MA'LUMOTLARNI SIQISHDA BITLI ALGORITMLARDAN FOYDALANISH

Farmonov Sherzodbek Raxmonjonovich
Farg'ona davlat universiteti amaliy matematika va informatika kafedrasi katta o'qituvchisi
e-mail: farmonovsh@gmail.com
Shavkatova Dilyoraxon Iqbol qizi
Farg'ona davlat universiteti talabasi
e-mail: dildoradhavkatova@gmail.com

Annotatsiya. *Ma'lumotlarni siqish (kompressiya) — bu ma'lumotlarni saqlash yoki uzatish uchun kerak bo'lgan joyni kamaytirish maqsadida ularni o'zgartirish jarayonidir. Bitli algoritmlar (yoki bit o'zgartirish algoritmlari) esa ma'lumotlarni bit darajasida siqish uchun ishlatiladigan metodlar bo'lib, ular ma'lumotlar oqimini qisqartirish va samarali tarzda saqlash imkonini beradi.*

Kalit so'zlar: *Run-Length Encoding (RLE), Data Compression, Lossless Compression, Encoding, Decoding, Compression Ratio, Huffman Coding, Lempel-Ziv-Welch (LZW), Bit-level Compression, Data Encoding, String Compression, Symbol Frequency, Run, Lossy Compression, Entropy.*

Аннотация. *Сжатие данных (сжатие) — это процесс изменения данных с целью уменьшения места, необходимого для хранения или передачи. Побитовые алгоритмы (или алгоритмы побитового сдвига) — это методы, используемые для сжатия данных на битовом уровне, что позволяет сокращать и эффективно хранить поток данных.*

Ключевые слова: *многосерийное кодирование (RLE), сжатие данных, сжатие без потерь, кодирование, декодирование, степень сжатия, кодирование Хаффмана, Лемпеля-Зива-Уэлча (LZW), битовое сжатие, кодирование данных, сжатие строк, частота символов, запуск. , Сжатие с потерями, Энтропия.*

ISSN 3060-4567

Annotation. *Data compression (compression) is the process of changing data in order to reduce the space required for storage or transmission. Bitwise algorithms (or bit-shifting algorithms) are techniques used to compress data at the bit level, which allows the data stream to be shortened and stored efficiently.*

Keywords: *Run-Length Encoding (RLE), Data Compression, Lossless Compression, Encoding, Decoding, Compression Ratio, Huffman Coding, Lempel-Ziv-Welch (LZW), Bit-level Compression, Data Encoding, String Compression, Symbol Frequency, Run, Lossy Compression, Entropy.*

Bitli algoritmlar haqida umumiy tushuncha

Bitli algoritmlar ma'lumotlarni o'zining asosiy qismlarida (ya'ni, bitlar darajasida) manipulyatsiya qilish orqali siqadi. Bunday algoritmlar oddiy ma'lumotlarni, masalan, matn, rasm yoki audio fayllarni, kamroq joy egallagan formatda saqlashga imkon beradi.

Ushbu algoritmlar ko'pincha quyidagi usullarga asoslanadi:

1. **Kodlash (Encoding) va dekodlash (Decoding)**: Bu jarayonda ma'lumotlar ma'lum bir qoidalar asosida kodlanadi va siqilgan holatga keltiriladi. Kodlash jarayoni ma'lumotlarni yangi bit ketma-ketligiga o'zgartiradi.

2. **Fikr va takrorlanishlarni bartaraf etish**: Siqish algoritmlari ma'lumotlarda takrorlanadigan yoki o'xshash qismlarni aniqlab, ularni yagona va qisqa ifodada saqlaydi.

Ma'lumotlarni siqishda ishlatiladigan bitli algoritmlar

1. **Huffman kodlash**: Huffman kodlash — bu eng mashhur siqish usullaridan biri. Huffman algoritmi ma'lumotlardagi eng ko'p uchraydigan belgilarga qisqa kodlar (bit ketma-ketliklari) tayinlaydi, kam uchraydigan belgilarga esa uzun kodlar tayinlanadi. Bu algoritm asosan *yunon* (lossless) siqish algoritmlarida qo'llaniladi.

