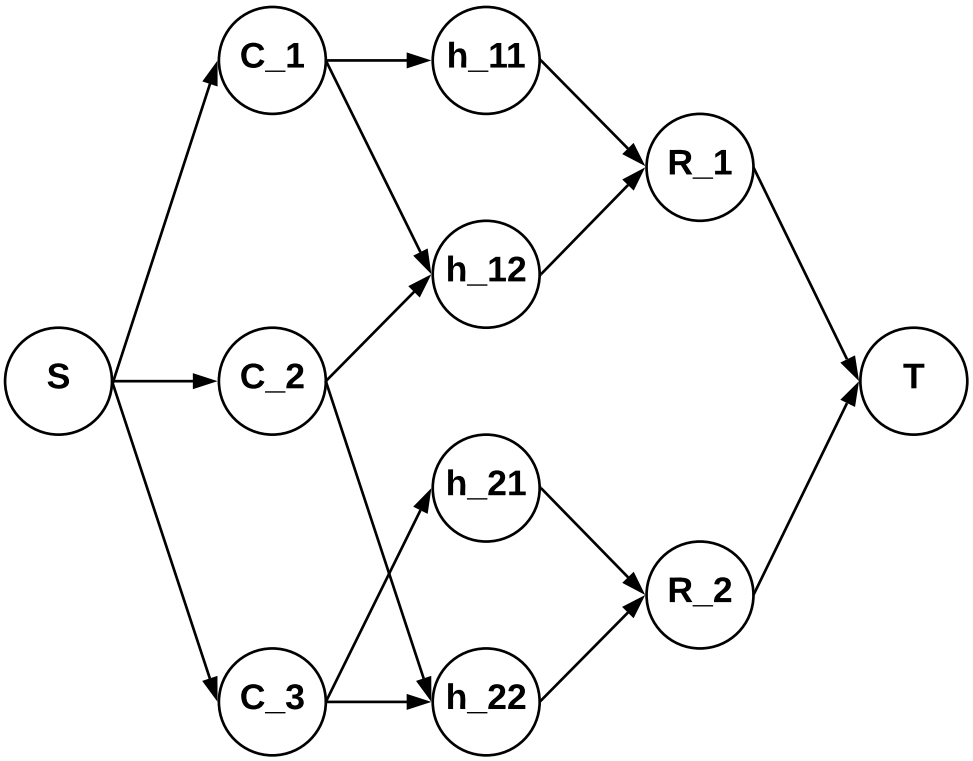
Similar to Problem 2 in the last homework, now we add another column of vertices between candidates and recruiters, which is the list of hours that the recruiter would able to interview. Note that we will list the hours (maybe repeatedly) for all recruiters, not just all the possible hours. For example if we only have 2 recruiters: R_1 can interview on hour h_1, h_2, and R_2 can interview on hour h_1, h_2, h_3, the list of hours in the graph would be $h_{11}, h_{12}, h_{21}, h_{22}, h_{23}$ (first two are for R_1 and the rest for R_2) rather than just h_1, h_2, h_3. The rest of the vertices are the same as before: a source, a list of candidates, a list of recruiters and a sink. We first connect the source to all candidates with capacity 1 for all. Then from the candidates to time, we connect a candidate to a time if and only if: 1. the candidate is available to be interviewed that time, and 2. the time corresponds to a recruiter that is qualified to interview this candidate (edge capacities are 1). Then from time to recruiters, we connect time vertices to their corresponding recruiters as stated before (edge capacities are 1). Finally we connect all recruiters to the sink with edge capacity q_j which are the max numbers of interviews each recruiter can do. The graph of an example of specific problem is shown down below:



An example of specific interview scheduling task: We have 3 candidates C_1, C_2, C_3 and 2 recruiters R_1, R_2. C_1 can be interviewed by R_1; C_2 can be interviewed by R_1, and R_2; C_3 can be interviewed by R_2. R_1 can do max 2 interviews and R_2 can do max 1 interview. The available time list for interviews/interviewees is: C_1: h_1, h_2; C_2: h_2, C_3: h_1, h_2; R_1: h_1, h_2; R_2: h_1, h_2. (Note that in the graph, h_ij means h_j corresponding to recruiter R_i)

Figure 1: The graph of an example of specific interview scheduling task

After the graph has been built, we simply run Ford-Fulkerson Algorithm in this graph to find the max flow. Then based on this max flow, and starting at S, we run BFS to T to find the matching, which is an interview schedule if one exists.

Proof of Correctness:

The only difference between this problem and the original bipartite problem is the time slots. We made the time slots separated by each recruiter and combine them together to a list in the end is because we would like each candidate can be constrained by both time slot and the recruiter. Only if those two conditions are satisfied, the candidate can then connect to the time slot h_{ij} (Time slot h_j corresponding to recruiter R_i). By doing this, we also made sure that each single candidate or recruiter is assigned to a single time slot (but a single time slot can be assigned to different candidates/recruiters). This is because each candidate node is the tail is at most one edge (proved in the textbook), and the capacity is one. So the flow come out of each candidate can only be one (i.e., candidate can only match one time slot in the final result). Since in the algorithm we have stated that candidates can only connect to the time slots that satisfy the two constraints (availability and qualification), we can make sure the match is correct from the candidate perspective. Also for recruiters we have list their available time slots and candidates can only connect to them via time slots satisfying the two constraints, and the capacities of the edges to sink are q_j which align with the max numbers of interview. Hence the match is also correct from the recruiter perspective. As we have discussed in the class and according to the text book, once the relationship is establish, the Ford-Fulkerson algorithm can be used to find a maximum matching in a bipartite graph (we now consider C_i and h_{ij} as a bipartite pair since h_{ij} will only and must connect to R_i in the matching result).

Time Complexity Analysis::

From the textbook we know Ford Fulkerson Algorithm get the result in $O(mC)$ time. In this problem, $m = (n + mT + Tk + k)$, $C = n$. We know $m > n$ since each candidate are qualified to be interviewed by at least one recruiter (i.e., $L_i \geq 1$). Therefore $m > n > k$ and mT is the largest term in $(n + mT + Tk + k)$. Hence the time complexity is $O(mTn)$.