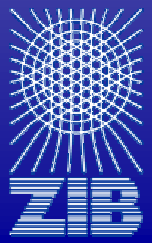
Constraint Integer Programming

SCIP

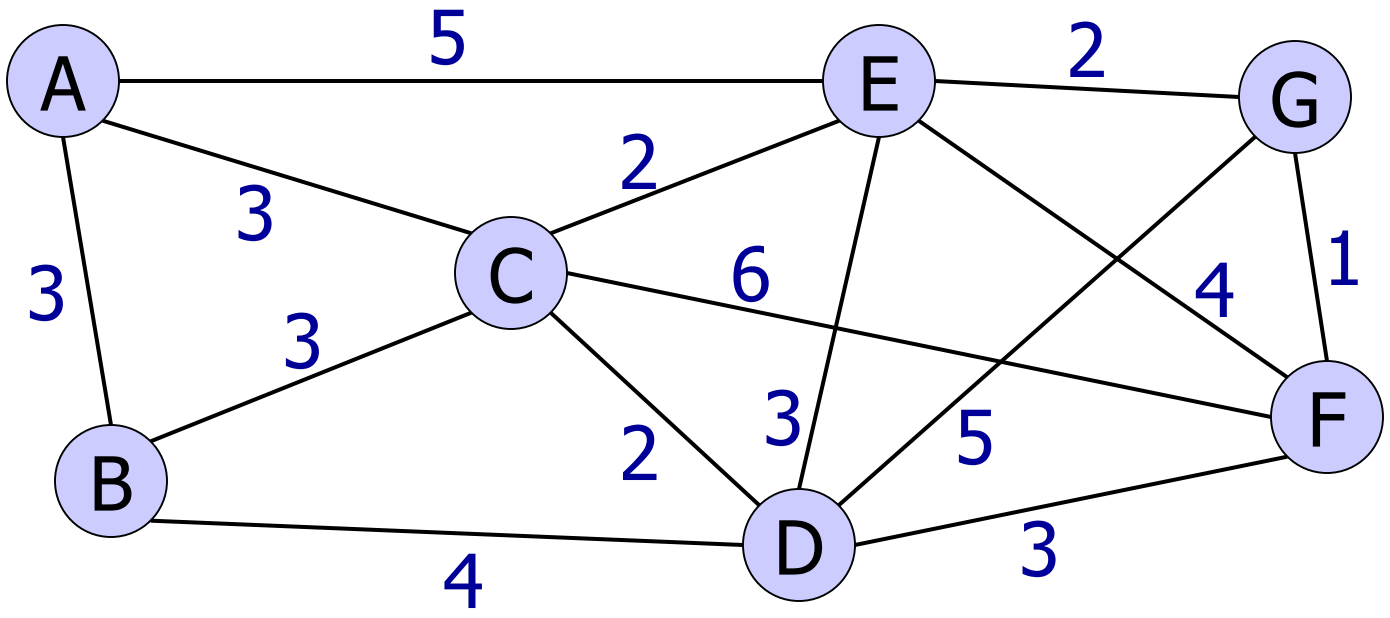
Solving Constraint Integer Programs



Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB)

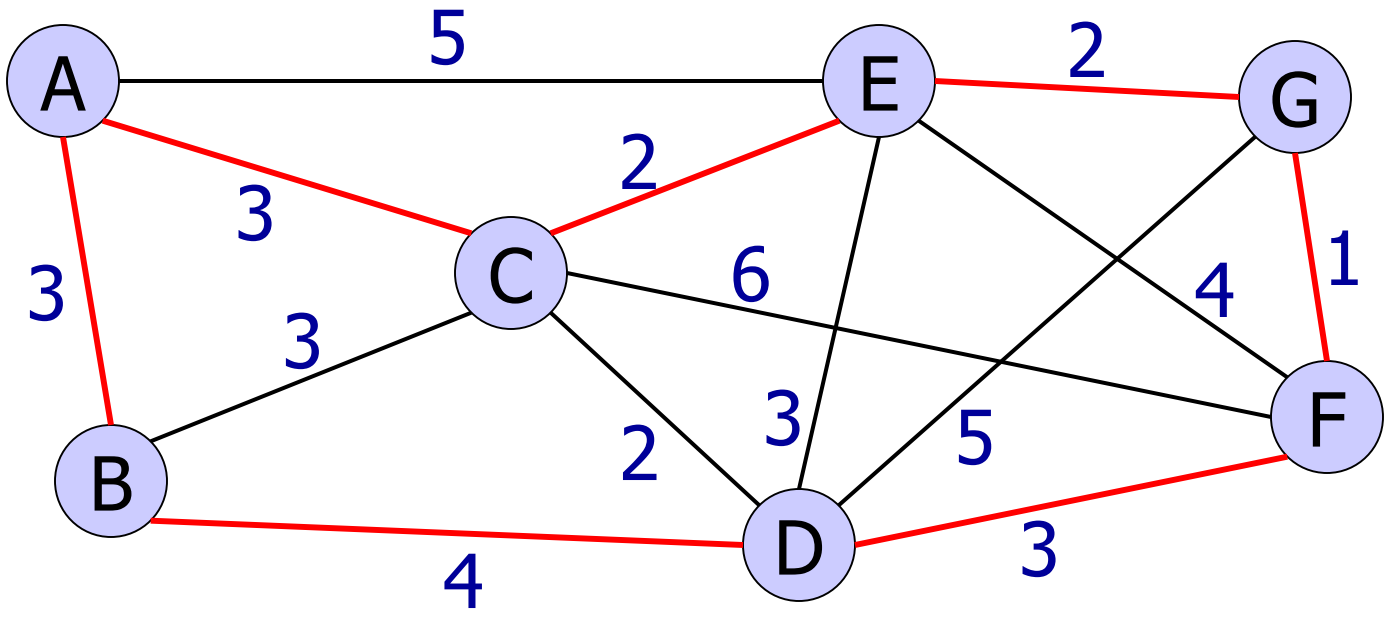
Tobias Achterberg
achterberg@zib.de

Example: Traveling Salesman Problem (TSP)



