Multi-terminal flows:

Similar to All-pairs SP.; but more difficult.

For this section, we will assume our graphs are Undirected. As of now, we do not have Corresponding results for the directed case.

This work is based on work of T.C. Hu and R.E. Gomory ('60's) and also found in the book by Ford and Fulkerson. [see also my notes, & work of Mayeda, book by Frank and Frisch}, and more recent version by W.K. Chen (a student of F&F]

There are three sections in this topic. $[v_{i,i} = \infty \; \forall i]$

Necessary & Sufficient Conditions

1. Given numbers $v_{i,j}$ $i = 1 \dots n$, $j = 1 \dots n$, $i \neq j$, are these values max flows from i to j ~~in~~ some ~~(Undirected) network~~ undirected (directed) network (graph)

i.e are there edge capacities in a graph $G = [V; E]$ with $|V| = n$ such that the resulting max flows are $\{ v_{i,j} \}$.

The problem on the previous page is called

Realizability Problem.

2. Find maximum flows $\{v_{i,j}\}$ in a graph $G = [V; E]$ with edge capacities $= u_{ij}$

$\downarrow$
undirected (directed?)

This is called the Analysis Problem

3. Design or Synthesis Problem

Find $\{u_{ij}\}$ such that max. flow between i and j is at least $v_{i,j}$ $\quad i = 1 \cdots n$, $j = 1 \cdots n$

& Minimize cost function of $\{u_{ij}\}$.

We will consider all three problems for undirected graphs; the case of directed graphs is more complicated is not completely resolved yet.

Promise # 2 in CS6363 — During this part of the course, we also prove Correctness of improvement algorithm for Min (Max) spanning Tree problem.

Spanning Tree Problem

Given an undirected graph $G=[V;E]$ with edgeweights
connected

$\left\{ r_{i,j} \; \begin{matrix} i\in V \\ j \in V \end{matrix}, i \neq j, (i,j)\in E \right\}$, find a spanning tree T whose total weight $\sum_{(i,j)\in T} r_{i,j}$ is maximum

---

Theorem A necessary and sufficient condition for a sp. tree T^* to be optimal is:

$$\left[(i,j) \notin T^*\right] \Rightarrow \left[\; r_{i,j} \leq \min_{(k,l)\in L^*_{i,j}} r_{k,l} = r_{pq} \qquad (I)\right.$$

Where $L^*_{i,j}$ is the unique cycle formed by adding $\underset{\notin T^*}{(i,j)}$ to T^*.

Proof: Necessity follows from the facts that if not

a) $\underbrace{T^* + (i,j) - (p,q)}_{=T} \in$ Sp. Tree

b) $\sum_{(k,l)\in T} r_{k,l} > \sum_{(k,l)\in T^*} r_{(k,l)}$

Contradiction to T^* max. Sp. Tree.

It is sufficiency that is harder.

To show sufficiency, we will show that any two spanning trees T_1 and T_2 that satisfy (I) have the same total weight. Actually we show a much stronger result. Let the weights of edges of T_1, T_2 (sorted in decreasing order) be

$$W_{T_1} = (w_1, w_2, \ldots, w_n) \quad w_i \geq w_{i+1} \quad i = 1 \ldots n-1$$

$$W_{T_2} = (w'_1, w'_2, \ldots w'_n) \quad w'_i \geq w'_{i+1} \quad i = 1 \ldots n-1$$

Then we show that $W_{T_1} = W_{T_2}$ (vectorially)

Edges of $T_1 \cup T_2$ (assuming $T_1 \neq T_2$) can be partitioned into 3 groups (disjoint sets)

black 1) $T_1 \cap T_2$ - edges (common to both)

2) T_1 - only edges
3) T_2 - only edges

$|T_1\text{-only}| = |T_2\text{-only}| \neq 0$ Since all span. trees have same # of edges

---

Assuming $T_1 \neq T_2$, let (x_1, x_k) be a T_1-only edge with weight α. Let the (unique) cycle C formed by adding (x_1, x_k) to T_2 contain edges

