60016 OPERATIONS RESEARCH

Branch & Bound Algorithm

23 November 2020

Solving (M)ILPs so far

- LP Relaxation
- Cutting Plane Algorithms (Gomory Cut, Knapsack Cover Cut)

Black Panther

Solving MILPs now: branch and bound

- Divide & Conquer
- Branch & Bound

The Hulk

Mixed Integer Programming

Mixed Integer Linear Programming (MILP, MIP) Problem:

$$
\begin{aligned}
&\min z = c^{\mathrm{T}} x \\
&\text{s.t. } Ax = b \\
&\qquad x_j \geq 0 \qquad\qquad \text{for } j \in N = \{1, \ldots, n\} \\
&\qquad x_j \in \mathbb{Z} \qquad\qquad \text{for } j \in Z \subseteq N.
\end{aligned}
$$

Note: $x_j \in N \setminus Z$ are continuous, $x_j \in Z$ are integer.

Additional assumption this lecture: Finite bounds $\underline{x}_j$, $\overline{x}_j$ for $j \in Z$: $x_j \in \{\underline{x}_j, \underline{x}_j + 1, \ldots, \overline{x}_j\}$.

Complete Enumeration Algorithm

Idea: Without loss of generality, assume $Z = \{1, \ldots, k\}$. Loop through all values of the integer variables solving LPs in the continuous variables:

for $x_1 \in \{\underline{x}_1, \underline{x}_1 + 1, \ldots, \overline{x}_1\}$ do
  for $x_2 \in \{\underline{x}_2, \underline{x}_2 + 1, \ldots, \overline{x}_2\}$ do
    ...
      for $x_k \in \{\underline{x}_k, \underline{x}_k + 1, \ldots, \overline{x}_k\}$ do
        Solve LP in $x_{k+1}, \ldots, x_n$ with $x_1, \ldots, x_k$ fixed.
        Record LP solution found.
      end for
    ...
end for

Return best LP solution solution found or report infeasibility.

Sanity Check. What is the complexity of complete enumeration? Would it ever be reasonable to use complete enumeration?

Divide and Conquer Principle

Structure search to scan only through a subset of solutions.

General Principle: Divide and Conquer

- *Divide* a large problem into several smaller subproblems.
- *Conquer* solution by working on the smaller subproblems.

Notation:

- P_i: i-th subproblem
- $x^*(P_i)$: optimal solution to the i-th subproblem

Divide and Conquer Principle

Application to MILP:

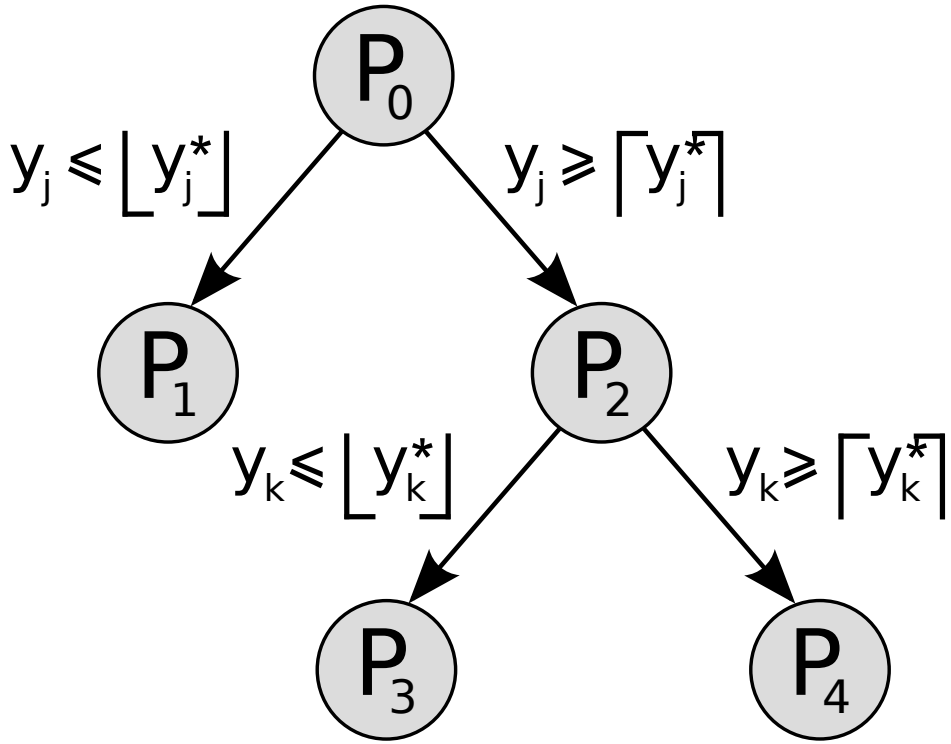
- Solve LP relaxation of problem $P_0 \Rightarrow x^*(P_0)$.
- Stop if solution satisfies integrality constraints.
- Otherwise choose non-integer $x_p^* \in x^*(P_0)$, $p \in Z$:
  - Divide (Branch): Create two subproblems, P_1 and P_2, adding to P_0 the constraints $x_p \leq \lfloor x_p^* \rfloor$ and $x_p \geq \lceil x_p^* \rceil$, respectively.
  - Conquer (Bound/Fathom): Recursively solve the new subproblems. If optimal solution of continuous relaxation of P_1 (P_2) is worse than any known feasible solution for P_0, disregard P_1 (P_2).

Any solution of P_0 satisfying integrality constraints is feasible in one of P_1 and P_2. Hence, we can solve P_0 by solving P_1 and P_2.

Sanity Check. Why do we divide each node into two subproblems? Why do we fathom nodes with continuous relaxation worse than any known feasible solution?

Divide and Conquer Principle

Recursive application of divide and conquer principle leads to binary tree:



Terminal nodes = problems that remain to be solved.

Branch & Bound for MILPs

Preliminaries:

- P_0 denotes original MILP problem:

$$\begin{aligned} \min\ & z = c^{\mathrm{T}} x \\ \text{s.t. } & Ax = b \\ & x_j \geq 0 && \text{for } j \in N = \{1, \ldots, n\} \\ & x_j \in \mathbb{N}_0 && \text{for } j \in Z \subseteq N. \end{aligned}$$

Sanity Check. We previously said that the methods today work for integer variables $\mathbb{Z}$. So why are we now specifying that the integer variables are in $\mathbb{N}_0$?

Branch & Bound for MILPs

Preliminaries:

- P_i: i-th subproblem
- $x^*(P_i)$: optimal solution for LP relaxation of P_i.
- $c^T x^*(P_i)$: optimal value for LP relaxation of P_i. Set $c^T x^*(P_i) = \infty$ if P_i is an infeasible LP.
- OPT denotes objective function value of best *feasible* solution (for P_0) found so far. At beginning, $OPT = \infty$.
- x_{OPT} is the solution producing the best objective OPT.
- A LP solution $x^*(P_i)$ is feasible for MILP P_0 if it satisfies all integrality constraints of P_0.

Branch & Bound Steps for MILPs

1. Initialization.
   - Initialize $OPT = \infty$.
   - Solve LP relaxation of $P_0 \Rightarrow x^*(P_0)$.
   - If $x^*(P_0)$ feasible for P_0, $OPT = c^T x^*(P_0)$, $x_{OPT} = x^*(P_0)$ and STOP.
   - Otherwise set list of problems to $\{P_0\}$.

2. Problem Selection. Extract from list a P_i having $c^T x^*(P_i) < OPT$. If no such P_i exists, STOP.

3. Variable Selection. Choose non-integer $x_p^* \in x_p^*(P_i)$ that must be integer in P_0 (i.e., $p \in Z$).

Sanity Check. How to chose the branching variable?

Branch & Bound for MILPs

4. Branch.
   - Create subproblems P' and P'' with $x_p \leq \lfloor x_p^*(P_i) \rfloor$ and $x_p \geq \lceil x_p^*(P_i) \rceil$, respectively.

5. Bound P'.
   - Solve LP relaxation of P'.
   - If $c^T x^*(P') < OPT$
     - If $x^*(P')$ feasible for P_0
       - Set $OPT = c^T x^*(P')$ and $x_{OPT} = x^*(P')$.
     - Else add P' to list for further inspection.

6. Bound P''. (Identical to step 5, but for P'') Go back to step 2.

Branch & Bound for MILPs

Output:

- $OPT = \infty$: P_0 is infeasible.
- $OPT < \infty$: P_0 is feasible and OPT its optimal value.

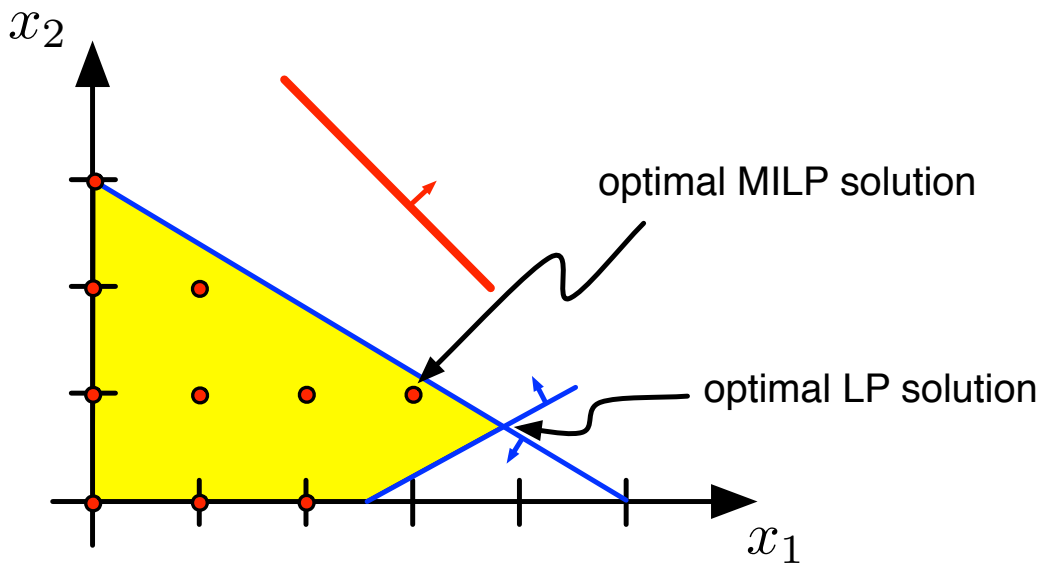
Optimal Solution: x_{OPT} associated to OPT.