- given a graph G=(V,E) with distances d_e

Example: Traveling Salesman Problem (TSP)



- given a graph G=(V,E) with distances d_e
- task: find shortest tour

Mixed Integer Program

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in R^p \times Z^q \end{aligned}$$

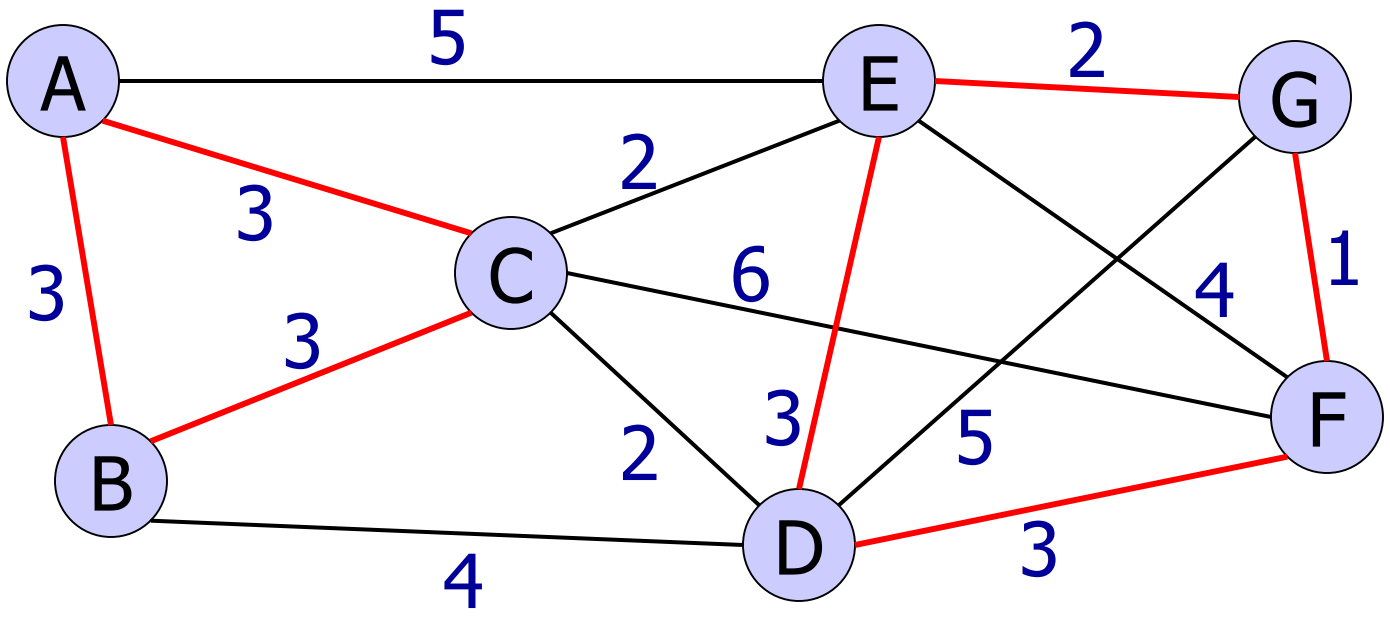
- linear objective function c
- highly structured feasible set F
  - described by linear constraints
- real or integer valued variables x

TSP as Integer Program

$$
\begin{aligned}
\min \quad & \sum_{e \in E} d_e x_e & & & \\
\text{s.t.} \quad & \sum_{e \in \delta(i)} x_e & = & 2 & \text{for all } i \in V \\
& \sum_{e \in \delta(U)} x_e & \geq & 2 & \text{for all } \emptyset \subset U \subset V \\
& x_e & \in & \{0,1\} & \text{for all } e \in E
\end{aligned}
$$

