CMPS 102
Introduction to Analysis of Algorithms
Winter 2017
Homework 4
Network Flow

Computer Science
UC Santa Cruz

March 14, 2017

Name: Froylan Valencia
Email: frvalenc@ucsc.edu

Solution to Problem 1

PROBLEM: Consider a set S of n computer scientist roommates who would like to determine "who will do the dishes after dinner" for the next m nights. Not all roommates have dinner at home every night, so let Si denote the subset of roommates who will have dinner at home on the i-th night and let $w_i \leq |S_i|$ be the number of people needed to do the dishes on the i-th night (some nights multiple people are needed).

For each roommate $j \in S$ and night $1 \leq i \leq m$, let $Q_i(j) = 0$ if j does not eat at home on the i-th night, otherwise let $Q_i(j) = w_i \div |S_i|$, i.e., if j eats at home on the i-th night. Now, for each roommate j, let

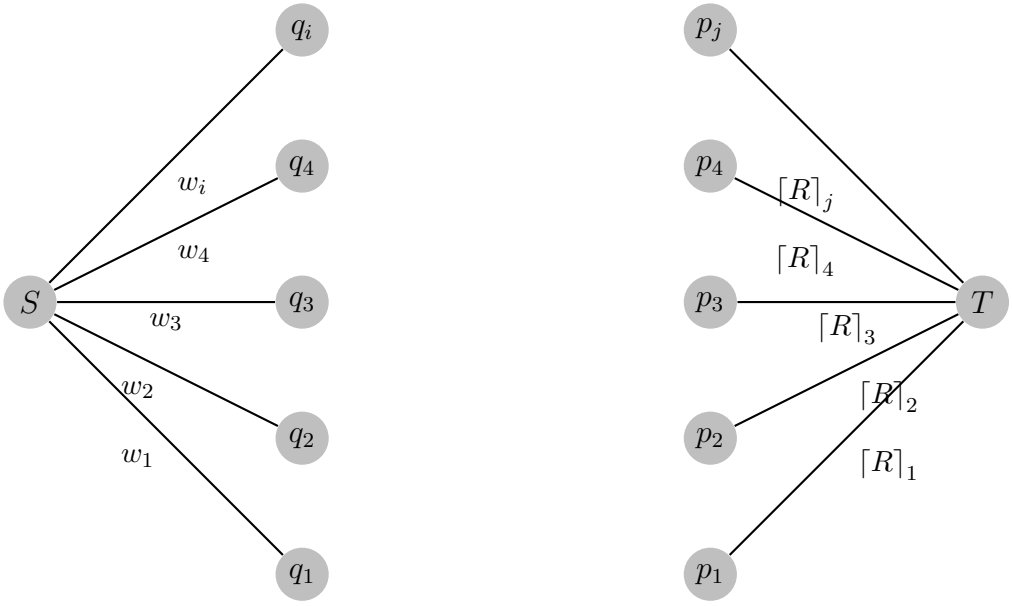
$$R_j = \sum_{i=1}^{m} Q_i(j)$$

Clearly, we can?t hope that everyone does dishes exactly Rj nights since, in general, Rj won?t even be an integer. Nevertheless, the computer scientists claim:

There exists a dish-washing plan under which

each roommate $j \in S$ does the dishes at most $\lceil R_j \rceil$ times.

Prove the claim. That is, be a computer scientist.



widthheightfigure

SOLUTION:

First we construct a graph as follows. We build the following network. There is a node q_i for each night i, a node p_j for each person j, and an edge (q_i,p_j) of capacity 1 if the person j is attending on a particular night i. We then connect a super-source s to each of the night nodes by an edge of capacity w_i the number of people needed to clean on a particular night and we connect each person node to super-sink t by an edge of capacity $\lceil R \rceil_j$

Finding that computing a fair cleaning schedule is equivalent to computing the maximum flow problem. The only thing we need to do is to prove that the value of the max flow is. First of all, it is obvious that for any given persons flow f, $|f| \leq |S_j|$ Thus if we are able to find a flow We claim that there is a feasible way to schedule dishes if and only if there is an s - t flow of value $\lceil R \rceil j$

We know that no person may be assigned more than $\lceil R \rceil_j$, since the person to sink edges have a capacity enforced, and every assigned person must be attending on that day, since we only add edge if a person present. Therefore, we have shown that if a flow exists, we can transform it into a fair schedule.

For computing it, we build graph and apply the Ford-Fulkerson algorithm. Runtime: Running Ford Fulkerson $O(|E| * F)$.

