ECS 122A Lecture 10 Jasper Lee

Greedy Algorithms

Def (From course textbook)

Construct a (partial) solution iteratively, via a seq of short-sighted decisions/choices, and hope that it works out at the end.

WARNING: "Most" greedy algs are wrong!

Note: But greedy algs can be good "approximation" algs (Very out of scope for 122A).

Fractional knapsack

Setting:
- Knapsack has capacity $C \geq 0$
- List of n divisible items (think liquid)
  - Weight $w_i \geq 0$
  - Reward $r_i \geq 0$

can take only a fraction p_i of item i and get reward $p_i r_i$.

Constraint: Total weight $\leq C$

What are some possible greedy strategies (which might be wrong)?

Alg candidates:

Greedily fill Knapsack w/:

- Smallest w_i?
- Biggest r_i?
- Biggest reward density r_i/w_i?

Counterexample:

- Smallest w_i: Capacity 1

  Item 1: weight 1, reward 0

  2: 100, 1

- Biggest r_i:

  Item 1: weight 100, reward 100

  Item 2: 1 2

Alg:

Sort items in decreasing r_i/w_i

total_weight $\leftarrow 0$

for $i = 1$ to n

  if total_weight $+ w_i \le C$:

    Take $p_i = 1$

  else take $p_i = \max((C - \text{total\_weight})/w_i, 0)$

Yeah but why is it correct?!

Correctness of greedy alg:

WRONG intuition (commonly held):

Make a series of locally opt choices to get a globally opt soln.

Issue:
- Partial solutions might not be evaluable
- Doesn't apply to "find me any soln" problems

Correct intuition: By induction

For every choice we made, we did not **** up.

Common framework to turn this into a proof:

Sometimes may not want to follow this template

Induction variable: # of choices made in alg / partial soln

Induction hypothesis:

For the partial soln constructed from the first i choices of the alg, there exists an optimal / feasible soln that is "compatible" w/ the partial soln.

Need to be specific about what "compatible" means.

Analyzing fractional knapsack alg:

Induction hypothesis for the loop:

Let $\rho_1, \ldots, \rho_i$ be the partial soln constructed at the end of i^{th} loop.

not "the"

Then exists an opt soln $\rho_1^{(opt)}, \ldots, \rho_n^{(opt)}$

s.t. $\rho_1 = \rho_1^{(opt)}, \ldots, \rho_i = \rho_i^{(opt)}$.

Base case $(i=0)$: trivially true.

Induction step:

Assume there is an opt soln

$\rho_1^{(opt)}, \ldots, \rho_n^{(opt)}$

so that $\rho_1 = \rho_1^{(opt)}, \ldots, \rho_i = \rho_i^{(opt)}$

Now consider the $i+1^{st}$ iter.

Either $\rho_{i+1} = 1$ or $\rho_{i+1} < 1$, either way

$\rho_{i+1} \geq \rho_{i+1}^{(opt)}$.

If $\rho_{i+1} = \rho_{i+1}^{(opt)}$ we're done.

If not then $\rho_{i+1} > \rho_{i+1}^{(opt)}$

There must be $(\rho_{i+1} - \rho_{i+1}^{(opt)}) \cdot w_i$ weight taken in items $i+2$ to n in $\rho^{(opt)}$, otherwise just take more items in $\rho^{(opt)}$ to give it at least as much reward.

Let's now construct a new opt soln $\rho'_1, \rho'_2, \ldots, \rho'_n$

s.t. $\rho'_1 = \rho_1, \ldots, \rho'_{i+1} = \rho_{i+1}$

Swapping argument. Common proof strategy for greedy alg.

To do so, start from $\rho^{(opt)}$ and <u>swap</u> $(\rho_{i+1} - \rho_{i+1}^{(opt)}) \cdot w_i$ weight from items $i+2$ to n to item $i+1$.

Since item $i+1$ has at least as much reward density as items $i+2$ to n, ρ' has reward no worse than $\rho^{(opt)}$.

So ρ' is an opt soln too.

By construction, $\rho'_1 = \rho_1, \ldots, \rho'_{i+1} = \rho_{i+1}$

Wrap up: At the end of n^{th} loop,

$\exists$ opt soln $\rho^{(opt)}$ s.t. $\rho_1 = \rho_1^{(opt)}, \ldots, \rho_n = \rho_n^{(opt)}$

So ρ is an opt soln

Choose your team tournament

Setting: 2 teams of n members each

Members have power levels
$A[1..n]$
$B[1..n]$ } not necessarily sorted

Each member competes with 1 member of other team

Team A chooses the matching

Can Team A win all matches?

Alg: (There is at least one other correct greedy alg)

Runtime? $O(n \log n)$
Sort A
Binary search tree for each B element

for $i = 1$ to n
find weakest A member n- weaker than $B[i]$.
if exists, match w/ $B[i]$ and remove this A member
if none exists, output "no".

Correctness:
(By induction.) If alg terminates and outputs yes, by construction we have a winning matching.

Need to show: If Team A can win, then we'll find a soln.

Induction hypothesis:

If Team A can win, then $\exists$ a winning matching that agrees w/ the matches made in first i loop iters.

Induction step:

Assume $\exists$ a winning matching M agree w/ the alg's first i match choices.

Now suppose alg matches $B[i+1]$ w/ $A[j]$.

If M matches $A[j]$ w/ $B[i+1]$, then done.

Otherwise, M matches $B[i+1]$ w/ some $A[j']$, and $A[j]$ w/ some $B[i']$ w/ $i' > i+1$.

Note that $A[j'] \geq A[j]$ by alg.

and since M is a winning matching for team A,

$$A[j'] \geq B[i+1]$$
$$A[j] \geq B[i']$$

So swap. Know $A[j] \geq B[i+1]$ since alg chose.

$$A[j'] \geq A[j] \geq B[i'].$$

Now matching M' still winning, and agrees w/ alg on first $i+1$ choices. //

Wrap up: If Team A can win, then after n^{th} loop will have found a winning matching :)