

Logistikada Traveling Salesperson algoritmining ahamiyati

Farmonov Sherzodbek Raxmonjonovich

*Farg'ona davlat universiteti amaliy matematika va informatika kafedrasi katta
o'qituvchisi*

farmonovsh@gmail.com

Sobirov Asadbek Nodirjon o'g'li

Farg'ona davlat universiteti talabasi

asadbeksobirov@yandex.ru

Annotatsiya. Hozirgi zamon logistika sohasida yuk tashish, tarqatish va omborlarni boshqarish samaradorligini oshirish masalalari dolzarb hisoblanadi. Bu borada Sayohat qiluvchi savdogar masalasi (Traveling Salesperson Problem, TSP) algoritmi optimal marshrutlarni rejalashtirish va transport xarajatlarini kamaytirishda muhim ro'l o'ynaydi. Ushbu maqolada TSP algoritmining logistikada qo'llanilishi va amaliyotdagi ahamiyati yoritilgan.

Kalit so'zlar: Traveling salesperson masalasi, logistika, yuk tashish, marshrutlarni optimallashtirish, ombor boshqaruvi, transport xarajatlari.

Annotation. In the field of modern logistics, issues of increasing the efficiency of cargo transportation, distribution, and warehouse management are relevant. In this context, the Traveling Salesperson Problem (TSP) algorithm plays a significant role in planning optimal routes and reducing transportation costs. This article highlights the application of the TSP algorithm in logistics and its practical importance.

Keywords. Traveling Salesperson Problem, logistics, cargo transportation, route optimization, warehouse management, transportation costs.

Аннотация. В современной сфере логистики актуальными являются вопросы повышения эффективности транспортировки грузов, их распределения и управления складами. В этой связи алгоритм задачи коммивояжера (Traveling Salesperson Problem, TSP) играет важную роль в планировании оптимальных маршрутов и снижении транспортных расходов. В данной статье освещено применение алгоритма TSP в логистике и его практическое значение.

Ключевые слова. Задача коммивояжера, логистика, грузоперевозки, оптимизация маршрутов, управление складом, транспортные расходы.

Kirish. Logistika zamonaviy iqtisodiyotning ajralmas qismi hisoblanadi. Yuk tashish xizmatlaridan boshlab omborni boshqarishgacha bo'lgan barcha jarayonlarda optimal marshrutlarni aniqlash zarur. TSP algoritmi eng qisqa

yo'llarni aniqlash va xarajatlarni minimallashtirishga imkon beradi. Bu nafaqat vaqtni tejaydi, balki yoqilg'i sarfi va resurslar boshqaruvini ham optimallashtiradi.

Sayohat qiluvchi savdogar masalasi (TSP). TSP — bir shahar (yoki nuqta)dan boshlanib, bir nechta shaharlarga (yoki nuqtalarga) tashrif buyurib yana boshlang'ich shaharga qaytadigan eng qisqa marshrutni topish masalasidir. Matematik nuqtai nazardan, TSP bir nechta shaharlar (mijozlar) orasidagi masofalar jadvaliga asoslanib, barcha mumkin bo'lgan marshrutlarni tahlil qiladi va umumiy masofasi eng kichik bo'lgan marshrutni tanlaydi.

TSPning logistikadagi qo'llanilishi. Yuk tashish marshrutlarini optimallashtirish

TSP yuk mashinalari yoki kuryerlar uchun eng qisqa marshrutni aniqlashda ishlatiladi. Bu jarayon quyidagi ustunliklarni beradi:

- Yoqilg'i tejalishi: Transport vositalari minimal masofa bosib o'tadi.
- Vaqtni tejash: Yuklarni belgilangan vaqt ichida etkazib berish imkoniyati oshadi.

Misol: Ombordan mijozlar punktlariga yuklarni yetkazib berishda TSP yordamida optimal marshrut aniqlanadi. Masalan, $O \rightarrow C \rightarrow A \rightarrow D \rightarrow B \rightarrow O$ marshruti, masofasi boshqa variantlarga nisbatan samaraliroq bo'lishi mumkin.

Ombor logistikasini boshqarish. Omborlarda buyurtmalar yig'ish jarayonini optimallashtirish uchun TSP algoritmi ishlatiladi. Masalan, bir xodim omborda harakatlanib buyurtmalarni yig'ishi kerak bo'lsa, eng qisqa yo'l belgilab beriladi. Natijada:

- Xodimlarning ishi samaraliroq bo'ladi.
- Ombordagi operatsion xarajatlar kamayadi.

