

FORD-BELLMAN ALGORITMI, ENG QISQA YO‘LLARNI TOPISH

Farmonov Sherzodbek Rahmonaliyevich

Farg‘ona davlat universiteti amaliy matematika va informatika kafedrasida o‘qituvchisi

farmonovsh@gmail.com

Ismoilova Muslimaxon Axmadjon qizi

Farg‘ona davlat universiteti 2-kurs talabasi
tursunovamuslimaxon05@gmail.com

Anotatsiya: Ford-Bellman algoritmi, eng qisqa yo‘llarni topish uchun ishlatiladigan asosiy algoritmlardan biridir. Bu algoritm grafdagi manba tugunidan boshqa barcha tugunlarga bo‘lgan eng qisqa masofalarni hisoblashga qaratilgan va ayniqsa manfiy og‘irliklar mavjud bo‘lgan grafiklarda samarali ishlaydi. Maqolada Ford-Bellman ushbu algoritmi amaliyotda qanday ishlatish mumkinligi va uning potensial imkoniyatlari haqida muhim fikrlarni taqdim etadi.

Kalit so‘zlar: Ford-Bellman algoritmi, eng qisqa yo‘l, graf, manfiy og‘irliklar, tarmoq tahlili, transport tizimlari, algoritmik samaradorlik, robototexnika, eng qisqa yo‘l algoritmi.

Annotation: The Ford-Bellman algorithm is one of the main algorithms used to find the shortest paths. This algorithm focuses on calculating the shortest distances from a source node to all other nodes in a graph and is particularly effective for graphs with negative edge weights. This paper presents key insights on how the algorithm can be applied in practice and explores its potential capabilities

Keywords: Ford-Bellman algorithm, shortest path, graph, negative weights, network analysis, transportation systems, algorithmic efficiency, robotics, shortest path algorithm.

Аннотация: Алгоритм Форда-Беллмана является одним из основных алгоритмов для нахождения кратчайших путей. Этот алгоритм направлен на вычисление кратчайших расстояний от исходной вершины до всех других вершин графа и особенно эффективно работает с графами, содержащими отрицательные веса рёбер. В статье представлены важные идеи о том, как можно использовать этот алгоритм на практике и его потенциальные возможности.

Ключевые слова: Алгоритм Форда-Беллмана, кратчайший путь, граф, отрицательные веса, анализ сети, транспортные системы, алгоритмическая эффективность, робототехника, алгоритм кратчайшего пути.

Ford-Bellman algoritmi graf nazariyasining muhim algoritmlaridan biridir va o'zining oddiyliigi va universal ishlash printsiipi bilan ko'plab amaliy masalalarda qo'llaniladi. Bu algoritmi bir nechta yirik tizimlarda, masalan, transport tarmoqlari, internet tarmog'i, va kompyuter tarmoqlarida yo'nalishlarni optimallashtirishda foydalaniladi. Ford-Bellman algoritmi asosan manfiy og'irliklarni qabul qilgan grafalarda qisqa yo'lni topishda ishlatiladi va manfiy sikllarni aniqlashda ham samarali bo'ladi. Bu maqolada Ford-Bellman algoritmining asosiy printsiiplari, ishlash usuli va C# dasturlash tilida uning kodini taqdim etish ko'rib chiqiladi.

1. Ford-Bellman Algoritmi: Nazariy Asoslar

Ford-Bellman algoritmi – bu manba tugunidan barcha boshqa tugunlarga bo'lgan eng qisqa yo'llarni hisoblash uchun ishlatiladigan algoritmi bo'lib, grafdagi har qanday yo'lning og'irligi manfiy bo'lishi mumkin. Algoritmning ishlash tamoyili shundan iboratki, har bir qirra (yoki yo'l) uchun masofa yangilanadi. Bu jarayon bir necha marta takrorlanadi, chunki birinchi navbatda eng qisqa yo'l topilmaydi, va vaqt o'tgan sari yanada yaxshi yechimlar kelib chiqadi.

Ford-Bellman algoritmi grafdagi barcha qirralar bo'yicha "relaxatsiya" jarayonini amalga oshiradi. Relaxatsiya jarayonida agar mavjud yo'lning masofasi boshqasiga qaraganda kichikroq bo'lsa, yangi masofa qabul qilinadi. Algoritm $V-1$ marta barcha tugunlar uchun bu jarayonni takrorlaydi, chunki eng qisqa yo'lni topish uchun $V-1$ o'zgartirishlar yetarli bo'ladi (V – grafdagi tugunlar soni). Agar bu jarayon V -marta takrorlansa, demak, grafda manfiy sikl mavjud bo'lishi mumkin, chunki bu turdagi sikllar cheksiz kichik qiymatlarni yaratishi mumkin.