Termination: Under assumption of finite bounds $\underline{x}_j$, $\overline{x}_j$ for $j \in Z$, algorithm terminates in finitely many steps.

Example 1: Problem Statement [1/15]

Example:

$$\begin{aligned} \text{maximise} \quad & x_1 + x_2 \\ \text{subject to} \quad & 3x_1 + 5x_2 \leq 15 \\ & x_1 - 2x_2 \leq \tfrac{5}{2} \\ & x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0 \end{aligned}$$

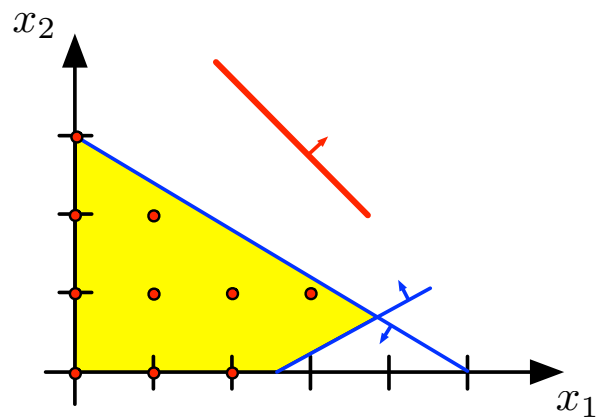


Example 1: Initialise [2/15]

Example:

$$\begin{aligned} \text{maximise} \quad & x_1 + x_2 \\ \text{subject to} \quad & 3x_1 + 5x_2 \leq 15 \\ & x_1 - 2x_2 \leq \tfrac{5}{2} \\ & x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0 \end{aligned}$$

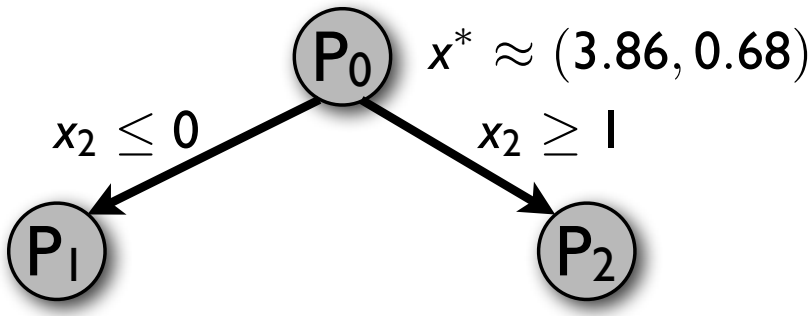
$x^* \approx (3.86, 0.68)$

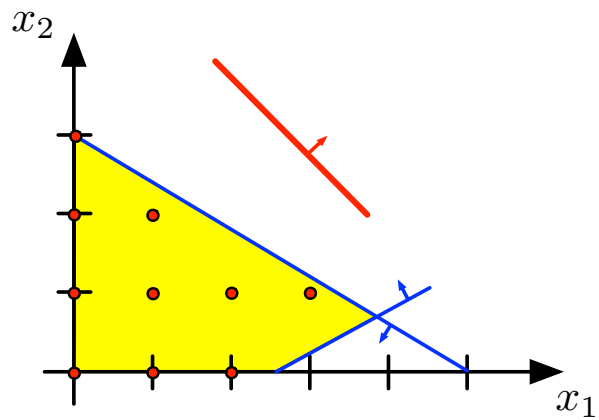


Example 1: Branch [3/15]

Example:

$$\begin{aligned} \text{maximise} \quad & x_1 + x_2 \\ \text{subject to} \quad & 3x_1 + 5x_2 \leq 15 \\ & x_1 - 2x_2 \leq \tfrac{5}{2} \\ & x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0 \end{aligned}$$





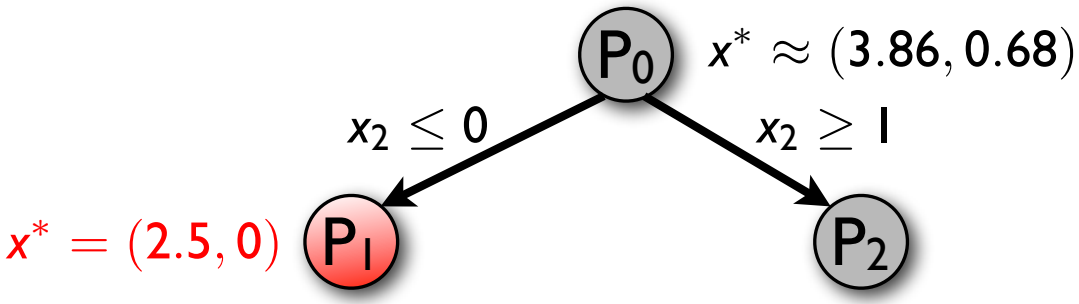
Example 1: Bound [4/15]

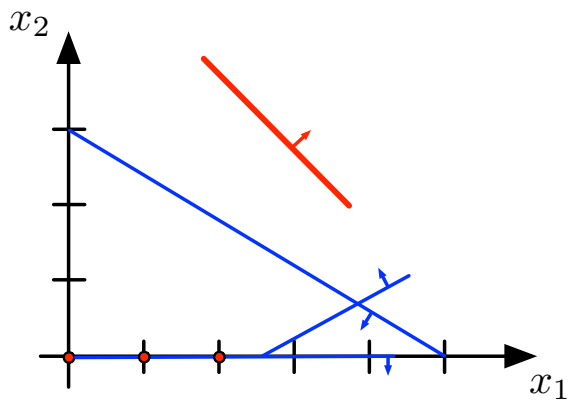
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$

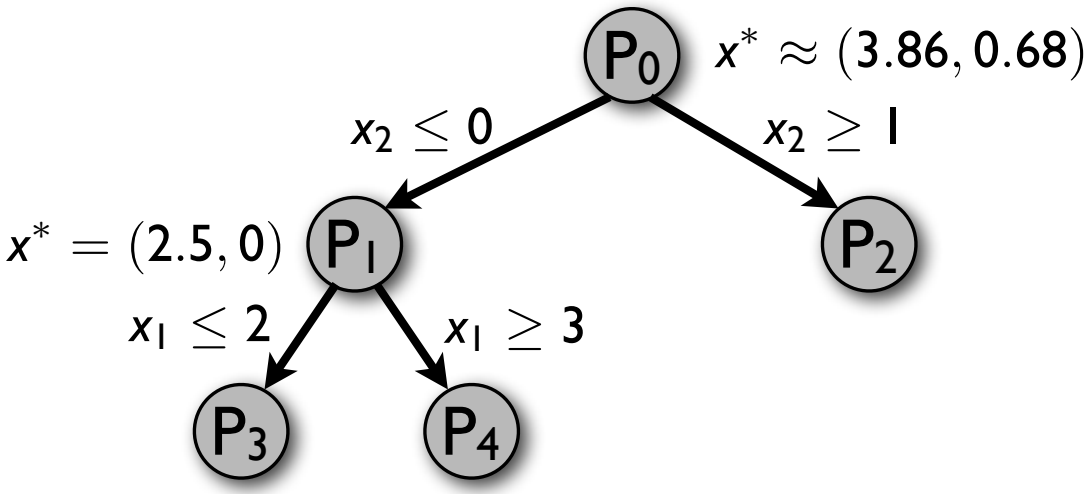


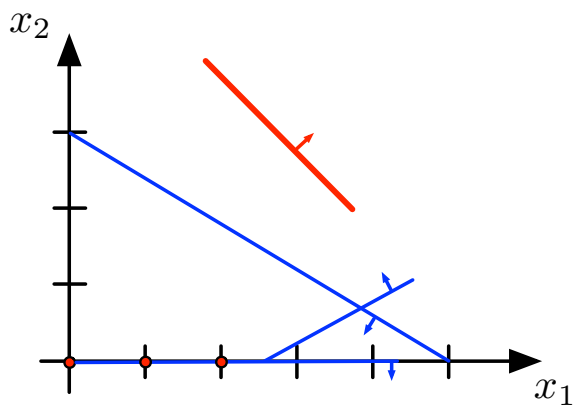


Example 1: Branch [5/15]

Example:

$$\begin{aligned} \text{maximise} \quad & x_1 + x_2 \\ \text{subject to} \quad & 3x_1 + 5x_2 \leq 15 \\ & x_1 - 2x_2 \leq \tfrac{5}{2} \\ & x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0 \end{aligned}$$





Sanity Check. Did we have to take this branching step?