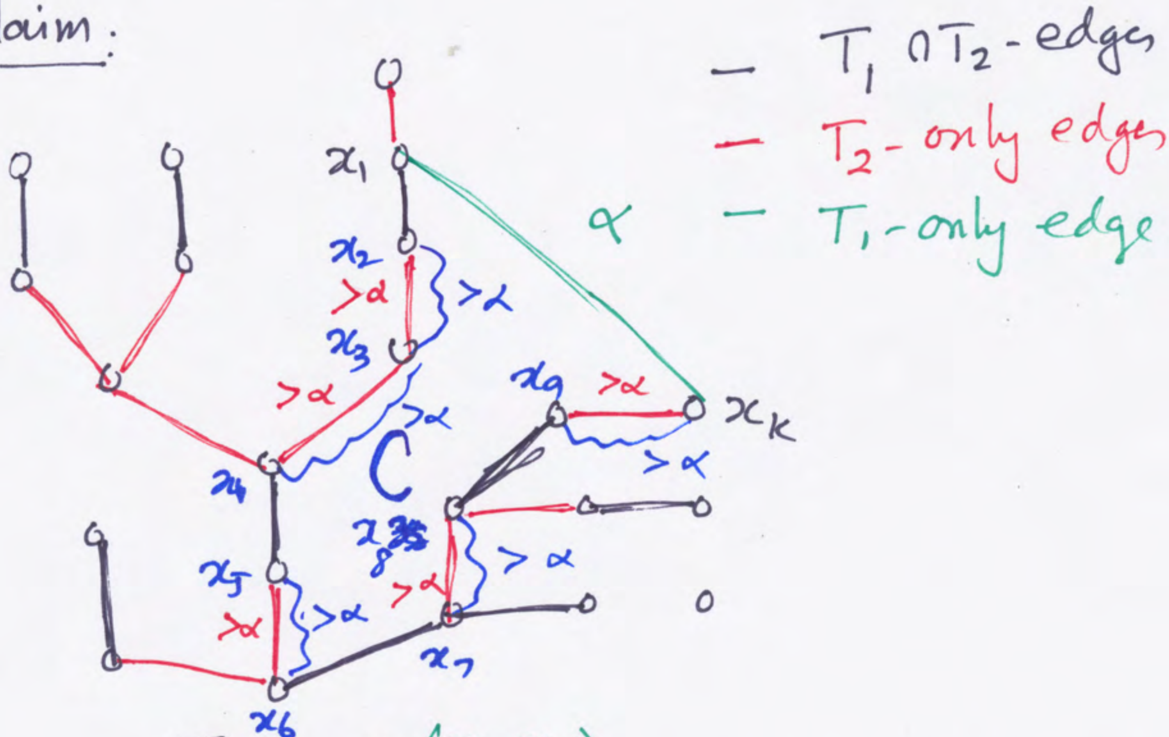
$$(x_1, x_2), (x_2, x_3) \cdots (x_{k-1}, x_k), (x_1, x_k)$$

(5)

<u>Claim</u> Among these edges in C, $\exists$ a T_2-only edge (x_i, x_{i+1}) whose weight is α.

<u>Remark</u>: $T_2 - (x_i, x_{i+1}) + (x_i, x_k) = T_2'$
is another spanning tree and set of edge weights of T_2 and T_2' are same. T_2' has one more edge in common with T_1. Repeating this process we can go from T_2 to T_1 with same set of edge weights which proves main result.

<u>Pf of Claim</u>:



$T_2 + (x_1, x_k)$

Since T_1 is a spanning tree, not all edges of cycle $+ (x_1, x_k)$ can be $T_1 \cap T_2$-edges. So there are T_2-only edges in C.

Because of I, if none of T_2-only edges in C have weight $= \alpha$, each edge of C that is a T_2-only edge has weight $> \alpha$. Each T_2-only edge in C forms a cycle when added to T_1. The remaining edges of such cycles are shown in blue. This creates one or more cycles in T_1 which is not possible.

All these edges have weight $> \alpha$ since (I) holds for both T_1 & T_2. (They are not (x_1, x_k)). Hence the claim and hence the theorem.

[This also shows correctness of Improvement algorithm for Max (min) Sp. trees. Promise #2 in CS6363]

Now we turn to Multiterminal Flows.

