Dynamic Programming

0-1 knapsack problem

Subsetsum problem

- A single resource
- Requests $\{1..n\}$ each take time w_i to process
- Can schedule jobs at any time between 0 to W.
- Objective: to schedule jobs such that we maximize the resource's utilization

$OPT(i)$ = value of the opt. sol. for requests $1..i$.

$$\begin{cases} \text{if } n \notin O \text{ then } OPT(n) = OPT(n-1) \\ \text{if } n \in O \text{ then } OPT(n) = w_n + OPT(n-1) \end{cases}$$

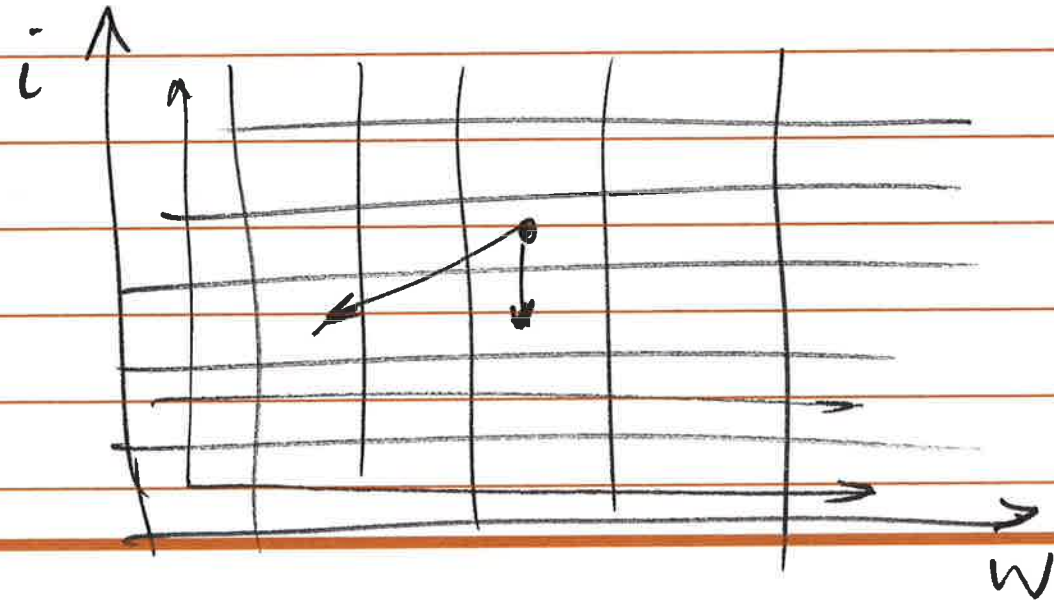
$OPT(i, w)$ = value of the opt. sol.
using a subset of the items $\{1..i\}$
with max. allowed weight w.

if $n \notin O$ then $OPT(n, w) = OPT(n-1, w)$

if $n \in O$ then $OPT(n, w) = w_n + OPT(n-1, w - w_n)$

① if $w < w_i$ then $OPT(i, w) = OPT(i-1, w)$

otherwise $OPT(i, w) = \max(OPT(i-1, w),$
$w_i + OPT(i-1, w - w_i))$



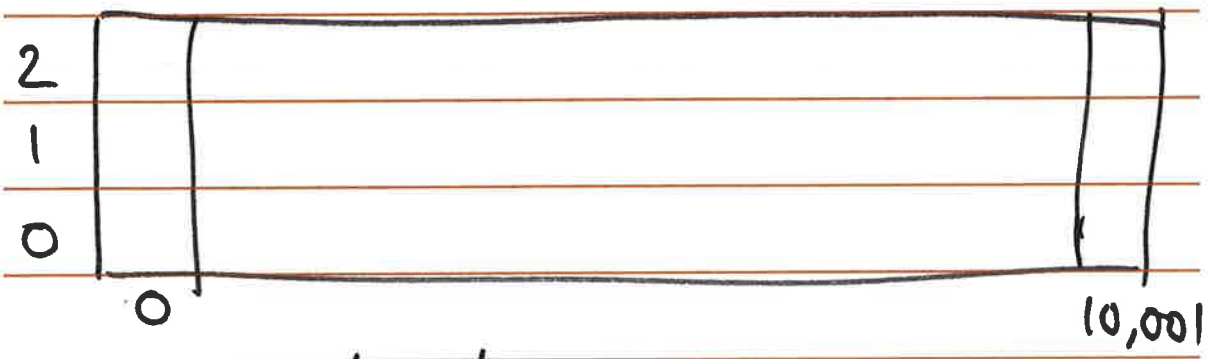
```
Subset_sum (n, W)
    array M[0, w] = 0  for each
                        w = 0 to W
for i = 1 to n
    for w = 0 to W
        use recurrence formula (1)
            to compute M[i, w]
    end for
end for
return M[n, W]
```

Takes $O(nW)$ ← pseudo polynomial O complexity

ex. $w_1 = 10,000$

$w_2 = 9,999$

$W = 10,001$



~ 30K elements

0-1 knapsack problem

each request has weight w_i & value v_i

if $n \notin O$, then $OPT(n, w) = OPT(n-1, w)$

if $n \in O$, then $OPT(n, w) = v_n + OPT(n-1, w-w_n)$