Knapsack problem:

Total capacity B
and n objects with:
integer weights $w_1, \ldots, w_n$
& integer values $v_1, \ldots, v_n$

Goal: Find subset S of objects that:
a) fits in the backpack
& b) maximizes the total value

In other words, find $S \subset \{1, \ldots, n\}$ where:

a) $\sum_{i \in S} w_i \leq B$

& b) maximizes $\sum_{i \in S} v_i$

Application: scheduling jobs.

Versions:

1) without repetition: one copy of each object
2) with repetition: unlimited supply of each object

First: one copy of each object

What about greedy approach?

Example: 4 objects: 1, 2, 3, 4

Values: 15, 10, 8, 1

weights: 15, 12, 10, 5

$B = 22$

Greedy: sort by $r_i = \frac{v_i}{w_i}$ so $r_1 > r_2 > r_3 > r_4$

greedy solution: objects 1 & 4 for total value = 16

optimal solution: objects 2 & 3 for total value = 18.

Dynamic Programming approach:

First, Define the subproblem

Initial attempt: Prefix

Let $K(j)$ = max value achievable using a subset of objects $1, \dots, j$

Second step: Express $K(j)$ in terms of $K(1), \dots, K(j-1)$.

Consider $K(j)$: given $K(j-1)$, can we add object j to the optimal for $K(j-1)$?

Need to know how much weight is available.

Want best solution for subset of $1, \dots, j-1$

with weight $\leq B$

& weight $\leq B - w_j$

Next round need $B, B - w_{j-1}, B - w_j, B - w_j - w_{j-1}$

So need to consider all possible weights.

Subproblem Definition:

For b & j where $0 \le b \le B$ & $0 \le j \le n$

let $K(b,j)$ = max value achievable using a subset of objects $1, \ldots, j$ & total weight $\le b$

Goal: Compute $K(B, n)$

Recurrence relation:

For $K(b,j)$:

either use object j:

then want best solution for subset of $1, \ldots, j-1$ with total weight $\le b - w_j$ (so j fits in)

or don't use object j:

then want best for $1, \ldots, j-1$ with total weight $\le b$.

Therefore,

if $w_j \le b$,

$$K(b,j) = \max\{v_j + K(b-w_j, j-1), K(b, j-1)\}$$

if $w_j > b$,

$$K(b,j) = K(b, j-1)$$

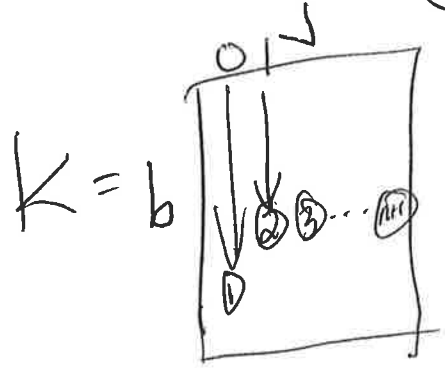
Base cases:

$$K(b,0) = 0$$

$$K(0,j) = 0$$

Recurrence for $K(b,j)$ uses $K(?, j-1)$

So fill table from $j=0 \to j=n$.



fill column by column

KnapsackNoRepeat$(B, w_1, \ldots, w_n, v_1, \ldots, v_n)$:

For $j = 0 \to n$, $K(0, j) = 0$
For $b = 0 \to B$, $K(b, 0) = 0$

For $j = 1 \to n$,
  For $b = 1 \to B$,
    if $w_j > b$,
      then $K(b, j) = K(b, j-1)$
      else $K(b, j) = \max\{v_j + K(b - w_j, j-1), K(b, j-1)\}$

Return$(K(B, n))$

<u>Running time:</u>

For loop of size $O(n)$ & inner loop of size $O(B)$, thus total running time of $O(nB)$.

<u>Now:</u> unlimited supply of each object

Try subproblem as before:

$K(b,j)$ = max value achievable using subset of objects $1,\ldots,j$ (possibly with repeats) & total weight $\leq b$.

if $w_j \leq b$

either include object j but then need to allow more copies of it so:

$$v_j + K(b-w_j, j)$$

← j instead of $j-1$

or don't include j so: $K(b, j-1)$.

Therefore,

if $w_j \le b$,

$$k(b,j) = \max\{k(b,j-1),\ v_j + k(b-w_j, j)\}$$

if $w_j > b$,

$$k(b,j) = k(b,j-1).$$

KnapsackRepeat$(B, w_1, \ldots, w_n, v_1, \ldots, v_n)$:

For $j = 0 \to n$, $k(0,j) = 0$

For $b = 0 \to B$, $k(b,0) = 0$

For $j = 1 \to n$

  For $b = 1 \to B$

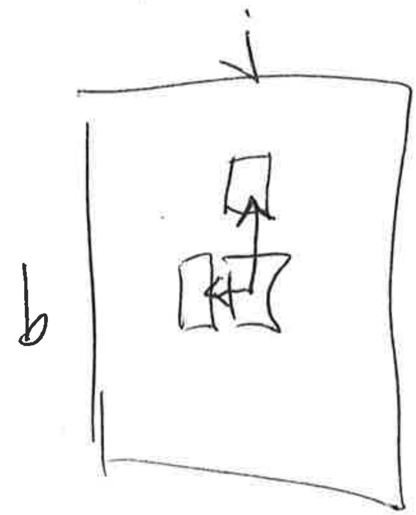
    if $w_j > b$

      then $k(b,j) = k(b,j-1)$

      else $k(b,j) = \max\{v_j + k(b-w_j, j), k(b,j-1)\}$

Return$(k(B,n))$

Filling the table:



$K(b,j)$ uses $\cancel{K(b,j-}$ $K(b-w_j, j)$

(earlier in same column)

or $K(b, j-1)$

(previous column)

Running time: $O(nB)$ as before.

Alternative:

$K(b)$ = max value achievable using total weight $\leq b$
& all objects $1, \ldots, n$ allowed.

Recurrence:

Try all possibilities for last object ℓ
need that $w_\ell \leq b$

Hence:

$$K(b) = \max_{\ell} \{ K(b - w_\ell) + v_\ell : 1 \leq \ell \leq n, w_\ell \leq b \}$$

K is one-dimensional but each entry
takes $O(n)$ time so
$O(nB)$ total time as before.