Home Assignment 1
Advanced Algorithms and Data Structure

Siyi Wu	Zhongxing Ren	Yu Pei	Qianxi Yang
KRW521	CPJ395	MQH875	HSB264

December 6, 2022

1 Group Part

1.1 Answer of exercise 26.1-1 in CLRS

Solution1:
Suppose the maximum flow of a $G = (V, E)$ is flow f, if after replacing (u,v) with new edges (u,x) and (x,v) and getting a new flow network $G' = (V', E')$, the flow conservation property still holds because the flow in and out of v_1 and v_2 do not change. Second, assume the number of cuts across the edge (u,v) in the original graph G is n. After splitting, there will be n cuts across the edge (u,x) and n cuts across (x,v). Since $c(u, x) = c(x, v) = c(u, v)$, $|f| = \sum_{v \in V} f(s, v)$ remains the same. Therefore, the maximum flow of $G' = (V', E')$ remains the same as G.

Solution2:
Suppose the set of min-cut distinguished by G is S, T. If the min-cut edge does not contain (u, v) then u, v, x nodes are in S or in T. In this case, the capacity of the min-cut is not affected, that is, the maximum flow $|f|$ remains unchanged. If the min-cut edge contains (u, v), then when the new node is added, there are two cases: 1. The new node x is added to S, then (x, v) replaces (u, v) when calculating the capacity of the cut, because $c(u, v) = c(x, v)$ so the capacity of the cut remains the same and the maximum flow $|f|$ is also the same.
2. The new node x is added to T, similarly (u, x) replaces (u, v), because $c(u, v) = c(u, x)$ so the capacity of the cut capacity remains the same, and the maximum flow $|f|$ remains the same.

1.2 Answer of exercise 26.1-4 in CLRS

To prove that $\alpha f_1 + (1 - \alpha) f_2$ is a flow, it must be shown to satisfy conservation of **Capacity constraint** and **Flow conservation**.

Proof of Capacity constraint:
First let $f = \alpha f_1 + (1 - \alpha) f_2$, so we have:

$$\forall u, v \in V, f(u, v) = \alpha f_1(u, v) + (1 - \alpha) f_2(u, v) \tag{1}$$

And because the value domain of $f_1(u, v)$ is $[0, c(u, v)]$, the value domain of $f_2(u, v)$ is $[0, c(u, v)]$, and the value domain of α is $[0, 1]$. So the question becomes a problem of finding the limit of a quadratic equation.

When $f_1(u, v)$ and $f_2(u, v)$ both take $c(u, v)$, $f(u, v)$ has a maximum value over the domain of definition:

$$f_{max}(u, v) = \alpha c(u, v) + (1 - \alpha) c(u, v) = c(u, v) \tag{2}$$

Similarly, when $f_1(u, v)$ and $f_2(u, v)$ both take $0 f(u, v)$ has a minimum value over the domain of definition:

$$f_{min}(u,v) = \alpha \times 0 + (1-\alpha) \times 0 = 0 \tag{3}$$

Therefore the value domain of f is $[0, c(u,v)]$.

Proof of Flow conservation:
By definition,

$$\begin{aligned}\sum_{v\in V} f(v,u) &= \sum_{v\in V}(\alpha f_1(v,u) + (1-\alpha) f_2(v,u)) \\ &= \alpha \sum_{v\in V} f_1(v,u) + (1-\alpha)\sum_{v\in V} f_2(v,u)\end{aligned} \tag{4}$$

$$\begin{aligned}\sum_{u\in V} f(u,v) &= \sum_{u\in V}(\alpha f_1(u,v) + (1-\alpha) f_2(u,v)) \\ &= \alpha \sum_{u\in V} f_1(u,v) + (1-\alpha)\sum_{u\in V} f_2(u,v)\end{aligned} \tag{5}$$

Because,

$$\alpha \sum_{v\in V} f_1(v,u) = \alpha \sum_{u\in V} f_1(u,v) \tag{6}$$

And,

$$(1-\alpha) \sum_{v\in V} f_2(v,u) = (1-\alpha) \sum_{u\in V} f_2(u,v) \tag{7}$$

Therefore,

$$\sum_{v\in V} f(v,u) = \sum_{u\in V} f(u,v) \tag{8}$$

1.3 Answer of exercise 26.1-7 in CLRS

It is sufficient to use the following structure instead of each node in the original diagram.