Misol: Agar bir faylda "a" harfi ko'pincha uchrasa, unda bu harfga qisqa kod (masalan, 0) tayinlanadi, "b" harfi esa kamroq uchrasa, unga uzunroq kod (masalan, 10) tayinlanadi.

2. **Run-Length Encoding (RLE)**: Run-Length Encoding (RLE) — bu oddiy va samarali siqish usuli bo'lib, ma'lumotlar ketma-ketligidagi bir xil elementlarning takrorlanishini bitta son va unga mos keluvchi qiymat bilan almashtiradi. Bu usul ayniqsa tasvirlar va matnlarda takrorlanadigan belgilar mavjud bo'lganda samarali ishlaydi.

Misol: "AAAAAABBBCCDAA" stringi RLE orqali "6A3B2C1D2A" ga siqilishi mumkin.

3. **Lempel–Ziv–Welch (LZW)**: LZW — bu algoritm ko'pincha *lossless* kompressiya usuli sifatida ishlatiladi. Bu algoritm ma'lumotlarda tez-tez uchraydigan bloklar yoki subqatorlarni identifikatsiya qiladi va ularni qisqa kodlar bilan almashtiradi. Bu usulni masalan, GIF va TIFF tasvir formatlarida, shuningdek, Unix tizimidagi *compress* va *zip* arxivlarida ishlatish mumkin.

4. **Arithmetic coding**: Arithmetic coding — bu yanada ilg'or va murakkab siqish algoritmi bo'lib, u simvollarni va ularning ehtimolliklarini bir xil arifmetik intervalga (ya'ni, son intervaliga) joylashtiradi. Bunda har bir belgi o'zining ehtimolligiga qarab intervalni bo'lib chiqadi, va oxir-oqibat bir butun son bilan ifodalanadi.

Bu usulda, masalan, "AAAABBB" stringi, A va B harflarining ehtimolligiga asoslanib, bittalik son sifatida siqiladi.

Bitli siqish algoritmlarining afzalliklari va kamchiliklari

Afzalliklari:

- **Hajmni qisqartirish**: Bitli siqish algoritmlari ma'lumotlarni ancha samarali tarzda siqadi, ayniqsa takrorlanishlar bo'lsa.
- **Samaradorlik**: Ba'zi algoritmlar (masalan, Huffman va LZW) katta hajmdagi ma'lumotlar uchun juda samarali.
- **Boshqarish osonligi**: Ko'pgina bitli algoritmlar juda oddiy va ularni ishlatish juda qulay, masalan, RLE yoki Huffman kodlashni amalga oshirish.

Kamchiliklari:

- **Murakkablik**: Ba'zi algoritmlar (masalan, Arithmetic Coding) juda murakkab va ularni ishlab chiqish yoki optimallashtirish vaqt olishi mumkin.

- **Maxsus holatlar**: Ba'zi siqish algoritmlari faqat maxsus turdagi ma'lumotlarga (masalan, rasm yoki matn) yaxshi ishlaydi, ammo boshqa turdagi ma'lumotlar uchun samarali bo'lmasligi mumkin.

Quyida **Run-Length Encoding (RLE)** siqish algoritmi asosida ishlaydigan bir masalani va unga mos C# dasturini keltiraman.

Masala: Run-Length Encoding (RLE) yordamida matnni siqish

Ma'lum bir matn berilgan. Ushbu matnda bir xil belgilar ketma-ket kelsa, ularni siqish kerak, ya'ni har bir takrorlangan belgi va uning sonini saqlash.

Masala sharti:

- Masalan, "AAAABBBCCDAA" matnini siqish kerak.
- Natijada "6A3B2C1D2A" formatida siqilgan matnni olishimiz kerak.

