дані зберігаються в нереляційній базі даних Firebase Realtime Database. Ця база даних забезпечує синхронізацію даних у реальному часі.

Дані зберігаються в JSON-форматі. Кожен документ у базі даних є об'єктом JSON, який має ключ і значення. Ключ є унікальним ідентифікатором документа, а значення – це вміст документа.

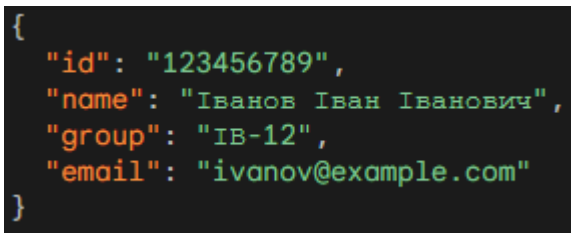


Рисунок 1. Вигляд документа JSON, який зберігає інформацію про студента

Для обробки даних у Firebase Realtime Database використовується API Firebase. Цей API дає змогу виконувати такі операції з даними [2]:

- фільтрація – вибір даних, які відповідають певним критеріям;
- сортування – впорядкування даних за певним критерієм;
- об'єднання – об'єднання даних із різних джерел;
- агрегація – обчислення узагальнених показників з даних.

Зображений на рис. 2 код виконує фільтрацію даних про студентів за групою.

```
// Створюємо базу даних
DatabaseReference db = FirebaseDatabase.instance.ref();

// Отримуємо список документів з бази даних
Query query = db.child("students").orderByChild("group");

// Виконуємо фільтрацію даних
query.whereEqualTo("group", "М-22");

// Читаємо дані з документа
query.onValue.listen((snapshot) {
  // Виводимо дані з документа
  print(snapshot.value);
```

Рисунок 2. Код, що виконує фільтрацію даних про студентів за групою

Також можна використовувати Firebase Realtime Database для створення інтерактивних видів, як-от таблиці, діаграми та карти. Для цього можна використовувати бібліотеки FirebaseUI і Google Maps Flutter SDK.

МІДС складається з таких компонентів [3]:

- клієнтський компонент забезпечує інтерфейс користувача для МІДС. Він реалізований на технологічному стеку Flutter, Dart;
- віртуальна машина виконує код клієнтського компонента;
- базовий компонент забезпечує базові можливості МІДС: зберігання даних, доступ до мережі та ін. Він реалізований на мові програмування Dart;
- Firebase забезпечує зберігання даних і аналітику МІДС.

Для обробки даних у МІДС використано такі методи [4]:

- метод збору даних, заснований на використанні API Firebase;

➢ метод зберігання даних, заснований на використанні Firebase Realtime Database;

➢ метод аналізу даних, заснований на використанні Firebase Analytics;

➢ метод представлення даних, заснований на використанні Firebase Realtime Database.

Отримані результати були обґрунтовані шляхом проведення дослідження і розробки системи. Дослідження показали, що МІДС відповідає поставленим вимогам і забезпечує ефективну обробку даних.

Внаслідок дослідження було розроблено МІДС для студентів факультету, яка забезпечує ефективну обробку даних. МІДС може бути використана для забезпечення студентів факультету актуальною інформацією.

Список використаних джерел

1. Russo, Daniel T. Data Science for Mobile App Development, 2018.
2. Мельник О. В., Морозов В. О. Обробка даних в мобільних системах, 2017.
3. Іванов М. В. Firebase Realtime Database: можливості та застосування, 2021.
4. Developer documentation for Firebase. URL: https://firebase.google.com/docs

УДК 519.8

Михайляк М. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

МОДЕЛЮВАННЯ ВПЛИВУ ФАКТОРІВ ПРИРОДОКОРИСТУВАННЯ НА ЕКОЛОГІЮ

Донецький національний університет імені Василя Стуса, м. Вінниця

У сучасному світі проблема впливу природокористування на екологію стає дедалі більш актуальною та нагальною. Збільшення населення, розвиток промисловості, швидке зростання економіки призводять до інтенсивного використання природних ресурсів, що може негативно сказатися на стані екосистем та загальній екологічній стійкості. Відтак важливо досліджувати та моделювати вплив різних факторів природокористування на екосистеми, а також визначати шляхи стійкого й ефективного природокористування для забезпечення екологічної рівноваги та довгострокової стійкості природних систем.

Одним з головних складників вивчення впливу природокористування на екологію є аналіз ключових факторів, що входять до цього процесу. Землекористування, лісозаготівлі, водокористування та інші аспекти визначають мас-

штаб і характер впливу на природні екосистеми. Моделювання різних сценаріїв та їх взаємодії з біорізноманіттям та кліматичними процесами дає змогу прогнозувати наслідки та розробляти стратегії для збереження біологічної різноманітності та природних ресурсів [1].

Додатковою важливою аспектною областю дослідження є визначення взаємозв'язків між різними видами природокористування та їх взаємодія з іншими екологічними чинниками. Наприклад, аналіз впливу змін у землекористуванні на якість ґрунтів і водних ресурсів може розкрити потенційні наслідки для рослинності та тваринного світу.

Важливо враховувати соціально-економічні аспекти природокористування, оскільки вони часто визначають обсяги та структуру використання ресурсів. Економічні інтереси можуть конфліктувати з екологічними цілями, тому важливо розробляти моделі, які враховують інтереси всіх сторін та сприяють досягненню балансу між екологічною стійкістю й економічним розвитком.

Моделювання також дає змогу вивчення потенційних екологічних криз, розвитку стресових сценаріїв і розроблення ефективних заходів для запобігання та пом'якшення їх наслідків. Це важливо для розробки сталого природокористування й забезпечення збереження природних ресурсів для майбутніх поколінь.

Математичні моделі є потужним інструментом для розуміння та аналізу складних систем, зокрема ті, які пов'язані зі змінами у природокористуванні та екосистемах [2]. Ось деякі з переваг математичних моделей:

- Системний аналіз. Математичні моделі дають змогу включати різні фактори та їх взаємодії у вигляді математичних рівнянь. Це допомагає проводити глибший аналіз впливу окремих елементів системи на загальний результат.
- Прогнозування та сценарійне планування. Математичні моделі можуть бути використані для розробки різних сценаріїв та прогнозування реакції системи на зміни у введених параметрах. Це корисно для розробки стратегій управління та запобігання можливим негативним наслідкам.
- Ефективність рішень. Математичні моделі дають змогу тестувати різні стратегії та рішення віртуально перед їх впровадженням у реальному середовищі. Це зменшує ризик непередбачуваних наслідків та допомагає вибрати оптимальний варіант.
- Візуалізація та поширення знань. Математичні моделі можуть бути візуалізовані, що спрощує розуміння складних взаємозв'язків між різними факторами системи. Це полегшує комунікацію та сприяє поширенню знань.
- Експерименти в умовах обмежених ресурсів. У реальному світі проведення експериментів може бути дорогим та затратним за часом. Математичні моделі дають змогу виконувати чисельні експерименти в обмежених умовах, враховуючи різні фактори та середовище.

➢ Управління ризиками. Моделі дають змогу визначити ризики та їхні можливі наслідки, що допомагає розробити стратегії для зменшення чи управління ризиками.

➢ Стимулювання креативності та інновацій. Математичні моделі можуть слугувати інструментом для розробки нових ідей та інновацій, даючи змогу тестувати та оцінювати їх можливі впливи.

І хоча математичні моделі мають свої обмеження та вимагають точних та релевантних даних для ефективного застосування, вони є важливим інструментом для розв'язання складних проблем у галузі природокористування та екології [3].

Можна розглянути простий приклад математичної моделі для вивчення впливу факторів природокористування на екологію. У цьому прикладі використано три основні фактори: землекористування, лісозаготівлі та водокористування.

Математична модель:

Z – землекористування;

L – лісозаготівлі;

W – водокористування;

E – екологічні показники (наприклад, рівень забруднення).

Можна використовувати прості лінійні функції для опису взаємозв'язків між цими факторами та екологічними показниками:

Вплив землекористування (Z):

$$E = aZ + b. \quad (1)$$

Вплив лісозаготівлі (L):

$$E = cL + d. \quad (2)$$

Вплив водокористування (W):

$$E = eW + f. \quad (3)$$

Загальна екологічна динаміка:

$$E = aZ + b + cL + d + eW + f. \quad (4)$$

Пояснення:

➢ коефіцієнти a, b, c, d, e, f – це параметри моделі, які треба визначити з урахуванням даних та конкретних властивостей системи;

➢ модель передбачає, що вплив кожного фактора лінійно залежить від його значення.

Для повноцінного розроблення та калібрування такої моделі потрібно використовувати конкретні дані та проводити додатковий аналіз.

Це лише простий приклад, і реальна модель може бути складнішою з використанням нелінійних функцій або інших методів моделювання, але це дає загальну ідею про те, як можна підходити до створення математичної моделі впливу факторів природокористування на екологію.

Продовжуючи розгляд математичних підходів у моделях для аналізу впливу природокористування на екологію, можна розглянути декілька додаткових аспектів:

- диференціальні рівняння (рівняння широко використовуються для висвітлення динаміки змін у часі. Вони дають змогу моделювати, як динамічні параметри змінюються у часі);
- системи рівнянь (моделі, що їх містять, можуть враховувати взаємодії між різними компонентами екосистеми);
- чисельні методи (використання чисельних методів дає змогу розв'язувати складні математичні задачі, які не завжди мають аналітичні розв'язки).

