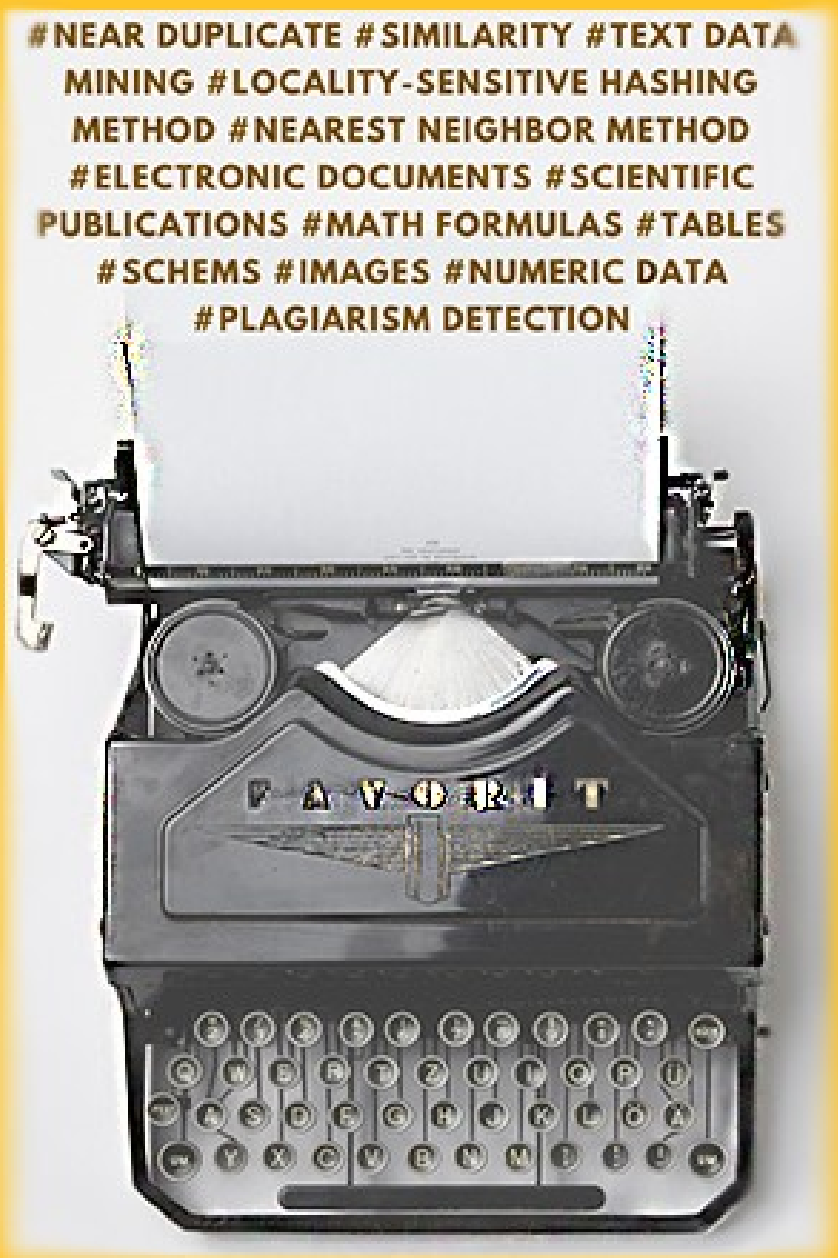
Лізунов П.П., Білощицький А.О, Кучанський О.Ю.,
Чала Л.Е., Діхтяренко О.В., Білощицька С.В.

МОНОГРАФІЯ

Інформаційна технологія пошуку неповних дублікатів у графічних даних, математичних формулах та таблицях



ЛІЗУНОВ П.П., БІЛОЩИЦЬКИЙ А.О., КУЧАНСЬКИЙ О.Ю.,
ЧАЛА Л.Е., ДІХТЯРЕНКО О.В., БІЛОЩИЦЬКА С.В.

Інформаційна технологія пошуку неповних дублікатів у графічних даних, математичних формулах та таблицях

Монографія

Київ 2017

УДК 004.912
ББК 73
Л 55

Рекомендовано до друку Вченою радою
Київського національного університету будівництва і архітектури
(протокол № 8 від 24 листопада 2017 р.)

Р Е Ц Е Н З Е Н Т И :

В.А. Рач
д-р техн. наук, проф.,
директор навчально-наукового інституту інформаційних та телекомунікаційних технологій
Університету «КРОК»

В.Д. Гогунський
д-р техн. наук, проф., заслужений діяч науки і техніки України,
завідувач кафедри управління системами безпеки життєдіяльності
Одеського національного політехнічного університету

Л 55 Лізунов П.П.
Інформаційна технологія пошуку неповних дублікатів у графічних даних, математичних формулах та таблицях [Текст] : монографія. / П.П. Лізунов, А.О. Білощицький, О.Ю. Кучанський, Л.Е. Чала, О.В. Діхтяренко, С.В. Білощицька. – К: КНУБА, 2017. – 136 с.

Монографія присвячена актуальній проблемі побудови моделей та методів виявлення неповних дублікатів у графічних даних, математичних формулах та таблицях.

Здійснено аналіз наукових розробок та програмного забезпечення, що дозволяє знаходити неповні дублікати зображень, схем та діаграм в електронних документах. Запропонована гібридна схема порівняння математичних формул, яка передбачає можливість аналізу подібностей та виявлення повного або часткового дублювання формул як з використанням шаблонів формульних об'єктів, що аналізуються, так і з застосуванням конвертації математичних формул у різних форматах на універсальну мову математичної розмітки. Представлено гібридні моделі та методи виявлення неповних дублікатів у таблицях, що базуються на методах найближчого сусіда та локально-чутливого хешування. Створено та описано модуль візуалізації інформації, інтерфейс користувача, структурні схеми супроводу наукових робіт в Департаменті атестації кадрів вищої кваліфікації та ліцензування Міністерства освіти і науки України та розмежування прав доступу в програмному комплексі визначення неповних дублікатів у електронних документах.

Монографія призначена для науковців, спеціалізованих вчених рад та організацій, в роботі яких виникає питання виявлення неповних дублікатів в електронних документах: дипломних та дисертаційних робіт, описаннях CAD проектів тощо. Може бути використана при викладанні дисциплін «Системи штучного інтелекту» та «Інтелектуальні системи та технології обробки даних».

УДК 004.912
ББК 73

ЗМІСТ

ВСТУП

Неповні дублікати або збіги – це фрагменти документів, які мають дубльований вміст даних визначеного типу (графічні дані, таблиці, числові дані, математичні формули тощо), але не є повністю дублікатами один одного. Під ідентифікацією або пошуком неповних дублікатів будемо розуміти одну з задач кластеризації, в якій визначаються документи, що є подібними в межах визначеного кластеру.

Перший розділ присвячено розробці та описанню моделей та методів аналізу графічних даних з метою виявлення в них подібностей або неповних дублікатів. Розглядаються методи аналізу зображень та їх фрагментів для визначення неповних дублікатів, а також прикладне програмне забезпечення для ідентифікації подібностей у графічних даних, діаграмах та математичних формулах.

У другому розділі наведені методи обробки та аналізу математичних формул. Розглянуті основні засоби побудови формул: спеціальні мови розмітки, редактори з графічним інтерфейсом, редактори для написання математичних та хімічних формул тощо. Розроблено метод порівняння математичних формул на основі шаблонів або зразків. У цьому методі формула відображається у вигляді тегу, який порівнюється з тегами інших формул, що зберігаються у відповідній базі.

У третьому розділі проведено аналіз сучасних графічних форматів та редакторів, які використовуються для створення растрових схематичних зображень у наукових текстах, розглянуто особливості топологічної структури таких графічних об’єктів, як схеми та діаграми. Підхід, що пропонується, передбачає проведення аналізу схем алгоритмів та діаграм за їх просторовою топологією, тобто за особливістю структури зв’язків між вузлами схеми або діаграми. Система автоматизованого аналізу просторової структури схем здійснює їх обробку, що передбачає реалізацію операцій фільтрації, аналізу структурних елементів схем, формування опису схем,

візуалізацію, сегментацію та кодування отриманих даних щодо об’єктів, які аналізуються.

У четвертому розділі наведені розроблені моделі та методи виявлення неповних дублікатів у таблицях. Розглянуті способи подання табличних даних та наведена постановка проблеми знаходження неповних дублікатів у таблицях. Описано модель для індексації даних таблиці та гібридний метод визначення неповних дублікатів у таблицях, який базується на методах локально-чутливого хешування та найближчого сусіда. Для виявлення подібності між послідовностями рядків використовується відстань Хеммінга. Для розрахунку відстаней між числовими даними використовуються відстань Евкліда, міська метрика та відстань Мінковського. У випадку, якщо контент електронного документа містить зображення або математичні формули, він досліджується окремо з використанням спеціальних методів. Розроблений метод призначено для знаходження неповних дублікатів у таблицях, що містять текстові та числові дані.

У п’ятому розділі описано підходи до візуалізації інформації про роботу системи визначення неповних дублікатів, наведено скріни з інтерфейсу відповідного додатка, а також принципи роботи програмних модулів системи визначення неповних дублікатів з урахуванням їх візуалізації. Також в розділі побудовано структурні схеми супроводу наукових робіт та розмежування прав доступу в системі перевірки ступеня унікальності наукових робіт.

РОЗДІЛ 1. МОДЕЛІ ТА МЕТОДИ АНАЛІЗУ ЗОБРАЖЕНЬ ТА ЇХ ФРАГМЕНТІВ ДЛЯ ВИЗНАЧЕННЯ ЗБІГІВ (НЕПОВНИХ ДУБЛІКАТІВ)

1.1. Проблема аналізу зображень та їх фрагментів для виявлення неповних дублікатів

Задача аналізу текстової інформації [1] та графічних даних [2–5] з метою виявлення подібностей та неповних дублікатів є актуальною для наукового співтовариства, фахових видань, спеціалізованих вчених рад, оскільки може бути цінним інструментом для уникнення зловживань та плагіату у сфері вищої освіти, сприяє академічній чесності. Аналіз зображень у електронних документах можна, наприклад, здійснювати з використанням стандартних програмних додатків та нестандартних додаткових модулей. У цьому випадку для покращення якості розпізнавання треба реалізувати процедури відновлення зашумлених зображень, аналізу форми та текстури, виділення фрагментів зображень, зіставлення двох зображень тощо.

Існує багато методів, які можуть бути ефективно застосовані для аналізу та розпізнавання графічних даних (рисунків, таблиць, формул) з метою ідентифікації збігів, неповних дублікатів або подібностей. У цілому, такі методи базуються на розпізнаванні ключових точок зображень, хешуванні зображень, застосуванні стохастичної геометрії, ланцюгів Маркова, перцептивного хешування тощо. Ці методи доволі успішно виконують задачу пошуку неповних дублікатів зображень, але мають деякі недоліки. Всі методи можна поділити на дві категорії: методи, які базуються на ключових точках, та методи, що базуються на аналізі пікселів зображення.

Перед застосуванням методів розпізнавання необхідно провести фільтрацію графічних даних, яка дозволяє здійснити обробку зображень таким чином, щоб їх було зручно аналізувати на подібність. Особливістю методів фільтрації є те, що вони дозволяють виділити локальні області зображення і розглянути їх особливі характеристики. Фільтрація передбачає використання деяких перетворень до всього зображення: бінаризації із

заданим пороговим значенням, фільтрації високих частот (фільтр Габора), фільтрації низьких частот (фільтр Гаусса), фільтрації Фур'є тощо.

Окремим видом фільтрації зображень є фільтрація ідентифікованих математичних функцій. Її принцип полягає у виявленні на зображенні елементарних математичних функцій: прямої, параболи, кола тощо. До фільтрів такого типу належать фільтри Хафа, Радона тощо.

У випадку складних зображень часто достатньо аналізувати не повне зображення, а тільки його контури. Основними методами фільтрації контурів є методи операторів Лапласа, Кенні, Робертса, Прюітта тощо.

Також з метою розпізнавання графічних даних, а саме: для класифікації та стиснення зображень використовується вейвлет-аналіз з набором класичних характеристичних функцій: вейвлети Морле і Хаара, вейвлет мексиканський капелюх тощо. Існують також особливі види фільтрів, що будуються на основі так званих риджлет-, курвлет- та бимлет-перетворень.

Оскільки реалізовані системи розпізнавання, як правило, націлені на пошук подібних графічних зображень, виникає необхідність у нових дослідженнях, націлених на виявлення неповних дублікатів у зображеннях та зображень, які є дублікатами, що були оброблені та модифіковані певним чином. Тема визначення подібностей та збігів у математичних формулах і таблицях висвітлена недостатньо, тому потребує подальших досліджень. Результати досліджень, що розглядають в даній монографії, можуть бути використані у вищих навчальних закладах для перевірки дипломних, курсових робіт та рефератів на плагіат, у спеціалізованих вчених радах для перевірки кандидатських та докторських дисертаційних робіт, у фахових виданнях для перевірки статей, монографій тощо.

При визначенні неповних дублікатів в електронних документах деякі складові контенту документів перетворюються в графічні елементи [6]. До таких складових належать графіки, формули, діаграми тощо. Тому методи визначення неповних дублікатів у зображеннях повинні бути адаптовані для знаходження графічних представлень вищезгаданих елементів. До методу

знаходження неповних дублікатів зображень можна висунути вимоги, які полягають у ідентифікації методом:

- точних відповідностей;
- обрізаних зображень;
- склеєних зображень (з кількох фрагментів);
- зображень зі зміненими розмірами;
- зображень зі зміненими кольорами;
- віддзеркалених та повернутих зображень.

Основна ідея пошуку – знайти неповні дублікати одного і того ж зображення, а не схожі на нього. Щодо інших вимог до способу визначення неповних дублікатів, точніше до результатів роботи цього способу – це досягнення максимально можливої повноти та мінімальної кількості помилок. Повнота визначається як відношення кількості знайдених неповних дублікатів до загальної кількість неповних дублікатів у базі даних.

Дублікатами називаються модифіковані копії одного і того ж зображення. Для знаходження дублікатів зображень можна використовувати методи пошуку схожих файлів, сигнатури MD5 для пошуку повністю ідентичних файлів або локально-чутливе хешування файлів (наприклад, метод ssdeep). Але ці способи недосконалі, оскільки навіть зміна формату файлу (наприклад, конвертація JPEG у PNG) повністю змінить його бінарну структуру. Тому методи пошуку, що базуються на обробленні зображень як бінарних файлів, – недоцільні, їх можна використовувати лише для знаходження повністю ідентичних файлів.

У випадку використання методу пошуку, заснованому на хешуванні файлів зображень за допомогою алгоритму MD5 (чи іншої подібної функції), навіть редагування EXIF-інформації (службова інформація про дату зйомки фото, модель фотоапарату тощо) призведе то того, що файли вважатимуться різними. Тому необхідно звернути увагу на підходи, що розглядають зображення як графічний об’єкт, а не бінарний файл, а саме: хешування значень кольору пікселів зображення, визначення ключових точок тощо.

Існує багато типів та форматів зображень, але всі вони можуть бути приведені до растрової графіки та збережені в одному з популярних форматів: PNG, TIFF, JPEG.

Серед підходів обробки графічної інформації можна виділити два основних напрямки: визначення ключових точок зображення та використання локально-чутливого хешування для пікселів зображення. Ці методи можуть бути поєднані і, в основному, дають задовільні результати під час пошуку подібностей в зображеннях.

Створення унікального способу визначення неповних дублікатів у зображеннях – робота, що потребує значних зусиль. Тому для побудови інформаційної системи визначення неповних дублікатів у контенті електронних документів вирішено обрати наявний спосіб визначення неповних дублікатів за допомогою перцептивного хешування та модифікувати його таким чином, щоб він відповідав вимогам системи.

Існують методи визначення неповних дублікатів у зображеннях, які дають результати з великою повнотою пошуку. Вони базуються на визначенні ключових точок зображення (рис. 1.1). Ці методи хоч і дають хороші результати у разі знаходження зображень, але мають один суттєвий недолік – вони знаходять всі схожі зображення, а не тільки дублікати.

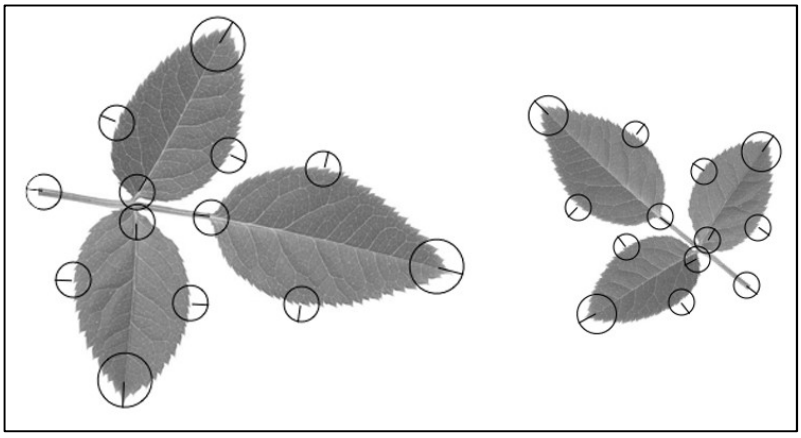
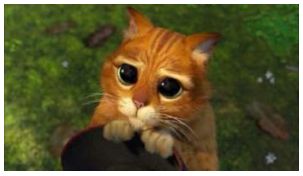
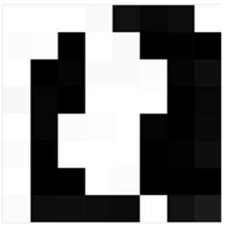


Рисунок 1.1 – Ключові точки на оригінальному
та модифікованому зображеннях

За допомогою ключових точок знаходиться максимальна кількість схожих зображень, але з’являється велика ймовірність хибних результатів. Маючи велику кількість помилкових результатів, ми перевантажуємо роботою експерта, який буде працювати з системою.

Розглянемо **метод індексації зображення на прикладі знаходження хешу за середнім значенням**. Цей тип індексації є одним з найпростіших, але підходить для пошуку однакових зображень там, де використання запозиченої графіки не намагаються приховати. Послідовність здійснення операцій та отримані результати подані у табл. 1.1. Перший крок – це зменшення роздільної здатності картинки, наприклад, до 8x8 пікселів, відповідно зміна пропорцій сторін до 1:1. Отримуємо прямокутник 64-ох точок. Далі зменшуємо кількість кольорів у зображенні, трансформувавши кольори у відтінки сірого.

Таблиця 1.1. Послідовність індексації зображення

№	Зображення на етапі	Коментарі
1		Вхідне зображення може бути будь-якого розміру
2		Зображення приведене до розміру 8x8 пікселів
3		Зображення переведене у відтінки сірого та визначений середній колір зображення
4		Результат порівняння кожного пікселя зображення з середнім кольором
5	00001110001001110 …	Визначення сигнатури

Будь-який сірий колір у моделі RGB має однакові значення всіх трьох кольорів (наприклад, RGB(75,75,75)), тому для розрахунків можна взяти лише одне число з трьох. Ми маємо зображення з 64 пікселів, кожен з яких має значення кольору від 0 до 255, і тепер можемо підрахувати середнє значення кольору. Для отримання середнього значення кольору числові значення кольорів пікселів додаються та діляться на загальну кількість пікселів. Наступна операція – переведення зображення у бітовий формат.

Знаючи середній колір зображення, проходимо за кожним пікселем зображення, порівнюючи його із середнім. Створюємо сигнатуру зображення: покроково, якщо піксель темніший середнього, додаємо до сигнатури одиницю, якщо світліший – нуль. Таким чином ми отримуємо рисунок в бітовому форматі (табл. 1.1, пункт 5), який можна також відобразити у вигляді двійкового числа. Отримане число можна перевести в будь-яку іншу систему числення.

Отримані сигнатури різних зображень можна порівнювати між собою за допомогою метрики Хеммінга, визначаючи відсоток подібності. Це дуже зручно, оскільки отримані сигнатури можна зберігати в базі, щоб не здійснювати ці операції кожного разу. Таким чином, індексація кожного зображення відбувається лише один раз. Крім того, елементи індексу тексту, що одержують за допомогою локально-чутливого хешування, також порівнюються за допомогою підрахування відстаней Хеммінга. Це дозволить використовувати уніфіковані засоби порівняння як для текстового індексу, так і для індексу зображень.

Всі оптимізації методу порівняння хешів текстового індексу будуть застосовуватися для індексів обох типів контенту. У результаті індексації таблиць також одержують бінарні рядки, які необхідно порівнювати за допомогою міри Хеммінга. Тому будуть розроблятися методи оптимізації підрахунку відстаней Хеммінга на великих наборах даних.

Використання перцептивного хешу. Для методу, що базується на перцептивному хешуванні, необхідно зменшувати зображення не до 8x8

пікселів, а як мінімум до 32x32. Замість приведення зображення до сірої гамми, використовується дискретне косинусне перетворення, таким чином виділяючи характерні особливості зображення. Двійкове значення сигнатури створюється так само, як і в попередньому методі. Цей метод дає теоретично позитивні результати, оскільки він визначає особливості зображення і у випадку хешування враховуються всі ці особливості (без поділу зображення на фрагменти), що теоретично повинно мінімізувати кількість помилкових результатів алгоритму.

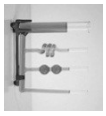
Дослідження можливостей Perceptual Hash. Для оцінки можливостей методу перцептивного хешування зображень створена колекція зображень, частина з яких були дублікатами, але з використанням різних модифікацій. Тестування здійснювалось за допомогою бібліотеки phashion. До дублікатів оригінального зображення застосовувалися такі модифікації:

- нелінійна зміна кольорів;
- накладання тексту;
- обрізка (5%, 20%, 50%);
- віддзеркалення;
- поворот(5°, 90°, 180°).

Також було взято два зображення, що не були дублікатами. У табл. 1.2 наведені результати роботи методу, тобто розрахована міра Хеммінга між початковим зображенням та модифікованим. За отриманими даними можна зробити висновки, що метод підрахування перцептивного хешу дає прийнятні результати лише на незначних модифікаціях. Були виявлені такі модифікації, як зміна кольорів, накладання тексту та незначна обрізка зображення (до 5% площі). З нахиленими або повернутими зображеннями алгоритм працює значно гірше.

Можна виявляти зображення, якщо вони нахилені не більш ніж на 5°, але й це вимагає збільшення порогового значення максимальної міри Хеммінга, за якою зображення вважаються дублікатами, що, в свою чергу, може призвести до помилкових результатів.

Таблиця 1.2. Результати роботи алгоритму Perceptual Hash з різними зображеннями

№	Зображення	Опис модифікацій	Відстань Хеммінга
1	2	3	4
1		Оригінальне зображення приведене у відтінки сірого	0
2		Нелінійна зміна кольорів зображення	2
3		Накладений поверх зображення текст	8
4		Незначна обрізка зображення (5% від низу зображення)	8
5		Обрізка приблизно 20% зображення	12
6		Обрізка 50% зображення	34
7		Віддзеркалене по горизонталі зображення	14
8		Поворот на 5°	14
9		Поворот на 90°	30
10		Поворот на 180°	16
11		Несхоже зображення, взяте для прикладу	34
12		Несхоже зображення, взяте для прикладу	32

Було вирішено взяти метод з використанням перцептивного хешу як базовий для інформаційної технології визначення неповних дублікатів. Оскільки сам метод не відповідає всім вимогам до ідентифікації неповних дублікатів у зображеннях, йому були додані певні модифікації.

Для подолання негативного ефекту обрізання або склеювання використовується метод визначення ключових точок зображення (алгоритм SURF). На відміну від наявних методів пошуку дублікатів за збігом ключових точок, пропонується використання ключових точок для поділу зображення на фрагменти. За отриманими ключовими точками зображення фрагментується таким чином, щоб окремі скупчення ключових точок потрапили на різні фрагменти. Наступні операції здійснюються як над початковим зображенням, так і над його копіями. Тобто кожен фрагмент вважається повноцінним зображенням у випадку збігу зображення з фрагментом іншого зображення.

Для знаходження неповних дублікатів у зображеннях, що були повернуті на значний кут, використовується метод визначення загального кута повороту. Він полягає у перетворенні початкового зображення на квадрат і створенні його копій з кутами повороту 5°, 10°, 15°, …, 180° та вибору одного з них, що відповідає певному критерію. Таким критерієм може бути середній колір одного з секторів зображення (приклад, зображений на рис. 1.2, крок повороту 30°).

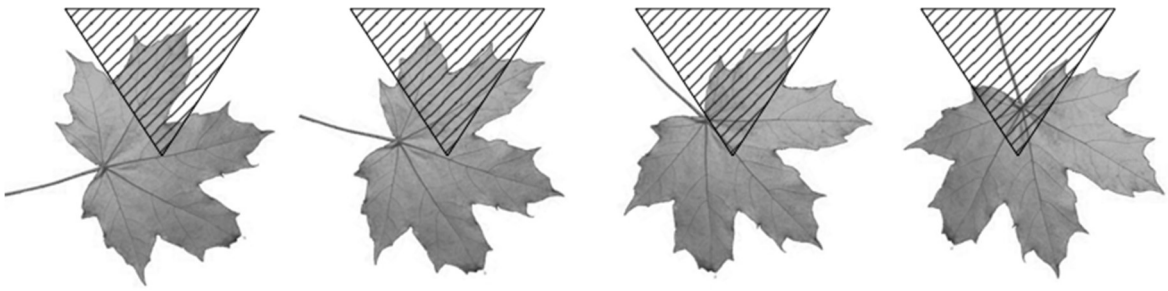


Рисунок 1.2 – Вибір кута повороту зображення

Колір одного пікселя зображення в кольоровій моделі RGB задається за допомогою трьох чисел від 0 до 255, наприклад, (255, 75, 13). Відтінки сірого задаються за допомогою 3-х однакових чисел, наприклад, (76, 76, 76). Взявши числове значення кольорів, можна вирахувати середнє значення кольору множини пікселів. Для кожного варіанта зображення вираховується середній колір пікселів, що потрапили до сектора. З усіх варіантів обирається таке повернуте зображення, для якого значення середнього кольору відповідає певному критерію, наприклад, мінімальному або максимальному значенню.

Для зображення та всіх його фрагментів визначається власний кут повороту. Після визначення кутів створюються віддзеркалені по горизонталі копії зображення та його фрагментів. Для кожного фрагмента вираховується перцептивний хеш. Таким чином, замість одного хешу зображення представлене множиною хешів. У випадку перевірки двох зображень перевіряється кожна пара хешів множин за допомогою міри Хеммінга і якщо відстань між будь-якими двома парами не перевищує порогового значення – ці зображення вважаються збігами.

Розглянемо основні поняття та сформулюємо проблему дослідження.

Загалом, можна виділити чотири типи дублікатів [7]:

а) точні дублікати – абсолютно однакові зображення, що не відрізняються жодним бітом. Приклад точних дублікатів картинок наведено на рис. 1.3;

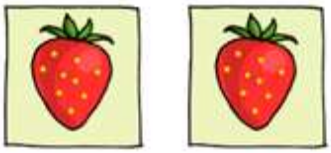


Рисунок 1.3 – Точні дублікати

б) тумбнейлерні дублікати (від англ. «thumbnail» – мініатюра) – зображення, що відрізняються тільки розміром, наприклад, репродукція на сайті картинної галереї та маленька картинка, яка на неї посилається.

Приклад тумбнейлерних дублікатів наведено на рис. 1.4;

Рисунок 1.4 – Тумбнейлерні дублікати

в) напівдублікати – картинки з напівпрозорими надписами зверху зображення, незначною корекцією кольору, обрізанням або рамкою. Приклад напівдублікатів наведено на рис. 1.5;

Рисунок 1.5 – Напівдублікати

г) розширені напівдублікати – картинки із дуже зміненими кольорами або пропорціями, а також фрагменти оригінальних зображень. Приклад таких зображень наведено на рис. 1.6.

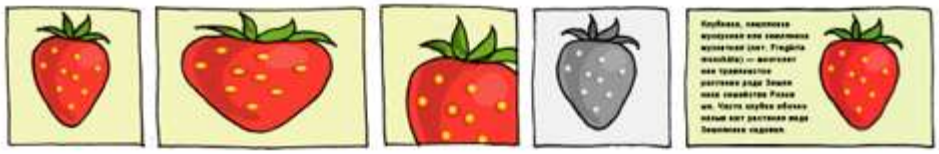


Рисунок 1.6 – Розширені напівдублікати

У цьому випадку повними дублікатами називатимемо точні дублікати з цієї класифікації, а неповними – всі інші. Проблема виявлення неповних дублікатів ускладнюється тим, що постає питання, на якому етапі модифікації зображення перестає бути дублікатом і стає самостійним зображенням. Приклади зображень-дублікатів та просто подібних зображень наведені на рис. 1.7 і 1.8 відповідно.

Адекватне оцінювання роботи алгоритму повинно здійснюватись за

допомогою експертів, а поняття про ступінь подібності є суб'єктивним. Будемо вважати, що зображення є неповнм дублікатом зображення, якщо на малюнку зображено такий самий об'єкт зі змінами, які під час поверхневого розглядання малюнку істотно не змінюють його сприйняття.

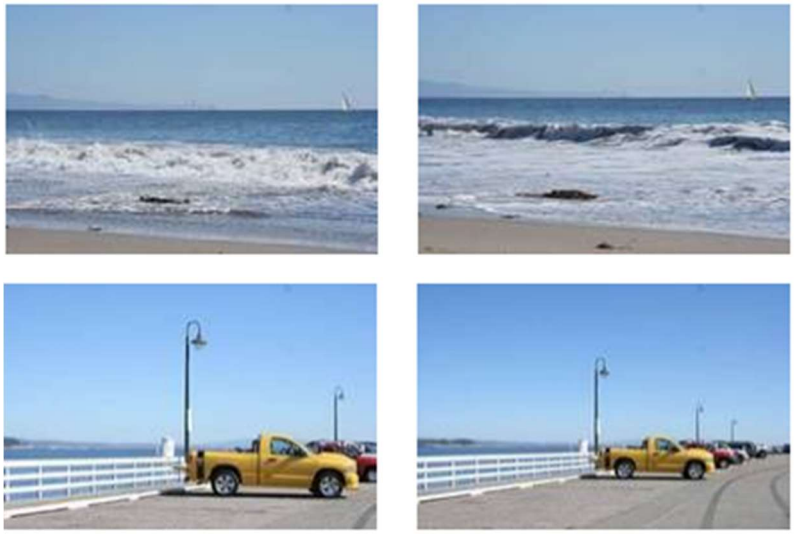


Рисунок 1.7 – Зображення-напівдублікати

Рисунок 1.8 – Візуально схожі зображення, які не є дублікатами один одного

Існує декілька напрямків пошуку за зображеннями: пошук за змістом (знайти фотографію людини або зображення, наприклад, берези), пошук за візуальним зразком (знайти зображення, подібні заданому), пошук за описом

(знайти зображення, помічене як «Цілком таємно» або «Київ») тощо. Кожен з напрямків пошуку має свої особливості та сфери використання [8].

У рамках цього дослідження розглядається пошук зображень за візуальним зразком. Він полягає у вилученні істотних властивостей зображень та побудові на їх основі сигнатур, що використовуватимуться для порівняння пар зображень. До кожної пари завжди належить зображення з колекції та зображення-зразок, подібне до якого бажає знайти користувач. Результатом порівняння є величина, яка називається візуальною подібністю (іноді тематичною близькістю) зображень. З точки зору інформаційного пошуку таке оцінювання, здійснене людиною-експертом, буде змістовною релевантністю, а розраховане в системі – формальною релевантністю.

Властивості оцінюються алгоритмами обчислення значень важливих чисельних величин і називаються ознаками. Подібних ознак можна виділити досить багато. У загальному випадку вони поділяються на дві групи: глобальні та локальні. До глобальних ознак належать основні кольори, текстури, форми, значущі елементи всього зображення. Локальні ознаки вираховуються для невеликих частин (блоків) зображення.

Для позначення значень ознак також вживають термін дескриптор. Набір ознак (іноді вживають поняття «вектор ознак»), що описує зображення для певної задачі, називається сигнатурою. Ознаки, відібрані з множини альтернативних для їх застосування в будь-якій задачі аналізу зображень і які ввійшли до сигнатури, називають ключовими. Під час пошуку зображень за зразком аналізуються не окремі екземпляри, а пари зображень, які зіставляються.

Тому на відміну від інших завдань аналізу в поточній проблемі від ознак зображень переходять до ознак пар, значення яких вираховують як абсолютні різниці значень відповідних ознак кожного з зображень пари. Така нескладна формула дозволяє моментально розрахувати їх для проіндексованих зображень і реалізувати на практиці пошук з прийнятними часовими характеристиками [9]. Ефективні методи пошуку в невеликих колекціях зображень з обмеженими ресурсами системи повинні максимально

лаконічно описувати кожне зображення і мати швидку процедуру порівняння пари зображень.

Метою дослідження в цьому розділі є вивчення ознак, за якими ідентифікується зображення, а також методів виявлення неповних дублікатів зображень з метою збільшення повноти і точності вибору зображення за зразком з колекції зображень.

Для досягнення поставленої мети необхідно вирішити такі задачі:

а) визначити поняття неповного/нечіткого дубліката зображення;

б) здійснити аналіз визначальних ознак зображень;

в) здійснити аналіз методів визначення неповних дублікатів у колекції зображень;

г) здійснити аналіз методів визначення неповних дублікатів за зразком;

д) запропонувати метод, що швидко і чітко працюватиме для колекції зображень.

1.2. Аналіз відомих наукових розробок та прикладного програмного забезпечення для пошуку неповних дублікатів зображень

Розглянемо основні класичні методи кластеризації колекції зображень, що можуть бути використані для пошуку неповних дублікатів у графічних даних електронних документів.

Детектор Харріса

Метод дозволяє визначати опорні точки зображень. Алгоритм заснований на вимірюванні інтенсивності яскравості зображення:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2, \qquad (1.1)$$

де $w(x,y)$ – функція вікна;

$I(x+u, y+v)$ – майбутня інтенсивність;

$I(x,y)$ – поточна інтенсивність.

Після апроксимації будується матриця часткових похідних, потім

розраховується її визначник і слід. Це дозволяє знайти кутові точки, що вважатимуться опорними:

$$R = \det M - k\left(trace\right)^2, \qquad (1.2)$$

де R – точка;

k – коефіцієнт k є(0.04,0.06);

M – матриця.

Значення R розраховується для кожної точки, після чого обираються тільки ті R, значення яких перевищує поріг. Наприкінці обираються локальні максимуми. Множина отриманих локальних максимумів і є вихідними даними алгоритма. Візуально метод можна проілюструвати, як показано на рис. 1.9 – 1.12.

