Optimization: Integer Optimization

Integer optimization

Integer optimization:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) \leqslant 0 \end{aligned} \qquad \text{where} \quad x = \begin{bmatrix} x_{\mathrm{r}} \\ x_{\mathrm{i}} \end{bmatrix} \quad x_{\mathrm{r}} \in \mathbb{R}^{n_{\mathrm{r}}}, \; x_{\mathrm{i}} \in \mathbb{Z}^{n_{\mathrm{i}}}$$

Types of integer optimization problems:

- Mixed integer optimization problem: $n_{\mathrm{r}}, n_{\mathrm{i}} > 0$
- All integer optimization problem: $n_{\mathrm{r}} = 0$
- Mixed integer linear programming (MILP):

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{s.t.} \quad & Ax \leqslant b \end{aligned} \qquad \text{where} \quad x = \begin{bmatrix} x_{\mathrm{r}} \\ x_{\mathrm{i}} \end{bmatrix} \quad x_{\mathrm{r}} \in \mathbb{R}^{n_{\mathrm{r}}}, \; x_{\mathrm{i}} \in \mathbb{Z}^{n_{\mathrm{i}}}$$

Approximation using real values only?

Integer parameters may represent various quantities:

1. number of smart phones produced by a factory per year
   $\rightarrow$ approximation by a real value will probably cause negligible errors

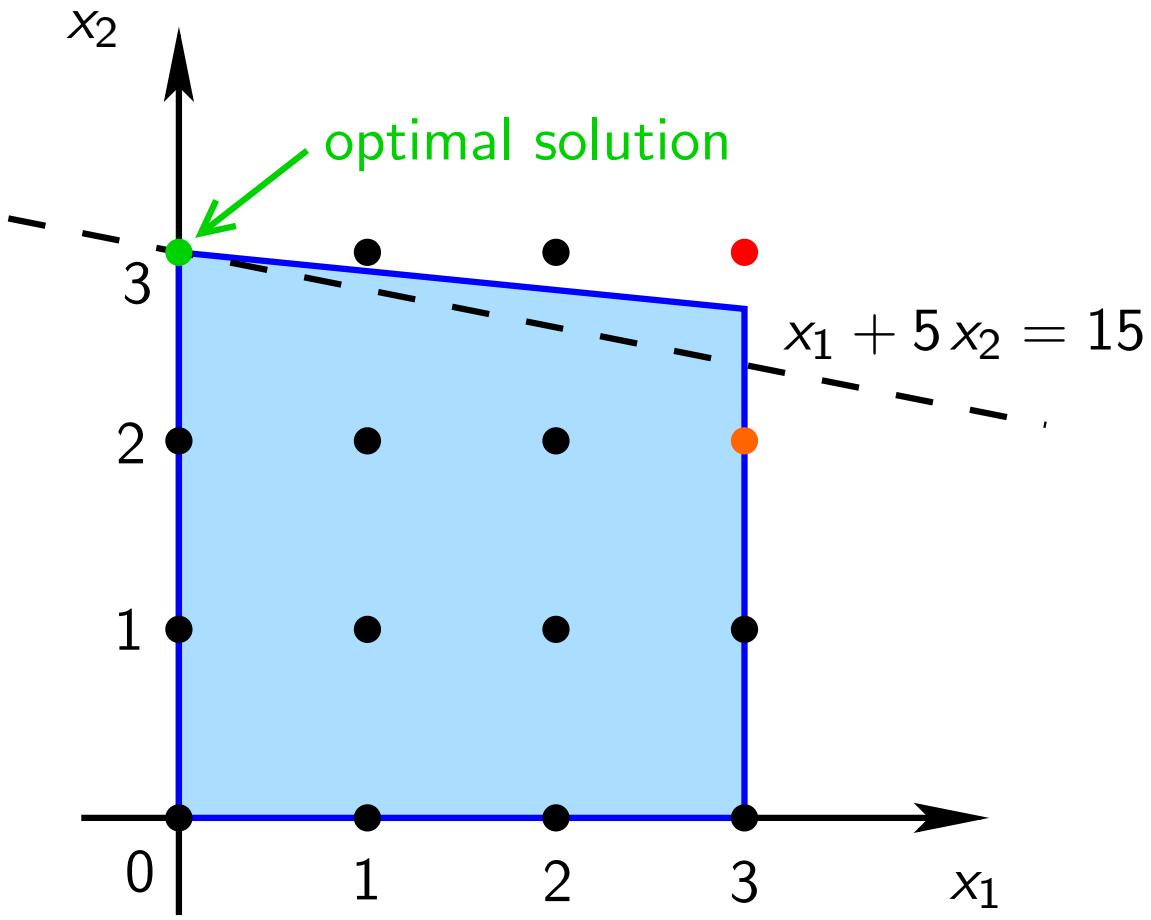
2. number of students that pass the "Modeling and Control of Hybrid Systems" examination
   $\rightarrow$ may or may not be approximated by a real value

