Bài tập CS112: Quy hoạch động

Nhóm : 2
Nguyễn Thiện Nhân 23521083
Trần Vinh Khánh 23520726

Đề Bài 1

Liên minh Hansa (Hanseatic League) là một mạng lưới các thành phố thương mại ven biển ở Bắc Âu, hình thành và phát triển mạnh mẽ trong suốt thế kỷ 12 đến 17 giúp thúc đẩy thương mại và bảo vệ quyền lợi kinh tế của các thành viên thông qua việc tạo ra một hệ thống giao thương mạnh mẽ, trao đổi hàng hóa như gỗ, ngũ cốc, cá, và hàng thủ công. Tuy nhiên các tàu buôn trong liên minh thường không sử dụng đường đi thẳng mà sẽ ghé qua nhiều thành phố khác nhau trên tuyến đường để tiếp tế và trao đổi. Bạn là một thương nhân đang trên hành trình từ London đến Novgorod. Sử dụng đồ thị bên dưới (trọng số trên cạnh là di chuyển chi phí thực tế, trọng số tại node là khoảng cách Euclide đến Novgorod).

- Hãy tính toán chi phí đường đi nếu sử dụng thuật toán tìm kiếm Greedy và UCS.
- Hãy viết chi tiết thông tin tuyến đường sử dụng đề di chuyển nếu sử dụng thuật toán tìm kiếm Greedy và UCS (Ví dụ: London – Amsterdam - Novgorod)
- Đường đi tìm được bởi thuật Greedy và UCS có tối ưu không?

Bài làm

Thuật toán Greedy

Ưu tiên các đỉnh có heuristic nhỏ nhất
Đường đi : $London => Hamburg => Falsterbo => Danzig => Visby => Talinn => Novgorod$
Chi phí $801 + 324 + 498 + 606 + 590 + 474 = 2866$

Thuật toán UCS

Đường đi : $London => Hamburg => Lubeck => Danzig => Riga => Tallinn => Novgorod$
Chi phí : 2496

Nhận xét

Hai thuật toán trên có thể chưa tối ưu

Bài 2

Đề bài

Kaiser là một cảnh sát kỳ cựu. Năm 2200, anh ấy đã có cuộc sống hạnh phúc với những thành tựu và chiến công đáng kể mà anh ấy gặt hái được và có một gia đình với 2 con 1 vợ vô cùng đầm ấm. Tuy nhiên vào một ngày không nắng, vợ anh ấy là Kayra đã bị một tên tội phạm thời gian bắt cóc và giam giữ.

Vì là một người vô cùng yêu thương vợ nên Kaiser ngày đêm đã cố gắng lần theo dấu vết mà tên tội phạm để lại và đã xác định được vợ anh ấy đang bị giam ở đâu đó trong N thành phố được đánh số từ 1 đến N được nối với nhau bởi M con đường một chiều.

Tuy nhiên vì tên bắt cóc vợ anh là một tên tội phạm thời gian khét tiếng nên một số con đường nối giữa các thành phố bị hắn đặt bẫy thời gian nên có độ dài là một số âm. Hãy giúp Kaiser xác định xem có chu trình âm trong thành phố hay không để anh ấy có thể tránh được nguy hiểm và tiếp tục đi giải cứu vợ.

Giải thuật

Phương pháp: Sử dụng thuật toán Bellman-Ford để tìm và phát hiện chu trình âm:

- Khởi tạo khoảng cách ban đầu từ đỉnh nguồn đến các đỉnh khác là $+\infty$, trừ đỉnh nguồn là 0.
- Thực hiện V - 1 lần duyệt tất cả các cạnh để cập nhật khoảng cách.
- Sau vòng lặp, duyệt thêm một lần tất cả cạnh:
  - Nếu có khoảng cách giảm thêm, chu trình âm đã xuất hiện.
  - Lần theo truy vết (predecessor) để xác định chu trình âm.
- Xuất ŸESnếu có chu trình âm và chu trình đó, nếu không thì xuất NO.

Độ phức tạp

- **Thời gian:** $O(V \cdot E)$, trong đó V là số đỉnh và E là số cạnh.
- **Không gian:** $O(V + E)$ để lưu đồ thị và mảng khoảng cách, truy vết.

Mã

```python
import sys

def detect_negative_cycle(N, edges):
    # Initialize distances and predecessor array
    distance = [float('inf')] * (N + 1)
    predecessor = [-1] * (N + 1)
    distance[1] = 0  # Assuming node 1 as the start

    # Relax edges |V| - 1 times
    for _ in range(N - 1):
        for u, v, w in edges:
            if distance[u] != float('inf') and distance[u] + w < distance[v]:
                distance[v] = distance[u] + w
                predecessor[v] = u

    # Check for negative weight cycle
    for u, v, w in edges:
        if distance[u] != float('inf') and distance[u] + w < distance[v]:
            # Negative cycle detected, reconstruct the cycle
            cycle = []
            x = v
            for _ in range(N):  # Move back N steps to find a cycle
                x = predecessor[x]

            # Find the cycle
            start = x
            cycle.append(start)
            x = predecessor[start]
            while x != start:
                cycle.append(x)
                x = predecessor[x]
            cycle.append(start)

            return "YES", cycle[::-1]  # Return the cycle in correct order

    return "NO", []

```

```python
# Example usage
if __name__ == "__main__":
    input = sys.stdin.read
    data = input().splitlines()

    N, M = map(int, data[0].split())
    edges = []
    for i in range(1, M + 1):
        u, v, w = map(int, data[i].split())
        edges.append((u, v, w))

    result, cycle = detect_negative_cycle(N, edges)
    if result == "YES":
        print(result)
        print(" ".join(map(str, cycle)))
    else:
        print(result)

```

Kết quả

Input:

```
4 5
1 2 1
2 4 1
3 1 1
4 1 -3
4 3 -2
```

Output:

```
YES
1 2 4 1
```

Giải thích:

- Chu trình âm được phát hiện là: $1 \to 2 \to 4 \to 1$.
- Các bước truy vết cho phép xác định chu trình đúng thứ tự.