Рисунок 1.9 – Вхідні зображення

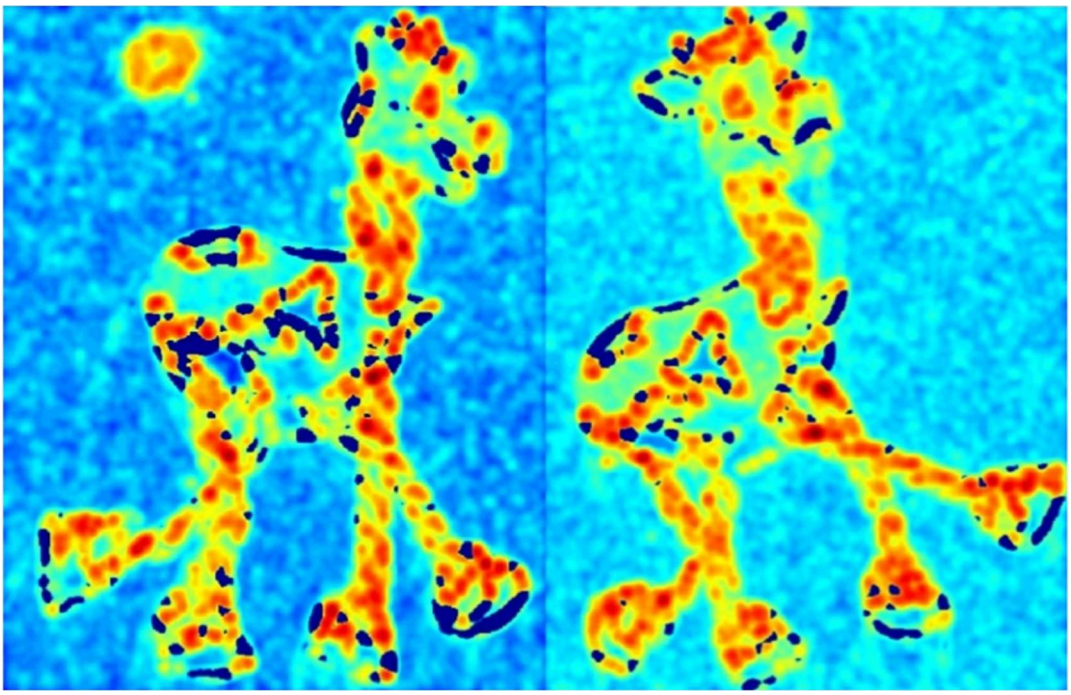


Рисунок 1.10 – Обчислення значення *R*

Рисунок 1.11 – Обчислення значення *R* > trashold

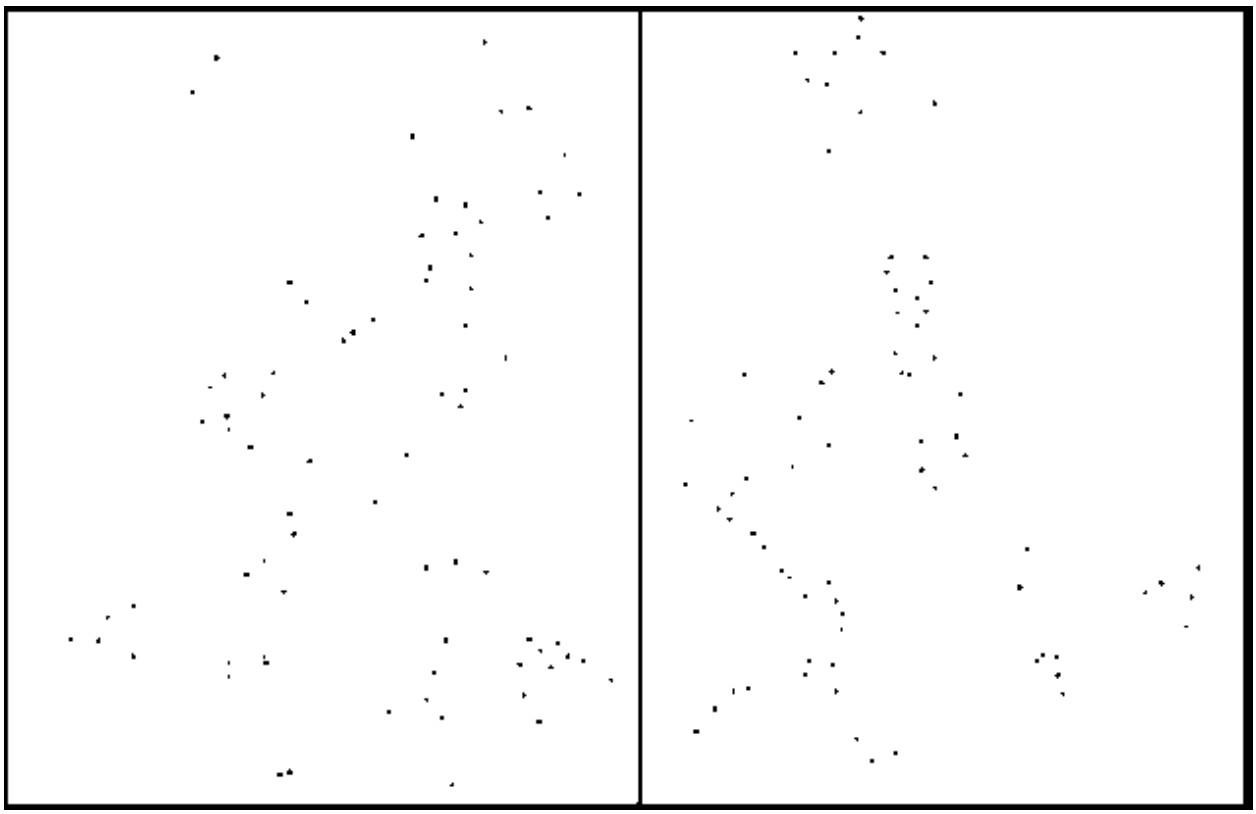


Рисунок 1.12 – Визначення точок локального максимуму *R*

Властивості методу: інваріантність до повороту, інваріантність до зміщення інтенсивності, не має інваріантності до зміни масштабу [10 – 14].

Алгоритми, вихідними даними яких є фрагменти, у більшості полягають у боротьбі з проблемою масштабу точкових алгоритмів. Окремим випадком є алгоритм Laplacian-of-Gaussian (LoG).

Laplasian-of-Gaussian (LoG)

Метод було розроблено для вирішення проблем визначення неповних дублікатів зображень у разі зміни масштабу зображення. Для цього:

- обирається функція, задана на фрагменті зображення, що є інваріантною до зміни масштабу (наприклад, середня інтенсивність);

- у кожній точці зображення ця функція розглядається як функція зміни розміру фрагмента;

- розраховується локальний максимум цієї функції. У цьому випадку точка максимуму інваріантна до зміни масштабу;

- розмір фрагмента, на якому досягається локальний максимум,

розраховається для кожного зображення окремо;

- відбувається стиснення точки разом з її оточенням. Знаходяться точки, що будуть мати максимальне значення на всіх масштабах.

Істинно особлива точка залишається такою на різних масштабах.

Поведінку такої функції у разі зміни масштабу наведено на рис. 1.13.

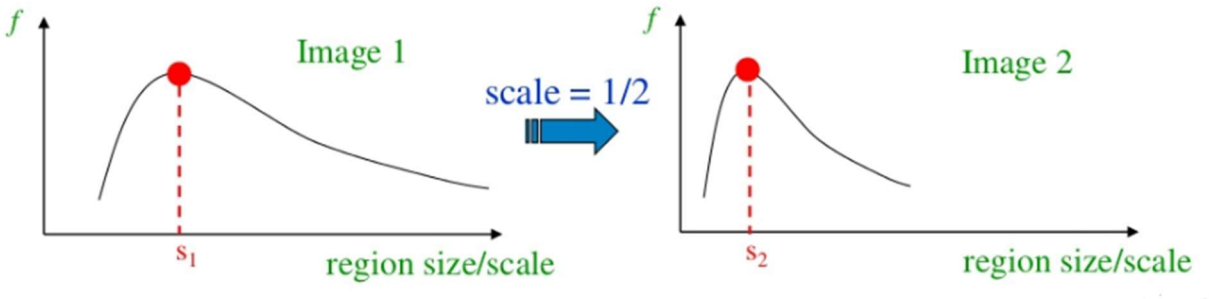


Рисунок 1.13 – Поведінка функції LoG у разі зміни масштабу

Таким чином, для кожної точки зображення її оточення «згортається» за формулою:

$$L(x,\sigma) = \sigma^2 (I_{xx}(x,\sigma) + I_{yy}(x,\sigma)), \tag{1.3}$$

де x,y – координати точки;

σ – масштаб.

Існують алгоритми, що дозволяють не тільки знайти однакові елементи зображень на різних масштабах, але й знайти, наприклад, елемент у повороті. Алгоритми, результатом роботи яких є дескриптори, дуже широко використовуються, тому що їх вихідні дані можуть бути подані на вхід стандартним алгоритмам кластеризації, таким як k-means, mean-shift тощо. Ці алгоритми описують фрагмент навколо точки за допомогою векторів або матриць ознак. Яскравим прикладом таких методів є SIFT-дескриптори.

SIFT-дескриптори

Технологія SIFT (масштабно-інваріантне перетворення ознак) є потужним засобом формування системи інваріантних структурних ознак [15]. Вона побудована на використанні кращих з сучасних базових принципів

локального оброблення, що містять у єдиному комплексі локальну фільтрацію, формування значущих ознак, аналіз простору перетворень, апроксимацію координат ознак тощо. Головний акцент у SIFT зроблений на стійкому визначенні крайових точок (КТ) зображення у плані відділення їх від інших стійких елементів (СЕ). У процесі багатоетапного оброблення шляхом аналізу відмінностей у просторі ознак встановлюється необхідна величина відстані між СЕ, що загалом призводить до високих характеристик розпізнавання. З цих причин перспективним є застосування SIFT у випадку розпізнавання зображень об'єктів в умовах просторових перешкод. Побудова модифікацій SIFT для цих умов робить новий крок у розвитку структурних методів розпізнавання.

Згідно з технологією SIFT [15] зображення $B(x,y)$ перетворюється на множину $L(x,y,k\sigma)=B(x,y)\otimes Q(x,y,k\sigma)$, кожний елемент якої одержують згорткою зображення з маскою гауссіана $Q(x,y,\sigma)$ (параметр σ). Величина σ змінюється дискретно в межах $[\sigma_0,\sigma_1]$.

Етап відбору стійких точок полягає у виборі найбільш стабільних КТ шляхом детального аналізу властивостей прилеглих даних у просторі $D(x,y,\sigma)$ на основі інтерполяції. Ставиться мета стабілізувати переміщення точок, викликані дискретними властивостями зображення і перетворень над ним. Для визначення більш точного положення КТ здійснюють розкладання $D(x,y,\sigma)$ у ряд Тейлора $D(c)=D+\frac{\partial D^T}{\partial c}c+\frac{1}{2}c^T\frac{\partial^2 D}{\partial c^2}c$, де $c=(x,y)$ – зсув від обраної КТ c^* у межах масштабу σ. Положення $\hat{c}$ екстремуму функції $D(c)$ за зсувом визначається з рівності нулю похідної. Якщо одна з компонент зсуву $\hat{c}$ перевищує 0,5, то координати нової КТ обчислюються з урахуванням поправки $c^*=c^*+\hat{c}$, і для неї процедура інтерполяції повторюється. Для забезпечення точності координати обчислюються у формі з рухомою комою.

Для усунення КТ з низькою контрастністю відкидаються ті, для яких член другого порядку не перевищує 0,03.

Усунення відгуків крайових точок здійснюється шляхом аналізу співвідношення між головними значеннями кривизни у напрямках для кожної виділеної КТ. Визначення кривизни зводиться до знаходження власних значень матриці $H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$ другого порядку аналогічно методу Харіса, де D_{uv} – похідна за змінними u, v. Власні значення α_1, α_2 матриці H у випадку фіксованого σ пропорційні величинам просторової кривизни. Відношення $r = \alpha_1 / \alpha_2$ власних значень ($\alpha_1 \geq \alpha_2$) є критерієм для виключення КТ зі списку. Відкидаються ті КТ, для яких одне з напрямків кривизни значно перевищує інше, що означає наявність «істотного» відгуку краю. Величини $\nu = tr^2(H) / det(H)$ та $\xi = (r+1)^2 / r$ є близькими. Значення ν є мінімальним, якщо α_1, α_2 приблизно рівні. Критерієм відхилення КТ є умова $\nu > \xi_0$, де ξ_0 – поріг для величини r_0. У практичному використанні $r_0 = 10$.

Кожній КТ призначається одна чи декілька орієнтацій. Цим досягається інваріантність до обернення. Обраховується амплітуда $m(x,y)$ та орієнтація $\theta(x,y)$ градієнта:

$$m(x,y) = \sqrt{[L(x+1,y) - L(x-1,y)]^2 + [L(x,y+1) - L(x,y-1)]^2},$$

$$\theta(x,y) = arctg \frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}. \qquad (1.4)$$

На основі відгуків значень і напрямків градієнта для точок у оточенні розміром 3σ (з центром у КТ) будується гістограма, яка містить 36 стовпчиків. Кожен стовпчик охоплює діапазон орієнтацій у 10 градусів. Значеннями гістограми є суми $v = \sum m_q$, що відповідають напряму з номером q. Для усунення викидів гістограма згладжується і нормується для забезпечення стійкості до змін яскравості, знаходиться напрямок з найбільшим відгуком. Піки гістограми відповідають панівним орієнтаціям

градієнта. Стовпці гістограми з відгуком, що перевищує 80% від максимуму, формують додаткові КТ за масштабом, які відрізняються лише напрямками. Ознакова інформація про об'єкт міститься у масиві напрямків $T = \{t_i\}$, $t_i = (x, y, m, \theta, \sigma)$. В оточенні 16×16 кожної точки за фіксованим масштабом аналізується множина з 16 фрагментів 4×4, що не перетинаються та утворюють розбиття. Для кожного з фрагментів будується гістограма напрямків, яка охоплює 360 градусів. Один напрямок відповідає діапазону у 40 градусів. Значеннями дескриптора є суми $v_i = \sum_{q=i} m_q$. Для забезпечення інваріантності до поворотів проводиться нормалізація шляхом зсуву щодо направлення КТ. У результаті для кожної КТ маємо дескриптор – вектор, що містить $16 \times 8 = 128$ значень, приклад якого наведено на рис. 1.14.

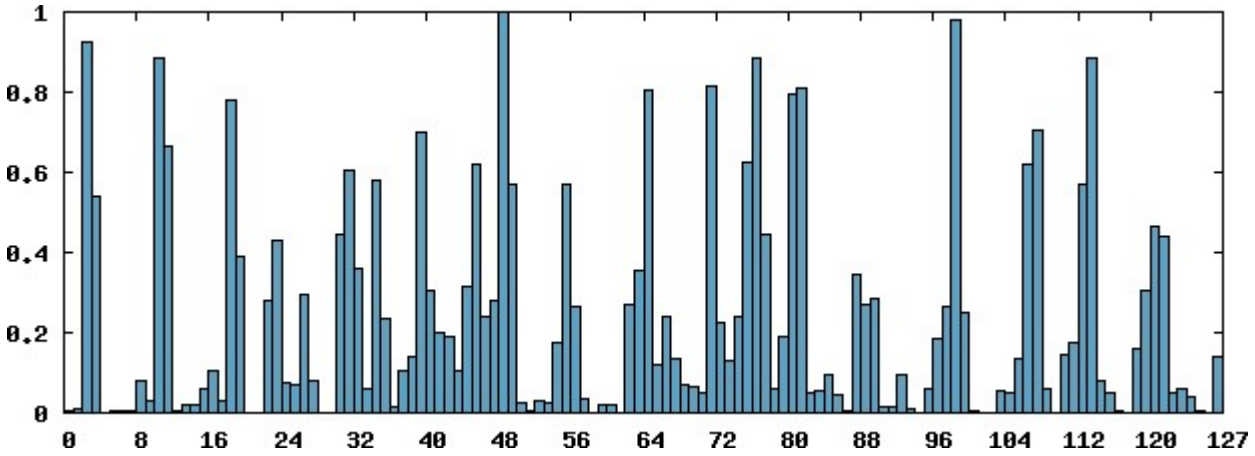


Рисунок 1.14 – Приклад дескриптора SIFT

Хеш-методи

Хеш-методи відрізняються своєю простотою. Як приклад розглянемо найпростіший алгоритм отримання хешу зображення [16].

Виходячи з визначення перетворення Фур'є, зображення – це двовимірний (залежність яскравості від горизонтальної та вертикальної координат) неперіодичний сигнал. Для RGB-зображення існує три таких сигнали: яскравість у каналах Red, Green и Blue. Оскільки в обробці сигналів і суміжних галузях перетворення Фур'є звичайно розглядається як

декомпозиція сигналу на частоти та амплітуди, поділимо умовно зображення на 3 частоти:

- на низькій частоті будуть міститися найбільші деталі, загальний розподіл яскравості та кольору, тобто форма об’єкта;

- на середній частоті знаходиться середня та дрібна деталізація, яка має назву «локальний контраст» і для знятих крупним планом об'єктів є фактурою поверхні;

- на високій частоті знаходиться наддрібна деталізація, про яку часто говорять «мікроконтраст» і яка відповідає за різкість.

Зрозуміло, що для порівняння зображень необхідно використовувати низькі частоти. Розглянемо алгоритм роботи методу з покроковими ілюстраціями. Вхідне зображення наведено на рис. 1.15.

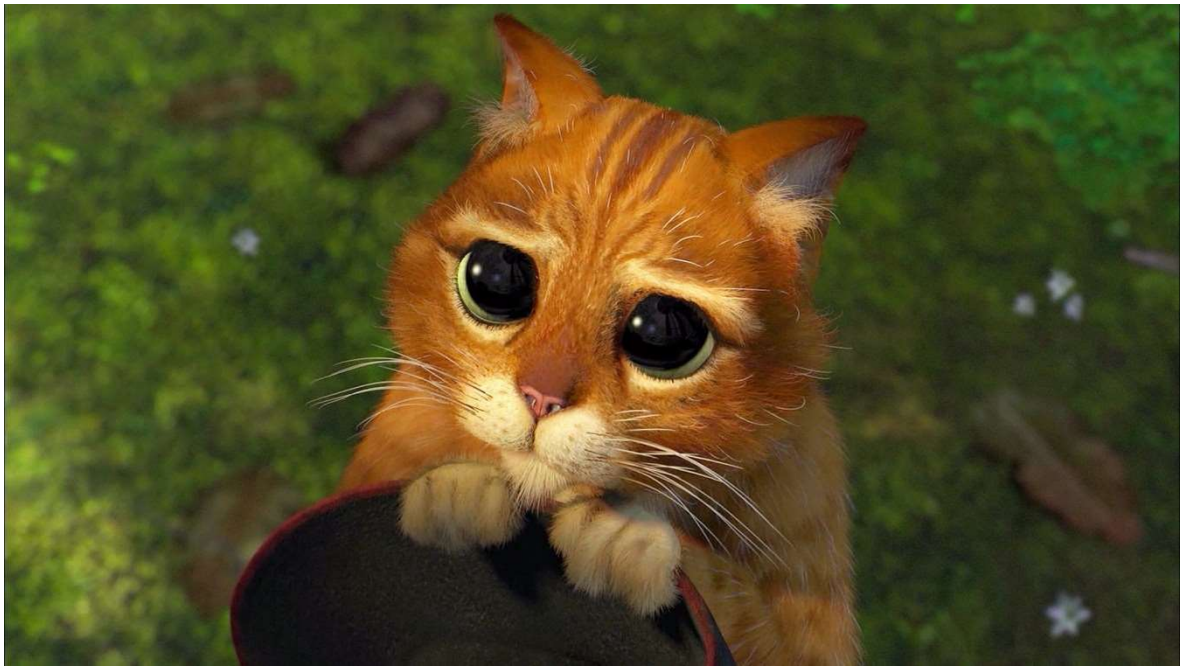


Рисунок 1.15 – Вхідне зображення

Крок 1. Зменшується розмір зображення. Найшвидший спосіб позбутися високих частот – зменшити зображення. У цьому випадку зображення зменшується до розподільної здатності 8x8, таким чином, загальне число пікселів становить 64. Можна не дбати про пропорції, зображення просто стискається в квадрат вісім на вісім. Отже, хеш буде

відповідати всім варіантам зображення, незалежно від розміру та співвідношення сторін. Ескіз наведено на рис. 1.16.

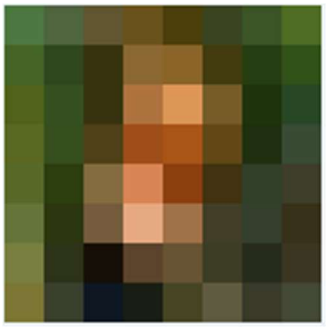


Рисунок 1.16 – Зменшене зображення

Крок 2. Прибирається колір зображення. Маленьке зображення переводиться в градації сірого так, що хеш зменшується втричі: з 64 пікселів (64 значення червоного, 64 зеленого і 64 синього) всього до 64 значень кольору. Знебарвлене зображення наведено на рис 1.17.

Рисунок 1.17 – Знебарвлене зображення

Крок 3. Далі для кожного з кадрів обчислюється середнє значення пікселів.

Крок 4. Формується ланцюжок бітів. Кожен піксель порівнюється із середнім значенням і, якщо він більше середнього значення, то до клітинки хешу записується 1, в іншому випадку – 0. Внаслідок цього отримуємо зображення, наведені на рис 1.18.

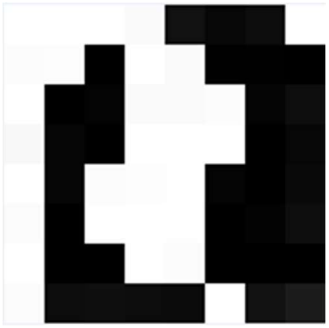


Рисунок 1.18 – Підсумкове зображення

Крок 5. Будування хешу. 64 окремих біта переводяться до єдиного 64-бітового значення. Порядок несуттєвий, якщо він зберігається незмінним.

Підсумковий хеш не зміниться, якщо картинку масштабувати, стиснути або розтягнути. Зміна яскравості або контрасту, або навіть маніпуляції з кольорами теж сильно не вплинуть на результат. І найголовніше: такий алгоритм відрізняється високою швидкістю.

Для порівняння пари зображень обчислюється відстань Хеммінга (підраховується кількість різних бітів). Нульова відстань означає, що це, швидше за все, однакові картинки (або варіації одного зображення). Дистанція 5 означає, що картинки чимось відрізняються одна від одної, але в цілому все одно досить схожі одна на одну. Якщо дистанція 10 або більше, то це, ймовірно, зовсім різні зображення.

1.3. Методи аналізу та дослідження ознак зображень та їх фрагментів для ідентифікації неповних дублікатів

Весь спектр методів аналізу зображень в інтелектуальних системах комп’ютерного зору (СКЗ) традиційно поділяється, відповідно до етапів оброблення зображення, на три групи [9, 10]: попереднє оброблення (фільтрація, сегментація, виділення контурів, характерних точок тощо); побудова описів (обчислення векторів або множин ознак, формування структурних елементів та моделей); інтерпретація описів (прийняття рішення про відповідність сформованих в процесі аналізу моделей та еталонних

описів, оцінка параметрів об’єктів). Важливу роль у цьому випадку відіграє те, як людина сприймає зображення й образи на них.

У людського зору існує інше, так зване синтетичне сприйняття, коли спостережуваний об'єкт сприймається одразу як цілісний образ. У випадку синтетичного сприйняття впізнавання відбувається рефлекторно, без докладного аналізу всього інформаційного поля зображення, часто лише за окремими фрагментами об’єкта. Наприклад, фахівцями з окремих деталей на основі візуального огляду впевнено розпізнаються численні марки автомобілів. За частковими фрагментами людський зір відновлює весь цілісний образ об’єкта, спираючись на апріорно задану модель. Здатність подумки добудовувати образ за уривчастими елементами – фундаментальна властивість зорового «апарата» людини. У біологічному сприйнятті, доповнюючи одна одну, ефективно використовуються обидві моделі зображення як повна (аналітична), коли аналізується вся множина точок зображення, так і часткова (синтетична). Наявні ж штучні побудови процесів сприйняття в СКЗ спираються, головним чином, на аналітичну модель, яка універсально підходить для всієї різноманітності візуальних об’єктів, представлених функцією яскравості в деякому полі зору.

Відмінні риси інтелектуальних СКЗ [11, 12]: вміння виділяти суттєву інформацію з наявної множини ознак; здатність до навчання та узагальнення знань з метою застосування їх у нових ситуаціях; відновлення та аналіз подій за неповної інформації про них; здатність визначати цілі та формулювати плани дій системи для їх досягнення. Ухвалення рішення за неповної інформації – одна із основних характеристик інтелектуальних систем.

Під час дослідження визначальні ознаки зображень були поділені на декілька груп. Першою групою ознак були гістограми. Для їх побудови використовувалися дві різні палітри: HSV і RGB. Гістограми для RGB-палітри будувалися таким чином: колірний простір розбивався на 32 рівні частини і розраховувалось, скільки пікселів зображення потрапляє до кожної з частин. Для HSV-палітри гістограми будувалися окремо за кожною з

компонент. Незважаючи на те, що ознаки, отримані з різних гістограм, засновані на колірних характеристиках окремих пікселів, вони непогано доповнювали одна одну, тому що HSV- і RGB-палітри пов'язані нелінійним перетворенням і відповідають за різні властивості об'єктів.

До другої групи належать ознаки, засновані на підрахунку граничних пікселів. Для обчислення цих ознак зображення переводилися в чорно-білий формат, а потім знаходились різкі яскравісні переходи. Після цього із зображення вгорі і внизу виділялися смужки висотою до 20% від загальної, і для цих смуг розраховувалась кількість «граничних» пікселів. Було зазначено, що в одному кластері бічні краї зображення можуть досить сильно відрізнятися (наприклад, невеликий поворот камери), але нижній і верхній край, зазвичай, залишаються без змін [13].

Питання використання колірних характеристик обговорюються в [14]. Виділяються такі ознаки, як загальний колір, найбільш частотні кольори, регулярність розміщення кольорів, складність образів і тощо. Робота [15] присвячена використанню текстурних ознак. Автори розрізняють ознаки контрастності, грубості, спрямованості, лінійних образів, регулярності та шорсткості текстур.

У [16] розглядаються різні ознаки форми: ознаки Фур'є (FD), ознаки кривизни простору (CSSD), ознаки Цернік-моментів (ZMD) і сіткові ознаки (GD). Аналізуються слабкі та сильні сторони зазначених ознак зображень.

У роботі [17] для пошуку зображень за вмістом пропонується використовувати ознаку однорідності текстури (Homogeneous Texture Descriptor, HTD), ознаку граничної гістограми (Edge Histogram Descriptor, EHD), масштабовану ознаку кольору (Scalable Color Descriptor, SCD), ознаку панівного кольору (Dominant Color Descriptor, DCD).

У роботі [18] запропонований алгоритм пошуку зображень на основі нечітких колірних гістограм, що обчислюються в просторі CIE L * a * b *. Пропонується, крім головної, використовувати набір додаткових гістограм, побудованих для певних ділянок зображення.

У роботі [19] розглядаються можливості колірної моделі HIS (Hue, Saturation, Intensity) під час пошуку зображень за вмістом. HSI – це циліндричне представлення колірної моделі RGB, що складається з таких компонент: колірний тон, насиченість та інтенсивність. Автор також пропонує використовувати модель HLD (Hue Layout Descriptor).

У [15] використовуються зменшені копії зображень для пошуку подібних. В основу роботи покладений той факт, що людське сприйняття дозволяє з достатньою точністю зрозуміти вміст зображень з малим розрішенням. Для кольорових зображень людині достатньо роздільної здатності 32x32, щоб досягти точності розпізнавання, яка перевищує 80%. У роботі [19] пропонується використовувати колірні ознаки. Для цього будується колірна гістограма, що враховує просторове розташування кольорів, та обчислюються колірні моменти. Для оцінки подібності двох зображень порівнюються параметри розподілу їх кольорів: математичне очікування за кожним з колірних каналів та попарні коваріації розподілів каналів.

У роботі [20] використовуються ознаки колірних автокорелограм, текстурна ознака Tamura, гістограми орієнтацій градієнтів.

Таким чином, існує досить велика кількість ознак, що могли б використовуватися в задачах пошуку зображень за візуальним зразком. Проте, більшість робіт містить в собі лише частину наявних ознак зображень, комбінуючи їх та розробляючи на їх основі сигнатури. У зв’язку з тим, що багато ознак можуть бути взаємопов’язані між собою, важливою задачею є їх раціональний відбір.

1.4. Побудова та дослідження модифікацій методів пошуку неповних дублікатів графічних даних

Існує багато способів, що беруть за основу різні ознаки зображень, користуються цією інформацією різними способами, додаючи різні коефіцієнти. Нижче розглянуті найбільш відомі методи, що дозволяють якомога точніше розв’язати поставлене завдання.

TinyImages

Спочатку генерується зменшена копія зображення роздільною здатністю 32x32 пікселя. Кожному пікселю приписується колір у просторі RGB.

Попередній пошук дублікатів здійснюється за кількома сигнатурами (векторами розмірності 64 і 128), остаточна перевірка проводиться попіксельним порівнянням. На першому кроці, незалежно від пропорцій, зображення стискається до розміру 20x20 пікселів і приводитися до сірого. Як перша сигнатура використовується центральна частина зображення, розміром 16x16 пікселів. Як друга сигнатура використовуються дескриптори особливих точок, координатами яких є екстремуми перетворення Difference of Gaussian (DoG) [8].

Вводиться емпіричний поріг, нижче якого сигнатури вважаються «близькими». Використовуючи «близькі» сигнатури, ми одержуємо групи кандидатів у дублікати. Аналізуючи координати «близьких» сигнатур на парі зображень, виділяється область перетину зображень. Для неї обчислюється попіксельна різниця і різниця карт градієнтів. І якщо достатній відсоток цих різниць менше порога, то пара зображень вважається дублікатами.

Ставимо у відповідність кожному зображенню кількість сигнатур інших зображень, близьких до сигнатур цієї картинки. Картинки упорядковано відповідно до зменшення лічильника. Для кожного невідзначеного зображення знаходимо близькі за сигнатурами. Для знайденого зображення та зображення запиту здійснюється пошук (за сигнатурами) загальної області. Порівнюється область перетину зображень, завдяки цьому робиться висновок про близькість зображень. Якщо поріг пройдено, то зображення вважаються напівдублікатамі і вони позначаються.

Після оброблення всіх знайдених зображень пошукове зображення також позначається. Процес триває до тих пір,поки всі зображення не будуть помічені. Для уточнення результату (і збільшення кластерів напівдублікатів) цей етап повторюється декілька разів з викиданням підклеєних картинок

(зі збереженням прив’язки) і зниженням порогів схожості. Після цього для кожного зображення визначаємо найбільш близьке до нього – «центр» кластера.

Метод «пошуку за зразком»

Для розв’язання завдання пошуку зображень у колекції за зразком може бути запропонований також алгоритм, в основі якого лежить представлення зображення у вигляді нечіткої колірної гістограми. Метод розрахунку нечіткої колірної гістограми заснований на методі, запропонованому в [12], і містить в собі такі кроки:

Крок 1. Приведення початкового зображення до розміру, що не перевершує 100 × 100 точок. Одночасно відбувається збереження пропорцій початкового зображення і бікубічна інтерполяція кольорів.

Крок 2. Приведення отриманого зображення в універсальний кольоровий простір CIE L*a*b*.

Крок 3. Обчислення значень функцій приналежності у кожній точці зображення.

Крок 4. Побудова гістограми значень функції приналежності.

В дослідженні, дотримуючись оригінального методу, розглядаються окремі функції приналежності для кожного колірного каналу.

Відмінною рисою запропонованого у дослідженні методу є те, що гістограми будуються без використання процедури дефаззіфікації. У кожній точці зображення будується вектор 75-мірного простору, компоненти якого є середнім арифметичним значень трьох функцій приналежності. Після цього відбувається усереднення цих векторів. Відстань між подібними векторами, що описують зображення, обчислюється на основі функції перетину гістограм.

На відміну від базового методу, в рамках якого для зображення обчислюється єдина гістограма, у цьому методі запропоновано обчислювати шість додаткових гістограм, що будуються для таких ділянок зображення:

- права половина зображення;
- ліва половина зображення;

- верхня половина зображення;
- нижня половина зображення;
- центральна частина зображення (велика);
- центральна частина зображення (маленька).

Модифікований хеш-метод

Для підвищення точності хеш-процедури, взятої за основу у розробленому методі, додамо вагові характеристики для чорно-білого зображення 8 × 8. У цьому випадку будемо вважати, що ступінь «значущості» точки на зменшеному зображенні збільшується залежно від кількості сусідів іншого кольору. Таким чином, коефіцієнт змінюється від 0 до 8.

Наведемо короткий опис розробленого комбінованого методу:

Крок 1. На етапі первинного оброблення вхідне зображення спочатку зменшується до здатності 10 × 10, після чого крайові значення обрізаються (до розмірності 8 × 8 із загальним числом пікселів 64). Такий спосіб не тільки дозволяє оразу позбутися всіляких «рамок» на зображенні, але і виділяє його основну частину. Таким чином, хеш буде відповідати всім варіантам зображення, незалежно від розміру та співвідношення сторін.

Крок 2. Забирається колір зображення. Маленьке зображення переводиться в градації сірого, тому хеш зменшується втричі: з 64 пікселів (64 значення червоного, 64 зеленого і 64 синього) всього до 64 значень.

Крок 3. Здійснюється приведення зображення до чорно-білого. Для кожного з кадрів обчислюється середнє значення пікселів, а потім кожен піксель порівнюється із середнім значенням (якщо він більше середнього значення, то в клітинку хешу записується 1, в іншому випадку – 0).

Крок 4. Здійснюється побудова хешу: 64 окремих біта переводяться в одне 64-бітове значення (порядок не має значення, якщо він зберігається постійним). Підсумковий хеш не зміниться, якщо зображення промасштабувати, стиснути або розтягнути. Зміна яскравості або контрасту,

а також маніпуляції з кольорами теж істотно не вплинуть на підсумковий результат.

Крок 5. Здійснюється порівняння аналізованого зображення з базовим. У цьому випадку необхідно використовувати зважену відстань Хеммінга, оскільки кожному бітові відповідає свій коефіцієнт. Для порівняння пари зображень обчислюється відстань Хеммінга (підраховується кількість різних бітів) з урахуванням ваг. У табл. 1.4 наведені співвідношення між класичною (D) і модифікованою (M) відстанями Хемминга для різних значень вагових коефіцієнтів W. У разі порівняння аналізованого і базового зображень нульова відстань означає, що це однакові зображення (або варіації одного зображення). У випадку від 0 до 5 зображення в цілому досить близькі один до одного (неповні дублікати). Якщо ж відстань становить від 6 до 9 зображення характеризуються окремими загальними ознаками, але дублікатами не є. Якщо відстань більше 9, то зображення вважаються різними.

Таблиця 1.4 – Корекція відстані між зображеннями з урахуванням ваг

W	0	1	2	3	4	5	6
D	1	1	1	1	1	1	1
M	0,8	0,85	0,9	0,95	1	1,05	1,1

Таким чином, запропонований метод дозволяє швидко і з прийнятною точністю знаходити неповні дублікати зображень. Він заснований на застосуванні комбінації хеш-методу і точних методів виявлення неповних дублікатів, пов’язаних з вибором визначальних точок на стиснутому й обробленому зображеннях. Проведено тестування запропонованого методу, яке підтвердило його працездатність: для презентативної вибірки зображень отримані прийнятні значення характеристик (повноти, точності і F-міри).

РОЗДІЛ 2. МЕТОДИ ОБРОБКИ Й АНАЛІЗУ МАТЕМАТИЧНИХ ФОРМУЛ З МЕТОЮ ВИЯВЛЕННЯ ПОДІБНОСТЕЙ

2.1. Огляд засобів побудови математичних формул

Майже всі наукові тексти, зокрема дисертаційні роботи та наукові статті, містять математичні формули та зображення. Для перевірки наукових текстових документів на наявність подібностей з публікаціями, що можна знайти в загальнодоступних мережевих джерелах, зазвичай використовують так звані системи «Антиплагіат», що реалізують технологію моніторингу та перевірки через стандартні пошукові алгоритми, які забезпечують достатньо швидкий та ефективний пошук повних або неповних дублікатів, а також гарантують коректне оброблення текстів [21–29]. Втім такі системи не дозволяють здійснювати перевірку зображень та математичних формул. У цьому випадку виникають проблеми, що неможливо вирішити за допомогою стандартних процедур. Додаткові специфічні проблеми виникають у процесі аналізу математичних формул в порівнюваних текстах з метою виявлення в них подібностей. Для вирішення цих проблем необхідно виробити нестандартні підходи.

Характерною рисою математичної інформації є використання складної та високорозвиненої двовимірної символьної системи позначень. Складність аналізу і розпізнавання математичних формул полягає в тому, що для знаходження неповних дублікатів необхідно аналізувати не просто графічне зображення, здійснюючи фільтрацію, виділення контурів і застосовуючи специфічні методи порівняння, а й текстову інтерпретацію формули, щоб ідентифікувати неповні дублікати за умови, що у формулі було змінено позначення літер, символи математичних операцій, форми дужок тощо [30]. Тому для знаходження неповних дублікатів математичних формул доцільно використовувати гібридні методи їх аналізу та порівняння.

У тексті формули можна наводити як малюнок або як графічний об’єкт, створений за допомогою одного з редакторів формул. Редактор формул –

комп’ютерна програма, призначена для створення та редагування математичних та інших формул. Редактори формул засновані на таких технологіях:

– застосування спеціальної мови розмітки, наприклад TeX, MathML у редакторі LaTex, Math для редактора OpenOffice;

– створення формул за допомогою графічного інтерфейсу: KFormula, MathType, MathCastmula, WIRIS Editor, MathCast (у цих редакторах формула будується зі складових, що надаються програмою);

– вбудовані компоненти: Math Expression Editor Light;

– символьні обчислення: Mathematica.

Спеціальні мови розмітки

Найбільш поширеними спеціальними мовами розмітки є LaTex та Math для OpenOffice.

LaTex – це система підготовки текстів, придатна, зокрема, для підготовки наукових публікацій, які містять математичні формули. Вона може використовуватись також для багатьох інших видів документів. LaTex побудована на базі TEX’a [31].

Мова розмітки TEX є досить поширеною мовою комп’ютерного верстання, розробленою Д. Кнутом [32]. До сих пір деякі відомі видання приймають до публікації технічні та математичні статті, що підготовлені саме у форматі TEX. Перевагою представлення формул у такому форматі є можливість повного управління зовнішнім виглядом будь-якої формули. Це є особливо важливим для запису складних математичних виразів, що містять матриці, системи рівнянь або нерівностей різних типів тощо. У цьому разі зображення формул у тексті є естетично привабливим. Мова розмітки текста TEX дещо близька до HTML та його узагальненню XML.

У форматі TEX формули записуються рядками. Одночасно використовуються макрокоманди мови, що починаються символом «\», за якими формується послідовність інших символів. Наприклад, грецька літера α записується як \alpha. Такий спосіб дозволяє записувати як прості, так і

складні символи та математичні вирази. Для роботи з формулами на мові TEX використовуються спеціальні фільтри, що контролюють ознаки початку та кінця формули. Приклади та правила запису формул на мові TEX можна знайти, наприклад, в [33].

Ще одним засобом запису формул у текстах є використання алгебраїчного синтаксиса за правилами TEX, але у цьому випадку формули на мові розмітки відокремлюються символами @@ з використанням текстового фільтра, що дозволяє розпізнавати ці обмежувальні символи. Переваги та недоліки такого представлення є такими ж, як і у разі використання мови TEX.