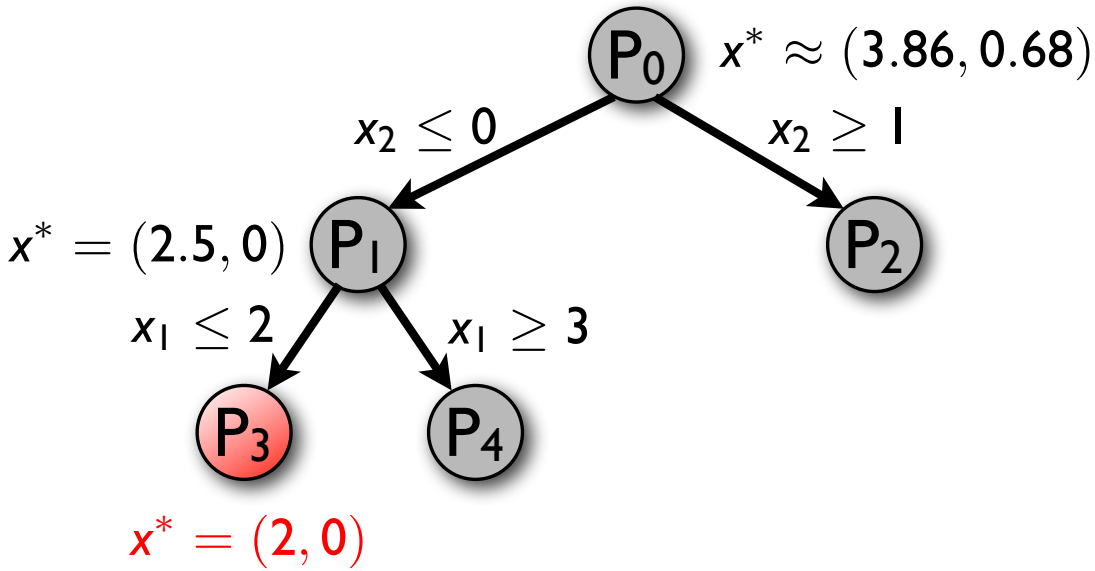
Example 1: Bound [6/15]

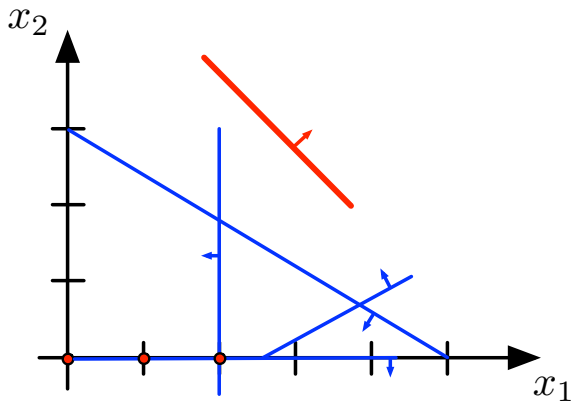
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$

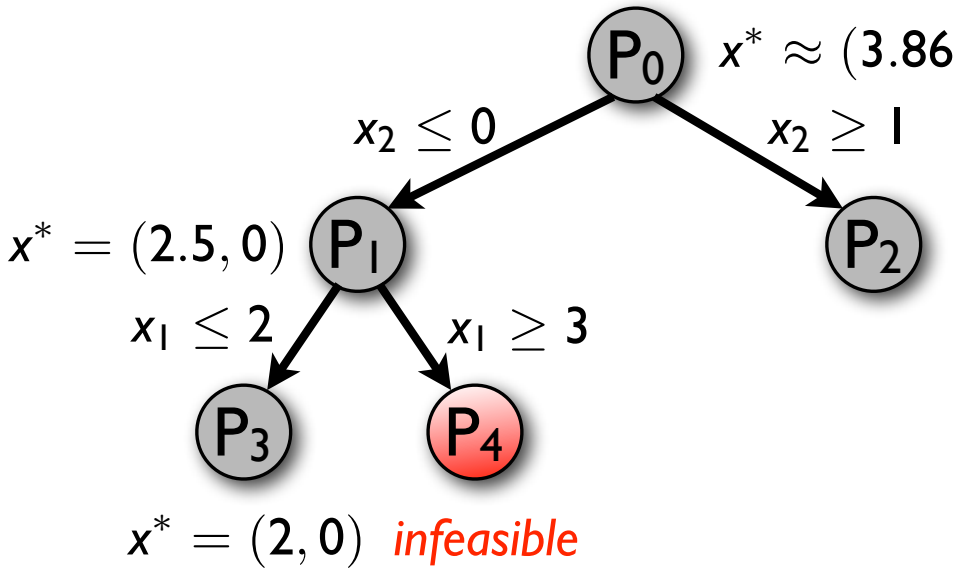


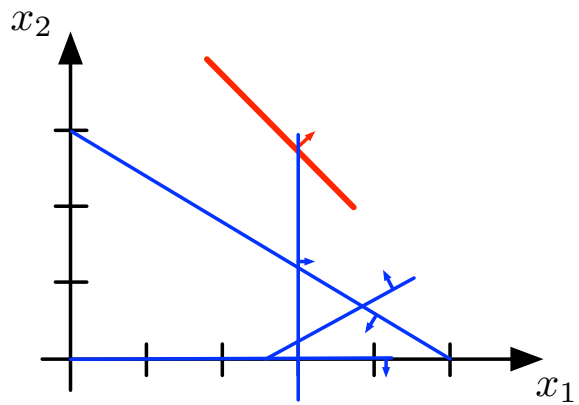


Example 1: Bound [7/15]

Example:

$$\begin{aligned} \text{maximise} \quad & x_1 + x_2 \\ \text{subject to} \quad & 3x_1 + 5x_2 \leq 15 \\ & x_1 - 2x_2 \leq \tfrac{5}{2} \\ & x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0 \end{aligned}$$





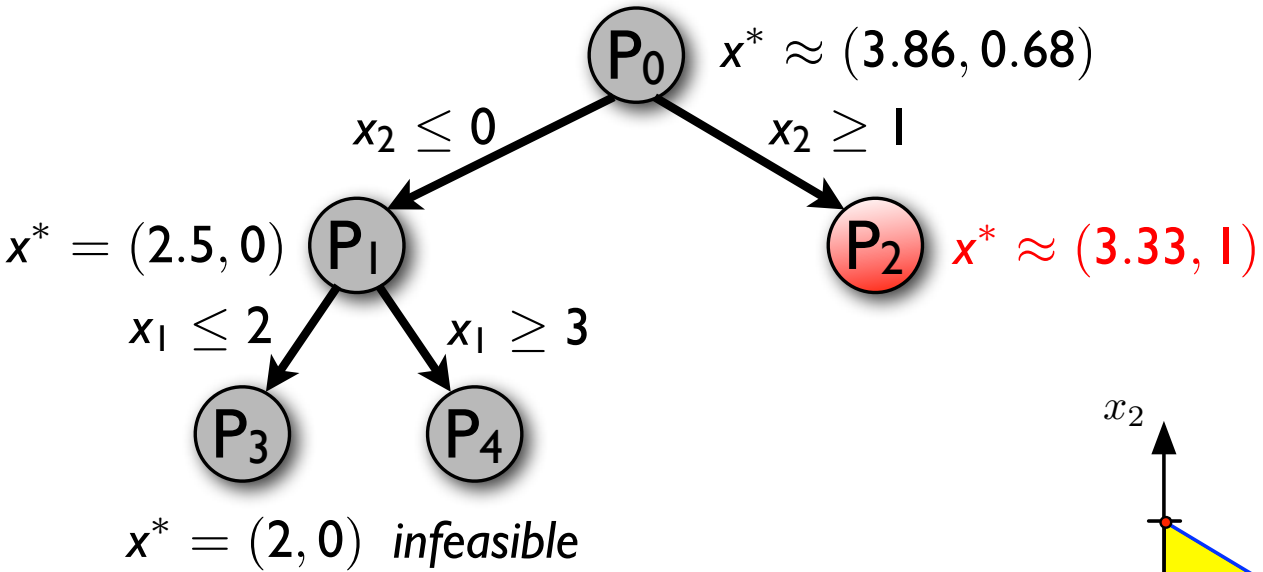
Example 1: Bound [8/15]

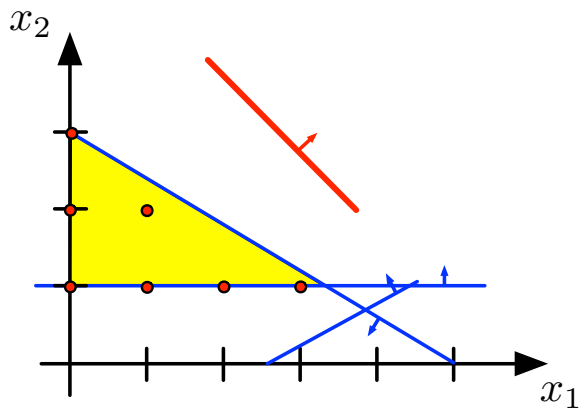
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$





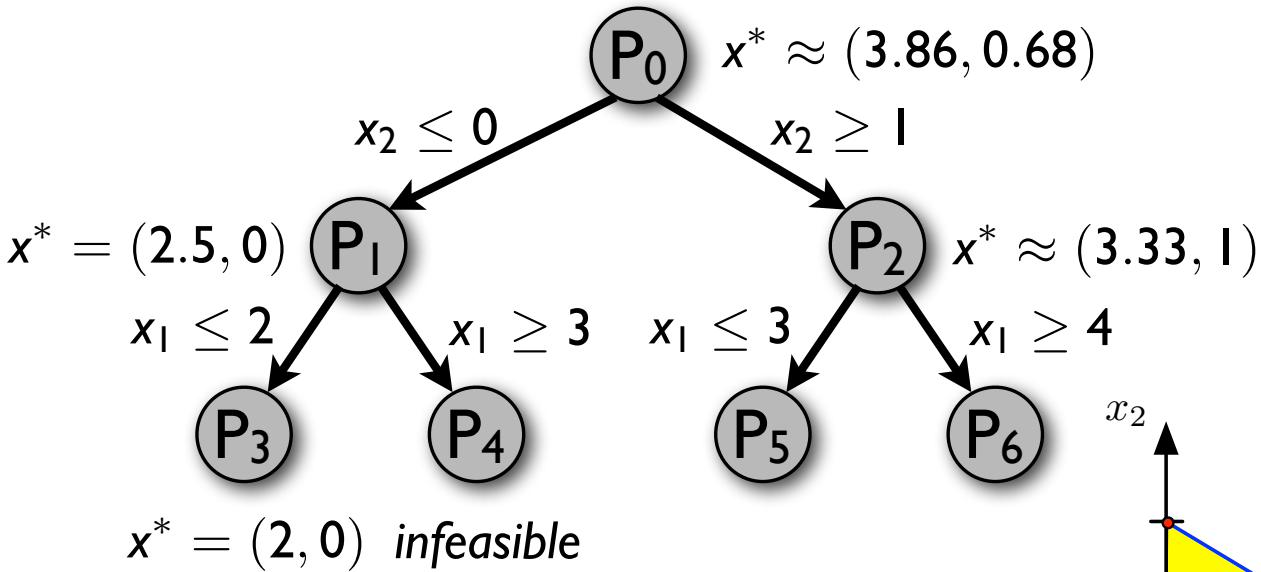
Example 1: Branch [9/15]

