Lecture 4.3

dinsdag 12 september 2023 13:23

We have the following theorem:

> Suppose all n item values in a knapsack instance are integers. Then we can compute an optimal solution in $O(n\, V_{tot})$ time, where V_{tot} is the total value of all items.

We define $weight(S) = \sum_{x_i \in S} weight(x_i)$ and $value(S) = \sum_{x_i \in S} value(x_i)$, i.e. the weight/value of a set is the total weight/value of the items in that set. Furthermore, we define S_i to be a set consisting of the first i items in X, i.e. $S_i = \{x_1, \ldots, x_i\}$. Finally, we let $S_0 = \emptyset$.

We are using a dynamic-programming strategy. As subproblems, we define (for $0 \leq i \leq n$ and $0 \leq j \leq V_{tot}$)
$A[i,j]$ = minimum weight of any subset $S \subseteq S_i$ whose total value is exactly j. We let $A[i,j] = \infty$ if no such subset S exists.
The table is useful because the maximum value of any subset $S \subseteq X$ with $weight(S) \leq W$ is equal to the largest j such that $A[n,j] \leq W$.

To solve this problem by dynamic programming, we first give a recursive formula for $A[i,j]$. Then, we compute all $A[i,j]$ by filling in a table. Finally, we find the largest j such that $A[n,j] \leq W$.

$$A[i,j] = \begin{cases} 0 \quad \text{(take the empty set)} & \text{if } j = 0 \\ \infty \quad \text{(from empty set, obtaining value} > 0 \text{ (this also value} = j\text{) is impossible)} & \text{if } j > 0 \text{ and } i = 0 \\ A[i-1, j] & \text{if } j > 0 \text{ and } i > 0 \text{ and value}(x_i) > j \\ \min\big(A[i-1,j],\ A[i-1, j - value(x_i)] + weight(x_i)\big) & \text{otherwise} \end{cases}$$

min (min weight of any S ... and that includes x_i → $A[i-1, j - value(x_i)] + weight(x_i)$; note: this case is not allowed when $value(x_i) > j$

min weight of any S ... and that does not include x_i) → $A[i-1, j]$