2. Ford-Bellman Algoritmining Samaradorligi

Ford-Bellman algoritmining vaqt murakkabligi bo'lib, bu yerda:

* grafdagi tugunlar soni,

* qirralar soni.

Bu algoritmining yaxshi tomoni shundaki, u manfiy og'irliklarga ega grafalarda ham ishlaydi. Biroq, uning samaradorligi katta grafalarda past bo'lishi mumkin. Agar va juda katta bo'lsa, uning ishlash vaqti sekinlashadi. Bu muammo Dijkstra algoritmi kabi boshqa qisqa yo'l algoritmlari yordamida hal qilinadi, ammo u faqat manfiy og'irliklar mavjud bo'lmagan holatlarda ishlaydi.

Ford-Bellman algoritmiga qaraganda Dijkstra algoritmi samaraliroq bo'lishi mumkin, lekin u manfiy og'irliklar bilan ishlay olmaydi. Shuning uchun, Ford-Bellman algoritmi juda foydali, ayniqsa manfiy og'irliklar mavjud bo'lsa.

3. Ford-Bellman Algoritmining Ilovalarida Qo'llanilishi

Ford-Bellman algoritmi real hayotdagi bir nechta tizimlarda qo'llaniladi:

Transport tizimlari: Yo'llar va transport tarmoqlarida eng qisqa yo'lni topish uchun,

Tarmoq tahlili: Kompyuter tarmoqlarida ma'lumotlarni uzatish uchun eng qisqa yo'llarni hisoblashda,

Moliyaviy tahlillar: Pul oqimlarini optimallashtirish va resurslarni taqsimlashda,

Robototexnika: Robotalar uchun eng qisqa yo'lni topish va ularning harakatini optimallashtirish.

Belman-Ford algoritimi $O(V * E)$ vaqt va $O(V)$ fazo

Bellman-ford yagona manbali eng qisqa yo'l algoritimidir . U salbiy qirralarning holatlarda samarali ishlaydi va salbiy sikllarni ham aniqlay oladi .

4.Algoritmning ishlash prinsipi

Algoritmning asosiy prinsipi quyidagicha:

1. Har bir tugun (vertex) uchun maksimal masofa qiymatini cheksiz (infinity) deb belgilaydi, lekin boshlang'ich tugun uchun qiymat 0 ga teng.

2. Grafning har bir qirrasini takroriy ravishda ko'rib chiqadi va tugunlar o'rtasidagi masofani yangilaydi.

3. Har bir qirra ustida tugunlar o'rtasidagi masofani yangilash natijasida qiymatlar o'zgarishi davom etadi.

4. Agar muddat belgilangan iteratsiyalardan keyin masofalar yangilanmasa, u holda algoritm tugaydi.

Bu algoritm $O(V * E)$ murakkablikka ega, bu yerda V - tugunlar soni, E esa qirralar soni.

5. Belman –ford algoritimi c# dasturida amalga oshirilishi.

// -Ford-Belman algoritimi

`using System;`

`using System.Collections.Generic;`

`class GfG {`

`static List<int> BellmanFord(int V, List<List<int>> edges, int src)`

`List<int> dist = new List<int>(new int[V]);`

`for (int i = 0; i < V; i++)`

`dist[i] = (int)1e8;`

`dist[src] = 0;`

`for (int i = 0; i < V; i++) {`

`foreach (List<int> edge in edges) {`

```

    int u = edge[0];
    int v = edge[1];
    int wt = edge[2];
    if (dist[u] != 1e8 && dist[u] + wt < dist[v]) {
        if (i == V - 1)
            return new List<int> { -1 };
        dist[v] = dist[u] + wt;
    }
}
}
return dist;
}
static void Main() {
    int V = 5;
    List<List<int>> edges = new List<List<int>>() {
        new List<int> { 1, 3, 2 },
        new List<int> { 4, 3, -1 },
        new List<int> { 2, 4, 1 },
        new List<int> { 1, 2, 1 },
        new List<int> { 0, 1, 5 }
    };
    int src = 0;
    List<int> ans = BellmanFord(V, edges, src);
    foreach (int dist in ans)
        Console.Write(dist + " ");
}
}

