MINIMAL BOG‘LANISH DARAXTINI TOPISHDA PRIM ALGORITMIDAN FOYDALANISH

Farmonov Sherzodbek Raxmonjonovich
Farg‘ona Davlat Universiteti Amaliy matematika va informatika kafedrasi katta o’qituvchisi
farmonovsh@gmail.com
Ibrohimjonova Dilraboxon Ibrohimjon qizi
Farg`ona davlat universiteti Amaliy matematika yo`nalishi 2-kurs talabasi
Servisec101@gmail.com

Annotatsiya. *Ushbu maqolada Prim algoritmi minimal bog`lanish daraxtini topish uchun eng samarali usullardan biri ekanligi, uning ishlash prinsipi, afzalliklari ko`rib chiqiladi. Shu bilan birga prim algoritmi qanday sohalarda qo`llanilishi tahlil qilinadi. Prim algoritmiga masala tuzilib, C# dasturlash tilida ishlanadi va tahlil qilinadi.*
**Kalit so`zlar:** *Prim algoritmi, graf nazariyasi, Tarmoq dizayni, Kompyuter tarmoqlari, minimal bog`lanish daraxti, Transport tizimlari, Robototexnika.*

Abstract. *In this article, Prim's algorithm is one of the most effective methods for finding a minimum spanning tree, its working principle, and its advantages are considered. At the same time, it will be analyzed in what fields the prim algorithm is used. A problem for Prim's algorithm is formulated, processed and analyzed in C# programming language.*
Keywords: *Prim's algorithm, graph theory, Network design, Computer networks, minimal connection tree, Transport systems, Robotics.*

Аннотация. *В данной статье алгоритм Прима — один из наиболее эффективных методов поиска минимального остовного дерева, рассмотрен принцип его работы и преимущества. При этом будет проанализировано, в каких сферах используется алгоритм прим. Задача для алгоритма Прима формулируется, обрабатывается и анализируется на языке программирования C#.*
Ключевые слова: *Алгоритм Прима, теория графов, Проектирование сетей, Компьютерные сети, минимальное дерево связей, Транспортные системы, Робототехника.*

Prim algoritmi, graf nazariyasida muhim o‘rin tutuvchi algoritmlardan biridir. U minimal bog‘lanish daraxtini (MST) topish uchun ishlatiladi va ko‘plab amaliy muammolarni hal qilishda qo‘llaniladi. Prim algoritmi 1930-yillarda Chex matematikasi Czech mathematician Vojtěch Jarník tomonidan ishlab chiqilgan bo‘lib, keyinchalik Robert C. Prim tomonidan takomillashtirilgan. Ushbu algoritm, bog‘lanish graflarida minimal og‘irlikdagi yo‘llarni aniqlashga yordam beradi, bu esa transport, telekommunikatsiya va boshqa sohalarda muhim ahamiyatga ega.

Prim algoritmi, o‘zining soddaligi va samaradorligi bilan ajralib turadi. U, asosan, og‘irliklari berilgan grafdagi eng kichik og‘irlikli qirralarni tanlash orqali minimal bog‘lanish daraxtini quradi. Ushbu maqolada Prim algoritmining asosiy tamoyillari, uning ishlash prinsipi, vaqt murakkabligi va amaliy qo‘llanilishlari haqida batafsil ma’lumot beriladi. Shuningdek, Prim algoritmining afzalliklari va kamchiliklari ham tahlil qilinadi, bu esa uni boshqa MST algoritmlari bilan solishtirish imkonini beradi.

Endi esa prim algoritmining ishlsh printsipi va uning amalga oshirilishi uchun zarur bo`lgan metodlar haqida batafsil bayon qilinadi.

Prim algoritmi, bir grafning minimal bog‘lanish daraxtini topish uchun quyidagi asosiy bosqichlardan iborat:

1. Boshlang‘ich nuqtani tanlash: Algoritm biron bir boshlang‘ich nuqtani (vertex) tanlaydi. Bu nuqta MST qurilishining boshlanishi bo‘ladi. Umuman olganda, har qanday nuqta tanlanishi mumkin, lekin ko‘pincha graflarning o‘ziga xos xususiyatlariga qarab eng kichik og‘irlikli nuqtani tanlash tavsiya etiladi.

2. Qirralarni tanlash: Tanlangan nuqtadan boshlab, grafdagi barcha qo‘shni nuqtalar (ya'ni, tanlangan nuqtaga bog‘langan nuqtalar) va ularning og‘irliklari hisobga olinadi. Har bir qo‘shni nuqtaga bog‘langan qirralarning og‘irliklari saqlanadi.

