МІЖНАРОДНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, програмної інженерії та комп'ютерних наук
Кафедра комп'ютерних наук

Пояснювальна записка

до кваліфікаційної роботи
першого (бакалаврського) рівня

на тему **Стиснення інформації в сучасних комп'ютерних мережах**

Виконав: студент 4 курсу, групи КІ-4
спеціальності
123 Комп'ютерна інженерія

Чу Нам Ань

Керівник Розенвассер Д.М.

Рецензент Григор'єва Т.І.

Одеса – 2023 р

ДОВІДКА

кафедри КН про виконану бакалаврську роботу

студента 4 курсу ФКПІтаКН групи КІ-4

Чу Нам Ань

на тему Стиснення інформації в сучасних комп'ютерних мережах

Висновок нормоконтролера

Пояснювальна записка до кваліфікаційної роботи виконана з незначними порушеннями ДСТУ та вимог внутрішнього положення КПІ, які не впливають

Нормоконтролер викл. каф. ТІІ (науковий ступінь, вчене звання, посада) 29.06.23 (підпис, дата) І.В. Клімішина (і. б. прізвище)

Висновок відповідального за наявність плагіату

Згідно з сертифікатом ID 1015327943 унікальність підтверджено

Відповідальна особа викл. каф. ТІ (науковий ступінь, вчене звання, посада) 29.06.23 (підпис, дата) І.В. Клімішина (і. б. прізвище)

Попередня експертиза (захист) бакалаврської роботи (бакалаврської роботи чи магістерської роботи)

студ. Чу Нам Ань (прізвище і б.) проведена "_____" _____ 20__ р.

Висновки

Студент Чу Нам Ань проаналізував декілька найпоширеніших програм-архіваторів, провів відповідні експерименти, зробив висновки та надав рекомендації

Виконана робота бакалавра відповідає вимогам стандартам та рекомендується до захисту до ЕК

Члени комісії (підпис) к.т.н., доц. Соловська І.М. (науковий ступінь, вчене звання, посада, прізвище і.б.)

(підпис) к.т.н., доц. Григор'єва Т.У. (науковий ступінь, вчене звання, посада, прізвище і.б.)

(підпис) к.т.н., доц. Чеха А.С. (науковий ступінь, вчене звання, посада, прізвище і.б.)

МІЖНАРОДНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, програмної інженерії та комп'ютерних наук
Кафедра комп'ютерних наук
Освітній ступінь бакалавр
Галузь знань 12 Інформаційні технології
Спеціальність 123 Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
к.т.н., доц.
_____ І.М.Соловська
"7" 04 2023 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ

Чу Нам Ань

1. Тема роботи: стиснення інформації в сучасних комп'ютерних мережах

керівник роботи доц. каф. КН Розенвассер Д.М.
затверджені наказом закладу вищої освіти від від 7 квітня 2023 р. № 587
2. Строк подання студентом роботи 10.06.2023 р.
3. Вихідні дані до роботи: перелік файлів різних типів та розмірів, які необхідно стиснути за допомогою сучасних архіваторів

4. Зміст розрахунково-пояснювальної записки
Розділ 1: типи даних та архіватори
Розділ 2: стиснення інформації та його види
Розділ 3: порівняння архіваторів

5. Перелік графічного матеріалу (з зазначенням обов'язкових креслень)
Слайд 1 – Схема стиснення даних
Слайд 2 – Приклади архіваторів
Слайд 3 – Порівняння архіваторів за ступенем стиснення
Слайд 4 – Порівняння архіваторів за швидкістю стиснення
Слайд 5 – Приклади стиснення

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	завдання видав	завдання прийняв

7. Дата видачі завдання 25.11.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Типи даних	25.11.2022 – 12.12.2022	вик
2	Види файлів	13.12.2022 – 31.12.2022	вик
3	Архіватори	01.01.2023 – 17.01.2023	вик
4	Стиснення даних в комп'ютерних мережах	18.01.2023 – 11.02.2023	вик
5	Стиснення без втрат	12.02.2023 – 28.02.2023	вик
6	Стиснення з втратами	01.03.2023 – 19.03.2023	вик
7	Порівняння за ступенем стиснення	20.03.2023 – 14.04.2023	вик
8	Порівняння за швидкістю стиснення	15.04.2023 – 09.05.2023	вик
9	Порівняння за якістю відновлення даних	10.05.2023 – 31.05.2023	вик
10	Висновки та рекомендації	01.06.2023 – 10.06.2023	вик

Студент _____ Чу Нам Ань
(підпис)

Керівник роботи _____ Д.М. Розенвассер
(підпис)

UNICHECK
by Turnitin

Ім'я користувача:
Анна Серединко

ID перевірки:
1015683842

Дата перевірки:
23.06.2023 14:26:58 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
23.06.2023 14:30:42 EEST

ID користувача:
100001433

Назва документа: Чу Нам

Кількість сторінок: 85 Кількість слів: 14258 Кількість символів: 95051 Розмір файлу: 1.10 MB ID файлу: 1015327943

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

29.3%
Схожість

Найбільша схожість: 5.63% з Інтернет-джерелом (https://er.chdtu.edu.ua/bitstream/ChSTU/3515/1/%D0%9A%D0%BE%D...

28.6% Джерела з Інтернету | 982 ..... Сторінка 87

1.98% Джерела з Бібліотеки | 18 ..... Сторінка 92

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | 7

Підозріле форматування | 24 сторінки

РЕЦЕНЗІЯ

на бакалаврську роботу студента Чу Нам Ань
на тему: «Стиснення інформації в сучасних комп'ютерних мережах»

Бакалаврська робота виконана на 50 с. текстової частини та 7-и демонстраційних слайдах та містить відповідні розділи згідно з завданням на бакалаврську роботу.

У бакалаврській роботі студента Чу Нам Ань розглянуті питання стиснення інформації у комп'ютерних мережах. Актуальність теми полягає в тому, що зменшення обсягу інформації в комп'ютерних мережах призводить до зменшення собівартості її зберігання та передавання.

У роботі пропонуються обрання архіватора за різними критеріями в залежності від інформації, що стискається та створення універсального архіватора для комп'ютерних мереж.

Практичне значення отриманих автором роботи результатів полягає в можливості впровадження отриманих результатів на реальних об'єктах.

Студентом показано достатню теоретичну підготовку.

Пояснювальна записка й графічні матеріали виконані охайно й відповідно до вимог ЄСКД.

Недоліки роботи:

- дослідження проведено тільки для безкоштовних архіваторів, що знаходяться у вільному доступі;
- в роботі відсутній опис механізму впровадження запропонованого архіватора в комп'ютерних мережах.

Названі недоліки не знижують цінності виконаної роботи.

Бакалаврська робота студента Чу Нам Ань відповідає вимогам до випускних кваліфікаційних робіт бакалаврів і заслуговує оцінки «задовільно».

Студент Чу Нам Ань заслуговує присвоєння за заявленою спеціальністю 123 Комп'ютерна інженерія кваліфікації бакалавр з комп'ютерної інженерії.

Рецензент
зав. кафедри ІТ  Т. І. Григор'єва

ВІДГУК КЕРІВНИКА

бакалаврської роботи студента Чу Нам Ань
на тему: «Стиснення інформації в сучасних комп'ютерних мережах»

Завдання стиснення інформації є актуальною темою для нових та існуючих комп'ютерних мереж.

У роботі розглядаються методи стиснення інформації зі втратами та без втрат.

Студент Чу Нам Ань добре розібрався з проблемами даної тематики, приділив увагу докладному аналізу методів стиснення інформації в Україні та світі.

Робота проводилася значною мірою самостійно.

Під час виконання наукової роботи студент Чу Нам Ань вивчив принципи стиснення інформації, принципи роботи програм-архіваторів даних формату ZIP, RAR, ARJ, PA та інших для стиснення даних різного походження, показав уміння користуватись навчальною та технічною літературою, розв'язувати інженерні задачі.

Студент Чу Нам Ань проаналізував декілька найпоширеніших програм-архіваторів, провів відповідні експерименти, зробив висновки та надав рекомендації.

Бакалаврська робота відповідає вимогам до кваліфікаційних робіт бакалаврів. Робота студента Чу Нам Ань заслуговує оцінки «задовільно».

Студент Чу Нам Ань заслуговує присвоєння за заявленою спеціальністю 123 Комп'ютерна інженерія кваліфікації бакалавр з комп'ютерної інженерії.

Керівник
доцент кафедри КН Д.М. Розенвассер

РЕФЕРАТ

Текстова частина бакалаврської роботи: 45 с., 31 рис., 7 табл., 11 джерел.

АРХІВАТОРИ, СТУПІНЬ СТИСНЕННЯ, ШВИДКІСТЬ СТИСНЕННЯ, ТИПИ ДАНИХ, ВИДИ ФАЙЛІВ

Об'єкт дослідження – сучасні архіватори, що використовуються для стиснення даних.

Мета роботи – надати рекомендації з обрання архіватора для стиснення даних різних типів.

Метод дослідження – аналітичний з використанням комп'ютерних технологій.

У бакалаврській роботі проведено аналіз файлів різних типів, методів стиснення даних та відомих сучасних архіваторів. Поставлені задачі для виконання роботи. За результатами досліджень надано рекомендації для обрання кращого архіватора, що використовується для стиснення даних. Під час аналізу проводилося порівняння за різними параметрами, такими як ступінь стиснення, швидкість стиснення, якість відновлення та інше.

ABSTRACT

The text part of the bachelor's work: 45 pp., 31 figures, 7 tables, 11 sources.

ARCHIVATORS, COMPRESSION LEVEL, COMPRESSION SPEED, DATA TYPES, FILE TYPES

The object of research is modern archivers used for data compression.

The purpose of the work is to provide recommendations for choosing an archiver for data compression of various types.

The research method is analytical with the use of computer technologies.

The bachelor's work analyzed files of various types, data compression methods, and well-known modern archivers. Tasks are set for the performance of the work. Based on the results of research, recommendations are provided for choosing the best archiver used for data compression. During the analysis, a comparison was made on various parameters such as compression ratio, compression speed, recovery quality and others.

ЗМІСТ

ВСТУП..... 11

1 ТИПИ ДАНИХ ТА АРХІВАТОРИ..... 12

1.1 Типи даних..... 12

1.2 Види файлів..... 15

1.3 Архіватори..... 18

1.4 Висновки до розділу 1..... 22

2 СТИСНЕННЯ ІНФОРМАЦІЇ ТА ЙОГО ВИДИ..... 23

2.1 Стиснення даних в комп'ютерних мережах 23

2.2 Стиснення без втрат..... 25

2.3 Стиснення з втратами..... 35

2.4 Висновки до розділу 2..... 41

3 ПОРІВНЯННЯ АРХІВАТОРІВ..... 43

3.1 Порівняння за ступенем стиснення 43

3.2 Порівняння за швидкістю стиснення 50

3.3 Висновки до розділу 3..... 55

ВИСНОВКИ ТА РЕКОМЕНДАЦІЇ..... 56

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ..... 57

ВСТУП

В процесі роботи перед нами часто виникає проблема нестачі пам’яті на пристрої. Користувач або знищує менш важливу інформацію, або записує на зовнішні диски чи на інші носії. Але є ще один вихід з цієї ситуації - архівування інформації.

Архівування – це стиснення певної інформації для економії простору та захисту її від небажаного доступу. Оскільки в стиснутому вигляді інформацією скористатись неможливо, скористатись цією інформацією з файла можна лише розархівувавши її. Архіватори - це програми для створення копій файлів, що мають менший розмір за допомогою певних методів стиснення інформації, а також для об'єднання копій декількох файлів в архівний файл. У наш час терабайтних обсягів жорсткого диска архівація файлів необхідна не для економії місця. У багатьох з нас особисті папки містять гігабайти інформації, яку ми після скачування навіть жодного разу не відкривали. Останнім часом ми розучилися дбайливо ставитися до дискового простору і акуратно складати папки в архівні файли.

На сьогоднішній день найбільш популярними програмами-архіваторами є WinRAR, PowerArchiver, а також безкоштовний 7-Zip. Вони дають можливість використовувати і інші архіватори для розпакування архівних файлів, які були стиснуті ними. У ці програми включені такі зручні функції як можливість створювати саморозпакування і багатотомні архіви. Останні підходять для відправки частинами по електронній пошті, що дуже зручно для використання в державних установах, враховуючи, що в більшості організацій поштові сервери мають обмеження на обсяг переданих даних.

.

1. ТИПИ ДАНИХ ТА АРХІВАТОРИ

1.1 Типи даних

Стиснення інформації - це процес перетворення інформації, що зберігається в файлі, до виду, при якому зменшується надмірність в її уявленні і відповідно потрібно менший обсяг пам'яті для зберігання. Всі процеси стиснення працюють з вхідним потоком інформації, мінімальною одиницею якого є біт. Метою стиснення зазвичай є отримання більш компактного вихідного потоку інформаційних блоків з деякого спочатку некомпактного вхідного потоку за допомогою деякого перетворення.

Цифровий аудіоформат - формат подання звукових даних, використаний при цифровому звукозапису, а також для подальшого зберігання записаного матеріалу на комп'ютері або електронних носіях інформації, так званих звукових носіях. При цьому, аудіофайл (файл, який містить звукозапис) - комп'ютерний файл, що складається з інформації про амплітуду і частоту звуку, збереженої для подальшого відтворення на комп'ютері або програвачі. Оцифрований звук з високою якістю вимагає величезних витрат дискової пам'яті. Спроби скоротити обсяг файлів, використовуючи стандартні архіватори не призводять до значного виграшу через специфічність звукових даних. Проте добитися досить значного рівня стиснення аудіоінформації вдається при використанні спеціальних методів, основаних на аналізі структури даних і наступним стисканням з деякими втратами. У 1988 році Міжнародною організацією стандартів ISO (International Standards Organization) був сформований комітет MPEG (Moving Pictures Expert Group, група експертів в області рухомих зображень), основним завданням якого є розробка стандартів кодування рухомих зображень, звуку і їх комбінації. За десять років свого існування комітет виробив ряд стандартів з даного питання. В результаті узагальнивши численні дослідження в цій області, був рекомендований ряд специфічних форматів для зберігання даних, відмінних за якістю результатів і швидкості потоку даних. В даний час існує три стандарти зберігання відеоданих: MPEG-1, MPEG-2 і MPEG-4. Спираючись на перші два формати, існують також формати зберігання звукової інформації - Layer-1, Layer-2 і Layer-3. Ці три звукові формати визначені для MPEG-1 і незначними розширеннями використовуються в MPEG-2. Всі три формати схожі один на

одного, але використовують різні рівні компромісу між стисненням і складністю.

Рівень Layer-1 - найбільш простий, не вимагає значних витрат на стиск, але і дає незначний ступінь стиснення.

Рівень Layer-3 - найбільш трудомісткий і забезпечує найкраще стиснення. В останній час цей формат завоював величезну популярність. Його часто називають MP3. Така назва пов'язана з розширенням звукових файлів, що зберігаються в цьому форматі. Основна ідея, на якій базуються всі методики стиснення аудіо сигналу з втратами, - зневага тонкими деталями звучання оригіналу, що лежать поза межами які сприймає людське вухо. Тут можна виділити кілька моментів.

Рівень шуму. Звуковий стиск базується на простому факті - якщо людина знаходиться поруч з сиреною, то навряд чи вона почує розмову людей, що знаходяться поряд. Причому це відбувається не тому, що людина звертає увагу на гучний звук, а тому, що людське вухо фактично втрачає звуки, що лежать в тому ж діапазоні частот, що і більш гучний звук. Цей ефект називається маскучий, він змінюється з різницею в гучності і частоті звуку.

Не менш важливим моментом кодування є використання психоакустичної моделі, що спирається на особливості людського сприйняття звуку. Стиснення з використанням цієї моделі засновано на видаленні свідомо нечутні частоти з більш ретельним збереженням звуків, що добре розрізняються людським вухом. На жаль тут не може бути точних математичних формул.

Сприйняття звуку людиною – складний, до кінця не вивчений процес, тому вибір методів стиснення виконується на основі аналізу прослуховування і порівняння по-різному стислих звуків групами експертів. Зате тут є практично необмежені можливості в сфері поліпшення психоакустичних моделей. Більшість існуючих алгоритмів для кодування людського голосу засноване на високій передбачуваності такого сигналу - універсальні алгоритми стиснення MPEG зі змінним успіхом намагаються застосувати цей прийом.

Формат мультимедіа набагато складніше ніж більшість інших форматів файлів через широке різноманіття даних, які вони повинні зберігати. Такі дані включають текст, зображення, аудіо та відеодані, машинну мультиплікацію та інші форми двояких даних типу цифрового інтерфейсу музичних інструментів та графічні шрифти. Типові формати мультимедіа не визначають нові методи для того, щоб зберегти ці типи даних. Замість цього вони пропонують можливість збереження даних в одному або більше існуючих форматах даних.

Наприклад, формат мультимедіа може містити текст, захищений як PostScript або RTF, але зазвичай не в форматі відкритого тексту ASCII [2]. Растрові дані можуть бути збережені як BMP або файли TIFF, а не як необроблені точечні малюнки. Так само аудіо, відео та дані анімації можуть бути збережені, використовуючи існуючі формати, підтримуючі формат файла мультимедіа. Для стиснення цифрових мультимедійних файлів використовуються спеціальні програми - кодеки. Це так звана «формула», яка визначає, яким чином можна упакувати відео та аудіоконтент. Кодеки виконують та формують операцію розкодування, у цьому випадку їх називають декодерами.

- Кодер (англ. Coder, encoder) - програма та/або пристрій, що використовуються для перетворення інформації з одного виду в іншій (кодування).
- Декодер (англ. Decoder) – теж саме, що і кодер, що здійснює перетворення в зворотньому напрямку.
- Кодек (англ. Codec) - кодер та декодер в одному блоці.
- Ступень стиснення - відношення розміру вхідного (не кодованого) файлу до розміру вихідного (кодованого) файлу. Наприклад, ступінь стиснення 11:1 означає, що закодированый файл у 12 разів менше оригіналу.
- Бітрейт (англ. Bitrate) - кількість бітів, відведене для запису одиниць часу аудіо-інформації. Змінюють зазвичай в кб/с, тобто кілобіт в секунду.

Більшість кодеків для звукових і візуальних даних використовують стиснення з втратами, щоб одержувати прийнятний розмір готового (стисненого) файлу. Існують також кодеки, що стискають без втрат (англ. Lossless codecs). Але для більшості застосувань вигідніше кодеки з втратами інформації, так як малопомітне погіршення якості виправдовується значним зменшенням обсягу даних. Єдиний виняток - ситуація, коли дані будуть піддаватися подальшій обробці, в цьому випадку повторювані втрати на кодуванні / декодуванні нададуть серйозний вплив на якість.

Найбільш популярними є такі кодеки:

- psd, bmp, rle, dib, gif, eps, jpg, pcx, raw, png, tif і ін. - зображення.
- flag, ogg, opus, wav, pcm, wma, mp3, aac, as3, dts, flac і ін. - аудіо;
- ffdshow, indeo, mjpeg, mpeg-1, mpeg-2, mpeg-4 (h.261, h.263, h.264), wmv - відео.

Текст (лат. Textum - зв'язок, з'єднання) - це об'єднана смислової зв'язком послідовність знакових одиниць, основними властивостями якої є зв'язність і цілісність. Текст є однією з найбільш поширених форм представлення інформації.

Це будь-яка записана мова, будь-який опис чого-небудь, будь-яка письмова повідомлення. Текст - це певна послідовність символів або будь-яке словесне висловлювання.

1.2 Види файлів

В наш чаш існують програми, основними складовими яких є файли. Існує велика кількість файлів. Кожен має своє призначення та функції. На сьогоднішній день найбільш відомі типи файлів: doc, txt, pdf, xls, jpeg, png, mp3, wav, exe, Всі з названих типів файлів мають свої функції – одні запускають програми, інші зберігають інформацію. Деякі з них без використання методів стиснення. Також є файли у стиснутому виді, на такі архівація діє набагато менше, так як надлишковість у стиснутих файлах вже зменшена.

Файл DOC (з англ. Microsoft Word Document) є документом, створеним для роботи з текстовими документами Microsoft Word. Цей формат може зберігати в собі велику кількість різноманітної інформації – формат тексту, данні вирівнювання, відступи, списки, абзаци та інше. Подібні файли можуть містити не лише текстовий документ з певною інформацією, але й зображення, таблиці та діаграми.

Формат TXT унікальний, тому що ці файли може відкрити будь-який текстовий редактор**.** Файли TXT мають роль блоків по зберіганню інформації, використовуваних з метою недопущення конфліктів з іншими форматами файлів. Файли, у яких дані пошкоджені, можна легко відновити, тому що користувач може далі працювати з даними які залишилися. Файли TXT містять мало елементів форматування, але дозволяють співвіднести прийняті набори елементів форматування з системним терміналом або простим текстовим редактором. Недоліком використання файлів TXT є низька ентропія внаслідок того, що файли TXT займають більше місця (в порівнянні з іншими текстовими файлами).

PDF (з англ. Portable Document Format) - популярний формат електронних документів, створений фірмою Adobe Systems з використанням багатьох можливостей мови PostScript. В першу чергу призначений для подання різних

документів в електронному вигляді, таких як книги, журнали, рекламні буклети, інструкції. Кожен документ PDF може містити текст, шрифти, зображення та мультимедіа-вставки. Для створення, редагування використовується програма Adobe Acrobat, а для перегляду підходить Foxit Reader, STDU Viewer, в тому числі офіційна безкоштовна програма Adobe Reader [8].

Розширення XLS - найпоширеніший формат даних для представлення інформації у вигляді електронних таблиць. Побудова діаграм, графіків і зображень, створення математичних формул - це лише незначний перелік операцій, які можуть бути згенеровані в XLS файл. Великоформатна таблиця, створювана в Microsoft Excel (програма для відкриття файлів XLS) - найбільш широко поширених програм великоформатних таблиць. Зберігає дані в таблиці з стовпчиків рядів і колонок. Стовпчики таблиці можуть містити дані, введені вручну, або підраховані комп'ютером результати з даних інших осередків. Часто застосовується для створення схем і графіків.