3. position of a switch in an electric circuit
   $\rightarrow$ cannot be approximated properly by a real value

Approximation using real values only? (continued)

$$\max_x \; x_1 + 5x_2$$

$$\text{s.t.} \quad x_1 + 10x_2 \leqslant 30$$
$$x_1 \leqslant 3$$
$$x_1, x_2 \geqslant 0$$
$$x_1, x_2 \in \mathbb{Z}$$

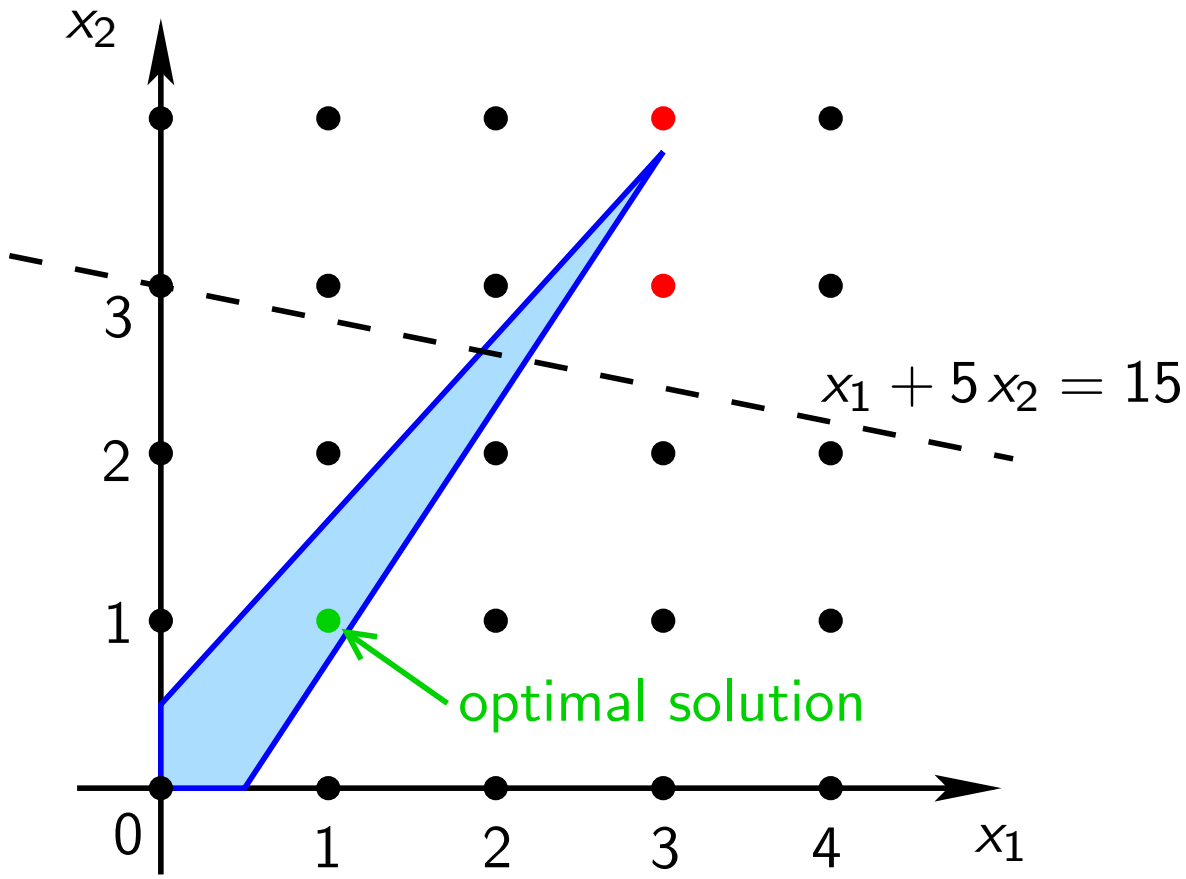


$\rightarrow$ rounded solutions not optimal

Approximation using real values only? (continued)

$$\max_x \; x_1 + 5x_2$$

$$\text{s.t.} \quad 38x_1 - 25x_2 \leqslant 19$$
$$-33x_1 + 30x_2 \leqslant 15$$
$$x_1, x_2 \geqslant 0$$
$$x_1, x_2 \in \mathbb{Z}$$



$\rightarrow$ rounded solutions not feasible

Complexity

Use enumeration?

combinatorial explosion
$\leftrightarrow$ growth in computer power / parallel computing

Example:

