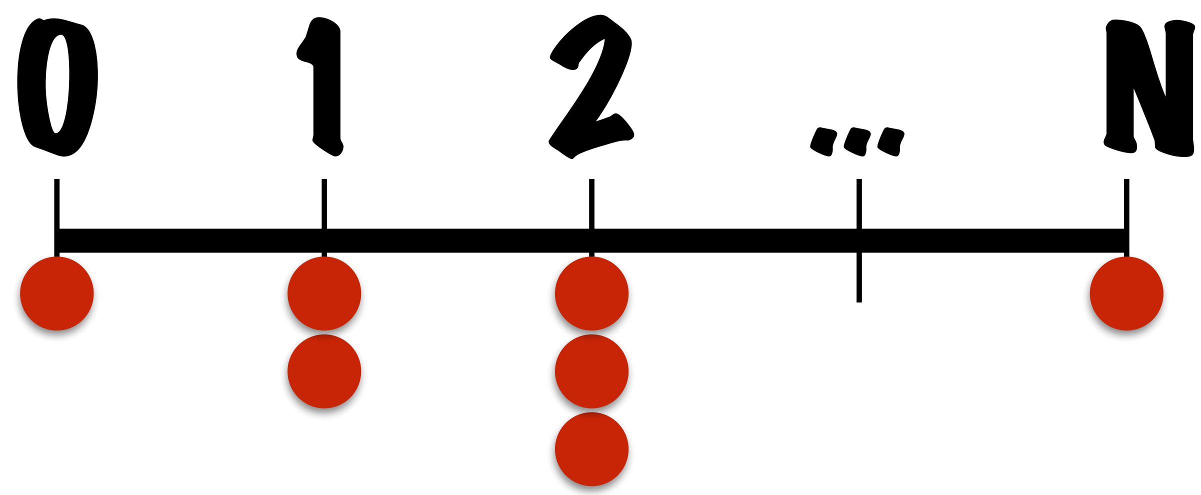
Knapsack and rounding

General case: algorithm

We already have an algorithm when values are small integers



Idea: modify input so that values are small integers

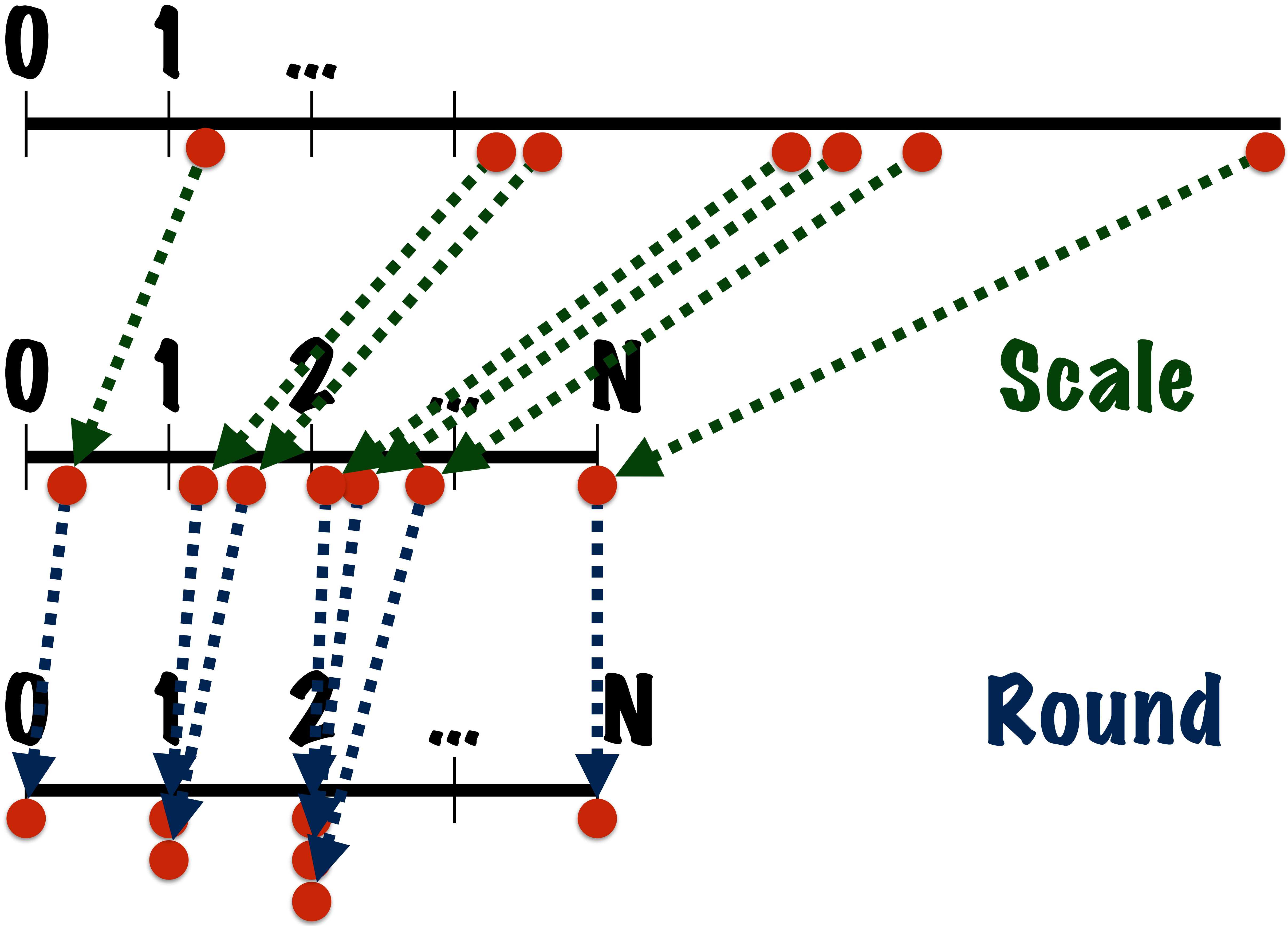
Q : How to modify input so that values are small integers?

A : Scale and round!

Discard items that don't fit

1. Multiply each value by something so that values are small
2. Round values to integers

Dynamic program for the scaled and rounded problem


0
1
...
0
1
2
...
N
Scale
0
1
2
...
N
Round

Effect of scaling:
Max scaled value = N

Multiply by $\alpha = \dfrac{\mathrm{N}}{\max \mathbf{v_i}}$

How do we pick N?

- small enough that the runtime is polynomial
- large enough that the resulting set of items has value close to the optimum

Algorithm Scale

1. Discard items not fitting
2. Let N=100 n
3. Let $v'_i \leftarrow \left\lfloor v_i \times \frac{N}{\max_j v_j} \right\rfloor$
4. Apply DP to $B, (s_i, v'_i)_i$
5. Output corresponding items

Runtime

$$\begin{aligned}
&\mathbf{For\ \ v = 0 \ldots nN : A[1, v] \leftarrow B + 1}\\
&\mathbf{A[1, v_1] \leftarrow s_1, A[1, 0] \leftarrow 0}\\
&\mathbf{For\ \ i = 2 \cdots n,}\\
&\quad \mathbf{For\ \ v = 0 \ldots v_i - 1 : A[i, v] \leftarrow A[i-1, v]}\\
&\quad \mathbf{For\ \ v = v_i, v_i + 1, \ldots, nN :}\\
&\qquad \mathbf{A[i, v] \leftarrow} \min(\mathbf{A[i-1, v], A[i-1, v - v_i] + s_i})\\
&\mathbf{Output\ \ } \max\{\mathbf{v : A[n, v] \le B}\}
\end{aligned}$$

$$\mathbf{N = 100 \times n \implies n^2 N = 100 \times n^3}$$

Knapsack and rounding