

# Algorithmische Graphentheorie

Sommersemester 2026

13. Vorlesung, Teil B

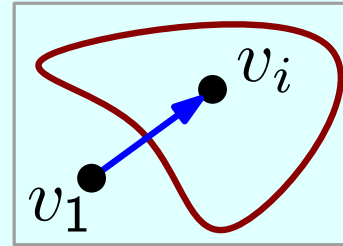
## Dynamische Programmierung – Das Rucksackproblem

# Erinnerung: DP von Bellman & Held-Karp

BellmanHeldKarp(Knotenmenge  $V$ , Abstände  $c: V \times V \rightarrow \mathbb{R}_{\geq 0}$ )

**for**  $i = 2$  **to**  $n$  **do**

└  $\text{OPT}[\{v_i\}, v_i] = c(v_1, v_i)$

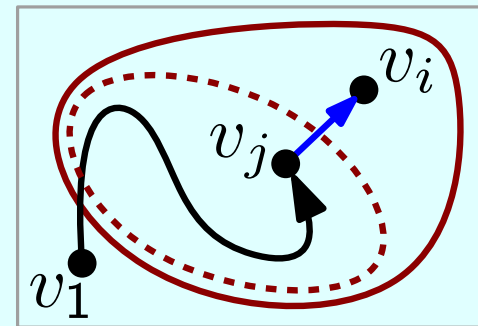


**for**  $j = 2$  **to**  $n - 1$  **do**

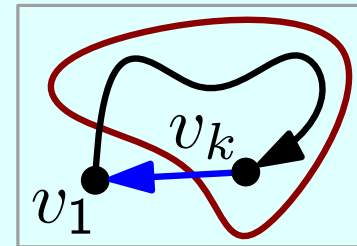
└ **foreach**  $W \subseteq \{v_2, \dots, v_n\}$  mit  $|W| = j$  **do**

└ **foreach**  $v_i \in W$  **do**

└  $\text{OPT}[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} \text{OPT}[W \setminus \{v_i\}, v_j] + c(v_j, v_i)$



**return**  $\min_{k \neq 1} \text{OPT}[V \setminus \{v_1\}, v_k] + c(v_k, v_1)$

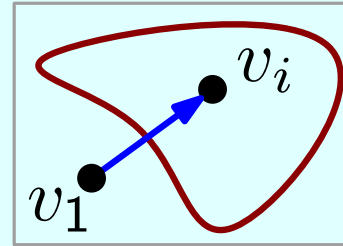


# Erinnerung: DP von Bellman & Held-Karp

BellmanHeldKarp(Knotenmenge  $V$ , Abstände  $c: V \times V \rightarrow \mathbb{R}_{\geq 0}$ )

**for**  $i = 2$  **to**  $n$  **do**

└  $\text{OPT}[\{v_i\}, v_i] = c(v_1, v_i)$

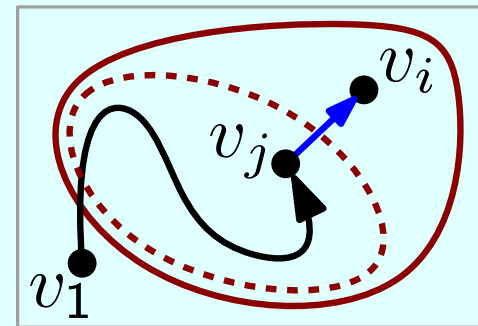


**for**  $j = 2$  **to**  $n - 1$  **do**

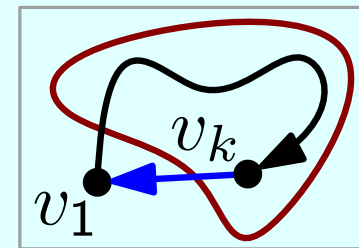
└ **foreach**  $W \subseteq \{v_2, \dots, v_n\}$  mit  $|W| = j$  **do**

└ **foreach**  $v_i \in W$  **do**

└  $\text{OPT}[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} \text{OPT}[W \setminus \{v_i\}, v_j] + c(v_j, v_i)$



**return**  $\min_{k \neq 1} \text{OPT}[V \setminus \{v_1\}, v_k] + c(v_k, v_1)$



Wie viele Paare  $(W, v_i)$  mit  $v_i \in W$  gibt's?

