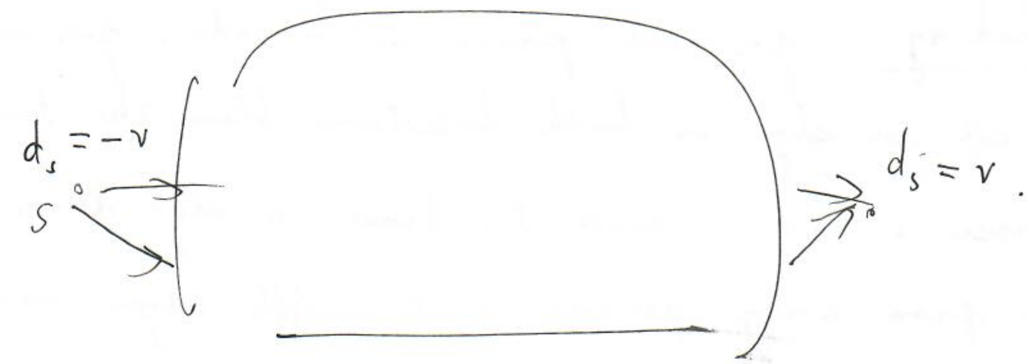
4. $S = (B \cap U) \cup (A \cap V)$

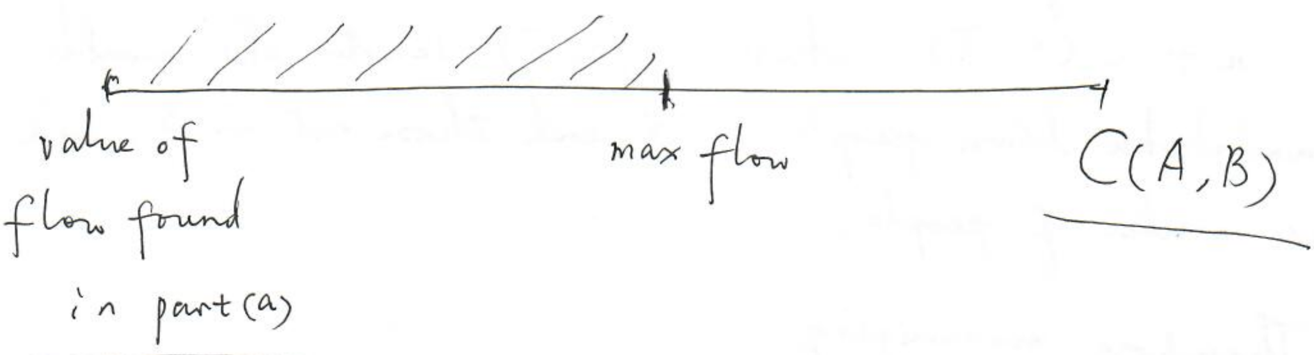
Pf: $\forall\, e = (u, v)$, $u \in U$, $v \in V$,

if e not covered by S, then $u \in A$, $v \in B$.

$\Rightarrow$ e is in the cut $\Rightarrow c(A, B) = \infty$. $\Rightarrow\Leftarrow$

5.





5 (a): Draw an edge from t to s, set demand everywhere to be 0, run circulation with the given upper and lower bounds.

Circulation exists $\Leftrightarrow$ flow exists in the original network.

The flow on the edge from t to s is the value of the flow, call it v.

5 (b) Let C be the capacity of an arbitrary cut.

Do binary search btwn v and C and find the max value s.t. a flow with that value exists.

Procedure to test if a flow with value v' exists:

in the original network, set $d_s = -v'$, $d_t = v'$, and $d_u = 0$ $\forall u \notin \{s, t\}$, and run circulation with upper & lower bounds.

---

2 (d) <u>Handshaking</u>.

Interpretation 1: If S can be $\emptyset$, then obviously ~~$\emptyset$ is the~~ we should take S to be $\emptyset$.

Interpretation 2: If S cannot be $\emptyset$.

If we set up the natural undirected graph representing the handshakes: one node per person, one edge per shake btwn the two corresponding nodes, then the question becomes: how many edges need to be removed to disconnect the graph?

One may check, for every pair of nodes u, v, whether there are $> k$ edge-disjoint paths between u & v. If this is true for all u & v, the S asked for in the problem doesn't exist.

This uses the max flow alg. $O(n^2)$ times.

One may save time by fixing u and enumerating v, asking whether there are $> k$ edge-disjoint paths btwn u & v.

2 (d) Handshaking.

Interpretation 1: If S can be ϕ, then obviously ~~ϕ is the~~ we should take S to be ϕ.

Interpretation 2: If S cannot be ϕ.

If we set up the ~~[illegible]~~ natural undirected graph representing the handshakes: one node per person, one edge per shake btwn the two corresponding nodes, then the question becomes: how many edges need to be removed to disconnect the graph?

One may check, for every pair of nodes u, v, whether there ~~is~~ are $> k$ edge-disjoint paths between u & v. If this is true for all u & v, the S asked for in the problem doesn't exist. This uses the max flow alg. $O(n^2)$ times.

One may save time by fixing u and enumerating v, asking whether there are $> k$ edge-disjoint paths btwn u & v. This uses max-flow $O(n)$ times. — Convince yourself that this is also correct.

There are in fact faster algorithms for this problem.