Tarqatish tarmog'ini rejalashtirish. TSP algoritmi tarqatish markazlaridan mijozlarga mahsulotlarni yetkazib berishda ishlatiladi. Bu jarayonda:

- Omborlar va mijozlar orasidagi eng samarali marshrutlar aniqlanadi.
- Ko'p zonali yetkazib berishda har bir mintaqa uchun yuk mashinalari optimal tarzda taqsimlanadi.

Misol: Katta logistika kompaniyalari, masalan, Amazon yoki DHL, TSP algoritmini tarqatish tarmoqlarini rejalashtirishda qo'llaydi.

Shahar transporti va avtomatlashtirilgan tizimlar

Shahar ichida avtobus yoki taksi xizmatlari uchun eng samarali yo'nalishlarni aniqlashda TSP qo'llaniladi. Bundan tashqari, robot changyutgichlar yoki avtomatlashtirilgan transport vositalarining yo'lini rejalashtirishda ham TSP algoritmi qo'llanadi.

TSP Algoritmini Amaliyotga Joriy Etish.

TSP algoritmini ishlatish uchun bir qancha yondashuvlar mavjud:

- Qo‘pol kuch yondashuvi (Brute Force): Barcha marshrutlarni ko‘rib chiqib eng qisqasini tanlash. Bu kichik miqyosdagi masalalarda samarali.
- Dinamik dasturlash: Masalan, Bellman-Held-Karp algoritmi yordamida masofalar tahlil qilinadi.
- Evristika usullari: Katta miqyosdagi masalalarda (masalan, minglab mijozlar) tezkor echimlar uchun ishlatiladi. Simulyatsion sovutish (Simulated Annealing) yoki genetika algoritmlari misol bo‘la oladi.

TSP Algoritmining Ahamiyati.

- Logistika xarajatlarini kamaytiradi: Marshrutlar optimallashtirish uchun transport xarajatlari sezilarli darajada qisqaradi.
- Samaradorlikni oshiradi: Yuk tashish va yetkazib berish tezligi ortadi.
- Atrof-muhitni himoya qiladi: Yoqilg‘i sarfi kamaygani uchun chiqindilar hajmi ham kamayadi.

Masala.

Bir yuk mashinasi quyidagi shaharlar orasida yuk tashishi kerak:

1. Ombor (O)
2. Mijoz A
3. Mijoz B
4. Mijoz C
5. Mijoz D

Shaharlar orasidagi masofalar (kilometrda) quyidagi jadvalda keltirilgan:

	Ombor (O)	A	B	C	D
O	0	10	20	15	30
A	10	0	25	17	28
B	20	25	0	35	22
C	15	17	35	0	18
D	30	28	22	18	0

Talab:

- Yuk mashinasi barcha mijozlarga (A, B, C, D) yuk etkazib berishi va yana omborga qaytishi kerak.
- Marshrut shunday tanlansin-ki, umumiy masofa minimal bo'lsin.

Yechim:**1. Qo'pol Kuch Yondashuvi (Brute Force)**

- Marshrutlarni hisoblash uchun barcha mumkin bo'lgan yo'nalishlarni (permutatsiyalarni) ko'rib chiqamiz.
- 4 ta mijoz uchun $4! = 24$ ta marshrutni tekshirish kerak:
 - Masalan: **O -> A -> B -> C -> D -> O** yoki **O -> C -> A -> D -> B -> O**

2. Masofalarni Hisoblash

Har bir marshrutning masofasini hisoblab chiqamiz va eng qisqa marshrutni aniqlaymiz.

Masalan:

1. O -> A -> B -> C -> D -> O:

- O -> A: 10
- A -> B: 25
- B -> C: 35
- C -> D: 18
- D -> O: 30

Umumiy masofa: $10 + 25 + 35 + 18 + 30 = 118$ km

2. O -> C -> A -> D -> B -> O:

- O -> C: 15
- C -> A: 17
- A -> D: 28
- D -> B: 22
- B -> O: 20

Umumiy masofa: $15 + 17 + 28 + 22 + 20 = 102$ km

Barcha marshrutlar ichida masofasi eng qisqa bo'lganini tanlaymiz.

