НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»
ім. Ігоря Сікорського

Навчально-науковий комплекс «Інститут прикладного системного аналізу»____
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри
_____ _____
(підпис) (ініціали, прізвище)
“___”_____20__ р.

Дипломна робота
на здобуття ступеня бакалавра

з напряму підготовки 6.050101 Комп’ютерні науки
(код і назва)

на тему: Аналіз алгоритмів та програмних засобів порівняння документів

Виконав: студент 4 курсу, групи Да-32
(шифр групи)

Іщенко Ярослав Іванович
(прізвище, ім’я, по батькові) (підпис)

Керівник к. т. н., доцент Капшук О.О.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант Економічний к. е. н., доцент Рощина Н.В.
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент к. т. н., доцент Крилов Є. В.
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

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

Студент _____
(підпис)

Київ – 2017 року

Національний технічний університет України
«Київський політехнічний інститут»
ім. Ігоря Сікорського

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

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки 6.050101 Комп'ютерні науки
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____2017 р.

ЗАВДАННЯ
на дипломну роботу студенту
Іщенко Ярославу Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз алгоритмів та програмних засобів порівняння документів
керівник роботи Капшук О.О., к.т.н., доцент _____ ,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від «10»травня 2017 р. № 1477-С

2. Термін подання студентом роботи 9. 06. 2017

3. Вихідні дані до роботи _____
Алгоритми порівняння електронних та паперових документів

Програмні засоби порівняння документів в різних форматах

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)

1. Аналіз алгоритмів порівняння документів.
2. Програмні засоби для порівняння документів та їх використання для виявлення плагіату.
3. Оцінка ефективності програмних засобів порівняння документів

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) Презентація, Таблиця порівняння алгоритмів- плакат, таблиця порівняння програмних засобів- плакат, алгоритми- плакат.

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., доцент, к.е.н.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	10.05.2017	
2	Збір інофрмації	17.05.2017	
3	Дослідження алгоритмів	24.05.2017	
4	Дослідження програмних засобів	31.05.2017	
5	Аналіз програмних засобів	2.06.2017	
6	Виконання економічного розділу	4.06.2017	
7	Оформлення дипломного проекту	5.06.2017	

* Консультантом не може бути зазначено керівника дипломної роботи.

8	Отримання допуску до захисту та подача роботи в ДЕК	10.06.2017	

Студент _____ Я.І. Іщенко
(підпис) (ініціали, прізвище)

Керівник роботи _____ О.О. Капшук
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

Бакалаверської дипломної роботи Іщенка Ярослава Івановича

на тему:”Аналіз алогоритмів та програмних засобів порівняння документів”

Дипломна робота присвячена дослідженню алгоритмів та програмних засобів порівняння документів.

У даній роботі ставиться завдання розглянути сучасні інструментальні засоби для порівняння документів. Було проаналізовано найпопулярніші сучасні алгоритми та програмні засоби існуючі на ринку рішення.

Повсюдне збільшення документообігу, що спостерігається в останні роки в усьому світі, диктує необхідність пошуку нових підходів до порівняння документів. Тенденції останніх кількох років в сегменті IT показують зростання створення і поширення різноманітних ресурсів для для перевірки текстів, сайтів, документів на плагіат, а також для порівняння двох чи більше документів.

Результат роботи – порівняльний аналіз та класифікація сучасних популярних програмних засобів та алгоритмів порівняння докуметів.

Загальний обсяг роботи 118 сторінок, 59 рисунків, 7 таблиць, 8 посилань. Ключові слова: Алгоритм diff, алгоритм шинглів, програмні засоби для порівняння документів, використання програмних засобів для перевірки докуметів в боротьбі з плагіатом.

АННОТАЦИЯ

Бакалаврской дипломной работы Ищенко Ярослава Ивановича

на тему «Анализ алогоритмив и программных средств сравнения документов"

Дипломная работа посвященая исследованию алгоритмов и программных средств сравнения документов.

В данной работе ставится задача рассмотреть современные инструментальные средства для сравнения документов. Были проанализированы самые популярные современные алгоритмы и программные средства существующие на рынке решения.

Повсеместное увеличение документооборота, наблюдается в последние годы во всем мире, диктует необходимость поиска новых подходов к сравнению документов. Тенденции последних нескольких лет в сегменте ИТ показывают рост и распространения различных ресурсов для для проверки текстов, сайтов, документов на плагиат, а также для сравнения двух или более документов.

Результат работы - сравнительный анализ и классификация современных популярных программных средств и алгоритмов сравнения докуметов.

Общий объем работы 118 страницы, 59 рисунков, 7 таблиц, 8 ссылок. Ключевые слова: Алгоритм diff, алгоритм шинглов, программные средства для сравнения документов, использование программных средств для проверки докуметов в борьбе с плагиатом.

ABSTRACT

to the bachelor thesis by Ishchenko Yaroslav Ivanovich on “Analysis of the algorithms and software tools for document comparison”

The thesis is devoted to research of algorithms and software for comparing documents.

In this paper, the task is to consider modern tools for comparing documents. The most popular modern algorithms and software tools existing in the market solution were analyzed.

The widespread increase in workflow that has been observed in recent years around the world dictates the need for new approaches to document comparison. The trends of the last few years in the IT segment show an increase in the creation and distribution of various resources for checking texts, sites, documents for plagiarism, and for comparing two or more documents.

The result of the work is comparative analysis and classification of modern popular software tools and algorithms for document comparison.

The total volume of work is 118 pages, 59 figures, 7 tables, 8 references.
Keywords: algorithm diff, shingle algorithm, software for comparison of documents, use of software for checking documents in the fight against plagiarism.

ЗМІСТ

ВСТУП

Нерідко при роботі доводиться порівнювати між собою різні модифікації документів, наприклад, вихідну і змінену редакції матеріалів, підготовлених в Word або у вигляді PDF-документів або презентацій, робочу та оновлені версії прайс-листів зі зміненими цінами в Excel, різні версії текстових документів і т .п. При цьому питання не в тому, яка з версій файлів є більш свіжої (це і так зрозуміло з властивостей файлів), а важливо, що саме змінилося в документах з точки зору вмісту. Порівнювати документи вручну - заняття невдячне через занадто великих витрат часу і можливості помилок, адже не помітити якусь важливу деталь при перегляді простіше простого. Набагато розумніше завдання порівняння файлів передоручити комп'ютера. В цілому, в плані порівняння Word-документів все йде досить благополучно і без використання допоміжних інструментів, хоча в версіях Word 2002 і Word 2003 дана можливість надійно прихована від чужих очей, і, ймовірно, не так багато користувачів про її існування взагалі здогадуються. Справа в тому, що для порівняння документів тут потрібно спочатку завантажити вихідний файл. Потім з меню "Сервіс" відкрити команду "Порівняти і об'єднати виправлення", вказати файл, порівнюваний з вихідним, і включити прапорець "Чорні рядки". Тільки після цих маніпуляцій кнопка "Об'єднати" перетвориться в кнопку "Порівняти", і при натисканні на даній кнопці програма і проведе порівняння файлів. Результати порівняння будуть показані в новоствореному документі в традиційному режимі рецензування. З появою Word 2007 все стало набагато простіше, оскільки тепер досить перемкнутися на вкладку "Рецензування", клацнути по кнопці "Порівняти" і вказати порівнювані версії документа. Теоретично, в Excel теж можливо порівняння документів вбудованими засобами, правда, тільки при роботі в режимі фіксації змін. Однак це незручно, оскільки кожну з змінених осередків доведеться переглядати, наводячи на неї миша, так як зміни, внесені в документ, відображаються в спливаючих вікнах (приблизно таких,

як звичайні примітки). Подруге, якщо названий режим не буде попередньо включений (команда "Сервіс"> "Виправлення"> "Виділити виправлення", прапорець "Відстежувати виправлення"), то зробити порівняння XLS-файлів потім виявиться неможливо. Що стосується швидкого порівняння PDF-документів, то така можливість, звичайно, мається на Acrobat 9 Pro і Acrobat 9 Pro Extended, але ці рішення встановлені далеко не на кожному комп'ютері. Тому при необхідності швидкого порівняння Excel-таблиць, PDF-документів, презентацій, а також документів в інших форматах, зокрема, текстових файлів і програмних кодів, доводиться вдаватися до використання додаткового інструментарію. Варіантів тут безліч, і це можуть бути як комплексні рішення, що дозволяють працювати з декількома файловими форматами, так і вузькоспеціалізовані утиліти. Чимала частина подібних рішень пропонується за пристойні гроші - скажімо, ціна одного з найвідоміших в цій сфері комплексних рішень Diff Doc становить $ 99,95, а вельми популярна серед програмістів утиліта Araxis Merge оцінюється в € 119. Дослідження показують, що майже 60% офісних працівників стикаються із завданням порівняння документів, при цьому кожен п'ятий співробітник займається такою діяльністю регулярно, наприклад щоб знизити ризик підписання некоректної версії договору через зміни, які могли бути внесені контрагентом. У 78% випадків юристи, представники договірних відділів, логісти, менеджери з продажу, фінансисти та інші офісні співробітники порівнюють паперові екземпляри документів з електронними, в 61% - файли Microsoft Word між собою, трохи рідше - копії у форматі PDF з документами MS Office (51%) і відскановані зображення документів з їх електронними дублікатами (37%). У наявності сильна розрізненість форматів даних, яка вкупі з низьким ступенем автоматизації істотно ускладнює процес пошуку невідповідностей в документах, особливо в разі їх порядкової звірки. Тож не дивно, що в деяких організаціях закривають очі на подібні речі, зовсім не

підозрюючи, наскільки ризикованим може бути такий підхід до організації документообігу в компанії.

1 АНАЛІЗ АЛГОРИТМІВ ДЛЯ ПОРІВНЯННЯ ДОКУМЕНТІВ

1.1 Підходи до порівняння версій файлів

Люди, які використовують системи контролю версій вихідного коду (SVN, Mercurial, Git і т.п.), напевно часто користуються можливістю порівняння версій файлів для перегляду внесених користувачами змін. Існує безліч незалежних програм порівняння версій (WinMerge, BeyondCompare і ін.). При порівнянні версій, як правило, дві версії файлу показуються поруч один з одним таким чином, щоб однакові (незмінним) частини документів були розташовані навпроти один одного, а змінилися (Додані та видалені) виділяються відповідним кольором.

Алгоритми можуть використовуватися для реалізації такого порівняння.

В якості найпростішого випадку розглянемо порівняння простих текстових (plain text) файлів.

Текстовий файл - це набір рядків (а точніше абзаців) тексту. Абзац - це рядок символів, що закінчує символами кінця рядка і перекладу покащика (в Windows) або тільки одним символом кінця рядка (в Unix / Linux). Не будемо відволікатися на те, що символи можуть представлені різними кодуваннями і будемо вважати, що ми маємо справу з однобайтним ASCII. Завдання порівняння версій такого файлу полягає у визначенні того, які абзаци не змінилися, а які змінилися, були видалені або додані.

Більшість алгоритмів порівняння файлів засновані на алгоритмі пошуку найдовшої підпослідовності (LCS, Longest Common Subsequence). Дійсно, найдовша спільна підпослідовність символів двох файлів можна вважати незмінною частиною, а всі інші ділянки (в залежності від їх приналежності до однієї з версій) є віддаленими (якщо вони належать старої

версії) або доданими (якщо належать нової). Існує кілька реалізацій алгоритму LCS, обчислювальна складність яких варіюється від поліноміальної (прямо пропорційною довжині порівнюваних підпослідовностей) до логарифмічною. Дані алгоритми також дуже вимогливі до пам'яті. Стає зрозуміло, що використовувати посимвольного подання документів до таких умовах нереально. Зручніше в якості одиниць порівняння використовувати абзаци. Їх можна порівнювати «як є», тобто безпосередньо як рядки, але в цілях оптимізації зручніше їх прохешіровать і порівнювати хеші, а до самих рядках звертатися тільки в разі збігу хеш (тому що від колізій при хешування ніхто не застрахований).[1]

Таким чином, перший етап алгоритму порівняння полягає в завантаженні списку абзаців документів в пам'ять (так, нам дійсно доведеться завантажити обидва порівнюваних документа цілком в пам'ять), їх хешування і застосуванні до отриманих хеш алгоритму пошуку найбільшою загальною збігається підпослідовності (LCS). На виході даного етапу ми отримаємо інформацію про гарантовано не змінилися (тобто збігаються) ділянках документів. Всі інші ділянки є зміненими. Якщо змінилося ділянці зі старої версії відповідає порожню ділянку в новій версії, значить цю ділянку був видалений. Якщо змінилося ділянці нової версії відповідає порожню ділянку в старій версії, значить цю ділянку було додано.

Складніша справа з зміненими ділянками, які були присутні та в старій, і в новій версії. Наприклад, між двома однаковими ділянками можемо розташовуватися змінина ділянка, яка в старій версії становила два абзаци, а в новій - три. Рівних (співпадаючих) абзаців серед цих п'яти абзаців немає. Якщо зупинитися на цьому кроці, можна просто при показі користувачеві виділити ці групи абзаців цілком як змінені і перекласти завдання аналізу докладних змін на користувача. Але можна поступити хитріше. Можна розрахувати оптимальну відповідність для абзаців зміненої ділянки, щоб точно знати, який якому відповідає. Для оцінки «схожості» кожної пари

абзаців можна застосувати алгоритм розрахунку так званої «дистанції редагування» (або відстані Левенштейна). Дистанція редагування - це мінімальна кількість операцій вставки, видалення і заміни одного символу на інший, необхідних для перетворення одного рядка в іншу. Обчисливши дистанцію редагування між кожною парою абзаців зміненої ділянки, ми можемо застосувати метод динамічного програмування для обчислення оптимального відповідності між абзацами зміненої ділянки. Самим неоптимальним варіантом розміщення при цьому є той, коли всі абзаци зміненого ділянки в старій версії вважаються віддаленими, а всі абзаци нової версії - доданими (тобто ніхто нікому не відповідає). Всі інші варіанти розміщення будуть більш оптимальні. Застосувавши метод динамічного програмування можна знайти найоптимальніший варіант відповідності. Тепер ми вже точно знаємо, який абзац зміненого ділянки якого відповідає і зможемо застосувати алгоритм порівняння (на цей раз вже посимвольного) для обчислення точних змін всередині абзацу.

1.2 Ієрархічний алгоритм diff при роботі зі складними документами

Широко відомі алгоритми порівняння двох інформаційних активів («плоских» файлів XML-файлів, Word-документів, різних моделей і т. д.). Однак бурхливий розвиток інформаційних додатків для підтримки офісної діяльності ставить нові завдання перед розробниками програмних засобів в цій галузі. Зокрема, від багатьох завдань по створенню, редагуванню і перевірці документів можуть використовуватися різні варіації алгоритма порівняння двох документів. Однак для того, щоб користувачі отримали повноцінні інструменти, існуючі алгоритми порівняння повинні бути доопрацьовані в наступних напрямках - мати можливість рекурсивно, але по-різному на кожному рівні (так звана ієрархічна неоднорідність), використовувати ієрархічну візуалізації результатів порівняння,

підтримувати порівняння документів «нечіткими» способами. Більш того, доцільно говорити не про окремий алгоритм, а про платформу, яка підтримує ціле сімейство різних базових алгоритмів, дозволяючи їх реалізовувати у вигляді кінцевих і зручних цільових сервісів. Алгоритм пошуку різниці двох файлів (diff) - в загальному випадку двох довільних інформаційних активів - є широко відомим як серед користувачів, так і серед розробників ПО. Порівнюються документи (відповідний алгоритм реализован, наприклад, в MS Office), файли з текстами програм (засоби контролю версій CVS та Subversion, утиліта UNIX diff) і т. д. Алгоритм diff використовується в якості складової частини алгоритмів злиття (merge) двох різних версій файлів в системах версіонного контролю, при синхронізації даних в системах з обмеженнями на трафік, при об'єднанні серверних і клієнтських даних (наприклад, в разі обриву з'єднання) і т. д. Однак при всій своїй популярності цей алгоритм має ліміт на застосування, головний з яких - зручна візуалізація результатів. Наприклад, якщо спробувати скористатися функцією diff для порівняння Word-документів, вбудованої в MSOffice, то якщо порівнювані документи чималі (10 сторінок і більше), а зроблених змін досить багато, то прийняти таку різницю виявляється дуже складно - користувач бачить величезну кількість виділеної інформації, серед якої і вилучені абзаци, і відмінності в правилах пунктуації. З іншого боку, очевидно, що у випадку з офісними документами дуже часто доводиться порівнювати розійшлися версії, які мають однакову структуру - розділи, підрозділи і т. д., і цю структуру можна було б використовувати при візуалізації результатів порівняння. Також виникає потреба порівнювати не тільки документи, а й пакети документів (наприклад, договір може містити багато додатків, які зберігаються в окремих файлах), т. д. застосовувати алгоритм порівняння рекурсивно, але на різних рівнях (при порівнянні папок, файлів, окремих розділів) алгоритм повинен працювати по-різному! І, нарешті, такий алгоритм повинен вміти порівнювати з текстами «нечітко», з точністю до

прогалин, порожніх рядків, різних назв подібних документів і т. д. В області документообігу існує велика кількість задач, де такий алгоритм міг бути затребуваний, і різні цілі призначені для користувача сервісу можна було б створювати на його основі.

Завдання знаходження найбільшої спільної підпослідовності (англ. Longest common subsequence, LCS) - задача пошуку послідовності, яка є підпослідовність декількох послідовностей (зазвичай двох). Часто завдання визначається як пошук всіх найбільших підпослідовностей. Це класична задача інформатики, яка має додатки, зокрема, в задачі порівняння текстових файлів (утиліта diff), а також в біоінформатики.

Підпослідовність можна отримати з деякої кінцевої послідовності, якщо видалити з останньої деякий безліч її елементів (можливо порожній). Наприклад, BCDB є підпослідовність послідовності ABCDBAB. Будемо говорити, що послідовність Z є спільною підпослідовність послідовностей X і Y, якщо Z є підпослідовність як X, так і Y. Потрібно для двох послідовностей X і Y знайти загальну підпослідовність найбільшої довжини. Зауважимо, що НОП може бути кілька.

Робота diff заснована на знаходженні найбільшою загальною підпослідовності (англ. Longest common subsequence, проблема LCS). Наприклад, є дві послідовності елементів:

a b c d f g h q z

a b c d e f g I j k r x y z

і треба знайти найбільш довгу послідовність елементів, яка представлена в обох послідовностях в однаковому порядку.[7] Це означає, що необхідно знайти нову послідовність, яка може бути отримана з першої послідовності видаленням деяких елементів або з другої послідовності видаленням інших елементів. В даному випадку такою послідовністю буде:

a b c d f g j z

Після отримання найбільшої загальної послідовності залишається тільки невеликий крок до отримання схожого на diff виведення:

e h i k q r x y

\+ -+ + -+++

1.3 Неоднорідна ієрархічність

При роботі з офісними документами можна виділити три рівня порівняння: порівняння складу двох пакетів документів, порівняння структури двох файлів, порівняння текстових фрагментів двох файлів, які належать однаковим елементам структури (Простіше кажучи, одного й того ж розділу). Перехід від окремих файлів до пакетів пов'язаний з тим, що папка з файлами (і,можливо, з вкладеними папками) є семантичної одиницею нашої предметної області - договір може містити файл власне з договором, а також багато додаткових файлів з різними додатками, тендерна документація також може бути розбита на численні форми-файли і додатки і т. д. У багатьох задачах важливо надати користувачеві зручний інструмент для знаходження різниці в сенсі складу файлів. При цьому виникає задача ідентифікації подібних файлів, є окремим випадком порівнюваних / зливаються (mapping / matching) позицій в задачах злиття XML-файлів, онтологій і баз даних. Після того, як алгоритм знайшов подібності та відмінності на рівні файлів, він повинен почати порівнювати між собою окремі файли, які він визначив як подібні. Тут потрібно переходити не до простого порівняння текстів (або XML-уявлень файлів), а працювати зі структурами порівнюваних файлів. Мова йде про зміст файлів (Про поділ їх на розділи і підрозділи), однойменних колонках в електронних таблицях, а також про виділення в документах загальних великих об'єктів великих таблиць в текстових документах, картинок (зокрема, Visio-картинок, вставлених в MS Word

файлах, всередині яких можна проводити подальше порівняння) і т. д. Виділення розділів, таблиць, картинок і ін. об'єктів - це знову процедура mapping / matching. На цьому рівні також можуть бути знайдена істотна різниця між порівнюваними документами. А для об'єктів, які визначені як подібні, можна запускати подальше порівняння. Для Word-файлів це буде порівняння текстових фрагментів.

Отже, в цілому у нас виходить єдиний багаторівневий алгоритм порівняння, що визначає всі види відмінностей двох пакетів документів, при цьому на різних рівнях цей алгоритм працює по-різному. У нього може бути інтегрований і алгоритм обчислення різниці двох дерев (при порівнянні структури заголовків двох файлів) і обчислення різниці «плоского» тексту для порівняння тестових фрагментів.

1.4 Ієрархічна візуалізація

Уявімо, що у нас є завдання порівняти два договори, які сильно відрізняються один від одного, проте мають, загальну структуру. Тоді на першому рівні результат порівняння можна помітити червоною іконкою. Далі, відкриваючи один з договорів і переглядаючи його зміст першого рівня, ми можемо побачити, що ряд його розділів теж позначені червоними іконками (вони теж сильно відрізняються від аналогічних розділів в іншому документі або в іншому документі їх немає взагалі), ряд розділів помічені рожевими іконками (вони відрізняються не сильно), а інші не позначені взагалі - вони ідентичні. Відкриваючи, в свою чергу, розділи, помічені червоною і рожевої іконкою, ми побачимо там розділи другого рівня, також помічені подібним чином, і т. д. В кінці кінців, ми дійдемо до листових розділів, які цікавлять нас і вже тут дивимося, власне, на текстову різницю. Важливо, що при цьому обсяг тексту, який відкривається нам на екрані, не буде дуже великим (за умови, звичайно, що розділи в документі не дуже великі, але і в цьому випадку ми можемо мати ієрархічну візуалізацію: на

першому рівні ми можемо переглянути тільки істотні відмінності, які не відображаються точково). Крім того, паралельно ми можемо мати ще й текстовий лог порівняння – для бажаючих (досвід показує, що такі знаходяться). такий текстовий лог дозволяє швидко поглянути на результати порівняння та побачити багато або мало відмінностей, і якщо їх, наприклад, багато, то скористатися ієрархічної візуалізацією.[1]

1.5 Алгоритм шинглів для веб-документів

1.5.1 Канонізація тексту

Канонізація тексту призводить оригінальний текст до єдиної нормальної формі.

Текст очищається від прийменників, сполучників, знаків пунктуації, HTML тегів, і іншого не потрібного «сміття», який не повинен брати участь в порівнянні. У більшості випадків так само пропонується видаляти з тексту прикметники, так як вони не несуть смислового навантаження.

Так само на етапі канонізації тексту можна приводити іменники до називному відмінку, єдиному числа, або залишати від них тільки корінь слова.

На виході маємо текст, очищений від «сміття», і готовий для порівняння (Рис. 1).

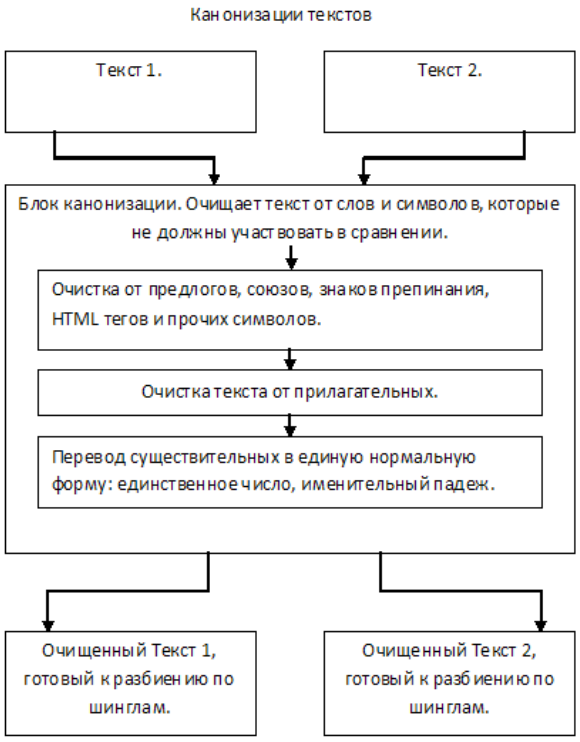


Рисунок 1- Канонізація тексту

1.5.2 Розбиття на шингли

Шингли (англ) - лусочки, виділені зі підпослідовності слів.

Необхідно ыз порівнюваних текстів виділити підпослідовності слів, що йдуть один за одним по 10 штук (довжина шингли). Вибірка відбувається внахлест, а не встик.

Таким чином, розбиваючи текст на підпослідовності, ми отримаємо набір шинглів в кількості рівній кількості слів мінус довжина шингли плюс один (кол_во_слов - дліна_шінгла + 1).

Нагадую, що дії по кожному з пунктів виконуються для кожного з порівнюваних текстів (Рис. 2).

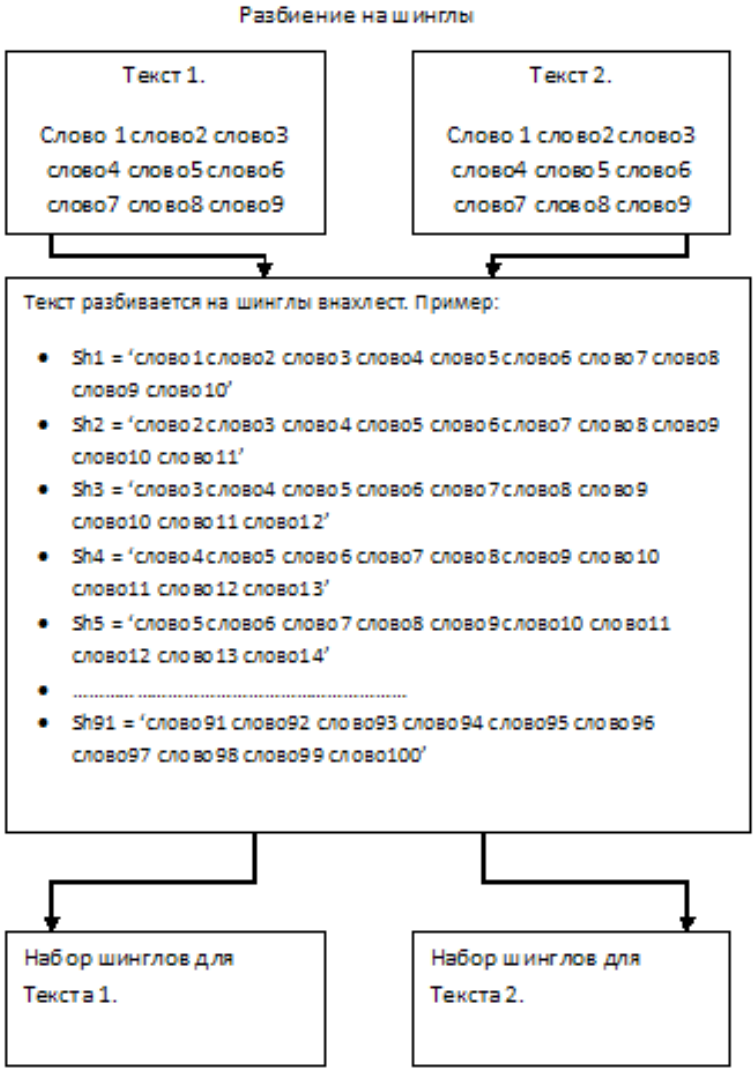


Рисунок 2 – Розбиття на шингли

1.5.3 Обчислення хеш шинглів за допомогою 84-х статичних функцій

Принцип алгоритму шинглів полягає в порівнянні випадкової вибірки контрольних сум шинглів (підпослідовностей) двох текстів між собою.