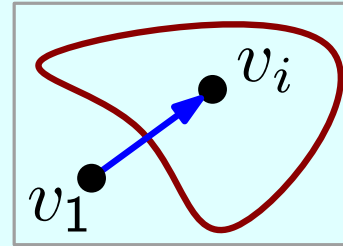
**Speicher:** (= Größe der DP-Tabelle)

# Erinnerung: DP von Bellman & Held-Karp

BellmanHeldKarp(Knotenmenge  $V$ , Abstände  $c: V \times V \rightarrow \mathbb{R}_{\geq 0}$ )

**for**  $i = 2$  **to**  $n$  **do**

└  $\text{OPT}[\{v_i\}, v_i] = c(v_1, v_i)$

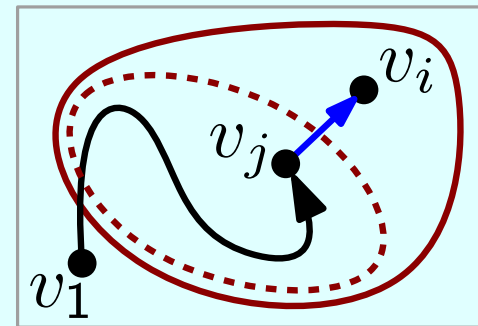


**for**  $j = 2$  **to**  $n - 1$  **do**

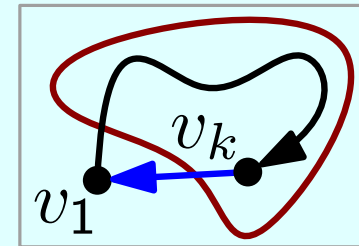
└ **foreach**  $W \subseteq \{v_2, \dots, v_n\}$  mit  $|W| = j$  **do**

└ **foreach**  $v_i \in W$  **do**

└  $\text{OPT}[W, v_i] = \min_{v_j \in W \setminus \{v_i\}} \text{OPT}[W \setminus \{v_i\}, v_j] + c(v_j, v_i)$



**return**  $\min_{k \neq 1} \text{OPT}[V \setminus \{v_1\}, v_k] + c(v_k, v_1)$



Wie viele Paare  $(W, v_i)$  mit  $v_i \in W$  gibt's?  $\leq n \cdot 2^{n-1}$

**Speicher:** (= Größe der DP-Tabelle)

$O(n \cdot 2^n)$

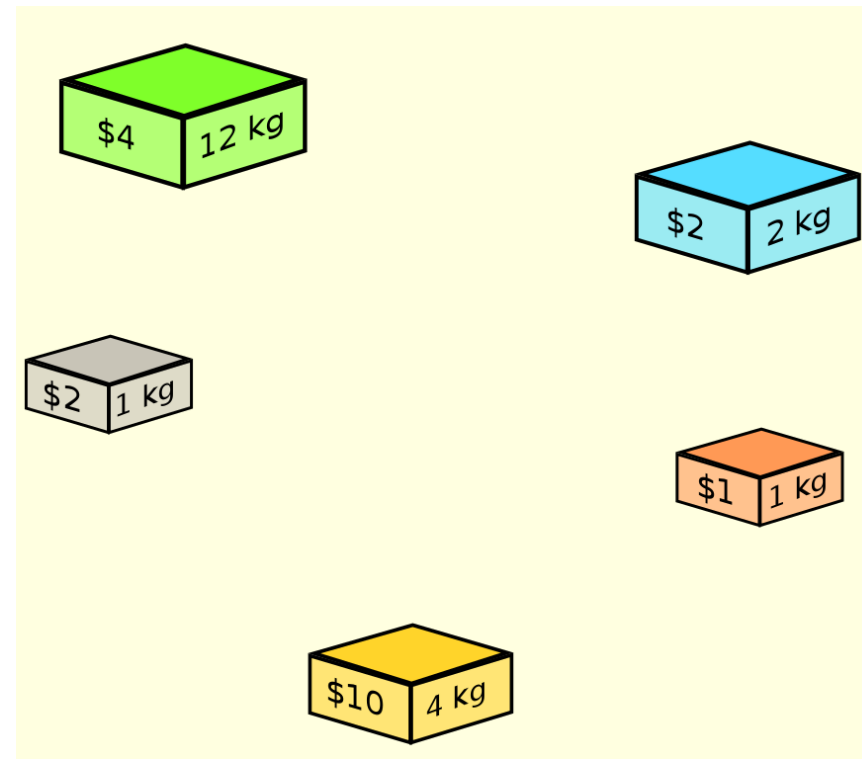
# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

