CS 577

Midterm

Q1

a) we know that the capacity for the tank (c) will always be greater than or equal to largest w_i i.e $c \geq \max\{w_i : i \in [n]\}$.

[redacted]: We have number of days (n), values of water needed w_i for $i \in [n]$ and maximum fills (f).

We need to find 'c' capacity of the storage tank so there is no water shortage and at most f fills.

Input: n (number of days), Array $w[1 \ldots n]$ (water values per day), f (maximum number of fills that can be used).

Output: c (capacity of storage tank such that no water shortage, at most f fills) $c \in \mathbb{N}$

→

procedure : minimum capacity(n, f, w[1..n]).

sum ← sum of array w.

if f == 0 :

return sum;

i ← largest element's index.

$c_1 \leftarrow w[i] + w[i-1]$

$c_2 \leftarrow w[i+1] + w[i]$

fact ← ⌈sum / f⌉

return max{ c_1, c_2, fact }.

b). we have n, $w[1..n]$, $l[1..n]$ (loss array), c.

we need for minimise the loss.

~~Our subproblems are for the first day~~ do we fill or not. If we fill, then we decrease

Input: n, $w(1..n)$, c, $l(1...n]$

Output: minimum loss.

procedure: minimum_loss($n, w(1...n], c, l(1...n]$.

$l \leftarrow l[1]$

sum $\leftarrow$ sum of array w.

fact $\leftarrow \lceil sum/c \rceil$

for $i = 1$ to n do

c).

Input: c, n, w[ ], l[ ].

output: a fill schedule.

procedure: fill-schedule (c, n, w[ ], l[ ]).

f_1 = black box (c, n, w[ ], l[ ]).

print (days:, f_1).

~~for: while~~

while (black box (c, n, w[ ], l[ ]) != null).

print (black box (c, n, w[ ], l[ ]).

return.

Thus we can see that we call black box $2n-1$ times and use print statement for $(2n-1)$ or $O(n)$ Time

Thus the calls are

$O(n)$. to black box.