# LP-based Branch and Bound

Jesper Larsen & Jens Clausen

jla,jc@imm.dtu.dk

Informatics and Mathematical Modelling
Technical University of Denmark



DTU

# Divide and conquer

In an effort to solve

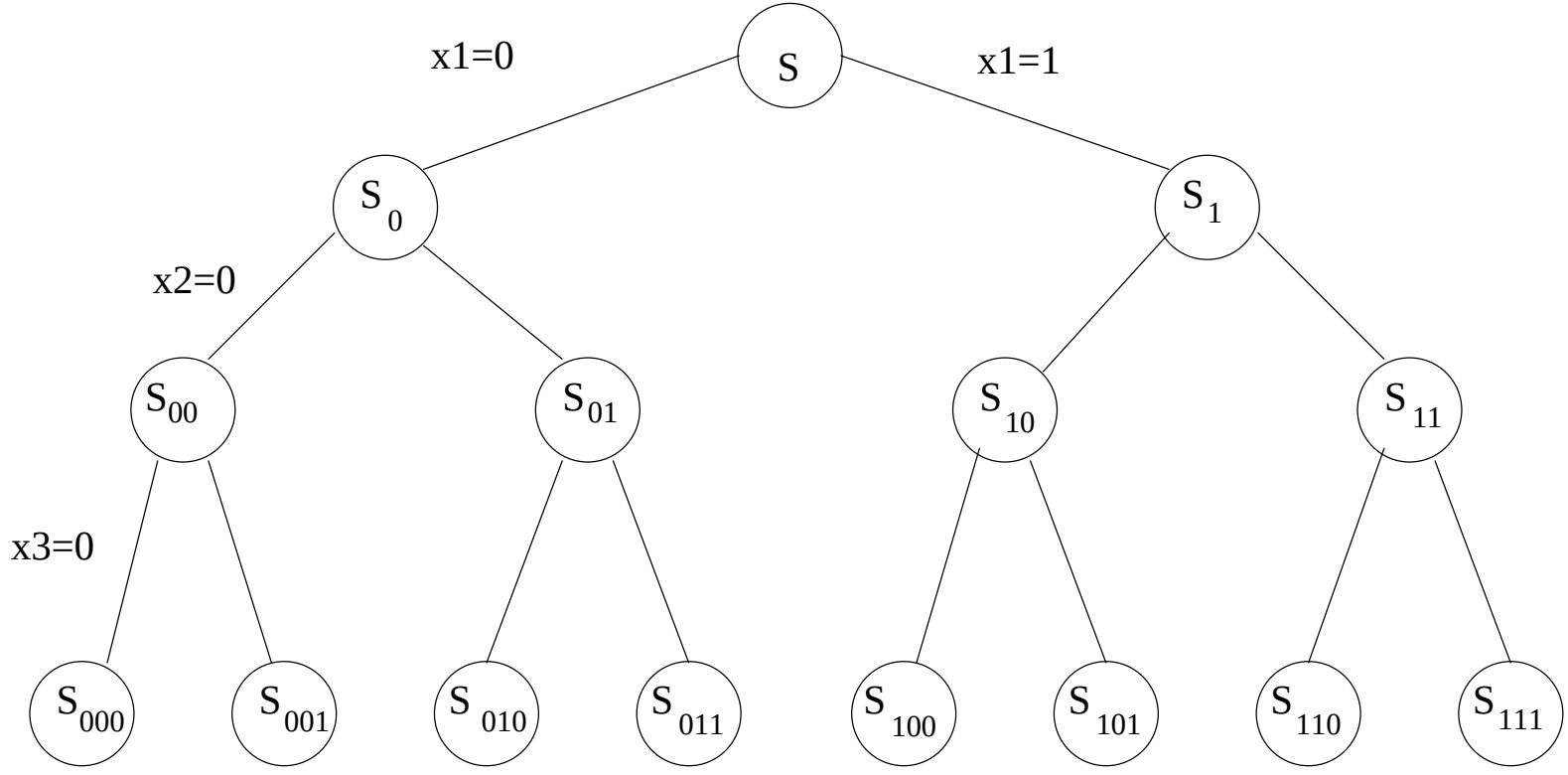
$$z = \max\{cx : x \in S\}$$

an idea would be to divide the problem into smaller problems that are easier to solve.