JPEG (Joint Photographic Experts Group - Об'єднана експертна група по фотографії, вимовляється "джейпег") – популярний серед графічних форматів, який застосовують для зберігання фото і подібних до них. Цей формат розроблений компанією C-Cube Microsystems як метод зберігання зображень з великою глибиною кольору, наприклад, одержуваних при скануванні фотографій з численними, ледь вловимими (а іноді і невловимими) відтінками кольору. Найбільша відмінність формату JPEG від інших полягає в тому, що в JPEG використовується алгоритм стиснення з втратами (а не алгоритм без втрат) інформації. Алгоритм стиснення без втрат так зберігає інформацію про зображення, що розпаковане зображення в точності відповідає оригіналу. При стисненні з втратами приноситься в жертву частина інформації про зображення, щоб досягти більшого коефіцієнта стиснення. Розпаковане зображення JPEG рідко відповідає оригіналу абсолютно точно, але дуже часто ці відмінності настільки незначні, що їх ледве можна (якщо взагалі можна) виявитих [9].

PNG (з англ. Portable Network Graphics) - один із популярних форматів у веб-графіці. Був створений в 1995 році на конференції Usenet як альтернатива формату GIF. З 1 жовтня 1996 року вже існував як повноправний графічний формат, що підтримується всіма браузерами та графічними редакторами. Зазвичай файли формату PNG мають розширення .png та MIME тип – image/png [9]. Цей формат використовує стиснення без втрат, на відміну від формату JPEG. На практиці .png підходить для збереження зображень із невеликою кількістю

кольорів. Логотипи, іконки, значки, схеми, скан-копії документів зазвичай розміщують на сайтах у форматі .png.

MP3 вважається форматом, який розроблений командою MPEG (з англ. Motion Pictures Expert Group — група експертів в області кінематографу) для зберігання аудіоінформації. Він є одним з найбільш поширених і популярних форматів цифрового кодування звукової інформації, який широко використовується в файлообмінних мережах для скачування музики. Має ступінь стиснення за алгоритмом кодування Layer 3, що забезпечує звучання високої якості, подібне CD Audio і досить компактний розмір файлу.

У форматі MP3 використовується алгоритм стиснення з втратами, розроблений для зменшення розміру даних, необхідних для відтворення запису, який точно відповідає оригінальному, але з відчутними втратами якості при прослуховуванні на звуковій системі. Принцип стиснення цього формату полягає в зниженні точності декількох частин звукового потоку, що практично непомітно для слуху на апаратурі низької точності відтворення звуку (наприклад, більшість портативних пристроїв, звукових карт, музичних центрів та інших), а також для людей похилого віку, у зв'язку з природними віковими змінами слухового апарату, але в більшості випадків чітко видно на апаратурі високої точності відтворення звуку. Цей метод називають психовізуальним кодуванням. Далі етап, на якому будується діаграма звуку у вигляді послідовності коротких відрізків часу, потім на ній видаляється інформація, що не помітна людському слуху, а решта інформації зберігається в компактному вигляді. MP3-файли можуть створюватися з високим або низьким бітрейтом, що впливає на якість файлу в результаті.

Формат аудіофайлу WAV або як його ще називають Waveform (англ. Waveform - "у формі хвиль") - один із перших аудіо-форматів. Зазвичай використовується для зберігання нестиснених аудіо-записів, ідентичних за якістю звуку на компакт-дисках (audio-CD). В середньому одна хвилина звуку у форматі WAV займає близько 10 мегабайт. Також його особливість у тому, що для кодування амплітуди виділяється фіксованим числом біт. Це відображається на розмірі вихідного файлу, але робить його дуже зручним для читання. Типовий хвильовий файл складається з заголовкової частини, тіла з аудіопотоком і хвоста для додаткової інформації, куди аудіоредактори можуть записувати власні метаданні [8].

Файли EXE (executable files - виконувані файли) - загальна назва комп'ютерних програм ОС DOS, OpenVMS, Windows і ін. Ресурси, растрові

зображення, іконки та ін. Елементи, необхідні для роботи програми, містяться саме в таких файлах. Після стиснення часто можна скористатися стислою версією файлу таким же чином, як і незжатою версією (стиснення проводиться виконуваючими архіваторами - робочими пакетами, програмними пакетами і т.п.). Більшість користувачів комп'ютерів знають, що для запуску певної програми в разі відсутності ярлика на інсталяційний файл необхідно відшукати файл EXE.

1.3 Архіватори

Архіватор — програмне забезпечення, що має змогу за рахунок застосування спеціальних методів для зменшення інформації створювати копії файлів меншого розміру, а також об'єднувати копії декількох файлів в один архівований файл, з якого можна при необхідності витягти файли в їх первинному вигляді. До основних можливостей сучасних архіваторів належать: занесення цілих груп файлів і підкаталогів в архів, оновлення і перевірка цілісних архівів, вилучення файлів з архіву, захист інформації від небажаного доступу.

Архіватори дозволяють економити від 10% до 90% дискового обсягу. Файлом, який поміщений в архів, можна скористатися, тільки після його вилучення, тобто після розархівації. Розархівувати файл можна або архіватором, або спеціальною програмою - розархіватором. Залежно від архіватора і формату, звичайно, архіву залежить швидкість стиснення і його якість. Також, є можливість і просто створення контейнера без використання стиснення взагалі. Це буває зручно, коли необхідно просто об'єднати всі елементи в один контейнер і швидко отримувати до них доступ без довгого вилучення. Ступінь стиснення залежить від того, який тип файлів архівуватиметься. Найкраще це працює з документами, складні ж елементи - бінарні, стискаються трохи гірше.

На сьогоднішній день, найбільше ми використовуємо такі архіватори, як WinRAR, 7-Zip, PowerArchiver, BandiZip. Кожен з них може послідовно об'єднувати файли в один контейнер, при цьому зберігаючи головні дані про них, щоб в подальшому можна було розпакувати. Це: час створення, зміни, точний розмір, ім'я та інші метадані, які зберігає операційна система.

WinRAR -– це файловий архіватор з високим ступенем стиснення, є одним із найкращих архіваторів за співвідношенням ступеня стиснення до швидкості роботи. Він являється 32-розрядною версією архіватора RAR для

Windows, потужного засобу створення архівів і управління ними. Існує кілька версій RAR для різних операційних систем, зокрема RAR для Windows, Linux, DOS, OS / 2, UNIX [7].

Рисунок 1.1 – Приклади архіваторів

Особливості WinRAR:

- ✓ використання оригінального високоефективного алгоритму стиснення даних;
- ✓ наявність графічної оболонки з підтримкою технології перетягування (dragdrop);
- ✓ можливість використання інтерфейсу командного рядка;
- ✓ управління архівами інших форматів (CAB, ARJ, LZH, TAR, GZ, ACE, UUE, BZ2, JAR, ISO);
- ✓ підтримка методу безперервного (solid) архівування, при якому може бути досягнута більш висока ступінь стиснення (від 10 до 50%), ніж дають звичайні методи, особливо якщо упаковується значна кількість невеликих файлів однотипного змісту;
- ✓ підтримка багатотомних архівів;
- ✓ утворення саморозпаковування (SFX) звичайних і багатотомних архівів за допомогою стандартного або додаткових модулів SFX;
- ✓ можливість відновлення фізично пошкоджених архівів;
- ✓ можливість створення і використання томів для відновлення;
- ✓ підтримка кодування Unicode в іменах файлів;

- ✓ наявність інших додаткових функцій, наприклад шифрування даних і імен файлів в архіві, додавання архівних коментарів, ведення протоколу помилок та ін.

7-Zip – єдиний безкоштовний архіватор, який є у відкритому доступі і має можливість поставити пароль. Архіватор підтримує розпакування архівів більшості популярних форматів, включаючи ZIP, ARJ, TAR і RAR. Цікаво, що розмір одержуваних ZIP-файлів виходить на 2-10% менше, ніж при використанні WinRAR. На сумісність це ніяк не впливає - ZIP-архіви, створені в 7-Zip, без проблем відкриваються як в інших архіваторах, так і вбудованими засобами Windows. 7-Zip використовує багатопоточність і дозволяє задіяти для стиснення, в залежності від алгоритму або формату, різну кількість потоків [3]. При створенні архівів, в яких файли стискаються незалежно один від одного (наприклад ZIP), програма може використовувати до восьми потоків одночасно. Для алгоритму стиснення LZMA (Lempel-Ziv-Markov chain-Algorithm) архіватор одночасно може використовувати до двох потоків. Неможливість використання більшої їх кількості пояснюється послідовним характером безперервного стиснення.

Bzip2 –алгоритм з відкритим початковим кодом для стиснення даних. Відповідно до традицій UNIX-програмування, Bzip2 виконує тільки одну функцію: стиснення і розпаковування одного файлу. При цьому до назви файлу за умовчанням додається розширення .bz2. Для упаковки декількох файлів їх зазвичай спершу архівують в один файл утилітою tar, і потім вже стискають за допомогою bzip2. Такі архіви зазвичай мають розширення .tar.bz2. Bzip2 стискає більшість файлів ефективніше, але повільніше, ніж традиційніші zip. В цьому відношенні він схожий на інші сучасні алгоритми стиснення. Згідно з автором, Bzip2 програє від 10 до 15 відсотків найкращому класу алгоритмів стиснення даних, відомих в цей час (PPM), але при цьому вдвічі швидший при стисненні і в 6 разів швидший при розпаковуванні.

При стисненні в форматі 7z також використовуються спеціальні фільтри-нормалізатори. Так, для більш оптимального стиснення 32-бітного x86-коду використовуються нормалізують конвертери BCJ і BCJ2. Крім того, програма має оптимізує дельта-конвертер для деяких типів мультимедійних даних, наприклад незжатих 24-бітних зображень.

Особливості 7-Zip:

- ✓ дуже висока ступінь стиснення в форматі 7z завдяки використанню вдосконаленого алгоритму Лемпела-Зіва;
- ✓ для форматів ZIP і GZIP форматів, 7-Zip забезпечує ступінь стиснення на 2-10% вище, ніж співвідношення наданих PKZip;
- ✓ сильне шифрування AES-256 в 7z і ZIP форматах;
- ✓ можливість архівів для формату 7z;
- ✓ інтеграція з Windows Shell;
- ✓ потужний файловий менеджер;
- ✓ вбудована утиліта для тестування продуктивності
- ✓ потужна команда версія рядка;
- ✓ локалізації для 87 мов.

PowerArchiver - комерційний архіватор для Microsoft Windows, що розробляється ConeXware Inc. Має вбудовану підтримку створення або вилучення безлічі різних типів архівів, у тому числі ZIP, 7z і Tar. Крім того, в ньому є можливість вилучення архівів RAR, ACE та багатьох інших. Термін дії оціночної версії програми складає 30 днів. PowerArchiver був випущений в березні 1999 року. У той час представляв себе як безкоштовне програмне забезпечення і був написаний на Borland Delphi [4]. У розряд shareware він став відноситься з червня 2001. Перша назва програми (до PowerArchiver) було EasyZip. В даний час існує консольна версія програми і плагін для Microsoft Outlook. Інтерфейс PowerArchive переведений на 15 мов.

Основні функції архіватора PowerArchiver:

- ✓ Створення і монтування ISO файлів;
- ✓ Вікно попереднього перегляду вмісту архіву;
- ✓ Запис CD / DVD / Blu-ray;
- ✓ Повна підтримка Windows 7;
- ✓ Передовий модуль архівування;
- ✓ Розширений планувальник завдань;
- ✓ Стиснення 7-Zip;
- ✓ Швидке вилучення RAR архівів;
- ✓ Зворотна сумісність зі старими форматами архівів.

BandiZip - вільний файловий архіватор для операційних систем Windows і Mac OS. Підтримує більше 20-ти форматів, в тому числі і RAR, ACE, ZIPX. Може інтегруватися в провідник Windows і створити та розпакувати архіви через контекстне меню. Під час створення нового архіву BandiZip надає

можливість вибрати ступінь стиснення, формат кінцевого файлу (ZIP, ZIPX, EXE, TAR, TGZ, LZH, ISO, 7Z). Присутня функція Drag & Drop, а також призначення пароля до архівів (тільки для форматів ZIP і ZIPX) [4].

Основні можливості:

- ✓ Створення та розпакування «звичайних» архівів;
- ✓ Настроєний і призначений для користувача інтерфейс;
- ✓ Підтримка 35-ти мов;
- ✓ Попередній перегляд файлів перед розпакуванням;
- ✓ Пошук файлів всередині архіву;
- ✓ Шифрування архівів за допомогою алгоритмів ZipCrypto і AES-256;
- ✓ Перевірка файлів на наявність помилок;
- ✓ Може працювати з диска або USB флеш-накопичувача.

1.4 Висновки

І все ж архіватори, створені ще за часів дискет і «слабкого» Інтернету, до сих пір є популярними у використовуванні будь-яким користувачем ПК програм. А вся справа в тому, що за допомогою архіву ми можемо перенести або переслати з одного пристрою на інший групу файлів і папок в компактному вигляді, при цьому зберігши їх структуру розміщення. А це значною мірою економить час. Стиснення інформації являє собою процес, що перетворює інформацію, в результаті якого вона зменшується. Стиснення даних може бути без втрат або з регульованою втратою інформації, під час якого змінюється зміст стискаються даних. При відновленні даних з файлів стиснутих другим способом повне їх відновлення неможливе. Такий метод не підійде для стиснення, наприклад, текстових документів. Стиснення даних з втратами підходить для зображень, аудіо і відеофайлів.

Архіватори характеризуються ступенем стиснення файлів, швидкістю роботи і набором функцій, які до них додаються. Серед цих функцій - можливість створення архівів з файлів поточного каталогу, витяг, видалення і додавання інформації в архів, перегляд вмісту архівного файлу, а також стислих файлів в архіві. Важливими функціями є також можливість створення багатотомних і архівів, захист інформації в архіві і підтримка архівних файлів, які були створені іншими архіваторами. Не менш важливою є і можливість повного або часткового відновлення інформації з пошкоджених архівів.

2 СТИСНЕННЯ ДАНИХ ТА ЙОГО ВИДИ

2.1 Стиснення даних в мережах зв'язку

Швидкий розвиток науки і техніки вимагає широкого використання пристроїв обробки, передавання і запам’ятовування постійно зростаючих об’ємів даних. У зв'язку з розвитком в мережах зв'язку, особливо в мобільному, рішення задач кодування мовної інформації має важливе значення.

Основними галузями застосування кодування мовної інформації є:

— ущільнення каналу зв'язку для забезпечення більшої пропускної спроможності;

— забезпечення конфіденційності переданої з технічного каналу зв'язку мовної інформації.

Перша область застосування обумовлена збільшенням обсягів переданої мовної інформації по каналу зв'язку: при зростанні кількості переданої інформації (наприклад, збільшення обсягів телефонних переговорів усередині фірми, збільшення кількості вхідних дзвінків) виникає необхідність забезпечення великої пропускної здатності використовуваних каналів зв'язку для підтримки інформаційного обміну [2].

Друга сфера застосування обумовлена тим, що одночасно з розвитком радіоелектронних засобів зв'язку розвиваються і засоби електронного шпигунства. Ця обставина ставить завдання захисту мовної інформації в ряд найбільш актуальних.

Треба зауважити, що мовна інформація категорично відрізняється від текстової інформації. При шифруванні тексту використовують певний обмежений набір символів, тому при роботі з текстом можна використовувати такі шифри, як шифри перестановки, шифри заміни, шифри збивання і так далі. Мову не можна представити набором будь-яких знаків, тому застосовуються інші принципи кодування. При кодуванні мовної інформації ставлять завдання максимального стиснення мовного сигналу. Ефективність (якість) кодування оцінюється по суб'єктивним сприйняттям мови. Якість сигналу вимірюється часто за п'ятибальною шкалою MOS (Mean Opinion Score - середня суб'єктивна оцінка). Оцінка за шкалою MOS визначається шляхом обробки усереднених оцінок, які дають групами слухачів кільком мовним сигналам, відтвореними різними гучномовця.

Основними властивостями будь-якого методу стиснення даних є:

— якість стиснення, тобто ставлення довжини (в бітах) стиснутого представлення даних до довжини вихідного уявлення;
— швидкість кодування і декодування, що визначаються часом, що витрачається на кодування і декодування даних;
— обсяг необхідної пам'яті.

Рисунок 2.1 – Схема стиснення даних

В області стиснення даних, як це часто трапляється, діє закон важіля: алгоритми, що які використовують більше ресурсів (час або пам'ять), зазвичай досягають кращої якості стиснення, або навпаки: менш ресурсомісткі алгоритми за якістю стиснення, як правило, поступаються більш ресурсомісткими. Таким чином, побудова алгоритму стиснення даних представляється досить нетривіальним завданням, так як необхідно домогтися досить високої якості стиснення при невеликому обсязі використовуваних ресурсів. Зрозуміло, що критерії оцінки методів стиснення з практичної точки зору залежні від передбачуваної області застосування. Наприклад, при використанні стиснення в системах реального часу необхідно забезпечити

високу швидкість кодування і декодування; для вбудованих систем критичний параметр - обсяг необхідної пам'яті; для систем довгострокового зберігання даних - якість стиснення і швидкість декодування і т. д. Без перебільшення можна сказати, що відомі тисячі різних методів стиснення даних, але багато них помітно поступаються іншим по всіх параметрах і тому не представляють інтересу.

2.2 Стиснення без втрат

Щоб зберегти всі дані які ви хочете бачити на своєму мобільному пристрої або передати швидко через мережу, повинен бути спосіб ефективно стискати та розпаковувати дані без втрати інформації. Стиснення даних без втрат (англ. Lossless data compression) - метод стиснення даних, при використанні якого, закодовані дані можуть бути відновлені з точністю до біта. Цей тип стиснення принципово відрізняється від стиснення даних з втратами. Стиснення без втрат використовується у випадках, коли важливо, щоб вихідні і розпаковані дані були ідентичними, або коли відхилення від вихідних даних були б небажаними [1]. Типовими прикладами є виконувані програми, текстові документи і вихідний код. Деякі формати файлів зображень, такі як PNG або GIF, використовують тільки стиснення без втрат, в той час як інші можуть використовувати методи з втратами. Аудіо формати без втрат найчастіше використовуються для архівування або виробництва, тоді як файли меншого розміру з втратами звуку зазвичай використовуються на портативних плеєрах і в інших випадках, коли простір для зберігання обмежений або точне відтворення аудіо не потрібно. Для кожного з типів цифрової інформації, як правило, існують свої оптимальні алгоритми стиснення без втрат. Алгоритми стиснення даних без втрат можна ефективно реалізувати на мобільному телефоні пристроїв, навіть з апаратними обмеженнями, такими як низька обчислювальна потужність, статична пам’ять та час автономної роботи. Можна виділити декілька основних алгоритмів для стиснення даних без втрат:

— Алгоритм Хаффмана, Шеннона-Фано;
— Адаптивне арифметичне кодування;
— Lempel-Ziv 77 (LZ77 та LZ78) - використовують словниковий метод;
— Lempel-Ziv-Welch (LZW) - використовує словниковий метод для посилання на майбутні дані, щоб точно відповідати цим даним вже в закодованому вигляді;

— PPM - алгоритм, який є статистичним алгоритмом стиснення даних основане на контекстному моделюванні та прогнозуванні;

— BWT- не є самостійним, зменшує розмір даних, це лише полегшує дані;

— RLE – найпростіший алгоритм, оснований на повторюванні символів, які йдуть один за одним.

Наступний крок цих алгоритмів - порівняння за допомогою основного тесту, щоб визначити, який з них є оптимальним для впровадження на мобільних пристроях.

Почнемо з алгоритму Шеннона-Фано:

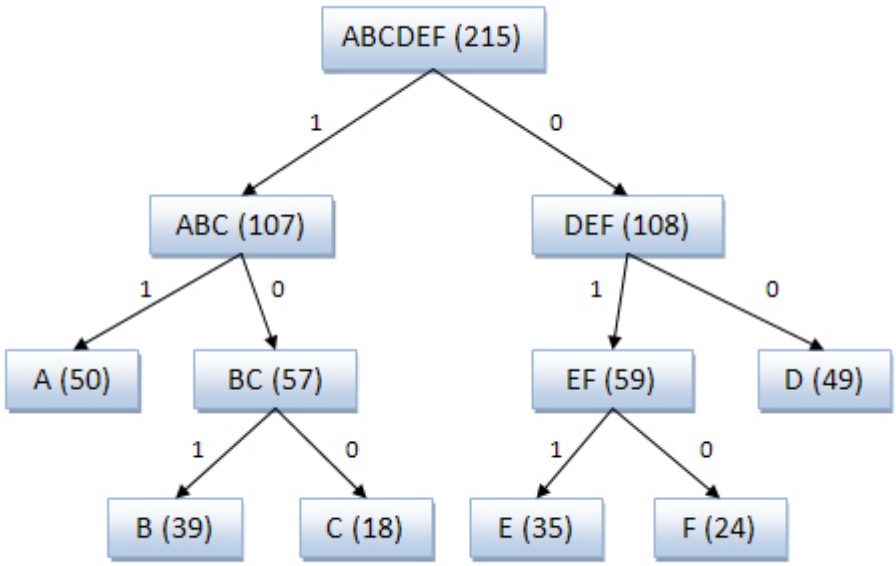


Рисунок 2.2 – Приклад алгоритму Шеннона-Фано

1. Усі m символів дискретного джерела розташовані у порядку зменшення за ймовірністю їх появи.

2. Сформована колонка символів поділяється на дві групи так, що сумарні ймовірності кожної групи трохи відрізняються одна від одної.

