

RABIN-KARP BIR TATBIG'I HAQIDA

Farmonov Sherzodbek Raxmonjonovich

*Farg'ona davlat universiteti amaliy matematika va
informatika kafedrası katta o'qituvchisi
farmonovsh@gmail.com*

Mahmudjonova Shoxistaxon Shuhratjon qizi

*Farg'ona davlat universiteti talabasi
shoxistamaxmudjonova3@gmail.com*

Annotatsiya: Ushbu maqolada Rabin-Karp algoritmi va uning matn ichida substring qidirishda qo'llanilishi yoritiladi. Algoritm hashing tamoyiliga asoslangan bo'lib, qidiruv jarayonini samarali va tezkor bajarishga imkon beradi. Rabin-Karp algoritmining asosiy g'oyasi — substringlarning xesh qiymatlarini hisoblash orqali moslikni aniqlashdir. Ushbu yondashuv, ayniqsa, katta hajmdagi matnlarda va bir nechta qidiruvlar talab etiladigan vaziyatlarda o'zining samaradorligini isbotlaydi. Maqolada algoritmning nazariy asosi, ishlash tamoyillari, matematik yondashuvi va turli amaliy qo'llanilish sohalari haqida batafsil ma'lumot beriladi.

Kalit so'zlar: Rabin-Karp algoritmi, substring qidirish, hashing, xesh funksiyasi, samaradorlik, matn qidirish, algoritmlar, plagiatni aniqlash, tabiiy tilni qayta ishlash (NLP), katta hajmdagi ma'lumotlar, texnologik rivojlanish, DNS qidiruv tizimlari, xesh kolliziyasi, kompyuter lingvistikasi, sun'iy intellekt, blokcheyn, Big Data.

Annotation: This article discusses the Rabin-Karp algorithm and its application in substring searching within a text. The algorithm is based on the hashing principle, allowing for efficient and fast execution of search processes. The main idea of the Rabin-Karp algorithm is to calculate the hash values of substrings to determine matches. This approach is particularly effective for large-scale texts and scenarios requiring multiple searches. The article provides detailed information about the theoretical foundation, working principles, mathematical approach, and various practical applications of the algorithm.

Keywords: Rabin-Karp algorithm, substring search, hashing, hash function, efficiency, text search, algorithms, plagiarism detection, natural language processing (NLP), large-scale data, technological development, DNS search systems, hash collision, computational linguistics, artificial intelligence, blockchain, Big Data.

Аннотация: В статье рассматривается алгоритм Рабина-Карпа и его применение для поиска подстрок в тексте. Алгоритм основан на принципе хэширования, что позволяет эффективно и быстро выполнять процессы поиска. Основная идея алгоритма Рабина-Карпа заключается в вычислении хэш-

значений подстрок для определения совпадений. Этот подход особенно эффективен для обработки больших текстов и задач, требующих множества поисков. В статье подробно рассматриваются теоретические основы, принципы работы, математический подход и различные практические области применения алгоритма.

Ключевые слова: алгоритм Рабина-Карпа, поиск подстрок, хэширование, хэш-функция, эффективность, текстовый поиск, алгоритмы, обнаружение плагиата, обработка естественного языка (NLP), большие данные, технологическое развитие, системы поиска DNS, хэш-коллизии, компьютерная лингвистика, искусственный интеллект, блокчейн, Big Data.

Matn ichida substringni qidirish muammosi dasturlashning asosiy masalalaridan biri bo'lib, uning samarali yechimi ko'plab sohalarda, jumladan, kompyuter lingvistikasi, ma'lumotlar tahlili va kiberxavfsizlikda muhim ahamiyatga ega. Katta hajmdagi matnlar ichida kerakli belgilar ketma-ketligini aniqlash uchun tezkor va ishonchli algoritmlarni qo'llash zarur. Matn qidirishning an'anaviy usullari, masalan, oddiy qidiruv yoki Brute Force usuli, kichik hajmdagi ma'lumotlar bilan ishlashda samarali bo'lsa-da, katta miqdordagi ma'lumotlarda ularning sekin ishlashi yoki resurslarni ko'p talab qilishi sababli yaroqsiz bo'lib qoladi. Bu borada Rabin-Karp algoritmi matn qidirish jarayonini samarali bajarish uchun mukammal yechim hisoblanadi. Ushbu algoritm, matematik hashing tamoyiliga asoslanib, xabarlarining yoki matn qismlarining xesh qiymatlarini taqqoslash orqali substringlarni tez va aniqlik bilan aniqlaydi. Klod Shenon va Uorren Viverning "Matematik kommunikatsiya nazariyasi"da ta'kidlanganidek, algoritmik samaradorlik katta hajmdagi ma'lumotlarni boshqarishda muhim ahamiyatga ega. Rabin-Karp algoritmi aynan shu tamoyilni amaliyotga tatbiq etib, zamonaviy dasturlash muammolariga samarali yechim taklif etadi.