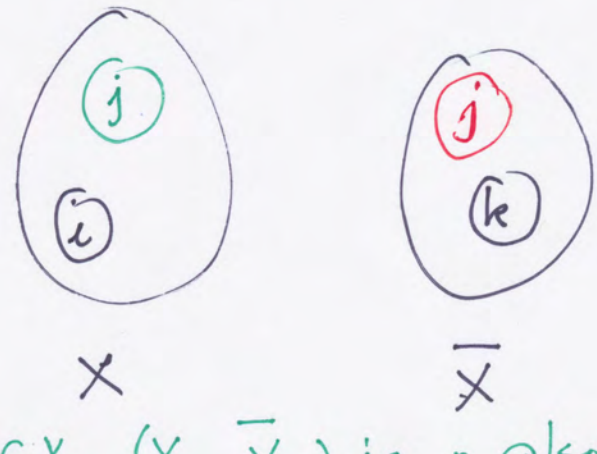
Since we are concerned with undirected graphs, $v_{i,j} = v_{j,i}$ $\forall i, j \in V$. ; Such v is called a Symmetric function. This condition is Necessary for these to be max flows in some undirected graph.

<u>Thm</u>: A necessary and sufficient condition for a nonnegative symmetric function v to be <u>realizable</u> is that it satisfy:

$$\text{(II)} \quad v_{i,k} \geq \min\{v_{i,j}, v_{j,k}\} \quad \forall \text{ ordered triples } (i,j,k),\ i \neq j \neq k,\ i,j,k \in V$$

Further if v is realizable, it is realizable by a tree graph

<u>Pf</u>: Let $(X, \overline{X} = V - X)$ be a min-cut that separates i and k. $[\, i \in X,\ k \in \overline{X} \,]$



Two possibilities for j indicated in green/red

In case $j \in X$, $(X, \overline{X})$ is also a cut that separates (j,k) $\therefore$ $v_{j,k} \leq U(X, \overline{X}) = v_{i,k}$

In case $j \in \overline{X}$, $(X, \overline{X})$ is also a cut that separates (i,j): $\therefore$ $v_{i,j} \leq U(X, \overline{X}) = v_{i,k}$.

$\therefore$ (II) follows in either case

Using (II) recursively, we have

$$v_{i,p} \geq \min\left[v_{i,j}, v_{j,k}, v_{k,l} \ldots v_{*,p}\right] \quad (\text{III})$$

Pf: Exercise left to you.

<u>Sufficiency</u>: Consider a (complete) graph (undirected) with edge weight v_{ij} which Satisfy (II) (and hence III)

Let T^* be a maximum spanning tree of G. Now consider T^* as input graph with Capacities equal to edge weights. If we solve maximum flow problem with this input we get $\{v_{i,j}\}$. To prove this, we use the fact since T^* is a max-sp. tree in G,

This also shows that realizable v can be realized on a tree graph

we use conditions for max sp. tree:

$$v_{i,p} \leq \min\left[v_{i,j}, v_{j,k}, \ldots v_{*,p}\right]$$

where $(i,p) \notin T^*$, and the rest are in the Cycle formed by adding (i,p) to T^*.

This together with (III) gives equality.

Thus, for an undirected graph, there at most $n-1$ distinct values among $\{v_{ij}\}$.

This completes realizability problem for undirected graphs. We now turn to Analysis Problem.

Two n-node graphs are *equivalent* if the same flow function v. We have just shown that every n-node graph is *equivalent* to a tree on n-nodes. The Analysis problem finds such a tree ~~that~~ is equivalent to a given undirected capacitated graph. This involves only $(n-1)$ maximum flow problem often on smaller graphs.

The process involves an operation that produces "Condensed" graphs. It involves "Contracting" some edges which is equivalent to setting their capacities $= \infty$. Condensing a set B of nodes = setting $u_{ij} = \infty \ \forall i,j \in B$.

Such a process can not decrease max flow for any pair of nodes. The use of this process will be clear later. We need a lemma before we proceed and it is the basis for condensing a set of nodes.

<u>Lemma</u>: Let $(X, \bar{X})$ be a min-cut separating s and t in G. Let p and q be nodes in G, $p \neq q$; $p \in X$, $q \in X$. Then $\exists$ a min-cut $(S, \bar{S})$ separating p and q in G such that either $\bar{X} \subseteq S$ or $\bar{X} \subseteq \bar{S}$.
[In other words nodes of $\bar{S}$ are <u>not</u> "split" by $(S, \bar{S})$]. This in turn implies that when we solve max-flow problem for p and q, we can "condense" nodes in $\bar{X}$.