Різновидами мови TEX є мови розмітки типу LaTex або MikTEX [34]. Математичні вирази обробляються LaTex'ом в особливому режимі роботи, що називається математичною модою. Загальний для TEX'а та LaTex'а контрольний символ переходу в цю моду є знак долара $. У наукових статтях, дисертаціях, монографіях тощо одиничні математичні символи або короткі, прості формули, що розташовані безпосередньо в тексті, відокремлюються одинарними знаками $...$, формули ж, які друкуються окремим рядком, – подвійними знаками – $$...$$. Зазначимо, що у мові Math для редактора OpenOffice використовуються подібні позначки форматування, які дещо відрізняються від LaTex, але значної різниці між ними немає.

Наприклад, у Math для створення формули:

$$\Delta G = \Delta G^0 + RT\, ln \frac{P_M^m P_N^n}{P_A^a P_B^b}$$

необхідно ввести текст:

%DELTA G= %DELTA G^0 + RT ln {P^m_M P^n_N} over {P^a_A P^b_B}.

Для позначення змінних використовуються букви латини, грецької мови (з використанням повного написання літер, «\alpha», «\beta») тощо. Для позначення дій користуються спеціальними позначками (наприклад, «^», «_»), математичними операторами (наприклад, «+», «–», «=», «\neq» (≠),

«\sum» (∑) тощо), стандартними математичними функціями (наприклад, «\sin», «\min», «\ln» тощо), спеціальними словами-операторами, якими визначаються дроби, біноміальні коефіцієнти, радикали (наприклад, «\sqrt», «\frac {чисельник} {знаменник}» тощо). Зазначимо, що для записування математичних формул на web-сторінках останнім часом використовується також Math-система ASCIIMathML.

Загальний принцип використання MathML полягає в тому, що математичні конструкції вбудовуються до звичайного HTML-документа та адекватно відображуються після його завантаження з мережі. Перше, що відрізняє мову розмітки MathML від аналогів, – це використання двох способів кодування виразів. Один з них пов'язаний з безпосередньою передачею синтаксису формули (presentation), другий, навпаки, відображає семантику виразу (*content*).

Презентаційна розмітка описує математичну символіку з виразами, що будуються з використанням деяких схем виведення та засобів розміщення таких фрагментів виразів, як дроби, верхні та нижні індекси. Семантична розмітка описує математичні об'єкти та функції, де для кожного вузла будується дерево виразу згідно з деякою конкретною схемою, а гілки цього дерева віповідають підвиразам.

Оскільки питання розміщення текстів, що містять математичні формули та вирази, на web-сторінках є важливим, останнім часом набули поширення практичні дослідження цієї проблеми. Зокрема, для цього використовуються Java-аплети, що візуалізують математичні формули в HTML-сторінках та дозволяють переглядати та редагувати їх у вікні браузера (апплет – прикладна програма на мові програмування Java у формі байт-коду, що виконується у веб-браузері з використанням віртуальної Java-машини). Наприклад, опис апплетів WebEQ Viewer Control та Input Control, розроблених фірмою Design Science, Inc для візуалізації формул, наведено в [35]. Крім того, для роботи з web-текстами, що містять формули, може використовуватись редактор формул DragMath Equation Editor. Після виклику цього редактора здійснюється запуск

Java-машини та з’являється вікно побудови формул, що нагадує відповідне вікно редактора типу Microsoft Equation. У цьому випадку правила розмітки формул в редакторах DragMath Equation Editor та Microsoft Equation є практично однаковими.

Таким чином, порівняння формул, створених у редакторах розглянутого типу, не викликає труднощів. За формулою створюється шаблон, що складається зі спеціальних символів, службових слів, позначень функцій. Під час порівняння формул порівнюються їх шаблони, і якщо вони збігаються, постає питання про можливу тотожність формул. За аналогією аналізу текстів, під час аналізу формул, створених за допомогою спеціальних мов розмітки, треба відкидати так звані «стоп-слова». У мовах математичної розмітки такими словами можуть бути символи форматування, пробілів тощо.

Редактори з графічним інтерфейсом

Для створення математичних текстів широко використовується редактор формул Equation editor, що має специфічні можливості для побудови формул і належить до комплекту Microsoft Office. Цей редактор формул – скорочена версія програми «MathType» від фірми Design Science. Створені за допомогою цього редактора формули можна використовувати в додатках корпорації Microsoft, насамперед в текстовому редакторі Word, а також у програмі презентацій PowerPoint та табличному процесорі Excel. В цьому редакторі формул можна створювати складні математичні формули, використовуючи символи та шаблони панелі інструментів (рис. 2.1).

Панель містить понад 150 математичних символів та 120 шаблонів дробів, сум, границь тощо. Шаблони можна вкладати один в одний для створення багатоступеневих формул. Формули у цьому випадку мають виключно ілюстративний характер, фактично вони є стилізованими малюнками, через що за ними не можна здійснювати обчислення. Формули в процесі створення оформлюються згідно з правилами запису математичних виразів, що прийняті в редакторі, але їх стилі та розмір можна змінювати за

бажанням користувача. Формула, створена в Microsoft Equation, є «об'єктом», що займає в документі прямокутну область, і може бути разташована зверху або всередині тексту.

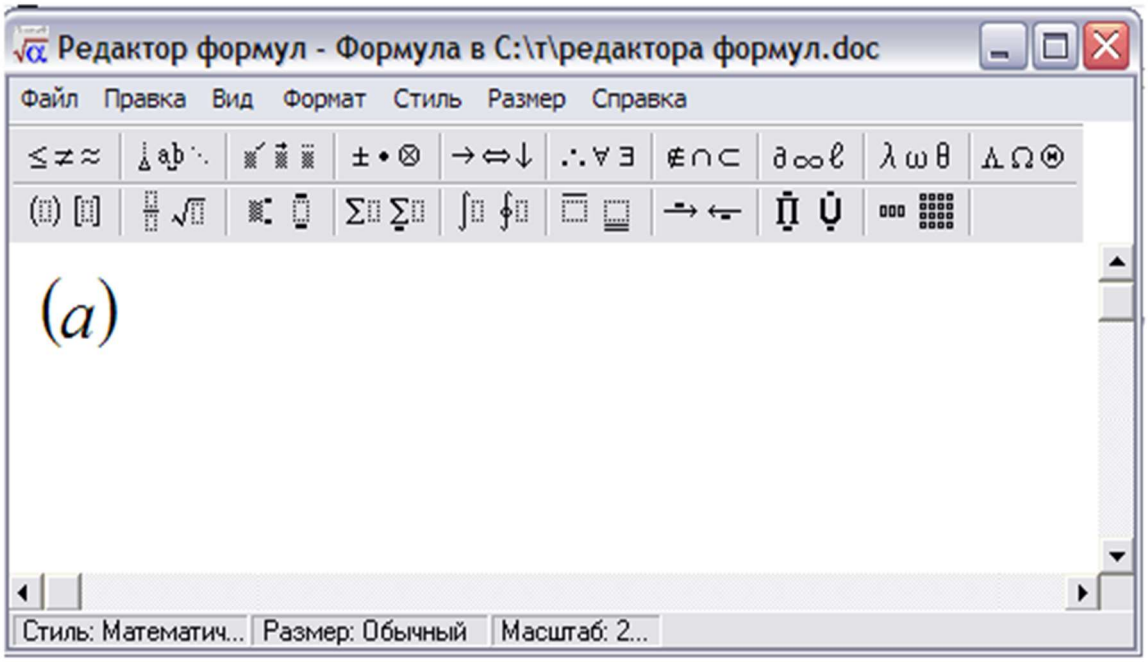


Рисунок 2.1 – Інтерфейс редактора MS Equation Editor 3.0

Назви деяких математичних функцій (наприклад, «sin», «cos») автоматично розпізнаються та пишуться прямим шрифтом.

MathType – це професійна версія редактора формул, що працює з Word, Corel WordPerfect та багатьма іншими додатками, а також значно покращує можливості Equation Editor. MathType більш інтегрований до текстового процесору. Цей редактор працює з формулами як з частиною тексту, а не картинками, вставленими в текст, завдяки чому зникає багато проблем.

Інтерфейс користувача програми MathType, як і інтерфейси більшості інших додатків Windows, є інтуїтивним та орієнтованим на візуальне сприйняття. Для кожного математичного поняття в MathType є шаблон, що містить символи та пробіли для заповнення. Всього налічується біля 175 шаблонів, зокрема дроби, радикали, суми, інтеграли, матриці та різні види дужок. Редактор MathType має інтерфейс, наведений на рис. 2.2.

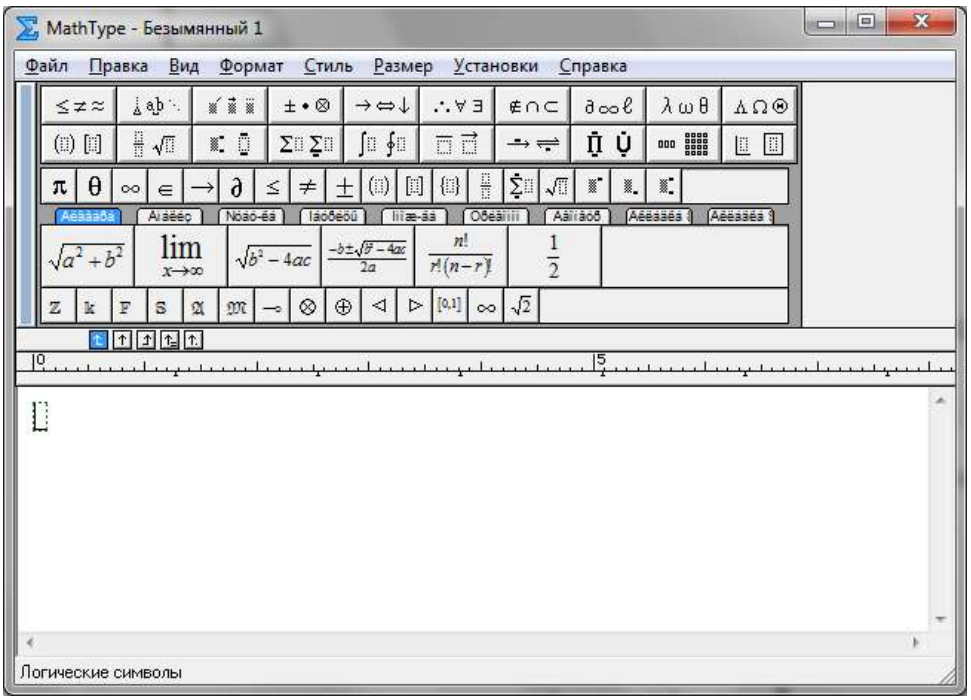


Рисунок 2.2 – Інтерфейс редактора MathType

Редактор формул у MS Word 2010 є однією з версій MathType та має такий вигляд (рис. 2.3).

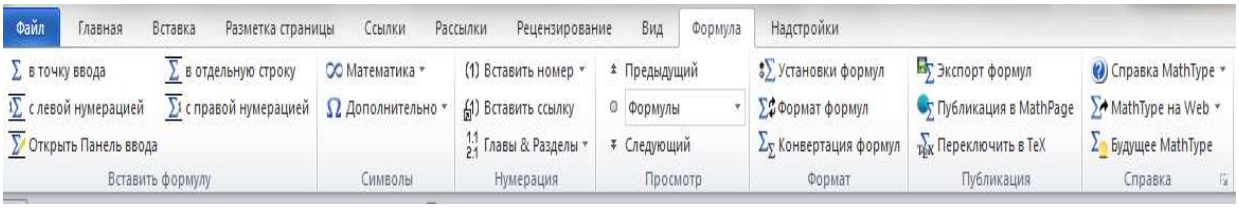


Рисунок 2.3 – Інтерфейс редактора формул у MS Word 2010

У MS Word завжди є можливість перетворити формулу, створену за допомогою MathType, на рисунок формату bmp. У цьому випадку порівняння формул має здійснюватись як порівняння малюнків, тим більше, що формат формул-малюнків буде співпадати за розміром (у випадку однакових формул).

Редактори написання хімічних формул

Крім звичайних математичних формул, у деяких наукових публікаціях можуть бути наявні хімічні формули. Для роботи з такими документами

інколи неможливо використовувати лише текстові або графічні редактори загального призначення. Редагування текстів, що містять хімічні терміни та формули, потребує використання спеціалізованих програмних інструментів.

Для написання хімічних формул авторами найчастіше використовується редактор LibreOffice Math. Хімічні формули зазвичай будуються у прямому зображенні. У разі використання Writer хімічні формули обробляються як об’єкти текстового документа, що полегшує їх подальше порівняння в наукових текстах.

Розробниками Microsoft Research та Центру Unilever було представлено нове розширення для MS Office: редактор Chem4Word, що дозволяє додавати та редагувати хімічні формули в документах, створених за допомогою Word, Excel та PowerPoint. Програмне забезпечення цього редактора хімічних формул, що підтримує версії MS Word 2007 та MS Word 2010, розповсюджується безкоштовно. Редактор Chem4Word дозволяє використовувати специфічні символи, позначки та вставляти до документів, які редагуються, хімічні формули та зображення.

В табл. 2.1 наведено приклад розмічення формул у редакторі LibreOffice Math.

Таблиця 2.1. Приклад розмічення хімічних формул у редакторі LibreOffice Math

Об’єкти	Формула	Розмічення
Молекули	H2SO4	H_2 SO_4 (пропуск між елементами обов’язковий)
Ізотопи	$^{238}_{92}\mathrm{U}$	U lsub 92 lsup 238
Іони	$\mathrm{SO_4^{2-}}$	SO_4^{2-{}} або SO_4^{2"-"}

Редагування текстів, що містять складну хімічну інформацію, а також створення та збереження хімічних структурних формул та схем хімічних

реакцій інколи здійснюються з використанням спеціалізованого хімічного редактора Symyx Draw [36]. Документи цього редактора можуть входити до складу документа Word або розміщуватися в окремих графічних файлах. В останньому випадку створені в Symyx Draw хімічні формули можна зберігати в одному з поширених графічних форматів (bmp, gif тощо).

2.2. Метод визначення ідентичних формул з різними літерними позначеннями на основі шаблонів

Порівняння формул з використанням шаблонів

Пошук за формулами є значно складнішим, ніж пошук за текстом (наприклад, формули x2 й a2 за шаблонами є ідентичними, а за йменуваннями змінних вони різняться).

Якщо формули збережені як об'єкт MathType, доцільно застосовувати порівняння за шаблоном.

Наприклад, потрібно порівняти формули:

$$W_1 \xleftarrow{F_{(w_1,L_1)}^{\downarrow}} L_1 \xleftarrow{F_{(w_2,L_1)}^{\downarrow}} W_2 \quad ma \quad Q_1 \xleftarrow{F_{(q1,P_1)}^{\downarrow}} P_1 \xleftarrow{F_{(q_2,P_1)}^{\downarrow}} Q_2$$

Віповідний шаблон для їх порівняння наведено на рис. 2.4.

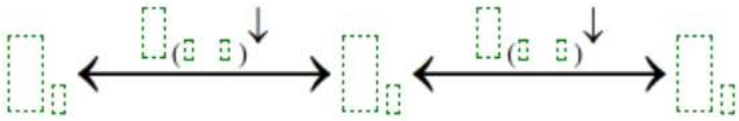


Рисунок 2.4 – Шаблон для порівняння формул

Таким чином, з'явиться можливість спочатку порівнювати шаблони, а потім – найменування змінних.

Треба зауважити, що деякі математичні та фізичні величини мають стійкі загальновживані позначення, тому слід зважати на контекст, у якому використовується формула. Зазвичай пояснення до формул подаються в абзацах до або після формули. Тому для різних галузей треба створити

словники з означенням величин та їх позначень. Наприклад, сила струму позначається як I, ймовірність – P тощо.

Стандартами оформлення документації (Державним стандартом України 3008–95, міждержавним стандартом 2.105–95 та іншими стандартами) встановлені правила оформлення формул та пояснень до них:

– позначення символів та числових коефіцієнтів, що входять у формули, має бути подано безпосередньо під формулою;

– кожне позначення подано з нового рядка без абзацу, дотримуючись послідовності, з якою вони розміщені у формулі;

– перший рядок розшифрування має починатися словом «де» без двокрапки після нього. Після формули (перед словом «де») ставиться кома.

Порівняння формул з використанням конверторів

Перспективним напрямком автоматизованого аналізу математичних формул є створення конверторів з різних форматів (TeX, Equation, MathType) до канонічного формату XML/MathML [37]. Такі конвертори мають формуватися у вигляді онлайнових сервісів, що є доступними як для користувачів, так і для систем, в яких використовуються математичні формули.

Традиційно для створення математичних текстів активно використовується редактор MS Word, що має специфічні можливості для побудови формул. Він є більш звичним для користувачів, ніж класична система роботи з математичними текстами (La)Tex, оскільки не потребує інсталяції додаткового програмного забезпечення. Разом з тим, слід зазначити, що поліграфічна якість математичних формул у MS Word є гіршою, ніж у системі TeX.

Ще однією проблемою є закритість форматів Microsoft. Цим зумовлена обмеженість використання word-форматів для публікації інформації у мережі Інтернет – доводиться конвертувати текст до html, причому зазвичай MS Word під час такого конвертування використовує застарілий метод переведення формул до растрової графіки (наприклад, до gif-формату).

Наявність різних стандартів представлення математичних та природничо-наукових текстів додає труднощів до вирішення проблеми порівняння математичних формул у документах. Використання декількох стандартів в дисертаційних роботах, періодичних виданнях, матеріалах конференцій, наукових монографіях призводить до того, що інформаційне середовище заповнюється документами, які представлені у важкосумісних форматах. Це створює погіршення цілісності інформаційного середовища та виникнення проблем з використанням текстів у різних представленнях.

Перспективним підходом до роботи з природничо-науковими текстами є розробка та застосування спеціальних діалектів (словників) XML. Умовним стандартом побудови математичних формул у Web-середовищі та розміщення у ньому математичних текстів вважається мова математичних символів MathML (Mathematical Markup Language) [38]. Ця мова є підмножиною розширеної мови розмітки XML (eXtensible Markup Language), що часто використовується для створення інших мов. Таке застосування XML є цілком природним і добре працює тоді, коли використання HTML для передачі даних нових типів обмежене узгодженням особливостей їх форматів.

До найбільш поширених мов розмітки, які основані на XML, належать:

– Wireless Markup Language (WML): формат даних для бездротових пристроїв, що працюють з протоколом WAP (мобільні телефони);

– Synchronized Multimedia Integration Language (SMIL): задає часову розмітку, зовнішній вигляд тощо для мультимедійних презентацій, визначає послідовність відображення мультимедійних файлів;

– Scalable Vector Graphics (SVG): використовується для описування двовимірної векторної графіки;

– Chemical Markup Language (CML): використовується для описування хімічних формул.

Разом зі специфікаціями, пов’язаними із задачею стандартизації структури текстів (наприклад, DocBook), MathML здатен сформувати

канонічний формат представлення природничо-наукових текстів, що використовує розмітку логічного рівня та дозволяє здійснення трансляції до будь-якого класичного або нестандартного формату, орієнтованого на зовнішнє представлення документів. Втім, слід зазначити, що мова MathML створювалася з метою її використання для побудови комунікативного ланцюжка та автоматизації роботи сервісів. Через це вона не орієнтована на безпосереднє користування внаслідок її складності.

Для генерування MathML-кодів необхідно використовувати WYSIWYG-редактори або конвертори з інших систем побудови математичних текстів. Зокрема, доцільно розробити програму-конвертор формул з документів Microsoft Word, що створені з використанням редактора формул Microsoft Equation Editor 3.0 та MathType 5.0 до формату MathML з оптимізацією для перегляду у різних Інтернет-браузерах. Реалізація такої програми може бути здійснена на мовах VBA або Java та базуватися на витяганні структур формул з RTF-файлів.

Застосування Java дозволяє здійснювати трансляцію формул незалежно від конкретного редактора MS Word. Наприклад, можна застосовувати редактор формул Equation Editor як WYSIWYG-редактор, що генерує MathML-код, а також дозволяє транслювати до стандартного формату раніше створені математичні тексти. Така технологія трансляції математичних формул на основі обробляння RTF-файлів, реалізована як онлайновий сервіс, може бути застосована для трансляції формул з документів Microsoft Office до різних форматів, в тому числі і до TeX.

Завдання, що були поставлені робочою групою W3C з математики під час створення MathML:

– забезпечення кодування матеріалів математичного характера для комунікацій усіх рівней освітнього та наукого типу;

– забезпечення кодування як математичної символіки, так і її значень;

– підтримка створення шаблонів та інших інструментів математичного редагування;

– забезпечення перетворень до інших математичних форматів як суто презентаційного, так і семантичного характеру.

Підтримку конвертування математичних формул з розміткою LaTex до представлення HTML може бути здійснено за допомогою програмного конвертора Pandoc. Для введення математики до HTML можуть бути використані спеціальні перетворювачі на основі MathML, Java-Script, онлайн-сервисів, код яких вноситься до сконвертованого HTML-файлу.

Приклад з результатами такого конвертування наведено на рис. 2.5.

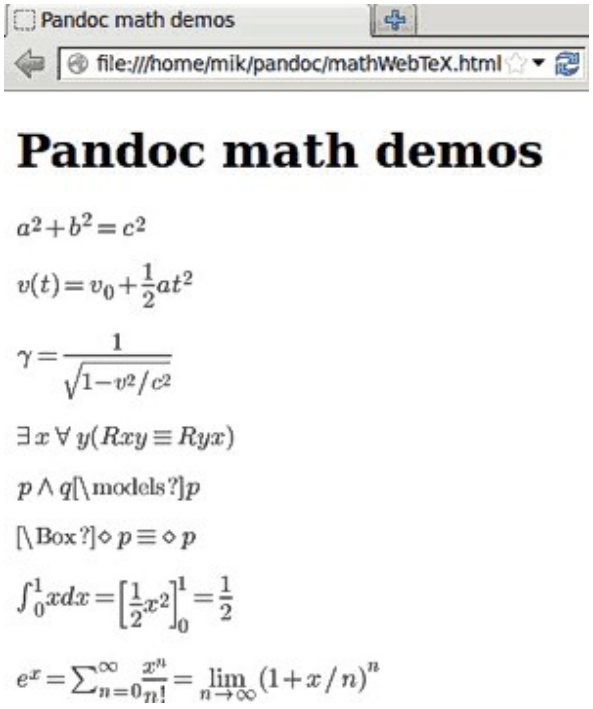


Рисунок 2.5 – Приклад конвертування формул з використанням програми Pandoc

Для конвертування математичних формул з розміткою LaTeX можно використовувати такі опції програми Pandoc:

- mathml – перетворює формули LaTeX до розмітки MathML;

- webtex – перетворює формули LaTeX за допомогою онлайн-сервісу Google Chart API;

- mathjax – перетворює формули LaTeX за допомогою розширення MathJax для MediaWiki;

- latexmathml – перетворює формули LaTeX за допомогою JS-бібліотеки Latexmathml.

Гібридна схема порівняння формул

Запропонована гібридна схема порівняння формул здійснює перевірку оригінальності формульного об’єкта, що аналізується, як за джерелами мережі Інтернет, так і за власними базами текстів з формулами (статей, дисертацій, курсових, дипломних та магістерських робіт тощо).

Джерела, що містять формули, можуть зберігатися як у вигляді повних текстів, необхідних для оцінки характеру подібностей (за результатами перевірки), так і у вигляді спеціального пошукового індексу, необхідного для швидкої перевірки збігу текстів та бази джерел.

Припустимо, що згідно з однією з наявних систем пошуку текстів (без урахування можливої близькості наявних там формул), що є близькими (за сучасними критеріями близькості текстів або їх фрагментів), була сформована обмежена множина (А) об’єктів з високим ступенем подібності до текстового об’єкта з формулами, оригінальність якого аналізується.

Розглянемо можливі варіанти представлення текстових об’єктів з формулами, що належать до множини А:

а) формули в текстах збережені як об’єкт MathType (варіант 1);

б) об’єкти збережені у вигляді тексту з графічними зображеннями формул (наприклад, у форматах GIF або JPEG) (варіант 2);

в) об’єкти, що містять формули, збережені повністю у вигляді документів у форматі PDF (варіант 3).

У першому випадку (варіант 1) доцільно застосовувати порівняння за шаблоном.

Алгоритм порівняння полягає в такому:

– формуються шаблони формульних об’єктів, що аналізуються (шаблони X);

– поступово формуються шаблони формул з текстових об’єктів з формулами, що належать до множини A (шаблони Y);

– здійснюється перевірка повних збігів структури шаблонів X з шаблонами Y (процедура такої перевірки є тривіальною) та формується клас шаблонів Z, що відповідають випадкам повного збігу за результатами перевірки;

– для шаблонів X та Y з однаковою структурою (тобто шаблонів з класу Z) здійснюється перевірка тотожності найменування змінних;

– у разі наявності однієї або кількох формул у множині Z, що мають повний збіг (як за шаблонами, так і за відповідними найменуваннями змінних) з формульними об’єктами, які аналізуються, здійснюється перевірка наявності посилань у текстових фрагментах;

– у разі відсутності посилань на первинне джерело, для випадків повного збігу (як за шаблонами, так і за відповідними найменуваннями змінних), фрагменти (сторінки) з однаковими формулами (в первинному джерелі та джерелі, що перевіряється на плагіат) автоматично заносяться до електронної форми «Індикація можливості формульного плагіату», що створюється для кожного аналізованого об’єкта.

Для варіанта 1 передбачається також можливість виявлення часткового дублювання формул.

Зауважимо, що редактор MathType містить транслятор формул та математичних виразів до MathML формату.

За необхідності аналізу та порівняння об’єктів, що збережені у вигляді тексту з графічними зображеннями формул (варіант 2), може бути застосована технологія, яка реалізована у програмі Infty [39]. Ця програма дозволяє звести завдання варіанту 2 до традиційного підходу роботи з відсканованими документами. Вона використовує технологію оптичного розпізнавання символів OCR (Optical character recognition) і застосовує

структурний аналіз до отриманого результату. У той час як ця технологія розпізнає складні математичні формули в документі, вона ігнорує іншу текстову інформацію. Використання програми Infty при розпізнаванні математичних формул із растрових графічних зображень дозволяє їх представляти мовою MathML та здійснювати в разі необхідності перетворення їх на текст.

У роботі [40] наведено результати досліджень щодо автоматизації процесу декомпозиції формул, записаних мовою MathML, з подальшою програмною реалізацією перетворення їх на текстовий опис. Пошук формул у тексті відбувається на основі удосконаленого методу трансформації синтаксичного дерева та методу пошуку за ключовими словами. Розроблений авторами [40] модуль знаходить математичні формули у документі відповідно до маски пошуку. Математичні формули в документі HTML/XHTML починаються із тегу $і закінчуються тегом$. У результаті всі теги, які розміщені між цими двома, виділяються. Отже, математична формула, яка записана мовою математичної розмітки MathML, відповідно до методу трансформації синтаксичного дерева і розроблених правил, перетворюється на текстовий опис і записується у текст документа.

Модуль перетворення формул на текстовий опис забезпечує пошук у тексті математичних формул та спеціальних символів відповідно до розроблених правил. Для перетворення MathML на текст використовується модель трансформації синтаксичного дерева [41].

Аналіз документів, у яких зустрічаються математичні формули у форматі PDF (варіант 3), як і раніше є складним та неостаточно вирішеним завданням. На теперішній час розпізнавання складних математичних формул із PDF-зображень можливе лише зі значними неточностями передавання змісту формули. Програма Infty може бути використана для аналізу як графічної, так і текстової інформації, що міститься в документі PDF. Для цього використовується розроблений авторами [40] дворівневий аналізатор

контенту і структури математичних формул з подальшим їх записом мовою MathML.

Зазначимо, що різноплановість представлення текстових об’єктів з формулами, що належать до множини A (варіанти 1, 2 та 3), підтверджує доцільність розробки гібридного методу, який забезпечує конвертування формул різних форматів (MathType, TeX, LaTex, JPEG, PDF тощо) до уніфікованого формату мови математичної розмітки MathML.

У загальному випадку, аналіз подібностей та виявлення повного або часткового дублювання формул може бути здійснений за такою схемою:

– сканування формули з відповідного інформаційного джерела (друкована стаття, електронний документ, зображення тощо);

– розпізнавання відсканованої формули та подання її мовою MathML. Розпізнавання формули або графічного зображення відбувається засобами програми Infty (серед засобів перетворення документа TeX на мову MathML найбільш якісною вважається програма TtM, розроблена А. Хатчінсоном (I. Hutchinson) [42]. Вона перетворює документ TeX у файл формату XHTML, до складу якого належать MathML формули. У результаті формується файл із розширенням *.mml, в якому записана формула;

– аналіз файлів формул, що порівнюються, з метою виявлення їх повного або часткового дублювання;

– запис результату аналізу до електронної форми «Індикація можливості формульного плагіату», що створюється для кожного аналізованого об’єкта.

Згідно з цією схемою можна виділити такі варіанти її реалізації на етапі перетворення математичних формул у різних форматах на мову математичної розмітки MathML (залежно від форматів представлення формул, що аналізуються) [43]:

1. **.doc → «GrindEQ Math» → TeX → Teacode LaTex → MathML*: перетворення за допомогою плагіну GrindEQ Math Utilities та онлайн-сервісу

Teacode LaTex на мову MathML набору формул, записаних у форматі текстового документа Microsoft Word;

2. *MathType → MathML*: перетворення на мову MathML набору формул, записаних засобами MathType;

3. **.html → виділення графічних об'єктів → JPG, PNG, GIF → Infty → MathML*: перетворення набору формул, записаних у вигляді графічних об'єктів веб-сторінки *.html, на мову MathML засобами програми Infty;

4. **.pdf → Infty → MathML*: перетворення на мову MathML набору формул, записаних у форматі Adobe Reader *.pdf, за допомогою програми для розпізнавання символів Infty.

5. *LaTex → Pandoc → MathML*: перетворення на мову MathML набору формул, записаних засобами LaTex.

Примітка: для варіанту 1 може виявитися достатнім застосування порівняння за шаблоном та відповідним алгоритмом, наведеним вище. За необхідності уточнення аналізу може бути і в цьому варіанті додатково застосовано схему з перетворенням на мову MathML формул, записаних засобами MathType.

Результатом реалізації запропонованої гібридної схеми може бути формування електронної форми «Індикація можливості формульного плагіату», що створюється для кожного аналізованого об'єкта, з визначенням загальних коефіцієнів подібності формульних документів, які перевіряються, та побудовою порівняльної гістограми.

Таким чином, аналіз наявних засобів побудови формул за допомогою сучасних редакторів та методів порівняння формул у наукових текстах дозволив розробити підхід до виявлення повних або часткових формульних дублікатів у публікаціях, курсових, дипломних проектах або магістерських роботах тощо.

Запропонована гібридна схема порівняння формул здійснює перевірку оригінальності формульного об'єкта, що аналізується, як за джерелами

мережі Інтернет, так і за власними базами текстів з формулами. Схема передбачає можливість аналізу подібностей та виявлення повного або часткового дублювання формул як з використанням шаблонів формульних об’єктів, що аналізуються, так і з застосуванням конвертації математичних формул у різних форматах на універсальну мову математичної розмітки MathML.

Одержані результати можуть бути використані для розширення функціональних можливостей наявних систем виявлення плагіату в наукових текстах, що містять формульні об’єкти [44].

РОЗДІЛ 3. АВТОМАТИЗОВАНИЙ АНАЛІЗ ПОДІБНОСТЕЙ СХЕМ ТА ДІАГРАМ В ЕЛЕКТРОННИХ ТЕКСТОВИХ ДОКУМЕНТАХ

3.1. Засоби побудови схем та діаграм

Існують ефективні засоби перевірки наукових текстових документів на можливу наявність подібностей з документами, які вже знаходяться в електронних бібліотеках або в ресурсах глобальної мережі Інтернет [24 – 29]. В той же час системи пошуку повних чи неповних дублікатів зазвичай не мають можливості аналізувати подібності графічних об’єктів, які є частиною порівнюваних документів (статей, доповідей, дипломних проектів, магістерських або дисертаційних робіт, звітів і ін.). До таких об’єктів належать усі ілюстративні елементи: рисунки, схеми, графіки, діаграми, фото, графічні зображення тощо [22].

Поширеним різновидом графічних елементів сучасних наукових текстів, що тематично пов’язані з проблемою розробки та використання інформаційних технологій обробки даних в складних системах різного функціонального призначення, є схеми перетворення інформації (насамперед, схематичні відображення алгоритмів системи) та спеціалізовані діаграми. Діаграми та схеми є важливими інструментами в науці, бізнесі та індустрії. Вони широко застосовуються у процесі розробки бізнес-звітів, наукових статей та доповідей, технічної документації та інших типів документів. Крім того, побудова діаграм є невід’ємною складовою будь-яких UML-засобів. UML – уніфікована розповсюджена мова моделювання. Для оптимізації дій проектувальника були створені CASE-засоби спеціального виду, що дозволяють професійно будувати діаграми UML, які зараз є необхідним елементом системного та структурно-функціонального аналізу [45]. Крім того, графічне представлення діаграм UML, що інколи наводяться в електронних текстових документах, може здійснюватися за допомогою деяких графічних редакторів. Зазначимо, що і схеми алгоритмів, і діаграми

UML є графічними зображеннями, що містять лінії, які відображають зв’язки між відповідними вузлами схеми або діаграми. При цьому топологія конкретних зображень разом з позначеннями, що зазвичай наявні на рисунках текстових документів, є певною мірою унікальною. Це означає, що порівняльний аналіз топології схем та діаграм в документах може доповнювати результати виявлення в них повних або неповних подібностей [46].

Виникають проблеми, які не завжди можна вирішити за допомогою стандартних процедур. Ці проблеми виникають, насамперед, з використанням під час розробки графічних документів різноманітних графічних редакторів та форматів графічних даних. Слід зазначити, що дотепер відсутні універсальні вимоги до імпортування графічних об’єктів в тексти статей або доповідей. Зазвичай редакторські вимоги полягають у зазначенні мінімальної роздільної здатності (наприклад, не менше 300 dpi), переліку рекомендованих форматів представлення рисунків (наприклад, *bmp, *.tif, *.jpg*) та рекомендацій щодо бажаного використання деяких графічних редакторів (наприклад, *MS Visio, MS Paint, CorelDraw, AdobeIllustrator*).

Інколи до вимог додаються умови імпортування графічного матеріалу в текст статті у вигляді об’єктів у градаціях сірого або чорно-білого, без використання обтікання та прив’язки об’єктів. Розміри рисунків та діаграм обмежуються при цьому заданими габаритами, а розміри шрифтів текстових об’єктів (підписи осей, шкала осей, легенди, підписи об’єктів тощо) у рисунках мають бути враховані так, щоб при масштабуванні цього рисунка остаточні розміри цих текстових об’єктів були пропорційні розміру основного тексту.

У тексті схеми та UML-діаграми можуть бути наведені як рисунки або як графічні об’єкти, створені за допомогою одного зі спеціальних графічних редакторів та сбережені в одному із форматів графічних даних

Формати графічних даних

У комп'ютерній графіці застосовують чимало форматів файлів для збереження растрових або векторних зображень [47]. Розглянемо деякі з таких форматів, які використовуються для побудови схем та діаграм в текстових електронних документах.

T1FF (Tagged Image File Format). Формат призначений для збереження растрових зображень високої якості (розширення імені файлу – .tif). Відноситься до числа широко розповсюджених, відрізняється перенесенням між платформами (IBM PC і Apple Macintosh). Для зменшення розміру файлу застосовується алгоритм стиску Лемпеля-Зіва-Велча (LZW).

PSD (PhotoShop Document). Власний формат програми Adobe PhotoShop (розширення імені файлу – .psd), що дає можливості збереження растрової графічної інформації.

PCX. Формат збереження растрових даних програми PC PaintBrush фірми Z-Soft (розширення імені файлу – .pcx). Відсутність можливості зберігати кольоровідокремлені зображення і інші обмеження призвели до втрати популярності формату.

PhotoCD. Формат розроблений фірмою Kodak для збереження цифрових растрових зображень високої якості (розширення імені файлу – .pcd).

Windows Bitmap. Формат збереження растрових зображень в операційній системі Windows (розширення імені файлу – .bmp). Відповідно підтримується всіма додатками, що працюють у цьому середовищі.

JPEG (Joint Photographic Group). Формат призначений для створення та збереження растрових зображень (розширення імені файлу – .jpg). Дозволяє регулювати співвідношення між ступенем стиску файлу і якістю зображення. Формат найчастіше рекомендують використовувати для електронних публікацій.

GIF (Graphics Interchange Format). Стандартизований у 1987 році як засіб збереження стиснутих зображень з фіксованою (256) кількістю кольорів

(розширення імені файлу – .gif). Набув популярності в Інтернеті завдяки високому ступеню стиску.

WMF (Windows MetaFile). Формат збереження векторних зображень операційної системи Windows (розширення імені файлу – .wmf). За визначенням підтримується всіма додатками цієї системи. Однак відсутність засобів для роботи зі стандартизованими колірними палітрами, прийнятими в поліграфії, і інші недоліки обмежують його застосування.

EPS (Encapsulated PostScript). Формат опису як векторних, так і растрових зображень мовою PostScript фірми Adobe, у фактичному стандарті в області допечатних процесів і поліграфії (розширення імені файлу – .eps). Оскільки мова PostScript є універсальною, у файлі можуть одночасно зберігатися векторна і растрова графіка.

