FORD-BELLMAN ALGORITMINING GRAFLARDA QO‘LLANILISHI

Farmonov Sherzodbek Rahmonjonovich
Farg’ona davlat universiteti amaliy matematika va informatika kafedrasi katta o’qituvchisi
farmonovsh@gmail.com
Rahimaliev Maronali Zokirzoda
Farg‘ona davlat unversiteti talabasi
rahimalievmardonali@gmail.com

Anotatsiya: *Ushbu maqolada Ford-Bellman algoritmining graflar va ularning strukturalarida qo‘llanilishi batafsil tahlil qilinadi. Algoritm graf nazariyasida eng qisqa yo‘lni topish uchun samarali vosita sifatida tanilgan bo‘lib, uning ishlash prinsipi, iteratsiyalar orqali masofalarni yangilashga asoslanadi.*

Kalitso‘zlar: *Ford-Bellman algoritmi, eng qisqa yo‘l, graflar, tarmoq tahlili, iteratsiya, salbiy og‘irlik, ma'lumotlar strukturalari, yo‘nalgan graflar, yo‘nalmagan graflar, adjacency matrix, adjacency list, sikl, tarmoq optimallashtirish, algoritmlar, graf nazariyasi.*

Abstract: *This paper provides a detailed analysis of the application of the Ford-Bellman algorithm in graphs and their structures. The algorithm is well-known as an efficient tool for finding the shortest path in graph theory, with its working principle based on updating distances through iterations.*

Keywords: *Ford-Bellman algorithm, shortest path, graphs, network analysis, iteration, negative weights, data structures, directed graphs, undirected graphs, adjacency matrix, adjacency list, cycle, network optimization, algorithms, graph theory.*

Аннотация: *В данной статье подробно анализируется применение алгоритма Форда-Беллмана в графах и их структурах. Этот алгоритм является эффективным инструментом для нахождения кратчайшего пути в теории графов, его принцип работы основан на обновлении расстояний через итерации.*

Ключевые слова: *Алгоритм Форда-Беллмана, кратчайший путь, графы, анализ сетей, итерации, отрицательные веса, структуры данных, ориентированные графы, неориентированные графы, матрица смежности, список смежности, цикл, оптимизация сетей, алгоритмы, теория графов.*

Ford-Bellman algoritmi eng qisqa yo'l masalalarini hal qilish uchun ishlatiladigan mashhur algoritmlardan biridir. Bu algoritm grafda bir nuqtadan boshqa barcha nuqtalarga eng qisqa yo‘lni topish uchun mo‘ljallangan bo‘lib, ayniqsa salbiy

og‘irliklarga ega bo‘lgan yo‘llar mavjud bo‘lsa ham samarali ishlaydi. Ford-Bellman algoritmi 1950-yillarda Richard Bellman va Lester Ford tomonidan ishlab chiqilgan bo‘lib, uning nomi shu ikki matematikning ismlaridan olingan. Algoritmning asosiy maqsadi - ma'lum bir manba nuqtasidan boshlash va har bir boshqa nuqtaga eng qisqa yo‘lni topishdir. Bu masala ko‘plab sohalarda, xususan, transport tarmoqlari, kompyuter tarmoqlari, va boshqa tizimlarda keng qo‘llaniladi. Ford-Bellman algoritmi graf nazariyasining asosiy qismi bo‘lib, grafda tugunlar va yo‘llar (yoki qirralar) mavjud. Har bir yo‘lning og‘irligi (yoki vaznini) belgilash orqali, algoritm bu og‘irliklar asosida eng qisqa yo‘llarni hisoblash imkoniyatini yaratadi. Ford-Bellman algoritmi uchun muhim jihat shundaki, u salbiy og‘irliklarga ega bo‘lgan yo‘llar bilan ishlay oladi.

Ford-Bellman algoritmining ishlash mexanizmi va tushunchalarini to‘g‘ri tushunish, uni turli xil masalalarda qo‘llash uchun muhimdir. Misol uchun, agar bizning maqsadimiz biror tarmoqda eng qisqa yo‘lni topish bo‘lsa, Ford-Bellman algoritmi yordamida buni samarali tarzda amalga oshirishimiz mumkin.

