Jeff Hildebrandt

Homework 3

COT 5405

**1)**

If OPT(j) is equal to the maximum number of robots that can be destroyed for the instance of the problem on x[1],x[2],…x[j] and f(j) is a function so that if j seconds have passed since the EMP was last used, it can destroy up to f(j) robots. So, where k is the number of seconds since the EMP was used, min(x[k], f(j)) would be how many robots could be killed that turn.

n = total number of seconds

OPT(0) = 0

*for j = 1; j <= n; j++*

*for i = j; i >= 0; i--*

*OPT(j) = max(OPT(i) + min(x[j], f(j-i)))*

*Return OPT(n)*

Runs in O(n^2) time

**2)**

n = final day where the tank needs to be empty

a = starting day where the tank is assumed empty

b = day when the tank is filled

b > a

P = price of delivery

L = max capacity of storage tank

S(a,b-1) = the cost to store the gas from day a to day b

OPT(a) = the optimal solution for days a through n where the tank is empty on day a

OPT(1) = starting day where the tank has to be filled on this day

*for a = n; a >=0; a--* (Note: here b is found as well as the number of gallons to buy)

*OPT(a) = P + () S(a, b-1)*

After all OPT values are found, start at OPT(1), from there OPT(b), then OPT(b)… until n is reached. Each time you get to another b day (day the tank was filled) , note how many gallons were bought and which day it was. Runs in O(n^2)

**3)**

**a)**

Given the following example:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Week 1** | **Week 2** | **Week 3** | **Week 4** | **Total** |
| **l** | 10 | 1 | 10 | 10 |  |
| **h** | 5 | 50 | 100 | 1 |  |
| **Algorithm Answer** | 0 | 50 | 10 | 10 | 70 |
| **Correct Answer** | 10 | 0 | 100 | 10 | 120 |

The algorithm doesn’t look far enough into the future. In this case, the algorithm incorrectly chose to pick the $50 task, when if it looked a little further it would see that a much better $100 task could have been chosen instead.

**b)**

OPT(i) = the max cumulative revenue of the current week and previous weeks

l[i] = low stress value of week i

h[i] = high stress value of week i

n = total weeks

if OPT goes out of bounds assume value = 0

*for i=1; i<=n; i++*

*OPT(i) = max(l[i] + OPT(i – 1), h[i] + OPT(i – 2))*

*return OPT(n)*

This algorithm calculates the maximum value as it iterates through the weeks. And it calls previous OPT values in O(1) time. Runs in O(n) time.