- ein Gewicht  $w_i \in \mathbb{Q}^+$ ,
- ein Wert  $v_i \in \mathbb{Q}^+$ ,

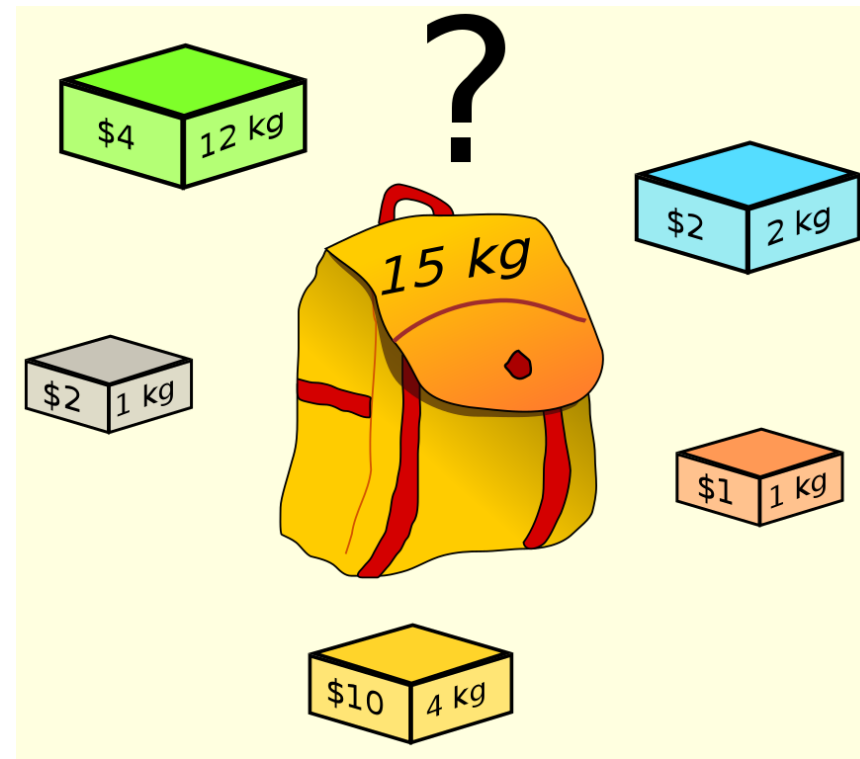


# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

- ein Gewicht  $w_i \in \mathbb{Q}^+$ ,
- ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .



# Das Rucksack-Problem

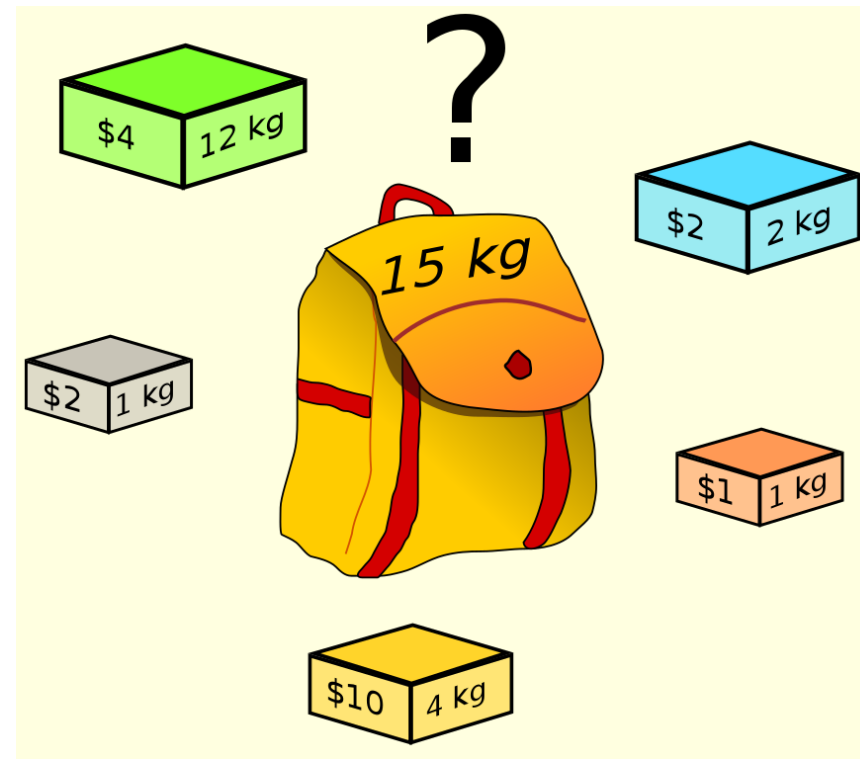
**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

