2019-1 Computer Algorithms Homework #6

Dae-Ki Kang

June 3, 2019

(Deadline : June 18)

1. Describe the algorithm that solves the following problem in pseudo code. Your algorithm has to use a network flow function (like `Ford-Fulkerson(G)`, `Edmonds-Karp(G)`, `Push-Relabel(G)`, or `Dinic(G)`) to find a max flow in a flow network. You have to explain your algorithm in detail, but you don't have to explain the network flow functions in detail, because they are already well-known.

   Mina wants to go on a trip during the a summer vacation.

   She has to pay in advance several things such as airfare, lodging, car rent, show tickets, etc. To pay them, she wants to use her mileage points from her credit card, airline, cell phone, etc.

   She can divide mileage points from one source (ex. airline mileage) to pay multiple things (ex. airfare and tickets). Also she can merge mileage points from multiple sources (ex. airline and cell phone) to pay one thing (ex. car rent).

   However, there can be a limitation in applying certain mileages to certain type of payments. For example, suppose she has 100,000 (in Korean Won) airline mileage and 50,000 cell phone mileage, and she wants to pay 90,000 for airfare and 70,000 for car rent [1], there can be a limitation that she can use airline mileage for both airfare and car rent, but she only can use cell phone mileage for car rent.

   With that limitation, it is not a good idea to use only airline mileage to pay for car rent, because, after that, she needs to pay 60,000 for car rent (after she use 90,000 for airfare and 10,000 for car rent from 100,000 airline mileage). Instead, it is a reasonable choice to use 50,000 cell phone mileage for car rent first, and then use 100,000 airline mileage for airfare and car rent, which results her to pay 10,000 for either car rent or airfare.

   Now given a set of mileages, a set of payments, and lists of feasible payments for each mileage, calculate the minimum amount of money that Mina has to pay.

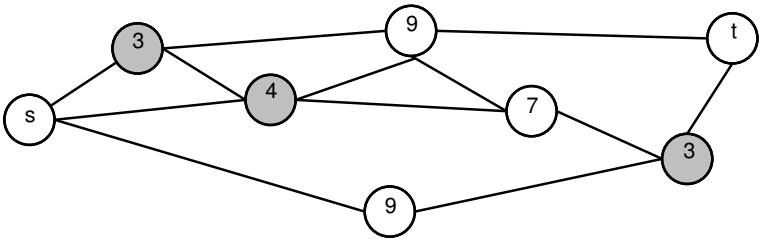
```
Input
2     <-- # of mileages
10 5  <-- mileage #0 and #1
2     <-- # of payments
9 7   <-- payment #0 and #1
0 1   <-- mileage #0 can be used for payment #0 and #1
1     <-- mileage #1 can be used for #1

Output
1     <-- minimum of payment 1 is needed
```

[1] Let me know if you know any nice place with these reasonable costs!

2. Describe the algorithm that solves the following problem in pseudo code. Your algorithm has to use a network flow function (like `Ford-Fulkerson(G)`, `Edmonds-Karp(G)`, `Push-Relabel(G)`, or `Dinic(G)`) to find a max flow in a flow network. You have to explain your algorithm in detail, but you don't have to explain the network flow functions in detail, because they are already well-known.

   Mina, the mysterious thief, has robbed a museum in a town s of their art collection. The police has found out that Mina tries to move the collection from the town s to a port town t. Since the art collection is too big to move by a truck, the police can easily guess that Mina will use a freight train.

   One example of the train network from s to t is as follows:



   The network has several stations and the police puts several policemen in stations except s and t to detect a suspicious train. The number of policemen needed for a station depends on the complexity of the station. The number inside a vertex (which denotes a station) is the number of necessary policemen to monitor the station.

   You are directing the police team. You need to deploy policemen to some stations, so that, for all possible paths from s to t, there are policemen to check trains. You want to economize the resources so you want to find the minimum number of policemen to accomplish this.

   For example, in the above train network, the vertices (or stations) in gray color are the stations which the policemen are assigned. As you can see, 10 policemen (3+4+3) are enough to monitor all possible paths from s to t.

```
Input
8               <-- # of stations, 0 is s and 7 is t.
0 3 4 9 9 7 3 0 <-- # of policemen needed for each vertex, 0 for s and t.
1 2 3           <-- adjacency list for vertex 0 (s)
0 2 4           <-- adjacency list for vertex 1
0 1 4 5
0 6
1 2 5 7
2 4 6
3 5 7
4 6             <-- adjacency list for vertex 7 (t)

Output
10
```

3. Describe the algorithm that solves the following problem in pseudo code. Your algorithm has to use a network flow function (like `Ford-Fulkerson(G)`, `Edmonds-Karp(G)`, `Push-Relabel(G)`, or `Dinic(G)`) to find a max flow in a flow network. You have to explain your algorithm in detail, but you don't have to explain the network flow functions in detail, because they are already well-known.