PDF (Portable Document Format). Формат опису документів, розроблений фірмою Adobe (розширення імені файлу – .pdf). Хоча цей формат в основному призначений для збереження документа цілком, його можливості дозволяють забезпечити ефективне представлення зображень в текстах.

Графічні редактори

Графічні редактори – це прикладні програми, призначені для створення й обробки графічних зображень (в тому числі, схем та діаграм) на комп'ютері в діалоговому режимі. Програми обробки графіки можна розділити на дві групи: растрові та векторні.

Розглянемо деякі з таких програм, які використовуються для побудови схем та діаграм в текстових електронних документах.

ABViewer. Редактор графічних файлів, що дозволяє працювати як з растровими, так і з векторними форматами файлів (зокрема з TIFF, JPEG, GIF).

b4Look. Редактор графічних файлів, що підтримує основні графічні формати: JPG, TIF, PNG, GIF, BMP. Програма використовує GDI + для максимально швидкої роботи з графічними файлами.

ACDSee. Редактор графічних файлів, що надає основні функції для обробки растрових зображень. До основних переваг програми слід віднести високу швидкість обробки графічних даних, багатопоточність, підтримку більшості відомих графічних форматів (зокрема, TIFF, JPEG, GIF, BMP), можливість конвертації зображень.

Adobe Photoshop. Цей пакет програм має потужні можливості для обробки зображень з використанням різноманітних фільтрів та ефектів. Пакет володіє засобами відновлення пошкоджених зображень, ретушування фотографій тощо. Підтримує як растрові (BMP, JPEG, GIF), так і векторні (AI, CDR) графічні формати.

CorelDraw. Професійний векторний графічний редактор, призначений для обробки і створення векторної та растрової графіки. Програма Microsoft Draw, що входить в комплект MS Office, широко використовується в наукових статтях для створення схем та діаграм.

Gliffy. Графічний редактор, призначений для створення різноманітних схем, діаграм (зокрема BPMN, UML, UI Design, Venn diagrams, SWOT). Високий рівень функціональності досягається завдяки використанню технології Flash.

3.2. Метод порівняння схем та діаграм в електронних документах

Аналіз просторової структури схем

Підхід, що пропонується, передбачає на першому етапі проведення аналізу схем алгоритмів та діаграм за їх просторовою топологією, тобто за особливістю структури зв’язків між вузлами схеми або діаграми [48].

Система автоматизованого аналізу просторової структури схем здійснює їх обробку, що передбачає реалізацію операцій фільтрації, аналізу структурних елементів схем, формування опису схем, візуалізацію, сегментацію та кодування отриманих даних щодо аналізованих об’єктів. Узагальнена структурна схема системи наведена на рис. 3.1.

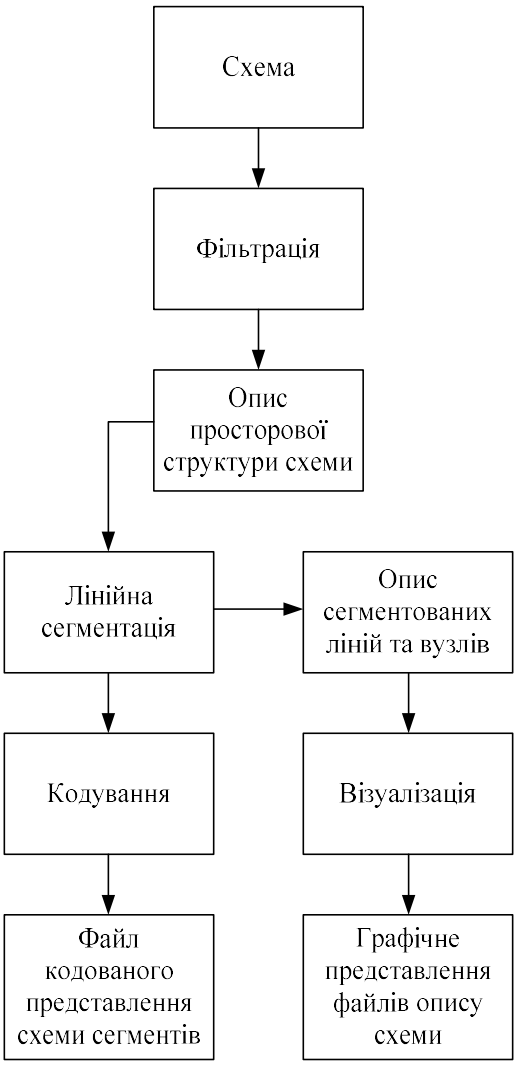


Рисунок 3.1 – Система аналізу просторової структури схем

На вхід системи надходить схема з текстового документу, що аналізується. Залежно від якості зображення та насиченості схеми здійснюється його попередня центроїдна фільтрація з метою видалення випадкових шумів (зайвих точок, що не є частиною інформації щодо структурних елементів зображення), після чого виділяються основні елементи. Далі реалізується центроїдна релаксація, що дозволяє виділити наявні на схемі криві лінії, кути, кола, інші замкнуті фігури, визначити їх геометричні характеристики для подального опису просторової структури схеми. Після релаксації оброблене зображення надходить до підсистеми лінійної сегментації, призначеної для обробки пересічних ліній, що

утворюють вузли та сегменти. Сегментовані лінії та вузли піддаються кодуванню для подальшого кодованого представлення схеми сегментів. Водночас результати сегметації можуть використовуватися для візуалізації та графічного представлення опису схеми.

Розглянемо докладніше окремі блоки системи автоматизованого аналізу просторової структури схем. Центральним елементом цієї системи є підсистема лінійної сегментації, основною функцією якої є автоматизація процедури пошуку ліній та вузлів на растровому зображенні з подальшим кодованим представленням схеми сегментів. Структурна схема підсистеми лінійної сегментації наведена на рис. 3.2.

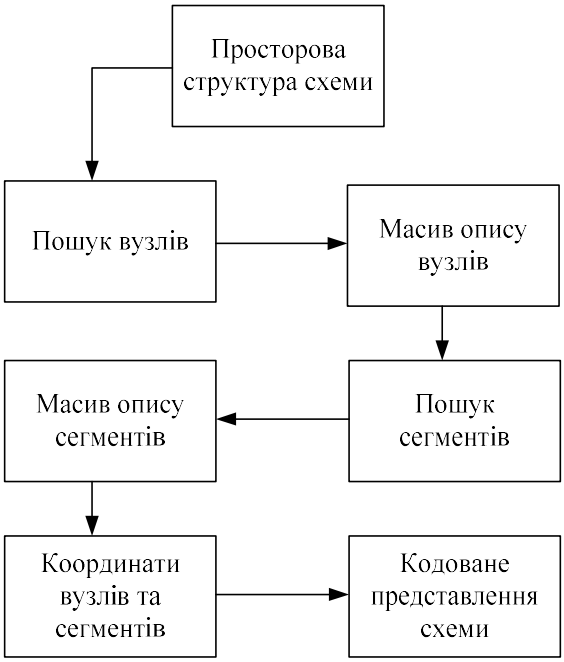


Рисунок 3.2 – Структура підсистеми лінійної сегментації

В цій підсистемі використовуються такі структури даних:

а) формат вхідного масиву точок зображення:

$$\begin{matrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{matrix}, \qquad (3.1)$$

де a_{ij} – елемент масиву точок, що відповідає точці зображення з координатами (i, j) та приймає значення 0 або 1; *i=1,2,...,m*; *j= 1,2,...,n*.

б) формат опису сегментів ліній:

$$\begin{array}{l} n_1 : x_{11,}\ y_{11}\ [\,x_{12}, y_{12}[\,x_{13}, y_{13}\,]] \\ n_2 : x_{21,}\ y_{21}\ [\,x_{22}, y_{22}[\,x_{23}, y_{23}\,]] \\ \ldots \\ n_k : x_{k1,}\ y_{k1}\ [\,x_{k2}, y_{k2}[\,x_{k3}, y_{k3}\,]] \end{array}, \qquad (3.2)$$

де n_i – номер i-го сегмента лінії; x_{i1}, y_{i1} – координати першої точки i-го сегмента лінії; x_{i2}, y_{i2} – координати другої точки i-го сегмента лінії (указуються, якщо точка (x_{i2}, y_{i2}) є вузлом); x_{i3}, y_{i3} – координати напрямку кодування від точки (x_{i1}, y_{i1}) (указуються, якщо точки (x_{i1}, y_{i1}) та (x_{i2}, y_{i2}) є вузлами).

Алгоритми пошуку вузлів та сегментів

Процедура *пошуку вузлів* застосовується для визначення в межах схеми, що обробляється, елементів, які є областями перетину ліній (замкнутими фігурами). Ця процедура передбачає обхід масиву точок схеми з одночасним формуванням масиву елементів вузла, визначенням координат центра вузлів та загальної кількості вузлів. При цьому в масиві вузлів здійснюється нумерація елементів, що дозволяє визначати, до якого з вузлів належить конкретна точка. Отримані за результатами цієї процедури дані використовуються надалі для пошуку сегментів ліній, кодування ліній та визначення координат сегментів для їх графічного представлення. Вхідною інформацією процедури пошуку вузлів є масив точок з файлу, формат якого має вигляд (3.1). Розмір цього масиву відповідає розміру схемного зображення. Вихідною інформацією цієї процедури є масив вузлів схеми з зазначенням номера кожного вузла для їх подальшої ідентифікації.

На вхід процедури надходить схемне фільтроване зображення з текстового документу, що аналізується, у вигляді масиву точок, кожний елемент якого може приймати значення 0 або 1 (в разі відсутності чи

наявності точки відповідно). Таким чином, структурні елементи схемного зображення формуються як сукупність точок зі значенням 1. Кожній одиниці зображення в масиві відповідає елемент масиву вузлів, значення якого формується таким чином:

- якщо значення елемента менше нуля, то цей елемент ще не є оброблений (це необхідно при обході масиву точок для виключення повторної обробки елементів);

- якщо значення елемента дорівнює нулю, то даному елементу не відповідає жоден з вузлів і відповідна точка в масиві точок не належить до вузла;

- якщо значення елемента більше нуля, то воно відповідає номеру вузла, до якого належить аналізована точка.

У загальному випадку зображення містить лінії одиничної товщини, тобто кожна точка окремої лінії може мати не більше двох сусідніх точок, що водночас визначають напрямок руху лінії. Лінію одиничної товщини можна описати таким чином:

$$S(x_k, y_k) = \{1,2\}, \tag{3.3}$$

де N – кількість точок в лінії; (x_k, y_k) – координати точки a_k; $S(x_k, y_k)$ – кількість сусідніх з a_k точок, $k = \overline{1, N}$.

Кількість сусідніх з a_k точок можна визначити так:

$$S(x_k, y_k) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} C[x_k + i, y_k + j] - C[x_k, y_k], \tag{3.4}$$

де $C[x_k, y_k]$ – масив точок вхідного зображення.

Для розглянутого варіанта опису точки початку та кінця лінії мають лише одну сусідню точку, що однозначно визначає напрямок руху лінії. Всі інші точки лінії мають по дві сусідніх точки (попередню та наступну). Це дозволяє уникнути надмірної інформації для опису лінії. Зображення, що містять лінії саме такого типу, можуть бути отримані на етапі попередньої

фільтрації. За таких умов точки, що мають більше ніж дві сусідніх, є вузловими та розглядаються як точки входу лінії до областей перетину ліній (замкнутих фігур). Це можна відобразити таким чином:

$$B[x,y]=\begin{cases} n, S(x,y)>2, \\ 0, S(x,y)\le 2 \end{cases} \quad (3.5)$$

де $B[x,y]$ – масив вузлових точок для зображення розміру $M \times N$; n – номер вузла, що обробляється.

Пошук вузлових елементів передбачає послідовний розгляд елементів масиву точок. Однак при перетині ліній точки, що є сусідніми з вузлом, можуть бути більше ніж дві сусідні точки. Цієї неоднозначності можна запобігти за допомогою визначення центра вузла, координати якого мають бути отримані як середнє арифметичне усіх точок, що належать до вузла. Зазначимо, що дві точки вважаються належними до одного вузла, якщо існує безперервний шлях з точок цього вузла, що їх з'єднує.

Алгоритм пошуку вузлів реалізовано у вигляді програмного модуля «SN (search nodes)»:

1. Початок;

2. Ініціалізація масиву вузлів;

3. i=0; j=0; z=0;

4. Якщо j>=N, то перехід до п.11;

5. Якщо i>=M, то перехід до п.10;

6. Якщо (apix[i][j]=1) та (apix2[i][j]<0) та (NC(i,j)>2), то перехід до п.7, інакше до п.9;

7. z=z+1;

8. NS(i,j,z);

9. i=i+1; перехід до п.5;

10. i=0; j=j+1; перехід до п.4;

11. Кінець.

У наведеному алгоритмі застосовано такі позначення: *M*, *N* – розміри вхідного зображення (схеми / діаграми); (apix[i][j], apix2[i][j] – масиви точок та вузлів відповідно; NC (neig count) – функція обчислення кількості сусідніх точок; NS (node select) – функція пошуку вузлових точок ; *z* – номер поточного вузла.

Процедура *пошуку сегментів* застосовується для визначення в межах схеми, що обробляється, сегментів ліній за результатами реалізації попередньої процедури пошуку вузлів та виділення структурних елементів зображення (схеми / діаграми). Це дозволяє отримати опис зображення схеми, що надалі кодується та обробляється в системі аналізу. Процедура лінійної сегментації передбачає виконання таких операцій:

- пошук окремих сегментів ліній, що не містять вузлів;
- пошук сегментів ліній, одна з початкових або кінцевих точок яких є вузлом;
- пошук сегментів ліній, обидві початкові та кінцеві точки яких є вузлами.

При цьому використовуються особливості, пов’язані з різними характеристиками структурних елементів графічного зображення. Формування результатів обробки визначається способом їх передавання до інших блоків системи (найчастіше, за допомогою ланцюгового кодування). Наприклад, у випадку опису сегментів ліній, повністю обмежених вузлами, виникає необхідність передавання додаткової координати для однозначного визначення напрямку руху при формуванні ланцюгового коду.

Вхідними даними для цієї процедури є: вхідний масив точок зображення; масив вузлів, що містить опис вузлових точок (їх області, центри та їх нумерацію для унікальної ідентифікації).

Пошук сегментів ліній здійснюється за припущенням їх одиничної товщини (лінії саме такого типу можуть бути отримані на етапі попередньої фільтрації). Ця процедура передбачає обхід масиву точок схеми та аналіз їх можливої належності до масивів елементів вузлів, що дозволяє визначити, до

якого із сегментів лінії належить конкретна точка. Кожній одиниці зображення в масиві відповідає елемент масиву сегментів, значення якого формується таким чином:

- якщо значення елемента менше нуля, то цей елемент ще не є оброблений (це необхідно при обході масиву точок для виключення повторної обробки елементів);
- якщо значення елемента дорівнює нулю, то даному елементу не відповідає жоден із сегментів ліній і відповідна точка в масиві точок не належить до цих сегментів;
- якщо значення елемента більше нуля, то воно відповідає номеру сегмента ліній, до якого належить аналізована точка.

Це можна описати таким чином:

$$D[x,y]=\begin{cases} n, S(x,y)<3 \\ 0, S(x,y)\geq 3 \end{cases},\ x=\overline{1,M},\ y=\overline{1,N}, \qquad (3.6)$$

де (x,y) – координати точки; D – масив сегментів; n – номер сегмента, що обробляється.

Алгоритм пошуку сегментів реалізовано у вигляді програмного модуля «SS (search segs)»:

1. Початок;

2. Ініціалізація масиву сегментів;

3. i=0; j=0; ; nsegs =0;

4. Якщо j>=N, то перехід до п.11;

5. Якщо i>=M, то перехід до п.10;

6. Якщо (apix[i][j]=1) та (apix1[i][j]<0) та (NC(i,j)=1) або (NC(i,j)=2), то перехід до п.7, інакше до п.9;

7. nsegs = nsegs +1;

8. NL(i,j, nsegs);

9. i=i+1; перехід до п.5;

10. i=0; j=j+1; перехід до п.4;

11. Кінець.

У наведеному алгоритмі застосовано такі позначення: *M*, *N* – розміри вхідного зображення (схеми / діаграми) по горизонталі та вертикалі; (apix[i][j], apix1[i][j] – масиви точок та сегментів відповідно; NC (neig count) – функція обчислення кількості сусідніх точок; NS (neig line) – функція обробки точок одного сегмента лінії; nsegs – номер поточного сегмента.

Алгоритм кодування сегментів

Процедура *кодування сегментів* застосовується для формування кодів трьох типів сегментів ліній за результатами попереднього пошуку сегментів ліній та вузлів. При цьому обробляються такі типи сегментів ліній: сегменти, що не мають вузлів; сегменти, одна з крайніх точок яких є вузлом; сегменти, обидві крайні точки яких є вузлами.

Вхідними даними для цієї процедури є: вхідний масив точок зображення (3.1); масив вузлів, що містить опис вузлових точок (їх області, центри та їх нумерацію для унікальної ідентифікації); масив сегментів, що містить опис сегментів ліній (точки, що належать лінії, координати їх перетину та їх нумерацію для унікальної ідентифікації).

При кодуванні сегментів ліній застосовується формат опису (3.2). Ця процедура передбачає пошук крайніх точок сегментів ліній, за якими визначається належність кожного з них до одного з трьох розглянутих вище типів. За результатами послідовного обходу координат точок зображення зі значеннями в масивах сегментів та вузлів однозначно визначається тип та координати сегмента, що обробляється, а також спосіб його кодування. Перевірка належності деякої точки зображення сегментам здійснюється таким чином: якщо цій точці відповідає додатне значення в масиві сегментів, то вона належить до сегмента з відповідним номером. Аналогічно перевіряється належність деякої точки зображення вузлам: якщо цій точці відповідає додатне значення в масиві вузлів, то вона належить до вузла з відповідним номером.

У свою чергу, порівняння масивів дозволяє робити такі висновки:

- якщо точці зображення відповідає додатне значення в масиві сегментів та нульове значення в масиві вузлів, то ця точка належить до сегменту і не належить до вузлів;

- якщо точка має додатне значення в масиві сегментів та додатне значення в масиві вузлів, то вона належить більш ніж до одного сегмента.

При кодуванні сегментів ліній визначається тип сегмента (відповідно з визначеним типом формуються до шести координат точок опису сегмента).

Алгоритм кодування сегментів реалізовано у вигляді програмного модуля «CS (code segs)»:

1. Початок;

2. Ініціалізація масивів та змінних;

3. Якщо пошук вузлів завершено, то перехід до п.5, інакше до п.4;

4. Пошук вузлів (SN), перехід до п.3;

5. Якщо пошук сегментів завершено, то перехід до п.7, інакше до п.6;

6. Пошук сегментів (SS), перехід до п.5;

7. i=0; j=0; lnum=0; liden =0;

8. Якщо j>=N, то перехід до п.15;

9. Якщо j>= M, то перехід до п.14;

10. Якщо (apix[i][j]=1) та и(apix1[i][j]<0) та ((NC (i,j)=1) або (NC (i,j)=2)), то перехід до п.11, інакше до п.13;

11. lnum = lnum +1; liden= liden+1;

12. LV (i,j, liden);

13. i=i+1; перехід до п.9;

14. i=0; j=j+1; перехід до п.8;

15. Кінець.

У наведеному алгоритмі застосовано такі позначення: lnum – порядковий номер сегмента, що обробляється; liden – ідентифікатор сегмента, що обробляється, в масиві сегментів; LV (line vect) – функція обробки точок сегмента та кодування його координат.

Підсистема порівняння схем та діаграм в текстових документах

Запропонований метод аналізу та кодування схем та діаграм може бути застосований для порівняння графічних об’єктів і подальшого їх зберігання в базі даних системи виявлення дублікатів електронних документів.

У підсистемі порівняння схем та діаграм, що містяться в текстових документах, можуть бути використані розглянуті вище алгоритми аналізу структури графічних об’єктів. Однією з функцій такої підсистеми є перевірка оригінальності графічного об’єкта (насамперед, схеми або діаграми UML), як за джерелами мережі Інтернет, так і за власними базами текстів з такими графічними об’єктами (статей, дисертацій, курсових, дипломних та магістерських робіт та ін.).

Припустимо, що згідно з використанням однієї з наявних систем пошуку текстів (без урахування можливої близькості наявних там схем або діаграми), що є близькими за сучасними критеріями близькості текстів або їх фрагментів, була сформована обмежена множина A, яка складається з об’єктів з високим ступенем подібності до текстового об’єкта X зі схемами або діаграмами, оригінальність якого аналізується. Як зазначалося вище, є різні варіанти представлення графічних об’єктів, що входять до документів з множини А. Втім слід зазначити, що лише частина з них застосовується в переважній більшості схем та діаграм текстових документів. Як правило, несумісні формати мають файли растрових та векторних зображень, хоча є формати, що дозволяють зберігати дані різних класів.

Чимало додатків орієнтовані на власні «специфічні» формати, перенесення їхніх файлів в інші програми змушує використовувати

спеціальні фільтри або експортувати зображення в «стандартний» формат. Зважаючи на особливості запропонованого вище методу аналізу та кодування схем та діаграм, зробимо наголос на варіанти представлення текстових об’єктів, що входять до множини *A*, коли вони містять графічні растрові зображення схем та діаграм (наприклад, у форматах GIF або JPEG). Таке представлення є достатньо поширеним в текстових електронних документах, що мають науковий характер. У той же час є програмні засоби, які в деяких випадках дозволяють конвертувати схеми та діаграми інших форматів до стандартного растрового варіанта.

Розглянемо задачу порівняння *L* схем $X_1, X_2, \ldots, X_L$ з текстового об’єкта *X* зі схемами або діаграмами, оригінальність якого аналізується і який має високий ступінь подібності до текстових об’єктів множини *A*. Алгоритм такого порівняння полягає у такому:

- схеми $X_1, X_2, \ldots, X_L$ обробляються згідно з розглянутими алгоритмами структурного аналізу, що дозволяють сформувати їх кодовані представлення $X_{11}, \ldots, X_{LL}$;

- аналогічно формуються кодовані представлення схем з текстових об’єктів, що входять до множини *A* (множина кодованих шаблонів *Y*);

- здійснюється перевірка повних збігів структури $X_{11}, \ldots, X_{LL}$ з кодованими шаблонами *Y* (процедура такої перевірки є тривіальною) та формується клас шаблонів *Z*, що відповідають випадкам повного збігу за результатами перевірки;

- у разі наявності однієї або кількох схем в множині *Z*, що мають повні структурні збіги зі схемами у документі *X*, здійснюється перевірка збігу текстових написів і пояснень у схемах;

- у разі відсутності у документі *X* посилань на первинне джерело для випадків повного збігу (як за структурою схем, так і за відповідними текстовими поясненнями) документ *X* автоматично заноситься до

електронної форми «Індикація можливості плагіату схем», що створюється для кожного аналізованого об’єкта.

Для розглянутого алгоритму може бути передбачена також можливість виявлення часткового дублювання схем. Цей алгоритм може доповнювати загальну гіпотетичну процедуру повного автоматизованого аналізу оригінальності документу, що перевіряється, з урахуванням подібностей тексту, формул, зображень, таблиць та ін.

Результатом реалізації запропонованої гібридної схеми може бути формування електронної форми «Індикація можливості плагіату схем та діаграм», що створюється для кожного об’єкта, з визначенням загальних коефіцієнтів подібності графічних елементів, які перевіряються, та побудовою відповідної порівняльної гістограми [49].

РОЗДІЛ 4. МОДЕЛІ ТА МЕТОДИ ВИЯВЛЕННЯ НЕПОВНИХ ДУБЛІКАТІВ У ТАБЛИЦЯХ

4.1. Формальна постановка проблеми знаходження неповних дублікатів у таблицях, які містять дані різних типів: текстова інформація, графічні зображення, числові дані тощо

Під таблицею розуміють розташування даних різних типів у рядках і стовпцях або у вигляді більш складної структури. Табличне представлення інформації широко застосовується в наукових дослідженнях, представленні результатів аналізу даних значного обсягу тощо. Вигляд таблиць значно варіюється за структурою, гнучкістю та позначенням і визначається залежно від особливості контенту, що має бути в них репрезентований. Особливістю таблиць у технічних та наукових статтях і в інших публікаціях є те, що вони, як правило, відділяються від основного тексту в окремий блок, що має нумерацію та назву [50]. У класичному розумінні під таблицею вважають контекст для читання [51], також таблиці є невід'ємною частиною інформаційних систем обробки даних [52]. У роботі [53] вважають табличне представлення даних одним з кроків процесу їх відображення та візуалізації.

Загалом, таблиця складається з розташованих впорядкованим чином рядків та стовпців. Термін «рядок» може бути визначений також як вектор, запис або кортеж. Традиційний термін «стовпець» часто інтерпретують як поле, атрибут або властивість, що має визначену назву або ім'я. Ця назва апріорі може складатися зі слова або послідовності слів, презентуватися числовими значеннями, формулою (формулами) або датою. Перетин визначеного рядка та стовпця визначає комірку, яка наповнюється контентом. В цілому, представлення таблиць дуже різноманітне: елементи таблиць можуть бути згруповані за деякими ознаками, розбиті на сегменти, вкладені з різним ступенем вкладеності, містити анотації, об'єкти типу формула, дата тощо.

У випадку приховання плагіату оригінальна таблиця може бути порівняно легко видозмінена: переставлені рядки та стовпці, змінені заголовки та назви полів. Також, якщо йде мова про недобросовісне запозичення, дані таблиці можуть бути суттєво змінені. Наприклад, якщо в оригінальній таблиці розміщені результати числового експерименту, то в запозиченні ці числові дані можуть бути навмисно змінені. Приклад деякої оригінальної таблиці вказано в табл. 4.1, а її модифікація – в табл. 4.2. Все це значно ускладнює порівняння таблиць на виявлення неповних дублікатів.

Таблиця 4.1. Приклад оригінальної таблиці

Назва компанії	**Прибуток**	**Кількість працівників**
Корпорація ТехБуд	225 750	560
Нова Компанія	785 960	1220
Альфа-Буд	895 975	1150

Таблиця 4.2. Приклад модифікації оригінальної таблиці

Компанія	***К-сть працівників***	***Приб.***
Альфа-Буд	1150	895 975
Корп. ТехБуд	560	225 750
Н. Компанія	1220	785 960

Розглядається задача виявлення неповних дублікатів у таблицях, що репрезентують результати експериментальних наукових досліджень і представляються в текстах дисертаційних та дипломних робіт, наукових публікаціях тощо. Оскільки дані в таких таблицях можуть представлятися різними типами, традиційні методи ідентифікації подібності для них повинні бути замінені комплексними або гібридними методами.

Дослідження, які розглянуті в цій роботі, можуть бути цінним інструментом для уникнення зловживань та плагіату в сфері вищої освіти, а також створення механізмів забезпечення боротьби з плагіатом на рівні держави. Запропонований метод може бути використаний для виявлення

неповних дублікатів у дисертаційних та дипломних роботах, а також у наукових публікаціях.

Задача знаходження неповних дублікатів (NDD) є традиційною задачею інтелектуального аналізу тексту. Зокрема для виявлення подібностей та неповних дублікатів у текстових даних використовують алгоритм розрахунку відстаней Левенштейна [54], що є рядковими метриками, які дозволяють розрахувати різницю між двома послідовностями символів. Також для знаходження підрядка в тексті, який подібний до заданого зразка, використовують алгоритм Bitap з модифікаціями Манбера та By [55]. Відстані між рядками в цьому випадку визначаються в термінах відстані Левенштейна.

Для задачі знаходження неповних дублікатів також застосовують алгоритм BK-tree [56, 57], що полягає в побудові метричного дерева для дискретних метричних просторів. Цей алгоритм може бути використаний для виявлення подібних рядків з врахуванням словника. У роботі [29] пропонується виявляти матеріал з підозрою на плагіат на підставі аналізу подібностей. Також методи пошуку неповних дублікатів на основі локально-чутливого хешування розглянуті в роботі [1].

У роботі [24] наведено математичну формалізацію для задачі пошуку неповних дублікатів в текстових даних. У роботі [25] наведено порівняння різних методів пошуку неповних дублікатів за середнім часом роботи алгоритмів та кількістю знайдених слів. Задача знаходження неповних дублікатів у числових даних пов’язана з ідентифікацією подібностей у часових рядах на основі співставлення зі зразком з використанням методу найближчих сусідів з заданою метрикою [58, 59].

У роботі [14] пропонується система, що заснована на пошуку зразків та їх співставленні. У роботі [15] розглядаються текстурні особливості, які відповідають зоровому сприйняттю людини, для цифрового аналізу текстур, а також розпізнавання графічної інформації. У роботі [16] розглядають порівняльний аналіз різних методів розпізнавання зображень. У роботі [17]

описано систему, яка здійснює пошук за ключовими словами та за кадрами з врахуванням зворотнього зв'язку під час обробляння зображень. Оскільки в таблиці можуть бути презентовані текстові, графічні та числові дані, розглянуті публікації можуть стати при нагоді у разі виявлення в таблицях неповних дублікатів. Проте на теперішній час нема чітко обґрунтованого методу для порівняння таблиць з урахуванням різних способів їх представлень.

Задача знаходження неповних дублікатів у таблицях є процесом ідентифікації таких таблиць, які найбільш подібні одна до одної. Подібність у цьому випадку виражається деяким функціоналом F, що задає відстань між таблицями. Якщо ця відстань не перевищує порогового значення $\lambda \in R$, то таблиці вважаються подібними, а отже, в даних цих таблиць наявні неповні дублікати.

Метою дослідження є побудова алгоритму аналізу таблиць на виявлення в них подібностей та побудова гібридного методу для виявлення неповних дублікатів у таблицях на основі методів локально-чутливого хешування та найближчого сусіда.

Для досягнення мети були поставлені такі завдання:

– визначення типів та класифікація представлення даних у табличному вигляді;

– проведення індексації контенту таблиці;

– побудова гібридного методу для виявлення неповних дублікатів у таблицях на основі знаходження неповних дублікатів у текстових даних та моделі ідентифікації подібностей у числових даних на основі методу найближчого сусіда.

Можна виділити шість різних типів контенту, який відображається в комірках таблиці:

1. Числовий. У цьому випадку враховуються тільки ті числові множини з ієрархії чисел, які не містять літер в записі числа, а також інших позначень (корінь квадратний, риска дробу тощо). Тобто до цього типу належать всі

дійсні числа. Слід зазначити, що під час попереднього розгляду таблиці дані числового типу з десятковим розділювачем мають бути уніфіковані, тобто в усіх таблицях, що аналізуються системою на знаходження неповних дублікатів, десятковим розділювачем має бути або крапка «.», або кома «,». Відповідно до міжнародного стандарту ISO 31-0, як десятковий розділювач можна використовувати і крапку, і кому.

2. Текстовий або рядковий. Цей тип є скінченою послідовністю символів з визначеного алфавіту.

3. Формула. Цей тип є графічним об’єктом, який створено за допомогою певного редактора формул або мови розмітки: MathType, KFormula, MathCastmula, TeX, MathML тощо.

4. Дата та час. Цей тип відображає запис, що містить число, місяць та рік, (іноді номер тижня). Відповідно до міжнародних стандартів ISO 8601, виділяють два типи запису дати: рік-місяць-число та рік.місяць.число (наприклад, 2016-05-30 та 2016.05.30). Згідно зі стандартом ГОСТ Р 6.30-2003 допускається запис число.місяць.рік (наприклад, 30.05.2016). У США застосовується запис місяць/число/рік (наприклад, 5/30/2016). Також для маркування виробів використовується запис рік+тиждень (наприклад, 1623). Всі форми запису дати можна чітко означити, тому у випадку, якщо розглядаються таблиці з комірками типу дата, формати повинні бути приведені до одного типу.

5. Рисунок у вигляді графічного об’єкта.

6. Комбінований, що містить одразу кілька з перелічених типів.

Розглянемо відомі типи табличного представлення інформації. Звичайна таблиця – це таблиця зі скінченною кількістю рядків та стовпців, яка не містить групування та об’єднання кількох комірок в одну, і кожна комірка таблиці містить дані визначеного типу (числовий, рядковий, дата, формула). У таблиці 4.3. в першому стовпці подано текстові дані, в другому стовпці – числові, в третьому – типу дата.

Таблиця 4.3. Приклад звичайної таблиці

ПІП	Зар. плата	Дата нарахування з/п
Іваницький І.А.	4523.66	22.10.2016
Таценко В.В.	4955.25	23.10.2016
Харченко Г.Я.	3120.35	22.10.2016

Багатовимірна таблиця – це таблиця, у якій дані нормалізовані та впорядковані у визначеній ієрархії. Прикладом є таблиця множення (табл. 4.4).

Таблиця 4.4. Приклад багатовимірної таблиці

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Таблиці створюються за допомогою табличних редакторів, які є спеціальними комп’ютерними програмами для побудови, аналізу та зберігання даних в табличній формі. Вигляд вікна програми Microsoft Excel показано на рис. 4.1.

Усі табличні редактори працюють за єдиним принципом: кожна комірка може містити числові, текстові дані або результати розрахунків за формулами з використанням вмісту інших комірок та стандартних математичних, статистичних та фінансових операцій і функцій. Окрім створення і редагування таблиць, у табличних редакторах можна створювати діаграми та графіки, а також там наявні інструменти для використання створених таблиць в якості баз даних.

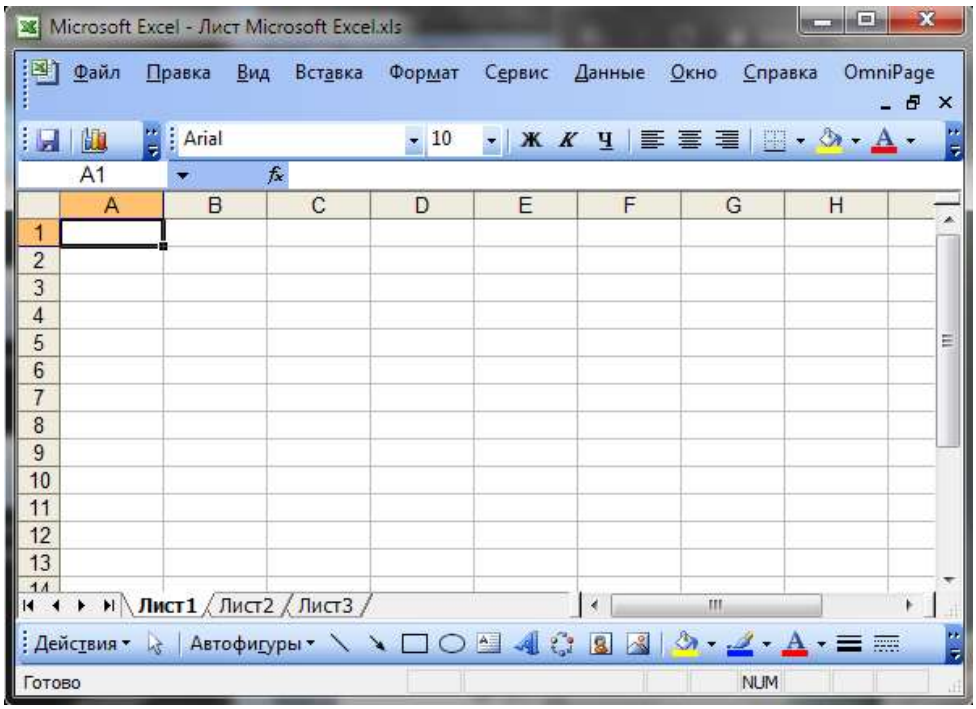


Рисунок 4.1 – Вигляд основного вікна табличного редактора Microsoft Excel

Усі табличні редактори працюють за єдиним принципом: кожна комірка може містити числові, текстові дані або результати розрахунків за формулами з використанням вмісту інших комірок та стандартних математичних, статистичних та фінансових операцій і функцій. Окрім створення і редагування таблиць, у табличних редакторах можна створювати діаграми та графіки, а також там наявні інструменти для використання створених таблиць в якості баз даних.

Найбільш популярними засобами для створення і редагування таблиць, якими користуються і під час написання наукових робіт є: Microsoft Excel, Corel Quattro Pro, Lotus 1-2-3, Gnumeric, LibreOffice Calc (open sourse software), KSpread тощо [60]. Як показує практика, переважна більшість таблиць у наукових статтях та дисертаційних роботах в Україні оформлюється за допомогою редактора Microsoft Excel. Останнє регламентується «Інструкцією з підготовки та подання матеріалів атестаційних справ та матеріалів щодо утворення спеціалізованих вчених рад і внесення до їх складу часткових змін в електронному вигляді» (Наказ ВАК №572 від 21.08.2010), яка передбачає, що інформація в електронному

виглядів повинна зберігатись у форматах .doc та .xls (формати відповідно для текстових та табличних даних Microsoft).

Задача пошуку неповних дублікатів в таблицях пов'язана з задачею пошуку неповних дублікатів в електронних документах взагалі. Це пов'язано з тим, що текст результатів наукових досліджень може складатися з контенту різного типу, так як і контент таблиці: текстові дані, числові значення, зображення, схеми, математичні формули. Тому для побудови концептуальної моделі пошуку неповних дублікатів у таблицях можна використати концептуальну модель пошуку неповних дублікатів електронних документів.