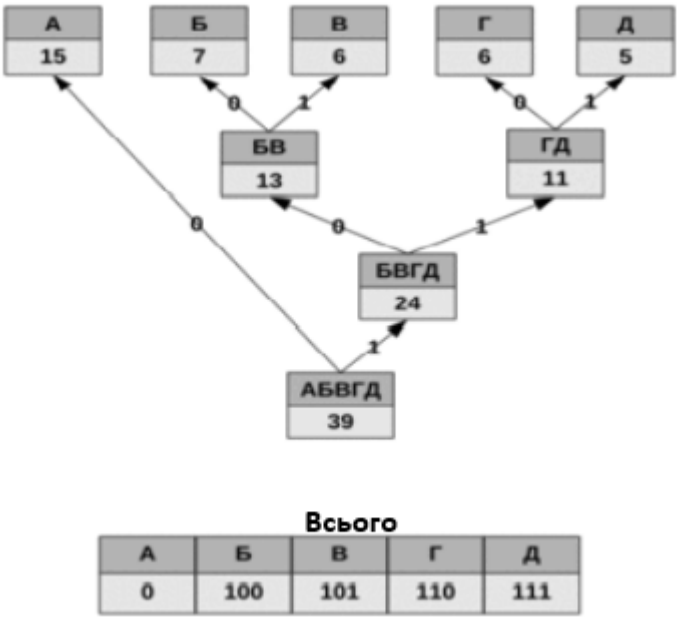
3. Верхня група кодується символом «1», а нижня - «0».

4. Кожна група поділяється на дві підгрупи з близькими загальними ймовірностями; верхня підгрупа кодується символом «1», а нижня - «0».

5. Процес поділу та кодування триває до тих пір, поки кожна підгрупа не містить один символ вихідного повідомлення.

6. Для кожного символу джерела пишеться код; зчитування коду здійснюється зліва направо.

Ось класичний алгоритм Хаффмана на вході отримує таблицю частот появи символів у повідомленні. Далі, на основі цієї таблиці будується дерево кодування Хаффмана (Н-дерево) (рис. 1.1).



Всього

А	Б	В	Г	Д
0	100	101	110	111

Рисунок 2.3 – Приклад побудови дерева Хаффмана

Етапи алгоритму:

1. Символи введеного алфавіту утворюють список вільних вузлів. Кожен аркуш має вагу, яка може дорівнювати або ймовірності, або кількості повторень символу в стисненому повідомленні.

2. Вибрано два вільних вузли дерева з найменшими вагами.

3. Їх батьки створені з вагою, рівною їх загальній вазі.

4. Батьківський додається до списку вільних вузлів, а два його нащадки вилучаються з цього списку.

5. Одна дуга, яка залишає батьківський, відображається в біт 1, інша - в біт 0. Значення бітів гілок, що походять від кореня, не залежать від ваги нащадків.

6. Етапи, починаючи з другого, повторюються, поки у списку вільних вузлів не залишиться лише один вільний вузол. Він буде вважатись коренем дерева.

Для статистичного стиснення без втрат векторів руху і даних, отриманих після квантування, використовуються відомі в теорії інформації підходи Найбільш часто застосовуються коди змінної довжини (наприклад, код Хаффмана) або Адаптивне арифметичне кодування. При цьому відомо, що Адаптивне арифметичне кодування більш ефективне з точки зору ступеня стиснення. На відміну від алгоритму Хаффмана, цей алгоритм не має жорсткої

постійної відповідності вхідних символів групам біт вихідного потоку. Це дає алгоритму велику гнучкість в поданні дрібних частот зустрічальності символів. Крім цього, перевершує алгоритм Хаффмана по ефективності стиснення, дозволяє стискати дані з ентропією, меншою 1 біта на кодований символ. Адаптивне арифметичне кодування - алгоритм стиснення інформації без втрат, який при кодуванні ставить у відповідність тексту дійсне число з відрізка [0; 1). Даний метод, як і алгоритм Хаффмана, є ентропійним, тобто довжина коду конкретного символу залежить від частоти розповсюдженості цього символу в тексті. Арифметичні коди вказують вищі результати стиснення, ніж алгоритм Хаффмана, для даних з нерівномірними розподілами ймовірностей кодованих символів.

Алгоритми стиснення без втрат Lempel-Ziv 77 (LZ77) та LZ78 - алгоритми, опубліковані Абрахамом Лемпелем та Якобом Зивом в 1977 і 1978 роках відповідно. Він став основою інших алгоритмів сім'ї LZ *: LZW, LZSS, LZMA та використовує словниковий підхід.

LZ77 використовується як основа для компресійних інструментів, таких як GZip. Алгоритм асиметричний, оскільки з часом, кодування набагато вибагливіше, ніж декодування. Алгоритм LZ77 використовує такі структури даних, як двійкові дерева, суфіксальні дерева та хеш-таблиці, який забезпечує швидкий пошук без потреби у великій пам’яті. LZ77 стискає дані шляхом заміни розділів даних посиланням на відповідність даних, які вже пройшли як кодер, так і декодер. Під час розпакування файлу не потрібен пошук даних, оскільки компресор видав явний потік літералу, розташування та довжини відповідності. Процес стає ще більш ефективним, якщо вікно повністю зберігається в кеш-пам’яті, тому отримання збігу відбувається швидко, незалежно від того де це відбувається у вікні. Алгоритм LZ77 працює, підтримуючи поточний покажчик на вхідних даних. Знайдені символи перед тим, як поточний символ складає буфер пошуку, були як символи, які з'являються після поточного символу, розміщуються в перспективний буфер. Буфери створюють вікно, яке показує розділ введення, який переглядається в даний час. Як поточний покажчик рухається вперед, так вікно рухається через вхід. Поки символи знаходяться в буфері перспективи, алгоритм шукає в буфері пошуку найдовший збіг. Замість надсилання відповідних символів вони кодуються зі зміщенням від поточного вказівника, розміром відповідності та символ у буфері перспективи [11]. Кодер та декодер повинні відстежувати останні 2 КБ або 4 КБ останніх даних. Кодер повинен зберігати ці дані для

пошуку збігів, поки декодер потребує щоб він розумів збіги, на які посилається кодер. LZ77 надає можливість збільшити пам'ять до підвищення продуктивності. З більшим вікном є покращення швидкості знаходження збігів.

Алгоритм LZ78 має трохи іншу ідею. Цей алгоритм використовує словниковий підхід, генеруючи тимчасовий словник під час кодування і декодування. Словник спочатку порожній і алгоритм намагається закодувати перший символ. На кожній ітерації намагаємося збільшити кодувальний префікс, поки такий префікс є в словнику. Кодові слова такого алгоритму будуть складатися з двох частин - номера в словнику найдовшого знайденого префікса і символу, який йде за цим префіксом. При цьому після кодування такої пари префікс з приписаним символом додається в словник, а алгоритм продовжує кодування з наступного символу.

Опублікування алгоритму LZW справило велике враження на всіх фахівців зі стиснення інформації. За цим послідувала велика кількість програм і додатків з різними варіантами цього методу. Цей метод дозволяє досягти одну з найкращих ступенів стиснення серед інших існуючих методів стиснення графічних даних, при повній відсутності втрат або спотворень у вихідних файлах. Використовується в файлах формату TIFF, PDF, GIF, PostScript та інших.

Процес стиснення виглядає наступним чином: послідовно зчитуються символи вхідного потоку і відбувається перевірка, чи існує в створеній таблиці рядків такий рядок. Якщо такий рядок існує, зчитується наступний символ, а якщо рядок не існує, в потік заноситься код для попереднього знайденого рядка, рядок заноситься в таблицю, а пошук починається знову. Наприклад, якщо стискають байтові дані (текст), то рядків в таблиці виявиться 256 (від "0" до "255"). Якщо використовується 10-бітний код, то під коди для рядків залишаються значення в діапазоні від 256 до 1023. Нові рядки формують таблицю послідовно, тобто можна вважати індекс рядка її кодом.

Формат, що відноситься до стиснення без втрат, який використовує алгоритм LZW є GIF До недавнього часу формат стиснення GIF був найпопулярнішим серед форматів, що використовуються для стиснення зображень. Розшифровується GIF як Graphics Interchange Format (Формат графічного обміну). Особливостями даного формату є підтримка до 256 кольорів, зберігання декількох зображень в одному файлі, за допомогою якого створюються анімації картинок (рисунок 2.5). В багатьох випадках це властивість використовується для створення мультиплікації в web-браузерах.

Завдяки кращому стисненню і більшій глибині кольору формат GIF був замінений0 на формат JPEG, але формат GIF в даний час користується популярністю серед інших додатків, однак поширенню сильно заважають юридичні перешкоди [11].

Рисунок 2.4 – Приклад GIF-анімації

Структура файлів формату JPEG має блоковий характер. Іншими словами, всі файли формату складаються з окремих блоків і ніяк не пов'язані один з одним. Деякі програми, які не мають можливість розпізнати тип блоків, пропускають їх. Для цього, кожен блок містить у собі в заголовку його розмір. Анімацію складають всі зображення, які йдуть одне за іншими змінюючи один одного, створюючи при цьому ілюзію руху. В файлі також можуть перебувати й інші блоки, і неважливо, будуть вони перебувати до або після (або навіть між ними):

— Коментарі. Прихований текст видно тільки через спеціальні програми, такі як аніматори GIF;
— Простий. Рядки символів з обмеженнями форматування. В даний час не використовується;
— Графічні блоки управління, які встановлюють вихідні параметри для окремих зображень;
— Глобальні та локальні колірні палітри, що визначають, які кольори будуть на зображеннях;
— Спеціальні блоки, які можуть використовуватися тільки програмами, які знають про їхнє існування.

Мінімально необхідний набір блоків - простий не анімований GIF (рисунок 2.5):

Рисунок 2.5 – Структура файла GIF

Дескриптор глобального екрану вказує розмір області, а також колір фону. Дескриптори окремих зображень визначають, в якому місці логічного екрана має розташовуватися зображення. Глобальна палітра кольорів. Поодинокі ілюстрації файлу можуть використовувати глобальну палітру кольорів, або самостійно встановити свою. Плюсом таблиці кольорів вважається скорочення файлу, також легкість у застосуванні під концепціях. Типи блоків. Потім розташовується змінна частина GIF-файлу. Зображення формується з блоків, що визначаються 1-байтовим кодом в джерелі блоку. Завершувач. Блок завершувача зазначає кінець файлу. Це являється однобайтовим блоком, що складається виключно з коду блоку.

Для декодування на вхід подається тільки закодований текст, оскільки алгоритм LZW може відтворити відповідну таблицю перетворення безпосередньо по закодованому тексту. Алгоритм генерує однозначно декодувальний код за рахунок того, що кожен раз, коли генерується новий код, новий рядок додається в таблицю рядків. LZW постійно перевіряє, чи є рядок вже відомий, і якщо так, виводить існуючий код без генерації нового. Таким чином, кожен рядок буде зберігатися в єдиному екземплярі і мати свій унікальний номер. Отже, під час декодування, під час отримання нового коду генерується новий рядок, а при отриманні вже відомого, рядок витягується з словника.

Плюси та мінуси використання цього алгоритму:

+ Не потребує обчислення ймовірностей народження символів або кодів;
+ Для декомпресії не треба зберігати таблицю рядків в файл для розпакування. Алгоритм побудований таким чином, що ми в змозі відновити таблицю рядків, користуючись тільки потоком кодів;
+ Даний тип компресії не вносить спотворень в вихідний графічний файл, і підходить для стиснення растрових даних будь-якого типу;
- Алгоритм не проводить аналіз вхідних даних, тому не оптимальний.

PPM - адаптивний статистичний алгоритм стиснення даних без втрат, заснований на контекстному моделюванні і прогнозі. Модель PPM використовує контекст - безліч символів в стислому потоці, що передують даному, щоб пророкувати значення символу на основі статистичних даних. Сама модель лише пророкує значення символу, безпосередньо стиснення здійснюється алгоритмами ентропійного кодування, як наприклад, алгоритм Хаффмана, арифметичне кодування. Довжина контексту, яка використовується при прогнозі, зазвичай сильно обмежена[5]. Ця довжина позначається n і визначає порядок моделі PPM, що позначається як PPMn. Необмежені моделі також існують і позначаються просто PPM*. Якщо передбачення символу по контексту з n символів не може бути вироблено, то відбувається спроба передбачити його за допомогою n-1 символів. Рекурсивний перехід до моделей з меншим порядком відбувається, поки передбачення не відбудеться в одній з моделей, або коли контекст стане нульової довжини n = 0. Моделі ступеня 0 і -1 слід описати особливо. Модель нульового порядку еквівалента нагоди контекстно-вільного моделювання, коли ймовірність символу визначається виключно з частоти його появи в стисливому потоці даних. Подібна модель зазвичай застосовується разом з кодуванням по Хаффману. Для отримання хорошої оцінки ймовірності символу необхідно враховувати контексти різної довжини. PPM є варіантом стратегії перемішування, коли оцінки ймовірностей, зроблені на підставі контекстів різної довжини, об'єднуються в одну загальну ймовірність. Отримана оцінка кодується будь-яким ентропійним кодером ЕК, зазвичай це якась разновідность арифметичного кодера. На етапі ентропійного кодування і відбувається власне стиснення. Велике значення для алгоритму PPM має проблема обробки нових символів, які ще не зустрічалися у вхідному потоці. Це проблема носить назву «проблема нульової частоти». Деякі варіанти реалізацій PPM вважають лічильник нового символу рівною фіксованою величиною, наприклад, одиниці. Інші реалізації, як наприклад, PPMD, збільшують псевдолічильник нового символу кожен раз, коли, дійсно, в потоці

з'являється новий символ. Іншими словами, PPMD оцінює ймовірність появи нового символу як відношення числа унікальних символів до загальної кількості використовуваних символів. Опубліковані дослідження алгоритмів сімейства PPM з'явилися в середині 1980-х років. Програмні реалізації не були популярні до 1990-х років, тому що моделі PPM вимагають значної кількості оперативної пам'яті. Сучасні реалізації PPM є одними з кращих серед алгоритмів стиснення без втрат для текстів на природній мові.

BWT-перетворення (Burrows-Wheeler Transform) - техніка для стиснення інформації (особливо текстів), заснована на перетворенні, яке відкрито в 1983 р. BWT є унікальним алгоритмом. По-перше, незвично саме перетворення, відкрите в науковій області, далекій від архіваторів. По-друге, BWT перетворення надзвичайно просте та сам BWT компресор складається з послідовності декількох розглянутих раніше алгоритмів і вимагає, тому, для своєї реалізації найрізноманітніших програмних навичок.

BWT не стискує дані, але перетворює блок даних в формат, виключно відповідний для компресії. Перш за все, слід відзначити одну з його особливостей. BWT керує відразу цілий блок даних. Тобто, йому заздалегідь відомі відразу всі елементи вхідного потоку або, принаймні, досить великого блоку. Це робить скрутним використання алгоритму в тих областях застосування, де потрібно стиснення даних "на льоту", символ за символом. В цьому відношенні BWT навіть більш вимогливий, ніж методи сімейства LZ, що використовують для стиснення ковзне вікно. Слід зазначити, що можлива реалізація стиснення даних на основі BWT, що обробляє дані послідовно по символам, а не по блокам [5]. Але швидкісні характеристики програм, що використовують таку реалізацію, будуть дуже далекі від досконалості. Найбільша проблема полягає в тому, що BWT вимагає розподіл оперативної пам'яті для всіх вхідних та вихідних потоків і необхідний великий буфер для виконання необхідних сортувань. Незважаючи на те, що стиснення на основі BWT могло бути виконано з дуже малою пам’яттю, звичайні налаштування використовують швидке сортування алгоритмів та структури даних, які потребують великої кількості пам'яті для подачі швидкості. Незалежно від проблем із пам'яттю, алгоритми, що реалізують BWT стиснення файлів на високому ступені стиснення.

Після того, як з'ясувалося, що наші дії цілком можливі, і дані ми не спотворили, можна перейти до питання розгляду корисності перетворення. Головне завдання перетворення Burrows-Wheeler Transform полягає в тому, щоб

переставити символи таким чином, щоб їх можна було легко стиснути. Щоб зрозуміти цей процес досить уявити потік даних, який складається з набору деяких стабільних поєднань символів. Поєднання символів, що дозволяє передбачити деякий невідомий символ, називається контекстом. Контекст називається стійким (стабільним), тому що розподіл частот символів, що безпосередньо прилягають до нього зліва і справа, змінюється в межах блоку. Таким чином, головна властивість перетворення в тому, що воно збирає разом символи, відповідні схожим контекстам. Чим більше стабільних контекстів в блоці даних, тим краще буде стискатися отриманий в результаті перетворення блок. Практика показує, що в результаті перетворення звичайних текстів більше половини з усіх символів такі ж.

Алгоритм RLE (англ. Run-Length Encoding) - один з найстаріших і найпростіших алгоритмів архівації, який замінює однакові символи, що йдуть підряд парою. Цей алгоритм ефективний для рядків, що містять багато ланцюжків символів, що повторюються, наприклад, результату перетворення Burrows-Wheeler. Стиснення в RLE відбувається за рахунок заміни ланцюжків однакових байт на пари "лічильник, значення". Кращий, середній і найгірший коефіцієнти стиснення - 1/32, 1/2, 2/1. Одна з реалізацій алгоритму така: шукають байт, що найменше раз зустрічається, його називають його префіксом і роблять заміни ланцюжків однакових символів на трійки "префікс, лічильник, значення". Якщо ж цей байт зустрічається в вихідному файлі один або два рази поспіль, то його замінюють на пару "префікс, 1" або "префікс, 2". Залишається одна невикористана пара "префікс, 0", яку можна використовувати як ознака кінця упакованих даних. До позитивних сторін алгоритму, можна віднести те, що він не вимагає додаткової пам'яті при роботі, і швидко виконується[5]. Алгоритм застосовується в форматах PCX, TIFF, BMP. Цікава особливість групового кодування в PCX полягає в тому, що ступінь архівації для деяких зображень може бути істотно підвищений всього лише за рахунок зміни порядку квітів в палітрі зображення.

2.3 Стиснення з втратами

Стиснення з втратами (англ. Lossy compression) – метод стиснення даних, при якому розпакований файл відрізняється від оригіналу, проте є корисним для використання. Перевага методів стиснення з втратами над методами стиснення без втрат полягає у великій мірі стиснення. При використанні

стиснення з втратами необхідно враховувати, що повторне стиснення зазвичай призводить до деградації якості. Однак, якщо повторне стиснення виконується без будь-яких змін, якість не змінюється. Стиснення з втратами найчастіше використовується для мультимедійних даних, особливо для потокової передачі даних та телефонії. Ідея, що лежить в основі всіх алгоритмів стиснення з втратами, досить проста: на першому етапі треба видалити незначну інформацію, а на другомі етапі до решти даних застосувати найбільш підходящий алгоритм стиснення без втрат [1]. Підходи тут можуть бути різними в залежності від типу стиснених даних. Якщо ми працюємо зі звуком, найчастіше видаляють частоти, які людина просто не здатна сприйняти. Чим більше втрати, тим вище ступінь стиснення і тим сильніше спотворення. Але помірних спотворень, як правило, людина не помічає. Це пов'язано з особливостями сприйняття звуку і зображень. Алгоритми побудовані так, щоб при розумній мірі компресії спотворення були практично непомітні. Але, зрозуміло, професійний фотограф, наприклад, або людина близькою до нього професії відразу впізнає спотворення фотографії, музикант або людина з хорошим музичним слухом відрізнить попередньо стиснений звук від оригіналу.

1. JPEG - досить потужний метод. Практично він є стандартом для повнокольорових зображень. Оперує алгоритм областями 8x8, на яких яскравість і колір змінюються порівняно плавно. Внаслідок цього, при розкладанні матриці такої області в подвійний ряд по косинусам значущими виявляються тільки перші коефіцієнти. Таким чином, стиснення в JPEG здійснюється за рахунок плавності зміни кольорів у зображенні. Алгоритм розроблений групою експертів в області фотографії спеціально для стиснення 24-бітових зображень. JPEG - Joint Photographic Expert Group - підрозділ в рамках ISO - Міжнародної організації зі стандартизації. Назва алгоритму читається ['jei'peg]. В цілому алгоритм заснований на дискретному косинусоїдальному перетворенні - ДКП (Discrete-Cosine Transform - DCT), що застосовується до матриці зображення для отримання деякої нової матриці коефіцієнтів. Для отримання вихідного зображення застосовується зворотне перетворення. ДКП розкладає зображення по амплітудам деяких частот. Таким чином, при перетворенні ми отримуємо матрицю, в якій більшість коефіцієнті або близькі, або дорівнюють нулю. Крім того, завдяки недосконалості людського зору, можна апроксимувати коефіцієнти більш грубо, без помітної втрати якості зображення [2]. Для цього використовується квантування

коефіцієнтів (quantization). У найпростішому випадку - це арифметичний побітовий рух вправо. При цьому перетворенні втрачається частина інформації, але може досягати великого ступеня стиснення.

Як працює алгоритм (якщо ми стискаємо 24-бітове зображення) показано на рис. 2.6:

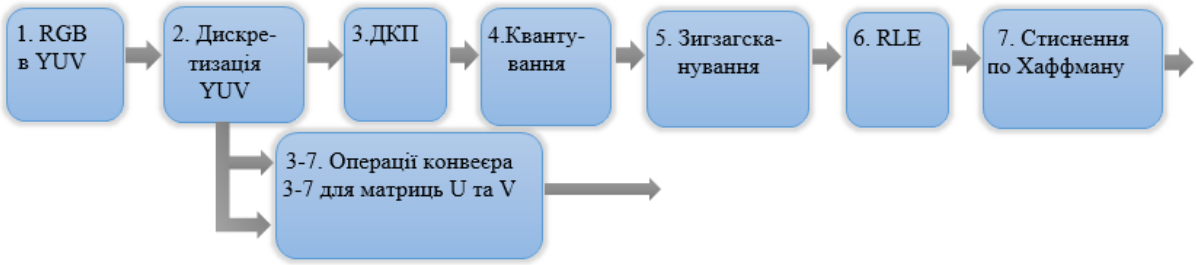


Рисунок 2.6. – Алгоритм стиснення JPEG

Крок 1.

Зображення з колірного простору RGB переводимо, з компонентами, що відповідають за червону (Red), зелену (Green) і синю (Blue) складові кольору точки, в колірний простір YCrCb. У ньому Y - кольорова складова, а Cr, Cb - компоненти, що відповідають за колір (хроматичний червоний і хроматичний синій). Через те, що людське око менш чутливе до кольору, ніж до яскравості, з'являється можливість архівувати масиви для Cr і Cb компонент з великими втратами і, відповідно, великими ступенями стиснення. Подібне перетворення вже давно використовується в телебаченні. На сигнали, що відповідають за колір, там виділяється більш вузька смуга частот. Спрощений перехід з колірного простору RGB в колірний простір YCrCb можна уявити за допомогою матриці переходу:

(2.1)

Зворотне перетворення здійснюється множенням вектора YUV на зворотну матрицю:

(2.2)

Крок 2.