Scheduling 10 batches, 10 machines, 5 days (=120 h)
Allocate machines to batches for each slot of 1 hour:

$$x_{ijk} = \begin{cases} 1 & \text{if batch } i \text{ uses machine } j \text{ in slot } k \\ 0 & \text{otherwise} \end{cases}$$

possibilities: $2^{10 \cdot 10 \cdot 120} \approx 10^{3612}$

So what?? ... just use fast computer with parallel processors

Complexity (continued)

- Processor at size of proton
- # processors = # protons in universe
- Clock period = time for light to traverse proton
- Run time is 10 $\times$ current age of universe

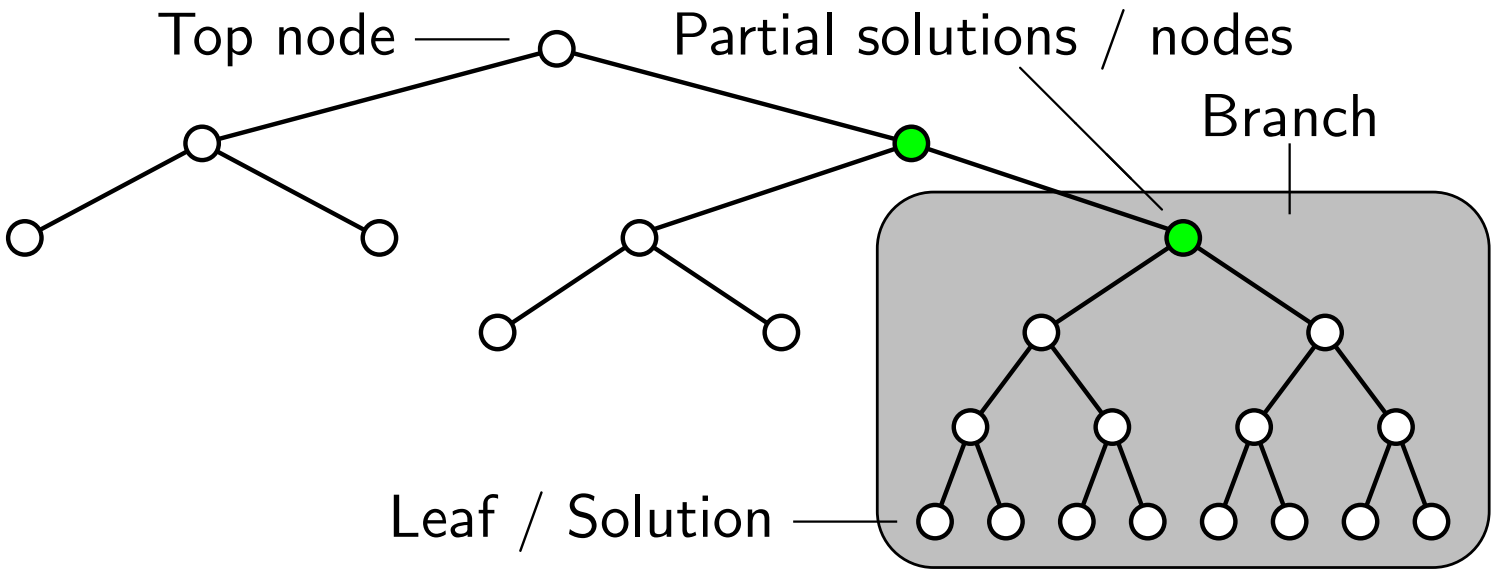
Number of evaluations $\approx 10^{168} \quad \leftrightarrow \quad 10^{3612}$ required

$\Rightarrow$ enumeration fails even for small-sized problems

Search

Essence of combinatorial optimization: searching through a tree

- Start at the top node: no decisions have been made yet
- Each decision results in an arc leading to a child node
- Leaf: no additional decisions can be made (each leaf represents a potential solution)