**Gesucht:** Teilmenge  $R$  von  $U$ , so dass





# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

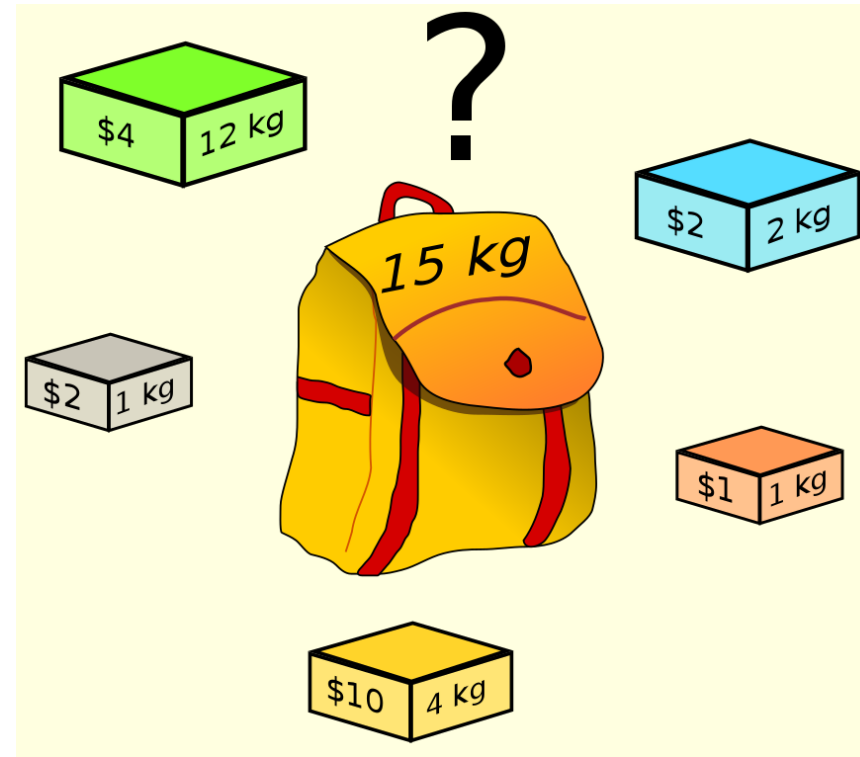
– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:



# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

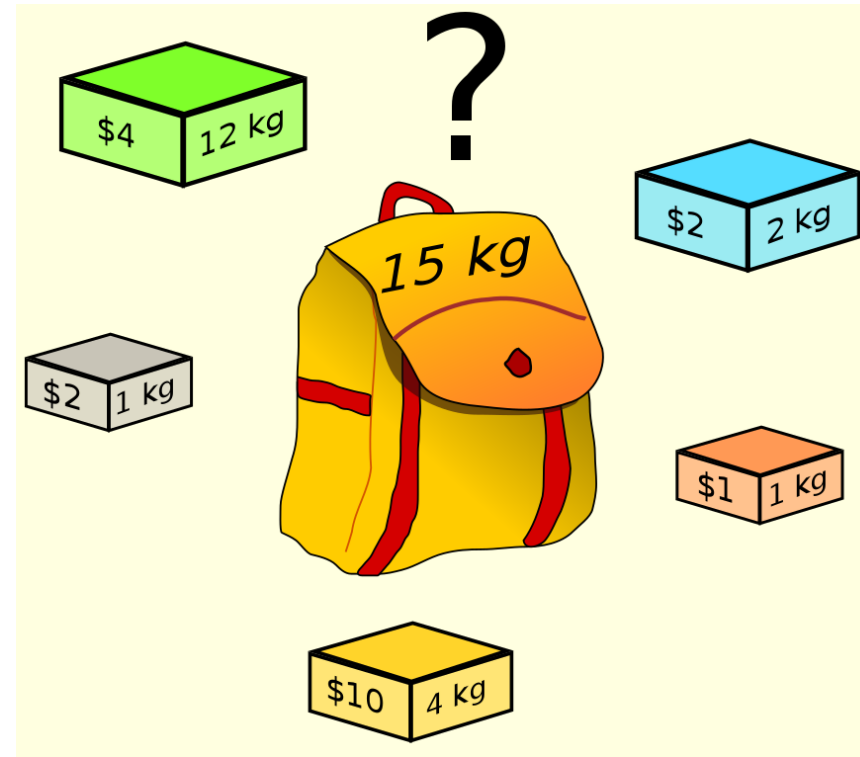
– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:

$$\sum_{i \in R} w_i \leq W$$



# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

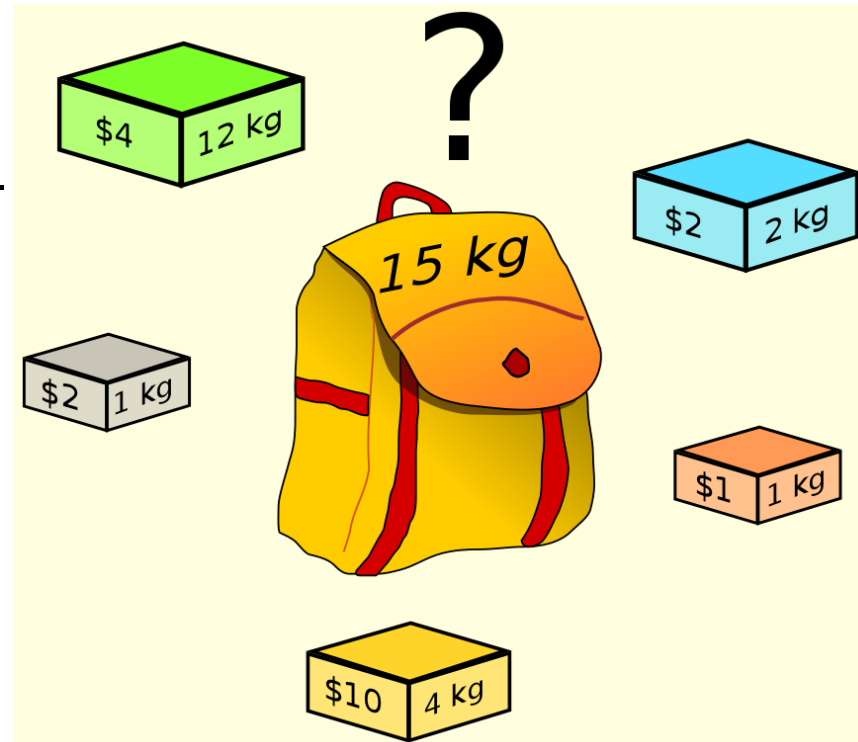
außerdem eine Gewichtsschranke  $W > 0$ .

**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:

$$\sum_{i \in R} w_i \leq W$$

– der Gesamtwert der Objekte in  $R$  maximal ist:



# Das Rucksack-Problem

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

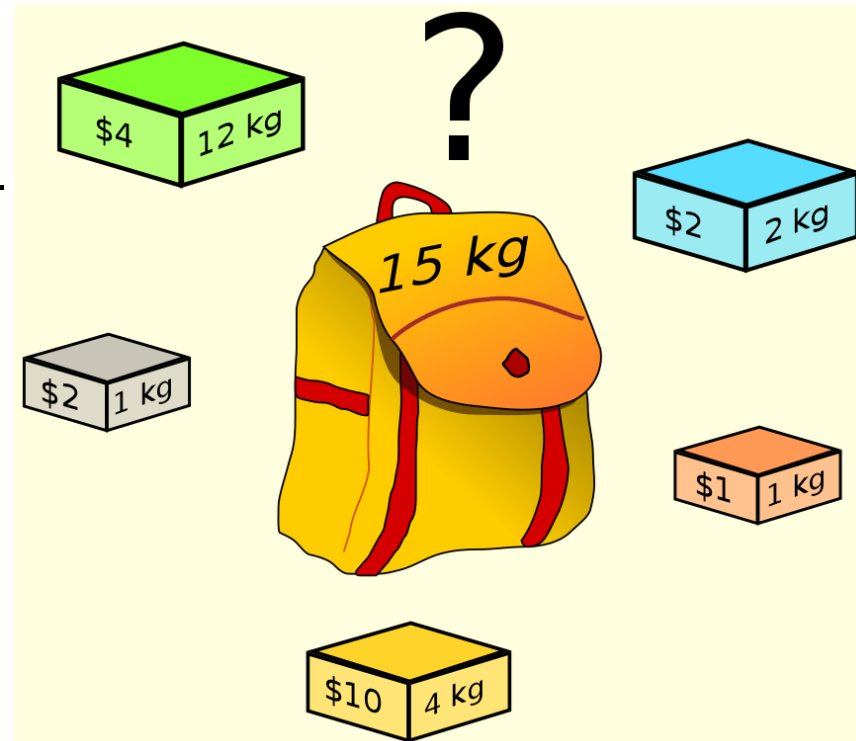
**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:

$$\sum_{i \in R} w_i \leq W$$

– der Gesamtwert der Objekte in  $R$  maximal ist:

$$\sum_{i \in R} v_i \text{ max!}$$



# Bad News

**Satz.** Das Rucksack-Problem ist NP-schwer.

# Bad News

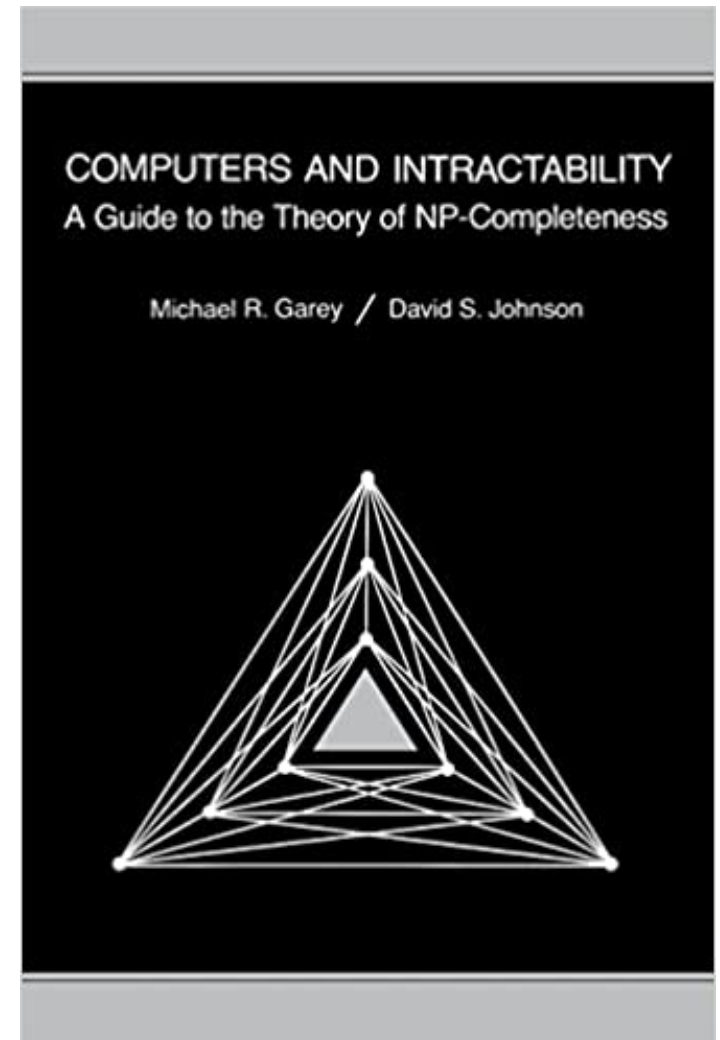
**Satz.** Das Rucksack-Problem ist NP-schwer.

*Beweis.*

# Bad News

**Satz.** Das Rucksack-Problem ist NP-schwer.

*Beweis.* Durch Reduktion vom Problem ...



