CS545 Spring, 2015

Homework Assignment #3
Due: March 26
Student: Shuo Yang

---

1. (a) Suppose we have 6 activities, for activity i, s_i is the start time, f_i is the finish time and d_i is the duration. And we sort them in order of increasing duration:

i	1	2	3	4	5	6
s_i	8	3	1	4	4	0
f_i	9	5	4	8	9	6
d_i	1	2	3	4	5	6

The greedy procedure when the activities are considered in order of increasing duration would be: 1) select activity 1 since it is the local best; 2) select activity 2 since it is compatible with activity 1; 3) no other activities can be selected since they are not compatible with either activity 1 or 2. So the procedure yields the solution $\{1, 2\}$, but the optimal solution should be $\{3, 4, 1\}$. Therefore the greedy procedure is not correct.

(b) With the same input, now let us sort them in order of increasing start-time:

i	1	2	3	4	5	6
s_i	0	1	3	4	4	8
f_i	6	4	5	8	9	9

The greedy procedure when the activities are considered in order of increasing duration is:

Function *GreedyActivitySelection*(S, F, n)
    $A := \{1\}$ // activity with the earliest start time
    $j := 1$ // indicates the activity with greatest start time in A
    for $i := 2$ to n
        if $S[i] \geq F[j]$
            $A := A \cup \{i\}$
            $j := i$
    return A

Running the above procedure with the same input yields the solution $\{1, 6\}$ which is certainly not an optimal solution. Therefore the greedy procedure is not correct.

(c) Consider the following 9 activities ordered by the number of overlaps (o_i):

i	1	2	3	4	5	6	7	8	9
s_i	4	0	7	1	2	3	5	6	6
f_i	6	3	10	4	4	5	7	8	9
o_i	2	2	2	3	3	3	3	3	3

When considering the order of increasing number of overlaps, the greedy procedure would first pick activity 1, then pick activity 2, and finally pick activity 3. Thus the solution derived by the greedy procedure is $\{1, 2, 3\}$, but the optimal solution should be $\{2, 6, 7, 3\}$. Therefore the greedy procedure is not correct.

2. Label n gas stations as $g_1, g_2, \cdots, g_n$ and let d_i be the distance between gas stations g_i and g_{i+1}, specially, d_0 denotes the distance between g_1 and city A, and d_n denotes the distance between city B and g_n. Let D be the total distance between city A and B, where $D = \sum_{i=0}^{n} d_i$.

Greedy algorithm

The greedy strategy is to refuel at the gas station whose distance from the current stop is closest to m but less than m.

Function $GreedyTripRefuel()$
  $A := \emptyset$ // the set of refueling stops
  $i := 0$ // indicates the farthest gas station one can make.
  $T_1 := 0$ // distance traveled before making a refuel
  $T_2 := 0$ // total distance traveled
  while $T_2 < D$
    if $(T_2 - T_1) \leq m$
      $T_2 := T_2 + d_i$
      $i := i + 1$
    else
      $i := i - 1$ // cannot make it to station g_i, so must refuel at station g_{i-1}
      $A := A \cup \{g_i\}$
      $T_2 := T_2 - d_i$
      $T_1 := T_2$
  return A

The algorithm runs in $O(n)$ time since there are at most n gas stations along the way.

Correctness

Lemma: Suppose A is a subset of an optimal solution where the latest refuel stop is station g_k. Let g_i be the gas station after g_k whose distance from g_k is closest to m but less than m. Then $A \cup \{g_i\}$ is also a subset of an optimal solution.

Proof. Since A is a subset of an optimal solution, let A^* be an optimal solution with $A \subseteq A^*$. Let g_j be the gas station in $A^* - A$ that is next to g_k. We have two cases:

- case-1: $i = j$. Then $A \cup \{g_i\} \subseteq A^*$, the lemma holds.
- case-2: $i \neq j$. Since the distance from g_k to g_i is closest to m but less than m, this distance must be greater than the distance from g_k to g_j. Thus we can replace g_j with g_i and still yields an optimal solution since g_i is closer to the future stops than g_j.

□

Theorem: $GreedyTripRefuel()$ finds an optimal solution.

Proof. Initially, A is an empty set, which is a subset of an optimal solution and g_k in this case would be city A. By the lemma, $A \cup g_i$ is a subset of an optimal solution where g_i is the gas station whose distance from city A is closest to m but less than m.

By induction on the number of iterations, when the function terminates, A is a subset of an optimal solution.

Since from the starting point, the greedy algorithm makes the least possible stops, there is no better solution with fewer stops than the one produced by the greedy algorithm. Therefore A is optimal. □

3.

4.