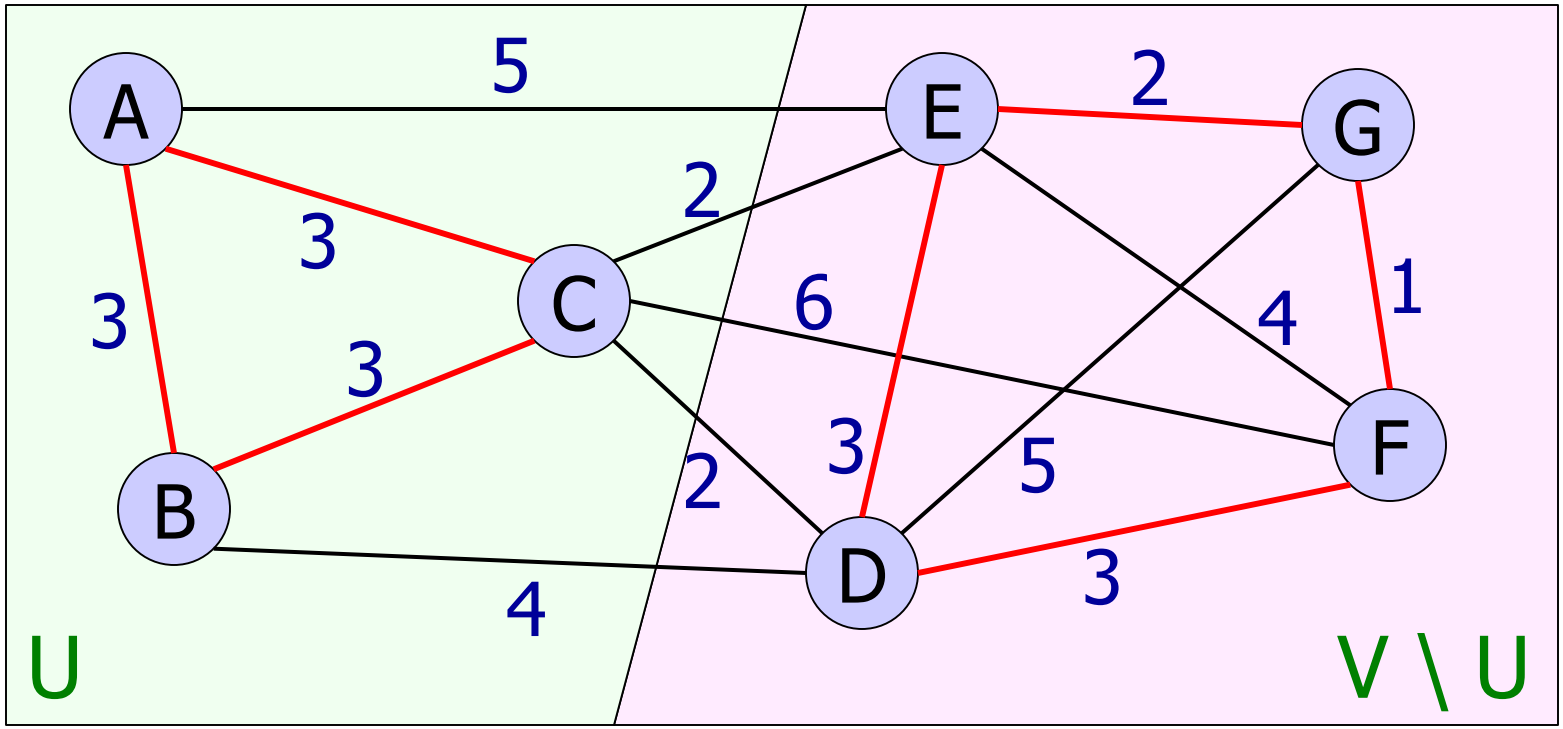
- $|E|$ binary variables: $x_e = 1 \Leftrightarrow$ edge e is in tour
- $|V|$ degree constraints
- $O(2^{|E|})$ subtour elimination constraints

Subtour elimination constraints



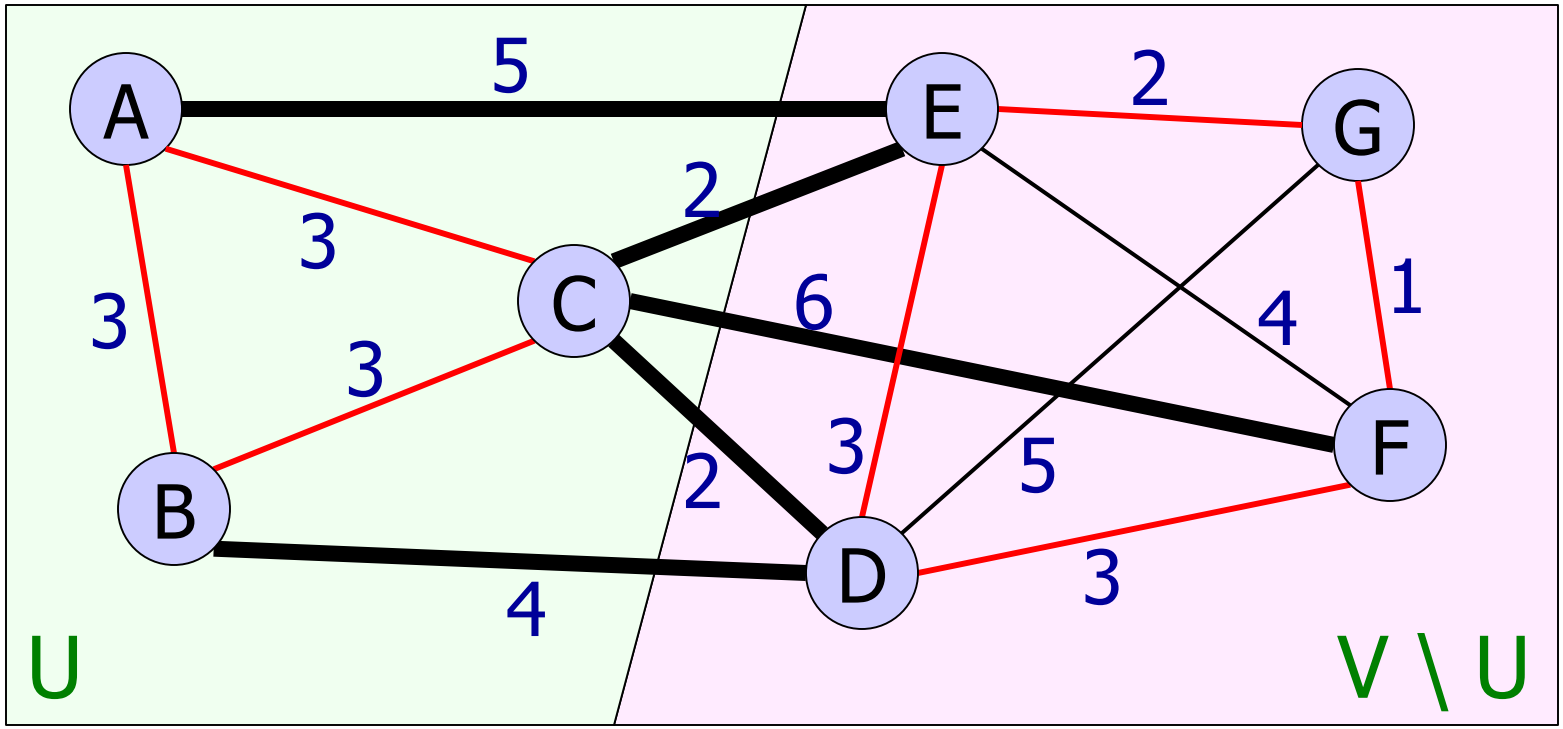
- solution is valid w.r.t. the degree constraints