Solution to Problem 2

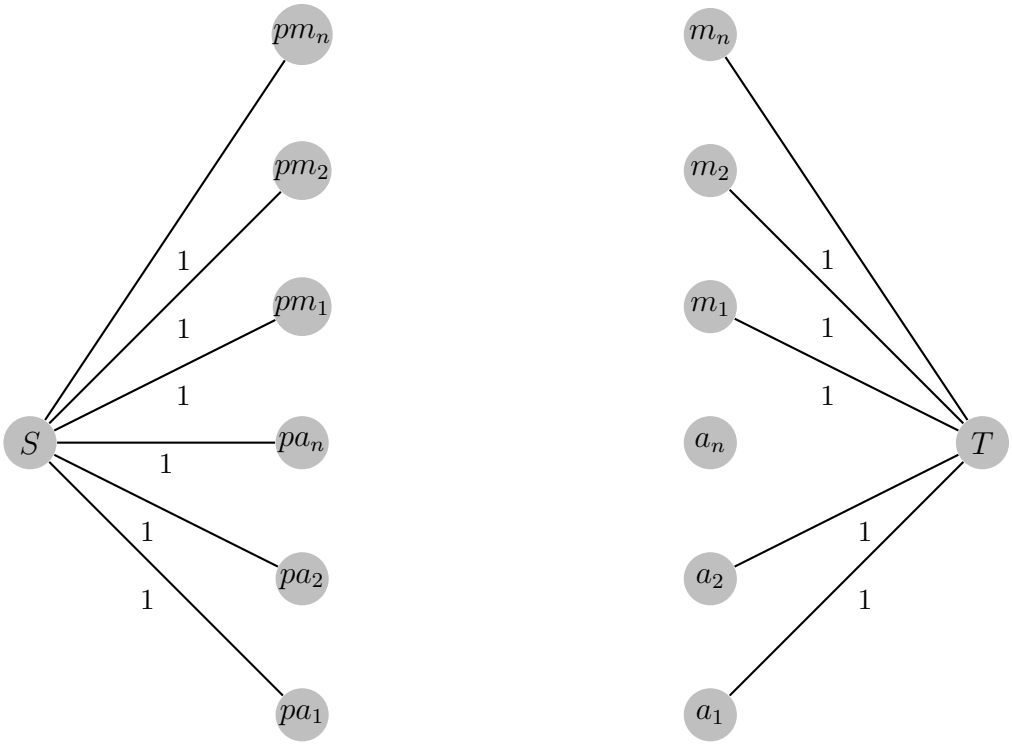
You are planning a dinner party for n friends who, naturally, have all sorts of dietary restrictions and allergies. Your plan is to cook some appetizer dishes and some main dishes and portion them so that you end up with n appetizer portions and n main portions. You email everyone1 , and for each person $i \in \{1, ...n\}$ you get back their list Ai of OK-to-eat appetizer dishes and their list Mi of OK-to-eat main dishes.

Design an efficient algorithm that takes as input the number of portions of each dish and the lists Ai , Mi , designs a max flow instance, has it solved, and uses the (value of the) maximum flow to decide whether or not it is possible to give each friend an appetizer and a main portion that is OK for them to eat.

SOLUTION:

The goal here is to come up with a solution that allows for each person to eat an appetizer and a main entree, we have n appetizer and n main dish portions

The graph contains two edges leaving the source for each person, of these two vertices one corresponds with a main dish serving and one will correspond with an appetizer serving that are connected to the sink. Each edge will have a capacity of 1 meaning due to the integrality theorem we will either have 0 or 1 through each edge. We know that in order for the problem to be satisfied we mush have n serving of main dishes and n serving of appetizers so we must have a max flow of 2n.



widthheightfigure

claim1: there is a satisfying assignment of entrees and appetizers

Proof. If there is a satisfying assignment of entrees/apps there is an integral flow of size 2n. We construct an assignment by choosing the saturated edges between person vertices and entree/app vertices as our assignments. Each persons will be assigned only the dishes that it prefers, since edges represent an interest in a dish. We are pushing a flow of n through all vertices going to entrees and a flow of n through vertices going to appetizers, so we have a total of 2n flow, every edge out of the source must be saturated, so every person must be assigned one entree and one app. Since each edge leading to T has a capacity of 1 there can only be one edge person pushing flow i.e. receiving the serving. Therefore, our assignment is a satisfying assignment.

claim 2: A satisfying assignment of entree/apps implies there is a max flow of size 2n in G.

Proof. If there is a satisfying assignment, we can create a flow of size 2n in the following manner: Saturate every edge out of S. Then each person vertex must push out exactly one flow, which we do by saturating the edges that correspond to the entree and app assigned to this person, then saturate edges from entree snack to T. We know that this last step is possible because this was built from a legal assignment, which means there are not two persons flowing into the same entree/app vertex. So given that a satisfying assignment exists, we have shown how to construct a flow of size 2n in G. Therefore, if there exist a pairing of apps and entrees there also exists a max flow of size 2n.

Runtime: we must build the graph and then run Ford- Fulkerson. Building the graph is $O(n(n+n))$, since there are $O(n+n+n)$ vertices, and $O(n(n+n))$ edges between person and entree/app vertices. Running Ford-Fulkerson is $O(|E| * F)$. We can put an upper bound of 2n on the flow and we get a runtime of $O(n(n+n)n) = O(n^3)$

Solution to Problem 3

You are an English teacher assigning presentations to students. Each of your n students will receive a topic and 31 fancy words that they must use in discussing the topic. You?ve announced the set of topics T and the set of fancy words F , and have received from each student i ? $1, ..., n$ a set $Ti \subseteq T$ of topics the student is interested in, and a set $Fi \subseteq F$ of fancy words that the student would like to use. You want to see if it is possible to assign to each student a topic and exactly 31 words such that:

Each topic is assigned to at most 2 students.

Each fancy word wi is assigned to at most ti students.

Describe a method that takes as input the sets Ti and Fi and the numbers t1 , . . . tk and returns either ?No?, or an assignment of topics and words that meets the requirements.

Solution to Problem 4

Let $G = (V, E)$ be a directed graph with a source $s \in V$, a sink $t \in V$, and where every edge $e \in E$ has capacity exactly 1. Let k be the value of the maximum flow in G.

Question: Given an arbitrary integer $1 \leq q \leq k$, can you always remove q edges from G so that in the resulting graph G' the value of the maximum flow is $k - q$?

SOLUTION:

Yes,

Let the edges E2 of size q be the edges removed and form the subgraph $G' = (V, E - E2)$, which has an s-t flow of value at least k-q. indeed consider any cut (S,T) of G'. There are at least k edges out of S in G and at most q edges have been deleted so there are at least $k - q$ edges out of S in G' Thus in the minimum cut of G' there are at least $k - q$ edges out of S in G' thus the minimum cut in G' has a flow of at least $k - q$