Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Lehrstuhl für
INFORMATIK I
Algorithmen & Komplexität

Institut für Informatik

Algorithmische Graphentheorie

Sommersemester 2021

3. Vorlesung

Lineares Programmieren

Prof. Dr. Alexander Wolff *Lehrstuhl für Informatik I*

Gewinnmaximierung

Sie sind Chef einer kleinen Firma, die zwei Produkte P_1 und P_2 herstellt. Produzieren Sie x_1 Einheiten P_1 und x_2 Einheiten P_2, so beträgt Ihr Gewinn in €

$$G(x_1, x_2) = 30x_1 + 50x_2$$

Gewinnmaximierung

Sie sind Chef einer kleinen Firma, die zwei Produkte P_1 und P_2 herstellt. Produzieren Sie x_1 Einheiten P_1 und x_2 Einheiten P_2, so beträgt Ihr Gewinn in €

$$G(x_1, x_2) = 30x_1 + 50x_2$$

Drei Mitarbeiter M_A, M_B und M_C produzieren die dafür jeweils notwendigen Einzelteile. Dabei brauchen sie eine bestimmte Zeit pro Teil und haben jeweils eine maximale Arbeitszeit, die die Produktion der Einzelteile einschränkt:

$$\begin{aligned} M_A: &\quad 4x_1 + 11x_2 \leq 880 \\ M_B: &\quad x_1 + x_2 \leq 150 \\ M_C: &\quad x_2 \leq 60 \end{aligned}$$

Gewinnmaximierung

Sie sind Chef einer kleinen Firma, die zwei Produkte P_1 und P_2 herstellt. Produzieren Sie x_1 Einheiten P_1 und x_2 Einheiten P_2, so beträgt Ihr Gewinn in €

$$G(x_1, x_2) = 30x_1 + 50x_2$$

Drei Mitarbeiter M_A, M_B und M_C produzieren die dafür jeweils notwendigen Einzelteile. Dabei brauchen sie eine bestimmte Zeit pro Teil und haben jeweils eine maximale Arbeitszeit, die die Produktion der Einzelteile einschränkt:

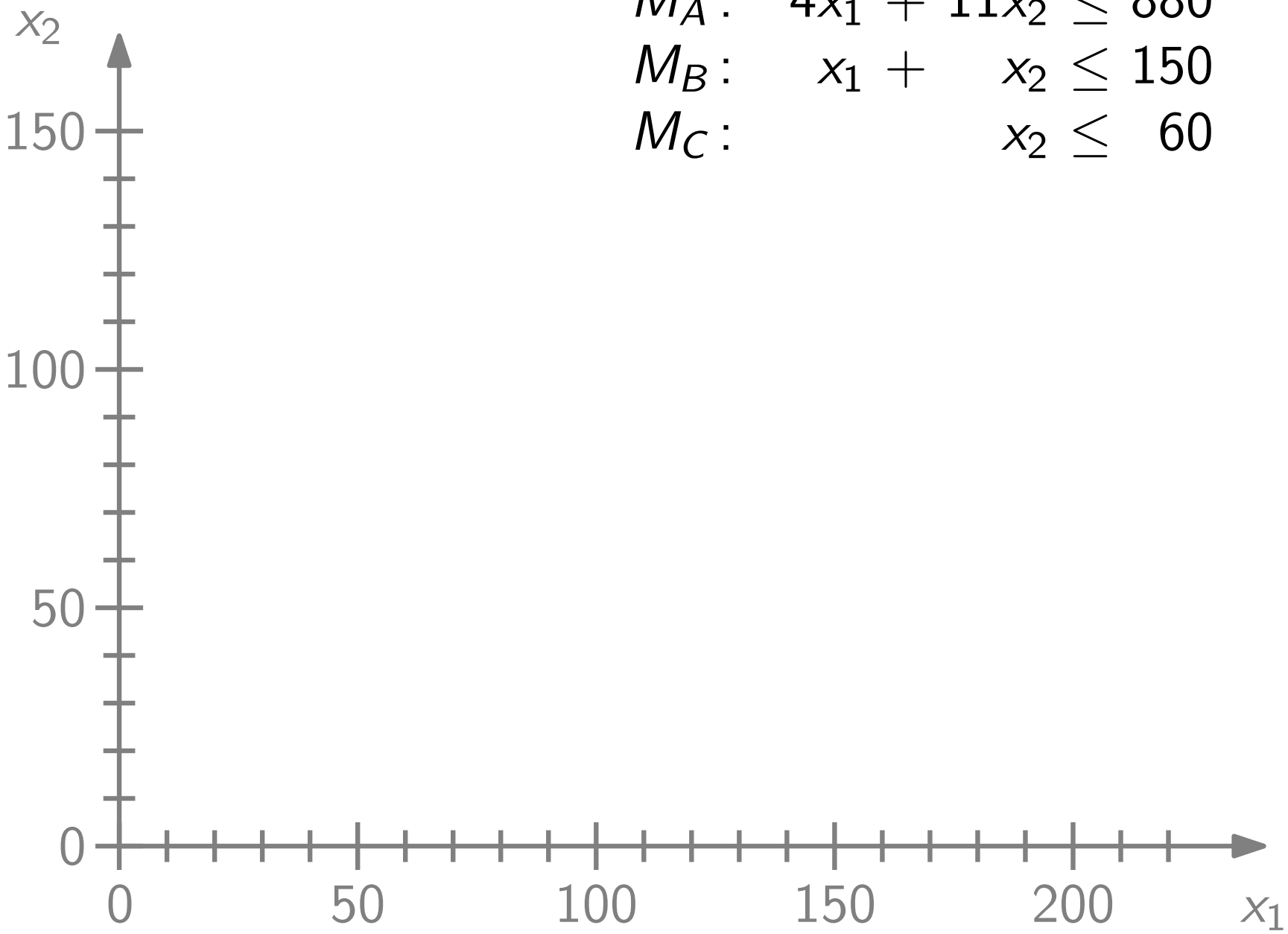
$$\begin{aligned} M_A: &\quad 4x_1 + 11x_2 \leq 880 \\ M_B: &\quad x_1 + x_2 \leq 150 \\ M_C: &\quad x_2 \leq 60 \end{aligned}$$

Welche Wahl von (x_1, x_2) maximiert den Gewinn?

Lösung

Lineare Beschränkungen:

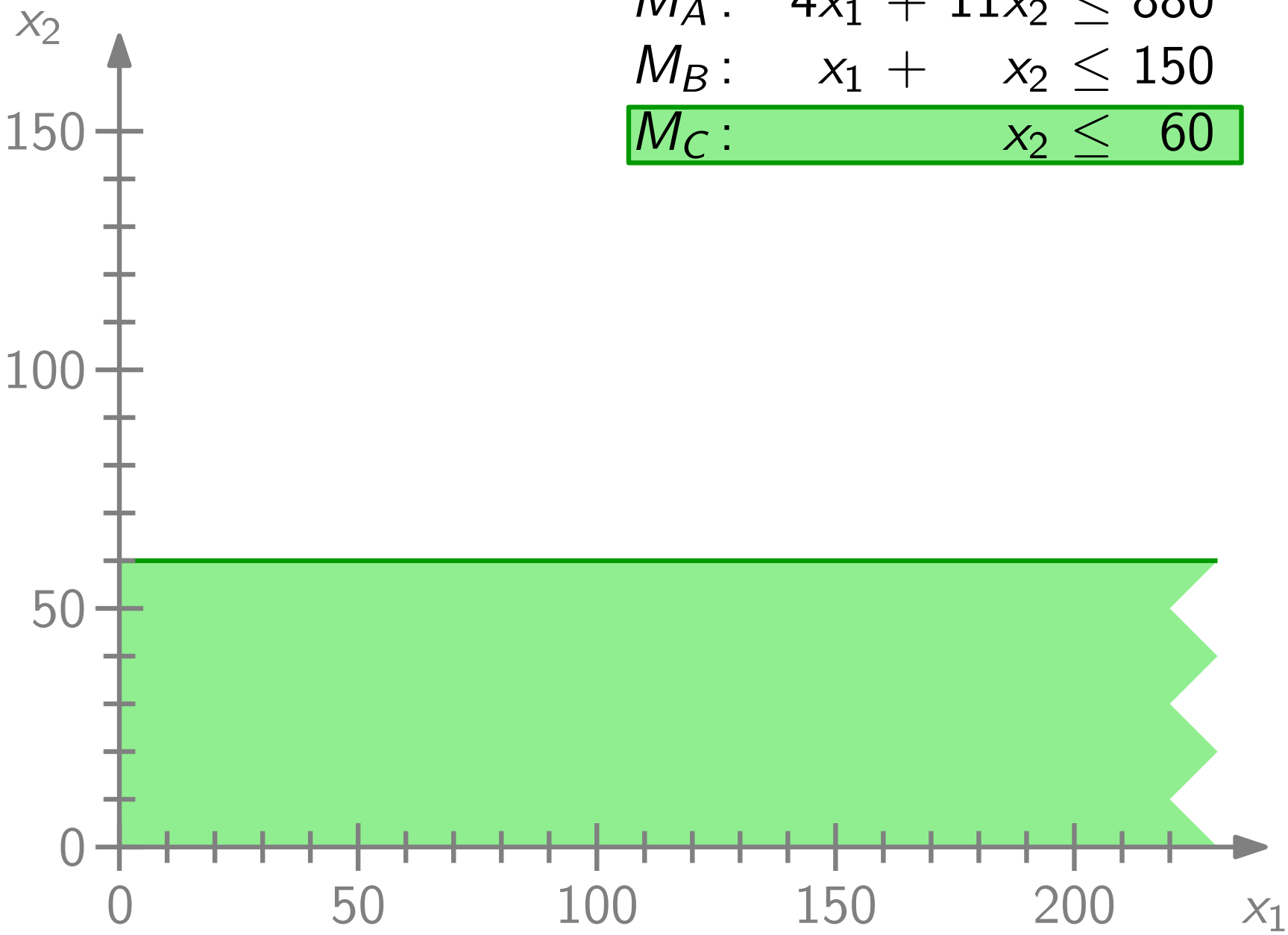
$$
\begin{aligned}
M_A: &\quad 4x_1 + 11x_2 \leq 880 \\
M_B: &\quad x_1 + x_2 \leq 150 \\
M_C: &\quad x_2 \leq 60
\end{aligned}
$$



Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$



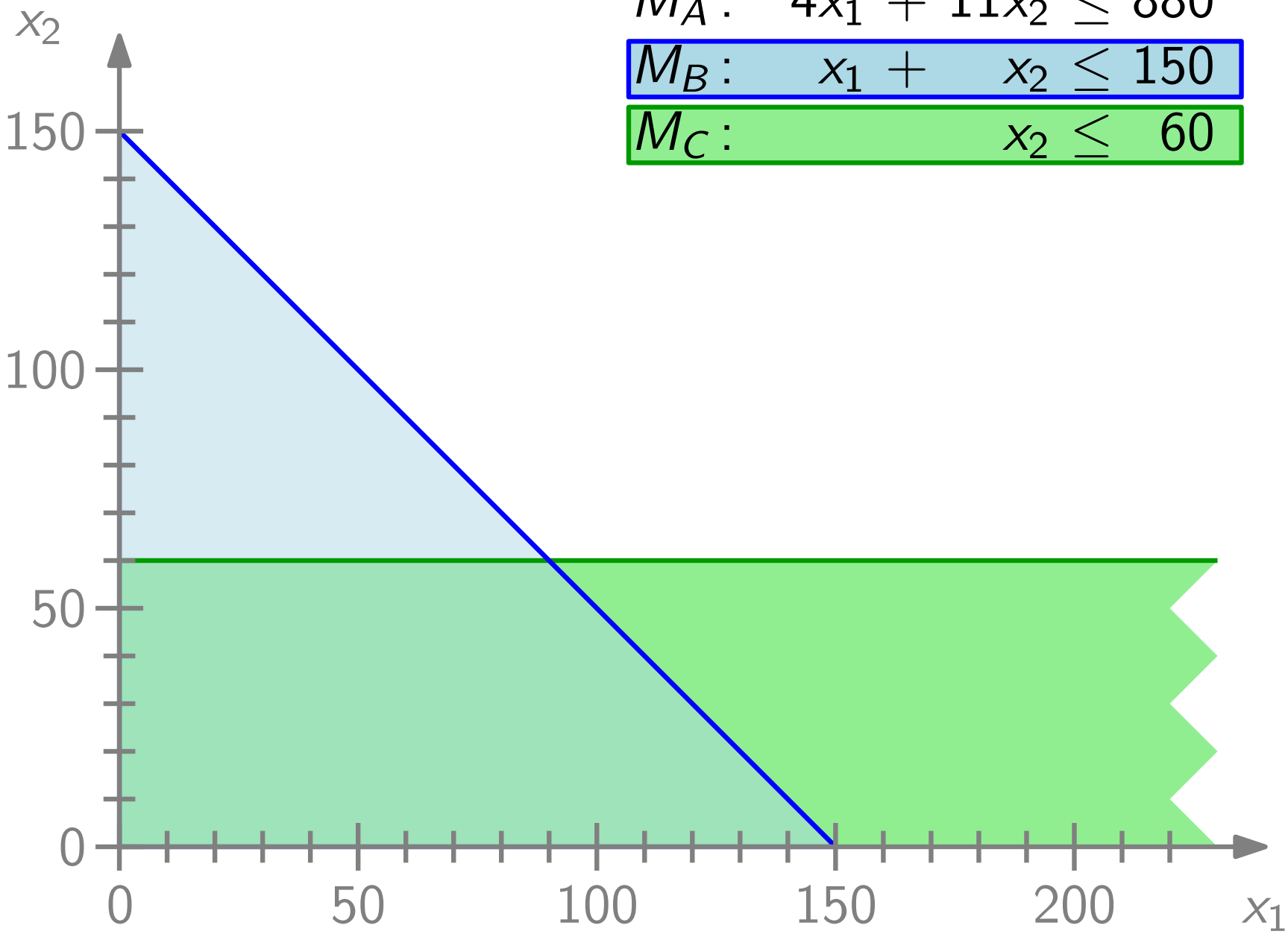
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$

$$M_B: \quad x_1 + x_2 \leq 150$$

$$M_C: \quad x_2 \leq 60$$



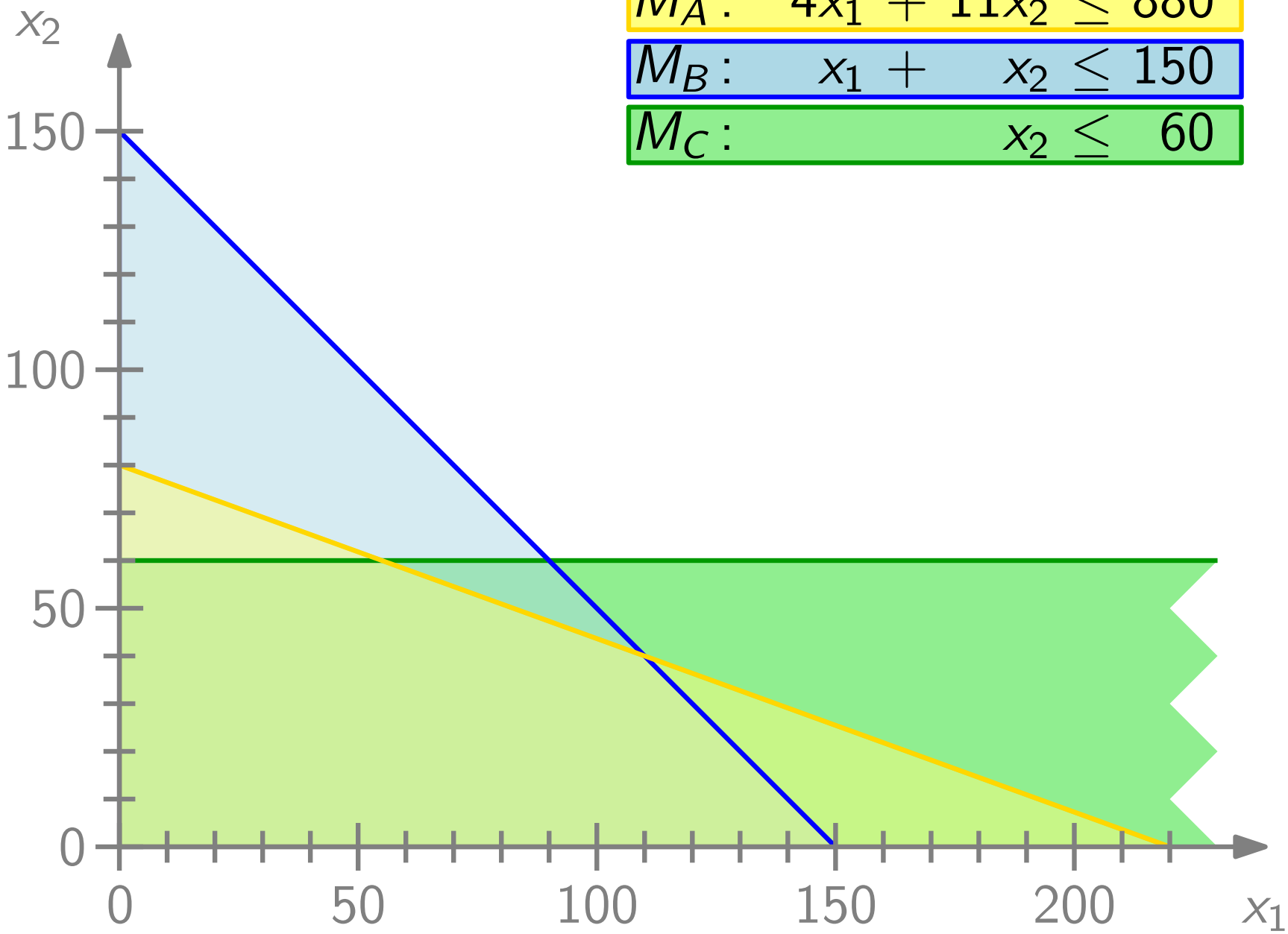
Lösung

Lineare Beschränkungen:

$M_A: \quad 4x_1 + 11x_2 \leq 880$

$M_B: \quad x_1 + x_2 \leq 150$

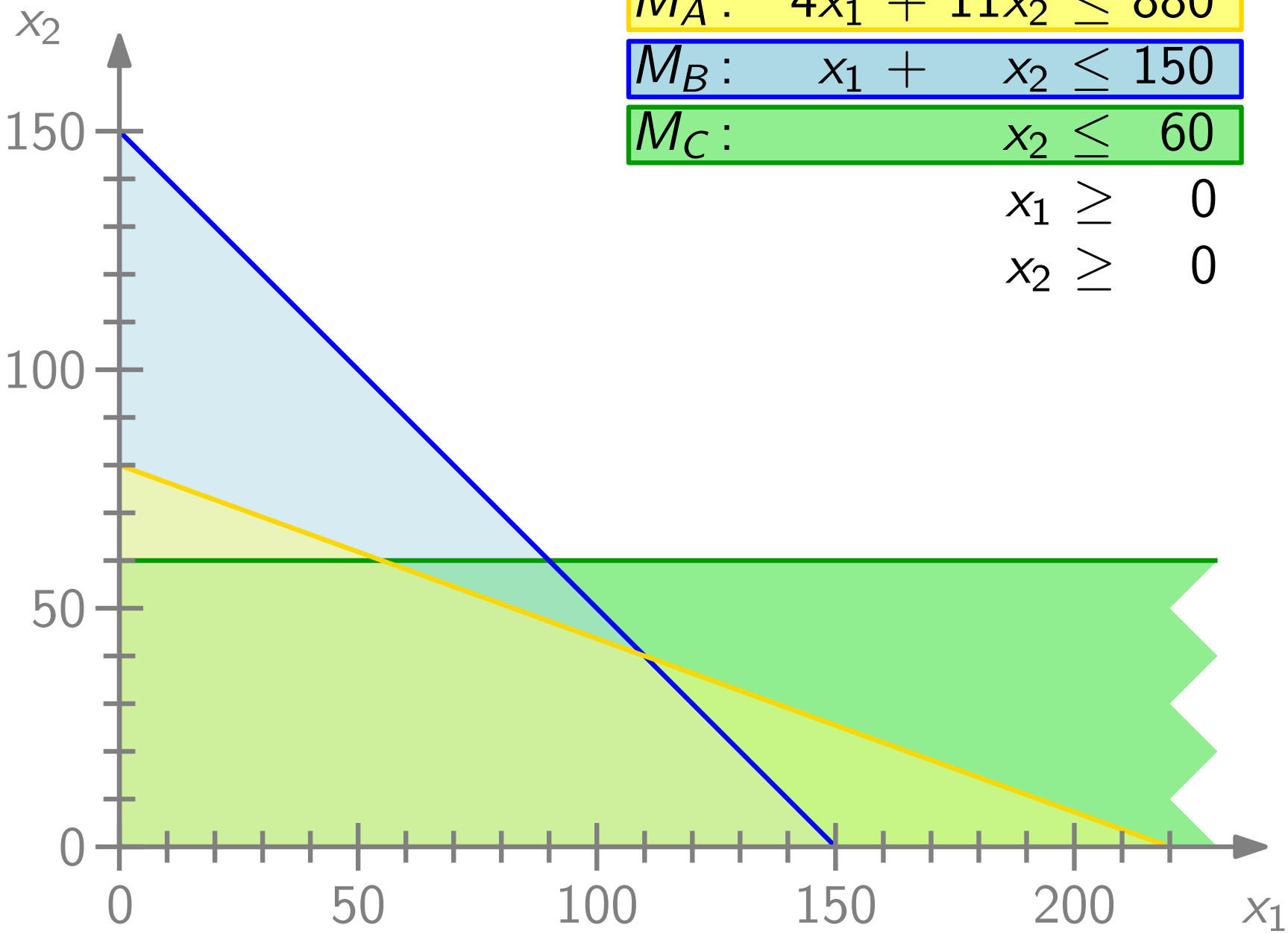
$M_C: \quad x_2 \leq 60$



Lösung

Lineare Beschränkungen:

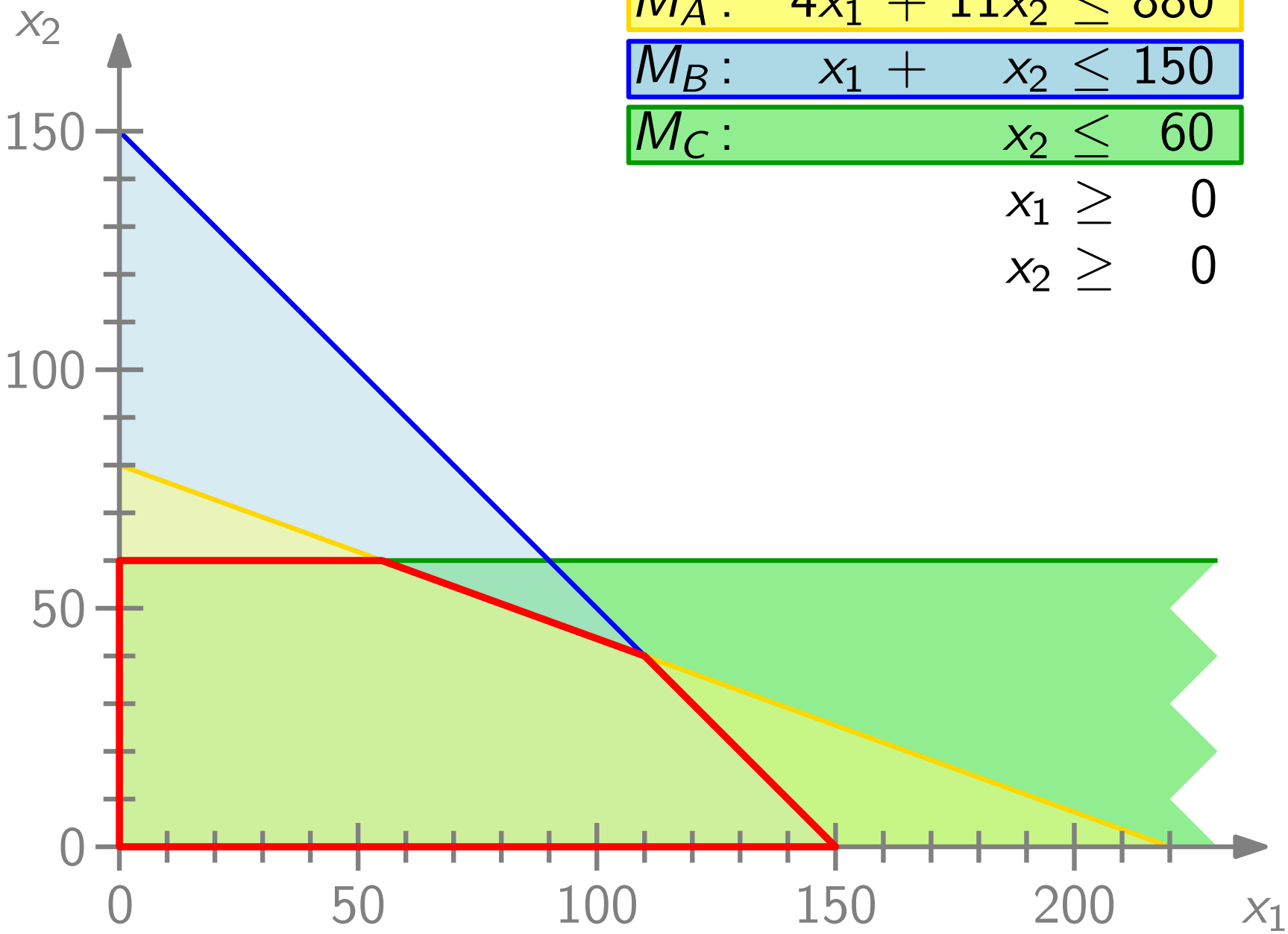
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$



Lösung

Lineare Beschränkungen:

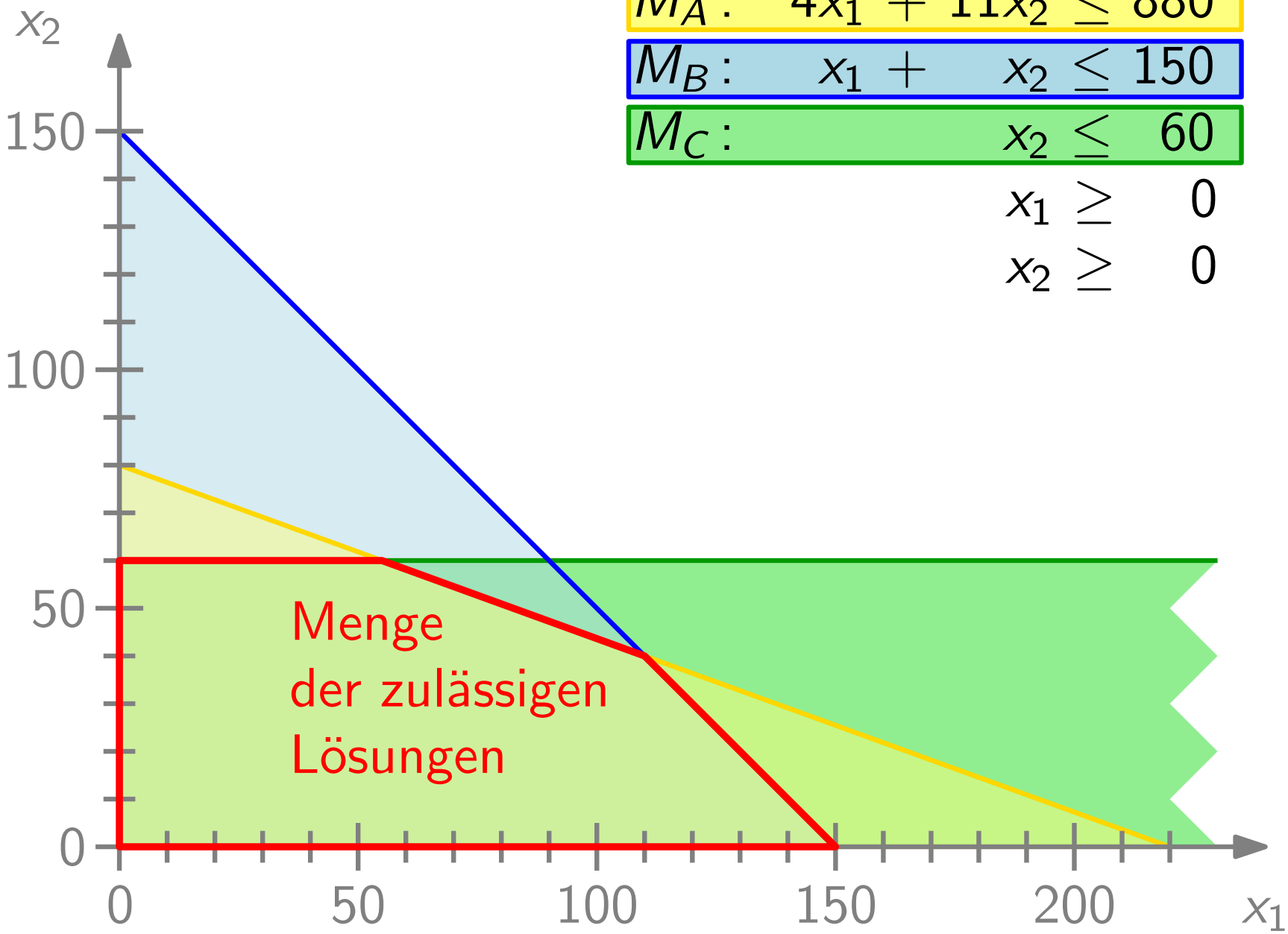
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$



Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$



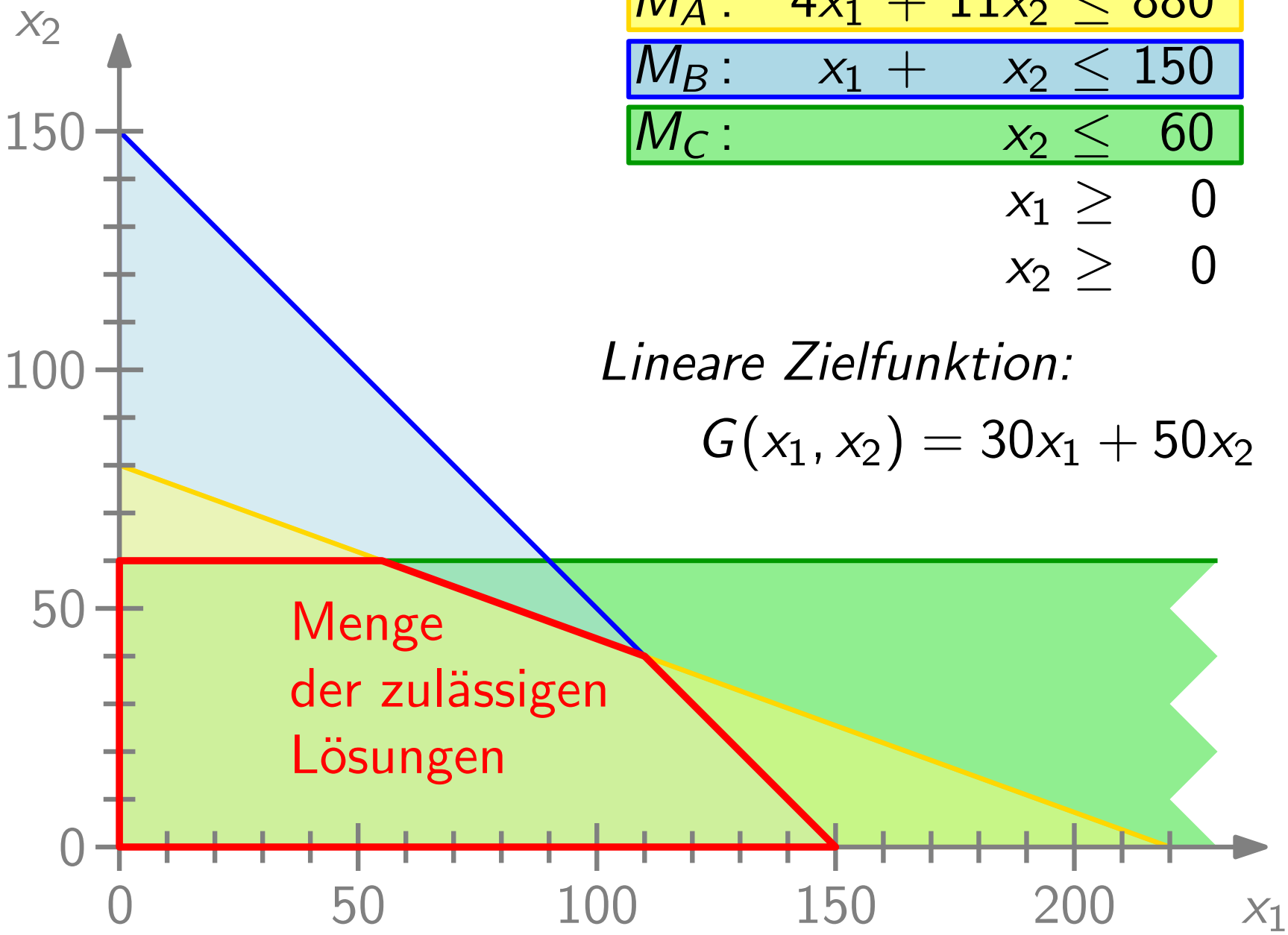
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2$$



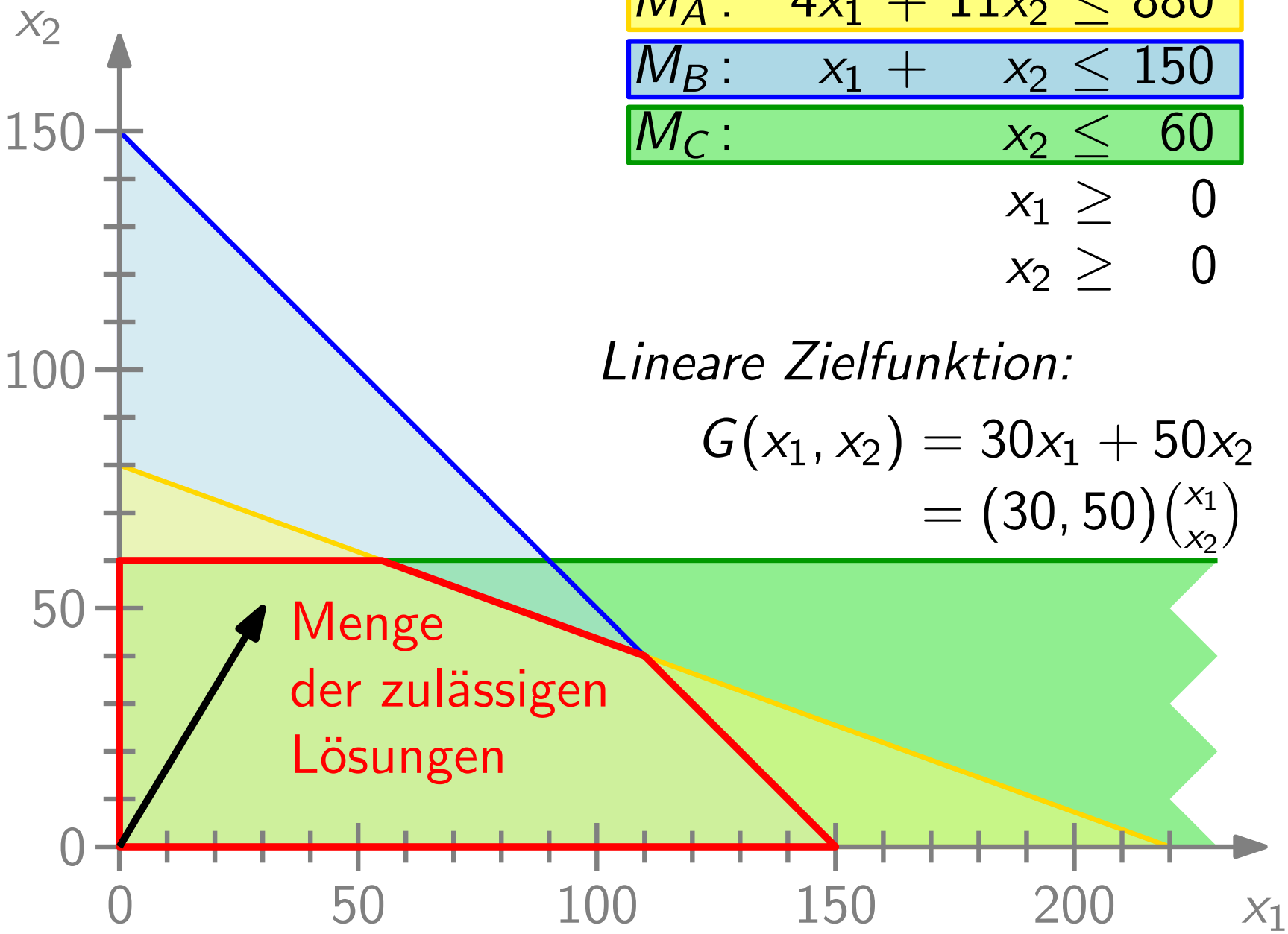
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



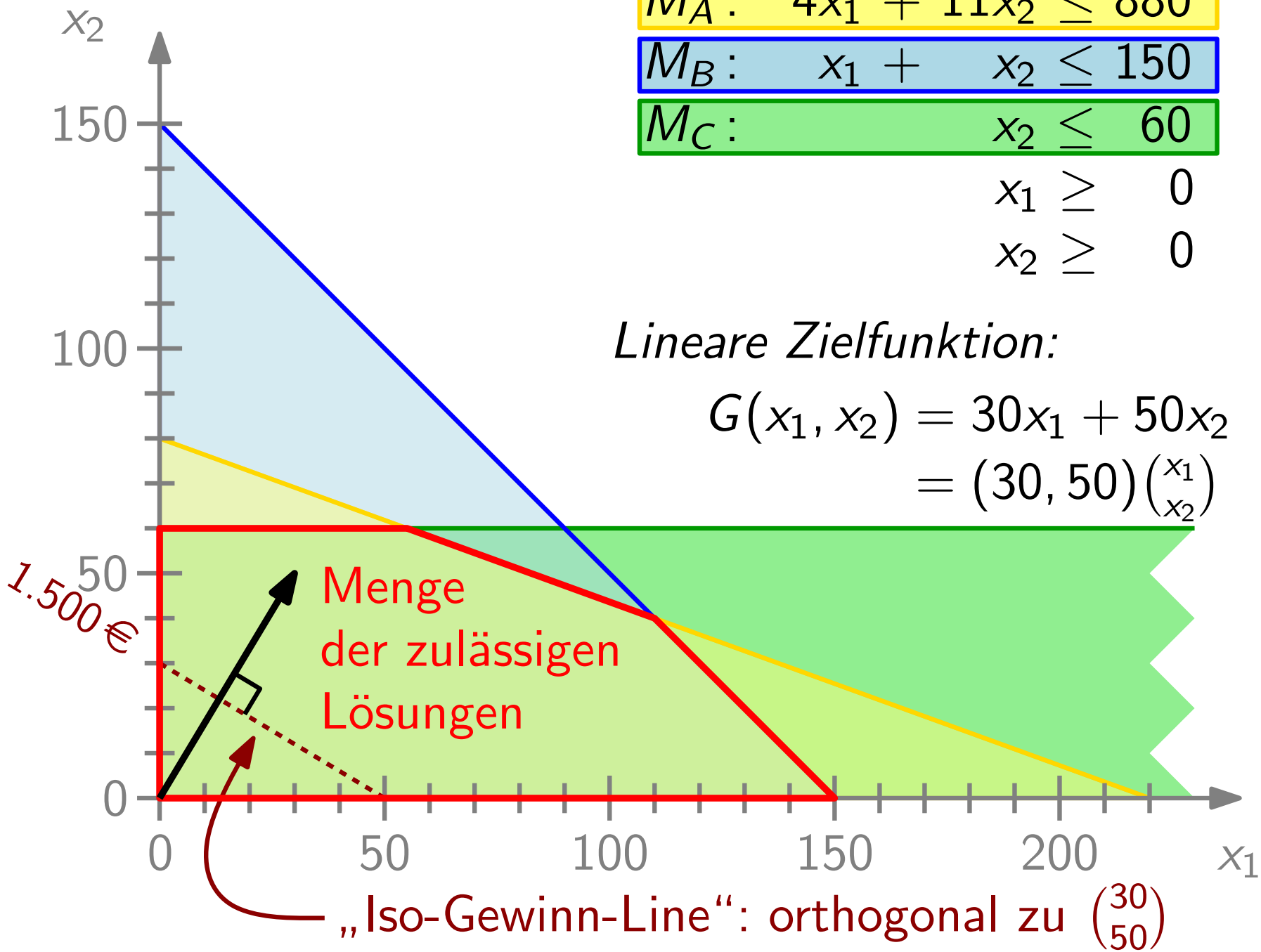
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



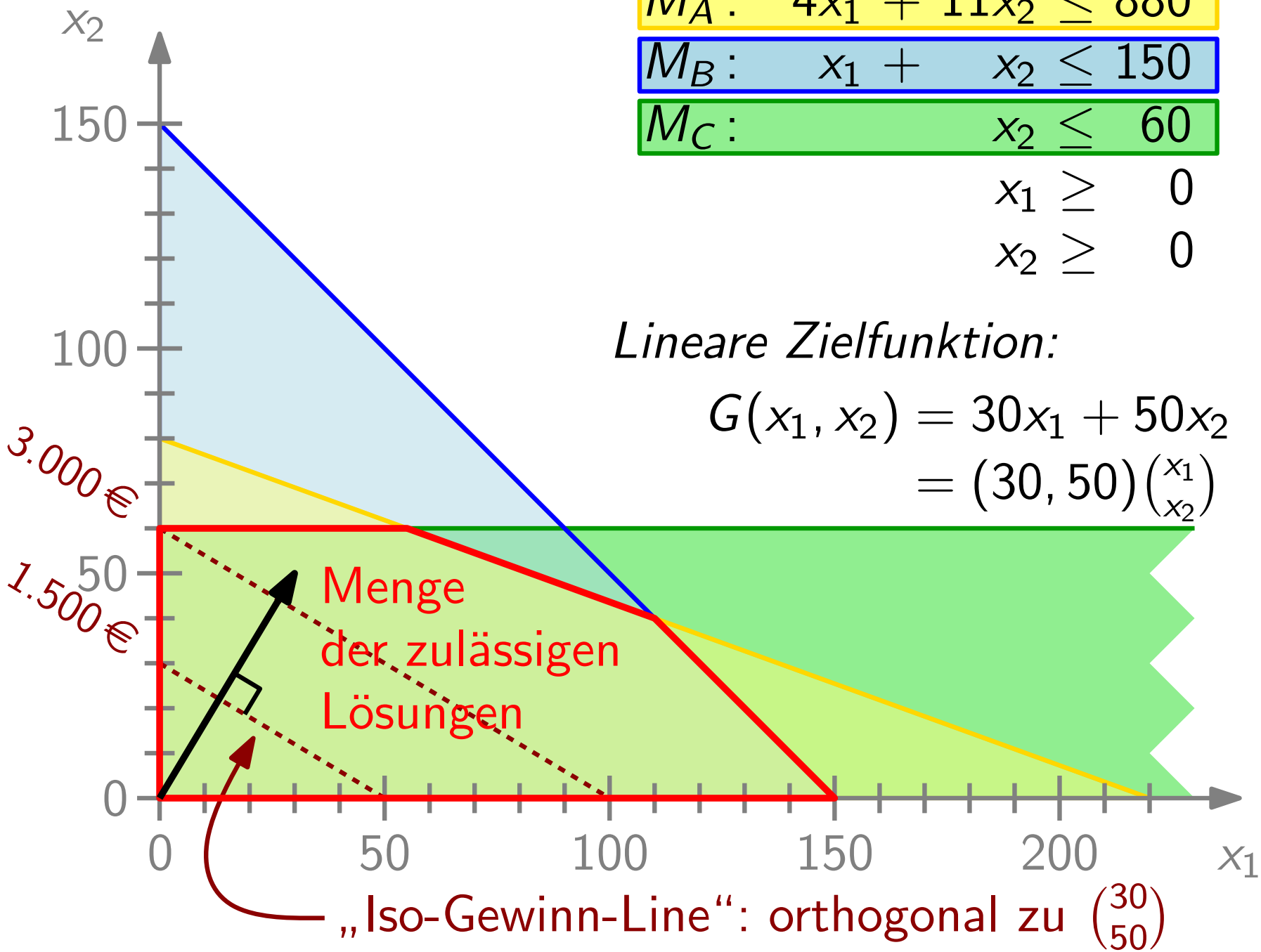
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



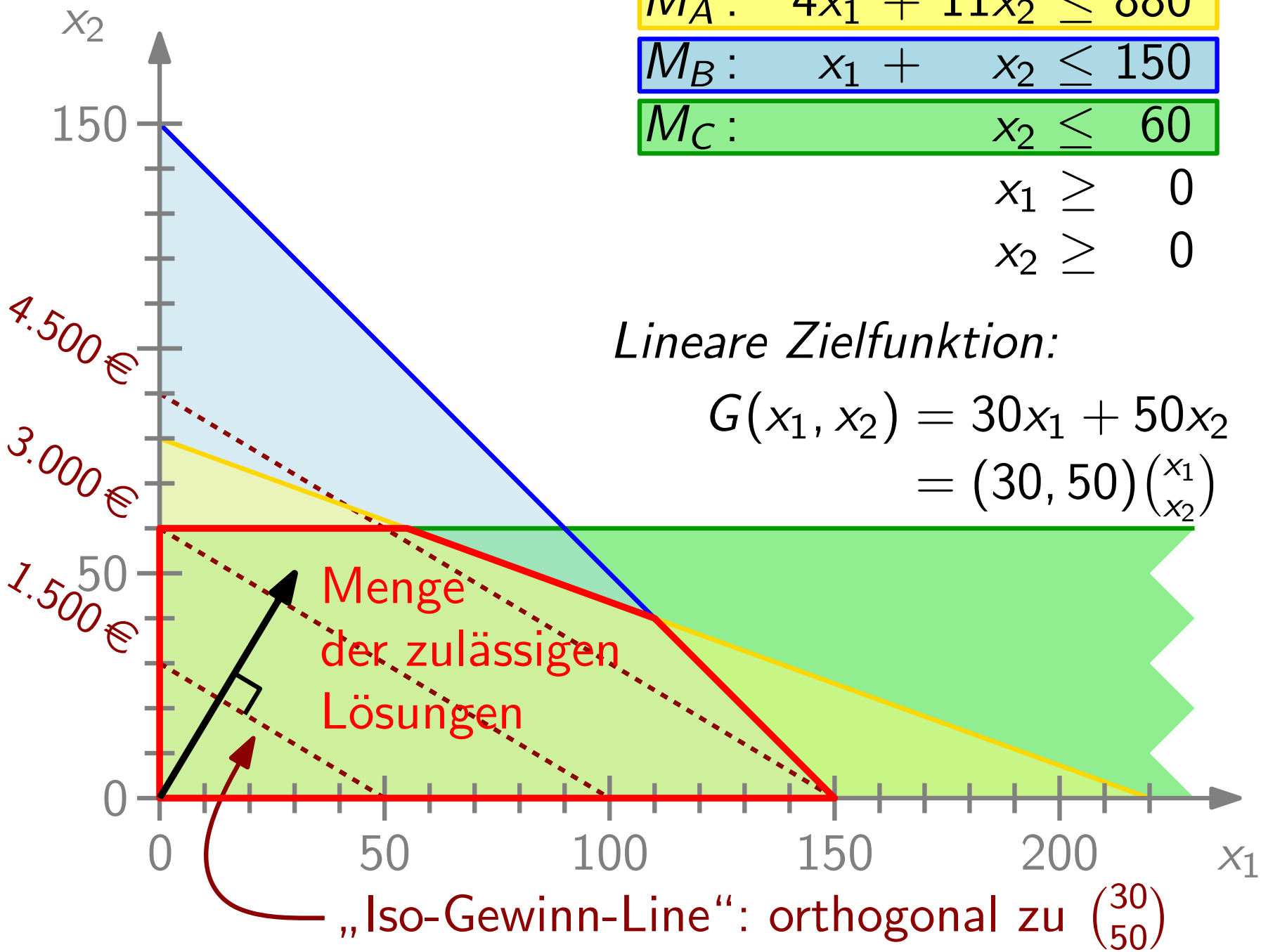
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



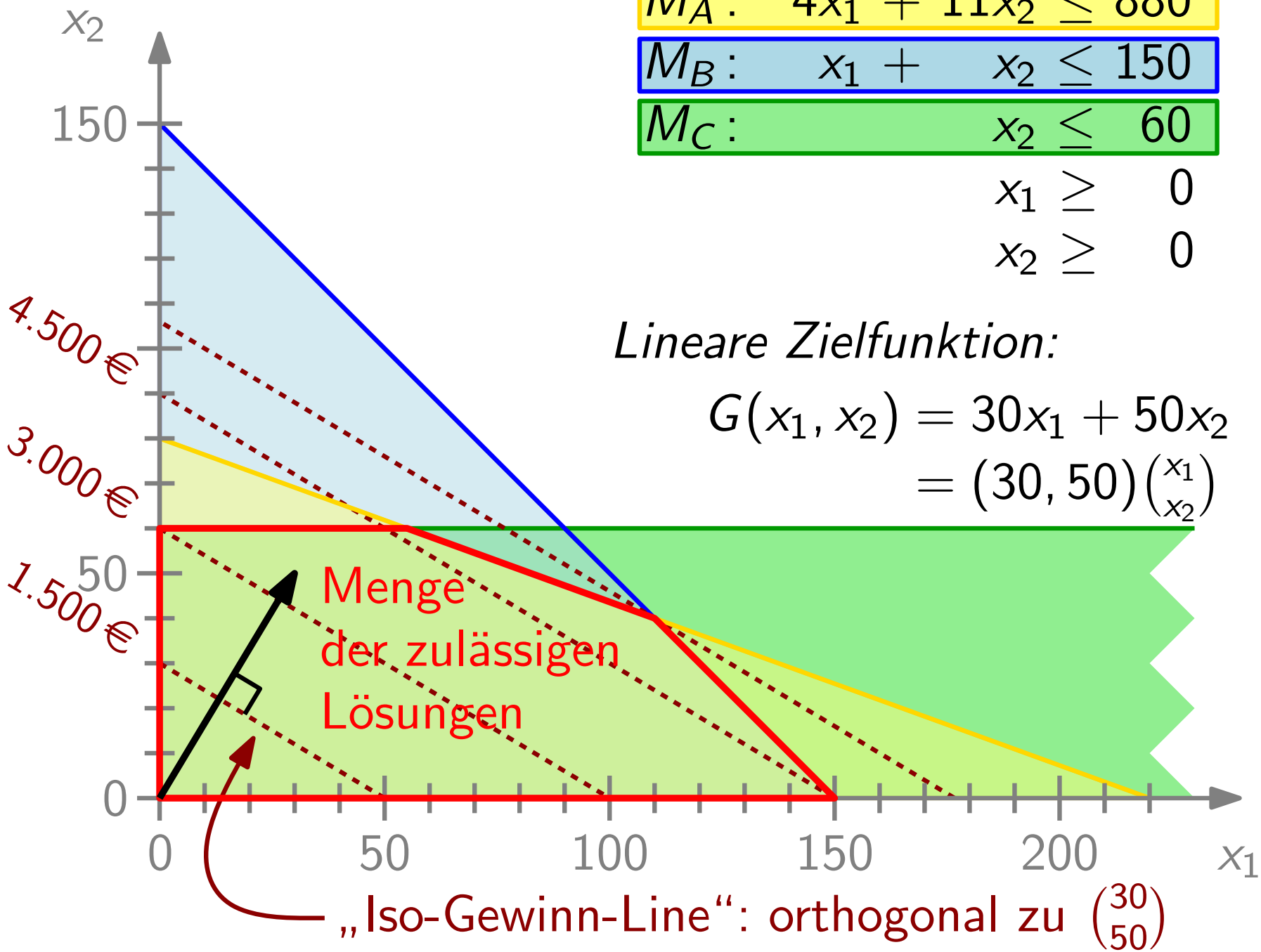
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



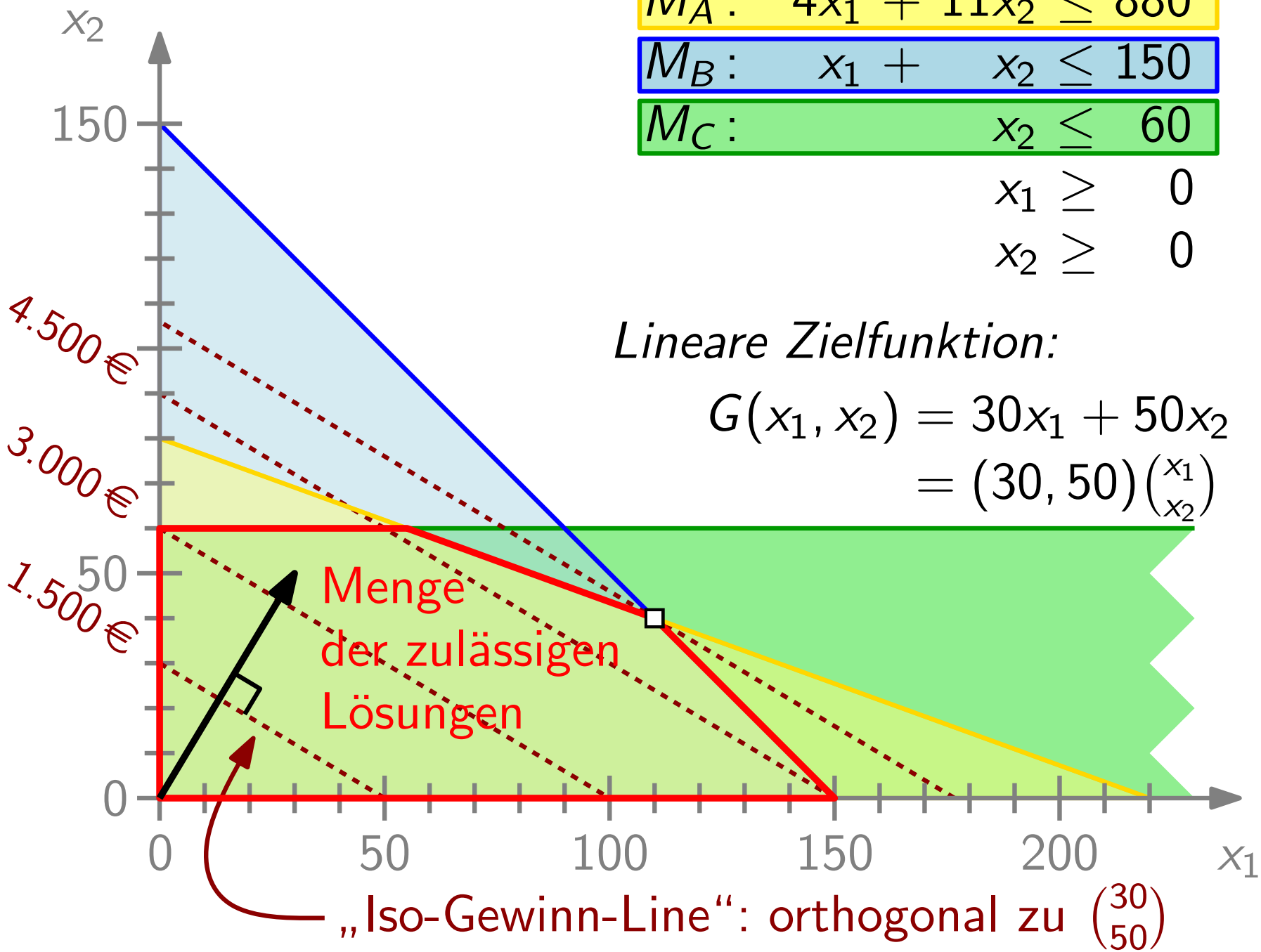
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



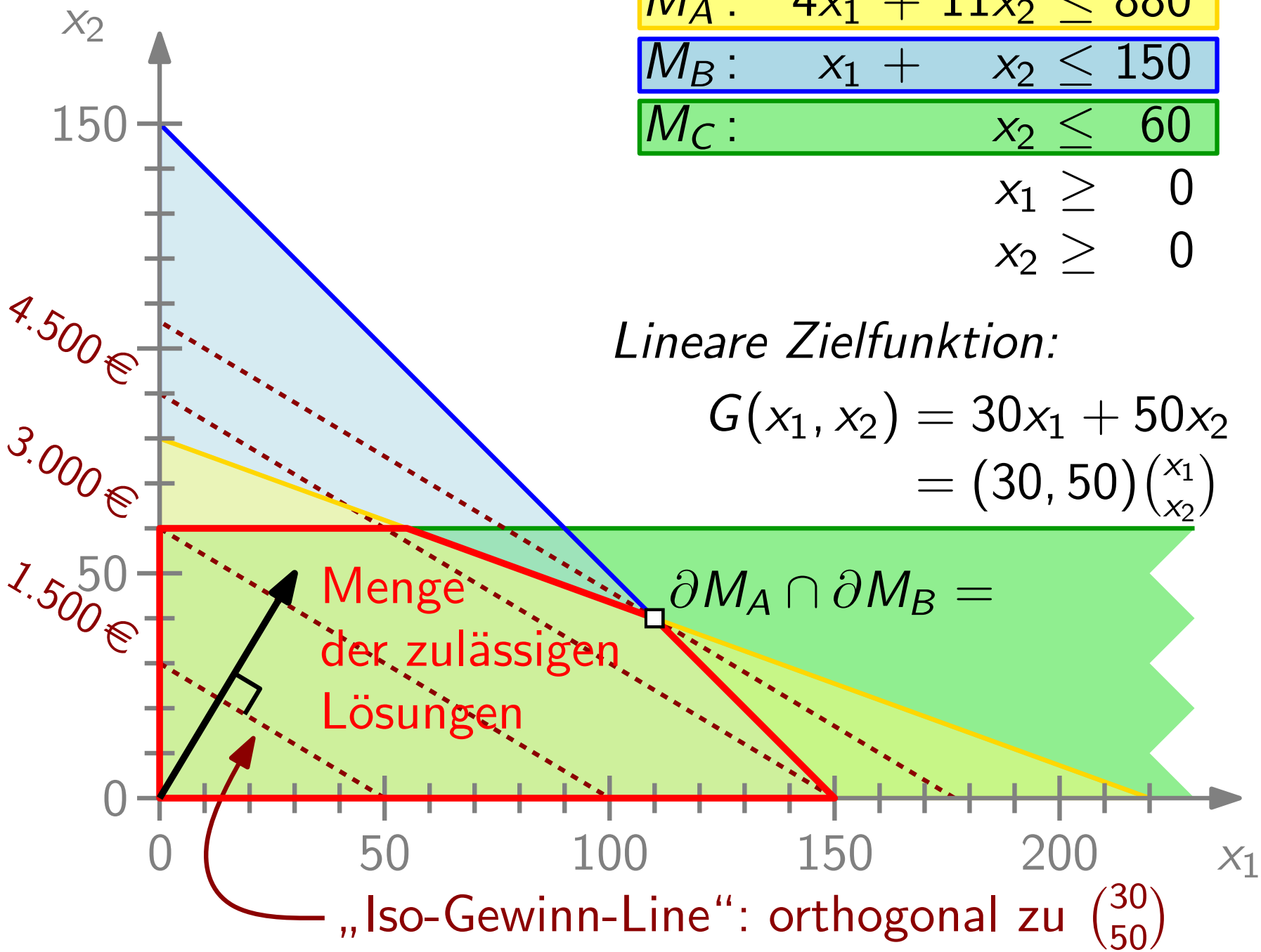
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