- $S = S_1 \cup S_2 \cup \ldots \cup S_K$ be a **decomposition** of $S$.
- $z^k = \max\{cx : x \in S_k\}$ for $k = 1, 2, \ldots, K$.
- $z = \max_k z^k$.

# Enumeration tree

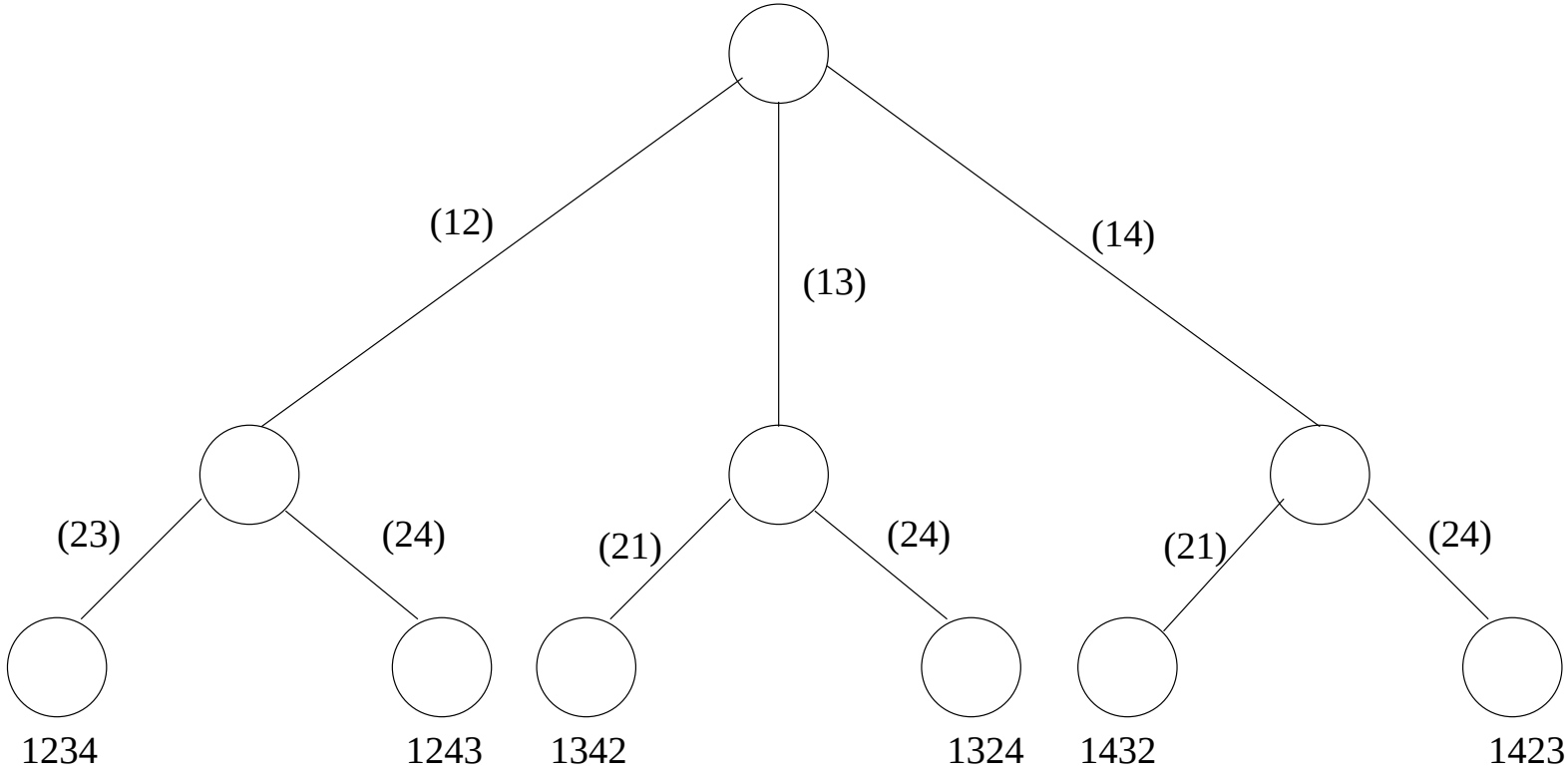
A way to represent the decomposition approach is via an enumeration tree.