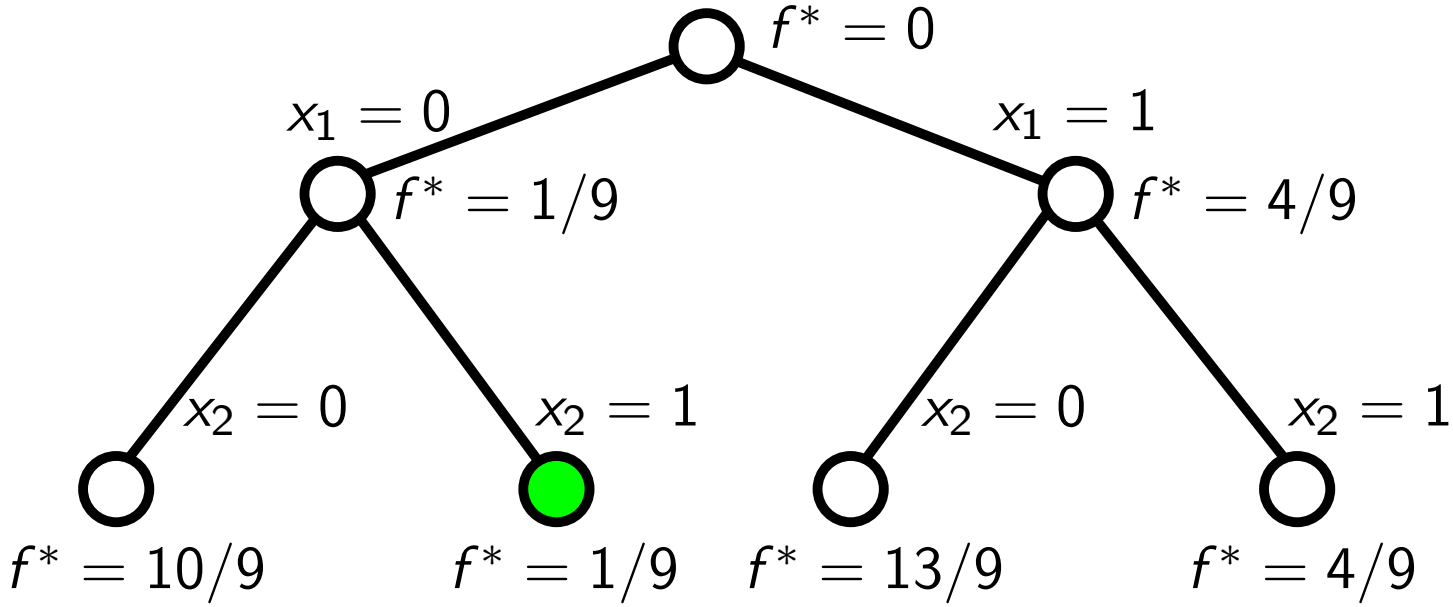
Search tree: Example

Consider

$$\min_{x_1,x_2\in\{0,1\}} f(x_1,x_2) = \min_{x_1,x_2} \left(x_1 - \frac{1}{3}\right)^2 + (x_2 - 1)^2$$

Search tree:

- Root node: x_1, x_2 both real-valued
- Intermediate layer: x_1 fixed, x_2 real-valued
- Leaf nodes: x_1, x_2 both fixed



Tree search

Categorization of optimization techniques based on basic tasks in tree searching:

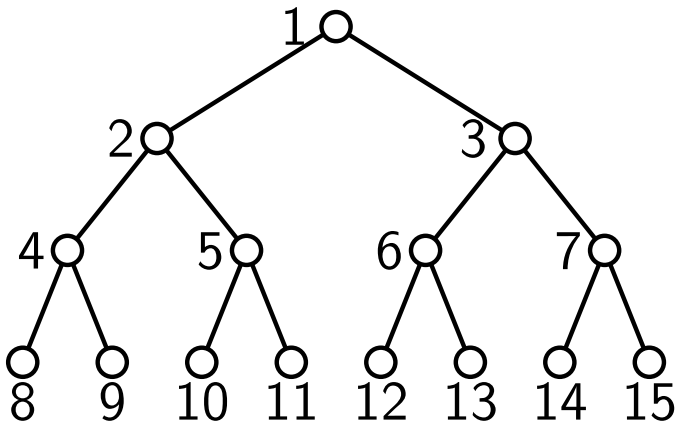
- search strategy
- test for the feasibility and optimality of leaves (i.e., of fully determined solutions)

Theoretically, these two are sufficient

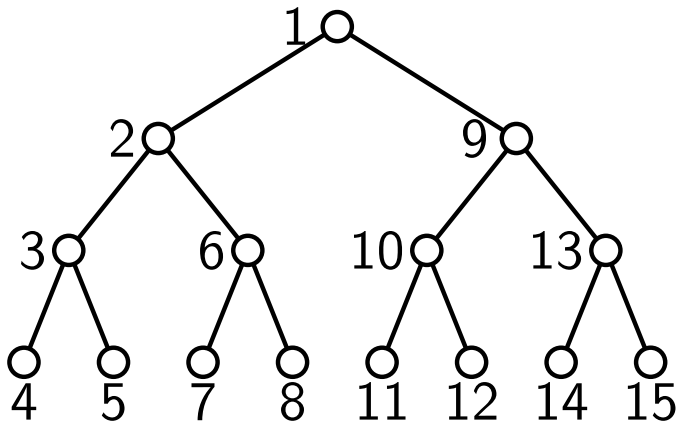
In order to improve speed of search, four basic refinements can be recognized:

1. *branch pruning*
2. branch merging
3. search rearrangement
4. problem decomposition

Search strategy



a. Breadth-first search



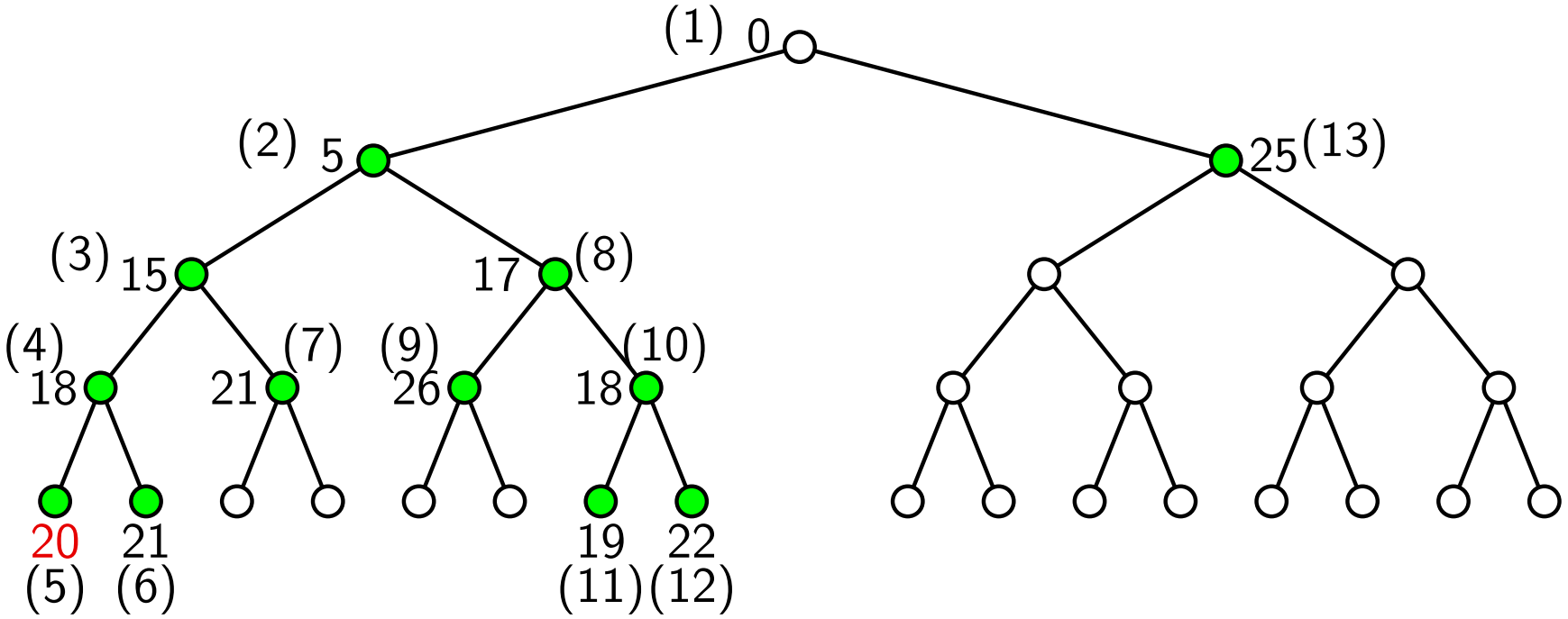
b. Depth-first search

a. Breadth-first search
   → requires large amounts of memory
b. Depth-first search
   → generates single candidate solutions step by step
c. Combination of breadth-first and depth-first
d. Generate fully determined candidate solution in one step
   → only leaves are generated and tested; used in many nonlinear programming solvers (e.g., hill-climbing and stochastic search)

Refinement: Branch pruning

- Branch pruning is obtained by test for feasibility and optimality of nodes (i.e., partial solutions)
- If partial solution is rejected, none of its descendants need to be evaluated. Therefore, a single test (at node) can replace many tests (at all descendants of the node)
- Three basic tests:
  1. Test of feasibility of the decisions made so far
  2. Test of feasibility of the descendants of this partial solution
  3. Evaluation of optimality of a partial solution
     (if lower bound for optimum in branch is larger than best solution so far)

Refinement: Branch pruning (continued)



Branch-and-Bound

- First depth-first search yields solution with cost 20. So $f^* \leftarrow 20$
- While backtracking, other partial solutions are evaluated. If lower bound on costs of partial solution is higher than f^*, there is no need to evaluate any other node of this branch (only the green nodes are evaluated)
- If during search better solution is found, then f^* is updated

Overview of integer optimization methods

Integer optimization methods

Most important approaches are:

- *Mixed Integer (Non)-Linear Programming (MI(N)LP)*
- Dynamic programming
- Constraint Logic Programming (CLP)
- Heuristic methods

