Integer Linear Programming: Introduction to Branch & Bound

Dr. Gwen Maudet

UNIVERSITÉ DU LUXEMBOURG

october 2025

Example

maximize $z = 2x_1 + x_2$

subject to $6x_1 + 5x_2 \leq 23$

$8x_1 + 3x_2 \leq 19$

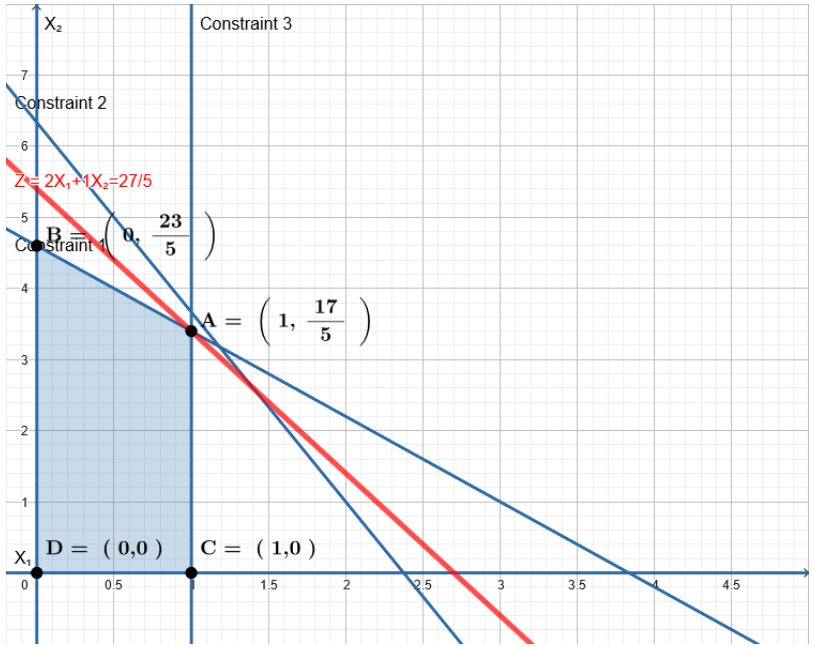
$x_1, x_2 \geq 0$