Algoritmning ishlash g'oyasi shundan iboratki, u qidirilayotgan substring va matndagi har bir substringning xesh qiymatlarini hisoblab, ular orasidagi moslikni aniqlaydi. Hashingning tezkorligi tufayli, xabarlarining bevosita belgilar bo'yicha taqqoslanishi talab qilinmaydi, bu esa algoritmnini yanada samaraliroq qiladi. Shu bilan birga, algoritmnining murakkabligi $O(n + m)$, bu esa katta hajmdagi matnlarda ham optimal ishlashni ta'minlaydi.

Zamonaviy texnologiyalar rivoji bilan Rabin-Karp algoritmi nafaqat matn qidirishda, balki fayllarni bir-biriga moslashtirish, plagiatni aniqlash, kompyuter lingvistikasi va hatto sun'iy intellekt sohalarida ham qo'llanilmoqda. Dasturiy ta'minot yaratish jarayonida bu algoritm matn ichidagi substringlarni tez va aniq aniqlash uchun dasturchilar uchun qulay vositadir. Shu bilan birga, algoritmnining ba'zi cheklovlari, masalan, xesh qiymatlarining kolliziyasi (bir xil xesh qiymatiga ega turli xabarlar) kabi

holatlar muammoli bo'lishi mumkin. Biroq, bu muammo algoritmnining asosiy tamoyillari va samaradorligini o'zgartirmaydi, aksincha, uni yanada chuqurroq o'rganish zaruratini oshiradi.

Rabin-Karp algoritmining ishlash tamoyillari

Rabin-Karp algoritmi matn qidirish masalalarini samarali hal qilish uchun yaratilgan. Uning asosiy g'oyasi xesh funksiyalaridan foydalanish orqali matnning qidirilayotgan qator (substring) bilan mos kelishini aniqlashdir. Bunda xesh qiymatlari — qidiruv jarayonini tezlashtiruvchi vosita sifatida ishlatiladi. Xo'sh, xesh funksiyasi qanchalik ishonchli va u qanday ishlaydi?

Xesh funksiyasi — bu matnning kichik qismini sonli qiymatga aylantiruvchi matematik mexanizmdir. Masalan, "abc" substringining xesh qiymati 97, 98, 99 harflarining ASCII kodlariga asoslangan bo'lishi mumkin. Agar xesh qiymatlari mos kelsa, bu ikki substringning mosligini bildiradi. Ammo algoritm shu yerda to'xtab qolmaydi. Xesh qiymatlar mos kelgan taqdirda ham substringlarning har bir belgisi bo'yicha bevosita taqqoslash amalga oshiriladi. Bu usul xesh funksiyadagi kolliziyalar (bir xil xesh qiymatiga ega bo'lgan turli substringlar) muammosini hal qilishga yordam beradi.

Algoritmnining bosqichlari

Rabin-Karp algoritmi uch asosiy bosqichdan iborat:

1. Qidirilayotgan substring xesh qiymatini hisoblash.
2. Matndagi substringlarning xesh qiymatlarini iterativ hisoblash.
3. Moslik aniqlangan taqdirda, substringlarni bevosita taqqoslash.

Ushbu jarayon $O(n+m)$ murakkablik bilan ishlaydi, bunda n — matnning uzunligi, m — qidirilayotgan substringning uzunligi. Bu usul oddiy Brute Force algoritmidan ko'ra tezroq, chunki u substringlarni har bir bosqichda to'liq taqqoslamaydi.

Ammo savol tug'iladi: bu algoritm o'ta katta hajmdagi ma'lumotlar bilan ishlaganda ham samarali bo'la oladimi? Javob shundaki, algoritm katta hajmdagi ma'lumotlar uchun juda samarali, ammo xesh funksiyasi sifatining yuqoriligi bunga hal qiluvchi ta'sir ko'rsatadi. Yaxshi xesh funksiyasi kolliziyalarni kamaytiradi, bu esa qidiruv jarayonining aniqligini oshiradi.

Zamonaviy dasturlashdagi qo'llanilishi

Rabin-Karp algoritmi nafaqat matn qidirish, balki boshqa sohalarida ham keng qo'llaniladi. Quyidagi sohalar bunga misol bo'la oladi:

1. **Plagiatni aniqlash:**
 - o Algoritm hujjatlar orasidagi o'xshashliklarni aniqlashda ishlatiladi. Masalan, ikki maqolaning bir xil bo'limlarini tezda aniqlash uchun Rabin-Karp algoritmi qo'llanilishi mumkin.

