Announcements

OK to import data structures

- Hw2 is available on Gradescope (one coding question and 2 written question. **Due Friday Feb 6** only one late day.
- **May need to start panini press after the shift ends** (problem 2)

hw1 and quiz1 grades open on Gradescope: **regrade open for one week after grades open**

7^{30} - 9 → by Monday

Prelim 1: Thursday, Feb 12. fill out this form, if you have a conflict.

Covers hw1-2, sections week 1-2, lectures through this Wednesday this week.

Monday's class and section next week is review.

Other prelim info and practice questions posted on Canvas, solutions will be posted after sections (Tuesday)

Dynamic programming V: Knapsack

The problem: n items with weight w_i and value v_i

max weight allowed W, items $\{1, 2, .., n\}$

The problem select $S \subseteq \{1, .., n\}$ such that

$\sum_{i \in S} w_i \leq W$ → permitted to take S

maximum $\sum_{i \in S} v_i$

Example $W = 20$

#	w	v	v/w
1	15	20	
2	10	15	
3	8	14	
4	7	13	13/7

optimum $\{2, 3\}$

Ideas to solve Knapsack problems

greedy ideas

1. max value (among $w_i \leq W$) X see above
2. min weight X see above
3. min v_i/w_i value density X see above

Dynamic Programming: what are good subproblem for Knapsack?

Order items: $1 \dots, n$ & consider last decision

What to do with item n

proposed sub problem: $Opt(i)$ = optimum items $\{1, \dots, i\}$ max value possible

base case $Opt(0) = 0$

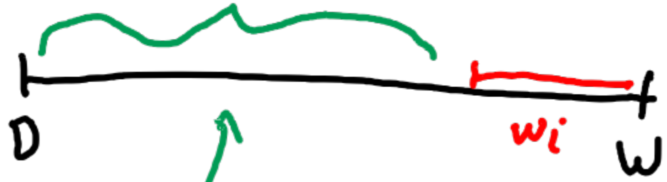
$Opt(1) = v_1$ (assuming $w_1 \leq W$)

need recurrence

$Opt(i) = \max\left(Opt(i-1),\ v_i + Opt\,?\right)$

item i not included

item i included



only $W - w_i$ weight allowed

Join by Web PollEv.com/evatardos772

Does the subproblem proposed work OK?

Opt(i) = max value for items {1, ..., i}

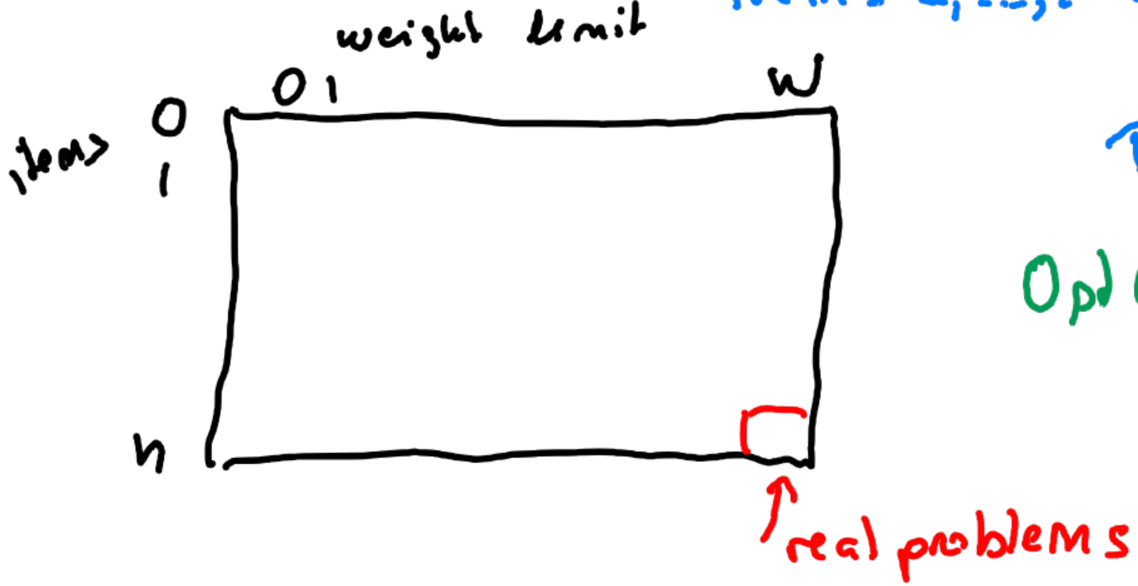
you choose ordering ??