У висновку можна зазначити, що вивчення впливу природокористування на екологію є надзвичайно складним та багатогранним завданням, що вимагає інтегрованого підходу. Аналіз ключових факторів, як-от землекористування, лісозаготівлі та водокористування, визначає масштаб і характер впливу на природні екосистеми. Загалом вивчення впливу природокористування на екологію та використання теоретичних моделей сприяє розумінню складності цього процесу і розвитку стратегій для сталого та збалансованого використання природних ресурсів.

Список використаних джерел

1. Природокористування: вебсайт. URL: https://st.kharkov.ua/wp-content/uploads/2022/01/pryrodokorystuvnnia.pdf (дата звернення: 27.11.2023).

2. Ніколюк П. К. Моделювання систем: навч. посіб. для здобувачів вищої освіти спеціальності 122 «Комп'ютерні науки». Вінниця: ДонНУ імені Василя Стуса, 2023 (дата звернення: 27.11.2023).

3. Моделювання. Основні поняття. URL: https://web.posibnyky.vntu.edu.ua/fmbt/avto6_bilichenko_modelyuvtehproces_avtotransportu/p2.html (дата звернення: 27.11.2023).

УДК 004.5

Підруцький Д. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

АНАЛІЗ ПЕРСПЕКТИВ ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Штучний інтелект (ШІ) стає все більш актуальним у сучасному світі, проникаючи в різні сфери нашого життя. Його потенціал для розвитку і вплив на різні галузі є предметом інтенсивних досліджень та обговорень. У цьому контексті важливо провести аналіз перспектив застосування штучного інтелекту з метою визначення його можливостей і викликів.

ШІ має потенціал трансформувати бізнес-процеси, вдосконалювати медичні технології, оптимізувати виробництво і поліпшувати кількісні та якісні показники різних секторів. Його застосування також пов'язане з етичними питаннями і потенційними загрозами для працевлаштування та безпеки.

Останні дослідження в галузі штучного інтелекту вказують на значний прогрес у сферах машинного навчання, обробки природної мови та комп'ютерного зору. Додатковий акцент робиться на вдосконаленні алгоритмів глибокого навчання та розвитку автономних систем.

Метою цієї роботи є систематичний аналіз перспектив застосування штучного інтелекту з урахуванням сучасного стану досліджень у цій області.

Для досягнення визначеної мети, потрібно виконати такі завдання:

1. Насамперед визначитись із тим, що являє собою штучний інтелект.
2. Проаналізувати переваги та недоліки штучного інтелекту.
3. Навести приклади, де зараз може або ще не застосовується ШІ.
4. Зробити остаточний підсумок аналізу цієї тематики.

ШІ – унікальний продукт технічного прогресу, що дає змогу машинам вчитися, використовуючи людський і власний досвід, пристосовуватися до нових умов у метах свого застосування, виконувати різнопланові завдання, які тривалий час були під силу лише людині, прогнозувати події й оптимізувати ресурси різного характеру [1]. Він являє собою обширну галузь досліджень, яка передбачає різноманітні теорії, методи, технології та практики. У цьому контексті важливо розглянути ключові поняття, які є основою для розвитку і застосування ШІ.

Як і будь-які технології, вони мають свої переваги та недоліки. Ознайомимося з перевагами [2]:

- ШІ може обробляти величезні обсяги даних та виконувати завдання швидше, ніж людина;

➢ ШІ може досягати високого рівня точності у виконанні завдань, а також підвищувати рівень безпеки у різних галузях;

➢ ШІ дає змогу автоматизувати рутинні завдання та оптимізувати процеси в різних галузях;

➢ можливість швидко адаптуватися до змінюваних умов та нових даних;

➢ здатність ШІ аналізувати великі обсяги даних для прогнозування та прийняття рішень;

ШІ має такі недоліки:

➢ може вимагати значних зусиль та часу залежно від завдання, розробки та налаштування алгоритмів штучного інтелекту;

➢ можливість систематичних помилок та непередбачуваних ситуацій;

➢ скорочення робочих місць через автоматизацію певних видів робіт;

➢ непередбачуваність та ризики у випадку неправильного аналізу або даних.

Тепер розберемо сфери, де ШІ застосовують [3]:

➢ Креативні ***індустрії (маркетинг, медіа тощо).*** Це перша сфера, яка спадає на думку після появи ChatGPT 4-ї версії. Користувачі можуть ставити запитання і швидко отримувати відповіді, генерувати тексти для статей, соцмереж, складати вірші та оповідання тощо.

➢ ***Медицина.*** Вже зараз ШІ аналізує рентгенівські знімки, результати комп'ютерної томографії тощо, щоб виявити рак чи інші хвороби. Це суттєво економить час медичних фахівців, що дає змогу приділити більше уваги пацієнтам. Як стверджує консалтингова компанія McKinsey, 15 % годин роботи медичних працівників можна автоматизувати за допомогою ШІ. Також ШІ уже використовується для аналізу даних і пошуку закономірностей у них, встановлення точного діагнозу та призначення лікування, розроблення нових ліків і роботизованої хірургії. Наприклад, IBM Watson може аналізувати структуровані та неструктуровані дані, які можуть бути критичними для лікування, а також вивчати медичну карту пацієнта і робити призначення. Отже, цей ШІ-інструмент уже фактично виконує функції лікаря.

➢ ***Банківська та фінансова сфера.*** У цій сфері ШІ, ймовірно, буде набирати обертів у застосуванні для аналізу поточних трендів і передбачення нових. Сьогодні далеко не усі фінансові установи користуються ШІ-інструментами, але якщо вони прагнуть залишитися конкурентоспроможними, то незабаром не матимуть іншого вибору. І звичайно, у цій сфері штучний інтелект може спростити виконання безлічі рутинних завдань на зразок підготовки та обробки документів. Це допоможе скоротити витрати, а працівникам – зосередитися на завданнях, які мають більшу цінність.

➢ ***Транспорт і логістика.*** Ця сфера розвивається завдяки штучному інтелекту: зокрема, машинне навчання вже допомогло зробити управління ланцюж-

ком постачання безперебійним процесом. Роботи на складах сортують і пакують товари, а ШІ-алгоритми будують оптимальні маршрути доставки замовлень. На черзі – революція автономних транспортних засобів. Tesla, Volvo, Volkswagen, Uber та інші компанії активно вкладаються в інновації, які згодом замінять водіїв-людей. Також сьогодні у різних країнах досліджують використання штучного інтелекту для оптимізації руху громадського транспорту та управління світлофорами.

Отже, у підсумку проведеного аналізу перспектив застосування штучного інтелекту можна зазначити, що ця технологія не лише визначає новий етап технічного прогресу, але й створює потужні можливості для інновацій та вдосконалення різних галузей. Застосування штучного інтелекту у медицині, виробництві, фінансах та інших секторах приводить до покращення ефективності, оптимізації процесів і створення нових можливостей, які покращать наше життя, водночас ШІ потрібно так само прямо пропорційно розвивати.

Список використаних джерел

1. Штучний інтелект (AI): що це таке і чому це важливо?: вебсайт. URL: https://cutt.ly/9wIzdlvQ (дата звернення: 23.11.2023).

2. Плюси і мінуси штучного інтелекту: вебсайт. URL: https://moyaosvita.com.ua/matematuka/plyusi-i-minusi-shtuchnogo-intelektu/ (дата звернення: 23.11.2023).

3. Вплив штучного інтелекту на сфери діяльності людини: вебсайт. URL: https://hub.kyivstar.ua/articles/yaki-sfery-zminyuvatyme-shtuchnyj-intelekt-u-najblyzhchomu-majbutnomu (дата звернення: 23.11.2023).

УДК 004.9

Рудь О. С., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

РОЛЬ СТАТИСТИЧНОГО НАВЧАННЯ У ВДОСКОНАЛЕННІ МЕДИЧНИХ ЗАСОБІВ ТА ТЕХНОЛОГІЙ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. Сьогодні медична галузь активно використовує сучасні технології для покращення діагностики, лікування та профілактики різних захворювань. Однією з ключових галузей, яка відіграє важливу у роль у цьому процесі, є статистичне навчання. Сучасні медичні засоби та технології великою мірою

базуються на аналізі великих обсягів даних, який стає можливим завдяки розвитку статистичного навчання.

Актуальність. Впровадження алгоритмів машинного навчання в медицину не лише полегшує діагностику та лікування, але й відкриває перспективи для інноваційних підходів до прогнозування захворювань і персоналізованої медичної практики, що робить цю тему надзвичайно актуальною.

1. Застосування статистичного навчання в діагностиці та прогнозуванні

Однією з найактуальніших областей використання статистичного навчання у медицині є діагностика захворювань та їх прогнозування. За допомогою алгоритмів машинного навчання системи можуть аналізувати величезні обсяги клінічних даних, результати обстежень та зображень, що надходять від різних медичних пристроїв. Використання цих даних у поєднанні з розвиненими моделями дає змогу виявляти патології на ранніх стадіях, коли їх ще не завжди можна виявити за допомогою стандартних методів.

