УДК 004.056

Т. В. Миронюк, *к.т.н.,*
доцент кафедри інформаційної безпеки та комп'ютерної інженерії,
e-mail: tanjamiron85@gmail.com
А. В. Чепеленко, *магістрант,*
e-mail: antonychepelenko@gmail.com
Черкаський державний технологічний університет,
б-р Шевченка, 460, м. Черкаси, 18006, Україна

ВДОСКОНАЛЕННЯ МЕТОДУ КОМПРЕСІЇ ДАНИХ НА ОСНОВІ КОДУ ФІБОНАЧЧІ

У статті розглядається ефективність компресії даних на основі коду Фібоначчі. Досліджується проблема використання коду змінної довжини для стискання даних на прикладі кодування текстового повідомлення. Визначено недоліки використання коду Фібоначчі при компресії текстових даних з великою кількістю унікальних символів. Запропоновано метод для збільшення ефективності компресії за допомогою коду Фібоначчі при великій кількості унікальних символів та проведено аналіз ефективності запропонованого методу порівняно зі стандартним методом.

Ключові слова: *дані, біт, скупчення даних, компресія даних, стискання без втрат, стиснення втрат, послідовність Фібоначчі.*

Вступ. На сьогодні більшість текстових документів, написаних навіть однією мовою, будуть містити значну кількість унікальних символів. Так, український алфавіт, який містить 33 букви-символи, буде представлений 66 унікальними символами, за рахунок того, що великі й малі букви – це різні символи в таблиці ASCII [8]. Додаймо до цього 10 цифр, знак пропуску, пунктуаційні знаки та отримаємо за 90 унікальних символів. Також не варто забувати про використання спеціальних символів та інших алфавітів (наприклад у підручниках математики, фізики, статтях з посиланнями чи цитати іншими мовами) і кількість унікальних символів у тексті перевищить значення 120 символів [6, 7]. Відповідно до цього стандартний метод генерації коду Фібоначчі [1, 4] буде менш ефективним, що й є основою для його покращення.

Також не варто забувати про форматування тексту, яке в різних текстових редакторах відрізняється, і при зчитуванні одного й того ж тексту в різних форматах може з'являтися значна кількість унікальних символів, які відповідають саме за форматування тексту. В таких випадках кількість унікальних символів починає перевищувати 150-180 і стандартний метод генерації коду Фібоначчі буде вже неефективним через його основні недоліки, а саме зменшення ефективності компресії при збільшенні кількості унікальних символів без збереження частоти розподілу цих символів. Вдосконалений метод навіть у таких випадках буде мати позитивне значення компресії даних.

Метою роботи є вдосконалення методу компресії даних на основі коду Фібоначчі за рахунок збільшення ефективності компресії на реальних даних.

Виклад основного матеріалу. Дані – це окремі фрагменти інформації, зазвичай відформатовані спеціальним способом. Вся програма ділиться на дві загальні категорії: дані та програми. Програми – це збірки інструкцій для обробки даних. Дані можуть існувати в різних формах – як цифри або текст на паперових носіях, як біти та байти, що зберігаються в електронній пам'яті. Біт – це найменша одиниця даних, що зберігаються на комп'ютері. Хоча комп'ютери надають інструкції, які можуть тестувати і управляти бітами, вони зазвичай призначені для зберігання даних та виконання інструкцій у бітних множинах, які називаються байтами. Байт складається з 8 бітів. У багатьох системах чотири восьмибітні байти (або октети) утворюють 32-розрядне слово. У таких системах тривалість інструкцій іноді виражається словом, тобто 32 біт у довжину, або наполовину словом, тобто 16 біт у довжину.

Стиснення даних визначається як представлення даних таким чином, що область

зберігання, необхідна для цільових даних, менша, ніж розмір вхідних даних. Існують два алгоритми для стиснення даних: перший безпосередньо стискає дані, тобто перетворює дані таким чином, щоб вихідна кількість бітів була менша, ніж вхідна, а другий виконує реконструкцію стиснутих даних до початкового стану. Стиснення тексту – важлива область стиснення даних без втрат. Дуже важливо, щоб реконструкція була ідентична тексту оригіналу, тому що дуже маленькі відмінності можуть призвести до висловлювань з різними значеннями. Символи зазвичай представлені за допомогою ASCII коду [8], де записано 256 символів. Частота кожного коду ASCII відрізняється одна від одної. Якщо текстові дані використовуються як введення, деякі символи зустрічаються найчастіше, і є багато символів, які ніколи не використовуються на вході. В основному використовуються алфавіт, цифри та деякі спеціальні символи. В алфавіті найчастіше використовуваними символами є голосні (табл. 1).