3. Minimal qirralarni aniqlash: Har safar MST ga yangi nuqta qo‘shilganda, eng kichik og‘irlikka ega qirra tanlanadi. Bu jarayon davomida MST ga qo‘shilgan nuqtalar va qirralar saqlanadi.

4. Qayta tiklash: Tanlangan qirra orqali yangi nuqta MST ga qo‘shilgandan so‘ng, yangi qo‘shni nuqtalar va ularning og‘irliklari qayta hisoblanadi. Bu jarayon, MST to‘liq qurilgunga qadar davom etadi.

5. Natijani chiqarish: Algoritm to‘liq qurilgan minimal bog‘lanish daraxtini chiqaradi. Ushbu daraxtda barcha nuqtalar bog‘langan bo‘lib, umumiy og‘irlik minimal qiymatga ega bo‘ladi.

Algoritmning murakkabligi: Prim algoritmining vaqt murakkabligi tanlangan ma'lumotlar tuzumi va usulga bog‘liq. Agar oddiy massivlar ishlatilsa, vaqt murakkabligi O(V^2) bo‘ladi, bu yerda V - nuqtalar soni. Agar prioritet navbatdan foydalanilsa (masalan, min-heap),vaqt murakkabligi O(E log V) ga tushadi, bu yerda E - qirralar soni.

Prim algoritmining minimal bog`lash daraxtini aniqlashdagi samaradorligi va amaliy qo`llanilishini o`rganamiz. Algoritmning asosiy natijalari quyidagilardan iborat:

1. Samaradorlik: Prim algoritmi, turli xil graf tuzilmalarida, masalan, to‘g‘ri va og‘irliklar bilan bog‘liq bo‘lgan graflarda yuqori samaradorlikni ko‘rsatdi. Algoritmning vaqt murakkabligi O(E log V) ga teng bo‘lib, bu uni katta graflar bilan ishlashda samarali qiladi.

2. Minimal og‘irlik: Algoritm orqali qurilgan minimal bog‘lanish daraxti har doim eng kichik og‘irlikka ega qirralarni tanlash orqali hosil qilinadi. Olingan MST, berilgan grafning barcha nuqtalarini bog‘lagan holda, umumiy og‘irlikni minimallashtirishga erishadi.

3. Moslashuvchanlik: Prim algoritmi turli xil ma'lumotlar tuzilmalarida (masalan, prioritet navbatlar, massivlar) qo‘llanilishi mumkin. Bu uning turli sohalarda, jumladan, tarmoq dizayni va transport tizimlarida keng qo‘llanilishiga imkon beradi.

4. Amaliy misollar: Tadqiqot davomida Prim algoritmining amaliy misollari keltirildi. Masalan, kompyuter tarmoqlarini loyihalashda, transport yo‘llarini optimallashtirishda va telekommunikatsiya tizimlarida qo‘llanilishi natijasida xarajatlar va resurslardan samarali foydalanish ta'minlandi.

5. Ta'sirchanlik: Prim algoritmi o‘zining oddiyligi va samaradorligi bilan ajralib turadi. Bu algoritmning oson tushunilishi va amalga oshirilishi, dasturchilar va muhandislar tomonidan keng qo‘llanilishiga olib keladi.

MASALA: Berilgan grafning bog`lanish xarajatlari matritsasini hisobga olib, prim algoritmi yordamida minimal bog`lanish daraxtini toping.

Masalaning shartlari:

1. Grafning tugunlari(vertex) soni V ga teng.
2. Har bir tugun o`rtasidagi bog`lanish xarajatlarini ifodalovchi VxV matritsa beriladi.
3. Bog`lanish xarajatlari ixtiyoriy bo`lishi mumkin, lekin o`z-o`zidan bog`lanish xarajatlari nolga teng bo`ladi.

