УДК 004.627 **doi: 10.32620/aktt.2024.6.10**

А. Є. ПЕРЕПЕЛИЦИН, А. І. ЧЕПЕЛЕВИЧ, А. А. ЛІТВІНОВ

Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна

АНАЛІЗ ФОРМАТІВ АРХІВІВ ТА ЇХ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ДЛЯ СТИСНЕННЯ ТЕКСТОВИХ ФАЙЛІВ

Предметом *вивчення в даній статті є поширені формати архівів для стиснення файлів, особливості їх реалізації, показники стиснення тексту і необхідний час для існуючих програм під різні платформи.* ***Метою*** *роботи є спрощення процесу вибору формату архіву та програмних засобів для роботи з ним для стиснення текстових файлів з урахуванням вимог часу, коефіцієнта стиснення і відкритості коду.* ***Завдання****: провести аналіз існуючих технологій та інструментів, які задіяні у процесі архівування даних; проаналізувати формати архівів, які широко використовуються; виконати аналіз важливих особливостей форматів архівів; виконати експериментальне дослідження параметрів стиснення для визначеного набору форматів та програмних засобів; запропонувати кроки інтеграції архівів до проекту системи із забезпеченням їх сумісності. Відповідно до поставлених завдань, були отримані наступні* ***результати****. Обговорюється застосування стиснення даних у системах зберігання інформації. Розглянуто доступні формати для архівування в Ubuntu. Виконано аналіз поширених форматів архівів. Проаналізовано особливості формату zip і формату rar для роботи з файлами великих розмірів. Виконано експериментальне дослідження параметрів стиснення для еталонних текстових файлів великого розміру з використанням десяти комбінацій на основі семи форматів та чотирьох програмних засобів. Запропоновано рекомендації щодо інтеграції архівів до проекту системи із забезпеченням сумісності.* ***Висновки.*** *Наукова новизна отриманих результатів полягає в тому, що проведений аналіз та експериментальне дослідження існуючих форматів архівів дозволяє спростити процес ухвалення рішення про використання відповідного формату на основі вимог до часу архівування, коефіцієнту стиснення та можливості використання програмної реалізації для конкретної платформи. Отримані результати дослідження дозволяють запропонувати використовувати формат архіву з відкритим кодом zpaq для стиснення тексту або набору версій проекту та документації для досягнення коефіцієнту стиснення який більш ніж удвічі менший ніж у формату rar, та на два відсотки менший ніж у форматах 7z і txz.*

Ключові слова: *формати архівів; rar; zip; 7z; tar.gz; tar.xz; zpaq; архівування даних; стиснення тексту.*

Вступ

У процесі проектування інженерних рішень та систем, у тому числі для аерокосмічній галузі, створюється велика кількість послідовних версій документації та файлів проектів. Для забезпечення компактного зберігання цієї інформації актуальною є задача пошуку технологічних рішень архівування.

Багато десятиліть розвивалася велика кількість форматів архівів [1], які враховували потреби як стиснення файлів [2], так і створення файлу зі збереженням інформації про час зміни, імена та структуру каталогів [3]. Залежно від цього створювалися формати, які являли собою безпосередньо архіви і дозволяли здійснювати об'єднання безлічі невеликих файлів в один файл, що давало змогу спростити процес роботи з ним на рівні файлової системи і прискорювало копіювання для магнітних дисків [4].

Надалі виникла необхідність стиснення таких форматів і стали з'являтися різні формати стиснення, які дозволяли залежно від набору даних у складі такого файлу здійснювати застосування набору алгоритмів стиснення.

Подальшим розвитком таких форматів стало використання багатотомних архівів, що давало змогу формувати архів для зберігання на різних накопичувачах із фіксованим розміром.

Хоча деякі формати архівів та архіватори [5] доступні з відкритим вихідним кодом, частина утиліт та форматів є комерційними. Але попит і популярність деяких форматів може наблизити момент публікації частини вихідного коду, як це було з компонентами розпакування для форматів rar. Але нові формати вже мають відкритий вихідний код, але все ще не підтримуються навіть у відкритих операційних системах (ОС).

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

Метою даної роботи є спрощення процесу вибору формату архіву та програмних засобів для роботи з ним для стиснення текстових файлів з урахуванням вимог часу, коефіцієнта стиснення і відкритості коду. Для досягнення цієї мети необхідно вирішити наступні **завдання**:

1) провести аналіз існуючих технологій та інструментів, яки задіяні у процесі архівування даних;

2) проаналізувати формати архівів, яки широко використовуються;

3) виконати аналіз важливих особливостей форматів архівів;

4) виконати експериментальне дослідження параметрів стиснення для визначеного набору форматів та програмних засобів;

5) запропонувати кроки інтеграції архівів до проекту системи із забезпеченням їх сумісності.

Структура цієї статті включає п'ять основних розділів. У розділі 1 розглядаються системи, де використовуються архіви як складова. У розділі 2 виконується аналіз поширених форматів архівів. Розділ 3 описує результати детального дослідження особливостей форматів zip і rar та інструментів для роботи з ними. Результати порівняння форматів архівів та запропоновані кроки по їх інтеграції при побудові систем наведено у розділах 4 і 5.