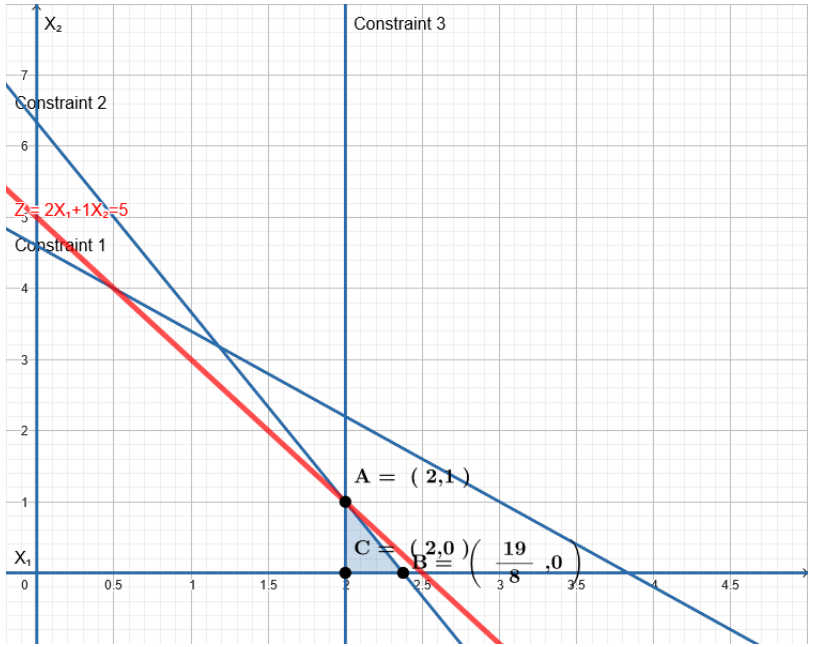
$x_1, x_2 \in \mathbb{N}$

○ Set of integer solutions

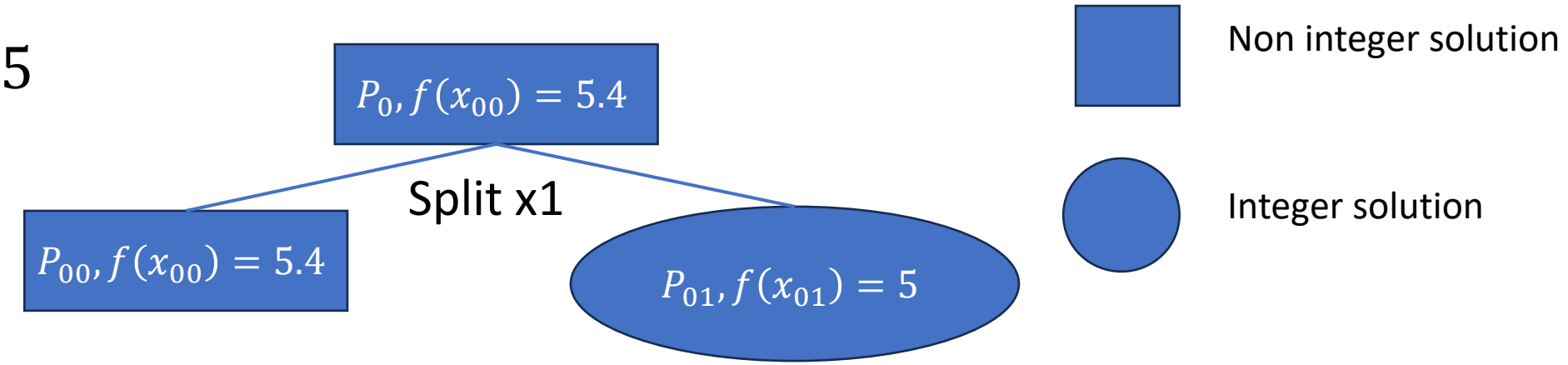
Divide step 1

- The best solution of P_0 is X1=1.182 , X2=3.182.
- Divide P_0 according to variable x_1. Solve graphically the subproblems P_{00} and P_{01}.



