УДК 621.391.251:681.3.053 **Жураковський Б.Ю.**

СПОСОБИ СТИСНЕННЯ ДАНИХ ПРИ АРХІВАЦІЇ

В роботі розглядаються способи стиснення даних при архівації в інформаційних каналах систем управління. Наведена класифікація способів стиснення при архівації. Розрахована залежність коефіцієнту стиснення від типу інформації для різних способів.

Ключові слова: способи стиснення, стиснення при архівації, середня довжина кодової комбінації, коефіцієнт стиснення.

Використання алгоритмів даного способу стиснення може забезпечити передачу великих об'ємів інформації (наприклад, бази даних, добову звітність та інші документи) у системі управління мережею за допомогою дискретних каналів передачі інформації [1].

Для того, щоб інформація займала менше місця, її необхідно закодувати, тобто представити у вигляді більш компактних кодів. Схеми кодування існують тільки для випадку, коли необхідно закодувати одну з можливого ряду подій, при цьому їхні імовірності повинні бути відомі. Іншими словами, якщо необхідно усунути тільки надмірність розподілу подій, то досить лише оптимально закодувати події. Такий підхід використовувався в перших алгоритмах стиснення. Оскільки він не дозволяв усунути інші види надмірності, ступінь стиснення була низкою[2].

У сучасних алгоритмах стиснення для перетворення різних видів надмірності відповідно ймовірностям подій використовують моделі вхідних даних. Для деяких типів надмірності перед моделюванням використають спеціальні перетворення, що трансформують ці види надмірності в інші.

Для кожного типу надмірності існують свої моделі. Для надмірності розподілу подій використають статистичну модель нульового порядку. Ця модель містить імовірності настання наступних подій поза залежністю від попередніх.

Для надмірності повторення подій використають моделі RLE - Run Length Encoding. Ці моделі містять імовірності появи ланцюгів однакових подій різної довжини.

Для надмірності ланцюгів подій використовують словникові моделі [3].

Для надмірності розподілу подій після настання деякої кількості попередніх подій використовують статистичні моделі вищих порядків або виконують спеціальні перетворення, що групуються.

Позиційна надмірність у початковому виді, як правило, не зустрічається, або її відносять до інших типів надмірності. Але потрібно відзначити, що надмірність, яка виникає в результаті спектровиділяючих перетворень та залежить від розподілу ймовірностей позиції перетвореного зображення, також можна класифікувати як позиційну надмірність [4].

Для просторової надмірності або використовують моделі на основі просторових предикторів і додаткові моделі помилок, або виконують спектровиділяючі перетворення, які не усувають надмірність даних, але групують інформацію таким чином, щоб надмірність можна було легко усунути.

Розглянемо класифікацію та види алгоритмів стиснення інформації для передачі даних (рис.1).

Статистичне кодування Хафмена. Вхідним символам, поданим послідовностями бітів однакової довжини зіставляються послідовності бітів змінної довжини. Довжина коду для символу пропорційна (з округленням до цілого) двійковому логарифму частоти його появи, що береться з оберненим знаком. Це кодування є префікс ним, що дає змогу декодувати його однопрохідним алгоритмом. Префікс ний код зручно подавати у вигляді двійкового кодового дерева, в якого шлях від кореня до вершини визначає код символу.

До недоліків кодів Хафмена можна віднести те, що мінімальна довжина кодового слова для них не може бути менше одиниці, тоді як ентропія повідомлення цілком може становити й 0,1, і 0,01 біт/букву. У цьому випадку код Хафмена стає істотно надлишковим. Проблема вирішується застосуванням алгоритму до блоків символів, але тоді ускладнюється процедура

кодування/декодування та значно розширюється кодове дерево, яке потрібно в остаточному підсумку зберігати разом з кодом.

