Applied Algorithm Design: Exam

Prof. Pietro Michiardi

Rules and suggestions

- The idea is to complete first the questions and then the exercises.
- Having **the good** answer for all questions will guarantee to pass the exam with 10/20.
- Exercises are difficult if you are not familiar with the course contents. Suggestion: pick one exercise that you think is more feasible for you and focus on that one.

Questions

1. In a content resolution protocol, you need to set the probability p for a process to access a shared resource that can be used by *one and only one* process at a time. Give an intuition on how to set p. Optionally, show mathematically what is the best p value.

2. Given a digraph $G = (V, E)$ and two nodes s and t, find the minimum number of edges whose removal disconnects t from s.
   (a) Given this problem, which algorithm can be used to find the solution?
   (b) Which is the running time of this algorithm?
   (c) Why is this question is important in real systems?

3. In class we have seen the Local Search algorithm to address a simple version of the the facility location problem, namely the k-median problem (Lecture 6). Give the intuition of a Local Search approach to solve the Uncapacitated Facility Location problem. You can use as a baseline the Local Search algorithm for the k-median problem.

4. Recall from Lecture 4 the problem of finding a s, t-minimum cut, given a graph G. Given the more general problem of finding a Global Minimum Weight Cut defined in Lecture 7, simply name[1] two algorithms (one deterministic, one randomized) to solve the Global Min-Cut problem.

5. In the Gale-Shapley Algorithm we saw in Lecture 1, we studied an important property of the result of different executions of the algorithm on the same input data. This property is desirable in case the Algorithm is executed in an Asynchronous environment (see Lecture 1). Briefly[2] discuss what is the property we hint at and what are the implications.

Exercise 1

The Center Selection Problem we have seen in class has been studied through the lenses of approximation algorithms.

Here, instead, we take a simple local search approach to the problem. In this problem, we are given a set of sites $S = \{s_1, s_2, ..., s_n\}$ in the plane, and we want to choose a set of k centers $C = \{c_1, c_2, ..., c_k\}$ whose covering radius[3] is as small as possible.

We start by arbitrarily choosing k points in the plane to be the centers $c_1, c_2, ..., c_k$. We then alternate the following two steps.

1. Given the set of k centers $c_1, c_2, ..., c_k$ we divide S into k sets: For $i = 1, 2, ..., k$ we define S_i to be the set of all the sites for which c_i is the closest center

2. Given this division of S into k sets, construct new centers that will be as "central" as possible relative to them. For each set S_i, we find the smallest circle in the plane that contains all points in S_i, and define the center c_i to be the center of this circle.

If steps 1) and 2) cause the covering radius to strictly decrease, then we perform another iteration; otherwise the algorithm stops.

The alternation of steps 1) and 2) is based on the following natural interplay between sites and centers. In step 1) we partition the sites as well as possible given the centers; and then in step 2) we place the centers as well as possible given the partition of the sites.

Question 1: Prove that this local search algorithms eventually terminates (give a simple sketch of the proof).

[1] Which really means only give the name!
[2] A couple of bullet items
[3] The farthest that people in any one site must travel to their nearest center.

Question 2: Consider the following statement.

There is an absolute constant $b > 1$ (independent of the input of the algorithm), so when the local search algorithm terminates, the covering radius of its solution is at most b times the optimal covering radius.

Decide whether you think this statement is true or false, and give a sketch of the proof of either the statement or its negation.

`[Recall]`: **The Center Selection Problem**

- We have a set $S = \{s_1, s_2, \cdots, s_n\}$ of n sites to serve
- We have a set $C = \{c_1, c_2, \cdots, c_k\}$ of k centers to place

The problem: Select k centers C placement so that maximum distance form a site to the nearest center is minimized

Exercise 2

A number of *peer-to-peer* systems on the Internet are based on *overlay networks*. Rather than using the physical Internet topology as the network on which to perform computation, these systems run protocols by which nodes choose collections of virtual "neighbors" so as to define a higher-level graph whose structure may bear little or no relation to the underlying physical network. Such an overlay network is then used for sharing data and services, and it can be extremely flexible compared with a physical network, which is hard to modify in real time to adapt to changing conditions.

Peer-to-peer networks tend to grow through the arrival of new participants, who join by linking into the existing structure. This growth process has an intrinsic effect on the characteristics of the overall network. Recently, people have investigated simple abstract models for network growth that might provide insight into the way such processes behave, at a qualitative level, in real networks.

Here's a simple example of such a model. The system begins with a single node v_1. Node then join one at a time; as each node joins, it executes a protocol whereby it forms a directed link to a single other node chosen uniformly at random from those already in the system. More concretely, if the system already contains nodes $v_1, v_2, ...v_{k-1}$ and node v_k wishes to join, it randomly selects one of $v_1, v_2, ..., v_{k-1}$ and links to this node.

Suppose we run this process until we have a system consisting of nodes $v_1, v_2, ...v_n$; the random process described above will produce a directed network in which each node other than v_1 has exactly one outgoing edge. On the other hand, a node may have multiple incoming links, or none at all. The incoming links to a node v_j reflect all the other nodes whose access into the system is via v_j; so if v_j has many incoming links, this can place a large load on it. To keep the system load-balanced, then, we'd like all nodes to have roughly comparable number of incoming links. That's unlikely to happen here, however, since nodes that join earlier in the process are likely to have more incoming links that nodes that join later. Let's try to quantify this imbalance.

Question 1: Given the random process described above, what is the expected number of incoming links to node v_j in the resulting network? Give an exact formula in terms of n and j, and also try to express this quantity asymptotically (via an expression without large summations) using $\Theta(\cdot)$ notation.

Question 2: The previous question precise a sense in which the nodes that arrive early carry an "unfair" share of the connections in the network. Another way to quantify the imbalance is to observe that, in a run of this random process, we expect many nodes to end up with no incoming links.

Give a formula for the expected number of nodes with no incoming links in a network grown randomly according to this model.