Mixed integer linear programming

- **Mixed Integer Linear Programming (MILP)**:

$$\min_x \quad c^T x \qquad \text{s.t.} \quad Ax \leqslant b \qquad \text{where} \quad x = \begin{bmatrix} x_\mathrm{r} \\ x_\mathrm{i} \end{bmatrix},\ x_\mathrm{r} \in \mathbb{R}^{n_\mathrm{r}},\ x_\mathrm{i} \in \mathbb{Z}^{n_\mathrm{i}}$$

- LP relaxation:

$$\min_x \quad c^T x \qquad \text{s.t.} \quad Ax \leqslant b \qquad \text{where} \quad x_k \in \mathbb{R} \ \text{ for all } k$$

  $\rightarrow$ can be solved using, e.g., simplex method
  $\rightarrow$ solve LP relaxation and round solution (in feasible direction)
  $\rightarrow$ feasible?
  optimal?

- $c^T x^*_{\mathrm{LP}} \leqslant c^T x^*_{\mathrm{MILP}} \rightarrow$ relaxed LP yields lower bound for MILP

Basic branch-and-bound algorithm for MILP

- Initialization:
  $U^* := f(\text{best feasible mixed integer/real solution}) = +\infty$
  $\mathcal{I}$: indices of integer variables
  solve LP relaxation of original problem $\rightarrow x^*$
- Iteration:
  - Branching:
    take most recently created subproblem
    select branching variable x_j ($j \in \mathcal{I}$, $x_j^* \notin \mathbb{Z}$)
    create two new subproblems with constraints

$$x_j \leqslant \lfloor x_j^* \rfloor \qquad x_j \geqslant \lfloor x_j^* \rfloor + 1$$

  - Bounding:
    compute lower bounds L_i using LP relaxation
  - Pruning: prune branch i if
    - $L_i > U^*$
    - no feasible solution for LP relaxation
    - LP solution x^* with $x_j^* \in \mathbb{Z}$ for all $j \in \mathcal{I}$
      $\rightarrow$ set $U^* = \min(U^*, L_i)$
- Stop if no remaining subproblems

Example: MILP using branch-and-bound

$$
\begin{aligned}
\min_x \quad & 20 - 4\,x_1 + 2\,x_2 - 7\,x_3 + x_4 \\
\text{s.t.} \quad & x_1 + 5\,x_3 \leqslant 10 \\
& x_1 + x_2 - x_3 \leqslant 1 \\
& 6\,x_1 - 5\,x_2 \leqslant 0 \\
& -x_1 + 2\,x_3 - 2\,x_4 \leqslant 3 \\
& x_1, x_2, x_3, x_4 \geqslant 0 \\
& x_1, x_2, x_3 \in \mathbb{Z}
\end{aligned}
$$

Integer variables: x_1, x_2, x_3, Continuous variable: x_4

Initialization: $U^* = \infty$, $\mathcal{I} = \{1, 2, 3\}$

LP relaxation:

$\rightarrow \quad x^* = [\,1.25 \;\; 1.5 \;\; 1.75 \;\; 0\,]$, $L_0 = 5.75$

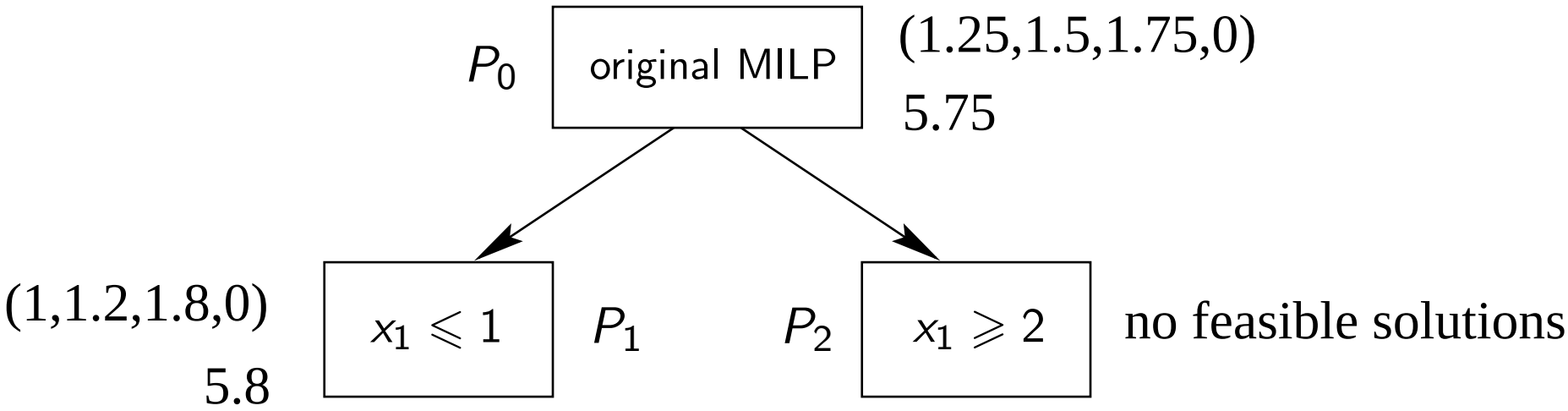
Example: MILP using branch-and-bound

Iteration 1:

Branching variable: x_1 with $x_1^* = 1.25$

Two new subproblems: P_1: P_0 + additional constraint $x_1 \leqslant 1$
P_2: P_0 + additional constraint $x_1 \geqslant 2$

LP relaxations: $P_1 \rightarrow x^* = [\,1 \;\; 1.2 \;\; 1.8 \;\; 0\,], \; L_1 = 5.8$
$P_2 \rightarrow$ no feasible solutions $\rightarrow$ prune



Example: MILP using branch-and-bound (cont.)

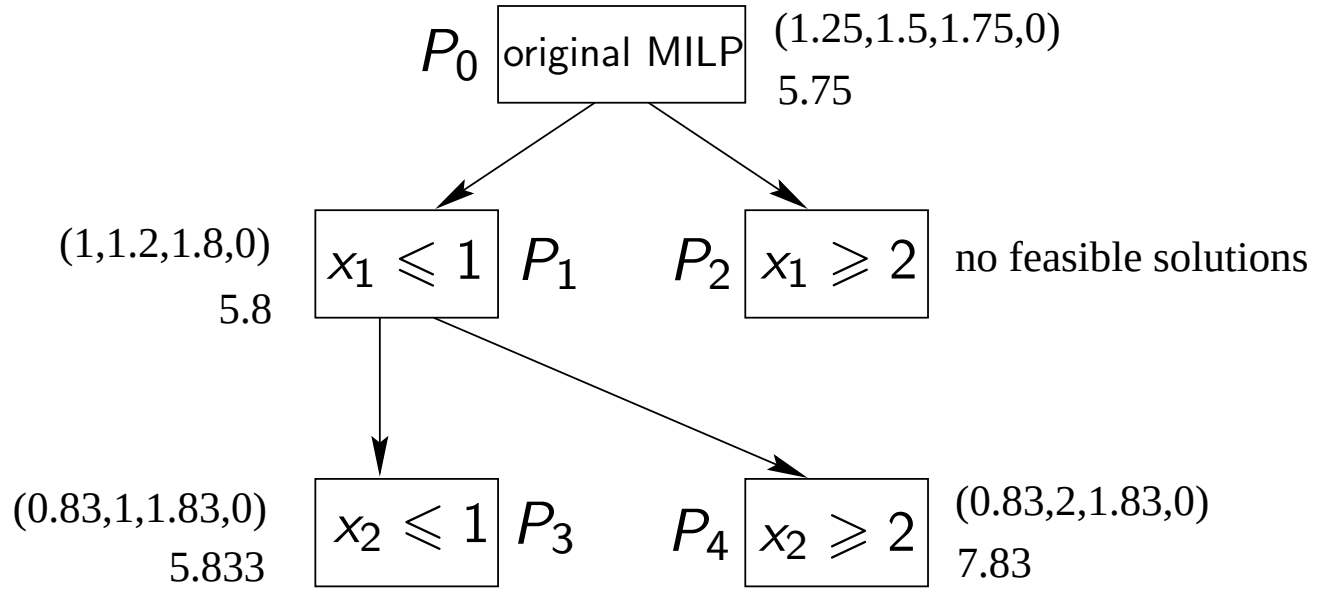
Iteration 2:

Select subproblem P_1 ,

Branching variable: x_2 with $x_2^* = 1.2$

Two new subproblems: P_3: P_1 + additional constraint $x_2 \leqslant 1$
P_4: P_1 + additional constraint $x_2 \geqslant 2$

LP relaxations $P_3 \rightarrow x^* = [\,0.833 \;\; 1 \;\; 1.833 \;\; 0\,], \;\; L_3 = 5.833$
$P_4 \rightarrow x^* = [\,0.833 \;\; 2 \;\; 1.833 \;\; 0\,], \;\; L_4 = 7.833$



Example: MILP using branch-and-bound (cont.)