3. Optimal Marshrut

Eng qisqa yo'l masofasi **102 km**, marshruti quyidagicha:

O -> C -> A -> D -> B -> O

Ushbu masalining c# dasturlash tilidagi yechimi:

```
using System;

class TSPEExample
{
    static int[,] distances = {
        { 0, 10, 20, 15, 30 }, // Ombor (O)
        { 10, 0, 25, 17, 28 }, // A
        { 20, 25, 0, 35, 22 }, // B
        { 15, 17, 35, 0, 18 }, // C
        { 30, 28, 22, 18, 0 }  // D
    };

    static int numberOfCities = 5;
    static int minDistance = int.MaxValue;
    static int[] bestRoute;

    static void GeneratePermutations(int[] route, int start, int end)
    {
        if (start == end)
        {
            CalculateRoute(route);
        }
        else
        {
            for (int i = start; i <= end; i++)
            {
                Swap(ref route[start], ref route[i]);
                GeneratePermutations(route, start + 1, end);
                Swap(ref route[start], ref route[i]);
            }
        }
    }

    static void Swap(ref int a, ref int b)
    {
        int temp = a;
        a = b;
        b = temp;
    }

    static void CalculateRoute(int[] route)
    {
        int totalDistance = 0;
        for (int i = 0; i < route.Length - 1; i++)
        {
            totalDistance += distances[route[i], route[i + 1]];
        }
        totalDistance +=
distances[route[route.Length - 1], route[0]]; // Return to start

        if (totalDistance < minDistance)
```

```

        {
            minDistance = totalDistance;
            bestRoute = (int[])route.Clone();
        }
    }

    static void Main()
    {
        int[] cities = { 0, 1, 2, 3, 4 }; // Ombor (0) va mijozlar
        GeneratePermutations(cities, 1, numberOfCities - 1);

        Console.WriteLine("Eng qisqa masofa: " + minDistance);
        Console.Write("Eng yaxshi marshrut: ");
        foreach (var city in bestRoute)
        {
            Console.Write((char)('O' + city) + " -> ");
        }
        Console.WriteLine("O");
    }
}

```

Ishlash jarayoni:

1. Masofa jadvali:
 - **distances** massivida shaharlar orasidagi masofalar saqlanadi.
2. Permutatsiyalarni yaratish:
 - **GeneratePermutations** funksiyasi barcha mumkin bo'lgan marshrutlarni generatsiya qiladi.
 - Masalan: O -> A -> B -> C -> D -> O.
3. Har bir marshrut uchun masofani hisoblash:
 - **CalculateRoute** funksiyasi qabul qilingan marshrutning umumiy masofasini hisoblaydi.
 - Agar bu masofa eng qisqa masofadan kichik bo'lsa, u yangi eng qisqa marshrut sifatida saqlanadi.
4. **Swap** funksiyasi:
 - Marshrutlarni yaratish uchun massiv elementlarini almashtiradi.
5. Natija chiqishi:
 - Eng qisqa masofa va unga mos keladigan marshrut chop etiladi.

7. Xulosa

TSP algoritmi logistika sohasida keng qo'llanilib, xarajatlarni kamaytirish va xizmat sifatini oshirishda muhim ahamiyatga ega. U omborlarni boshqarishdan tortib, kuryerlik xizmatlariga qadar ko'plab jarayonlarni samarali rejalashtirishga yordam beradi. Kelajakda ushbu algoritmnı avtomatlashtirish va sun'iy intellekt bilan birlashtirish logistikaning yanada rivojlanishiga xizmat qiladi.

Adabiyotlar

1. Applegate, D. L., Bixby, R. E., Chvátal, V., & Cook, W. J. (2007). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press.
2. Laporte, G. (1992). "The Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms". *European Journal of Operational Research*, 59(2), 231–247.
3. Dantzig, G. B., & Ramser, J. H. (1959). "The Truck Dispatching Problem". *Management Science*, 6(1), 80–91.
4. Gutin, G., & Punnen, A. P. (Eds.). (2006). *The Traveling Salesman Problem and Its Variations*. Springer Science & Business Media.
5. Toth, P., & Vigo, D. (2002). *Vehicle Routing: Problems, Methods, and Applications*. SIAM.
6. Ropke, S., & Pisinger, D. (2006). "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". *Transportation Science*, 40(4), 455–472.
7. Mitrovic-Minic, S., Krishnamurti, R., & Laporte, G. (2004). "Double-horizon Based Heuristics for the Dynamic Pickup and Delivery Problem with Time Windows". *Transportation Research Part B: Methodological*, 38(8), 669–685.
8. Winston, W. L. (2004). *Operations Research: Applications and Algorithms*. Cengage Learning.
9. Clarke, G., & Wright, J. W. (1964). "Scheduling of Vehicles from a Central Depot to a Number of Delivery Points". *Operations Research*, 12(4), 568–581.
10. Ballou, R. H. (2004). *Business Logistics/Supply Chain Management: Planning, Organizing, and Controlling the Supply Chain*. Pearson.

11.Khan Academy.

Introduction to the Traveling Salesman Problem. [Online Resource]

12.Google AI Blog.

"Optimization Challenges in Vehicle Routing". [Online Resource]