Наприклад, алгоритми можуть виявляти певні патерни у зображеннях медичних знімків, які лікарі можуть пропустити або важко помітити. Також вони можуть аналізувати результати лабораторних тестів та інші клінічні параметри для швидкої і точної діагностики. Це допомагає лікарям приймати інформовані рішення щодо лікування, а також сприяє покращенню прогнозів щодо перебігу захворювання.

За допомогою статистичного навчання розробляються моделі прогнозування, що дають змогу визначити ймовірність розвитку конкретних захворювань у пацієнтів на основі їхньої історії захворювань, генетичних факторів та інших даних. Це відкриває можливості для раннього втручання та індивідуальної стратегії лікування, спрямованої на попередження розвитку конкретних ускладнень [1].

2. Оптимізація лікування та індивідуалізація підходів

Алгоритми машинного навчання можуть аналізувати різноманітні клінічні дані та інформацію про пацієнта для прийняття інформованих рішень щодо лікування.

Наприклад, на основі історії лікування та реакції на попередні методи можна прогнозувати ефективність різних терапевтичних стратегій. Системи, що використовують статистичне навчання, можуть адаптувати та оптимізувати лікування на основі відповіді пацієнта протягом часу, що сприяє максимальній ефективності та мінімізації негативних наслідків. Статистичне навчання також дає змогу враховувати індивідуальні особливості пацієнта, як-от генетичні фактори та особливості метаболізму. Це важливо для розробки персоналізованих підходів до лікування, що має велике значення у випадках, коли стандартні методи можуть бути менш ефективними або призводити до побічних ефектів [2].

3. Профілактика та попередження захворювань

Аналізуючи великі обсяги даних про здоров'я пацієнтів, зокрема й генетичні, клінічні та соціальні параметри, алгоритми машинного навчання можуть ідентифікувати особливі групи ризику для конкретних захворювань.

Наприклад, моделі можуть прогнозувати ймовірність розвитку серцево-судинних захворювань на основі генетичних факторів та звичок життя. Це дає змогу розробляти персоналізовані рекомендації з попередження ризикових факторів, що сприяє зменшенню виникнення захворювань та покращенню загального стану здоров'я населення. Статистичне навчання також використовується для розробки систем прогнозування епідемій та поширення інфекційних захворювань. Аналізуючи глобальні дані про захворювання та спостерігаючи їх розвиток, можна реагувати на потенційні загрози та вживати ефективних заходів щодо контролю поширення захворювань.

4. Ризики та перспективи використання

Незважаючи на величезний потенціал статистичного навчання у медицині, існують ризики, як-от забезпечення конфіденційності медичних даних та розробка етичних стандартів використання алгоритмів у практиці. Однак із постійними дослідженнями та строгим врахуванням цих аспектів статистичне навчання в медицині обіцяє залишатися каталізатором інновацій та значним інструментом для покращення якості медичної допомоги.

Висновки. Статистичне навчання визнане ключовим інструментом для вдосконалення медичних засобів та технологій. Воно не лише допомагає у точній діагностиці та ефективному лікуванні, але й відкриває нові можливості у профілактиці та індивідуалізації медичних підходів. За умови вирішення етичних та конфіденційних питань статистичне навчання матиме все більший вплив на розвиток медичної науки та покращення якості надання медичних послуг.

Список використаних джерел

1. Слабкий Г. О., Миронюк І. С. Біостатистика: методичні рекомендації. Ужгород: 2020. 122 с.

2. Василенко О. А., Сенча І. А. Математично-статистичні методи аналізу в прикладних дослідженнях: навч. посіб. Одеса: 2012. 166 с.

УДК 004.415

Сіклічук А. С., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Січко Т. В., канд. техн. наук, доцент, доцент кафедри інформаційних і прикладних технологій

ТЕСТУВАННЯ API ЯК ІНСТРУМЕНТ ВИЯВЛЕННЯ ПОМИЛОК У РОБОТІ КЛІЄНТ-СЕРВЕРНИХ СИСТЕМ

Донецький національний університет імені Василя Стуса, м. Вінниця

Вступ. У сучасному програмуванні й розробці програмного забезпечення важливим етапом є не тільки створення функціональності, але й гарантування її надійності та стабільності через проведення належного тестування. Зокрема, тестування API (Application Programming Interface) є важливим складником у забезпеченні правильної взаємодії між різними компонентами програм та сервісів. Найбільш поширеним інструментом для розв'язання цієї задачі підходить автоматизоване тестування, яке за допомогою різних мов програмування, фреймворків та бібліотек дає змогу зробити цей процес набагато швидшим та більш організованим.

Актуальність. API є невід'ємною частиною клієнт-серверної взаємодії. Він являє собою набір правил та інструкцій, які визначають, як програми або компоненти програмного забезпечення можуть взаємодіяти один з одним. Цей термін використовується для опису інтерфейсу, який дає змогу взаємодіяти з вебсерверами та отримувати або передавати дані через мережу Інтернет. Для цього використовується протокол HTTP, що забезпечує доступ до різних послуг чи ресурсів через мережу. Саме тому тестування коректної поведінки інтерфейсів є надважливим в сучасному циклі тестування вебдодатків.

Важливість тестування API можна продемонструвати на прикладі роботи з відкритою API-документацією. Ця документація знаходиться на сервісі Swagger [1], на якому безпосередньо можна здійснювати саме тестування. Вона має назву PetStore і являє собою набір інформації, яка описує функціональність та використання самого інтерфейсу. Загалом це зразок додатка, що можна використовувати для тестування. Додаток імітує онлайн-зоомагазин, і користувачі можуть додавати та отримувати інформацію про своїх вихованців. Уявимо, що у нас є версія додатка Pet Store, але з графічним інтерфейсом. Відповідно на сайті має відображатися інформація про нашого улюбленця, яку ми заздалегідь внесли в особистому кабінеті. Але ця інформація відсутня або відображається частково. Є два варіанти того, що може слугувати причиною: помилка на клієнті, тобто front-end не відображає правильно дані, або помилка на сервері, тобто back-end має певні проблеми в роботі логіки і відправляє на клієнт неправильні дані. Тут і стає в нагоді API-документація. За її допомогою можна створити тест, що перевірить,

чи коректно відпрацьовує логіка додатка. Реалізовувати його ми будемо за допомогою мови програмування Python та фреймворка для тестування Pytest. Це все можна було б реалізувати і за допомогою самого сервісу Swagger, але написані власноруч тести зручніше зберігати та оптимізувати, якщо з часом логіка додатка буде змінюватися. Запуск у вигляді так званих тест-ранів (набір тестів) і обробка помилок буде здійснюватися швидше, і ми будемо отримувати чіткі та зрозумілі результати. Також за такого підходу легше зберігати тестові дані, які будуть використовуватися під час кожного запуску програми.

Перед тим, як написати код для перевірки правильності отримання даних про домашнього улюбленця, потрібно його створити. Розглянемо наявний для цього запит в API-документації від СВАГЕР (рис. 1).

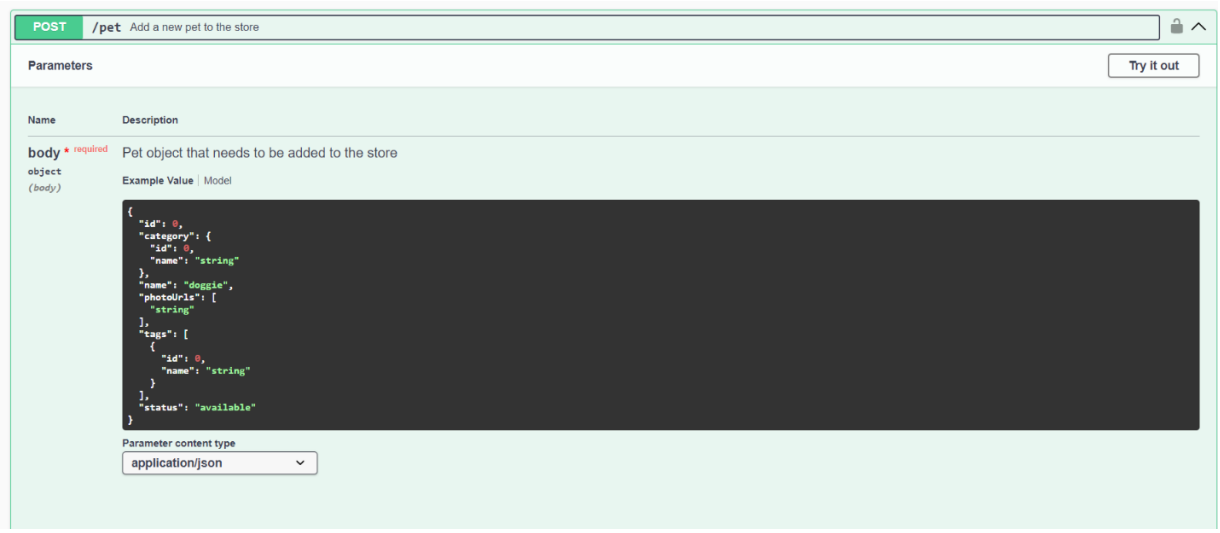


Рисунок 1. Структура запиту на створення інформації про тварину

Структура запиту складається з методу POST, URL та тіла запиту. В тілі у нас передається на сервер вся інформація в форматі JSON.

Для отримання даних про улюбленця використовується запит, що складається з методу GET та URL (рис. 2). Пошук тваринки відбувається за ID, який попередньо призначається під час створення.