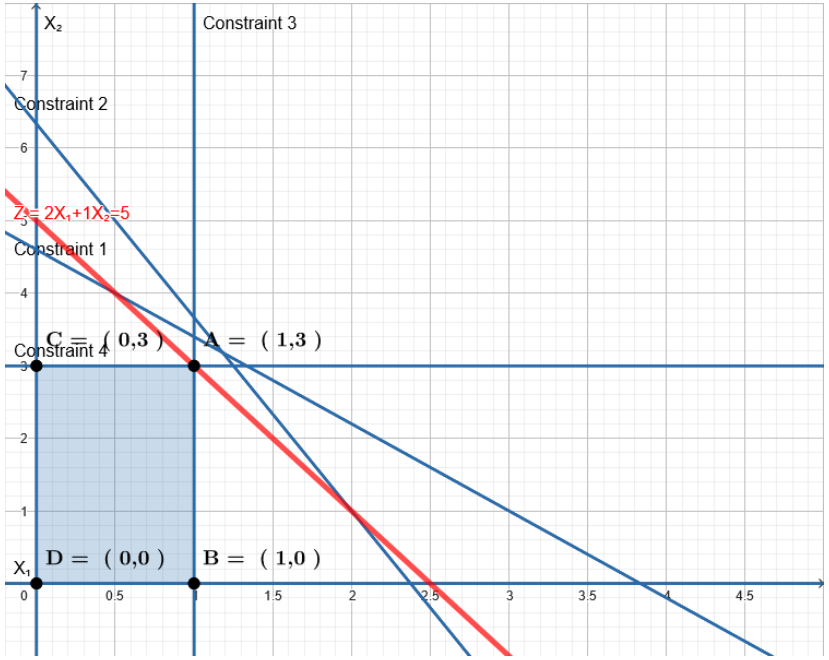


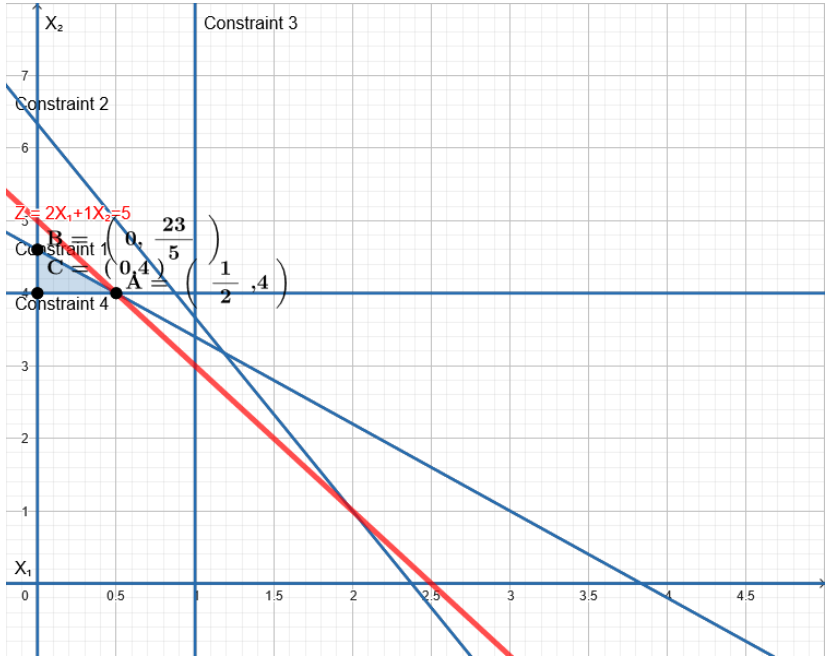
- $f(x^*) = 5$


$P_0, f(x_{00}) = 5.4$
Split x1
$P_{00}, f(x_{00}) = 5.4$
$P_{01}, f(x_{01}) = 5$
Non integer solution
Integer solution

