Assignment 4

Algorithms: Correctness and Complexity

Due 2014 Dec 2.

Q0 [20] MARS

You are on the operations team of the CSA's Mars Ascend and Return Samples mission. The rover has identified hundreds of samples (set S) that could be returned. Each sample $x \in S$ has a real weight $w(x)$ and a real value $v(x)$. A subset of S is optimal if it maximizes the total value while the total weight is under or equal to $w_{\max}$.

(a) [10] Give a greedy algorithm for this problem. Your algorithm need not find an optimal subset, but it should do a reasonable job at selecting a reasonable subset.

(b) [10] Give counter-examples showing that each of the three most obvious ways of ranking the samples can lead your algorithm to find suboptimal subsets.

Q1 [10] You work for ZipTrip.com, the travel site for people in a hurry. They guarantee to find the quickest sequence of flights from location x to location y that such that the first flight leaves on or after time t.

Their database consists of a set of flights where each flight is associated with a starting airport, a destination airport, a boarding time, and a landing time.[1] Your algorithm needs to find the set of flights that gets a passenger from airport x at time t to airport y the soonest. Layovers where the passenger must change plane must be at least 30 minutes between landing and boarding time, 60 minutes if the passenger must change terminals, or 120 minutes if the passenger must pass through immigration and customs.. Find an efficient algorithm to solve this problem. Keep in mind that ZipTrip.com will be processing many such queries on the same data base, so some preprocessing of the data base is reasonable to do.

---

[1] In reality a flight might have stops.. However we can treat such a flight as several simple flights for our purposes.

Q2 [**20**] As mentioned in class, the greedy algorithm for making change fails for certain mixes of denominations, for example $\{1, 5, 10, 25\}$. Develop a dynamic programming solution for the following problem. Input: a natural number t and a set S of coins, each having a natural number value. Output: A minimal sized subset of coins that have total value t.

(a) [**5**] Develop a recursive algorithm to determine whether there is or is not a subset of S that has total value t.

(b) [**5**] Develop an efficient top-down (memoizing) algorithm to solve the problem in part (a)

(c) [**5**] Develop an efficient bottom-up algorithm to solve the problem of part (a).

(d) [**5**] Modify your solution to either part (b) or (c) to output a minimal sized subset of coins that adds up to t, if there is one.