1. Аналіз технологій архівування даних

Розвиток форматів архівів потребував введення різних модифікацій, зокрема появи різних видів шифрування. У переважній більшості випадків застосовується стиснення на основі блокових шифрів. Утиліти, які використовують такі формати, припускають можливість введення пароля для архівування.

Поява нових форматів архівів пов'язана з необхідністю додавання нових якостей і, можливо, поліпшення стиснення. Частина таких форматів зумовлена комерційними та маркетинговими міркуваннями, у випадках якщо такі рішення є платними.

Для Unix систем існує набір стандартних архівів, які від самого початку є повністю рішеннями з відкритим вихідним кодом, та історично підтримуються вбудованими архіваторами, а також утилітами, що можуть бути встановлені або входять до складу стандартного набору утиліт у складі ОС.

Особливістю таких архівів є значна підтримка та більшість, як комерційних архіваторів, так і інших рішень, які підтримують ці архіви. До них можна віднести формати архівів zip, tar, gz.

Ubuntu 16.04 за замовчуванням пропонує створення архіву з таким переліком підтримуваних форматів архівів: 7z, z, ar, bz2, cbz, cpio, ear, gz, iso, jar, lzh, lzma, tar, tar.z, tar.bz2, tar.gz, tar.lz, tar.lzma, tar.lzo, tar.xz, war, xz, zip.

2. Аналіз поширених форматів архівів

Rar, створений у 1993 році, став одним з найпопулярніших форматів архівів завдяки високому ступеню стиснення, підтримці відновлення даних, шифрування та можливості створювати безперервні та багатотомні архіви [6]. Хоча rar є пропрієтарним форматом, для створення архівів потрібна ліцензія. Розпакування можливе за допомогою вільного програмного забезпечення, що робить його кросплатформним і широко вживаним.

Zip був створений у 1989 році компанією PKWARE і швидко став найпопулярнішим форматом архівів. Він підтримує широку сумісність на різних платформах, хороший ступінь стиснення, захист паролем і можливість створювати багатотомні архіви. Однак оригінальна версія zip має обмеження на розмір файлів і архівів до 4 ГБ, що стало проблемою для сучасних потреб.

Zip64, представлений у 2001 році, є розширенням формату zip, призначеним для подолання обмежень на розмір файлів і архівів, встановлених оригінальним zip. Zip64 підтримує архіви та файли розміром понад 4 ГБ, зберігаючи при цьому сумісність з існуючим програмним забезпеченням для архівації. Однак використання Zip64 вимагає програмної підтримки, яку старіші програми можуть не підтримувати.

7z, формат, який використовується програмою 7-Zip, був вперше випущений у 1999 році. Він пропонує високий рівень стиснення, підтримку сучасних алгоритмів, таких як LZMA, можливість шифрування, створення багатотомних архівів і підтримку великих розмірів файлів. Однак 7z підтримується за замовчуванням не на всіх платформах, тому для роботи з цим форматом потрібне спеціальне програмне забезпечення.

Tar (Tape Archive – стрічковий архів) був створений у 1979 році і використовується переважно в UNIX-подібних системах для об'єднання декількох файлів в один архів без стиснення. Tar підтримує метадані файлів, такі як дозволи та мітки часу, що робить його надійним інструментом архівації. Однак tar сам по собі не забезпечує стиснення даних і зазвичай використовується у поєднанні з форматами стиснення, такими як gzip.

Gz або **gzip** був створений у 1992 році і став стандартним засобом стиснення даних, особливо на Unix-подібних системах. Цей формат забезпечує швидке стиснення і розпакування, що робить його ідеальним для стиснення великих файлів і потоків даних. Основним обмеженням gzip є те, що він підтримує стиснення лише одного файлу; його часто використовують у поєднанні з tar для архівації кількох файлів.

Xz – це новий формат, який потенційно може замінити gz для розповсюдження скомпонованих файлів у Linux. tar.xz, представлений у 2009 році, поєднує в собі два формати: tar для архівування та xz для стиснення даних. Формат tar.xz став популярним на UNIX-подібних системах завдяки високому ступеню стиснення, який перевершує інші методи, такі як gzip і bzip2.

Він використовується для пакування багатьох файлів в один архів без зміни їхньої структури, а стиснення за алгоритмом LZMA2 дозволяє значно зменшити розмір архіву. Незважаючи на відсутність вбудованої підтримки шифрування та відновлення даних, tar.xz активно використовується для розповсюдження програмного забезпечення та архівування в середовищах, де важлива ефективність стиснення.

Zpaq, представлений у 2009 році як розвиток paq, підтримує стиснення з прогресивним оновленням, що дозволяє додавати нові файли до архіву без необхідності перестворювати весь архів. Zpaq також підтримує відновлення даних, але його відносна повільність і обмежена підтримка на сучасних платформах звужують сферу його застосування [7].

