GRAFDAGI BARCHA JUFTLIKLAR ORASIDAGI ENG QISQA YO‘LLARNI TOPISH JOHNSON ALGORITMI

Farmonov Sherzodbek Raxmonjonovich
Farg’ona davlat universiteti amaliy matematika va informatika kafedrasi katta o’qituvchisi
farmonovsh@gmail.com
Turg’unova Surayyoxon Ulug’bek qizi
Farg’ona davlat universiteti talabasi
surayyoturgunova16@gmail.com

Anotatsiya:Johnson algoritmi qisqa yo‘llarni hisoblashda ishlatiladigan samarali metodlardan biridir.Bu algoritm, grafdagi barcha juftliklar orasidagi eng qisqa yo‘llarni topish uchun mo‘ljallangan bo‘lib, u o‘zining yuqori samaradorligi bilan ajralib turadi. Ushbu algoritmning afzalligi, u grafning salohiyatli qisqa yo‘llarini hisoblashda tezkorlikni va samaradorlikni ta’minlaydi, ayniqsa, yirik grafalar bilan ishlashda.

Kalit so’zlar: Johnson algoritmi, qisqa yo’llarni hisoblash, graf, eng qisqa yo’llar, samaradorlik, qayta tuzish, salohiyatli qisqa yo’llar, tezkorlik, yirik graflar, afzalliklari, cheklovlari, amaliy qo’llanilishi, samaradorlikni baholash.

Annotation: Johnson's algorithmis one of the efficient methods used forcalculating shortest paths.This algorithm is designed to find the shortest paths between all pairs of vertices in a graph and stands out due to its high efficiency. The advantage of this algorithm is that it ensures speed and efficiency when calculating potential shortest paths in a graph, especially when working with large graphs.

Keywords: Johnson’s algorithm, shortest path calculation, graph, shortest paths, efficiency, reconstruction, potential shortest paths, speed, large graphs, advantages, limitations, practical applications, efficiency evalution.

Аннотация: Алгоритм Джонсона является одним из эффективных методов для вычисления кратчайших путей. Этот алгоритм предназначен для нахождения кратчайших путей между всеми парами вершин в графе и выделяется своей высокой эффективностью. Преимущество данного алгоритма заключается в том, что он обеспечивает скорость и эффективность при вычислении потенциальных кратчайших путей в графе, особенно при работе с большими графами.

Ключевые слова: Алгоритм Джонсона, вычисление кратчайших путей, граф, кратчайшие пути, эффективность, перестройка, потенциальные кратчайшие пути, скорость, большие графы, преимущества, ограничения, практическое применение, оценка эффективности.

Johnson algoritmi haqida qisqacha ma’lumot: Johnson algoritmi- graf nazariyasida eng qisqa yo‘llarni toppish masalalarini yechish uchun ishlatiladigan

samarali algoritmlardan biridir. U, ayniqsa, yo‘llarning og‘irliklari butun sonlar bilan belgilangan yo‘llar uchun qulay hisoblanadi. Algoritmning asosiy maqsadi, graflar orasidagi barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llarni topishdir. Dijkstra algoritmining takomillashtirilgan shakli bo‘lgan Johnson algoritmi, O(n^2 * log n) vaqt murakkabligi bilan ishlaydi va bu uni katta miqdordagi nuqtalar va qirralarga ega bo‘lgan grafalarda samarali yechimga aylantiradi.

Johnson algoritmi ikki bosqichda ishlaydi. Birinchi bosqichda, qo‘shimcha manba nuqtasi qo‘shilib, Bellman-Ford algoritmi yordamida barcha juft nuqtalar orasidagi eng qisqa yo‘llar hisoblanadi. Ikkinchi bosqichda esa, Dijkstra algoritmi yordamida asl grafdagi barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llar hisoblanadi, bunda birinchi bosqichdan olingan natijalar asosida graflarning og‘irliklari mos ravishda o‘zgartiriladi.

Ushbu maqolada, Johnson algoritmining ishlash prinsipi, uning afzalliklari va dasturiy ta'minotdagi qo‘llanilishi haqida batafsil ma'lumot beriladi. Algoritmning samarali ishlash prinsiplari uni turli sohalarda, masalan, transport tizimlarida yoki kompyuter tarmoqlarida qo‘llash uchun keng imkoniyatlar yaratadi.

