COL 351

TUTORIAL SHEET 11

1. **[KT-Chapter7]** Give a polynomial time algorithm for the following minimization analogue of the max-flow problem. You are given a directed graph $G = (V, E)$, with a source s and sink t, and numbers (capacities) l_e for each edge $e \in E$. We define a flow f, and the value of a flow, as usual, requiring that all nodes except s and t satisfy flow conservation. However, the given numbers are lower bounds on edge flow, i.e., they require that $f_e \geq l_e$ for each edge e, and there is no upper bound on flow values on edges.

   (a) Give a polynomial time algorithm that finds a feasible flow of minimum possible value (Hint: Start with any flow from s to t which obeys the lower bounds, and try to send a maximum flow from t to s in a suitable modification of the graph G).

   (b) Prove an analogue of the max- flow min-cut theorem for this problem.

2. **[KT-Chapter7]** You have a collection of n software applications, $\{1, \ldots, n\}$, running on an old system; and now you would like to port some of these to the new system. If you move application i to the new system, you expect a net (monetary) benefit of $b_i \geq 0$. The different software applications interact with one another; if applications i and j have extensive interaction, then the you will incur an expense if you move one of i or j to the new system but not both – let's denote this expense by $x_{ij} \geq 0$. So if the situation were really this simple, you would just port all n applications, achieving a total benefit of $\sum_i b_i$. Unfortunately, there's a problem. Due to small but fundamental incompatibilities between the two systems, there's no way to port application 1 to the new system; it will have to remain on the old system. Nevertheless, it might still pay off to port some of the other applications, accruing the associated benefit and incurring the expense of the interaction between applications on different systems. So this is the question: which of the remaining applications, if any, should be moved? Give a polynomial-time algorithm to find a set $S \subseteq \{2, \ldots, n\}$ for which the sum of the benefits minus the expenses of moving the applications in S to the new system is maximized.

3. You are given a directed graph and special vertices s and t in the graph. Give an efficient algorithm to find the maximum number of vertex disjoint paths (i.e., no two paths should share a common vertex) from s to t. How will you solve this problem if the graph is undirected?

4. **[KT-Chapter7]** There is a set of k people $\{p_1, \ldots, p_k\}$ and for a sequence of N days, you are given a subset S_i of people who want to go work together in a car. You want to find a fair driving schedule. We say that the total driving obligation of a person

p_j over a set of days is the expected number of times that p_j would have driven, had a driver been chosen uniformly at random from among the people going to work each day. More concretely, the driving obligation for p_j can be written as

$$\Delta_j := \sum_{i:p_j \in S_i} \frac{1}{|S_i|}.$$

Ideally, we would like to require that p_j drives at most Δ_j times; unfortunately, Δ_j may not be an integer. So lets say that a driving schedule is a choice of a driver for each day–that is, a sequence $p_{i_1}, \ldots, p_{i_n}$ with $p_{i_t} \in S_t$ for each day t - and that a fair driving schedule is one in which each p_j is chosen as the driver on at most $\lceil \Delta_j \rceil$ days. Prove that for any sequence of sets $S_1, \ldots, S_n$ there exists a fair driving schedule. Give an efficient algorithm to compute a fair driving schedule.

5. Prove the analogue of max-flow min-cut theorem in the circulation problem when we have lower and upper bounds on edge capacities and flow conservation needs to hold at every vertex.