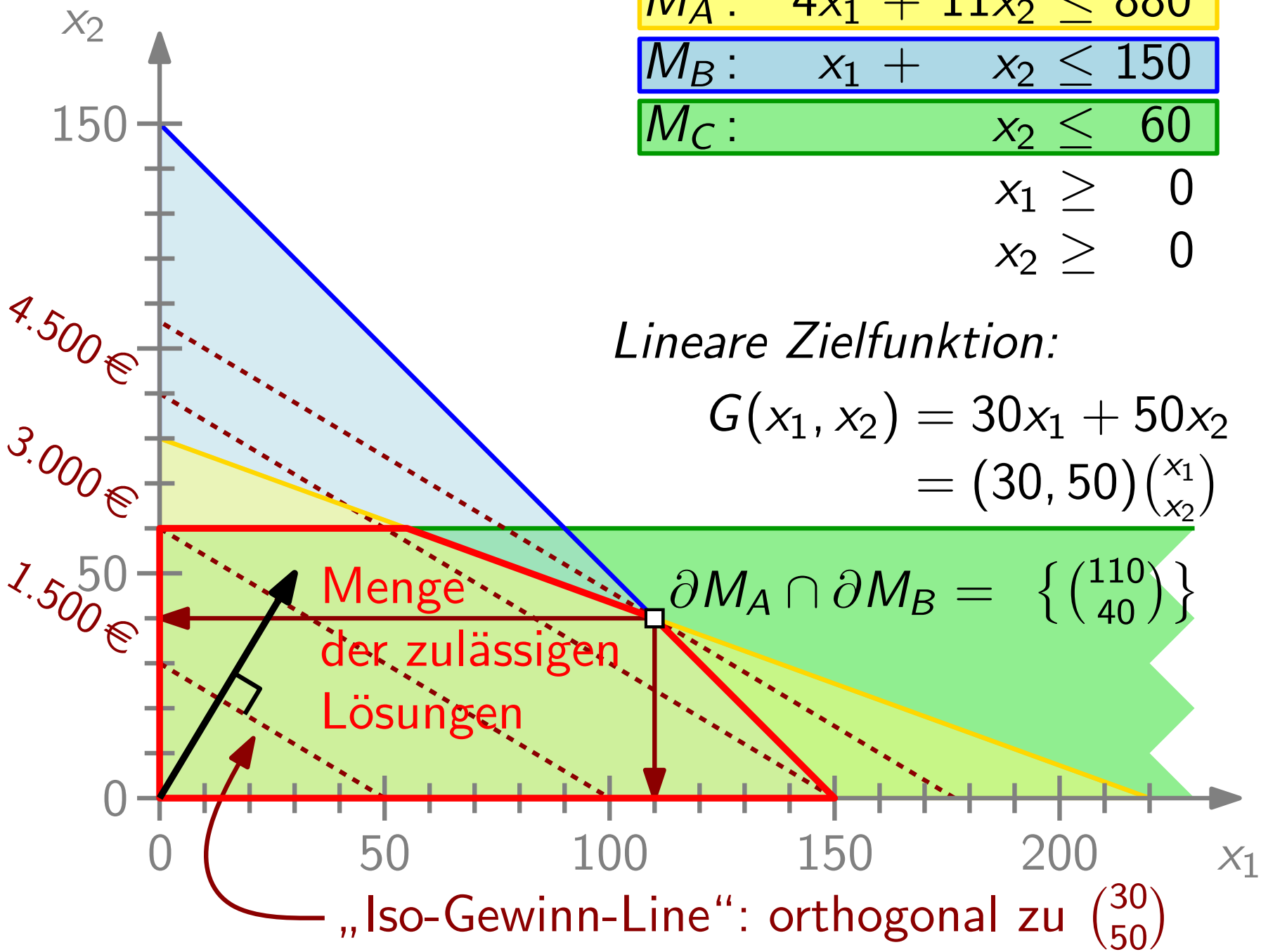
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



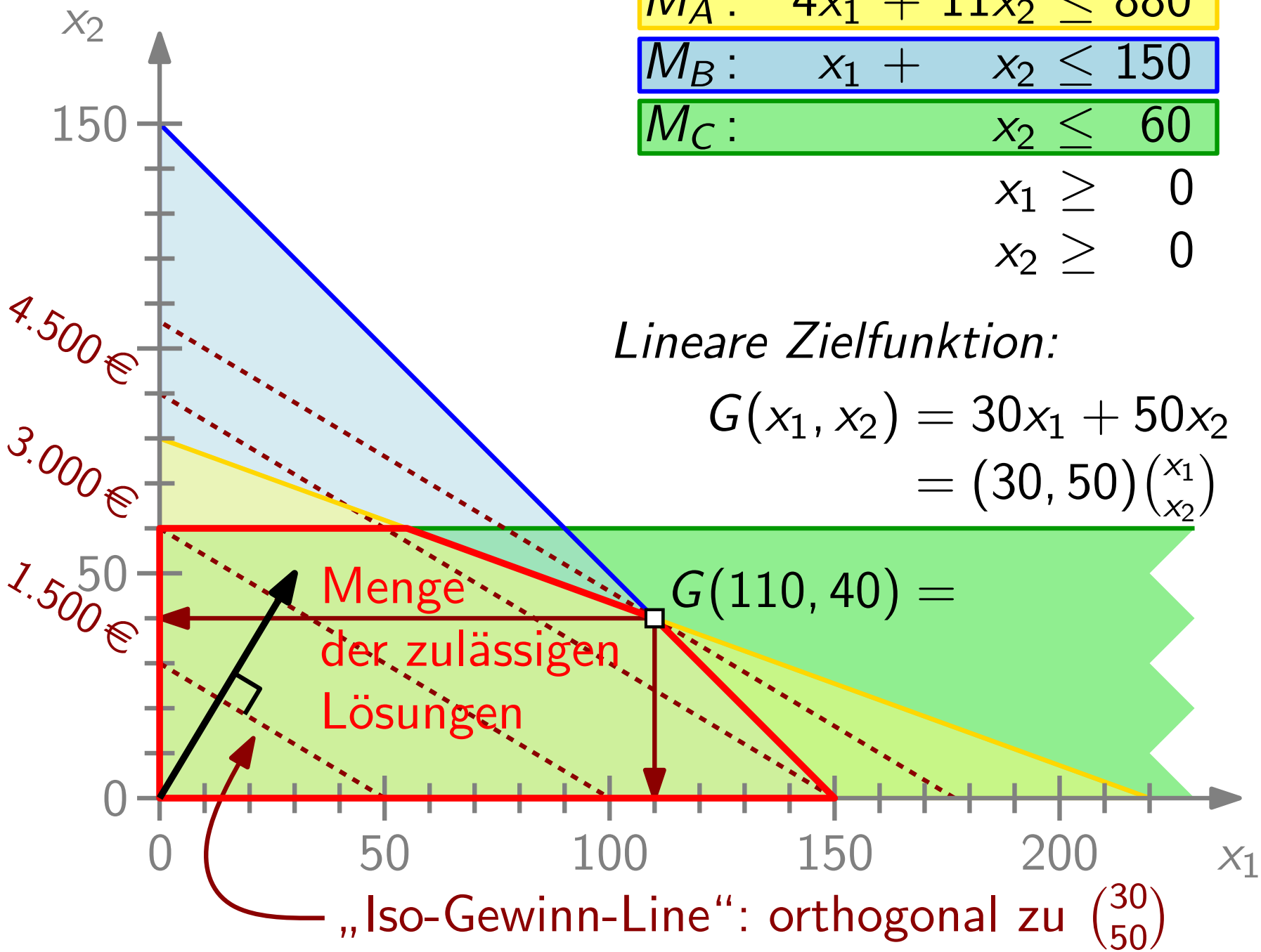
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



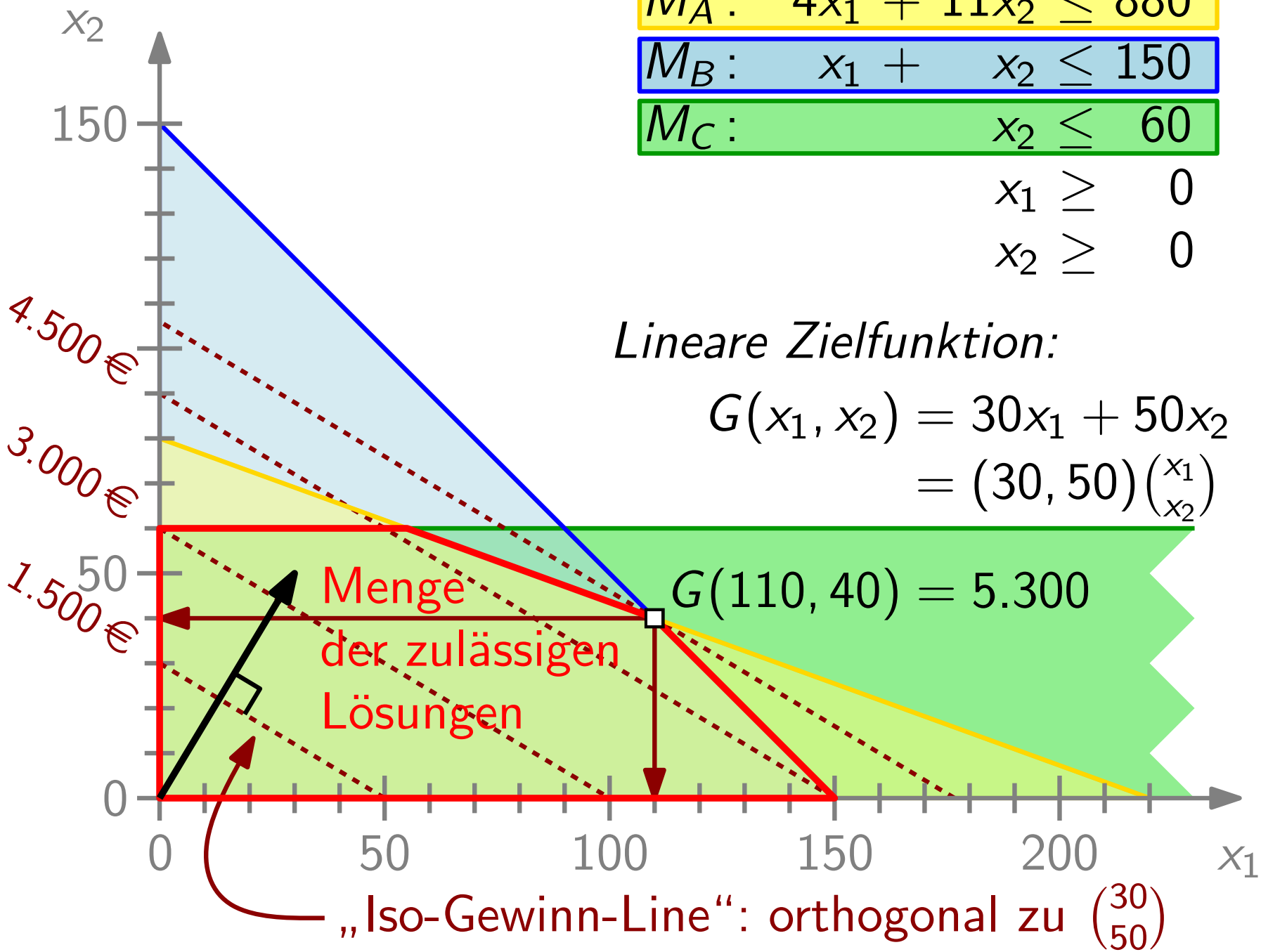
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



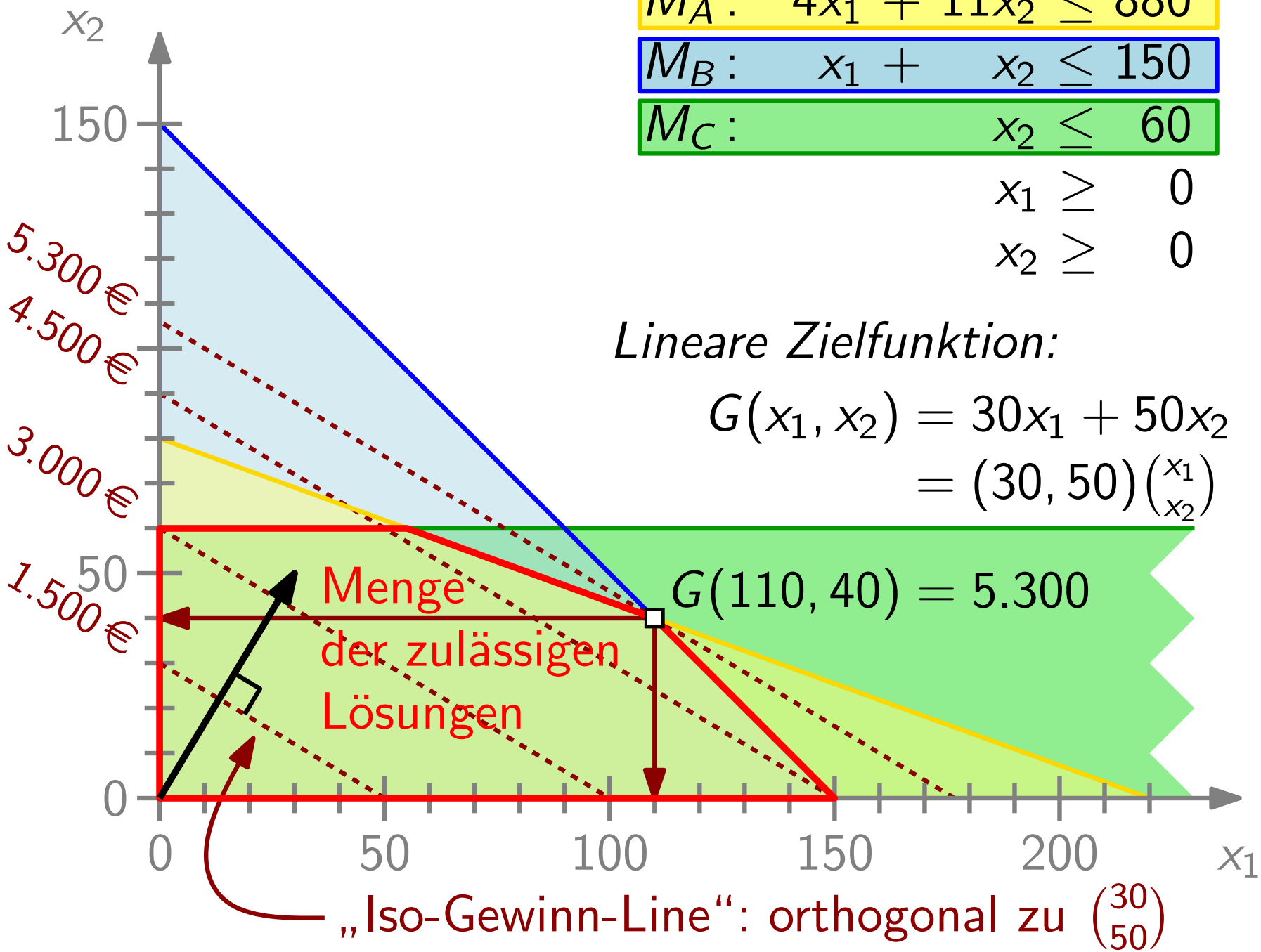
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



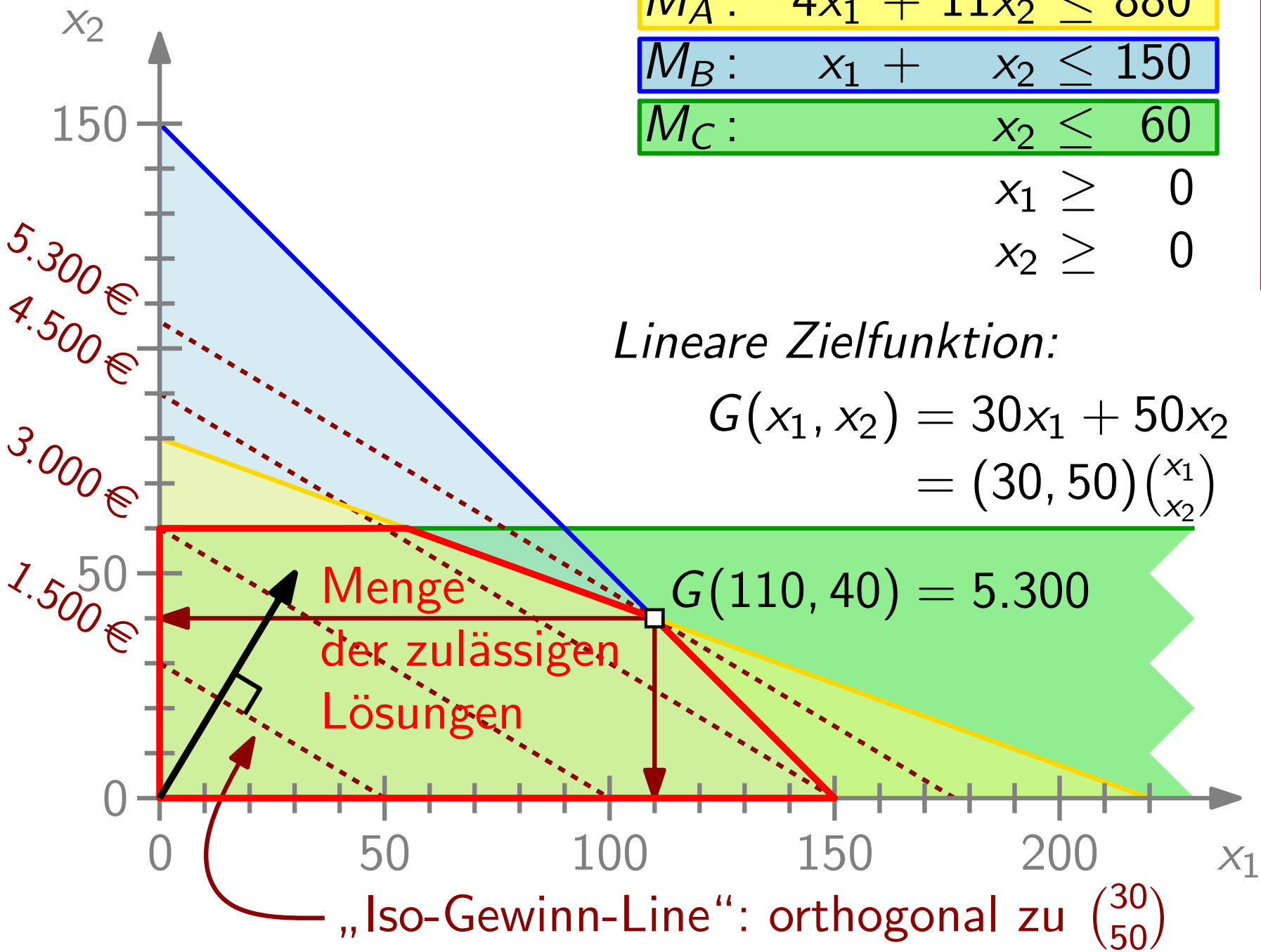
Lösung

Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

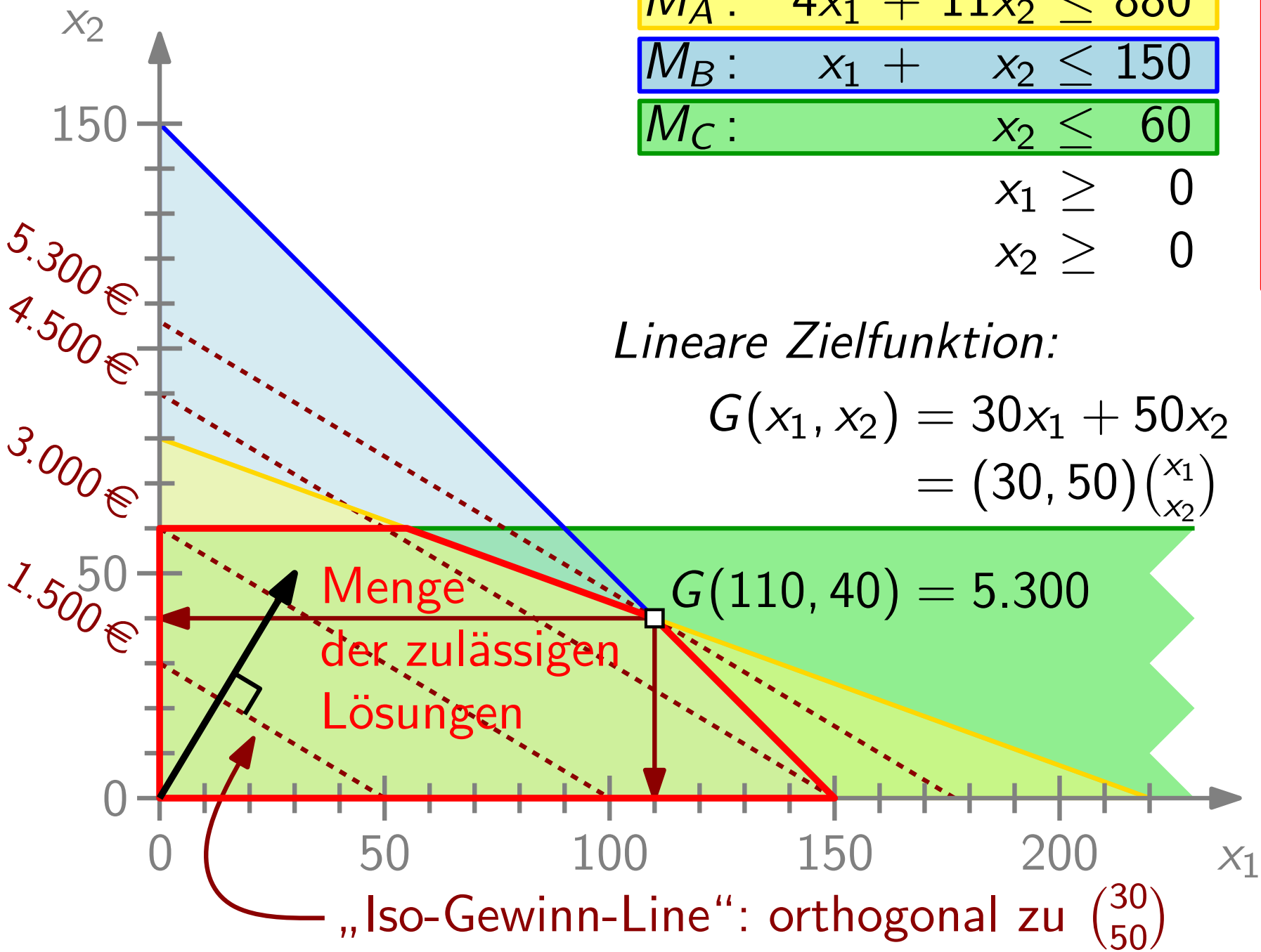
Lineare Beschränkungen:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

Lineare Beschränkungen:

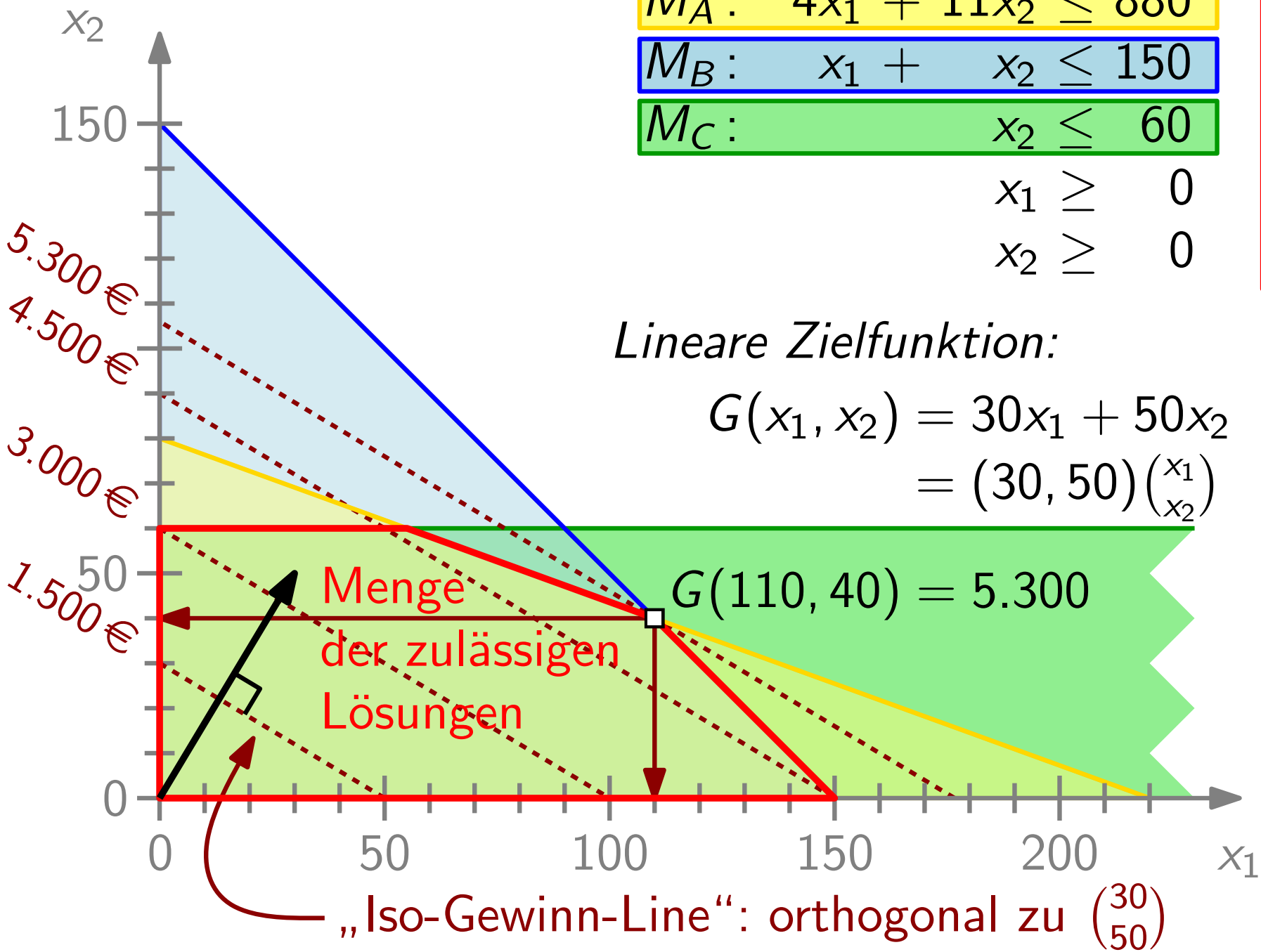
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

$x \geq 0$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

Lineare Beschränkungen:

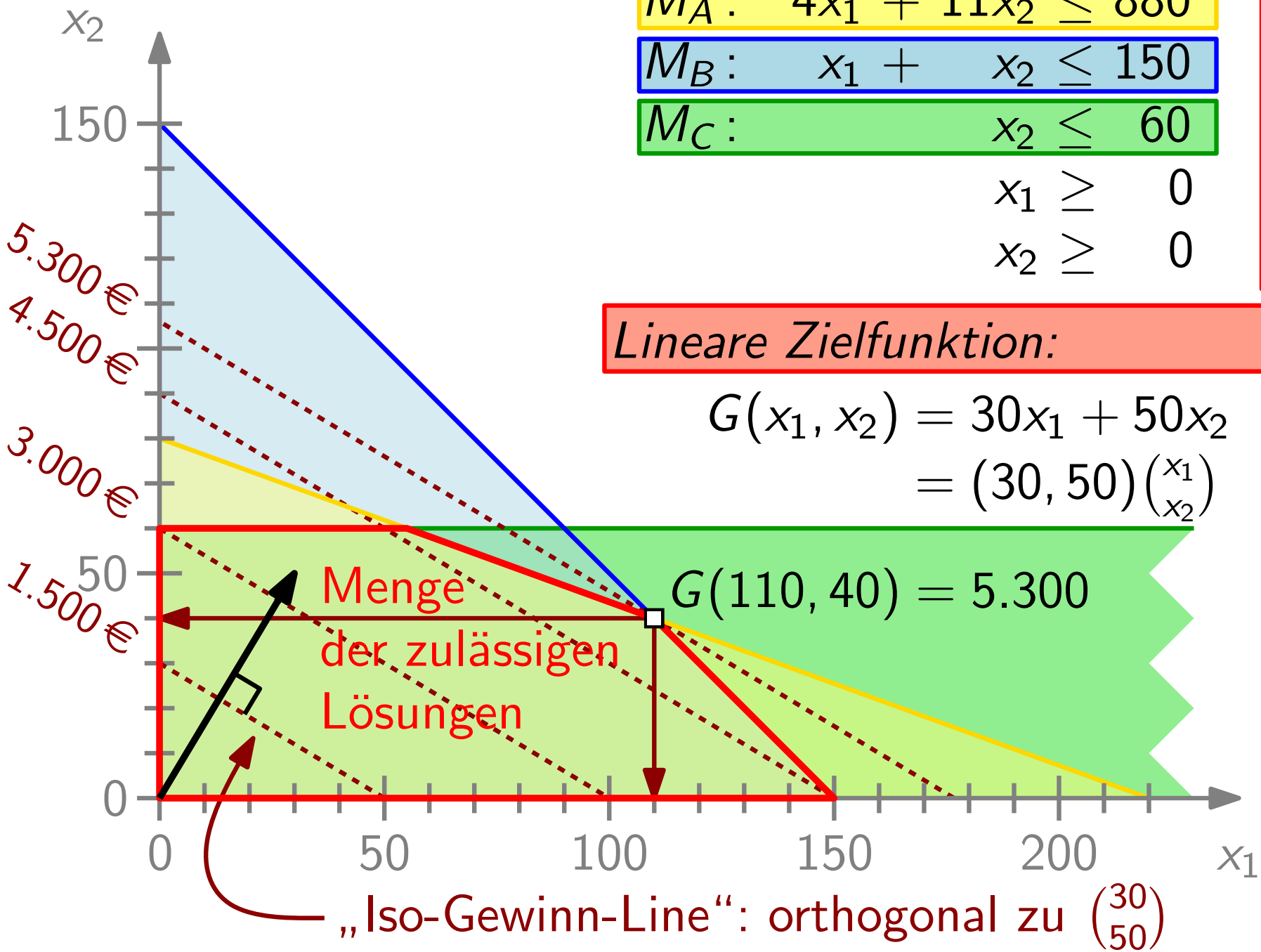
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

$x \geq 0$

Lineare Zielfunktion:

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

Lineare Beschränkungen:

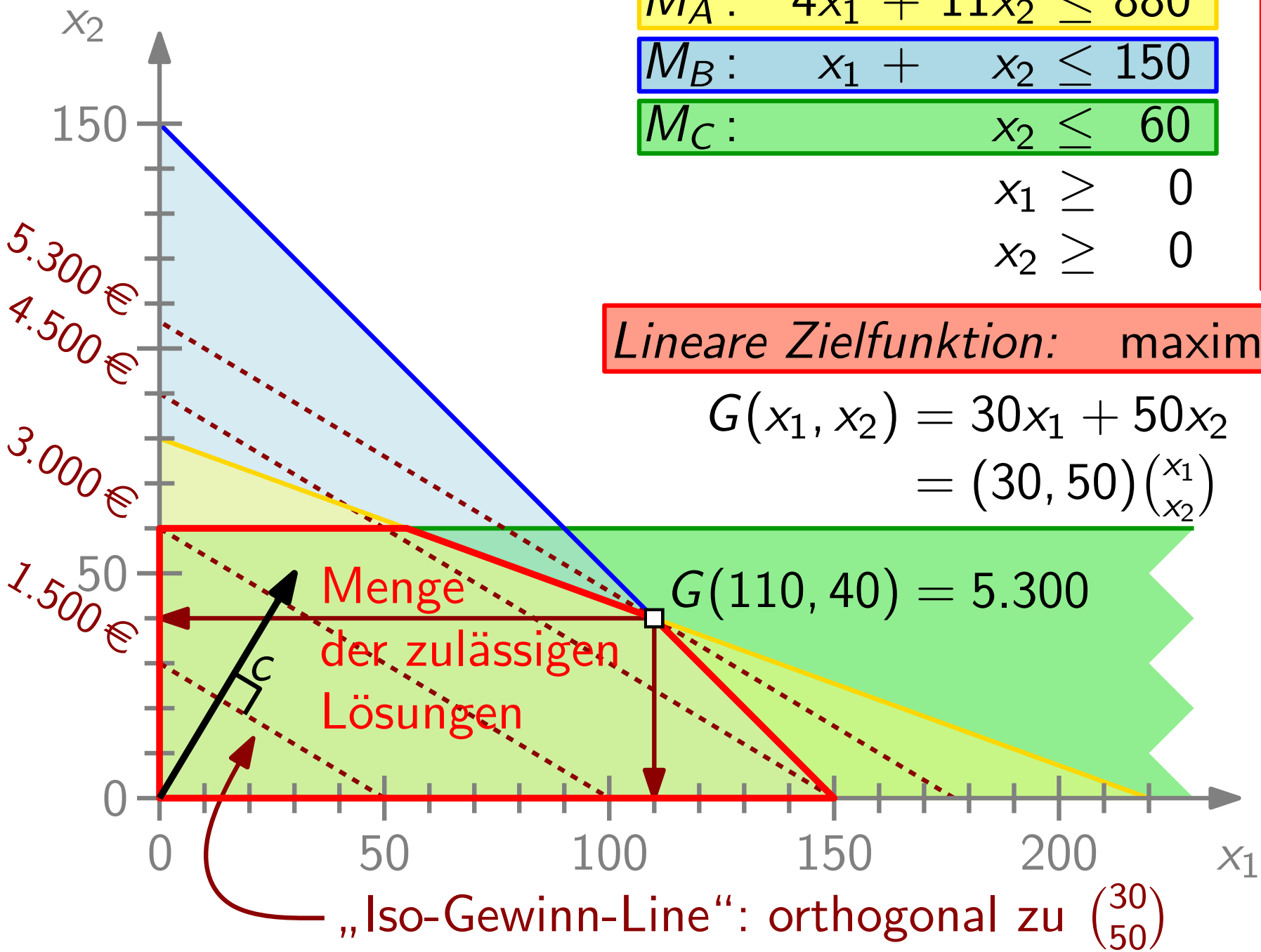
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

$x \geq 0$

Lineare Zielfunktion: maximiere $c^T x$

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

Lineare Beschränkungen:

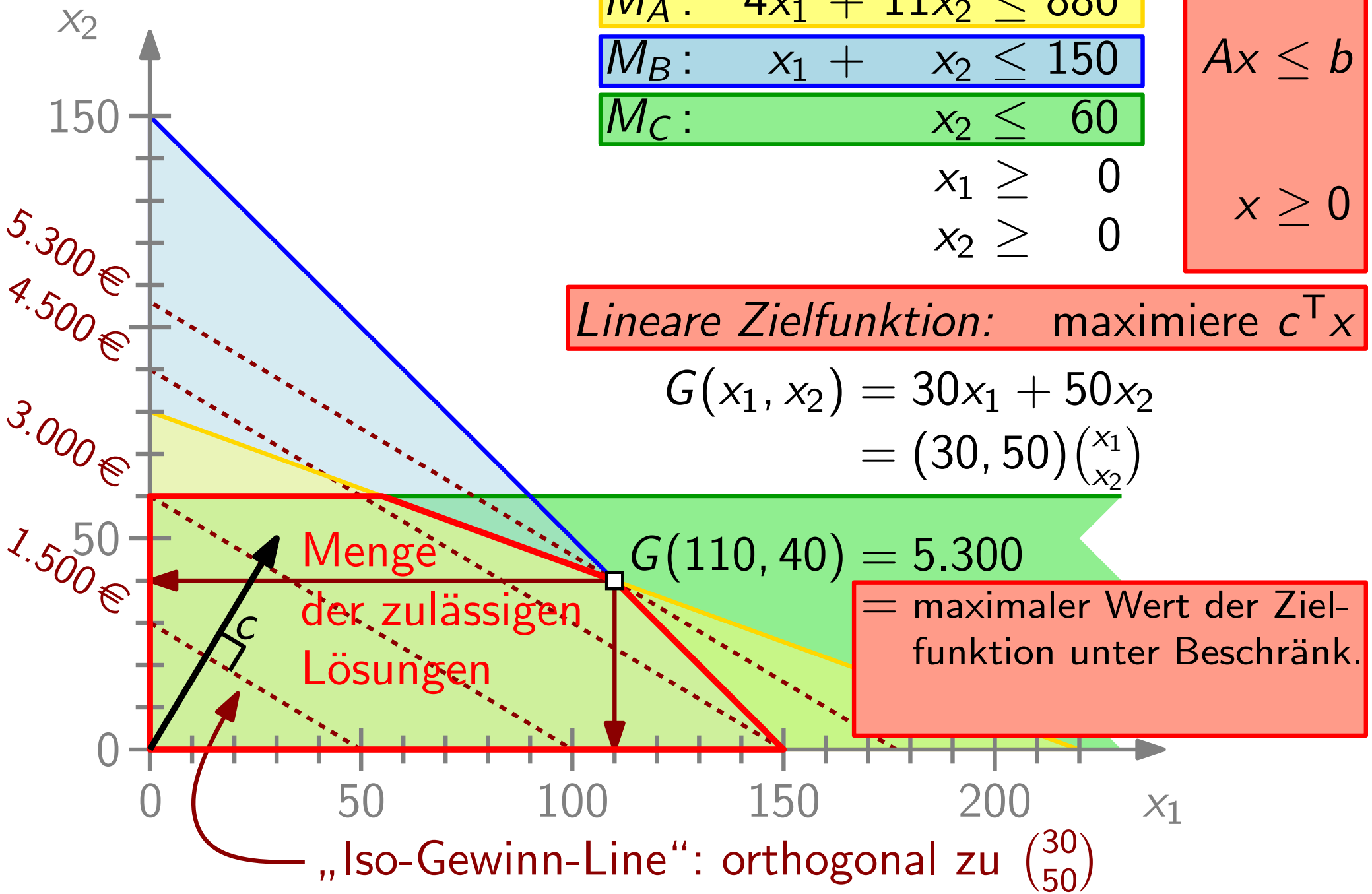
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

$x \geq 0$

Lineare Zielfunktion: maximiere $c^T x$

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lösung

Lineare Beschränkungen:

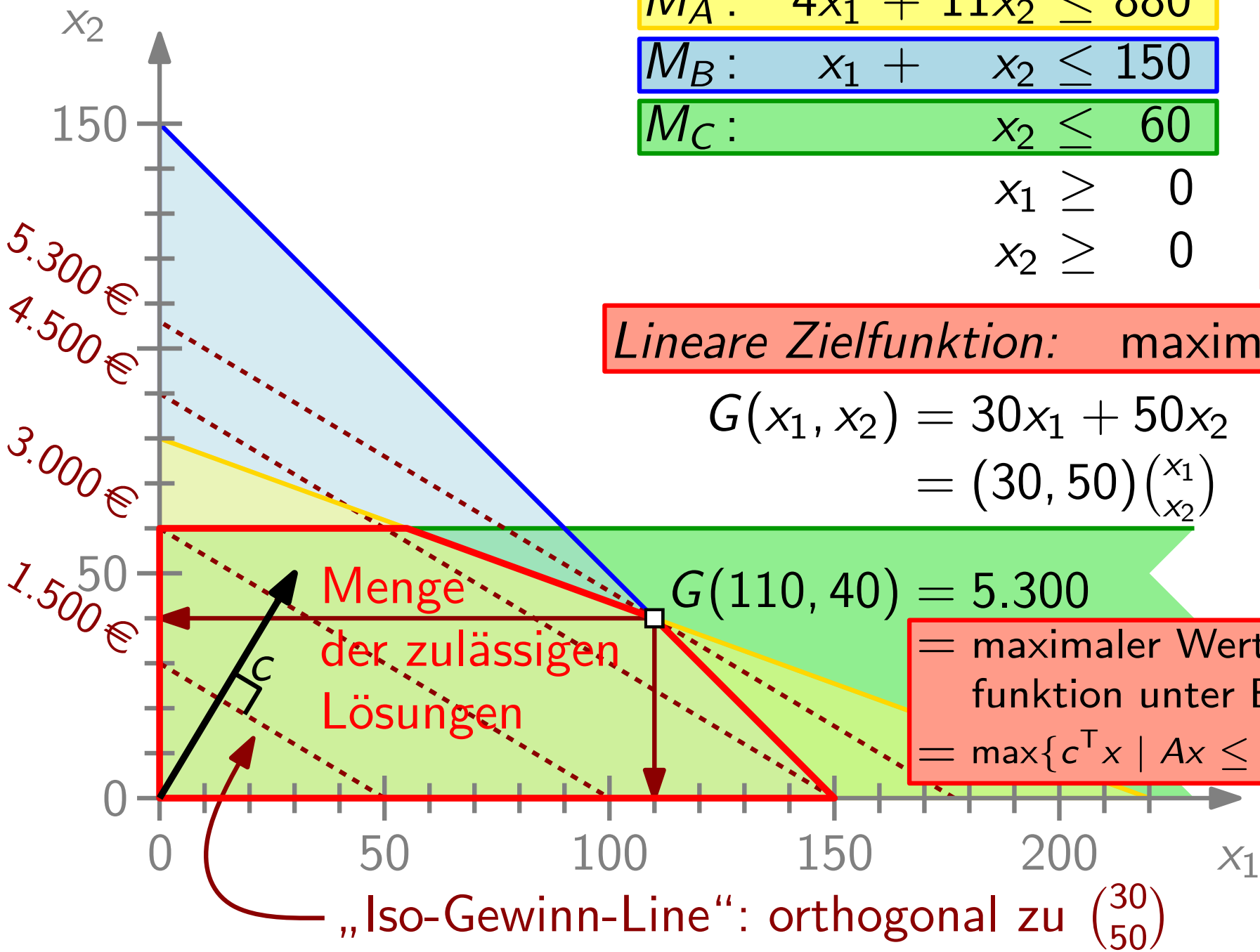
$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$Ax \leq b$

$x \geq 0$

Lineare Zielfunktion: maximiere $c^T x$

$$G(x_1, x_2) = 30x_1 + 50x_2 = (30, 50)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$



Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

Gesucht:

Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$

Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n \quad \text{mit} \quad x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

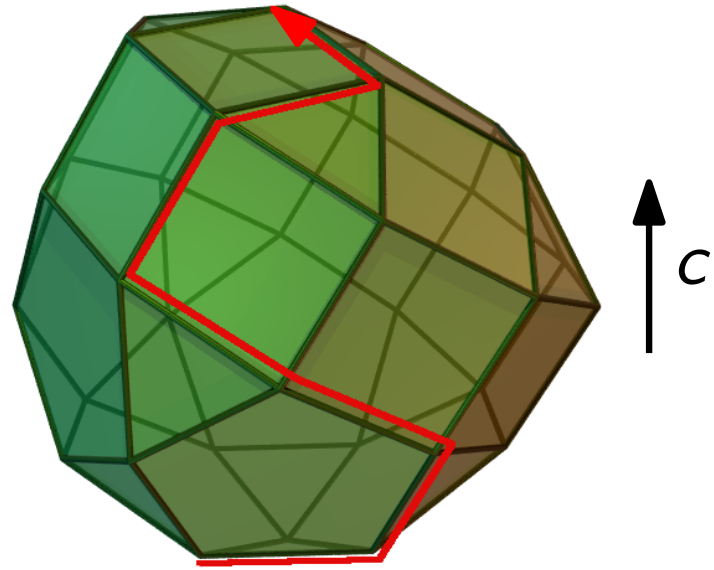
Satz. [Dantzig, 1947]
Der Simplex-Algorithmus löst lineare Programme.

Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

Satz. [Dantzig, 1947]
Der Simplex-Algorithmus löst lineare Programme.

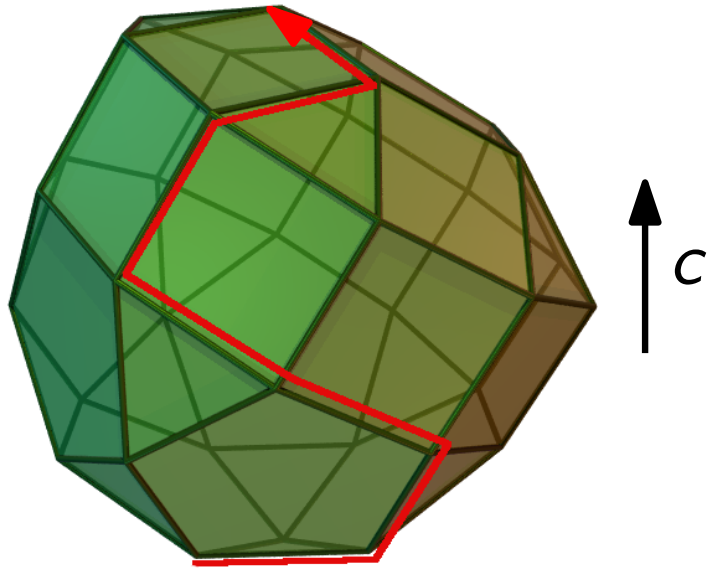


Lineares Programmieren I

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

Satz. [Dantzig, 1947]
Der Simplex-Algorithmus löst lineare Programme.



Satz. [Klee & Minty, 1972]
Es gibt Beispiele, auf denen der Simplex-Algorithmus exponentielle Zeit benötigt.

Lineares Programmieren II

Satz. [Khachiyan, 1979]
Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

Lineares Programmieren II

Satz. [Khachiyan, 1979]
Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei $L =$ Anz. Bits in der Eingabe.

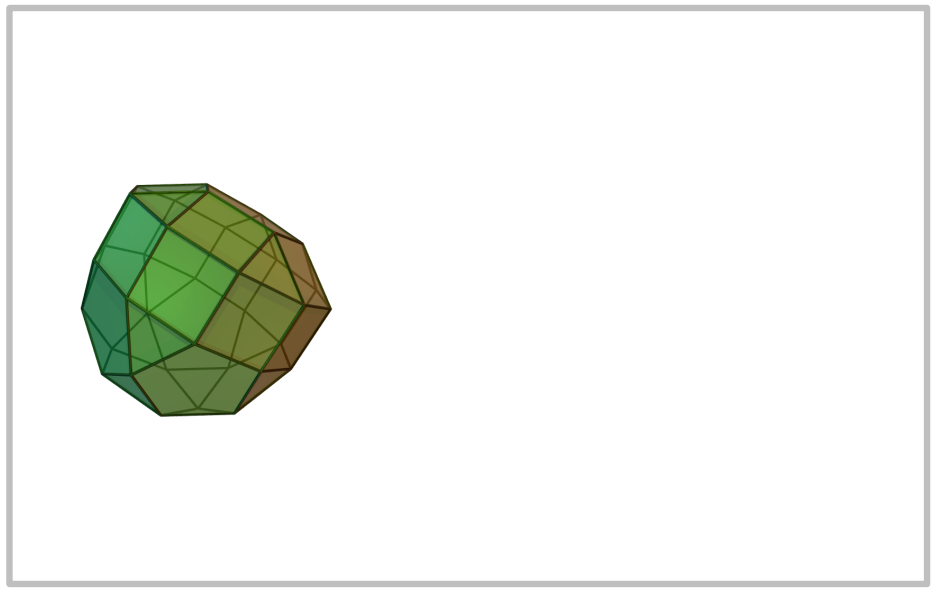
Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

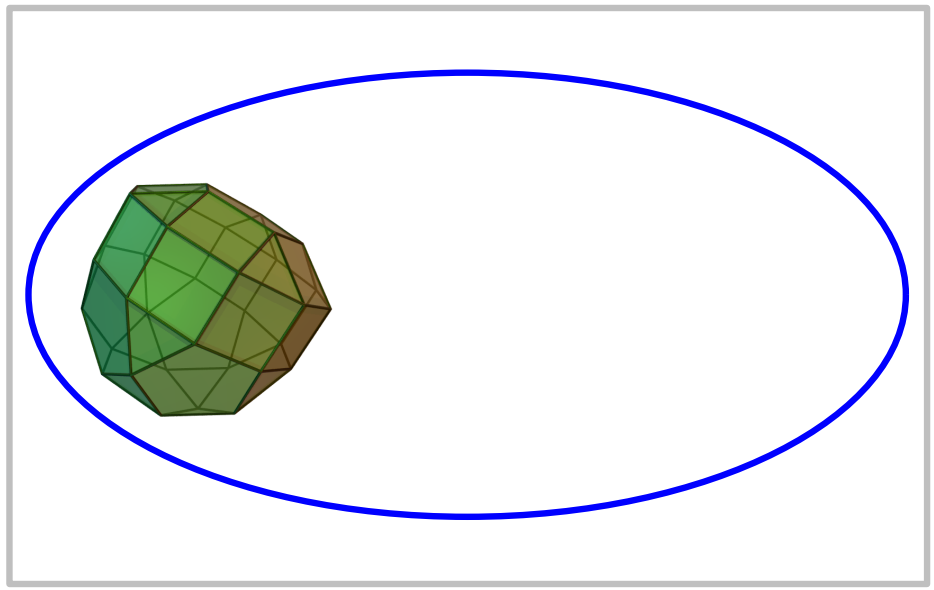


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

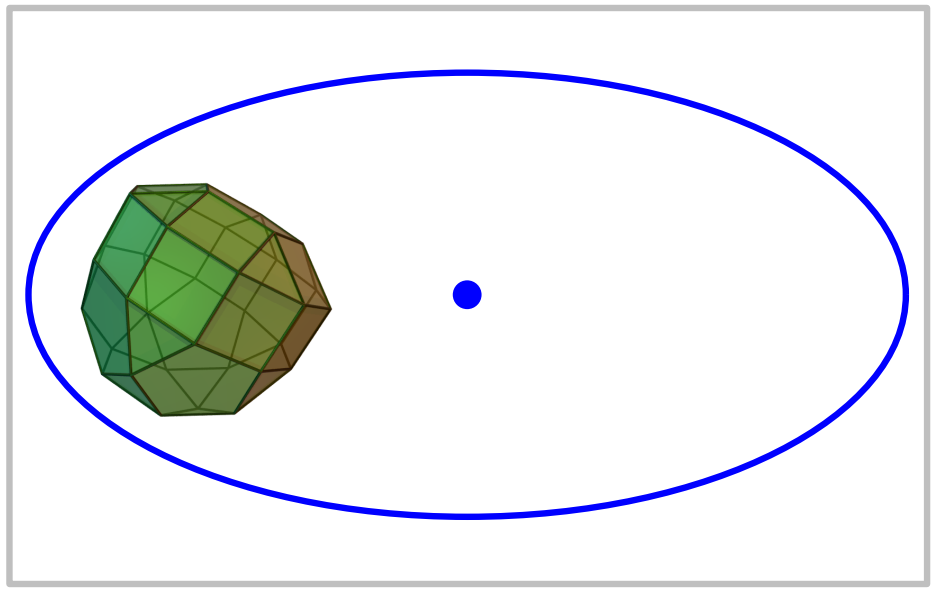


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

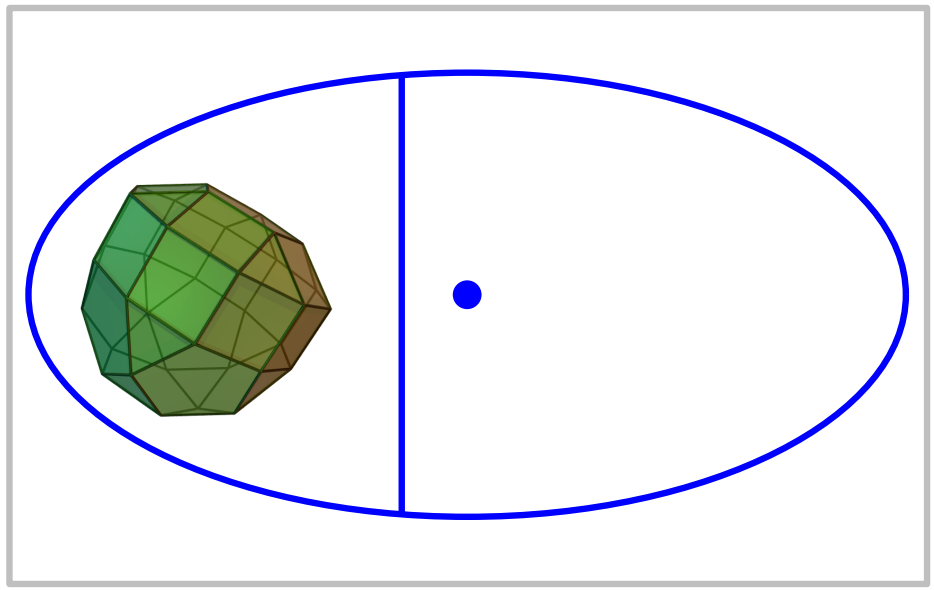


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei $L =$ Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

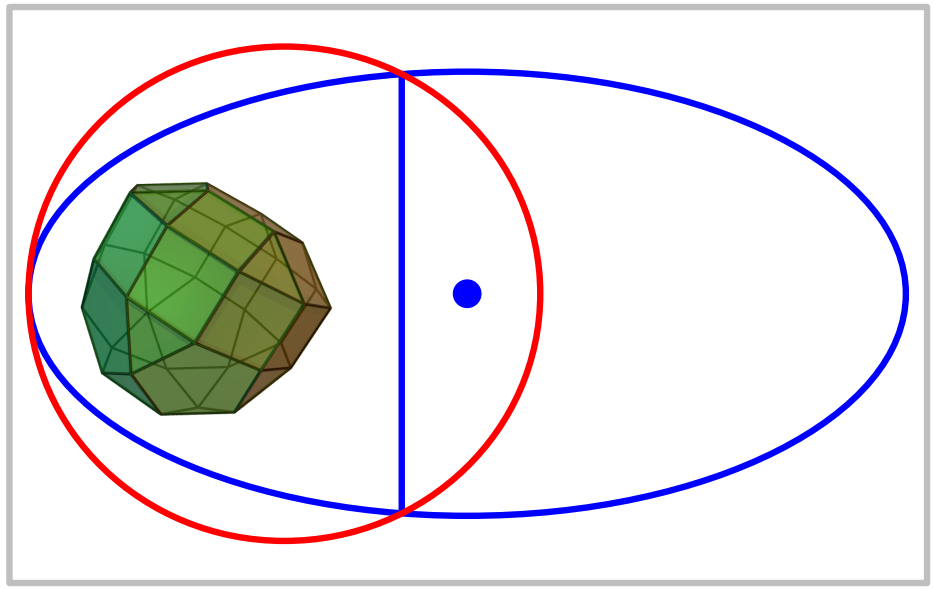


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

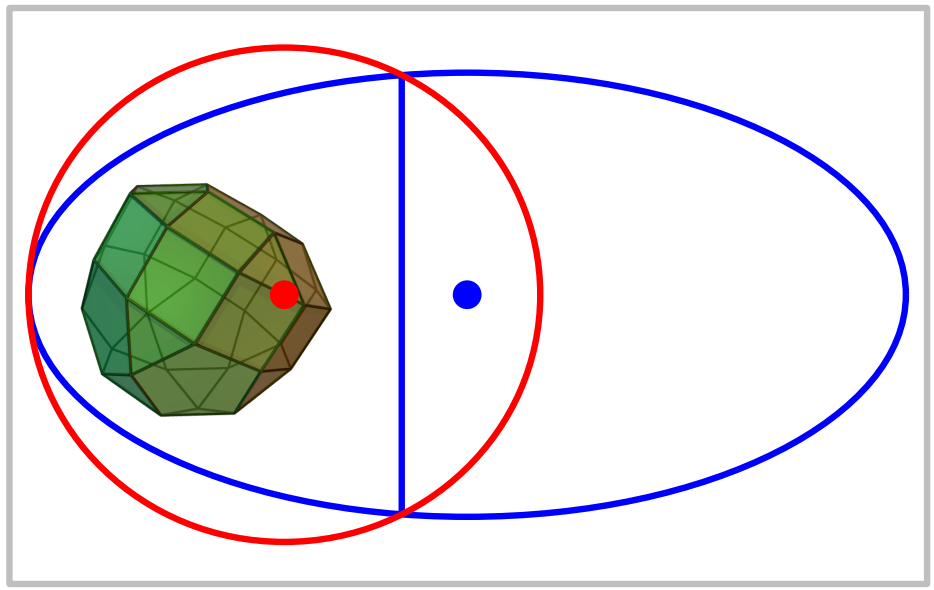


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ

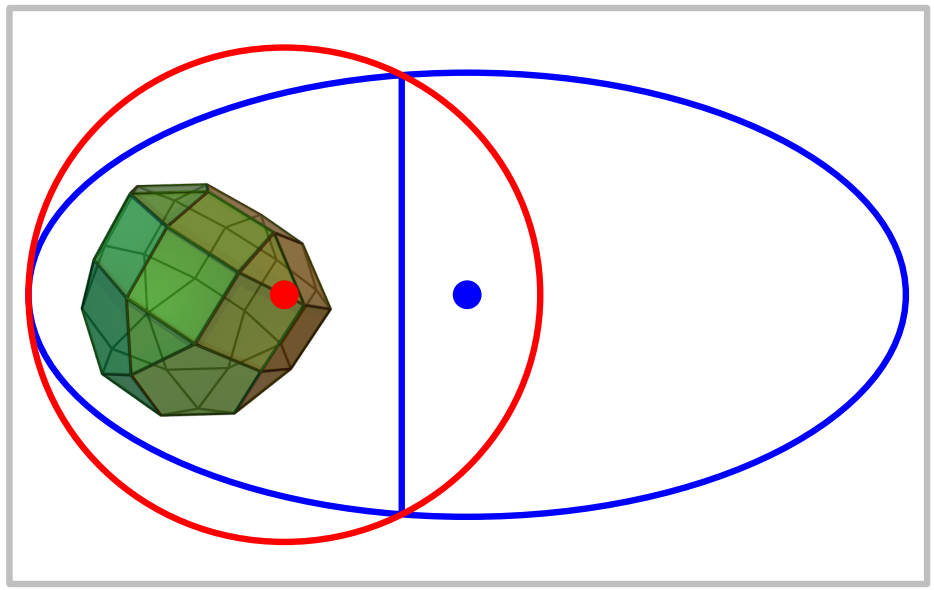


Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ



Narendra Karmarkar
*1957 Gwalior, Indien

Satz. [Karmakar, 1984]

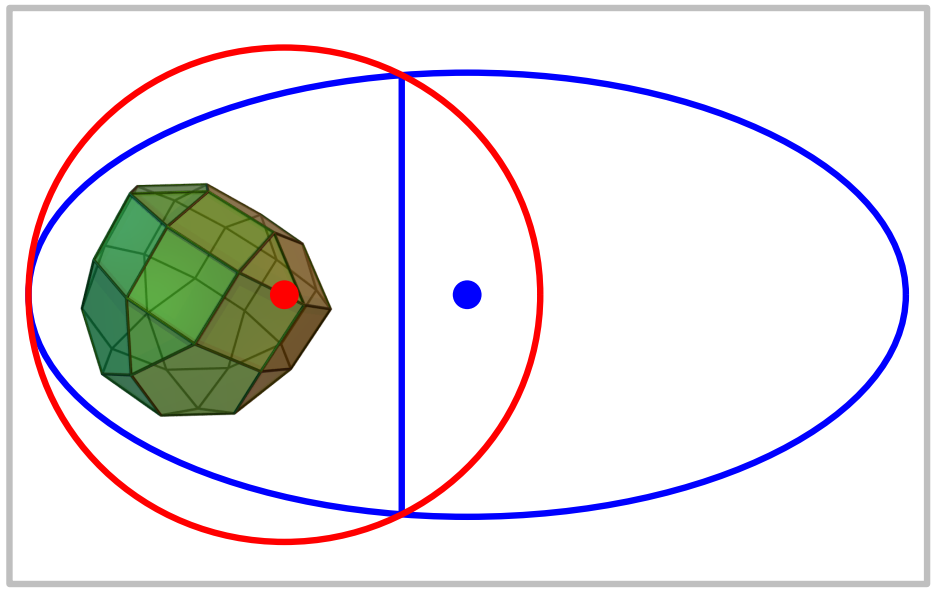
Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^{3.5})$ Zeit *numerisch stabil* lösen.

Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ



Narendra Karmarkar
*1957 Gwalior, Indien

Satz. [Karmakar, 1984] *Innere-Punkt-Methode*

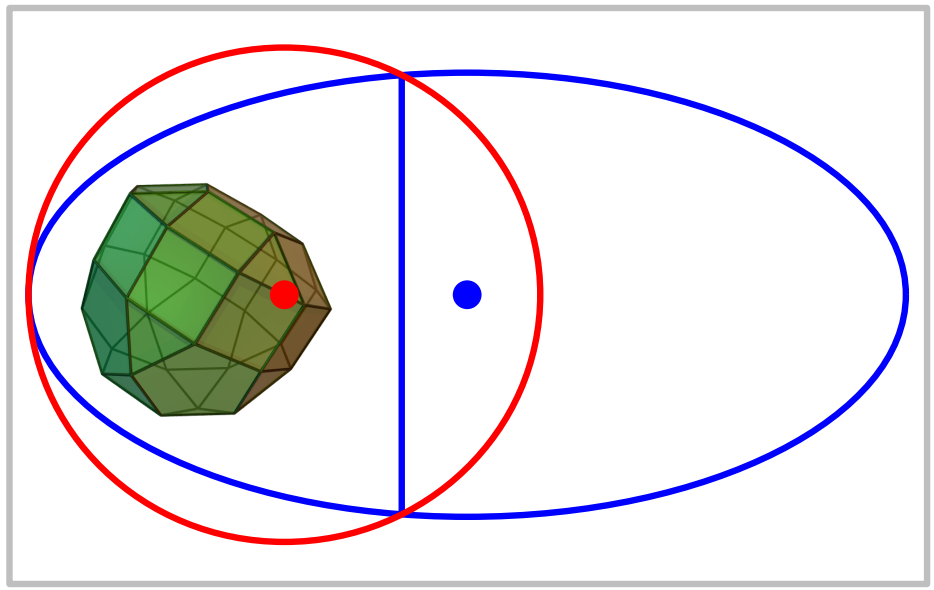
Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^{3.5})$ Zeit *numerisch stabil* lösen.