Таблиця 1

Середньостатистичні частоти букв та пропуску між словами в українській мові [5]

___	0,138	і	0,044	д	0,027	г	0,013	ж	0,007
о	0,086	р	0,043	л	0,027	ч	0,011	ш	0,005
н	0,068	е	0,042	п	0,025	х	0,011	є	0,005
а	0,064	с	0,037	з	0,020	ї	0,010	щ	0,004
и	0,055	к	0,033	я	0,019	ц	0,010	ф	0,003
в	0,046	м	0,029	ь	0,016	й	0,009	ґ	0.000
т	0,045	у	0,027	б	0,013	ю	0,008		

256 символів нечасто використовуються при компресії тексту. Різниця в цій частоті символів створює гарну можливість стиснення даних, притому кодом не фіксованої довжини, а кодом змінної довжини.

Компресія на основі Фібоначчі [2] генерує код змінної довжини. Використовуються традиційні методи заміни введених символів на специфічний код, наприклад кодові слова та ряд Фібоначчі [9] для реалізації стиснення.

У математиці числа Фібоначчі – це послідовність ряду чисел, в якому кожне наступне число дорівнює сумі двох попередніх: 1; 1; 2; 3; 5; 8; 13; 21; 34; 55; 89; 144. Часто, особливо в сучасному застосуванні, в послідовність доданий ще один елемент, а саме початковий 0: 0; 1; 1; 2; 3; 5; 8; 13; 21; 34; 55; 89; 144.

Математично визначається послідовність F_n чисел Фібоначчі за рекурентним співвідношенням

$$F_n = F_{n-1} + F_{n-2}, \qquad (1)$$

де, наприклад, $F_1 = 1$ і $F_2 = 1$ або $F_0 = 0$ і $F_1 = 1$.

При компресії кожне число Фібоначчі з послідовності (табл. 2) відображає кількість кодових слів, а номер числа з послідовності + 1 – це довжина цього кодового слова. Для компресії даних, які представляються через ASCII-код, максимально буде використовуватися 13 перших чисел із послідовності, тому що сума перших 12 чисел: 0 + 1 + 1 + 2 + 3 + 5 + 8 + + 13 + 21 + 34 + 55 + 89 = 233, що менше 256, тобто мінімальна необхідна кількість – це 13.

Таблиця 2

Послідовність Фібоначчі

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}
0	1	1	2	3	5	8	13	21	34	55	89	144

На основі цієї послідовності і буде будуватися код для компресії даних.

Ефективне саморозмежування подання довільного числа – це кодування Фібоначчі. Фібоначчі-кодування базується на тому, що будь-яке натуральне число n може бути однозначно виражено як сума різних чисел Фібоначчі, а тому не буде двох однакових послідовних чисел. Це означає, що, якщо буде використано бінарне представлення коду Фібоначчі, то не буде випадків двох послідовних одиниць.

Процес кодування на основі послідовності Фібоначчі [3] можна записати у вигляді наступного алгоритму:

Нехай n – це число, що відображає символ відповідно до ASCII-коду, S – вихідний масив, $F(i)$ – це число з послідовності Фібоначчі.

1. Знаходимо найбільше число $F(i)$, яке менше, ніж вхідне значення.
2. Додаємо і до вихідного масиву S.
3. Заміняємо n на $n - F(i)$.
4. Переходимо до кроку 1.

При використанні динамічного коду для компресії даних необхідно також ввести унікальний ключ для коректного розпізнавання закодованих даних і подальшої декомпресії даних. Оскільки в послідовності не може бути двох послідовних одиниць, можна це використати. Так, найкоротший можливий код, що буде показувати закінчення кодового слова, – це «11».

Отримані в результаті дослідження дані наведено в табл. 3.