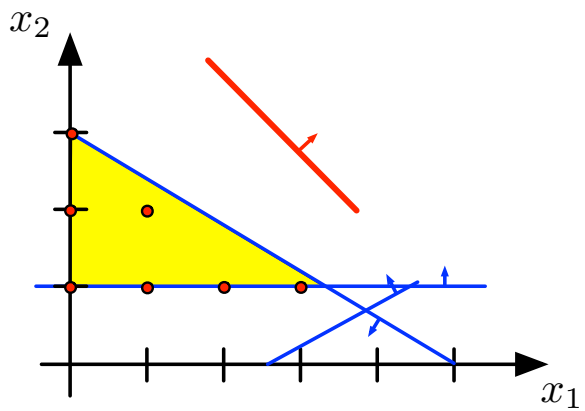
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$





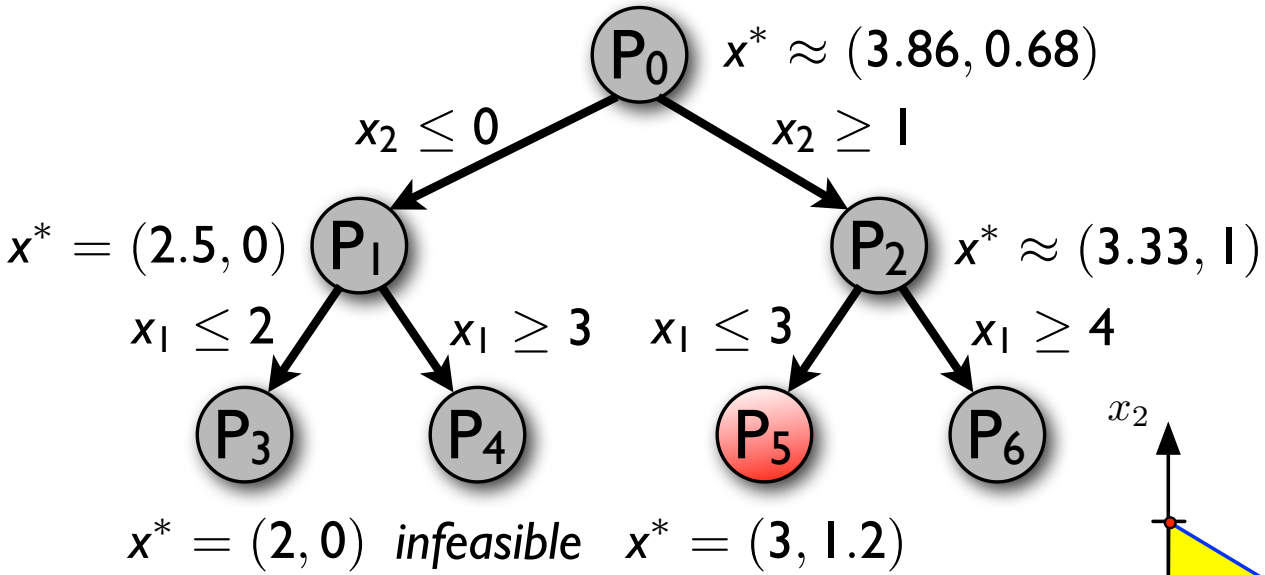
Example 1: Bound [10/15]

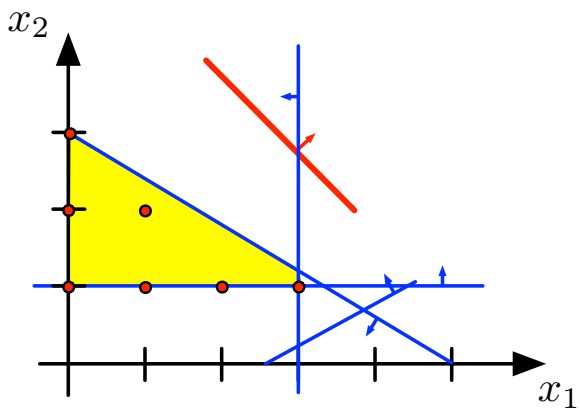
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$





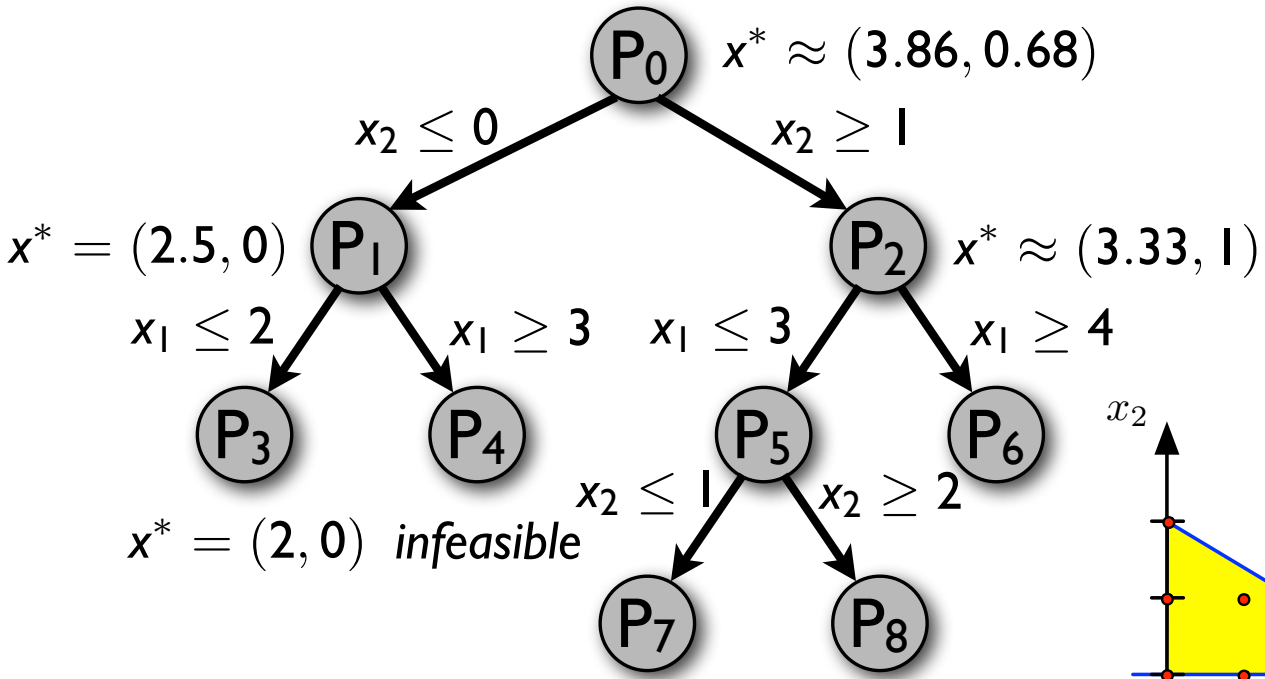
Example 1: Branch [11/15]

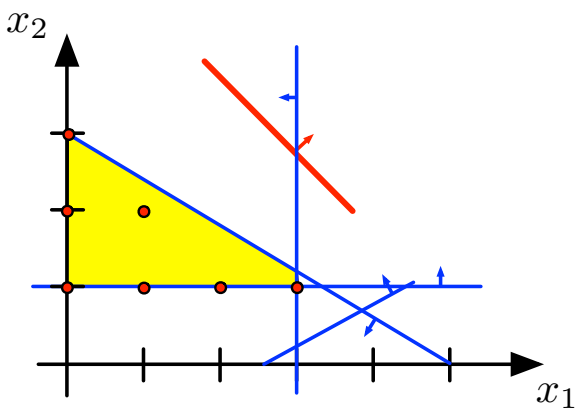
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$





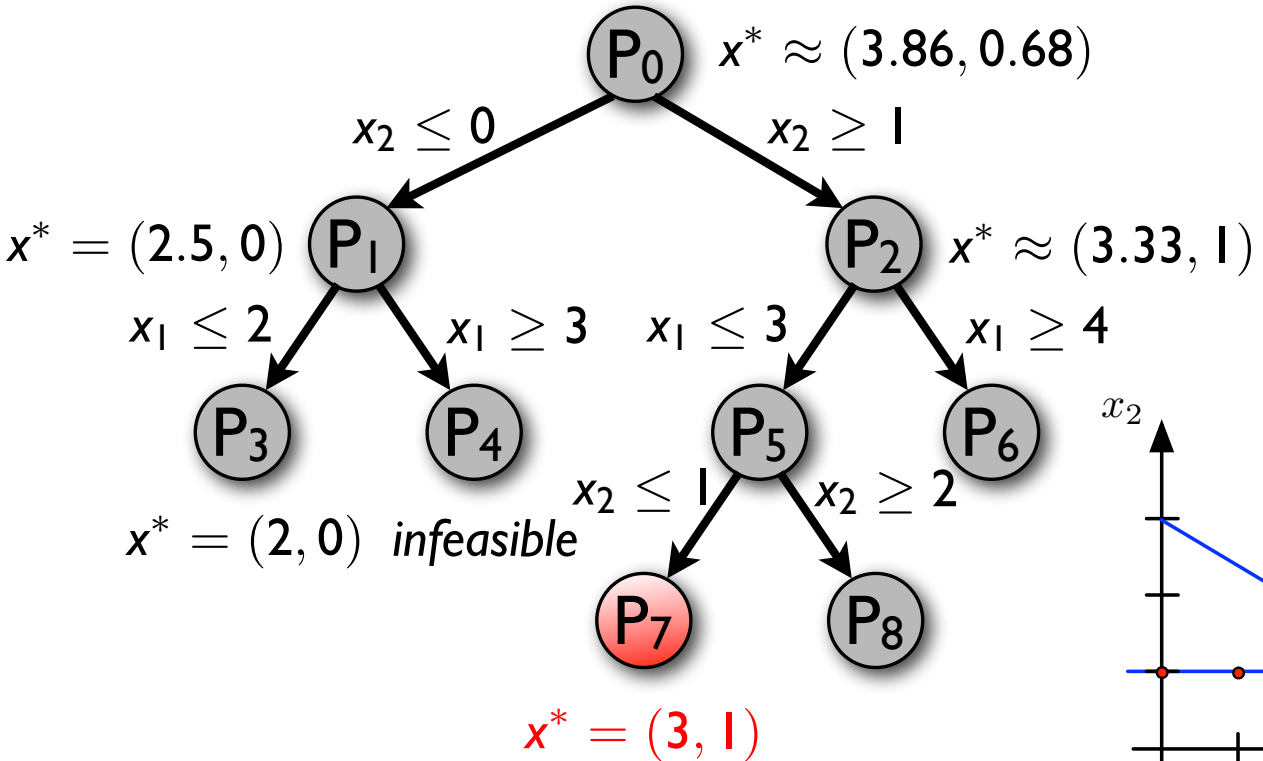
Example 1: Bound [12/15]