Проблема алгоритму полягає в кількості порівнянь, адже це безпосередньо позначається на продуктивності. Збільшення кількості шинглів для порівняння характеризується ростом операцій, що критично позначиться на продуктивності.

Пропонується представити текст у вигляді набору контрольних сум, розрахованих через 84х унікальні між собою статичні хеш функції.

Для кожного шингли розраховується 84 значення контрольної суми через різні функції (наприклад SHA1, MD5, CRC32 і т.д., всього 84 функції). Тому кожен із текстів буде представлений, можна сказати, у вигляді двовимірного масиву з 84 рядків, де кожен рядок характеризує відповідну з 84х функцій контрольних сум.

З отриманих наборів будуть випадковим чином відібрані 84 значення для кожного з текстів і порівняні між собою відповідно функції контрольної суми, через яку кожен з них був розрахований. Таким чином, для порівняння буде необхідно виконати всього 84 операції (Рис. 3).

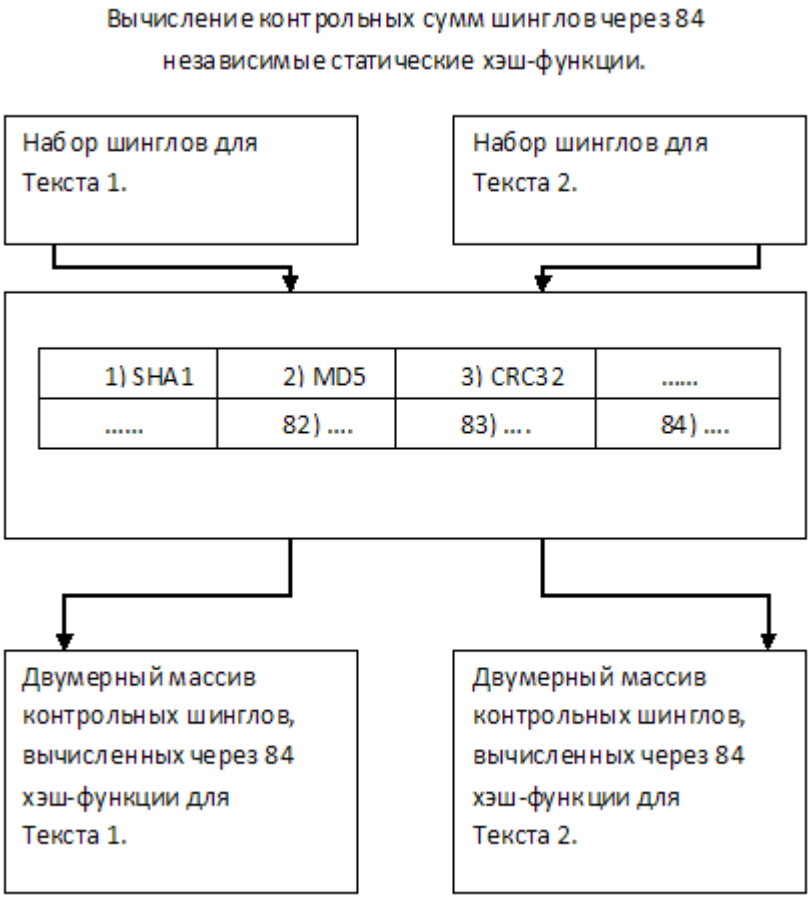


Рисунок 3 – Обчислення хеш шинглів за допомогою 84-х статичних функцій

1.5.4 Випадкова вибірка 84 значень контрольних сум

Як я описував вище, порівнювати елементи кожного з 84х масивів між собою – дуже складно. Для збільшення продуктивності виконаємо випадкову вибірку контрольних сум для кожної з 84х рядків двовимірного масиву, для обох текстів. Наприклад, будемо вибирати саме мінімальне значення з кожного рядка.

Отже, на виході маємо набір з мінімальних значень контрольних сум шинглів для кожної з хеш функцій(Рис. 4).

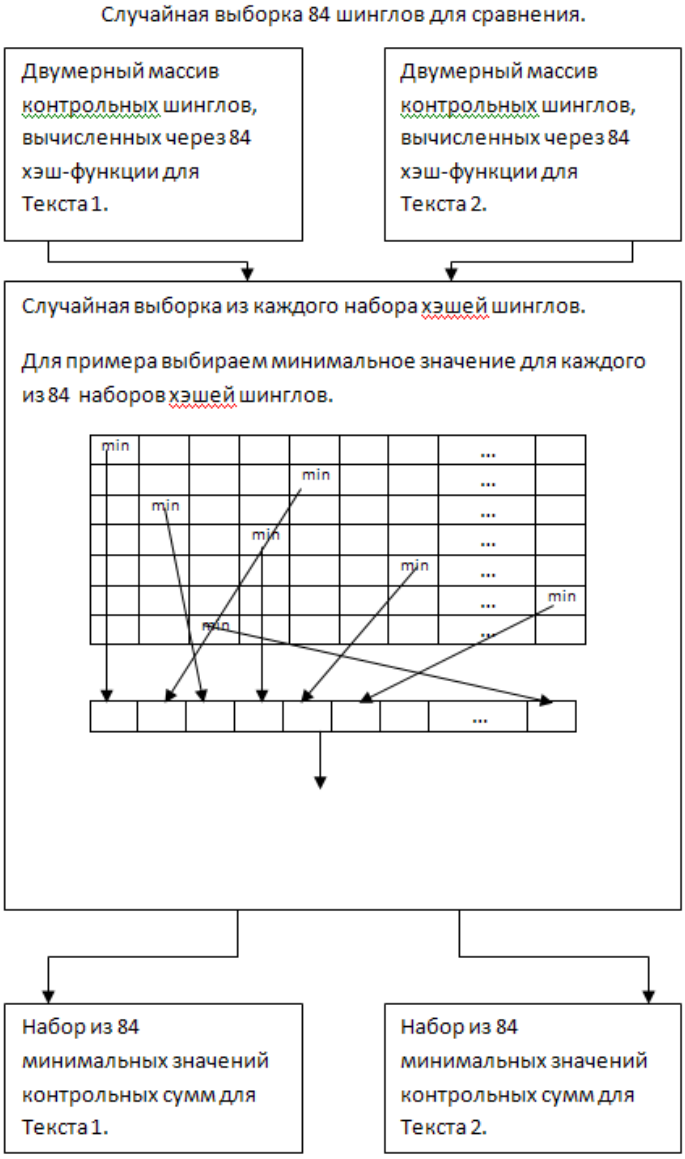


Рисунок 4 - Випадкова вибірка 84 значень контрольних сум

1.5.5 Порівняння, визначення результату

І останній етап - порівняння. Порівнюємо між собою 84 елемента першого масиву з відповідними 84ю елементами другого масиву, вважаємо відношення однакових значень, з цього отримуємо результат (Рис. 5).[8]

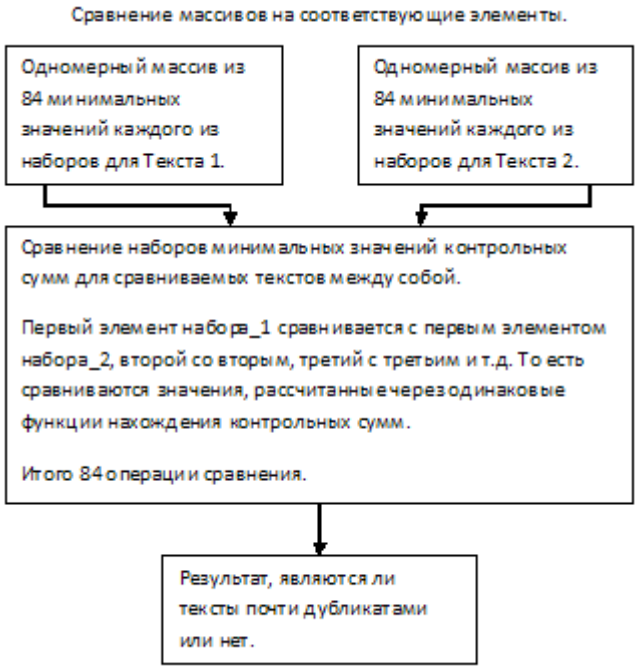


Рисунок 5 - Порівняння

1.5 Алгоритм порівняння документів в форматі LATEX

Розглядається задача побудови різниць, що виникають при редагуванні документів в форматі LATEX. Кожен документ подається у вигляді синтаксичного дерева, вузли якого називаються токенами. Будується мінімально можливе текстове представлення документа, що не міняє синтаксичне дерево. Весь текст розбивається на фрагменти, границі яких відповідають токені. За допомогою алгоритму Хіршбергом будується відображення послідовності текстових фрагментів початкового документа в

аналогічну последовність відредагованого документа, відповідне мінімальній редагуючій відстані. Будується відображення символів текстів, відповідне відображення послідовностей текстових фрагментів. У синтаксичних деревах виділяються токени такі, що символи відповідних фрагментів тексту при відображенні або всі не змінюються, або всі видаляються, або всі додаються. Для дерев, утворених іншими токенами, будується відображення за допомогою алгоритму Zhang-Shasha.

1.6 Алгоритм Хіршбергом

Найбільш ефективним з точки зору кількості споживаної пам'яті є алгоритм Хіршбергом.

Нехай потрібно побудувати відображення послідовності a1 ... an в послідовність

b1 ... bn. Ідея рекурсивного переходу алгоритму ґрунтується на двох равенствах:

δL (a1 ... an, b1 ... bm) = δL (an ... a1, bm ... b1),

δL (a1 ... aiai + 1 ... an, b1 ... bm) = min

j∈1, ..., m-1

(ΔL (a1 ... ai, b1 ... bj) + δL (ai + 1 ... an, bj + 1 ... bm)).

Розглянемо всілякі випадки.

1. n = 0 (перша послідовність порожня). Тоді всі елементи другої послідовності вважаються доданими.

2. m = 0 (друга послідовність порожня). Тоді всі елементи першої послідовності вважаються віддаленими.

3. n = 1 (перша послідовність складається з одного елемента a1). Якщо в другій послідовності є елементи, рівні a1, то перший з них вважається

чином a1, а остальні - доданими. Якщо таких елементів немає, то b1 вважається чином a1, а решта -доданими.

4. m = 1 (друга послідовність складається з одного елемента b1). Якщо в першій послідовності є елементи, рівні b1, то перший з них вважається прообразом b1, а решта - віддаленими. Якщо таких елементів немає, то a1 вважається прообразом b1, а решта - віддаленими.

5. n, m ≥ 2. Перша послідовність розбивається на дві рівні за можливості частини a1 ... ai і an ... ai + 1, де i = [n2]. Друга послідовність розбивається на дві частини b1 ... bj * I bn ... bj * + 1 так, щоб мінімізувати суму:

j* = Arg min j∈1, ..., m-1(ΔL (a1 ... ai, b1 ... bj) + δL (an ... ai + 1, bm ... bj + 1)).

Будуються відображення a1 ... ai в b1 ... bj* I an ... ai + 1 в bm ... bj* + 1.

Другі частини послідовностей записуються в зворотному порядку для можливості повторного використання розрахованого відстані Левенштейна їх префіксів.

1.7 Алгоритм Вагнера-Фішера

Наївний підхід полягає у виконанні двох кроків:

1) побудова матриці відстаней між всілякими парами префіксів методами динамічного програмування [Беллмана, 1960];

2) відновлення редакційного приписи за допомогою покрокового аналізу можливих зворотніх переходів.

Так, наприклад, влаштовані алгоритми Вагнера-Фішера і Нідлмана-Вунша.

ПРИКЛАД 1. Розглянемо застосування обчислення редактирующего відстані і відображеня за допомогою алгоритму Вагнера-Фішера в тому

випадку, коли з слова «ЛОГІКА» виходить слово «АЛГЕБРА».

На Рисунку 6, а представлена матриця L, побудована динамічно по рекурентним формулами. У кожній її клітині знаходиться значення редактирующего відстані для вдповідних префіксів, в правій нижній клітці - редагує відстань для порівнюваних слів.

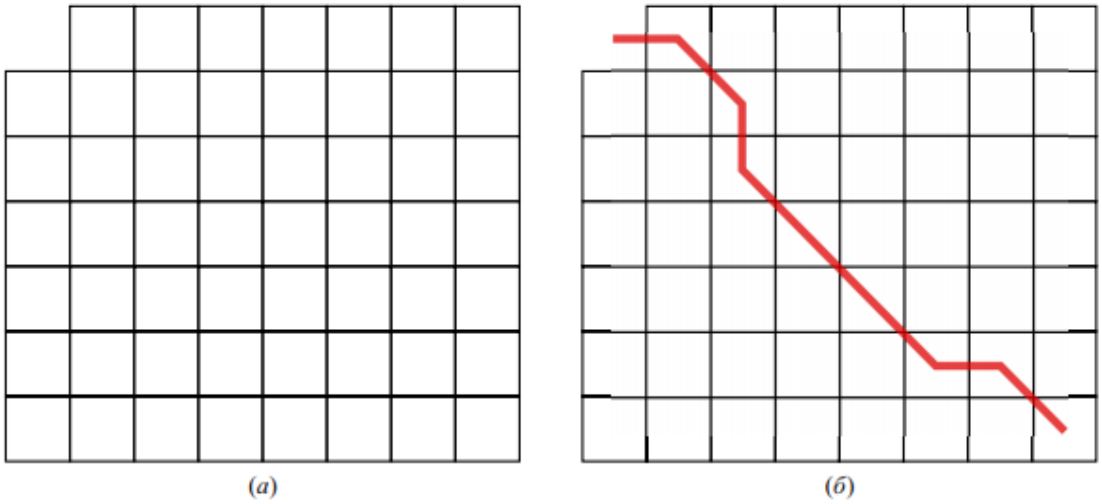


Рисунок 6 Приклад матриці редагуючої відстані для префіксів слів (а). Відновлення послідовності операцій (б)

На рисунку 6, б відображена відновлена послідовність кроків для отримання підсумкового редактирующего відстані. У напрямку ламаної можна визначити, яка операція відбувається на кожному кроці:

• вправо: (L (i, j) = L (i - 1, j) + 1) - доданий символ,

• вниз: (L (i, j) = L (i, j - 1) + 1) - віддалений символ,

• по діагоналі: (L (i, j) = L (i-1, j-1) +1) - символ змінений;

(L (i, j) = L (i-1, j-1)) – символ не змінений.

Це дозволяє побудувати відображення, представлене на малюнку 2 (D означає, що символ

вилучений, I - доданий, C - змінений).

Подібний підхід виявляється вкрай вимогливий до пам'яті: необхідно

зберігати кількість елементів, пропорційне добутку довжин порівнюваних послідовностей.

Тому на практиці він рідко використовується.

1.8 Порівняння двох одинакових форматів документів

1.8.1 Порівняння двох файлів формату doc

Застосовується до: Word 2016 Word 2013 Word 2010 Word 2007.

Даний параметр дозволяє порівняти два документи і вивести на екран тільки незбіжні фрагменти. Порівнянні документи не змінюються. При такому способі порівняння відмінності між документами завжди відображаються в новому, третьому документі.

Якщо потрібно порівняти виправлення, зроблені кількома рецензентами, не слід використовувати цю можливість. Виберіть команду Об'єднання виправлень від декількох авторів в одному документі.

1. Відкрийте документи, які потрібно порівняти.

2. На вкладці Рецензування в групі Порівняти натисніть кнопку Порівняти.

3. Виберіть пункт Порівняння двох версій документа (юридична примітка).

4. В поле Вихідний документ вкажіть документ, який буде використовуватися в якості вихідного.

5. В поле Змінений документ виберіть документ, який потрібно порівняти з уже відкритим документом.

6. Клацніть Більше, а потім вкажіть параметри порівняння документів.

Поруч з Показувати зміни виберіть відображення змін на рівні знаків або на рівні слів.

7. Якщо результати порівняння не потрібно виводити в третьому документі, вкажіть документ, в якому повинні бути відображені зміни.

8. Натисніть кнопку ОК.

9. Якщо будь-яка з версій документа містить записані виправлення, на екрані з'явиться повідомлення. Щоб прийняти виправлення і порівняти документи, натисніть кнопку Так.

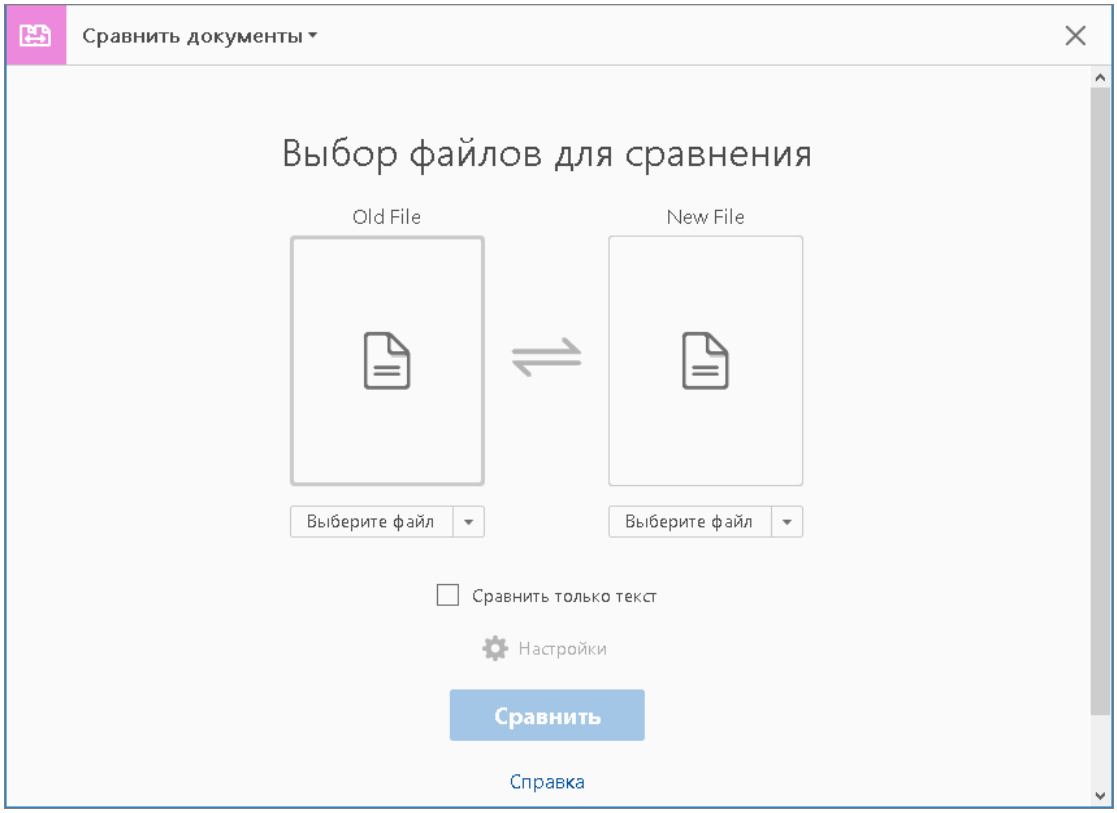
10.Відкриється новий документ, в якому будуть прийняті виправлення, записані в оригінальному документі, а зміни, відмічені в другому з порівнюваних документів, відобразяться у вигляді виправлень.

Порівнянні документи не змінюються.

1.8.2 Порівняння двох версій файлу PDF (Acrobat Pro DC)

За допомогою абсолютно нового інструменту Порівняння документів можна швидко і точно визначити відмінності двох версій файлу PDF.

1. Виберіть Інструменти> Порівняти документи (Рис. 7).



На рисунку 7- Головна сторінка Acrobat Pro DC

2. Клацніть зліва Вибрати файл, щоб вибрати більш стару версію файлу для порівняння. Клацніть праворуч Вибрати файл, щоб вибрати більш нову версію файлу для порівняння.

3. Клацніть Змінити файл, потім виберіть вже відкритий файл або знайдіть і виберіть потрібний файл. Натисніть значок «Поміняти місцями», що розташовується між слайдами, щоб поміняти попередній і новий файли місцями (Рис. 8).

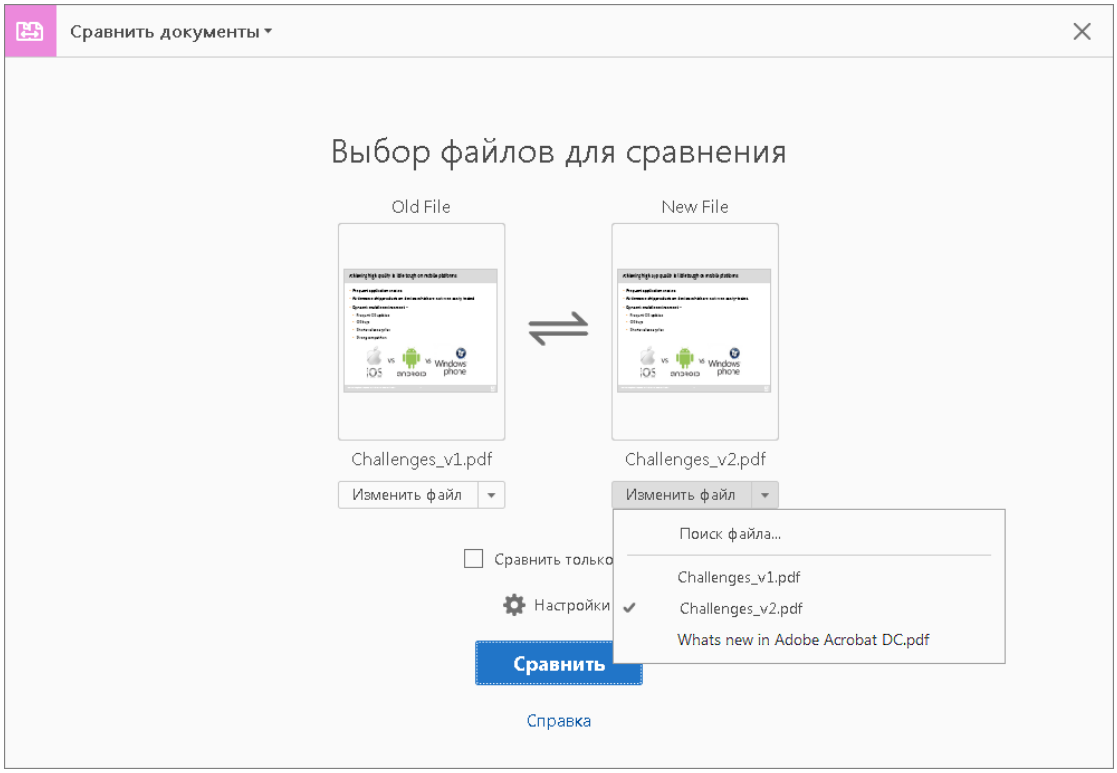


Рисунок 8 - вибір файлів для перевірки

4. Щоб ігнорувати відмінності між графічними елементами, встановіть прапорець Порівняти тільки текст.

5. Налаштування можна змінити, клацнувши значок шестерінки «Налаштування», який дозволяє відкрити наступне діалогове вікно (Рис. 9).

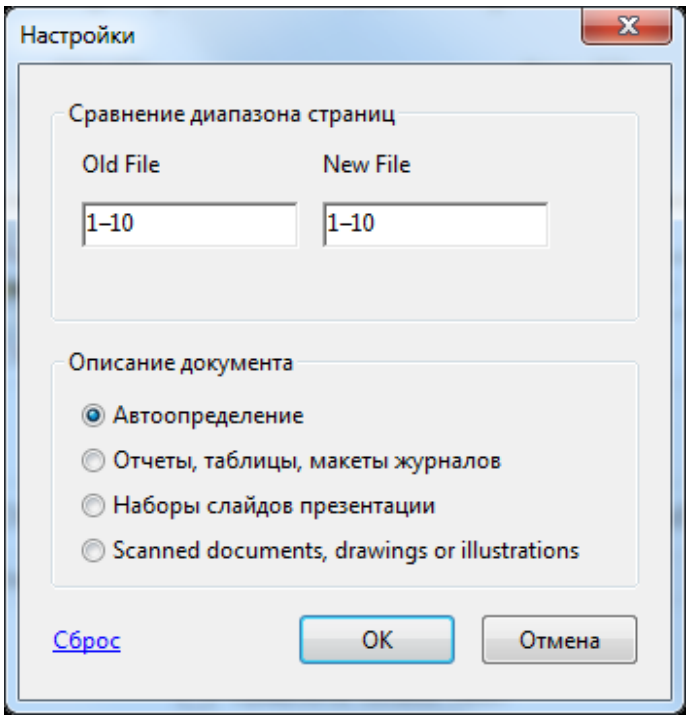
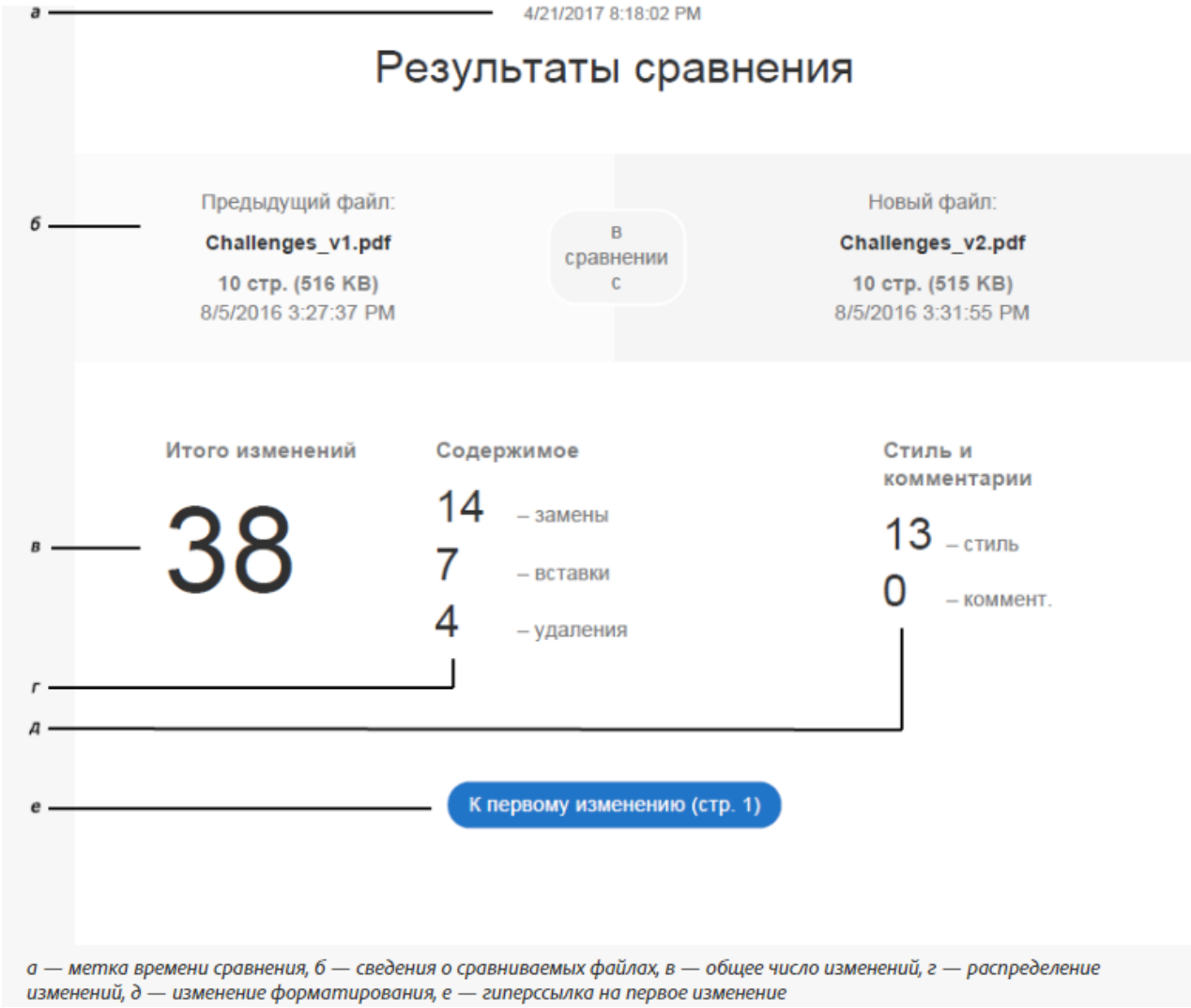


Рисунок 9 - Настройки

6. Натисніть «Порівняти». Acrobat DC відобразить результати в новому документі. На першій сторінці представлена зручна для сканування зведення всіх відмінностей між файлами (Рис. 10).