C# dasturi:

```
using System;
using System.Text;
class Program
{    // Run-Length Encoding (RLE) algoritmi
   static string RunLengthEncode(string input)
   {       StringBuilder encodedString = new StringBuilder();
      // Agar input bo'sh bo'lsa, bo'sh satr qaytariladi
      if (string.IsNullOrEmpty(input))
      {       eturn "";       }
      int count = 1;
      // Matnni birin-ketin tekshirib chiqamiz
      for (int i = 1; i < input.Length; i++)
      {          if (input[i] == input[i - 1])          {
count++;  // Agar hozirgi belgi oldingi bilan bir xil bo'lsa, sanagichni oshiramiz          }          else
                {          // Aks holda, oldingi belgi va uning takrorlanish sonini qo'shamiz
```

```
                encodedString.Append(count);
                encodedString.Append(input[i - 1]);
                count = 1;  // Yangi belgi uchun sanagichni qayta boshlaymiz
            }
        }       // Oxirgi belgi va uning sanini qo'shish
        encodedString.Append(count);
        encodedString.Append(input[input.Length - 1]);
        return encodedString.ToString();
    }   static void Main(string[] args)
    {       // Input matn
        string inputString = "AAAABBBCCDAA";
        // RLE siqishni amalga oshirish
        string encodedString = RunLengthEncode(inputString);
        // Natijani chiqarish
        Console.WriteLine("Original matn: " + inputString);
        Console.WriteLine("Siqilgan matn: " + encodedString);
    }}
```

Dastur qanday ishlaydi:

1. **RunLengthEncode** funksiyasi:

- Bu funksiya matnni (string) olib, uning Run-Length Encoding (RLE) siqish algoritmasi yordamida siqilgan versiyasini qaytaradi.
- StringBuilder yordamida siqilgan matn quriladi. Har bir belgining qancha marta takrorlanganligi hisoblanadi va ular ketma-ketligi sifatida qo'shiladi (masalan, "6A" 6 ta 'A' ni bildiradi).

2. **Main** funksiyasi:

- Bu erda dastur AAAABBBCCDAA matnini oladi va RunLengthEncode funksiyasini chaqirib, natijani chiqaradi.

Masalaning ishlashini tushunish:

Agar dasturga "AAAABBBCCDAA" matni kiritilsa:

- "AAAA" — bu 4 ta 'A' ni bildiradi, ya'ni "4A".

- "BBB" — bu 3 ta 'B' ni bildiradi, ya'ni "3B".
- "CC" — bu 2 ta 'C' ni bildiradi, ya'ni "2C".
- "D" — bu bitta 'D' ni bildiradi, ya'ni "1D".
- "AA" — bu 2 ta 'A' ni bildiradi, ya'ni "2A".

Natijada siqilgan matn quyidagicha bo'ladi:

"4A3B2C1D2A"

Dasturdan foydalanish:

1. Dastur ishga tushirilganda, sizga biror matn kiritilishi kerak bo'ladi (yoki kodda berilgan matnni o'zgartirishingiz mumkin).

2. Dastur avvalgi matnni RLE siqish algoritmi bilan siqib, siqilgan natijani chiqaradi.

Bitli siqish algoritmlari ma'lumotlarni saqlash va uzatish samaradorligini oshirishda juda muhim vositalardir. Ular ko'plab real dunyo ilovalarida, masalan, fayl siqish dasturlari, rasm va video formatlarida ishlatiladi. Har bir algoritmning o'ziga xos afzalliklari va kamchiliklari mavjud, shuning uchun aniq bir holatda eng yaxshi siqish usulini tanlash muhimdir.

FOYDALANILGAN ADABIYOTLAR:

1. Marcin Jamro. C# Data Structures and Algorithms. Second Edition. Published by Packt Publishing Ltd., in Birmingham, UK. 2024. – 349 p.

2. Дж.Эриксон. Алгоритмы.: – М.: " ДМК Пресс ", 2023. – 528 с.

3. Hemant Jain. Data Structures & Algorithms using Kotlin. Second Edition. in India. 2022. – 572 p.

4. Н. А. Тюкачев, В. Г. Хлебостроев. C#. Алгоритмы и структуры данных: учебное пособие для СПО. – СПб.: Лань, 2021. – 232 с.