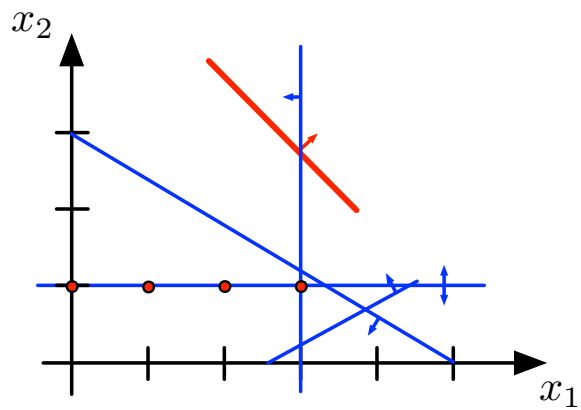
Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$





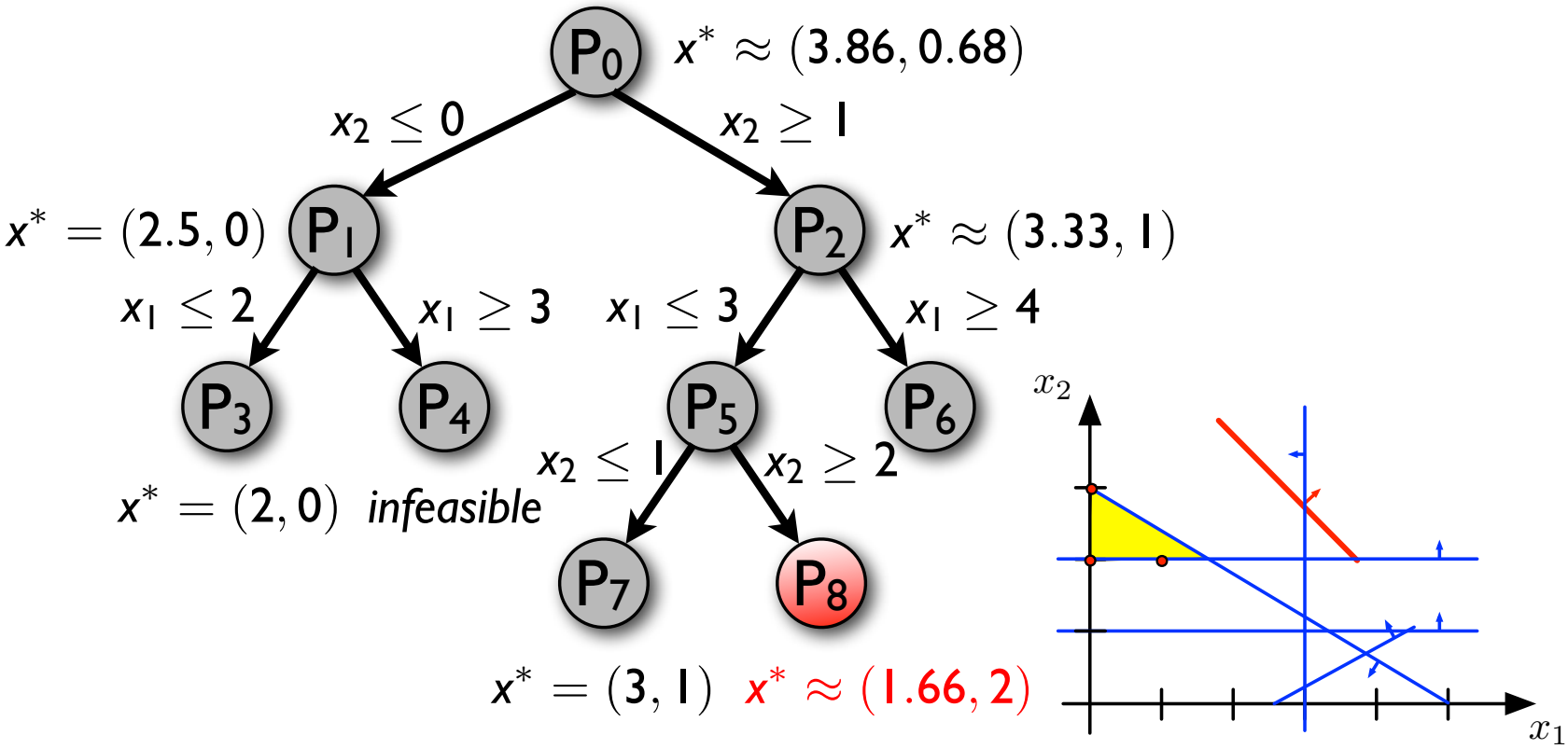
Example 1: Bound [13/15]

Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$



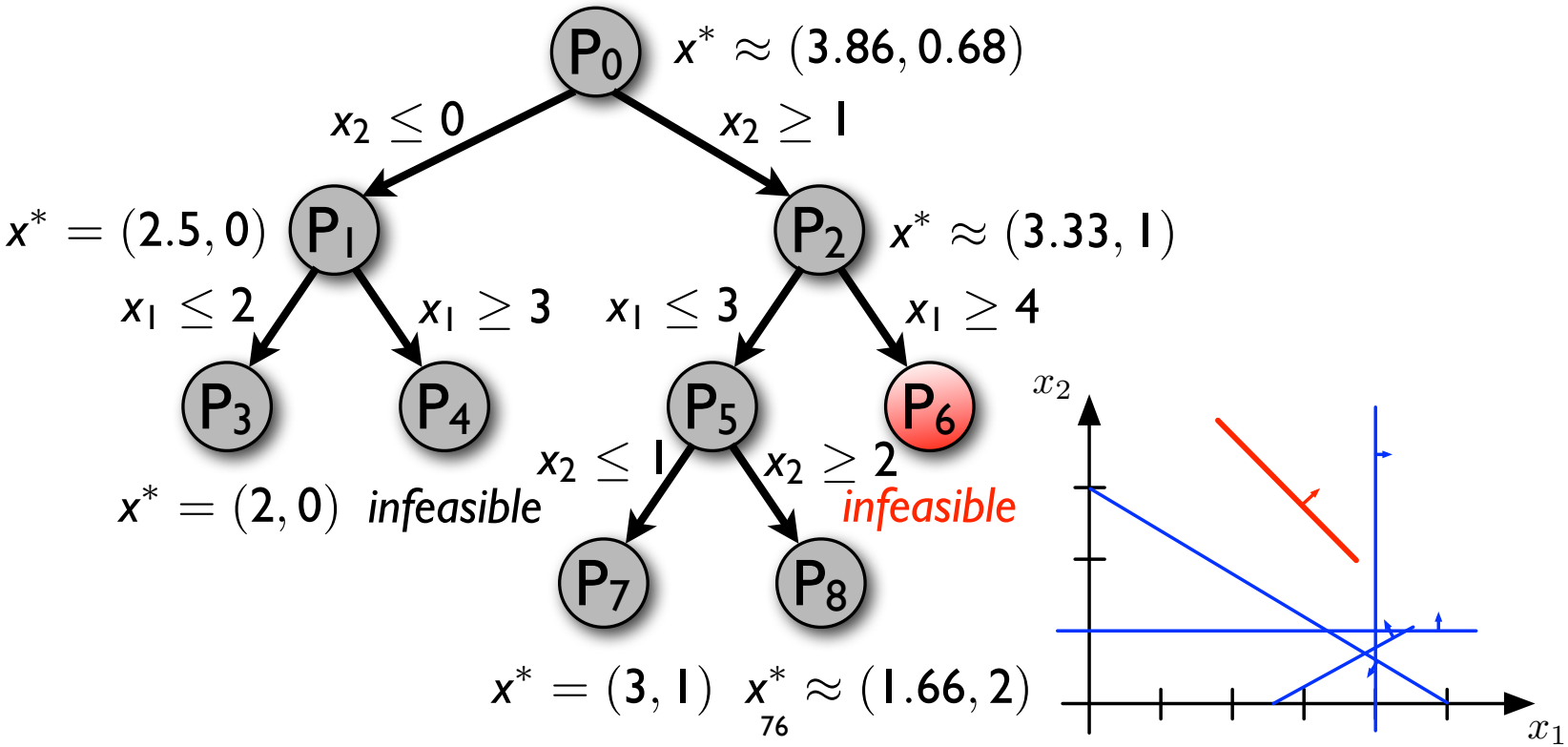
Example 1: Bound [14/15]