а — метка времени сравнения, б — сведения о сравниваемых файлах, в — общее число изменений, г — распределение изменений, д — изменение форматирования, е — гиперссылка на первое изменение

Рисунок 10 - результати

7. Натисніть кнопку До першого зміни для перегляду кожного відмінності.

8. Вкажіть спосіб перегляду відмінностей. Одночасний перегляд: використовуйте одночасний перегляд (Ctrl + \) для паралельного перегляду результатів з показом відмінностей, виділених в тексті. Попередній файл відображається зліва, а новий файл відображається праворуч. Зміни виділені в кожній лінії, що зв'язує файли для зручності перегляду. Клацніть лінію, щоб відобразити спливаючу примітку з усіма відповідними змінами. Для відкриття спливаючого меню можна також клацнути виділене вміст (Рис. 11).

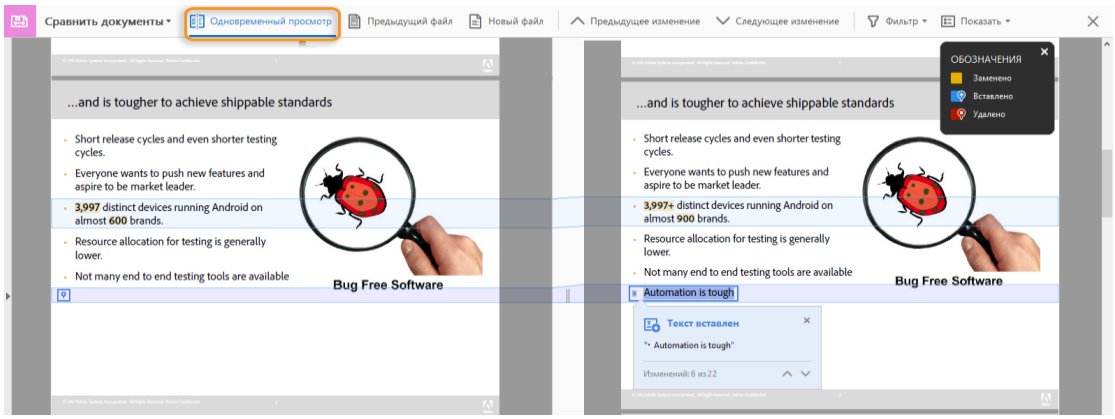


Рисунок 11- Меню

9. Клацніть правою кнопкою миші зміна або відповідне спливаюче меню, щоб встановити потрібний статус для зміни (Рис. 12).

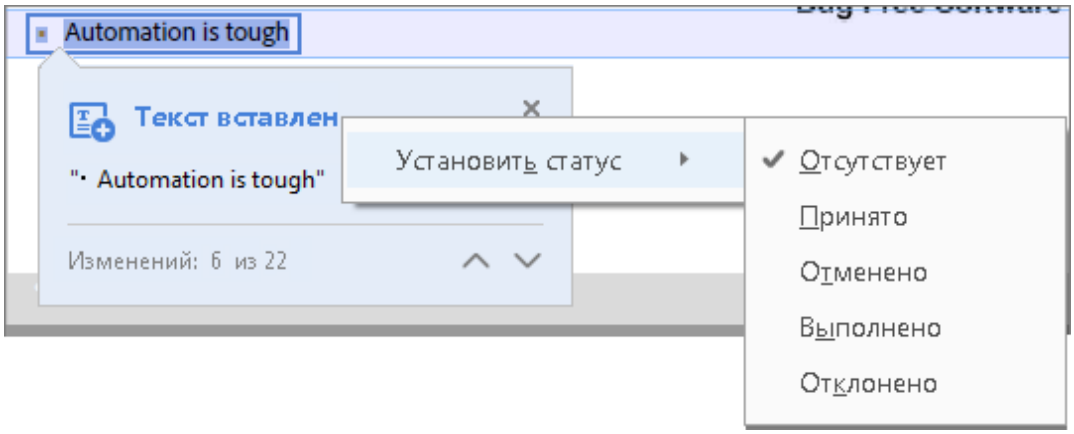


Рисунок 12 - Спливаюче меню

10. Посторінково: використовуйте кнопку Попередній файл або Новий файл (Ctrl + /) на панелі інструментів для перегляду окремого файлу з усіма виділеними змінами (Рис. 13).

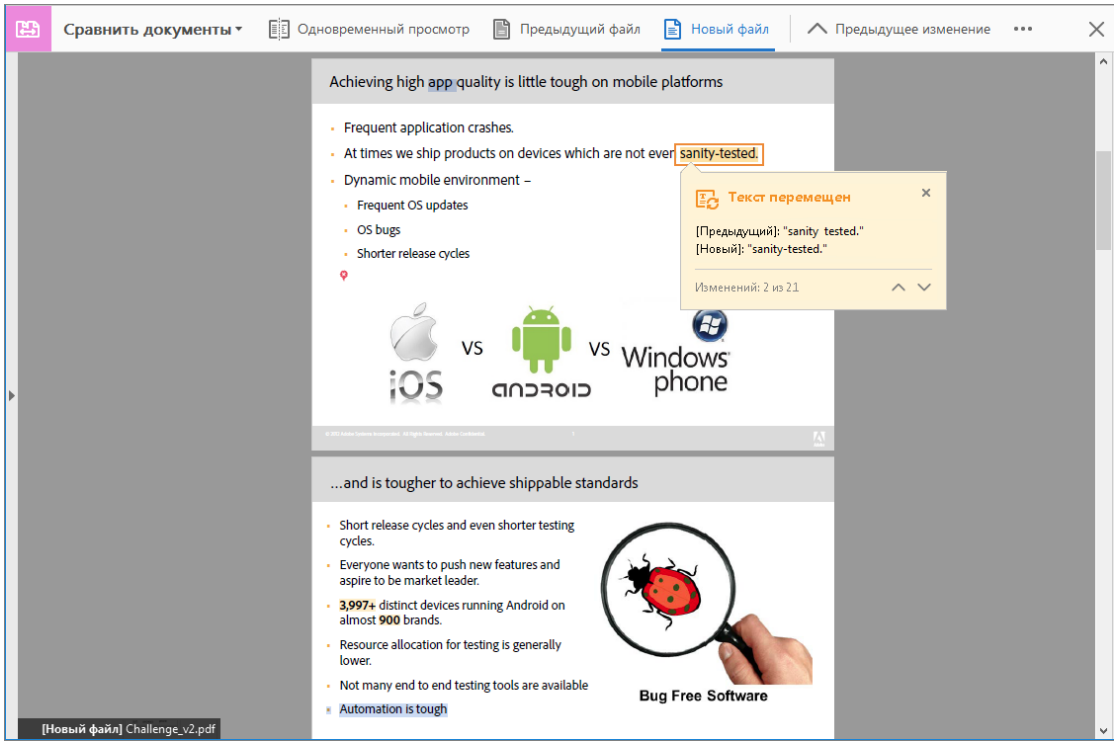


Рисунок 13 - Відображення змін

11.Для перегляду всіх змін: Скористайтеся параметрами Фільтр і Показати.За замовчуванням відмінності в коментарях, форматуванні, а також фонові відмінності відключені. Щоб переглянути ці зміни в звіті, виберіть ці параметри з меню Фільтр на панелі інструментів. Також можна відфільтрувати результати порівняння, вибравши певні типи змін в меню «Фільтр». В меню Показати можна вибрати, чи потрібно відображати або приховувати умовні позначення, рядки і виділення, а також результати (Рис. 14,15).

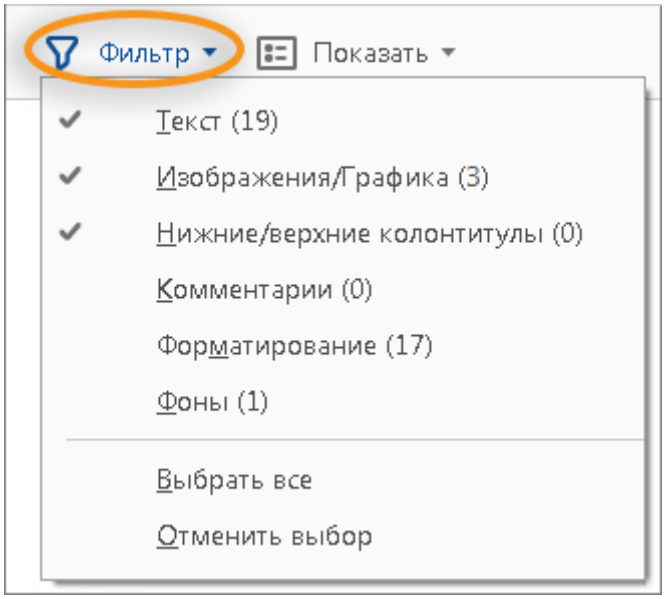


Рисунок 14 - Меню фильтр

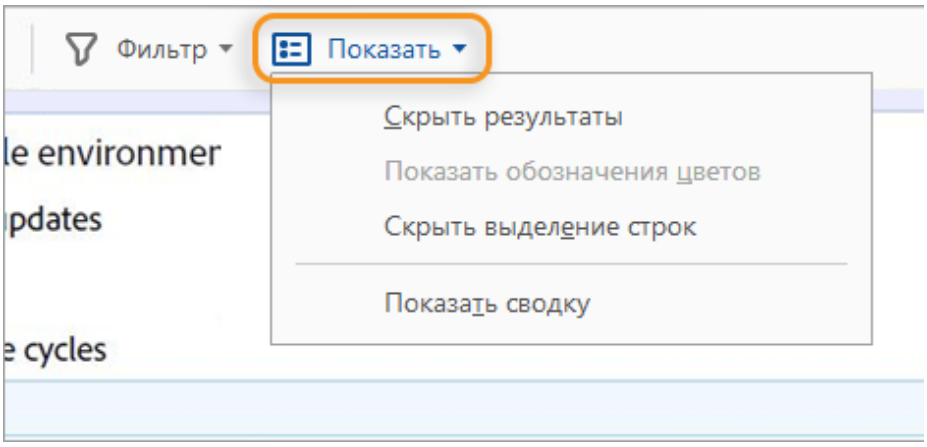


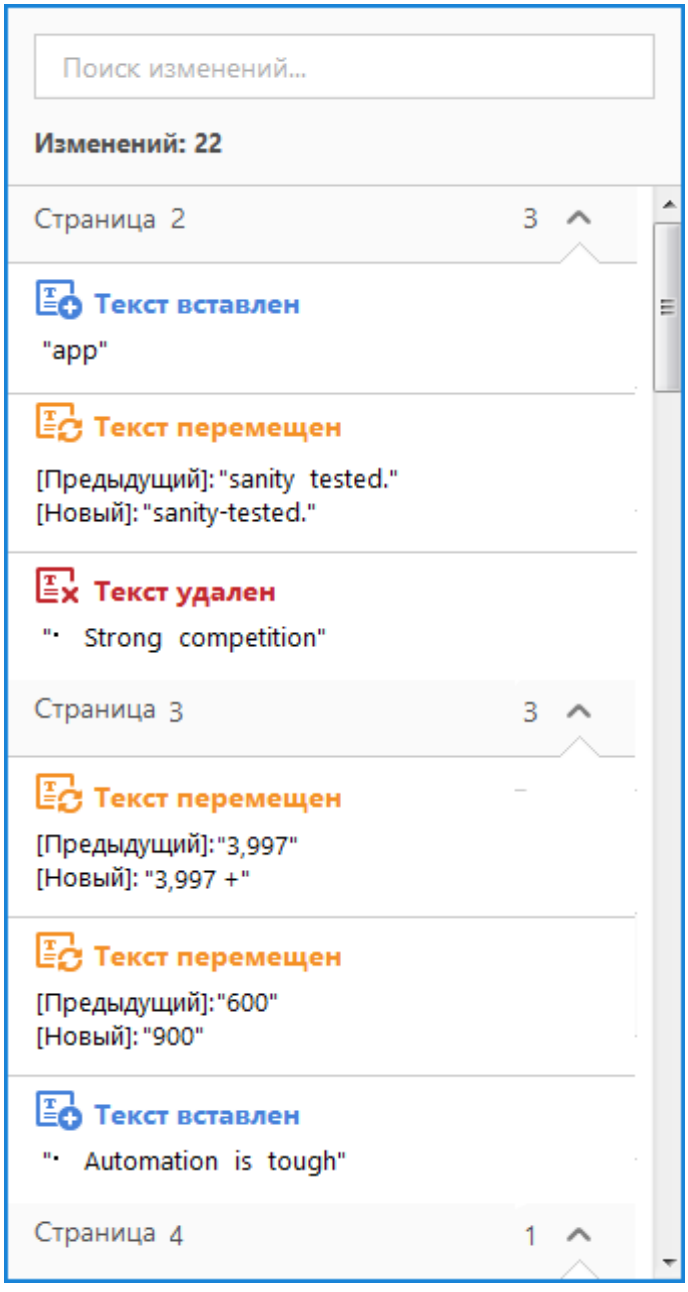
Рисунок 15- меню показать

Використовуйте кнопку Попередня зміна (Ctrl +.) Або Наступна зміна (Ctrl +,) для переходу від одного зміни до іншого(Рис. 16).

Рисунок 16 - Спливаючі нотатки

Кнопки «Попередня зміна» і «Наступна зміна» також доступні у спливаючих нотатках з докладним описом змін.

Використовуйте праву панель. Клацніть трикутник в центрі правого вікна Acrobat, щоб відкрити праву панель. На цій панелі перераховані всі зміни, видимі в звіті «Результати порівняння». За допомогою текстового поля пошуку можна знайти певну зміну (Рис. 17).



На рисунку 17- Результат перевірки

Збережіть файл «Результати порівняння» («Файл»> «Зберегти»). Натисніть кнопку у вигляді хрестика в правому верхньому куті панелі інструментів, щоб закрити інструменти «Порівняння документів».

1.10 Порівняння двох різно форматних докуметів

1.10.1 Порівняння документів PDF і Word

Документ PDF можна порівняти з документом Word. Типовий сценарій має місце, якщо з документа Word був створений PDF-файл, після чого документ був змінений, і тепер потрібно знайти ці відмінності.

PDF-файл передається в програму Convert Assistant для перетворення в документ Word. Порівняння двох документів виконується в Word, що дозволяє виконати візуальне або текстове порівняння з результатом, що відображається в тимчасовому файлі, який при бажанні можна зберегти.

Процедура порівняння документів:

1. Виберіть пункти «Додому»> «Обробка»> Порівняння документів. Відкриється діалогове вікно «Порівняння документів».
2. Прийміть відкритий в даний момент документ PDF в якості більш старого або натисніть кнопку Огляд, щоб вибрати більш новий файл в діалоговому вікні «Відкрити». В полі Редакція виберіть будь-яку з доступних редакцій документа.
3. Натисніть кнопку Огляд і виберіть в якості типу файлу документ Word (DOC або DOCX), а потім виберіть потрібний документ Word в діалоговому вікні «Відкрити».
4. Виберіть тип звіту для представлення результату порівняння: Поруч або поєднаний (див. Опис нижче).

5. Натисніть кнопку ОК. Підтвердіть перетворення з формату PDF і натисніть кнопку Перетворити все в програмі PDF Converter Assistant. Копія вихідного PDF-файлу, доступна тільки для читання, буде відображена в Microsoft Word з розширенням PDF.
6. При необхідності підтвердіть перетворення файлу з формату RTF в файл Word.

1.11 Висновок до розділу 1

В даному розділі буду досліджено алгоритми, які використовуються в сучасних програмних засобах для порівняння документів.

Diff алгоритм реализован, наприклад, в MS Office, файли з текстами програм (засоби контролю версій CVS та Subversion, утиліта UNIX diff) і т. д. Алгоритм diff використовується в якості складової частини алгоритмів злиття (merge) двох різних версій файлів в системах версіонного контролю, при синхронізації даних в системах з обмеженнями на трафік, при об'єднанні серверних і клієнтських даних (наприклад, в разі обриву з'єднання) і т. д. Однак при всій своїй популярності цей алгоритм має ліміт на застосування, головний з яких - зручна візуалізація результатів. Наприклад, якщо спробувати скористатися функцією diff для порівняння Word-документів, вбудованої в MSOffice, то якщо порівнювані документи чималі (10 сторінок і більше), а зроблених змін досить багато, то прийняти таку різницю виявляється дуже складно - користувач бачить величезну кількість виділеної інформації, серед якої і вилучені абзаци, і відмінності в правилах пунктуації.

Проблема алгоритму полягає в кількості порівнянь, адже це безпосередньо позначається на продуктивності. Збільшення кількості шинглів для порівняння характеризується ростом операцій, хто критично позначиться на продуктивності.

2 ПРОГРАМНІ ЗАСОБИ ТА ІНТЕРНЕТ СЕРВІСИ ДЛЯ ПОРІВНЯННЯ ДОКУМЕНТІВ

2.1 ABBYY Comporator

Програма дозволяє швидко виявити різночитання як в текстових документах (Word, Excel, PowerPoint, ODT, ODS, RTF, TXT), так і графічний (GIF, TIFF, BMP, JBIG2, JPEG, JPEG2000, PNG, PDF), наприклад, відсканований або у вигляді фотографії, а також PDF без текстового шару.

ABBYY компаратор працює з текстами російською та англійською мовами. Його інтерфейс і управління досить просте: спочатку відкривається оригінал, а потім в поле порівняння додається змінений варіант. ABBYY Comporator_screen

Додаток проаналізує всі тексти і вкаже області з невідповідністю, серед яких можуть виявитися видалення, додавання або зміна тексту. При цьому несуттєві відмінності, наприклад, у форматуванні, накреслення, пробілу та табуляції, може бути проігноровані.

Дуже зручно, що перегляд знайдених невідповідностей на обох документах виконується синхронно. За підсумками порівняння результат може бути представлений у вигляді таблиці із зазначенням відмінностей або як PDF-документ з коментарями в місцях змін.

ОС: Windows XP / Vista / 8 / 8,1

Сайт: abbyy.ru

Ціна: 23 000 рублів

2.2 Compare Suite Pro

Compare Suite Pro - корисна утиліта, за допомогою якої здійснюється порівняння і подальша синхронізація файлів наступних форматів: Microsoft Word і Excel, веб-сторінок Файловий, PDF, RTF, виконавчі файли, а також

відомості зображень і файли мультимедія. Даний інструмент допоможе всім, хто працює з великим оборотом документів, дозволяє спростити спостереження за змінами в файлах, а в результаті отримати детальний звіт і поділитися ним з іншими. Особливості Compare Suite Pro: - швидке порівняння файлів різних форматів (.txt, .doc, .xls, .pdf, .htm, .html, .exe, .dll, .com, .sys, .ocx, .cpl, .bin); - наявність особливих фільтрів для окремих типів формату документа; - виділення всіх змін і відмінностей в залежності від використовуваного способу порівняння; - наявність 3 способів порівняння; - існує функція злиття для текстових файлів, а опція повернення і повтор допоможе уникнути помилок; - можливість формування інформативного звіту. У програмі порівняльний аналіз проводиться декількома способами. 2 стандартний спосіб: «по символам» і «за словами». При використанні цих варіантів ви отримуєте порівняльна інформацію про дати створення файлів, відсоток збігу даних, про кількість рядків, які були додані / видалені / змінені. Унікальний спосіб: «за ключовими словами» дає список ключового слова два тексти (загальне і окреме для текстів). Після порівняння ви отримуєте детальний підсумкові відомості, занесений в звіті, який презентується в зручному форматі .doc або .html. Робота зі звітами прості: в утиліті можна створити звіт з вихідної і зміненим варіантом тексту або ж вибрати варіант створення звіту, що складається тільки з фрагментів зміненого тексту. У зв'язку з цим звіти діляться на повні і скорочені.

2.3 WinMerge

WinMerge - є Open Source Порівняння і інструментом злиття для Windows. Може порівнювати як файли, так і папки, відображаючи відмінності в візуальної текстовій формі, які легко зрозуміти і обробити.

WinMerge є вельми корисною для визначення місць, які змінилися між версіями проекту, а потім вона дозволяє об'єднувати зміни між версіями.

WinMerge можна використовувати в якості зовнішнього інструменту визначення різниць / злиття, або як автономне додаток.

На додаток, WinMerge має безліч допоміжних можливостей, які роблять процес порівняння, синхронізації і злиття настільки простим, наскільки це можливо:

- порівняння файлів
- Візуальна підсвічування зміни і злиття текстові файли
- Зручний редкатор з підсвічуванням синтаксими, нумерація рядка і перенесення рядка
- Підсвічування змін всередині рядка
- Панель відмінностей показують відмінності поточного файлу в два вертикальних панелях
- Панель розташування відображає карту порівнюваних файлів
- Виявлення переміщених рядків
- Порівняння вмісту каталогів
- Фільтрація файлів, заснована на регулярних виразах, дозволяє включати і виключати елементи з вибірки
- Швидке порівняння, що враховують розміри файли і дати
- Порівняння одного каталогу, або порівняння включаючи підкаталоги
- Може відображати результат порівняння папок у вигляді дерева
- управління версії
- Створюють файли патчі (Обичний-, context- і Єдині формати)
- Дозволити конфлікти файлів
- Початкова підтримка інтеграції з Visual SourceSafe і Rational ClearCase
- Інтеграція з контекстним меню (64-підтримує бітові версії Windows)
- Підтримка архівів здійснюється за допомогою 7-Zip
- Підтримка плагінів.
- локалізуемое інтерфейс

- Онлайн-керівництво і встановлене HTML-керівництво

2.4 Araxis Merge

Araxis Merge - програма для візуального порівняння, злиття і синхронізації папок. Вбудований редактор розпізнає різні формати документів: вихідний код, веб-сторінки, XML, PDF, Microsoft Office, зображення і т. п. Також в Merge передбачена інтеграція з популярними системами управління версіями і іншими середовищами розробки.

У Merge задіяна вкладочному інтерфейс, підтримується збереження сеансів і робоче простору з усіма параметрами в окремому файлі. Стрічка Стрічка розділена на секції, завдяки цьому розташування команд легко запам'ятовується, інструменти для роботи з текстом завжди під рукою. Всі дії, пов'язані з редагуванням і навігацією по тексту, доступні на стрічці. Панель інструментів ретельно налаштовується тільки в Mac OS, можливі й інші відмінності між версіями Merge, в залежності від платформи. Розташування панелей легко змінити на вертикальне чи горизонтальне розташування, можна додати додаткові інформаційні колонки. Таким чином, програма зручна, її інтерфейс продуманий до дрібниць.

Злиття 4 Підтримує режим Порівняння. Додаткові режими (порівняння зображень і виконавчі дані) не такі цікаві з огляду на відсутність інструментів редагування. Тому далі мова буде йти про порівняння файлів і директорій.

Текстовий редактор підтримує підсвічування синтаксису, нумерацію рядків. При порівнянні, крім змін, зазначених відповідним кольором, зручно відстежувати зв'язки між рядками за допомогою сполучних ліній (Зразки ліній). Навівши курсор на відповідний блок, можна застосувати для нього операцію злиття з сусіднім файлом (функція Point-і-клацання зрощування).

Тут вловлюється аналогія зі згаданим в першій частині огляду програми SmartSynchronize. До документу можна додавати закладки і коментарі. Експорт звітів, з занесенням всіх відмінностей, здійснюється в форматах DIFF, HTML, HTML-слайдшоу і XML.

Другий основний режим роботи Araxis Merge - порівняння і синхронізація каталогів. Сильна сторона цього інструменту - підтримка різних джерел: віртуальна файлова система, мережеві диски, проекти та інші джерела. При порівнянні, доступні дві панелі з відображенням структури каталогів, також нескладно активувати режим тристороннього злиття.

Конфігурація фільтрів для директорій і файлів доступна в розділі Фільтри налаштувань. Вони діляться на візуальному (висновок тільки потрібні дані) і фільтри вибір (вибір файли за заданими критеріями).

У переліку доступних операцій - об'єднання папок, побайтово порівняння і порівняння за розміром і датою. При автосінхронізациі файли з конфліктами не обробляються і відкладаються для прийняття користувачем рішення.

Потенційні можливості Araxis Злиття значно зростає, якщо брати в розрахунку інтеграції з Mercurial, Git, Subversion, Perforce і інші середовища. Можливість розширення забезпечується вбудованими в програму плагінами.

2.5 Advego Plagiatus

Одна з найстаріших програм для Windows, з перевірки унікальності..

Її безглуздо використовувати для розрахунків з клієнтами: унікальність виходить свідомо низькою, а клієнт потім ніяк не доведуть, що 90%, отриману ПРИ перевірку ЦИМ - добре дивом, а не погано.

Переваги Advego:

- Немає обмежень на кількість перевіряється тексту.
- Потужна підтримка.

- Багато функцій і налаштувань.
- Швидке оновлення.

Недоліки Advego:

- Трохи повільніше основного конкурента - програми Etxt Антиплагіат.
- Маленька довжина шингли за замовчуванням.
- Зайва підозрілість.
- Часта поява капчті.

