CS6363 - Design and Analysis of ~~Con~~ Algorithms

Homework-5

Name: T. HIMA SRI

UTDID: 2021624644

1) Solve Exercise 25.2-6 on page 700 in CLRS:
How can we use the output of the Floyd-Warshall algorithm to detect the presence of a negative-weight cycle?

Sol: We use the Floyd-Warshall algorithm to compute the all pairs shortest paths.

I/P: ~~A matrix~~ Set of vertices V; A weight matrix W which contains the distance btw. the vertices.

O/P: A matrix showing the shortest path distance btw. any 2 pair of vertices.

Recursive solution:

Let $d_{ij}^{(k)}$ → weight of ~~the~~ a shortest path from vertex i to vertex j for which all intermediate vertices are in set {1, 2, --- k}.

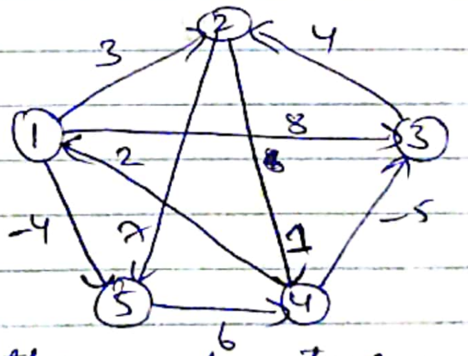
$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0. \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1 \end{cases}$$

Algorithm:

```
FLOYD_WARSHALL(W)
    n = W.rows          // no. of vertices
    D^(0) = W           // ∵ for k=0, d_ij^(k) = w_ij
    for k = 1 to n
        let D^(k) = (d_ij^(k)) be a new n×n matrix.
        for i = 1 to n
            for j = 1 to n
                d_ij^(k) = min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
    return D^(n).
```

Running time:
Since the algorithm has 3 loops, it takes $O(n^3)$.

Eg Given graph: G:



Executing the algorithm each steps, we obtain the following matrices.

$$D^{(0)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(1)} = \begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$D^{(2)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix} \qquad D^{(3)} = \begin{pmatrix} 0 & 3 & 8 & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

$$D^{(4)} = \begin{pmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix} \qquad D^{(5)} = \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

Methods to detect -ve weight cycle from Floyd-Warshall algorithm output:

① We know that
$d_{ij}^{(k)}$: represents a path from i to j with distance d_{ij} comprising intermediate vertices $\{1, 2, \ldots k\}$

Therefore, there is a negative weight cycle if and only if $d_{ii}^{(n)} < 0$ for some vertex i.

$d_{ii}^{(n)} \rightarrow$ Is a path weight from vertex i to itself containing all intermediate vertices $\{1, \ldots, n\}$

If this value is negative; it means that -ve weight cycle exists on vertex i.

<u>Case 1</u>: If the negative weight cycle has 1 vertex; i.e $d_{ii}^{(0)} = $ -ve which is $w_{ii} < 0$ hence d_{ii} starts out negative and since d

values are never increased, it is also negative when algorithm terminates

Case 2: If the negative weight cycle has at-least 2 vertices, let 'i' be the vertex which has -ve weight cycle.

We know that $d_{ii}^{(k)}$ is calculated as below:

$$d_{ii}^{(k)} = \min(d_{ii}^{(k-1)}, d_{ik}^{(k-1)} + d_{ki}^{(k-1)})$$

Let k be the highest numbered vertex on the cycle. Neither $d_{ik}^{(k-1)}$ & $d_{ki}^{(k-1)}$ can include k as the intermediate vertex; and i & k are on the negative-weight cycle. Therefore $d_{ik}^{(k-1)}$ & $d_{ki}^{(k-1)}$ have correct shortest path weights.

Since $i \to k \to i$ is a negative weight cycle, the sum of those 2 weights is negative, so $d_{ii}^{(k)}$ is set to a negative value. Since $d_{ii}^{(k)}$ values are never increased, the algorithm terminates in a negative value for $d_{ii}^{(n)}$.

2) Alternatively, we could just run the FLOYD-WARSHALL algorithm for one extra iteration.

If there are negative-weight cycles, some d values will change.

Otherwise, there will be no change in the d values obtained earlier

2) Solve Exercise 26.1-6 on page 714 in CLRS.

Professor Adam has 2 children who, unfortunately dislike each other. The problem is so severe that not only do they refuse to walk to school together, but in fact each one refuses to walk on any block that other child has stepped on that day. The children have no problem with their paths crossing at a corner. Fortunately both the professor's house and school are on corners, but beyond that he is not sure if it is going to be possible to send both of his children to the same school. The professor has a map of his town. Show how to formulate the problem of determining whether both his children can go to the same school as a maximum-flow problem?

Sol: Given:

Children cannot walk on the same block/path.

Children can have same crossing.

House and School are at 2 corners.

To show:

Formulate the problem of determining whether both children can go to same school as a maximum flow problem.

Flow Network:

A flow network $G=(V,E)$ is a directed graph in which each edge $(u,v) \in E$ has a non-negative capacity $c(u,v) \geq 0$.

In the given problem, let us take

Source : s = Corner of the Professor's house

Sink : t = Corner on which the School is located.

Let us define the flow network of this problem as below:

$G=(V,E)$

V = Set of vertices

Create a vertex for each corner

$\therefore$ Set of Vertices = set of all corners.

E = Edges between the vertices.

If there is a street/block between corners u and v, create directed edges (u,v) and (v,u) accordingly.

$\therefore$ G is a network with each corner as vertex and the street between the corners as edges.

Let us say capacity of each edge $(u,v) \in E$ is one because both children will not ~~cross~~ move on the same path.

1. $c(u,v) = 1$.

