

# Optimierung

Auf der Suche nach dem Besten

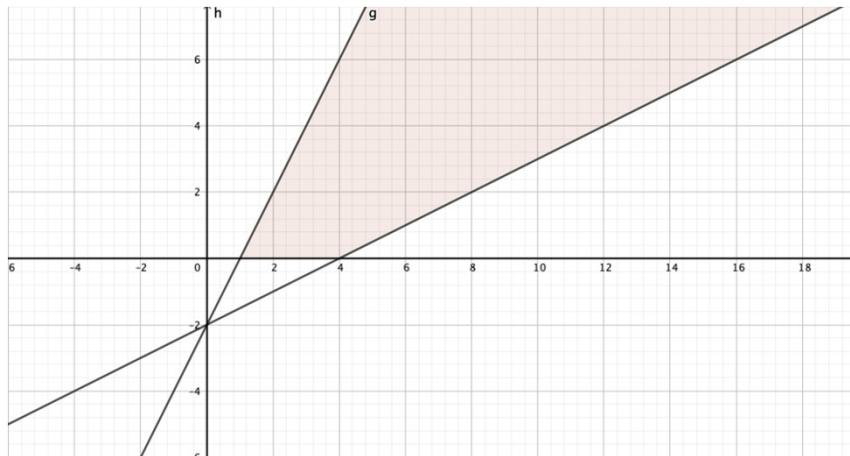
# Lineare Programmierung

$$f(x, y) = x + y + 2$$

$$x - 2y \leq 4$$

$$2x - y \geq 2$$

$$x, y \geq 0$$

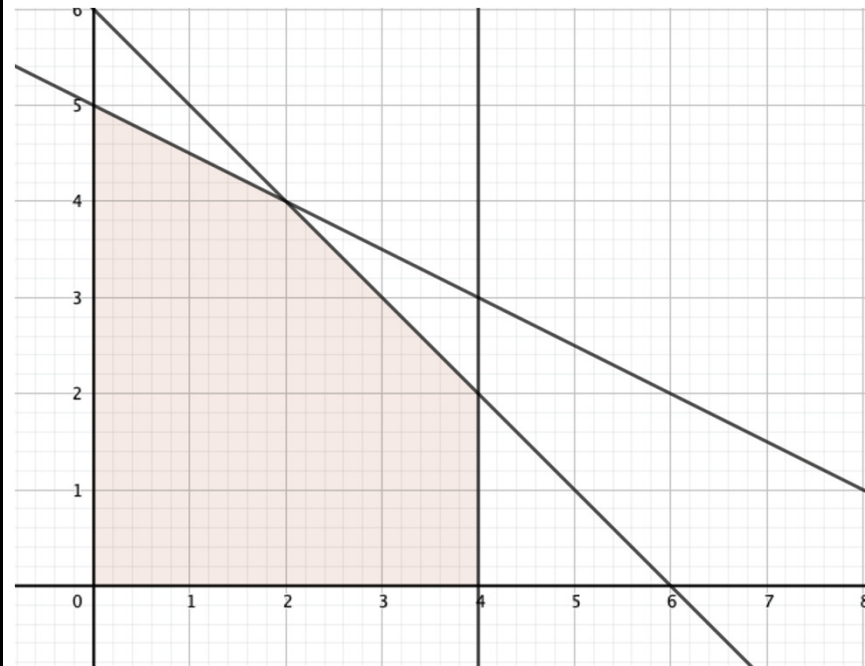


$$f(x, y) = 2x + 3y - 4$$

$$2x + 4y \leq 20$$

$$2x + 2y \leq 16$$

$$x, y \geq 0$$



# Simplex-Algorithmus

- Systematisches Absuchen aller Eckpunkte
- basiert auf Matrixumformungen

$\begin{array}{cccc c} 2 & 4 & 1 & 0 & 0 & 20 \\ 2 & 2 & 0 & 1 & 0 & 12 \\ 4 & 0 & 0 & 0 & 1 & 16 \\ \hline 2 & 3 & 0 & 0 & 0 & 4 \end{array}$	$\begin{array}{cccc c} 0 & 4 & 1 & 0 & -1/2 & 12 \\ 0 & 2 & 0 & 1 & -1/2 & 4 \\ 1 & 0 & 0 & 0 & 1/4 & 4 \\ \hline 0 & 3 & 0 & 0 & -1/2 & -4 \end{array}$
$\begin{array}{cccc c} 2 & 4 & 1 & 0 & 0 & 20 \\ 2 & 2 & 0 & 1 & 0 & 12 \\ 4 & 0 & 0 & 0 & 1 & 16 \\ \hline 2 & 3 & 0 & 0 & 0 & 4 \end{array}$	$p(4, 0, 12, 4, 0)^T$ $\begin{array}{cccc c} 0 & 0 & 1 & -2 & 1/2 & 4 \\ 0 & 1 & 0 & 1/2 & -1/4 & 2 \\ 1 & 0 & 0 & 0 & 1/4 & 4 \\ \hline 0 & 0 & 0 & -1.5 & 1/4 & -10 \end{array}$
$\begin{array}{cccc c} 2 & 4 & 1 & 0 & 0 & 20 \\ 2 & 2 & 0 & 1 & 0 & 12 \\ 4 & 0 & 0 & 0 & 1 & 16 \\ \hline 2 & 3 & 0 & 0 & 0 & 4 \end{array}$	$p(4, 2, 4, 0, 0)^T$

# Rucksackproblem

- Mehrere Objekte mit definierten Eigenschaften
- Beschränkte Kapazität
- Variationen
  - 0/1
  - Beschränkt
  - Unbeschränkt

Handwritten mathematical formulas on a light-colored background:

$$\sum_i C_i X_i \rightarrow \max!$$
