stay ahead + exchange

Name: ______________________ Wisc id: ______________________

Coin Changing

1. *CLRS 3rd edition (p. 446).* Consider the problem of making change for n cents using the fewest number of coins. Assume that each coin's value is an integer.

   (a) Describe a greedy algorithm to make change consisting of quarters (25¢), dimes (10¢), nickels (5¢), and pennies (1¢). Prove that your algorithm yields an optimal solution.

   Solution:

   algo: to reduce the amount of cents used to make change it would be necessary that my algorithm chooses the highest amount of coin that doesn't exceed the desired amount @ each chose.

   Prove: use exchange to prove this. g_i will represent my greedy approach and o_i will be optimal approach.

   $g_1 \leq g_2 \leq \dots \leq g_n$ $o_m \leq g_n$ $m = n$

   $o_1 \leq o_2 \leq \dots \leq o_m$

   replace o_1 with g_1, since $o_1 \leq g_1$ then $\{g_1, o_2, \dots, o_m\}$ is optimal. Stabbing each interval to replace the optimal solution makes this greedy algorithm the equal or even better than the original optimal sol. $\{g_1, g_2, \dots g_m\}$

   (b) Suppose that the available coins are in the denominations that are powers of c, i.e., the denominations are c^0, c^1, ..., c^k for some integers $c > 1$ and $k \geq 1$. Show that the greedy algorithm always yields an optimal solution.

   Solution:

   it would be the same

   (not like that for toilet paper problem in sample exam)

(c) Give a set of coin denominations for which the greedy algorithm does not yield an optimal solution. Your set should include a penny (1¢) so that there is a solution for every value of n.

Solution:

12¢ 1¢, 2¢, 6¢, 9¢

9¢, 2¢, 1¢ X

6¢, 6¢

(d) Give an $O(nk)$-time algorithm that makes change for any set of k different coin denomination, assuming one of the coins is a penny.

Solution:

$$DP[i] = \min_{j=1 \dots n} \left(m[i - b_j] \right) + 1$$

min # of bundles to achieve i bundles

n diff bundle totals

$i = 0$
$m[0] = 0$
$i < 0$
$m[0] = \infty$

I Love Train Stations

stay ahead

2. There are towns that lies on a straight road, and the government is planning to build a railroad path along this road. You, as the project manager of this construction, needs to decide where to build train stations. Every town must be within distance R of a train station. The goal is to minimize the number of train stations built.

   (a) Consider the following algorithm: repeatedly build train stations where you can maximize the number of towns newly covered. Show that this algorithm is not optimal by giving a counter-example.

Solution:

this algorithm wants a train station at every town but can be minimized if a train station was in between both towns so both towns can use the train station as long as they are both ≤ R distance away

Instead of building 2 train stations we could use 1

   (b) Give an algorithm and prove that it's optimal.

Solution:

go along road once a town is hit go distance R away then build a train station. then once at next town check if there is already a TS R distance away if not go R distance to build if yes go to next town & check again

Give Me Classroom

3. Lecture j starts at s_j and finishes at f_j. Find the minimum number of classrooms to schedule all lectures so that no two occur at the same time in the same room.

Solution: we need another classroom if 2 intervals intersect times. (for 2 lectures)

Algo:

- sort by finish time
- iterate through each lecture, if theres an open classroom, schedule the class

• represent d as the amount of lectures that all conflict w/ one another.

the optimal solution will have exactly d amount of classrooms scheduled.

proof by contradiction: "my algorithm, G, has more than d classrooms scheduled".

• if there are d amount of strongly conflicted lectures, G will sort these class rooms by f_j first. next will check each lecture, and if they conflict then add another classroom. since all d lectures conflict, G will schedule d classrooms. theres a contradiction b/c G was said to create more than d.

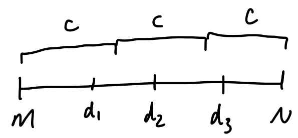
∴ G schedules d classrooms ∴ is optimal.

Name: ______________________ Wisc id: ______________________

Breakpoint Selection

1. You are driving on a straight road from Madison to Nosidam. Your car has a fuel capacity of C, which means that you can drive C miles after refueling. There are refueling stations along the way at $d_1, d_2, ..., d_n$. You drive from Madison with a full fuel tank. Your goal is to reach Nosidam with the fewest amount of times to refuel.