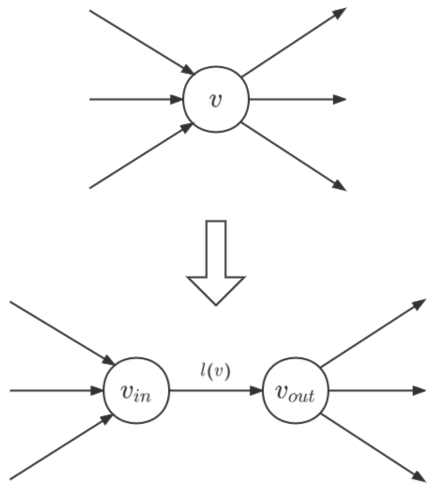

Figure 1: Substitution Structure

Because in this structure, any flow passing through v must pass through the edges (v_{in}, v_{out}), in order to get from the edges going into v to the edges going out of v. Therefore, v_{in} and v_{out} are equivalent to bound together, and as long as the flow passes through v_{in}, it must also pass through v_{out}. In addition, the edge (v_{in}, v_{out}) only limits the node capacity and does not affect $c(u, v)$ and $c(v, u)$ in the original graph, so the maximum flow does not change as a result.

Therefore G' should have $2|V|$ nodes, $|E| + |V|$ edges.

1.4 Answer of exercise 26.2-2 in CLRS

The flow across the $cut(\{s, v_2, v_4\}, \{v_1, v_3, t\})$ is:
$f(S,T) = f(s, v_1) + f(v_2, v_1) + f(v_4, v_3) + f(v_4, t) - f(v_3, v_2)$
$= 11 + 1 + 7 + 4 - 4 = 19$
The capacity of this cut is:
$c(S,T) = c(s, v_1) + c(v_2, v_1) + c(v_4, v_3) + c(v_4, t)$
$= 16 + 4 + 7 + 4 = 31.$

1.5 Answer of exercise 26.2-3 in CLRS

The Edmonds-Karp algorithm is executed as follows.

