Solutions for Exercise Sheet 12

Handout: December 5th — Deadline: December 12th, 4pm

Question 12.1 (0.5 marks)

Suppose that you modify GREEDY-ACTIVITY-SELECTOR to use the following greedy strategies. State whether each strategy would yield an optimal solution or not. If they do, then provide a proof of optimality. If they don't, then provide an example instance where the strategy fails.

1. Always select the activity of least duration amongst those that are compatible with all previously selected activities
2. Always select the compatible activity that overlaps with the fewest remaining activities
3. Always select the last activity to start that is compatible with all previously selected activities
4. Always select the compatible activity with the earliest start time

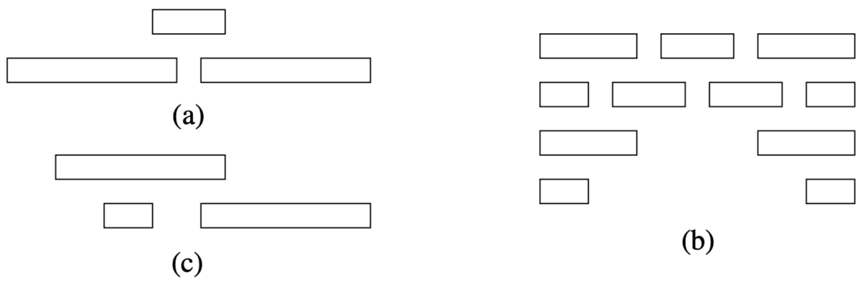

Figure 1: Examples

Solution.

1. The strategy fails for example on this instance: $s_1 = 1, f_1 = 5, s_2 = 4, f_2 = 6, s_3 = 5, f_3 = 10$. (See Fig. 1 (a)).
2. The strategy fails for example with the activities in Fig. 1 (b). The activity that overlaps least is the second one on the first row. Selecting this activity first forces a solution with only 3 activities while the optimal solution has 4 activities.
3. This strategy is symmetric to the one used at lecture and works for the same reason, and you can prove that the greedy choice is *safe*.

   Consider any nonempty subproblem S_k, and let a_m be an activity in S_k with the latest start time. Then a_m is included in some maximum-size subset of mutually compatible activities of S_k. Consider an optimal solution A_k. If it contains a_m, you're done. Otherwise swap a_m in with a_j out, where a_j is the activity in the solution with latest start time. Since a_m cannot start earlier than a_j, it must also be compatible with all the other activities or a_j wouldn't be compatible either. CVD.

4. The strategy fails for example if the first activity to start is also the last one to finish. Then all the other activities are force out of the solution. Another example is provided in Fig. 1 (c).

Question 12.2 (0.25 marks)

Prove that the fractional knapsack problem has the greedy choice property, hence always finds an optimal solution.

Solution.

The Greedy algorithm sorts the items according to value per gram: let the greedy solution be $a_1, a_2, \ldots, a_m/i$, of decreasing value per gram and where a_m/i may be some fraction of a_m if it doesn't all fit or exactly a_m if it fits (i.e. then $i = 1$). Let's assume there is a different optimal solution using some a_{m+i} object(s) with lower value per gram instead of one (or more) of the a_i objects in the greedy solution. Then we could just swap them in and out and achieve the same solution value.

Question 12.3 (0.5 marks)

Eddy takes part in a cycle race from start s_1 to finish s_n with feed stations $s_2, \ldots, s_{n-1}$ along the way and distances d_i between s_i and s_{i+1}. To save time, Eddy plans to stop at the smallest possible number of stations. He knows that he can cycle distance ℓ without stopping for supplies, where $\ell > d_i$ for all $1 \leq i \leq n-1$.

(a) Design a greedy algorithm that computes the minimal number of stops for Eddy.

(b) Argue why your greedy strategy yields an optimal solution.

Solution.

(a) The greedy strategy is to choose at any feed station where Eddy stops the most distant feed station he can reach. Thus from any s_k he should stop at $s_{k'}$ where $k < k'$ and

$$\sum_{i=k}^{k'} d_i \leq \ell < \sum_{i=k}^{k'+1} d_i.$$

An informal description like the above is sufficient. In pseudocode, the algorithm could look like this. In this code, we count the number of stops in a variable *s-nr*. We sum up distances from the last stop in a variable *loc-sum*, and when distances exceed ℓ, and i is the current loop counter, we count a stop at feed station s_{i-1} (the one just before Eddy exceeds his maximum distance of ℓ). The algorithm then continues with *loc-sum* $= d_i$ as that is the distance Eddy has travelled since stopping at s_{i-1}.

```
FEED-STATION($d_1, \ldots, d_{n-1}, \ell$)
1  $s\text{-}nr = 0$
2  $loc\text{-}sum = 0$
3  for $i = 1$ to $n-1$
4      $loc\text{-}sum = loc\text{-}sum + d_i$
5      if $loc\text{-}sum > \ell$
6          $s\text{-}nr = s\text{-}nr + 1$
7          $loc\text{-}sum = d_i$
8  return $s\text{-}nr$
```

NB: the algorithm can easily be extended to print out the actual feed stations by adding a line that prints s_{i-1} inside the *if* statement.

(b) Let $S = (s_{o_1}, \ldots s_{o_m})$ be an optimal sequence of feed stations. We show that the greedy choice s_{g_1} for the first feed station is always safe. Formally, similar to Theorem 16.1 for activity selection, we show that there is a solution with a minimum number of stops that includes s_{g_1} as first stop.

If $s_{o_1} = s_{g_1}$ the claim is true and there is nothing to prove. Otherwise, we must have $s_{o_1} < s_{g_1}$ as s_{g_1} is the furthest reachable feed station. If we replace s_{o_1} with s_{g_1}, we get a new solution $S' = S \setminus \{s_{o_1}\} \cup \{s_{g_1}\}$. This solution is feasible as $d(s_1, s_{g_1}) \leq \ell$ and the distance to the second feed station has not increased: $d(s_{o_1}, s_{o_2}) \leq \ell$ and $s_{o_1} < s_{g_1}$ imply $d(s_{g_1}, s_{o_2}) \leq \ell$. Since we only replaced a feed station, both solutions have the same number of stops, $|S'| = |S|$. Hence we have shown that there is solution, S', with a minimum number of stops that makes a greedy choice for the first feed station.

We can now consider the subproblem given by all stops from s_{g_1}. Iterating the above argument shows that the greedy algorithm will compute an optimal solution.

Question 12.4 (0.25 marks)

Implement both Recursive-Activity-Selector(s, f, k, n) and Greedy-Activity-Selector(s, f, n). The algorithms take in input the array s of starting times, the array f of ending times and the number of activities n. The elements are already sorted according to finish times.