Lab07-Network Flow

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2019.

* Name: ChenxiaoYang Student ID: 517021910540 Email:cxyang1@outlook.com

1. Remember the network problems in last lab?

   Consider there is a network consists n computers. For some pairs of computers, a wire i exists in the pair, which means these two computers can communicate with delay t_i.

   The distance of two computers are defined as their communication delay. Design an algorithm to compute the maximum distance in the network.

   You need to provide the pseudo code and analyze the time complexity.

   Solution.

   Algorithm 1: Maximum Distance Algorithm

   Input: Graph G = (V, E), (computer $\in V$, wire $\in E$).

   Output: The maximum distance in the network.

   ```
   for i ← 1 to n do
       for j ← 1 to n do
           if wire w connects i, j then
               c_ij = t_w;
           else
               c_ij = ∞;
   for k ← 1 to n do
       for i ← 1 to n do
           for j ← 1 to n do
               c_ij = c_ik + c_kj;
   return max value of matrix C;
   ```

   The algorithm uses Floyd Algorithm to compute the distance of each pair of computers, and then find the maximum value of those distances. The time complexity of Floyd Algorithm is $O(n^3)$ and finding maximum value is $O(n^2)$. Therefore, the total time complexity is $O(n^3)$ □

2. Suppose you are traveling through a country defined as a directed graph $G = (V, E)$. You start from vertex s and want to go to e. For each $i \in E$, there is a w_i regarding the cost for traveling via i. Some times $w_i < 0$ which means you can earn money by traveling.(Do not ask why.) Here you need to design an algorithm satisfying the following demands:

   (a) Find the minimum cost from s to e. The problem guarantee that there is a path from s to e.

   (b) Indicate whether there is a circle in G, that by traveling through this circle you can earn money continually.

   You need to provide the pseudo code and analyze the time complexity.

Solution.

Algorithm 2: Traveling Minimum Cost Algorithm

Input: Graph G = (V, E), vertices s and e.

Output: Whether there is a negative circle and the minimum cost from s to e.

```
for node u ∈ V do
    M[0, u] = ∞;
M[0, s] = 0;
for i ← 1 to n do
    for node v ∈ V do
        if M[e] has been updated in previous iteration then
            for edge(e, v) ∈ E do
                M[i, e] = min{M[i − 1, e], M[i − 1, v] + w(e, v)};
if M[n, u] = M[n − 1, u] for all u ∈ V then
    return M[n, e];
else
    There is a negative cirlce in G;
```

□

3. (Bonus)Suppose you are a staff of SJTU who is in charge of arranging lessons. Suppose you have n time slots, n lessons and n professors. Clearly, you should assign exactly one time slot and one lesson to every professor. A lesson or a time slot should be assigned to exactly one professor.

For each professor, he will prefer some certain time slots among these n time slots, and prefer to taught some certain lessons among these n lessons.

A professor will be satisfied iff you arrange him both his preferred lesson and preferred time slot. Your goal is to satisfy as many professors as you can. Design an algorithm to output how many professors can you satisfy at most.

Solution. We construct a graph with n time slot nodes, n lesson nodes, n pre-professor nodes and n post-professor nodes. In the graph, directed edges of capacity 1 are used to connect them.

For the time slots and lessons that a professor prefers, we connect the time slot nodes to pre-professor node, post-professor node to lesson nodes. Also, pre-professor node is connected to the corresponding post-professor node. A source node connects to all time slot nodes. All lesson nodes connects to a target node. An example is shown in Figure 1.

To compute how many professors can you satisfy at most, we only need to compute the maximal flow of this graph.

The connection between pre-professor node and post-professor node ensures that each professor can only choose one lesson and one time slot.

Similarly, the source-timeSlots and lessons-target connection ensures that each lesson and time slot can only be chosen once.

If the flow value of each edge is either 0 or 1, then a sub-flow from source to target with value 1 represents that the professor in this sub-flow is satisfied. According to the Integrality

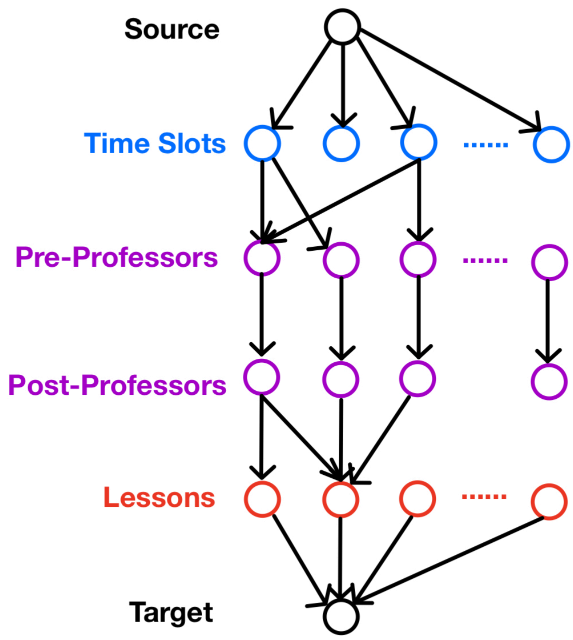

Figure 1: An example of constructed flow

Theorem in course slide, we can always convert a maximal flow with non-integer flow value to a flow with only value 0 or 1. Therefore, this method is correct.

To compute the maximal flow of this graph, we can use Ford-Fulkerson algorithm with time complexity of $O(mn)$.

$\square$