(Sol) HW07 Max-Flow Min-Cut

1. [12 points] Job Assignment Problem

We have n computing jobs $(x_1, x_2, \cdots, x_n)$ and m machines $(y_1, y_2, \cdots, y_m)$. Let S_i be the set of machines on which job x_i can run (some jobs can't run on some machines due to compatibility issues). Each machine can run (up to) two different jobs (note that the same machine can't run the same job more than once). The objective is to run as many jobs as possible.

Formally, an assignment is a mapping from jobs to machines such that each machine gets at most two jobs (some jobs may not be assigned to any machines). The size of an assignment is the number of jobs that are assigned to machines (note that we have no reason to assign one job to multiple machines, so we will assume that we will assign each job either to a machine or to none). The size of an assignment is the number of jobs assigned to machines.

Example:
Suppose $n = 4, m = 2$ with $S_1 = \{y_1, y_2\}$, $S_2 = \{y_1\}$, $S_3 = \{y_2\}$,and $S_4 = \{y_2\}$. We can assign x_1, x_2 to y_1 and x_3, x_4 to y_2, which is optimal (as we can run four jobs).

Here is a slightly different example with $n = 4, m = 2$: $S_1 = \{y_2\}$, $S_2 = \{y_1\}$, $S_3 = \{y_2\}$, and $S_4 = \{y_2\}$. In this case, we can run x_2 on y_1, but we'll have to choose two from $\{x_1, x_3, x_4\}$ to run on y_2; so, the optimal solution (assignment) is of size 3.

We can reduce this problem to the Max-Flow problem.

(a) [8 points] Describe your reduction. This means, you need to clearly describe the following:

- [2 points] Construction - How you would construct a flow network G from an arbitrary input instance $(n, m, \{S_i\})$.
  - Create $\{s, t\}$.
    For each job x_i, create a corresponding node x_i.
    For each machine y_j, create a corresponding node y_j.
    Note that G will have $2 + n + m$ nodes.

- Create an edge from s to x_i with unit capacity (capacity 1).
  Create an edge from y_j to t with capacity 2.
  Create an edge from x_i to y_j with unit capacity if and only if x_i and y_j are compatible (according to S_i).
  Note that G will have $n + m + \sum_i |S_i|$ edges (the summation is at most $n * m$).

- [4 points] Conversion - How you would convert a max-flow in G into an assignment.
  - Given a max-flow f^* in G, we can convert it into a job assignment as follows. Assume that f^* is an integral max-flow (i.e., $f^*(e)$ is an integer for all e).
    - If $f^*(x_i, y_j) = 1$ then we assign job x_i to machine y_j.

    Note: During the conversion (post-processing) step, you must explain how you can convert a max-flow (f^*) into an assignment because the original problem (job assignment) demands an assignment, not a max-flow. If this is not clearly explained in your answer, a small deduction applies.
- [2 points] Relationship - What is the relationship between the size of an assignment and the value of a max-flow in your reduction? (e.g., are they equal? is one twice the other? etc.)
  - $val(f^*)$ is equal to the maximum possible number of jobs assigned to machines.

    Every augmenting path $s \rightarrow$ job $\rightarrow$ machine $\rightarrow t$ has a bottleneck value of 1, as every edge from a job to a machine has a capacity 1. Every augmenting path is one more job assigned to a machine, so the max flow value is the max possible number of jobs assigned to machines.

(b) [6 points] Analyze the running time of your reduction. This involves two steps in addition to the running time of finding a max flow.

- [3 points] Running time of the Construction step (express your answer in terms of $n, m, \{S_i\}$ using Big-O).

  As described above, creating nodes (V) takes $O(2 + n + m)$ time and creating edges takes $O(n + m + \sum_i |S_i|)$ time.

Overall, this step takes $O(n + m + \sum_i |S_i|)$ time (or, $O(nm)$ is a loose, but correct upper-bound, too).

- [3 points] Running time of the Conversion step (express your answer in terms of $n, m, \{S_i\}$ using Big-O).

  Given f^*, we simply need to check if $f^*(x_i, y_i) = 1$ for each pair or not, and this can be implemented in $O(\sum_i |S_i|)$ time (or, $O(nm)$ is fine, too).
  (Note: One common mistake was NOT to describe HOW you can convert a max-flow into an assignment.)