Subtour elimination constraints



- solution is valid w.r.t. the degree constraints
- subtour constraints: $\sum_{e \in \delta(U)} x_e \geq 2 \text{ for all } \emptyset \subset U \subset V$

Subtour elimination constraints



- solution is valid w.r.t. the degree constraints
- subtour constraints: $\sum_{e \in \delta(U)} x_e \geq 2 \text{ for all } \emptyset \subset U \subset V$

Constraint Program

$$\begin{aligned} \min \quad & c(x) \\ \text{s.t.} \quad & x \in F \\ & x \in \mathbb{Z}^n \end{aligned}$$

- arbitrary objective function c
- arbitrary feasible set F
  - described by arbitrary constraints
- integer valued variables x

TSP as Constraint Program

$$\begin{aligned} \min \quad & \text{length}(x) \\ \text{s.t.} \quad & \text{alldiff}(x_1,\ldots,x_n) \\ & x \in \{1,\ldots,n\}^n \end{aligned}$$

- $|V|$ integer vars: x_v is position of node in tour
- objective function
  - length(x): length of tour $x_1 \to \ldots \to x_n \to x_1$
- all-different constraint
  - $\text{alldiff}(x_1,\ldots,x_n) :\Leftrightarrow x_u \neq x_v$ for all $u \neq v$

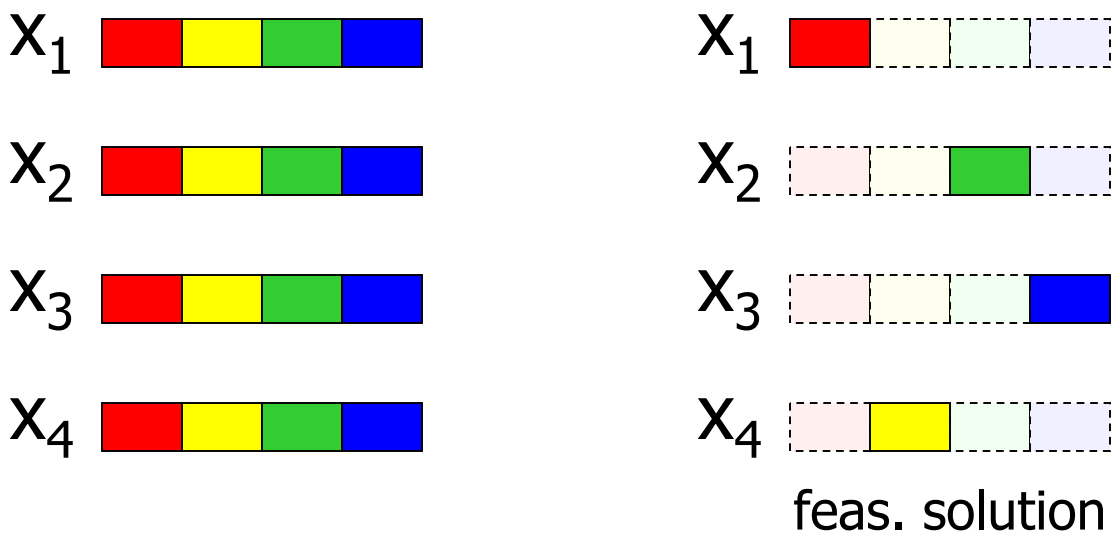
Solving Mixed Integer Programs

- branching
  - split problem into smaller subproblems
  - solve subproblems recursively $\rightarrow$ branching tree
- bounding
  - solve LP relaxations $\rightarrow$ lower bound on objective value
- cutting planes
  - strengthen LP relaxations to get better bounds
- pricing
  - dynamic addition of variables

Solving Constraint Programs

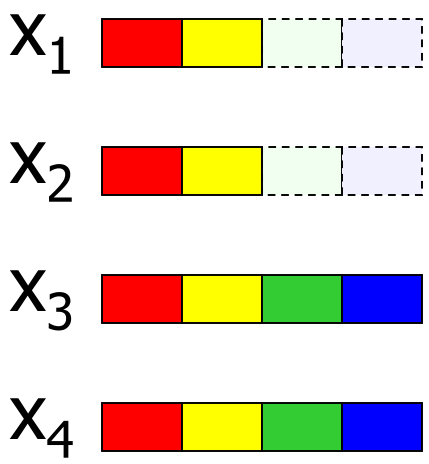
- branching
  - split problem into smaller subproblems
  - solve subproblems recursively $\rightarrow$ branching tree
- bounding
  - only "pseudo solution" available
- domain propagation
  - tighten domains by inference