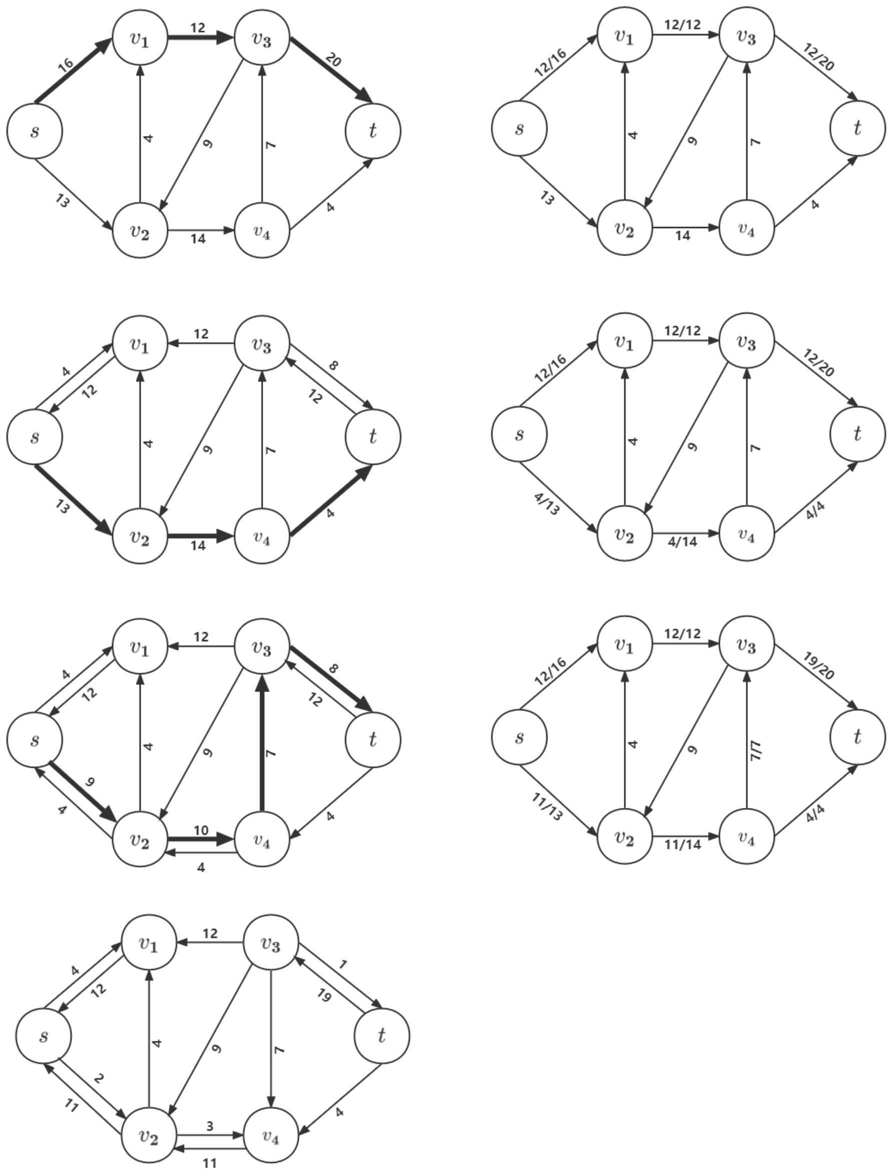

Figure 2: The execution of the Edmonds-Karp algorithm

1.6 Answer of exercise 26.2-4 in CLRS

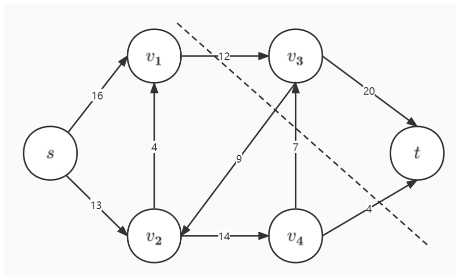

Figure 3: The minimum cut

A minimum cut (S,T) is $S = \{s, v_1, v_2, v_4\}$ and $T = \{v_3, t\}$. The capacity of this cut is $c(S,T) = 12 + 7 + 4 = 23$. The augmenting path in graph (c) cancels flow (v_3, v_2) in graph (a) and flow (v_2, v_1) in graph (b).

1.7 Answer of exercise 26.2-7 in CLRS

Proof:
Firstly, we prove that f_p satisfies capacity constraint:
Because $c_f(p) = min\{c_f(u,v) : (u,v) \in p\}$ (p is the augmentation path in residual network), $c_f(p) \le c_f(u,v), (u,v) \in p$. So for $(u,v) \in p$, $f_p(u,v) = c_f(p) \le c_f(u,v)$. For $(u,v) \notin p$, $f_p(u,v) = 0 < c_f(u,v)$. So for every edge (u,v) in residual network, $f_p(u,v) \le c_f(u,v)$. So f_p satisfies capacity constraint.
Secondly, we prove that f_p satisfies flow conservation:
For $u \in V - \{s,t\}$ in residual network, if u is on the augmentation path, then

$$\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v) = f_p(u,v) = c_f(p) \tag{9}$$

if u isn't on the augmentation path, then

$$\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v) = 0 \tag{10}$$

So, for $u \in V - \{s,t\}$ in residual network, the equation

$$\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v) \tag{11}$$

always holds. So f_p satisfies flow conservation.
So f_p is a flow in residual network G_f, and $|f_p| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s) = c_f(p) > 0$