○ Misol uchun, yirik matnlar to‘plamida plagiarizmni aniqlash faqat substring qidirishni emas, balki bir nechta substringlarni bir vaqtning o‘zida qidirishni talab qiladi. Rabin-Karp algoritmi bu talabni samarali bajaradi.

2. **DNS qidiruv tizimlari:**

○ Rabin-Karp algoritmi domen nomlarini tezkorlik bilan qidirish uchun ishlatiladi. Bu tizimda matn qidirish o‘rniga, domenlar xesh qiymatlar orqali taqqoslanadi.

3. **Kompyuter lingvistikasi va NLP (Natural Language Processing):**

○ Tabiiy tilni qayta ishlashda algoritm so‘zlarni matn ichida qidirishda, o‘xshash jummalarni aniqlashda va hatto til grammatikasini tahlil qilishda qo‘llaniladi.

Rabin-Karpning afzalliklari va cheklovlari

Rabin-Karp algoritmining asosiy afzalligi uning hashing tamoyiliga asoslanganligidir. Hashing yordamida xabarlarini bevosita taqqoslash ehtiyoji kamayadi, bu esa algoritmni sezilarli darajada tezlashtiradi. Bundan tashqari, bu algoritm bir nechta substringni bir vaqtning o‘zida qidirishga imkon beradi, bu esa uni katta hajmdagi qidiruv talab qilinadigan vaziyatlarda ayniqsa foydali qiladi.

Ammo, algoritmnining ayrim cheklovlari ham mavjud. Xesh funksiyasining tanlovi bu algoritmnining samaradorligiga bevosita ta’sir qiladi. Agar xesh funksiyasi yaxshi tanlanmagan bo‘lsa, xesh qiymatlarning kolliziyasi oshib ketishi mumkin, bu esa algoritmni sekinlashtiradi. Shuningdek, Rabin-Karp algoritmi real vaqt rejimida ishlatiladigan tizimlarda boshqa qidiruv algoritmlariga qaraganda nisbatan ko‘proq xotira talab qilishi mumkin.

Savol tug‘iladi: ”Algoritmnining bu cheklovlari uni amalda qo‘llashga qanchalik to‘sqinlik qiladi?” Javob shuki, zamonaviy dasturlash vositalari yordamida xesh funksiyalarining sifatini oshirish va kolliziyalarni kamaytirish mumkin, bu esa Rabin-Karp algoritmini yanada ishonchli qiladi.

Algoritmnining kelajagi: Rabin-Karpning zamonaviy yechimlardagi o‘rni

Bugungi texnologiyalar davrida Rabin-Karp algoritmi o‘z ahamiyatini yo‘qotmagan. Hashing tamoyiliga asoslangan bo‘lsa-da, algoritm sun’iy intellekt va mashinani o‘rganish (Machine Learning) texnologiyalari bilan uyg‘unlashmoqda. Masalan:

• Sun’iy intellekt tizimlari Rabin-Karp algoritmini tabiiy tilni qayta ishlashda (NLP) qo‘llab, katta hajmdagi matnlarni qismlarga ajratish va taqqoslashda foydalanmoqda.

• Blokcheyn texnologiyasida matn va ma’lumotlarni himoyalash va tezkor qidiruv jarayonlarini amalga oshirish uchun Rabin-Karpning xesh tamoyillari muhim o‘rin tutadi.

Masala: Berilgan asosiy matn (text) ichida berilgan substring (pattern) mavjudligini aniqlang va agar mavjud bo'lsa, u boshlanadigan indeksni qaytaring. Rabin-Karp algoritmi yordamida substring qidirish jarayonini amalga oshiring.

Masalaning Tahlili:

1. **Vazifa:** Matn ichida substringni izlash uchun Rabin-Karp algoritmini qo'llash kerak. Bu algoritmnining asosiy afzalligi — substringlarni hashing orqali tezda taqqoslashdir.

2. **Yondashuv:**

○ **Xesh funksiyasi:** Substringni xesh qiymatga o'zgartirish, bu taqqoslash jarayonini tezlashtiradi.

○ **Oyna algoritmi (sliding window):** Matnni substring uzunligida harakatlantirish va har bir qism uchun xesh qiymatni hisoblash.

○ **Kolliziya nazorati:** Xesh qiymatlar mos kelganda, belgilarni qo'shimcha ravishda taqqoslash.

3. **Murakkablik:**

○ $O(n + m)$ — bu yerda n asosiy matn uzunligi va m substring uzunligi. Bu murakkablik oddiy qidiruv (Brute Force) algoritmidan yaxshiroqdir.