Таблиця 3

Відображення чисел у коді Фібоначчі

Позиція	1	2	3	4	5	6	7	8	9	10	11	Відображення в коді Фібоначчі
Значення	1	2	3	5	8	13	21	34	55	89	144	
Ранг												
1	1											11
2	0	1										011
3	0	0	1									0011
4	1	0	1									1011
5	0	0	0	1								00011
6	0	1	0	1								01011
7	1	0	0	1								10011
8	0	0	0	0	1							000011
9	0	0	1	0	1							001011
10	0	1	0	0	1							010011
11	1	0	0	0	1							100011
12	1	0	1	0	1							101011

Процес стиснення даних за допомогою послідовності Фібоначчі [3] чітко розділений на стадії:

1. Зчитати вхідний файл і визначити частоту символів.
2. Відсортувати символи за частотами в порядку спадання замовлення.
3. Обчислити код Фібоначчі кожного рейтингу.
4. Створити таблицю для кодування.
5. На основі вхідного файлу і таблиці для кодування провести компресію даних.

Для подальшого дослідження наведемо приклад компресії даних. Для цього виконаємо компресію повідомлення: «Тестове повідомлення».

Результати отриманих обрахунків подано в табл. 4.

Таблиця 4

Процес обрахунку результатів компресії для першого прикладу

Символ	Частота	Ранг	Код Фібоначчі	Кількість бітів до компресії	Кількість байтів після компресії
е	3	1	11	24	6
о	3	2	011	24	9
в	2	3	0011	16	8
н	2	4	1011	16	8
__	1	5	00011	8	5
д	1	6	01011	8	5
і	1	7	10011	8	5
м	1	8	000011	8	6
л	1	9	001011	8	6
п	1	10	010011	8	6
с	1	11	100011	8	6
т	1	12	101011	8	6
я	1	13	0000011	8	7
Т	1	14	0001011	8	7

Дослідивши та підрахувавши наведені в табл. 4 результати, отримуємо:

- загальна кількість байтів до компресії – 160;
- загальна кількість байтів після компресії – 90.

В результаті буде отримано таке значення ефективності компресії:

$$1-\frac{90*100}{160}=43{,}75\%\,. \qquad (2)$$

За обрахунками отриманих вище даних визначено, що символи, які займали по 8 біт пам'яті, після компресії почали займати від 2 до 7 біт кожен. Але, якщо кількість різних символів збільшиться, тоді знадобиться використовувати все більше рангів для кодування.

Переглянувши результати, наведені в табл. 2, можна побачити, що в подальшому деякі символи при компресії почнуть займати більше, ніж 8 біт і максимально будуть займати 13 біт, за умови, що у вхідному файлі використовуються майже всі символи із ASCII-таблиці. В такому випадку вихідний файл може, навпаки, займати більше місця, ніж вхідний.

Цей алгоритм також дуже залежить від частоти розподілення символів. Якщо розподілення символів буде рівномірним, то цей алгоритм значно втратить ефективність компресії.

Розглянемо приклад найменшої ефективності алгоритму. Для цього візьмемо повідомлення розміром 160 біт, але символи в ньому не будуть повторюватися. Для прикладу використаємо повідомлення наступного типу: «1234567890йцукенгшщз». Це повідомлення містить рівно 20 різних символів розміром 160 біт.

Виконаємо компресію даних та наведемо результати виконання в табл. 5.

Таблиця 5

Процес обрахунку результатів компресії для другого прикладу

Символ	Частота	Ранг	Код Фібоначчі	Кількість бітів до компресії	Кількість байтів після компресії
1	2	3	4	5	6
1	1	1	11	8	2
2	1	2	011	8	3
3	1	3	0011	8	4
4	1	4	1011	8	4
5_	1	5	00011	8	5
6	1	6	01011	8	5
7	1	7	10011	8	5
8	1	8	000011	8	6
9	1	9	001011	8	6
0	1	10	010011	8	6
й	1	11	100011	8	6
ц	1	12	101011	8	6
у	1	13	0000011	8	7
к	1	14	0001011	8	7
е	1	15	0010011	8	7
н	1	16	0100011	8	7
г	1	17	0101011	8	7
ш	1	18	1000011	8	7
щ	1	19	1001011	8	7
з	1	20	1010011	8	7

Підрахувавши отримані в табл. 5 результати, було визначено, що:

- загальна кількість байтів до компресії – 160;
- загальна кількість байтів після компресії – 114.