$$\sum_i m_i X_i \leq \text{limit}$$
$$X_i \in \{0, 1\}$$
$$i = 1, \dots, n$$

# Travelling Salesman Problem

- Mehrere Städte
- Kürzester Weg
- Anwendungen
  - Logistik
  - Öffentliche Verkehrsmittel
  - Telekommunikation

The whiteboard contains the following handwritten mathematical formulation for the Traveling Salesman Problem (TSP):

$$\begin{aligned} & \text{TSP} \\ & \min \sum_i \sum_{j \neq i} c_{ij} x_{ij} \\ & \left. \begin{aligned} \sum_{i+j} x_{ij} &= 1 \\ \sum_{j+i} x_{ij} &= 1 \end{aligned} \right\} \forall i, j = 1, \dots, n \\ & x_{ij} \in \{0, 1\} \quad \forall i = 1, \dots, n \\ & \sum_{i \in Q} \sum_{j \in Q} x_{ij} \leq |Q| - 1 \\ & \forall Q \subseteq \{1, \dots, n\} \end{aligned}$$

# Ansatz - Brute force

```
11 if(length(values) != length(weights))
12     disp("Anzahl der Nutzwerte und Gewichte stimmen nicht überein ")
13     return;
14 end
15
16 %Initialisierung
17 m = length(values);
18 limit = ceil(limit);
19 items = zeros(m,1);
20 value = 0;
21
22 p = perms(1:m);
23
24 tic
25 for i = 1:rows(p)
26     weight = 0;
27     tmpValue = 0;
28     tmpItems = zeros(m,1);
29     j = 1;
30
31     while ((j <= m) && (weight+weights(p(i,j)) <= limit) ) % = columns(p)
32         weight += weights(p(i,j));
33         tmpItems(p(i,j)) = 1;
34         j += 1;
35     end
36
37     tmpValue = sum(values(p(i,1:j-1)));
38
39     if (tmpValue > value)
40         value = tmpValue;
41         items = tmpItems;
42     end
```