2.6 Etxt Антиплагиат

Гідний Конкурент Advego Plagiatus. На жаль, володіє всіма недоліками останнього, але інтерфейс у цієї програми суб'єктивно краще. Плюс, як мені здалося, працює програма спритніші, трохи більше коректніше шукає плагіат і у неї перевірена робоча партнерська програма.

Переваги Etxt:

- Немає обмежень на кількість перевіряється тексту.
- Багато функцій і налаштувань.
- Швидке оновлення.
- Приємний інтерфейс.
- Суб'єктивно трохи швидше Advego Plagiatus.

Недоліки Etxt Advego аналогічні:

- Маленька довжина шингли за замовчуванням.
- Зайва підозрілість.
- Часта поява капчті.

2.7 Content-watch.ru

Content-годинник може похвалитися цікавими алгоритмами роботи з перевірки унікальності текстів. Основна його перевага, на мій погляд - відсутність паніки і помилкових спрацьовувань. Тобто, якщо вже сервіс визначив рерайт, то рерайт дійсно має місце бути. Тому його можна рекомендувати саме копирайтерам, як надійний засіб позбавитися від необгрунтованих претензій замовника.

Плюс для професійних веб-розробників є можливість реалізації платних програмних перевірок, при цьому вартість API найдешевша на ринку на сьогоднішній день.

- Отже, переваги Content-watch.ru:
- Відсутність паніки і помилкові спрацьовування.
- Швидка і точна перевірка.
- Стабільна робота.
- Найдешевша реалізація API.

Недоліки Content-watch.ru:

- Обмеження за обсягом перевіряється тексту до 20 000 знаків.
- Невдалий застарілий інтерфейс.

2.8 Text.ru

Дуже популярний сервіс онлайн-перевірки текстів на унікальність.

Якість алгоритму перевірки текстів не дуже революційно, навіть у чомусь, на моєму погляд, поступається попередній сервіс Content-watch.ru, але зате Текст.ру «бере» гарний інтерфейс, різноманітний потрібний і не дуже потрібні функції, потужний партнерська програма. Ну і домен: простий і запам'ятовується.

Мені подобається цей сервіс за рахунок глибокого пошуку рерайт. На жаль, в ньому досить часті випадки помилкового спрацьовування (хоча і не так часто як в Advego і Etxt). Втім, є і зворотна сторона: якщо Text.ru показав, що унікальність 100%, значить текст - воістину унікальний.

Таким чином, переваги Text.ru:

- Найбільший набір функцій серед усіх сервісів.
- Глибоке вишукування рерайт (добре для замовника, але погано для виконавця).
- Красивий і різноманітний інтерфейс.
- Можливість додавання завдання в черзі.
- Наявність посилань на результати перевірки.
- Можливість підключення API, а також масовий (але платний) онлайну-перевірка всіх сторінок сайту на унікальність.

Недоліки Text.ru:

- Часто бувають технічні збої.
- Часом дуже довга черга і в результаті вкрай низька швидкість перевірки.
- Обмеження за обсягом перевіряється тексту - 15 000 знаків.
- Дуже незручна, та ще й платна перевірка тексту за посиланням.
- Немає підсвічування частин тексту для кожного знайденого домену.

2.9 Висновки до розділу 2

В другому розділі мною були розглянуті функціональні можливості програмних засобів для перевірки документів.

Я вияснив, що функціонал платних та безплатних програм майже не відрізняется. Хоча головною їх перевагою є те що вони вони можуть порівнювати паперові і електронні документи.

Я проаналізував 5 основних популярних сервісів по перевірці унікальності текстів. Але мені очевидні на сьогоднішній день два лідери: Text.ru і Content-watch.ru.

Причому, Text.ru найкраще підійде замовникам, так як дозволяє провести максимально глибоку перевірку (нехай іноді і з помилковими спрацьовуваннями.

А ось виконавцю краще все-таки підійде Content-watch.ru або Miratools.ru (якщо ненагружен), так як в них набагато менше помилкових спрацьовувань і не доведеться нескінченно переписувати тексти. Аналізувати унікальність технічного рерайта в Text .ru - майже неможливо, так як сервіс реагує на найменші збіги з приводу і без.

3 АНАЛІЗ ФУНКЦІОНАЛЬНИХ МОЖЛИВОСТЕЙ ПРОГРАМНИХ ЗАСОБІВ ТА ІНТЕРНЕТ СЕРВІСІВ ДЛЯ ПОРІВНЯННЯ ДОКУМЕНТІВ

3.1 Програмний засіб для порівняння документів ABBYY Comparator

Рішенням проблеми механічного порівняння документів може бути автоматичне порівняння документів. Саме для виконання цього завдання і призначений ABBYY Comparator, завдяки вбудованій технології оптичного розпізнавання текстів (Optical Character Recognition, OCR) забезпечує порівняння документів російською та англійською мовами як в текстових форматах, так і в графічних (відскановані матеріали або їх фотографії, PDF без текстового шару тощо). Користувачеві потрібно тільки завантажити відскановане зображення документа - і текст буде автоматично розпізнано перед пошуком відмінностей з еталонним документом.

Програма дозволяє порівняти два документи будь-якої комбінації підтримуваних форматів:

PDF (* .pdf);

файли зображень (* .jpeg, * .tiff, * .png, * .j2k, * .bmp, * .jbig2, * .gif, * .pdf та інші);

Microsoft Office:

Word (* .doc, * .docx);

Excel (* .xls, * .xlsx);

PowerPoint (* .ppt, * .pptx);

Rich Text Format (* .rtf);

Open Document (* .odt, * .ods, * .odp);

текстовий документ (* .txt);

HTML (* .html);

XPS (потрібно Microsoft .NET Framework 3.x).

ABBYY Comparator має простий, інтуїтивно зрозумілий інтерфейс з мінімальною кількістю налаштувань і не вимагає спеціального навчання для початку роботи - досить перетягнути мишею файли з документами у вікно програми, визначитися з настройками мовних параметрів на вкладці «Порівняння» і дочекатися обробки завантажених даних. При бажанні можна задати додаткові налаштування порівняння документів для отримання більш точних результатів. Наприклад, включити пошук помилок і розбіжностей в пунктуації, а також змінити режим обробки PDF-документів(Рис. 18).

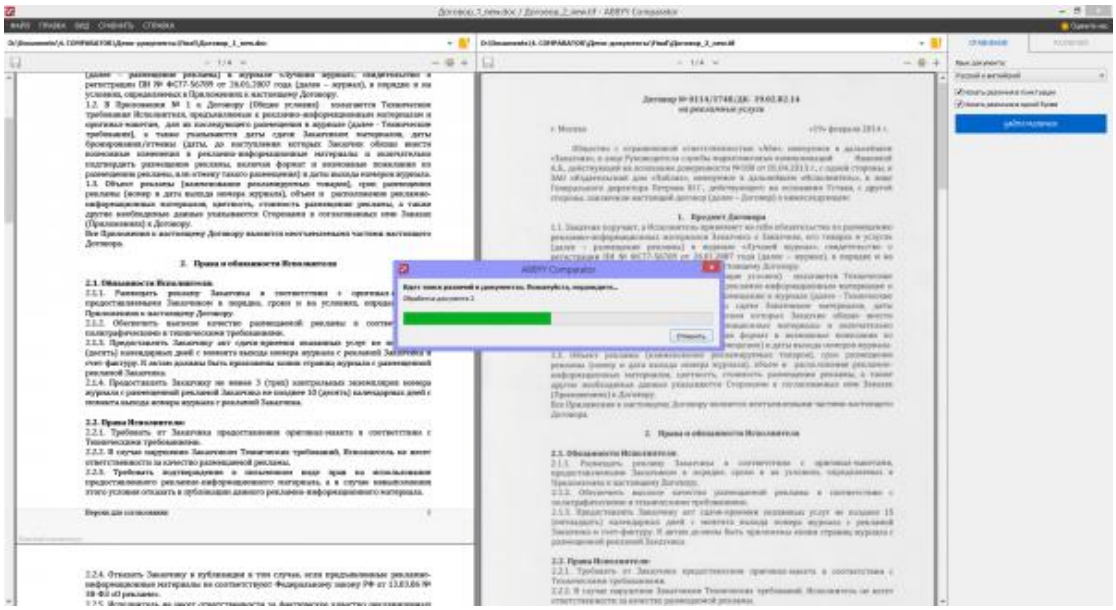


Рисунок 18- ABBYY Comparator

Програма знаходить наступні види відмінностей: видалення, додавання та виправлення. Несуттєві відмінності у форматуванні, накресленні,

прогалини і табуляції ігноруються. Всі виявлені невідповідності відображаються на окремій панелі, а також підсвічуються по тексту в обох документах. При перемиканні по сформованому списку невідповідностей синхронно проглядаються обидва документи, які можна розташувати горизонтально чи вертикально(Рис. 19).

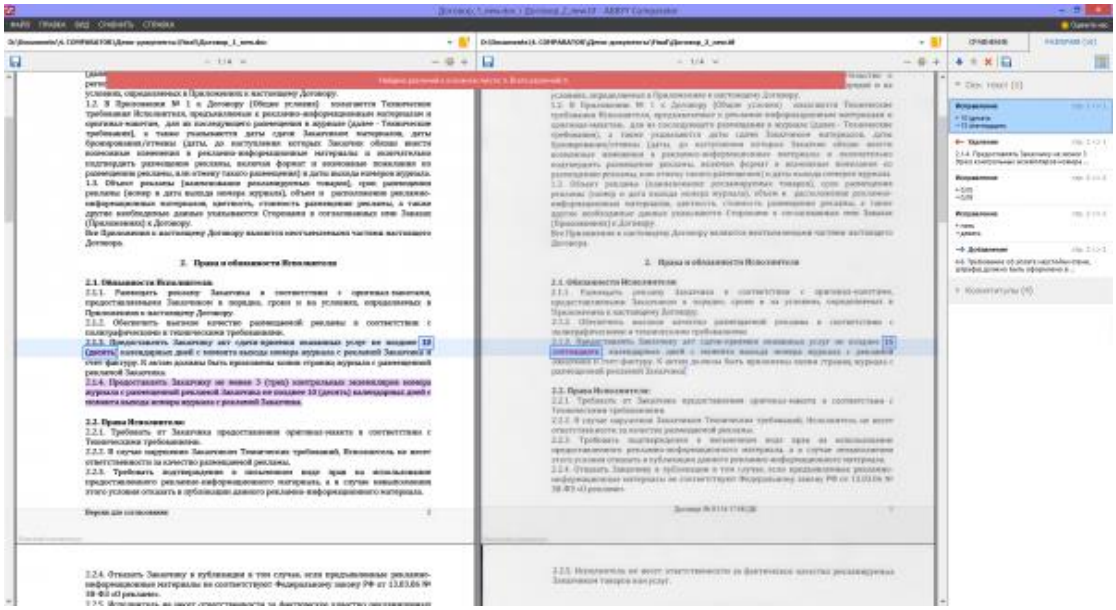


Рисунок 19- ABBYY Comparator

Паралельний перегляд знайдених невідповідностей в обох документах(Рис. 20).

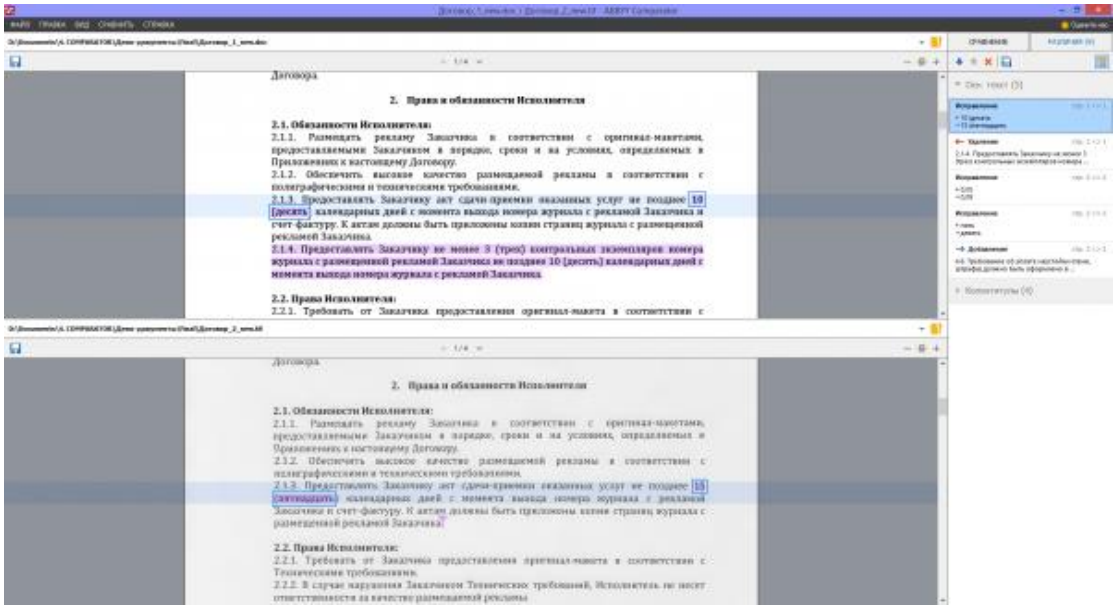


Рисунок 20- ABBYY Comparator

Можна переміщатися між відмінностями на спеціальній панелі, і знайдені відмінності будуть синхронно підсвічуватися на порівнюваних документах.

Важливою особливістю ABBYY Comparator є можливість створення докладного звіту про відмінності і подальшого його збереження у вигляді таблиці в форматі Word або PDF-документа з коментарями в місцях змін. Отриману таблицю можна використовувати, наприклад, для вставки в звіт про узгодження документів. Несуттєві відмінності перед формуванням звіту можна видалити(Рис. 21).

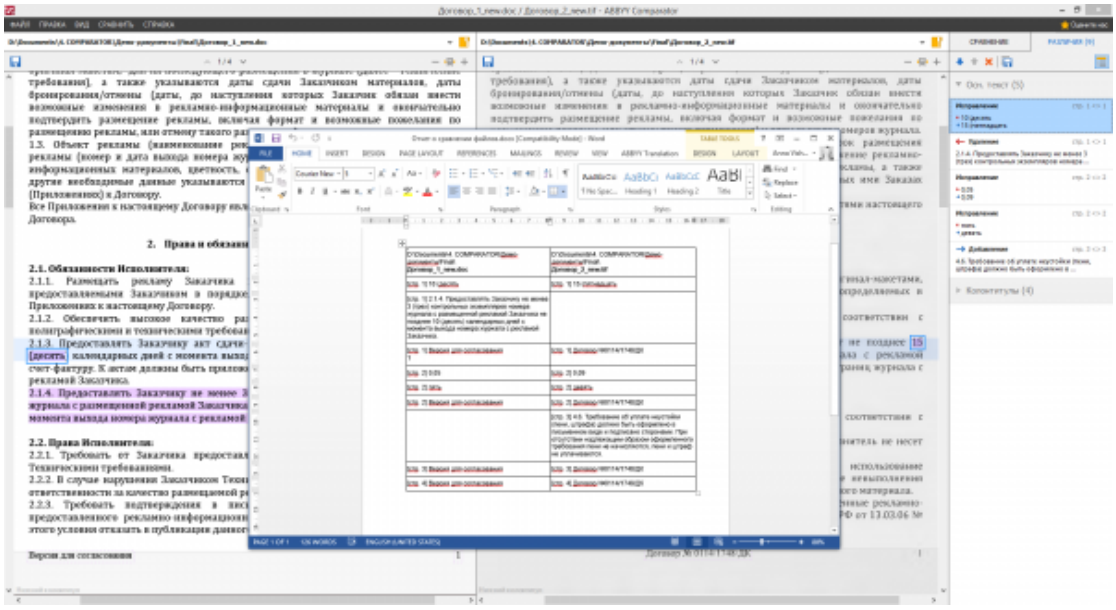


Рисунок 21- результати перевірки

Звіт про знайдені відмінності експортується в файл у форматі * .docx.

Підтримує збереження результатів порівняння у вигляді PDF-документа з коментарями, який можна створити на основі будь-якого з порівнюваних документів.

Таблиця 1- Програмні вимоги продукту

Назва вимоги	Параметри вимог
Операційні системи:	Windows 8; Windows 8.1; Windows 7; Windows Vista; Windows XP SP2.

Продовження таблиці 1

Назва вимоги	Параметри вимог
Вимоги до комп'ютера:	• процесор з тактовою частотою 1 ГГц; • оперативна пам'ять: 1024 Мб і вище; • вільне місце на диску: 300 Мб для звичайної установки і 300 Мб для роботи програми; • відеокарта і монітор з роздільною здатністю не менше 1024x768 точок; • клавіатура, миша або інший пристрій.
Системні вимоги до програмного забезпечення:	• Microsoft Office 2007 (SP3 і вище) / 2010/2013; • Microsoft .Net 4.0 (потрібне додаткове місце на диску: для 32-розрядних (x86) процесорів - 850 МБ, для 64-розрядних (x64) - 2 ГБ).
Додаткові вимоги:	інтернет-з'єднання для активації серійного номера.
Формати документів для порівняння:	• PDF (* .pdf); • Файли зображень (* .jpeg, * .tiff, * .png, * .j2k, * .bmp, * .jbig2, * .gif, * .pdf та ін.); • Microsoft Office: • Word (* .doc, * .docx); • Excel (* .xls, * .xlsx); • PowerPoint (* .ppt, * .pptx). • Rich Text Format (* .rtf); • Open Document (* .odt, * .ods, * .odp); • Текстовий документ (* .txt); • HTML (* .html);

3.2 Програма для порівняння текстових файлів Compare Suite

Програма для порівняння текстових файлів Compare Suite - це зручна і прекрасно оптимізована утиліта для аналізу і синхронізації практично будь-яких документів, що використовуються при роботі на комп'ютері. Для порівняння документів, створених в офісному пакеті від Microsoft, файлів PDF, і інших типів. Так само легко і надзвичайно швидко утиліта дозволяє зробити аналіз і порівняння цілих каталогів з документами, хоча основне призначення Compare Suite - програма для порівняння текстових файлів. Запустимо Compare Suite, і виберемо в меню пункт "Нове порівняння файлів". Ця ж команда виконується комбінацією гарячих клавіш Shift + Ctrl + F. Потім потрібно вказати програмі, які саме текстові файли, нам потрібно порівняти, найзручніше використовувати прийом drag-and-drop - помістивши мишкою кожен порівнюваний файл свого віконце(Рис. 22).

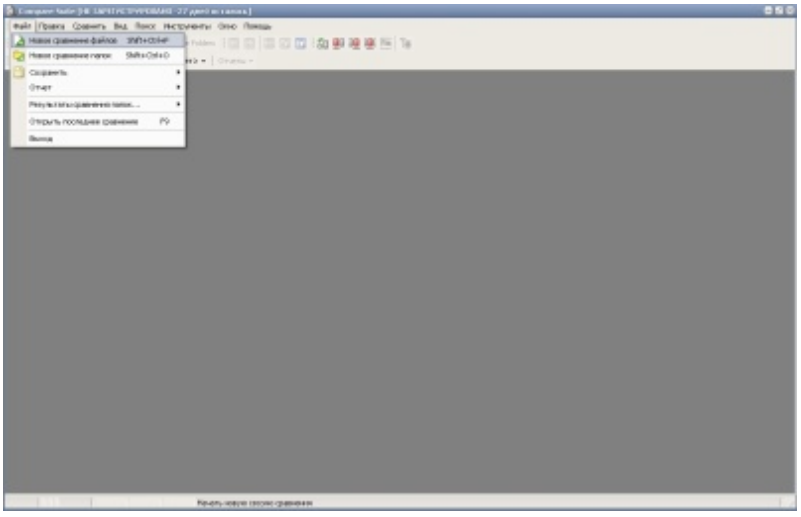


Рисунок 22- Програма для порівняння документів Compare Suite

Після цього ніяких додаткових дій не потрібно - свою роботу програма для порівняння текстових файлів Compare Suite виконає автоматично. Вибравши в меню інший метод порівняння файлів, з трьох можливих, (посимвольний, пословно або по "ключовими словами") ви запустите нову процедуру порівняння. Там же в меню можна виставити і умови для порівняння, наприклад - "ігнорувати всі прогалини в тексті". Пункти меню "Перейти до наступного (попереднього) відмінності" автоматично перемістять положення курсора на необхідну сходинку. Ці ж дії ініціюються гарячими клавішами "Ctrl + N" і "Ctrl + P"(Рис. 23).

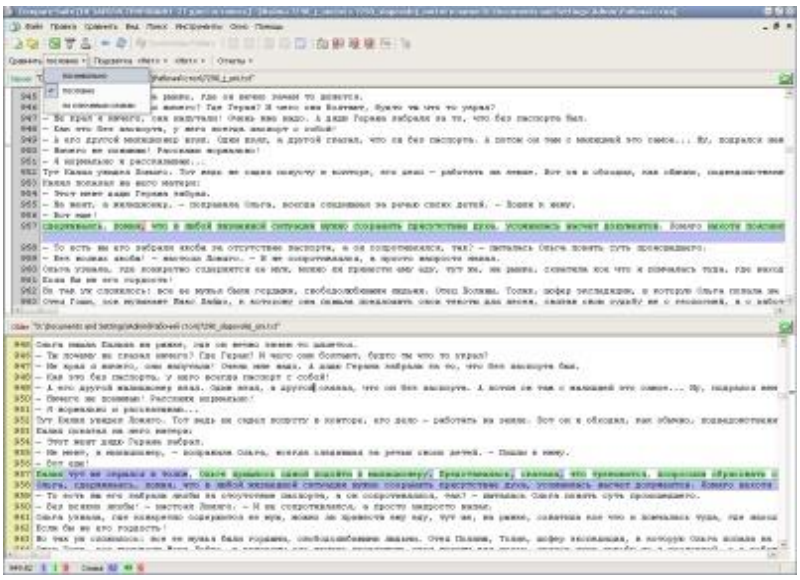


Рисунок 23- Порівняння тексту

Особливо слід вказати на те, що програма для порівняння текстових файлів Compare Suite має можливість для "підсвічування" спеціальних символів, які використовуються в різних мовах програмування і розмітки текстів(Рис. 24).

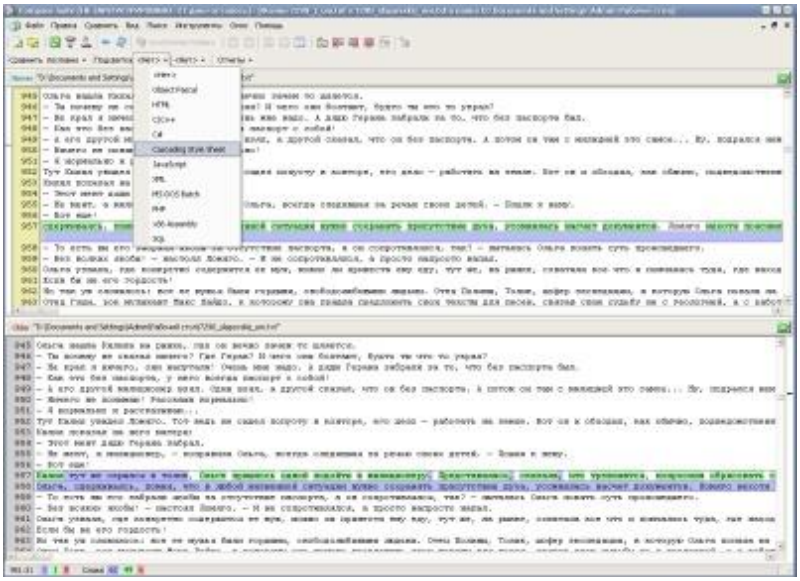


Рисунок 24- Релагування тексту

Крім того, програма для порівняння текстових файлів Compare Suite дає можливість порівняти зміст текстового файлу з вмістом буфера обміну, і вести редагування тексту прямо в своєму вікні. Відредагований файл, зрозуміло, потім можна зберегти. Програма для порівняння текстових файлів Compare Suite коректно працює з різними кодуваннями тексту. На знімках з екрану, наведених в якості ілюстрації, "правий" текстовий файл був спеціально перетворений кодування UTF-8, тобто в ті самі "кракозябри", перед якими пасує вбудований "Блокнот" Widows. Так само вільно програма для порівняння текстових файлів Compare Suite звертається з текстовими файлами форматів різних операційних систем, як Windows, так і Unix і Mac систем(Рис. 25).

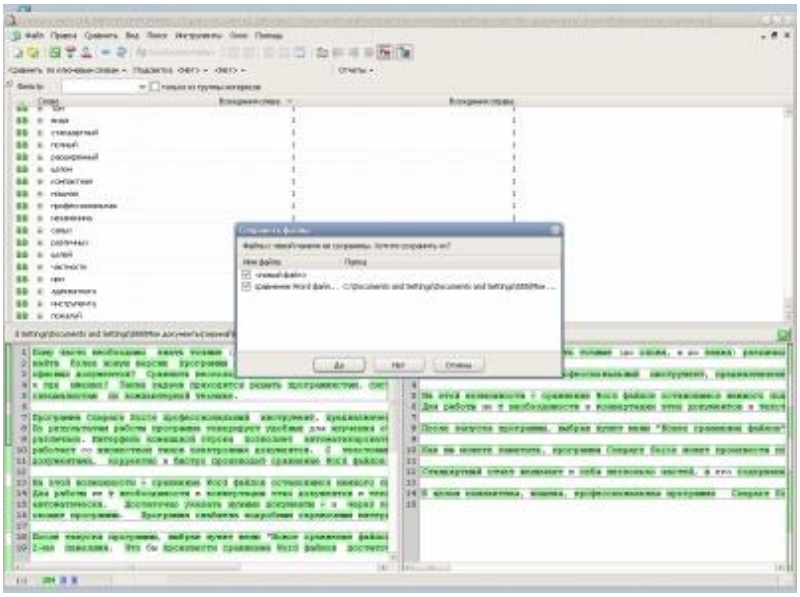


Рисунок 25- Результати порівняння документів

Після завершення аналізу тексту, програма для порівняння текстових файлів Compare Suite генерує звіт в HTML форматі, що власне і є головним і основним перевагою даного програмного продукту. У звіті в наочній формі будуть приведені всі наявні відмінності порівнюються текстових файлів.

Таблиця 2- Програмні вимоги продукту

Операційні системи:	Windows, 10, 7;
Вимоги до комп'ютера:	15 МБ вільного дискового простору; 50 Мб вільного місця на диску для тимчасових даних, які Compare Suite створює при порівнянні великих структур папок (близько 1000 файлів і папок) і створення розширеного звіту. Примітка: для простого порівняння і повного звіту ви не обмежені вільного місця на диску;
Системні вимоги до	Internet Explorer 4 або вище - для правильних

програмного забезпечення:	звітів порівняння; MS Office, з Microsoft Word, Microsoft Excel встановлений для порівняння документів Word, і Excel;
Додаткові вимоги:	інтернет-з'єднання для активації серійного номера.
Формати документів для порівняння:	• PDF (* .pdf); • Файли зображень (* .jpeg, * .tiff, * .png, * .j2k, * .bmp, * .jbig2, * .gif, * .pdf та ін.); • Microsoft Office: • Word (* .doc, * .docx); • Excel (* .xls, * .xlsx); • PowerPoint (* .ppt, * .pptx).