Розбиваємо вихідне зображення на матриці 8x8. З кожної формуємо 3 робочі матриці ДКП - по 8 біт окремо для кожної компоненти. При великих ступенях стиснення цей процес може виконуватися трохи складніше. Зображення ділиться по компоненту Y - як і в першому випадку, а для компонентів Cr і Cb матриці набираються через рядок і через стовбчик. Тобто з вихідної матриці розміром 16x16 виходить тільки одна робоча матриця ДКП. При цьому, як можна помітити, що ми втрачаємо 3/4 корисної інформації про кольорові складові зображення і отримуємо відразу стиснення в два рази. Ми можемо робити так завдякі роботі в просторі YCrCb. На результуючому RGB зображенні, як показала практика, це позначається несильно.

Крок 3.

Застосовуємо ДКП до кожної робочої матриці. При цьому, отримуємо матрицю, в якій коефіцієнти в лівому верхньому кутку відповідають низькочастотної складової зображення, а в правому нижньому - високочастотної. Поняття частоти випливає з розгляду зображення як двовимірного сигналу (аналогічно розгляду звуку як сигналу). Плавна зміна кольору відповідає низькочастотної складової, а різкі скачки - високочастотної.

Крок 4.

На данному етапі відбувається квантування, якщо простими словами, це просто розподіл робочої матриці на матрицю квантування поелементно. Для кожної компоненти (Y, U і V), в загальному випадку, задається своя матриця квантування q [u, v] (далі МК).

(2.3)

Здійснюється управління ступенем стиснення та відбуваються найбільші втрати. Якщо задаючи МК з великими коефіцієнтами, то отримаємо більше нулів і, отже, більшу ступінь стиснення. У стандарт JPEG включені рекомендовані МК, побудовані дослідним шляхом. Матриці з більшою чи меншою мірою стиснення отримують шляхом множення вихідної матриці на деяке число gamma. З квантуванням пов'язані і специфічні ефекти алгоритму. При великих значеннях коефіцієнта gamma втрати в низьких частотах можуть бути настільки великі, що зображення розпадеться на квадрати 8x8. Втрати в високих частотах можуть проявитися в так званий "ефект Гіббса", коли навколо контурів з різким переходом кольору утворюється своєрідний "німб".

Крок 5.

Переводимо матрицю 8x8 в 64-елементний вектор за допомогою "зигзаг" -сканування, тобто беремо елементи з індексами (0,0), (0,1), (1,0), (2,0)….

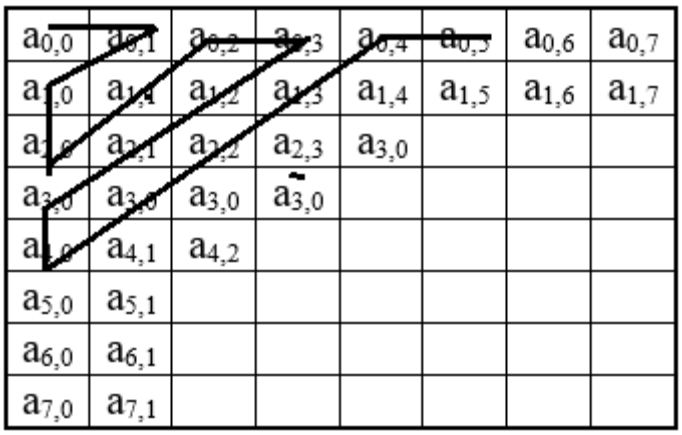


Рисунок 2.7 – Матриця 8x8 за допомогою "зигзаг" –сканування

Крок 7.

Згортаємо вектор за допомогою алгоритму групового кодування. При цьому отримуємо пари типу (пропустити, число), де "пропустити" є лічильником пропускаються нулів, а "число" - значення, яке необхідно поставити в наступну комірку. Так, вектор 42 3000-2 00001 ... буде згорнуто в пари (0,42) (0,3) (3, -2) (4,1)

Крок 8.

Згортаємо отримані пари кодуванням по Хаффману з фіксованою таблицею. Процес відновлення зображення в цьому алгоритмі повністю симетричний. Метод дозволяє стискати деякі зображення в 10-15 разів без серйозних втрат.

Багато в чому недолік методу стиснення JPEG полягає в тому, що при стисненні ніяк не враховується специфіка зображення, структура і характерні ділянки. Саме врахування специфіки зображення лежить в основі фрактального методу [3].

2. Фрактал - це структура, виділена при аналізі зображення, і володіє схожою формою незалежно від її розмірів. В даному форматі стиснення використовується система функцій (Iterated Function System - IFS). Найбільш поширеним прикладом фрактальної зображення є зображення папороті (рис. 2.3), яке складається з 4-х афінних перетворень. Фрактальна компресія - це пошук самоподібних областей і визначення для них параметрів афінних перетворень [10]. Даний алгоритм зручний тим, що в деяких випадках дозволяє отримати дуже високі коефіцієнти стиснення при прийнятій візуальній якості для реальних фотографій природних об'єктів.

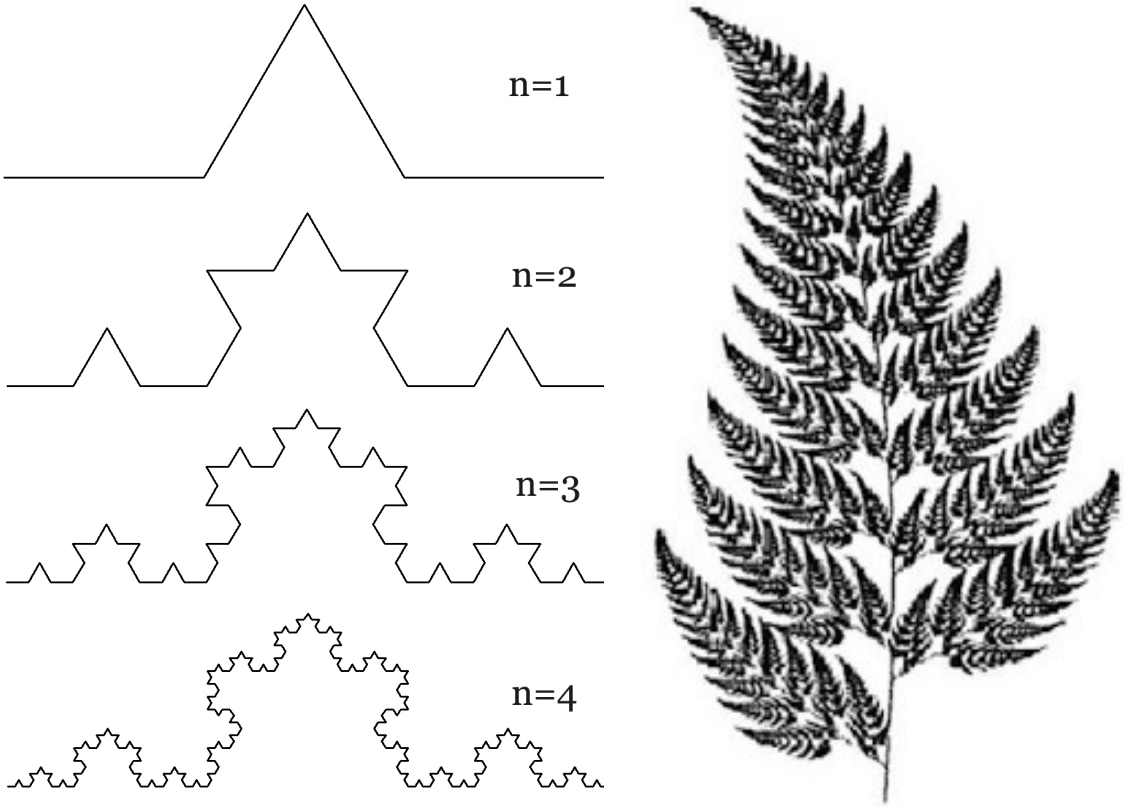


Рисунок 2.8 – Зображення, згенеровані
за допомогою IFS

Збереження якості при стисненні не є єдиним плюсом даного формату. Так, фрактальний архіватор дозволяє, наприклад, при розпаковці довільно змінювати розширення зображення без появи ефекту зернистості. Для фрактального алгоритму компресії, як і для інших алгоритмів стиснення з втратами, дуже важливі механізми, за допомогою яких можна буде регулювати ступінь стиснення і ступінь втрат. До тепер розроблений досить великий набір таких методів. По-перше, можна обмежити кількість перетворень, свідомо забезпечивши ступінь стиснення не нижче фіксованої величини. По-друге, можна зажадати, щоб в ситуації, коли різниця між оброблюваним фрагментом і найкращим його наближенням буде вище певного порогового значення, цей фрагмент дробився обов'язково. По-третє, можна заборонити дробити фрагменти розміром менше, припустимо, чотирьох точок. Змінюючи порогові значення і пріоритет цих умов, можна дуже гнучко управляти коефіцієнтом компресії зображення: від побітної відповідності, до будь-якого ступеня стиснення. На малюнку 2.9 показаний кодер фрактального стиснення.

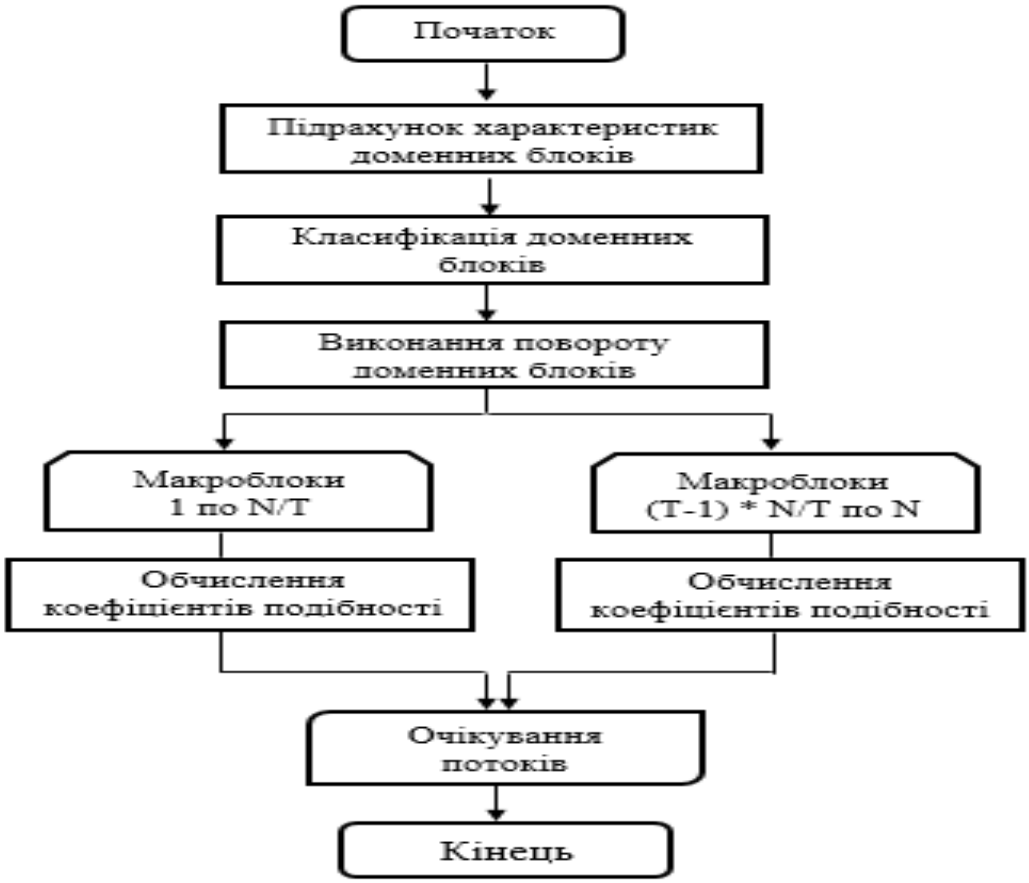


Рисунок 2.9 – Функціональна схема кодера фрактального стиснення

Основна складність фрактального стиснення полягає в тому, що для пошуку відповідних доменних блоків потрібен повний перебір. Оскільки при цьому переборі щораз повинні порівнятися два масиви, дана операція виходить досить довгою. Порівняно з простим перетворенням її можна звести до операції скалярного добутку двох масивів, однак ця операція також тривала.

Таблиця 2.1 – Порівняння алгоритмів стиснення зображень

Формат зображення	Доступні кольори	Стиснення	Розмір файла (середні значення)	Підходить краще для:
JPEG	16,7 млн	З втратами	Невеликий (< 1 МБ)	Зберіганння
Фрактальне стиснення	16 млн	З втратами	Великий	Інтернет

З огляду на все критерії при виборі кращого методу, включаючи ступінь стиснення, якість зображення і область використання, найоптимальнішим алгоритмом є JPEG. Використовують і оптимізують формат JPEG частіше за

інших форматів через те, що більшість сучасних зображень мають велику кількість колірних відтінків і градієнтів. При стисненні навіть більше ніж на 50-70%, наші очі можуть і не помітити відмінностей від оригіналу великого розміру. Крім цього, дуже низька швидкість кодування фрактального стиснення робить його ефективним. Але якщо ми говоримо про зберігання зображень, варто визнати, що JPEG з цим завданням впорається погано, так як при багаторазовому стисканні якість помітно погіршиться. За якістю колірного відображення також перевершує .jpeg, але розмір файлу буде на 30-40% більше. У таблиці 2.1 наведено порівняння перерахованих вище алгоритмів стиснення зображень

2.4 Висновки

Виходячи з цього розділу ми зрозуміли, що всі методи стиснення можна розділити на дві великі групи: стиснення з втратами і стиснення без втрат. Стиснення без втрат застосовується в тих випадках, коли інформацію потрібно відновити з точністю до біта, пікселя і т.д. Такий підхід є можливим при стисненні, наприклад, текстових даних. У деяких випадках, однак, не потрібно точного відновлення інформації та допускається використовувати алгоритми, що реалізують стиснення з втратами, яке, на відміну від стиснення без втрат, зазвичай простіше реалізується і забезпечує більш високу ступінь архівації.. Існує багато практичних алгоритмів стиснення даних, але всі вони базуються на трьох теоретичних способах зменшення надлишковості даних. Перший спосіб полягає в зміні вмісту даних, другий – у зміні структури даних, а третій – в одночасній зміні як структури, так і вмісту даних. Якщо при стисненні даних відбувається зміна їх вмісту, то метод стиснення є незворотнім, тобто при розархівуванні даних з архіву не відбувається повне відновлення інформації. Такі методи часто називаються методами стиснення з регульованими втратами інформації.

Зрозуміло, що ці методи можна застосовувати тільки для таких типів даних, для яких втрата частини вмісту не приводить до суттєвого спотворення інформації. До таких типів даних відносяться відео- та аудіодані, а також графічні дані. Методи стиснення з регульованими втратами інформації забезпечують значно більший ступінь стиснення, але їх не можна застосовувати до текстових даних. Прикладами форматів стиснення з втратами інформації можуть бути: JPEG для графічних даних; MPG – для для відеоданих; MP3 – для аудіоданих. Якщо при стисненні даних відбувається тільки зміна структури

даних, то метод стиснення є зворотнім. У цьому випадкові з архіву можна відновити інформацію повністю. Зворотні методи стиснення можна застосовувати до будь-яких типів даних, але вони дають менший ступінь стиснення у порівнянні з незворотними методами стиснення. GIF є прикладом стиснення без втрати інформації.

3 ПОРІВНЯННЯ АРХІВАТОРІВ

Всі алгоритми стиснення оперують вхідним потоком інформації з метою отримання більш компактного вихідного потоку за допомогою деякого перетворення. Основними технічними характеристиками процесів стиснення і результатів їх роботи є:

- Ступінь стиснення - відношення обсягів вихідного і результуючого потоків;
- Швидкість стиснення - час, що витрачається на стиснення деякого обсягу інформації вхідного потоку, до отримання з нього еквівалентного вихідного потоку;
- Якість стиснення - величина, що показує, на скільки сильно упакований вихідний потік при застосуванні до нього повторного стиснення за тим же або іншим алгоритмом.

3.1 Порівняння за ступенем стиснення

Під ступенем стиснення розуміють відношення розмірів стисненого файлу і вихідного, виражене у відсотках. Ступінь стиснення залежить від використовуваної програми стиснення, методу стиснення і типу вихідного файлу. Найкраще стискаються файли графічних образів, текстові файли, файли даних, ступінь стиснення яких може досягати 5 - 40%, менше стискуються файли виконуваних програм і завантажувальних модулів - 60 - 90%. Майже не стискуються архівні файли. Програми для архівації відрізняються використовуваними методами стиснення, що відповідно впливає на ступінь стиснення [6].

Ступінь стиснення файлів характеризується коефіцієнтом k, який можна розрахувати за формулою:

(3.1)

де k – ступінь стиснення, S_0 – об’єм вихідних даних, S_c - об’єм даних, що стиснули.

Очевидно, що вищий ступінь стиснення означає більшу ефективність алгоритму; k = 1 означає, що алгоритм не здійснює стиснення, а k <1 означає,

що алгоритм шкідливий, тобто обсяг результуючого повідомлення більший за початковий.

Отже проаналізуємо за ступенем стиснення файли, розміром 16,3 Мб та 0,229 Мб. На початку ми заархівували файли різного формату зі звичайним, швидким та максимальним режимами стиснення:

1. Формат .doc

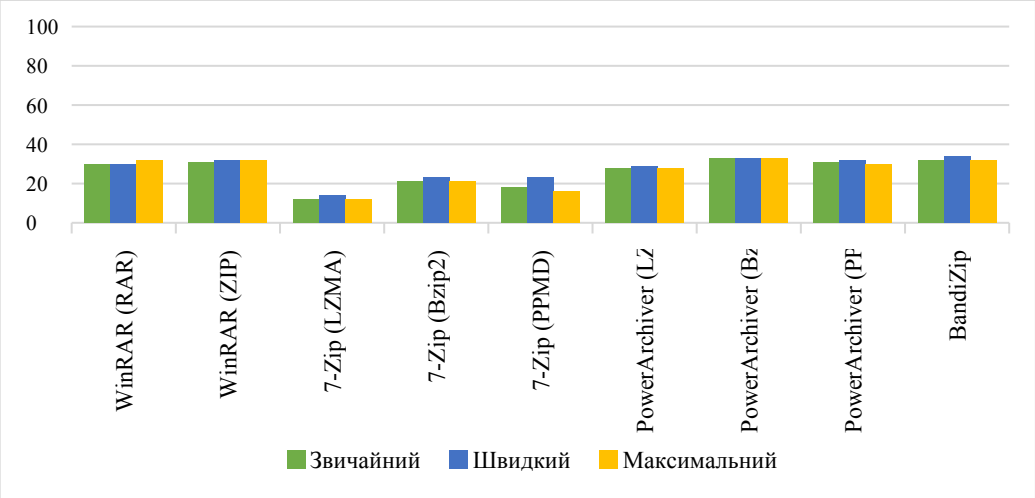


Рисунок 3.1 – Стиснення файлу .doc розміром 16,3 Мб

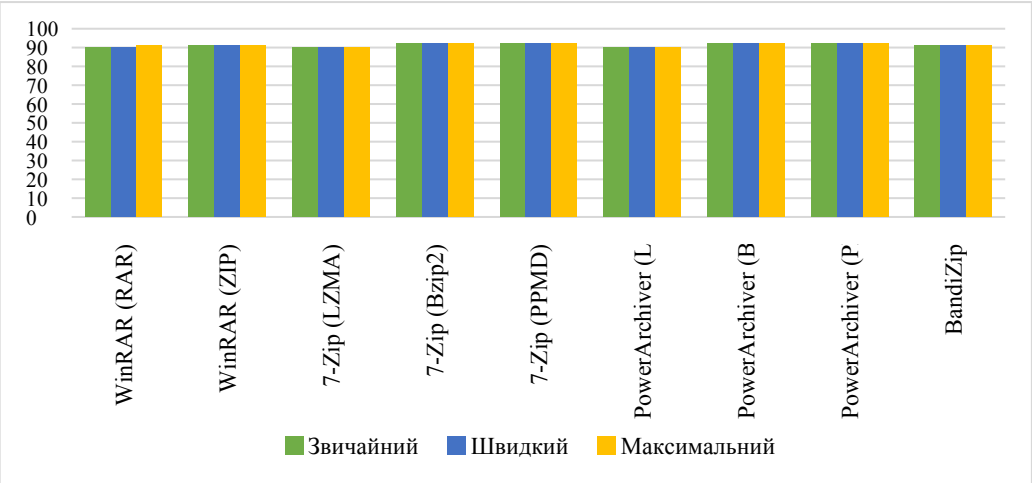


Рисунок 3.2 – Стиснення файлу .doc розміром 0,229 Мб

2. Формат .txt

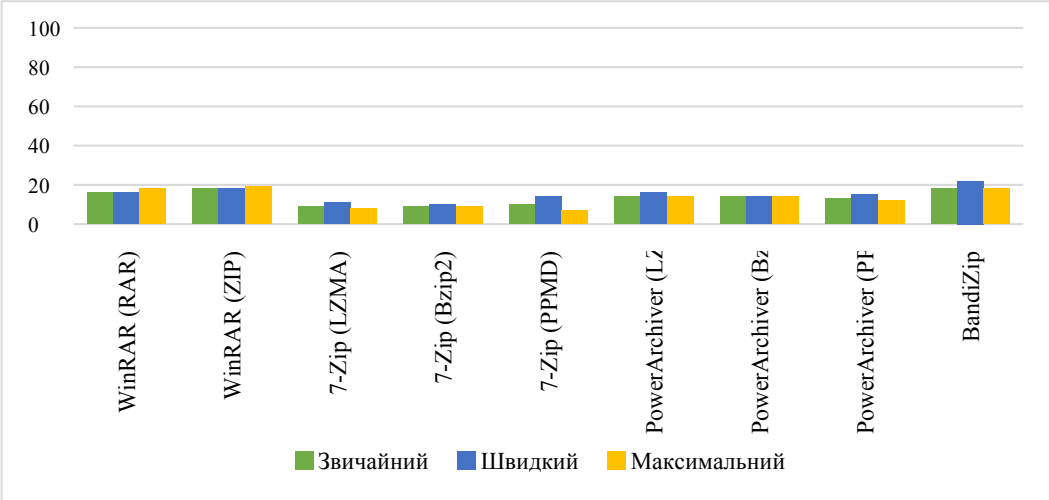


Рисунок 3.3 – Стиснення файлу .txt розміром 16,3 Мб

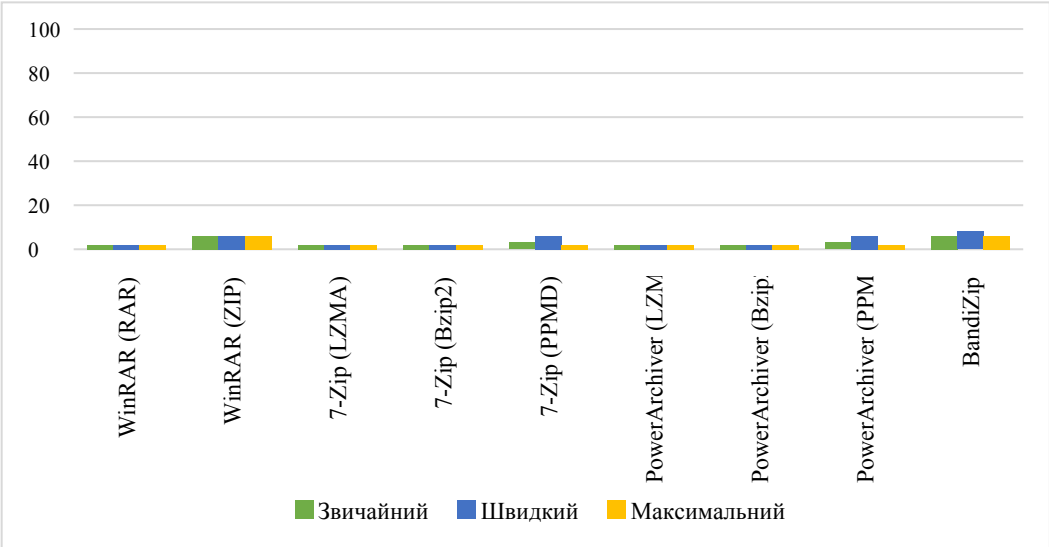


Рисунок 3.4 – Стиснення файлу .txt розміром 0,229 Мб

3. Формат .pdf

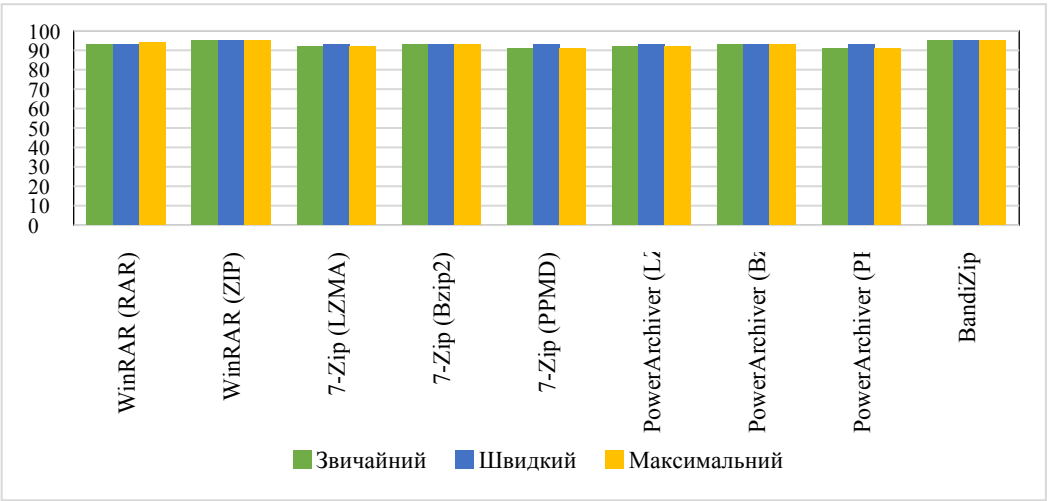


Рисунок 3.5 – Стиснення файлу .pdf розміром 16,3 Мб

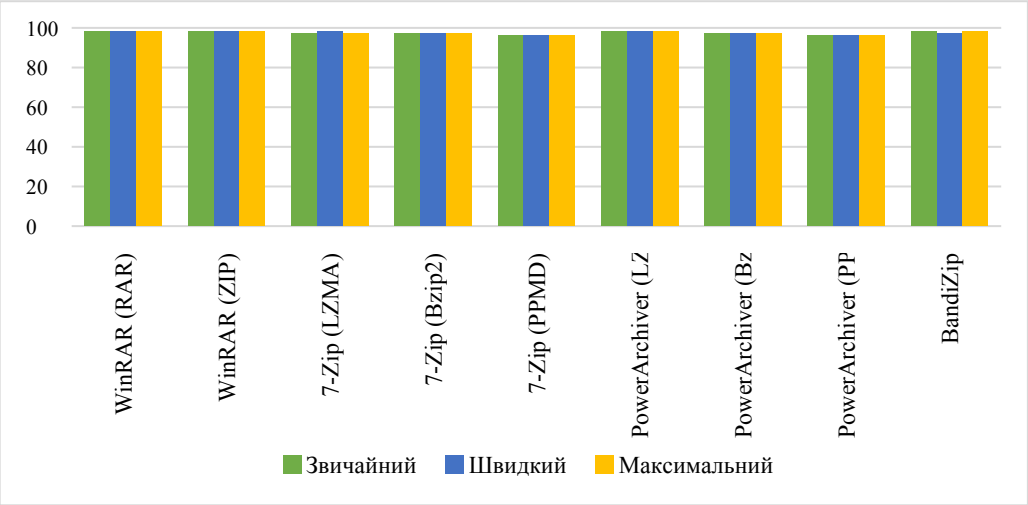


Рисунок 3.6 – Стиснення файлу .pdf розміром 0,229 Мб

4. Формат .png

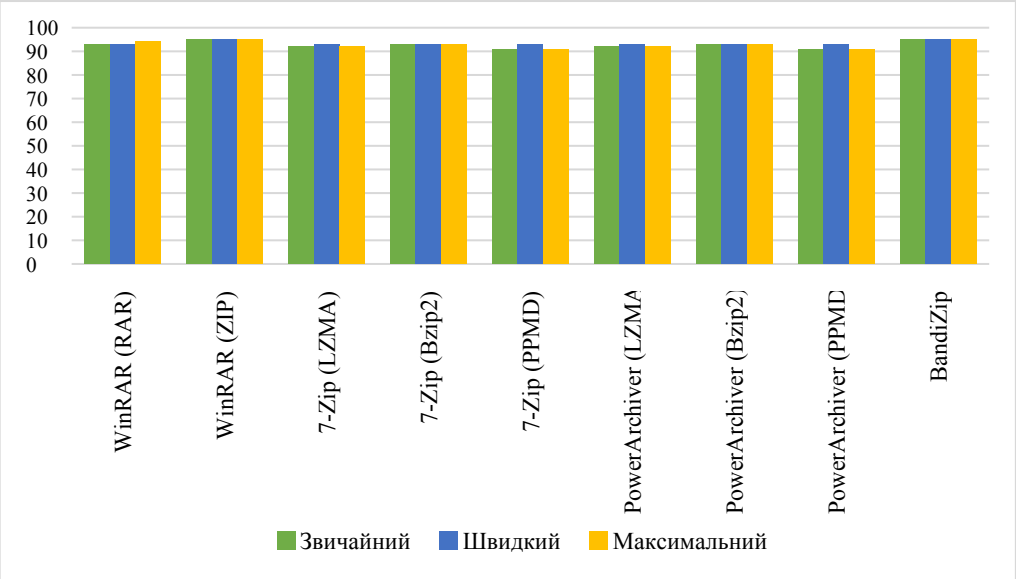


Рисунок 3.7 – Стиснення файлу .png розміром 16,3 Мб

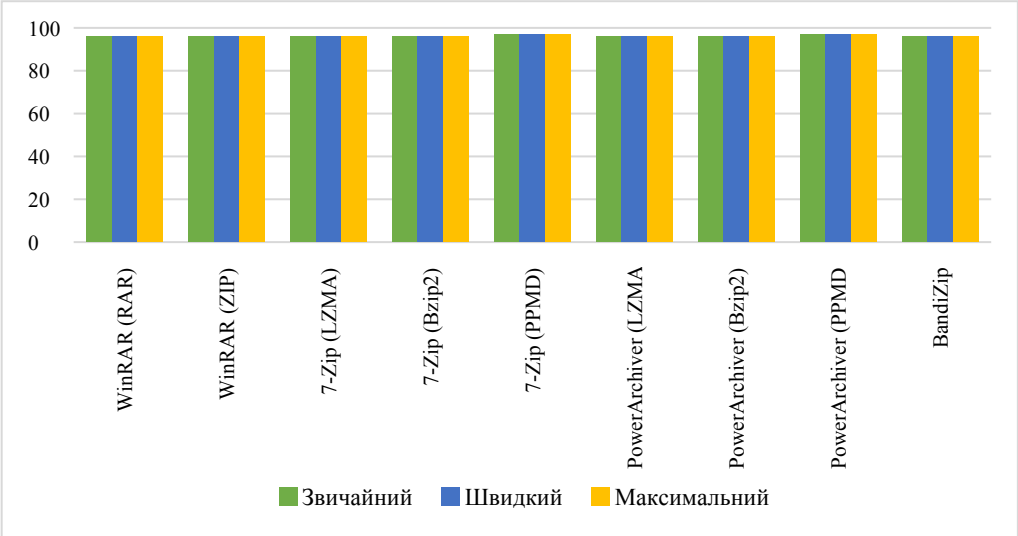


Рисунок 3.8 – Стиснення файлу .png розміром 0,229 Мб

5. Формат .mp3

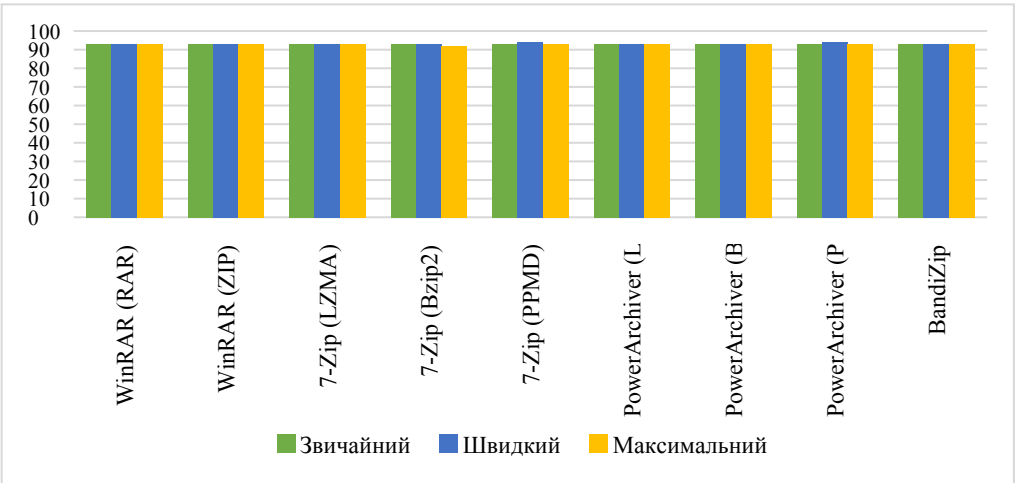


Рисунок 3.9 – Стиснення файлу .mp3 розміром 16,3 Мб

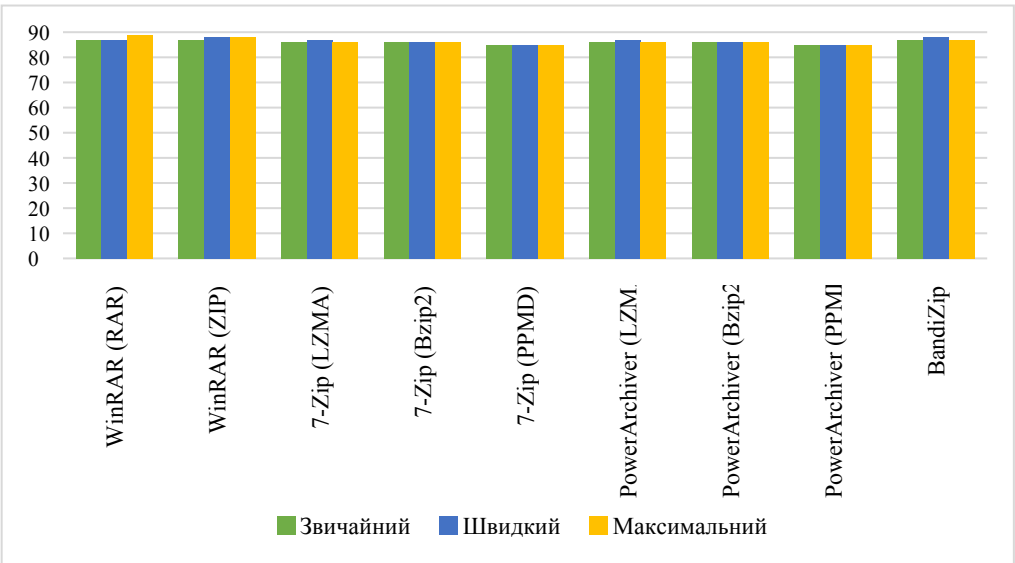


Рисунок 3.10 – Стиснення файлу .mp3 розміром 0,229 Мб

6. Формат .wav

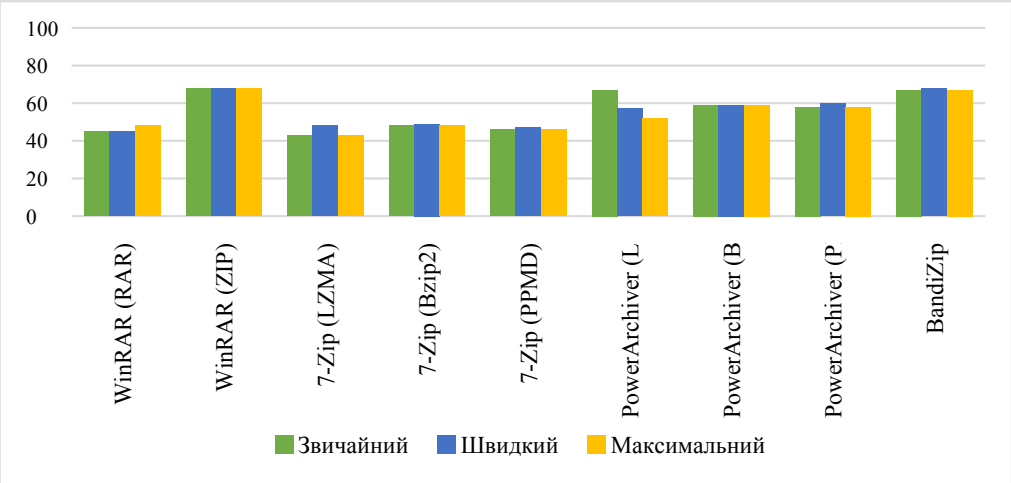


Рисунок 3.11 – Стиснення файлу .wav розміром 16,3 Мб

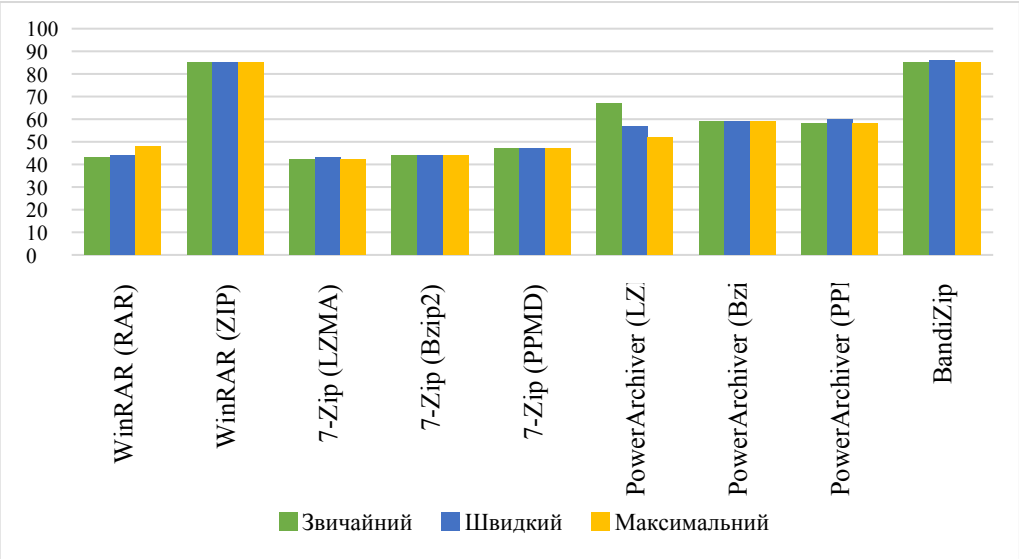


Рисунок 3.12 – Стиснення файлу .wav розміром 0,229 Мб

7. Формат .exe

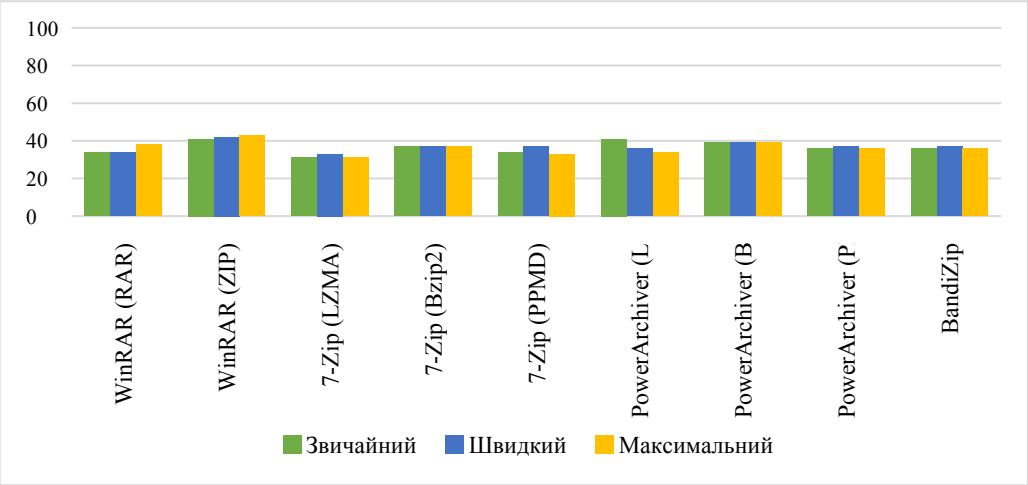


Рисунок 3.13 – Стиснення файлу .exe розміром 16,3 Мб

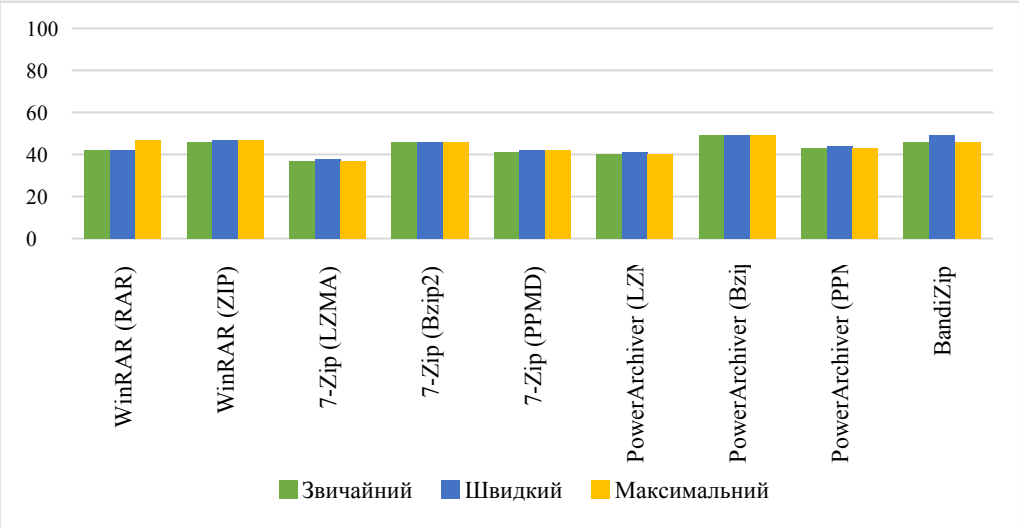


Рисунок 3.14 – Стиснення файлу .exe розміром 0,229 Мб

8. Формат .xls

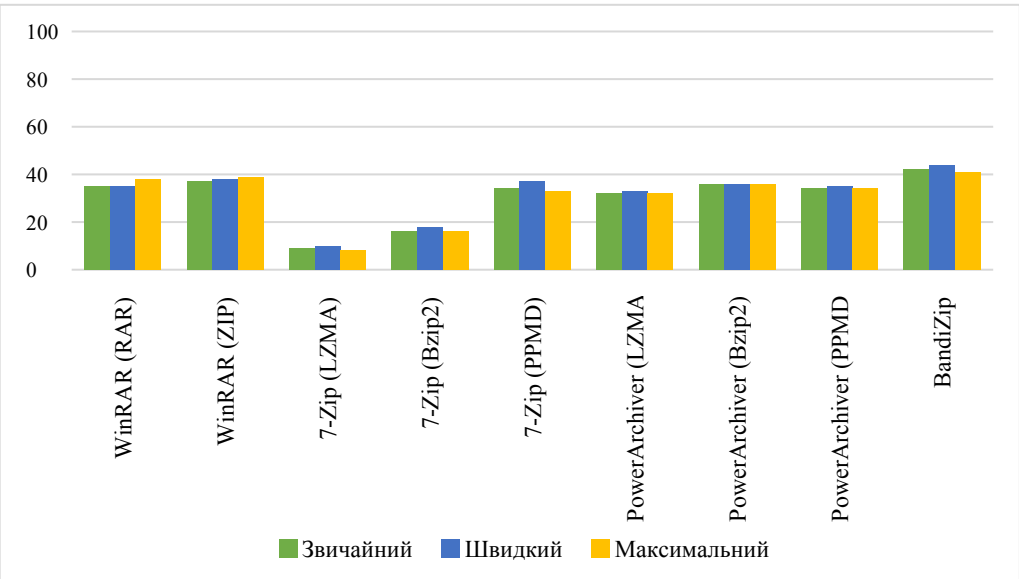


Рисунок 3.15 – Стиснення файлу .xls розміром 16,3 Мб

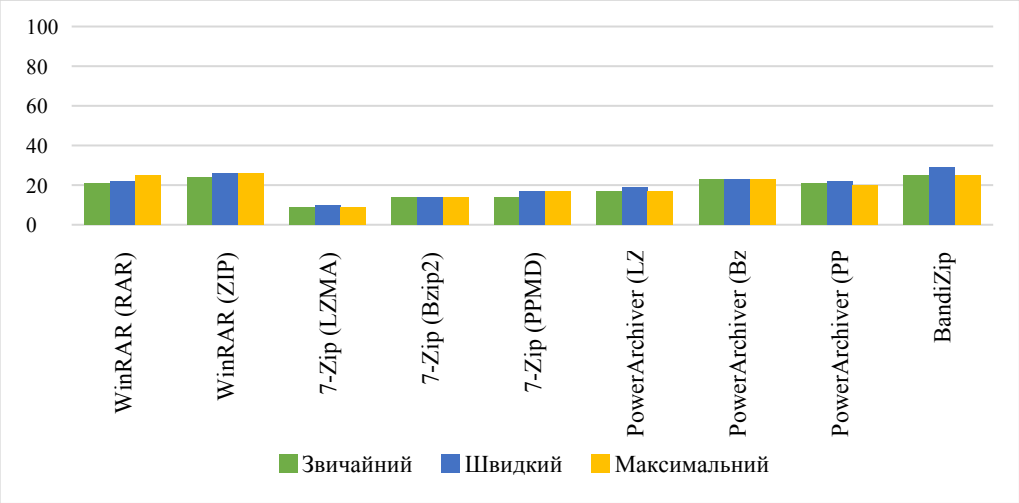


Рисунок 3.16 – Стиснення файлу .xls розміром 0,229 Мб

9. Формат .jpeg

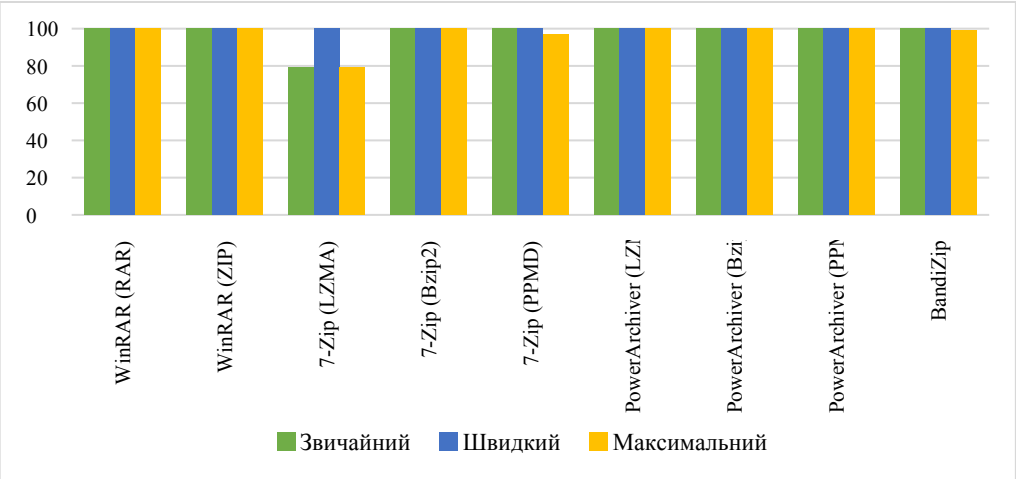


Рисунок 3.17 – Стиснення файлу .jpeg розміром 16,3 Мб

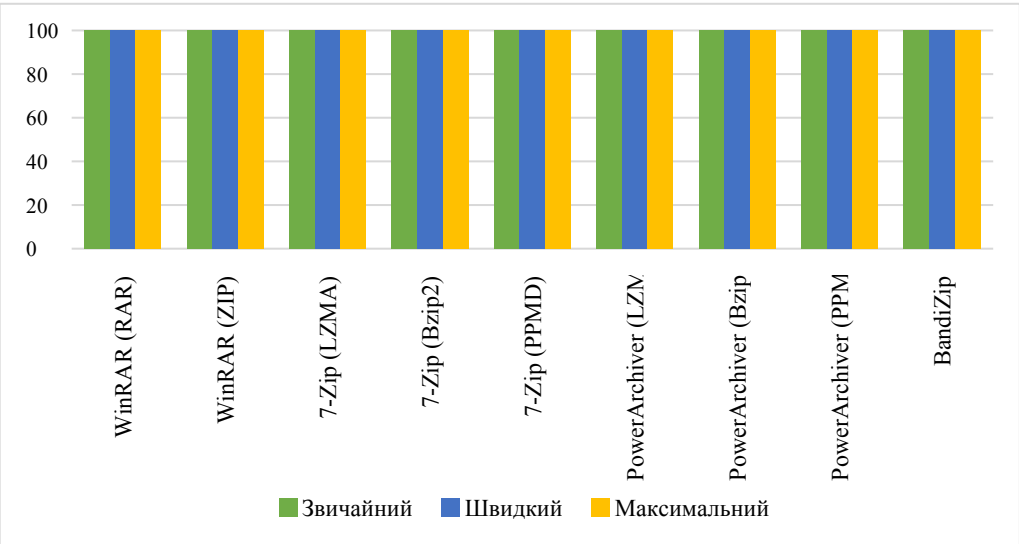


Рисунок 3.18 – Стиснення файлу .jpeg розміром 0,229 Мб

Отже з проведеного аналізу можна зробити висновок

1. WinRAR архіватор має можливість стискати інформацію у ZIP та у RAR архіви. Тому проводимо дослідження стиснення інформіції у архів ZIP та у архів RAR. Отриманні дані занесемо у табл. 1 та табл. 2 (Додаток Б).
2. У архіватора 7-Zip коефіцієнт компресії дуже залежить від даних використаних для випробувань. Зазвичай 7-Zip (у форматі 7z) стискує на 30 – 70% краще ніж у форматі zip. Також 7-Zip стискує у форматі ZIP на 2-10% краще чим більшість інших програм-архіваторів, що працюють з форматом ZIP. У табл. 3 (Додаток Б) описані результати стиснення папки із самою програмою 7-Zip за допомогою різних методів стиснення при звичайному, швидкому та максимальному режимі стиснення.
3. PowerArchiver – універсальний архіватор з підтримкою всіх популярних форматів. Більш докладно про стиснення даних за допомогою цього архіватора можна побачити

3.2 Порівняння за швидкістю стиснення

Наскільки ми знаємо, архіватори можуть добре стискати файли. Та як швидко це відбувається і від чого залежить, проаналізуємо наразі за швидкістю стиснення