Рисунок 2. Структура запиту на отримання даних про тварину

Перед тим, як писати тести, потрібно зробити set-up та підготувати тестові дані. Налаштування середовища можна зробити за допомогою вбудованого в Python pip або за допомогою Poetry [3]. Це інструменти для управління залежностями, як-от встановлені бібліотеки, пакети та ін. Різниця лише в тому, що pip зберігає інформацію про залежності, а poetry зберігає інформацію про сам проєкт загалом, і зберігає це в одному місці, що є дуже зручним.

У Pytest є так звані фікстури – функції, що дають змогу виконувати певні підготовчі дії перед тим, як запускати тести, і позначаються декоратором @pytest.fixture. У цьому проєкті в нас буде реалізовано дві фікстури: одна зчитує BASE URL та з'єднує його з нашим PATH (частинка URL, яка визначає конкретний ресурс або місце на вебсервері), а інші фікстура реалізує логування, що дає змогу переглянути вміст відповіді від сервера після надсилання запиту:

```
import logging
import pytest
from requests_toolbelt import sessions

URL = "https://petstore.swagger.io/v2/"

@pytest.fixture(scope="function")
def logger():
  logger = logging.getLogger("api")
   return logger

@pytest.fixture(scope="function")
def api_client():
   api_client = sessions.BaseUrlSession(base_url=URL)
   return api_client
Нехай в нас є тварина, інформація про яку зберігається у змінній post_body:
post_body = {
   "id": 1,
   "category": {
      "id": 0,
      "name": "category1"
   },
   "name": "Bob",
   "photoUrls": [
      "string"
   ],
   "tags": [
      {
         "id": 0,
         "name": "string"
      }
```

```
    ],
    "status": "available"
}
```

Структура тесту складається з надсилання запиту на інформацію про тварину, отримання статус-коду відповідно до того, чи була операція успішна, та валідацію отриманих даних з тими даними, які ми вказували під час створення тварини (змінна post_body).

```
def test_find_by_valid_id(api_client, logger):
    petID = 1
    response = api_client.get(url=f"pet/{petID}")
    logger.info(response.text)
    response_body = response.json()
    assert response.status_code == 200
    assert post_body['id'] == response_body['id']
```

Запускаємо тест та перевіряємо результати. Якщо логіка програми відпрацьовує коректно, то на виході сервер поверне нам статус-код 200 та інформацію про нашу тварину у форматі JSON, що відповідає тим даним, які ми вносили під час створення вихованця [рис. 3].

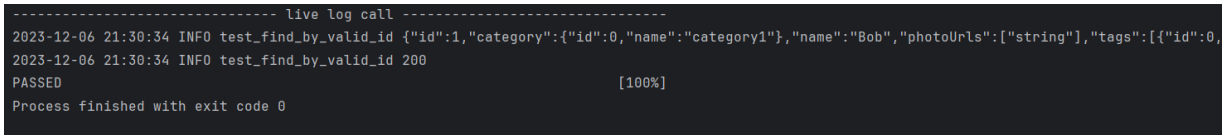


Рисунок 3. Результат виконання тесту

Отже, ми маємо коректні дані та статус-код 200, що значить правильне відпрацювання логіки програми. Тому якщо вважати, що ми маємо такий вебдодаток, який реалізовано для загального користування, і він має графічний користувацький інтерфейс, а інформація про нашу тваринку не відображається, хоча вона була попередньо занесена до програми, то проблему потрібно шукати саме на стороні front-end розробників.

Висновки. Ми визначили, що проблема у некоректному відображені даних зовсім не стосується логіки програми, сервер правильно обробляє та надсилає дані на клієнт, що полегшить роботу розробникам та збереже їх час.

Список використаних джерел

1. Swagger. URL: https://petstore.swagger.io/#/ (дата звернення: 04.12.2023).
2. Pytest. URL: https://docs.pytest.org/en/7.4.x//#/ (дата звернення: 04.12.2023).
3. Poetry. URL: https://python-poetry.org/ (дата звернення: 04.12.2023).

УДК 004.4

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Труханська В. О., здобувачка 3 курсу спеціальності 122 Комп'ютерні науки,
Ніколюк П. К., д-р фіз.-мат. наук, професор, професор кафедри інформаційних технологій

ПАТЕРНИ ПРОЄКТУВАННЯ У ПРОГРАМУВАННІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Патерни проєктування – рішення для певних проблем, які часто зустрічаються під час проєктування архітектури програмного забезпечення. Різниця патернів та готових бібліотек у тому, що патерн не є конкретним кодом чи рішенням, а лише загальним принципом для вирішення проблеми, яка виникає під час проЄктування програми. Патерн не можна скопіювати, його завжди потрібно реалізовувати під конкретну архітектуру ПЗ [1].

Плюсами використання патернів є:

1. Стандартизація коду – патерни мають готові рішення, які можна просто реалізовувати у своїх проєктах, не витрачаючи час на проєктування власної архітектури.

2. Типові рішення – більшість патернів перевірені та оптимізовані, що дає змогу уникнути зайвих проблем з архітектурою проєкту.

3. Зміни у структурі – патерни спрощують внесення змін до проєкту без значного впливу на всю програму.

4. Сприйняття структури – патерни значно покращують читабельність коду проєкту, що дає змогу іншим розробникам швидше орієнтуватися у проєкті, його функціях та архітектурі.

Розглянемо основні патерни, які найчастіше використовуються в сучасному програмному забезпеченні.

Одинак (Singelton) – патерн проєктування, який передбачає, що один клас має один екземпляр та одну точку вводу у програмі відповідно. Тобто патерн призначений для того, щоб гарантувати, що клас має тільки один екземпляр і забезпечує глобальний доступ до нього. Використання патерну сприяє поліпшенню читабельності коду, оскільки він визначає єдину точку доступу до об'єкта, уникнення дублювання та спрощення механізмів управління ресурсами.

Прототип (Prototype) – патерн проєктування, який призначений для копіювання об'єктів без розгляду їх внутрішньої структури та реалізації, шляхом створення нового екземпляру класу методами, реалізованими в цьому ж класі.

Фабричний метод (Factory) – патерн проєктування, який визначає загальний тип та ознаки об'єкта, реалізованого класом, і дає змогу визначати різні

значення цих ознак для його підкласів. Цей метод сприяє легкості розширення програмних систем, оскільки дає змогу додавати нові класи без зміни коду вже наявних класів.

Абстрактна фабрика (Abstraction factory) – патерн проєктування, що дає змогу створювати групи пов'язаних об'єктів, не прив'язуючись до конкретних класів створюваних об'єктів.

Будівельник (Builder) – патерн проєктування, що дає можливість покроково створювати складні об'єкти, використовуючи один код [2].

Інновації у сфері патернів проєктування можуть бути спрямовані на покращення ефективності, гнучкості та адаптивності в програмному забезпеченні. Однією з інновацій може стати розробка динамічних патернів проєктування. Основна ідея полягає в тому, щоб розширювати та змінювати патерни в реальному часі, щоб забезпечити адаптивність системи до динамічних вимог. Можливі переваги динамічних патернів.

1. Автоматичне адаптування (система може самостійно визначати, коли необхідно застосовувати той чи інший патерн, в залежності від поточного стану або обставин).

2. Автоматичне управління ресурсами (система може динамічно вирішувати, коли застосовувати ефективні патерни для оптимізації використання ресурсів).

3. Динамічні зміни у реальному часі (можливість миттєвої зміни патернів без перезапуску програмного забезпечення, що дає змогу більш швидко реагувати на змінні умови).

4. Самоналаштування і адаптація (можливість системи вивчати та адаптуватися до оптимальних патернів на основі досвіду та даних використання у залежності від умов у системі).

5. Зменшення кількості коду (динамічне застосування патернів може призвести до меншого обсягу коду та більшої гнучкості).

Згадані патерни дають змогу підвищити адаптивність системи до змін та зробити розробку програмного забезпечення більш гнучкою та ефективною. Важливо, щоб динамічні зміни патернів були добре збалансованими та мали чіткі стратегії управління, щоб уникнути складнощів у розумінні й обслуговуванні системи [3].

Список використаних джерел

1. Що таке патерни проєктування. URL: https://robotdreams.cc/uk/blog/301-chto-takoe-patterny-proektirovaniya (дата звернення: 12.11.2023).

2. Огляд патернів проєктування. URL: https://www.globallogic.com/ua/insights/blogs/dive-into-the-selenium-patterns/ (дата звернення: 13.11.2023).

3. Design Patterns in the Teaching of Programming. URL: https://www.sciencedirect.com/science/article/pii/S1877042814044218 (дата звернення: 12.11.2023).

УДК 519.6

Цегольник В. В., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Хмелівський Ю. С., асистент кафедри інформаційних технологій

ОПТИМІЗАЦІЯ ВИКОРИСТАННЯ СОНЯЧНОЇ ЕНЕРГІЇ ЗА ДОПОМОГОЮ МЕТОДІВ ОПТИМІЗАЦІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

Сонячна енергія є відновлюваним джерелом енергії, яке має значний потенціал для задоволення потреб людства у енергії. За даними Міжнародного агентства з відновлюваної енергетики (IRENA), у 2022 р. встановлена потужність сонячних електростанцій у світі досягла 1 200 ГВт, що на 21 % більше, ніж у 2021 р. З кожним роком люди на дахах своїх будинків все більше починають встановлювати сонячні батареї з метою економії електроенергії.

