I have read and agree to the collaboration policy. Lynne Diep.

Lynne Diep
Homework Heavy Grading
Collaborators: none

Homework 4-1
Due: June 5, 2017

a) Algorithm –
Define vertices v_{in} and v_{out} to replace v
For all v except s and t
    Each edge that goes into v, now goes into v_{in}
    Each edge that exits out of v, not exits v_{out}
    Set edge weight v_{in} and $v_{out} = C_v$
Set each edge that does not equal v_{in} and v_{out} to infinity (∞)*
    *Because we do not want the flow to be restricted, we want original v's to be bottleneck in Ford-Fulkerson. We also want v's to be initially reachable from s.
Run Ford-Fulkerson on this new graph
Return max-flow

Runtime – O(mnC)
Space – O(m+n)

Correctness of algorithm in part c

b) An analogue s-t cut of the node-capacitated graph is a partition (A,B) of V. Where s ∈ A and t ∈ B.
Where A is the set of nodes reachable from s, and B is the set of nodes that are not reachable from s.
Capacity is Σ C_v, where v_{edge} on to A. We define v_{edge} to be edge through vertex v.

c) Proof (same time):
i) ∃ a cut (A,B) such that v(f) = cap(A.B)
ii) Flow f is a max flow
iii) There is no augmenting relative to f
(i) → (ii)
(ii) → (iii)
(iii) → (i)

(i) → (ii)
$v_f = \Sigma_{fout}(v) - \Sigma_{f in}(v)$
$\leq \Sigma_{fout}(v)$
$\leq \Sigma$ C_v, where v_{edge} out of A
= cap(A,B)

(ii) → (iii)
Let f be a flow. If there exists an augmenting path, then we can improve f by sending flow along the path.
(iii) → (i)
f = flow, no augmenting paths
A = set of vertices reachable from s in residual graph
By definition of A, s ∈ A
By definition of f, s ∉ A
v(f) = $\Sigma_{f_{out}}(v)$ - $\Sigma_{f_{in}}(v)$
≤ Σ C_v, where v_{edge} out of A
= cap(A,B)

Hence, why the max-flow min-cut theorem holds for the analogue, and proves correctness of the algorithm for part a.