Justin Wong (jujwong@ucsc.edu)
Homework #4 Network Flow
CS 102
March 12th, 2015

1. Junior Jamborees

Given n party favors and n snacks, there are n children with preferences for party favors F_i and acceptable snacks S_i. Is it possible to assign everyone one favor and snack of their choice?

Starting with a source s connect this to all vertices in the list S_i. Connect all vertices in list S_i to a sink t. Then give each edge capacity of 1 to edges from source to nodes in S_i and nodes in F_i to sink. The output of the edges between F_i and t will be 1, while the sum of these edges will be n, so no two edges can flow into the same node coming from S_i. Similarly from s to S_i the edges have capacity 1 and the sum of these edges will be n, so no two edges can flow out of the same node in S_i to the same node in F_i. Use the Ford-Fulkerson algorithm find a maximum flow from s to t. This will be a maximum matching because any matching will give you a flow of the same value, where max flow should be equal to n.

2. Camp ABC

Given a list of *n* students' S_i working spots and a list of *c* cabins with their locations design an algorithm to decide if it is possible to assign kids to cabins so that no student is too far from their working spot and no cabin has more than $\lceil n/c \rceil$ students assigned to it. There is a function that takes in a spot and a cabin and returns whether or not it is too far.

Starting with a source *s* connected to all the nodes in the list S_i, the edge capacities will be 1 with the number of edges being equal to the number of students *n*. Connect all the nodes in the list *c* to a sink *t,* the edge capacity will be $\lceil n/c \rceil$. Use the function that checks if spots and cabins are valid and connect nodes from S_i to the nodes in *c* according to the function each with edge capacity of 1 because each students is assigned to one cabin. The edge capacity of $\lceil n/c \rceil$ coming out of the cabins is from multiple students being assigned to a singular cabin, the sum of the edges leaving the cabins will be *n* as $\lceil n/c \rceil$ multiplied by the number of cabins is simply *n*. This shows that the conservation of flow, as all the flow from the source will reach the end.

Using the Ford Fulkerson algorithm to find max flow will give us *n* if there is a valid assignment of students to cabins. This algorithm will determine if there is a valid flow, which in turns tells us if the assignment is fair.

3. Haute Couture

m is a list of designers and n is a list of articles of clothing, C_i is the set of articles they can work on and what design school they went to, and each designer can only be assigned up to 3 articles. Is it possible to assign *k* designers to each item with no designer being given more than 3 articles and no committee contains only designers from the same school?

Starting with a source *s* connect to each of the *m* designers nodes each with edge capacity of 3 because of the limitation of three designers per clothing item. Take each *n* clothes node and connect them to a sink *t*. Without anything else this represents the designers to clothes without the restrictions of the different schools of design, if we connected the designers to clothes here. There would be no way of differentiating what schools of design were intended, indicated by the lack of the use of the set C_i. Using the set C_i construct a gadget of the three different schools and connect the *m* designer nodes each to a gadget representing the 3 schools and preventing any articles from being worked on by exclusively the same school. Connect the designer nodes to gadgets with the edge capacity of 1, thus denying the designer the ability to work on the same project and providing the list of schools they went to. From the gadgets connect them to the articles of clothing each with the edge capacity of k-1 as the clothing can have multiple incoming edges but the flow from each individual designer will be 1. Connecting *n,* the articles of clothing, to the sink, *t,* will have edge capacities of k. Use Ford Fulkerson algorithm to solve find the max flow.

If there is a possible assignment of clothing worked on by at least two different schools of designers and using *k* designers then the max flow would be equal to the number of clothes worked on multiplied by the number of designers, hence Max Flow =*nk*.

4. 7.22 Matrix Madness

M is an *n* x *n* matrix with each entry either a 0 or 1. M_{ij} is the row *i* and entry *j*. The diagonal is m_{ii}. It is possible to swap rows and columns. It is rearrangeable if it is possible to swap some rows and columns and end up in a state such that all diagonals of *M* are 1's.

(a) Here is a matrix with at least one 1 in each row and each column that is not rearrangeable.

$$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix} \qquad \text{and} \qquad \begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{matrix}$$

(b) To determine if a matrix *M* of size $n\ x\ n$ with 0-1 entries is rearrangeable, all the diagonals are 1's. The matrix *M* must first have at least n entries equal to 1's. Then we start by converting it to a network flow, using an adjacency matrix. Create a list of n nodes I from the rows of the matrix and connect these to a source s each with an edge capacity of 1. Similarly with the values of the columns of the matrix, create a list of n nodes J but connect these to a sink t with edge capacities of 1. The nodes in I will have an edge connecting it to a node in J if for some value i both I_i and J_i are both 1, this edge capacity is 1 as we don't want to recount any values in *M* twice. If we have a max flow value of n, then we know that the matrix is rearrangeable.

A max flow of n means that the matrix is rearrangeable of size $n\ x\ n$. S is the set where both I_i and J_i have the value of 1. The values of flow going into I all have 1 so as to count a single entry for that row. The values leaving J all have 1 meaning that at the position of m_{ij} there is a 1. This ensures that each row value with 1 and column value of 1 does not share its value with another in bipartite matching. Using Ford Fulkerson if the max flow value of the graph is n then the matrix is rearrangeable.

5. 7.27 Carpool ***not answered**

$S = \{p_1,\ldots,p_k\}$ is the people. Total driving obligation of p_j over a set of days is the expected number of times that p_j would have driven chosen uniformly at random from those going to work. The carpool lasts d days and on the i^{th} day a subset of S_i from S of people who go to work. Total driving obligation Δ_j for p_j is $\Delta j = \sum i: p \in S_i \frac{1}{|Si|}$.