Порівняно з декількома попередніми роками збільшення використання сонячних батарей є дуже значним. Це можна пояснити декількома факторами. По-перше, вартість сонячних батарей знизилася вдвічі за останні 10 років. Це зробило сонячну енергію доступнішою для більшого кола споживачів. По-друге, технології виробництва сонячних батарей постійно розвиваються, що дає змогу підвищувати їх ефективність і надійність, а також у багатьох країнах світу існують державні програми підтримки сонячної енергетики, як-от субсидії, податкові пільги та квоти на використання відновлюваних джерел енергії [1].

Однак сьогодні сонячні батареї все ще залишаються дороговартісним продуктом для отримання енергії у власне використання, а їх ефективність досі є під питанням економічної вигоди для людей, що не мають величезної площі для їх устаткування чи не живуть у місці, що постійно перебуває під освітленням сонячних променів. Окрім того, що кількість енергії виробляється невелика і тільки в певний період дня, її необхідно або використовувати одразу, або зберігати також у недешевих акумуляторах чи теплових накопичувачах. І оскільки енергії не так і багато, для того, щоб вона стала більш конкурентоспроможною з традиційними джерелами енергії, необхідно вирішити низку проблем, пов'язаних із її використанням – як її отримати більше чи використовувати довше. Методи оптимізації мають для цього декілька рішень.

За допомогою методів оптимізації можна знайти такі рішення, які дають змогу максимально ефективно використовувати сонячну енергію з урахуванням таких факторів:

- Оптимальне розміщення сонячних батарей. Розміщення сонячних батарей має важливий вплив на їх продуктивність. Методами оптимізації можна знайти

таке розміщення, яке буде максимально ефективним із погляду захоплення сонячного світла.

➢ Оптимальний вибір типу сонячних батарей. Сонячні батареї бувають різних типів і мають різні характеристики. Методами оптимізації можна знайти такий тип сонячних батарей, який буде оптимальним для конкретних умов використання.

➢ Оптимальне управління сонячними батареями. Сонячними батареями можна керуватись так, щоб максимально ефективно використовувати їх енергію, оскільки однією з основних проблем є їх низька ефективність. Сьогодні ефективність сонячних батарей становить приблизно 20 %. Це означає, що лише 20 % сонячного світла, яке потрапляє на сонячну батарею, перетворюється в електричну енергію. Методами оптимізації можна розробити такі алгоритми управління сонячними батареями, які будуть відповідати конкретним потребам споживача [2].

Розглянемо модель реальної проблеми. Нам необхідно визначити серед сонячних батарей різних характеристик, які поставити в себе на території найвигідніше, маючи обмежену площу. Нехай у нас є площа розміром 100 м^2. Доступні нам сонячні батареї наведені в таблиці 1. Кожна сонячна батарея має такі основні параметри: розмір, потужність і ефективність. Для знаходження кількості енергії, що приносить сонячна плита, використовуватимемо формулу добутку потужності та ефективності плити. Інтенсивність освітлення вважатимемо нормальною впродовж усього часу роботи.

Таблиця 1 – Характеристики сонячних плит

Розмір	**Потужність**	**Ефективність**
5 м^2	300 Вт	0,18
10 м^2	600 Вт	0,2
15 м^2	900 Вт	0,25

Для розв’язання поставленої задачі можна використовувати різні методи оптимізації. Один із можливих підходів – це використання методу симплекс-таблиці. У цьому методі розглядається кожне можливе розташування сонячних батарей як один розв’язок задачі оптимізації. Для кожного розв’язку розраховується загальна потужність сонячних батарей. Потім зазвичай використовується метод симплекс-таблиці для пошуку найбільш оптимального розв’язку, тобто розв’язку, який забезпечує максимальну загальну потужність сонячних батарей [3].

Модель буде мати такий вигляд:

$$Z = 300 * 0{,}18 * x1 + 600 * 0{,}2 * x2 + 900 * 0{,}25 * x3 \rightarrow \max;$$

$$\begin{cases} 5 * x1 + 10 * x2 + 15 * x3 \leq 100; \\ x1 \geq 0, x2 \geq 0, x3 \geq 0. \end{cases}$$

На рис. 1, використовуючи надбудову Розв'язувач платформи Excel, налаштуємо модель на знаходження максимального значення й обов'язково вказуємо пошук розв'язку в цілих числах, оскільки ми досліджуємо кількість батарей для встановлення, що продукуватимуть нам максимальну кількість енергії.

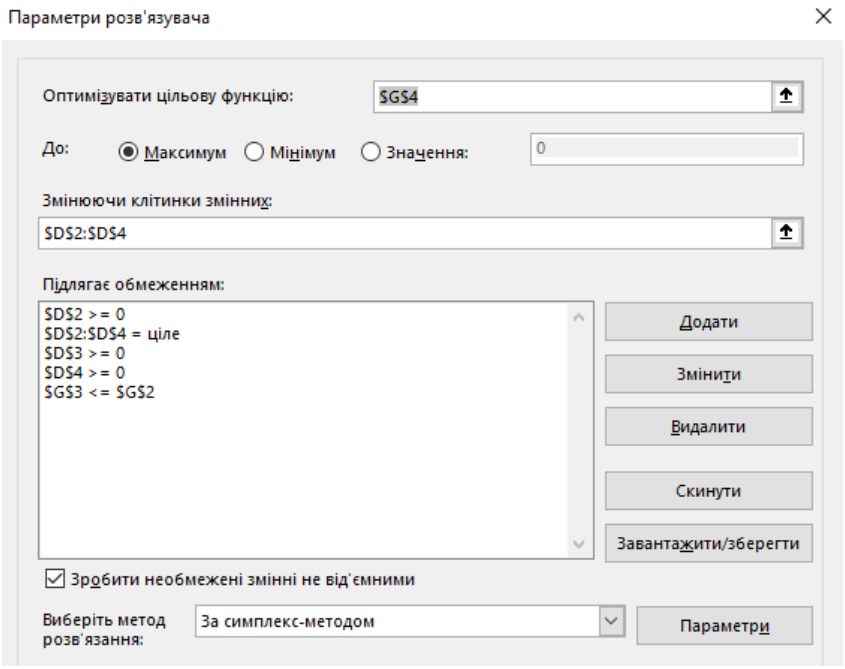


Рисунок 1. Параметри розв'язувача з даними моделі

Отже, з рис. 2 бачимо, що максимально отримана енергія з досліджуваними даними становитиме 1 470 Вт, або ж 1,47 кВт за годину, за нормальних умов інтенсивності світла.

	A	B	C	D	E	F	G
1	Розмір	Потужність	Ефективність	Результат			
2	5	300	0,18	0		Площа (дана):	100
3	10	600	0,2	1		Площа (заповнена):	100
4	15	900	0,25	6		Отримана енергія:	1470
5							

Рисунок 2. Результат розв'язання задачі

У реальних умовах кількість енергії, яку продукує сонячна батарея за 1 год, може бути нижчою, ніж розрахункова, оскільки на ефективність сонячної батареї також впливають фактори температури, затінення, інші погодні умови. Зазвичай сонячна батарея потужністю 100 Вт виробляє від 10 до 20 Вт год енергії

за 1 год. А оскільки сумарно за результатом розрахунків ми отримали 7 сонячних батарей, що потужніші в 6 та 9 разів, то провівши відповідні розрахунки, можемо вважати, що отримана кількість енергії є правдоподібною.

Список використаних джерел

1. IRENA: Vast majority of new renewables in 2022 have lower costs than fossil fuel–fired electricity. URL: https://balkangreenenergynews.com/irena-vast-majority-of-new-renewables-in-2022-have-lower-costs-than-fossil-fuel-fired-electricity/

2. Guide to Solar Optimization. URL: https://arkresources.com.au/guide-to-solar-optimization/

3. Simplex Method. URL: https://www.britannica.com/topic/simplex-method

УДК 004.021

Чемес В. С., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

ПОРІВНЯННЯ АЛГОРИТМІВ ПОШУКУ НАЙКОРОТШОГО ШЛЯХУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Однією з ключових задач, пов'язаних із пошуком найефективніших шляхів у графах та мережах, є пошук найкоротшого шляху між двома вершинами графу. Для вирішення цієї проблеми було розроблено різноманітні алгоритми, серед яких важливе місце займають алгоритми A* та Дейкстри.

Алгоритм Дейкстри – це метод для знаходження найкоротших шляхів у зважених графах із невід'ємними вагами ребер. Винахідником є голландський математик і інженер Едсгер Дейкстра (1959). Використовується для задач маршрутизації в телекомунікаційних мережах.

Основна ідея алгоритму полягає в систематичному відстеженні та оновленні найкоротших шляхів до кожної вершини з початкового вузла графу. Алгоритм використовує вагу шляху, що розповсюджується через ребра, обираючи вершину з найменшою вагою як поточний вузол. Оновлює ваги сусідніх вершин, враховуючи вагу ребра та активну вершину.

Кроки алгоритму:

1. Ініціалізація (встановлення ваги початкового вузла як 0, інші – як нескінченність).

2. Вибір вузла (обирається вершина з найменшою вагою як поточний вузол).