5. Mykel J. Kochenderfer. Tim A. Wheeler. Algorithms for Optimization. Published by The MIT Press., in London, England. 2019. – 500 p.

6. Рафгарден Тим. Совершенный алгоритм. Графовые алгоритмы и структуры данных. – СПб.: Питер, 2019. - 256 с.

7. Ахо Альфред В., Ульман Джеффри Д., Хопкрофт Джон Э. Структуры данных и алгоритмы. – М.: Вильямс, 2018. – 400 с.

8. Дж.Хайнеман, Г.Поллис, С.Стэнли. Алгоритмы. Справочник с примерами на C, C++, Java и Python, 2-е изд.: Пер. с англ. — СпБ.: ООО "Альфа-книга", 2017. — 432 с.

9. Farmonov, S., & Nazirov, A. (2023). C# DASTURLASH TILIDA GRAY KODI BILAN ISHLASH. B CENTRAL ASIAN JOURNAL OF EDUCATION AND INNOVATION (Т. 2, Выпуск 12, сс. 71–74). Zenodo.

10. Farmonov, S., & Toirov, S. (2023). NETDA DASTURLASHNING ZAMONAVIY TEXNOLOGIYALARINI O'RGANISH. *Theoretical aspects in the formation of pedagogical sciences*, *2*(22), 90-96

11. Raxmonjonovich, F. S. (2023). C# dasturlash tilida fayl operatsiyalari qo'llashning qulayliklari haqida. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, *2*(2), 439-446.

12. Raxmonjonovich, F. S. (2023). C# tilida ArrayList bilan ishlashning afzalliklari. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, *2*(2), 470-474.

13. Farmonov Sherzodbek Raxmonjonovich, & Rustamova Humoraxon Sultonbek qizi. (2024). C# DASTURLASH TILIDA TO'PLAMLAR BILAN ISHLASH. Ta'lim Innovatsiyasi Va Integratsiyasi, 11(10), 210–214. Retrieved from http://web-journal.ru/index.php/ilmiy/article/view/2480.

14. Raxmonjonovich, F. S., & Ravshanbek o'g'li, A. A. (2023). Zamonaviy dasturlash tillarining qiyosiy tahlili. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, 2(2), 430-433.

15. Farmonov, S., & Rasuljonova, Z. (2024). OB'EKTGA YO'NALTIRILGAN DASTURLASH ZAMONAVIY DASTURLASHNING ASOSI SIFATIDA. *Центральноазиатский журнал образования и инноваций*, *3*(1), 83-86.

16. Farmonov, S., & Ro'zimatov, J. (2024). DASTURLASH TILLARINI O'RGANISHDA ONLINE TA'LIM PLATFORMALARIDAN FOYDALANISH. Theoretical aspects in the formation of pedagogical sciences, 3(1), 5-10.

ISSN 3060-4567

17. Farmonov, S. R., & qizi Xomidova, M. A. (2024). C# VA JAVA DASTURLASH TILLARIDA FAYLLAR BILAN ISHLASHNING TURLI USULLARINING SAMARADORLIGI HAQIDA. *Zamonaviy fan va ta'lim yangiliklari xalqaro ilmiy jurnal*, *1*(9), 45-51.

18. Raxmonjonovich, F. S. (2024). C# VA MASHINA TILI. *Ta'lim innovatsiyasi va integratsiyasi*, *12*(1), 59-62.

19. Farmonov, S. (2023). C# DASTURLASH TILIDA GRAY KODI BILAN ISHLASH. *Центральноазиатский журнал образования и инноваций*, *2*(12 Part 2), 71-74.

20. Farmonov, S., & Jo'rayeva, M. (2023, December). DASTURLASHDA POLIMORFIZMNING AHAMIYATI. In *Международная конференция академических наук* (Vol. 2, No. 13, pp. 5-8).

21. Farmonov, S., & Usmonaliyev, U. (2024). O'ZBEKISTON RESPUBLIKASI IT SOHASINING RIVOJLANISH ISTIQBOLLARI. *Бюллетень педагогов нового Узбекистана*, *2*(1), 59-62.