Solution:



I choose a greedy algo that consists of at each station decide whether you can make it to the next station with the amount of gas still in the tank. if you can, then go to next station, if you can't fill up tank to its capacity

Prove by stay ahead:

for my greedy algorithmm, I will represent g_i $\forall_i = 1 \dots n$ as my locations where I stop to fill up my tank. For the optimal algorithmm w/ least amount of stops I will rep as O_j $\forall_j = 1 \dots n$. My algo garentees that I pick the farthest station I can go without running out of gas $\therefore$ Claim: $g_i \geq O_i$.

for $g_m \geq O_n$ = destination,

$g_m \geq dst$

$g_m = dst$ b/c no more breakpoints

$\therefore$

$m = n$

BC: $i=1$, $g_1 \geq O_1$ due to nature of greedy

IH: assume $g_k \geq O_k$.

$g_k + C \geq O_k + C$ since $g_k \geq O_k$ adding c to each makes them go the same distance ad g_k is closer to dest. so $g_k + C \geq O_k + C$

$g_{k+1} \geq O_{k+1}$

$\therefore$

g_k is optimal

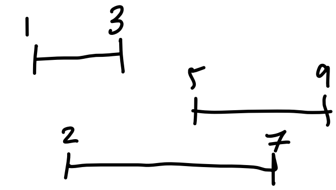
Interval Stabbing

2. There are n intervals $[[a_1, b_1], [a_2, b_2], ...[a_n, b_n]]$. We say that a set of point S **stabs** all the intervals if every interval in the set contains at least one point in S. Find a minimal set of point S.

Solution:

n intervals : [1, 3], [5, 9], [2, 7]

choose $S = \{ \quad \}$?



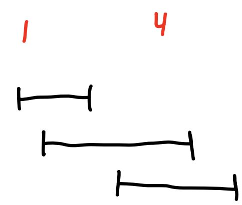
$S = \{1, 4, 7\}$

S stabs these intervals

not optimal

cuz $S = \{2, 7\}$ is optimal

or $S = \{3, 5\}$



Algo:

- sort by right endpoint
- start from intervals in ascending order
- make the first stab value to the largest # in this interval.
- go to next interval that hasn't been stabbed yet + go through same process

prove by exchange:

let G represent the stab intervals for my algo, $g_1 < g_2 < \dots < g_m$. let O represent the stab interval for the optimal algo. $o_1 < o_2 < \dots < o_n$.

BC: since $o_1 \leq g_1$, I can replace o_1 with g_1 as g_1 will be the the furthest stab that covers the first interval (after sorting). $O = \{g_1, o_2 \dots, o_n\}$

due to definition of greedy algo $g_i \; \forall i = 1 \dots m$ can be replaced with each $O_j \; \forall j = 1 \dots n$.

$\therefore O = \{g_1, g_2 \dots g_m\}$ $\therefore$ my algo is the optimal sol.

Preemptible Job Scheduling

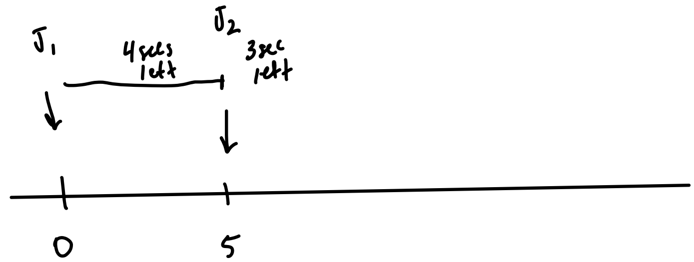
stop it even during execution

3. There are n jobs. Job i is available at time s_i, and it require p_i processing time. Jobs are preemptible. Design an algorithm to minimize $\sum_i c_i$ where c_i is the time when job i is completed. Prove its correctness.

Solution:

Algo:

- if there are no jobs running start processing time for job
- if there is jobs i that start time conflict with job that is processing, process the job that has the least amount of time left to process.
- continue process until all jobs are processed



$J_1 \rightarrow 5+3+4 = 12$

$J_2 \rightarrow 5+3 = 8$ ✓

$J_1 \rightarrow 5+4 = 9$

$J_2 \rightarrow 5+4+3 = 12$

by doing the less amount of time left, the faster job waits less time & therefore minimizes the total amount of time to complete each job.

prove by exchange:

x, y 2 jobs

x < y in remaining time

x : 2

y : 3

lets say O does y job first

	t_1			t_2	t_3	t_4	t_5
OPT	y			y	x	y	x
OPT'	x			x	y	y	y

OPT : $\min(c_x, c_y) + \max(c_x, c_y)$

OPT' : $t_{px} + \max(c_x, c_y)$

$t_2 \leq t_3$