C# Kod:

using System;

using System.Collections.Generic;

class RabinKarpSearch

{

static **int** Prime = 101;

// Xesh funksiyasi

static **long** CalculateHash(string str, **int** end)

{

long hash = 0;

for (**int** i = 0; i < end; i++)

{

hash += (**long**)(str[i] * Math.Pow(Prime, i));

}

return hash;

}

// Hashni qayta hisoblash (sliding window uchun)

static **long** RecalculateHash(string str, **int** oldIndex, **int** newIndex, **long** oldHash,

```

int patternLen)
{
    long newHash = oldHash - str[oldIndex];
    newHash /= Prime;
    newHash += (long)(str[newIndex] * Math.Pow(Prime, patternLen - 1));
    return newHash;
}

// Substringlarni taqqoslash (xesh qiymatlar mos kelganda)
static bool CheckEquality(string str1, string str2, int start1, int end1, int start2,
int end2)
{
    if (end1 - start1 != end2 - start2)
        return false;

    while (start1 <= end1 && start2 <= end2)
    {
        if (str1[start1] != str2[start2])
            return false;

        start1++;
        start2++;
    }
    return true;
}

// Rabin-Karp algoritmi
public static List<int> RabinKarp(string text, string pattern)
{
    int textLen = text.Length;
    int patternLen = pattern.Length;
    long patternHash = CalculateHash(pattern, patternLen);
    long textHash = CalculateHash(text, patternLen);
    List<int> result = new List<int>();

    for (int i = 0; i <= textLen - patternLen; i++)
    {
        if (patternHash == textHash && CheckEquality(text, pattern, i, i +
patternLen - 1, 0, patternLen - 1))

```

```

    {
        result.Add(i);
    }
    if (i < textLen - patternLen)
    {
        textHash = RecalculateHash(text, i, i + patternLen, textHash, patternLen);
    }
}
return result;
}

static void Main(string[] args)
{
    string text = "abracadabra";
    string pattern = "abra";

    List<int> result = RabinKarp(text, pattern);

    if (result.Count > 0)
    {
        Console.WriteLine("Pattern found at indices: " + string.Join(", ", result));
    }
    else
    {
        Console.WriteLine("Pattern not found in the text.");
    }
}
}

```

Kodning tushuntirishi:

1. **Xesh hisoblash:** CalculateHash funksiyasi substring yoki matnning bir qismi uchun xesh qiymatni hisoblaydi. Bu hashning asosiy tuzilishi Prime asosida bo'ladi.
2. **Hashni qayta hisoblash:** RecalculateHash funksiyasi oyna harakatlanayotganda yangi xeshni hisoblash uchun ishlatiladi. Eski qiymatni yangilash orqali hisoblash tezlashadi.
3. **Taqqoslash:** Xesh qiymatlar bir xil bo'lsa, substringning haqiqiy mosligini CheckEquality yordamida tekshiramiz.
4. **Natija:** Agar moslik aniqlansa, substringning boshlang'ich indeksi natijalar ro'yxatiga qo'shiladi.

Natija:

Agar matn "abracadabra", substring esa "abra" bo'lsa, dastur chiqishi:

Pattern found at indices: 0, 7

Rabin-Karp algoritmi matn qidirishning zamonaviy va samarali yechimlaridan biri sifatida o'zining nazariy hamda amaliy ahamiyatini saqlab kelmoqda. Uning asosida yotgan hashing tamoyili, axborotni tez va aniqlik bilan tahlil qilishni ta'minlab, katta hajmdagi ma'lumotlar bilan ishlash imkoniyatlarini sezilarli darajada kengaytiradi. Ushbu algoritm, bir vaqtning o'zida bir nechta substringlarni qidirish, o'xshashliklarni aniqlash va katta hajmdagi ma'lumotlarni boshqarish kabi muammolarni hal qilishda muhim rol o'ynaydi.

Biroq algoritm faqat texnik samaradorligi bilan emas, balki dasturchilar va tadqiqotchilar uchun yaratgan yangi imkoniyatlari bilan ham ajralib turadi. Rabin-Karp nafaqat algoritm sifatida, balki turli dasturiy tizimlarda keng qo'llanilishi mumkin bo'lgan moslashuvchan yondashuvni taklif etadi. Masalan, plagiatni aniqlash, DNS tizimlari, tabiiy tilni qayta ishlash va fayl mosligini aniqlash kabi zamonaviy masalalarda uning foydaliligini isbotlash mumkin.

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. С#. 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 С, С++, 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. В 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. 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>.

12. 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*.

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

14. 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*.

15. 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*.

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

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

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

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