Ford-Bellman algoritmi asosiy nazariyaga ega bo‘lib, uning turli xil ma'lumotlar strukturalari yordamida optimallashtirilgan variantlari mavjud. Ma'lumotlar strukturalari – bu algoritmning samaradorligini oshirish uchun ishlatiladigan tashkil etish metodlari bo‘lib, ular algoritmning ishlash tezligini va xotira sarfini optimallashtirishga yordam beradi. Ford-Bellman algoritmiga asoslangan ba'zi variantlar matritsa (adjacency matrix) yoki ro‘yxat (adjacency list) kabi ma'lumotlar strukturalarini qo‘llashni o‘z ichiga oladi. Bu strukturalar orqali grafaning tugunlari va ularning o‘zaro bog‘lanishlari ko‘rsatiladi, bu esa algoritmning samarali ishlashiga yordam beradi. Algoritmning ishlash usulini to‘liq tushunish, uning qanday qilib turli xil grafalarda samarali ishlashini aniqlash imkonini beradi. Ford-Bellman algoritmining asosiy yutuqlaridan biri shundaki, u salbiy og‘irliklarga ega bo‘lgan grafalarda ham samarali ishlaydi, lekin salbiy sikllar mavjud bo‘lsa, natijalar xato bo‘lishi mumkin. Bu algoritmning keng qo‘llanishi va ahamiyati nafaqat nazariy, balki amaliy masalalarda ham katta ahamiyatga ega. Ford-Bellman algoritmining o‘rganilishi, ayniqsa uning graflar va ma'lumotlar strukturalarida qo‘llanilishiga alohida e'tibor qaratilmoqda, chunki bu algoritmni ko‘plab turli tizimlarda qo‘llash orqali samarali yechimlar topish mumkin. Graflar matematika va kompyuter fanlarida muhim o‘rin tutadi chunki ular ko‘plab amaliy muammolarni model qilishda qo‘llaniladi. Graflar tugunlar (vertices) va ularni bog‘lovchi yo‘llardan (edges) tashkil topgan tuzilmalar bo‘lib, har bir yo‘l ikki tugun orasidagi aloqani ifodalaydi. Graflar, masalan, transport tizimlarida, tarmoq tahlilida va boshqa ko‘plab sohalarda ishlatiladi. Graflar orqali ma'lumotlar strukturalarini va tizimlarning o‘zaro aloqalarini tasvirlash mumkin. Har bir grafikda tugunlar va qirralar bo‘lib, ular o‘rtasida qandaydir bog‘lanish mavjud. Graflar, shuningdek, qirralarning yo‘nalgan yoki yo‘nalmagan bo‘lishiga qarab o‘ziga xos turlarga ajratiladi.

Graflarning turlari asosan ikki xil bo‘ladi: yo‘nalgan graflar (directed graphs) va yo‘nalmagan graflar (undirected graphs). Yo‘nalgan grafikda har bir qirra (edge) biror

yo‘nalish bo‘yicha bog‘lanishni ifodalaydi, ya'ni har bir yo‘l bir tugundan boshqa tugunga yo‘nalgan bo‘ladi. Masalan, kompyuter tarmog‘ida ma'lumotlarning uzatilishi yo‘nalgan grafik sifatida tasvirlanishi mumkin chunki ma'lumotlar faqat ma'lum bir yo‘nalishda o‘tadi. Graflarning ma'lumotlar strukturalarida qanday tashkil etilishi ham juda muhim. Graflarni kompyuterda saqlash va ularni qayta ishlash uchun bir necha xil ma'lumotlar strukturalari qo‘llaniladi. Eng keng tarqalgan strukturalardan biri — **adjacency matrix** (matritsa). Matritsa yordamida grafadagi har bir tugun uchun boshqa tugunlar bilan bo‘lgan aloqalar ko‘rsatiladi. Matritsa ikki o‘lchovli massiv sifatida tasvirlanadi, bunda massivning i-chi satri va j-chi ustuni i va j tugunlarining o‘rtasidagi bog‘lanishni ifodalaydi. Agar i va j tugunlar o‘rtasida bog‘lanish mavjud bo‘lsa, matritsada mos keluvchi hujayra qiymati 1 bo‘ladi, aks holda esa 0 bo‘ladi.