Domain Propagation (CP)



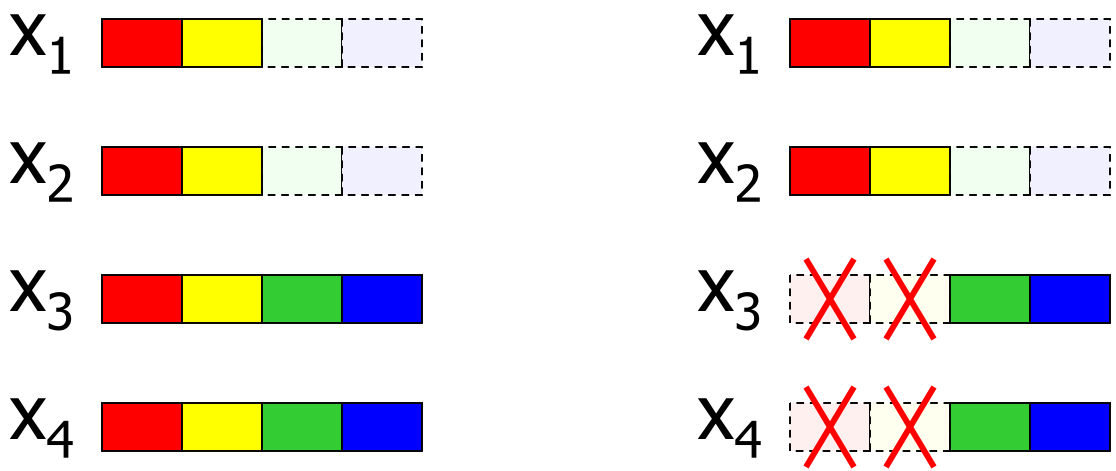
- all-different constraint:
  - each variable must take a different value

Domain Propagation (CP)



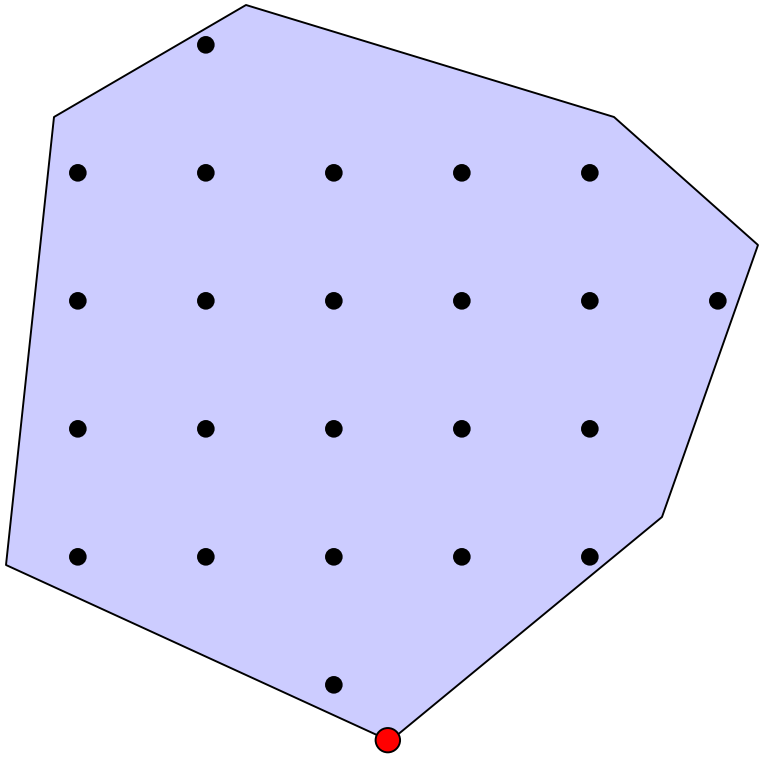
- some values are impossible (due to branching)

Domain Propagation (CP)



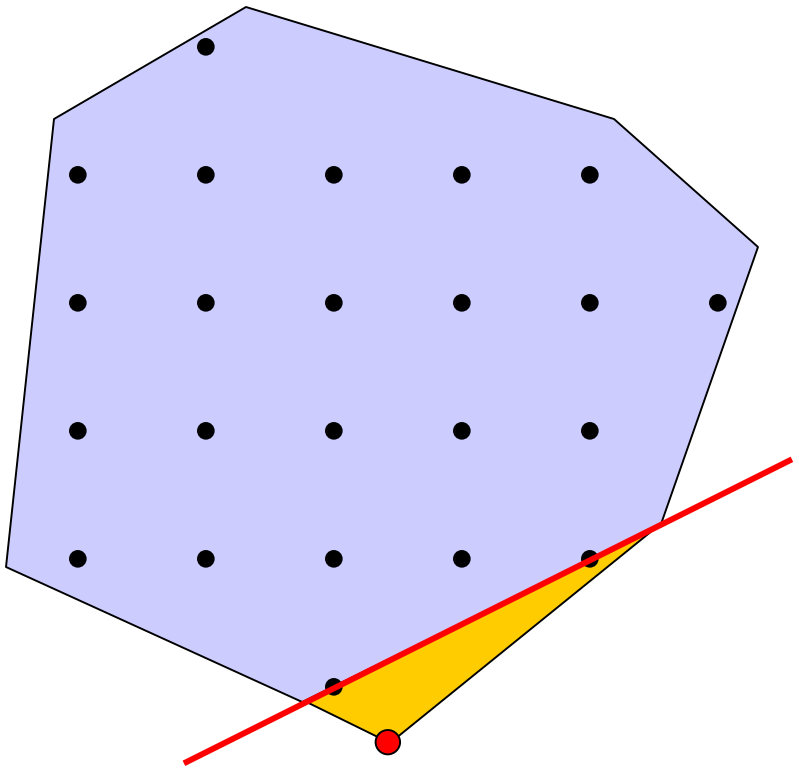
- some values are impossible (due to branching)
- infer other impossibilities in domains

