KNAPSACK

input: A triple $I = (v, w, L)$ where $v = (v_1, v_2, \ldots, v_n)$ is a sequence of non-negative *values*, $w = (w_1, w_2, \ldots, w_n)$ is a sequence of positive integer *weights*, and w is a non-negative integer *weight limit*. We call each integer $i \in \{1, 2, \ldots, n\}$ an *item*, so v_i is the value of item i and w_i is its weight.

output: The maximum value of any subset $S \subseteq \{1, 2, \ldots, n\}$ of items having total weight at most w. (The value of a subset S is $\sum_{i \in S} v_i$. The total weight of S is $\sum_{i \in S} w_i$.)

Here is the algorithm. Fix any input instance $I = (v, w, L)$.

subproblems. For each $i \in \{0, 1, 2, \ldots, n\}$ and $\ell \in \{0, 1, 2, \ldots, L\}$, define $V[i, \ell]$ to be the maximum value achievable using only items $\{1, 2, \ldots, i\}$ and total weight at most ℓ. The given instance I has solution $V[n, L]$.

recurrence. If $i = 0$ or $\ell = 0$, then$V[i, 0] = 0$ (using here that the weights are positive). For $i, \ell > 0$, if $w_i > \ell$, then $V[i, \ell] = V[i - 1, \ell]$. Otherwise

$$V[i, \ell] = \max \begin{cases} V[i-1, \ell] \\ V[i-1, \ell - w_i] + v_i. \end{cases}$$

The intuition is that an optimal set S for $V[i, \ell]$ either includes i or it doesn't. If it doesn't include i, then it can be any optimal solution for $V[i - 1, \ell]$. Otherwise (it includes i), it can be the union of $\{i\}$ and any optimal solution for $V[i - 1, \ell - w_i]$.

time. There are $O(nL)$ subproblems. For each, the right-hand side of the recurrence can be evaluated in constant time. So the total time to compute $V[n, L]$ is $O(nL)$.

correctness. To prove correctness, we would give a more careful proof tha the recurrence relation is correct. Correctness of the algorithm would follow by induction (on i).

This algorithm is equivalent to running the LONGEST PATH algorithm on an appropriate DAG.

External resources on KNAPSACK

- CLRS Chapter 16, Exercise 16.2–2
- Dasgupta et al. Chapter 6.
- Kleinberg & Tardos Chapter 6.
- MIT lecture video
  - https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-006-introduction-to-algorithms-fall-2011/lecture-videos/lecture-21-dp-iii-parenthesization-edit-distance-knapsack