CS157 Homework 5

BY CHAOQIAN AND SILAO_XU

March 3, 2013

Problem 1

Bilbo has a long journey ahead of him and wants to stay in comfortable hotels along his route each night, but can only walk 20 miles in a day. Fortunately, he has a guidebook charting the locations of all the hotels along his route. (Bilbo already knows which route he is taking; he only has to choose where along his route to stay each night.)

1. (8 points) Find a greedy algorithm for Bilbo to compute how to finish his journey in the least number of days. Points on this problem will only be given for the proof that your algorithm is optimal; more points will be given for simpler and clearer proofs.

   - ***Greedy Algorithm***

     Let start point be s and the termination be T. The following algorithm will find the next hotel as far as possible within 20 miles for each hop.

     ```
     01 def Find-Way(s):
     02     n = Get-Next(s)
     03     if n = T then do:
     04         return 0
     05     else:
     06         return 1 + Find-Way(n)
     07 end
     08
     09 def Get-Next(s):
     10     n = s.nextHotel
     11     while n.distFromStart - s.distFromStart ⩾ 20 do:
     12         s = n
     13         n = s.nextHotel
     14     end
     15     return n
     16 end
     ```

   - ***Correctness***

     - **Claim**. *Find-Way* terminates.

       Proof: The *Find-Way* would recurse on the next route point (the hotel along the route he will live in at night). Since the destination of the route is known, therefore line 3 would become true at some point.

     - **Claim**. *Find-Way* satisfies the feasibility that he needs to walk at most 20 miles every day in the trip.

       Proof: Based on the assumptFind-MinCost-Wayion that the distance between every consecutive pair of hotels would not exceed 20 miles, in *Get-Next* line 10, the first next hotel we look at would not 20 miles further than s. Also, in *Get-Next* line 11 we limit that the furthest hotel from s would not exceed 20 miles.

So we conclude that *Find-Way* satisfies the feasibility that Bilbo would walk at most 20 miles every day in the trip.

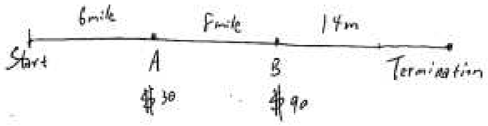
- **Claim**. *Find-Way* satisfies the optimality criteria that he can finish his journey in the least number of days.

  Proof: Let S^* be the optimal solution got from *Find-Way*. Assume that there is an optimal solution S that require less days than solution S^* for Bilbo to finish the tour, there must be a day d, in the entire trip such that Bilbo could walk more than *Find-Way* solution. However, based on the fact that the distance between every consecutive pair of hotels would not exceed 20 miles and the algorithm we construct in *Get-Next* line 11, if in day d solution S Bilbo would walk more than the miles in solution S^*, then Bilbo have to walk more than 20 miles. Contradiction!

  So, no other solutions would be better than S^*, *Find-Way* satisfies the optimality criteria that Bilbo can finish his journey in the least number of days.

2. (3 points) After studying your algorithm for the previous problem, Bilbo realizes that, actually, the different hotels cost different amounts, and what he actually wants to do is minimize the total cost of his journey. (Luckily, his guidebook also lists the cost of each hotel.) He thinks of the following greedy algorithm: wherever he is, for each hotel within a day's walk of him (20 miles), he computes the "cost per mile" of staying there, dividing its cost by the amount of progress he would make by staying there; given this list of costs, he then chooses to spend his next night at the hotel with the best cost per mile. Demonstrate for Bilbo that being greedy can be costly, that is, describe an example where Bilbo's algorithm gives suboptimal performance.

   Suppose the route is as follows:

   Based on the greedy algortihm he constructs, in the very beginning, hotel A is his next destination since $\frac{30}{6m} < \frac{90}{6m+8m}$, and the second hotel he would arrive would be B since within 20 miles, B is the only hotel. In this case, Bilbo would live in 2 hotels spending 120. But the optimal solution would be walk through hotel B in his first day and that would only charge him 90.

3. (4 points) Find a dynamic programming algorithm for Bilbo's problem. Make it clear to Bilbo why it works, including an explanation of the meaning of any tables you ask Bilbo to construct. (Your solution for this part shFind-MinCost-Wayould look like an explanation, not a proof).

   - ***DP Algorithm***

     The pseudocode is as follows:

     ```
     s ← departure point
     t ← termination point
     N ← total number of hotels
     label each hotel in the route from s to t from 1 to N
     construct a (N+1) array ''hotels'' in which
         each element i stores the minimum cost from s to hotel i
     construct a (N+1) array ''hops'' in which each element i stores
     ```

```
    the previous hotel number Bilbo would live in before hotel i

def Find-MinCost-Way(t):
    hotels[0] ← 0
    hops[0] ← s
    for j from 1 to N do:
        for each hotel i with 20 miles before location j:
            hotels[j]  ←  min{hotels[i] + j.cost}
            hops[j]  ← hotel index of min{hotels[i]}
        end
    end
    for each hotel k within 20 miles before t then do:
        return min{hotel(k)}
    end
end
```

- ***Correctness*** (Proof by Induction)

  Base Case:

  If the distance from s to t is less than 20 miles, assume there is c hotels in the route. Then from s to hotel i (i ranges from 1 to c), the optimal cost of every intermediate destination, hotel i, is the cost of hotel i itself. Also, we have set hotel number 0 is of cost 0 and therefore the minimum cost from departure point to destination is of cost 0. Base case holds.

  Inductive Hypothesis:

  Assume in intermediate destination hotel k, in its previous 20 miles, there were m hotels and from s to hotel i (i ranges from $k - m$ to k) we have constructed optimal route for it.

  We need to prove that we would have the minimal cost starting from s, ending at hotel $k+1$.

  In hotel $k+1$'s previous 20 miles, there are no more than $m+1$ hotels. So based on our inductive hypothesis, from s to hotel i (i ranges from $k+1-(m+1)$ to k), we have constructed optimal route for it. For the route from s to hotel $k+1$, we could find the optimal route by choosing the minimal cost from the previous $m+1$ hotels' route and add hotel $k+1$'s cost.

  So we could find the optimal route from s to the termination.