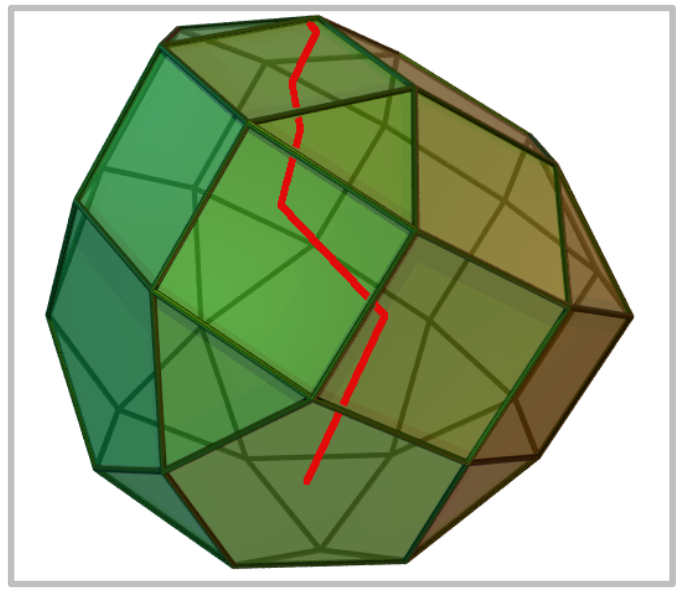
Lineares Programmieren II

Satz. [Khachiyan, 1979] *Ellipsoidmethode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^6)$ Zeit lösen, wobei L = Anz. Bits in der Eingabe.

Leonid Khachiyan
*1952 Leningrad
†2005 South Brunswick, NJ





Narendra Karmarkar
*1957 Gwalior, Indien

Satz. [Karmakar, 1984] *Innere-Punkt-Methode*

Ein Lineares Programm der Dim. n lässt sich in $O(L^2 \cdot n^{3.5})$ Zeit *numerisch stabil* lösen.

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n \quad \text{mit} \quad x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

Ganzzahlige lineare Programmierung (ILP)

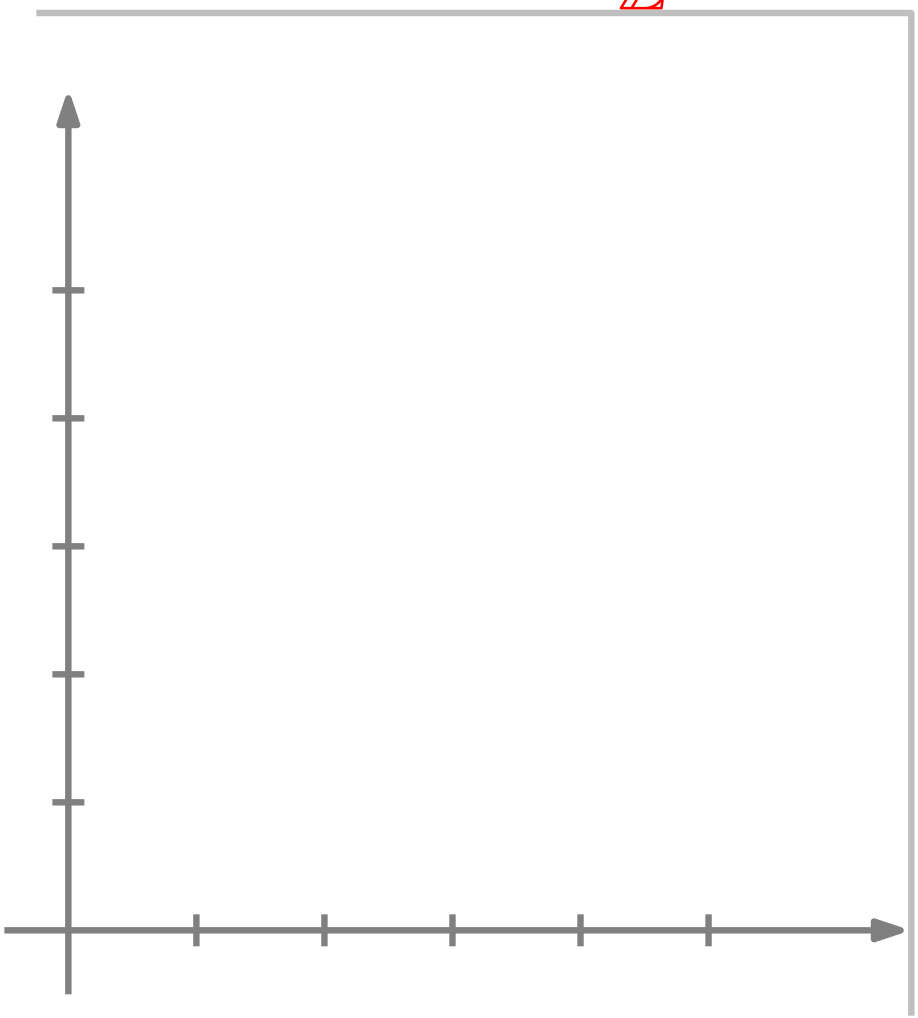
Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

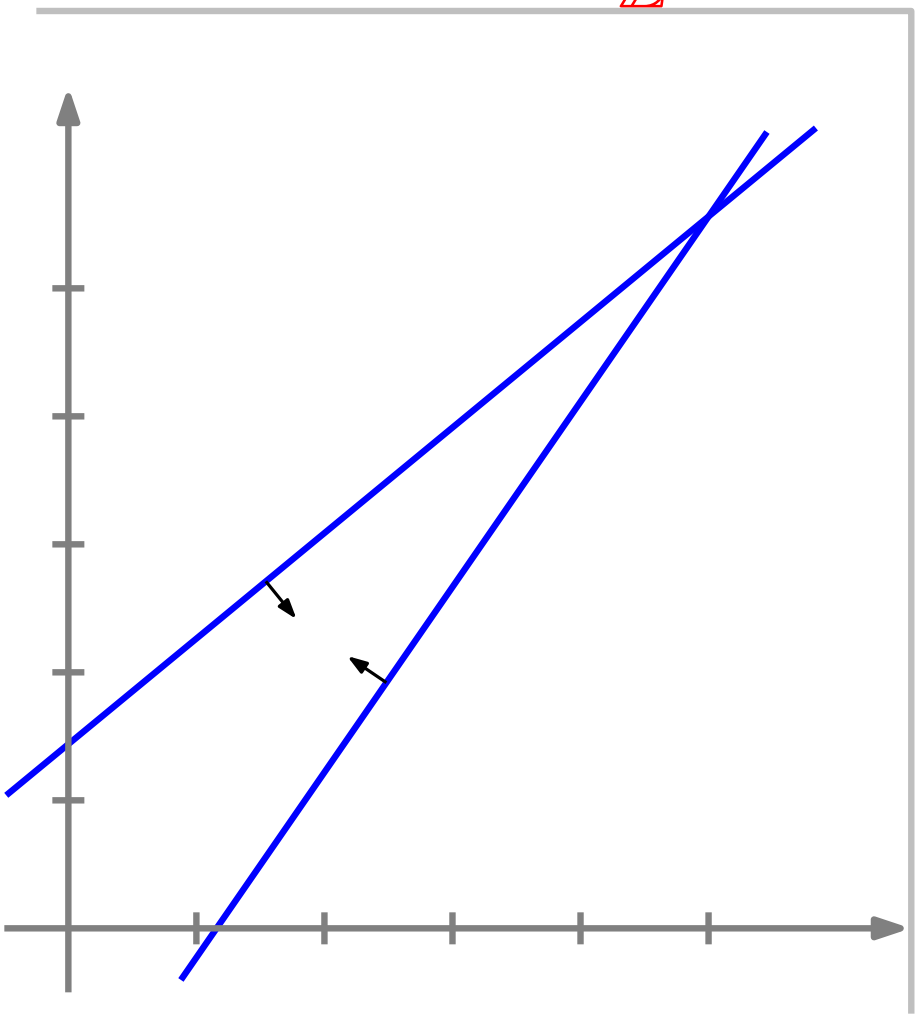
Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

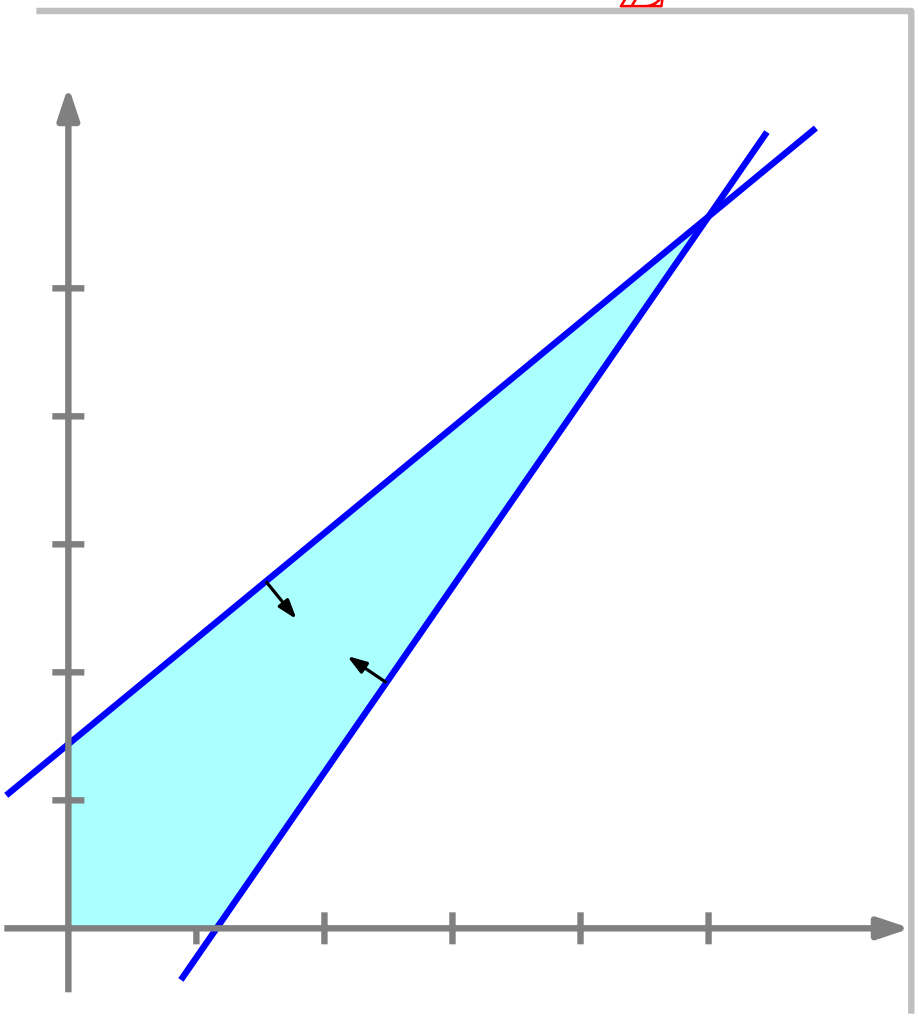
Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

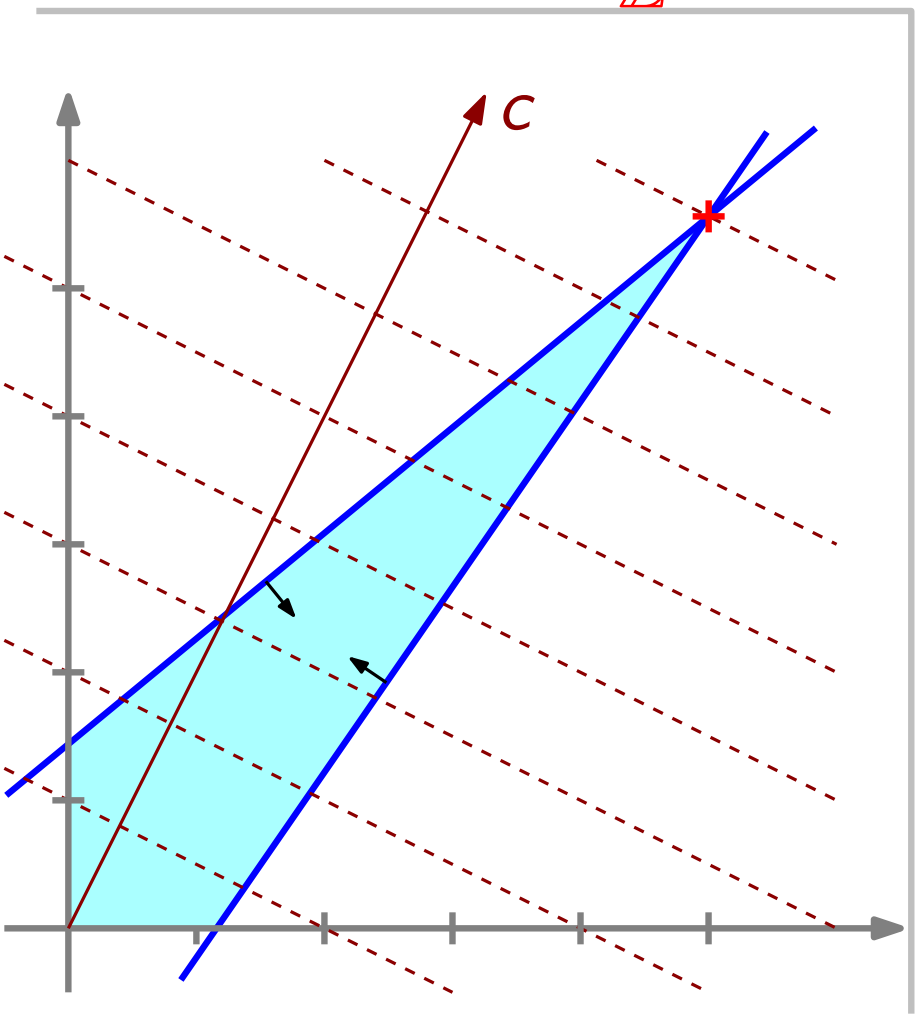
Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

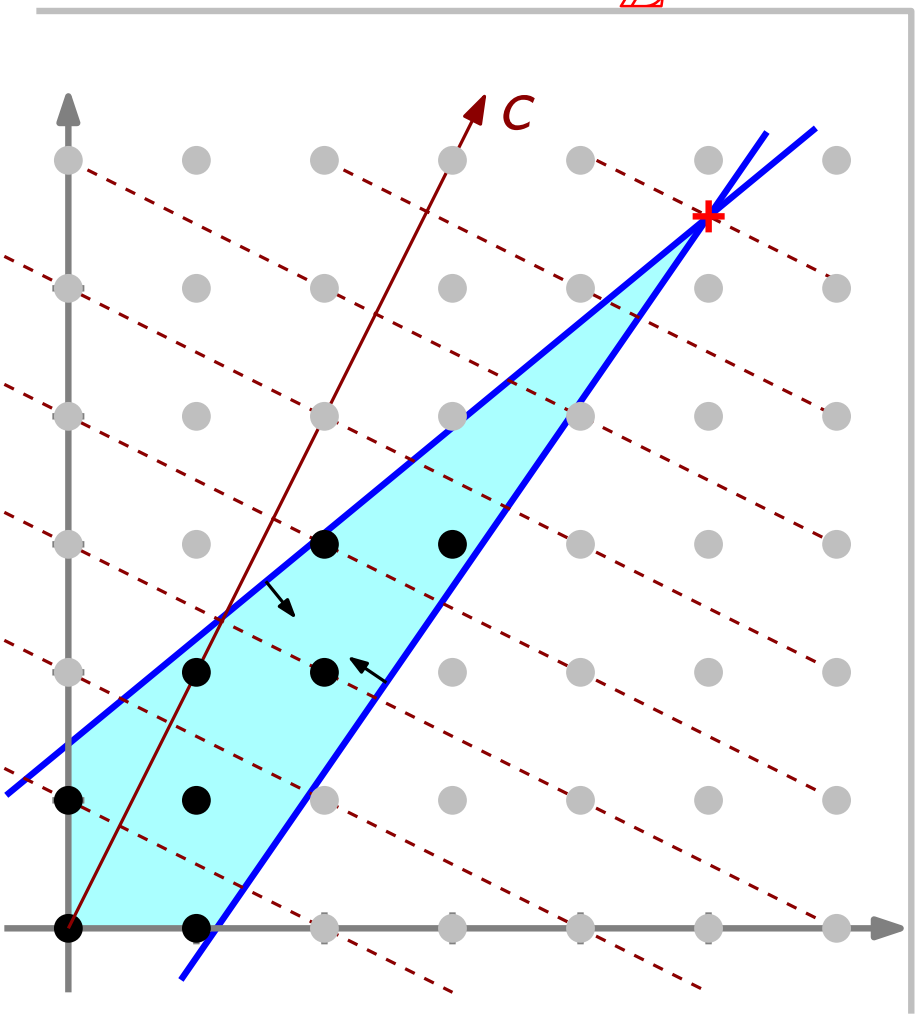
Gesucht: $x^\star \in \mathbb{R}^n$ (durchgestrichen, $x \in \mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

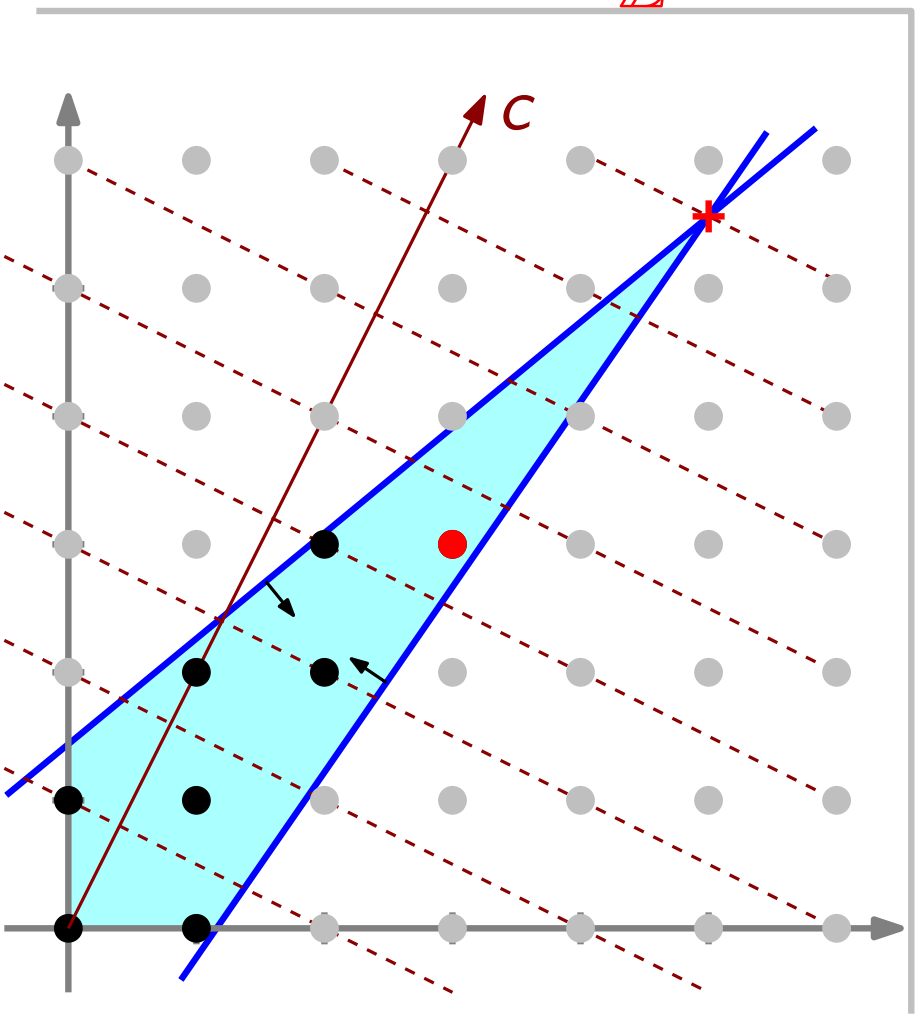
Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^n$

Gesucht: $x^\star \in \cancel{\mathbb{R}^n}\ \mathbb{Z}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

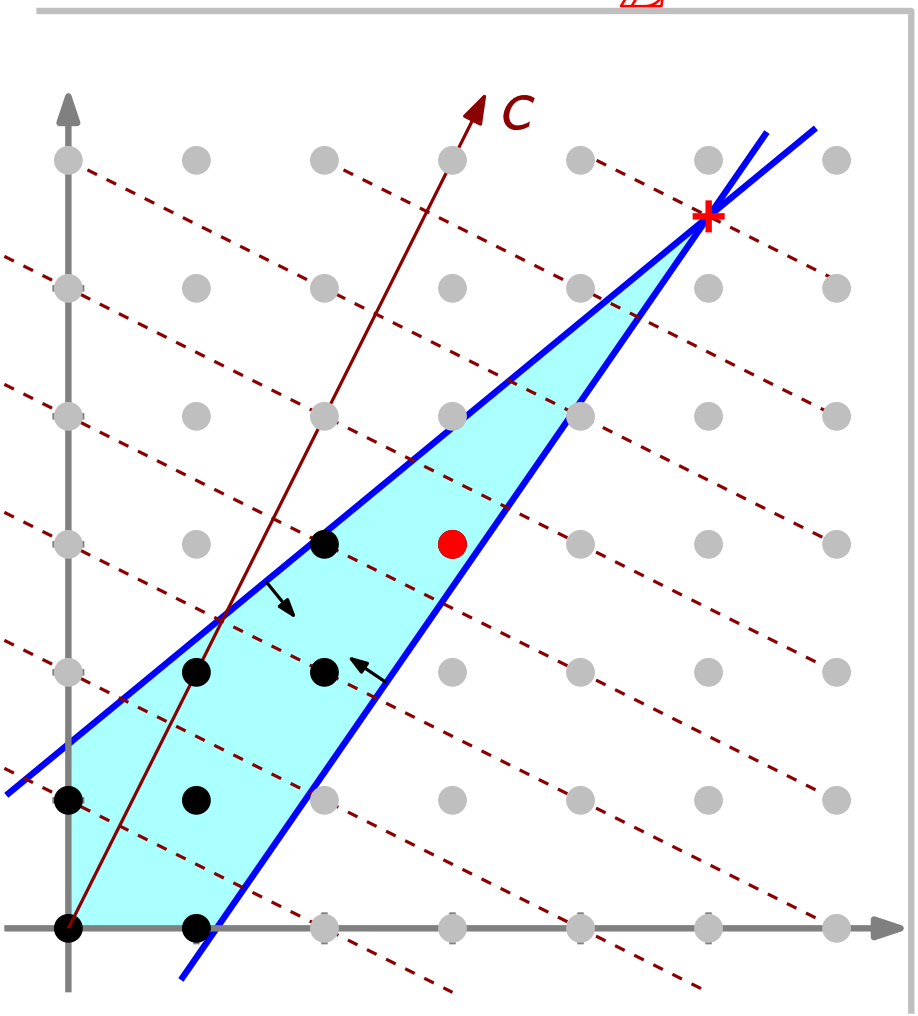
Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

häufig $\{0,1\}^n$

$\mathbb{Z}^n$

$x \in \mathbb{Z}^n$

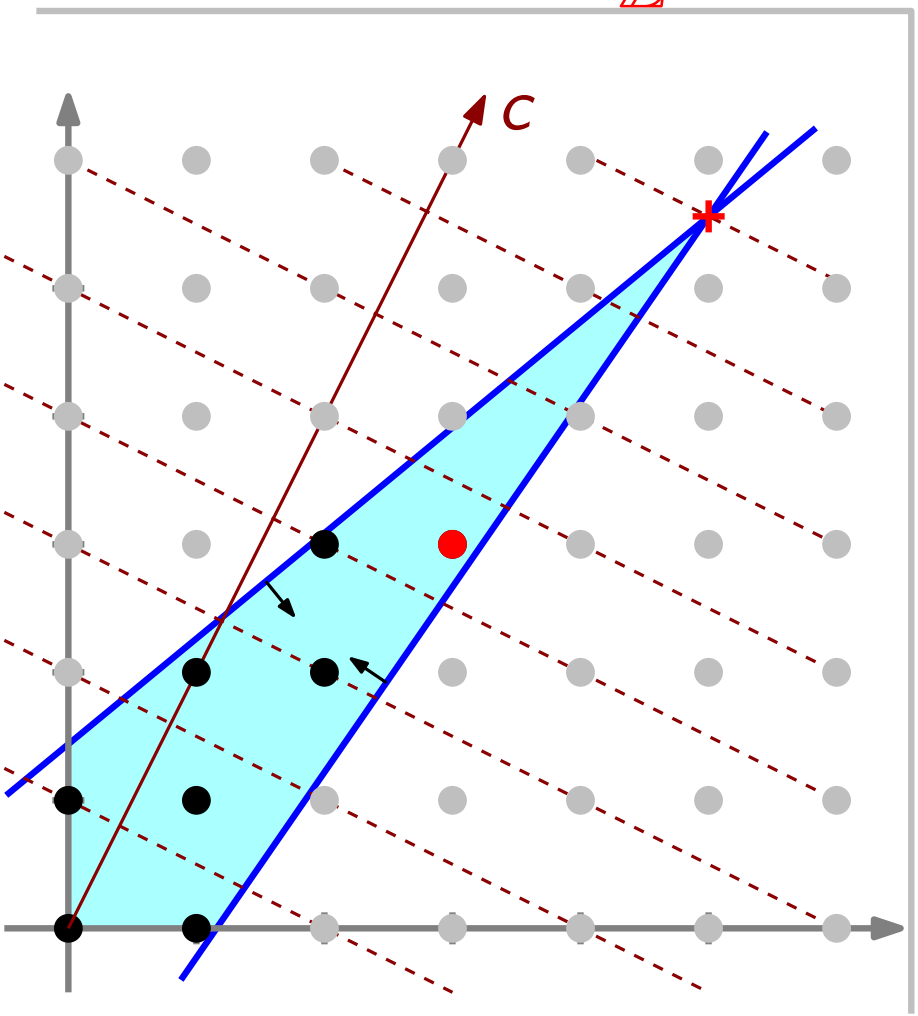


Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$

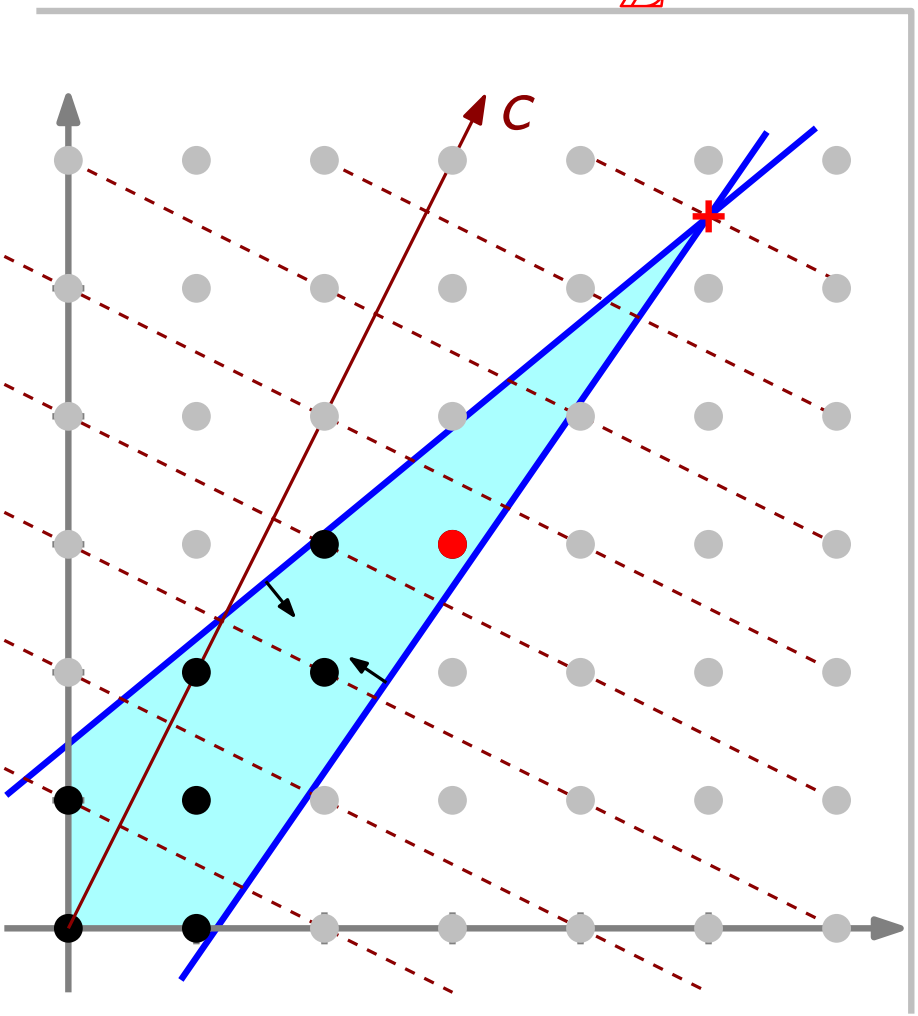
häufig $\{0,1\}^n$ oder $\mathbb{Z}^n$ $\geq$ $x \in \mathbb{Z}^n$



Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$

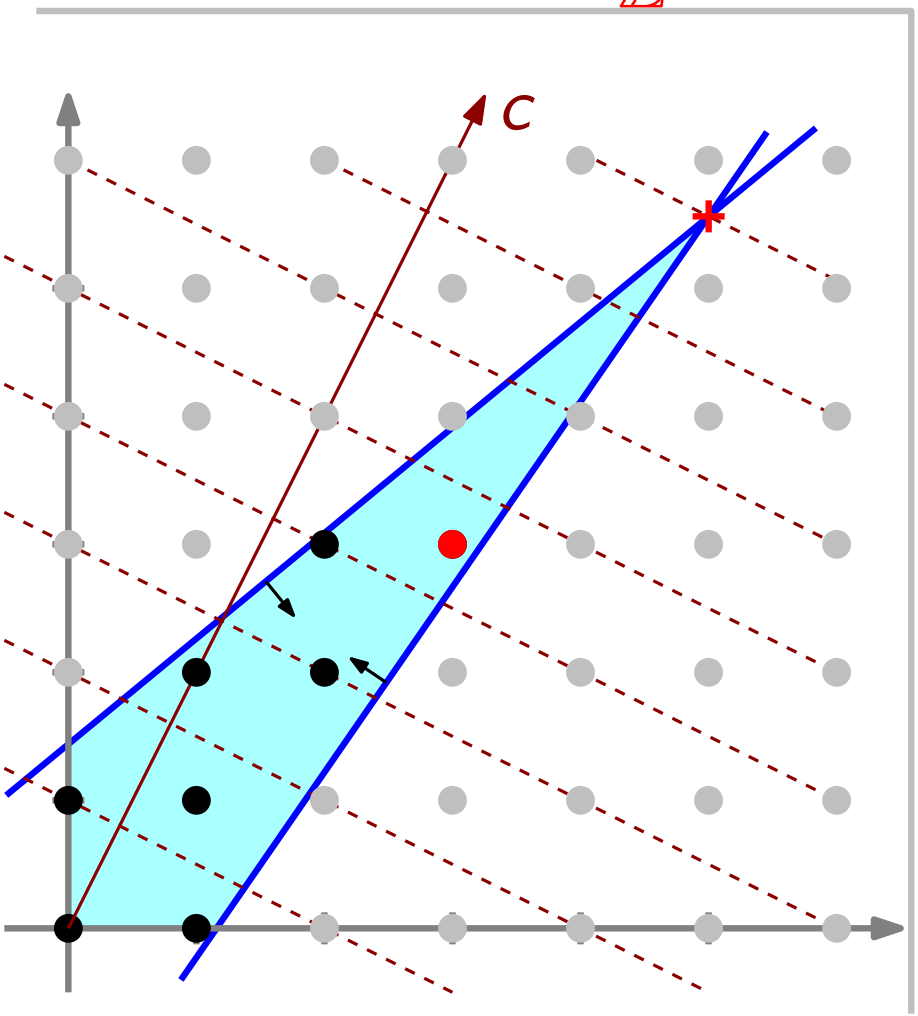
Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

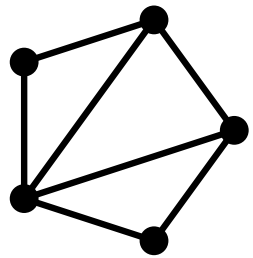
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



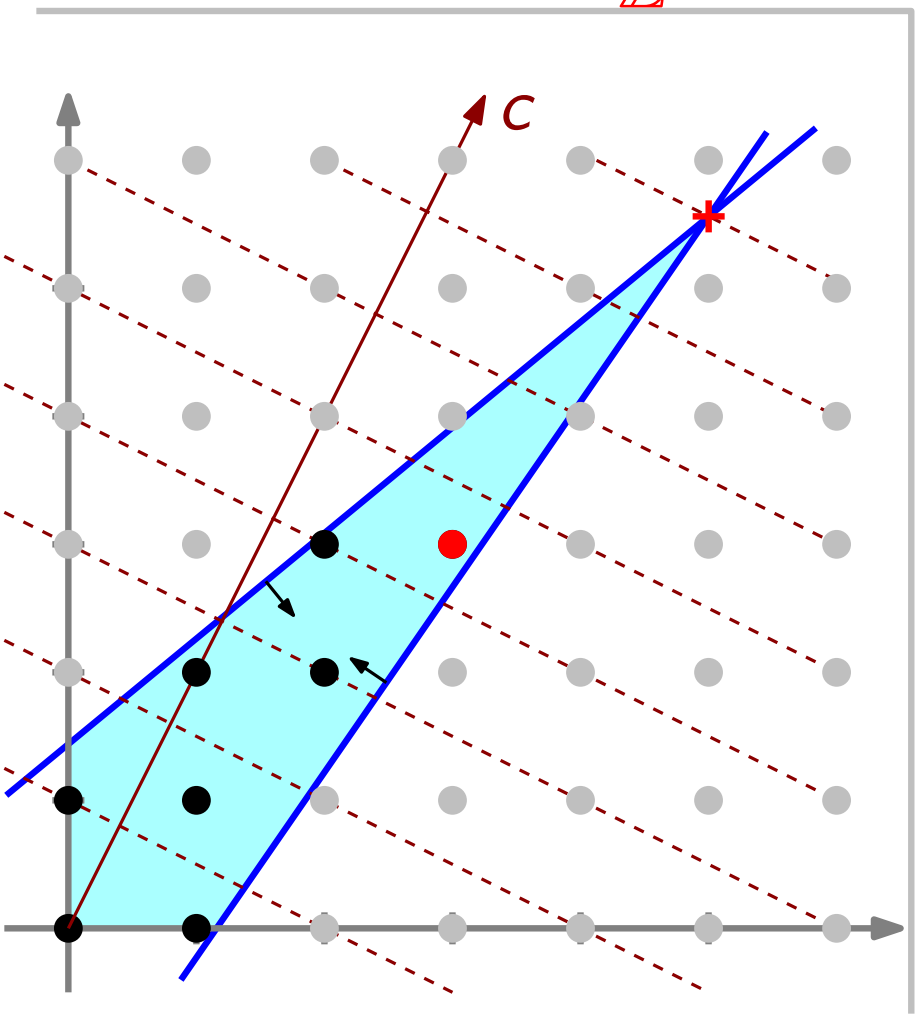
Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

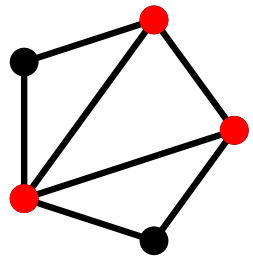
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



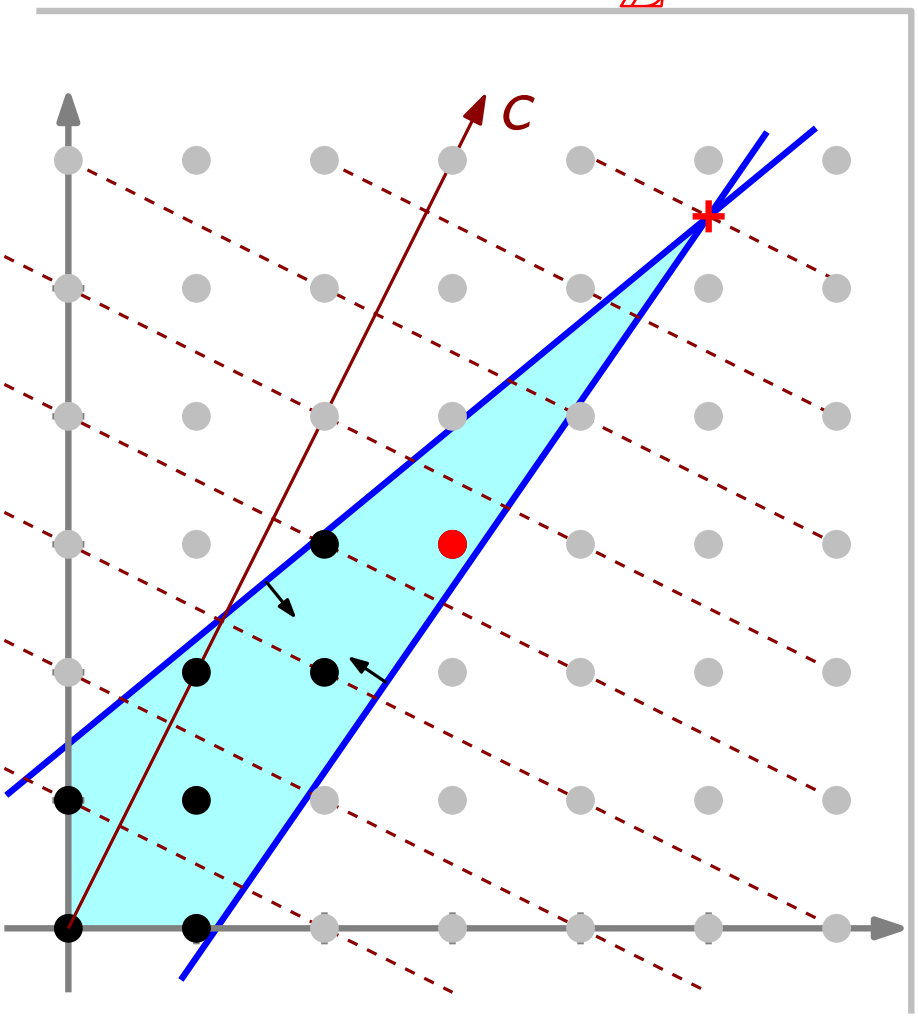
Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

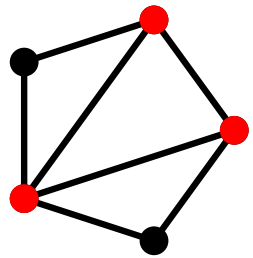
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

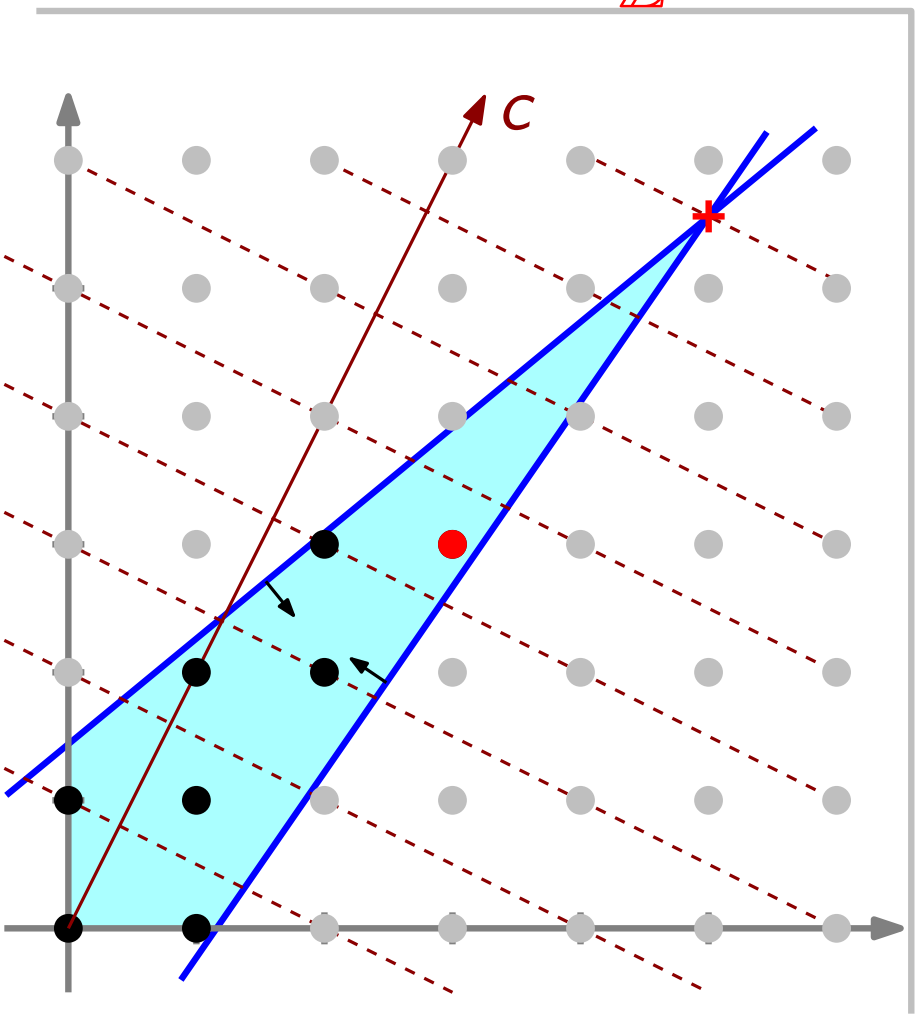
Ziel: $|V'|$ minimal!

Aufgabe. Modellieren Sie dieses Problem als ILP!

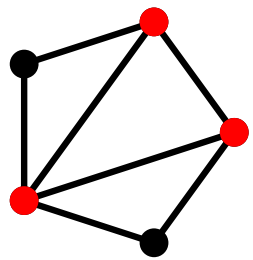
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

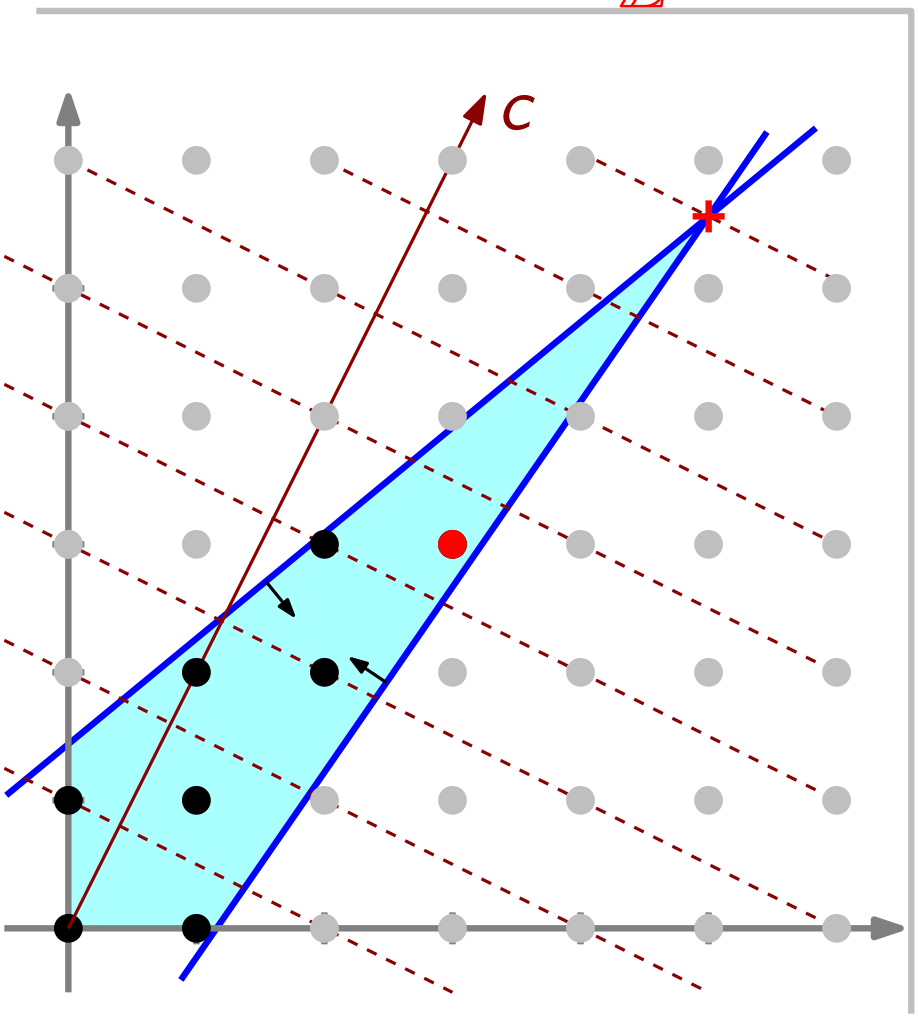
Aufgabe. Modellieren Sie dieses Problem als ILP!

Tipp. Verwenden Sie für jeden Knoten v eine Variable $x_v \in \{0,1\}$, mit $x_v = 1 \Leftrightarrow v \in V'$

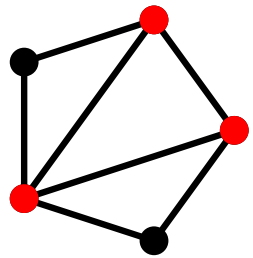
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ ($\geq$ oder $\arg\min$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

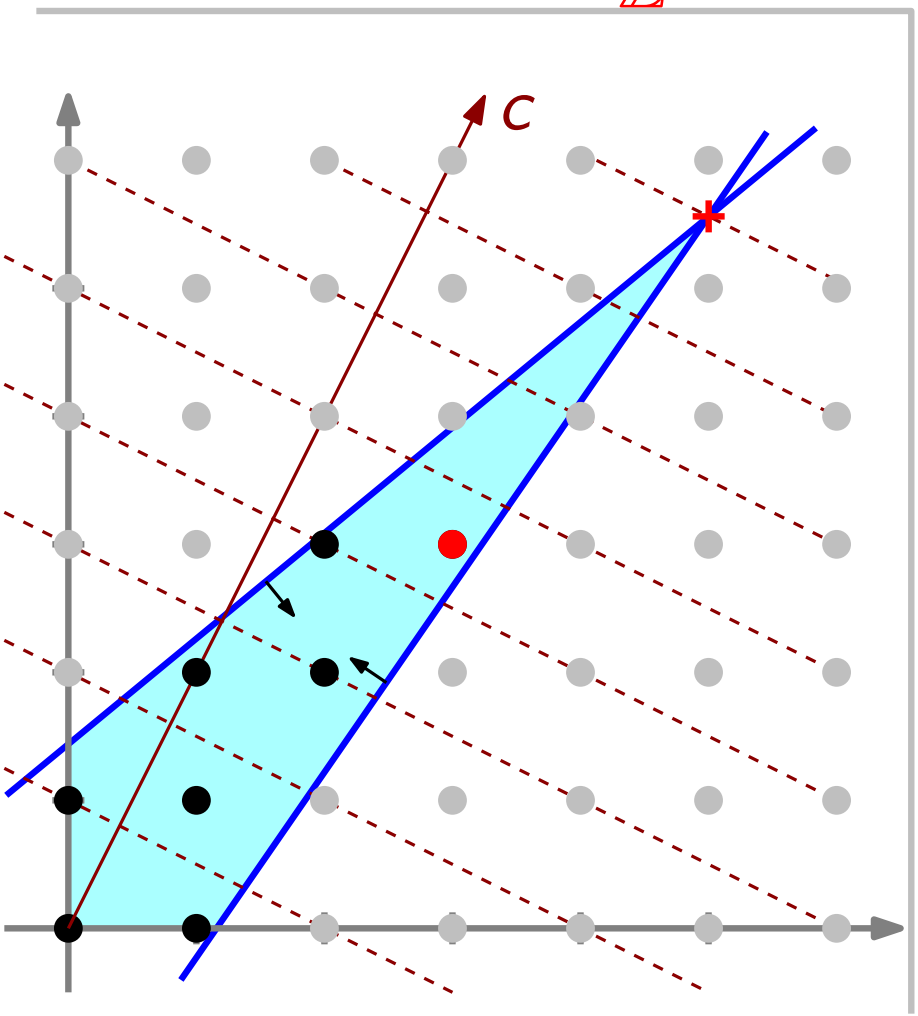
Ziel: $|V'|$ minimal!

Modell. Für $v \in V$ sei $x_v \in \{0, 1\}$.

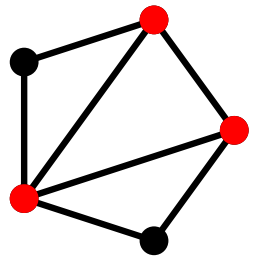
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ ($\geq$ oder $\arg\min$; $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

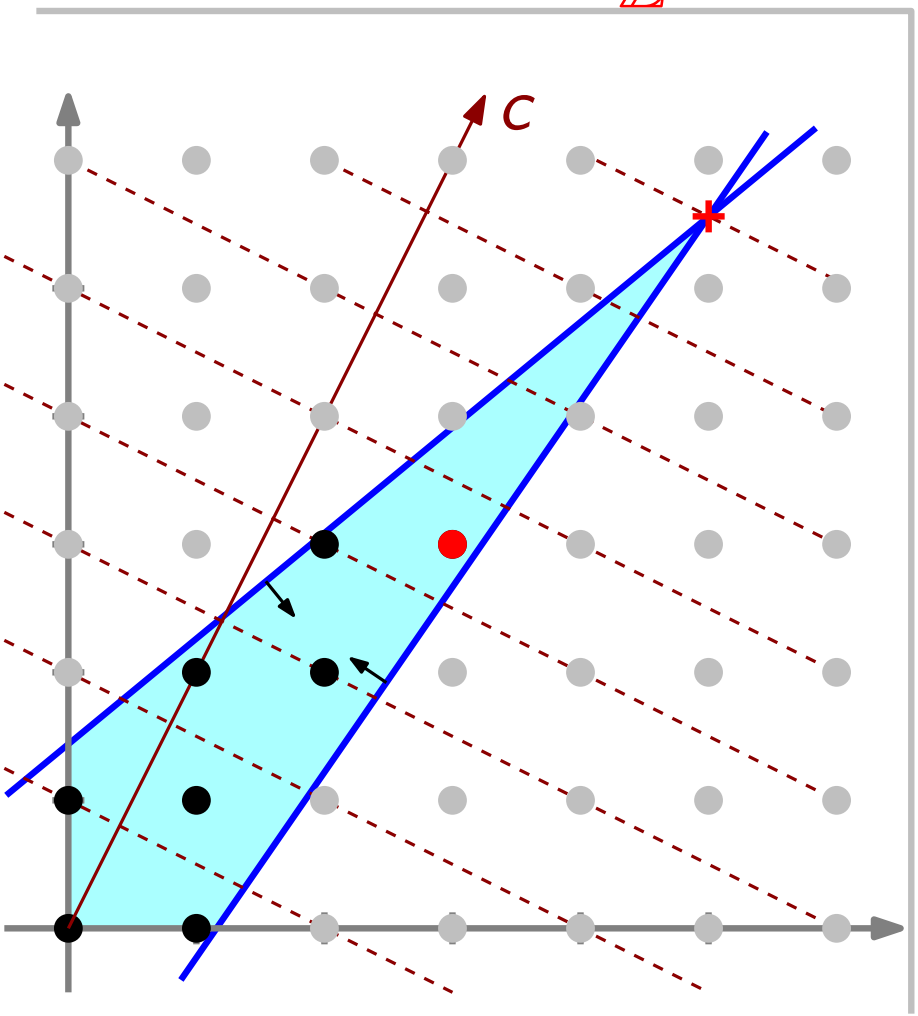
Modell. Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel:

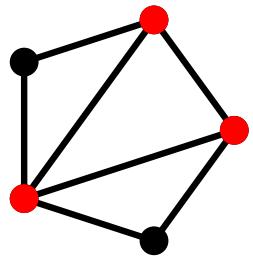
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ ($\geq$ oder $\arg\min$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

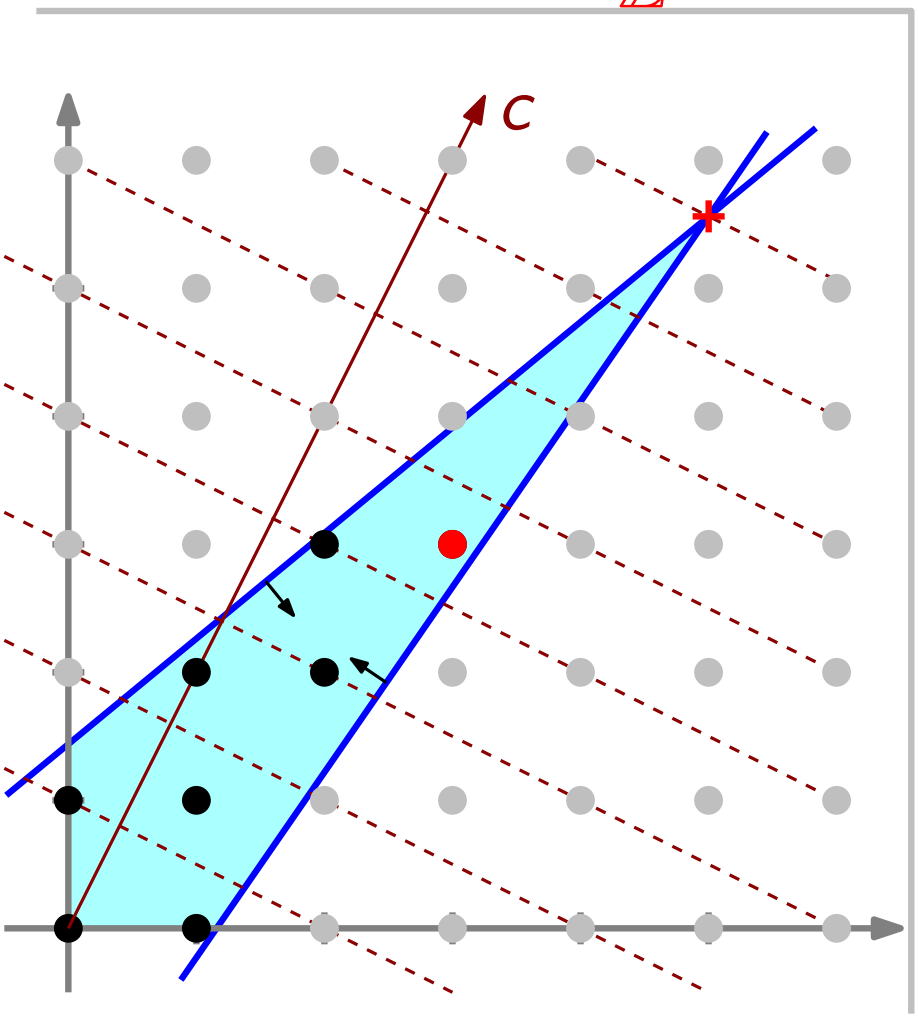
Modell. Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel: minimiere $\sum_{v \in V} x_v$

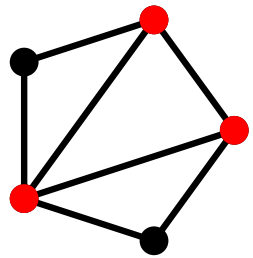
Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ ($\geq$ oder $\arg\min$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$



Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

Modell. Für $v \in V$ sei $x_v \in \{0,1\}$.

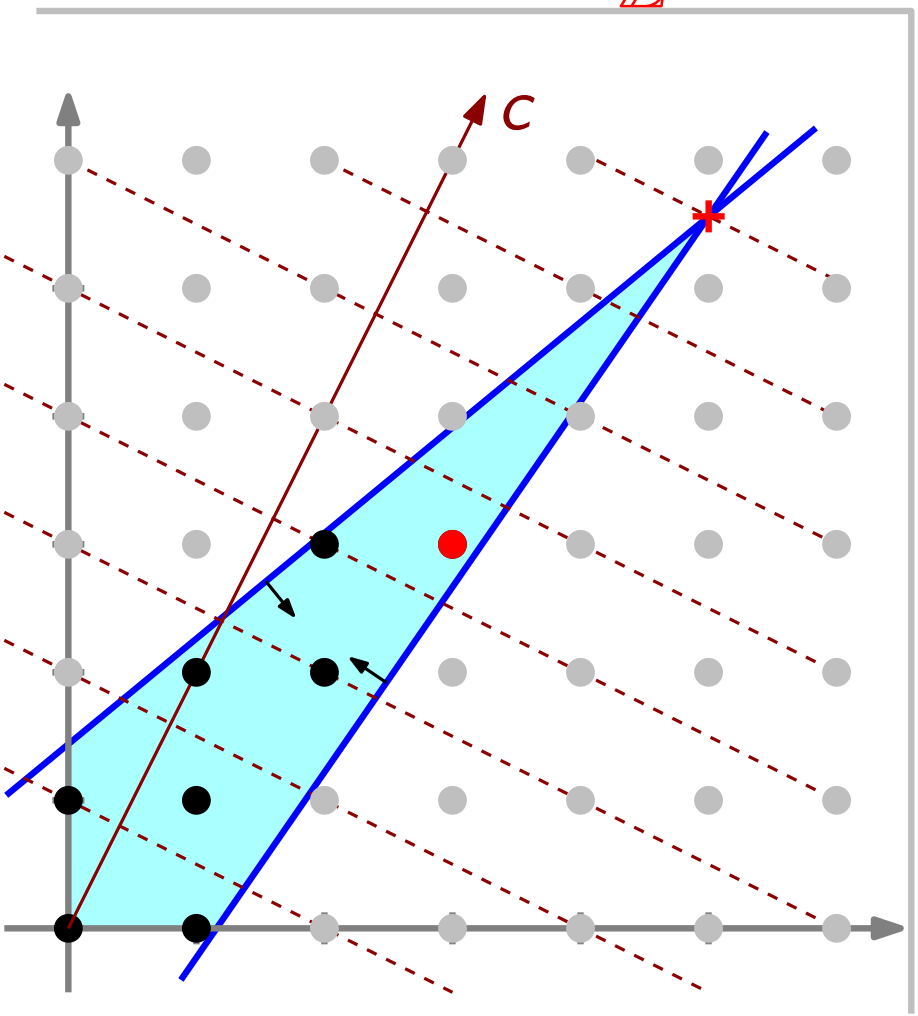
Ziel: minimiere $\sum_{v \in V} x_v$

Beschränkungen:

Ganzzahlige lineare Programmierung (ILP)

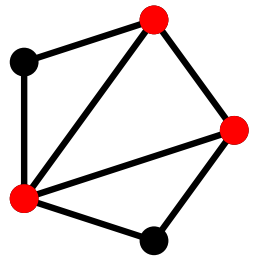
Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$

Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.



Ziel: $|V'|$ minimal!

Modell. Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel: minimiere $\sum_{v \in V} x_v$

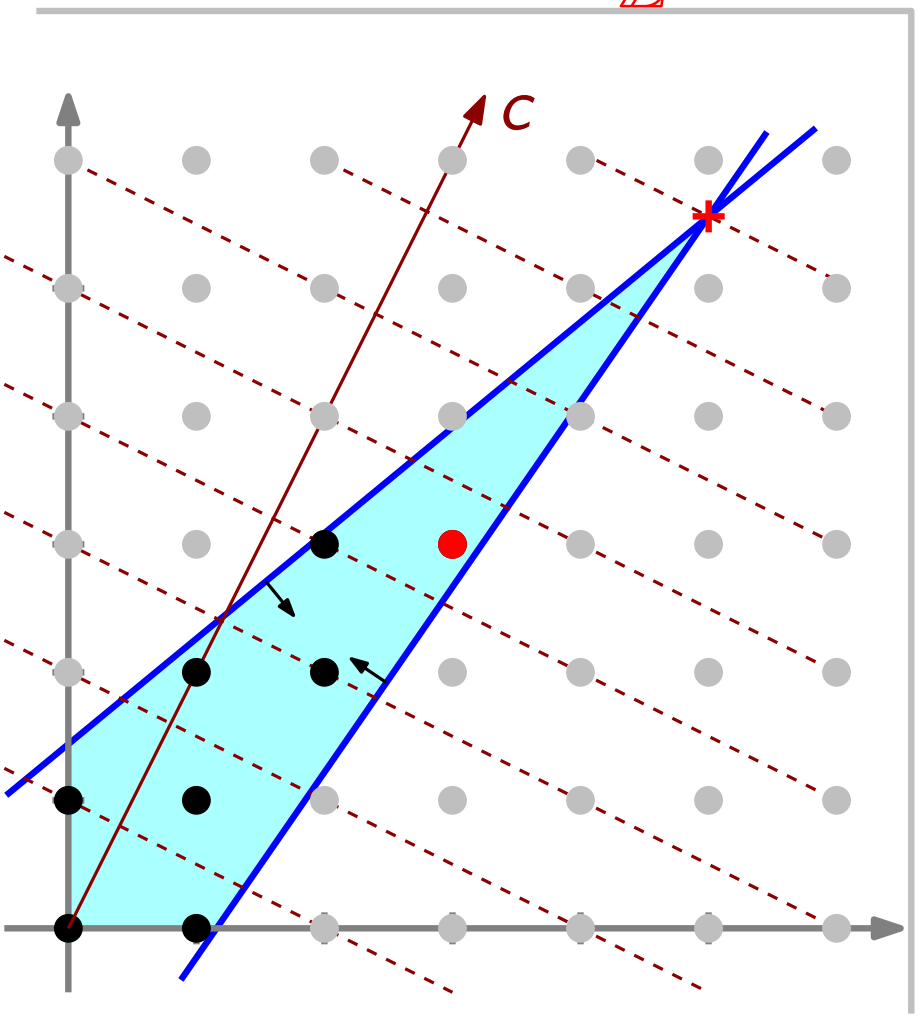
Beschränkungen:

für jede Kante $uv \in E$ fordern wir $x_u + x_v \geq 1$.