Example: maximise $x_1 + x_2$

subject to $3x_1 + 5x_2 \leq 15$

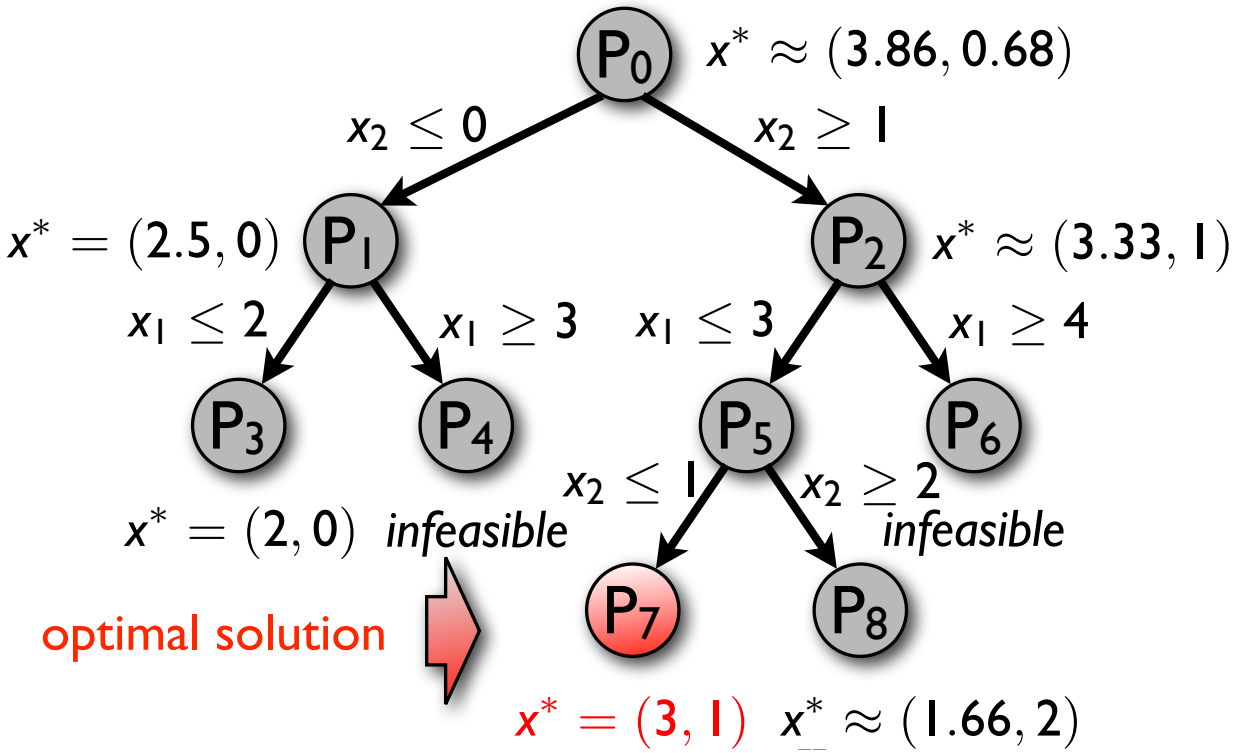
$x_1 - 2x_2 \leq \frac{5}{2}$

$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$



Example 1: Terminate [15/15]

Example: maximise $x_1 + x_2$
subject to $3x_1 + 5x_2 \leq 15$
$x_1 - 2x_2 \leq \frac{5}{2}$
$x_1, x_2 \geq 0, \quad x_1, x_2 \in \mathbb{N}_0$



Sanity Check. Are there methods we could have used to explore this tree more effectively?

Example 2 [1/11]

Assume the following problem is given:

$$\max\ 2x_1 + 3x_2 + x_3 + 2x_4$$

subject to

$$\begin{aligned} 5x_1 + 2x_2 + x_3 + x_4 &\leq 15 \\ 2x_1 + 6x_2 + 10x_3 + 8x_4 &\leq 60 \\ x_1 + x_2 + x_3 + x_4 &\leq 8 \\ 2x_1 + 2x_2 + 3x_3 + 3x_4 &\leq 16. \end{aligned}$$

The bounds are $x_1 \in [0, 3]$, $x_2 \in [0, 7]$, $x_3 \in [0, 5]$ and $x_4 \in [0, 5]$. Furthermore, $x_j \in \mathbb{N}_0$ for all $j = 1, \ldots, 4$.

Example 2 [2/11]

Change to minimisation objective:

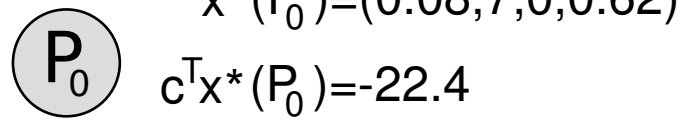
$$\min \ -2x_1 - 3x_2 - x_3 - 2x_4$$

subject to

$$\begin{aligned} 5x_1 + 2x_2 + x_3 + x_4 &\leq 15 \\ 2x_1 + 6x_2 + 10x_3 + 8x_4 &\leq 60 \\ x_1 + x_2 + x_3 + x_4 &\leq 8 \\ 2x_1 + 2x_2 + 3x_3 + 3x_4 &\leq 16. \end{aligned}$$

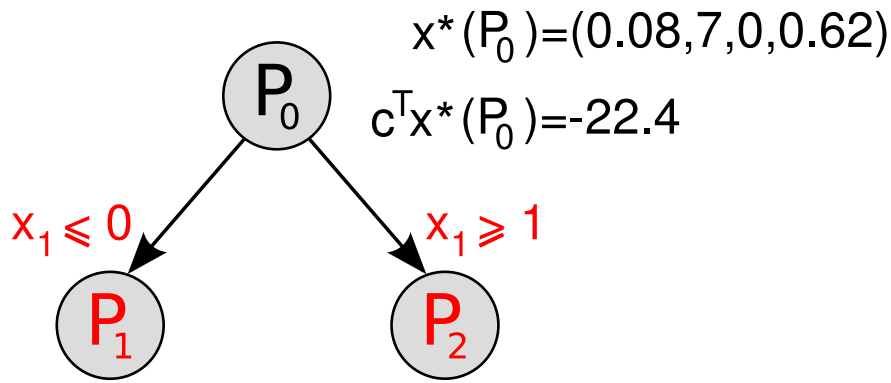
$x_1 \in [0, 3]$, $x_2 \in [0, 7]$, $x_3 \in [0, 5]$ and $x_4 \in [0, 5]$. $x_j \in \mathbb{N}_0$ for all $j = 1, \ldots, 4$.

Example 2: Initialisation [3/11]



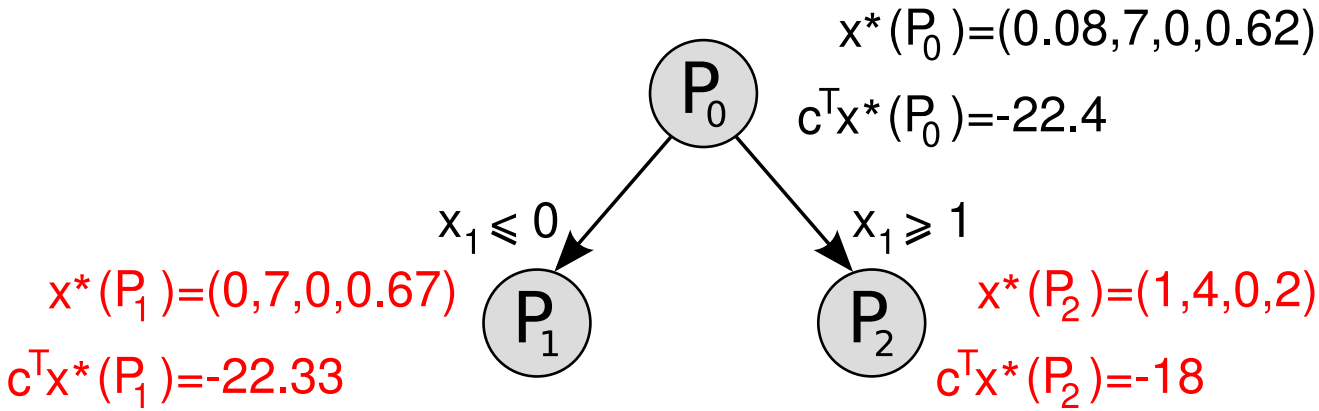
Problem list: $\{P_0\}$, $OPT = \infty$.

Example 2: Branch [4/11]



Problem list: $\{\}$, $OPT = \infty$.

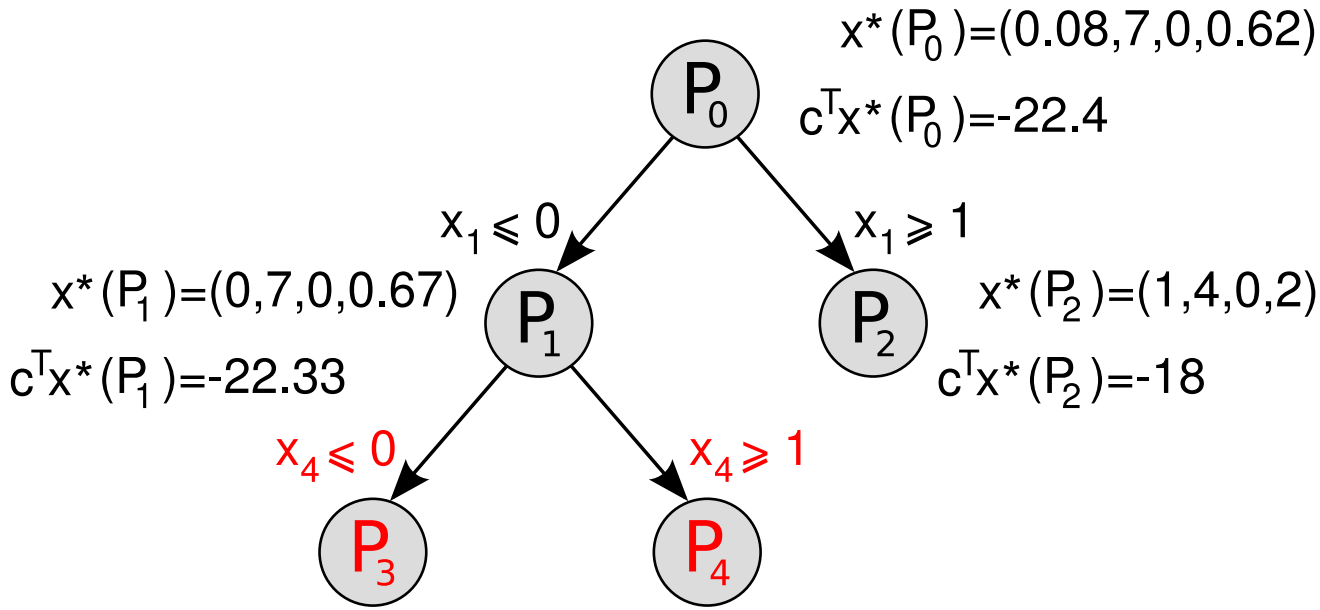
Example 2: Bound [5/11]



Problem list: $\{P_1\}$, $OPT = -18$, $x_{OPT} = x^*(P_2)$.