Now, the maximum flow problem comes down to below as:

We wish to find the maximum flow of 2 units between source & sink where the capacity of each edge is 1.

Thus, we can make the children move from professor house (source) to school (sink) with the capacity of only one child through each edge/street.

The flow $f(u,v)$ in the network should follow the capacity constraint i.e

$$0 \leq f(u,v) \leq c(u,v)$$

And also, it satisfies the flow conservation:

For all $u \in V - \{s,t\}$

$$\sum_{v \in V} f(u,v) = \sum_{v \in V} f(v,u)$$

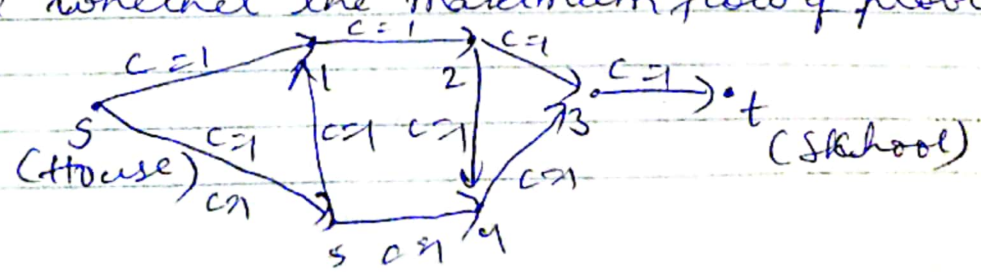
Therefore, the maximum flow of this problem is:

$G: (V,E)$ where V is the corners of the town, $E \rightarrow$ street between the corners i.e directed edge (u,v) btw the corners u,v.

Capacity of each edge $c(u,v) = 1$

Find whether the maximum flow of problem is 2.

Eg.



3) Solve Exercise 26.2-9 on page 731 in CLRS.

Suppose that both f and f' are flows in a network G and we compute flow $f \uparrow f'$. Does the augmented flow satisfy the flow conservation property? Does it satisfy the capacity constraint?

Sol: Given that:

f & f' are flows in a network G.

Augmented flow $f \uparrow f'$ is computed.

To show:

Does Augmented flow satisfy flow conservation property?

Does Augmented flow satisfy capacity constraint

Capacity Constraint:

For a flow f in a ~~regat~~ network G,

$f: V \times V \rightarrow R$.

For $(u, v) \in V$, we require $0 \leq f(u,v) \leq c(u,v)$ where $c(u,v)$ is the capacity btw. the vertices u & v.

Flow Conservation:

For all $u \in V - \{s,t\}$ we require.

$$\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$$

where s & t are source & sink vertices
In other words;
Net Incoming flow ~~b/w 2 vertices~~ on any edge
= Net Outgoing flow ~~b/w 2 vertices~~ on any edge.

Similarly,
Augmentation of a flow f by f'; is defined as

$$f \uparrow f' : V \times V \rightarrow R$$

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{if } (u,v) \in E; \\ 0 & \text{otherwise} \end{cases}$$

a) To check whether it satisfies Flow Conservation property:

Note that f and f' are two flows in a network G.
And for any network G; we know that there are no anti-parallel edges

$\therefore$ if $(u,v) \in E$; then $(v,u) \notin E$ and $f(v,u) = 0$.

Hence, we can define augmentation flow of f & f' as:

~~since $f'(v,$~~

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) & \text{if } (u,v) \in E \\ 0 & \text{otherwise} \end{cases}$$

Since f' is a flow in network G; $f'(v,u) = 0$

To check the flow-conservation property, let us calculate $\sum_{v \in V} f \uparrow f'(v,u)$ for $u \in V - \{s,t\}$

$$\sum_{v \in V} f \uparrow f'(v,u) = \sum_{v \in V} \left[f(v,u) + f'(v,u) \right]$$

$$= \sum_{v \in V} f(v,u) + \sum_{v \in V} f'(v,u) \quad \text{--- (1)}$$

Since f & f' are flows of network G, they follow flow-conservation.

$$\Rightarrow \sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$$

$$\text{||}^{ly} \quad \sum_{v \in V} f'(v,u) = \sum_{v \in V} f'(u,v).$$

Therefore (1), can be replaced as

$$\sum_{v \in V} f \uparrow f'(v,u) = \sum_{v \in V} f(u,v) + \sum_{v \in V} f'(u,v)$$

$$= \sum_{v \in V} \left(f(u,v) + f'(u,v) \right)$$

$$= \sum_{v \in V} \left(f \uparrow f'(u,v) \right)$$

(by augmentation flow def'n)

$\therefore$ We arrived at

$$\sum_{v \in V} f \uparrow f'(v,u) = \sum_{v \in V} f \uparrow f'(u,v)$$

satisfies.

$\Rightarrow$ This proves that $f \uparrow f'$ ~~follows~~ Flow-Conservation property

b) To check whether $f \uparrow f'$ satisfies Capacity constraint:

$\Rightarrow$ We ~~arrived~~ ~~derived~~ know that

$$f \uparrow f'(u,v) = \begin{cases} f(u,v) + f'(u,v) & \text{if } (u,v) \in E \\ 0 & \text{otherwise.} \end{cases}$$

Let us a consider a network G with a single edge (s,t)

Let capacity of that edge $c(s,t) = 1$.

If we define our flows $f(s,t) = 1$ & $f'(s,t) = 1$.

Then, we have
$$\begin{aligned} f \uparrow f'(s,t) &= f(s,t) + f'(s,t) \\ &= 1 + 1. \\ &= 2 \\ &> c(s,t). \end{aligned}$$

This counterexample shows that $f \uparrow f'(s,t) > c(s,t)$ which violates the capacity constraint.

Hence, the augmented flow $f \uparrow f'(u,v)$ does not satisfy capacity constraint.