Задача пошуку неповних дублікатів електронних документів формулюється так: нехай D – деякий вхідний документ, а $\overline{D} = \{D_1, D_2, \text{K}, D_p\}$ – документи, які зберігаються в базі даних, p – кількість документів в базі. Завдання полягає у визначенні таких документів $\overline{\overline{D}} = \{D_1^*, D_2^*, \text{K}, D_l^*\}$ або їх фрагментів, для яких виконується умова:

$$F\left(D, D_i^*\right) < \lambda, i = \overline{1,l}, \overline{\overline{D}} \subset \overline{D}, l < p,$$

де λ – погорове значення, а F – відстань між документами.

На рис. 4.2. зображено концептуальну модель пошуку неповних дублікатів електронних документів.

Згідно з нею процедура виявлення дублікатів складається з трьох основних етапів:

1. Визначення типів даних, які представляють контент документу.

2. Побудова канонічного вигляду та кодування за методом, що відповідає ідентифікованому типу даних.

3. Формування та перевірка умов на наявність неповних дублікатів. На цьому етапі розраховується подібність між фрагментами вхідного документу та документів, які входять до бази даних.

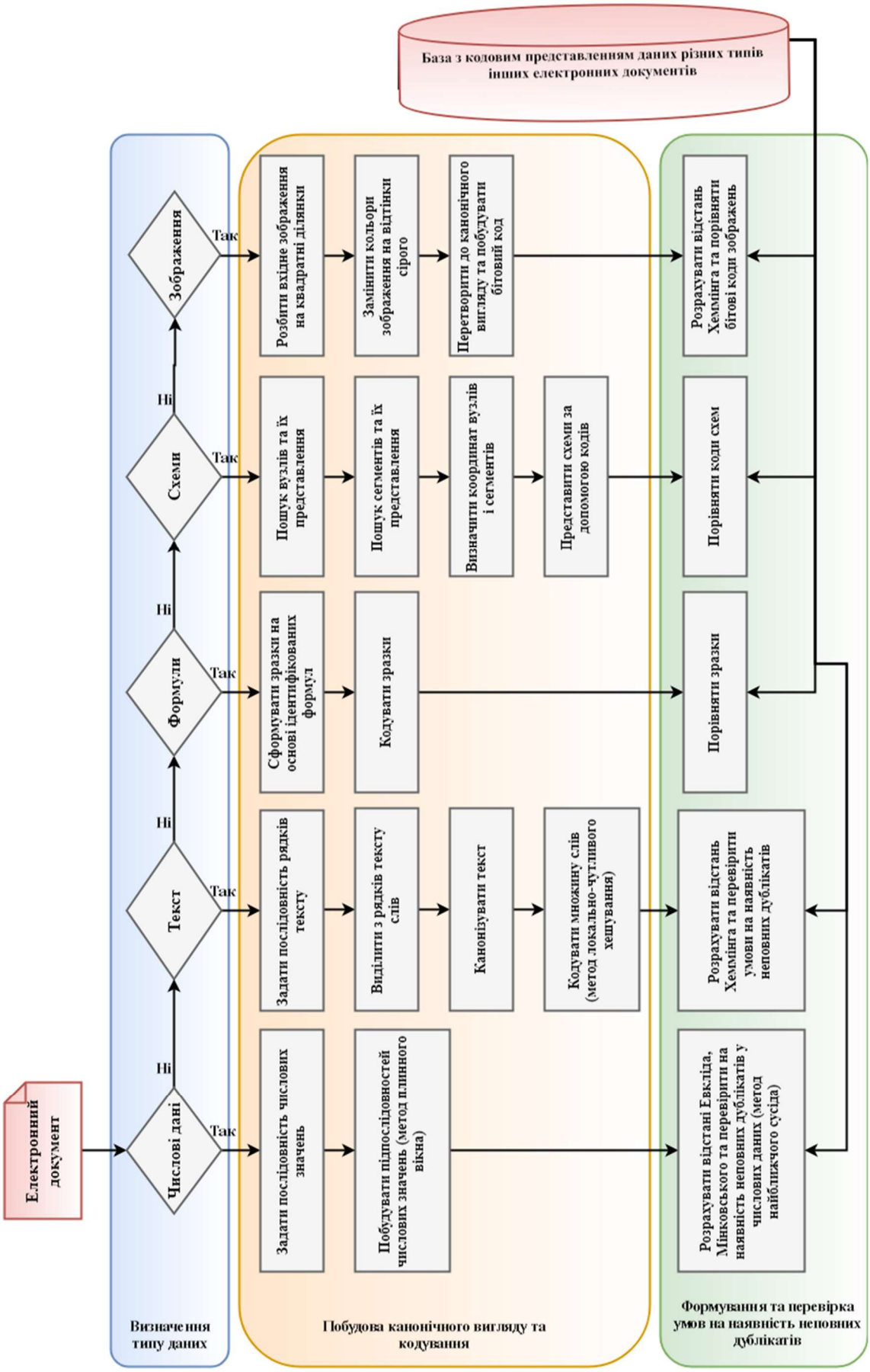


Рисунок 4.2 – Концептуальна модель пошуку неповних дублікатів в електронних документах

У випадку, якщо контент документу містить визначений тип даних (числові значення, текстові дані, математичні формули, схеми, зображення), то дані цього типу відокремлюються від основного документу та аналізуються окремо на наявність неповних дублікатів. Загалом процедура пошуку неповних дублікатів електронних документів та таблиць подібні.

4.2. Модель індексації даних таблиці

Нехай B – деяка вхідна таблиця , а $\overline{B} = \left\{ B_1, B_2, \text{K}, B_p \right\}$ – таблиці, які відібрані з тексту та зберігаються в загальній базі таблиць, p – кількість таблиць у базі. Завдання полягає у визначенні такої множини таблиць $B^* = \left\{ B_1^*, B_2^*, \text{K}, B_l^* \right\}$, що $B^* \subset \overline{B}$, $l < p$, для яких виконується умова:

$$F\left(B, B_i^*\right) < \lambda, \; i = \overline{1,l}, \tag{4.1}$$

де λ – порогове значення, а F – відстань між таблицями. Наявність у базі хоча б однієї такої таблиці з множини $\overline{B}$, для якої виконується умова (4.1) свідчить про те, що таблиця B є запозиченою.

Коміркою K_{ij} деякої таблиці B називається такий елемент таблиці, який утворюється шляхом перетину i -го рядка та j -го стовпця цієї таблиці.

Контентом або даними комірки K_{ij} деякої таблиці B називається таке значення (числове, текстове, типу дата тощо), яке відповідає цій комірці таблиці. Позначимо контент комірки K_{ij} через $Cont\left(K_{ij}\right)$, $i = \overline{1, r_1}$, $j = \overline{1, r_2}$, де r_1 – кількість рядків таблиці B, а r_2 – кількість стовпців таблиці B.

Як було визначено в попередньому пункті, контент комірок може вичерпуватися такими типами: числовий, текстовий, дата, рисунок, формула, комбінований. У випадку, якщо в таблиці наявні рисунки та формули, вони виділяються для дослідження окремо. Контент типу дата після приведення до єдиного формату можна вважати звичайним текстовим рядком. Таким чином,

спростивши можливі варіанти типів контенту комірок можна вважати, що контент таблиці представляється у вигляді двійки [61]:

$$B = \langle N, S \rangle, \tag{4.2}$$

де N – кортеж з числових даних комірок таблиці B, а S – кортеж з текстових даних комірок таблиці B.

Нехай $I = \{1, 2, \text{K}, r_1\}$ – множина індексів або номерів рядків таблиці, а $J = \{1, 2, \text{K}, r_2\}$ – множина індексів або номерів стовпців таблиці B. Проглянемо всі комірки таблиці та визначимо тип їх даних. Якщо дані деякої комірки належить до числового типу, то відповідний контент представляється у вигляді елемента множини $\overline{N}$, якщо текстового типу, то додається до множини $\overline{S}$ за правилом:

$$\overline{N} = \left\{ k \;\middle|\; k \in Cont\left(K_{ij}\right), k \in R, i \in I, j \in J \right\}, \tag{4.3}$$

$$\overline{S} = \left\{ k \;\middle|\; k \in Cont\left(K_{ij}\right), k \in T, i \in I, j \in J \right\}, \tag{4.4}$$

де множина T – множина символів потужності L, $card\left(T\right) = L$:

$$T = \{t_1, t_2, \text{K}, t_L\},$$

t_i – окремий символ, $i = \overline{1, N}$, $t_i \in A$, A – множина атомарних символів формальної мови або алфавіт.

Представимо множини $\overline{N}$ та $\overline{S}$ у вигляді числової послідовності та послідовності рядків відповідно довжини v та w, тобто $N = \{n_1, n_2, \text{K}, n_v\}$ – послідовність з числових значень контентів комірок, $v = card\left(N\right)$, $n_i \in N$, $i = \overline{1, v}$, а $S = \{s_1, s_2, \text{K}, s_w\}$ – послідовність з рядків контентів комірок $w = card\left(S\right)$, $s_j \in S$, $j = \overline{1, w}$.

Розглянемо окремо представлення цих послідовностей у вигляді, зручному для застосування моделей ідентифікації подібності та пошуку неповних дублікатів.

Розглянемо послідовність $S=\{s_1,s_2,\text{K},s_w\}$. Кожен її елемент містить текст, що може складатися з одного або кількох слів. Проглянемо послідовно всі елементи послідовності від s_1 до s_w та виберемо з них слова. Слово довільного елемента уніграму S задається у вигляді послідовності

$$S_n^{\beta}=\{t_1,t_2,\text{K},t_{\beta}\},$$

де $n\in R$ – порядковий номер слова, β – довжина слова, $t_j\in A$, $t_j\notin C$, $j=\overline{1,\beta}$, $C=\{"\_","\,","\,","\,".","\,"\ ","\,"-","\,":","\,";","\,"\#"\}$ – всі небуквенні символи елементів послідовності S, $S_n^{\beta}\in s_j$, $j\in\{1,2,\text{K},w\}$.

Сформуємо новий уніграм з усіх слів - елементів уніграму рядків S, попередньо виключивши з розгляду так звані стоп-слова. Перелік стоп-слів задамо у вигляді множини

$$M=\{"i","та","але","або","тощо","і т.п.","і т.д."\}.$$

Тоді нова послідовність слів має такий вигляд:

$$W=\{S_1^{\beta_1},S_2^{\beta_2},\text{K},S_m^{\beta_m}\},$$

де β_j, $j=\overline{1,m}$ – довжини слів, а m – їх кількість. Елементи такої послідовності є словами в канонізованій формі. Використовуючи метод плинного вікна, побудуємо сукупність послідовностей:

$$E_1=\{S_1^{\beta_1},S_2^{\beta_2},\text{K},S_h^{\beta_h}\},$$

$$E_2=\{S_2^{\beta_2},S_3^{\beta_3},\text{K},S_{h+1}^{\beta_{h+1}}\},$$

$$\dots$$

$$E_{m-h+1}=\{S_{m-h+1}^{\beta_{m-h+1}},S_{m-h+2}^{\beta_{m-h+2}},\text{K},S_{m-1}^{\beta_{m-1}},S_m^{\beta_m}\},$$

де h – розмір вікна або кількість елементів побудованих послідовностей $E_1,E_2,\text{K},E_{m-h+1}$.

Далі за методом локально-чутливого хешування представимо сукупність послідовностей $F(W)=(E_1,E_2,\text{K},E_{m-h+1})$ у вигляді бітових рядків, тобто

$$\Delta(W)=\left(I(E_1),I(E_2),\text{K},I(E_{m-h+1})\right), \qquad (4.5)$$

де $I(E_k)$ – елемент індексу, що задає бітовий рядок, який однозначно є послідовністю E_k, $k=\overline{1,m-h+1}$. Тобто:

$$I(E_k)=\{\delta_{k1},\delta_{k2},\text{K},\delta_{kc}\}, \qquad (4.6)$$

де $\delta_{kx}\in\{0,1\}$, $k=\overline{1,m-h+1}$, $x=\overline{1,c}$, c – кількість бітів, що є бітовою послідовністю.

Розглянемо числову послідовність $N=\{n_1,n_2,\text{K},n_v\}$ вхідної таблиці B та побудуємо для неї набір підпослідовностей за методом плинного вікна, тобто:

$$K_1=\{n_1,n_2,\text{K},n_g\},$$

$$K_2=\{n_2,n_3,\text{K},n_{g+1}\},$$

…

$$K_{v-g+1}=\{n_{v-g+1},n_{v-g+2},\text{K},n_{v-1},n_v\},$$

де v – кількість елементів послідовності N, а g – розмір вікна або кількість елементів підпослідовностей $K_1,K_2,\text{K},K_{v-g+1}$. Оскільки елементи побудованих підпослідовностей є дійсними числами, $n_i\in R$, $i=\overline{1,v}$, то ці підпослідовності можуть бути g-вимірними векторами. Тобто, якщо вважати, що задано простір R^g, який має евклідову структуру, можна визначити на цьому просторі метрику ρ між будь-якими двома векторами простору $a\in R^g$ та $b\in R^g$: $\rho(a,b)$. Причому ця метрика буде задовольняти аксіому тотожності, тобто:

$$\rho(a,b)=0\Leftrightarrow a=b,$$

аксіому симетрії:

$$\rho(a,b)=\rho(b,a)$$

та аксіому трикутника для деякого вектора $c \in R^g$:

$$\rho(a,c)\le\rho(a,b)+\rho(b,c).$$

Така метрика або міра близькості (подібності) між такими векторами, які є числовими значеннями контентів таблиць, – складова, яка визначатиме ступінь подібності цих таблиць.

4.3. Гібридний метод визначення збігів або неповних дублікатів у таблицях

Відповідно до постановки задачі, нехай B – вхідна таблиця, а B_1, B_2, K, B_p – таблиці, які відібрані та зберігаються в загальній базі таблиць, p – кількість таблиць у базі. Завдання полягає у визначенні таких таблиць з бази, для яких виконується умова (4.1) [61 – 63].

Нехай побудовано послідовність з текстових даних $S=\{s_1, s_2, \text{K}, s_w\}$ та послідовність з числових значень $N=\{n_1, n_2, \text{K}, n_v\}$ для таблиці B. Оскільки база з таблицями вже відома, то очевидно, що кожна така таблиця є проіндексованою. Тобто для кожної з таблиць B_1, B_2, K, B_p відомі послідовності з текстових даних:

$$S^y=\{s_1^y, s_2^y, \text{K}, s_w^y\}$$

та послідовності числових даних:

$$N^y=\{n_1^y, n_2^y, \text{K}, n_v^y\},\ y=\overline{1,p}.$$

Також, очевидно, що для послідовностей слів задано елементи індексу

$$I\left(E^{\,y}_{k_y}\right)=\{\delta^y_{k_y1}, \delta^y_{k_y2}, \text{K}, \delta^y_{k_yc}\},$$

$\delta^{y}_{k_y,x} \in \{0,1\}$, $k_y = \overline{1, m_y - h + 1}$, $x = \overline{1,c}$, c – кількість бітів, що є бітовою послідовністю, а m_y – кількість слів в послідовностях:

$$W^y = \left\{ S_1^{y,\beta_1}, S_2^{y,\beta_2}, \text{K}, S_{m_y}^{y,\beta_{m_y}} \right\},$$

що містять слова S_j^{y,β_j} в канонізованій формі, β_j, $j = \overline{1, m_y}$ – довжини слів.

Побудуємо для рядкової послідовності $S = \{s_1, s_2, \text{K}, s_w\}$ вхідної таблиці B уніграм зі слів в канонізованій формі:

$$W = \left\{ S_1^{\beta_1}, S_2^{\beta_2}, \text{K}, S_m^{\beta_m} \right\},$$

де β_j, $j = \overline{1, m}$ – довжини слів, а m – їх кількість. Далі методом плинного вікна визначимо послідовності $E_1, E_2, \text{K}, E_{m-h+1}$ і за методом локально-чутливого хешування побудуємо елементи індексу:

$$I(E_k) = \{\delta_{k1}, \delta_{k2}, \text{K}, \delta_{kc}\},$$

де $\delta_{kx} \in \{0,1\}$, $k = \overline{1, m - h + 1}$, $x = \overline{1,c}$, c – кількість бітів, що є послідовністю.

Розрахуємо відстані Хеммінга від елементів кожного індексу послідовностей вхідної таблиці до елементів індексу послідовностей тих таблиць, що знаходяться в базі, за формулою:

$$H\left(I(E_k), I\left(E^y_{k_y}\right)\right) = \frac{1}{c} \sum_{j=1}^{c} \left| \delta_{kj} - \delta_{k_y j} \right|, \qquad (4.7)$$

$k = \overline{1, m - h + 1}$, $k_y = \overline{1, m_y - h + 1}$, $y = \overline{1, p}$.

За умови

$$H\left(I(E_k), I\left(E^y_{k_y}\right)\right) < \lambda_H \qquad (4.8)$$

для наперед заданого значення параметра $\lambda_H \in [0,1]$, то з ймовірністю 1 можна стверджувати, що елемент індексу з номером k подібний до елемента індексу з номером k_y таблиці з номером y. Це означає, що таблиця з номером y може бути подібною до вхідної таблиці з порогом λ_H, тобто в ній міститься неповний дублікат.

Побудуємо для числової скінченої послідовності $N=\{n_1,n_2,\text{K},n_v\}$ вхідної таблиці B набір підпослідовностей $K_1,K_2,\text{K},K_{v-g+1}$. Вважаємо також, що для кожної з таблиць B_1,B_2,K,B_p на основі їх послідовностей числових даних $N^y=\{n_1^y,n_2^y,\text{K},n_v^y\}$, $y=\overline{1,p}$ побудовані підпослідовності $K_1^y,K_2^y,\text{K},K_{v-g+1}^y$ за методом плинного вікна:

$$K_1^y=\{n_1^y,n_2^y,\text{K},n_g^y\}$$

$$K_2^y=\{n_2^y,n_3^y,\text{K},n_{g+1}^y\}$$

…

$$K_{v-g+1}^y=\{n_{v-g+1}^y,n_{v-g+2}^y,\text{K},n_{v-1}^y,n_v^y\}.$$

Якщо представити побудовані підпослідовності $K_1,K_2,\text{K},K_{v-g+1}$ та $K_1^y,K_2^y,\text{K},K_{v-g+1}^y$ у вигляді кортежів, то міри подібності між ними визначаються на основі відстані Евкліда, міської метрики або відстані Мінковського. Отримаємо такі y матрицей відстаней:

$$\rho_1\left(K_u,K_r^y\right)=\sqrt{\sum_{j=r}^{g+r-1}\left(n_{j+u-r}-n_j^y\right)^2}, \tag{4.9}$$

$$\rho_2\left(K_u,K_r^y\right)=\sum_{j=r}^{g+r-1}\left|n_{j+u-r}-n_j^y\right|, \tag{4.10}$$

$$\rho_3\left(K_u,K_r^y\right)=\left(\sum_{j=r}^{g+r-1}\left|n_{j+u-r}-n_j^y\right|^t\right)^{\frac{1}{t}}, \tag{4.11}$$

$y=\overline{1,p}$, $u=\overline{1,v-g+1}$, $r=\overline{1,v-g+1}$, t – параметр відстані Мінковського.

Далі знаходимо мінімальні значення для кожного рядка матриць $\rho_\tau\left(K_u,K_r^y\right)$ по $r=\overline{1,v-g+1}$, отримаємо для кожного $y=\overline{1,p}$ відстані:

$$\zeta_\tau\left(K_u,K_{\min}^y\right)=\min_{r=\overline{1,v-g+1}}\left\{\rho_\tau\left(K_u,K_r^y\right)\right\} \tag{4.12}$$

У випадку $u=\overline{1,v-g+1}$, для фіксованого $\tau=\overline{1,3}$.

Нормалізуємо значення отриманих відстаней за формулою:

$$\zeta_\tau^N\left(K_u,K_{\min}^y\right)=\frac{\zeta_\tau\left(K_u,K_{\min}^y\right)-\min\limits_{u=\overline{1,v-g+1}}\left\{\zeta_\tau\left(K_u,K_{\min}^y\right)\right\}}{\max\limits_{u=\overline{1,v-g+1}}\left\{\zeta_\tau\left(K_u,K_{\min}^y\right)\right\}-\min\limits_{u=\overline{1,v-g+1}}\left\{\zeta_\tau\left(K_u,K_{\min}^y\right)\right\}},\qquad(4.13)$$

$y=\overline{1,p}$, $u=\overline{1,v-g+1}$, $\tau=\overline{1,3}$.

Якщо виконується умова

$$\zeta_\tau^N\left(K_u,K_{\min}^y\right)<\lambda_\rho\qquad(4.14)$$

для наперед заданого значення параметра $\lambda_\rho\in[0,1]$, то з ймовірністю 1 можна стверджувати, що вектор з номером u подібний до вектора таблиці з номером y, тобто таблиця містить неповний дублікат. Чим більше значення λ_ρ, тим більш жорсткі вимоги до пошуку неповних дублікатів.

Алгоритм виявлення неповних дублікатів у таблицях B_1,B_2,K,B_p відносно таблиці B, яка задається двійкою (4.2), складається з таких кроків:

1. Виокремити з вхідної таблиці B графічні об'єкти: зображення та формули. Вони досліджуються окремо. Для порівняння формул може бути застосований метод, який базується на порівнянні зразків або шаблонів.

Метод знаходження неповних дублікатів у математичних формулах складається з таких етапів:

1) ідентифікація формул у текстах, що аналізуються;

2) створення зразків на основі ідентифікованих формул;

3) порівняння зразків знайдених формул між собою;

4) перевірка контексту формул, які мають подібний зразок, з урахуванням наявності загальновживаних позначень у цих формулах;

5) формули з подібним зразком та контентом вважаються неповними дублікатами.

2. За наявності у таблиці B комірок з даними типу дата пропонується привести всі дати у відповідність з єдиним форматом, напр. «число.міс.рік», і розглядати дані комірки як комірки з контентом текстового типу.

3. Весь контент комірок числового типу привести до єдиного типу: десятковий розділювач відображати у вигляді коми «,».

4. Видалити з таблиці стовпець «№ п/п», що відображає нумерацію рядків, якщо останній існує.

5. Побудувати послідовність з текстових даних S та послідовність з числових значень N. Одночасно звертати увагу на комірки з комплексним контентом: у випадку, якщо в певній комірці вказано і числові, і текстові дані, текст даної комірки та числові дані розподіляються за окремими елементами послідовностей.

6. Побудувати для послідовності S вхідної таблиці B послідовність зі слів у канонізованій формі W. Далі визначити послідовності $E_1, E_2, \text{K}, E_{m-h+1}$ і за методом локально-чутливого хешування побудувати елементи індексу $I(E_k)$.

7. Далі розрахувати відстані Хемінга від елементів кожного індексу послідовностей вхідної таблиці до елементів індексу послідовностей тих таблиць, які знаходяться в базі за формулою (4.7) та перевірити умову (4.8) для заданого порогового значення $\lambda_H \in [0,1]$. Якщо умова виконується, то це означає, що неповний дублікат ідентифіковано.

8. Побудувати для числової послідовності N вхідної таблиці B набір підпослідовностей $K_1, K_2, \text{K}, K_{v-g+1}$.

9. Представити підпослідовності $K_1, K_2, \text{K}, K_{v-g+1}$ та $K_1^y, K_2^y, \text{K}, K_{v-g+1}^y$ у вигляді векторів та розрахувати відстані за формулами (4.9)–(4.11). За основу можна обрати одну з цих з формул.

10. Застосувати формули (4.12)–(4.13) та перевірити умову (4.14). Якщо для заданого значення порога $\lambda_\rho \in [0,1]$ ця умова виконується, то можна стверджувати, що неповний дублікат ідентифіковано. Значення λ_ρ та λ_H визначаються заздалегідь шляхом експерименту.

11. Визначаючи неповні дублікати за даним методом, особливу увагу слід приділяти тим таблицям, для яких виконуються окремо умови (4.8) та (4.14), особливо якщо розрахована відстань суттєво менша порогу. Якщо ж ці умови виконуються одночасно для визначених значень λ_ρ та λ_H, то така таблиця однозначно містить неповний дублікат і повинна бути окремо розглянута експертом на наявність запозичення.

Таким чином, описано та формалізовано гібридний метод виявлення неповних дублікатів на основі методу локально-чутливого хешування та методу найближчого сусіда. Цей метод може бути використано в антиплагіат-системах, а також інших системах, що призначені для проведення інтелектуального аналізу на ідентифікацію подібностей інформації, яка презентована у вигляді таблиці.

Перевагою описаного методу є те, що під час порівняння наукових робіт на виявлення неповних дублікатів таблиці, які ідентифікуються в текстах, вони можуть бути перевірені на основі розробленого гібридного методу, що враховує одразу текстові та числові дані, презентовані в таблицях.

У результаті проведених досліджень було встановлено:

1. Визначаючи типи контенту комірок та проводячи індексацію таблиці, можна вважати, що таблиця задається у вигляді сукупності текстових та числових даних (4.2). У цьому випадку дані типу «дата» відображають в одному з відомих форматів, а графічні дані та об’єкти типу «формула» досліджуються окремо від таблиць.

2. Першим етапом під час аналізу таблиці є індексація даних. Для текстових даних формуються скінчені послідовності зі слів у канонізованому вигляді, з яких на основі методу локально-чутливого хешування будуються бітові послідовності. Для числових даних формуються скінчені числові послідовності.

3. Для виявлення неповних дублікатів у таблицях може бути застосовано гібридний метод, що знаходить подібності між текстовими та числовими даними окремо та узагальнює отримані результати. Подібність між даними у випадку вхідних текстових даних виражається відстанню Хемінга з заданим пороговим значенням. У випадку числових даних, подібність визначається на основі методу найближчих сусідів з заданими метричними відстанями, а вектори, між якими відстані розраховуються, є безпосередньо числовим контентом таблиці з приведеним до єдиного формату десятковим розділювачем.

Запропонований гібридний метод виявлення неповних дублікатів у таблицях дозволяє ідентифікувати подібності між текстовими та числовими даними таблиць окремо, а потім узагальнює отримані результати. Для текстових даних формуються послідовності зі слів у канонізованому вигляді, з яких на основі методу локально-чутливого хешування будуються бітові послідовності. Подібність між даними в цьому випадку розраховується на основі відстані Хеммінга з заданим пороговим значенням. Ідентифікація подібності між числовими даними таблиць реалізується на основі методу найближчих сусідів із заданими метричними відстанями. Метод дозволяє ідентифікувати неповні дублікати, які наявні в даних вхідної таблиці відносно множини таблиць, що відібрані з наукових публікацій та дипломних і дисертаційних робіт. Слід зазначити, що метод призначено для знаходження неповних дублікатів у таблицях, що містять тільки текстові та числові дані. За наявності в контенті даних таблиць рисунків та формул, ці об’єкти досліджуються окремо за допомогою специфічних методів.

Розроблений метод може бути реалізований в системах, які призначені для проведення інтелектуального аналізу текстової та таблично представленої інформації для ідентифікації подібностей та виявлення неповних дублікатів, зокрема в антиплагіат-системах.

РОЗДІЛ 5. СТРУКТУРНІ СХЕМИ ТА ВІЗУАЛІЗАЦІЯ ПОШУКУ НЕПОВНИХ ДУБЛІКАТІВ, А ТАКОЖ СУПРОВОДУ НАУКОВИХ РОБІТ В СИСТЕМІ ДЕПАРТРАМЕНТУ АТЕСТАЦІЇ КАДРІВ ВИЩОЇ КВАЛІФІКАЦІЇ ТА ЛІЦЕНЗУВАННЯ МІНІСТЕРСТВА ОСВІТИ І НАУКИ УКРАЇНИ

5.1. Інтерфейс системи визначення неповних дублікатів

У системі виявлення неповних дублікатів в електронних документах не визначено локального інтерфейсу, проте система виконує необхідні завдання як сервіс операційної системи, динамічно завантажуючи та перевіряючи необхідні матеріали. Окремо реалізовано веб-додаток, що дозволяє налагодити зв'язок між користувачем системи та системою визначення неповних дублікатів. Цей веб-додаток має власний інтерфейс [6], особливості якого будуть розглянуті в цьому пункті.

Задачі, які можуть бути реалізовані з використанням інтерфейсу системи визначення нечітких дублікатів в електронних документах, такі:

1. Імпорт документів до системи.
2. Визначення параметрів пошуку неповних дублікатів та формування завдання пошуку неповних дублікатів.
3. Групування імпортованих документів за визначеними категоріями та папками.
4. Динамічна зміна стану кожного документа: документ, який імпортується, індексується тощо.
5. Генерація звітів про виконання системою пошуку неповних дублікатів.
6. Відображення фрагментів текстів, що мають подібності. Відображення інформації про загальний відсоток унікальності, розподіл нечітких дублікатів у тексті документа, наявність нечітких дублікатів в зображеннях, таблицях та формулах.

Веб-додаток забезпечує роботу одразу кількох користувачів, розподіляючи документи на перевірку та встановлюючи особливості

налаштування одних користувачів від інших. Це здійснюється за допомогою багатокористувацького режиму роботи системи. У системі визначено такі групи користувачів: адміністратор системи, користувачі першого, другого та третього рівнів. Особливостями прав адміністратора є те, що він має можливість переглядати, редагувати та видаляти будь-який документ або профіль користувача різного рівня. Інші користувачі можуть вносити зміни лише у власні профілі.

Рівень користувача визначає доступ до ресурсів системи. Ресурсом у цьому випадку є процесорний час. Задачі користувачів другого та третього рівня отримують 85% та 15% вільних ресурсів відповідно. Якщо система не має інших задач, то поточна задача отримує 100% ресурсів на її виконання.

Вхід до системи визначення нечітких дублікатів відбувається за адресою: http://dsr.univ.kiev.ua:8080/login. У цьому випадку треба ввести логін і пароль для входу в систему (рис. 5.1).

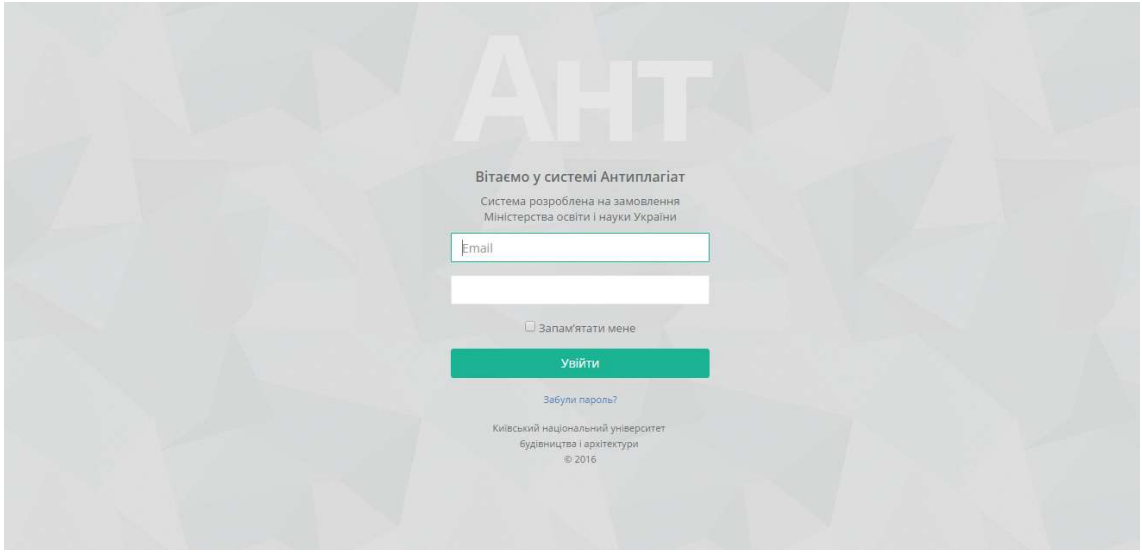


Рисунок 5.1 – Сторінка для доступу до системи визначення нечітких дублікатів

Після проходження перевірки, користувач потрапляє на особисту сторінку. Зокрема, залежно від рівня доступу, він має можливість переглянути документи, що були завантажені раніше для перевірки на наявність неповних дублікатів. На рис. 5.2. вказано такий перелік документів,

який можна оновити або знайти тільки ті документи, які цікавлять користувача на теперішній час, здійснюючи пошук за ключовими словами та назвою.

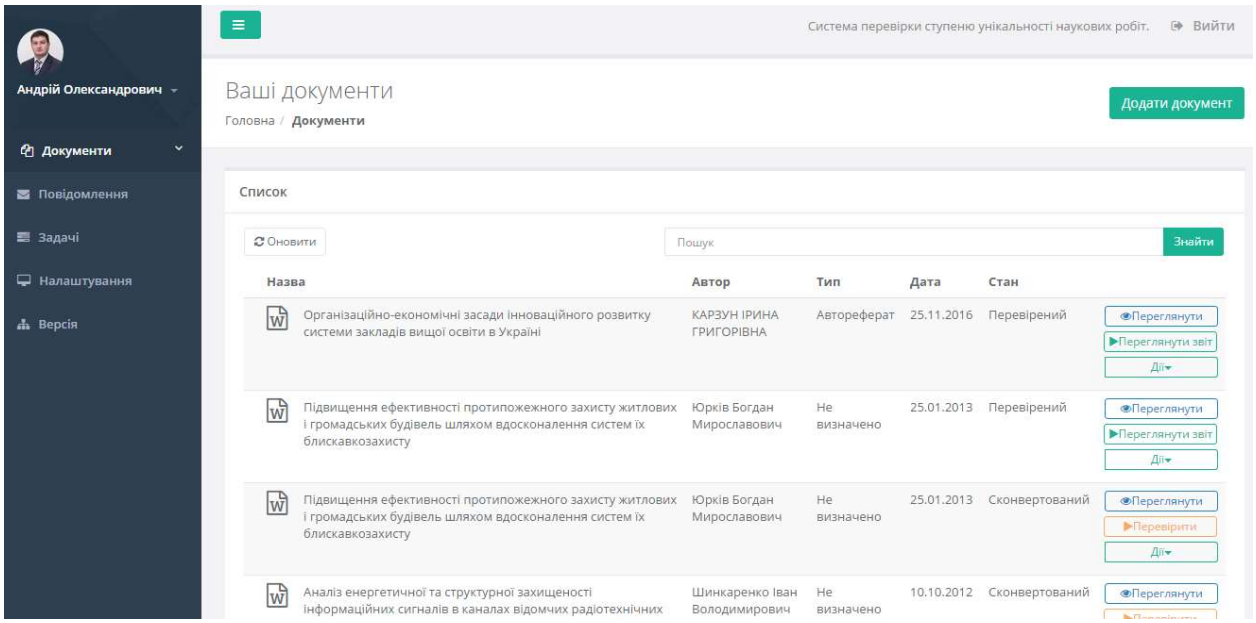


Рисунок 5.2 – Сторінка для перегляду завантажених користувачем документів

Також користувач має можливість обмінюватися повідомленнями з іншими користувачами системи та адміністратором (рис. 5.3).

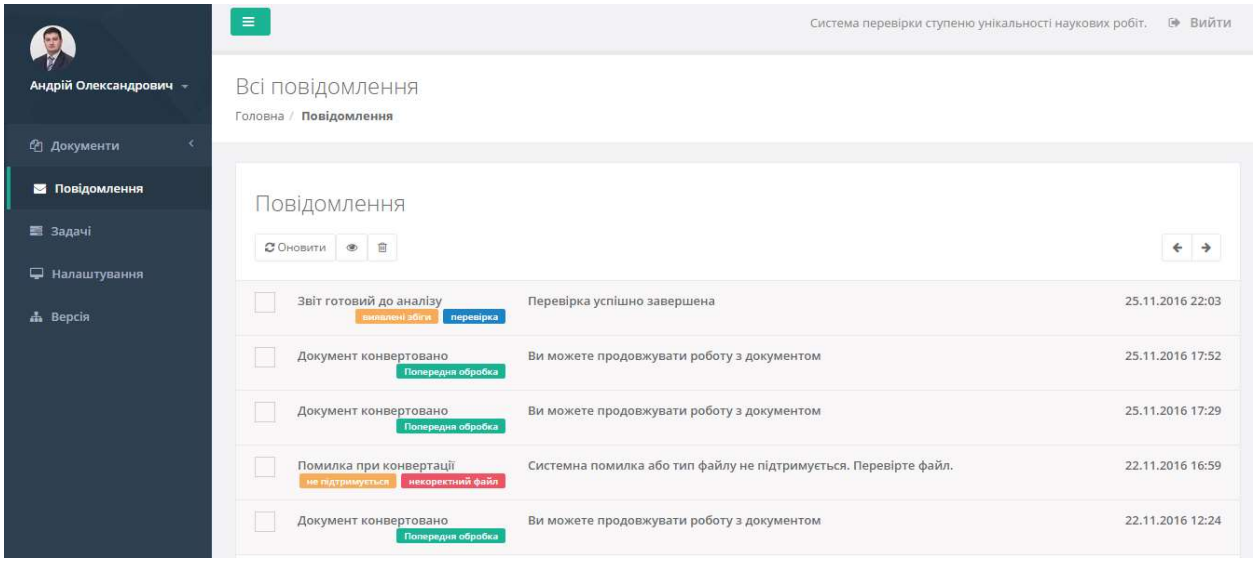


Рисунок 5.3 – Сторінка для перегляду повідомлень користувача

Задачі, які сформовані користувачем, відображаються в переліку на відповідній вкладці. Користувач має можливість перезапустити задачі та оновити загальний перелік (рис. 5.4).