Pea – це архівний формат, запроваджений у 2006 році та розроблений за допомогою програми PeaZip. Особлива увага приділяється безпеці даних, підтримується шифрування, перевірка цілісності та можливість розбиття архіву на частини. Формат має повністю відкритий вихідний код і забезпечує гідний рівень стиснення, але його головною проблемою залишається безпека. Він широко використовується у проектах з відкритим вихідним кодом, але його поширення обмежене порівняно з популярними форматами.

Lzh, створений у 1988 році в Японії, забезпечував хороший для свого часу ступінь стиснення і був широко популярний в Японії, особливо для розповсюдження програмного забезпечення. Однак зараз lzh використовується лише в Японії, а його підтримка обмежена сучасними архіваторами.

Ace, створений у 1999 році, пропонував високий рівень стиснення і можливості відновлення даних, але зараз вважається застарілим. Обмежена кількість сучасних архіваторів підтримує цей пропрієтарний формат.

Arc, створений у 1985 році, був одним із перших форматів, який поєднував функції стиснення та архівування даних, що зробило його відомим у 1980-х роках. Хоча arc широко використовувався на початку розвитку Інтернету, з часом його витіснили більш сучасні формати, такі як zip, які забезпечували кращий ступінь стиснення та підтримку структури каталогів.

Paq – це серія форматів архівів, відомих своїми надзвичайно високими ступенями стиснення, які були досягнуті завдяки використанню складних моделей даних. Перші версії paq з'явилися на початку 2000-х років, і хоча paq забезпечує найвищий рівень стиснення, його використання вимагає значних обчислювальних ресурсів, що робить його повільним у роботі.

Zipx, представлений у 2009 році компанією WinZip, є розширеним форматом zip, який використовує поліпшені алгоритми стиснення, такі як JPEG і WAVPACK, для більш ефективного зменшення розміру певних типів даних. Однак zipx підтримується лише в останніх версіях таких програм, як WinZip, що обмежує його сумісність з іншими архіваторами.

Rk – формат архівів, розроблений для забезпечення високого рівня стиснення. Використовується в програмі RARLAB, але менш поширений у порівнянні з іншими форматами, такими як rar або zip. Відрізняється високою швидкістю стиснення і ефективністю при стисненні великих файлів. Однак формат rk є пропрієтарним і не має широкої підтримки в програмному забезпеченні з відкритим кодом.

3. Аналіз важливих особливостей архівів

Аналіз формату zip. Серед популярних форматів архівів, таких як zip, існують обмеження, пов'язані з розміром такого файлу. Це обмеження пов'язане з можливістю використання фіксованих типів даних для позначення довжини файлу.

Тому у випадку, якщо для позначення довжини файлу використовується беззнакове 32-х бітне значення, максимальний розмір файлу не може перевищувати 4 ГБ і для позначення більшої довжини потрібно використовувати додаткові поля.

Одним зі стандартів довжини типів даних є 32-х бітні поля, що пов'язано з історією еволюції самих процесорів. У зв'язку з цим було створено формат із більшою довжиною полів даних та інших полів, які відповідають за можливість роботи з файлами у складі такого архіву, який отримав назву zip64.

За замовчуванням більшість засобів архівування не дають змоги вибирати цільовий формат під час створення архіву zip.

З цієї причини частина засобів використовують за замовчуванням визначення розміру або типу формату залежно від кількості та розміру файлів у складі такого архіву. Тому доцільно знайти засіб і знайти можливість використання вихідного коду для створення власних рішень, під час побудови рішень, що використовують zip з 64-х бітними полями довжини файлу.

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

Аналіз особливостей формату rar. Іншою проблемою є можливість використання пропрієтарних рішень, таких як rar. Недоліком такого архіву, такого формату, є необхідність використання компонента із закритим вихідним кодом для архівування [8]. При цьому перевагою такого формату є наявність вихідного коду для розархівування і велика кількість корисних функцій, таких як безперервний архів, інформація для відновлення, томи для відновлення, можливість шифрування різних файлів, використання контрольних сум різної довжини в різних версіях архіву.

Існують недокументовані властивості такого типу архіву, і в різних версіях архівів використовувалися різні версії засобів відновлення та визначення ідентичності файлів і додаткових елементів у складі таких рішень.

Так у 4-ій версії формату rar, у середовищі rar, існувала віртуальна машина, яка давала змогу виконувати деякі функції для спрощення процесу розархівування. Під час експлуатації такої можливості, існувала така можливість модифікувати поведінку в процесі розархівування і виконувати деякі дії в складі архіву. У разі зловмисного використання такої поведінки, існувала можливість виконувати деякі дії під час відкриття такого архіву в стандартних засобах відкриття цього формату, однак пізніше таку можливість було видалено із засобів перегляду.

Додатково недокументованою особливістю такого формату є алгоритм пошуку фрагментів, що збігаються. Для версії 4 така довжина становила 10 МБ. Це означає, що в разі зустрічі однакових послідовностей довільної довжини в обсязі архівування до 10 МБ шукали і повністю виробляли такі послідовності, що давало змогу зменшити підсумковий розмір результуючого архіву.

При збільшенні розміру такого архіву такий алгоритм архівування не застосовувався і кожен дублікат у складі архіву стискався незалежно. На такі параметри не впливали параметри безперервного стиснення, а в пізніших версіях архіву таку можливість було винесено до складу параметрів архівування, і водночас вона не повторює повністю недокументовані особливості попередніх версій.