Ikkinchi keng tarqalgan struktura esa **adjacency list** (ro‘yxat) bo‘lib, bu tuzilma grafadagi har bir tugun uchun uning qo‘shni tugunlari ro‘yxatini saqlaydi. Adjacency list da har bir tugun uchun alohida ro‘yxat mavjud bo‘lib, u o‘sha tugun bilan bog‘langan barcha tugunlarni o‘z ichiga oladi. Masalan, agar grafada A tuguni B va C tugunlari bilan bog‘langan bo‘lsa, A tugunining ro‘yxatida B va C tugunlari bo‘ladi. Adjacency list tuzilmasi xotira jihatidan juda samarali hisoblanadi chunki faqat mavjud bog‘lanishlar uchun joy ajratiladi. Shuningdek, bu tuzilma katta grafalar uchun samaraliroq chunki kerakli joy faqat o‘zaro aloqada bo‘lgan tugunlar uchun ajratiladi. Graflarni ifodalash uchun yana bir nechta usullar mavjud. Masalan, **edge list** (qirralar ro‘yxati) tuzilmasi orqali grafikdagi barcha qirralar ro‘yxati shaklida saqlanadi. Bu tuzilma juda oddiy va undan grafdagi barcha qirralarni tahlil qilishda foydalanish mumkin. Biroq, edge list tuzilmasi nafaqat yirik grafikalarni samarali saqlashga imkon bermaydi, balki qo‘shni tugunlar o‘rtasidagi bog‘lanishni aniqlashda ham kamchiliklarga ega bo‘lishi mumkin. Shu sababli grafikni saqlash uchun tuzilmani tanlashda uning hajmi va ishlash samaradorligi hisobga olinishi kerak. Ford-Bellman algoritmi graf nazariyasida eng qisqa yo‘l masalasini hal qilish uchun keng qo‘llaniladi. Graflar, tugunlar va ular orasidagi bog‘lanishlardan iborat bo‘lib, Ford-Bellman algoritmi grafaning har bir tuguniga eng qisqa masofani topishga qaratilgan. Bu algoritmning ishlash prinsipi, masofalarini hisoblashda har bir tugundan boshlab, uning qo‘shni tugunlariga nisbatan masofalarni yangilashga asoslanadi. Graflarni tahlil qilishda Ford-Bellman algoritmi tugunlar orasidagi aloqalarni ko‘rib chiqish uchun qirralar (edges) va og‘irliklardan foydalanadi. Ford-Bellman algoritmi asosan, grafikdagi har bir tugunni bir necha marta ko‘rib chiqadi. Masalan, grafikda mavjud bo‘lgan barcha qirralar orqali har bir tugun uchun eng qisqa masofani hisoblash uchun kerakli yangilanishlarni amalga oshiradi. Bu jarayon tarmoqda masofa yangilanayotgan vaqt davomida grafikning barcha qirralari tahlil qilinadi, ya'ni har bir tugundan barcha qo‘shni tugunlarga eng qisqa yo‘lni topish uchun yangilanishlar ro‘y beradi.

Graflarda eng qisqa yo‘lni topishda Ford-Bellman algoritmi har bir tugun uchun eng qisqa masofani iteratsiya (takrorlash) usuli bilan yangilab boradi. Bu jarayonda, agar

yangi hisoblangan masofa kichik bo‘lsa, u holda tugunning masofasi yangilanadi. Ushbu yangilanishlar bir nechta marta amalga oshiriladi va har bir marta yangilanish tugunlar o‘rtasidagi mavjud yo‘llarni ko‘rib chiqish orqali eng qisqa yo‘lni yangilaydi. Bu jarayon, agar grafada salbiy og‘irliklar bo‘lsa ham, doimiy ravishda takrorlanadi. Ford-Bellman algoritmi graflarda sikllarning mavjudligini ham hisobga oladi. Agar grafda salbiy og‘irlikli sikl bo‘lsa, bu algoritm o‘zining maqsadiga erisha olmaydi, chunki salbiy sikllar orqali masofalar cheksiz kamayib boradi. Bunday holatda Ford-Bellman algoritmi xatolikni aniqlash imkoniyatiga ega bo‘lib, sikllarni aniqlab ularning mavjudligini bildirish mumkin.

Misol. Bir nechta omborlar (A, B, C) va do'konlar (X, Y, Z) mavjud. Har bir ombor va do'kon o'rtasidagi transport yo'llarining xarajatlari (masofalar yoki vaqt) belgilangan. Maqsad — omborlardan do'konlarga eng arzon yoki eng tezkor yo'lni tanlash.