[siehe Buch von Garey & Johnson]

Was tun?



Was tun? – Mach das Problem leichter!

# Was tun? – Mach das Problem leichter!

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \mathbb{Q}^+$ ,

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

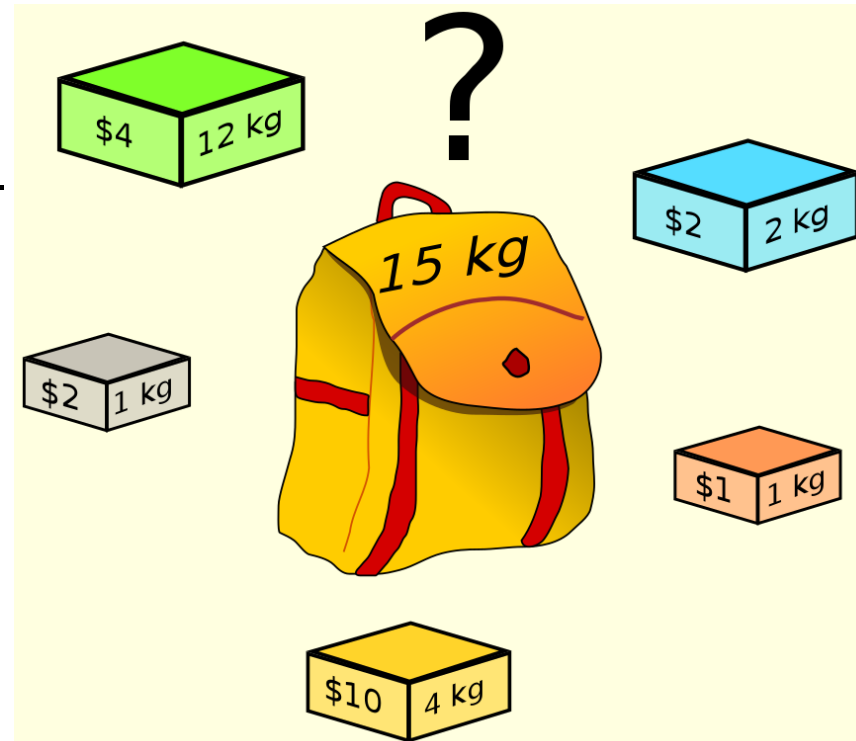
**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:

$$\sum_{i \in R} w_i \leq W$$

– der Gesamtwert der Objekte in  $R$  maximal ist:

$$\sum_{i \in R} v_i \text{ max!}$$



# Was tun? – Mach das Problem leichter!

**Gegeben:** Menge  $U$  von Objekten und für jedes  $i \in U$ :

– ein Gewicht  $w_i \in \cancel{\mathbb{Q}^+}$ ,  $\mathbb{N}$

– ein Wert  $v_i \in \mathbb{Q}^+$ ,

außerdem eine Gewichtsschranke  $W > 0$ .

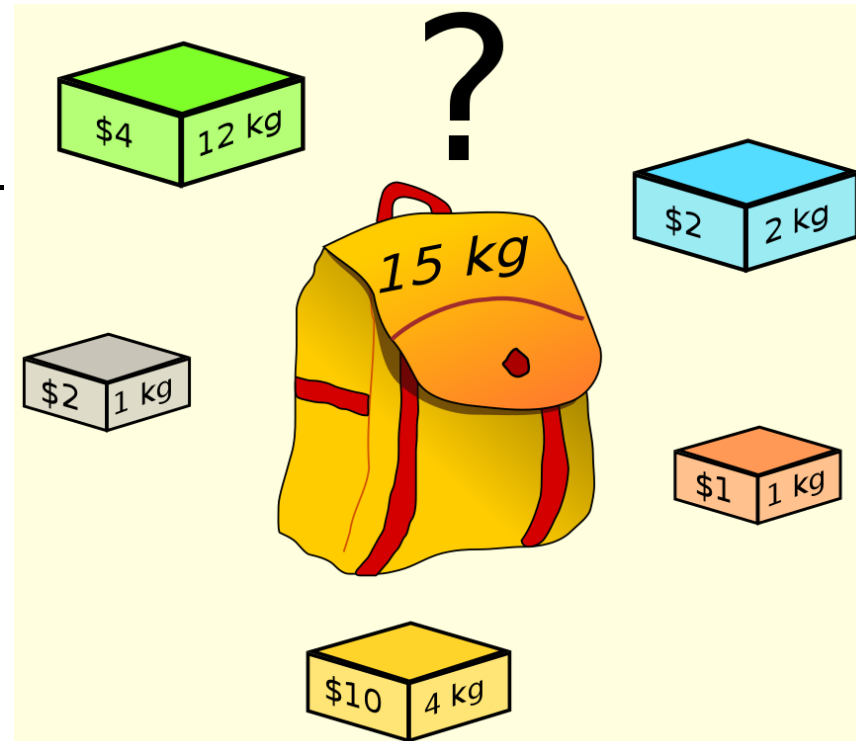
**Gesucht:** Teilmenge  $R$  von  $U$ , so dass