1.8 Answer of exercise 26.2-9 in CLRS

If both f and f' are flows in a network G, then:

$$\begin{cases} 0 \leq f(u,v) \leq c(u,v) \\ 0 \leq f'(u,v) \leq c(u,v) \\ (f \uparrow f')(u,v) = f(u,v) + f'(u,v) \end{cases} \tag{12}$$

where $(u,v) \in E$ (E is the set of edge in network G), so

$$0 \leq (f \uparrow f')(u,v) \leq 2c(u,v) \tag{13}$$

so the augmented flow doesn't satisfy the capacity constraint and it can't be regarded as a flow in network G.
For f, the value of input flow of u ($u \in V - \{s,t\}$, V is the set of vertices of network G) equals to that of output flow of u. For f', the value of input flow of u ($u \in V - \{s,t\}$, V is the set of vertices of network G) equals to that of output flow of u. So, for $f \uparrow f'$, the value of input flow of u ($u \in V - \{s,t\}$, V is the set of vertices of network G) equals to that of output flow of u. So the augmented flow satisfies the flow conservation.

1.9 Answer of exercise 26.3-2 in CLRS

Proof:
Here we use induction to prove the therom 26.10. After the first loop of Ford-Fulkerson, because capacity function c takes on only integral values and all f(u,v) ($(u,v) \in E$, E is the set of edge in network) in the flow network are initially 0, so the capability of every edge in residual network only takes integral values, so $c_f(p)$ ($c_f(p) = min\{c_f(u,v) : (u,v) \in p\}$ where p is the augmentation path in residual network) takes on only integral values making all f(u,v) $(u,v) \in E$ of the flow network after augmentation are integral values (**line 5 to line 8 in FORD-FULKERSON**). So $|f|$ ($|f| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$) takes on only integral values.
Then, we assume that after nth loops, $|f|$ and all f(u,v) ($(u,v) \in E$, E is the set of edge in network) of the flow network are all integral values.
Next, because after nth loops $|f|$ and all f(u,v) ($(u,v) \in E$, E is the set of edge in network) of the flow network are both integral values, in (n+1)th loop, the capability of every edge in residual network only takes integral values, so $c_f(p)$ ($c_f(p) = min\{c_f(u,v) : (u,v) \in p\}$ where p is the augmentation path in residual network) takes on only integral values making all f(u,v) $(u,v) \in E$ of the flow network after (n+1)th loop are integral values (**line 5 to line 8 in FORD-FULKERSON**). So $|f|$ ($|f| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$) also takes on only integral values after (n+1)th loop.
So here we have proved that "after each loop in **FORD-FULKERSON** $|f|$

and all f(u,v) ($(u, v) \in E$, E is the set of edge in network) of the flow are all integers if the capacity function c takes on only integral values". So, as the flow that after the final loop of **FORD-FULKERSON**, the maximum flow also has the fact that $|f|$ and all f(u,v) ($(u, v) \in E$, E is the set of edge in network) of the maximum flow are all integers if the capacity function c takes on only integral values.

2 Own Summary of Each Group Member

2.1 Own summary of Siyi Wu (KRW521)

2.1.1 Flow network

1. Definition: $G = (V, E)$ is a directed graph in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$.
2. Capacity constraint: $\forall u, v \in V, 0 \leq f(u, v) \leq c(u, v)$
3. Flow conservation: $\forall u \in V - \{s, t\}, \sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$

2.1.2 Residual networks & Cuts

Residual networks

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & \text{if } (u, v) \in E \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