Iteration 3:

Select subproblem P_3

Branching variable: x_1 with $x_1^* = 0.833$

Two new subproblems: P_5: P_3 + additional constraint $x_1 \leqslant 0$

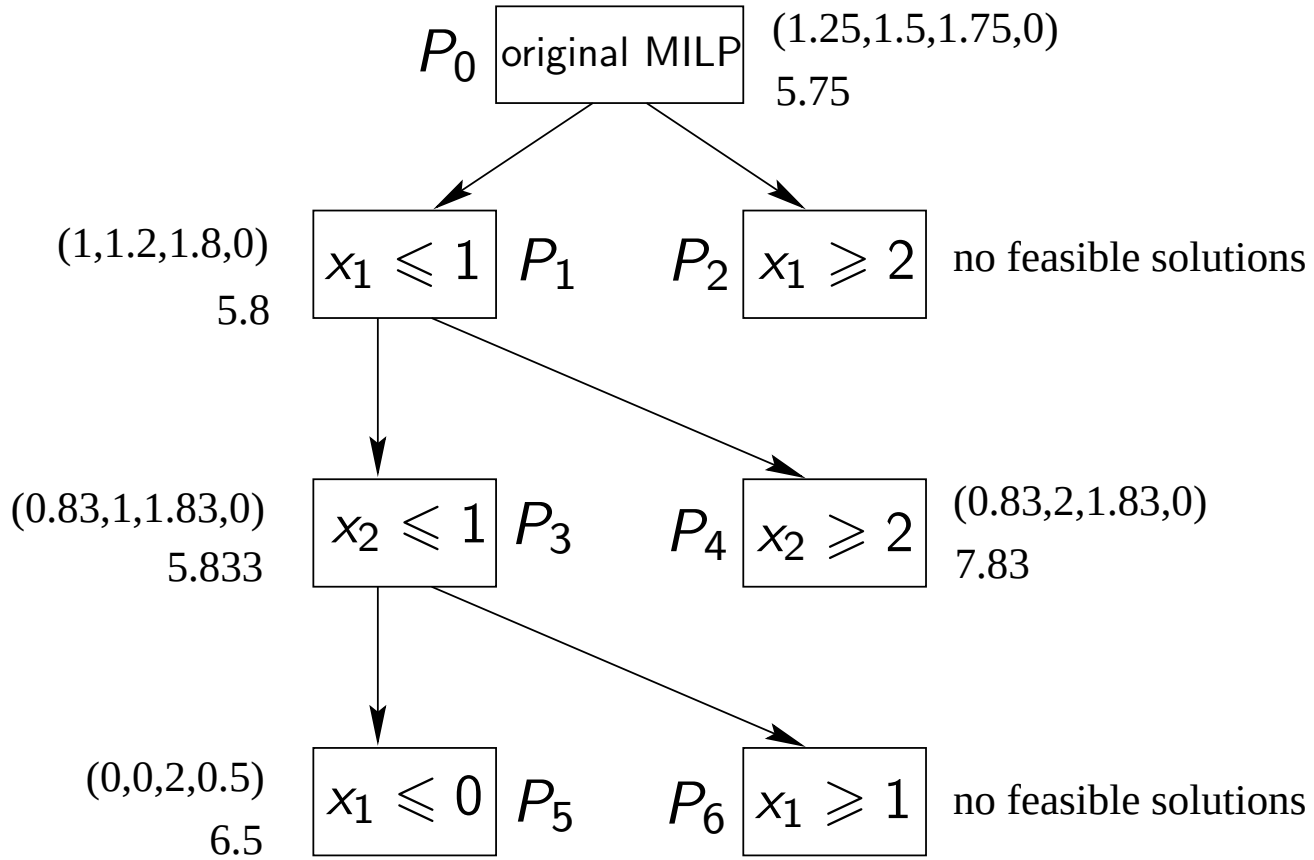
P_6: P_3 + additional constraint $x_1 \geqslant 1$

LP relaxations

$P_5 \rightarrow$ $x^* = [\,0 \;\; 0 \;\; 2 \;\; 0.5\,]$, $L_5 = 6.5$
feasible integer/real solution of MILP
set $U^* = 6.5$

$P_6 \rightarrow$ no feasible solutions

Example: MILP using branch-and-bound (cont.)



$L_4 = 7.833 > U^* \Rightarrow$ problem P_4 can be fathomed

$\rightarrow$ no remaining subproblems

$\Rightarrow$ optimal solution: $x = [\,0 \;\; 0 \;\; 2 \;\; 0.5\,]$

Heuristic search techniques

Heuristic search techniques

- *Random search*
- *Genetic algorithms*
- *Simulated annealing*
- Tabu search
- Randomized algorithms
- ...

$\rightarrow$ no guaranteed convergence to (global) optimum
but: "good" solutions on the average

Summary

- Defined integer optimization + complexity
- Search strategies
- Integer optimization methods
  - mixed integer (non)-linear programming
    — branch-and-bound
  - heuristic methods: random, genetic, simulated annealing