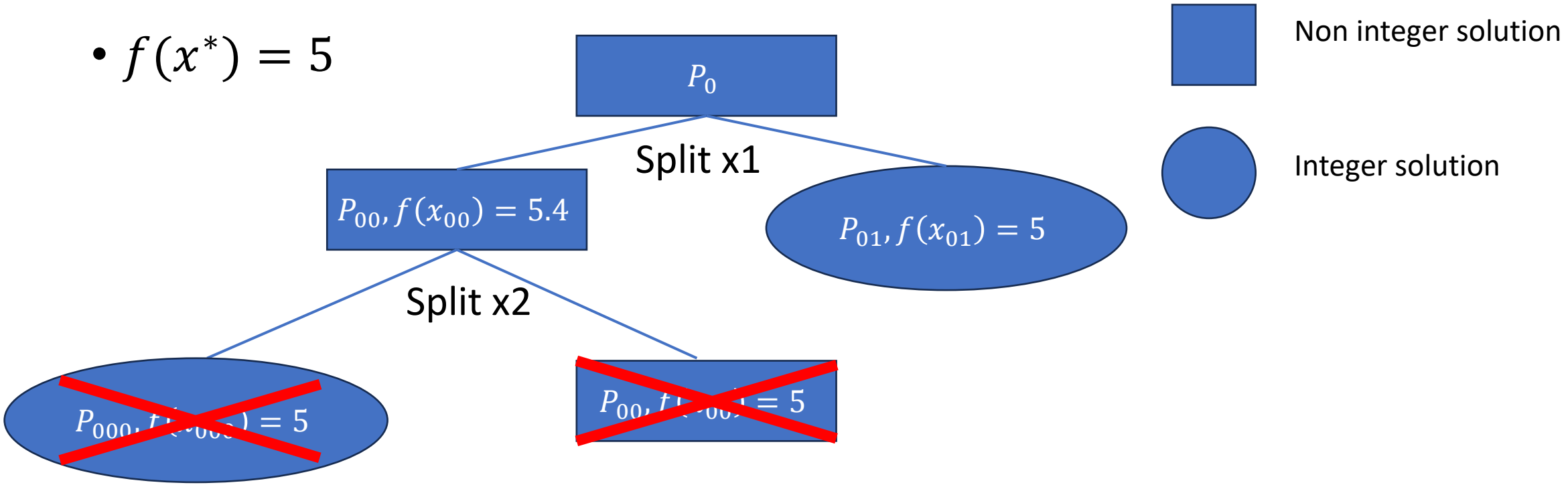
Divide step 2

- The best solution of P_{00} is x1=1, x2=3.4
- Divide P_{00} according to variable x_2. Solve graphically the subproblems P_{000} and P_{001}.

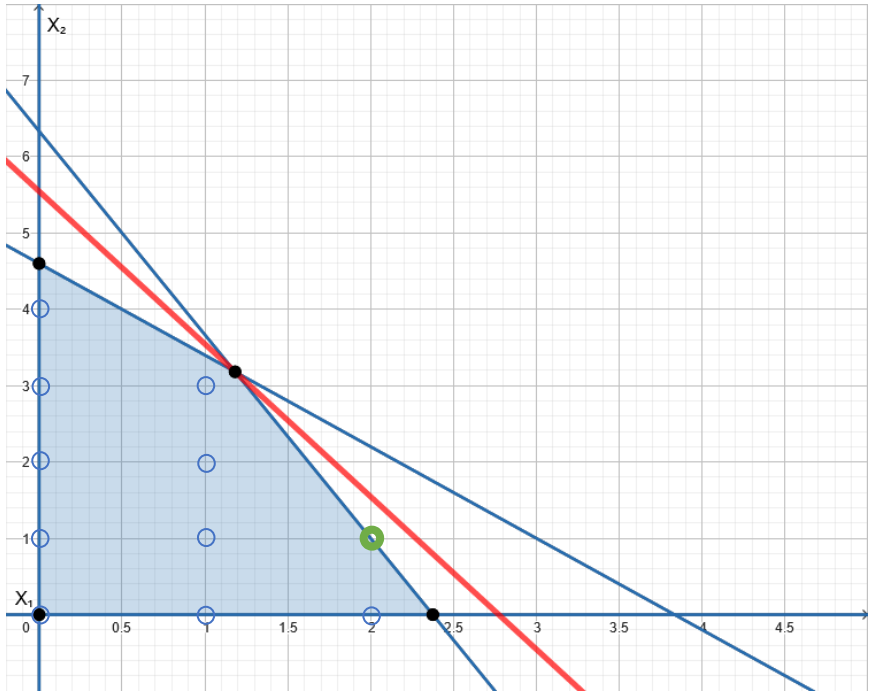




- $f(x^*) = 5$


P_0
Split x1
$P_{00}, f(x_{00}) = 5.4$
$P_{01}, f(x_{01}) = 5$
Split x2
$P_{000}, f(x_{000}) = 5$
$P_{00}, f(x_{00}) = 5$
Non integer solution
Integer solution