Grafik Tashkili:

- Omborlar (A, B, C) grafikdagi tugunlar sifatida.
- Do'konlar (X, Y, Z) ham grafikdagi tugunlar sifatida.
- Har bir ombor va do'kon o'rtasidagi transport yo'llari qirralar sifatida ko'rsatiladi va har bir qirra og'irligi transport xarajatlarini ifodalaydi.

```
using System;
using System.Collections.Generic;

class Program
{
    // Maxsus chegara uchun qiymat
    static int INF = int.MaxValue;

    // Ford-Bellman Algoritmi
    static void FordBellman(int[,] graph, int source, int n)
    {
        // Dastlabki masofalar, barcha masofalar INF ga teng
        int[] dist = new int[n];
        for (int i = 0; i < n; i++)
            dist[i] = INF;
        dist[source] = 0;

        // Vaqtning har bir bosqichida masofalarni yangilash
        for (int i = 0; i < n - 1; i++)
        {
            for (int u = 0; u < n; u++)
            {
                for (int v = 0; v < n; v++)
```

```
            {
                if (graph[u, v] != INF && dist[u] != INF && dist[u] + graph[u, v] < dist[v])
                {
                    dist[v] = dist[u] + graph[u, v];
                }
            }
        }
    }
    // Natijalarni chiqarish
    Console.WriteLine("Eng qisqa yo'l masofalari:");
    for (int i = 0; i < n; i++)
    {
        if (dist[i] == INF)
            Console.WriteLine("Ombor {0}dan Do'kon {1}ga yo'l yo'q.", source, i);
        else
            Console.WriteLine("Ombor {0}dan Do'kon {1}ga masofa: {2}", source, i, dist[i]);
    }
}

static void Main()
{
    // Omborlar va do'konlar o'rtasidagi transport xarajatlarini ko'rsatish
    // 0 - Ombor A, 1 - Ombor B, 2 - Ombor C, 3 - Do'kon X, 4 - Do'kon Y, 5 - Do'kon Z
    int n = 6; // 3 ombor + 3 do'kon
    int[,] graph = new int[n, n];

    // Barcha elementlarni INF bilan to'ldirish (yo'l yo'q)
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            graph[i, j] = INF;

    // Yo'llarni belgilash (masofalar)
    graph[0, 3] = 10; // Ombor A dan X ga
    graph[0, 4] = 20; // Ombor A dan Y ga
    graph[1, 3] = 15; // Ombor B dan X ga
    graph[1, 5] = 30; // Ombor B dan Z ga
    graph[2, 4] = 25; // Ombor C dan Y ga
```

```
        graph[2, 5] = 5;  // Ombor C dan Z ga

        // Ford-Bellman algoritmini ishlatish
        Console.WriteLine("Ombor A dan barcha do'konlarga eng qisqa yo'lni topish:");
        FordBellman(graph, 0, n); // Ombor A (0)dan barcha do'konlarga yo'lni hisoblash

        Console.WriteLine("\nOmbor B dan barcha do'konlarga eng qisqa yo'lni topish:");
        FordBellman(graph, 1, n); // Ombor B (1)dan barcha do'konlarga yo'lni hisoblash

        Console.WriteLine("\nOmbor C dan barcha do'konlarga eng qisqa yo'lni topish:");
        FordBellman(graph, 2, n); // Ombor C (2)dan barcha do'konlarga yo'lni hisoblash
    }
}
```

Natija: Kodni ishga tushirgandan so'ng, har bir ombordan barcha do'konlarga eng qisqa yo'l masofalari chiqariladi. Masalan, "Ombor A dan Do'kon X ga masofa: 10" yoki "Ombor B dan Do'kon Z ga masofa: 30".

Ford-Bellman algoritmi graflar orqali eng qisqa yo‘lni topishda samarali va kuchli vosita sifatida ajralib turadi. Uning ishlash prinsipi grafdagi har bir tugun uchun masofalarni yangilash va iteratsiyalar orqali eng qisqa yo‘lni aniqlashga asoslanadi. Algoritm salbiy og‘irliklarga ega qirralar bilan ishlashda ham samarali, bu uni tarmoq tahlili, transport tizimlari va boshqa sohalarda qo‘llash imkoniyatini beradi. Biroq, salbiy og‘irlikli sikllar mavjud bo‘lsa, algoritm noto‘g‘ri natijalarga olib kelishi mumkin. Shunday bo‘lsa-da Ford-Bellman algoritmi graf nazariyasidagi boshqa algoritmlar bilan taqqoslaganda uning salbiy og‘irliklar bilan ishlash imkoniyati va graflarning murakkab tuzilmalarini tahlil qilishdagi samaradorligi uni amaliyotda keng qo‘llashga imkon beradi

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

10. 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.

11. Raxmonjonovich, F. S. (2023). Dasturlashda interfeyslardan foydalanishning ahamiyati. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, *2*(2), 425-429.

12. 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.

13. 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.

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

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

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