0-1 knapsack problem

Problem statement:

- Given \( n \) items \( \{1, 2, \ldots, n\} \)
- Item \( i \) is worth \( v_i \), and weight \( w_i \)
- Find a most valuable subset of items with total weight \( \leq W \)

*Rule: have to either take an item or not take it ("0-1 Knapsack") – can't take part of it.*

Example:

- Given

<table>
<thead>
<tr>
<th>( i )</th>
<th>( v_i )</th>
<th>( w_i )</th>
<th>( v_i/w_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Total weight \( W = 5 \)

- Find a most valuable subset of items with total weight \( \leq W \)
0-1 knapsack problem

Problem statement, \textit{mathematically} – version 1:

\textit{Find a subset } \mathcal{S} \subseteq \{1, 2, \ldots, n\} \text{ such that}

\text{maximize } \sum_{i \in \mathcal{S}} v_i

\text{subject to } \sum_{i \in \mathcal{S}} w_i \leq W
0-1 knapsack problem

Problem statement, *mathematically* – version 2:

Let $x = (x_1, x_2, \ldots, x_n)$, and

$$x_i = \begin{cases} 
1 & \text{$i$-th item is in the knapsack} \\
0 & \text{$i$-th item is not in the knapsack}
\end{cases}$$

Then the knapsack problem is

$$\max_{x_i \in \{0, 1\}} \sum_{i=1}^{n} v_i x_i$$

$$\text{s.t. } \sum_{i=1}^{n} w_i x_i \leq W$$
0-1 knapsack problem

The brute-force algorithm

- $2^n$ feasible solutions
- Total cost = $O(n \cdot 2^n)$
0-1 knapsack problem

Three possible greedy strategies:

1. Greedy by highest value $v_i$

2. Greedy by least weight $w_i$

3. Greedy by largest value density $\frac{v_i}{w_i}$

All three approaches generate feasible solutions. However, we cannot guarantee that any of them will always generate an optimal solution!
0-1 knapsack problem

Example:

<table>
<thead>
<tr>
<th>i</th>
<th>(v_i)</th>
<th>(w_i)</th>
<th>(v_i/w_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Total weight \(W = 5\)

Greedy by value density \(v_i/w_i\):

▶ take items 1 and 2.
▶ value = 16, weight = 3
▶ Leftover capacity = 2

Optimal solution

▶ take items 2 and 3.
▶ value = 22, weight = 5
▶ no leftover capacity

Question: how about greedy by highest value? by least weight?
0-1 knapsack problem

Extra example: Given the following six items with $W = 100$:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$v_i$</th>
<th>$w_i$</th>
<th>$v_i/w_i$</th>
<th>Greedy by</th>
<th>optimal solution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>value</td>
<td>weight</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>100</td>
<td>0.4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>35</td>
<td>50</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>18</td>
<td>45</td>
<td>0.4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>20</td>
<td>0.2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>5</td>
<td>0.4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Total value</td>
<td></td>
<td></td>
<td>40</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>Total weight</td>
<td></td>
<td></td>
<td>100</td>
<td>80</td>
</tr>
</tbody>
</table>

All three greedy approaches generate feasible solutions, but none of them generate the optimal solution.