

9 Ganzzahlige Lineare Optimierung

Branch&Bound:

Bsp.:

A	b
---	---

$$\begin{pmatrix} -5 & -2 & -1 & -2 \\ -1 & 1 & 1 & -2 \\ -3 & -1 & -1 & -3 \end{pmatrix} \quad \begin{pmatrix} -8 \\ -2 \\ -5 \end{pmatrix}$$

$$c = (-2, -1, 1, 2)$$

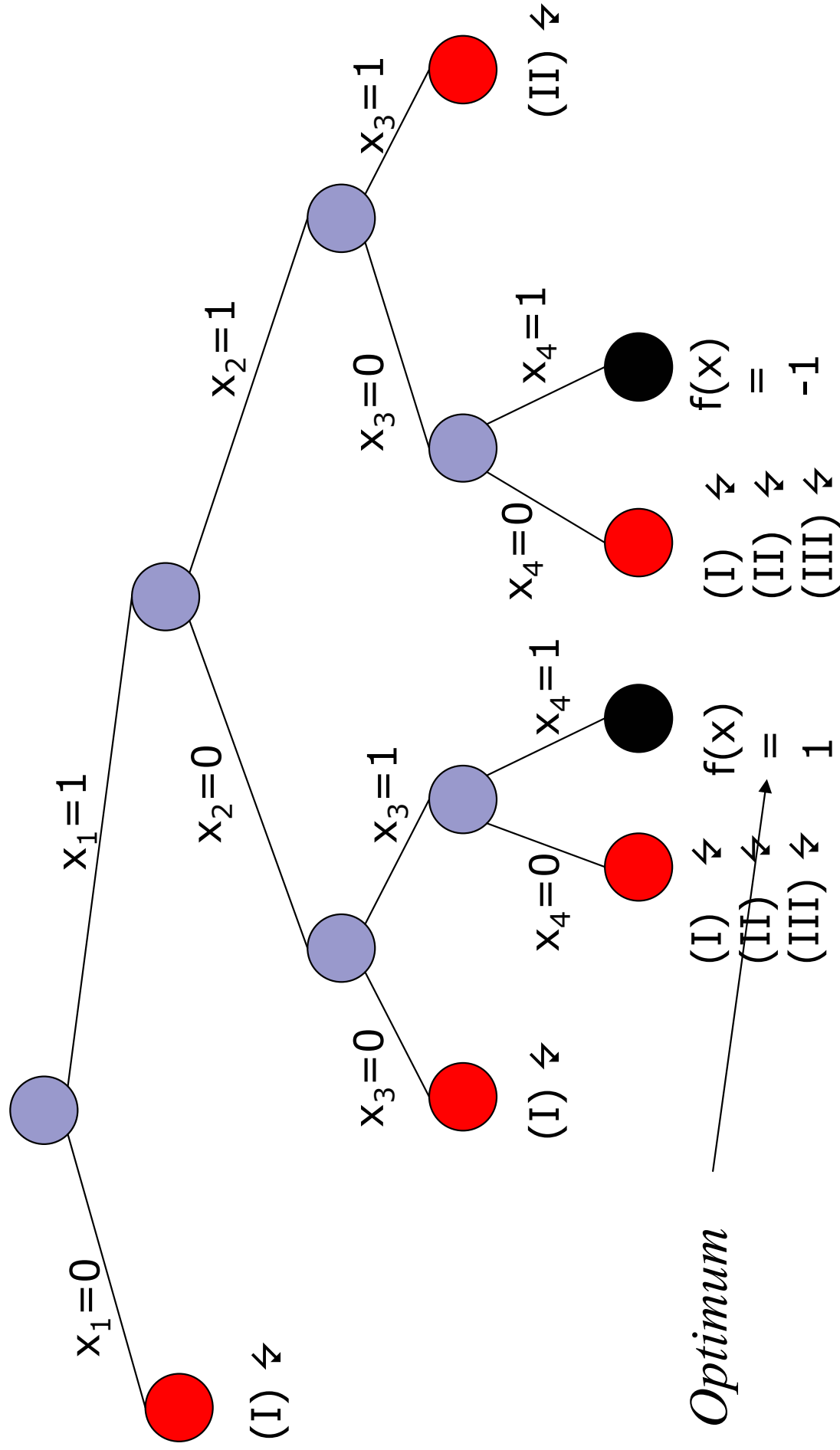
ges.: $x \in \{0, 1\}^4$

$$\begin{aligned} \text{mit} \quad & -5x_1 - 2x_2 - x_3 - 2x_4 \leq -8 \quad (\text{I}) \\ & -x_1 + x_2 + x_3 - 2x_4 \leq -2 \quad (\text{II}) \\ & -3x_1 - x_2 - x_3 - 3x_4 \leq -5 \quad (\text{III}) \end{aligned}$$

$$f(x) = -2x_1 - x_2 + x_3 + 2x_4 \rightarrow \max$$

9 Ganzzahlige Lineare Optimierung

Branch&Bound:



9 Ganzzahlige Lineare Optimierung

Branch&Bound:

```
function BranchAndBound(Zustandsbaum G = (V,E), s ∈ V) {  
  
    opt = Approximation(s);  
    best_sol sei Lsg. mit c(best_sol)=opt;           // obere Schranke  
    H = {(s, Relaxation(s))};                     // untere Schranke  
  
    while (H ≠ ∅) {  
        wähle und lösche nächstes Element (w,uw) aus H; // Select  
        generiere alle Nachfolger w1,...,wb von w in G; // Branch  
  
        for (i = 1 to b do) {  
            u = Relaxation(wi); l = Approximation(wi); // Schranken berechnen  
            if (l > opt) {                               // neue beste Lsg  
                opt = l; best_sol sei Lsg. mit c(best_sol) = l;  
                H = H \ {(x,ux) | ux ≤ opt};          // Cutoffs: Bound  
            }  
            if (u < opt) H = H ∪ {(wi,u)};  
        } // for  
    } // while  
    return best;  
}
```

9 Ganzzahlige Lineare Optimierung

Branch&Bound:

Die Knoten beschreiben Zustände, in denen einige Variablen bereits gesetzt sind (**feste** Variablen), andere noch nicht (**freie** Variablen).

Heuristiken leiten Suche in vielversprechende Teile des Zustandsgraphen. Sind Ergebnisse von Heuristiken Schranken, so heißen sie Relaxationen (oder zulässige Heuristiken, admissible, ...)

Bsp.:

- Suche kann abgebrochen werden, wenn für ein $i \in \{1, \dots, m\}$
$$\sum_{x_j \text{ frei}} \min\{A_{ij}, 0\} + \sum_{x_j \text{ fest}} A_{ij} \bullet x_j > b_i$$
- Def.: Es sei $P=(A,b,c)$ ein IP. Das LP $L=(A,b,c)$ heißt LP-Relaxation von P .
 $L \in \mathcal{P}$, d.h., es ex. ein polynomial zeitbeschränkter (d.h. oft: einen schnellen) Algorithmus für L .

9 Ganzzahlige Lineare Optimierung

Branch&Bound:

Beobachtung: Es sei $P = (A,b,c)$ ein IP, x optimale Lsg. für P .
Es sei $R = (A,b,c)$ LP-Relaxation und y optimal für R .
Dann gilt: $c^T x \leq c^T y$.

Bew: x optimal für $P \Rightarrow x$ zulässig für $P \Rightarrow x$ zulässig für $R \Rightarrow$
 $c^T x \leq c^T y$, da y optimal für R .

Lemma: Es sei v Knoten im Suchgraph, $P(v)$ sein IP, $R(v)$ LP-Relaxation von $P(v)$. Es sei x' optimale Lsg von $R(v)$.
Dann gilt: Jede zulässige Lsg x für $P(v)$ hat Kosten
 $c^T x \leq c^T x'$

Beweis: x zulässig für $P(v) \Rightarrow x$ zulässig für $R(v)$
 $\Rightarrow c^T x \leq c^T x'$ (x' optimal)

Damit: Es sei x^{\sim} bisher beste bekannte Lsg von P (Wurzel)

Die Suche unter einem Knoten v kann abgebrochen werden,
wenn $c^T x' \leq c^T x^{\sim}$

9 Ganzzahlige Lineare Optimierung

Branch&Bound:

- Verschiedene Implementierungen der Menge H führen zu verschiedenen Selects und damit zu verschiedenem Verhalten:
 - H Stack → LIFO-Select → Tiefensuche
 - H Queue → FIFO-Select → Breitensuche
 - H Priority Queue bzgl. L_w → Best-First-Select → Bestensuche
- Bei Implementierung von H als Queue oder Priority-Queue ist der Speicheraufwand linear in der Größe des Zustandsgraphen.
- Die Qualität der Schranken bestimmt den Suchaufwand
- Die Größe einzelner Teilbäume ist nicht vorhersagbar

9 Ganzzahlige Lineare Optimierung

Anwendung von Branch&Bound bei allgemeinen MIPs:
geg.: ein gemischt ganzzahliges Lineares Programm.

1. Falls alle der ganzzahligen Variablen bei der optimalen Lösung der LP-Relaxation bereits ganzzahlige sind, haben wir eine Lösung für das MIP gefunden.
2. Sonst sind diejenigen Variablen, die als ganzzahlig deklariert wurden, aber in der LP-Relaxation nicht ganzzahlig gesetzt wurden, mögliche Branchingvariablen. Bsp.: x habe den Wert 4,356. Da x in der Lösung unsrerer MIPs ganzzahlig sein soll, kann man sagen $x \neq 4.356$. Man fordert $x \leq 4$ oder $x \geq 5$. Es ergeben sich 2 neue MIPs:

MIP1: $MIP \cup \{x \leq 4\}$ und MIP2: $MIP \cup \{x \geq 5\}$