Перевагою таких форматів є можливість використовувати різні алгоритми відновлення, які дають змогу використовувати додаткову інформацію, на відміну від формату zip.

Більшість проєктів використовує архіви для однієї з двох цілей: власне для економії місця та стиснення файлів, і для фіксації структури та часу їхньої зміни. Однак процес додавання є найскладнішим етапом для роботи з архівами незалежно від того, чи є вони повністю відкритими, чи є зворотною розробкою закритого формату.

Процес стиснення вимагає додавання та аналізу всієї безлічі файлів, що входять до складу такого архіву, і якщо весь набір файлів не відомий на момент початку архівування, то виникає необхідність вибору правильних параметрів.

Більшість архіваторів дають змогу виконувати динамічне архівування, і в цьому разі всі архіви не містять конкретного заголовка, а розбиваються на безліч томів і безпосередньо мають у своєму розпорядженні поруч із файлами у складі архіву інформацію про такі файли. Таким чином процес вилучення й аналізу файлів у складі архівів потребуватиме доступу до всіх томів у складі архіву й аналізу всього вмісту цих файлів.

Для прискорення процесу роботи з архівами існує можливість розміщення додаткової інформації в заголовках, деякі формати підтримують такі блоки даних на початку архіву, деякі наприкінці, а деякі і на початку, і наприкінці.

Додавання інформації для відновлення є додатковою особливістю архівів для забезпечення можливості відновлення в разі пошкодження архівного файлу.

Така можливість дає змогу за рахунок завадостійкого кодування або інших засобів відновлення здійснювати захист даних у разі збою, що є актуальним не лише для накопичувачів і жорстких дисків, а й для поточного стану флеш-пам'яті, за якої збій окремих кластерів цілком можливий через деградаційні відмови.

Такий захист дає змогу відновлювати фіксовану кількість модифікованих блоків у складі архіву і залежно від використовуваного алгоритму відновлення забезпечує до половини обсягу додаткової інформації.

Наприклад, у разі використання коду Ріда-Соломона існує можливість відновлювати незалежно від кількості таких відмов до половини довжини блоків додаткового відновлення, і якщо така кількість становить, наприклад, 3% від розміру архіву, то можна відновити бінарно ідентичний файл із кількістю модифікацій до 1.5%.

Цікавим завданням є можливість додавання до складу формату zip, який повністю відкритий і надається з вихідним кодом засобів для відновлення. Існує технічна можливість поєднання декількох форматів у складі формату zip для того, щоб отримати результуючий формат zip, що підтримуватиметься всіма засобами перегляду за замовчуванням і при цьому міститиме інформацію для відновлення у форматі zip, яку може бути відновлено спеціалізованими типами архівів, можливість таких поєднань дає змогу розширити властивості наявних форматів з відкритим кодом та додати до них ті функції, які не підтримуються за замовчуванням.

Можливість використовувати максимальне стиснення і модифікація цього параметра залежно від використовуваних файлів є додатковим критерієм для роботи з архівами.

Більшість сучасних архіваторів, як 7-Zip та WinRar, автоматично визначають алгоритми стиснення залежно від набору даних. Існують специфічні формати даних, які підтримують можливість стиснення аудіоданих залежно від кількості доріжок, якщо аудіодані не використовують компресію, то в них існує закономірність зміни миттєвого значення сигналу, за рахунок чого за умови використання фіксованих форматів можна суттєво знизити час для стиснення та водночас досягти максимального ступеню стиснення, порівнюючи з режимом автоматичного вибору.

Це означає, що існують передустановки, на підставі яких можна підвищити ефективність процесу використання таких рішень [9].

Стиснення тексту також є важливим параметром архіву [10]. Так само, як і стиснення без втрат істинності зображень у raw від представлення байтових масивів.

Додатковою можливістю під час використання архівів є можливість саморозпакування для різних ОС у разі відсутності встановлених засобів для розпакування.

Такі рішення дають змогу здійснювати не тільки вилучення файлів, а й найпростіші кроки зі встановлення. Деякі дистрибутиви програм можуть бути засновані на саморозпаковувальних архівах для конкретної ОС.

Архіви, що саморозпаковуються, являють собою комбінацію бінарного файлу, який здійснює роботу з розпакуванням архіву з додаванням до нього поспіль самого тіла архіву, для того, щоб інструменти для розпакування могли коректно працювати з таким архівом, існує набір можливих алгоритмів пошуку інформації в самому архіві.

Однією з можливостей є використання в заголовку деякої маски, яка гарантовано не зустрічається в складі саморозпаковувального бінарного файлу (виконуваного бінарного файлу), і до таких стандартних заголовків належать ключові слова та фіксовані магічні набори, такі як використовують у WinRar. При цьому існує максимальна довжина такого заголовка, блоку для самого вилучення з тієї причини, що під час збільшення довжини такого блоку даних збільшується ймовірність випадкової зустрічі потрібної маски у складі файлу, що може призвести до хибної ідентифікації такого файлу, тому розмір частин, що саморозпаковуються, обмежується кількома сотнями кілобайтів.