Optimal solution graphically



Set of integer solutions

One optimal solution

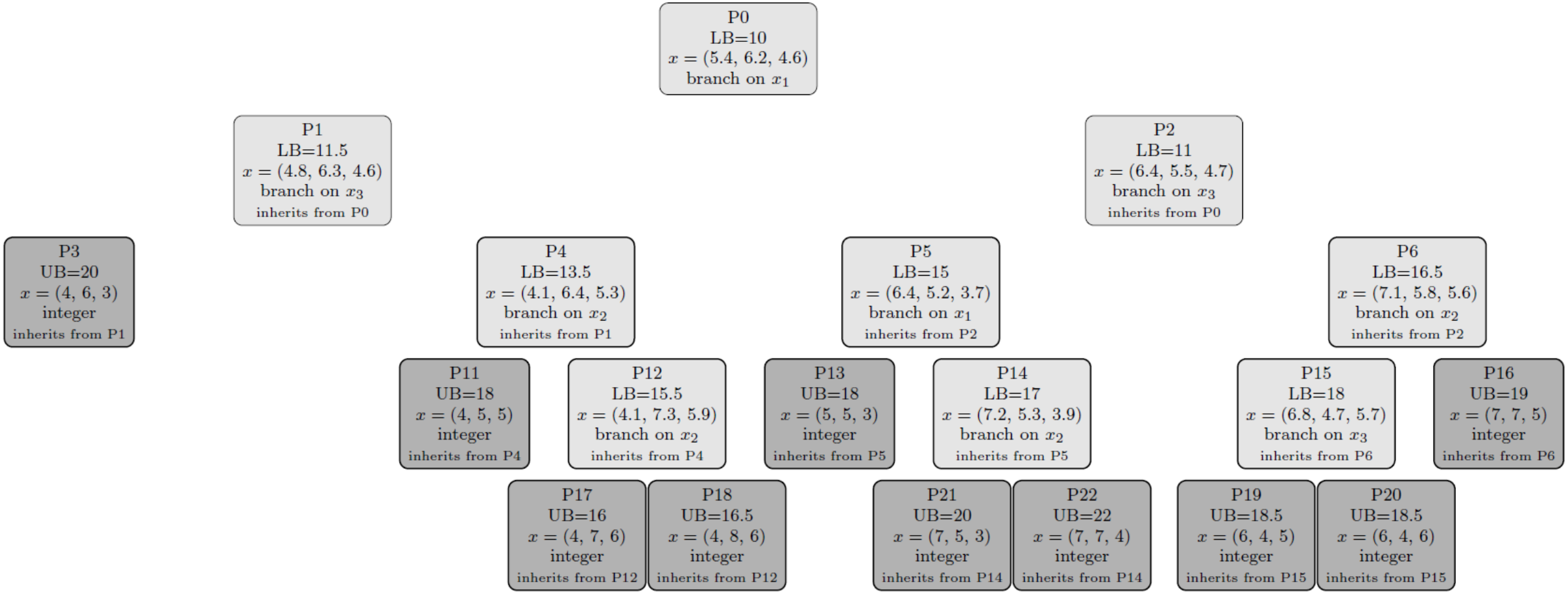
Going Deeper in Branch and Bound