Після виконання обрахунку було отримано таке значення ефективності компресії:

$$1-\frac{114*100}{160}=28{,}75\%\,. \qquad (3)$$

Порівнявши результати із першим прикладом, було визначено, що ефективність компресії зменшилася на 15 %.

Як було зазначено вище, однією з основних проблем існуючого методу компресії є його використання на даних з великою кількістю унікальних символів. Тому, відштовхуючись від отриманих даних, було запропоновано метод, який збільшує ефективність при середньому значенні унікальних символів від 100 одиниць, але зменшує ефективність при меншому значенні унікальних символів.

Головним принципом запропонованого методу є розширення таблиці кодування введенням нових закінчень. Так, можна ввести додаткові закінчення «1111», «111111» і т. д., але при цьому доведеться видалити варіанти, коли символ буде закодований тільки закінченням, тобто видалити такі коди Фібоначчі: «11», «1111» і т. д.

Результати кодів Фібоначчі із застосуванням запропонованого методу наведено в табл. 6.

Таблиця 6

Таблиця кодів Фібоначчі згідно з новим методом

Ранг	Код згідно з новим методом
1	011
2	0011
3	1011
4	00011
5	10011
6	01011
7	01111
8	000011
9	100011
10	010011
11	001011
12	101011
13	001111
14	101111

Відповідно при компресії даних із нерівномірною частотою унікальних символів такий метод дає втрату близько 5 %, але при компресії файлів із рівномірною частотою унікальних символів цей метод дає покращення компресії на 10-15 %.

Результати тестування запропонованого методу наведено в табл. 7.

Таблиця 7

Результати порівняння компресії

Розмір файлу, байт	Кількість унікальних символів	Розмір після компресії методом Фібоначчі, байт	Розмір після компресії методом збільшення ефективності, байт
43010	119	31670	31007
60825	137	47860	45762
6944	99	4468	4545
1173	29	590	639
131134	256	170591	155819
93458	184	95034	87153

Відповідно до результатів, поданих в табл. 7, видно, що запропонований метод ефективніший за стандартний метод при кількості унікальних символів, більшій за 110-120 символів.

Код Фібоначчі можна також використовувати для шифрування даних. У цьому випадку запропонований метод має більш складну логіку шифрування, тому що при компресії / шифруванні даних з кількістю понад 220 унікальних символів запропонований метод зменшить вихідний розмір даних на 10-15 %.

Висновки. В процесі дослідження розглянуто ефективність компресії даних на осно-

ві коду Фібоначчі. Досліджено проблему використання коду змінної довжини для стискання даних на прикладі кодування тестового повідомлення. Розглянуто загальний принцип компресії даних, її різновиди та проаналізовано зв'язок між розподілом частоти вживання певних літер і можливістю застосування компресії на основі кодів динамічного розміру. Також розглянуто загальний принцип побудови таблиць кодування для цього методу і проведено практичні обрахунки ефективності методу для випадків розподілу символів згідно зі статистикою та при рівномірному розподілі символів. На основі обрахунків ефективності компресії в різних випадках виокремлено основні недоліки методу для подальшого вдосконалення алгоритму.

Враховуючи визначені недоліки, проведено вдосконалення методу компресії на основі кодів Фібоначчі, що забезпечило підвищення ефективності компресії реальних даних.

Запропоновано метод для збільшення ефективності компресії за допомогою коду Фібоначчі при великій кількості унікальних символів та проведено аналіз ефективності запропонованого методу порівняно зі стандартним методом. Визначено, що вдосконалений метод з виключенням має вищий відсоток компресії над стандартним методом, але результати суттєво залежать від частотного розподілу символів. Вдосконалений метод спрямований саме на компресію текстів, в яких використовується декілька мов, спеціальні символи, а також на компресію файлів будь-якого формату, де стандартний метод має малий, або навіть від'ємний показник компресії.