3.3 Програма для порівняння текстових файлів WinMerge

WinMerge є Open Source інструментом порівняння і злиття для Windows. WinMerge може порівнювати як файли, так і папки, відображаючи відмінності в візуальної текстовій формі, які легко зрозуміти і обробити.

WinMerge є вельми корисною для визначення місць, які змінилися між версіями проекту, а потім вона дозволяє об'єднувати зміни між версіями. WinMerge можна використовувати в якості зовнішнього інструменту визначення різниць / злиття, або як автономне додаток.

3.3.1 Особливості

На додаток, WinMerge має безліч допоміжних можливостей, які роблять процес порівняння, синхронізації і злиття настільки простим, наскільки це можливо:

Загальна:

- Supports Microsoft Windows 2000 / XP / 2003 / Vista / 2008/7/8/2012
- Обробка текстових форматів файлів Windows, Unix і Mac
- підтримка Unicode
- Інтерфейс з закладками

Порівняння файлів:

- Візуальна підсвічування змін і злиття текстових файлів
- Зручний редкатор з підсвічуванням синтаксими, нумерацією рядків і перенесенням рядків
- Підсвічування змін всередині рядка
- Панель відмінностей показує відмінності поточного файлу в двох вертикальних панелях
- Панель розташування відображає карту порівнюваних файлів
- Виявлення переміщених рядків

Порівняння вмісту каталогів:

- Фільтрація файлів, заснована на регулярних виразах, дозволяє включати і виключати елементи з вибірки
- Швидке порівняння, що враховує розміри файлів і дати
- Порівняння одного каталогу, або порівняння включаючи

підкаталоги

- Може відображати результат порівняння папок у вигляді дерева

Управління версіями:

- Створює файли патчів (Normal-, Context- і Unified формати)
- Дозволити конфлікти файлів
- Початкова підтримка інтеграції з Visual SourceSafe і Rational ClearCase

Інше:

- Інтеграція з контекстним меню (підтримує 64-бітові версії Windows)
- Підтримка архівів здійснюється за допомогою 7-Zip
- Підтримка плагінів.
- локалізуемое інтерфейс
- Онлайн-керівництво і встановлене HTML-керівництво

Вікно порівняння файлів відображає два файли, відкритих в редакторі в двох панелях (Рис. 26).

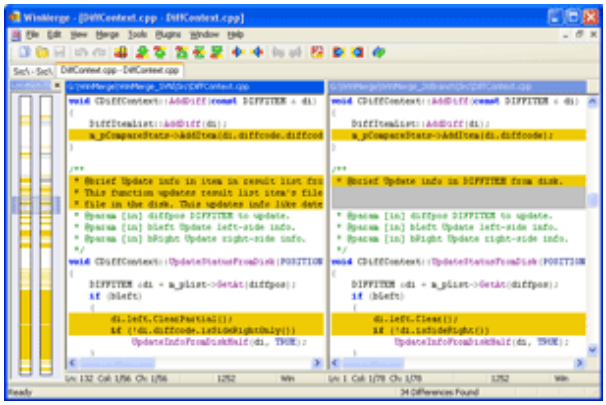


Рисунок 26- програма WinMerge

Редагування дозволяє користувачеві легко робити невеликі зміни без необхідності відкривати кожен раз файли в редакторі або IDE(Рис. 27).

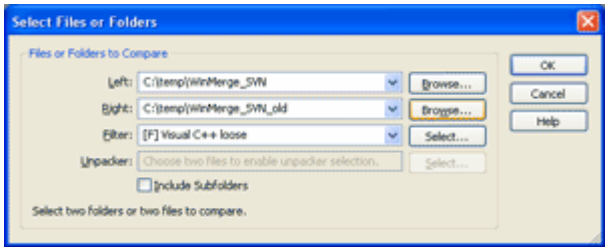


Рисунок 27- рудагування файлів

WinMerge дозволяє вибирати / відкривати шляхи і файли різними шляхами. Імпользованіе відповідного вікна - один з них(Рис. 28).

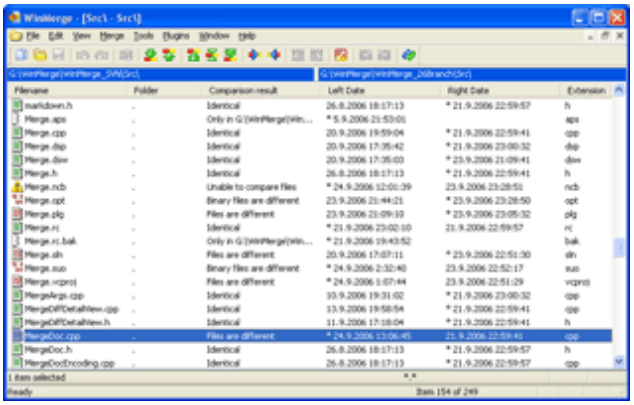


Рисунок 28- вибирання файлу

Функція порівняння каталогів показує всі файли і підкаталоги з папок, які порівнюються, у вигляді списку. Порівняння каталогів дозволяє проводити синхронізацію каталогів копіюванням або видаленням файлів і підкаталогів. Цей функціонал може бути різнобічно налаштований на свій розсуд(Рис. 29).

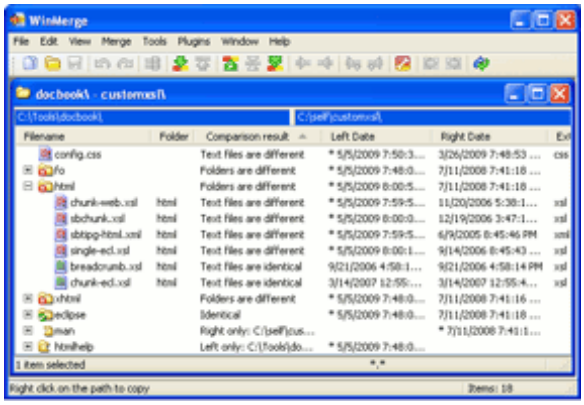


Рисунок 29- порівняння каталогів

В цьому режимі директорії, в яких знаходяться файли, можуть розгортатися і згортатися. Це зручно для організації більш простий навігації по структурі каталогів. Режим перегляду дерева доступний тільки в рекурсивном порівнянні(Рис. 30).

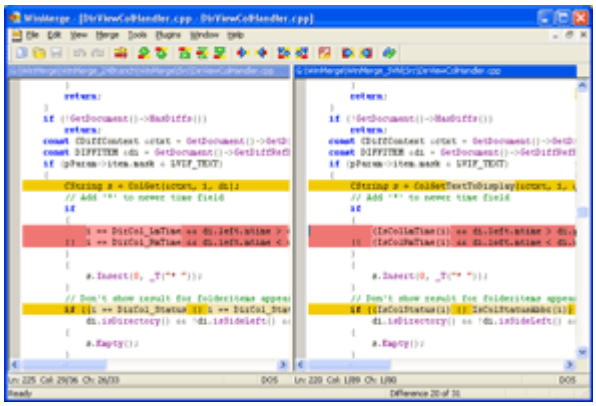


Рисунок 30- рекурсивне порівняння

Часто виникає необхідність побачити точне відмінність у середньому рядку. WinMerge може підсвічувати різноманітні області між рядками (Рис. 31).

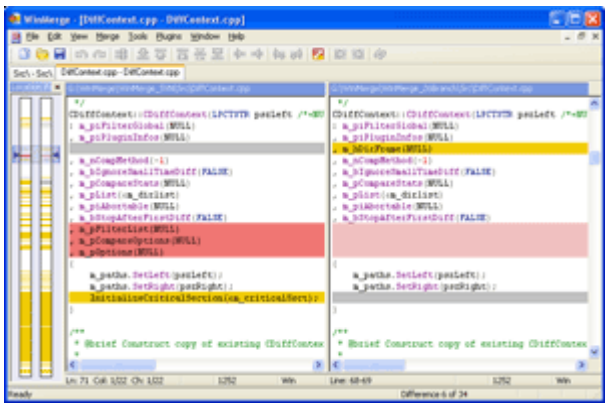


Рисунок 31- точні відмінності між рядками

Область с полосами местоположения показывает общую картину разницы файлов, которые сравниваются.

3.4 Програма для порівняння текстових файлів Araxis Merge

Araxis Merge - Мультиплатформенне додаток, що має свої версії в системах Windows і Mac OS X.

Додаток поширюється у вигляді DMG-образу. Крім самого додатка, до складу образу входять зразки AppleScript-скриптів, які можна використовувати в роботі, зразки файлів для порівняння і кілька утиліт, які можна використовувати разом з додатком у режимі командного рядка. На сайті розробника можна знайти докладну інформацію про те, як встановлювати ті чи інші компоненти, що входять в дистрибутив програми.

Додаток Araxis Merge, традиційно для більшості mac-програм, має простий і зрозумілий інтерфейс, що дозволяє навіть без підтримки рідного для користувача язикапонять зрозуміти принцип роботи програми та порядок

його використання. До слова сказати, українська, а рівно і будь-яка інша локалізація, відмінна від англійської, в Araxis Merge взагалі відсутнє. Вікно програми має акуратний, без особливих графічних вишукувань, інтерфейс, що включає два вікна для зіставлення файлів і панель інструментів, яка розташувалася у верхній частині вікна(Рис. 31).

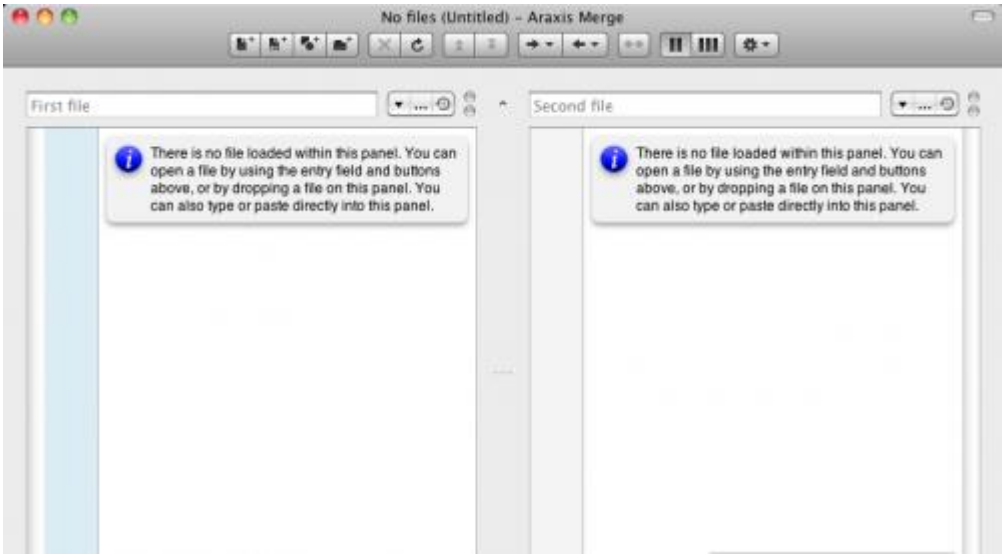


Рисунок 32- Вікно програми[6]

Araxis Merge - досить сучастний додаток, який вміє порівнювати між собою не тільки текстові файли. Можна виділити 4 основні режими роботи програми: порівняння текстових файлів, порівняння папок, порівняння зображень і, нарешті, порівняння бінарних файлів (програм). Для запуску кожного режиму існує окремі пункти в меню програми, як варіант, окремі групи кнопок на панелі інструментів. При запуску додатку вікно відкривається в стандартному режимі порівняння, який встановлюється в меню налаштувань (Рис. 32).

Рисунок 33- меню налаштувань[6]

Принцип роботи досить простий: в кожну з панелей додається по одному файлу, а потім за кількома критеріями порівнюється їх вміст. Наприклад, текстові файли можуть порівнюватися один з одним в режимах лінія-лінія, слово-слово, символ-символ. Додавання файлів здійснюється декількома способами. Перший, і найпростіший - перетягуванням файлів на кожну з панелей вікна програми. Другий, традиційний не тільки для Windows, але і частково для Mac OS X - через меню програми (пункт File-> Open File ...). Там же знаходиться ще один пункт меню, що дозволяє відкрити текстовий файл, використовуючи більш 100 різних кодувань (Рис. 33).

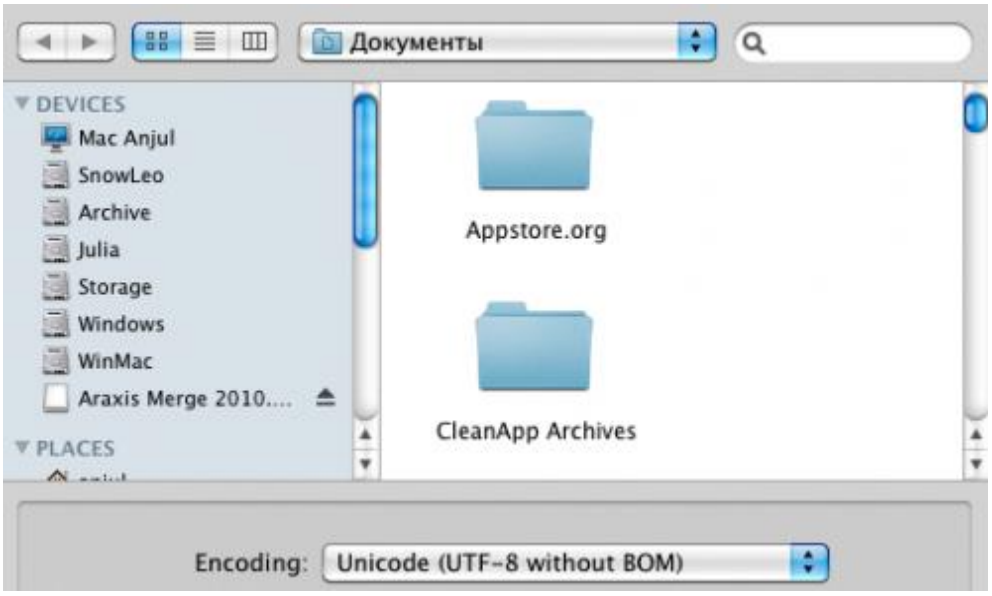


Рисунок 34- пункт меню, що дозволяє відкрити текстовий файл[6]

Третій, найбільш зручний, спосіб додавання файлів для порівняння - використання невеликої тулбару, розташованого в верхній частині кожної

панелі. Цей спосіб зручний, насамперед тим, що пропонує цілих три варіанти імпорту файлів при мінімумі рухів мишкою по екрану. Дивіться самі: можливо додавання файлів не тільки через вікно Finder, а й використовуючи історію відкриття файлів, причому журнал ведеться окремо для кожної з панелей. Нарешті, натиснувши на кнопку Available Versions, можна відкрити одну з попередніх версій файлу, якщо користувач їх вже не видалив(Рис. 34).

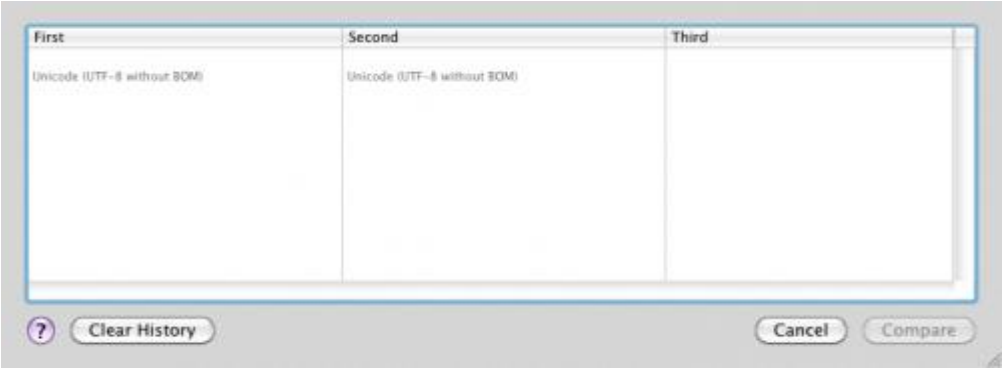


Рисунок 35- відкривання попередньої версії якщо її видалили[6]

Araxis Merge дійсно вміє порівняти не тільки два, але також і три файли одночасно. Для відображення третьої панелі можна скористатися відповідною кнопкою на панелі інструментів (у вигляді римської цифри III). Для повернення назад в режим двох панелей слід скористатися кнопкою «II«. В роботі, як правило, найчастіше порівнюються тільки два файли, а не три, і тому режим трьох вікон за замовчуванням не використовується.

3.4.1 Порівняння текстів

В процесі тестування програми завантажили в обидві панелі зразки тестових файлів, які йшли в комплекті з додатком. Як видно на скріншоті, різні типи змін відзначаються різними кольорами. Так, наприклад, якщо в одному з файлів присутністю текстовий блок, відсутній в іншому файлі, то цей блок підсвічується червоним або зеленим кольором, причому лініями від

однієї панелі до іншої показується місце в тексті, де на думку програми, мав би перебувати цей відсутній абзац. Зміни не такі глобальні, а стосуються окремих слів або символів, також підсвічуються окремим кольором. А в самому низу екрану, на інфо-панелі буде показана кодування текстових файлів, кількість і характер знайдених відмінностей, а також загальне число рядків і колонок в тексті(Рис. 35).

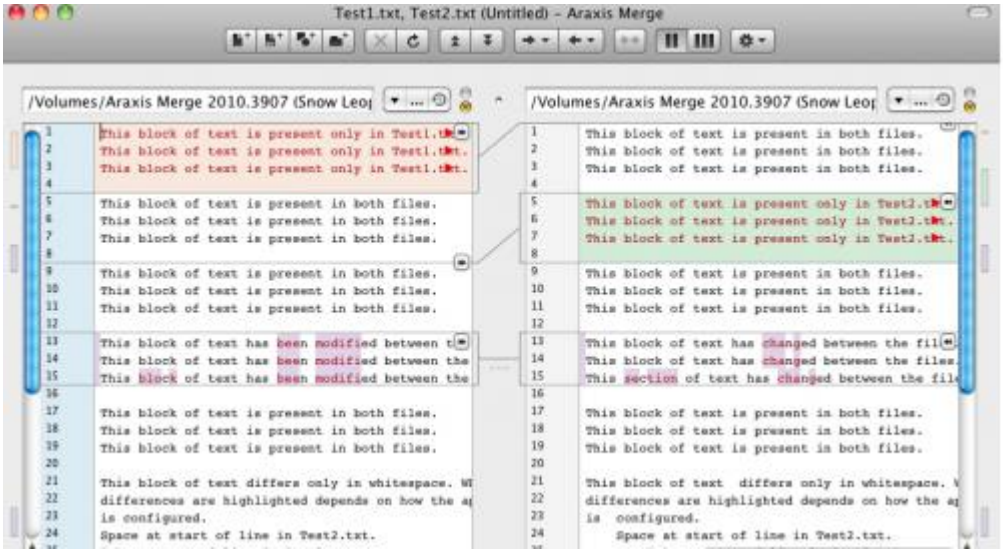


Рисунок 36- Невідповідності у тексті[6]

Araxis Merge - вона вміє не тільки порівнювати, але об'єднувати текстові файли між собою. Досить скористатися однією з двох кнопок, розташованих все на тій же панелі інструментів, як зміст лівої панелі буде записано поверх змісту правої, і навпаки, зміст правої замінить зміст лівої. Ще однією характерною особливістю програми є можливість редагування файлів «на місці», прямо в вікні Araxis Merge. Якщо порівнювані файли мають статус «для читання», то замінити вміст одного файлу іншим або просто відредагувати його не вийде. Про наявність цього режиму буде сигналізувати індикатор жовтого кольору, розташований в правій частині міні-тулбару кожної з панелей(Рис. 36).

Рисунок 37- міні-тулбар[6]

3.4.2 Порівняння зображень

На першому скріншоті видно, що при практично однаковому змісті, зображення мають певні відмінності(Рис. 37).

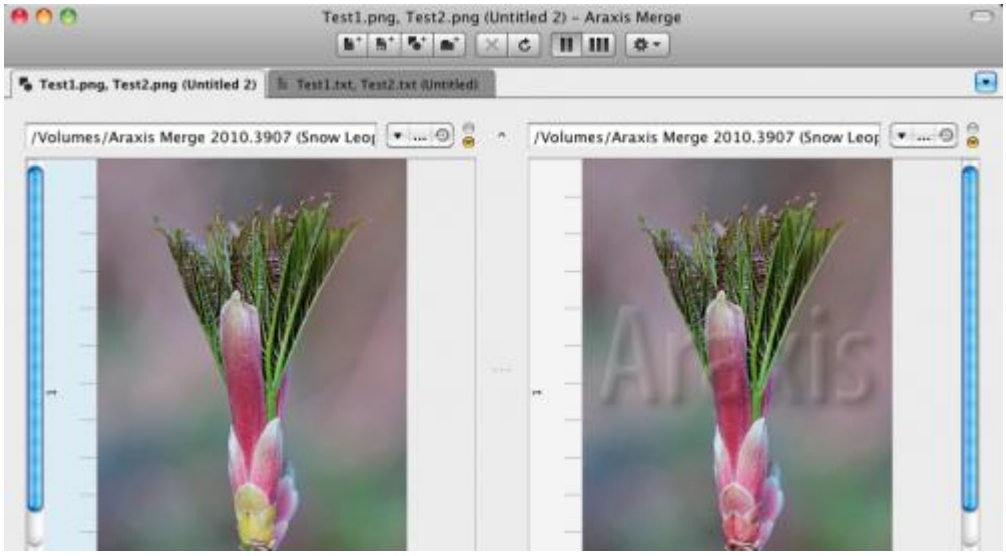


Рисунок 38- порівняння зображень[6]

Добре, якщо зображення мають відмінності в досить великих деталях, які можна відразу помітити, а якщо зміни будуть стосуватися дрібниць? Тут на допомогу приходять інструменти з нижньої панелі інструментів. Ці інструменти, зокрема, дозволяють приховати / показати всі однакові деталі, приховати / показати відмінності, або, як варіант, підсвітити відмінності на обох зображеннях. Використання інструменту Swap дозволить плавно поєднати обидва зображення, аж до того, що вони стануть ідентичні один одному або, як варіант, поміняються місцями один з одним. Повзунок Scale

дозволить плавно і одночасно збільшити обидва зображення, щоб мати можливість розглянути дрібні деталі(Рис. 38).

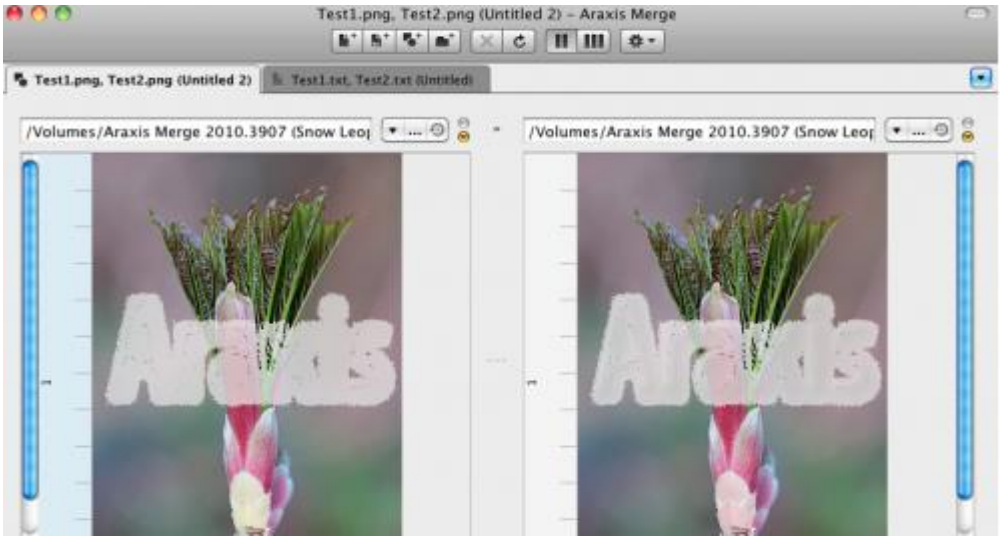


Рисунок 39- результати порівняння зображень[6]

На відміну від текстових файлів, файли зображень можна об'єднати між собою.

Як можна помітити на останньому скріншоті, Araxis Merge в своїй роботі використовує систему вкладок, маючи кожну нову пару або трійку порівнюваємих файлів в окремій вкладці, дозволяючи не загружати Робочий стіл масою відкритих вікон.

3.5 Програма для перевірки на плагіат Advego Plagiatus

Програма Advego Plagiatus широко використовується рерайтерами і копирайтерами для перевірки статей на унікальність.

Переваги програми:

- Приємний і зрозумілий інтерфейс;

- Безліч налаштувань;
- Робота з 5 пошуковими системами (Google, Bing, Nigma, Яндекс, Yahoo);
- Наявність вбудованого редактора тексту;
- швидкодія;
- Точність результатів;
- Уважне ставлення розробників: оновлення, виправлення помилок, додавання додаткових опцій.