Different branching, searching strategies

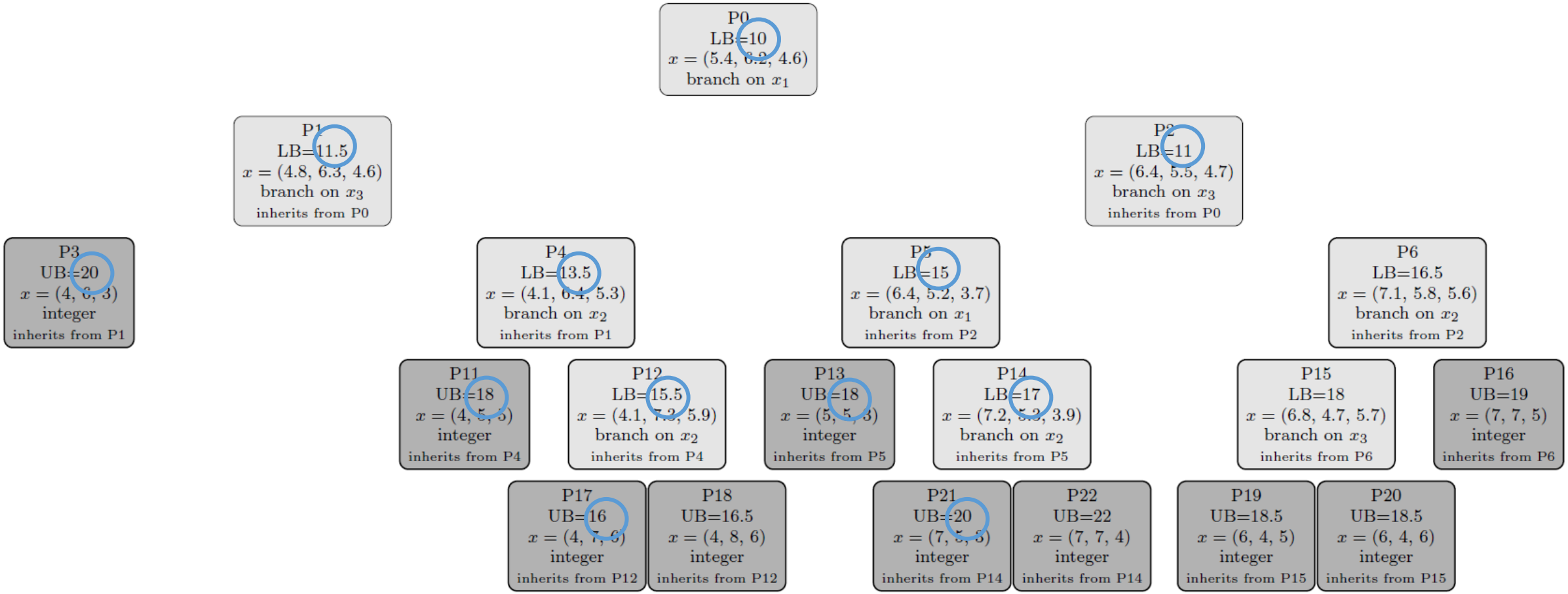
From Constraint Integer Programming, Achterberg, PhD thesis, 2007[1]