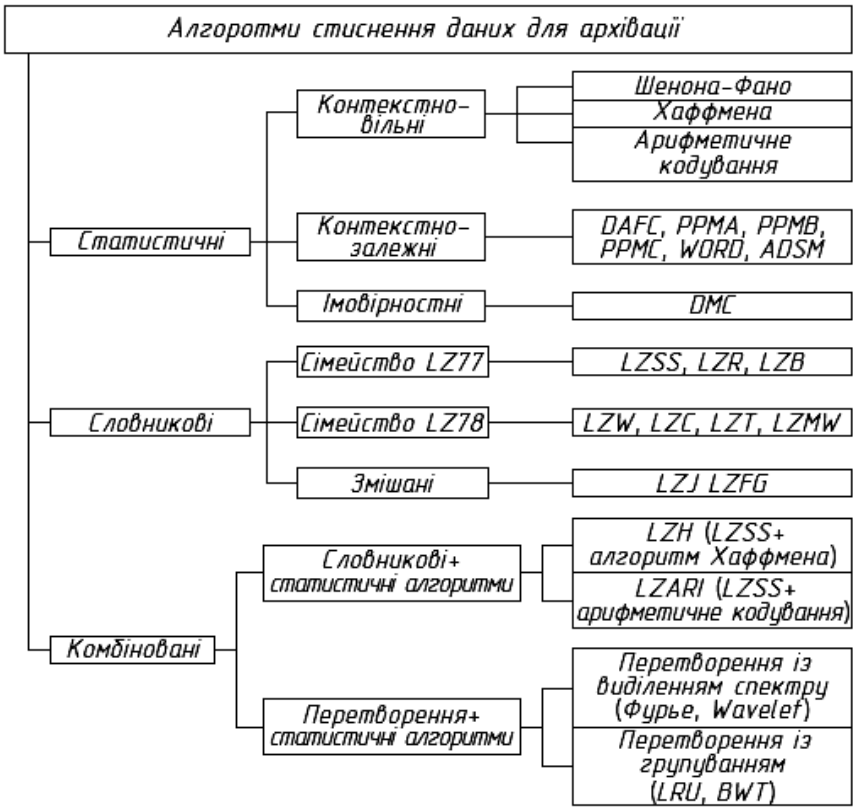

Рис. 1. Класифікація алгоритмів стиснення даних для архівації

Код Хафмена забезпечує середню довжину коду, що співпадає з ентропією, тільки у тому випадку, коли ймовірності символів джерела є цілими негативними ступенями двійки: 1/2 = 0,5; 1/4 = 0,25; 1/8 = 0,125; 1/16 = 0,0625 і т.д. На практиці така ситуація зустрічається дуже рідко або може бути створена блокуванням символів.

Арифметичне кодування. При арифметичному кодуванні текст подається дійсними числами в інтервалі від 0 до 1. У ході кодування тексту, відображаючий його інтервал зменшується, а кількість біт для його представлення збільшується. Чергові символи тексту зменшують величину інтервалу виходячи із значень їх ймовірностей, які визначаються моделлю. Більш ймовірні символи роблять це в меншій мірі, ніж менш ймовірні, а, отже, додають менше біт до результату. Перед початком роботи відповідному тексту інтервал є (0; 1). При обробці чергового символу його ширина звужується за рахунок виділення цьому символу частини інтервалу. Довжина інтервалу для символу дорівнює ймовірності його появи в повідомленні. Розміщення інтервалу ймовірності символу не має значення. Важливо тільки те, щоб і кодер, і декодер розташовували символи за однаковими правилами.

Результати стиснення, досягнуті даним алгоритмом змінюються від 4,8-5,3 біт/символ для коротких текстів до 4,5-4,7 біт/символ для довгих.

Алгоритм Зіва-Лемпеля. Практично всі словникові методи кодування належать родині алгоритмів з роботи двох ізраїльських учених - Зіва й Лемпеля, опублікованої в 1977 році. Сутність їх полягає в тому, що фрази в стисненому тексті заміняються вказівкою на те місце, де фрази вже раніше з'являлися.

Це сімейство алгоритмів називається методом Зіва-Лемпеля та позначається як LZ-стиснення. Цей метод швидко пристосовується до структури тексту й може кодувати короткі функціональні слова, тому що вони дуже часто в ньому з'являються. Нові слова й фрази можуть також формуватися із частин слів, які раніше зустрічались.

