IE 512 Graphs, Networks, and Algorithms

# Final

This is a 24-hours take home final exam. Read *all* the questions before starting to write down a solution for one of the problems. There are 6 problems, so try to be very efficient in solving and writing the solutions. Contact `swoh@illinois.edu` if you have any questions.

**Problem 1**

($a$) **(3pt)** In this problem, we prove a sufficient condition for uniqueness of a Minimum Spanning Tree. Show that a graph has a unique minimum spanning tree if, for every cycle in the graph, the edge with the largest weight in the cycle is unique.

($b$) **(3pt)** Show that the converse is not true by giving a counter-example.

**Problem 2 (The dual of maximum flow.)** In the maximum flow problem, we are given a directed graph $G = (V, E)$ with a source node $s$ and a sink node $t$. Each edge $(i, j) \in E$ is associated with a capacity $c_{ij}$. A flow consists of a vector valued variable $f = \{f_{ij}\}_{(i,j)\in E}$, satisfying the capacity condition ($0 \le f_{ij} \le c_{ij}$ for all edges) and conservation condition (total incoming flow is equal to the total outgoing flow at each node that is not $s$ or $t$). The value of the flow is the total quantity leaving the source (or equivalently arriving the sink):

$$size(f) = \sum_{i:(s,i)\in E} f_{si}$$

This can be formulated as a linear program:

$$\begin{array}{ll} \text{maximize} & size(f) \\ \text{subject to} & 0 \le f_{ij} \le c_{ij}, \ \ \forall (i,j) \in E \\ & \displaystyle\sum_{k:(k,i)\in E} f_{ki} = \sum_{k:(i,k)\in E} f_{ik}, \ \ \forall i \notin \{s,t\} \end{array}$$

Consider a general directed network $G = (V, E)$, with edge capacities $c_{ij}$'s.

($a$) **(2pt)** Write down the dual of the general flow LP above. Use a variable $y_{ij}$ for each directed edge $(i, j)$, and $x_i$ for each node $i \notin \{s, t\}$.

($b$) **(3pt)** Show that any solution to the general dual LP must satisfy the following property: for any directed path from $s$ to $t$ in the network, the sum of the $y_{ij}$ values along the path must be at least 1.

($c$) **(3pt)** What are the intuitive meaning of the dual variables? Show that any $s - t$ cut in the network can be translated into a dual feasible solution whose cost is exactly the capacity of that cut. More precisely, given a $s - t$ cut, construct $y_{ij}$'s and $x_i$'s from the cut, such that the dual variables satisfy all the constraints in the dual LP.

**Problem 3 (7pt)** Recall the homework problem 6.1 on iterative power control with interference. We add a modification of receiver noise in this problem. Define the signal to interference plus noise ratio (SINR) for a transmitter $i$ as

$$\frac{G_{ii}P_i}{N_i \sum_k P_k \; + \; \sum_{k \neq i} G_{ik} P_k} \, .$$

We want to maximize the minimum SINR:

$$\begin{array}{ll} \text{maximize} & \min_i \dfrac{G_{ii}P_i}{N_i \sum_k P_k \; + \; \sum_{k \neq i} G_{ik} P_k} \\ \text{subject to} & P_i > 0 \end{array}$$

For the following parts, we solve an instance of this problem for a matrix $G = [G_{ij}]$ describing the signal interference, and a vector $N = [N_i]$ describing the noise, provided in `interference.m` on course website.

Design and implement on MATLAB an iterative algorithm (similar to the one from Homework 6.1 ($b$)) that finds the optimal power assignment $P_i$'s that maximizes the minimum SINR for given $G$ and $N$. Also, find the optimal value (of the objective function) of the optimization problem above.

**Problem 4** We have an $n \times n$ grid like below. A subset of the cells are 'point cells' (marked as black).

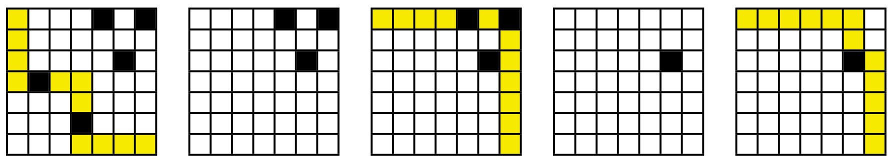


**Figure 1:** Greedily covering the point cells with three monotone paths.

A monotone path in the grid starts at the top-left cell and at each cell can only move to either right or down by one step (if there exists a cell to the right or below). Eventually, the path ends at the bottom-right cell. The goal is to cover as many 'point cells' as possible, with a single monotone path.