4. Експериментальне порівняння форматів архівів

Методологія дослідження включає в себе порівняння стиснення та часу цього процесу з фіксованим середовищем, в якому він проводиться. Для зручності та простоти повторення було обрано варіант використання інтерфейсу командного рядка для роботи з архіваторами. За допомогою створеного PowerShell-скрипту було проведено процес архівації з використанням чотирьох архіваторів, а саме 7-Zip, WinRar, PeaZip і ZPAQ. Для 7-Zip та PeaZip тести проводилися на форматах 7z і zip, для WinRar – на rar і zip, а для ZPAQ – тільки на форматі zpaq.

Було використано наступні версії архіваторів: 7-Zip версії 24.08, WinRar 7.01, ZPAQ 6.34 та PeaZip версії 9.9.1.

Результати експериментального стиснення великих текстових файлів 29.lst [11] і 32.lst [12] з наборами поліномів наведено в таблиці 1 і таблиці 2.

Таблиця 1

Експериментальне порівняння параметрів стиснення для еталонного текстового файлу 29.lst з використанням засобів PeaZip, WinRar, 7-Zip і ZPAQ

Формат	Додаток	Стиснено, байти	Час, секунди	Стиснено, %
.pea	PeaZip	48632929	32,025	29,3553
.zip	WinRar	47919293	8,026	28,9245
.rar	WinRar	39773798	160,03	24,0078
.zip	7-Zip	31328111	65,029	18,9099
.zip	PeaZip	31328111	74,024	18,9099
.tar.gz	7-Zip	31309350	66,204	18,8986
.7z	PeaZip	19602370	50,024	11,8322
.7z	7-Zip	19602338	53,024	11,8321
.tar.xz	7-Zip	19602324	37,077	11,8321
.zpaq	ZPAQ	15111298	389,935	9,1213

Таблиця 2

Експериментальне порівняння параметрів стиснення для еталонного текстового файлу 32.lst з використанням засобів PeaZip, WinRar, 7-Zip і ZPAQ

Формат	Додаток	Стиснено, байти	Час, секунди	Стиснено, %
.pea	PeaZip	186667018	50,019	30,9062
.zip	WinRar	184656153	29,026	30,5732
.rar	WinRar	154317901	969,022	25,5502
.zip	7-Zip	123474764	247,125	20,4435
.zip	PeaZip	123474764	253,019	20,4435
.tar.gz	7-Zip	123474653	236,075	20,4435
.7z	PeaZip	80296601	104,087	13,2946
.7z	7-Zip	80291137	221,018	13,2937
.tar.xz	7-Zip	80290728	159,094	13,2936
.zpaq	ZPAQ	67726048	931,953	11,2133

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

Було використано такі параметри стиснення: для 7z – параметр -mx9, для rar– параметр -m5 і для zpaq64 – параметр -m6 (рисунок 1 і рисунок 2).

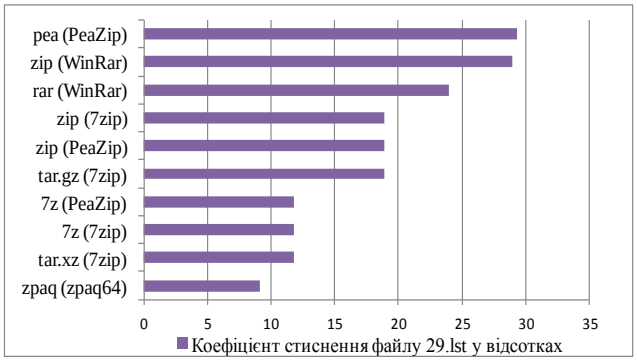

Рис. 1. Порівняння ступеня стиснення еталонного загальнодоступного текстового файлу 29.lst

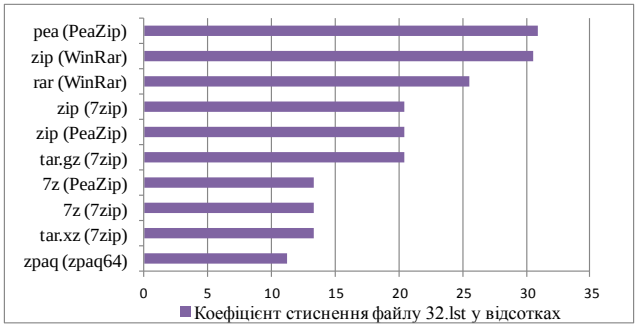

Рис. 2. Порівняння ступеня стиснення еталонного загальнодоступного текстового файлу 32.lst

Під час тестування записувалася інформація про розмір вхідного/вихідного файлу, ступінь стиснення, час початку та закінчення архівування, а також час, витрачений на архівування.

Ці параметри використовуються для порівняння типів і архіваторів. Для забезпечення рівності результатів у кожному тесті використовується 1 потік і максимальний рівень стиснення. Платформа для всіх тестів – однакове середовище на базі Acer nitro 5 an515-44 з Ryzen 5 4600H і Windows 11.