   To make a lot of easy money, Mr. DSJo, a stupid C programmer, is trying to do match fixing in a national programming league. Supported by one big politician, he asks all N players in the league to help him in the match fixing.

   The league is composed of many 1 vs 1 games. Every game ends with a win/lose result (i.e. no tie result). After the league, the player with most wins will be the champion. If top two players have the same number of wins, then there will be two champions.

   DSJo is trying to set the match-fixing up to make one player X to become a champion, who has been least expected to be a champion by gamblers. Since it will look suspicious if X becomes a champion with too high win number, DSJo tries to find the minimum number of wins for X to become a champion. However, since DSJo has learned nothing but alcohol drinking when he was in school, he does not know what to do.

   Given a set of wins for each player and remaining games, write a program that determine if it is possible to do match-fixing and calculate the minimum number of wins for X whose player number is 0. [2]

```
Input
3       <-- 3 test cases (<=50)

2 2     <-- 2 players (2<=N<=12) and 2 remaining games (0<=M<=100)
3 3     <-- player 0 has 3 wins so far, and player 1 has 3 wins so far.
0 1     <-- 1st of the 2 remaining games is player 0 vs player 1.
0 1     <-- 2nd of the 2 remaining games is player 0 vs player 1.

3 3     <-- 3 players and 3 remaining games
4 2 2   <-- player 0: 4 wins, player 1: 2 wins, player 2: 2 wins
1 2     <-- 1st of the 3 remaining games is player 1 vs player 2.
1 2     <-- 2nd of the 3 remaining games is player 1 vs player 2.
1 2     <-- 3rd of the 3 remaining games is player 1 vs player 2.

4 4     <-- 4 players and 4 remaining games
5 3 3 2 <-- player 0: 5 wins, player 1: 3 wins, player 2: 3 wins, player 3: 2 wins
0 1     <-- 1st of the 4 remaining games is player 0 vs player 1.
1 2     <-- 2nd of the 4 remaining games is player 1 vs player 2.
2 3     <-- 3rd of the 4 remaining games is player 2 vs player 3.
1 3     <-- 4th of the 4 remaining games is player 1 vs player 3.

Output
5       <-- 5 wins for the 1st case
-1      <-- impossible for the 2nd case
5       <-- 5 wins for the 3rd case
```

[2] We just calculate it from curiosity. We don't have to let it know to stupid DSJo. :)

4. Describe the algorithm that solves the following problem in pseudo code. Your algorithm has to use a network flow function (like `Ford-Fulkerson(G)`, `Edmonds-Karp(G)`, `Push-Relabel(G)`, or `Dinic(G)`) to find a max flow in a flow network. You have to explain your algorithm in detail, but you don't have to explain the network flow functions in detail, because they are already well-known.

   Again, Mr. wicked DSJo has founded a company named USD Corporation. USD Corporation is trying to apply for some national projects. Wicked DSJo has flattered one corrupted politician to influence the evaluation of the project proposals, so it is always 100% possible to earn any national projects if USD Corporation applies for them.

   But not all national projects are profitable. Because a project needs some equipments and money is needed to buy equipments.

   For example, the following is a table of national projects that USD Corporation is considering.

Table 1: National Projects

Project Name	Expected Profit	Equipments
CK	260	Starport
LINC	60	Starport, Control Tower, Armory
ACE	140	Factory, Armory
KMOVE	350	Factory
BK	500	Factory, Machine Shop

   And this is a table of equipments that USD Corporation needs to accomplish a certain project [3].

Table 2: Equipments

Equipments	Price
Starport	250
Control Tower	100
Armory	150
Factory	300
Machine Shop	100

   If one equipment is used for more than one project, you can buy only one.

   Here, net profit is expected profit minus equipment price. For example, if USD Corporation accepts to work on ACE and KMOVE project, expected profit is 490 and, since Factory and Armory cost 450, the net profit is 40.

   Now, write the program that automatically calculate maximum net profit from the given national projects and equipments.

```
Input
2       <-- # of test cases (<=50)

2 2     <-- # of national projects (<=100) and # of equipments (<=100)
10 10   <-- expected profits of national projects #0 and #1
5 10    <-- prices of equipments #0 and #1
1 0     <-- for project #0, need to buy the equipment #0
1 1     <-- for project #1, need to buy the equipment #0 and #1

5 5     <-- Case 2: The format is the same as above.
260 60 140 350 500
250 100 150 300 100
1 0 0 0 0
1 1 1 0 0
```

[3] It may look like from Starcraft game things, but, trust me, it isn't. So don't worry and please take this as a realistic situation!

```
0 0 1 1 0
0 0 0 1 0
0 0 0 1 1

Output
5
460
```