<u>Pf</u>: Let $(Y, \bar{Y})$ be a min-cut separating p and q in G.
Since $p \in X$, $q \in X$, and p and q are separated by $(Y, \bar{Y})$, we may assume without loss, $p \in X \cap Y$; $q \in X \cap \bar{Y}$.

We may also assume without loss, that $s \in X \cap Y$.

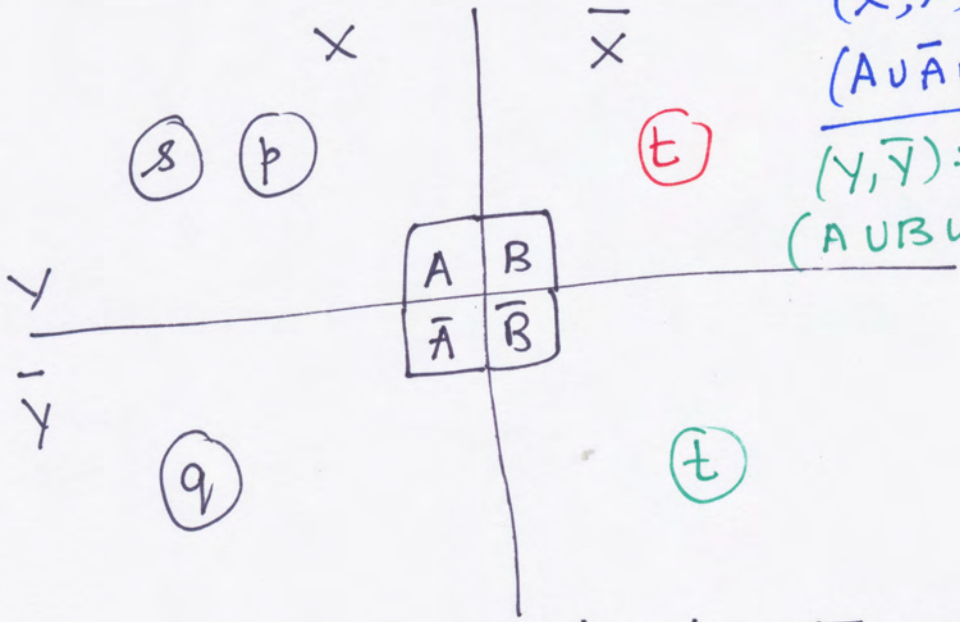
Let us denote ~~by~~: $X \cap Y = A$; $X \cap \overline{Y} = \overline{A}$;

$\overline{X} \cap Y = B$; $\overline{X} \cap \overline{Y} = \overline{B}$.

Hence $s \in A$; $p \in A$; $q \in \overline{A}$

$(X, \overline{X})$: min-cut for $(s, t$
$(A \cup \overline{A} \cup \overline{B}, B)$: a cut ,,

$(Y, \overline{Y})$: min-cut for $(p, q$
$(A \cup B \cup \overline{B}, \overline{B})$: a cut for(



There are two cases: (i) $t \in \overline{X} \cap Y = B$ red
(ii) $t \in \overline{X} \cap \overline{Y} = \overline{B}$ green

We treat Case (i) ; Case (ii) is similar and is left to you.

Next page

$$U(X,\bar{X}) = U(A,B) + U(\bar{A},B) + U(A,\bar{B}) + U(\bar{A},\bar{B})$$
$$\leq U(A \cup \bar{A} \cup \bar{B}, B) \quad \text{since this cut separates } s \text{ from } t$$
$$= U(A,B) + U(\bar{A},B) + U(\bar{B},B)$$

Min cut sep s from t

Hence $U(A,\bar{B}) + U(\bar{A},\bar{B}) - U(\bar{B},B) \leq 0$ (I)

Similarly, since $(Y,\bar{Y})$ is a min-cut separating p and q, and $(A \cup B \cup \bar{B}, \bar{A})$ is also a cut separating p & q, we have

$$U(Y,\bar{Y}) = U(A,\bar{A}) + U(A,\bar{B}) + U(B,\bar{A}) + U(B,\bar{B})$$
$$\leq U(A \cup B \cup \bar{B}, \bar{A})$$
$$= U(A,\bar{A}) + U(B,\bar{A}) + U(\bar{B},\bar{A})$$

