CS 3343 Analysis of Algorithms – Fall 2011

11/1/11

9. Homework

Due **11/8/11** before class

1. **Product (6 points)**

   Let A and B be two sets of n positive integers. You get to reorder each set however you like. After reording, let a_i be the i-th element of A and b_i be the i-the element of B. The goal is to maximize the function $\Pi_{i=1}^{n} a_i^{b_i}$. You will develop a greedy algorithm for this task.

   (a) Describe a greedy idea on how to solve this problem, and shortly argue why you think it is correct (not a formal proof).

   (b) Describe your greedy algorithm in pseudocode. What is its runtime?

2. **Points on the line (6 points)**

   Given a sequence $A = \{a_1, \ldots, a_n\}$ of points on the real line. The task is to determine the smallest set of unit-length (closed) intervals that contains all of the input points. Consider the following two greedy approaches:

   (a) Let I be an interval that covers the most points in A. Add I to the solution, remove the points covered by I from A, and recurse/continue.

   (b) Add the interval $I = [a_1, a_1 + 1]$ to the solution, remove the points covered by I from A, and recurse/continue.

   One of these approaches is correct, the other one is not. Show which of the approaches is not correct by finding a counter-example. (The counter example consists of an example input and two "solutions" – one is the actual optimal solution, the other is the solution computed by the greedy algorithm, which is not as good as the optimal solution.)

Flip over to back page $\Longrightarrow$

Practice Problems

(Not required for homework credit.)

1. **Product**

   Let A and B be two sets of n positive integers. You get to reorder each set however you like. After reording, let a_i be the i-th element of A and b_i be the i-the element of B. The goal is to maximize the function $\Pi_{i=1}^{n} a_i^{b_i}$. In the homework you will develop a greedy algorithm for this task.

   Consider $A = \{2, 5, 10, 20\}$ and $B = \{1, 2, 3, 4\}$. What would be the optimal reorderings of A and B, and why?

2. **Knapsack**

   Consider the Knapsack problem: A thief robbing a store finds n items, where the i-th item is worth v_i dollars and weighs w_i pounds (all v_i and w_i are integers). The thief wants to maximize the total value of the items he takes, but he can only carry at most W pounds in his knapsack. Which items should he take? (In order to understand the problem better, you may want to consider an example of items with values $12, 15, 4$ and weights $2, 3, 1$, and $W = 4$.)

   There are two versions of this problem: in the **0-1 Knapsack** the thief has to make a binary decision whether the thief takes the item or not. In the **fractional Knapsack** problem the thief is allowed to take only a part of the item (this does not make sense with TVs or such, but if the items are dust of precious metals or piles of different kinds of jewels this is more reasonable).

   Design a greedy algorithm for the **fractional Knapsack** problem. Just describe the basic idea of your algorithm, argue shortly why it is correct (no proof), and give its runtime. Then show how the same greedy algorithm, when applied to the **0-1 Knapsack** problem, does not work by giving a counter example (which shows that non-greedy gives a better solution).

3. **DFS, BFS**

   Run BFS and DFS on the graph below, starting on vertex a. Write the visit times (and for DFS finish times) into the vertices. Assume that vertices are ordered alphabetically in the adjacency lists.