Cutting Planes (MIP)



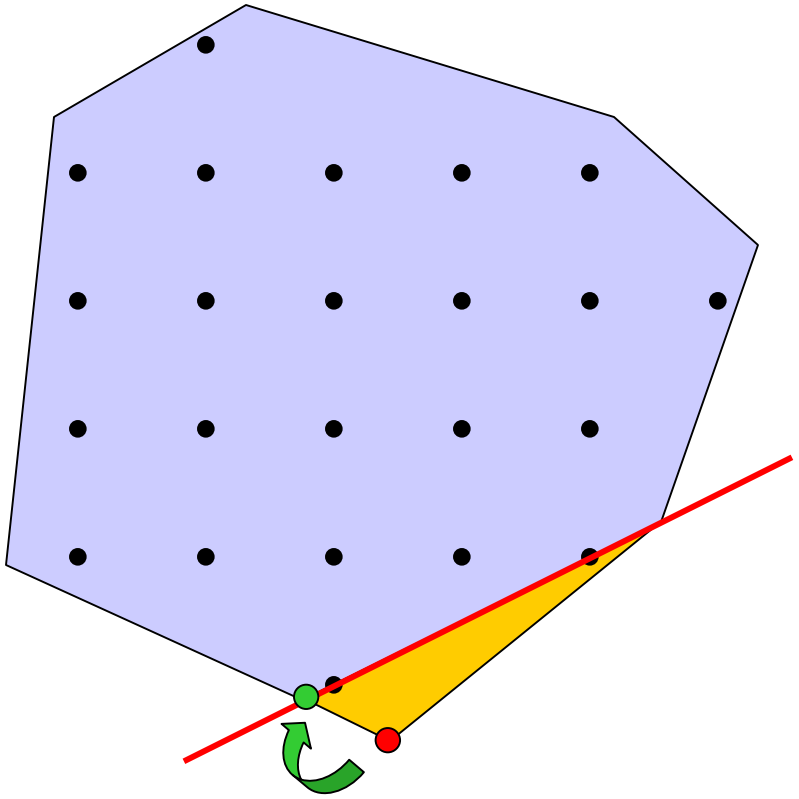
- current solution is fractional

Cutting Planes (MIP)



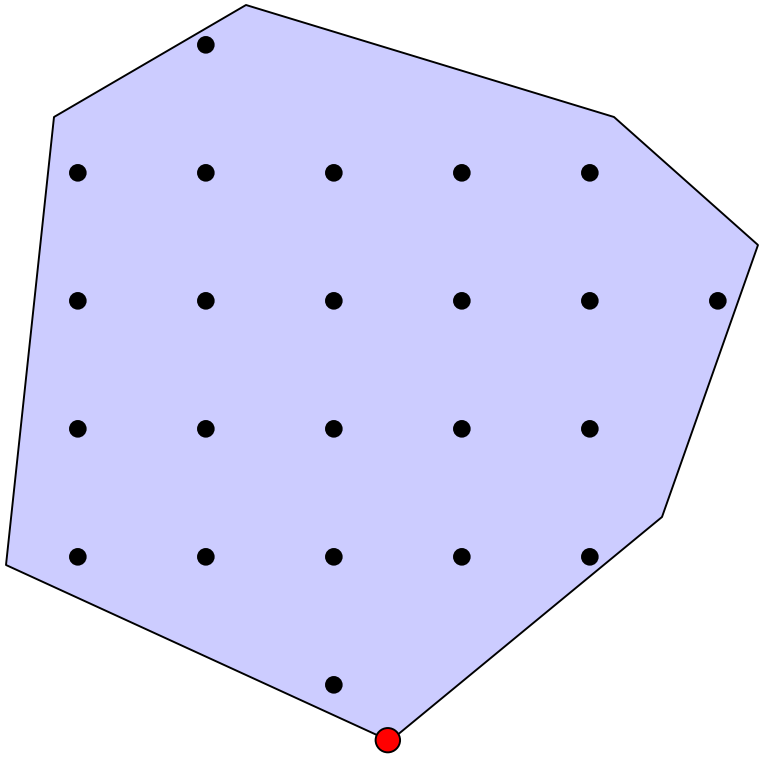
- fractional solution is separated by a cutting plane

Cutting Planes (MIP)