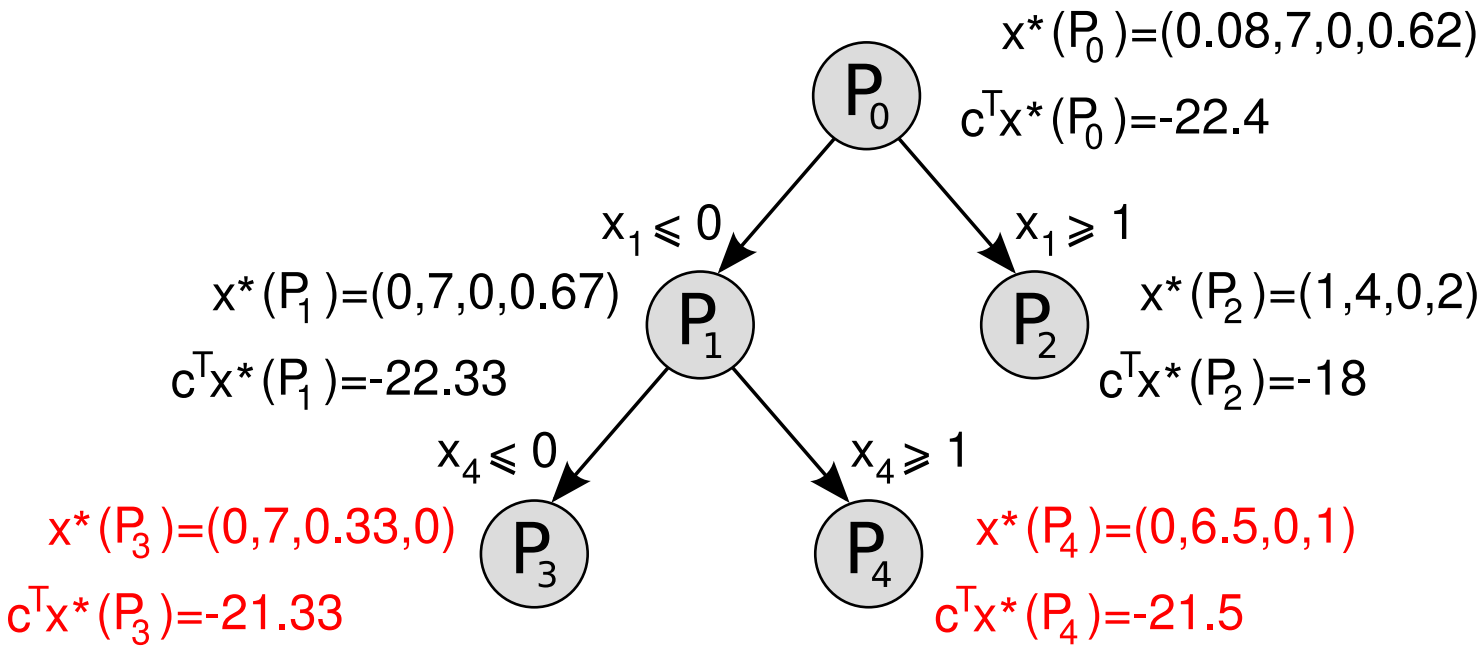
Sanity Check. Why can we stop exploring the child nodes of P_2?

Example 2: Branch [6/11]



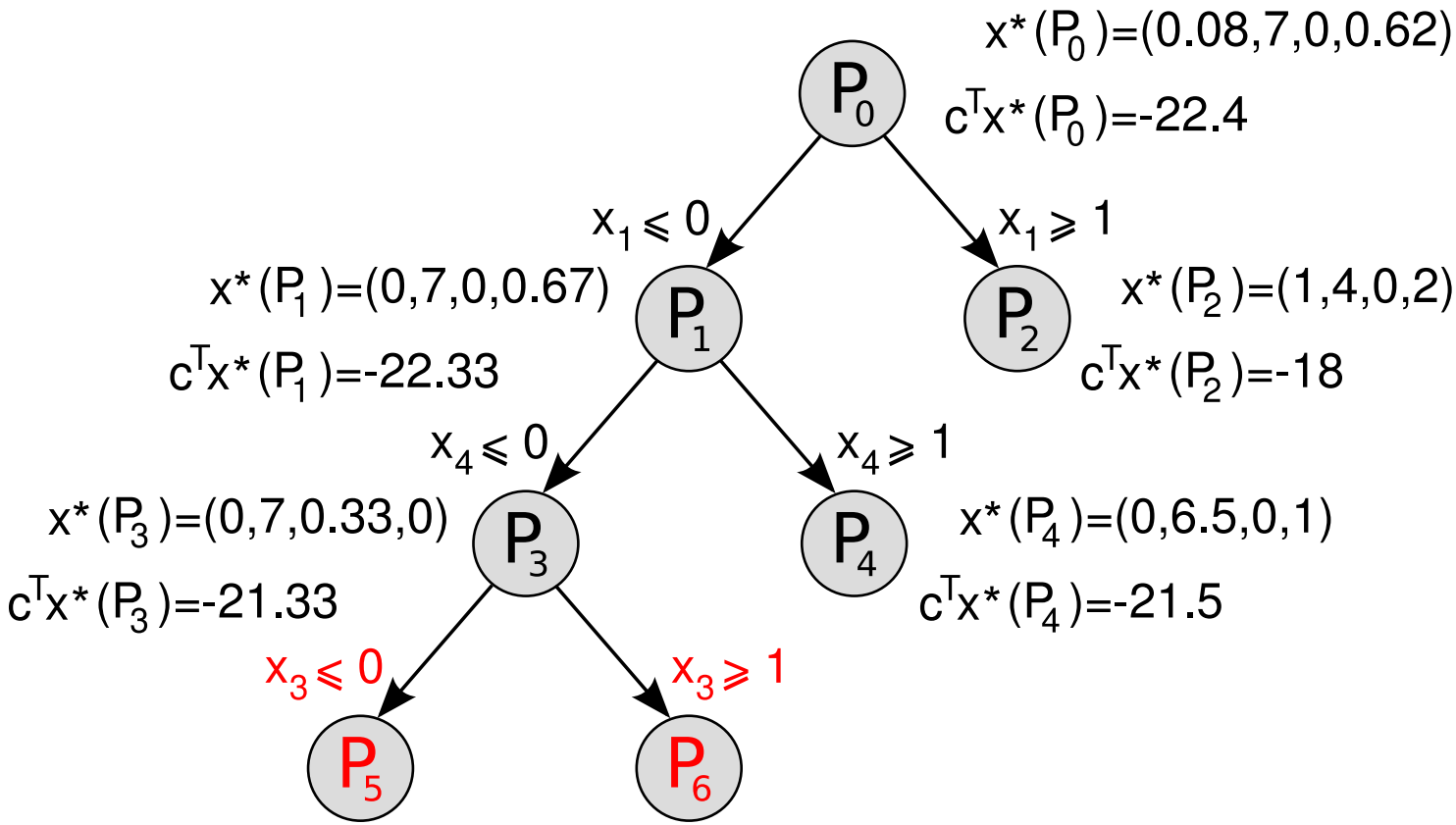
Problem list: $\{\}$, $OPT = -18$, $x_{OPT} = x^*(P_2)$.

Example 2: Bound [7/11]



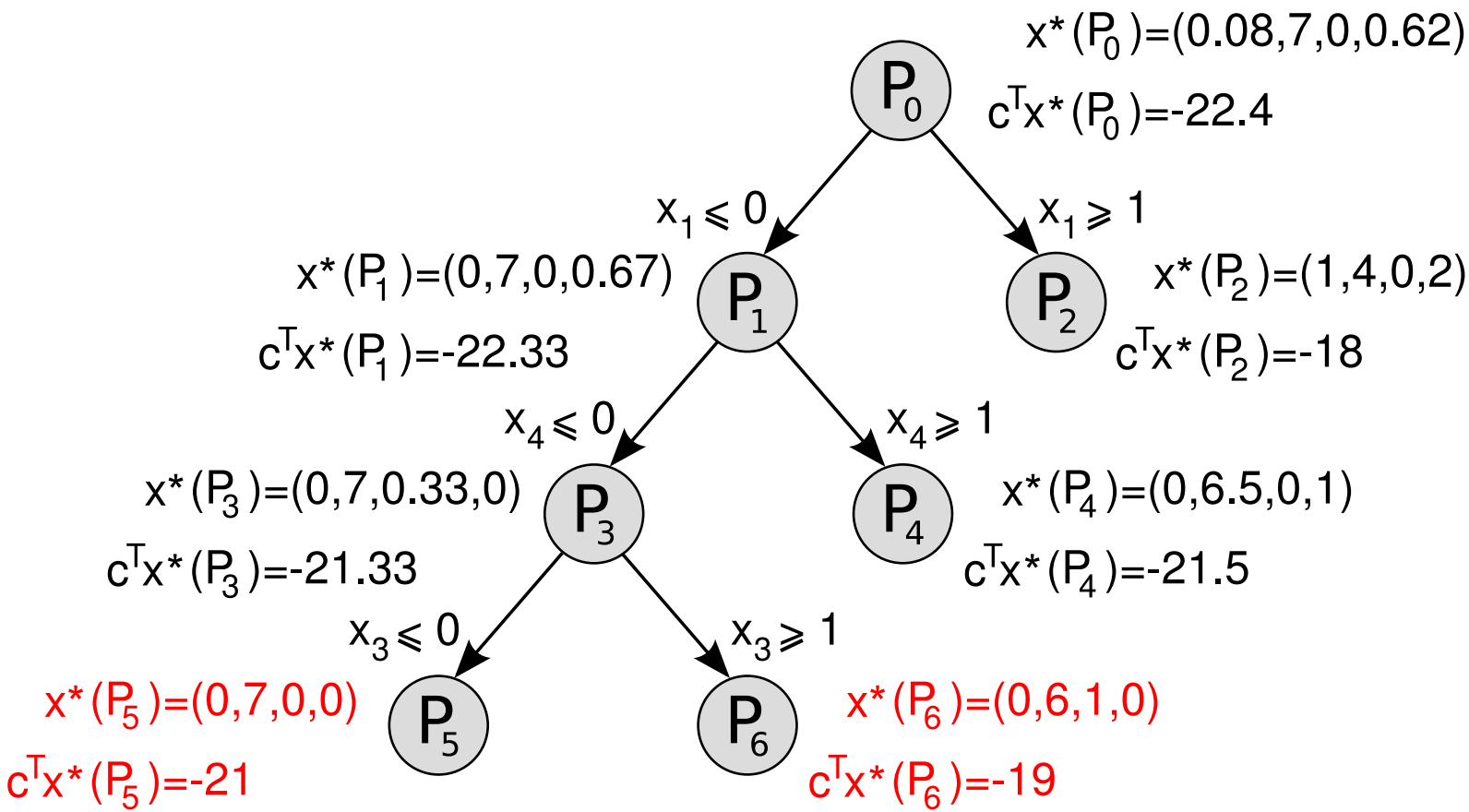
Problem list: $\{P_3, P_4\}$, $OPT = -18$, $x_{OPT} = x^*(P_2)$.