3. Оновлення ваги (обчислення ваг нових шляхів до сусідніх вершин. Оновлення ваги, якщо нова вага менша від поточної).

4. Позначення вузла (позначення поточного вузла як відвіданого).

5. Повторення (повторення кроків 2–4, доки не будуть відвідані всі вершини або досягнутий кінцевий вузол).

Алгоритм завершується, коли всі вершини відвідані або досягнутий кінцевий вузол. Важливо, що він працює лише для графів із невід'ємними вагами ребер і не обробляє від'ємні ваги [1].

Алгоритм A* є комбінацією точних методів та евристичних оцінок для ефективного пошуку найкоротшого шляху в графах. Його основна ідея полягає в тому, щоб враховувати як вартість досягнення поточної вершини, так і передбачувану вартість досягнення цільової вершини.

Кроки алгоритму A* включають:

1. Ініціалізація (встановлення початкової і цільової вершин, визначення початкового шляху).

2. Створення списків (відкритий список для неперевірених вершин та закритий список для вже оброблених).

3. Вартості (розрахунок вартостей для кожної вершини на основі вартості досягнення та евристичної оцінки до цільової вершини).

4. Вибір вершини (вибір вершини з найменшою загальною вартістю з відкритого списку).

5. Перевірка сусідів: (розгляд сусідніх вершин, оновлення вартостей, якщо новий шлях коротший).

6. Додавання та видалення вершин (додавання поточної вершини до закритого списку та видалення з відкритого).

7. Повторення (продовження кроків до досягнення цільової вершини або опорожнення відкритого списку).

8. Відновлення шляху (якщо цільова вершина досягнута, відновлення найкоротшого шляху від цільової до початкової вершини).

A* є ефективним та універсальним алгоритмом для пошуку шляхів у різних галузях застосування [2].

Тепер переглянемо програмну реалізацію кожного алгоритму на мові програмування Python. Ми розглянемо кожен з алгоритмів, який відображає найкоротший можливий шлях проходження від початкової точки (0, 0) до кінцевої точки (499, 499) і також відображає випадкові перешкоди, кількість яких становить 300.

Результати виконання кожного з алгоритмів можна розглянути на рис. 1 та 2 відповідно.

Execution Time: 6.765911102294922 seconds – це час виконання алгоритму Дейкстри.

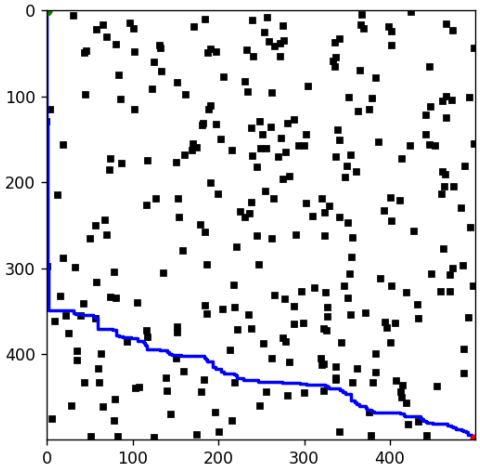


Рисунок 1. Шлях, утворений алгоритмом Дейкстри

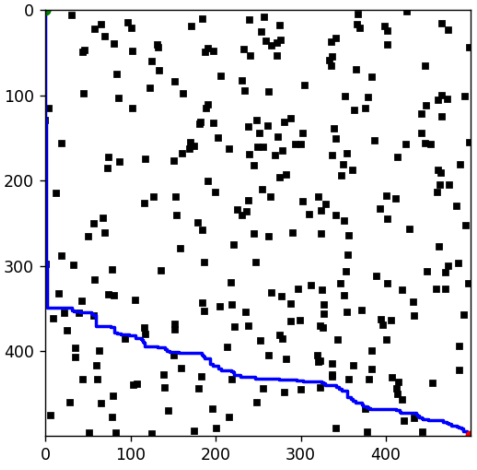


Рисунок 2. Шлях, утворений A-алгоритмом*

Execution Time: 0.8401174545288086 seconds – це час виконання A*-алгоритму. Як ми можемо помітити, A*-алгоритм виконується швидше за алгоритм Дейкстри у ≈8,05 разів. Алгоритм A* може бути більш швидким, порівняно з алгоритмом Дейкстри, у випадках, коли є евристична інформація про можливі вартості шляхів до кінцевої точки.

Основна відмінність між алгоритмами полягає в тому, що алгоритм A* додатково використовує евристичну функцію для оцінки залишкової вартості шляху до кінцевої точки. Ця евристична інформація дає змогу алгоритму A* більш ефективно обирати напрямок пошуку, спрямовуючись до очікувано більш «перспективних» шляхів.

Порівняно з алгоритмом Дейкстри, який розглядає всі можливі шляхи, алгоритм A* може прискорити пошук, концентруючись на областях, які, ймовірно, приведуть до оптимального шляху. Це особливо корисно у великих графах або задачах із великою кількістю можливих шляхів [3, 4].

Висновки. Алгоритм A* та алгоритм Дейкстри є ефективними методами пошуку найкоротших шляхів у графах, проте вони використовують різні стратегії для досягнення цієї мети.

Алгоритм Дейкстри гарантує знаходження найкоротшого шляху у графі з невід'ємними вагами ребер і розглядає всі можливі шляхи від початкової точки до кінцевої, вибираючи завжди найкоротший на поточний момент, але може бути витратним за часом, особливо у великих графах.

Алгоритм A* використовує евристичну інформацію для оцінки залишкової вартості шляху до кінцевої точки і спрямовує пошук у напрямку, який, імовірно, приведе до оптимального шляху, що може підвищувати ефективність у великих графах, але вимагає уважного вибору та налаштування евристичної функції для досягнення оптимальності.

Загалом вибір між алгоритмом A* та алгоритмом Дейкстри залежить від конкретних умов задачі. Якщо є можливість коректно визначити евристичну інформацію і вона допомагає зменшити простір пошуку, то A* може бути більш вигідним. У випадках, коли точна інформація важлива, алгоритм Дейкстри може бути більш оптимальним вибором.

Список використаних джерел

1. Dijkstra's algorithm. URL: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm (дата звернення: 27.11.2023).

2. A* search algorithm. URL: https://en.wikipedia.org/wiki/A*_search_algorithm (дата звернення: 28.11.2023).

3. Dijkstra's Algorithm. URL: https://www.programiz.com/dsa/dijkstra-algorithm (дата звернення: 30.11.2023).

4. A* Search Algorithm. URL: https://www.geeksforgeeks.org/a-search-algorithm/ (дата звернення: 01.12.2023).

УДК 311

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,

Волонтир Л. О., канд. техн. наук, доцент, доцент кафедри інформаційних технологій

МОЖЛИВОСТІ ВИКОРИСТАННЯ РАНГОВОЇ КОРЕЛЯЦІЇ В РЕАЛЬНОМУ ЖИТТІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Зв'язок між різними змінними є ключовим елементом аналізу даних у багатьох галузях. У світі різноманітних даних, коли розподіл даних може бути неоднорідним або відсутній нормальний характер, методи кореляції є важливим інструментом для виявлення та вимірювання ступеня зв'язку між змінними [1].

Один із таких методів – рангова кореляція – відіграє ключову роль у визначенні ступеня взаємозв'язку між змінними, які можуть мати порядковий характер або не відповідати умовам нормального розподілу. Використання рангових коефіцієнтів кореляції дає змогу робити висновки про взаємозв'язок між змінними навіть у випадках, коли інші методи кореляції втрачають свою ефективність.

Розглянемо основні можливості використання рангової кореляції в реальному житті [2]:

- У медичних дослідженнях рангова кореляція відіграє важливу роль у визначенні зв'язків між різними параметрами, як-от ефективність лікування,

симптоми хвороби, рівень болю та інші фактори, що вимірюються на порядковій шкалі. Особливо корисна рангова кореляція в ситуаціях, де точні числові значення можуть бути складними для отримання або коли дані мають обмеженість у нормальному розподілі. Наприклад, у випадках дослідження ефективності лікування рангова кореляція може допомогти встановити зв'язок між застосуванням конкретного методу терапії та зменшенням симптомів у пацієнтів [3].

➢ У соціальних науках рангова кореляція відіграє значну роль у встановленні зв'язків між різними соціальними явищами, уподобаннями, психологічними характеристиками та іншими аспектами, що можуть бути виміряні за порядковою шкалою. У сфері соціології рангова кореляція може бути використана для визначення зв'язку між соціальною класифікацією, освітою або доходами та певними соціальними поведінками чи уподобаннями. Наприклад, у психологічних дослідженнях рангова кореляція дає змогу встановлювати зв'язок між рівнем задоволення від життя та різними факторами, як-от рівень стресу, соціальна підтримка або стиль життя. Це допомагає розуміти, які аспекти нашого життя впливають на загальний рівень задоволення.

➢ У фінансовому аналізі рангова кореляція відіграє важливу роль у вивченні зв'язків між різними фінансовими показниками та рейтингами. Вона може бути використана для встановлення ступеня взаємозв'язку між рівнем доходу компаній, їх ринковою капіталізацією, фінансовими показниками та іншими фінансовими характеристиками.

Наприклад, у фінансовому аналізі рангова кореляція може допомогти визначити, які фактори найбільше впливають на дохідність компанії чи її рейтинг. Це може включати аналіз взаємозв'язку між фінансовими показниками (наприклад, оборотність активів, прибутковість, фінансова стабільність тощо) та рейтингом компанії на фондовому ринку або кредитною агенцією.