Результати тривалості архівування важливи для перехресного порівняння.

5. Запропоновані рекомендації щодо інтеграції архівів із забезпеченням сумісності

Додатковою можливістю ідентифікації файлу є додавання множинних заголовків, що дають змогу ідентифікувати такий файл у разі його пошкодження, за замовчуванням формат .zip передбачає наявність заголовка на початку та наприкінці, і на основі інформації в таких заголовках є можливість ідентифікувати такий файл, навіть якщо до нього додано надто довгий блок для самолікування або якщо заголовок було пошкоджено.

Деякі формати і бібліотеки для роботи з архівами дають змогу виконувати безпосередньо стиснення потоку в реальному часі, якщо навіть потік не доступний повністю, такі формати дають змогу виконувати компресію і стискати потік з розбиттям його на окремі блоки, для практичного використання таких стиснутих потоків потрібно додавати до них заголовки, для того, щоб засоби розархівування могли їх ідентифікувати.

Під час вибору засобів і конкретних технологій для роботи з архівами необхідно взяти до уваги вимоги до швидкодії та до швидкості розроблення таких рішень, тому що в разі відсутності конкретних вимог, наприклад, до мови програмування, може бути можливість вибору за швидкістю роботи. Так, наприклад, реалізація з використанням стандартних бінарних файлів таких, як динамічні бібліотеки, дає змогу суттєво підвищити швидкість роботи з наявними форматами за рахунок використання машинних кодів у складі таких рішень.

У разі якщо існує необхідність перенесення на різні платформи, наприклад для реалізації стиснення до складу Android рішень на різних процесорах, існує можливість написання таких рішень мовами проміжного рівня.

За можливості використання окремих бінарних блоків існує можливість написання компактних рішень для кожного з наявних процесорів, що дасть змогу підвищити швидкість роботи процесу стискання та розпакування таких архівів, але зробить сам файл доволі громіздким через необхідність зберігання скомпільованих заготовок для різних архітектур мобільних процесорів.

Дискусія

Під час розгляду всієї безлічі вимог до автоматичних засобів архівування та розархівування слід взяти до уваги можливість використання платних рішень та можливість використання рішень із відкритим вихідним кодом, швидкість роботи такого рішення, а також можливість його розархівування на різних платформах. Під час розгляду параметрів існує можливість ухвалення рішення про доцільність використання конкретного формату архіву залежно від набору вимог.

Однак для реалізації таких завдань необхідно виконати додаткове дослідження і додаткову розробку, що може перевищити позитивний ефект і вигоди від використання рішень з відкритим кодом. Однією з проблем під час побудови та інтеграції систем архівування до складу програмних рішень є

наявність хороших бібліотек, які дають змогу безпосередньо працювати з архівами, усю множину проблем можна поділити на процес роботи в режимі читання для розархівування та процес створення нового архіву або модифікацію наявного.

Для більшості платформ існують бібліотеки, які дають змогу нативно виконувати процес розархівування, водночас частина з цих бібліотек не є повністю протестованими та містять деякі дефекти, які можуть проявлятися для різних локалізацій, певних розмірів файлів і їхньої кількості та деяких полів у складі архівів, це означає, що в разі комбінації деяких умов такі архіви не можуть бути підтримуваними.

Наприклад, деякі засоби перегляду не можуть здійснювати коректне розшифрування зашифрованих архівів або здійснювати перегляд архівів у разі якщо використовується UTF-8 шляхи у складі такого архіву, подібною проблемою також є портування подібних рішень між різними ОС, через що частина рішень з графічним інтерфейсом не дає змоги коректно працювати з рішеннями, що загалом є повністю відкритими, але водночас не повністю протестованими в процесі перенесення, і у зв'язку з цим є потреба повноцінного аналізу таких рішень таких рішень для можливості прийняття рішень о доцільності використання конкретного формату архіву в залежності від набору вимог до проекту.

Висновки

Основним внеском цієї роботи є проведене аналітичне та експериментальне дослідження форматів архівів з точки зору можливості як персонального використання, так і інтеграції в інформаційні системи.

За результатами цього порівняння можна зробити висновок, що серед розглянутих форматів архівів zpaq забезпечує найвищий рівень стиснення і найбільшу тривалість цього процесу для більшості випадків.

Але в той же час формат 7z в PeaZip показує найшвидший процес додавання зі збереженням найвищого рівня стиснення.

Можна запропонувати наступні рекомендації щодо використання форматів архівів. Для зберігання даних на дисках з можливими пошкодженими кластерами краще використовувати формат rar з додаванням запису про відновлення. Для найвищого рівня стиснення текстової інформації краще використовувати формат zpaq. Для решти практичних випадків доцільно використовувати 7z у складі програми PeaZip.

Використання tar.xz може забезпечити значний ступінь стиснення, що є доцільним в версіях Linux.

Напрямком подальших досліджень є детальний розгляд задіяних алгоритмів стиснення та оптимізація реалізацій для конкретної платформи. Тривалість вилучення також є параметром, який може бути включений до кінцевих метрик, що може спростити процес прийняття рішень при виборі формату архіву.