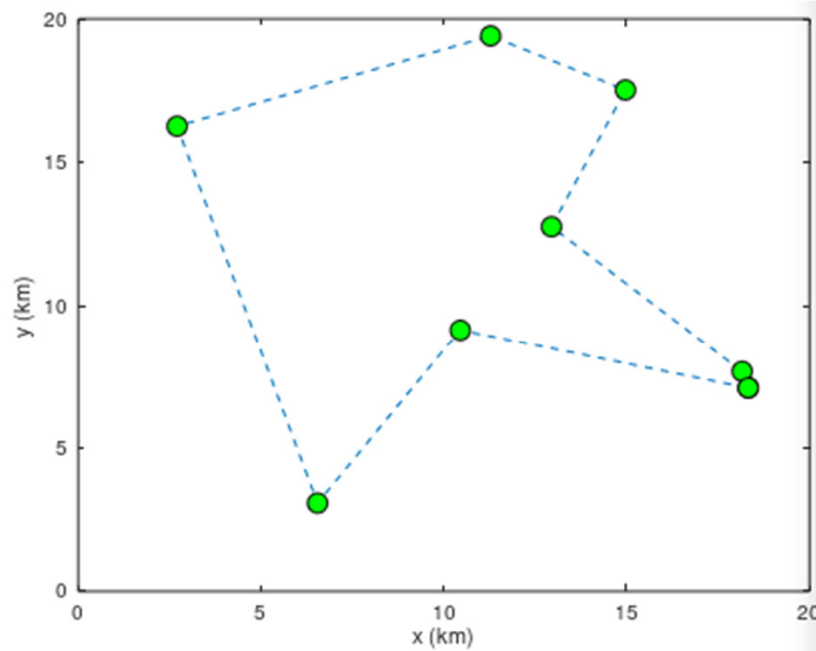
# Ansatz – Greedy-Methoden

```
7 len = length(gewicht);
8 %Matrix erstellen mit drei Spalten und so vielen Reihen, wie Objekte da sind.
9 objekte = zeros(len, 3);
10
11 objekte(:, 1) = gewicht;
12 objekte(:, 2) = nutzen;
13
14 %Verhältnis von Nutzen und Masse in dritte Spalte
15 for i = 1:len
16     objekte(i, 3) = objekte(i, 2)/objekte(i, 1);
17 end
18
19 %Nach Nutzen/Masse-Verhältnis sortieren
20 objekte = sortrows(objekte, -3);
21
22 gepackt = zeros(len, 2);
23 masse = 0;
24
25 for i = 1:len
26
27     if (masse + objekte(i, 1)) <= limit
28         gepackt(i,1) = objekte(i,1);
29         gepackt(i,2) = objekte(i,2);
30         masse += objekte(i, 1);
31     end
32
33 end
34
35 gepackt( all(~gepackt,2), : ) = [];
36 wert = sum(gepackt(:,2));
```

# Vergleich

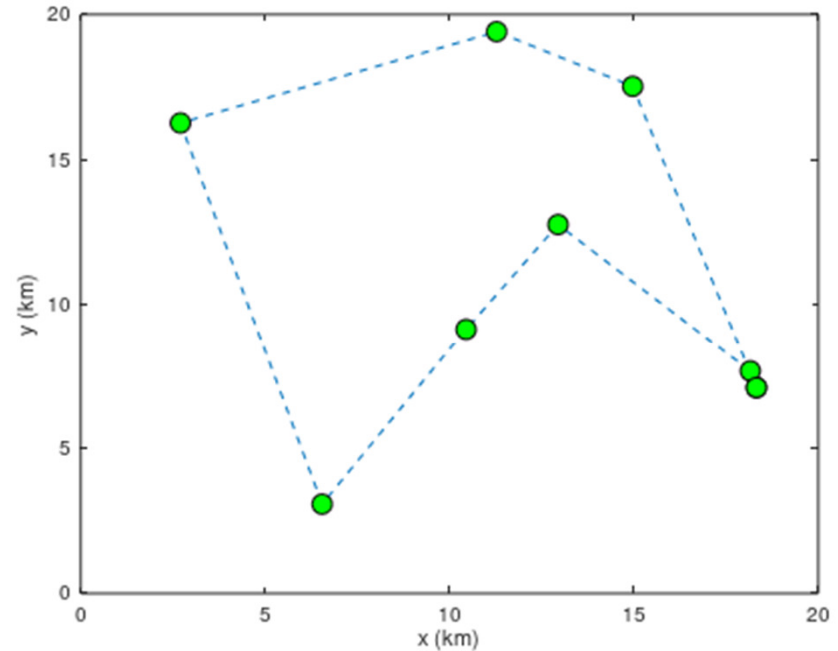
**Brute force**

$$s_{ges} = 55,4 \text{ km}$$



**Greedy**

$$s_{ges} = 57,4 \text{ km}$$







Tobias Ebenführer  
Paul Brüllmeir  
Vera Pamminger  
Laurenz Stadler  
Felix Weißenberger

## DI Rainer Schneckenleitner

Richard Emeder  
Karoline Moser  
Belinda Eder  
Niklas Hockl  
Justus Wertal