– das Gewicht der Objekte in  $R$  höchstens  $W$  ist:

$$\sum_{i \in R} w_i \leq W$$

– der Gesamtwert der Objekte in  $R$  maximal ist:

$$\sum_{i \in R} v_i \text{ max!}$$



# Rekursive Definition

Sei  $n$  die Anzahl der Objekte.

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei



# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(\quad)$ .

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

**Berechnung.**

$$f(1, w) =$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 \\ \end{cases}$$

**Berechnung.**

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ \end{cases}$$

**Berechnung.**

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ \text{sonst.} \end{cases}$$

**Berechnung.**

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**



# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**

und für  $2 \leq i \leq n$ :

$$f(i, w) =$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} & \text{if } w_i \leq w, \\ & \text{else.} \end{cases}$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} v_i + f(i-1, w-w_i) & \text{if } w_i \leq w, \\ f(i-1, w) & \text{else.} \end{cases}$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} v_i + f(i-1, w - w_i) & \text{if } w_i \leq w, \\ f(i-1, w) & \text{else.} \end{cases}$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} \max\{v_i + f(i-1, w - w_i), f(i-1, w)\} & \text{if } w_i \leq w, \\ f(i-1, w) & \text{else.} \end{cases}$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

**Berechnung.**  
 – Laufzeit?

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} \max\{v_i + f(i-1, w - w_i), f(i-1, w)\} & \text{if } w_i \leq w, \\ f(i-1, w) & \text{else.} \end{cases}$$

# Rekursive Definition

Sei  $n$  die Anzahl der Objekte. O.B.d.A. sei  $U = \{1, 2, \dots, n\}$ .

Sei  $w \in \{1, \dots, W\}$  und  $1 \leq i \leq n$ .

**Idee:** Def.  $f(i, w) :=$  maximaler Wert von  $\sum_{k \in R} v_k$ , wobei  
 $R \subseteq \{1, 2, \dots, i\}$  und  $\sum_{k \in R} w_k \leq w$ .

Dann suchen wir  $f(n, W)$ .

$$f(1, w) = \begin{cases} v_1 & \text{falls } w_1 \leq w, \\ 0 & \text{sonst.} \end{cases}$$

und für  $2 \leq i \leq n$ :

$$f(i, w) = \begin{cases} \max\{v_i + f(i-1, w - w_i), f(i-1, w)\} & \text{if } w_i \leq w, \\ f(i-1, w) & \text{else.} \end{cases}$$

**Berechnung.**

– Laufzeit?

– Speicher?

# Dynamisches Programm

- Probieren Sie selbst, die Einträge der DP-Tabelle  $f[i, w]$  in der richtigen Reihenfolge zu befüllen!



# Dynamisches Programm

- Probieren Sie selbst, die Einträge der DP-Tabelle  $f[i, w]$  in der richtigen Reihenfolge zu befüllen!
- Wenn Sie  $f(n, W)$ , also den maximalen Wert, berechnet haben – wie kriegen Sie dann heraus, welche Menge von Objekten in den Rucksack passt und dabei den maximalen Wert hat?

# Ausblick

Master-Vorlesungen: – „Advanced Algorithms“ und  
– „Approximationsalgorithmen“

Spannendes Video (30') von Thomas van Dijk zu praktischen Aspekten der Implementierung eines Algorithmus für das Rucksack-Problem:

<https://go.uniwue.de/algoprac4>