# Enumeration

- In an **explicit** enumeration all possible solutions are investigated in order to find the optimal solution.
- In an **implicit** enumeration all possible solutions are in the **worst-case** investigated in order to find the optimal solution.

# Enumeration tree for the TSP



# Implicit Enumeration

In order to overcome large problems we need to do more than just divide and solve leaf nodes.

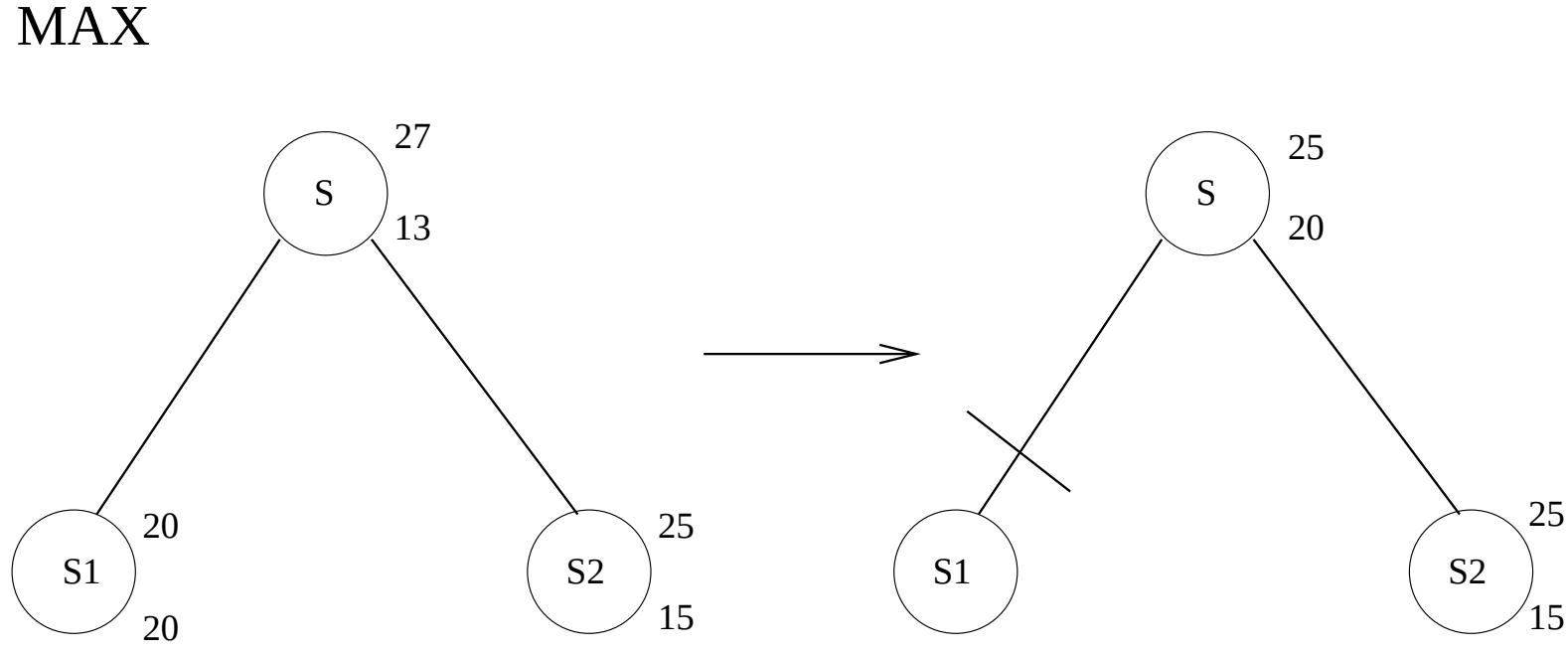
- How can we put together bound information?
- How can we use some bounds on the values of $\{z^k\}$ intelligently?