Masalani C# da yechimi

```
using System;
class Program
{
    const int V = 5; // Tugunlar soni

    // Minimal bog'lanish daraxtini topish uchun Prim algoritmi
    static void Prim(int[,] graph)
    {
        int[] parent = new int[V]; // MST ni saqlash uchun
        int[] key = new int[V];    // Minimal xarajatlar
        bool[] mstSet = new bool[V]; // MST ga qo'shilgan tugunlar

        // Barcha tugunlarni boshida chetga chiqaramiz
        for (int i = 0; i < V; i++)
        {
            key[i] = int.MaxValue; // Xarajatlarni cheksizga tenglaymiz
            mstSet[i] = false;     // Hech bir tugun MST ga qo'shilmagan
        }

        key[0] = 0; // Birinchi tugunni tanlaymiz
        parent[0] = -1; // Birinchi tugun MST ning ildizi

        for (int count = 0; count < V - 1; count++)
```

```
    {
        // Minimal kalitni tanlang
        int u = MinKey(key, mstSet);

        mstSet[u] = true; // Tugunni MST ga qo'shamiz

        // Qo'shni tugunlar o'rtasidagi bog'lanish xarajatlarini yangilaymiz
        for (int v = 0; v < V; v++)
        {
            if (graph[u, v] != 0 && !mstSet[v] && graph[u, v] < key[v])
            {
                parent[v] = u;
                key[v] = graph[u, v];
            }
        }
    }

    PrintMST(parent, graph);
}
// Minimal kalitni topish funksiyasi
static int MinKey(int[] key, bool[] mstSet)
{
    int min = int.MaxValue;
    int minIndex = -1;

    for (int v = 0; v < V; v++)
    {
        if (!mstSet[v] && key[v] < min)
        {
            min = key[v];
            minIndex = v;
        }
    }
    return minIndex;
}

// MST ni chop etish funksiyasi
static void PrintMST(int[] parent, int[,] graph)
{
    Console.WriteLine("Tugunlar orasidagi bog'lanishlar:");
    for (int i = 1; i < V; i++)
```

```
        {
            Console.WriteLine($"Tugun {parent[i]} - Tugun {i}: {graph[i, parent[i]]}");
        }
    }

    static void Main()
    {
        // Bog'lanish xarajatlari matritsasi
        int[,] graph = new int[,] {
            { 0, 2, 0, 6, 0 },
            { 2, 0, 3, 8, 5 },
            { 0, 3, 0, 0, 7 },
            { 6, 8, 0, 0, 9 },
            { 0, 5, 7, 9, 0 }
        };
        Prim(graph);
    }
}
```

Prim algoritmi minimal bog‘lanish daraxtini (MST) aniqlashda samaradorligi va amaliy qo‘llanilishi bilan tanilgan. Biroq, uning qo‘llanilishi va samaradorligi haqida bir qancha muhokamalar mavjud.

1. Afzalliklari:

Samaradorlik: Prim algoritmi o‘zining O(E log V) vaqt murakkabligi bilan katta graflarda tez ishlash imkoniyatini beradi. Bu, ayniqsa, tarmoq dizayni va transport tizimlarida katta ahamiyatga ega.

Oddiylik: Algoritmning oddiy tushunilishi va amalga oshirilishi ko‘plab dasturchilar va muhandislar tomonidan afzal ko‘riladi. Bu esa uning ta'lim jarayonida keng qo‘llanilishiga olib keladi.

Ishonchlilik: Prim algoritmi har doim to‘g‘ri natija beradi, ya'ni eng kichik og‘irlikka ega bo‘lgan bog‘lanish daraxtini hosil qiladi. Bu uning ishonchli algoritm sifatida tan olinishi uchun asosiy sababdir.

2. Kamchiliklari:

Grafning zichligi: Prim algoritmi zich graflarda yaxshi ishlasa-da, kam zichlikdagi graflarda boshqa algoritmlar, masalan, Kruskal algoritmi, samaraliroq bo‘lishi mumkin. Bu holatda, Prim algoritmi ko‘proq vaqt talab qilishi mumkin.

Prioritet navbatning zarurati: Algoritmning samarali ishlashi uchun prioritet navbatni ishlatish zarur. Bu esa qo‘shimcha murakkabliklarni keltirib chiqarishi mumkin, ayniqsa, dasturiy ta'minot ishlab chiqishda.

O'zgaruvchan graflar: Agar graf o'zgaruvchan bo'lsa (masalan, qirralarning og'irliklari o'zgarayotgan bo'lsa), Prim algoritmini qayta ishlatish zarurati paydo bo'ladi. Bu esa real vaqt rejimida ishlashda muammolarni keltirib chiqarishi mumkin.

3. Taqqoslash:

Kruskal algoritmi: Kruskal algoritmi ham minimal bog‘lanish daraxtini aniqlashda keng qo‘llaniladi. U qirralarni tartiblash orqali ishlaydi va kam zichlikdagi graflarda yanada samarali bo'lishi mumkin. Biroq, Kruskal algoritmi graflarni tartiblashda ko‘proq vaqt talab qilishi mumkin.