Декодування стислого тексту здійснюється прямо - відбувається проста заміна вказівки готовою фразою зі словника, на яку та вказує. На практиці LZ-метод досягає високого стиснення, його важливою властивістю є дуже швидка робота декодера. (Коли ми говоримо про текст, то припускаємо, що кодуванню піддається деякий вектор даних з кінцевим дискретним алфавітом, і це не обов'язково текст у буквальному значенні цього слова.)

Більшість словникових методів кодування мають ім'я авторів ідеї методу Зіва й Лемпеля, і часто вважають, що всі вони використовують один і той самий алгоритм кодування. Насправді різні представники цього сімейства алгоритмів дуже сильно розрізняються в деталях своєї роботи.

Всі словникові методи кодування можна розбити на дві групи.

Методи, що належати до першої групи, знаходячи в послідовності ланцюга, що кодується, символи які раніше вже зустрічалися, замість того, щоб повторювати ці ланцюги, замінюють їхніми вказівками на попередні повторення.

Словник у цій групі алгоритмів у неявному виді зберігається в оброблюваних даних, зберігаються лише вказівки на ланцюги, що зустрічаються серед повторюваних символів.

Алгоритми другої групи на додаток до вихідного словника джерела створюють словник фраз, що представляє собою повторювані комбінації символів вихідного словника, що зустрічаються у вхідних даних. При цьому розмір словника джерела зростає, і для його кодування буде потрібно більше число біт, але значна частина цього словника буде являти собою вже не окремі букви, а буквосполучення або цілі слова. Коли кодер виявляє фразу, що раніше вже зустрічалася, він заміняє її індексом словника, що містить цю фразу. При цьому довжина коду індексу виходить менше або набагато менше довжини коду фрази.

Всі методи цієї групи базуються на алгоритмі, розробленому й опублікованому Лемпелем і Зівом в 1978 році, - LZ78. Найбільш ефективним на даний момент представником цієї групи словникових методів є алгоритм LZW, розроблений в 1984 році Террі Велчем.

Перетворення Барроуза-Уілера. Алгоритм стиснення даних на основі перетворення Барроуза-Уілера (BWT) - це обернений алгоритм перестановки символів у вхідному потоці, що дозволяє ефективно стиснути отриманий у результаті перетворення блок даних.

Процедура перетворення відбувається так:

- Виділяється блок із вхідного потоку.
- Формується матриця всіх перестановок, отриманих у результаті циклічного зрушення блоку.
- Всі перестановки сортуються відповідно до лексикографічного порядку символів кожної перестановки.
- На вихід подається останній стовпець матриці й номер рядка, що відповідає оригінальному блоку.

Ефективне стиснення відбувається за рахунок того, що букви, пов'язані зі схожими контекстами, групуються у вхідному потоці разом.

Алгоритм PPM. Алгоритм PPM (prediction by partial matching) - це метод контекстно-обмеженого моделювання, що дозволяє оцінити ймовірність символу залежно від попередніх символів. Рядок символів, безпосередньо попередньому потоковому символу, будемо називати контекстом. Моделі, у яких для оцінки ймовірності використаються контексти довжиною не більш ніж N, прийнято називати моделями порядку N.

Імовірність символу може бути оцінена в контекстах різних порядків. Наприклад, символ "o" у контексті "to be or not t" може бути оцінений у контексті першого порядку “t”, у контексті іншого порядку “_t”, у контексті третього порядку “t_t” і т.д. Він також може бути оцінений у контексті нульового порядку, де ймовірності символів не залежати від контексту, і в контексті мінус першого порядку, де всі символи рівно ймовірні. Контекст мінус першого порядку використається для того, щоб виключити ситуацію, коли символ буде мати нульову ймовірність і не зможе бути закодований. Це може трапитися, якщо ймовірність символу не буде оцінена в жодному з контекстів (що можливо, якщо символ у них раніше не

зустрічався).