# Bounds

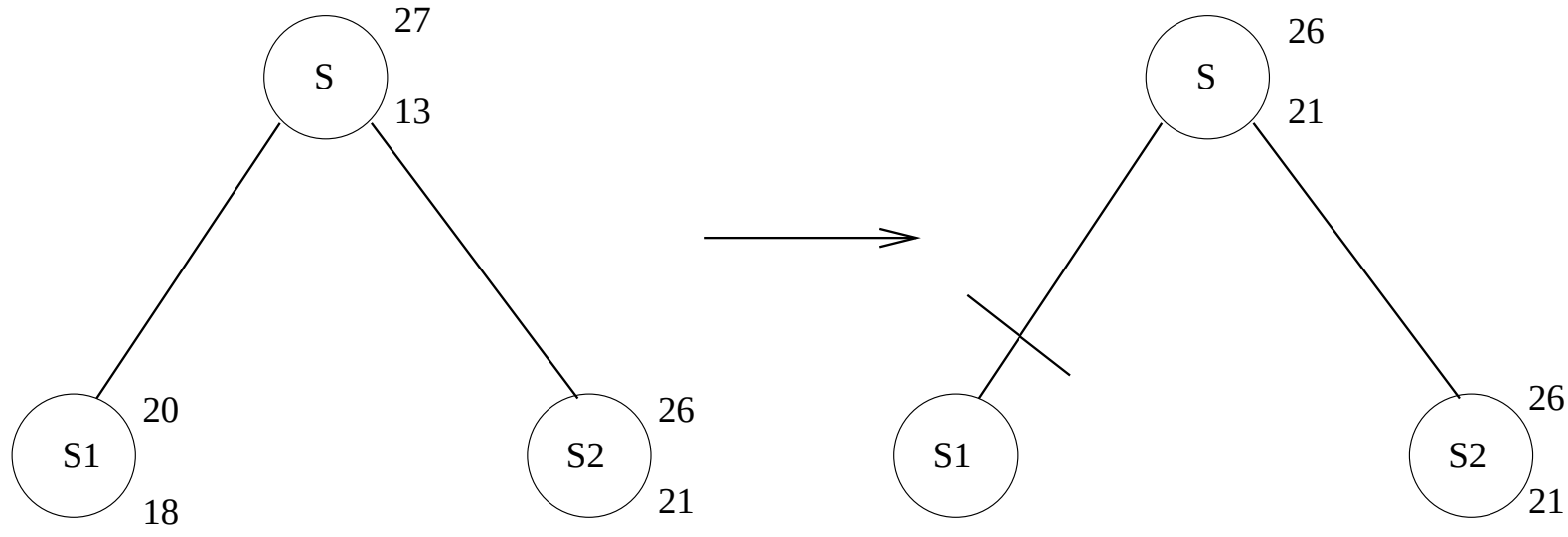
Let $S = S_1 \cup S_2 \cup \ldots \cup S_K$ be a **decomposition** of $S$, and let $\bar{z}^k$ be an upper bound on $z^k$ and $\underline{z}^k$ be a lower bound on $z^k$. Then

- $\bar{z} = \max_k \bar{z}^k$ is an upper bound on $z$.
- $\underline{z} = \max_k \underline{z}^k$ is a lower bound on $z$.