Which implies

$$U(A,\bar{B}) + U(B,\bar{B}) - U(\bar{B},\bar{A}) \leq 0 \quad \text{(II)}$$

Recall $U(Z,\bar{Z}) = U(\bar{Z},Z)$ since G is undirected. Adding (I) & (II) we get and using we get

$$2U(A,\bar{B}) \leq 0$$

But $U \geq 0$; $U(A,\bar{B}) = 0$. (III)

Using this in I) and (II) we get (13)
(III)

$$U(\bar{A},\bar{B}) - U(\bar{B},B) \leq 0 \quad (I')$$

$$U(B,\bar{B}) - U(\bar{B},\bar{A}) \leq 0 \quad (II')$$

Which implies $U(\bar{A},\bar{B}) = U(B,\bar{B})$ — (IV)

This inturn, implies

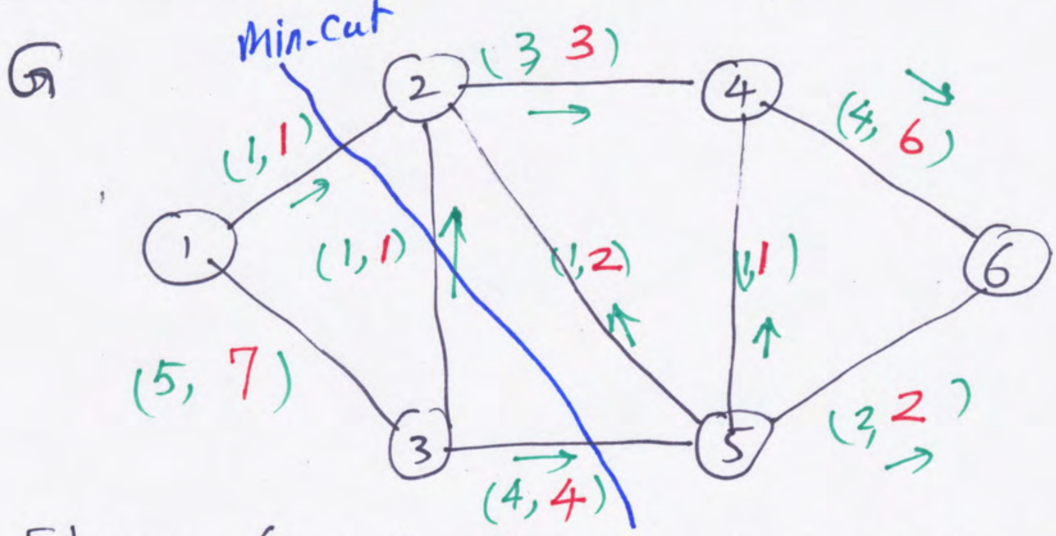
$$U(Y,\bar{Y}) = U(A \cup B \cup \bar{B}, \bar{A})$$

$\therefore \exists$ a (p,q) min cut that has $\bar{X}$ contained in one side and these nodes can be condensed without loss.

The above procedure is called "uncrossing" procedure and is used in other contexts later on.

Now we describe the main algorithm used in Analysis Problem. This is due to T.C. Hu + R.E. Gomory. [See also Mayeda, etc. referred to in Printed Lec. Notes.

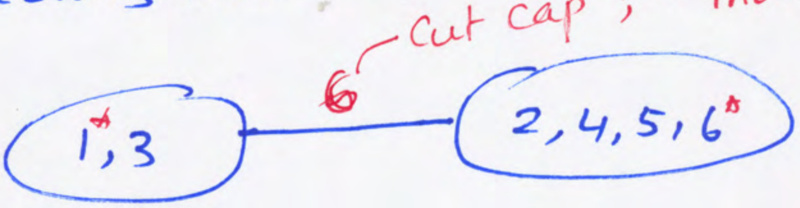
An Example: (Book by F F)



Step 1: (Arbitrarily) select an s-t pair; s = 1, t = 6. & Solve max-flow problem.