1. Формат .doc

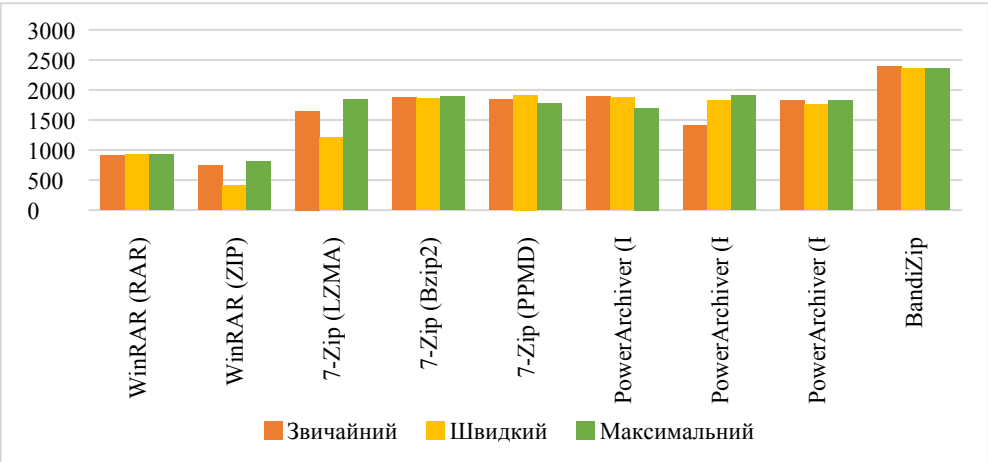


Рисунок 3.19 – Стиснення файлу .doc розміром 16,3 Мб

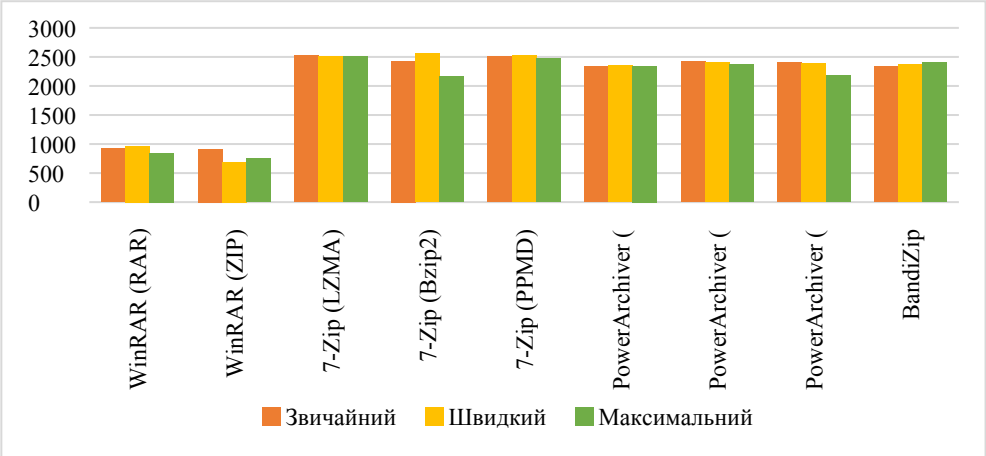


Рисунок 3.20 – Стиснення файлу .doc розміром 0,229 Мб

2. Формат .txt

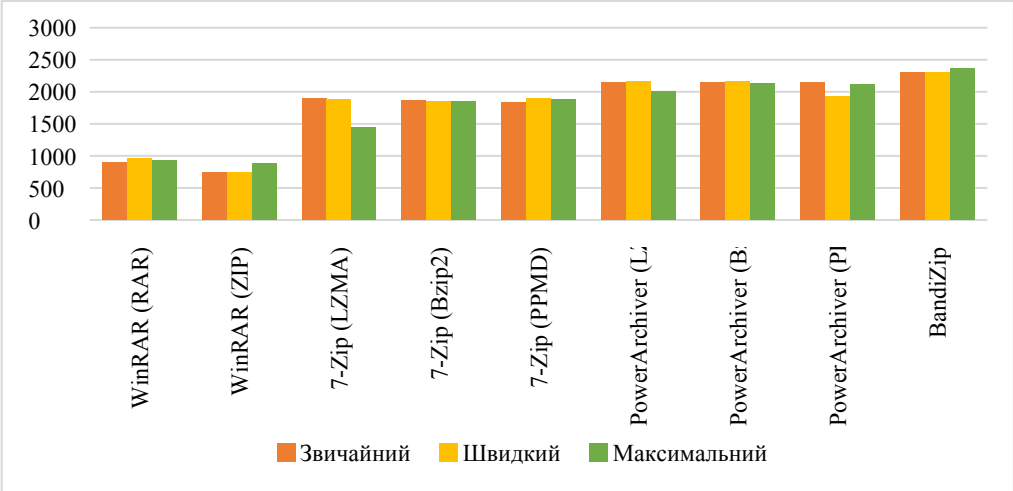


Рисунок 3.21 – Стиснення файлу .txt розміром 16,3 Мб

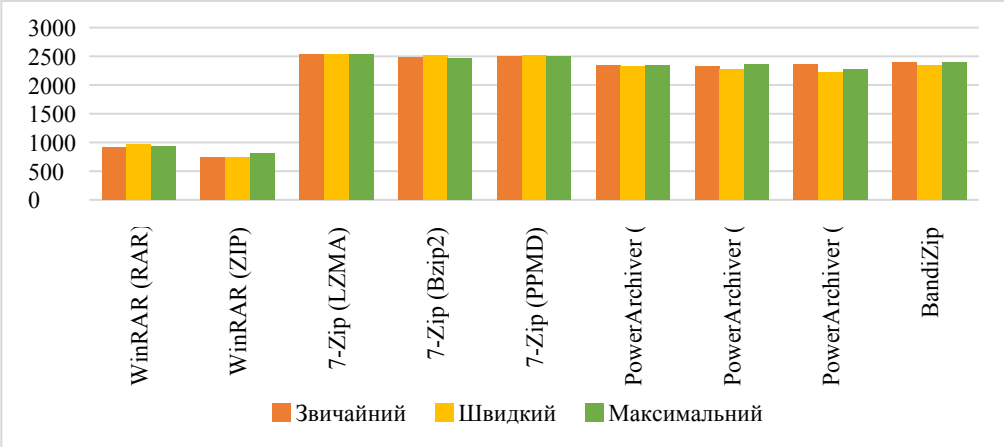


Рисунок 3.22 – Стиснення файлу .txt розміром 0,229 Мб

3. Формат .pdf

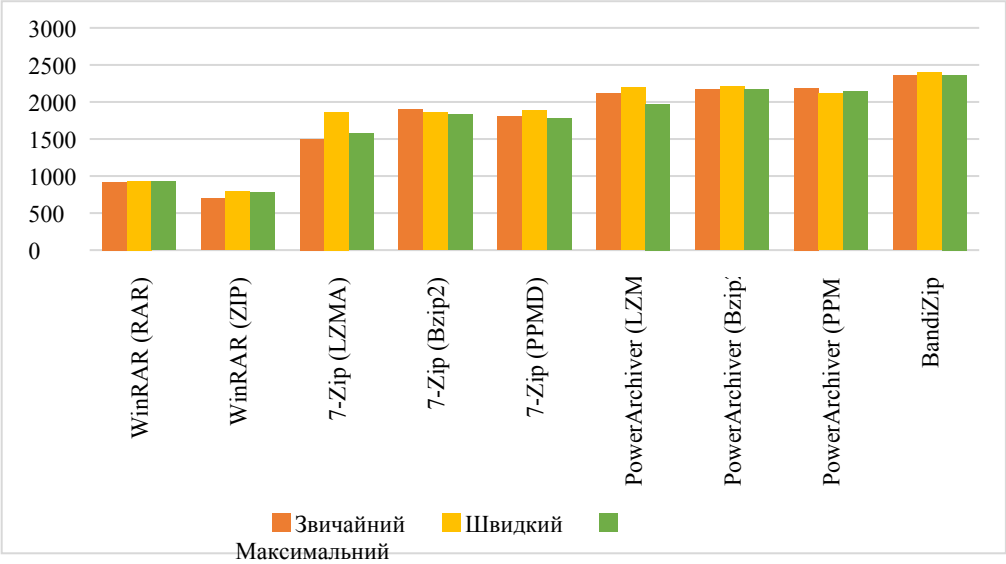


Рисунок 3.23 – Стиснення файлу .pdf розміром 16,3 Мб

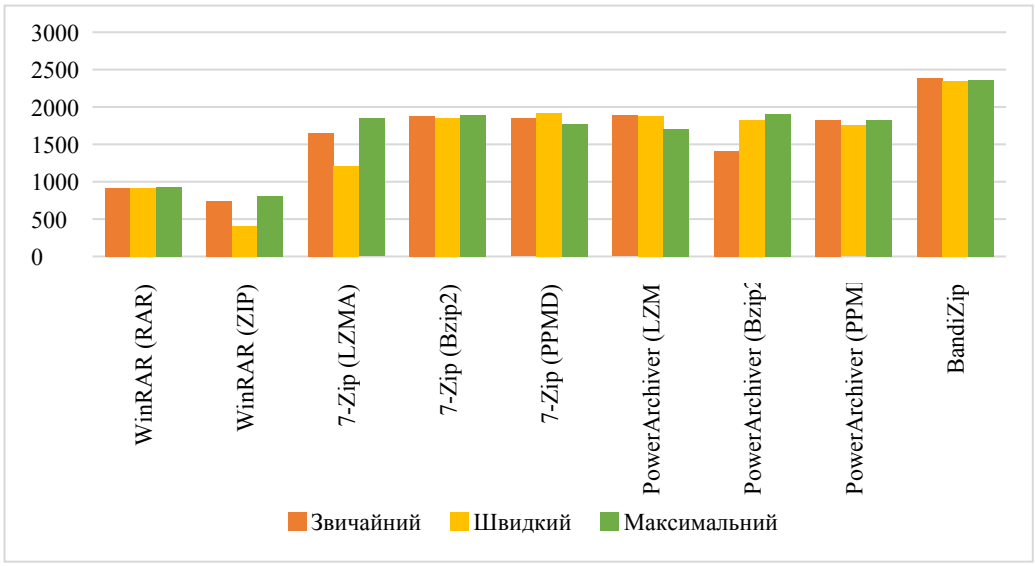


Рисунок 3.24 – Стиснення файлу .pdf розміром 0,229 Мб

4. Формат .png

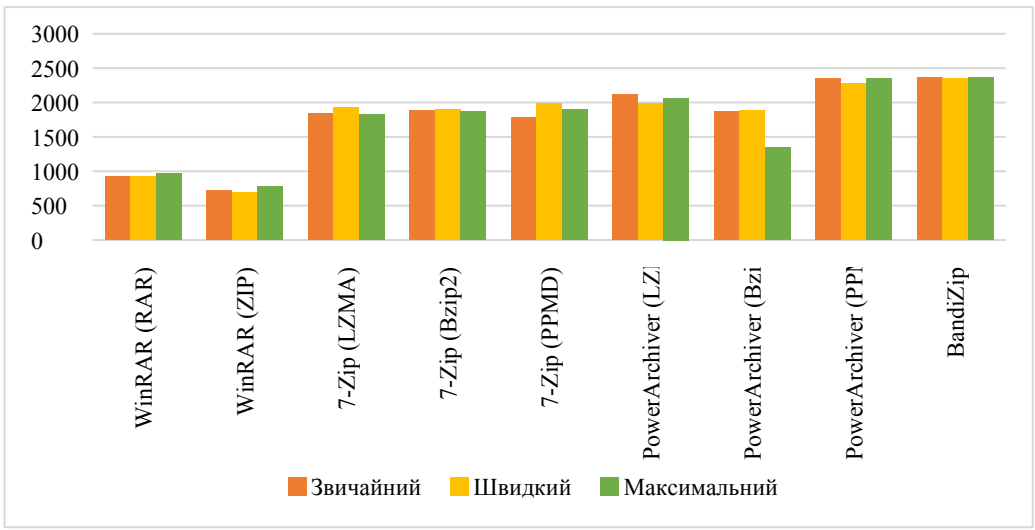


Рисунок 3.25 – Стиснення файлу .png розміром 16,3 Мб

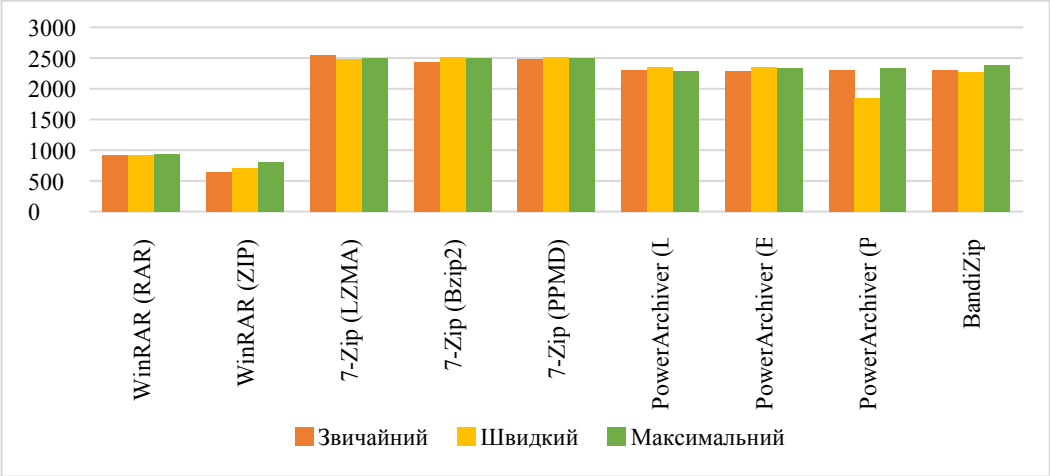


Рисунок 3.26 – Стиснення файлу .png розміром 0,229 Мб

5. Формат .mp3

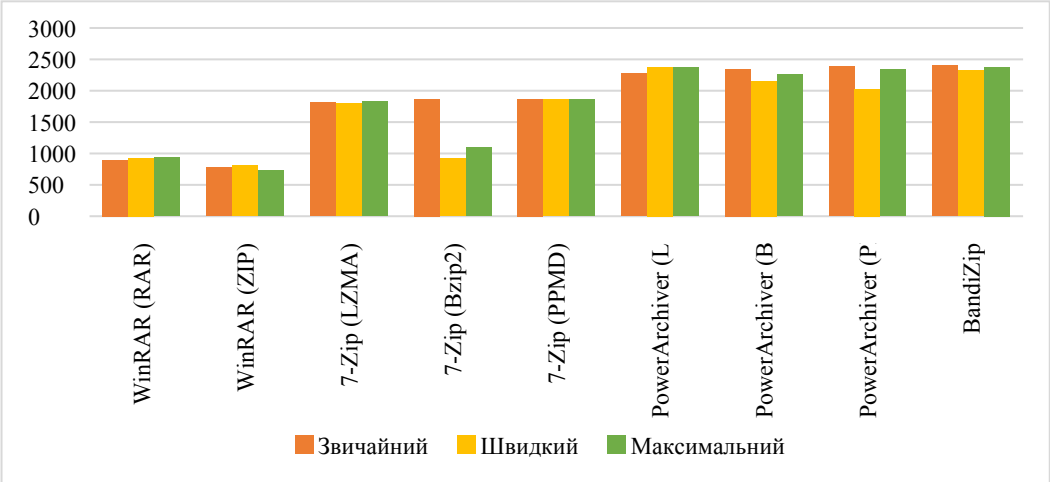


Рисунок 3.27 – Стиснення файлу .mp3 розміром 16,3 Мб

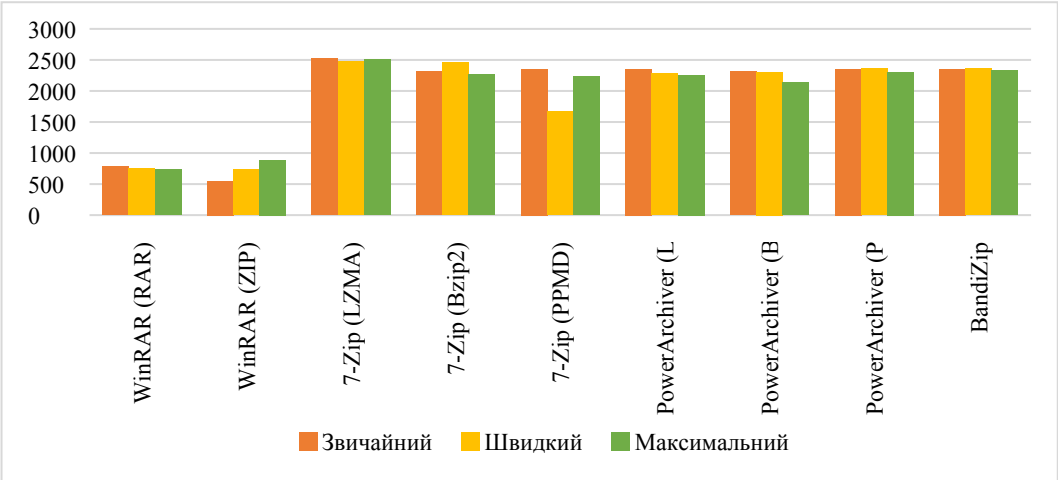


Рисунок 3.28 – Стиснення файлу .mp3 розміром 0,229 Мб

6. Формат .wav

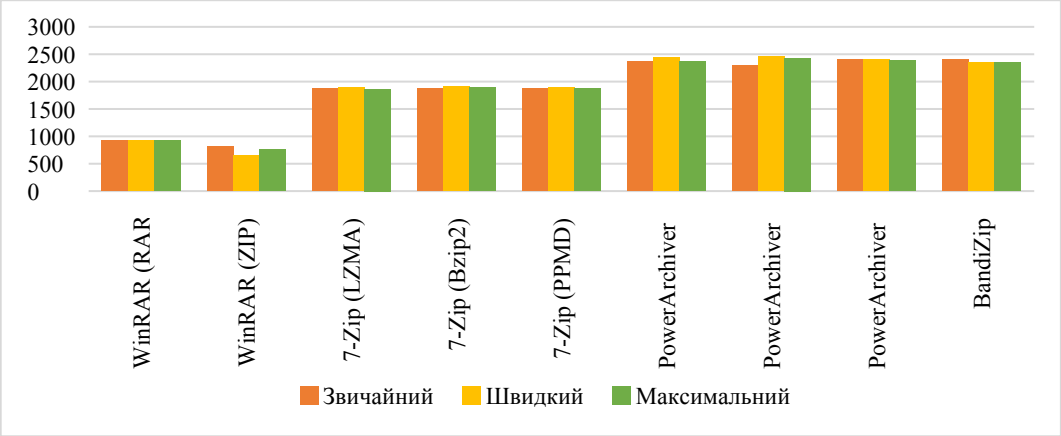


Рисунок 3.29 – Стиснення файлу .wav розміром 16,3 Мб

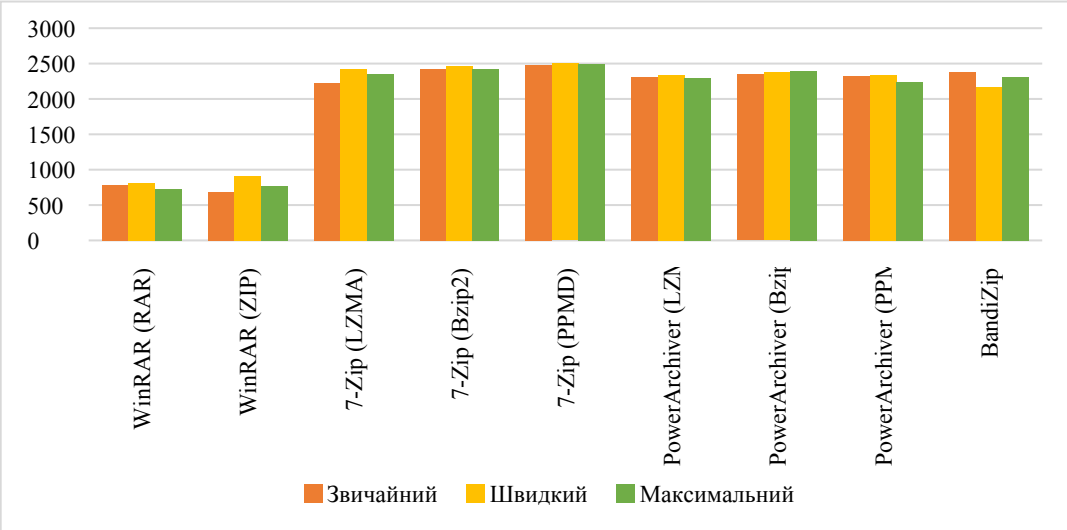


Рисунок 3.30 – Стиснення файлу .wav розміром 0,229 Мб

7. Формат .exe

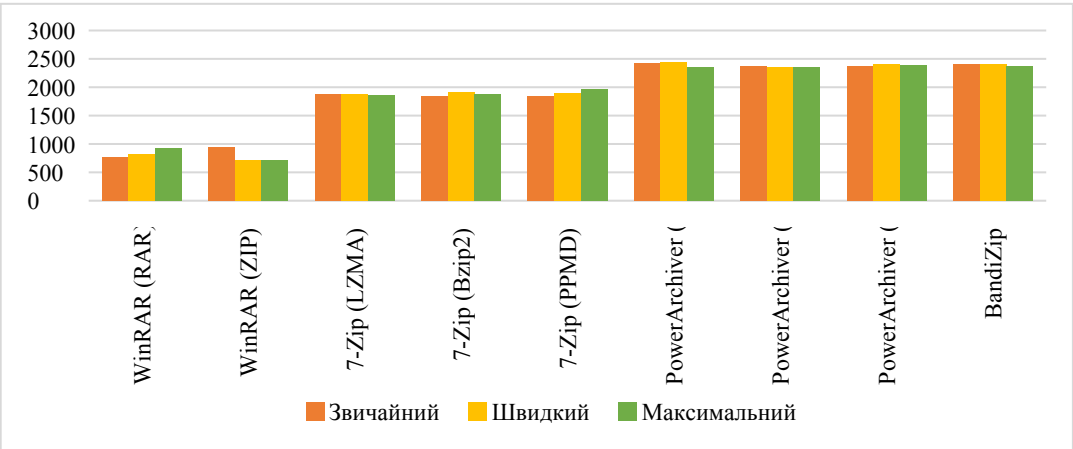


Рисунок 3.31 – Стиснення файлу .exe розміром 16,3 Мб

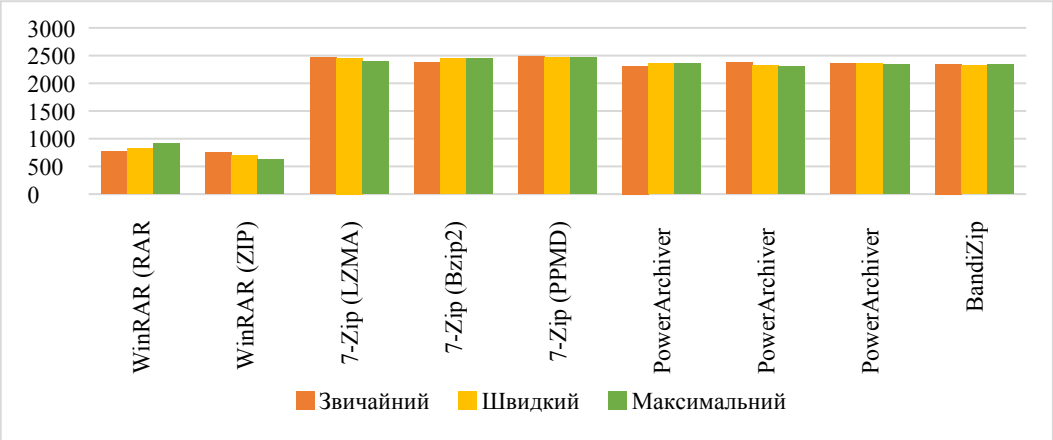


Рисунок 3.32 – Стиснення файлу .exe розміром 0,229 Мб

8. Формат .xls

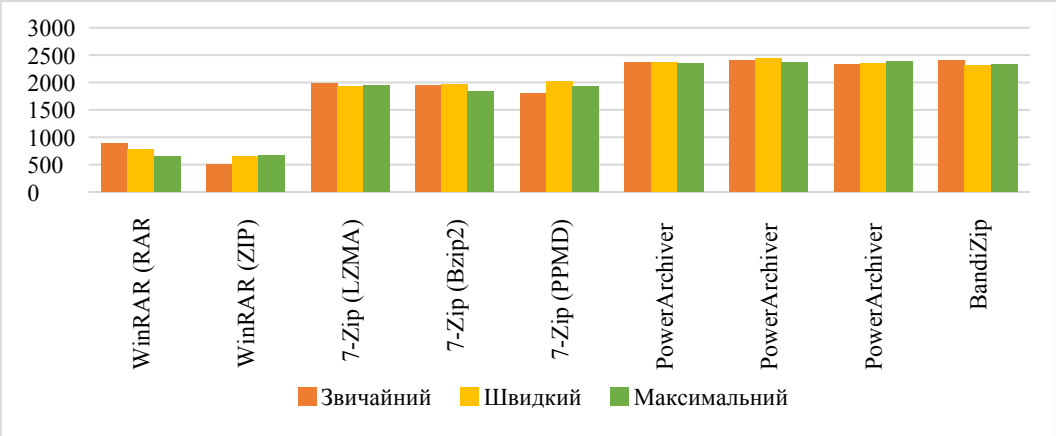


Рисунок 3.33 – Стиснення файлу .xls розміром 16,3 Мб

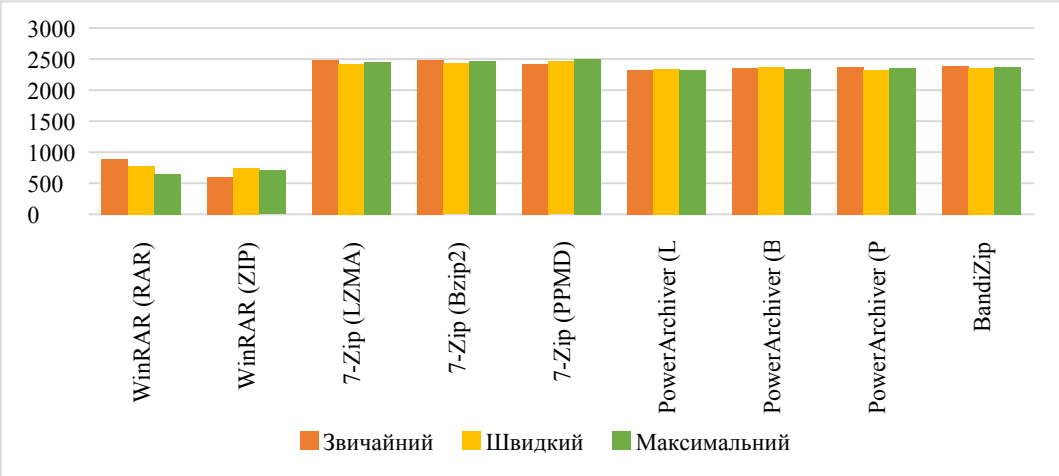


Рисунок 3.34 – Стиснення файлу .xls розміром 0,229 Мб

9. Формат .jpeg

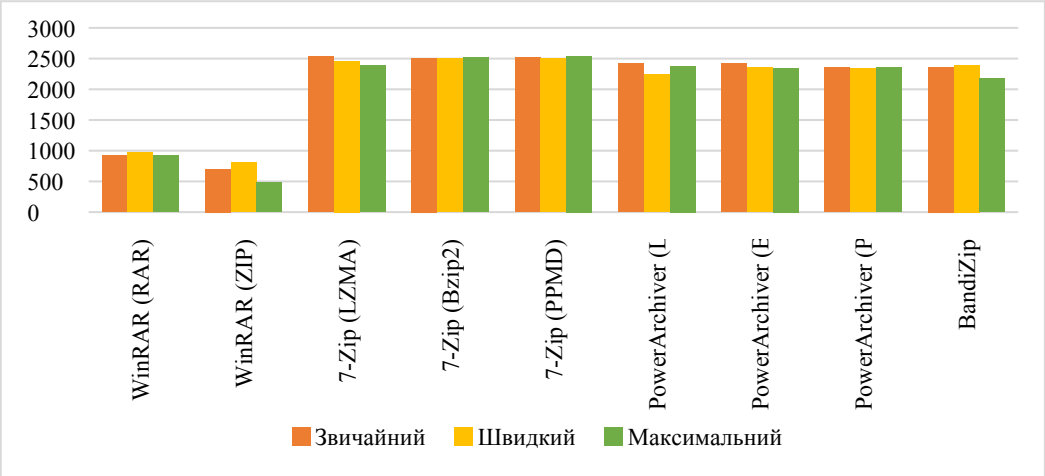


Рисунок 3.35 – Стиснення файлу .jpeg розміром 16,3 Мб

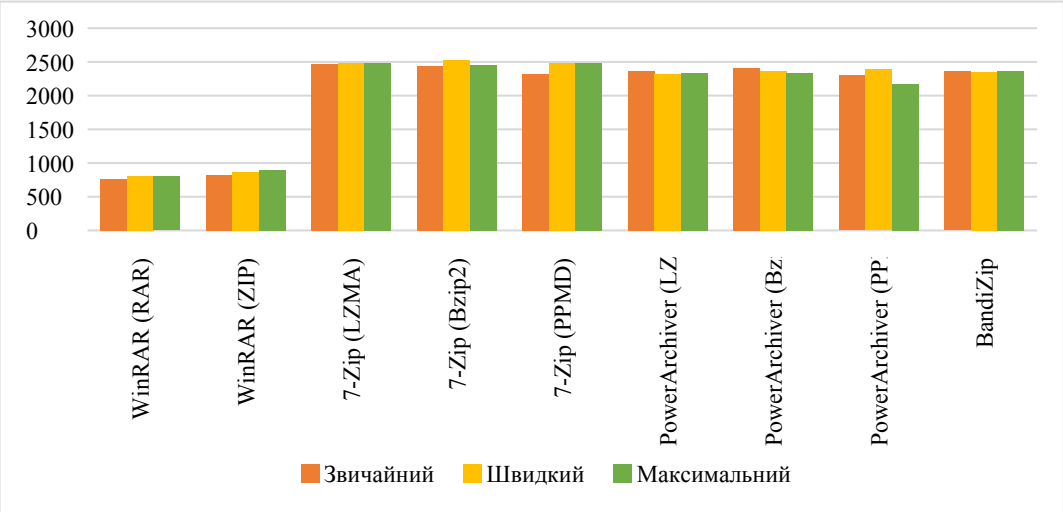


Рисунок 3.36 – Стиснення файлу .jpeg розміром 0,229 Мб

3.3 Висновки

Згідно проведених дослідів, найкращим з архіваторів показав себе 7zip. При дослідженні було виявлено, що один режим не дуже відрізняється від іншого, а метод стиснення потрібно обирати за параметром, який потребується найбільше: швидкість або ступінь стиснення файлу. Також при проведенні дослідження було виявлено що 7zip не підтримує стиснення таких форматів як: mp3, pdf, exe та jpeg.

Стиснення аудіоданих може зменшити пропускну здатність. Алгоритми стиснення звуку реалізовані в програмному забезпеченні як аудіокодеки. Як при стисненні з втратами, так і при втратах, надмірність інформації зменшується, використовуючи такі методи, як кодування, дискретне косинусне перетворення та лінійне передбачення, щоб зменшити обсяг інформації, що використовується для представлення нестиснутих даних.

Алгоритми стиснення з втратою звуку забезпечують більш високе стиснення і використовуються в численних програмах аудіо. Ці алгоритми майже всі покладаються на психоакустику для усунення або зменшення вірності менш чутних звуків, зменшуючи тим самим простір, необхідний для їх зберігання або передачі.

Стиснення звуку без втрат забезпечує представлення цифрових даних, які можна декодувати в точну цифрову копію оригіналу. Коефіцієнти стиснення складають приблизно 50–60% від початкового розміру, що аналогічно таким для загального стиснення даних без втрат. Коли аудіофайли підлягають обробці, або шляхом подальшого стиснення, або для редагування, бажано працювати з незмінним оригіналом (без стиснення або стиснення без втрат). Обробка з втратою стисненого файлу для певних цілей зазвичай дає кінцевий нижчий результат від створення того самого стисненого файлу з нестисненого оригіналу. На додаток до редагування чи змішування звуку, стиснення звуку без втрат часто використовується для архівного зберігання або як головних копій.