Boruvka algoritmi: Boruvka algoritmi ham MSTni aniqlashda ishlatiladi va ba'zi hollarda Prim va Kruskal algoritmlaridan samaraliroq bo'lishi mumkin. U bir necha bog‘lanish daraxtlarini bir vaqtning o‘zida hosil qilib, ular orasidagi eng kichik qirralarni tanlaydi.

4. Kelajak istiqbollari:

Prim algoritmini yanada rivojlantirish va optimallashtirish uchun yangi yondashuvlar va metodlarni o‘rganish zarur. Masalan, parallel hisoblash usullarini qo‘llash orqali algoritmning ishlash tezligini oshirish mumkin. Shuningdek, o'zgaruvchan graflar uchun dinamik MST algoritmlarini ishlab chiqish ham muhimdir. Bu esa real vaqt rejimida graf tuzilmalari bilan ishlashni osonlashtiradi. Prim algoritmi quyidagi sohalarda qo`llaniladi:

1. Tarmoq dizayni: Prim algoritmi kompyuter tarmoqlarini, telefon tarmoqlarini va boshqa aloqa tizimlarini loyihalashda ishlatiladi. Tarmoqning minimal narxda bog‘lanishini ta'minlash uchun qirralarning og‘irliklari tarmoq qurilish xarajatlariga mos kelishi mumkin.

2. Transport tizimlari: Transport tarmoqlarini optimallashtirish, masalan, yo‘l yoki temir yo‘l tarmoqlarini loyihalashda, Prim algoritmi foydalidir. U eng qisqa va iqtisodiy yo‘llarni aniqlashda yordam beradi.

3. Elektr energiyasi tarqatish: Elektr energiyasi tarqatish tarmoqlarini loyihalashda ham Prim algoritmi qo‘llaniladi. U energiya manbalaridan iste'molchilarga eng samarali va arzon yo‘llarni aniqlashga yordam beradi.

4. Geografik axborot tizimlari (GAT): Prim algoritmi GATlarda, masalan, xaritalarda bog‘lanishlarni aniqlash va optimallashtirishda ishlatiladi. U yer yuzasidagi ob'ektlar o'rtasidagi eng qisqa yo'llarni aniqlashda yordam beradi.

5. Telekommunikatsiya: Telekommunikatsiya tarmoqlarida signal uzatishni optimallashtirish va qamrov hududlarini kengaytirishda Prim algoritmi qo‘llaniladi.

6. Robototexnika: Robotlar uchun yo‘l harakatini rejalashtirishda, ularning harakat yo‘llarini optimallashtirishda Prim algoritmi foydalidir.

7. Kompyuter grafikasi: Kompyuter grafikalarida ob'ektlar o‘rtasidagi bog‘lanishlarni aniqlash va minimal xarajatga ega bo‘lgan grafik tuzilmalarni yaratishda Prim algoritmidan foydalaniladi.

8. Ma'lumotlar uzatish: Ma'lumotlar uzatish jarayonlarida, masalan, ma'lumotlar bazalarini optimal tarzda bog‘lashda Prim algoritmi yordamida bog‘lanishlarni aniqlash mumkin.

Prim algoritmi-minimal bog`lanish daraxtini topishning eng samarali usullaridan biri xisoblanadi. Prim algoritmi tarmoq dizayni, transport tizimlari, robototexnika, kompyuter grafikasi vas hu kabi sohalarda qo`llaniladi.

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). Array ma’lumotlar tizimini talabalarga o’qitishda Blockchain metodidan foydalanish. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 541-547.
12. Raxmonjonovich, F. S. (2023). Dasturlashda interfeyslardan foydalanishning ahamiyati. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 425-429.
13. Raxmonjonovich, F. S. (2023). Dasturlashda obyektga yo’naltirilgan dasturlashning ahamiyati. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 434-438.
14. Raxmonjonovich, F. S. (2023). Dasturlash tillarida fayllar bilan ishlash mavzusini Blended Learning metodi yordamida o'qitish. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 464-469.
15. Raxmonjonovich, F. S. (2023). DASTURLASHDA ISTISNOLARNING AHAMIYATI. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 475-481.
16. Raxmonjonovich, F. S. (2023). Dasturlashda abstraksiyaning o’rni. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 482-486.

17. 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.

18. 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.

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

20. 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.

21. 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.

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

23. 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.

24. 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.

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

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

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

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

29. Raxmonjonovich, F. S., & Xasan o'g'li, X. O. (2023). DASTURLASHDA SANA VA VAQTLAR BILAN ISHLASH. Ta'lim innovatsiyasi va integratsiyasi, 11(11), 3-6.