Department of Electronics Engineering
National Chiao Tung University
Algorithms, Fall 2017

December 6, 2017
Iris Hui-Ru Jiang

DEE3504:
Homework #3 (due 11:59AM, December 21, 2017)

Instructions: Submit your solution to TAs at ED413. No late submission allowed. Please list your collaborator and/or references (if any) for each problem.

Reading:

1. Chapters 5 and 6 including solved exercises.

Exercises:

1. (25 pt) 5.3 Suppose you're consulting for a bank that's concerned about fraud detection, and they come to you with the following problem. They have a collection of n bank cards that they've confiscated, suspecting them of being used in fraud. Each bank card is a small plastic object, containing a magnetic stripe with some encrypted data, and it corresponds to a unique account in the bank. Each account can have many bank cards corresponding to it, and we'll say that two bank cards are equivalent if they correspond to the same account.
   It's very difficult to read the account number off a bank card directly, but the bank has a high-tech "equivalence tester" that takes two bank cards and, after performing some computations, determines whether they are equivalent. Their question is the following: among the collection of n cards, is there a set of more than $n/2$ of them that are all equivalent to one another? Assume that the only feasible operations you can do with the cards are to pick two of them and plug them in to the equivalence tester. Show how to decide the answer to their question with only $O(n \log n)$ invocations of the equivalence tester.

2. (25 pt) There are n items in a store. For $i = 1, 2, \ldots, n$, item i has weight $w_i > 0$ and worth $v_i > 0$. A thief can carry a maximum weight of W pounds in a knapsack. Each item can be broken into smaller pieces, so the thief may decide to carry only a fraction x_i of item i, where $0 \leq x_i \leq 1$. Item i contributes $x_i w_i$ to the total weight in the knapsack, and $x_i v_i$ to the value of the load. Design an algorithm which can maximize $\sum_{i=1..n} x_i v_i$ subject to $\sum_{i=1..n} x_i w_i \leq W$ in $O(n \log n)$ time.

3. (25 pt) We are planning a hiking trip with total distance n kilometers. On the route, there are hotels located i kilometers from the starting point for every integer i. The hotel located i kilometers from the starting point has cost c_i. Assume that we want to walk at most 20 kilometers each day, design an efficient algorithm to minimize the total hotel cost. You may assume c_n=0. (Partial points for larger runtime.)

4. (25 pt) 6.11 Suppose you're consulting for a company that manufactures PC equipment and ships it to distributors all over the country. For each of the next n weeks, they have a projected *supply* s_i of equipment (measured in pounds), which has to be shipped by an air freight carrier.
   Each week's supply can be carried by one of two air freight companies, A or B.

- Company A charges a fixed rate r per pound (so it costs $r \cdot s_i$ to ship a week's supply s_i).
- Company B makes contracts for a fixed amount c per week, independent of the weight. However, contracts with company B must be made in blocks of four consecutive weeks at a time.

A *schedule*, for the PC company, is a choice of air freight company (A or B) for each of the n weeks, with the restriction that company B, whenever it is chosen, must be chosen for blocks of four contiguous weeks at a time. The *cost* of the schedule is the total amount paid to companies A and B, according to the description above.

Give a polynomial-time algorithm that takes a sequence of supply values s_1, s_2, …, s_n and returns a *schedule* of minimum cost. (Partial points for larger runtime.)

Example. Suppose $r = 1$, $c = 10$, and the sequence of values is

11, 9, 9, 12, 12, 12, 12, 9, 9, 11.

Then the optimal schedule would be to choose company A for the first three weeks, then company B for a block of four consecutive weeks, and then company A for the final three weeks.