Example 2: Branch [8/11]



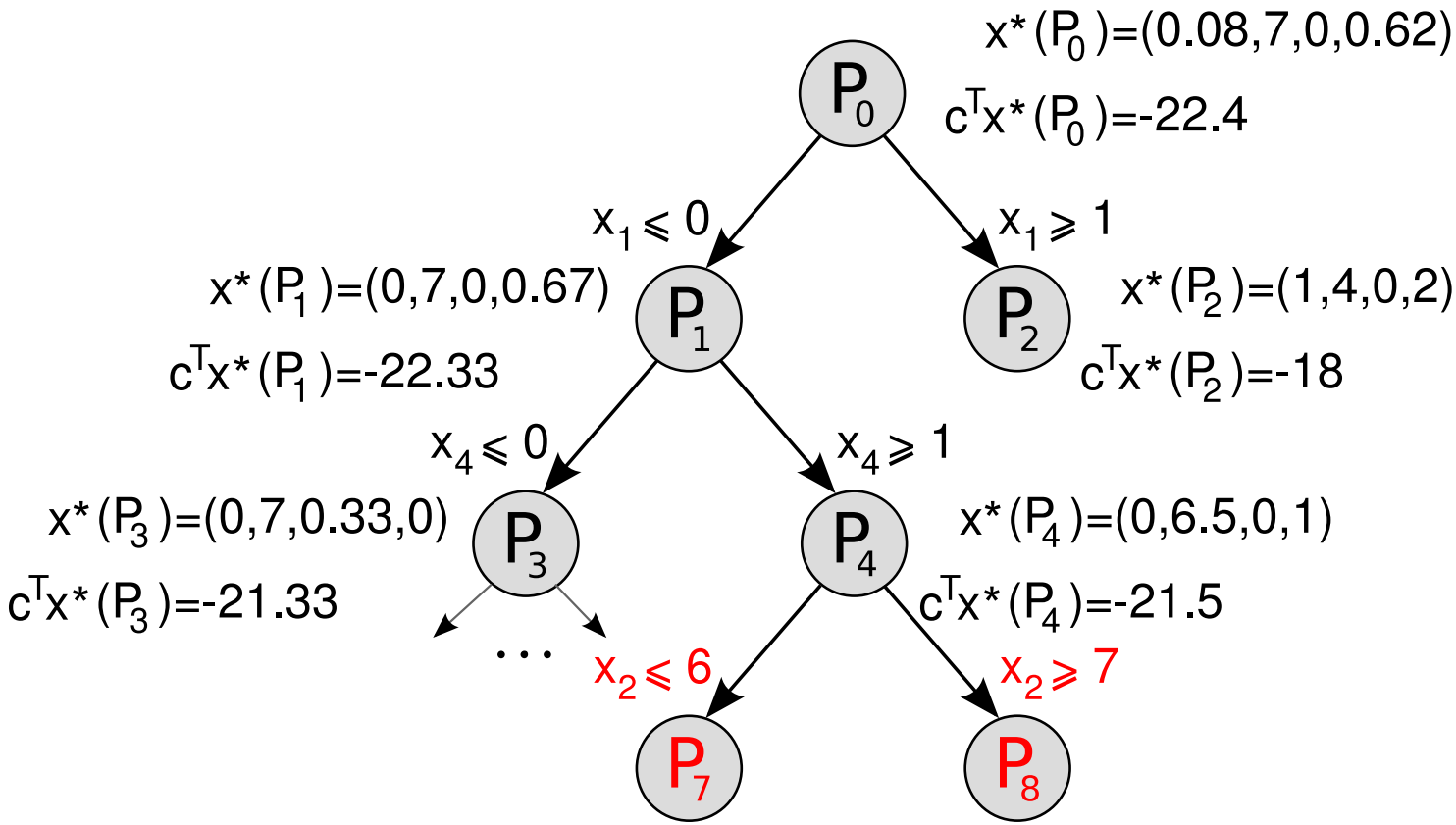
Problem list: $\{P_4\}$, $OPT = -18$, $x_{OPT} = x^*(P_2)$.

Example 2: Bound [9/11]



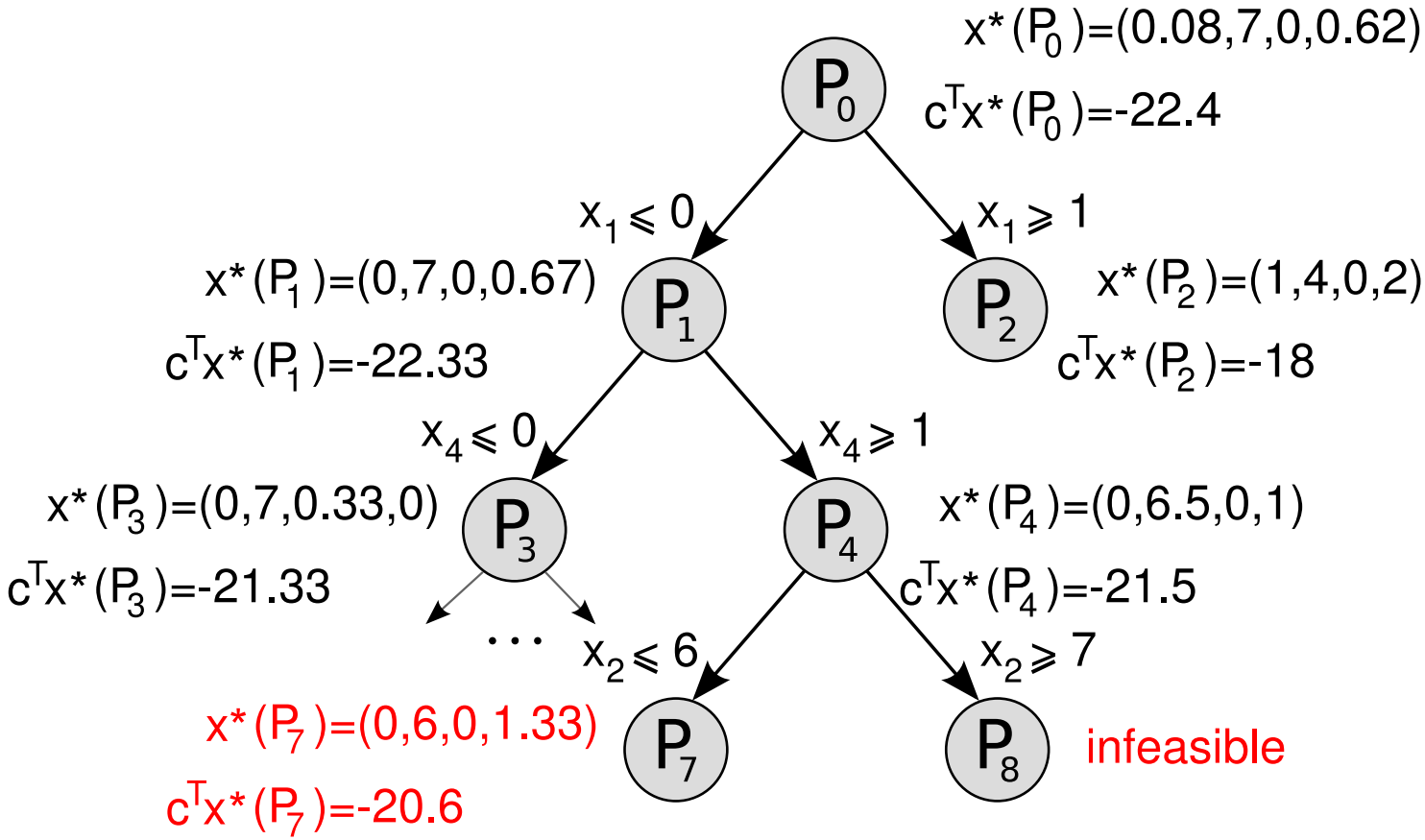
Problem list: $\{P_4\}$, $OPT = -21$, $x_{OPT} = x^*(P_5)$.
(Note: the P_4 branch can at best match $OPT = -21$, since $c^T x^*(P_4) = -21.5$.)

Example 2: Branch [10/11]



Problem list: $\{\}$, $OPT = -21$, $x_{OPT} = x^*(P_5)$.

Example 2: Bound [11/11]



Problem list: $\{\}$, STOP: $OPT = -21$, $x_{OPT} = (0, 7, 0, 0)$.

Branch-and-Cut

MILP solvers typically use a variant of branch-and-bound called branch-and-cut.

- Branch-and-cut combines branch-and-bound and (typically several types of) cutting plane algorithms.
- At each search tree node, cuts are dynamically generated and added to the LP relaxation to tighten the feasible set.
- Cuts are further categorised into local cuts that are valid only for a subtree of the branch-and-bound tree, and global cuts that are valid for all nodes.

Typically outperforms stand-alone methods, e.g. branch-and-bound and cutting plane algorithms.