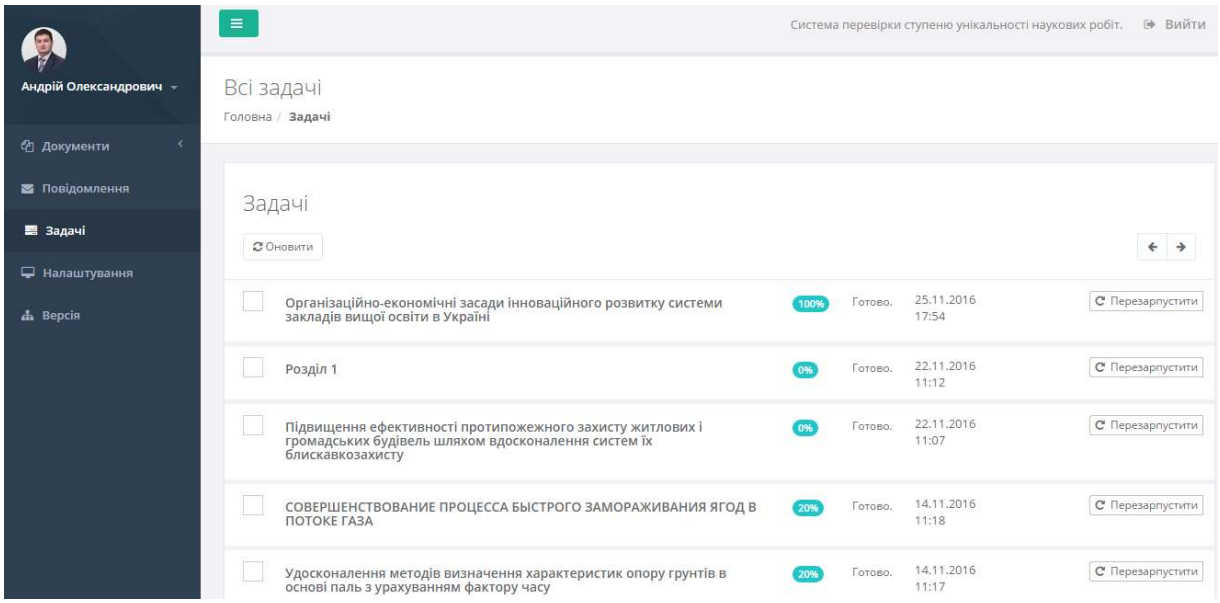


Рисунок 5.4 – Сторінка для перегляду завдань користувача в системі

Також у системі є можливість коригування параметрів пошуку неповних дублікатів: мінімальний збіг, розмір цитати, метод перевірки, точність порівняння. Така інформація визначається на вкладці «Налаштування» (рис. 5.5).

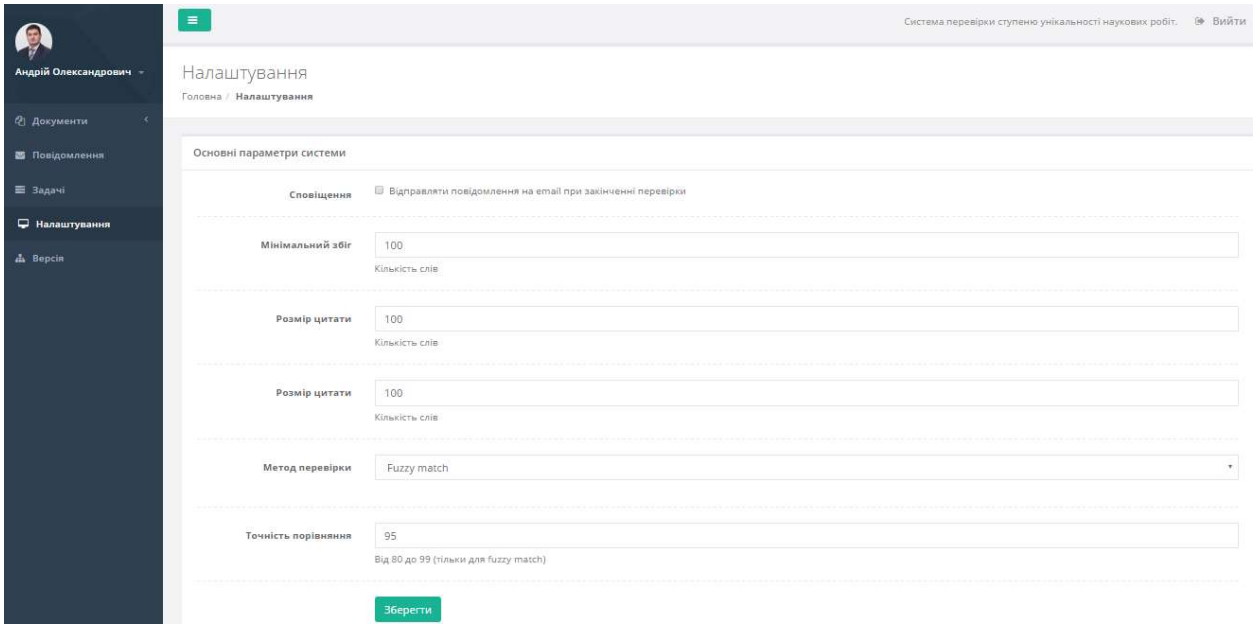


Рисунок 5.5 – Налаштування параметрів визначення неповних дублікатів у системі

Також користувач має можливість переглянути і відредагувати власний профіль (рис. 5.6) та переглянути версію системи з особливостями, що в ній реалізовані (рис. 5.7).

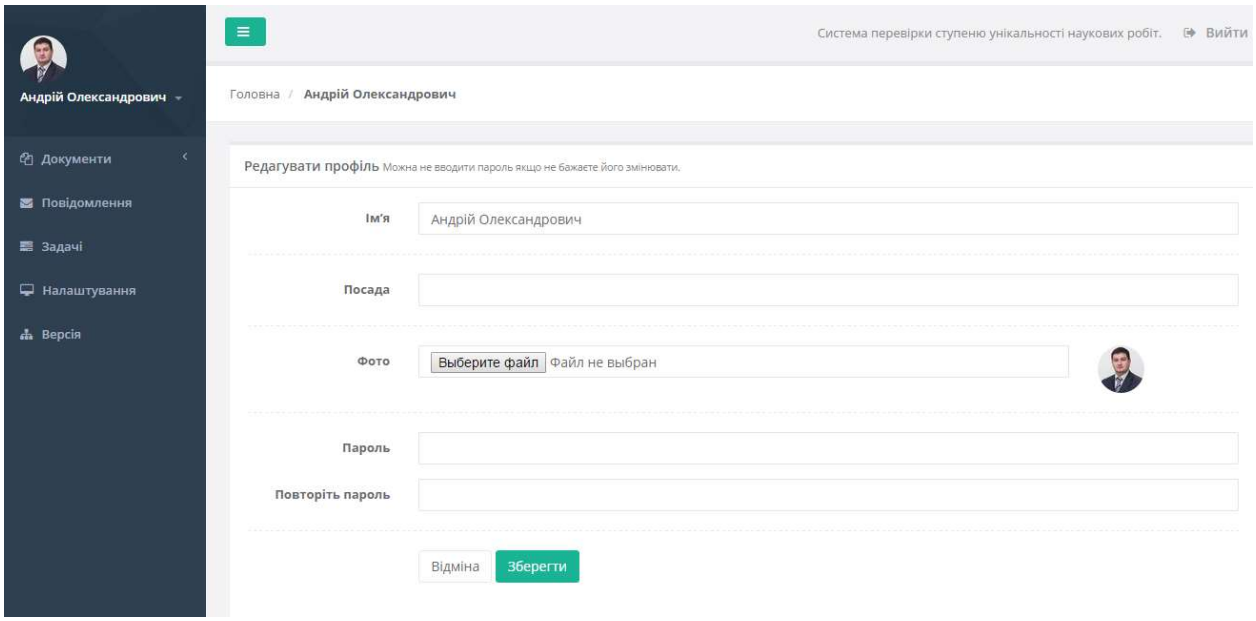


Рисунок 5.6 – Налаштування профілю користувача

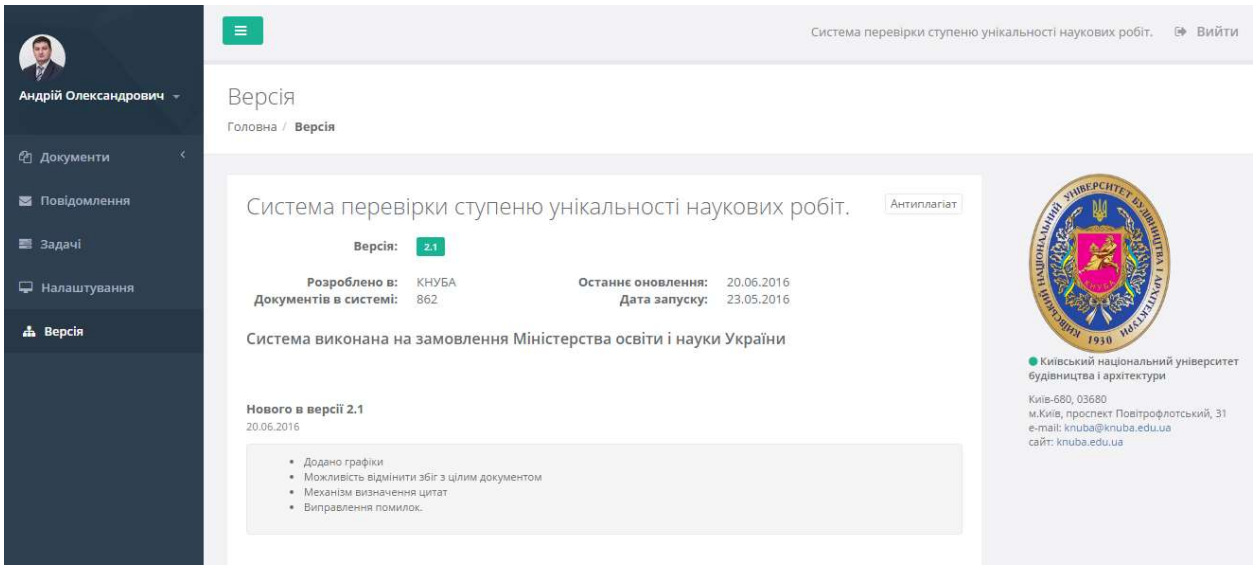


Рисунок 5.7 – Перегляд версії системи

Вікно завантаження користувачем документа для перевірки має вигляд, відображений на рис. 5.8. Тут необхідно вказати назву файлу, його параметри та деякі інші характеристики: автори, галузь знань, організація, від якої

представлено документ тощо. Відображення завантаженого документа вказано на рис. 5.9.

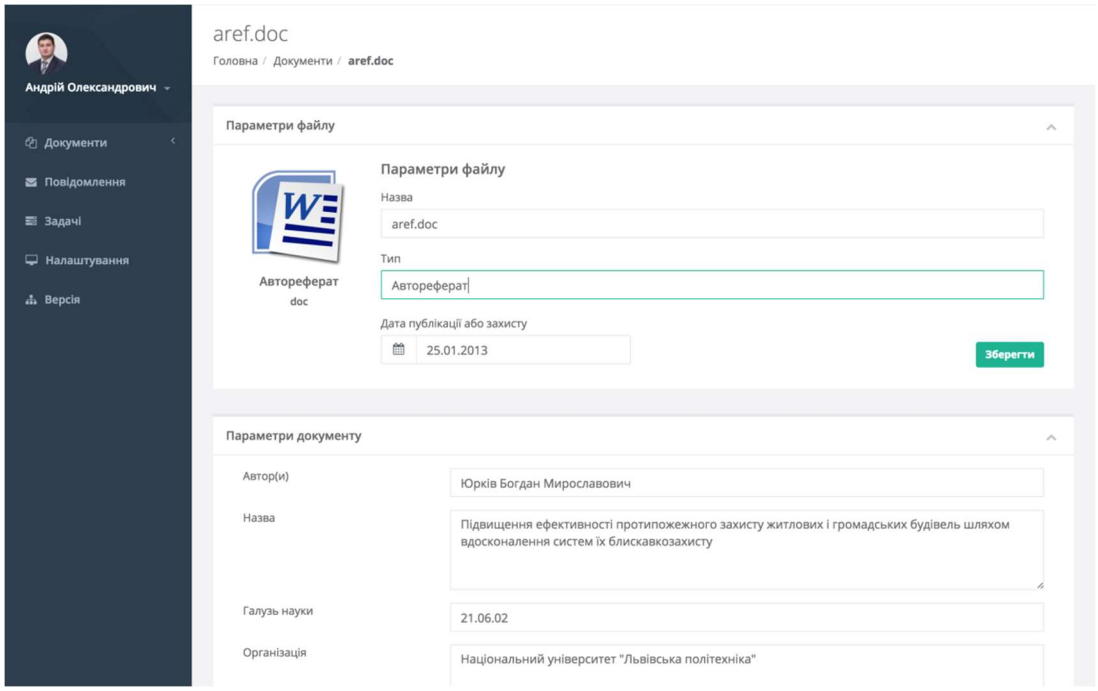


Рисунок 5.8 – Завантаження документа до системи
та визначення його характеристик

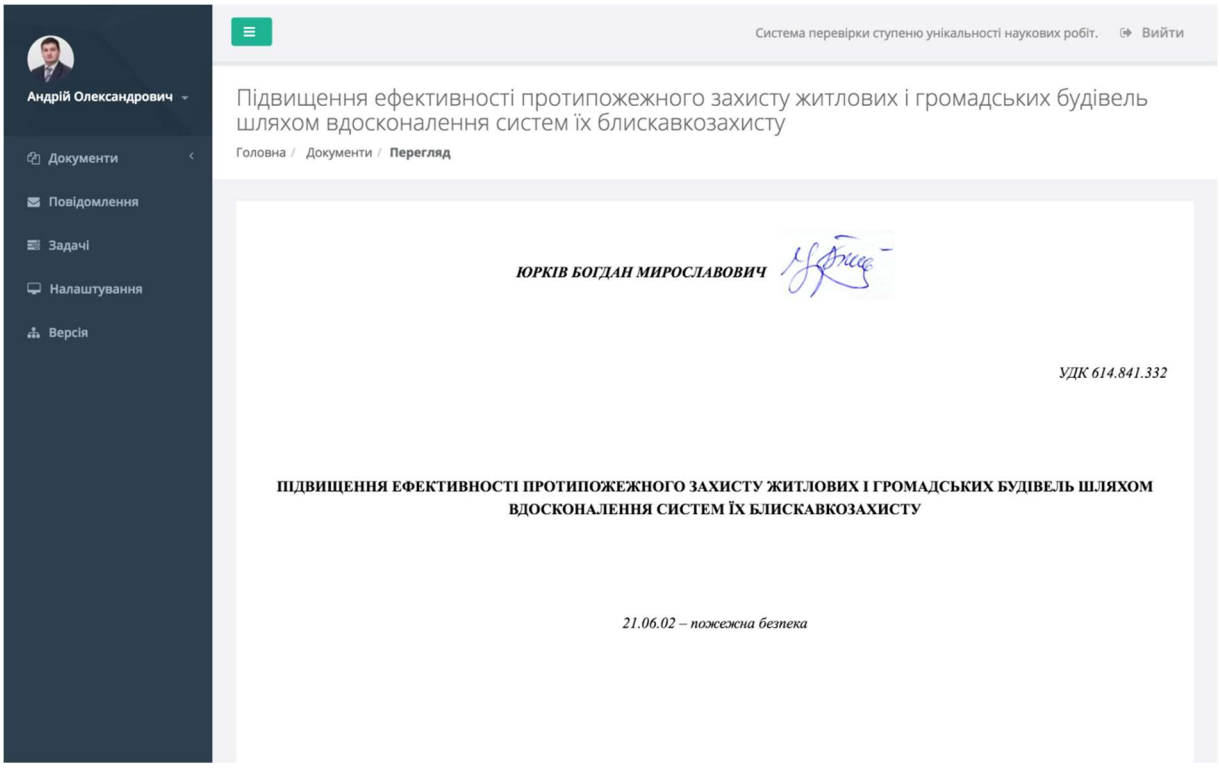


Рисунок 5.9 – Відображення вмісту завантаженого документа

Перегляд завантажених файлів відображається на сторінці (рис. 5.10).

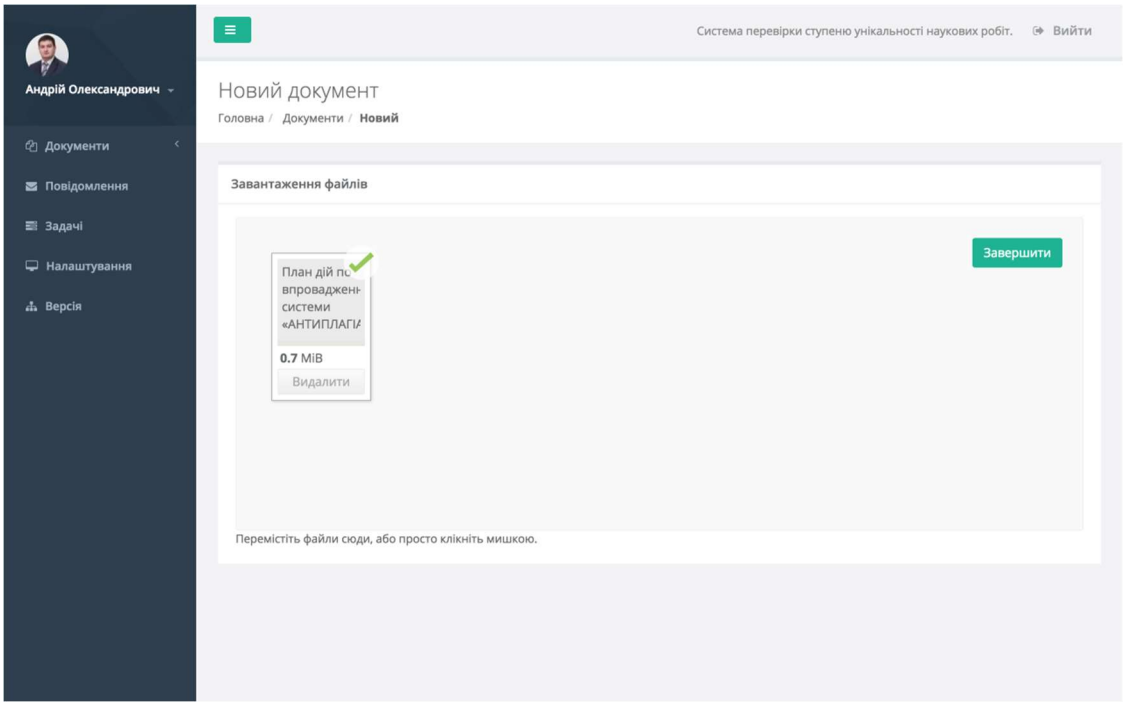


Рисунок 5.10 – Відображення завантаженого документа

Головна сторінка системи визначення ступеня унікальності наукових робіт вказана на рис. 5.11.

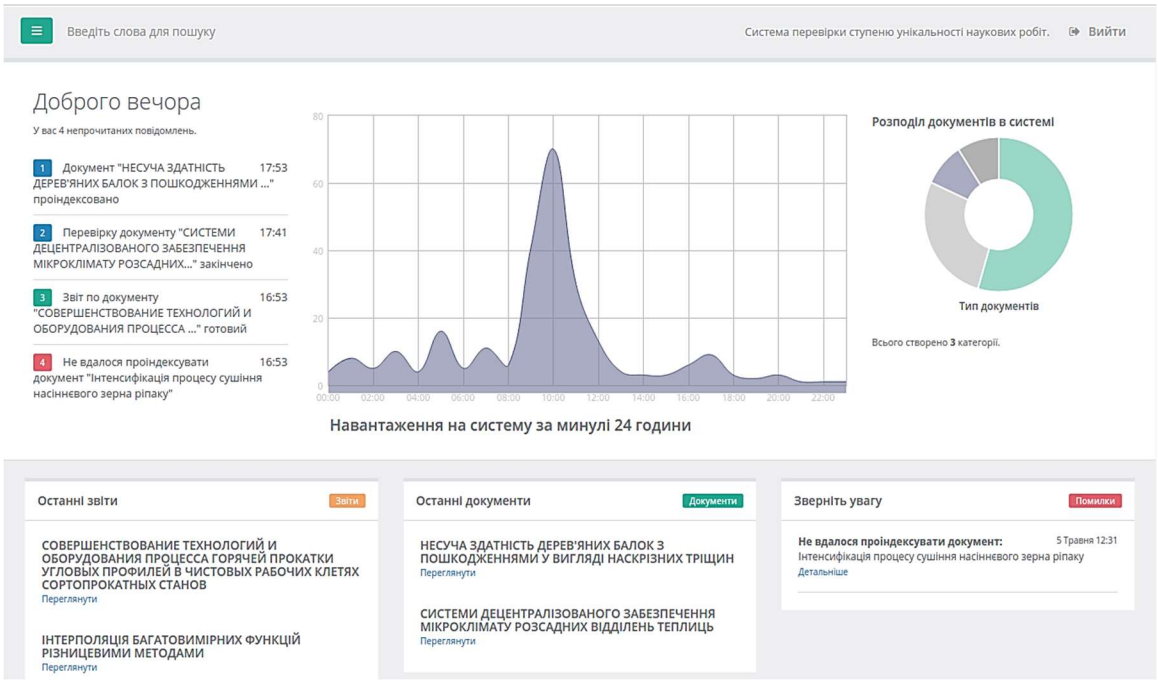


Рисунок 5.11 – Головна сторінка системи визначення ступеня унікальності наукових робіт

Основними розділами, що доступні у розробленому веб-додатку, є:

- файловий менеджер з можливістю групування документів за створеними папками або автоматично створеними секціями, такими як кластер, рік захисту/публікації, галузь наук тощо;
- сторінка повідомлень, де відображаються повідомлення щодо останніх подій у системі, таких як закінчення процесів індексації, перевірки, поява помилок тощо;
- розділ звітів про перевірку документів, де можна переглянути або скачати звіт у вигляді файлу.

Звіт про перевірку може мати кілька виглядів. Узагальнений вигляд відображає тільки відсоткові дані про перевірку та виділяє неповні дублікати безпосередньо в документі (рис. 5.12). Відношення унікального тексту до неунікального відображається у вигляді кругової діаграми (в правій частині сторінки). Аналогічно відображається і розподіл неповних дублікатів по документах, в яких знайшлися запозичення.

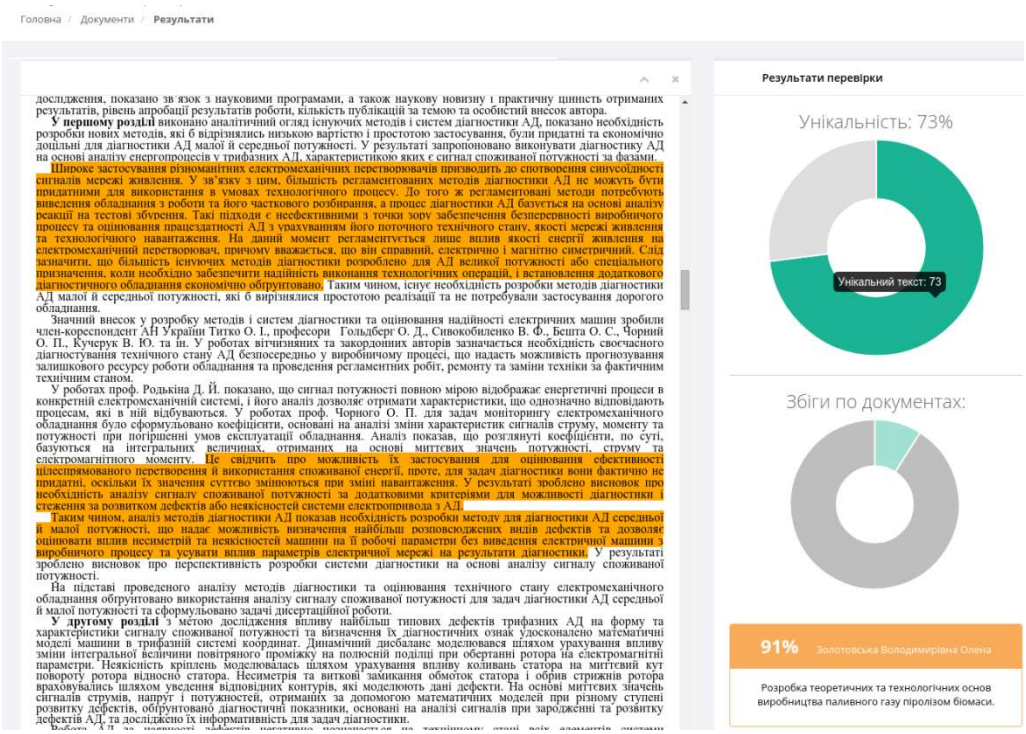
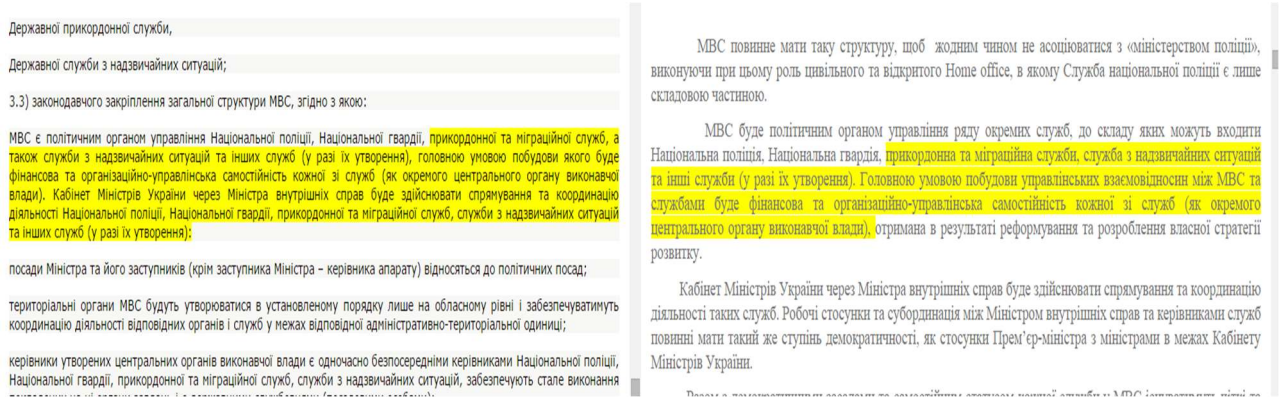


Рисунок 5.12 – Узагальнений вигляд звіту про неповні дублікати

На лівій частині сторінки відображається діаграма унікальності (співвідношення унікального тексту до неунікального) та розподіл запозичень у документах. Діаграма цього розподілу (рис. 5.13) є також досить важливим елементом, вона вказує, скільки контенту було взято з різних документів. На сторінці (рис. 5.14) перегляду неповних дублікатів є можливість побачити візуально фрагменти тексту, де є підозра на запозичення.

Рисунок 5.13 – Діаграма розподілу запозичень у документах


Державної прикордонної служби,

Державної служби з надзвичайних ситуацій;

3.3) законодавчого закріплення загальної структури МВС, згідно з якою:

МВС є політичним органом управління Національної поліції, Національної гвардії, прикордонної та міграційної служб, а також служби з надзвичайних ситуацій та інших служб (у разі їх утворення), головною умовою побудови якого буде фінансова та організаційно-управлінська самостійність кожної зі служб (як окремого центрального органу виконавчої влади). Кабінет Міністрів України через Міністра внутрішніх справ буде здійснювати спрямування та координацію діяльності Національної поліції, Національної гвардії, прикордонної та міграційної служб, служби з надзвичайних ситуацій та інших служб (у разі їх утворення);

посади Міністра та його заступників (крім заступника Міністра – керівника апарату) відносяться до політичних посад;

територіальні органи МВС будуть утворюватися в установленому порядку лише на обласному рівні і забезпечуватимуть координацію діяльності відповідних органів і служб у межах відповідної адміністративно-територіальної одиниці;

керівники утворених центральних органів виконавчої влади є одночасно безпосередніми керівниками Національної поліції, Національної гвардії, прикордонної та міграційної служб, служби з надзвичайних ситуацій, забезпечують стале виконання

МВС повинне мати таку структуру, щоб жодним чином не асоціюватися з «міністерством поліції», виконуючи при цьому роль цивільного та відкритого Home office, в якому Служба національної поліції є лише складовою частиною.

МВС буде політичним органом управління ряду окремих служб, до складу яких можуть входити Національна поліція, Національна гвардія, прикордонна та міграційна служби, служба з надзвичайних ситуацій та інші служби (у разі їх утворення). Головною умовою побудови управлінських взаємовідносин між МВС та службами буде фінансова та організаційно-управлінська самостійність кожної зі служб (як окремого центрального органу виконавчої влади), отримана в результаті реформування та розроблення власної стратегії розвитку.

Кабінет Міністрів України через Міністра внутрішніх справ буде здійснювати спрямування та координацію діяльності таких служб. Робочі стосунки та субординація між Міністром внутрішніх справ та керівниками служб повинні мати такий же ступінь демократичності, як стосунки Прем'єр-міністра з міністрами в межах Кабінету Міністрів України.

Рисунок 5.14 – Режим перегляду збігів у контексті документів

Експерт має можливість за зображеннями неповних дублікатів побачити і відредагувати висновок системи, якщо збіг є помилковим або якщо збіг є стандартною фразою чи визначенням, яке не є плагіатом. Також є можливість додати позначену фразу до переліку стоп-слів, внаслідок чого

база стоп-слів динамічно поповнюється і дає змогу системі визначати збіги більш точно.

Реалізований інтерфейс користувача простий, зрозумілий і гнучкий. Він дозволяє швидко орієнтуватися користувачу в деталях роботи системи і без особливих зусиль проаналізувати завантажені електронні документи та визначити неповні дублікати в тестах, графічних зображеннях, математичних формулах і таблицях.

5.2. Принципи роботи програмних модулів системи визначення неповних дублікатів з урахуванням їх візуалізації

Візуалізація програмних модулів системи визначення нечітких дублікатів є важливим елементом забезпечення її максимальної доступності для користувачів. Саме тому було запропоновано і реалізовано архітектуру системи таким чином, щоб взаємодія з користувачами відбувалася через мережу Інтернет. Крім цього, слід зазначити, що система визначення неповних дублікатів може ефективно працювати як на одному сервері, так і розділена на кілька серверів для оптимізації процесу пошуку. За допомогою глобальної мережі Інтернет налагоджується необхідний зв'язок між основними компонентами системи визначення неповних дублікатів: базою даних, сервером обробки даних, а також веб-сервером.

На рис. 5.15 показана загальна архітектура системи визначення неповних дублікатів. Причому інтерфейс користувача, який реалізується через веб-додаток та пов'язаний з відповідним веб-сервером, посідає в цій архітектурі одне з основних місць. Це пов’язано з тим, що візуальні компоненти системи дозволяють легко здійснювати управління діями системи, бачити статус перевірки, індексації, завантаження документів тощо [6].

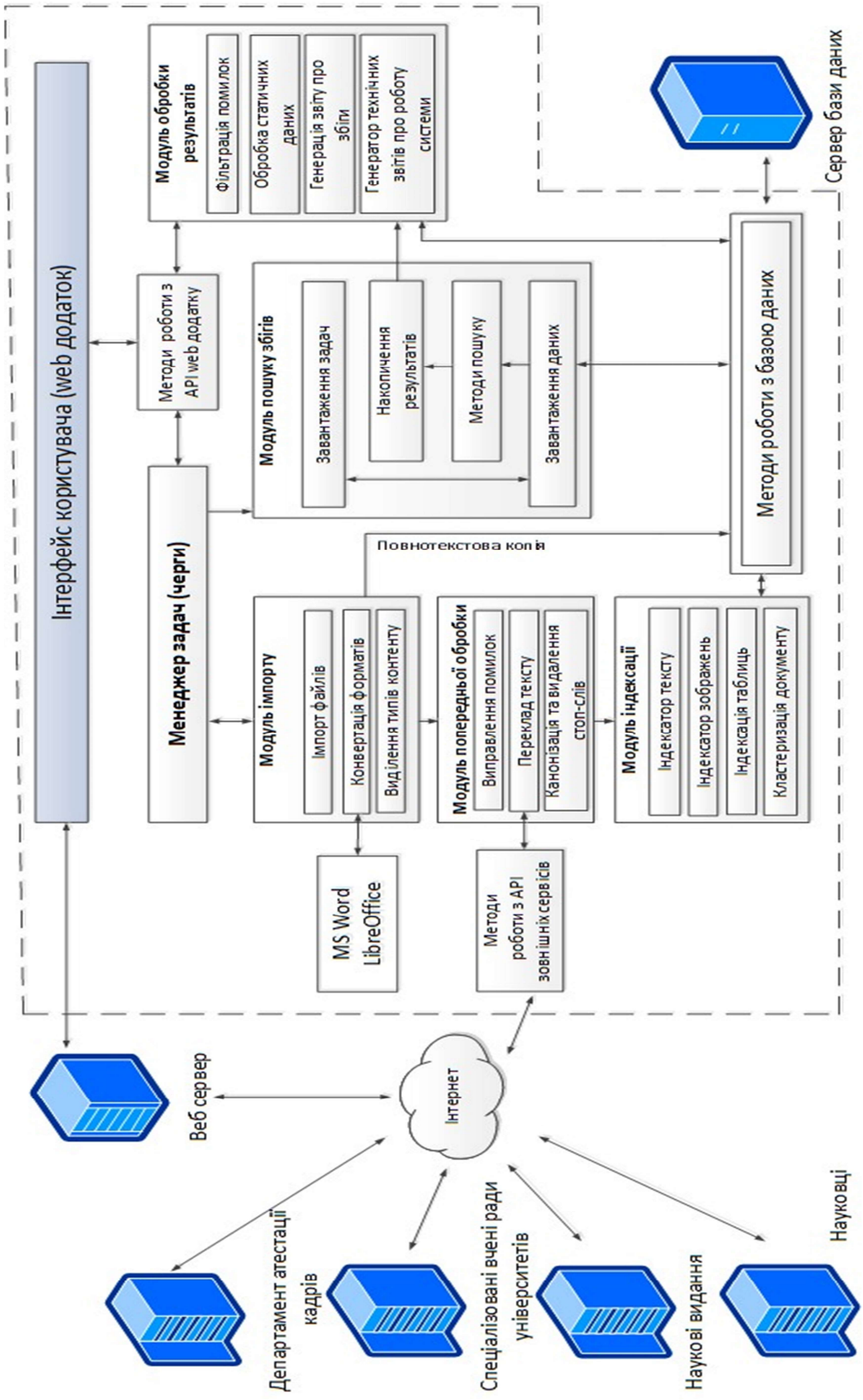


Рисунок 5.15 – Архітектура системи визначення неповних дублікатів

Розглянемо систему імпортування файлів до системи визначення неповних дублікатів. Під час роботи системи часто виникає необхідність працювати з файлами, які мають різний формат. Зрозуміло, що різні файли мають різну структуру і здійснювати опрацювання файлів різного формату складно. Саме тому важливо конвертувати всі файли, які завантажуються до системи, до єдиного формату.

Приведення завантажених файлів відбувається з втратами, оскільки складно коректно відобразити всі графіки, формули, таблиці та діаграми, що містяться в оригінальному файлі. Тому важливо відсортувати інформацію в документі за пріоритетами. Інформацію, яка більш важлива для пошуку неповних дублікатів, треба залишити та перетворювати в першу чергу, іншу – перетворити до примітивного вигляду. З цих міркувань, нам необхідно сформувати вимоги до документа, який перетворюємо: він не повинен мати складної структури, в ньому має підтримуватися форматування текстових, табличних даних, графічних зображень.

Візуалізація завантаження документів до системи визначення нечітких дублікатів реалізована за технологією Drag and Drop. На рис. 5.16 показано головне вікно для завантаження нових документів. Результати завантаження вказані на рис. 5.8 – 5.10.