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. Geg. Graph $G = (V, E)$

Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

Modell. (0-1-ILP)

Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel: minimiere $\sum_{v \in V} x_v$

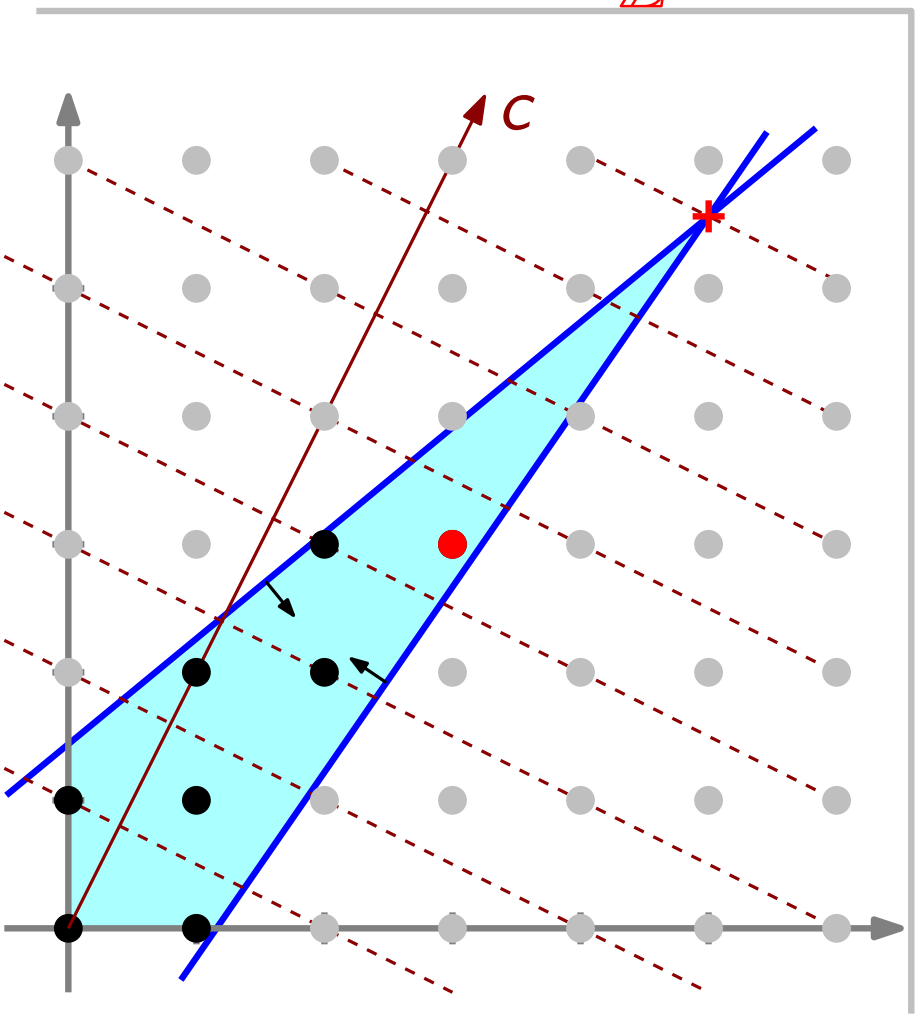
Beschränkungen:

für jede Kante $uv \in E$ fordern wir $x_u + x_v \geq 1$.

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. (NP-schwer)

Geg. Graph $G = (V, E)$

Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

Modell. (0-1-ILP)

Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel: minimiere $\sum_{v \in V} x_v$

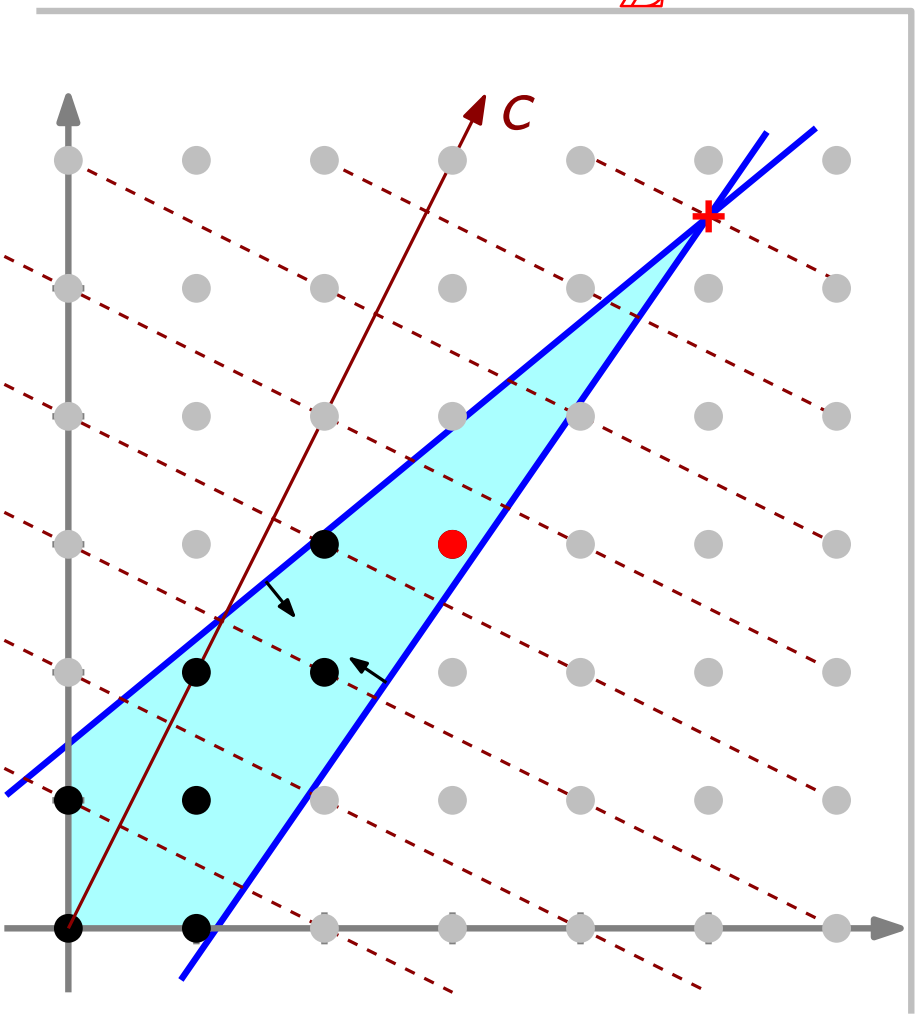
Beschränkungen:

für jede Kante $uv \in E$ fordern wir $x_u + x_v \geq 1$.

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, oder $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem. (NP-schwer)

- Geg. Graph $G = (V, E)$
- Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.
- Ziel: $|V'|$ minimal!

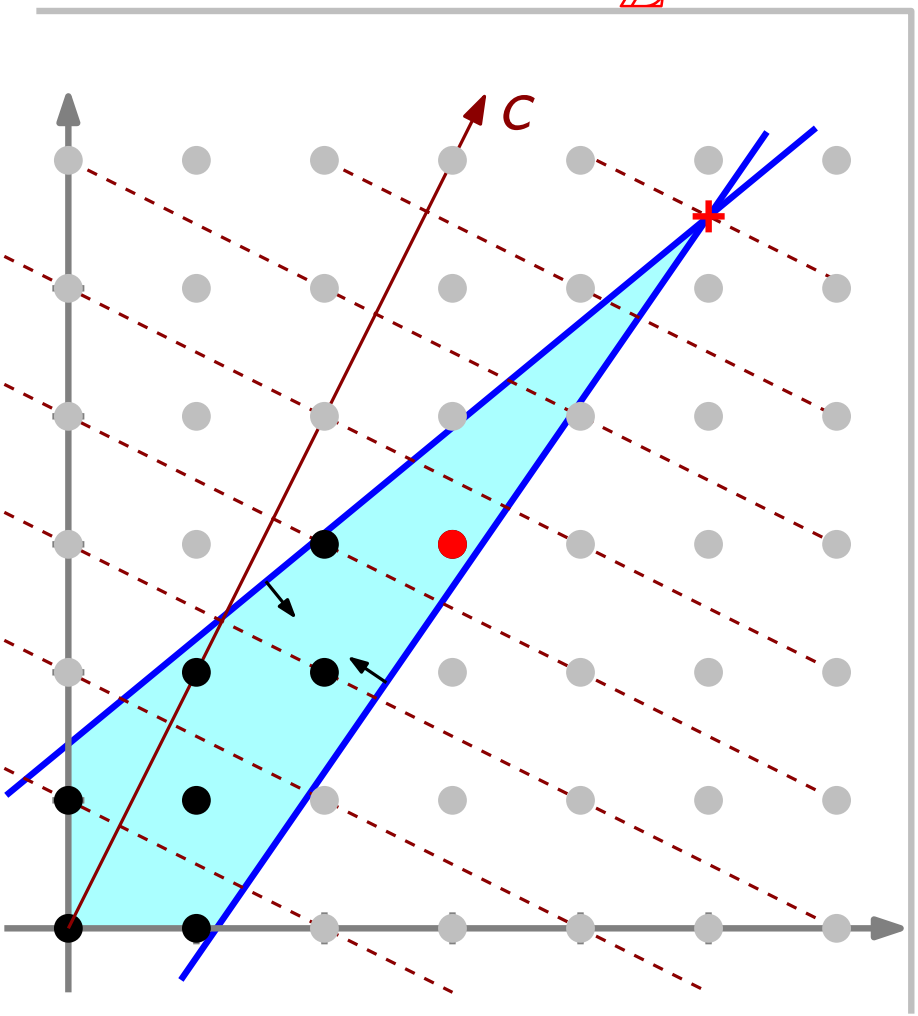
Modell. (0-1-ILP ⇓)

- Für $v \in V$ sei $x_v \in \{0,1\}$.
- Ziel: minimiere $\sum_{v \in V} x_v$
- Beschränkungen:
  - für jede Kante $uv \in E$ fordern wir $x_u + x_v \geq 1$.

Ganzzahlige lineare Programmierung (ILP)

Gegeben: $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$

Gesucht: $x^\star \in \mathbb{R}^n$ (häufig $\{0,1\}^n$, $\mathbb{Z}^n$) mit $x^\star = \arg\max\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0\}$ (oder $\arg\min$, $\geq$, $x \in \mathbb{Z}^n$)



Problem.

NP-schwer

Geg. Graph $G = (V, E)$

Ges. Knotenüberdeckung, d.h. $V' \subseteq V$, so dass jede Kante mind. einen Endpunkt in V' hat.

Ziel: $|V'|$ minimal!

Modell.

0-1-ILP

$\Downarrow$

NP-schwer

Für $v \in V$ sei $x_v \in \{0,1\}$.

Ziel: minimiere $\sum_{v \in V} x_v$

Beschränkungen:

für jede Kante $uv \in E$ fordern wir $x_u + x_v \geq 1$.

Bsp. II: ILP-Formulierung für MaxClique

Geg. ungerichteter, ungewichteter Graph $G = (V, E)$
Ges. Clique in G,

Bsp. II: ILP-Formulierung für MaxClique

Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G,

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

Bsp. II: ILP-Formulierung für MaxClique

Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\qquad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

Bsp. II: ILP-Formulierung für MaxClique

Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\qquad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Bsp. II: ILP-Formulierung für MaxClique

Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\qquad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!

Bsp. II: ILP-Formulierung für MaxClique

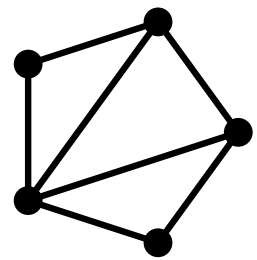
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \left(V', \{u'v' \in E : u' \in V', v' \in V'\}\right)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Bsp. II: ILP-Formulierung für MaxClique

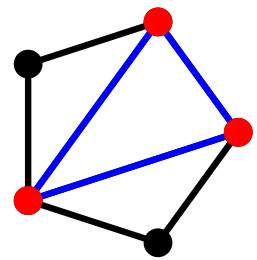
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\qquad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Bsp. II: ILP-Formulierung für MaxClique

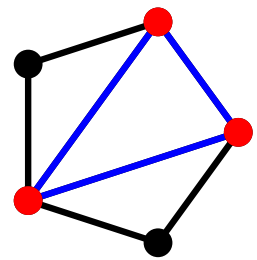
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$.

Bsp. II: ILP-Formulierung für MaxClique

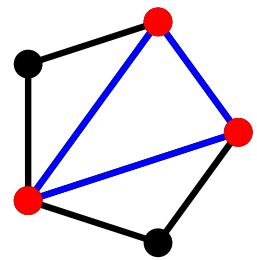
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\qquad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Bsp. II: ILP-Formulierung für MaxClique

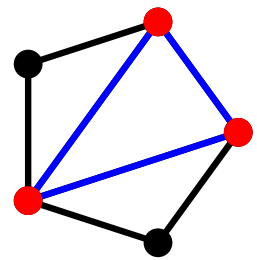
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Bsp. II: ILP-Formulierung für MaxClique

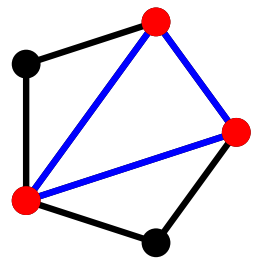
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Maximiere $\sum_{v \in V} x_v$

Bsp. II: ILP-Formulierung für MaxClique

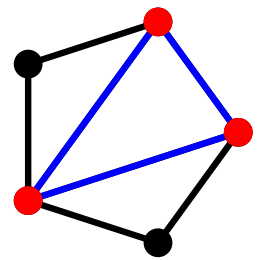
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Maximiere $\sum_{v \in V} x_v$

unter den Nebenbedingungen:

Bsp. II: ILP-Formulierung für MaxClique

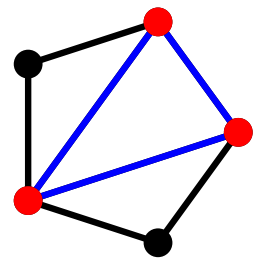
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Maximiere $\sum_{v \in V} x_v$

unter den Nebenbedingungen:

für jede Nicht-Kante $\{u, v\} \in \binom{V}{2} \setminus E$:

Bsp. II: ILP-Formulierung für MaxClique

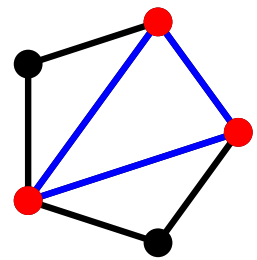
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Maximiere $\sum_{v \in V} x_v$

unter den Nebenbedingungen:

für jede Nicht-Kante $\{u, v\} \in \binom{V}{2} \setminus E$: $\quad x_u + x_v \leq 1.$

Bsp. II: ILP-Formulierung für MaxClique

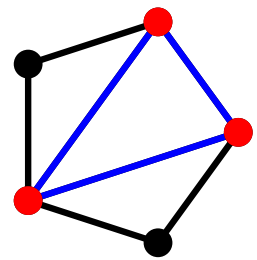
Geg. ungerichteter, ungewichteter Graph $G = (V, E)$

Ges. Clique in G, $\quad G[V'] := \big(V', \{u'v' \in E : u' \in V', v' \in V'\}\big)$

d.h. $V' \subseteq V$, sodass der von V' induzierte Graph $G[V']$ vollständig

m.a.W. $V' \subseteq V$, so dass für alle $\{u', v'\} \in \binom{V'}{2}$ gilt $u'v' \in E$.

Ziel: $|V'|$ maximal!



Variable:

Für $v \in V$ sei $x_v \in \{0, 1\}$. (Damit codieren wir die Menge V'!)

Zielfunktion:

Maximiere $\sum_{v \in V} x_v$

unter den Nebenbedingungen:

für jede Nicht-Kante $\{u, v\} \in \binom{V}{2} \setminus E$: $\quad x_u + x_v \leq 1.$

D.h. wenn u und v *nicht* benachbart sind, darf höchstens einer in die Clique.

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion:

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen:

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

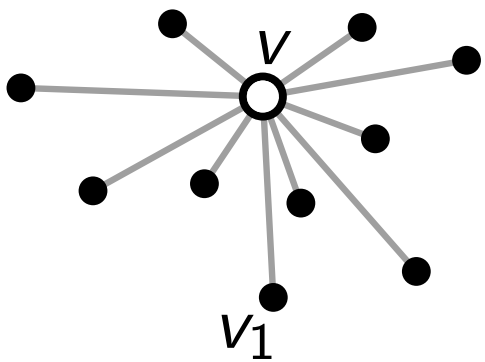
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$\sum_{u \in \mathrm{Adj}[v]} x_{uv} =$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

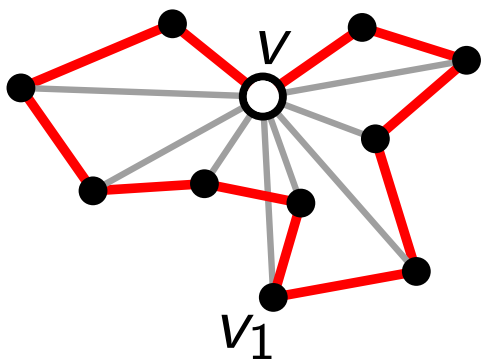
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} =$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

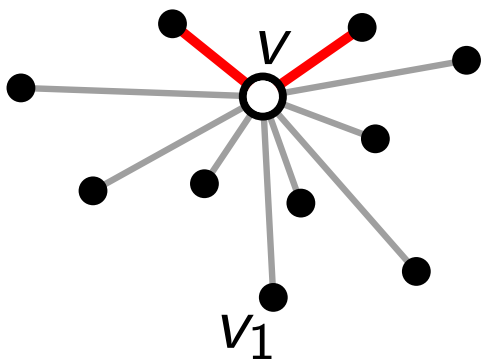
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} =$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

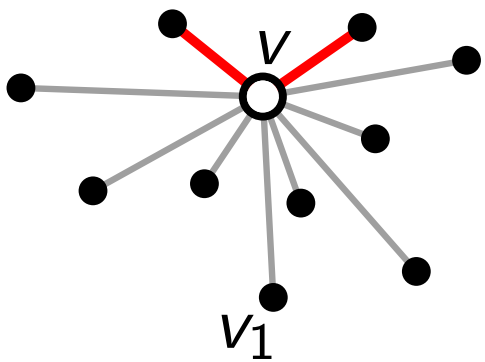
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

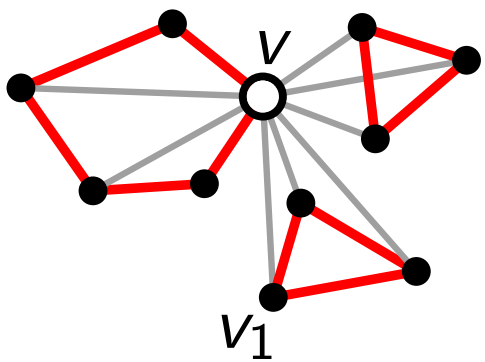
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

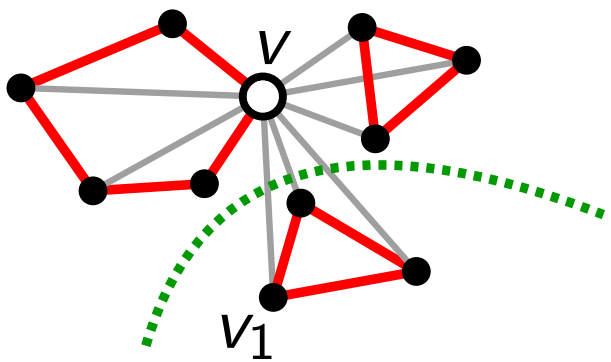
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

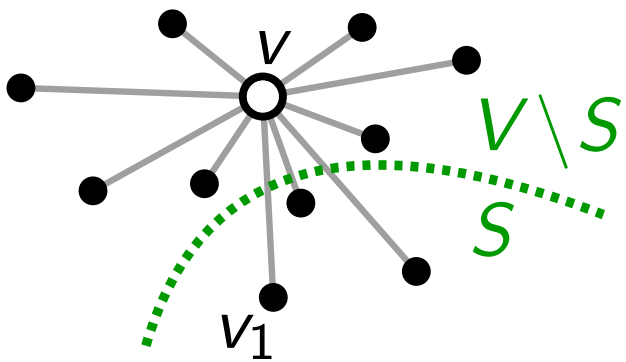
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

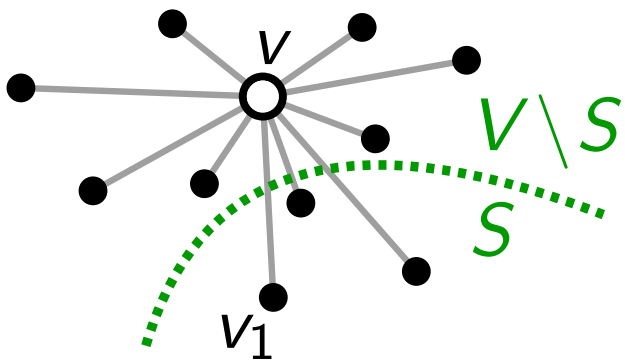
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

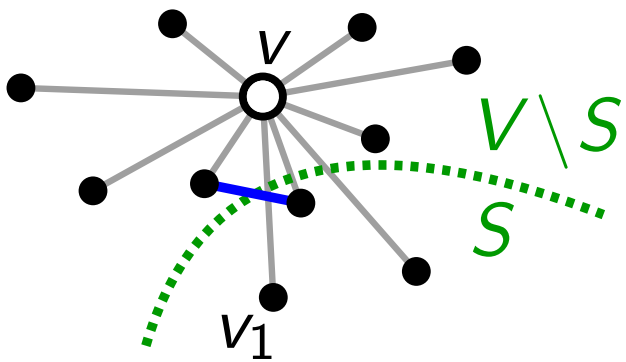
Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

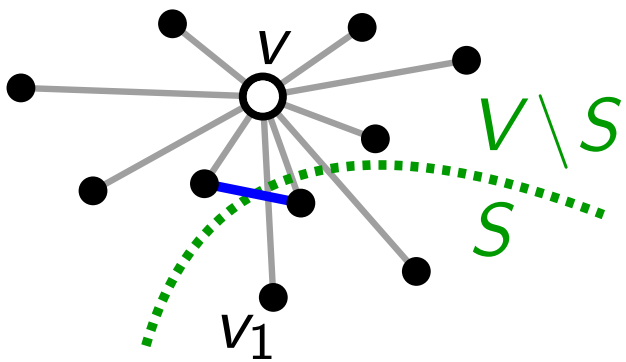
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 1$$

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

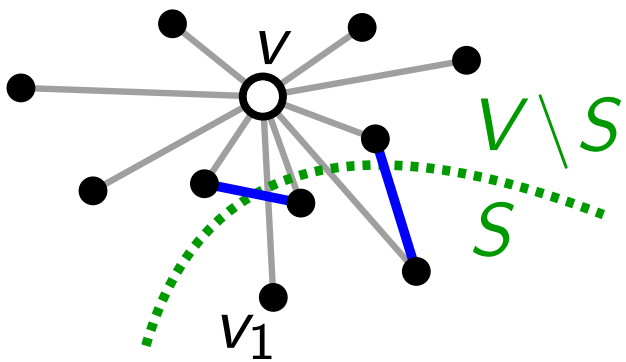
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 1$$

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

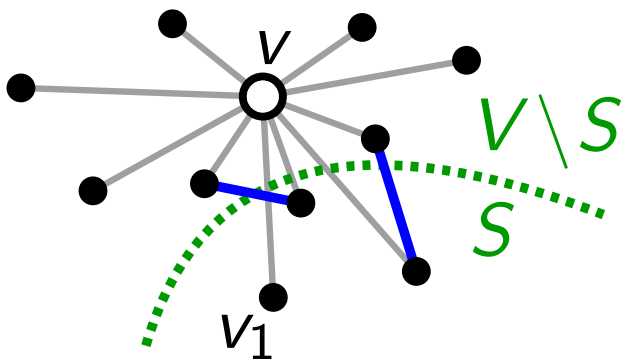
Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$

Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

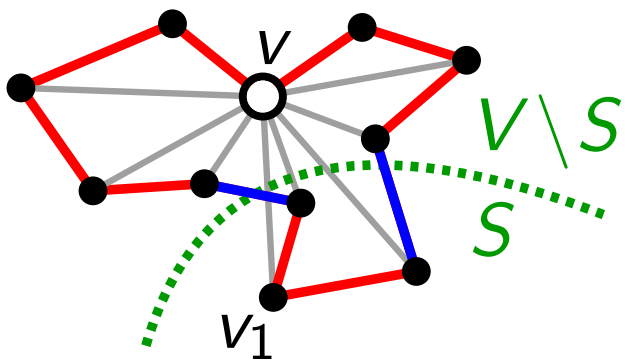
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Problem: Wie viele solcher Schnitte gibt's?

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

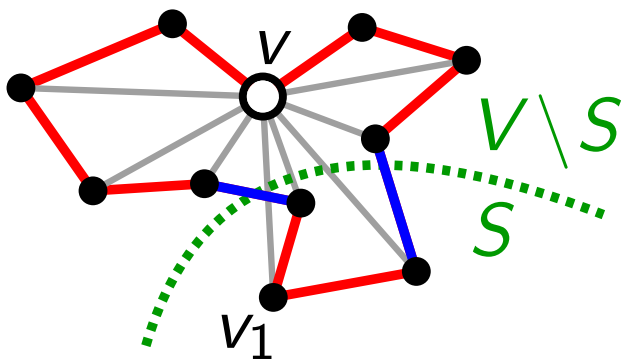
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Gegeben: vollständiger Graph $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ und Kantenkosten $c\colon E \to \mathbb{R}_{\geq 0}$

Gesucht: Hamiltonkreis K in G mit minimalen Kosten $c(K)$.
d.h. Permutation σ von $\langle 1, \ldots, n \rangle$, die $c(v_{\sigma(n)}, v_{\sigma(1)}) + \sum_{i=1}^{n-1} c(v_{\sigma(i)}, v_{\sigma(i+1)})$ minimiert.

Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

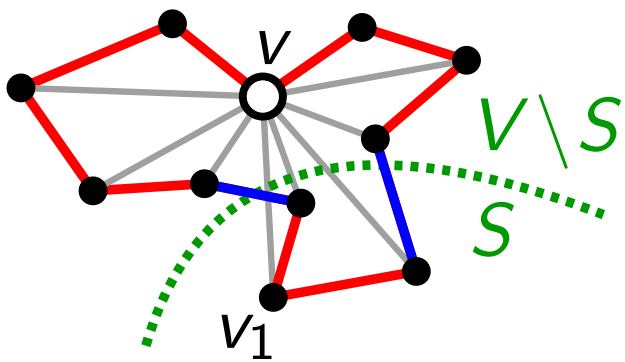
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Lösung:

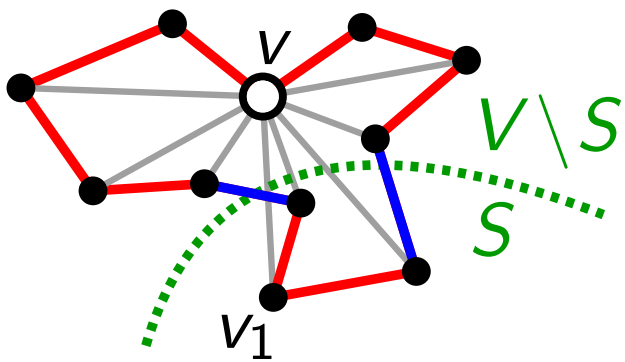
Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$

Bsp. III: ILP-Formulierung für TSP

Lösung: • Stelle Formulierung nur mit Knotenbedingungen auf.

Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

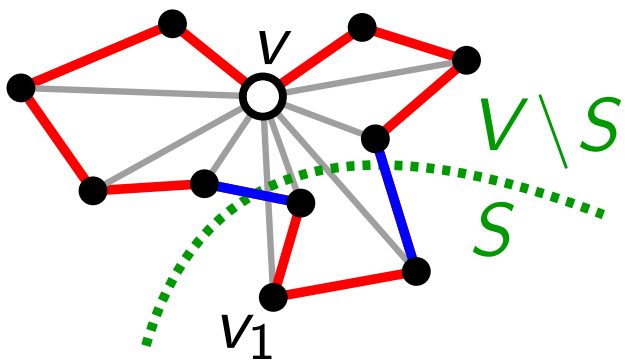
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Lösung:
- Stelle Formulierung nur mit Knotenbedingungen auf.
- Löse Formulierung F mit ILP-Solver (Cplex o.ä.)

Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

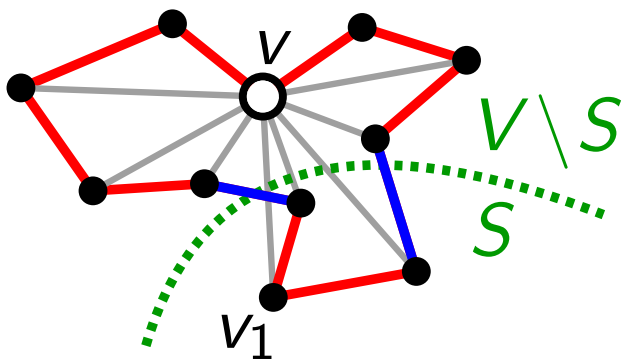
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Lösung:
- Stelle Formulierung nur mit Knotenbedingungen auf.
- Löse Formulierung F mit ILP-Solver (Cplex o.ä.)
- Solange Lösung aus mehreren Kreisen besteht:

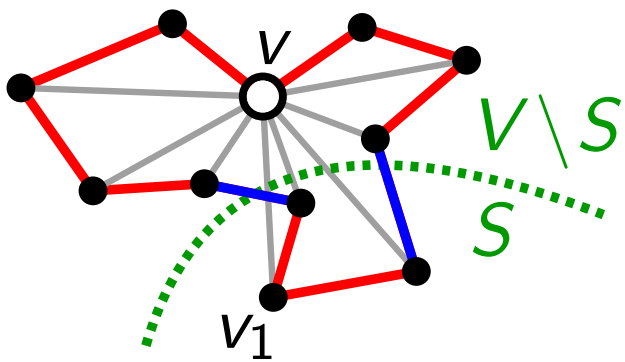
Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$

Bsp. III: ILP-Formulierung für TSP

Lösung:
- Stelle Formulierung nur mit Knotenbedingungen auf.
- Löse Formulierung F mit ILP-Solver (Cplex o.ä.)
- Solange Lösung aus mehreren Kreisen besteht:
  – Für jeden Kreis K

Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

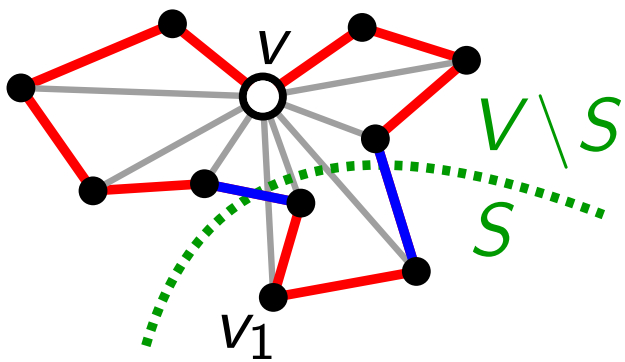
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Lösung:
- Stelle Formulierung nur mit Knotenbedingungen auf.
- Löse Formulierung F mit ILP-Solver (Cplex o.ä.)
- Solange Lösung aus mehreren Kreisen besteht:
  - Für jeden Kreis K
    füge die Schnittbed. für $(K, V \setminus K)$ zu F hinzu.

Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

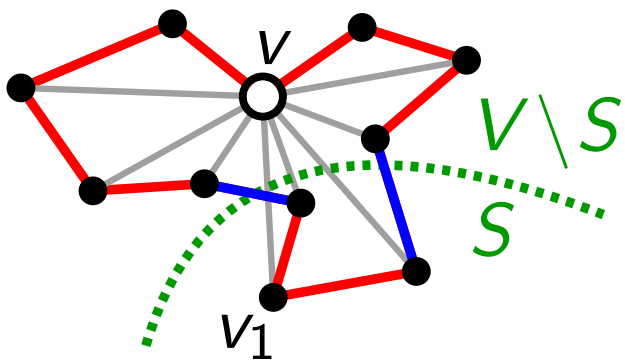
Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$

für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$



Bsp. III: ILP-Formulierung für TSP

Lösung:
- Stelle Formulierung nur mit Knotenbedingungen auf.
- Löse Formulierung F mit ILP-Solver (Cplex o.ä.)
- Solange Lösung aus mehreren Kreisen besteht:
  - Für jeden Kreis K
    füge die Schnittbed. für $(K, V \setminus K)$ zu F hinzu.
  - Fahre mit der Lösung des ILPs fort („Warmstart“).

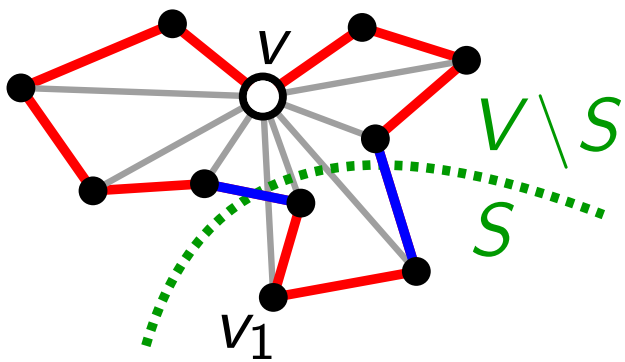
Problem: Es gibt $2^{n-1} - 1$ solcher Schnitte!

Variable: Für $uv \in E$ sei $x_{uv} \in \{0, 1\}$.

Zielfunktion: Minimiere $\sum_{uv \in E} c(u, v) \cdot x_{uv}$

Nebenbedingungen: für jeden Knoten $v \in V$:

$$\sum_{u \in \mathrm{Adj}[v]} x_{uv} = 2$$



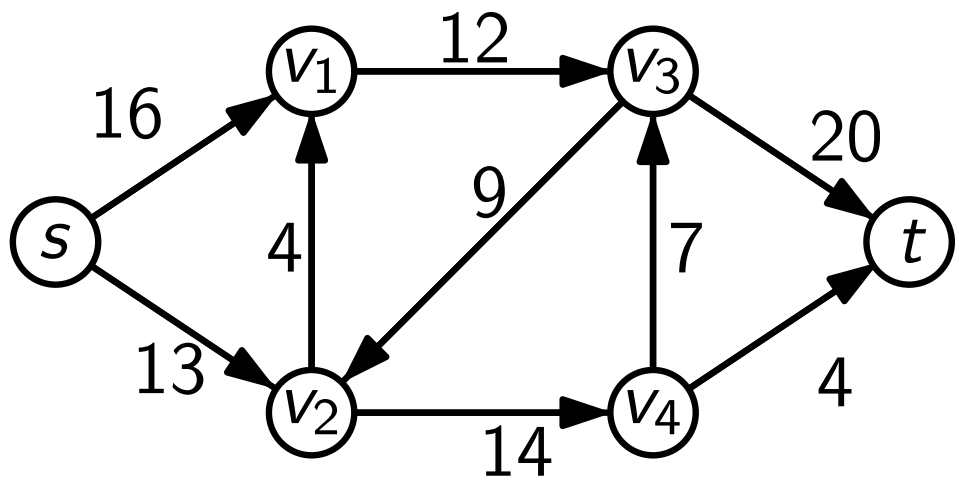
für jeden Schnitt $(S, V \setminus S)$ mit $\begin{cases} v_1 \in S \\ S \neq V \end{cases}$

$$\sum_{u \in S, v \in V \setminus S} x_{uv} \geq 2$$

□

Bsp. IV: Ein Transportproblem

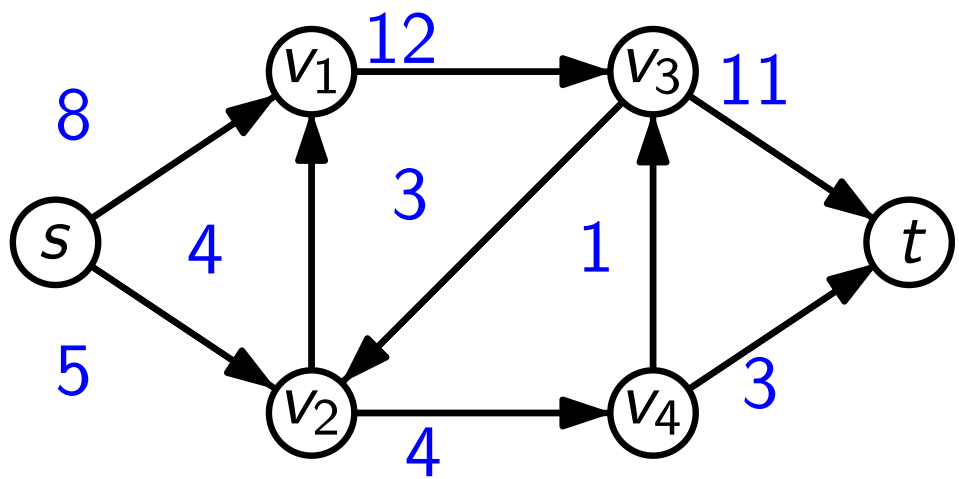
Ihre Firma möchte eine möglichst große Menge einer Ware von ihrem Lager s zu einem Kunden am Zielort t transportieren. Aufgrund bestehender Transport-Kapazitäten kann nur eine begrenzte Menge der Ware pro Transportabschnitt (Kante) transportiert werden. Welche Menge pro Tag können Sie zum Großkunden senden?



Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

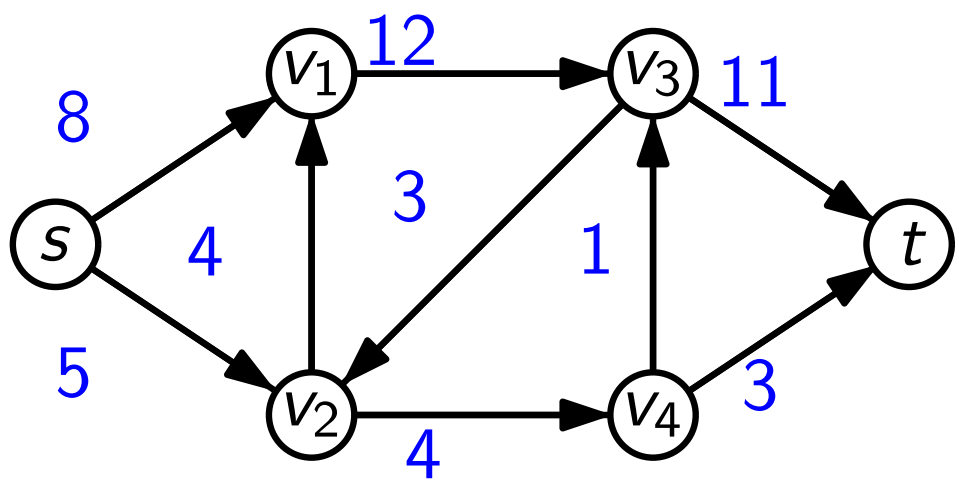
$$\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv) \;-\; \sum_{w \in \mathrm{Adj}[v]} f(vw) \;=\; 0.$$



Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

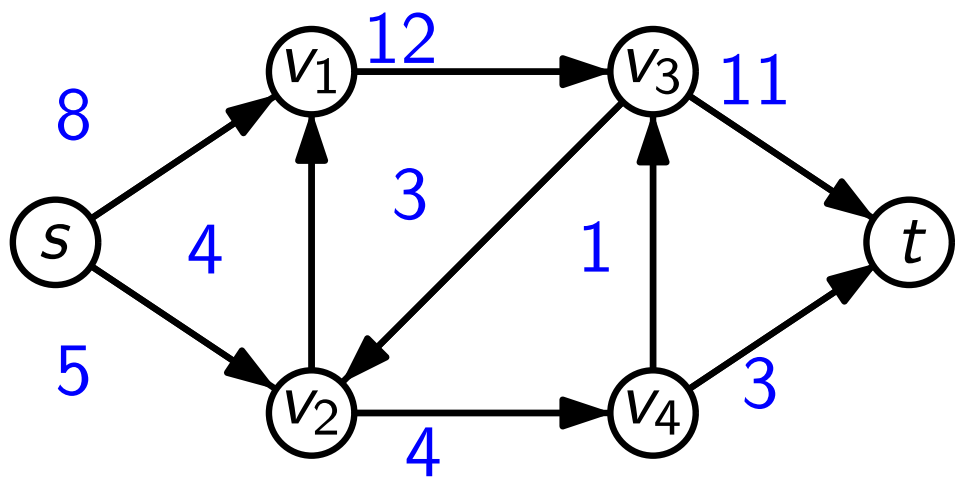
$$\underbrace{\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)} = 0.$$



Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

$$\underbrace{\underbrace{\sum_{\{u \in V\,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)}}_{\mathrm{Nettozufluss}_f(v)} = 0.$$

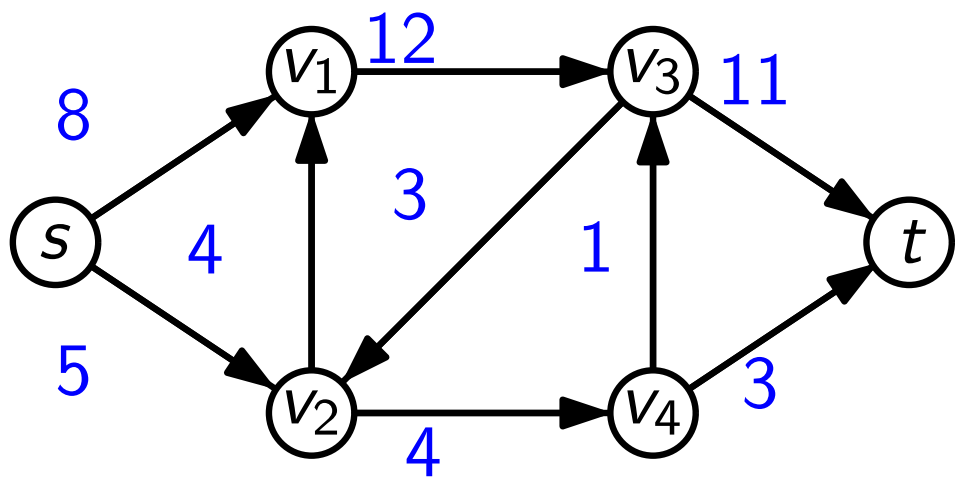


Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

Fluss-erhaltung:

$$\underbrace{\underbrace{\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)}}_{\mathrm{Nettozufluss}_f(v)} = 0.$$

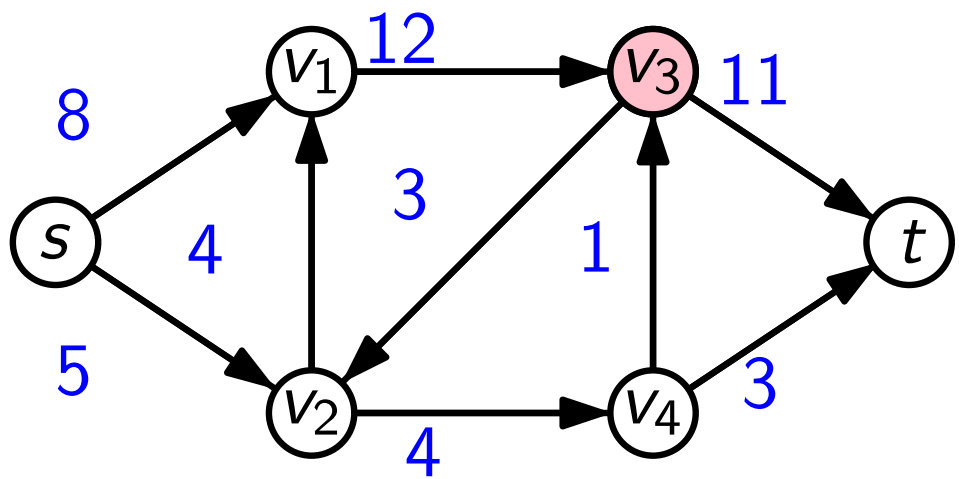


Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

Fluss-erhaltung:

$$\underbrace{\underbrace{\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)}}_{\mathrm{Nettozufluss}_f(v)} = 0.$$

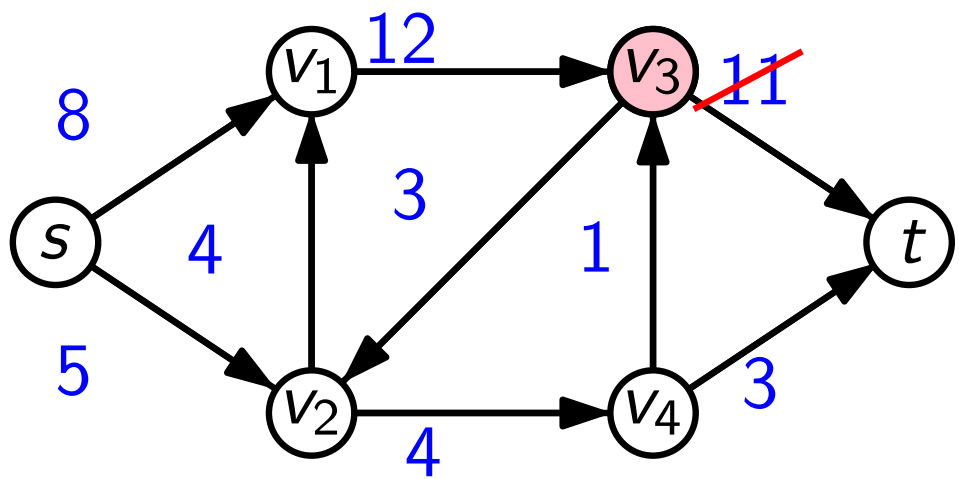


Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

Fluss-erhaltung:

$$\underbrace{\underbrace{\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)}}_{\mathrm{Nettozufluss}_f(v)} = 0.$$

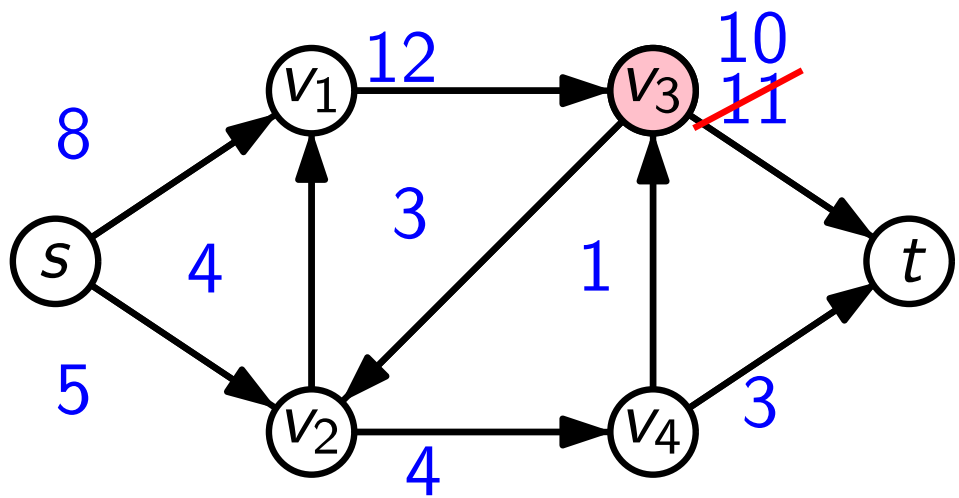


Modellierung durch Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Eine Funktion $f : E \to \mathbb{R}_{\geq 0}$ heißt *s-t-Fluss* („*Fluss*“),
wenn für jeden Knoten $v \in V \setminus \{s, t\}$ gilt

Fluss-erhaltung:

$$\underbrace{\underbrace{\sum_{\{u \in V \,:\, v \in \mathrm{Adj}[u]\}} f(uv)}_{\mathrm{Zufluss}_f(v)} - \underbrace{\sum_{w \in \mathrm{Adj}[v]} f(vw)}_{\mathrm{Abfluss}_f(v)}}_{\mathrm{Nettozufluss}_f(v)} = 0.$$

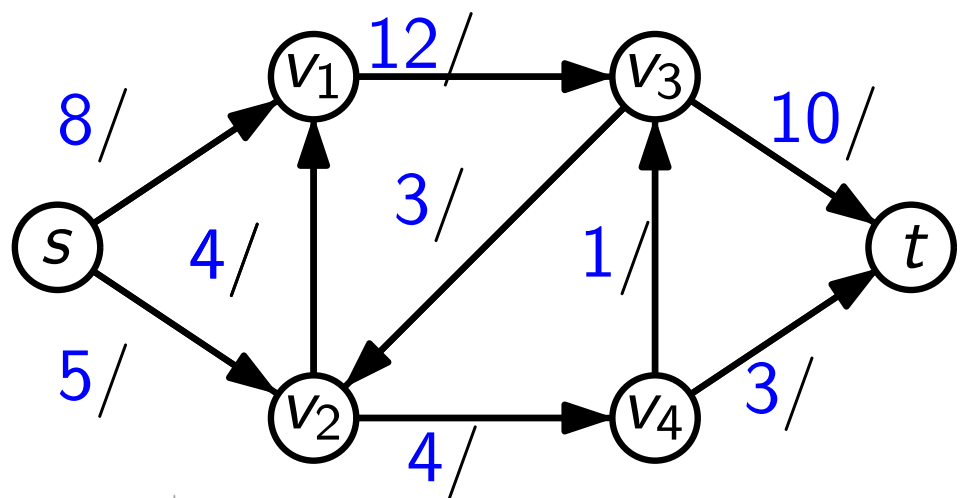


Zulässige und maximale Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Seien durch $c\colon E \to \mathbb{R}_{>0}$ Kantenkapazitäten* gegeben.
Ein Fluss f ist *zulässig*, wenn für jede Kante $e \in E$ gilt

$$0 \leq f(e) \leq c(e).$$

f/c



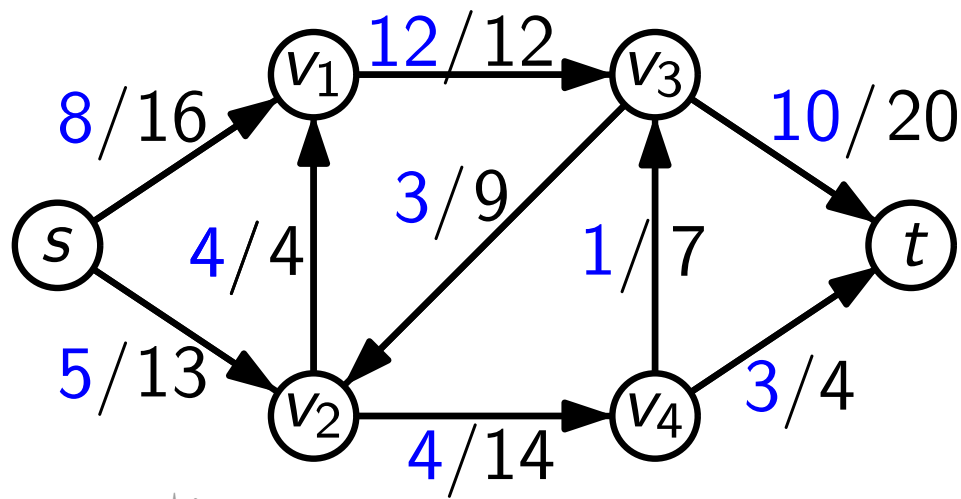
*) Diese Funktion hat nichts mit dem Zielfunktionsvektor bei den LPs zu tun!

Zulässige und maximale Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Seien durch $c\colon E \to \mathbb{R}_{>0}$ Kantenkapazitäten* gegeben.
Ein Fluss f ist *zulässig*, wenn für jede Kante $e \in E$ gilt

$$0 \leq f(e) \leq c(e).$$

f/c



*) Diese Funktion hat nichts mit dem Zielfunktionsvektor bei den LPs zu tun!

Zulässige und maximale Flüsse

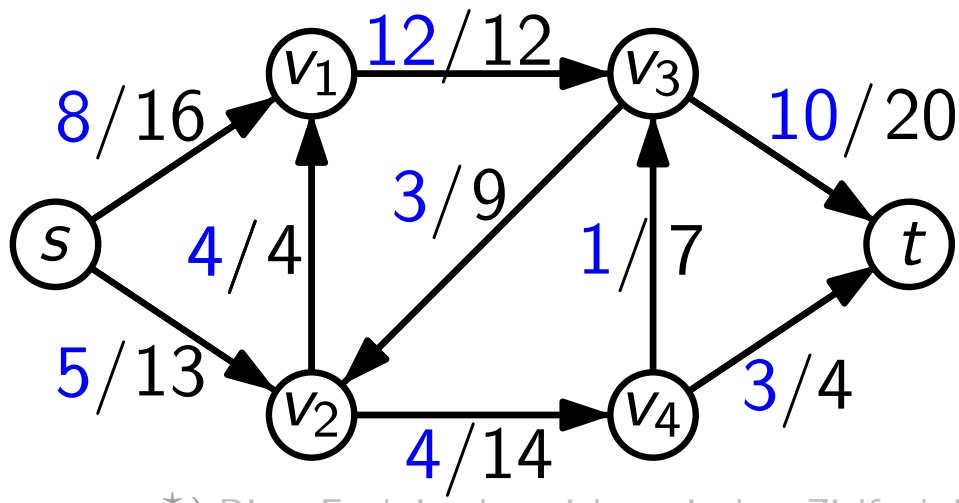
Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Seien durch $c\colon E \to \mathbb{R}_{>0}$ Kantenkapazitäten* gegeben.
Ein Fluss f ist *zulässig*, wenn für jede Kante $e \in E$ gilt

$$0 \leq f(e) \leq c(e).$$

Der *Wert* $|f|$ eines zulässigen Flusses f ist

$$\text{Nettozufluss}_f(t).$$

f/c



*) Diese Funktion hat nichts mit dem Zielfunktionsvektor bei den LPs zu tun!

Zulässige und maximale Flüsse

Def. Sei $G = (V, E)$ ein gerichteter Graph mit $s, t \in V$.
Seien durch $c\colon E \to \mathbb{R}_{>0}$ Kantenkapazitäten* gegeben.
Ein Fluss f ist *zulässig*, wenn für jede Kante $e \in E$ gilt

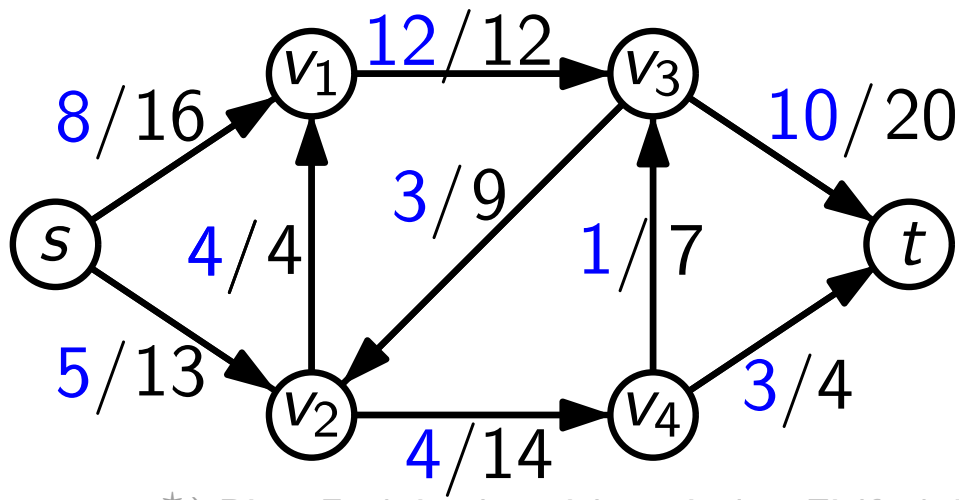
$$0 \leq f(e) \leq c(e).$$

Der *Wert* $|f|$ eines zulässigen Flusses f ist

$$\text{Nettozufluss}_f(t).$$

Ein zulässiger Fluss f ist *maximal*,
wenn für jeden zulässigen Fluss f' gilt $|f'| \leq |f|$.

f/c



*) Diese Funktion hat nichts mit dem Zielfunktionsvektor bei den LPs zu tun!

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c \colon E \to \mathbb{R}_{>0}$.

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c \colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert.

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c \colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c \colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält

– zulässig ist

– maximal ist

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\colon\, v \in \text{Adj}[u]\}} f(uv) \;-\; \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist

– maximal ist

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\colon\, v \in \text{Adj}[u]\}} f(uv) \;-\; \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

$$0 \leq f(e) \leq c(e),$$

– maximal ist

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\colon\, v \in \text{Adj}[u]\}} f(uv) \;-\; \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

$$0 \leq f(e) \leq c(e),$$

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$$|f| = \text{Nettozufluss}_f(t) \text{ maximiert.}$$

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) \;-\; \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

$$0 \leq f(e) \leq c(e),$$

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$$|f| = \text{Nettozufluss}_f(t) \text{ maximiert.}$$

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

Variable

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) - \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

$$0 \leq f(e) \leq c(e),$$

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$$|f| = \text{Nettozufluss}_f(t) \text{ maximiert.}$$

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

Variable

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) - \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

Konstante

$$0 \leq f(e) \leq c(e),$$

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$$|f| = \text{Nettozufluss}_f(t) \text{ maximiert.}$$

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

Variable

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) - \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

Konstante

$$0 \leq f(e) \leq c(e),$$

$|V| - 2 + |E|$ *lineare Beschränkungen!*

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$|f| = \text{Nettozufluss}_f(t)$ maximiert.

Aufgabe

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) - \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

Variable

– zulässig ist, d.h. für jede Kante e garantiert:

$$0 \leq f(e) \leq c(e),$$

Konstante

$|V| - 2 + |E|$ *lineare Beschränkungen!*

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$|f| = \text{Nettozufluss}_f(t)$ maximiert.

lineare Zielfunktion!

Aufgabe

YES, it's an LP!

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $s, t \in V$ und Kantenkapazitäten $c\colon E \to \mathbb{R}_{>0}$.

Geben Sie eine Methode an, die einen maximalen s-t-Fluss f konstruiert, also eine Funktion $f : E \to \mathbb{R}_{\geq 0}$, die

– den Fluss erhält, d.h. für jeden Knoten $v \notin \{s, t\}$ sicherstellt:

Variable

$$\text{Nettozufluss}_f(v) = \sum_{\{u \in V\,:\, v \in \text{Adj}[u]\}} f(uv) - \sum_{w \in \text{Adj}[v]} f(vw) = 0,$$

– zulässig ist, d.h. für jede Kante e garantiert:

Konstante

$$0 \leq f(e) \leq c(e),$$

$|V| - 2 + |E|$ *lineare Beschränkungen!*

– maximal ist, d.h. unter allen zulässigen s-t-Flüssen

$|f| = \text{Nettozufluss}_f(t)$ maximiert.

lineare Zielfunktion!

Flussalgorithmen

Kann man maximale Flüsse (= Spezialfall eines LPs) auch mit maßgeschneiderten kombinatorischen Algorithmen berechnen?

Flussalgorithmen

Kann man maximale Flüsse (= Spezialfall eines LPs) auch mit maßgeschneiderten kombinatorischen Algorithmen berechnen?

Hoffnung: Das könnte schneller gehen –
und strukturelle Einsichten liefern.

Flussalgorithmen

Kann man maximale Flüsse (= Spezialfall eines LPs) auch mit maßgeschneiderten kombinatorischen Algorithmen berechnen?

Hoffnung: Das könnte schneller gehen –
und strukturelle Einsichten liefern.

Fortsetzung folgt!