A. Yes, can be made to work
B. No, this does not work
C. I don't know

The dynamic program: subproblems, base case and recurrence

Assume W integer
& w_i all integers

$Opt(i, w)$ = max value possible
items $1, \ldots, i$ & weight limit w



Recurrence:

$$Opt(i,w) = \max\left(Opt(i-1, w),\ v_i + Opt(i-1, w-w_i)\right)$$

item i
not included

item i included
assuming
$w_i \leq w$

The dynamic programming algorithm

$Opt(0, w) = 0$ all w

$Opt(i, 0) = 0$ all i

For $i = 1, \ldots, n$

  For $w = 1, \ldots, W$

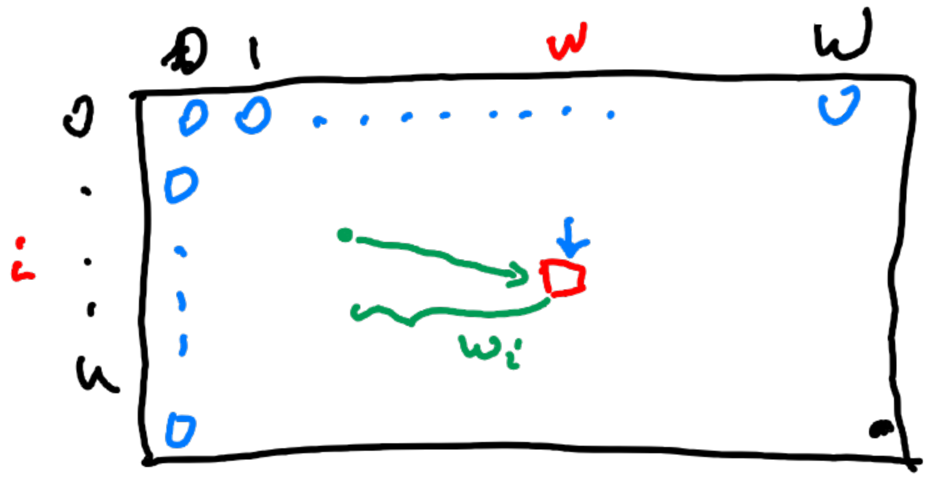
    if $w_i > w$ then

      $Opt(i, w) = Opt(i-1, w)$

    else

      $Opt(i, w) = \max\left(Opt(i-1, w),\ v_i + Opt(i-1, w-w_i)\right)$

Return $Opt(n, W)$



Running time: loops n & W $\Rightarrow O(nW)$

Correctness, running time, and extracting the solution

Correctness:

base ? obvious

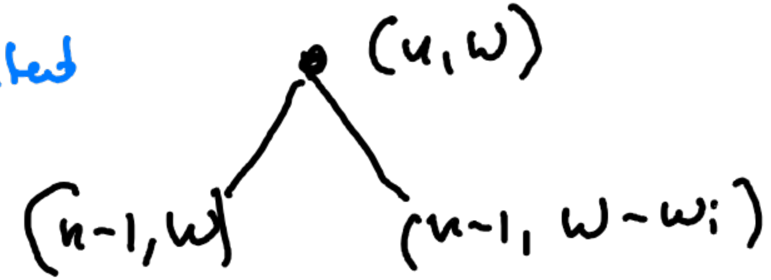
induction: explain recurrence

include English statement for what Opt is

$Opt(i, w)$ = max value possible
items $1, \dots, i$ of weight limit w

If time permits: benefit of a recursive solution with memoization

use hash table to store computed values



Recursive version

compute $Opt(n, W)$

if $n=0$ or $W=0$

return 0

else if $w_n > W$

check if $Opt(n-1, W)$ already computed

$Opt(n-1, W) = X$ & save it

return X

else

check if $Opt(n-1, W)$ & $Opt(n-1, W-w_i)$ computed

$\max(Opt(n-1, W), v_i + Opt(n-1, W-w_i)) = Y$ & save both

return Y