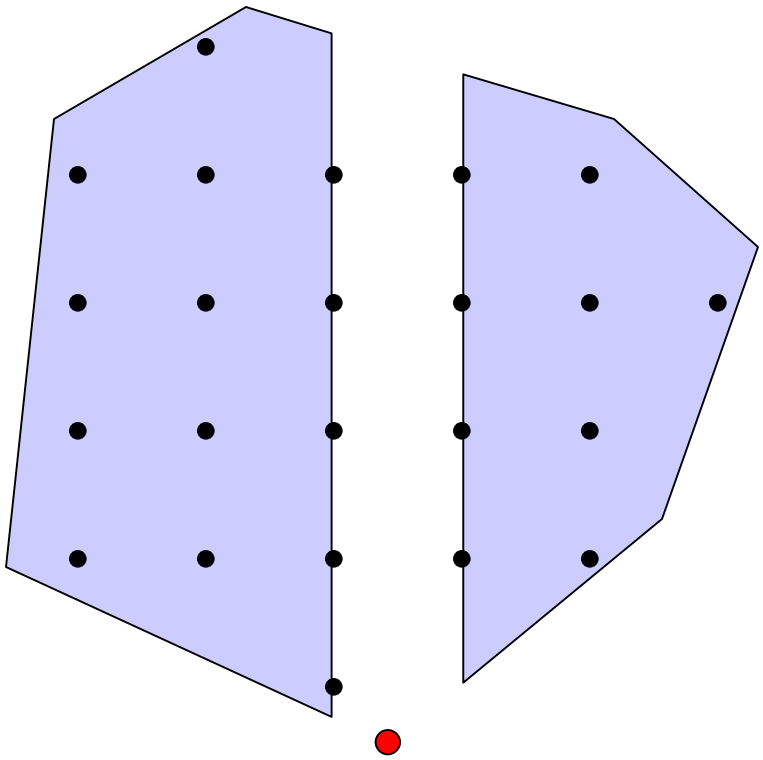
- fractional solution is separated by a cutting plane
- resolving LP relaxation yields new solution

Branching (MIP + CP)



- current solution is fractional

Branching (MIP + CP)



- split problems into sub problems to cut off current solution

Branch and Bound (MIP + CP)

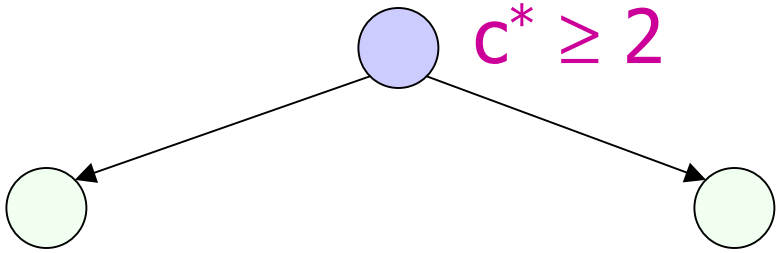
- root node defines global problem

Branch and Bound (MIP + CP)

$c^* \geq 2$

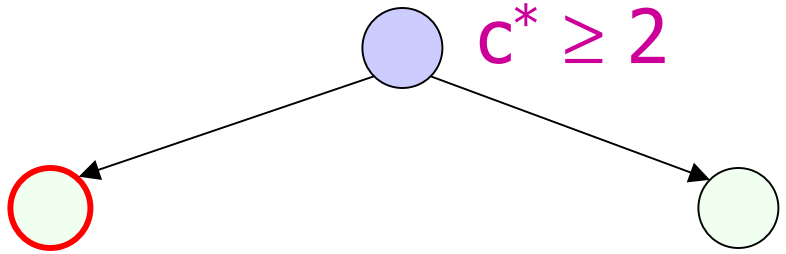
- relaxation yields lower bound

Branch and Bound (MIP + CP)



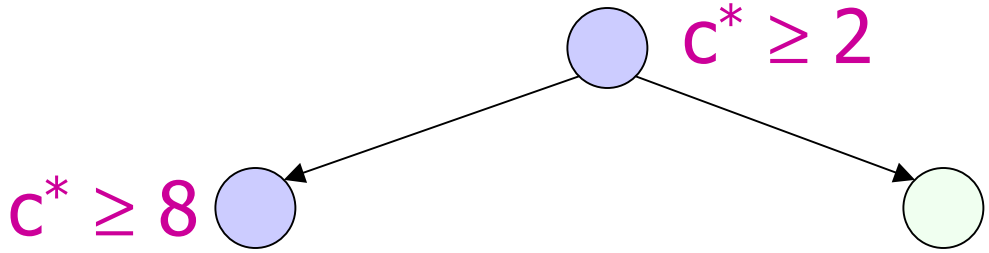
- relaxation yields lower bound
- branching decomposes problem into subproblems

Branch and Bound (MIP + CP)



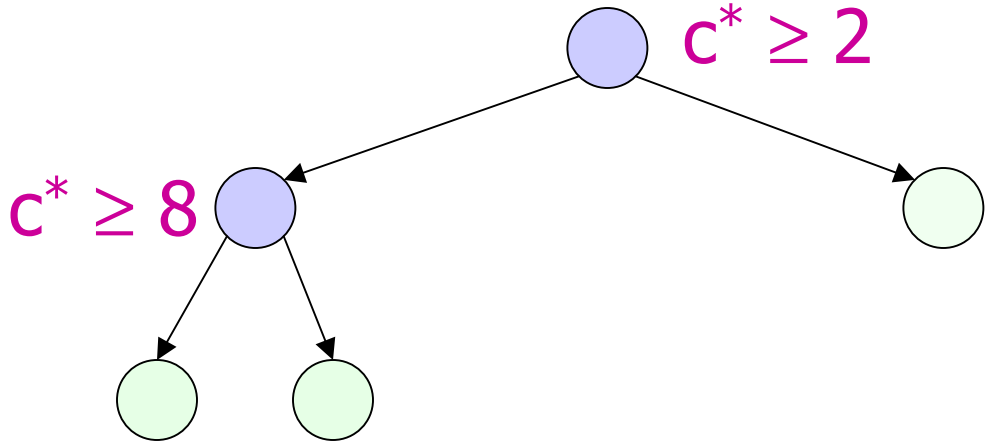
- relaxation yields lower bound
- branching decomposes problem into subproblems
- relaxation is solved for subproblems