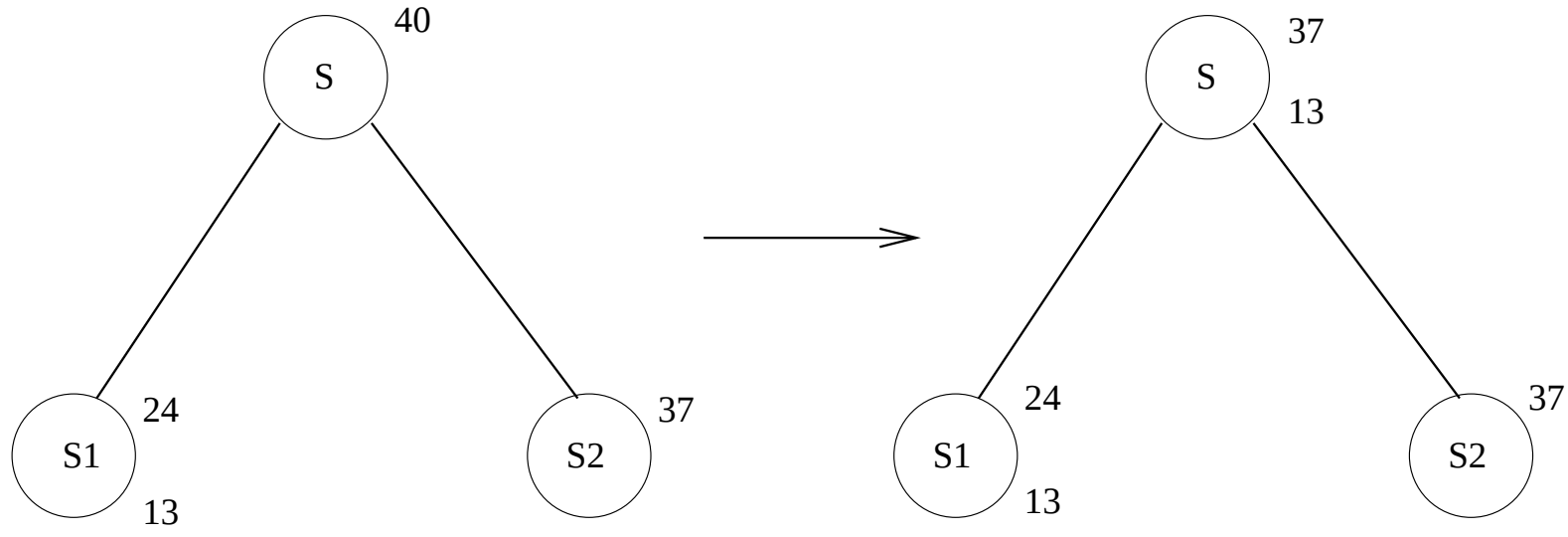
# Pruned by optimality



# Pruned by bound



# No pruning possible



# Pruning possibilities

Based on the three cases we have:

- Pruning by optimality: $z^t = \max\{cx : x \in S_t\}$ has been solved.
- Pruning by bound: $\bar{z}^t \leq \underline{z}$
- Pruning by infeasibility $S_t = \emptyset$
- Branching

# Practical aspects

- **Storing the tree:** List of *active* nodes, best known dual bound, variable lower and upper bounds, optimal/near-optimal basis.
- **How to bound:** LP-relaxation and LP-solver.
- **How to branch:**
  - Branch on most *fractional* variable.
  - Estimate the cost of forcing $x_j$ to become integer.
  - **[binary:]** Branch on closest to 0-1.
- **How to choose a node:** Next time!!

# Preprocessing

**Idea:** Detect and eliminate redundant constraints and variables, and tighten bounds where possible.

- Tightning bounds: use known bounds on some variables to tighten bounds on others.
- Redundant constraints
- Variable fixing (by duality)

# Preprocessing - for BIPs

- Generating logical inequalities
- Combining pairs of logical inequalities
- Simplifying