Ми виміряли ступінь стиснення, швидкість стиснення окремо проаналізували архіватори. Швидкість стиснення і декомпресії кожен алгоритм вимірювали за допомогою програми, яка називалась стисненням та процедури декомпресії кожного алгоритму із статично зв’язаних бібліотек. Ми обмежили вибір алгоритмів тими, які, як правило, мають вищий ступінь стиснення.

ВИСНОВКИ ТА РЕКОМЕНДАЦІЇ

Завжди існувала та існує на сьогоднішній день проблема з розміром при передачі та зберіганні інформації. На фоні розвитку комп’ютерних технологій можна подумати що використання ефективних методів стиснення не потрібне. Але разом із розвитком цих технологій збільшується і обсяг інформації для зберігання, передачі. Також, з погляду користувача за передачу інформації потрібно платити, а чим більше інформації тим більше грошей. Також фізичні носії не мають безкоштовних аналогів. Саме стиснення інформації покращить швидкість передачі інформації та допоможе заощадити на фізичних носіях.

Метою стиснення є зменшення кількості біт у порівнянні з джерелом, необхідних для зберігання або передачі заданої інформації. Для стиснення інформації часто використовується програми-архіватори даних такі як ZIP, RAR, PA, &ZIP та інші.

На даний час людям пропонують використовувати різні архіватори. Кожен архіватор має деякі переваги над іншими, але не існує архіватора який був би у всьому набагато кращим за своїх конкурентів.

Тому у даній роботі було проведено дослідження того, який коефіцієнт стиснення найкращий у кожного архіватора до певних типів файлів та який архіватор виграє по швидкості стиснення.

Роблячи прогнози, можна припустити, що програмне забезпечення для архівування буде розроблятися з використанням алгоритмів PAQ або LZMA, хоча не виключено, що відбудеться перехід на більш складний математичний алгоритм CSE.

Зараз ці алгоритми набирають популярності завдяки дуже хорошому коефіцієнту стиснення (хоча працюють доволі повільно). Але завдяки збільшенню швидкодії комп'ютера, швидкість роботи стає менш критичною.

ПЕРЕЛІК ПОСИЛАНЬ

1. Evaluating Lossless Data Compression Algorithms for Use on Mobile Devices [електроний ресурс] / Literature Synthesis Paul Brittan. Режим доступу:https://people.cs.uct.ac.za/~pbrittan/privacy_brittan_petzer/downloads/paul_litReview.pdf
2. Comparison of Brotli, Deflate, Zopfli, LZMA, LZHAM and Bzip2 Compression Algorithms [Електроний ресурс]/ http://www.gstatic.com/b/brotlidocs/brotli-2015-09-22.pdf
3. Офіційний сайт архіватора 7-Zip [електронний ресурс] / Режим доступу: https://7-zip.org.ua/
4. Жураковський Б.Ю. Способи стиснення даних при архівації / Сучасний захист інформації. – 2013. - №2. – С. 65-68.
5. Рак Т.Є., Смотр О.О., Зачко О.Б. Стиснення даних. Архіватори. Навч. пос. – Львів: ЛДУ БЖД, 2007. - 30 с.
6. Корпань Я. В. Методи та алгоритми компактного представлення графічної інформації в комп'ютерних системах / Технологічний аудит та резерви виробництва. – 2015. – № 3/2 (23). – С. 32–36.
7. WinRAR. Матеріал з Вікіпедії — вільної енциклопедії. [електроний ресурс] / Режим доступу: https://ru.wikipedia.org/wiki/WinRAR
8. Основні файлові формати [електронний ресурс] / Режим доступу: https://stud.com.ua/43312/informatika/osnovni_faylovi_formati
9. V.Larin. Investigation of the mechanism for processing predicted frames in the technology of compression of transformed images in computer systems and special purpose networks. / O.Tymochko, I. Sheviakov, A. Abdalla // Системи обробки інформації.– № 4. – Х. ХУПС. 2020. — С. 24 — 31.
10. Archiver comparison [Електроний ресурс] / Режим доступу: http://warp.povusers.org/ArchiverComparison/.
11. Comparison of compression/archive formats (7Z vs RAR vs ZIP) [Електроний ресурс] / Режим доступу: Comparison of compression/archive formats (7Z vs RAR vs ZIP).