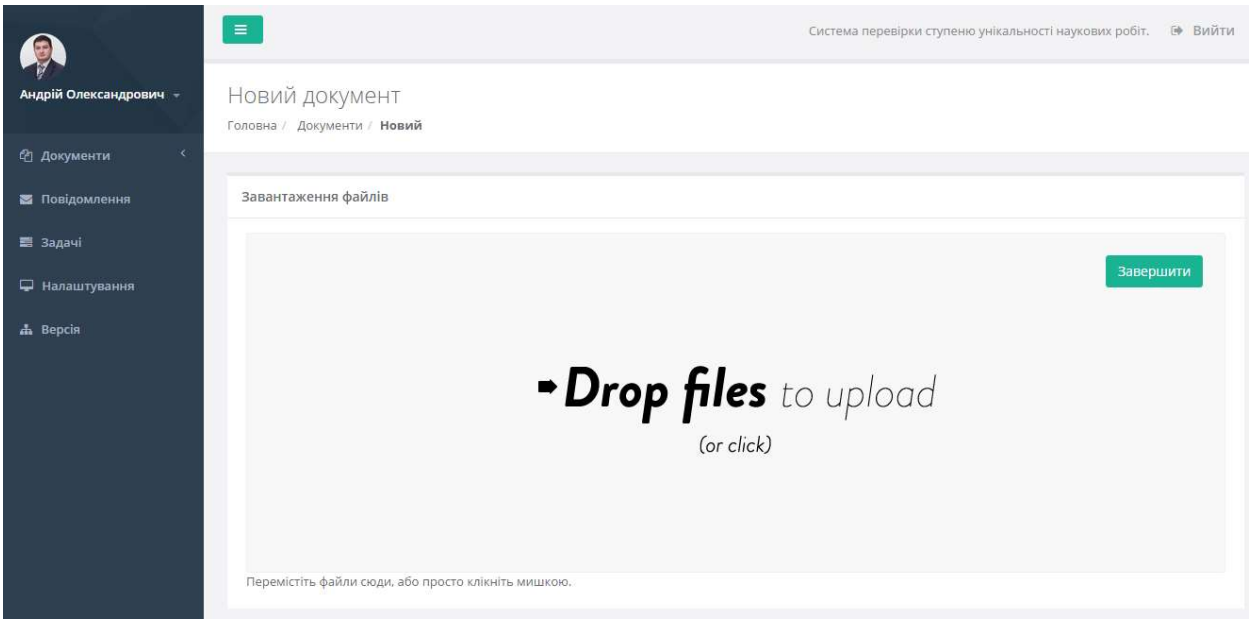


Рисунок 5.16 – Вікно завантаження нового документа до системи

У системі було визначено, що основним форматом, в який здійснюється конвертування документів, є формат HTML. Кожен рядок файлу HTML є тегом, який має вигляд слова зі знаками «<» та «>» по краях ( ). Знаки розподіляються на одинарні та парні (один вказує на початок певного контенту, інший – на його кінець).

Зображення в HTML, як правило, формуються окремими файлами та зберігаються окремо, а до документа вставляється посилання на цей файл. Проте у HTML є також можливість відображати файл зображення в тексті, при цьому використовується кодування BASE64 [64]. Окремі теги також існують для форматування таблиць, математичних формул тощо.

Для конвертації документів у формат HTML було вирішено використовувати наявне програмне забезпечення. Конвертацію файлів MS Office краще проводити за допомогою власне офісного пакету Microsoft, але за потреби може бути використано безкоштовний офісний пакет LibreOffice.

Імпорт документу має ряд етапів:

1. Отримання нового файлу і збереження його на магнітному диску.
2. Визначення формату вхідного файлу.
3. Визначення наявного конвертера для отриманого формату.
4. Запуск конвертера та отримання HTML-документа.
5. Очищення HTML-документа від зайвої інформації (стилів, скриптів).
6. Виділення типів контенту документа.
7. Збереження HTML-документа в базі даних.

Для конвертації у системі використовується офісний пакет LibreOffice та утиліта pdf2htmlEX. Було розроблено спосіб для перетворення звичайних текстових файлів у HTML. Таким чином, модуль імпорту документів отримав можливість імпортувати такі формати файлів: DOC, DOCX, RTF, TXT, PDF, ODT DOT та інші менш поширені формати.

Розглянемо схему модулів попередньої обробки та індексації імпортованих документів. Модуль попередньої обробки призначено для розбору контенту документів, які завантажуються до системи, сортування даного контенту тощо. Модуль індексації використовується для перетворення контенту документа на індекс, тобто на своєрідну структуру, яка дозволяє без суттєвих втрат ресурсів зберігати документ у системі, а також швидко здійснювати пошук неповних дублікатів. Індексація електронного документа здійснюється тільки один раз, а пошук неповних дублікатів, де за основу береться даний документ, проводиться неодноразово. По суті, під час індексації документа здійснюються тільки дві операції: обрахування хешів та фрагментація.

Першим етапом в обробці документа є виправлення помилок: заміна літер, наявність невидимих символів, друкарські помилки. Така обробка здійснюється на основі бібліотеки Hunspell.

Потім, у випадку, якщо документ містить текстовий контент, визначається мова тексту. Якщо мова відрізняється від стандартної в системі, то здійснюється переклад тексту. Засоби перекладу тексту включають в себе засоби локального перекладу та перекладу за допомогою онлайн систем (використовуючи API). Як локальна система перекладу використовується система Pragma 6, а як системи онлайн-перекладу – перекладачі Google, Яндекс та Bing.

Після перекладу тексту відбувається вилучення стоп-слів та стоп-фраз і канонізація. Канонізований текст поділяється на шингли і створюються елементи індексу за допомогою локально-чутливого хешування. Аналогічно текстовим даним документа, текстові дані таблиць також проходять всі етапи попередньої обробки та канонізації, і тільки після цього відбувається фрагментація таблиці та створення її індексу за допомогою локально-чутливого хешування.

У випадку наявності в контенті графічних зображень, вони досліджуються окремо. Для цього створюється віддзеркалена копія. Оригінал

і копія зображення проходять фрагментацію за визначеними ключовими точками. Для кожного отриманого елемента визначається власний кут повороту та будується перцептивний хеш. Для обраховування власного кута повороту було вирішено обертати не картинку, а сам сектор, в якому вираховується середній колір положення. Таким чином, відбувається економія ресурсів системи, оскільки поворот зображення є доволі затратною операцією. Також, перед поворотом, необхідно виконати деякі перетворення, оскільки дублікати зображень можуть мати різне розширення, пропорції та інші змінені параметри, що може вплинути на кінцевий результат.

Після фрагментації, визначення кутів повороту та обрахування перцептивних хешів отримуємо множину хешів різних фрагментів або варіантів зображення. Ця множина зберігається в базі даних і вважається індексом зображення.

Розглянемо підходи до перекладу тексту та проекції координат перекладеного тексту на початковий. При індексації електронного документу необхідно виконати розпізнавання мови тексту. Якщо мова тексту не базова, то зробити переклад. Базовою мовою в системі є українська, оскільки більшість документів, що будуть завантажуватися до системи – україномовні.

Для перекладу текст фрагментується на короткі уривки з одного чи кількох речень до 500 символів в одному фрагменті. Це необхідно для проекції позицій перекладу на оригінальний текст.

Базуючись на методі співставлення оригінального і перекладеного тексту, модуль має такі функції:

1. Поділ тексту на речення.
2. Формування запиту.
3. Переклад з використанням API однією з систем перекладу.
4. Проекція позицій оригінального тексту на перекладений.

Після проведення аналізу документа на наявність неповних дублікатів користувач може продивитися статистику: коефіцієнт унікальності

документа та документи, що зберігаються в базі системи та мають неповні дублікати з вхідним документом (рис. 5.17). Також окремо визначається візуалізація розподілу збігів за документами (рис. 5.18).

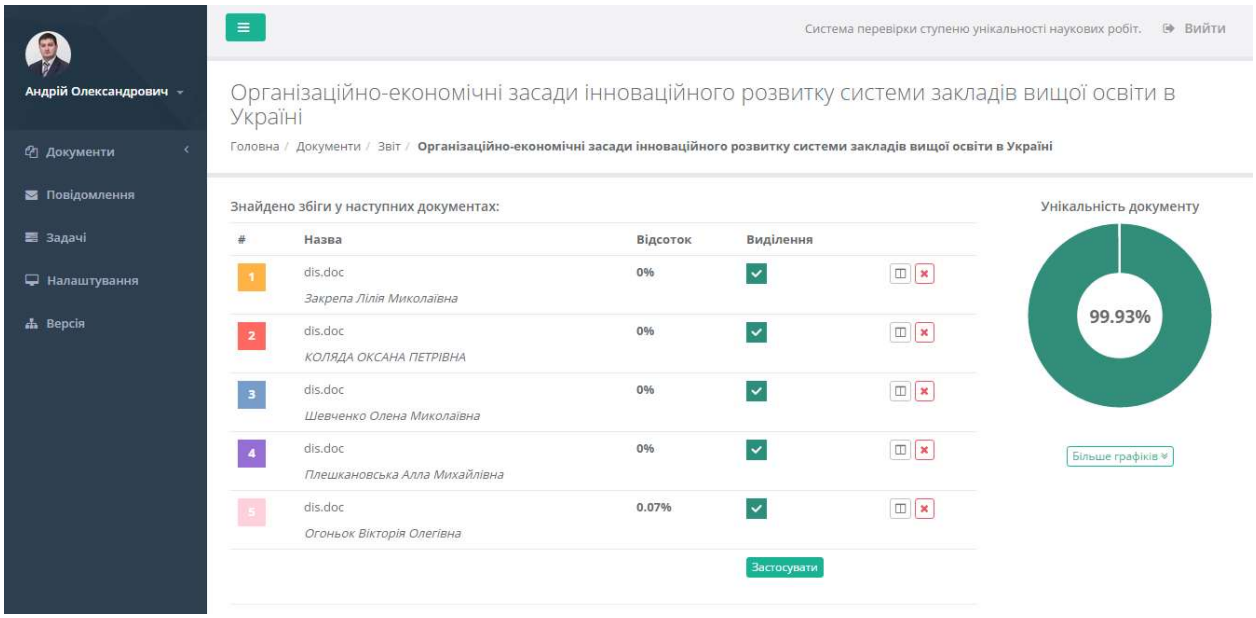


Рисунок 5.17 – Візуалізація інформації про результат перевірки документа на наявність неповних дублікатів

Рисунок 5.18 – Візуалізація інформації про розподіл збігів за документами

Візуалізація інформації у системі має важливе значення для користувача, який може легко зорієнтуватися та здійснити перевірку документів на наявність неповних дублікатів. Також візуалізація є важливим елементом відображення результатів роботи системи: результатів імпорту документів, оновлення, індексації, перевірки на неповні дублікати тощо.

В розробленій системі пошуку збігів виконується перевірка по різних елементах документа, таких як текст, таблиці, формули та зображення. Відповідно є декілька режимів переглядів збігів, що дозволяють отримати загальну інформацію про унікальність документу, та переглянути конкретні знайдені збіги. Для запуску перевірки документу, він має бути завантажений в систему і конвертований. Конвертація відбувається автоматично відразу після завантаження, але тільки якщо формат файлу підтримується системою. На даний момент підтримуються файли форматів "txt", "doc", "docx", "pdf", "xls", "xlsx" та "odt". Користувач системи має змогу запустити перевірку з розділу системи "Всі документи" (рис. 5.19).

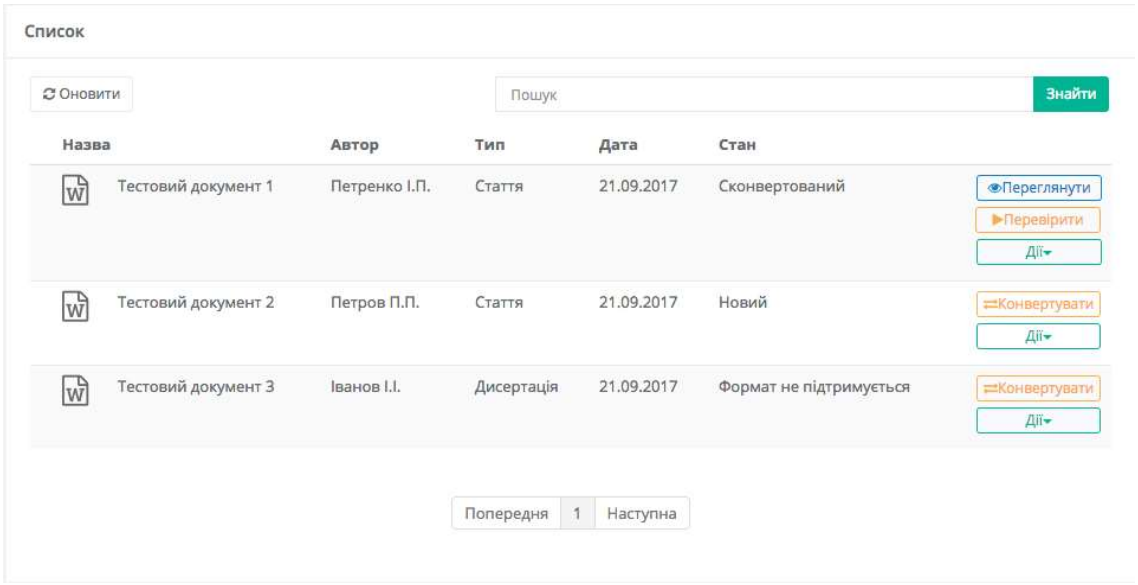


Рисунок 5.19 — Список документів, завантажених до системи

На рисунку 5.19 зображено три завантажені документи з різними статусами. "Тестовий документ 1" завантажено і конвертовано успішно, тому користувачу доступна опція "Перевірити". "Тестовий документ 2" система не конвертувала, можливо виникла якась помилка і автоматична конвертація не була запущена. Користувач може запустити конвертацію вручну за допомогою опції "Конвертувати". Формат тестового документу 3 не підтримується системою, як видно з поля "Статус", користувач може видалити даний документ або запустити конвертацію ще раз через певний

проміжок часу. Система оновлюється, додаються нові формати, тому цілком ймовірно, що в майбутньому даний формат також буде підтримуватися.

Для перевірки обраного документу необхідно натиснути на кнопку "Перевірити" і завдання на перевірку буде додано в чергу. Відслідкувати за процесом виконання перевірки користувач системи може перейшовши в розділ "Задачі".

На рисунку 5.20 зображено 2 задачі, одна з яких виконана, а інша в процесі виконання. Загалом, є кілька стадій перевірки, таких як індексація (якщо не була виконана для даного документу раніше), швидкий пошук для визначення кола схожих документів, детальний пошук для визначення точного місця збігів, обробка даних для розрахунку відсотків збігів та генерації звіту. Після того як перевірка закінчена, опція "Перевірити" (рис. 5.19) зникає і на її місці з'являється опція "Переглянути звіт".

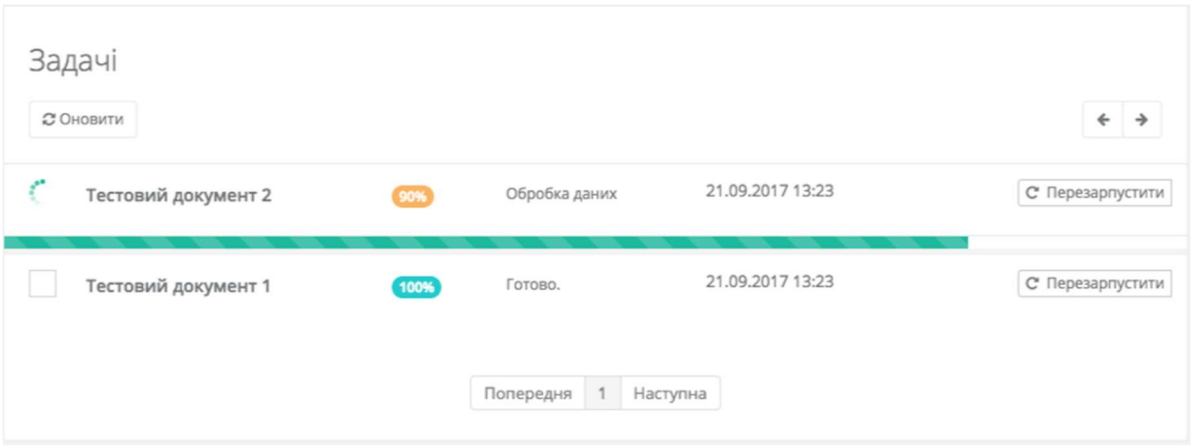


Рисунок 5.20 — Список поточних задач в системі

Користувачам системи доступні для перегляду декілька форм звіту. Натиснувши "Переглянути звіт", користувач потрапляє на загальний звіт по документу. Дану форму звіту можна умовно розділити на п'ять частин. У верхній частині сторінки користувач бачить список документів, у яких знайдено збіги з поточним документом, загальний відсоток унікальності та опції роботи з отриманими даними (рис. 5.21).

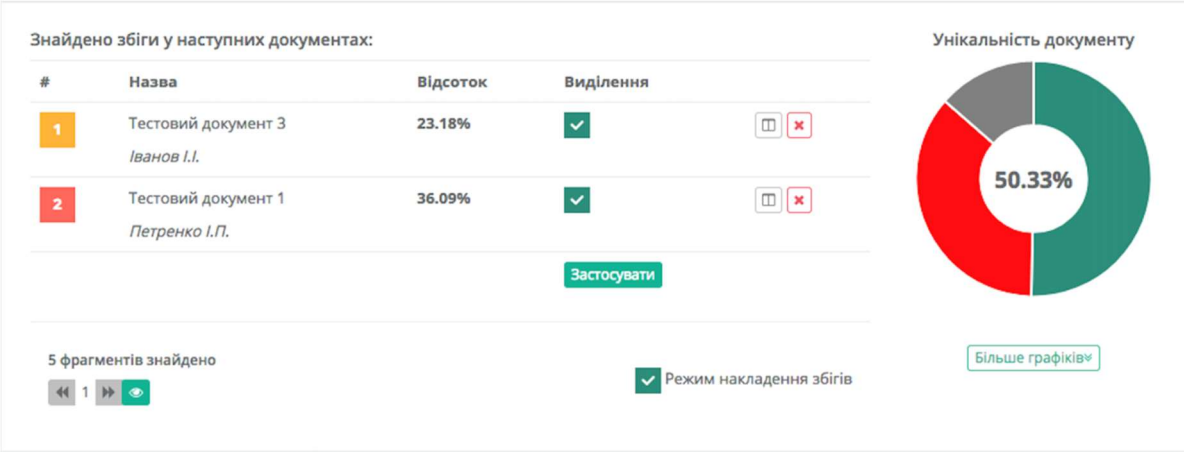


Рисунок 5.21 – Загальний звіт. Основні дані

Кожному з документів, у яких знайдено збіги, присвоюється колір, який використовується для виділення збігів у поточному документі. Про це детальніше буде описано нижче. Також ми можемо бачити відсоток збігу з поточним документом по кожному зі знайдених позицій. В цій же таблиці справа є дві кнопки. Кнопка зі значком "Книги" відкриває режим перегляду збігів з даним документом. Кнопка зі значком "хрестик" видаляє всі збіги поточного документу з обраним і перераховує статистику. Ця опція необхідна для видалення зі звіту збігів поточного документа з роботами того ж самого автора. Самоцитування не вважається запозиченням, але системі не завжди вдається правильно визначити авторів роботи. Діаграма "Унікальність документу" відображає співвідношення унікального та запозиченого або цитованого тексту. Зелений колір на діаграмі відповідає унікальному тексту, темно-сірий — цитатам і червоний — запозиченням. Загалом унікальність документу в нашому прикладі становить 50,33%.

Оскільки збіги можуть накладатися (один і той самий фрагмент тексту знайдений в кількох різних документах), то сумарний відсоток збігів по знайдених документах не дорівнює відсотку показаному на діаграмі. На схемі 5.22 показано приклад розрахунку загального відсотку збігів. В наведеному прикладі перевіряється "Документ В" і він збігається з "Документом Б" на 40% та з "Документом А" теж на 40%.

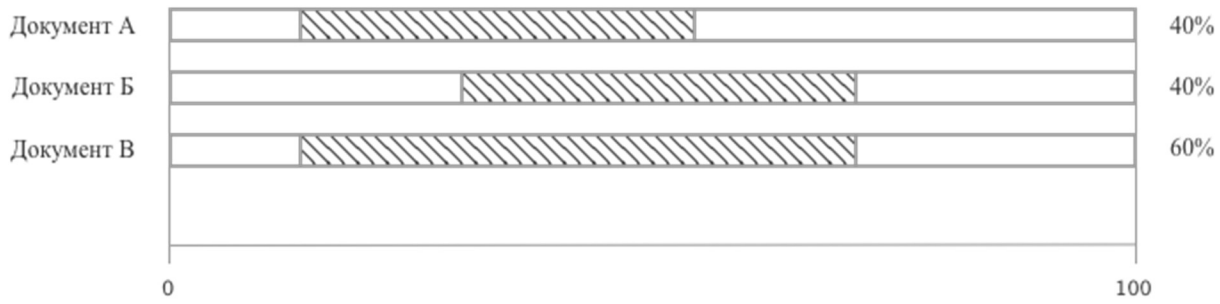


Рисунок 5.22 – Розрахунок унікальності документу

Сумарний відсоток збігів при цьому дорівнює 60% і залежить від того наскільки збіги в документах "А" та "Б" об’єднуються. Одиницею виміру тексту для визначення даних відсотків є слова.

На рис. 5.23 можна побачити інші компоненти які використовуються для фільтрування чи навігації між збігами. Зокрема, в правому нижньому кутку є перемикач збігів та інформація про кількість фрагментів загалом. Кнопка "Більше графіків" вмикає відображення додаткових діаграм (рис 5.21). Таких діаграм дві: "Розподіл збігів по довжині документу" та "Розподіл збігів по документах".

Рисунок 5.23 – Загальний звіт. Додаткові дані

Розподіл збігів по довжині документу показує щільність фрагментів, що збіглися на певному проміжку документа. Ця інформація може бути корисна для перевірки строго структурованих робіт, наприклад дисертацій. В дисертаціях в першому розділі практично завжди аналізуються сторонні джерела за тематикою роботи, тому там можна допустити велику концентрацію запозичень.

Розподіл збігів в документах показує відсоток збігів, що знайшлися у різних документах. У нашому прикладі (рис. 5.23) збігів, що знайшлися в тестовому документі 3, на 30% більше ніж збігів, що знайшлися в тестовому документі 1, при загальній неунікальності поточного документу в 36,9%.

Основну частину загального звіту займає перегляд текстових збігів (рис. 5.24). В даній формі відображається весь текст документу з урахуванням форматування.

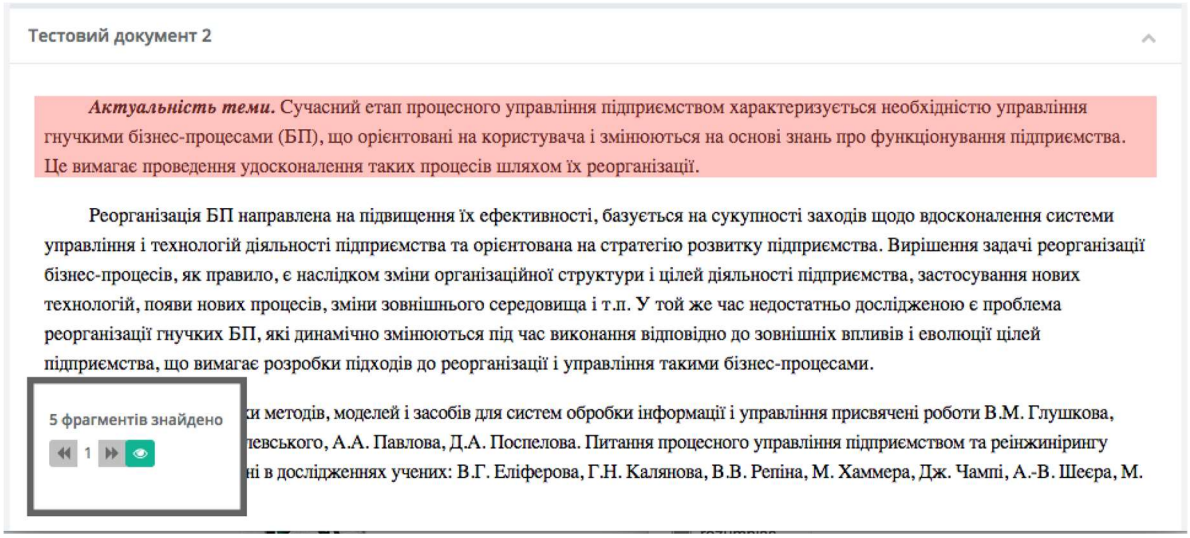


Рисунок 5.24 — Загальний звіт. Перегляд текстових збігів

В нижньому лівому куті рисунку 5.24 зроблена вставка з рисунку 5.19, тому що основна задача наведеного компоненту саме оперувати з текстовими збігами. При натисненні на кнопку зі значком "око" область перегляду тестових збігів прокручується до поточного збігу всередині документу. Всі збіги пронумеровані, номер збігу також вказаний на компоненті управління переглядом збігів. За допомогою кнопок з іконками "вперед" і "назад" можна перемикати поточний збіг, при цьому область з текстом буде автоматично прокручуватися до початку поточного збігу. В даному режимі текст в збігах виділяється кольором відповідно до присвоєного кольору документа (рис. 5.24).

Збіги можуть накладатися один на одного, в такому випадку вони будуть виділені одразу кількома кольорами (кольори будуть змішані). Для того, щоб оцінити збіги з одним документом і уникнути перекриття з іншими, можна вимкнути підсвічування для одного чи кількох знайдених

документів. Для цього необхідно зняти галку в графі "Виділення" і натиснути "Застосувати". Кількість збігів також буде перерахована, буде відображена кількість збігів з документами для яких увімкнено "Виділення". В загальному звіті також наведені збіги за зображеннями (рис. 5.25) .

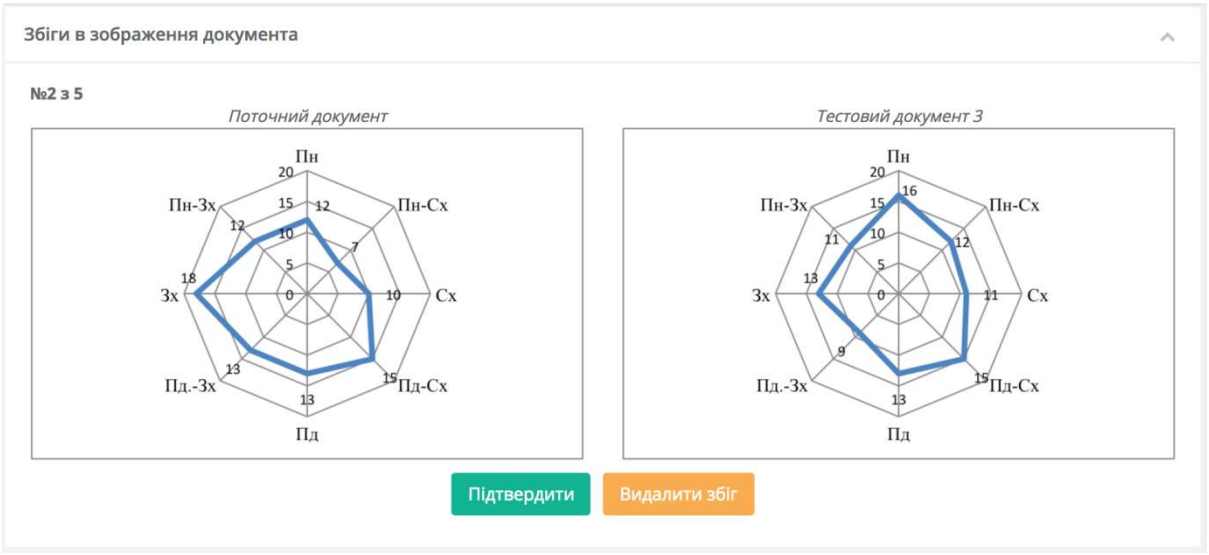


Рисунок 5.25 – Загальний звіт. Перегляд текстових збігів

В даній частині звіту наведені попарно розташовані зображення, які система вважає збігами. Оскільки для визначення копій використовується перцептуальне хешування, існує ненульова ймовірність помилки. Тому знайдені копії не враховуються при підрахунку загального відсотку унікальності документу. Дана інформація повинна допомагати експерту, що працює з системою зробити остаточний висновок щодо унікальності даної роботи. У оператора системи також є можливість вплинути на результати перевірки, скоригувавши їх. В даному випадку під кожною парою зображень є опції "Підтвердити" або "Видалити збіг". В разі підтвердження система більше не буде відображати ці опції і видалити збіг стане неможливо. Також оператор може видалити збіг зі звіту, якщо вважає, що система припустилася помилки. Дану операцію також неможливо відмінити, тому користувач системи повинен бути впевнений у своєму рішенні.

Аналогічно до роботи зі збігами в графічних зображеннях, відбувається відображення і робота зі збігами в формулах (рис. 5.26).

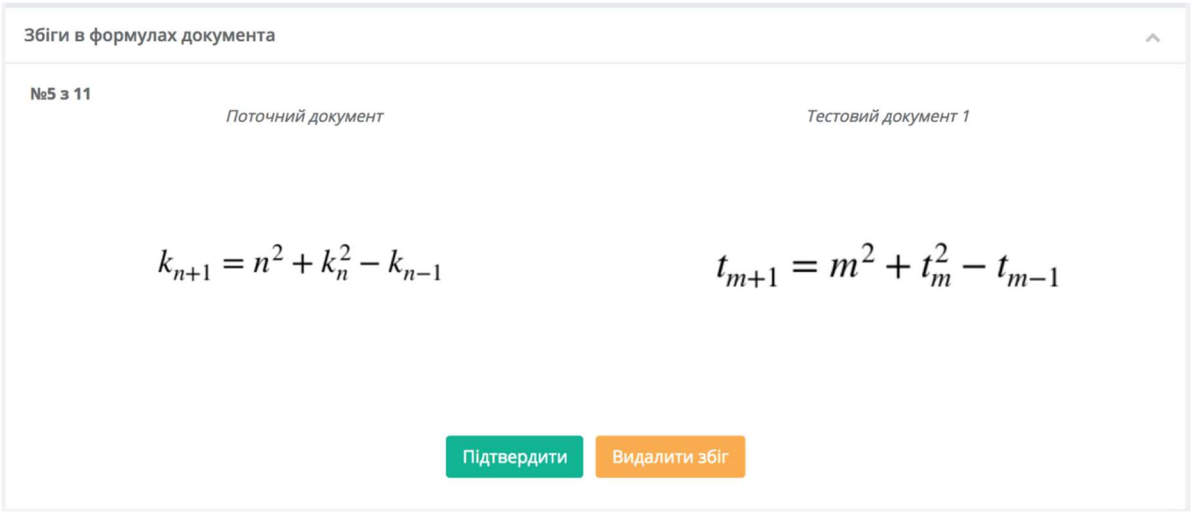


Рисунок 5.26 – Загальний звіт. Перегляд збігів у формулах

Для пошуку збігів в формулах, вони конвертуються в LaTex. Далі аналізується структура формули не звертаючи уваги на літери, які було використано для означення значень в формулі. Таким чином однакові за структурою формули будуть визначені системою як збіги. Експерт при роботі з системою має можливість підтвердити або видалити знайдений збіг. Даний метод роботи з формулами не використовується для коротких формул, тому при аналізі структури вираховується і "складність" формули. Складність формули визначається загальною кількістю математичних операцій (додавання, множення, піднесення до степеня тощо), виконаних у формулі. Також береться до уваги загальна кількість складових формули, наприклад, в формулі може бути кілька знаків дорівнює. Як і у випадку зі збігами зображень, дані збіги не впливають на загальний відсоток унікальності документу і служать допомогою експерту у прийнятті кінцевого рішення.

Інший спосіб перегляду результатів пошуку збігів — попарне порівняння документів. Цей тип звіту можна переглянути для кожного зі знайдених документів. Щоб перейти до попарного порівняння, необхідно натиснути на значок "Книга" (рис. 5.21) навпроти обраного для порівняння документу.

При попарному порівнянні документів користувач системи може працювати з текстовими збігами (рис. 5.27).

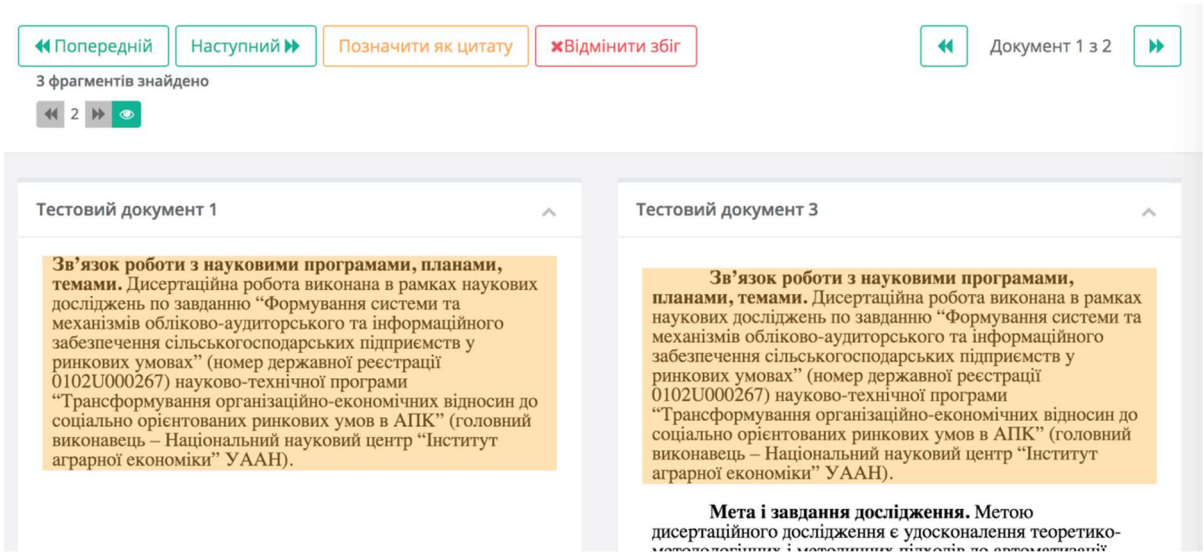


Рисунок 5.27 – Попарне порівняння документів

В даному типі звіту документи розташовані поряд один з одним, що дозволяє наглядно перевірити текстовий збіг. При натисканні кнопок "Попередній", "Наступний" відбувається перехід до попереднього або наступного фрагменту відповідно. При цьому контейнери обох документів прокручуються до місця початку звіту. В даному режимі користувачу системи доступний повний текст обох документів. Позиція контейнерів не зафіксована, є можливість прокручувати один або інший документ для отримання додаткових даних. Повернутися до поточного збігу можна натиснувши кнопку зі значком "Око", тоді обидва контейнери будуть знову прокручені до позиції початку збігу.

У випадку, якщо система не змогла розпізнати цитату автоматично, користувач системи має змогу позначити поточний збіг як цитату. Система пошуку орієнтується на правильне позначення цитування згідно норм оформлення дисертацій. Недотримання цих норм або використання інших стандартів може привести до пропуску системою цитат. У випадку, якщо ця сама цитата буде знайдена у іншому документі, вона буде позначена як збіг.

Для пошуку текстових збігів використовуються алгоритми нечіткого пошуку, тому тут теж можливі помилкові спрацювання. Користувачу

системи надається можливість повністю видалити збіг, якщо він помилковий. Цю дію неможливо відмінити, після видалення будь-якого збігу (чи позначення його як цитати) загальний відсоток унікальності та збігів в документі буде перераховано. Система пошуку збігів не ставить метою визначення точної унікальності документу, вона лише надає експерту необхідні дані та інструменти. Остаточний висновок про унікальність має зробити експерт.

Після того як користувач переглянув інформацію, надану в звіті по перевірці, та виконав необхідні коригування, він може експортувати звіт в .pdf файл. Для цього необхідно повернутися до перегляду загального звіту та вибрати "Експортувати звіт" з меню документа. Експортований звіт містить в собі загальну статистику по документу та інформацію про документи, у яких знайшлися збіги. В звіті наведені загальні відсотки унікальності, збігів та цитат. По кожному з документів у яких знайшлися збіги наведено їх назви, авторів, рік захисту або публікації роботи та відсоток збігів з поточним документом. Під час генерації звіту система "вирізає" фрагменти тексту, що позначені як збіг, разом з певною кількістю тексту до і після збігу. Ця кількість тексту вимірюється у словах і може бути задана у налаштуваннях системи. За бажанням, користувач може зробити цю кількість рівною нулю і отримувати в експортованому звіті лише текст зі збігів. Але тоді фрагменти збігів будуть вирвані з контексту.

В системі є можливість експортувати звіт визначення збігів у рисунках і формулах. В даний звіт будуть винесені зображення, розташовані в дві колонки. Права колонка містить зображення з документу, що перевірявся, ліва — знайдене схоже зображення. Всі знайдені зображення підписані, наведена інформація про назву, автора та рік захисту або публікації роботи, у якій знайдено дане зображення. Звіт по знайдених формулах аналогічний за своєю структурою звіту по знайдених зображеннях. За бажанням користувача системи дані типи звітів можуть бути експортовані як окремі файли або об'єднані в один великий файл.

5.3. Структурна схема супроводу дисертаційних робіт

В системі супроводу наукових робіт ролі користувачів розподіляються наступним чином:

- адміністратори системи;
- галузеві експерти;
- звичайні користувачі.

Адміністратори системи займаються внесенням нових робіт до системи та валідацією. Задача адміністратора – внести усі файли роботи, які необхідні для її аналізу та подальшого розгляду. При внесенні роботи в базу необхідно вказати ряд данних, таких як назва роботи, дата захисту, автор, галузь наук, керівник тощо. Дані поля є обов'язкові, окрім них ще є ряд полів, які можна заповнити, проте вони не будуть використовуватися для аналізу, а введені лише в інформаційних цілях.

Кожна з дисертацій, окрім файлу самої дисертації та файлу автореферату, повинна супроводжуватися такими документами:

1. Опис документів атестаційної справи.
2. Титульний аркуш.
3. Рішення щодо присудження наукового ступеня.
4. Присутні на захисті та результати голосування.
5. Протокол – стенограма (розшифровка засідання спеціалізованої вченої ради).
6. Супровідний лист.
7. Реєстраційно-облікова картка (МОН та УкрІНТЕІ).