Список літератури

1. Using Fibonacci compression codes as alternatives to dense codes. URL: https://ieeexplore.ieee.org/document/4483325 [Accessed 2 Sep. 2018].
2. Data compression using Fibonacci sequence. URL: http://www.academia.edu/32058265/Data_Compression_using_Fibonacci_Sequence [Accessed 2 Sep. 2018].
3. Fast Fibonacci encoding algorithm. URL: http://ceur-ws.org/Vol-567/paper 14. pdf
4. Fibonacci data compression. URL: http://code.activestate.com/recipes/577554-fibonacci-data-compression [Accessed 4 Sep. 2018].
5. Частоти повторюваності букв і біграм у відкритих текстах українською мовою. URL: http://ecobio.nau.edu.ua/index.php/ZI/article/viewFile/1968/1959 (Дата звернення: 3.09.2018).
6. Frequency of character combinations for three languages. URL: https://www.joyofdata.de/blog/frequency-of-character-combinations [Accessed 2 Sep. 2018].
7. Letter frequency of different languages. URL: http://letterfrequency.org/letter-frequency-by-language [Accessed 4 Sep. 2018].
8. ASCII table and description. URL: http://www.asciitable.com/ [Accessed 4 Sep. 2018].
9. Fibonacci sequence. URL: https://www.mathsisfun.com/numbers/fibonacci-sequen ce.html [Accessed 3 Sep. 2018].

References

1. Using Fibonacci compression codes as alternatives to dense codes. URL: https://ieeexplore.ieee.org/document/4483325 [Accessed 2 Sep. 2018].
2. Data compression using Fibonacci sequence. URL: http://www.academia.edu/32058265/Data_Compression_using_Fibonacci_Sequence [Accessed 2 Sep. 2018].
3. Fast Fibonacci encoding algorithm. URL: http://ceur-ws.org/Vol-567/paper 14. pdf
4. Fibonacci data compression. URL: http://code.activestate.com/recipes/577554-fibonacci-data-compression [Accessed 4 Sep. 2018].
5. Frequencies of recurrence of letters and digrams in open texts in Ukrainian. URL: http://ecobio.nau.edu.ua/index.php/ZI/article/viewFile/1968/1959 [Accessed 3 Sep. 2018] [in Ukrainian].
6. Frequency of character combinations for three languages. URL: https://www.joyofdata.de/blog/frequency-of-character-combinations [Accessed 2 Sep. 2018].
7. Letter frequency of different languages. URL: http://letterfrequency.org/letter-frequency-by-language [Accessed 4 Sep. 2018].
8. ASCII table and description. URL: http://www.asciitable.com/ [Accessed 4 Sep. 2018].
9. Fibonacci sequence. URL: https://www.mathsisfun.com/numbers/fibonacci-sequen ce.html [Accessed 3 Sep. 2018].

T. V. Myronyuk, *Ph.D.*,
associate professor of information security and computer engineering chair
e-mail: tanjamiron85@gmail.com
A. V. Chepelenko, *master*
e-mail: antonychepelenko@gmai.com
Cherkasy State Technological University,
Shevchenko blvd, 460, Cherkasy, 18006, Ukraine

ANALYSIS AND IMPROVEMENT OF DATA COMPRESSION METHOD BASED ON FIBONACCI CODE

This article discusses the effectiveness of data compression based on Fibonacci code. The problem of using variable-length code for data compression is studied on the example of a test message encoding. The general principle of data compression, its variants are considered and the connection between the distribution of the frequency of the use of certain letters and the possibility of applying compression on the basis of codes of dynamic size is analyzed. Also, the general principle of constructing the coding tables for this method is considered, and practical calculations of the efficiency of the method for cases of symbols distribution according to statistics and at a uniform distribution of symbols are carried out. Based on calculations of compression efficiency in different cases, the main disadvantages of the method are outlined for further improvement of the algorithm.

Taking into account certain deficiencies, the compression method on the basis of Fibonacci codes has been improved, which ensures the increase of the efficiency of real data compression.

A method for increasing the compression efficiency by means of Fibonacci code with a large number of unique symbols is offered and the analysis of the efficiency of the proposed method compared with the standard method has been conducted. It has been determined that the improved exception method has a higher percentage of compression over the standard one, but the results are significantly dependent on the frequency division of the symbols. The improved method is directed on the compression of texts using multiple languages, special symbols, as well as on the compression of files of any format where the standard method has a small or even negative compression ratio.

Keywords: *data, bit, data aggregation, data compression, lossless compression, loss compression, Fibonacci sequence.*