Додаток А

ПЕРЕЛІК КОПІЙ ДЕМОНСТРАЦІЙНОГО МАТЕРІАЛУ

Додаток Б
ТАБЛИЦІ З РЕЗУЛЬТАТАМИ

Таблиця 1 – Приклади стиснення архіватором WinRAR у форматі RAR

Початковий файл	Початковий розмір, МБ	Режим стиснення	Розмір після стиснення, МБ	Ступінь стиснення, %	Швидкість стиснення ,Кбіт/с
.doc	16,3	Звичайний	5,059	30	914
		Швидкий	5,087	30	920
		Максимальний	5,502	32	927
.txt		Звичайний	2,776	16	901
		Швидкий	2,816	16	963
		Максимальний	3,101	18	924
.pdf		Звичайний	15,539	93	918
		Швидкий	15,554	93	934
		Максимальний	15,720	94	930
.png		Звичайний	15,934	94	928
		Швидкий	15,944	94	928
		Максимальний	15,974	95	972
.mp3		Звичайний	15,720	93	899
		Швидкий	15,730	93	929
		Максимальний	15,783	93	932
.wav	16,3	Звичайний	7,619	45	919
		Швидкий	7,677	45	925
		Максимальний	8,097	48	930

.exe		Звичайний	5,670	34	903
		Швидкий	5,744	34	919
		Максимальний	6,427	38	910
.xls		Звичайний	5,937	35	913
		Швидкий	6,048	35	943
		Максимальний	6,430	38	922
.jpeg		Звичайний	16,636	100	924
		Швидкий	16,636	100	971
		Максимальний	16,631	100	917
.doc	0,229	Звичайний	0,208	90	917
		Швидкий	0,208	90	968
		Максимальний	0,209	91	847
.txt		Звичайний	0,006	2	910
		Швидкий	0,006	2	959
		Максимальний	0,007	2	931
.pdf	0,229	Звичайний	0,226	98	908
		Швидкий	0,226	98	959
		Максимальний	0,227	98	955
.png		Звичайний	0,222	96	925
		Швидкий	0,222	96	920
		Максимальний	0,223	96	942
.mp3		Звичайний	0,202	87	782
		Швидкий	0,202	87	756

		Максимальний	0,207	89	738
.wav		Звичайний	0,101	43	779
		Швидкий	0,102	44	807
		Максимальний	0,111	48	724
.exe		Звичайний	0,098	42	764
		Швидкий	0,099	42	817
		Максимальний	0,111	47	918
.xls		Звичайний	0,051	21	892
		Швидкий	0,054	22	779
		Максимальний	0,059	25	652
.jpeg		Звичайний	0,229	100	763
		Швидкий	0,229	100	811
		Максимальний	0,229	100	801

Таблиця 2 – Приклади стиснення архіватором WinRAR у форматі ZIP

Початковий файл	Початковий розмір, МБ	Режим стиснення	Розмір після стиснення, МБ	Ступінь стиснення, %	Швидкість стиснення, Кбіт/с
.doc		Звичайний	5,337	31	742
		Швидкий	5,469	32	407
		Максимальний	5,528	32	810
.txt		Звичайний	3,108	18	739
		Швидкий	3,220	18	734
		Максимальний	3,329	19	882
.pdf		Звичайний	15,820	95	694

	16,3	Швидкий	15,831	95	796
		Максимальний	15,836	95	778
.png		Звичайний	15,926	94	727
		Швидкий	15,938	94	689
		Максимальний	15,948	94	772
.mp3		Звичайний	15,731	93	775
		Швидкий	15,739	93	812
		Максимальний	15,739	93	725
.wav		Звичайний	11,411	68	818
		Швидкий	11,454	68	654
		Максимальний	11,480	68	758
.exe		Звичайний	6,962	41	945
		Швидкий	7,099	42	714
		Максимальний	7,161	43	706
.xls		Звичайний	6,353	37	503
		Швидкий	6,552	38	650
		Максимальний	6,610	39	675
.jpeg		Звичайний	16,607	100	702
		Швидкий	16,608	100	805
		Максимальний	16,608	100	480
.doc	0,229	Звичайний	0,210	91	913
		Швидкий	0,210	91	692
		Максимальний	0,210	91	748

.txt		Звичайний	0,015	6	733
		Швидкий	0,016	6	746
		Максимальний	0,016	6	805
.pdf		Звичайний	0,226	98	789
		Швидкий	0,226	98	789
		Максимальний	0,226	98	870
.png		Звичайний	0,222	96	637
		Швидкий	0,222	96	703
		Максимальний	0,222	96	802
.mp3		Звичайний	0,203	87	551
		Швидкий	0,204	88	733
		Максимальний	0,204	88	888
.wav		Звичайний	0,196	85	679
		Швидкий	0,196	85	906
		Максимальний	0,196	85	773
.exe		Звичайний	0,107	46	753
		Швидкий	0,110	47	698
		Максимальний	0,110	47	627
.xls		Звичайний	0,058	24	601
		Швидкий	0,062	26	748
		Максимальний	0,063	26	707
.jpeg		Звичайний	0,229	100	824
		Швидкий	0,229	100	861

		Максимальний	0,229	100	899

Таблиця 3 – Приклади стиснення архіватором 7-Zip різними методами

Початковий файл	Початковий розмір, МБ	Режим стиснення	Метод стиснення	Розмір після стиснення, МБ	Ступінь стиснення, %	Швидкість стиснення, Кбіт/с
.doc	16,3	Максимальний	LZMA	2,094	12	1846
		Звичайний		2,110	12	1648
		Швидкий		2,491	14	1212
		Максимальний	Bzip2	3,526	21	1896
		Звичайний		3,539	21	1874
		Швидкий		3,867	23	1854
		Максимальний	PPMD	2,823	16	1771
		Звичайний		3,101	18	1846
		Швидкий		3,938	23	1918
.txt	16,3	Максимальний	LZMA	1,438	8	1444
		Звичайний		1,562	9	1901
		Швидкий		1,988	11	1879
		Максимальний	Bzip2	1,646	9	1851
		Звичайний		1,647	9	1871
		Швидкий		1,764	10	1849
		Максимальний	PPMD	1,317	7	1887
		Звичайний		1,799	10	1838
		Швидкий		2,406	14	1899
.pdf	16,3	Максимальний	LZMA	15,453	92	1572

		Звичайний		15,456	92	1499
		Швидкий		15,588	93	1866
		Максимальний	Bzip2	15,506	93	1833
		Звичайний		15,514	93	1898
		Швидкий		15,586	93	1859
		Максимальний	PPMD	15,170	91	1779
		Звичайний		15,293	91	1803
		Швидкий		15,603	93	1889
.png		Максимальний	LZMA	15,891	94	1824
		Звичайний		15,894	94	1839
		Швидкий		15,986	95	1931
		Максимальний	Bzip2	15,900	94	1874
		Звичайний		15,931	94	1891
		Швидкий		15,937	94	1899
		Максимальний	PPMD	15,957	95	1894
		Звичайний		16,025	95	1786
		Швидкий		16,142	96	1986
.mp3		Максимальний	LZMA	15,757	93	1827
		Звичайний		15,763	93	1813
		Швидкий		15,827	93	1800
		Максимальний	Bzip2	15,701	92	1093
		Звичайний		15,716	93	1867
		Швидкий		15,730	93	919

		Максимальний	PPMD	15,746	93	1861
		Звичайний		15,847	93	1869
		Швидкий		16,000	94	1864
.wav	16,3	Максимальний	LZMA	7,281	43	1866
		Звичайний		7,296	43	1874
		Швидкий		8,079	48	1889
		Максимальний	Bzip2	8,080	48	1887
		Звичайний		8,095	48	1869
		Швидкий		8,223	49	1906
		Максимальний	PPMD	7,754	46	1877
		Звичайний		7,870	46	1872
		Швидкий		8,029	47	1891
.exe	16,3	Максимальний	LZMA	5,185	31	1862
		Звичайний		5,203	31	1869
		Швидкий		5,638	33	1877
		Максимальний	Bzip2	6,160	37	1877
		Звичайний		6,225	37	1844
		Швидкий		6,206	37	1915
		Максимальний	PPMD	5,624	33	1961
		Звичайний		5,724	34	1844
		Швидкий		6,206	37	1903
.xls	16,3	Максимальний	LZMA	1,502	8	1945
		Звичайний		1.512	9	1982

		Швидкий		1,790	10	1931
		Максимальний	Bzip2	2,767	16	1830
		Звичайний		2,771	16	1940
		Швидкий		3,088	18	1956
		Максимальний	PPMD	5,624	33	1927
		Звичайний		5,724	34	1800
		Швидкий		6,206	37	2018
.jpeg	16,3	Максимальний	LZMA	13,333	79	2385
		Звичайний		13,334	79	2540
		Швидкий		16,711	100	2460
		Максимальний	Bzip2	16,520	100	2516
		Звичайний		16,551	100	2510
		Швидкий		16,711	100	2501
		Максимальний	PPMD	16,276	97	2543
		Звичайний		16,508	100	2513
		Швидкий		15,655	100	2507
.doc	0,229	Максимальний	LZMA	0,207	90	2510
		Звичайний		0,207	90	2525
		Швидкий		0,207	90	2504
		Максимальний	Bzip2	0,212	92	2161
		Звичайний		0,213	92	2432
		Швидкий		0,213	92	2559
		Максимальний	PPMD	0,213	92	2477

		Звичайний		0,213	92	2501
		Швидкий		0,213	92	2528
.txt		Максимальний	LZMA	0,005	2	2537
		Звичайний		0,005	2	2534
		Швидкий		0,006	2	2531
		Максимальний	Bzip2	0,006	2	2463
		Звичайний		0,006	2	2489
		Швидкий		0,006	2	2510
		Максимальний	PPMD	0,005	2	2498
		Звичайний		0,009	3	2495
		Швидкий		0,016	6	2516
.pdf	0,229	Максимальний	LZMA	0,225	97	2498
		Звичайний		0,225	97	2483
		Швидкий		0,226	98	2460
		Максимальний	Bzip2	0,223	97	2412
		Звичайний		0,223	97	2507
		Швидкий		0,223	97	2480
		Максимальний	PPMD	0,221	96	2510
		Звичайний		0,221	96	2543
		Швидкий		0,221	96	2540
.png		Максимальний	LZMA	0,222	96	2507
		Звичайний		0,222	96	2553
		Швидкий		0,222	96	2489

		Максимальний	Bzip2	0,223	96	2507
		Звичайний		0,223	96	2534
		Швидкий		0,223	96	2516
		Максимальний	PPMD	0,225	97	2501
		Звичайний		0,225	97	2477
		Швидкий		0,225	97	2510
.mp3	0,229	Максимальний	LZMA	0,199	86	2510
		Звичайний		0,199	86	2525
		Швидкий		0,201	87	2480
		Максимальний	Bzip2	0,199	86	2266
		Звичайний		0,199	86	2309
		Швидкий		0,199	86	2454
		Максимальний	PPMD	0,197	85	2239
		Звичайний		0,197	85	2353
		Швидкий		0,197	85	1664
.wav	0,229	Максимальний	LZMA	0,098	42	2353
		Звичайний		0,098	42	2228
		Швидкий		0,113	49	2426
		Максимальний	Bzip2	0,103	44	2463
		Звичайний		0,103	44	2420
		Швидкий		0,103	44	2469
		Максимальний	PPMD	0,109	47	2489
		Звичайний		0,109	47	2475

		Швидкий		0,109	47	2501
.exe		Максимальний	LZMA	0,086	37	2395
		Звичайний		0,086	37	2469
		Швидкий		0,090	38	2457
		Максимальний	Bzip2	0,107	46	2454
		Звичайний		0,107	46	2379
		Швидкий		0,107	46	2451
		Максимальний	PPMD	0,098	42	2469
		Звичайний		0,095	41	2477
		Швидкий		0,098	42	2460
.xls		Максимальний	LZMA	0,022	9	2451
		Звичайний		0,022	9	2483
		Швидкий		0,025	10	2418
		Максимальний	Bzip2	0,034	14	2472
		Звичайний		0,034	14	2489
		Швидкий		0,034	14	2426
		Максимальний	PPMD	0,041	17	2501
		Звичайний		0,035	14	2418
		Швидкий		0,042	17	2460
.jpeg	0,229	Максимальний	LZMA	0,230	100	2489
		Звичайний		0,230	100	2463
		Швидкий		0,230	100	2480
		Максимальний	Bzip2	0,230	100	2449

		Звичайний		0,230	100	2434
		Швидкий		0,230	100	2528
		Максимальний	PPMD	0,231	100	2483
		Звичайний		0,231	100	2326
		Швидкий		0,231	100	2486

Таблиця 4 – Приклади стиснення архіватором PowerArchiver різними методами

Початковий файл	Початковий розмір, МБ	Режим стиснення	Метод стиснення	Розмір після стиснення, МБ	Ступінь стиснення, %	Швидкість стиснення, Кбіт/с
.doc	16,3	Максимальний	LZMA	4,768	28	1701
		Звичайний		4,768	28	1886
		Швидкий		4,985	29	1879
		Максимальний	Bzip2	5,623	33	1911
		Звичайний		5,623	33	1405
		Швидкий		5,623	33	1820
		Максимальний	PPMD	5,192	30	1820
		Звичайний		5,301	31	1830
		Швидкий		5,489	32	1752
.txt		Максимальний	LZMA	2,550	14	2005
		Звичайний		2,550	14	2141
		Швидкий		2,825	16	2161
		Максимальний	Bzip2	2,466	14	2131
		Звичайний		2,466	14	2152
		Швидкий		2,466	14	2161

		Максимальний	PPMD	2,142	12	2113
		Звичайний		2,303	13	2139
		Швидкий		2,707	15	1931
.pdf	16,3	Максимальний	LZMA	15,451	92	1974
		Звичайний		15,451	92	2113
		Швидкий		15,588	93	2193
		Максимальний	Bzip2	15,515	93	2172
		Звичайний		15,515	93	2170
		Швидкий		15,515	93	2209
		Максимальний	PPMD	15,191	91	2144
		Звичайний		15,241	91	2188
		Швидкий		15,609	93	2111
.png	16,3	Максимальний	LZMA	15,973	95	2066
		Звичайний		15,973	95	2120
		Швидкий		16,049	95	1991
		Максимальний	Bzip2	15,951	94	1343
		Звичайний		15,951	94	1871
		Швидкий		15,951	94	1889
		Максимальний	PPMD	16,074	95	2353
		Звичайний		16,077	95	2350
		Швидкий		16,155	96	2281
.mp3	16,3	Максимальний	LZMA	15,756	93	2371
		Звичайний		15,756	93	2276

		Швидкий		15,827	93	2376
		Максимальний	Bzip2	15,717	93	2261
		Звичайний		15,717	93	2345
		Швидкий		15,717	93	2157
		Максимальний	PPMD	15,781	93	2342
		Звичайний		15,825	93	2382
		Швидкий		16,003	94	2028
.wav	16,3	Максимальний	LZMA	8,854	52	2374
		Звичайний		11,336	67	2363
		Швидкий		9,705	57	2446
		Максимальний	Bzip2	10	59	2432
		Звичайний		10	59	2291
		Швидкий		10	59	2457
		Максимальний	PPMD	9,793	58	2382
		Звичайний		9,798	58	2398
		Швидкий		10,112	60	2407
.exe	16,3	Максимальний	LZMA	5,661	34	2350
		Звичайний		6,966	41	2417
		Швидкий		5,992	36	2437
		Максимальний	Bzip2	6,553	39	2347
		Звичайний		6,553	39	2363
		Швидкий		6,553	39	2353
		Максимальний	PPMD	6,015	36	2387

		Звичайний		6,038	36	2363
		Швидкий		6,238	37	2406
.xls		Максимальний	LZMA	5,430	32	2350
		Звичайний		5,430	32	2358
		Швидкий		5,725	33	2358
		Максимальний	Bzip2	6,218	36	2368
		Звичайний		6,218	36	2401
		Швидкий		6,218	36	2446
		Максимальний	PPMD	5,818	34	2379
		Звичайний		5,882	34	2332
		Швидкий		6,014	35	2350
.jpeg	16,3	Максимальний	LZMA	16,680	100	2379
		Звичайний		16,680	100	2426
		Швидкий		16,713	100	2244
		Максимальний	Bzip2	16,524	100	2345
		Звичайний		16,524	100	2418
		Швидкий		16,524	100	2358
		Максимальний	PPMD	16.442	100	2355
		Звичайний		16,480	100	2353
		Швидкий		16,637	100	2342
.doc	0,229	Максимальний	LZMA	0,207	90	2342
		Звичайний		0,207	90	2329
		Швидкий		0,207	90	2350

		Максимальний	Bzip2	0,213	92	2366
		Звичайний		0,213	92	2423
		Швидкий		0,213	92	2409
		Максимальний	PPMD	0,213	92	2184
		Звичайний		0,213	92	2406
		Швидкий		0,213	92	2382
.txt	0,229	Максимальний	LZMA	0,005	2	2350
		Звичайний		0,005	2	2347
		Швидкий		0,006	2	2319
		Максимальний	Bzip2	0,006	2	2353
		Звичайний		0,006	2	2321
		Швидкий		0,006	2	2279
		Максимальний	PPMD	0,005	2	2281
		Звичайний		0,008	3	2360
		Швидкий		0,016	6	2228
.pdf	0,229	Максимальний	LZMA	0,225	98	2337
		Звичайний		0,226	98	2345
		Швидкий		0,226	98	2259
		Максимальний	Bzip2	0,223	97	2309
		Звичайний		0,223	97	2347
		Швидкий		0,223	97	2326
		Максимальний	PPMD	0,221	96	2353
		Звичайний		0,221	96	2319

		Швидкий		0,221	96	2319
.png	0,229	Максимальний	LZMA	0,222	96	2286
		Звичайний		0,222	96	2304
		Швидкий		0,222	96	2350
		Максимальний	Bzip2	0,223	96	2342
		Звичайний		0,223	96	2281
		Швидкий		0,223	96	2345
		Максимальний	PPMD	0,225	97	2334
		Звичайний		0,225	97	2309
		Швидкий		0,225	97	1846
.mp3	0,229	Максимальний	LZMA	0,199	86	2249
		Звичайний		0,199	86	2353
		Швидкий		0,201	87	2276
		Максимальний	Bzip2	0,200	86	2139
		Звичайний		0,200	86	2316
		Швидкий		0,200	86	2304
		Максимальний	PPMD	0,197	85	2304
		Звичайний		0,197	85	2353
		Швидкий		0,197	85	2363
.wav	0,229	Максимальний	LZMA	0,141	61	2296
		Звичайний		0,141	61	2306
		Швидкий		0,183	79	2342
		Максимальний	Bzip2	0,187	81	2390

		Звичайний		0,187	81	2345
		Швидкий		0,187	81	2374
		Максимальний	PPMD	0,192	83	2237
		Звичайний		0,192	83	2324
		Швидкий		0,192	83	2342
.exe	0,229	Максимальний	LZMA	0,093	40	2353
		Звичайний		0,093	40	2314
		Швидкий		0,097	41	2363
		Максимальний	Bzip2	0,114	49	2309
		Звичайний		0,114	49	2369
		Швидкий		0,114	49	2324
		Максимальний	PPMD	0,101	43	2337
		Звичайний		0,101	43	2360
		Швидкий		0,103	44	2353
.xls		Максимальний	LZMA	0,041	17	2324
		Звичайний		0,041	17	2329
		Швидкий		0,046	19	2342
		Максимальний	Bzip2	0,054	23	2339
		Звичайний		0,054	23	2358
		Швидкий		0,054	23	2369
		Максимальний	PPMD	0,048	20	2358
		Звичайний		0,050	21	2369
		Швидкий		0,052	22	2326

.jpeg	0,229	Максимальний	LZMA	0,230	100	2332
		Звичайний		0,230	100	2360
		Швидкий		0,230	100	2327
		Максимальний	Bzip2	0,230	100	2342
		Звичайний		0,230	100	2404
		Швидкий		0,230	100	2363
		Максимальний	PPMD	0,231	100	2179
		Звичайний		0,231	100	2298
		Швидкий		0,231	100	2387

Таблиця 5 – Приклади стиснення архіватором BandiZip

Початковий файл	Початковий розмір, МБ	Режим стиснення	Розмір після стиснення, МБ	Ступінь стиснення, %	Швидкість стиснення, Кбіт/с
.doc	16,3	Максимальний	5,405	32	2363
		Звичайний	5,432	32	2393
		Швидкий	5,734	34	2353
.txt		Максимальний	3,125	18	2382
		Звичайний	3,146	18	2301
		Швидкий	3,815	22	2306
.pdf		Максимальний	15,835	95	2360
		Звичайний	15,836	95	2360
		Швидкий	15,855	95	2395
.png	16,3	Максимальний	15,919	94	2358
		Звичайний	15,935	94	2360

		Швидкий	15,994	95	2345
.mp3		Максимальний	15,724	93	2379
		Звичайний	15,744	93	2398
		Швидкий	15,785	93	2319
.wav		Максимальний	11,331	67	2355
		Звичайний	11,338	67	2398
		Швидкий	11,524	68	2345
.exe		Максимальний	6,015	36	2374
		Звичайний	6,038	36	2401
		Швидкий	6,238	37	2398
.xls		Максимальний	6,959	41	2329
		Звичайний	6,981	42	2404
		Швидкий	7,405	44	2314
.jpeg		Максимальний	16,625	99	2182
		Звичайний	16,626	100	2363
		Швидкий	16,620	100	2395
.doc	0,229	Максимальний	0,210	91	2398
		Звичайний	0,210	91	2334
		Швидкий	0,211	91	2374
.txt		Максимальний	0,015	6	2395
		Звичайний	0,015	6	2395
		Швидкий	0,020	8	2345
.pdf	0,229	Максимальний	0,226	98	2371

		Звичайний	0,226	98	2369
		Швидкий	0,225	97	2324
.png		Максимальний	0,222	96	2393
		Звичайний	0,222	96	2309
		Швидкий	0,222	96	2264
.mp3		Максимальний	0,203	87	2334
		Звичайний	0,203	87	2355
		Швидкий	0,204	88	2366
.wav		Максимальний	0,196	85	2314
		Звичайний	0,196	85	2379
		Швидкий	0,197	86	2173
.exe		Максимальний	0,107	46	2339
		Звичайний	0,108	46	2350
		Швидкий	0,114	49	2327
.xls		Максимальний	0,059	25	2342
		Звичайний	0,059	25	2385
		Швидкий	0,068	29	2363
.jpeg		Максимальний	0,230	100	2369
		Звичайний	0,230	100	2358
		Швидкий	0,230	100	2355