[1]: Achterberg, Constraint Integer Programming, 2007, PhD thesis

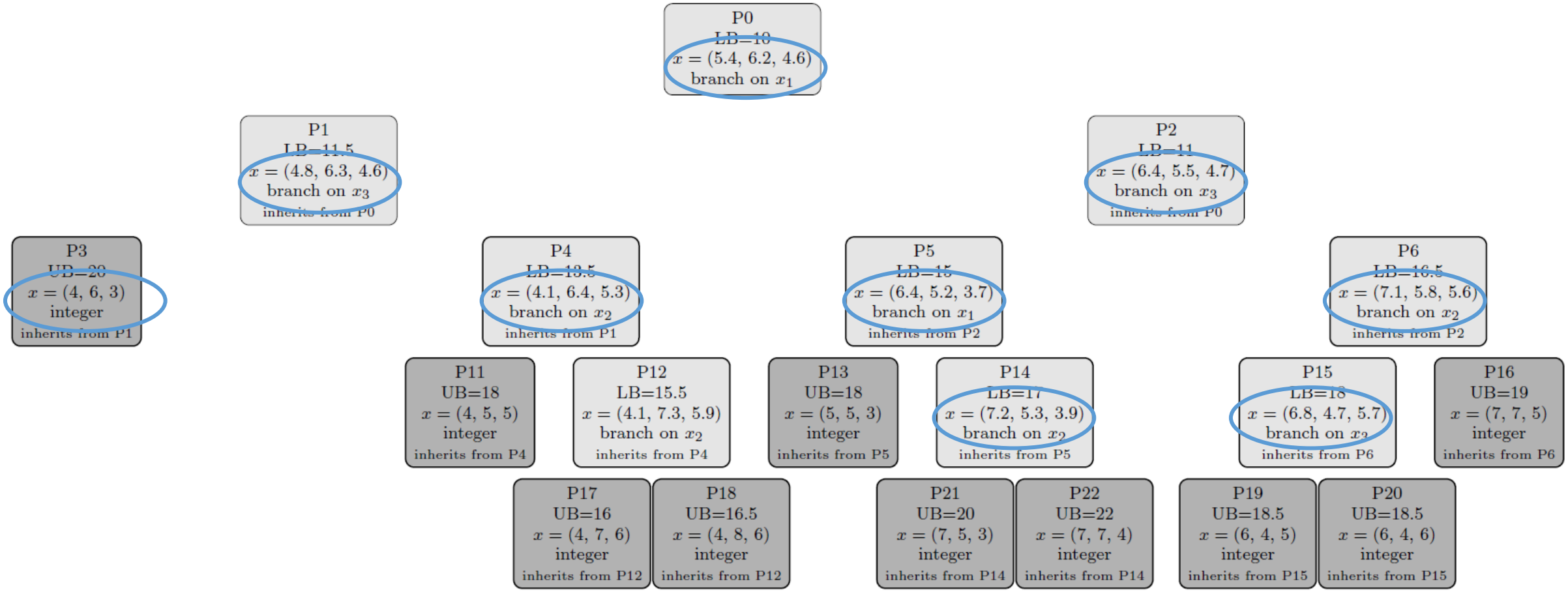
Recap about the B&B tree



Recap about the B&B tree



Recap about the B&B tree



Recap about the B&B tree

- Relaxed solutions (LB, UB) always get bigger from a parent to a child in minimization problem : → less solutions so poorer solutions
- When you branch on a variable, the children will respect for sure $x_i^k \leq \lfloor x_i^k \rfloor$ and $\lfloor x_i^k \rfloor \leq x_i^k$
  - If solution is integer, then the subproblem is solved

Branching

Branching Strategies

Answer the question: which variable to select to split one problem into 2 subproblems?

- The objective is to have the highest lower bound on both side of a child= the best approximation of optimal solution of the subproblem

→good approximation = good pruning possibilities.

In [1], show that a random branching can lead to ~+200% of solving time compared to a tuned one.

[1]: Achterberg, Constraint Integer Programming, 2007, PhD thesis

Branching Strategies: most infeasible

Consider a subproblem P^k, non integer solution x^k, $\mathrm{I^k} = \{i \in [1, n] \mid x_i^k \notin \mathbb{N}\}$

For one solution candidate x_i^k, it fractional parts are $f_{i-}^k = x_i^k - \lfloor x_i^k \rfloor$ and $f_{i+}^k = \lfloor x_i^k \rfloor - x_i^k$

- Known one: variable the closest to 0.5: $\max_{i \in I^k}(\min(f_{i-}^k, f_{i+}^k))$

→ in practice not used because not significantly better than random.

Branching Strategies: strong branching

Consider a subproblem P^k, non integer solution x^k, $\mathrm{I^k} = \{i \in [1, n] \mid x_i^k \notin \mathbb{N}\}$.

The trick: doing the branching for each variable, and compare the increasing in the lower bound → tighter lower bound for a same optimal solution →better chance of pruning

For $i \in \mathrm{I}^k$, P^k leads to P_{i-}^k and P_{i+}^k, with improvement of the LB being Δ_{i-}^k for P_{i-}^k and Δ_{i+}^k for P_{i+}^k. We want the variable that maximizes the increase of lower bound

One way is:

$$\max_{i \in I^k} (\Delta_{i-}^k \times \Delta_{i+}^k)$$

Drawback: doing the branching for each candidate then computing the LB is really expensive

Branching Strategies: Pseudo Cost

Consider a subproblem P^k, non integer solution x^k, $\mathrm{I}^{\mathrm{k}} = \{i \in [1, n] \mid x_i^k \notin \mathbb{N}\}$

The trick: instead of computing the improvement of the lower bound for each case, just use experience and approximate it!

