Dynamic Programming

Knapsack Problem

Problemset

- given n items of known weights w1, . . . , wn and values v1, . . . , vn and a knapsack of capacity W,
- find the most valuable subset of the items that fit into the knapsack.
- assume that all the weights and the knapsack capacity are positive integers; the item values do not have to be integers.
- To design a dynamic programming algorithm, we need to derive a recurrence relation that expresses a solution to an instance of the knapsack problem in terms of solutions to its smaller subinstances.

Main Idea

- Let us consider an instance defined by the first i items, $1 \leq i \leq n$, with weights $w1, \ldots, wi$, values $v1, \ldots, vi$, and knapsack capacity j, $1 \leq j \leq W$.
- Let F(i, j) be the value of an optimal solution to this instance, i.e., the value of the most valuable subset of the first i items that fit into the knapsack of capacity j.
- We can divide all the subsets of the first i items that fit the knapsack of capacity j into two categories: those that do not include the ith item and those that do

Main Idea

1. Among the subsets that do not include the ith item, the value of an optimal subset is, by definition, $F(i - 1, j)$.
2. Among the subsets that do include the ith item (hence, $j - w_i \geq 0$), an optimal subset is made up of this item and an optimal subset of the first $i - 1$ items that fits into the knapsack of capacity $j - w_i$. The value of such an optimal subset is $v_i + F(i - 1, j - w_i)$.

recurrence:

$$F(i, j) = \begin{cases} \max\{F(i-1, j),\ v_i + F(i-1, j-w_i)\} & \text{if } j - w_i \geq 0, \\ F(i-1, j) & \text{if } j - w_i < 0. \end{cases}$$

$$F(0, j) = 0 \text{ for } j \geq 0 \quad \text{and} \quad F(i, 0) = 0 \text{ for } i \geq 0.$$

Solution table

		0	$j-w_i$	j	W
	0	0	0	0	0
	$i-1$	0	$F(i-1, j-w_i)$	$F(i-1, j)$	
w_i, v_i	i	0		$F(i, j)$	
	n	0			goal

Example

item	weight	value
1	2	\$12
2	1	\$10
3	3	\$20
4	2	\$15

capacity $W = 5$.