Johnson algoritmi (Johnson's algorithm) qisqacha, barcha juft nuqtalar orasidagi eng qisqa yo'ldan o‘tishni topish uchun ishlatiladigan algoritmdir. U asosan yo‘llar (shortest paths) masalalarini yechishda qo‘llanadi. Johnson algoritmi, grafikda mavjud bo‘lgan barcha juft nuqtalar o‘rtasidagi eng qisqa masofalarni topishga mo‘ljallangan.

Johnson algoritmi asosan quyidagi bosqichlarga bo‘linadi:

1. Harfli (relabeling) o‘zgarishlar bosqichi:

Har bir nuqtaga qo‘shimcha og‘irlik kiritiladi, bu esa grafdagi hech qanday negative og‘irliklar bo‘lmasligini ta'minlaydi.

Bu bosqich Bellman-Ford algoritmi yordamida amalga oshiriladi. Bu algoritm har bir nuqta uchun virtual "potensial" qiymatlarni hisoblab chiqadi.

2. Dijkstra algoritmini qo‘llash:

Har bir nuqtadan boshqa nuqtalarga bo‘lgan eng qisqa yo‘llarni hisoblash uchun Dijkstra algoritmi ishlatiladi.

Har bir nuqta uchun Dijkstra algoritmi, Dijkstra ishlatishga to‘g‘ri keladigan grafning modifikatsiya qilingan versiyasida amalga oshiriladi. Bu yerda, har bir nuqtaga kiritilgan potensial qiymatlar asosida to‘g‘ri masofalar hisoblanadi.

3. Natijalarni qayta hisoblash:

Natijaviy eng qisqa yo‘llar original grafiga qaytariladi, ya'ni har bir yo‘l uchun asosiy og‘irliklar va potensiallar olinadi.

Shundan so‘ng, barcha juft nuqtalar orasidagi eng qisqa yo‘llar aniqlanadi.

Shu tarzda, Johnson algoritmi, grafdagi barcha juft nuqtalar orasidagi eng qisqa yo‘llarni toppish uchun kuchli va samarali usul bo‘lib, grafda negative og‘irliklar bo‘lishi mumkin bo‘lsa ham ishlaydi.

Johnson algoritmining asosiy prinspi: Johnson algoritmining asosiy prinsipi- bu barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llarni topish. Bu algoritm grafda negative og‘irliklar bo‘lishi mumkin bo‘lgan hollarda ishlaydi. Algoritm quyidagi prinsipga asoslanadi:

1. Potensiallarni hisoblash (relaxation):

Algoritmning birinchi qadamida Bellman-Ford algoritmi ishlatiladi. Bu bosqichda, har bir nuqtadan barcha boshqa nuqtalarga eng qisqa yo‘llarni hisoblash uchun bir "virtual" potensial qiymati kiritiladi. Potensiallar grafdagi negative og‘irliklarni bartaraf etish uchun kerak bo‘ladi.

Bellman-Ford algoritmi orqali har bir nuqta uchun potensial qiymatlar hisoblanadi. Potensiallar grafni o‘zgartirmasdan, masofalar hisobi uchun yordam beradi va negative og‘irliklarning kiritilishiga imkon beradi.

2. Dijkstra algoritmini ishlatish:

Potensiallar hisoblangandan so‘ng, har bir nuqtadan boshqa nuqtalarga eng qisqa yo‘llarni toppish uchun Dijkstra algoritmi ishlatiladi. Bu bosqichda grafdagi har bir nuqtaga kiritilgan potensial qiymatlar yordamida, Dijkstra algoritmi eng qisqa yo‘llarni hisoblaydi.

Potensial qiymatlar og‘irliklarni yangilashga yordam beradi va grafda negative og‘irliklar bo‘lsa ham to‘g‘ri natijalarga olib keladi.

3. Natijaviy yo‘llarni qayta tiklash:

Dijkstra algoritmi orqali hisoblangan eng qisqa yo‘llar asl grafiga qaytariladi. Yani, potensiallar yordamida hisoblangan yo‘llar asl grafiga ko‘chiriladi, shundan so‘ng haqiqiy eng qisqa yo‘llar olinadi.

Johnson algoritmi potensiallarni hisoblash va Dijkstra algoritmini qo‘llashni birlashtiradi, shuning uchun u negative og‘irliklarga ega bo‘lgan grafda ham samarali ishlaydi.

Asosiy tamoyil: Negativ og‘irliklarni o‘zgartirish uchun potensiallarni hisoblash.