Consider for one variable x_i, and μ_i being the number of time it has been used for branching already in other parts of the tree, each time P^l to P_{i-}^l being Δ_{i-}^l and Δ_{i+}^l for P_{i+}^l; from this experience, $\frac{\Delta_i^l}{f_{i.}^l}$ the LB gain per fractional part. The mean increasing of LB per fractional is $\varphi_{i-} = \frac{\sum \frac{\Delta_{i-}^l}{f_{i-}^l}}{\mu_i}$ and $\varphi_{i+} = \frac{\sum \frac{\Delta_{i+}^l}{f_{i+}^l}}{\mu_i}$.

- Then the pseudo cost for one variable candidate x_i is

$$\varphi_{i-} \times f_{i-}^k \times \varphi_{i+} \times f_{i+}^k$$

We then choose the variable that maximizes the pseudo cost.

→no need for extra computations: more lightweight

Branching Strategies: hybrid

- Pseudo cost is a good approximation, but can't be used at the beginning
- Strong branching is expensive but the the best choices to make + choices at the beginning of the solving are the mot important

- → In modern solvers (like SCIP open source solver), hybrid approach:
  - Use strong branching when the dept is lower than 10
  - Else, use pseudo-cost

It exist many other fin tuning for quantifying the quality of a variable for branching….

Searching

Search strategy

Answer the question: according to a set of unsolved subproblems, which one to solve the first?

The objective is to explore the node (or subproblem) that leads to the best solution.

2 components:

- Node comparing method: how to quantify the quality of a node
- Node selection method: which node will be explored in the end

Search Strategy: LB node comparing

- The objective is to rank open subproblems of the tree search. Usually, **a score is given to all node**, to then define the ranking.
- Lower Bound(LB): the lower bound on the node can be an estimation of how will be the integer solution of the subproblem: $e_{LB}(P^k) = f(x^k)$

Search Strategy: BP node comparing

- The objective is to rank open subproblems of the tree search. Usually, **a score is given to all node**, to then define the ranking.
- Best projection (BP): consider $\phi(x^k) = \sum_{i=1}^{n} \min(f_{i-}^k, f_{i+}^k)$ the distance to the closest integer solution.

$$e_{BP} = f(x^k) + \frac{(f(x^*) - f(x^0))}{\phi(x^0)} \phi(x^k)$$

increase of the score per fractional unit

Search Strategy: BE node comparing

- The objective is to rank open subproblems of the tree search. Usually, **a score is given to all node**, to then define the ranking.
- Best Estimate (BE): use the estimator of increasing of the LB according to the experience φ_{i-} φ_{i+}.

$$e_{BE} = f(x^k) + \sum_i \min(\varphi_{i-} f_{i-}^k, \varphi_{i+} f_{i+}^k)$$

Search Strategy: node selection

There is a total ranking of all the open nodes. Which one to select the first:

- **Best First Search (BFS):** select the best one

→advantage: if the ranking is accurate, the best choice is the best node

- **Dept First Search (DFS):** according to the node explored, select the best child; if children are completed, get the closest ancestor.

→advantage: DFS is the fastest way to have one feasible solution

+in practice, the search tree can be big (exponential), and DFS allow to minimize the number of open nodes

Search Strategy: node selection

- Hybrid version
  - As long as the children are unprocessed, do DFS. When you reach integer solutions, you select the next node by BFS, and then from that node on, you do again DFS.
  - →plunging to integer solution but do not get stuck in part of the tree

  - Condition can be given to stop the DFS even before finding a good solution: if the score is too far from the best score, then switch again to BFS.

Exercices

University of Luxembourg

Merci | Thank you | Danke

UNIVERSITÉ DU LUXEMBOURG