Скориставшись таким помічником, замовник може бути впевнений в унікальності матеріалу, а копірайтер або рерайтер зарекомендує себе як відмінного фахівця.[3]

Програма поширюється безкоштовно. Завантажити її можна прямо на сайті розробника(Рис. 40):

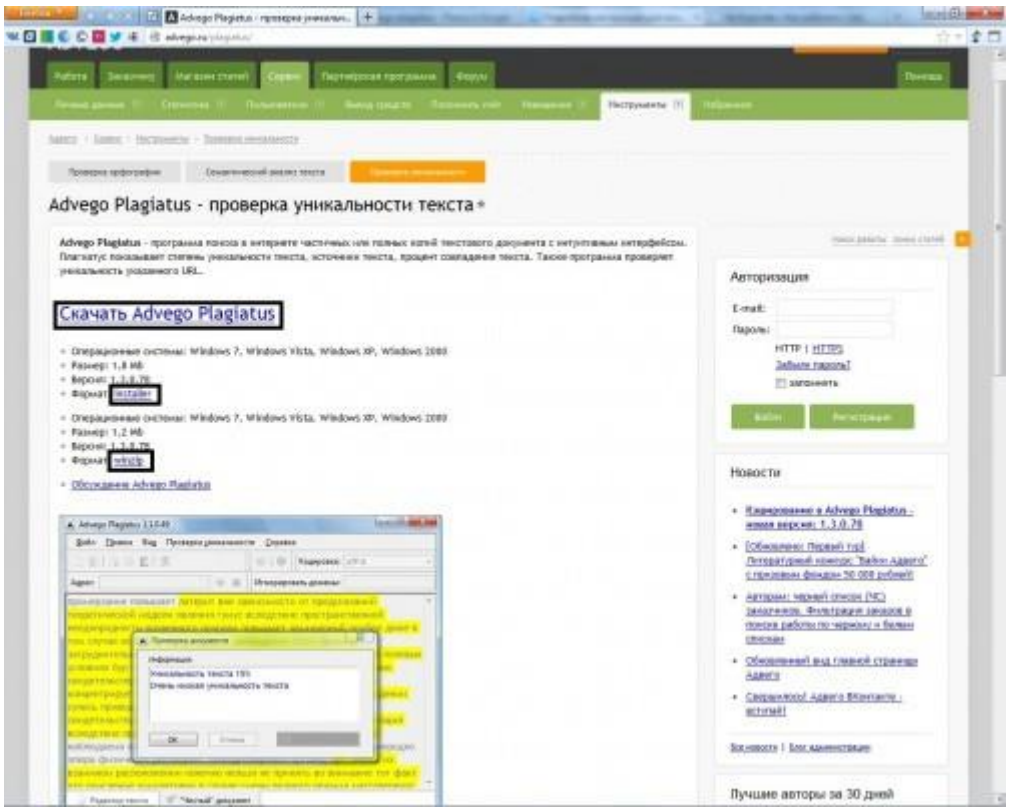


Рисунок 40- Сайт програми

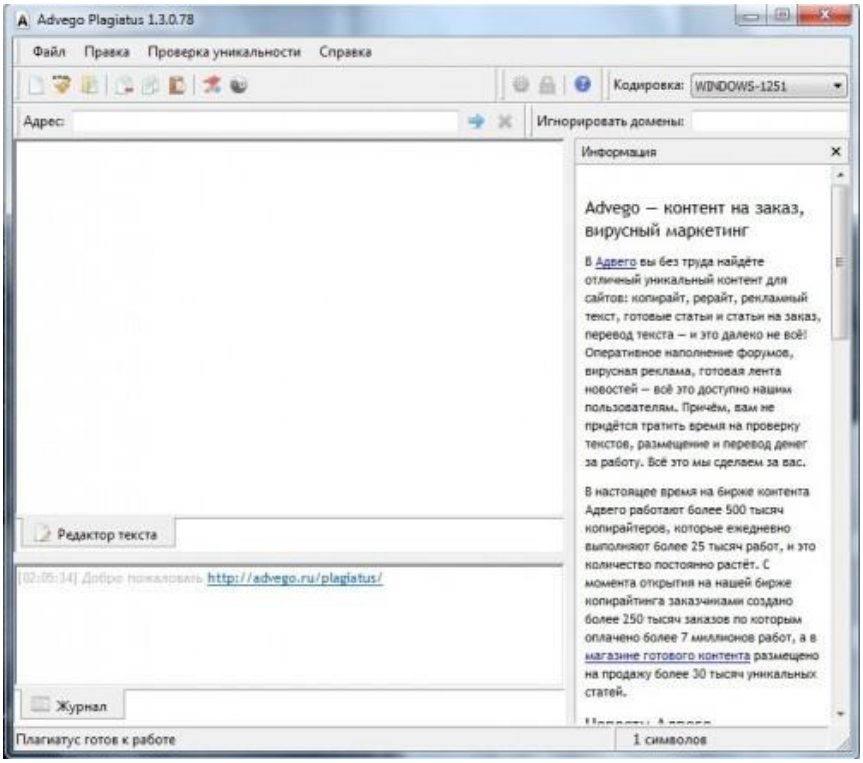


Рисунок 41 - Інтерфейс програми

Інтерфейс Advego Plagiatus розділений на декілька блоків:

Панель управління;

Робоче поле;

Поле результатів;

Блок інформації.

Саме віконце можна розтягнути хоч на весь екран. Це зручно при роботі з великим обсягом матеріалу. Блок інформації (праворуч) можна закрити, натиснувши на хрестик.

Панель управління являє собою перелік функцій. Якщо навести курсор і клацнути на «Файл» або будь-який інший пункт, відкриється список, що випадає.

У другому рядку винесені кнопки, для спрощення взаємодії з функціями програми, а також вікно з вибором кодування символів. За замовчуванням це Windows-1251.

Третій рядок включає в себе два текстових поля:

«Адреса» дозволяє перевірити унікальність інтернет сторінки (сюди вставляється посилання, після чого потрібно натиснути на синій хрестик і вибрати метод перевірки);

«Ігнорувати домени» - тут можна вказати домени, які будуть проігноровані при перевірці унікальності.

Наступне поле - редактор тексту. Туди поміщається матеріал, який потрібно перевірити на унікальність. Обсяг статті не обмежений, але чим більше символів, тим довше вона буде перевірятися.

До речі, кількість символів без пробілів зазначено в правому нижньому кутку програми, в рядку стану.

Останнє поле - «Журнал», блок результату. Тут поетапно описується хід перевірки, а також її результати. Журнал стирається відразу після закриття програми.

Якщо за один сеанс роботи перевірити кілька статей, кожен факт перевірки буде підписано відповідно. Відвівши бігунок вгору, можна подивитися результати за минулі перевірки в цьому сеансі роботи.[4]

Базові настройки програми

Налаштування програми Адвего Плагіатус можна відкрити, натиснувши на шестірню в другому рядку, або відкривши пункт «Перевірка унікальності» в першій, і вибравши у спадному меню пункт «Налаштування» (Рис. 42):

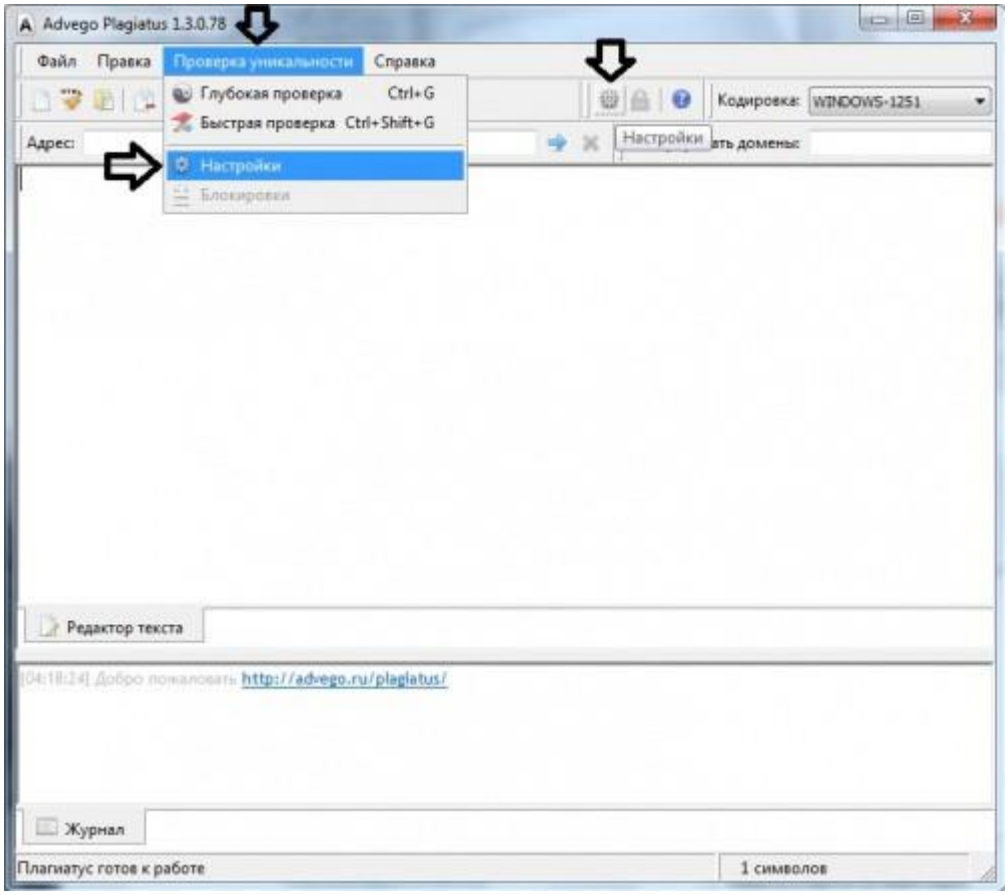


Рисунок 42 - Перевірка на плагіат

За замовчуванням вже виставлені оптимальні настройки, тому багато користувачів до них не звертаються. Втім, тут є досить зручні функції, не звертати на них уваги - собі на шкоду:

Пошук

Розділ «Пошук» - це широка налаштування функціональності програми. Тут можна:

Підкоригувати нижню планку унікальності (в залежності від вимог до тексту в кожному конкретному випадку);

Налаштувати облік збігів з одного ресурсу (у відсотках);

Змінити «Розмір шингли».

Зверніть увагу:

Шингл - це кількість слів, що йдуть підряд. Текст автоматично ділиться на шингли і програма «проганяє» їх з пошуку. Чим менше розмір шингли, тим більше буде збігів.

Золотою серединою вважається розмір шингли в 4 слова, але деякі замовники вимагають від копірайтерів змінити значення, наприклад, на 3. Якщо таких вимог немає, змінювати нічого не потрібно.

Змінити «Розмір фрази»:

Адвего Плагіатус посилає в пошуковик кілька слів (в залежності від розміру фрази), укладених в лапки. Значення за замовчуванням є оптимальним для роботи. Змінювати його потрібно в тому випадку, якщо умови роботи того вимагають;

У пункті «Пошукова система» можна вибрати додаткові пошукові системи. За замовчуванням це Google і Яндекс.

Перевірка унікальності в Advego Plagiatus

Готовий текст потрібно вставити в редактор тексту в головному вікні програми. Далі натиснути на одну з кнопок - швидка перевірка (червоний прапорець) або глибока перевірка (значок інь і янь).

Перший метод випадково підбирає кілька пошукових фраз, другий - поетапно і точно проганяє кожну з них. Краще вибирати глибоку перевірку(Рис. 43):

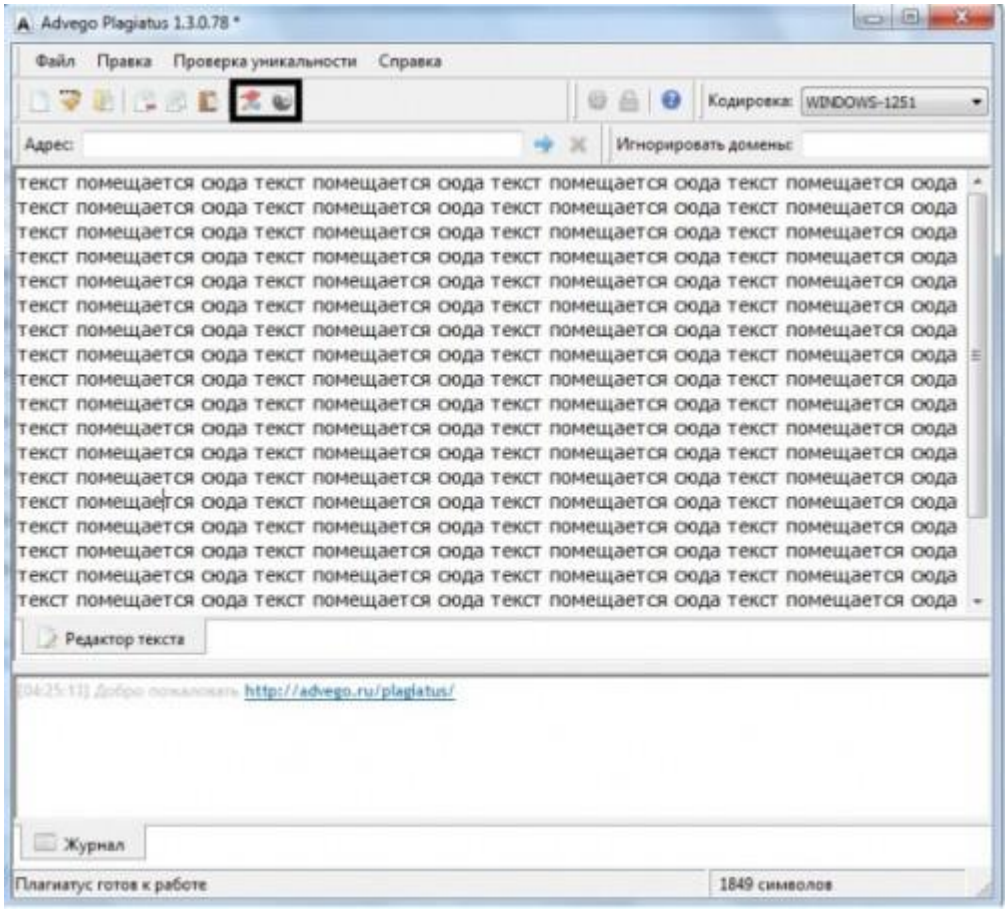


Рисунок 43- перевірка на унікальність

Через деякий час висвічується результат перевірки (унікальність тексту n% / n%. Вердикт). Число зліва - кількість оригінального тексту без збігів. Праворуч - підозра на рерайт(Рис. 44).

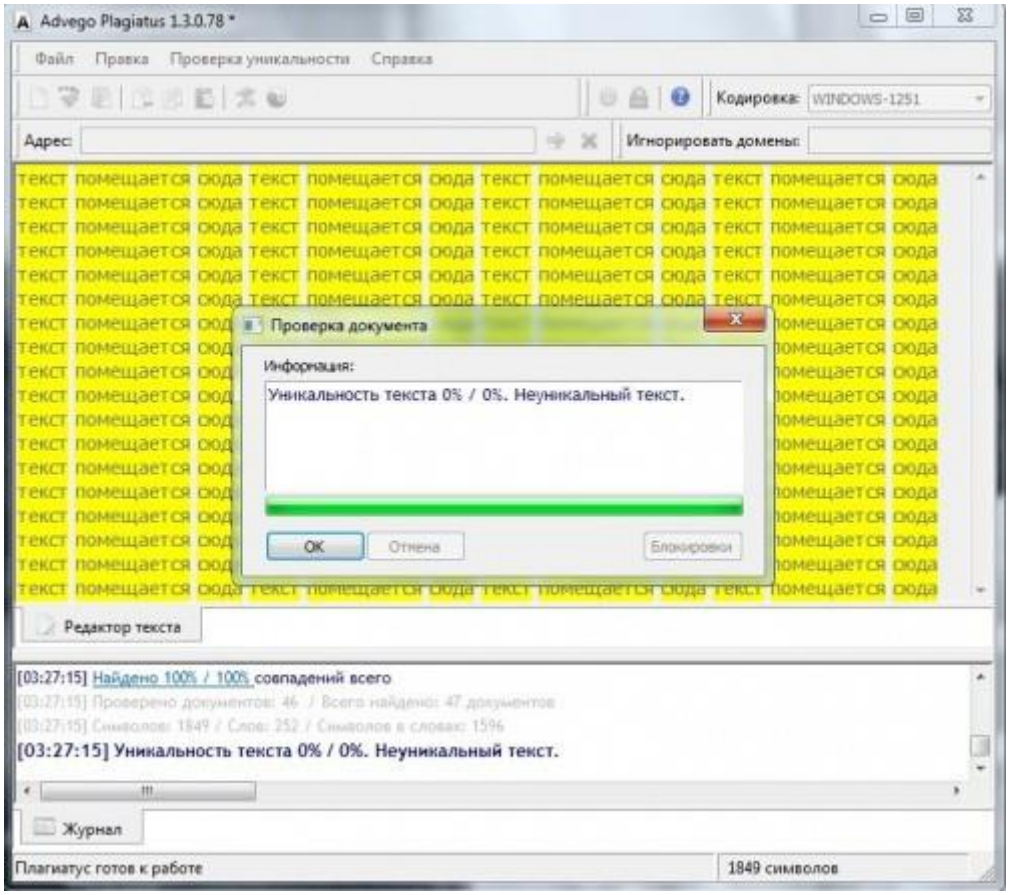


Рисунок 44- Результати перевірки на унікальність

Неунікальний текст підсвічується жовтим маркером.

Програмні вимоги продукту

3.6 Програма для перевірки на плагіат Etxt Антиплагиат

Можливості Etxt Антиплагиат

- Визначати унікальність тексту по збереженим копіям 7 пошукових систем (Яндекс, Google, Bing, Rambler, QIP, Nigma, Yahoo).
- Виділяти неунікальні фрагменти кольором і підраховувати відсоток збігів.
- Перевіряти на унікальність контент на всіх сторінках сайту.
- Виконувати перевірку тексту на поверхневий рерайт.[5]

- Вести пакетну перевірку текстів, збережених в одній папці.
- Порівнювати два завантажених тексту.
- Перевіряти на унікальність зображення.
- Видавати основні параметри сайту (PR, ТІЦ, число сторінок в каталогах пошукових систем і ін.) При SEO-перевірці.

Всі можливості програми можна побачити, відкривши меню «Операції» (Рис. 45).

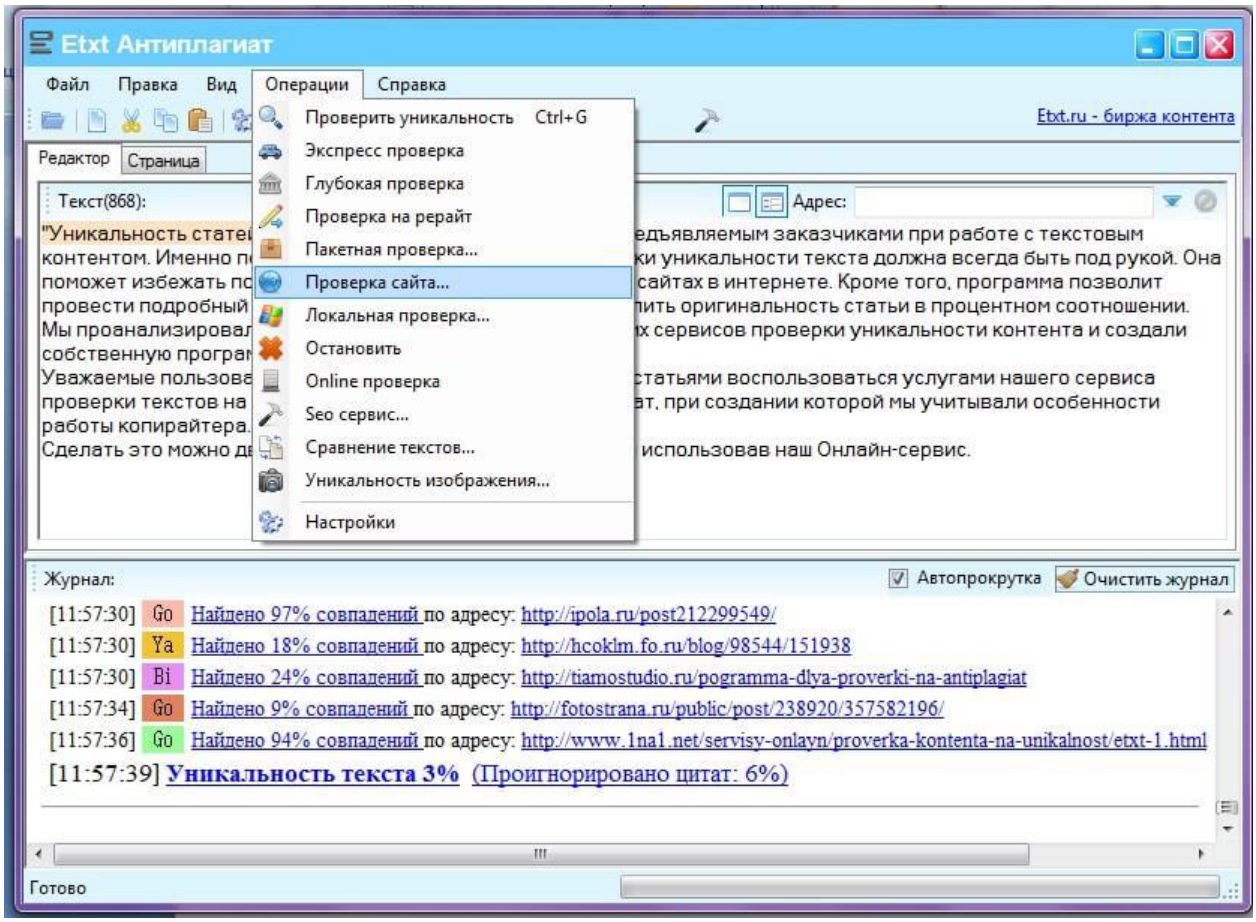


Рисунок 45- меню «Операції»

Настройка

Вкладка «Загальні»

- Відзначте галочками пошукові системи, до індексу яких повинна звертатися програма. Зазвичай для коректної перевірки досить вибрати Яндекс, Google, Rambler, Bing, Qip та Yahoo.
- Встановіть поріг унікальності - відсоток збігів, при досягненні якого програма продовжує перевірку. Для найбільш точного результату можна задати значення «0». В іншому випадку програма буде перевіряти текст до першого дубля, що збігається мінімум на 50% і зупиниться, так і не знайшовши всіх інших збігів.
- Вкажіть кількість слів у шинглів - мінімальному досліджуваному фрагменті тексту. Найчастіше при перевірці використовується значення «3» (Рис. 46).

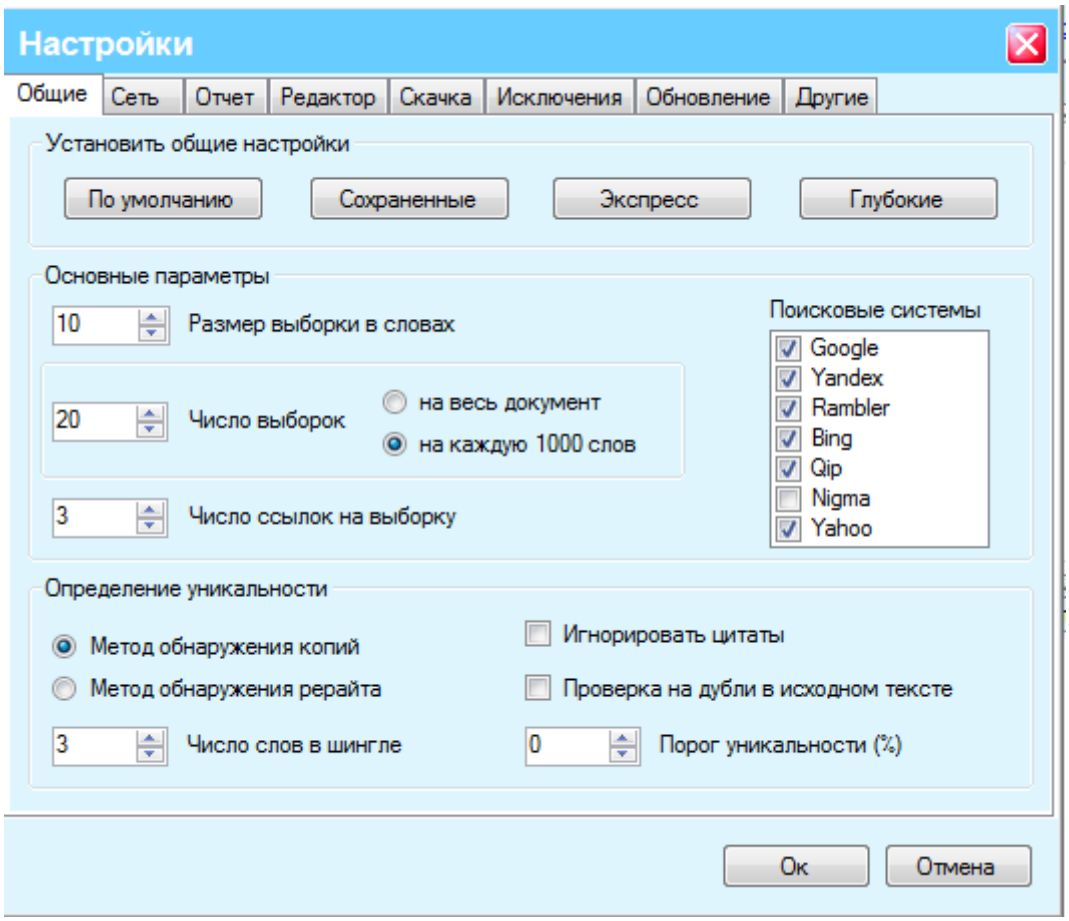


Рисунок 46- Настройки

Вкладка **«Інші»**

Від таких установок залежить як глибина перевірки, так і швидкість роботи програми. Оптимального балансу дозволяють домогтися наступні настройки.

- Мінімальний інтервал між суміжними запитами до пошукача - 5.
  При підвищенні цього значення перевірка буде повільніше, зате доведеться рідше вводити капчу.
- Максимальне число спроб здійснення запитів - 20.
  Чим частіше програма буде знову звертатися до ПС в разі невдалої спроби, тим глибше перевірка і більше запитів капчи.
- Таймаут закачування - 15.
  При підвищенні цього значення знижується швидкість перевірки, але при цьому починають враховуватися збіги з «важкими» сторінками і файлами pdf, на завантаження яких потрібно більше часу.
- Максимальна кількість завантажуються одночасно сторінок - 10.
  Чим більше цей показник, тим швидше перевірка і частіше запит капчи.
- Для коректної роботи Etxt Антілагіата також необхідно поставити галочку в розділі «Показувати капчу (Ya, Ni, Ma, Qi, Ra, Go)» на тій же вкладці(Рис. 47).

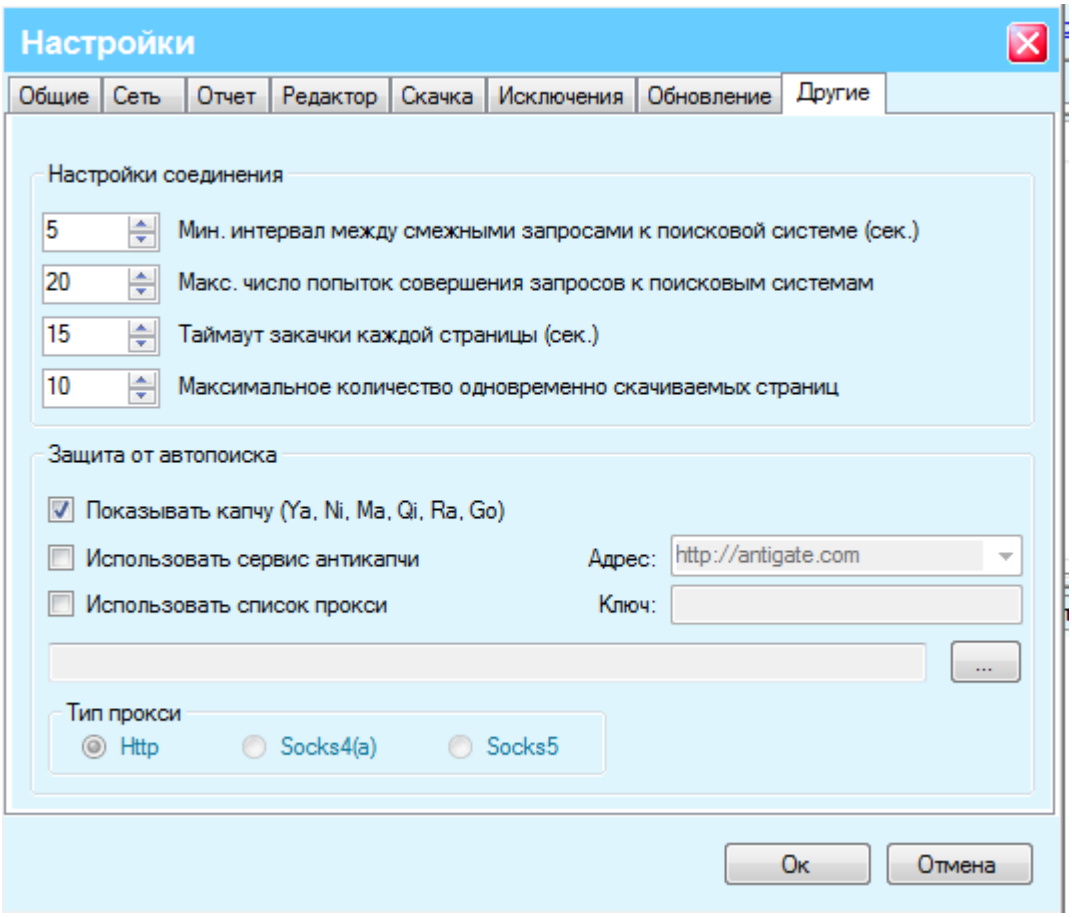


Рисунок 47- мінімальні налаштування для перевіркм тексту

Всі інші настройки можна залишити «за замовчуванням».