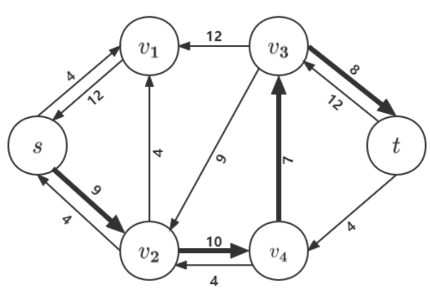

Figure 4: Residual network and an Augmenting path

Cuts

$$f(S,T) = \sum_{u\in S}\sum_{v\in T} f(u,v) - \sum_{u\in S}\sum_{v\in T} f(v,u) \tag{16}$$

$$c(S,T) = \sum_{u\in S}\sum_{v\in T} c(u,v) \tag{17}$$

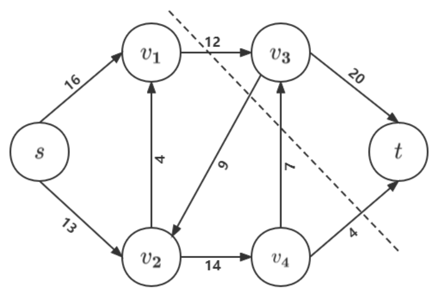

Figure 5: A minimum cut of a flow network

2.1.3 Ford-Fulkerson & Edmond-Karp

Ford-Fulkerson algorithm

1. Definition: Ford-Fulkerson algorithm is a way to calculate the maximum flow.

2. Time complexity: $O(E|f^*|)$

3. Depth first

Edmond-Karp algorithm

1. Definition: Edmond-Karp algorithm is a modified algorithm of Ford-Fulkerson algorithm

2. Time complexity: $O(VE^2)$

3. Breadth first

2.2 Own summary of Zhongxing Ren (CPJ395)

- Start with a question: There are two cities: city A and city B, there is a road network between them, but each road has a capacity of transporting the goods, so how to transport the goods as much as possible from city A to city B is a question. $->$ Abstract to "how to figure out the max flow in a flow network".
- Flow net work definition: **Capacity constraint** and **Flow conservation**
- Using an hourglass-shape graph to introduce an intuitive equivalent relation between value of max flow and minimum cut capacity.
- Definition of augmentation flow:

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{if } (u,v) \in E \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

- Rigorous derivation: Augmentation flow (will be used in **line 5 to line 8 in FORD-FULKERSON**) $->$ $|f| \leq c(S,T)$ $->$ $|f|$ rigorously increases after each loop until $|f|$ = some c(S,T) (min of c(S,T)) $->$ f is the max flow and its value $|f|$ = some c(S,T) (min of c(S,T))
- Definition of residual network:

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & \text{if } (u,v) \in E \\ f(v,u) & \text{if } (v,u) \in E \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

- More rigorous: Basic process of Ford-Fulkerson (augment flow in each loop until $|f|$ = some c(S,T) (min of c(S,T))): building residual network $->$ finding augmentation path $->$ figuring out f_p $->$ augment flow $->$ new loop until no augmentation path can be found in residual network, then we can get the max flow f and its value $|f|$.
- A sample for implementing the Ford-Fulkerson algorithm and Edmonds-Karp algorithm

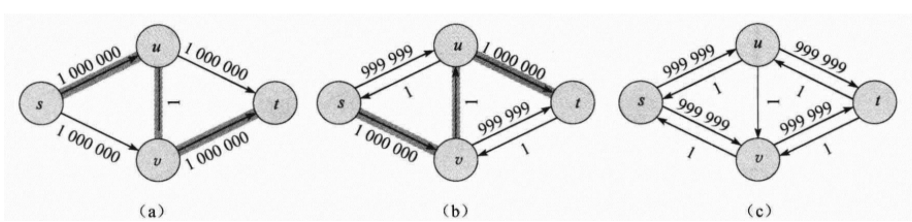

Figure 6: algorithm implementation (Figure 26-7 in CLRS [1])

And it is also a good sample to introduce the consuming time ($O(E|f^*|)$ and $O(VE^2)$) of these two algorithms (DFS, BFS in searching the augmentation path in residual network)

- Using of **Max Flow**: Maximum bipartite matching (adding source point and sink point additionally, unit capacity edges)

2.3 Own summary of Yu Pei (MQH875)

- Flow network: two properties, $|f|$
- Residual network G_f and residual capacity $c_f(u, v)$ of edge (u, v)
- Cut: flow $f(S, T)$ and capacity $c(S, T)$ of a cut
- Augmenting Path: residual capacity of path p: $c_f(p)$
- Max-Flow and Min-Cut
- Ford-Fulkerson Algorithm: time complexity
- Edmonds-Karp algorithm
- Algorithm implementation instance:

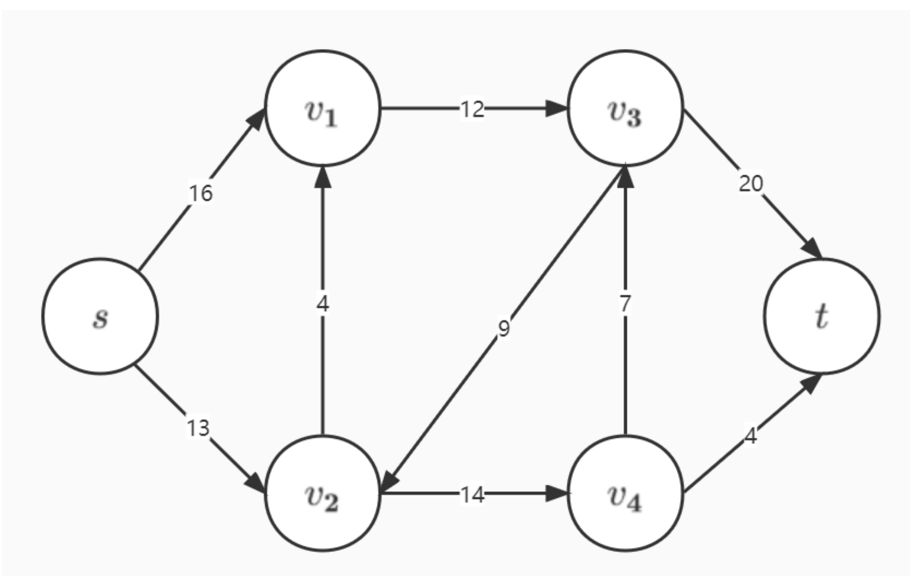

Figure 7: Algorithm implementation instance

- Maximum bipartite matching

2.4 Own summary of Qianxi Yang (HSB264)

2.4.1 Flow network and Max flow

A network is a directed graph G with vertices V and edges E connected with a function c, and each edge $e \in E$ has a non-negative integer value: the capacity of e. and we also need to additionally label two vertices, one as the source and one as the sink.

Beyond capacity, a flow in a flow network is a function f that again gives each edge e a non-negative integer value, namely the flow. The function has to fulfill the following two constraints: **Capacity constraint** and **Flow conservation**, one makes sure that the flow of an edge cannot exceed the capacity, and the other one makes sure that the sum of the incoming flow of a vertex v has to be equal to the sum of the outgoing flow of v.

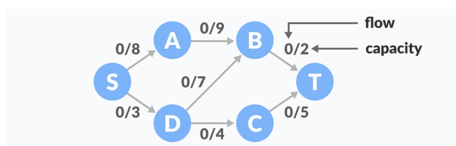

Figure 8: Flow network graph

The value of the flow of a network is the sum of all the flows that go through at the source s, and the Max flow is to Find this maximal flow of a flow network.

2.4.2 Ford-Fulkerson method

First, we need to define the concept of **residual capacity** and **augmenting path**, a residual capacity of a directed edge is the capacity minus the flow, and the augmenting path is a simple path in the residual graph, i.e. along the edges whose residual capacity is positive.

The Ford-Fulkerson method works as follows. First, we set the flow of each edge to zero. Then we look for an augmenting path from s to t. If such a path is found, we can increase the flow along these edges. Then we keep on searching for augmenting paths and summing the flow. Once we could not find another augmenting path, the flow is maximal.

2.4.3 Edmonds-Karp algorithm

Edmonds-Karp algorithm is just an implementation of the Ford-Fulkerson method that uses BFS for finding augmenting paths.

References

[1] Thomas H.Cormen, Charles E.Leiserson, Ronald L.Rivest, Clifford Stein, Introduction to Algorithms, third edition