Engineering and Applied Science Programs for Professionals
Whiting School of Engineering
Johns Hopkins University
605.621 Foundations of Algorithms
Homework 7
Due at the end of Module 14

Total Points 100/100

Collaboration groups will be set up in Blackboard by the end of the week. Make sure your group starts an individual thread for each collaborative problem and subproblem. All members of the collaboration group are expected to participate fully in solving collaborative problems, and peers will assess performance at the end of the assignment. Note, however, that each student is required to write up their solutions individually. Common solution descriptions from a collaboration group will not be accepted. Furthermore, to receive credit for a collaborative problem, each student in the collaboration group must actively and substantively contribute to the collaboration.

Self-Study Problems

All of the following problems come from the textbook and have solutions posted on the web at:

http://mitpress.mit.edu/algorithms.

You are permitted to use this site to examine solutions for these problems as a means of self-checking your solutions. These problems will not be graded.

Problems: 5.2-4, 5.2-5, 5.3-2, 5.3-4.

Problems for Grading

1. **Problem 1 Chapter 5**
   20 Points Total

   CLRS 5.3-3: Suppose that instead of swapping element $A[i]$ with a random element from the subarray A[i,...,n], we swapped it with a random element from anywhere in the array.

   Algorithm 1 Permute with All

   ```
   PermuteWithAll($A$)
   1  $n = A.length$
   2  for $i = 1$ to $n$
   3       swap $A[i]$ with Random$(1, n)$
   ```

   Does this code produce a uniform random permutation? Why or why not?

2. **Problem 2 Parts a and b**
   20 Points Total 10 Points Each

   Suppose you are acting as a consultant for the Port Authority of a small Pacific Rim nation. They are currently doing a multi-billion-dollar business per year, and their revenue is constrained almost entirely by the rate at which they can unload ships that arrive in the port. Here is a basic sort of problem they face.

   A ship arrives with n containers of weight $w_1, w_2, ..., w_n$. Standing on the dock is a set of trucks, each of which can hold K units of weight. (You may assume that K and w_i are integers.) You can stack multiple containers in each truck, subject to the weight restrictions of K. The goal is to minimize the number of trucks that are needed to carry all the containers. This problem is NP-complete.

   A greedy algorithm you might use for this is the following. Start with an empty truck and begin piling containers $1, 2, 3, ...$ into it until you get to a container that would overflow the weight limit. (These containers might not be sorted by weight.) Now declare this truck "loaded" and send it off. Then continue the process with a fresh truck. By considering trucks one at a time, this algorithm may not achieve the most efficient way to pack the full set of containers into an available collection of trucks.

   (a) Give an example of a set of weights and a value for K where this algorithm does not use the minimum number of trucks.

   (b) Show that the number of trucks used by this algorithm is within a factor of two of the minimum possible number for any set of weights and any value of K.

3. **Problem 3 Chapter 5 Parts a and b** ***Note this is a Collaborative Problem***
   40 Points Total 20 Points Each

   A number of peer-to-peer systems on the internet are based on overlay networks. Rather than using the physical internet 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 in real networks at a qualitative level.

   Here is a simple example of such a model. The system begins with a single node v_1. Nodes 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 $v_1, v_2, \ldots, v_{k-1}$ and node v_k wishes to join, it randomly selects one of $v_1, v_2, \ldots, v_{k-1}$ and links to that node.

   Suppose we run this process until we have a system consisting of nodes $v_1, v_2, \ldots, v_n$; 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. Then to keep the system load-balanced, we would like all nodes to have a roughly comparable number of incoming links. That is unlikely to happen, however, since nodes that join earlier in the process are likely to have more incoming links than nodes that join later. Let us try to quantify this imbalance as follows.

   (a) 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 using $\Theta(\cdot)$ notation.

   (b) Part (a) makes 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.

4. **Problem 4** ***Note this is a Collaborative Problem***
20 Points Total

Consider the following problem. There is a set U of nodes, which we can think of as users (e.g., these are locations that need to access a service, such as a web server). You would like to place servers at multiple locations. Suppose you are given a set S of possible sites that would be willing to act as locations for servers. For each site $s \in S$, there is a fee $f_s \geq 0$ for placing a server at that location. Your goal will be to approximately minimize the cost while providing the service to each of the customers.

So far, this is very much like the Set Cover Problem: The places s are sets, the weight of a set is f_s, and we want to select a collection of sets that covers all users. But there is one additional complication. Users u can be served from multiple sites, but there is an associated cost d_{us} for serving user u from site s. As an example, one could think of d_{us} as a "distance" from the server. When the value d_{us} is very high, we do not want to serve user u from site s, and in general, the service cost d_{us} serves as an incentive to serve customers from "nearby" servers whenever possible.

So here is the question, which we call the Facility Location Problem. Given the sets U and S with associated costs d and f, you need to select a subset $A \subseteq S$ at which to place the servers (at a cost of $\sum_{s \in A} f_s$) and assign each user u to the active server where it is cheapest to be served, $\min_{s \in A} d_{us}$. The goal is to minimize the overall cost

$$\sum_{a \in S} f_s + \sum_{u \in U} \min_{s \in A} d_{us} \tag{1}$$

Give a $\rho(n)$-approximate algorithm for this problem.