Після внесення адміністратором усіх необхідних документів система аналізує їх та виконує валідацію. Файли перевіряються на коректність, вони повинні бути непошкоджені та непорожні. Інформація про дисертацію також проходить валідацію, дата захисту не повинна бути в майбутньому, код

галузі наук повинен бути вказано вірно тощо. Якщо всі дані надано вірно, робота ставиться в чергу на перевірку в системі пошуку збігів у накових роботах. При цьому система перевіряє наявність експертів за тематикою (галузі науки) завантаженої роботи. Якщо в систему потрапляє робота з тематикою, за якою немає визначених експертів, адміністратору системи пропонується вибрати одного з існуючих експертів для перегляду даної роботи. Після цього повноваження адміністратора завершуються, перевірка не може бути відмінена, а робота видалена з системи.

Дисертаційна робота перевіряється за рядом ознак та на предмет запозичень тексту, формул, зображень або таблиць з інших наукових робіт. Перевірка за ознаками включає в себе:

1. Перевірка результатів голосування (чи є голоси проти).
2. Перевірка базової освіти здобувача (чи вона відповідає галузі наук, в якій захищається здобувач).
3. Перевірка наукового ступеню керівника роботи (чи отримано докторську ступінь, відповідність галузі наук тощо).
4. Перевірка об’єму наукової роботи.

Незалежно від результатів перевірки ознак наукової роботи, робота також проходить перевірку на наявність запозичень. Якщо знайдено значний відсоток запозичень або не виконуються вимоги при перевірці за ознаками, то система призначає експерта для розгляду роботи. Схема роботи системи супроводу дисертаційних робіт показана на рис. 5.28.

Експерти в системі супроводу наукових робіт розподілені за галузями наук. Один і той же експерт може відноситися відразу до кількох газузей, якщо це дозволяє його компетенція. Відповідно в певних галузях може бути по кілька експертів. Після проходження роботою перевірки і знаходження значного відсотку збігів або невідповідності певним критеріям МОН України, робота призначається на перевірку експерту. Система вибирає експерта випадково, але при цьому намагається рівномірно розподілити навантаження по експертах. Вибраний експерт отримує повідомлення про

призначену роботу на свою електронну адресу. Увійшовши в свій кабінет, експерт бачить роботи, що очікують перевірки. Він має змогу переглянути збіги з іншими науковими роботами та збіги зображень і формул. Також експерт отримує доступ до усіх завантажених файлів, може скачувати оригінал будь-якого з них. Після закінчення розгляду роботи експертом, від нього вимагається написати текстовий висновок за результатами перевірки.

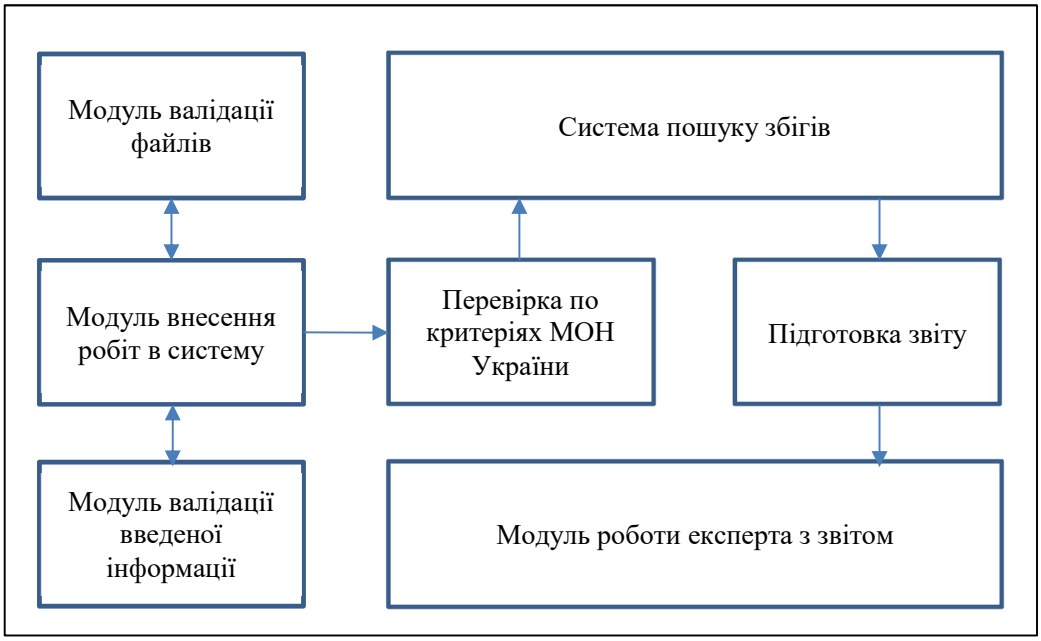


Рисунок 5.28 – Схема роботи системи супроводу наукових робіт

Звичайні користувачі можуть зареєструватися в системі, але вони не мають таких повноважень як адміністратори або експерти. Натомість вони можуть переглядати статус розгляду своєї роботи та отримати доступ до результатів пошуку збігів. Для того щоб отримати доступ до власної роботи необхідно створити аккаунт та отримати ідентифікатор аккаунту. Після цього необхідно надати ідентифікатор одному з адміністраторів для того, щоб він прив’язав роботу до аккаунта користувача.

Система супроводу наукових робіт підтримує генерацію документів. По кожній з робіт, яка була завантажена в систему і пройшла перерірку, формується проект наказу. Сама ж робота додається в список до розгляду на

засіданні Департаменту атестації кадрів вищої кваліфікації та ліцензування МОН. Після фінального розгляду робіт, адміністратор вносить рішення по кожній з них і статус роботи змінюється на “Підтримано” або “Відхилено”. Для робіт, які були підтримані, проект наказу переходить у фінальну версію, адміністратор може використати функцію автоматичного вивантаження наказу на сайт міністерства (для цього створюється його .pdf версія).

Протягом всього часу розгляду роботи в системі користувач, що має доступ до роботи, може бачити її статус. Статусів всього чотири (рис. 5.29):

1. Завантаження.
2. Автоматична перевірка
3. Експертна перевірка
4. Розгляд (змінюється на “Підтримано” або “Відхилено”)

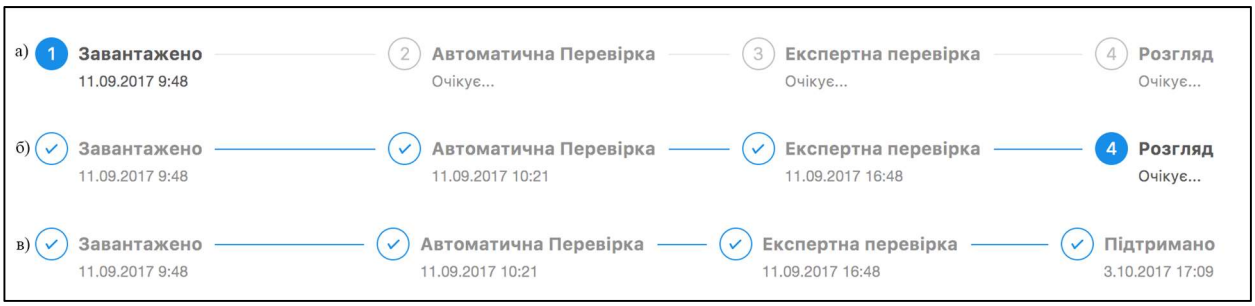


Рисунок 5.29 – Статуси завантаженої роботи в системі

Щойно робота завантажена і провалідована, вона отримує статус “Завантажено”. В процесі розгляду статус змінюються, а на шкалу прогресу додаються дати зміни статусу. Також, якщо автор зареєстрований в системі і адміністратор надав даноу роботу для доступу з його аккаунту, у автора є можливість увімкнути повідомлення. Тоді при кожній зміні статусу автор буде отримувати електронний лист з інформацією про поточний статус.

5.4. Структурні схеми розмежування прав доступу в системі перевірки ступеня унікальності наукових робіт

У системі перевірки ступеня унікальності наукових робіт є можливість гнучко налаштувати рівні доступу до завантажених робіт. Для цього в системі існує можливість створювати організації та департаменти. Користувач системи повинен обов'яково належати до одного з департаментів. Департамент в свою чергу обов'яково повинен належати до однієї з організацій. Архітектура зв'язків між елементами наведена на рис. 5.30.

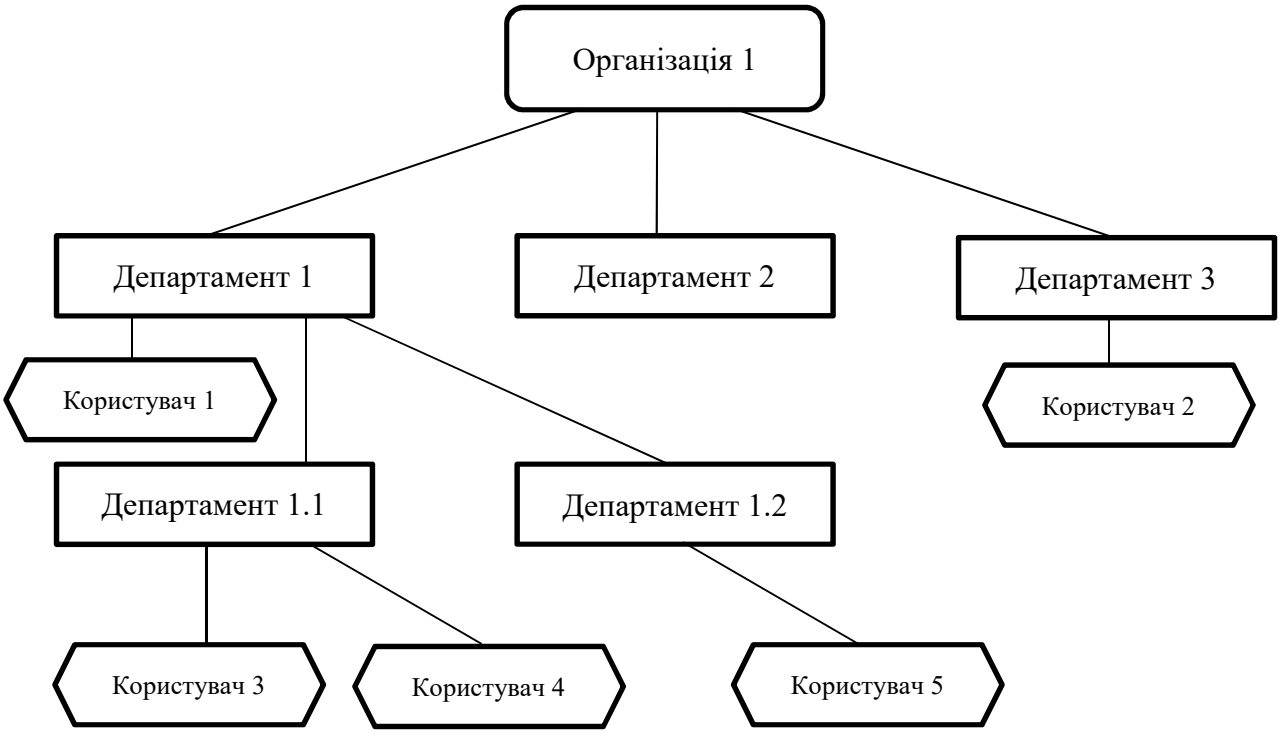


Рисунок 5.30 – Структурна схема одного з варіантів організації департаментів

Таким чином в системі перевірки ступеня унікальності наукових робіт є можливість повторити структуру організації, в якій використовується дана система. Рівні розмежування доступу базуються на структурі організації. Всього існує три рівні доступу:

1. Власні документи.
2. Рівень департаменту.
3. Рівень організації.

Рівень “Власні документи” максимально обмежує доступ користувача, він може оперувати тільки тими документами, які сам завантажив. Всі інші документи, незалежно від того, в якій організації чи департаменті знаходиться користувач є недоступними для нього.

Рівень “департаменту” дозволяє доступ до усіх документів, які були завантажені користувачами поточного департаменту та всіх субдепартаментів, що знаходяться нижче в ієрархії системи. Таким чином, користувач 1 що знаходиться в департаменті 1 (рис. 5.30) з рівнем доступу “Департамент” буде мати доступ до всіх документів, завантажених ним, а також до документів, завантажених користувачами 3 і 4. Але він не буде мати доступ до документів, завантажених користувачами 2 і 5.

Рівень “Організація” надає доступ до будь-якого документу, завантаженого користувачами, які знаходяться в тій же організації, що і користувач з даним рівнем доступу.

Користувачі організації можуть мати права адміністратора. Ці права не впливають на доступ до документів. Користувач з правами адміністратора, але з рівнем доступу “власні документи” не отримає доступу до інших документів. Натомість адміністратори організації мають змогу редагувати інформацію про організацію, створювати департаменти і субдепартаменти, додавати нових користувачів в систему (рис. 5.31).

Інтерфейс системи визначення унікальності наукових робіт наочно демонструє ієрархію організації з можливістю додавати департаменти на будь-якому рівні. Вкладеність департаментів може бути якзавгодно великою. Система не обмежує кількість або рівень вкладеності субдепартаменту.

Рисунок 5.31 – Інтерфейс управління структурою організації

При додаванні нового користувача адміністратор не може поставити йому рівень доступу до документів вище, ніж має сам. Тобто адміністратор з рівнем доступу “Організація” може надати новому користувачу будь-який рівень доступу, тоді як адміністратор з рівнем доступу “власні документи” може надати новому користувачу тільки такий же рівень доступу. На рисунку 5.32 зображена форма додавання нового користувача адміністратором.

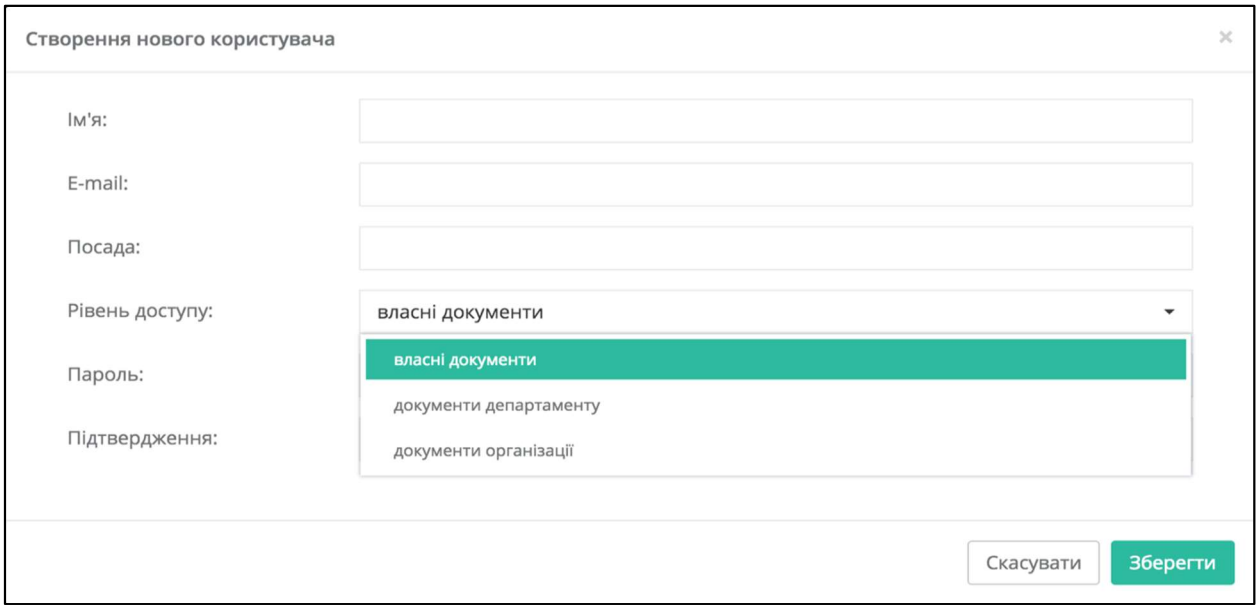


Рисунок 5.32 – Додавання нового користувача в обраний департамент

Адміністратори системи, окрім внесення інформації, необхідної для входу в систему, такої як електронна адреса та пароль, можуть також вносити ім’я та посаду нового користувача. Дана інформація дозволяє краще орієнтуватися в складі департаменту (рис. 5.33).

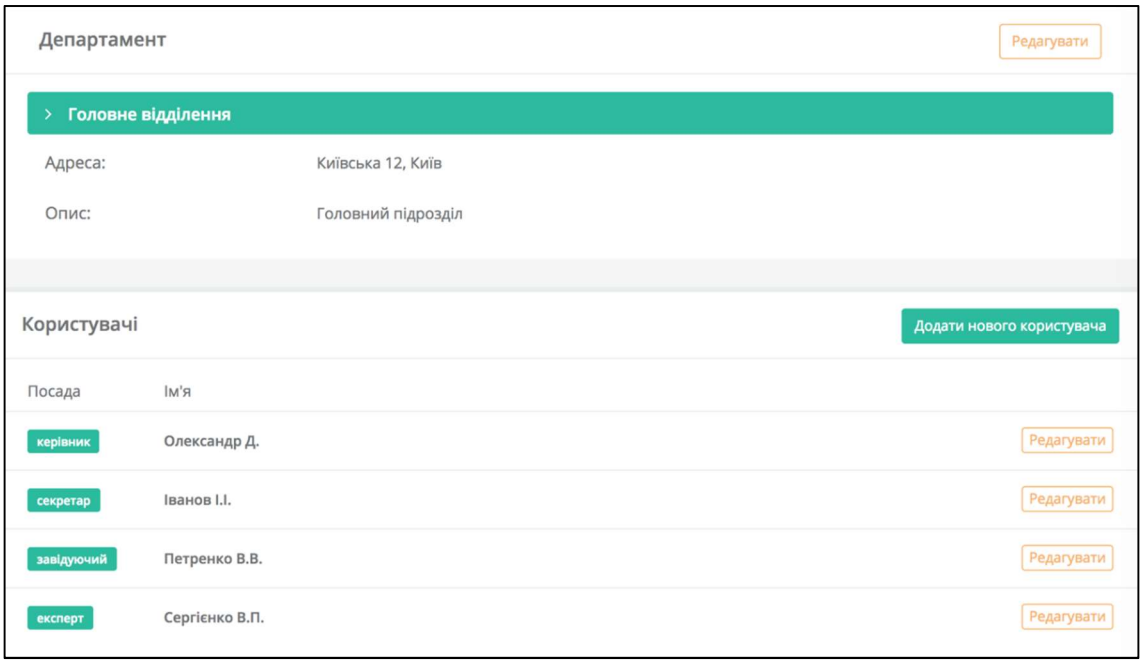


Рисунок 5.33 – Вигляд сторінки департаменту

Після того як користувача було додано, адміністратор має змогу редагувати інформацію про користувача, змінити будь-яку інформацію або пароль (рис. 5.34).

Рисунок 5.34 – Редагування існуючого користувача

Адміністратор також може видаляти департаменти або користувачів. Видалення департаменту можливе тільки якщо немає жодного користувача, що належить до цього департаменту. Всі користувачі департаменту повинні також бути видалені або перенесені. Видалення користувача не видаляє дані про користувача з бази, а лише маркує його як видаленого. Такий користувач більше не відображатиметься в списку користувачів департаменту та не матиме можливості входу в систему.

ВИСНОВКИ

У монографії наведено аналіз сучасних графічних форматів та редакторів, які використовуються для растрових схематичних зображень в наукових текстах, та особливості топологічної структури таких графічних об’єктів, як схеми та діаграми. Це дозволило створити підхід до виявлення повних або неповних дублікатів цих об’єктів у публікаціях, курсових, дипломних або магістерських проектах тощо. Підхід, що пропонується, передбачає проведення аналізу схем алгоритмів та діаграм за їх просторовою топологією, тобто за особливістю структури зв’язків між вузлами схеми або діаграми.

Система автоматизованого аналізу просторової структури схем здійснює їх обробку, що передбачає реалізацію операцій фільтрації, аналізу структурних елементів схем, формування опису схем, візуалізацію, сегментацію та кодування отриманих даних щодо аналізованих об’єктів. Система передбачає можливість аналізу подібностей та виявлення повного або часткового дублювання схем та діаграм за результатами кодованого представлення їх топологічної структури, що відображує зв’язки між вузлами, а також перевірки збігів текстових написів і пояснень в схемах та діаграмах. Запропонований метод автоматизованого аналізу та кодування схем та діаграм може бути застосований для порівняння цих графічних об’єктів та подальшого їх зберігання в базі даних системи виявлення дублікатів електронних текстових документів.

Складність аналізу і розпізнавання математичних формул, які містяться в текстових документах, полягає в тому, що для знаходження неповних дублікатів необхідно аналізувати не просто графічне зображення (здійснюючи фільтрацію, виділення контурів і застосовуючи специфічні методи порівняння), а й текстову інтерпретацію формули. Це необхідно для того, щоб мати змогу ідентифікувати неповний дублікат, за умови, що в формулі було змінено позначення літер, символи математичних операцій, форми дужок тощо. Тому для знаходження неповних дублікатів

математичних формул було розроблено гібридний підхід, що передбачає використання шаблонів, створених згідно з особливостями графічних редакторів, та спеціальних конверторів формул з різних форматів до канонічного формату.

Запропонована гібридна схема порівняння формул здійснює перевірку оригінальності формульного об’єкта, що аналізується, як за джерелами мережі Інтернет, так і за власними базами текстів з формулами. Схема передбачає можливість аналізу подібностей та виявлення повного або часткового дублювання формул як з використанням шаблонів формульних об’єктів, що аналізуються, так і з застосуванням конвертації математичних формул у різних форматах на універсальну мову математичної розмітки.

Запропоновано гібридний метод виявлення неповних дублікатів у таблицях. Цей метод дозволяє ідентифікувати подібності в текстових та числових даних таблиць окремо, а потім узагальнює отримані результати. Для текстових даних формуються послідовності зі слів в канонізованому вигляді, з яких на основі методу локально-чутливого хешування будуються бітові послідовності. Подібність в цьому випадку розраховується на основі відстані Хеммінга з заданим пороговим значенням. Ідентифікація подібності між числовими даними таблиць реалізується на основі методу найближчих сусідів з заданими метричними відстанями. Метод дозволяє ідентифікувати неповні дублікати, які наявні в даних вхідної таблиці в порівнянні з множиною таблиць, які відібрані з наукових публікацій та дипломних і дисертаційних робіт.

Створено та описано модуль візуалізації інформації та інтерфейс користувача в програмному комплексі визначення неповних дублікатів у електронних документах. Описано схеми розмежування прав доступу в системі та супроводу дисертаційних робіт.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Білощицький А.О. Ефективність методів пошуку збігів у текстах / А.О. Білощицький, О.В. Діхтяренко // Управління розвитком складних систем. – 2013. – № 14.– С. 144 – 147.

2. Hawkins J. On Intelligence [Text] / Jeff Hawkins. – Times Books, 2004. – 272 p.

3. Ту Дж. Принципы распознавания образов [Текст] / Дж. Ту, Р. Гонсалес. – М.: Мир, 1978. – 411 с.

4. Яне Б. Цифровая обработка изображений [Текст] / Б. Яне. – М: Техносфера, 2007. – 587 с.

5. Гонсалес Г. Цифровая обработка изображений [Текст] / Г. Гонсалес, Г. Вудс. – М: Техносфера, 2005. – 1072 с.

6. Лізунов П.П. Моделі та методи визначення нечітких збігів в контенті електронних документів [Текст]: монографія / П.П. Лізунов, А.О. Білощицький, О.В. Діхтяренко. – К.: КНУБА, 2015. –128 с.

7. Яндекс. Картинки и дубликаты изображений [Электронный ресурс] – режим доступа: yandex.ru/company/technologies/duplicates/

8. Варламов А. Д., Шарапов Р. В. Поиск визуально подобных изображений на основе машинного обучения [Текст] / А. Д. Варламов, Р. В. Шарапов // Труды 14-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» : RCDL-2012, Переславль-Залесский, Россия, – 2012. – С.113-120.

9. Слесарев А.В. Яндекс на РОМИП 2010: Поиск похожих изображений и дубликатов [Электронный ресурс] / А.В. Слесарев, И.Б. Мучник, Д.К. Михалев // Труды РОМИП 2010 . – Режим доступа : http://romip.ru/ru/2010/

10. Павлидис Т. Алгоритмы машинной графики и обработки изображений / Т. Павлидис; [пер. с англ. В.П. Ярославского]. – М.: Радио и связь, 1986. – 400 с.

11.Сироджа И.Б. Квантовые модели и методы искусственного интеллекта для принятия решений и управления / И.Б. Сироджа. – К.: Наукова думка, 2002. –420с.

12. Фу К. Робототехника / К. Фу, Р. Гонсалес, К. Ли; [пер. с англ. А.А. Сорокина, А.В. Градецкого, М.Ю. Рачкова; под. ред. В.Г. Градецкого]. – М.: Мир. – 1989.–624с.

13. Пименов В. Ю. Простые методы поиска изображений по содержанию [Электронный ресурс] / Труды РОМИП, 2010. – Режим доступа: http://romip.ru/ru/2010/

14. Mojsilović R. Matching and retrieval based on the vocabulary and grammar of color patterns / R. Mojsilović, J. Kovačević, J. Hu, R. J. Safranek, S. K. Ganapathy // IEEE Trans. Image Processing, – 2000, volume 9, pp. 38-54.

15. Tamura H. Texture features corresponding to visual perception / H. Tamura, S. Mori, T. Yamawaki // IEEE Transactions on System, Man and Cybernatic. – 1978, volume 8(6), pp. 460–473.

16. Zhang D. Content-Based Shape Retrieval Using Different Shape Descriptors: A Comparative Study / D. Zhang, G. Lu // In IEEE International Conference on Multimedia and Expo, 2001, pp.289-293.

17. Quack T. A System for Largescale, Contentbased Web Image Retrieval / T. Quack, U. Monich, L. Thiele, B. Manjunath // MM’04, October 1016, 2004, New York, USA. – P. 120-123.

18.Волосных Д.Ф Использование визуальныхособенностей восприятия компонент цветовой модели HSI при поиске изображений по содержанию [Электронный ресурс] / Труды РОМИ 2010 – Режим доступа: http://romip.ru/ru/2010/

19. Васильева Н. Взвешенный CombMNZ для комбинирования результатов поиска изображений по цветовым признакам [Электронный ресурс] / Н. Васильева, Ю. Гладышева // Труды РОМИП 2010 – Режим доступа : http://romip.ru/ru/2010/

20. Мельниченко А. ЛММИИ на РОМИП-2009: Методы поиска изображений по визуальному подобию и детекции нечетких дубликатов изображений [Электронный ресурс] / А. Мельниченко, А. Гончаров // Труды РОМИП 2009 – Режим доступа : http://romip.ru/ru/2009/

21. Стадник А.С. Анализ кадров видеоряда и вычисление продолжительности сцены используя алгоритм перцептивного хэша [Электронный ресурс] / А.С. Стадник // Информатика и компьютерные технологии-2011 – Режим доступа: http://ea.donntu.edu.ua:8080/jspui/bitstream/123456789/3955/1/4_%D0%A1%D1%82%D0%B0%D0%B4%D0%BD%D0%B8%D0%BA.pdf

22. Чалая Л.Э. Поиск неполных дубликатов в системах анализа цифровых изображений / Л. Э. Чалая, П. Ю. Попаденко // Вісник Кременчуцького національного університету імені Михайла Остроградського. – 2014. – Вип. 5. – С. 42 – 47

23. Антиплагиат [Электронный ресурс]. – Режим доступа: antiplagiat.ru

24. Білощицький А.О. Оптимізація системи пошуку збігів за допомогою використання алгоритмів локально чутливого хешування наборів текстових даних [Текст] / А.О. Білощицький, О.В. Діхтяренко // Управління розвитком складних систем. – 2014. – № 19.– С. 113 – 117.

25. Білощицький А.О. Метод вилучення помилкових збігів текстів в електронних документах [Текст] / А.О. Білощицький, С.Д. Криштоф, С.В. Білощицька, О.В. Діхтяренко // Управління розвитком складних систем. – 2015. – № 22(1). – С. 144 – 150.

26. Михайловський Ю.Б. Система Anti-Plagiarism як інструмент запобігання плагіату в навчальній та науковій діяльності [Текст] / Ю.Б. Михайловський, Н.А. Длугунович // Вісник Хмельницького національного університету. Технічні науки. – 2013. – № 3. – С. 162 – 168.

27. Лупаренко Л. А. Інструментарій виявлення плагіату в наукових роботах: аналіз програмних рішень [Текст] / Л.А. Лупаренко // Інформаційні технології і засоби навчання. – 2014. – Т. 40. – №2.– С. 151 – 169.

28. Шарапова Е.В. Исследование возможностей системы «Антиплагиат» для обнаружения заимствований [Текст] / Е.В. Шарапова // Перспективы науки и образования. – 2013 – №3.– С. 215 – 218.

29. Shenoy M. Automatic Plagiarism Detection Using Similarity Analysis [Text] / M. Shenoy, K.C. Shet, U.D. Acharya // Advanced Computing: An International Journal. – 2012. – № 3 (3). – P. 59–62.

30.Очков В.Ф. Формулы в научно-технических публикациях: проблемы и решения [Текст] / В. Ф. Очков // Электронный журнал Cloud of Science. – 2014. – Т. 1. – № 3. – 421 – 455.

31. Kopka H. A Guide to LATEX and Electronic Publishing [Text] / H. Kopka, P. Daly. – Addison-Wesley, Fourth edition. – 2003. – 660 p.

32. Кнут Э. Все про TeX. [Текст]/Э. Кнут. – М.: "Вильямс", 2003. – 560 с.

33. Очков В.Ф. Встроенные вычисления и отображение формул в электронных и печатных изданиях [Текст] / В.Ф. Очков, Е.П. Богомолова, Е.В. Никульчев., С. Герк // Известия высших учебных заведений. Проблемы полиграфии и издательского дела. № 6. 2015. – С. 45 – 58.

34. Гуссенс М. Путеводитель по пакету LaTeX и его web-приложениям [Текст] / М. Гуссенс, С. Ратц. – М.: Мир, 2001. – 601 с.

35. Гайтан О.М. Елементи технології реалізації автоматизованого адаптивного контролю знань студентів в комп’ютерних системах навчання [Текст] / О.М. Гайтан // Радіоелектронні і комп’ютерні системи. – 2014. – № 4 (68). – С. 97 – 105.

36. Душкевич.О.Г. Редактор химических формул Symyx Draw: Учебно-методическое пособие для студентов химических специальностей [Текст] / О. Г. Душкевич. – Минск.: БГУ, 2014. – 40 с.

37. Давидов М.В. Метод та інформаційна технологія озвучення математичних формул українською мовою [Текст] / М.В. Давидов, О.А. Лозицький, В.В. Пасічник// Штучний інтелект. – 2013. №1. – С. 233-245.

38. Елизаров А.М. Веб-технологии для математика: основы MathML [Текст] / А. Елизаров, Е. Липачев, М. Малахальцев. – М.: ФМЛ, 2010. – 192 с.

39. Baker J. A linear grammar approach to mathematical formula recognition from PDF [Текст] / J. Baker, A. Sexton, V. Sorge // Proc. of Intelligent Computer Mathematics, 2009. – P. 127–133.

40. Давидов М. В. Метод озвучення математичних формул та символів українською мовою [Текст] / М.В. Давидов, О.А. Лозицький, Ю.В. Нікольський // Наукові праці [Чорноморського державного університету імені Петра Могили]. Сер. : Комп'ютерні технології. – 2013. – Т. 213, Вип. 201. – С. 24–29.

41. Ehrig H. Handbook of Graph Grammars and Computing by Graph Transformations [Текст] / H.Ehrig, G.Engels, H.–J. Kreowski, G.Rozenberg. – Volume 2: Applications, Languages and Tools World . – Scientific, 1999. – 132 p.

42. Ian Hutchinson. Web Publishing Mathematics With MathML [Електронний ресурс] / Hutchinson Ian // IAP 2004. – Режим доступу : http://web.mit.edu/acs/iap05/mathml/mathmlfuture.pdf.

43.Josef B. Baker, Alan P. Sexton, Volker Sorge, Masakazu Suzuki : Comparing Approaches to Mathematical Document Analysis from PDF. ICDAR 2011: 463-467 [Електронний ресурс]. – Режим доступу: http://www.inftyproject.org/en/index.html.

44.Лізунов П. П. Гібридний підхід до аналізу та розпізнавання математичних формул з метою виявлення в них подібностей [Текст] / П.П. Лізунов, А.О. Білощицький, Л.Е. Чала, С.В. Білощицька, О.Ю. Кучанський, С.Г. Удовенко // Управління розвитком складних систем. – 2016. – № 27. – С. 145 – 155.

45. Боровик В.М. Програмні компоненти проектування діаграм UML[Текст] / В.М. Боровик, О.І. Труш, О.М. Крива // Проблеми інформатизації та управління. – 2010. –№ 3(31). – С. 14 – 19.

46. Гороховатский В.А. Исследование мер структурного соответствия компонентных объектов [Текст] / В.А. Гороховатский // Системи обробки інформації. – 2009. – Вип. 2 (76). – С. 36 – 42.

47. Обробка графічної інформації [Текст] : навч. посібник / В. А. Романюк, О.М. Сальніков, В. Г. Малюк та ін.; під заг. ред. В. А. Романюка. – Х. : Акад.ВВ МВСУ, 2013. – 112 с.

48. Вдовин А.М. Исследование планарных элементов пространственной структуры изображений [Текст] / А.М. Вдовин, Б.С. Хаба, А.И. Мурынов, В.Е. Лялин // Химическая физика и мезоскопия. – 2001. – 3(2). – С.134 – 147.

49.Лізунов П. П. Аналіз подібностей схем та діаграм в електронних документах [Текст] / П. П. Лізунов, А. О. Білощицький, Л. Е. Чала, С.В. Білощицька, О. Ю. Кучанський, С. Г. Удовенко // Управління розвитком складних систем. – 2016. – № 28. – С. 160 – 169.

50.Fink A. How to Conduct Surveys. [Text] / A. Fink. – Thousand Oaks: Sage Publications, 2005. – 224 p.

51. Ehrenberg A.S.C. A Primer in Data Reduction [Text] / A. Ehrenberg. – Wiley, Chrichester, UK, 1982. – 324 p.

52. Bertin J. Graphics and Graphic Information-Processing [Text] / J. Bertin. – Walter de Gruyter Berlin, New York, 1981. – 279 p.

53. Card S.K. Reading an Information Visualization: Using Vision to Think [Text] / S.K. Card, J.D. MacKinlay, B. Shneiderman. – Morgan Kaufmann, San Francisco, 1999. – 712 p.

54.Su Z. Plagiarism detection using the Levenshtein distance and Smith-Waterman algorithm [Text] / Z. Su, B. R. Ahn, K. Y Eom, M. K. Kang, J. P. Kim, M. K. Kim // IEEE 3rd International Conference on Innovative Computing Information and Control. – 2008. – P. 49-56.

55. Wu S. A fast algorithm for multi-pattern searching [Text] / S. Wu, U. Manber // Technical Report TR-94-17, Department of Computer Science, University of Arizona. – 1994.

56. Burkhard W. A. Some approaches to best-match file searching. [Text] / W.A. Burkhard, R.M. Keller // Com. of the ACM. – 1973. – 16(4). – С. 230–236.

57. Baeza-Yates R. Proximity matching using fixed queries trees [Text] / R. Baeza-Yates, W. Cunto, U. Manber, S. Wu // 5th Combinatorial Pattern Matching, LNCS 807. – 1994. – P. 198 – 212.

58.Кучанський О. Ю. Прогнозування часових рядів методом зіставлення зі зразком [Текст] / О. Ю. Кучанський, В. В. Ніколенко // Управління розвитком складних систем. – 2015. – Вип. 22. – С. 101–106.

59. Кучанський О. Прогнозування часових рядів методом селективного зіставлення зі зразком / О. Кучанський, А. Білощицький // Eastern-European Journal of Enterprise Technologies. – 2015. – Vol. 6, N 4(78). – P. 13–18.

60. Liebowitz S. Network Effects and the Microsoft Case [Text] / S. Liebowitz, S. Margolis // In Ellig, Jerome. Dynamic Competition and Public Policy: Technology, Innovation, and Antitrust Issues. Cambridge: Cambridge University Press. – 2001. – P. 160 – 193.

61. Lizunov P. Detection of near duplicates in tables based on the locality-sensitive hashing method and the nearest neighbor method [Text] // P. Lizunov, A. Biloshchytskyi, A. Kuchansky, S. Biloshchytska, L. Chala. Eastern-European Journal of Enterprise Technologies. – 2016, Vol. 6, Issue 4 (84), P. 4–10.

62. Biloshchytskyi A. Evaluation methods of the results of scientific research activity of scientists based on the analysis of publication citations [Text] // A. Biloshchytskyi, A. Kuchansky, Yu. Andrashko, S. Biloshchytska, O. Kuzka, O. Terentyev / Eastern-European Journal of Enterprise Technologies. – 2017, Vol. 3, Issue 2 (87), P. 4–10. doi:10.15587/1729-4061.2017.103651

63.Biloshchytskyi A. Conceptual Model of Automatic System of Near Duplicates Detection in Electronic Documents [Text] / A. Biloshchytskyi, A. Kuchansky, S. Biloshchytska, A. Dubnytska // 14-th International Conference “The Experience of Designing and Applications of CAD Systems in Microelectronics”, IEEE. – 2017. – P. 381-384.

64. Josefsson S. The base16, base32, and base64. Data encodings [Text] / S. Josefsson. – RFC Editor, United States, 2006. – 18 p.