Існують два основних підходи до обчислення розподілу ймовірностей наступного символу на основі ймовірностей символів у контекстах. Перший підхід називається “повне перемішування”. Він припускає призначення важелів контекстам різних порядків й одержання сумарних ймовірностей додаванням імовірностей символів у контекстах, помножених на важелі цих контекстів. Застосування такого підходу обмежено двома факторами. По-перше, не існує швидкої реалізації даного алгоритму. По-друге, не розроблений ефективний алгоритм обчислення важелів контекстів. Примітивні ж підходи не забезпечують досить високої точності оцінки й, як наслідок, ступені стиснення.

Другий підхід називається “методом виключень”. При цьому підході спочатку робиться спроба оцінити символ у контексті найвищого порядку. Якщо символ кодується, алгоритм переходити до кодування наступного символу. У противному випадку кодується “відхід” й уживає спроба закодувати символ у контексті меншого порядку. І так далі, поки символ не буде закодований.

Вибір алгоритму кодування. Серед вищенаведених алгоритмів стиснення даних для архівації розглянемо для порівняння на рис. 2 статистичні контекстнозалежні коди DAFC, ADSM та словникові коди LZSS (кодування однорідних даних) та LZW (кодування даних різних типів). Коди вибрані за еквівалентною можливістю (складністю) реалізації та можливістю архівувати текстові, програмні, графічні та об’єктні файли.

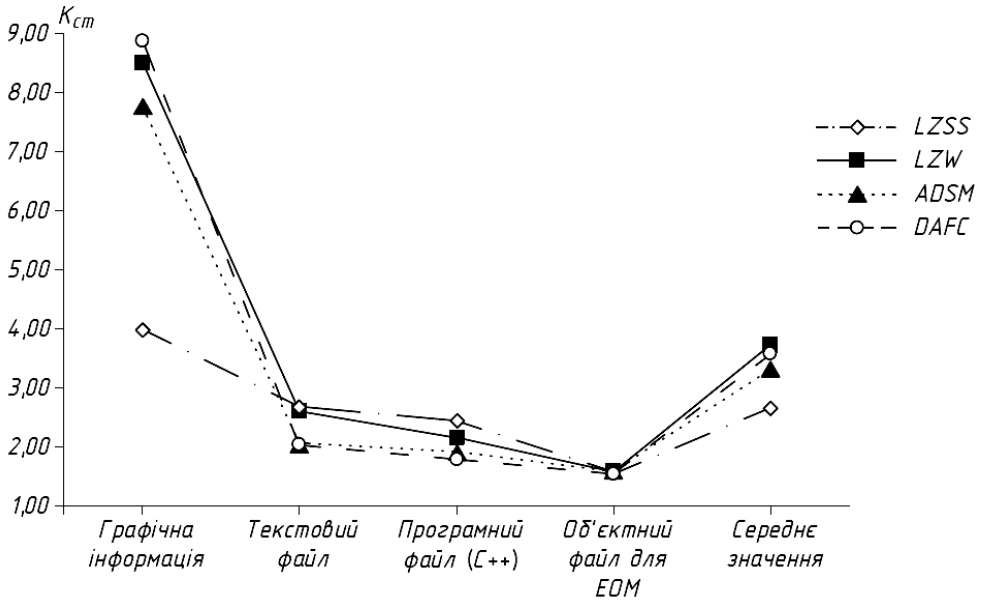

Рис. 2. Залежність коефіцієнту стиснення від типу інформації

Проаналізувавши приведену залежність коефіцієнту стиснення від типу інформації та вибраного коду ми можемо зробити висновок, що для передачі різнорідної інформації найбільшу ефективність стиснення дає код Зіва-Лемпеля-Велча (LZW).

ЛІТЕРАТУРА

1. Кричевский Р. Е. Сжатие и поиск информации. - М.: Радио и связь, 1989. - 168 с.
2. Цымбал В. П. Теория информации и кодирование: Учебник - 4 е изд., перераб. и доп. - К.: Вища школа, 1992 - 263 с.
3. Кохманюк Д. Сжатие данных: как это делается.Index Pro. - 1992. - №1. - С.18-29; 1993. - №2. - С. 30-49.
4. Жураковський Ю.П., Полторак В.П., Теорія інформації та кодування. – К.: Вища школа 2001. - 255 с.

Надійшла: 29.03.2013 р.

Рецензент: д.т.н., проф. Олійник В.Ф.