Внесок авторів: формулювання мети і завдань дослідження – **А. Є. Перепелицин**, **А. І. Чепелевич**, **А. А. Літвінов**, дослідження поширених форматів архівів – **А. І. Чепелевич,** аналіз важливих особливостей форматів zip, rar і zpaq – **А. Є. Перепелицин**, експериментальне дослідження стиснення набору форматів – **А. А. Літвінов**, запропоновані кроки інтеграції архівів – **А. Є. Перепелицин**.

Конфлікт інтересів

Автори заявляють, що немає конфлікту інтересів щодо цього дослідження, фінансового, особистого, авторського чи іншого, який міг би вплинути на дослідження та результати, представлені в статті.

Фінансування

Дослідження проведено без фінансової підтримки.

Доступність даних

Рукопис не має пов'язаних даних.

Використання засобів штучного інтелекту

Автори підтверджують, що не використовували генеративних технологій штучного інтелекту при створенні представленої роботи та дослідженні.

Автори прочитали та погодилися з опублікованою версією рукопису.

Література

1. Large Text Compression Benchmark, Matt Mahoney (June 4, 2024). – Available at: http://www.mattmahoney.net/dc/text.html – 08.09.2024.

2. Improvable Deflate Algorithm [Text] / W. Weimin, G. Huijiang, H. Yi, F. Jingbao, & W. Huan // Proceedings of 2008 3rd IEEE Conference on Industrial Electronics and Applications. – 2008. – P. 1572–1574. DOI: 10.1109/ICIEA.2008.4582783.

3. Kim, H. Improving Small File I/O Performance for Massive Digital Archives [Text] / H. Kim, & H. Yeom // Proceedings of 2017 IEEE 13th International Conference on e-Science (e-Science 2017). – 2017. – P. 256–265. DOI: 10.1109/eScience.2017.39.

4. Sparenberg, H. Use-case-optimized data storage of scalable media files [Text] / H. Sparenberg, V. Bruns, & S. Foessel // Third International Conference on Innovative Computing Technology

ISSN 1727-7337 (print)
ISSN 2663-2217 (online)

(INTECH 2013). – 2013. – P. 35–39. DOI: 10.1109/INTECH.2013.6653633.

5. Prabavathy, B. Optimized private cloud storage for heterogeneous files in an university scenario [Text] / B. Prabavathy, P. Ramya, & C. Babu // Proceedings of 2013 IEEE International Conference on Recent Trends in Information Technology (ICRTIT 2013). – 2013. – P. 323–328. DOI: 10.1109/ICRTIT.2013.6844224.

6. An, X. Optimized Password Recovery for Encrypted RAR on GPUs [Text] / X. An, H. Jia, & Y. Zhang // Proceedings of 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems. – 2015. – P. 591–598. DOI: 10.1109/HPCC-CSS-ICESS.2015.270.

7. Reference-Based Compression of FASTQ Data Using Longest Match Model [Text] / S. Bai, J. Chen, Z. Lu, & W. Li // Proceedings of 2021 IEEE 4th International Conference on Information Communication and Signal Processing (ICICSP 2021). – 2021. – P. 598-603. DOI: 10.1109/ICICSP54369.2021.9611973.

8. Wei, Y. An Automatic Carving Method for RAR File Based on Content and Structure [Text] / Y. Wei, N. Zheng, & M. Xu // Proceedings of 2010 Second International Conference on Information Technology and Computer Science. – 2010. – P. 68–72. DOI: 10.1109/ITCS.2010.23.

9. Kawano, H. Hierarchical Storage Systems and File Formats for Web Archiving [Text] / H. Kawano // Proceedings of 2011 21st International Conference on Systems Engineering. – 2011. – P. 217-220. DOI: 10.1109/ICSEng.2011.46.

10. P, N. The Study of Text Compression Algorithms and their Efficiencies Under Different Types of Files [Text] / N. P, M. Sathya, & T. Vengattaraman // Proceedings of 2023 IEEE 1st International Conference on Optimization Techniques for Learning (ICOTL 2023). – 2023. – P. 1–8. DOI: 10.1109/ICOTL59758.2023.10435164.

11. Maximal Length LFSR Feedback Terms (29). – Available at: https://users.ece.cmu.edu/~koopman/lfsr/29.dat.gz – 08.09.2024.

12. Maximal Length LFSR Feedback Terms (32). – Available at: https://users.ece.cmu.edu/~koopman/lfsr/32.dat.gz – 08.09.2024.

References

1. *Large Text Compression Benchmark*, Matt Mahoney (June 4, 2024). Available at: http://www.mattmahoney.net/dc/text.html (accessed September 08, 2024).

2. Weimin, W., Huijiang, G, & Yi, H. Fan Jingbao and Wang Huan. Improvable Deflate Algorithm. *Proceedings of 2008 3rd IEEE Conference on Industrial Electronics and Applications*, 2008, pp. 1572-1574. DOI: 10.1109/ICIEA.2008.4582783.

3. Kim, H., & Yeom, H. Improving Small File I/O Performance for Massive Digital Archives. *Proceedings of 2017 IEEE 13th International Conference on e-Science (e-Science 2017)*, 2017, pp. 256-265. DOI: 10.1109/eScience.2017.39.