➢ Управління ресурсами охоплює широкий спектр діяльності, зокрема й управління людськими ресурсами, матеріальними активами, фінансами та іншими ресурсами в організаціях. Рангова кореляція може бути застосована в цій сфері для аналізу пріоритетів, ефективності робочих груп, прийняття управлінських рішень та вирішення питань взаємодії між різними ресурсами.

Наприклад, у випадках управління людськими ресурсами рангова кореляція може бути використана для визначення зв'язку між показниками, як-от рівень задоволеності працівників, їхня продуктивність та інші соціальні фактори. Це допомагає визначити фактори, які впливають на ефективність роботи персоналу, та виявити стратегії підвищення рівня задоволення і продуктивності працівників.

➢ Економічні дослідження використовують рангову кореляцію для аналізу зв'язків між різними економічними показниками та факторами. Це може сто-

суватися досліджень економічного зростання, розподілу доходів, співвідношення економічних факторів із соціальними показниками та багатьма іншими аспектами економіки. Наприклад, у дослідженнях економічного зростання рангова кореляція може використовуватися для встановлення зв'язку між показниками ВВП на душу населення, рівня інвестицій, технологічного прогресу та ін. факторів, що можуть впливати на економічний розвиток країни.

Список використаних джерел

1. Рангова кореляція. URL: http://elbib.in.ua/rangova-korelyatsiya-doslidjennya-sotsialno-ekonomichnih-i-politichnih-protsesiv.html (дата звернення: 01.12.2023).

2. Rank Correlation Coefficient. URL: https://www.sciencedirect.com/topics/mathematics/rank-correlation-coefficient (дата звернення: 01.12.2023).

3. When rank correlation using. URL: https://www.dataversity.net/what-is-spearmans-rank-correlation-and-how-is-it-useful-for-business-analysis/ (дата звернення: 01.12.2023).

УДК 519.17:164.01

Юстименко Є. А., здобувач 3 курсу спеціальності 122 Комп'ютерні науки,
Сеник І. О., асистент кафедри інформаційних технологій

МОДЕЛІ НА ОСНОВІ ТЕОРІЇ ГРАФІВ У ЛОГІСТИЦІ

Донецький національний університет імені Василя Стуса, м. Вінниця

Логістика в сучасному світі вимагає постійної оптимізації та ефективного управління ланцюжками постачання. Використання моделей на основі теорії графів стає потужним інструментом для аналізу, планування та оптимізації логістичних процесів. Використовуючи графи та сучасні технології в інформаційних системах, можна здобути дуже потужний інструмент для логістичних потреб. Ця публікація пропонує огляд різних застосувань графових моделей у логістиці та їх вплив на оптимізацію логістичних систем.

У комп'ютерних науках граф – це абстрактна структура даних, яка являє собою колекцію вузлів (вершин) та зв'язків між ними (ребер). Вона використовується для моделювання взаємозв'язків між об'єктами в комп'ютерних науках, логістиці, економіці тощо [1].

У графі вузли являють собою об'єкти, а ребра – зв'язки між цими об'єктами. Ребра графу можуть бути напрямленими або ненапрямленими. Графи можуть бути орієнтованими, коли ребра мають напрям, або неорієнтованими, коли вони не мають напряму. Кожне ребро може мати додаткову інформацію,

як-от вага чи мітка, що відображає його характеристики або відносини між вершинами.

Графи широко використовуються для вирішення різноманітних завдань, зокрема для моделювання мереж, аналізу соціальних структур, оптимізації маршрутів, файлових систем тощо.

Графові моделі в плануванні маршрутів

➢ Однією з ключових областей у логістиці є оптимізація транспортних маршрутів. Застосування графових моделей дає змогу ефективно планувати оптимальні маршрути доставки, враховуючи різноманітні фактори, як-от відстань, час, вартість і обмеження. Планування оптимальних маршрутів є актуальною проблемою і в мирний час, і в умовах війни, коли з'являються великі обсяги переміщення ресурсів у межах та поза межами нашої країни.

Управління запасами за допомогою графових структур

➢ Графові моделі допомагають оптимізувати розміщення та управління запасами на складах. Вони дають змогу аналізувати оптимальні шляхи для зберігання товарів, мінімізуючи витрати на зберігання та оптимізуючи обслуговування замовлень.

➢ Графи можна використовувати для моделювання складів, де вершини представляють різні місця зберігання, а ребра – шляхи, по яких товари переміщуються між цими місцями. Ця модель дає змогу враховувати різні фактори, як-от час, відстань, вартість доставки, обсяги запасів і обмеження, що допомагає у розв'язанні задач ефективного розподілу та руху товарів.

Моделі мереж та оптимізація логістичних систем

➢ Графові структури дають змогу представити логістичні системи у вигляді мережі зв'язків між різними елементами. Моделі мереж та їх оптимізація в логістиці відіграють ключову роль у плануванні, управлінні та вдосконаленні логістичних систем. Вони використовують графові структури для моделювання та аналізу мережі постачання, складів, транспорту й інфраструктури, забезпечуючи оптимальне функціонування системи. Застосування графових структур у цій сфері може сприяти більш інтелектуальному й точному плануванню та управлінню логістичними потоками. Це дає змогу покращити управління логістичними системами, зменшити витрати, підвищити ефективність та реагувати на зміни у внутрішньому і зовнішньому середовищі.

Використання графів для аналізу та візуалізації логістичних даних

➢ Графові моделі стають потужним інструментом для аналізу та візуалізації великих обсягів логістичних даних. Візуалізація логістичних даних за допомогою графових структур дає змогу зрозуміти складні зв'язки та шляхи переміщення товарів чи інформації. Це полегшує прийняття рішень, допомагає виявити можливість оптимізації логістичних процесів і дає змогу оперативно виявляти та вирішувати проблеми у логістичних процесах.

Застосування графових моделей у логістиці є перспективним напрямом для вдосконалення управління логістичними системами. Їх потужність у плануванні, оптимізації та аналізі робить їх важливим інструментом для досягнення ефективності й оптимальності у логістиці.

Список використаних джерел

1. Поняття та побудова графів. URL: https://disted.edu.vn.ua/courses/learn/14043 (дата звернення: 12.11.2023).

2. Теорія графів. URL: https://ela.kpi.ua/bitstream/123456789/35854/1/Teoriia_hrafiv.pdf (дата звернення: 13.11.2023).

3. Graph theory. URL: https://www.xomnia.com/post/graph-theory-and-its-uses-with-examples-of-real-life-problems/ (дата звернення: 12.11.2023).

УДК 004.065

Яценко В. В., здобувач 2 курсу спеціальності 122 Комп'ютерні науки,

Вєтров О. С., старший викладач кафедри прикладної математики та кібербезпеки

ОГЛЯД АЛГОРИТМІВ ЧИТАННЯ ТА ЗАПИСУ В РЕЛЯЦІЙНИХ БАЗАХ ДАНИХ ТА СПОСОБИ ЇХ ОПТИМІЗАЦІЇ

Донецький національний університет імені Василя Стуса, м. Вінниця

Методи аналізу і обробки упорядкованих і неупорядкованих даних для добування знань називається наукою про дані [1]. Упорядкована інформація, об'єднана між собою за певними критеріями, називається базою даних. Правила представлення, організації даних і їх зв'язків називається моделлю даних [2]. Існують реляційна, об'єктно-орієнтована, графова, мережева, ієрархічна моделі даних. Найпопулярніша модель – реляційна модель представляє сутності у вигляді двовимірних таблиць. Для роботи з реляційними базами даних використовується SQL [3]. SQL – це мова структурованих запитів, що дає змогу користувачу взаємодіяти з даними у реляційних базах даних.

Ресурси knowledgehut, gigaspaces досліджують метод аналізу даних у реальному часу [4, 5]. Згідно з їх дослідженням, сучасні компанії все частіше використовують метод аналізу даних у реальному часі, оскільки зі звичайними методами збору й аналізу інформації організації не зможуть швидко реагувати на зміни і відповідно миттєво приймати рішення залежно від ситуації. Швидка обробка даних критично важлива, оскільки дає змогу виявити загрози, проблеми й недоліки у різних галузях на початковому етапі і швидко зреагувати на

подібні події. Відтак швидкість читання і запису даних критично важливі для швидкого опрацювання інформації та реагування на неї.

Метою дослідження є аналіз алгоритмів читання і запису та методики їх оптимізації. Необхідно дослідити, як у реляційних базах даних відбувається читання і запис даних, дослідити роботу запитів, надати можливі методи підвищення продуктивності алгоритмів і підсумувати інформацію.

