

Dynamic Programming Knapsack Algorithm

Input: Nonnegative integers $n, c_1, \dots, c_n, w_1, \dots, w_n$ and W

Output: A subset $S \subseteq \{1, 2, \dots, n\}$ such that

$$\sum_{j \in S} w_j \leq W \text{ and } \sum_{j \in S} c_j \text{ is maximum}$$

(1) let G be any upper bound on the value of the optimum solution, e.g. $G := \sum_{j=1}^n c_j$

(2) let $x(0,0) := 0$ and $x(0,k) := \infty$ for $k = 1, 2, \dots, G$

(3) For $j := 1$ to n do

For $k := 0$ to G do

Set $s(j,k) := 0$ and $x(j,k) := x(j-1,k)$.

For $k := c_j$ to G do

If $x(j-1, k-c_j) + w_j \leq \min\{W, x(j,k)\}$ then:

Set $x(j,k) := x(j-1, k-c_j) + w_j$ and $s(j,k) = 1$.

(4) let $k := \max\{i \in \{0, \dots, G\} : x(n,i) < \infty\}$. Set $S = \emptyset$.

For $j := n$ down to 1 do

If $s(j,k) = 1$ then set $S := S \cup \{j\}$ and $k := k - c_j$.

Meaning of $x(j,k)$: the minimum total weight of a subset $S \subseteq \{1, 2, \dots, j\}$ with $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} c_i = k$.

$x(j,k)$ are computed recursively:

$$x(j,k) = \begin{cases} x(j-1, k-c_j) + w_j & \text{if } c_j \leq k \text{ and } x(j-1, k-c_j) + w_j \leq \min\{W, x(j,k)\} \\ x(j-1, k) & \text{otherwise} \end{cases}$$

Meaning of $s(j,k)$: $s(j,k)$ equals 1 if the upper case applies and 0 otherwise.