Зрозуміло, ви можете самі змінювати налаштування на ці та інші вкладках в залежності від розв'язуваних завдань. Якщо ваш текст буде перевіряти на унікальність замовник, простежте за тим, щоб у вас збігалися настройки Etxt Антиплагіат щоб уникнути непорозумінь.

Перевірка унікальності

Спосіб 1. Скопіювати текст у вікно програми і натиснути «Перевірити унікальність».

Спосіб 2. Скопіювати текст, натиснути правою кнопкою миші по вікну програми і натиснути «Вставити і перевірити».

Спосіб 3. Клацнути «Файл» - «Відкрити», вибрати потрібний текстовий

документ, натиснути «Перевірити унікальність» (Рис. 48).

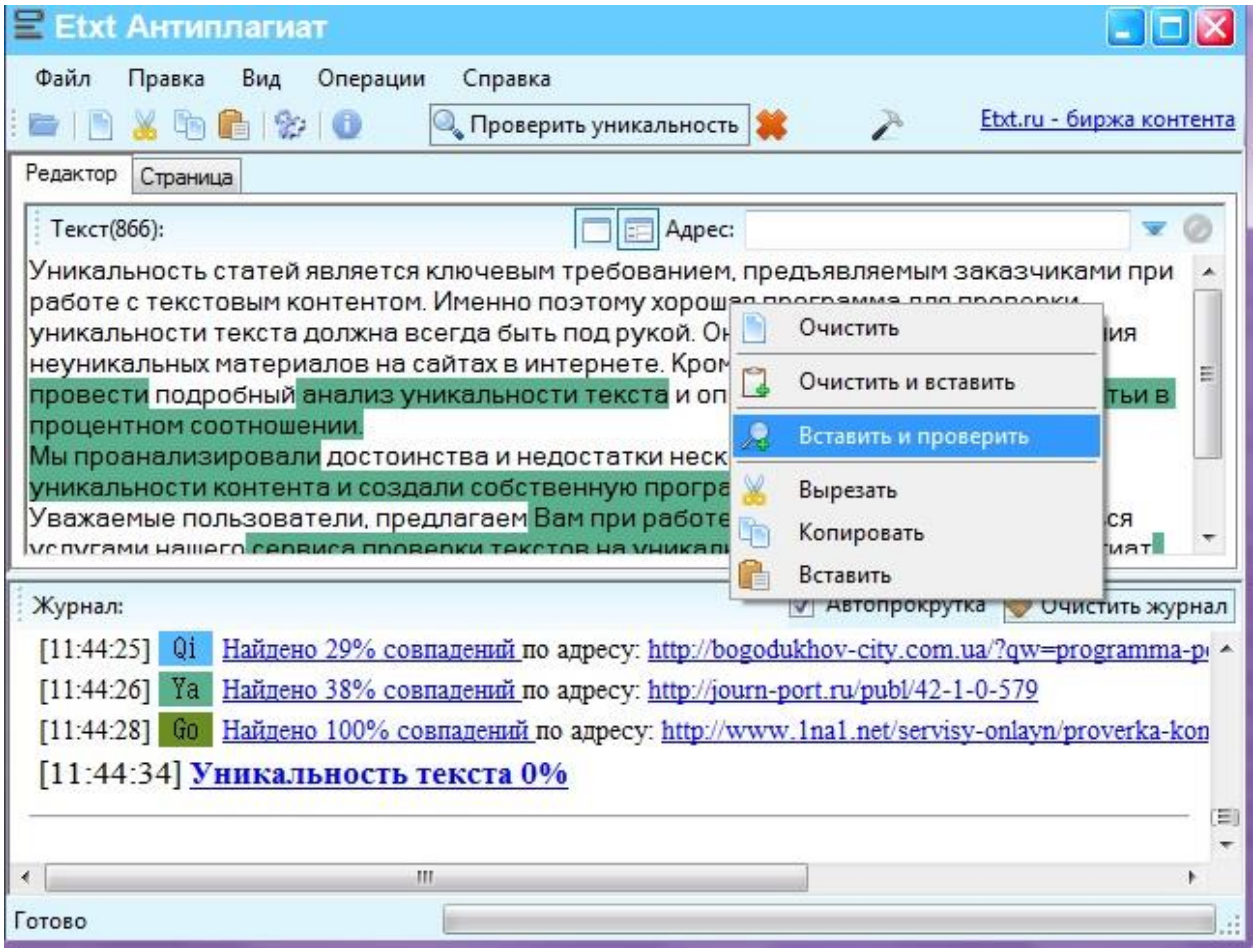


Рисунок 48- Перевірка на унікальність

Налаштування виключень

Зручна опція, якщо ви робите рерайт окремих фрагментів тексту і вам не важливі збіги з однією або декількома сторінками сайту клієнта. В цьому випадку можна виключити при перевірці збігу з однієї вихідної сторінкою або всіма сторінками домену.

Насамперед скопіюйте і збережіть в «Блокноті» URL сторінок, які не слід враховувати при перевірці на унікальність. Зайдіть в Etxt Антиплагіат, клікніть по іконці «Налаштування», відкрийте вкладку «Винятки», поставте

галочку поруч з потрібною опцією і завантажте щойно збережений файл з адресами(Рис. 49).

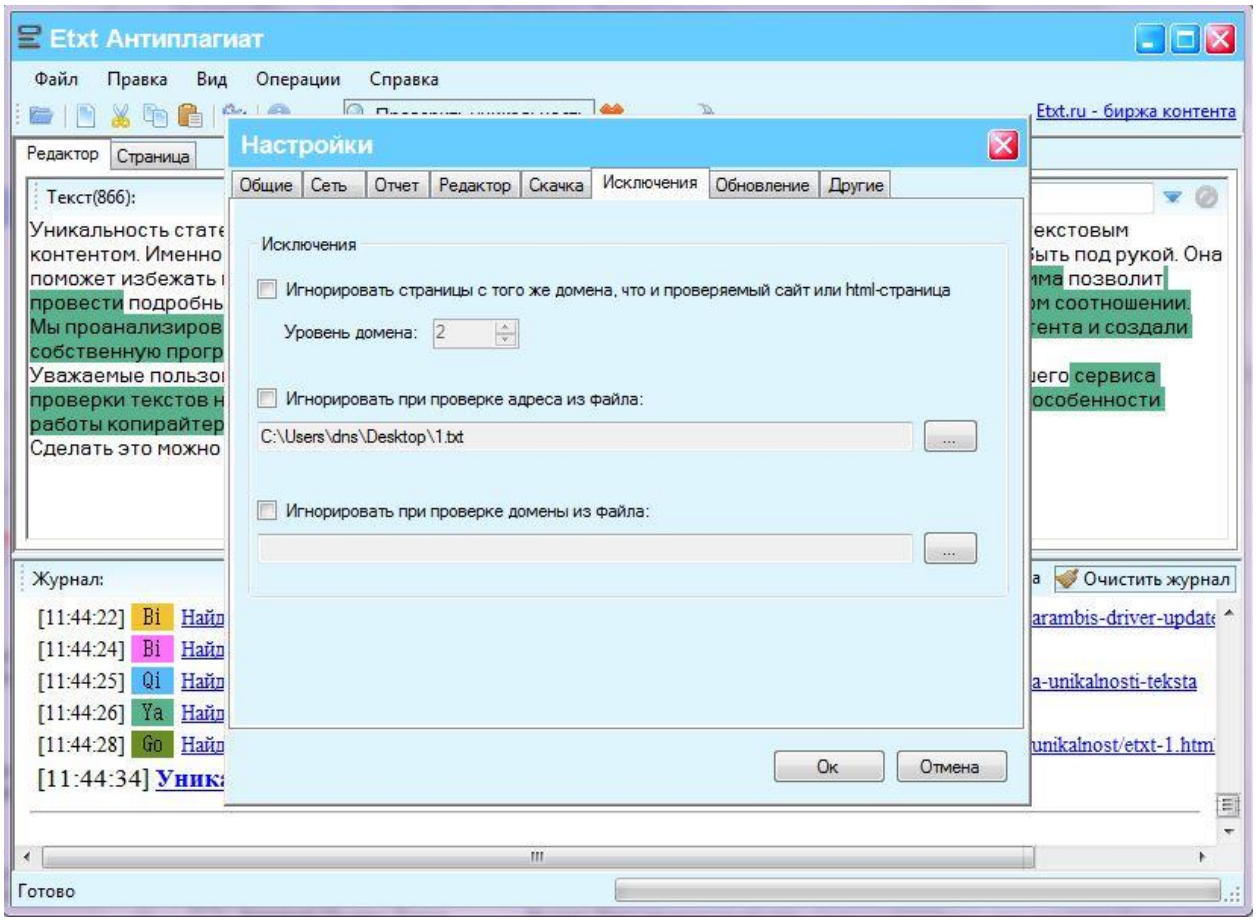


Рисунок 49 - Завантаження перевірки

Ігнорування цитат

Цікава «фішка», яка допоможе при перевірці текстів з прямою мовою, витягами з нормативних актів і іншими елементами, які не можна або неможливо зарерайтіть. До речі, до них найчастіше ставляться назва компанії і контактні дані, які кочують з тексту в текст. Щоб включити ігнорування цитат, укладіть потрібні фрагменти в лапки в тексті і натисніть «Ігнорувати цитати» в настройках на вкладці «Загальні» (Рис. 50).

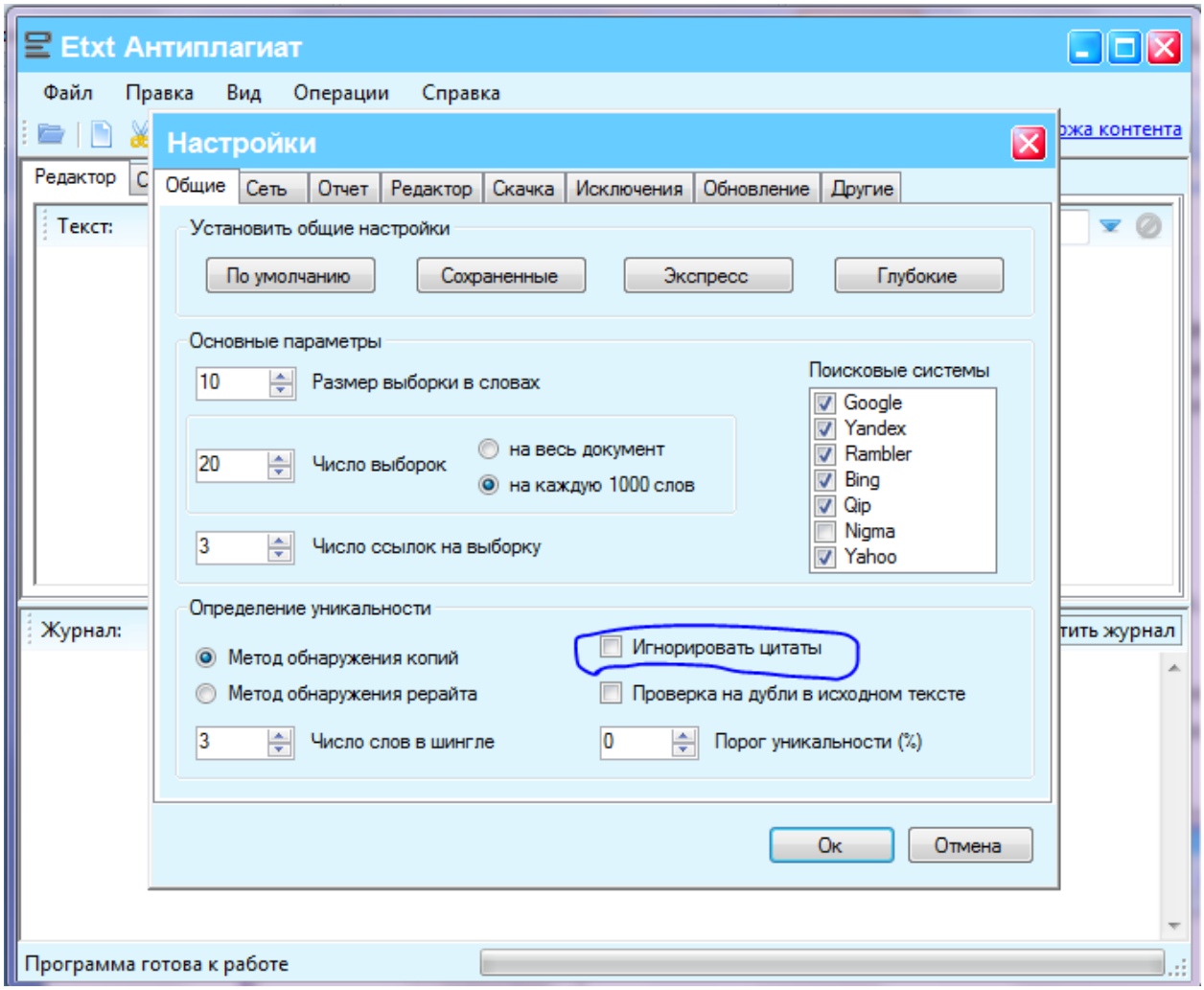


Рисунок 50 - уключення функції ігнорування цитат

Автозбереження

Відмінний варіант в тому випадку, якщо вам потрібен доступ до різних версій тексту, які ви кілька разів перевіряли на унікальність і потім редагували. А ще це справжній порятунок, якщо у вас з-за технічних проблем (наприклад, відключити електроживлення) не зберігся текст у використовуваному редакторі.

Зайдіть в меню «Файл», виберіть «Автозбереження» і клікніть по потрібному документу зі списку (за замовчуванням назви містять час і дату перевірки відповідного тексту, тому знайти потрібну версію буде неважко)

(Рис. 51).

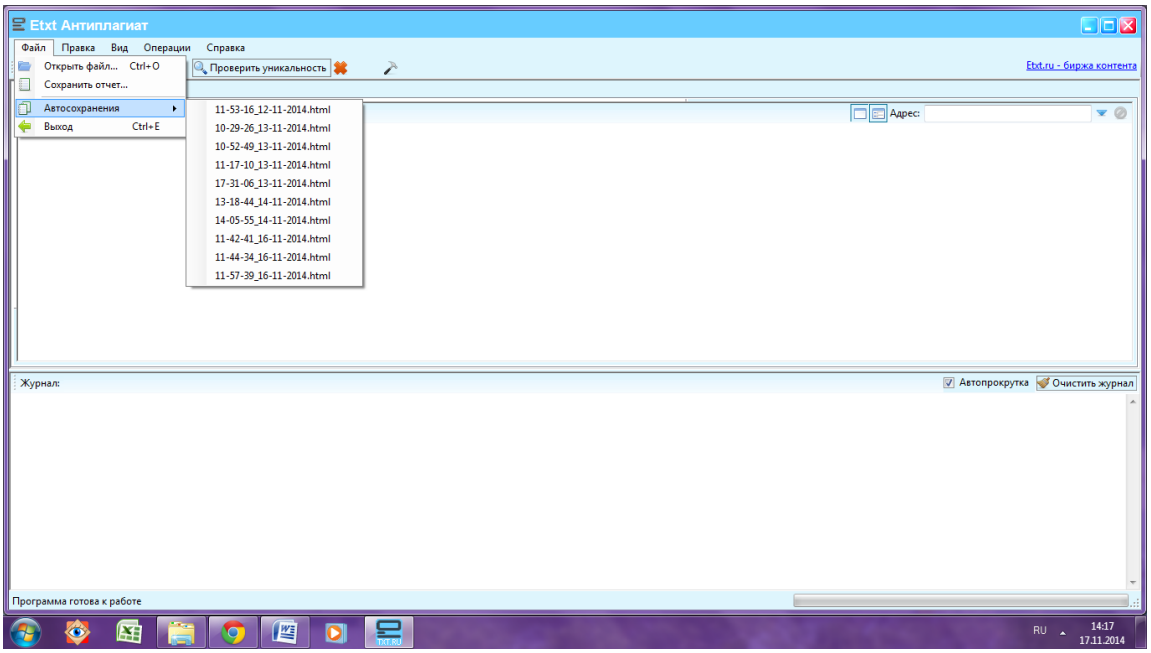


Рисунок 51- Автозбереження

Кількість автозбереження текстів можна змінити в настройках на вкладці «Звіт» (Рис. 52).

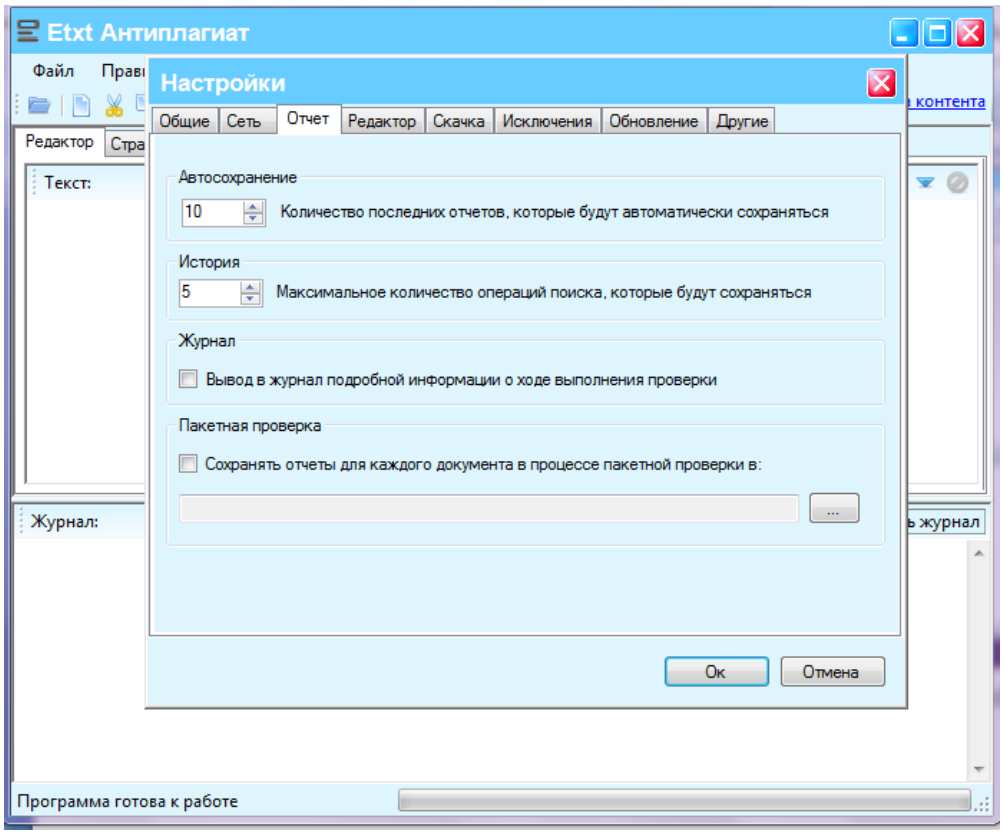


Рисунок 52- Вкладка звіт

І найголовніше: пристойною унікальністю тексту, яка порадує і замовника, і Яндекс або іншу ПС, вважається 95%. Однак повністю покладатися на Etxt Антиплагіат не варто: якщо якісь сайти випали з індексу пошукових систем, програма не знайде з ними збігів. Так що якщо текст вам здається підозріло унікальним, варто перевірити ще раз його ще раз через деякий час.

3.8 Онлайн сервіси для перевірки унікальності тексту Content-watch.ru

Розробники стверджують, що перевірка тексту проводиться за унікальною методикою, що не використовує шингли. Перевірка проводиться тільки на сайті без додаткового очікування і черг. Після реєстрації Ви можете перевірити 20 текстів на добу. Як правило, цього буває достатньо.

Сервіс має простий і зрозумілий інтерфейс без реклами і зайвих настирливих елементів. Швидкість, як і обіцяли розробники, дійсно дуже висока, при цьому немає черг на перевірку. Але, як і в будь-якому алгоритмі, Content watch дає збої: перевіряючи один і той же текст, Ви можете отримати різні результати. За весь час використання це траплялося зі мною лише двічі.

3.9 Онлайн сервіси для перевірки унікальності тексту Text.ru

Text.Ru - це і онлайн-сервіс (на даний момент кращий) перевірки текстів на унікальність, топова біржа копірайтингу і магазин готових статей одночасно. Сервіс надає можливість безкоштовної перевірки текстів на унікальність незареєстрованим користувачам обмежене, але досить велика,

число раз. Після реєстрації обмеження на кількість перевірок знімається, але залишається обмеження на обсяг - безкоштовно можна перевіряти тексти до 15 тисяч символів. При платних ж перевірках і у власників PRO-акаунтів знімається і це останнє обмеження, причому швидкість перевірки істотно зростає за рахунок зміни пріоритету черговості.

Text.Ru - перевірка тексту сайту на унікальність онлайн

По-перше, реєструємося на сайті. Потім натискаємо на кнопку «Подарунки» і активуємо подарунковий пакет символів (Рис. 53).

Рисунок 53- Онлайн сервіси для перевірки унікальності тексту Text.ru

3.10 Використання програмних засобів для перевірки документів в боротьбі з плагіатом

Плагіат (англ. Plagiarism) - це використання, перефразування і підведення підсумків роботи в будь-якій формі без підтвердження посиланнями на джерела і уявлення її як своєї власної роботи.

Плагіат з появою Інтернету перетворився в серйозну проблему. Потрапивши в Інтернет, знання стає надбанням всіх, дотримуватися

авторське право стає все важче і навіть неможливо. Поступово стає складніше визначити первісного автора.

Стрімкий розвиток мережі Інтернет поряд зі зростаючою комп'ютерною грамотністю сприяє проникненню плагіату в різні сфери людської діяльності: плагіат є гострою проблемою в освіті, промисловості та науковому співтоваристві .

Плагіат є злочином. Це вводить в оману читачів, приносить шкоду автору, і надає незаслужені блага плагіатор .

Широкий доступ до вітчизняної та зарубіжної літератури, багаторазове збільшення числа професійних видань, публікацій в Інтернеті - все це практично зводить нанівець будь-які було редакторські прагнення «перевірити» або «встановити» справжність і оригінальність аргументів і фактів, які використовуються в рукописах, пропонованих до публікації .

На ілюстрації представлена класифікація методів комп'ютерного виявлення плагіату з технічної точки зору.

Методи характеризуються по типу оцінки подібності.

Глобальна оцінка використовує великі частини тексту або документа для знаходження подібності в цілому, в той час як локальні методи на вході перевіряють обмежений сегмент тексту.

В даний час найбільш поширеним підходом є Дактилоскопія:

З ряду документів вибирається набір з декількох подстрок, які і є «відбитками». Розглянутий документ буде порівнюватися з «відбитками» для всіх документів колекції. Знайдені відповідності з іншими документами вказують на загальні сегменти тексту.

Перевірка документа дослівним перекриттям тексту представляє собою класичне порівняння рядків.

Перевірка підозрілих документів в цій ситуації вимагає розрахунку і

зберігання ефективно порівнянні подання всіх документів в довідковій колекції, які порівнюються попарно. Як правило, використовують моделі, такі як суфіксне дерево або суффіксний масив, які були адаптовані для виконання цього завдання в контексті комп'ютерного виявлення плагіату. Однак зіставлення подстроки є нежиттєздатним рішенням для перевірки великих колекцій документів (алгоритм відпрацьовує в середньому 2h порівнянь, де h - довжина рядка, в якій ведеться пошук).

Аналіз "безлічі слів" (англ.) Рос. є спрощенням уявлення, що використовується в обробці природної мови і пошуку інформації. У цій моделі текст представлений як невпорядкований набір слів. Документи представлені у вигляді одного або декількох векторів, які використовуються для попарного обчислення подібності.

Цитування - комп'ютерний метод виявлення плагіату, призначений для використання в наукових документах, що дозволяє використовувати цитати і довідковий матеріал. Визначає загальні цитати двох наукових робіт.

Шаблон цитат є підпослідовності, що містять не тільки загальні цитати для двох документів, а й подібний порядок і близькість цитат в тексті, що є основними критеріями для визначення шаблону цитат.

Стильометрія або вивчення мовних стилів - це статистичний метод для виявлення авторства анонімних документів і для комп'ютерної перевірки на плагіат.

Будуються стилометрическое моделі для різних фрагментів тексту, уривків, які стилістично відрізняються від інших. І шляхом порівняння моделей можна виявити плагіат.

Наприклад, аналіз на основі послідовностей частин мови. Розглядається спосіб розбиття тексту на фрагменти однорідності. Як параметри розбиття беруться різні послідовності частин мови. Далі проводиться аналіз фрагментів. І в результаті для тексту знаходяться

послідовності, які виділяли з текстів фрагменти, тобто алгоритм виділяє з тексту фрагменти неоднорідності, що мають різні частоти народження обраної послідовності частин мови, що показує на можливий плагіат в даному місці(Рис. 54).

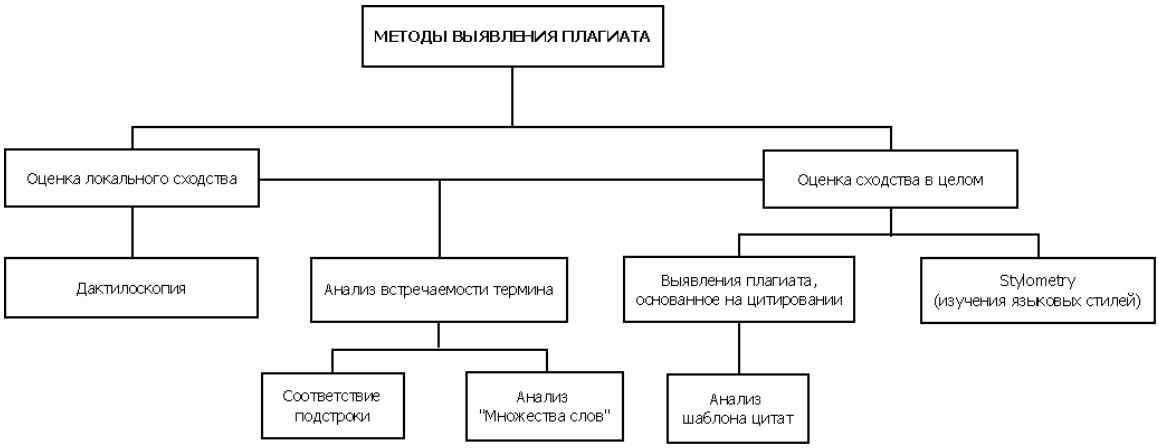


Рисунок 54- Методи виявлення плагіата

В даний час існує досить велика кількість сервісів і програм, що дозволяють будь-яким чином виявити запозичення. [1] У числі таких можна назвати: систему «Антиплагіат», Advego Plagiatus, Unplag, miratools.ru, istio.com, Praide Unique Content Analyser II, Plagiatinform, Copyscape (англ.) Рос .