4. Sparenberg, H., Bruns, V., & Foessel, S. Use-case-optimized data storage of scalable media files. *Third International Conference on Innovative Computing Technology (INTECH 2013*), 2013, pp. 35-39. DOI: 10.1109/INTECH.2013.6653633.

5. Prabavathy, B., Ramya, P., & Babu, C. Optimized private cloud storage for heterogeneous files in an university scenario. *Proceedings of 2013 IEEE International Conference on Recent Trends in Information Technology (ICRTIT 2013)*, 2013, pp. 323-328. DOI: 10.1109/ICRTIT.2013.6844224.

6. An, X., Jia, H., & Zhang, Y. Optimized Password Recovery for Encrypted RAR on GPUs. *Proceedings of 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, 2015, pp. 591-598. DOI: 10.1109/HPCC-CSS-ICESS.2015.270.

7. Bai, S., Chen, J., Lu, Z., & Li, W. Reference-Based Compression of FASTQ Data Using Longest Match Model. *Proceedings of 2021 IEEE 4th International Conference on Information Communication and Signal Processing (ICICSP 2021*), 2021, pp. 598-603. DOI: 10.1109/ICICSP54369.2021.9611973.

8. Wei, Y., Zheng, N., & Xu, M. An Automatic Carving Method for RAR File Based on Content and Structure. *Proceedings of 2010 Second International Conference on Information Technology and Computer Science*, 2010, pp. 68-72. DOI: 10.1109/ITCS.2010.23.

9. Kawano, H. Hierarchical Storage Systems and File Formats for Web Archiving. *Proceedings of 2011 21st International Conference on Systems Engineering*, 2011, pp. 217-220. DOI: 10.1109/ICSEng.2011.46.

10. P, N., Sathya, M. & Vengattaraman, T. The Study of Text Compression Algorithms and their Efficiencies Under Different Types of Files. *Proceedings of 2023 IEEE 1st International Conference on Optimization Techniques for Learning (ICOTL 2023)*, 2023, pp. 1-8. DOI: 10.1109/ICOTL59758.2023.10435164.

11. *Maximal Length LFSR Feedback Terms (29)*. Available at: https://users.ece.cmu.edu/~koopman/lfsr/29.dat.gz (accessed September 08, 2024).

12. *Maximal Length LFSR Feedback Terms (32)*. Available at: https://users.ece.cmu.edu/~koopman/lfsr/32.dat.gz (accessed September 08, 2024).

Надійшла до редакції 10.10.2024, розглянута на редколегії 18.11.2024

ANALYSIS OF ARCHIVE FORMATS AND PROGRAM SOLUTIONS FOR COMPRESSION OF TEXT FILES

Artem Perepelitsyn, Alona Chepelevych, Andrii Litvinov

The subject of study in this article and research is widely used archive formats for file compression, features of their implementation, text compression rates, and the time required for existing programs for different platforms. The **goal** of the work is to simplify the process of choosing an archive format and program solutions for working with it for compressing text files, with taking into account time requirements, compression ratio, and open source. The **task** is to perform an analysis of existing technologies and tools involved in the data archiving process, to analyze archive formats that are widely used, to perform an analysis of important features of archive formats, to perform an experimental study of compression parameters for a specific set of formats and software tools, to propose steps for integrating archives into the project of system with ensuring the compatibility. According to the tasks, the following **results** were obtained. The application of data compression in information storage systems is discussed. The available formats for archiving in Ubuntu are considered. The detailed analysis of widely used archive formats is performed. The features of the zip and rar formats for working with large files are analyzed. An experimental study of compression parameters for large-sized reference text files using ten combinations based on seven formats and four software tools is performed. Compression parameters of the text with use of the same archive formats using different software tools are investigated. Recommendations for integrating archives into the project of system with ensuring the compatibility are proposed. The use of zpaq for the compression of text information is proposed. **Conclusions.** The scientific novelty of the obtained results is in the fact that the analysis and experimental study of existing archive formats allows simplifying the process of making a decision on using the required archive format based on the requirements for archiving time, compression ratio, and the possibility of using software implementation for a specific platform. The obtained research results allow to propose the use of the open source archive format zpaq for compressing text or a set of project versions and documentation to achieve a compression ratio that is twice better than for rar format, and two percent better than for 7z and txz formats.

Keywords: archive formats; rar; zip; 7z; tar.gz; tar.xz; zpaq; archiving of data; text compression.

Перепелицин Артем Євгенович – канд. техн. наук, доц., доц. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Чепелевич Альона Ігорівна – студ. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Літвінов Андрій Андрійович – студ. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Artem Perepelitsyn – PhD, Associate Professor at the Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.perepelitsyn@csn.khai.edu, ORCID: 0000-0002-5463-7889, Scopus Author ID: 56332607800.

Alona Chepelevych – Student at the Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.i.chepelevych@student.csn.khai.edu, ORCID: 0009-0009-5307-7101.

Andrii Litvinov – Student at the Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.a.litvinov@student.csn.khai.edu, ORCID: 0009-0006-3247-4579.