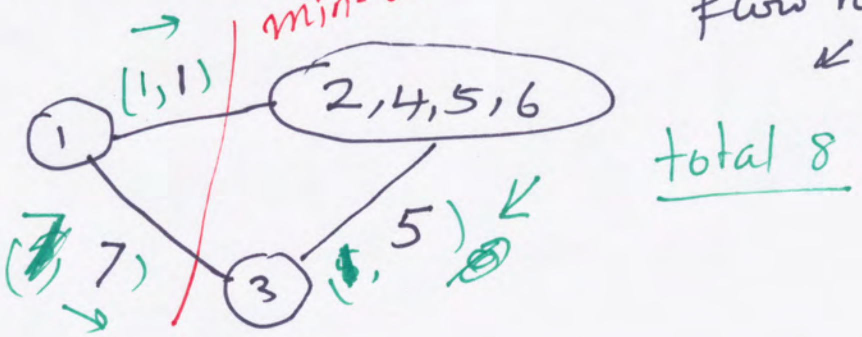
Flows in green

Results indicated by

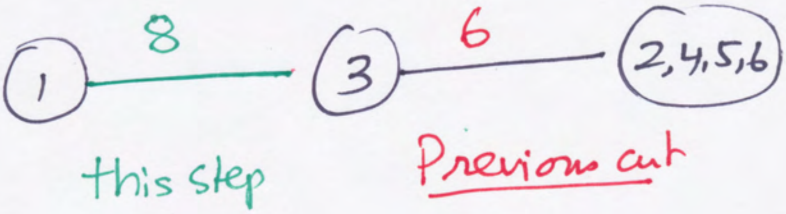


Step 2: Select (arbitrarily) an s-t completely in single condensed set in above picture. At this step, we select s = 1, t = 3. {2, 4, 5, 6} are condensed; condensed network:

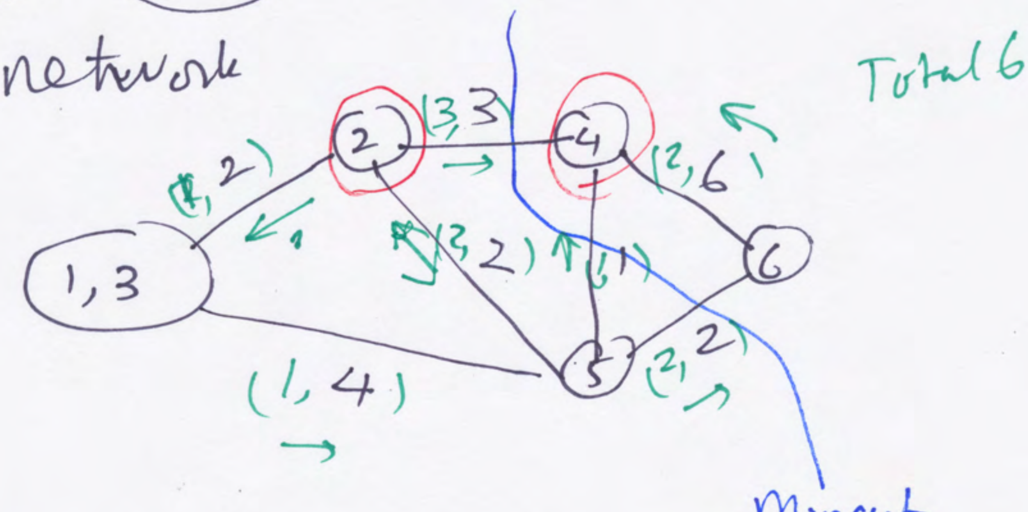
Solve 1-3 max flow here

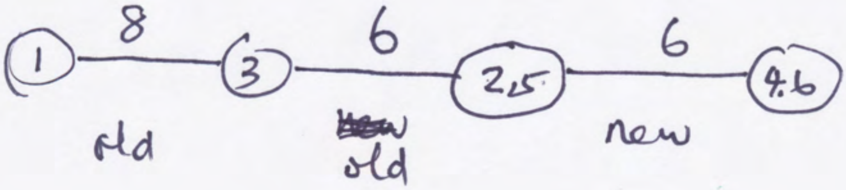


Since (2,4,5,6) is on side of (3), new "tree"



Step 3: $s = 2$, $t = 4$; If remove in the above (2,4,5,6) ; there may be many pieces — each is condensed. In this case we have only one piece (1) — (3) So (1,3) is condensed. Condensed network