1. Система «Антиплагіат»

   Система розроблена компанією «Форексис» [2]. Система здійснює онлайн пошук по великій кількості документів, що зберігаються в своїй основі системи, по базам даних партнерів, в тому числі: Російська державна бібліотека, Наукова електронна бібліотека ELibrary.ru, компанія Lexpro, а також по базі даних користувача. «Антиплагіат» здійснює пошук по мережі Інтернет власними засобами і тому має меншу оперативністю ніж системи, що використовують Яндекс. XML. У безкоштовної версії системи доступна тільки скорочена форма звіту.

2. Програма Advego Plagiatus

   Програма здійснює онлайн перевірку з використанням пошукових систем [2]. На відміну від аналогічних систем, Advego Plagiatus не використовує Яндекс.XML (безкоштовний сервіс, що надає можливість виробляти автоматичні пошукові запити до «Яндексу» і публікувати його видачу у себе на ресурсі). Програма видає відсоток збігу тексту і виводить знайдені джерела. Програма не перетворює літери, тобто немає перетворення регістру, немає обробки і зміни латинських букв в російських словах на аналогічні літери російського алфавіту для текстів російською мовою. Також відсутня підтримка пошуку по власній базі; через особливості роботи виникають ситуації, коли результати перевірки відрізняються від разу до разу.

3. сервіс Unplag

   Сервіс перевірки на плагіат Unplag [4] може здійснювати перевірку на плагіат як в режимі реального часу онлайн, так і порівнювати документ зі збереженою базою документів в бібліотеці користувача. Підтримує роботу з різними типами документів. Є персональна і корпоративна програми. Також працює з системою управління курсами Moodle, Canvas, Blackboard, Sakai .

4. сервіс www.miratools.ru

   Сервіс дозволяє здійснювати онлайн перевірку тексту на плагіат [7]. Система використовує результати видачі пошукових систем. Існує можливість заміни англійських букв на російські. Також є можливість зміни довжини і кроку шингли (англ.) Рос., Використовуваних для перевірки. За результатами перевірки видається відсоток збігів і знайдені

джерела. Система не працює з власною базою, існує обмеження на довжину тексту в 3000 символів і обмеження на кількість перевірок протягом доби.

5. сервіс www.istio.com

Сервіс здійснює перевірку тексту на наявність запозиченого контенту з використанням пошукових систем Яндекс.XML і Yahoo.com. [8]. За результатами перевірки видається повідомлення про те, чи є текст унікальним чи ні, і видається список подібних сторінок сайтів. Сервіс надає додаткові кошти для аналізу текстів, наприклад, перевірку орфографії, аналіз найбільш частотних слів і т. Д. У системи немає перетворення букв і пошук по власній базі.

6. Програма Praide Unique Content Analyser II

Програма перевіряє тексти з використанням пошукових систем [5]. Є можливість вибору використовуваних пошукових систем, містить засоби додавання нових пошукових систем. Перевірка здійснюється шинглі, довжину яких можна змінювати. Можна задавати кількості слів перекриття шинглів. Виводиться докладний звіт з перевірки в кожній пошуковій системі. У програмі відсутня заміни букв, обробка стоп-слів і немає підтримки роботи з власною базою.

7. система Plagiatinform

Система перевіряє документи на наявність запозичень як в локальній базі, так і в мережі Інтернет [6]. Система вміє знаходити плагіат у вигляді документів, скомпонованих з «перемішаних» шматків тексту декількох джерел. Перевірка може здійснюватися з використанням швидкого або

поглибленого пошуку. Результати перевірки видаються у вигляді наочного звіту. Відсутня перетворення букв. Відсутня можливість вільного використання або тестування системи.

8. сервіс Copyscape

Система Copyscape (англ.) Рос. дозволяє здійснювати пошук копій веб-сторінок в Інтернеті [7]. Система повертає список веб-сторінок, у яких є схожий за змістом текст. Сервіс здійснює перевірку на наявність запозиченого контенту з використанням пошукових систем Google і Yahoo!

Перевіряється тільки контент веб сторінки, тобто, для з'ясування унікальності тексту необхідно опублікувати текст на сайті і ввести в системі адреса сторінки. Без реєстрації існує обмеження на кількість перевірок на місяць і на кількість відображуваних результатів - 10 сайтів. Для зареєстрованих користувачів немає обмежень на кількість перевірок і виведених результатів, але кожен запит коштує 5 центів.

3.11 Висновки до розділу 3

В 3 розділі були протестовані функціональні можливості програм порівняння документів та онлайн сервісів антиплагіт.

Я проаналізував 5 основних популярних сервісів по перевірці унікальності текстів. Але мені очевидні на сьогоднішній день два лідери: Text.ru і Content-watch.ru.

Причому, Text.ru найкраще підійде замовникам, так як дозволяє провести максимально глибоку перевірку (нехай іноді і з помилковими спрацьовуваннями.

А ось виконавцю краще все-таки підійде Content-watch.ru або Miratools.ru (якщо ненагружен), так як в них набагато менше помилкових спрацьовувань і не доведеться нескінченно переписувати тексти. Аналізувати унікальність технічного рерайта в Text .ru - майже неможливо, так як сервіс реагує на найменші збіги з приводу і без.

Також була затронута тема плагату і, як програми для визначення плагіату допомагаю з ними боротися.

4.ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначеного для порівняння документів. Програмний продукт був розроблений за допомогою мови програмування C++ у середовищі розробки Qt. Інтерфейс користувача створений за допомогою технологій графічного інтерфейсу інструментарію Qt.

Програмний продукт є крос-платформним та рекомендується для використання на персональних комп’ютерах під управлінням операційних систем Windows та Linux.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

1. визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам:

2. ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.
3. для кожної функції визначаються повні річні витрати й кількість робочих часів.
4. для кожної функції на основі оцінок попереднього пункту визначається кількісна характеристика джерел витрат.
5. після того, як для кожної функції будуть визначені їх джерела витрат,
6. проводиться кінцевий розрахунок витрат на виробництво продукту.

4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА. Оскільки основні проектні рішення стосуються всієї системи, кожна окрема підсистема має їм задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу гетероскедастичних процесів в економіці та фінансах.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

1. програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;

2. забезпечувати високу швидкість обробки великих об’ємів даних у реальному часі;
3. забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;
4. передбачати мінімальні витрати на впровадження програмного продукту.

4.1.1 Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного продукту, який будує набір моделей прогнозу та перевіряє їх точність на певному наборі даних. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – розпізнавання вхідних даних;

F_3 – інтерфейс користувача.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування C++;

б) мова програмування Java;

Функція F_2:

а) написання алгоритмів вручну;

б) використання готових бібліотек;

Функція F_3:

а) інтерфейс користувача, створений за технологією Qt;

б) інтерфейс користувача, створений за технологією Windows Forms.

4.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 53). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 3).

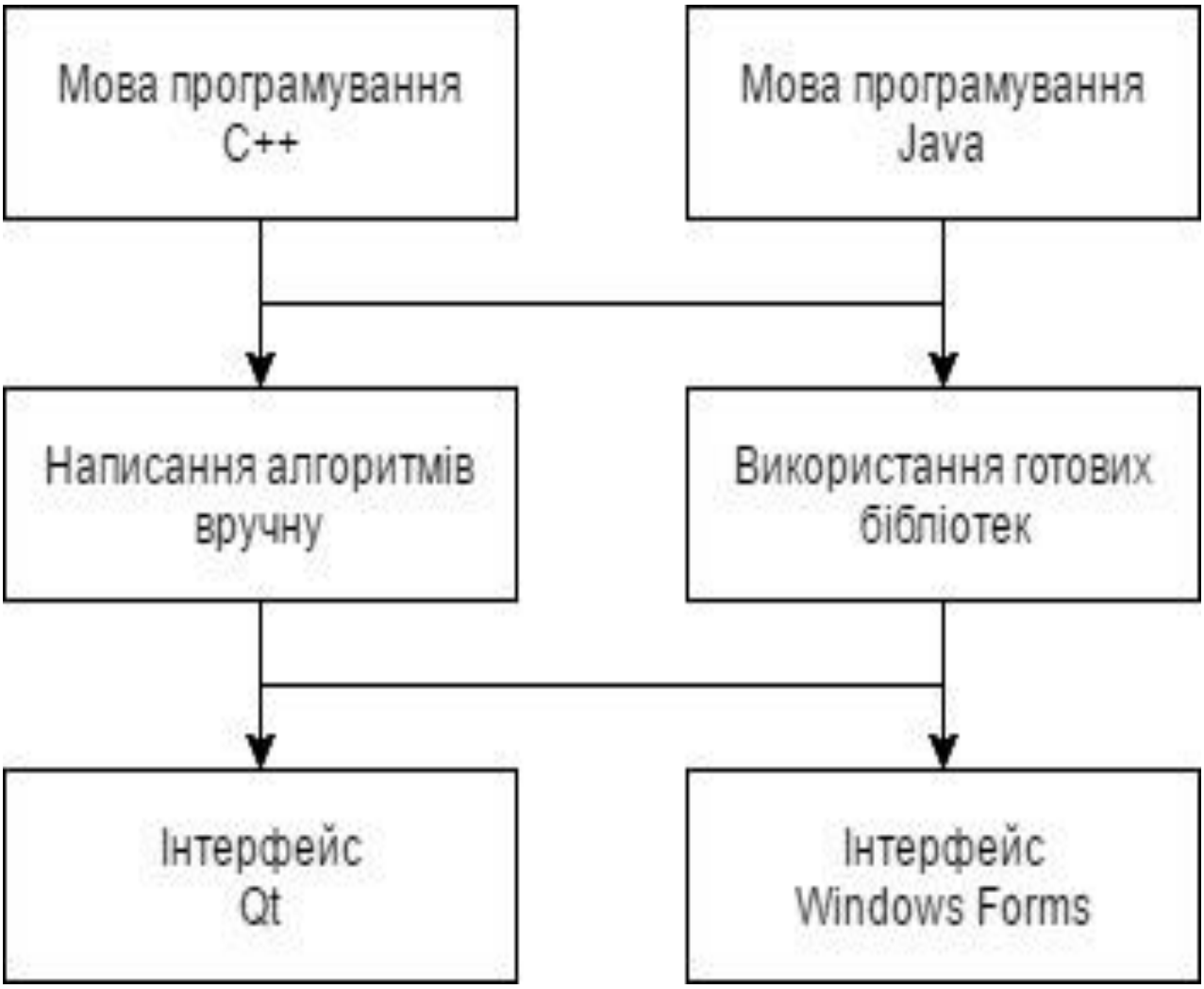


Рисунок 55 – Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 3 – Позитивно-негативна матриця

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	А	Кросплатформеність, швидкість, масштабованість	Складність для вивчення і компіляції
	Б	Об’єктність, безпека, автоматичне керування пам’яттю	Нижча швидкодія, неспроможність низькорівневого програмування
F2	А	Контроль вихідних даних	Затрачений час на написання
	Б	Перевірені методи обрахунків	Інтеграція
F3	А	Безкоштовна ліцензія, простота у освоєнні, кросплатформеність	Громідзкість вихідної програми
	Б	Простота у використанні	Відсутня кросплатформеність

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто

відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам.

Функція *F1:*

Оскільки програма має буди швидкою і підтримувати великі об’єми даних, то варіант «Б» має бути відкинутий.

Функція *F2:*

Оскільки контроль важливий при написанні програмного забезпечення, тому зупиняємось на варіанті «А».

Функція *F3:*

Обираємо варіант А, тому що в планах на майбутнє закладена ідея кросплатформенності.

4.2 Обґрунтування системи параметрів ПП

4.2.1 Опис параметрів

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу,

що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

1.*X1* – швидкодія мови програмування;

2.*X2* – об’єм пам’яті для збереження даних;

3.*X3* – час обробки даних;

4.*X4* – потенційний об’єм програмного коду.

X1: Відображає швидкодію операцій залежно від обраної мови програмування.

X2: Відображає об’єм пам’яті в оперативній пам’яті персонального комп’ютера, необхідний для збереження та обробки даних під час виконання програми.

X3: Відображає час, який витрачається на дії.

X4: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

4.2.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП.

Будуються графічні характеристики параметрів – рис. 56 – рис. 58.

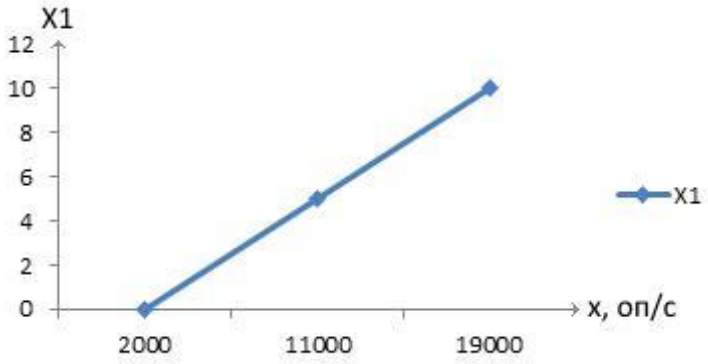


Рисунок 56 – X1, швидкодія мови програмування

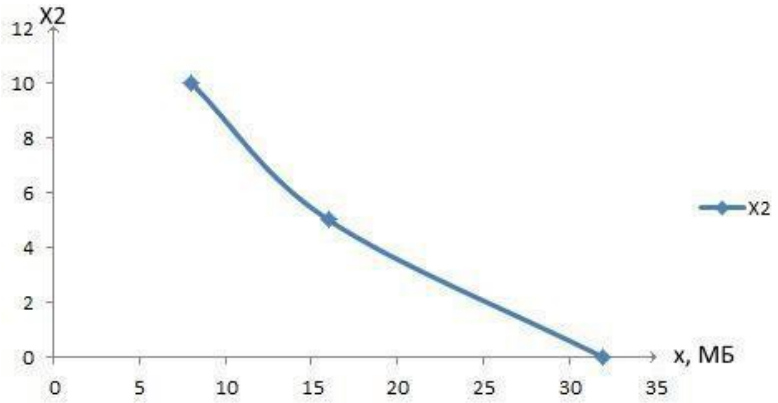


Рисунок 57 – X2, об’єм пам’яті для збереження даних

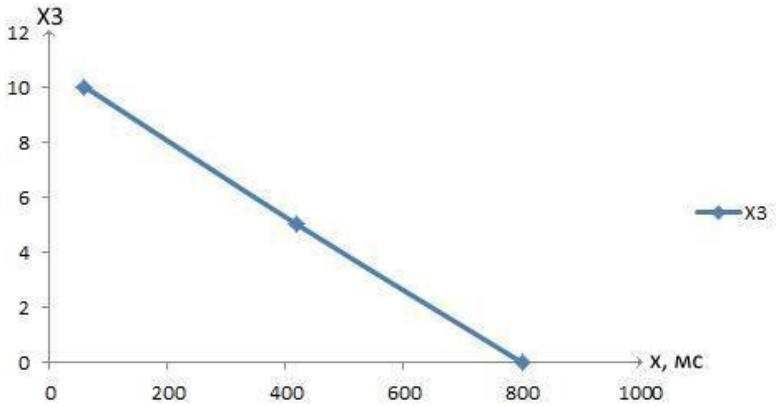


Рисунок 58 – X3, час обробки даних алгоритмом

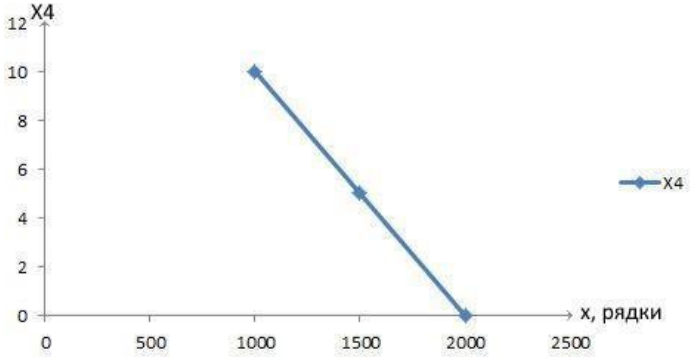


Рисунок 59 – X4, потенційний об’єм програмного коду

4.2.3 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходження параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

1. визначення рівня значимості параметра шляхом присвоєння різних рангів;
2. перевірку придатності експертних оцінок для подальшого використання;
3. визначення оцінки попарного пріоритету параметрів;
4. обробку результатів та визначення коефіцієнту значимості.

Таблиця 4 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	2	4	4	3	4	4	4	25	2,5	6,25
X2	Об’єм пам’яті для збереження даних	Мб	1	3	1	2	1	2	3	13	-9,5	90,25
X3	Час обробки даних алгоритмом	Мс	5	6	6	4	5	6	4	36	13,5	182,25
X4	Потенційний об’єм програмного коду	кількість строк коду	2	2	2	4	2	2	2	16	-6,5	42,25
	Разом		10	15	13	13	12	14	15	90	0	321

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 90,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 22{,}5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 321.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 321}{7^2(5^3 - 5)} = 1{,}31 > W_k = 0{,}67$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0.67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 6.

Таблиця 5 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	>	>	>	>	>	>	>	1.5
X1 і X3	<	<	<	<	<	<	<>	<	0.5
X1 і X4	<>	>	>	<	>	>	>	>	1.5
X2 і X3	<	<	<	<	<	<	<	<	0.5
X2 і X4	<	>	<	<	<	<>	>	<	0.5
X3 і X4	>	>	>	<>	>	>	>	>	1.5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1{,}5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю $A = \| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}$, де $b_i = \sum_{i=1}^{N} a_{ij}$.

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$K_{\text{вi}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}$, де $b_i' = \sum_{i=1}^{N} a_{ij} b_j$.

Як видно з таблиці 7, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 6 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$
X1	1,0	1,5	0,5	1,5	4,5	0,18	16,25	0,275
X2	0,5	1,0	0,5	0,5	2,5	0,1	9,25	0,157
X3	1,5	1,5	1,0	1,5	5,5	0,22	21,25	0,36
X4	0,5	1,5	0,5	1,0	3,5	0,14	12,25	0,208
Всього:					16	0,64	59	1

4.3 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(об’єм пам’яті для збереження даних) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 800 мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 6.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} ,$$

де n – кількість параметрів; $_в$ – коефіцієнт вагомості i–го параметра;B_i – оцінка i–го параметра в балах.

Таблиця 7 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	11000	4.5	0,215	0,774
F2(X2)	А	16	2.5	0,283	0,962
F3(X3,X4)	А	800	5.5	0,348	0,835
	Б	80	3.5	0,154	0,154

За даними з таблиці 7 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 0,81 + 0,25 + 1.21 = 2,27

K_{K2} = 0,81 + 0,25 + 0,49 = 1,55

Як видно з розрахунків, кращим є другий варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.4 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

При цьому варіант 3 має додаткове завдання:

1. Реалізація методів аналізу;

А варіант 4 має інше додаткове завдання:

2. Обробка інтерфейсу готових бібліотек.

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 2.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для

кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_М \cdot K_{СТ} \cdot K_{СТ.М}, \quad (4.1)$$

де $T_Р$ – трудомісткість розробки ПП; $К_П$ – поправочний коефіцієнт; $К_{СК}$–

коефіцієнт на складність вхідної інформації; $К_М$ – коефіцієнт рівня мови програмування; $К_{СТ}$ – коефіцієнт використання стандартних модулів і

прикладних програм; К – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $T_Р = 90$ людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_П = 1.7$.

Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх завдань рівний 1: $К_{СК} = 1$. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ} = 0.8$. Тоді, за формулою 6.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм другої групи складності, степінь новизни Б), тобто $T_Р = 27$ людино-днів, $К_П = 0.9$, $К_{СК} = 1$, $К_{СТ}$

$= 0.8$:

$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44$ людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

T_I = (122.4 + 19.44 + 4.8 + 19.44) · 8 = 1328,64 людино-годин;

T_{II} = (122.4 + 19.44 + 6.91 + 19.44) · 8 = 1345.52 людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 20 000 грн., один фінансовий аналітик з окладом 15 500 грн. Визначимо зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} \text{ грн.},$$

де М – місячний оклад працівників; T_m – кількість робочих днів тиждень; t – кількість робочих годин в день.

$$С_Ч = \frac{20000 + 20000 + 15500}{3 \cdot 21 \cdot 8} = 110 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_Ч \cdot Т_i \cdot К_Д,$$

де $С_Ч$– величина погодинної оплати праці програміста; Т – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 110 * 1 328,64 * 1.2 = 175 380,48 грн.

II. $С_{ЗП}$ = 110 * 1 345,52 * 1.2 = 177 608,64 грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП}$ · 0,22 = 175 380,48 * 0,22 = 38 583,7 грн.

II. $С_{ВІД} = С_{ЗП}$ · 0,22 = 177 608,64 * 0,22 = 39 073,9 грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 6000 грн., з

коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 * 20\ 0000 * 0,2 = 48\ 000$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 48\ 000 * (1 + 0.2) = 57\ 600$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0,22 = 57\ 600 \cdot 0,22 = 12\ 672$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 50 000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 50\ 000 = 14\ 375$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 50\ 000 \cdot 0.05 = 2\ 875$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) * t_З * К_В = (365 – 104 – 8 – 16) * 8 * 0.9 = 1706.4$

годин, де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} * N_С * К_З * Ц_{ЕН} = 1706,4 * 0,75 * 0,9733 * 1,93819 = 2\ 414,26$ грн.,

де N_C – середньо-споживча потужність приладу; $K_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} * 0.67 = 50\ 000 ; 0,67 = 33\ 500$ грн.

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$С_{ЕКС} = 57\ 600 + 12\ 672 + 14\ 375 + 2\ 875 + 2\ 414,26 + 33\ 500 = 123\ 436,26$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ} = 123\ 436,26 / 1\ 706,4 = 72,33$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в

залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot Т$$

I. $С_М = 72,33 * 1\ 328,64 = 96\ 100,53$ грн.;

II. $С_М = 72,33 * 1\ 345.52 = 97\ 321,46$ грн.;

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} * 0,67$$

I. $С_Н = 175\ 380,48 * 0,67 = 117\ 504,92$ грн.;

II. $С_Н = 177\ 608,64 * 0,67 = 118\ 997,78$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$$

I. $С_{ПП} = 175\ 380,48 + 38\ 583,7 + 96\ 100,53 + 117\ 504,92 = 427\ 569,63$ грн.;

II. $С_{ПП}$ = 177 608,64 + 39 073,9 + 97 321,46 + 118 997,78 = 433 001,78 грн.;

4.5 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

КТЕРj =ККj/СФj,

$К_{ТЕР1}$ = 2,27 / 427 569,63 = $5.31 \cdot 10^{-6}$;

$К_{ТЕР2}$ = 1,55 / 433 001,78 = $3.58 \cdot 10^{-6}$;

Як бачимо, найбільш ефективним є другий варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1}$= $5.31 \cdot 10^{-6}$.

4.6 Висновки до розділу

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

ВИСНОВКИ

В ході данної дипломної роботи було досліджено найбільш поширені алгоритми та програмні засоби для порівняння документів та системи антиплагиат. Також була сформована актуальність роботи та визначенні задачі.

У першому розділі були проанізовані найпоширеніші алгоритми, які на сьогодніший день використовуются в програмних засобах для порівняння документів. В даному розділі буду досліджено алгоритми, які використовуються в сучасних програмних засобах для порівняння документів.

Diff алгоритм реализован, наприклад, в MS Office, файли з текстами програм (засоби контролю версій CVS та Subversion, утиліта UNIX diff) і т. д. Алгоритм diff використовується в якості складової частини алгоритмів злиття (merge) двох різних версій файлів в системах версіонного контролю, при синхронізації даних в системах з обмеженнями на трафік, при об'єднанні серверних і клієнтських даних (наприклад, в разі обриву з'єднання) і т. д. Однак при всій своїй популярності цей алгоритм має ліміт на застосування, головний з яких - зручна візуалізація результатів. Наприклад, якщо спробувати скористатися функцією diff для порівняння Word-документів, вбудованої в MSOffice, то якщо порівнювані документи чималі (10 сторінок і більше), а зроблених змін досить багато, то прийняти таку різницю виявляється дуже складно - користувач бачить величезну кількість виділеної інформації, серед якої і вилучені абзаци, і відмінності в правилах пунктуації.

Проблема алгоритму шинглів полягає в кількості порівнянь, адже це безпосередньо позначається на продуктивності. Збільшення кількості шинглів для порівняння характеризується ростом операцій, хто критично позначиться на продуктивності.

В другому розділі мною були розглянуті функціональні можливості програмних засобів для перевірки документів.

Я вияснив, що функціонал платних та безплатних програм майже не відрізняется. Хоча головною їх перевагою є те що вони вони можуть порівнювати паперові і електронні документи.

Я проаналізував 5 основних популярних сервісів по перевірці унікальності текстів. Але мені очевидні на сьогоднішній день два лідери: Text.ru і Content-watch.ru.

Причому, Text.ru найкраще підійде замовникам, так як дозволяє провести максимально глибоку перевірку (нехай іноді і з помилковими спрацьовуваннями.

А ось виконавцю краще все-таки підійде Content-watch.ru або Miratools.ru (якщо ненагружен), так як в них набагато менше помилкових спрацьовувань і не доведеться нескінченно переписувати тексти. Аналізувати унікальність технічного рерайта в Text .ru - майже неможливо, так як сервіс реагує на найменші збіги з приводу і без.

Також була затронута тема плагату і, як програми для визначення плагіату допомагаю з ними боротися.

ПЕРЕЛІК ПОСИЛАНЬ

1. Д.В.Луцив, Д.В.Кознов, В.С.Андреев Системное программирование. Том. 7. / Д.В.Луцив, Д.В.Кознов, В.С.Андреев – Москва: Генеза, 2012. - 57с.
2. К.В.Чувилин. КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ И МОДЕЛИРОВАНИЕ / К.В.Чувилин – Москва: Вариант, 2015. - С. 329–345
3. Офіційний сайт abbyy – Режим доступу: https://www.abbyy.com/media/6837/abbyy_comparator.pdf - Дата доступу: 11.05.17
4. Офіційний сайт comparesuite – Режим доступу: http://www.comparesuite.ru/articles/programma-dlya-sravneniya-tekstovyx-fajlov.htm - Дата доступу: 20.05.17
5. Офіційний сайт winmerge – Режим доступу: http://winmerge.org/about/screenshots/ - Дата доступу: 20.05.17
6. Офіційний сайт adobe – Режим доступу: https://helpx.adobe.com/ru/acrobat/using/compare-documents.html/ - Дата доступу: 21.05.17
7. Diff - Wikipedia - Режим доступу: https://ru.wikipedia.org/wiki/Diff/ - Дата доступу: 1.06.17
8. Подходы к сравнению версий файлов - Режим доступу: https://habrahabr.ru/post/65944/ - Дата доступу: 5.06.17