Розглянемо, як виконуються запити у базах даних. Для дії над даними необхідно здійснити запит мовою SQL. Процес здійснення запиту можна поділити на кілька етапів. На першому етапі здійснюється розбір тексту запиту. Відбувається синтаксичний аналіз і перевірка запиту на помилки. Цей процес називається парсингом запиту. Якщо на цьому етапі буде знайдена синтаксична помилка – обробка запиту припиниться і виведеться помилка. Якщо етап синтаксичного аналізу буде пройдено вдало, він буде розділений на частини і буде створено дерево послідовності виконання запиту. Дерево являє собою кроки, які необхідно здійснити для виконання запиту. На наступному етапі дерево парсингу передається до алгебризатора. Алгебризатор перевіряє назви таблиць, атрибутів, полів, вказаних у запиті. Перевіряються і розпізнаються всі посилання, вказані у запиті. Якщо хоча б одне з посилань – некоректне, робота над запитом буде зупинена і виведеться відповідна помилка. Процес, що проводиться алгебризатором називається прив’язкою запиту. Згодом, якщо всі посилання пройшли перевірку, відбувається перевірка типів цих посилань і визначаються агрегатори – відбувається прив’язка агрегаторів. Фактично на цьому етапі відбувається перевірка, чи дійсно імена вказані у запиті, існують у вказаних таблицях, чи правильно вказані назви полів. Якщо всі перевірки були успішно пройдені, створюється дерево процесора запитів. Це двійковий файл, що містить дані про запит у вигляді кодованого значення – хеш-значення. За допомогою хеш-значення визначається, чи має цей запит актуальний план виконання. План вважається доцільним, якщо не було здійснено змін у таблиці. Якщо у запиту існує дійсний план виконання, процес обробки запиту припиняється і використовується план виконання запиту для оптимального виконання операцій, заданих користувачем. Якщо ж такого плану не існує, виконання запиту переходить на наступний етап. Відбувається планування виконання запиту – пошуку найоптимальнішого виконання цього запиту. Оптимізатор запитів здійснює пошук і оцінку всіх шляхів виконання запиту. Оцінюється швидкість виконання і ресурсозатратність кожного методу виконання запиту. Залежно від складності запит може пройти повну оптимізацію й отримати план на основі повної переоцінки всіх можливих шляхів виконання і може отримати тривіальний план. Тривіальні плани отримують запити, настільки легкі у виконанні, що запуск повної оптимізації не є раціональним рішенням.

Зчитування даних із бази даних здійснюється за допомогою запиту SELECT FROM. За допомогою цього запиту можна вибирати певні поля. Для вибірки певного поля з певної таблиці необхідно вказати назву поля і назву таблиці, якій це поле належить, наприклад: SELECT name FROM employee, де name – це назва поля, дані з якого потрібно зчитати, employee – назва таблиці де знаходиться це поле. Також можна обрати всі поля з певної таблиці, використовуючи синтаксис SELECT*. Дані вибірки можуть бути відсортовані за допомогою ORDER BY. Також можна обмежити вибірку за допомогою LIMIT.

Для запису даних у базу даних використовується запит INSERT INTO VALUES. За допомогою цього запиту можна записувати значення в певні або в усі стовпці. За допомогою запиту UPDATE SET WHERE можна оновити дані у певних таблицях.

Першим методом оптимізації є нормалізація бази даних. Нормалізація бази даних дасть змогу зменшити надмірність даних, імовірність появи аномалій, пришвидшити запис даних у базу даних. До того ж нормалізовані бази даних мають більш ефективні індекси, що також позитивно вплине на продуктивність читання і запису. Але з іншого боку, це збільшить кількість з'єднань, через які запиту потрібно пройти для пошуку необхідної інформації, що відповідно сповільнить процес читання даних таблиць.

Другий метод оптимізації читання і запису – оптимізація запитів. Насамперед, використання оператора OR не є раціональним. SQL не може обробити оператор в одній операції. У разі використання оператора UNION та задання подвійної умови в якості двох умов SQL сможе використати індекси, і запит відбудеться швидше та ефективніше. По-друге, часте використання JOIN і підзапитів у запиті сповільнює процес аналізу запиту. SQL не зможе якісно побудувати план виконання й оптимізувати запит. До того ж не гарантується, що побудований план буде максимально ефективним. Розділення запиту на кілька менших запитів зменшить кількість JOIN і підзапитів, і так прискорить аналіз і виконання. По-третє, варто уникати SELECT DISTINCT. Такі запити відбуваються значно повільніше. До того ж використання SELECT* у всіх запитах не є раціональним рішенням. Потрібно вказувати лише потрібні поля, і якщо це можливо, виставляти ліміт вибірки. Це значно прискорить швидкість роботи запиту і значно зменшить використання ним ресурсів.

Третім методом оптимізації є індексація даних. Індексування значно прискорює пошук інформації у базі даних. Існує два види індексів: кластеризовані і некластеризовані індекси. Кластеризовані індекси – індекси, які сортують дані в таблиці за ключовим значенням. Кластеризований індекс визначає порядок розміщення даних у таблиці. На кожну таблицю може бути створений лише один кластеризований індекс. Кластеризовані індекси значно пришвидшують послідовне читання інформації. Такі індекси також значно прискорять вибірки

великої кількості полів. До того ж більша продуктивність буде помітна і дід час роботи з діапазонами. Однак кластеризований індекс ускладнює запис даних у таблиці, особливо якщо відбувається багато непослідовних вставок. В такому випадку продуктивність запису даних буде знижена. Некластеризований індекс – індекс, який зберігає посилання на певні дані, і на відміну від кластеризованого не впорядковує їх фізично. Некластеризований індекс дає зиогу швидко отримати доступ до певного поля, але якщо буде вибірка з багатьох полів, то цей метод індексації не буде раціональним у використанні. Перевагою некластеризованого індексу також є пришвидшення роботи INSERT- і UPDATE-запитів, оскільки для здійснення таких операцій не потрібне сортування даних. Використання індексів дійсно пришвидшує роботу з даними, але занадто велика кількість індексів може сповільнити швидкість запису даних і зайняти багато дискового простору.

Четвертий метод оптимізації – кешування. Це процес зберігання даних, що часто використовуються в оперативній пам'яті або на жорсткому диску користувача. Існує декілька методик кешування: Cache-Aside, Read-through Cache, Write-back Cache, Write-through Cache.

Cache-aside працює так, що якщо даних немає в пам'яті, відбувається звернення до бази даних. Кеш оновлюється новими даними, і тоді повертається результат запиту. Перевагою цієї методики є вирішення проблеми частих зчитувань. До того ж дані зберігаються в кеші, лише якщо вони потрібні, що заощаджує пам'ять системи. Read-through Cache працює аналогічно, за винятком того, що користувач завжди звертається до кешу. Переваги і недоліки аналогічні першому методу.

Write-back Cache – дані постійно записуються в кеш, але не зберігаються в базі даних доти, доки не відбудеться запит на збереження внесених змін. Отже в пам'яті завжди буде актуальна інформація, а запис даних у базу даних буде значно ефективніший. Недоліком способу є можлива втрата даних у разі зупинки системи.

Write-through Cache записує дані в кеш і одразу переносить їх у базу даних. Перевагою методу є те, що у пам'яті завжди буде актуальна інформація, але необхідно робити два записи одночасно – у кеш і в базу даних, що може бути ресурсозатратно. Отжк методики кешування пришвидшують доступ до даних, що регулярно використовуються шляхом зменшення кількості звернень до бази даних. Це відповідно призводить до зменшення навантаження на базу даних і значного скорочення часу обробки запиту. Вибір методики кешування залежить від потреб користувача.

П'ятий метод оптимізації – використання масових операцій, або об'єднання малих запитів у один. Методика масових операцій – виконання багатьох

однотипних операцій одночасно. Це дає змогу значно прискорити обробку даних. Також, можна власноруч об’єднати запити в один, що значно пришвидшить запис і оновлення даних у таблиці.

У підсумку зазначимо, що швидка обробка інформації критично важлива. Велика швидкість аналізу інформації дає змогу ефективно виявляти проблеми на перших етапах та відповідно негайно реагувати і приймати миттєві рішення. Особливо важливою стає швидкість обробки запитів читання і запису в базу даних, оскільки від цього буде залежати вчасність прийняття рішення. SQL володіє вбудованими методами оптимізації запитів і виконує їх за найбільш ефективним планом, але методи оптимізації все ж відіграють важливу роль у продуктивності системи. Ефективна структура бази даних, оптимізовані запити, правильно налаштовані індекси, а також використання різних методик кешування і масових операцій можуть значно збільшити швидкість і ефективність системи.

Список використаних джерел

1. What is data science. URL: https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science (дата звернення: 25.10.2023)

2. What is data model. URL: https://cedar.princeton.edu/understanding-data/what-data-model (дата звернення: 25.10.2023)

3. What is SQL. URL: https://www.geeksforgeeks.org/what-is-sql/ (дата звернення: 25.10.2023)

4. Real-time data analytics. URL: https://www.knowledgehut.com/blog/data-science/real-time-data-analytics (дата звернення: 25.10.2023)

5. Real-Time Data Processing. URL: https://www.gigaspaces.com/data-terms/real-time-data-processing (дата звернення: 25.10.2023)

Наукове видання

Колектив авторів

МАТЕРІАЛИ IV ВСЕУКРАЇНСЬКОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ СТУДЕНТІВ, АСПІРАНТІВ ТА МОЛОДИХ ВЧЕНИХ

Редактор О. А. Солдатова
Технічний редактор Т. О. Важеніна-Гопрак

Підписано до друку 26.12.2023.
Формат 60×84/16. Папір офсетний.
Друк – цифровий. Умовн. друк. арк. 17,21.
Тираж 30. Зам. 1.

Донецький національний університет імені Василя Стуса
21021, м. Вінниця, 600-річчя, 21
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру
серія ДК № 5945 від 15.01.2018