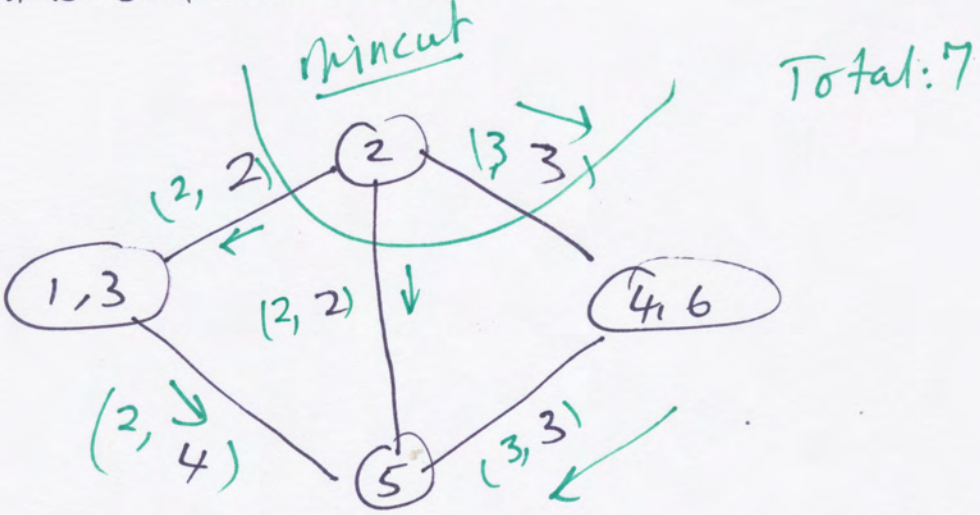


Step 4: $s = 2$, $t = 5$. Removing (2,5) from previous structure, we get two pieces

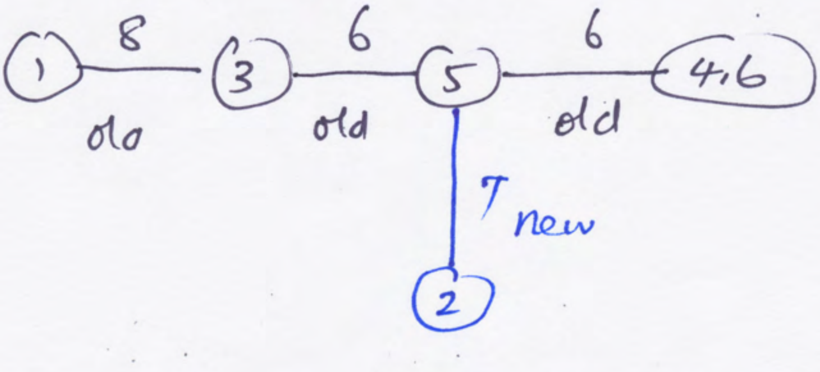
$\{1,3\}$ and $\{4,6\}$

So now (1,3) is condensed as is (4,6)