Johnson algoritmining muhim jihatlari quyidagilar:

1. Grafda negative og‘irliklarga qarshi chidamli: Johnson algoritmi grafda negative og‘irliklar mavjud bo‘lishi mumkin, lekin ularni hisoblashda yuzaga keladigan muammolarni bartaraf etadi. Algoritm potensiallar yordamida negative og‘irliklarni xavfsiz tarzda boshqaradi, shuning uchun grafdagi barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llarni to‘g‘ri hisoblaydi.

2. Barcha juft nuqtalar orasidagi eng qisqa yo‘llarni topadi: Algoritm barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llarni hisoblashga mo‘ljallangan. Bu, masalan, Floyd-Warshall algoritmi bilan o‘xshash bo‘lsa-da, Jonson algoritmi katta graflarda samaraliroq ishlaydi.

3. Effektivlik: Johnson algoritmi O(V^2 * log(V) + VE) murakkablikka ega bo‘lib, bu uni katta va kompleks grafiklar uchun samarali qiladi. Bu murakkablik, ya'ni

V – nuqtalar soni, E – qirralar soni bilan bog‘liq, shuningdek, potensiallarni hisoblash va Dijkstra algoritmi yordamida eng qisqa yo‘llarni topishning kombinatsiyasidan kelib chiqadi.

4. Dijkstra algoritmining ishlatilishi: Johnson algoritmi Dijkstra algoritmini har bir nuqtadan boshlanib, qolgan nuqtalar o‘rtasidagi eng qisqa yo‘llarni hisoblash uchun ishlatadi. Dijkstra algoritmi grafdagi negative og‘irliklarni hisobga olmaydi, ammo Jonson algoritmi orqali grafni oldindan o‘zgartirish (potensiallarni hisoblash) orqali buni hal qiladi.

5. Flexibilite: Johnson algoritmi turli xil grafiklar uchun moslashuvchan bo‘lib, agar grafda yo‘qotilgan yoki o‘zgargan qirralar bo‘lsa, ularni qayta hisoblash imkonini beradi. Bu algoritmni dinamik tarmoqni boshqarish yoki grafikni muntazam ravishda yangilab borish uchun foydalidir.

6. Optimal yechim: Johnson algoritmi barcha juft nuqtalar o‘rtasidagi eng qisqa yo‘llarni toppish uchun optimal yechimni taqdim etadi. U Floyd-Warshall algoritmi bilan taqqoslaganda, katta graflarda tezroq ishlashi mumkin, chunki har bir nuqtadan faqat Dijkstra algoritmi ishlatiladi, bu esa ko‘proq samarali hisoblanadi.

Johnson algoritmi grafdagi barcha juft nuqtalar orasidagi eng qisqa yo‘llarni topishda yuqori samaradorlik va keng qamrovli yondashuvni ta'minlaydi. Uning asosiy afzalliklari negative og‘irliklar bilan ishlash qobiliyati, samaradorlik, va kengaytirilgan moslashuvchanlikdir.

Johnson algoritmining internet tarmog’idagi afzalliklari: Jonson algoritmi asosan og‘irliklari musbat bo‘lgan yo‘llar bilan bog‘liq yo‘nalgan grafalarda ishlatiladi. Internet tarmog‘ida yoki boshqa tarmoqlarni optimallashtirishda, masalan, ma’lumot uzatish yo‘llarini tanlashda ushbu algoritmning bir qator afzalliklari mavjud:

1. Ko‘p juftlar o‘rtasida eng qisqa yo‘lni hisoblash: Johnson algoritmi, agar grafning barcha juft nuqtalari orasidagi eng qisqa yo‘llarni topish kerak bo‘lsa, samarali ishlaydi. Bu tarmoqda bir nechta nuqtalar orasidagi optimal marshrutlarni aniqlash uchun foydalidir.

2. Og‘irliklari musbat bo‘lgan grafalarda samarali: Johnson algoritmi, faqat musbat og‘irliklarga ega bo'lgan yo‘llar bilan ishlaydi. Bu esa Internet tarmoqlarida ko‘p holatlarda uchraydi, chunki ko'plab tarmoqda uzatish narxlari yoki tezliklar ijobiy qiymatlar bilan belgilanadi.

3. Foydalanish uchun ancha oson: Algoritmning ishlash prinsipi nisbatan oddiy bo'lib, Dijkstra algoritmini optimallashtirgan shaklini qo‘llaydi, bu esa uni tarmoq va boshqa real dunyo masalalarida ishlatish uchun qulay qiladi.