```

Natija : 0 5 6 6 7

Bu dastur bitta manbadan eng qisqa yolni topadi ,

Ford-Bellman algoritmi o'zining oddiyligi va manfiy og'irliklar mavjud bo'lsa ham ishlash imkoniyati bilan mashhur. C# dasturlash tilida kodlashda uning samaradorligi va ishlash prinsipi aniq ko'rsatilgan. Biroq, katta grafalarda uning ishlash tezligi past bo'lishi mumkin, shuning uchun ba'zi maxsus vaziyatlarda boshqa

algoritmlar, masalan, Dijkstra algoritmi yoki A* algoritmi, samaraliroq bo'lishi mumkin.

Foydalanilgan Adabiyotlar

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.
2. Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269–271.
3. Bellman, R. (1958). On a Routing Problem. *Quarterly of Applied Mathematics*, 16, 87-90.
4. GeeksforGeeks. (2024). Ford-Bellman Algorithm. Retrieved from <https://www.geeksforgeeks.org/bellman-ford-algorithm-dp-23/>
1. L OF EDUCATION AND INNOVATION (T. 2, Выпуск 12, сс. 71–74). Zenodo.
5. Farmonov, S., & Toirov, S. (2023). NETDA DASTURLASHNING ZAMONAVIY TEXNOLOGIYALARINI O'RGANISH. Theoretical aspects in the formation of pedagogical sciences, 2(22), 90-96
6. 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.
7. Raxmonjonovich, F. S. (2023). Dasturlashda interfeyslardan foydalanishning ahamiyati. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, 2(2), 425-429.
8. Raxmonjonovich, F. S. (2023). Dasturlashda obyektga yo'naltirilgan dasturlashning ahamiyati. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, 2(2), 434-438.
9. 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.
10. Raxmonjonovich, F. S. (2023). DASTURLASHDA ISTISNOLARNING AHAMIYATI. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, 2(2), 475-481.
11. Raxmonjonovich, F. S. (2023). Dasturlashda abstraksiyaning o'rni. *Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari*, 2(2), 482-486.
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. 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.
14. Raxmonjonovich, F. S. (2023). C# tilida ArrayList bilan ishlashning afzalliklari. Yangi O'zbekiston taraqqiyotida tadqiqotlarni o'rni va rivojlanish omillari, 2(2), 470-474.
15. Farmonov Sherzodbek Raxmonjonovich, & Rustamova Humoraxon Sul-tonbek 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>.
16. 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.
17. Farmonov, S., & Rasuljonova, Z. (2024). OB'KTGA YO'NALTIRILGAN DASTURLASH ZAMONAVIY DASTURLASHNING ASOSI SIFATIDA. Центральноазиатский журнал образования и инноваций, 3(1), 83-86.
18. 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.
19. 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.
20. Raxmonjonovich, F. S. (2024). C# VA MASHINA TILI. Ta'lim innovatsiyasi va integratsiyasi, 12(1), 59-62.
21. Farmonov, S. (2023). C# DASTURLASH TILIDA GRAY KODI BILAN ISHLASH. Центральноазиатский журнал образования и инноваций, 2(12 Part 2), 71-74.
22. Farmonov, S., & Jo'rayeva, M. (2023, December). DASTURLASHDA POLIMORFIZMNING AHAMIYATI. In Международная конференция академических наук (Vol. 2, No. 13, pp. 5-8).
23. Farmonov, S., & Usmonaliyev, U. (2024). O'ZBEKISTON RESPUBLIKASI IT SOHASINING RIVOJLANISH ISTIQBOLLARI. Бюллетень педагогов нового Узбекистана, 2(1), 59-62.

24. 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.
25. Bertsekas, D. P. (1998). Network Optimization: Continuous and Discrete Models. Athena Scientific
26. Garey, M. R., & Johnson, D. S. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman.
27. Zhang, L., & Li, Y. (2010). "A New Shortest Path Algorithm Based on the Bellman-Ford Algorithm." International Journal of Computer Science and Network Security, 10(2), 225-230.
28. Tarjan, R. E. (1977). "Finding shortest paths." SIAM Journal on Computing, 6(3), 503-517.
29. Kleinberg, J., & Tardos, É. (2006). Algorithm Design. Pearson.
30. Bertsekas, D. P. (1998). Network Optimization: Continuous and Discrete Models. Athena Scientific.