Condensed network



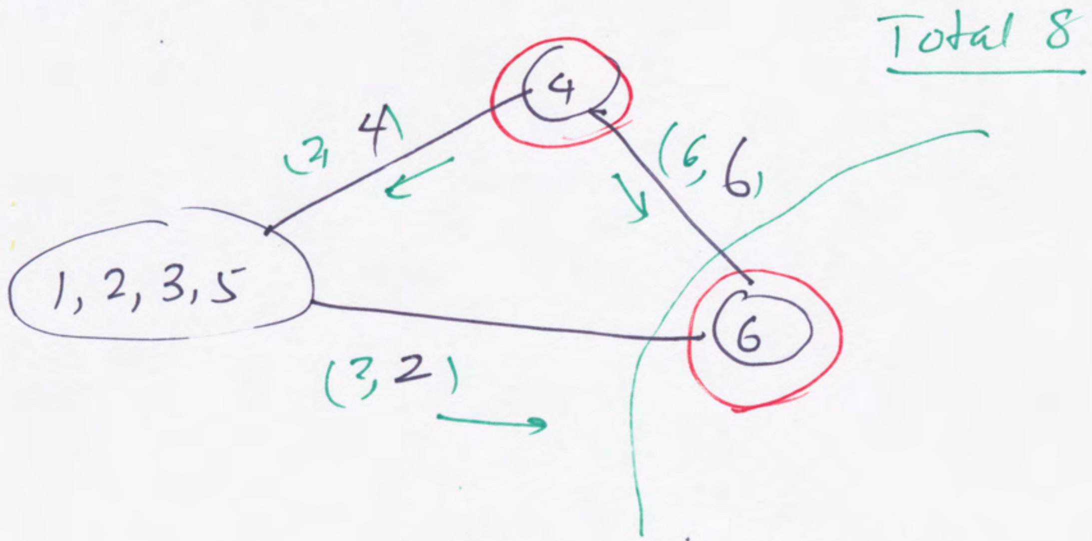
We split (2,5) so that (2) is by itself

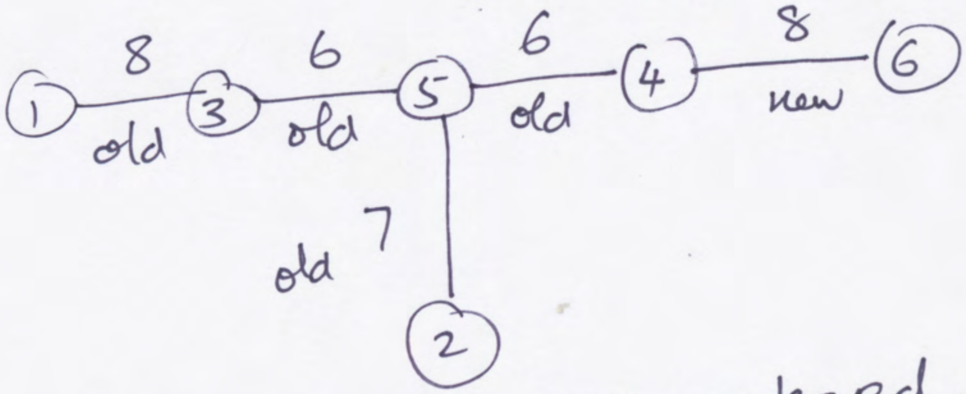


One more step to go with $s = 4$, $t = 6$

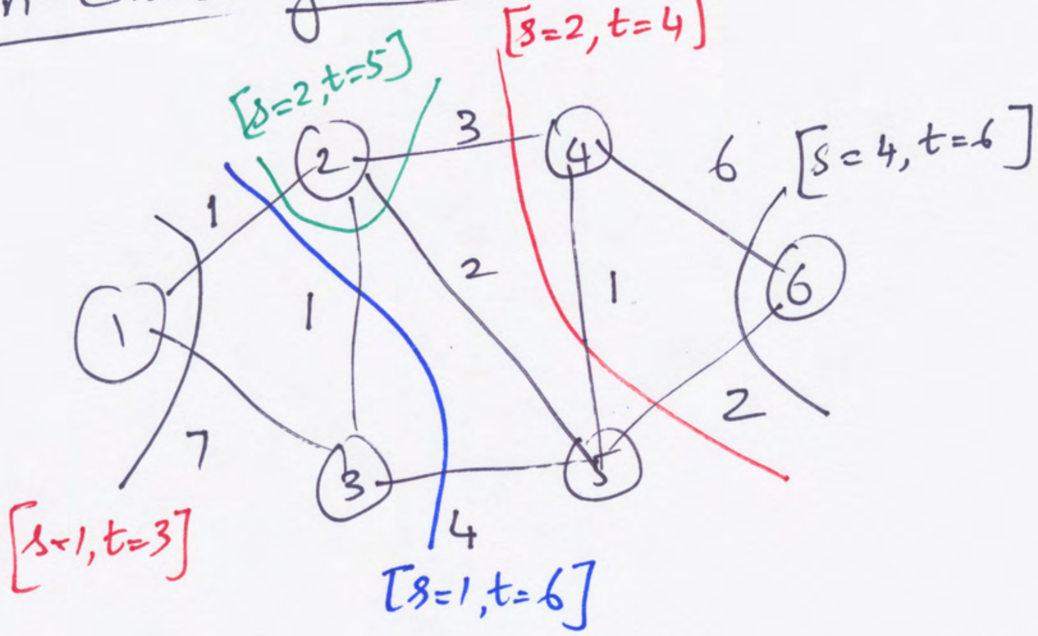
Here all of the rest is condensed into one piece.

Condensed Network





Non-crossing cuts encountered.



Now we prove that this works correctly.