4. O‘zgaruvchan tarmoqlarni boshqarish: Tarmoqdagi o‘zgarishlarni (masalan, qobiliyat yoki tarmoq sharoitining o‘zgarishi) hisobga olishda Jonson algoritmi tez va samarali natijalar beradi. Tarmoqda yuzaga keladigan o‘zgarishlarni tezda tahlil qilib, optimallashtirishni amalga oshirish mumkin.

5. Resurslarni tejash: Dijkstra algoritmi faqat bitta nuqtadan boshqa nuqtalarga yo‘lni hisoblashda samarali bo‘lsa, Jonson algoritmi butun graf bo‘ylab eng qisqa yo‘llarni topishga yordam beradi. Bu tarmoqda resurslarni samarali taqsimlash uchun qulaydir.

6. Tarmoqning ishlash samaradorligini oshirish: Internet tarmog‘ida optimal marshrutni tanlash orqali uzatish tezligi va tizimning umumiy samaradorligini oshirish mumkin. Jonson algoritmi orqali tarmoq resurslaridan maksimal darajada foydalanish mumkin.

Johnson algoritmi Internet tarmog'ida eng qisqa yo'llarni topish, tarmoqni optimallashtirish va samaradorlikni oshirishda muhim vosita hisoblanadi.

Masala:

Ikkita operatsion tizim mavjud, har bir mahsulotga ikkala tizimda birma-bir ishlov beriladi. Har bir mahsulot uchun birinchi tizimda ishlov berish vaqti va ikkinchi tizimda ishlov berish vaqti berilgan. Mahsulotlar ketma-ketligini shunday tartiblash kerakki, umumiy ishlab chiqarish vaqti minimal bo'lsin.

```
using System;
using System. Collections.Generic;
using System.Linq;
class Program
{
   static void Main()
   {
      // Mahsulotlar va ularning Tizim 1 va Tizim 2 da ishlov berish vaqtlari
int [,] jobs = {
         {3, 2}, // Mahsulot 1 (Tizim 1: 3, Tizim 2: 2)
         {2, 3}, // Mahsulot 2 (Tizim 1: 2, Tizim 2: 3)
         {1, 4}, // Mahsulot 3 (Tizim 1: 4, Tizim 2: 1)
         {1, 4} // Mahsulot 4 (Tizim 1: 1, Tizim 2: 4)
      };
      // Jonson algoritmi bo'yicha mahsulotlarni tartiblash
      var result = JohnsonAlgorithm(jobs);
      // Natijani chiqarish
Console.WriteLine("Optimal tartib:");
      foreach (var job in result)
      {
Console.WriteLine($"Mahsulot {job + 1}");
      }
   }
   static List<int>JohnsonAlgorithm(int[,] jobs)
```

```
        {
            List<int> group1 = new List<int>(); // Tizim 1 uchun guruh
            List<int> group2 = new List<int>(); // Tizim 2 uchun guruh
    // Mahsulotlar ro'yxatini va ular bilan bog'liq minimal vaqtlarga qarab guruhlarni ajratish
            for (int i = 0; i<jobs.GetLength(0); i++)
            {
                if (jobs[i, 0] < jobs[i, 1]) // Tizim 1 vaqti kichik
                    group1.Add(i);
                else // Tizim 2 vaqti kichik
                    group2.Add(i);
            }
            // Group1 va Group2'ni alohida tartibda saralash
            group1 = group1.OrderBy(i => jobs[i, 0]).ToList();
            group2 = group2.OrderByDescending(i => jobs[i, 1]).ToList();
            // Har ikkala guruhni birlashtirish
            group1.AddRange(group2);
            return group1;
        }
    }
```

Natija: Optimal tartib:

Mahsulot 1,Mahsulot 2,Mahsulot 3,Mahsulot 4.

Maqolada bu algoritmning tezkor internet tarmoqlarida ma’lumot uzatishni optimallashtirishdagi ahamiyati tahlil qilindi. Mahsulotlar tartibi Johnson algoritmi orqali optimal bo’lib, samarali ishlov berish vaqtini ta’minlaydi. Bu algoritm ishlab chiqarish jarayonini optimallashtirishda samarali va tez yechimlarni taklif etadi.

Foydalanilgan adabiyotlar ro’yxati:

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.