($a$) **(4pt)** Describe an efficient algorithm for finding a monotone path that covers the maximum number of 'point cells'. [hint: formulate the problem as min-cost flow problem]

($b$) **(3pt)** Now consider the problem of covering all point cells using multiple monotone paths. The goal is to cover all point cells using as small number of mutiple paths as possible. A greedy heuristic for finding one solution is to iteratively apply the above algorithm. At each step, find the monotone path that covers maximum number of point cells that are not already covered, and repeat until all point cells are covered. Prove by counter example that this algorithm does not always give the optimal solution.

($c$) **(Optional)** Describe an efficient algorithm to compute the smallest set of monotone paths that covers every point cells.

**Problem 5** **(7pt)** There are $n$ students in a class. We want to choose a subset of $k$ students as a committee. There has to be $m_1$ number of freshmen, $m_2$ number of sophomores, $m_3$ number of juniors, and $m_4$ number of seniors in the committee. Each student is from one of $k = m_1 + m_2 + m_3 + m_4$ departments. Exactly one student from each department has to be chosen for the committee. We are given a list of students, their home departments, and their class(freshman, sophomore, junior, senior).

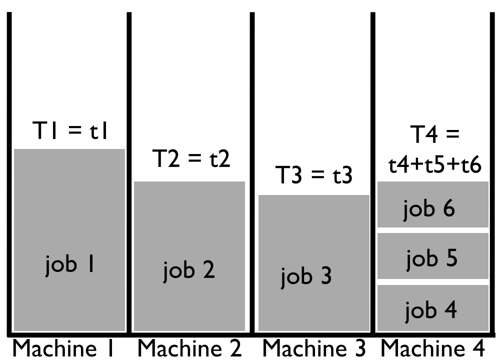
Describe an efficient algorithm to select who should be on the committee such that the constraints are satisfied.

**Problem 6** **(Competitive Analysis.)** We consider the following scheduling problem. There are $M$ machines and $J$ jobs. Each job $j$ takes $t_j$ time to finish independent of which machine is used. We want to come up with an assignment for each machine $i$. Let $A_i$ be the set of jobs assigned to machine $i$. Then, the completion time for this assignment and this machine is $T_i = \sum_{j \in A_i} t_j$. We want to find assignments such that $(i)$ each job is assigned to one machine, and $(ii)$ the maximum completion time is minimized.

$$\text{minimize}_{\{A_i\}_{i\in\{1,\ldots,M\}}} \max_{i\in\{1,\ldots,M\}} T_i$$

We let this minimum value be $OPT(L)$, where $L = \{t_j\}_{j\in\{1,\ldots,J\}}$ is the list of completion times.

Now consider a *greedy* approach to solve this problem. First sort the jobs in an decreasing order such that $t_1 \geq t_2 \geq \ldots \geq t_J$. Then, the greedy algorithm iteratively assigns at iteration $k$, the current job $k$ with completion time $t_k$ to the machine with the smallest load at current time. Once we assign a job to a machine, we never change the assignment. In this sense you can think of it as an online algorithm, working on a sorted and online input data.




Let $GA(L)$ be the maximum completion time $\max_{i\in\{1,\ldots,M\}} T_i$ for the greedy algorithm. We want to prove that for any input $L$,

$$\frac{GA(L)}{OPT(L)} \leq \frac{3}{2} \, .$$

Prove this claim step by step in the following.

$(a)$ **(1pt)** If $J \leq M$, prove that the greedy algorithm is optimal.

$$GA(L) = OPT(L)$$

$(b)$ **(1pt)** Now for any $J$, let $i$ be one of the machines that is assigned maximum load $GA(L)$ using the greedy algorithm. If $i$ has only one job, then prove that the greedy algorithm is optimal.

$$GA(L) = OPT(L)$$

$(c)$ **(1pt)** Again for any $J$, let $i$ be one of the machines that is assigned maximum load $GA(L)$ using the greedy algorithm. Let $j$ be the last job assigned to machine $i$. Prove that

$$GA(L) - t_j \leq \frac{1}{M} \sum_{i=1}^{M} T_i \, .$$

$(d)$ **(1pt)** If $J > M$, prove that, for any input $L$,

$$t_{M+1} \leq \frac{1}{2} OPT(L) .$$

$(e)$ **(1pt)** Prove that even the optimal algorithm requires the maximum completion time to be at least the average completion time. Precisely, prove

$$\frac{1}{M} \sum_{i=1}^{M} T_i \leq OPT(L)$$

$(f)$ **(1pt)** Using $(a), \cdots, (e)$, prove that

$$\frac{GA(L)}{OPT(L)} \leq \frac{3}{2} .$$

$(g)$ **(Optional)** Now, use similar ideas from above to prove that

$$\frac{GA(L)}{OPT(L)} \leq \frac{4}{3} .$$