Branch and Bound (MIP + CP)



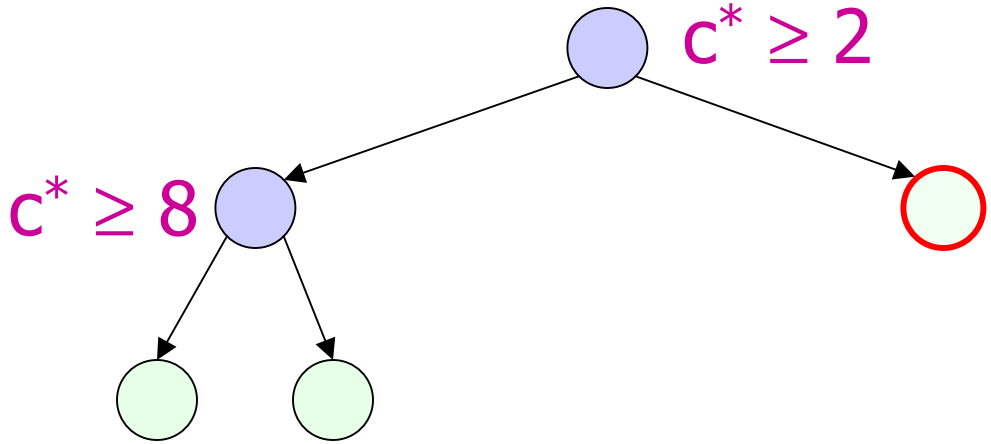
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



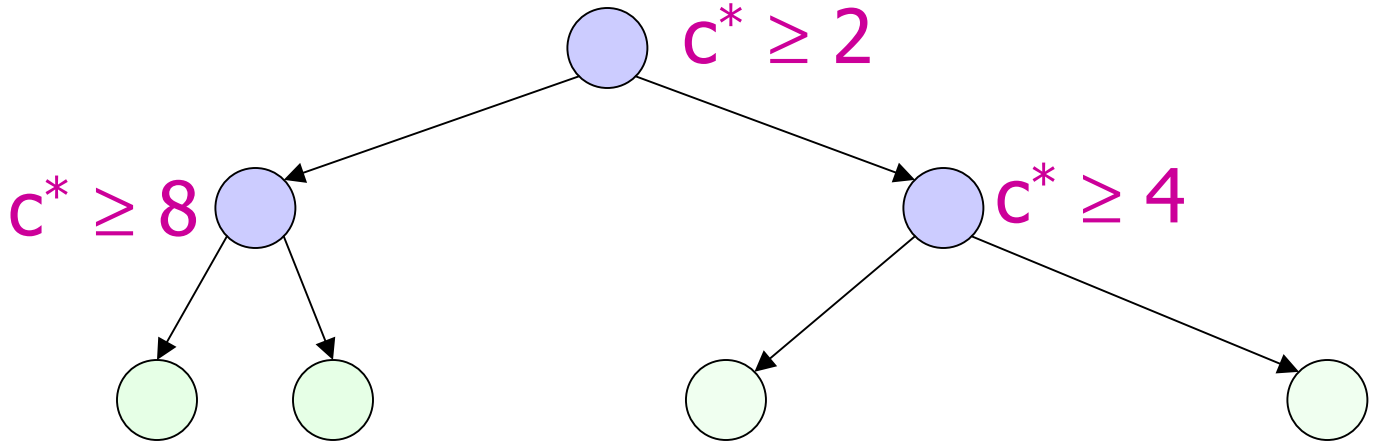
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



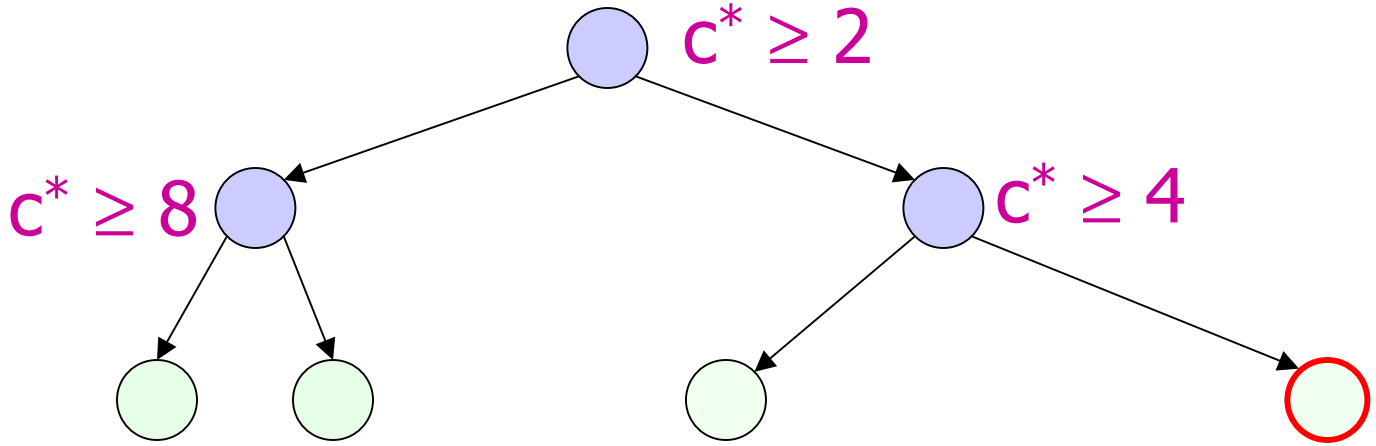
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



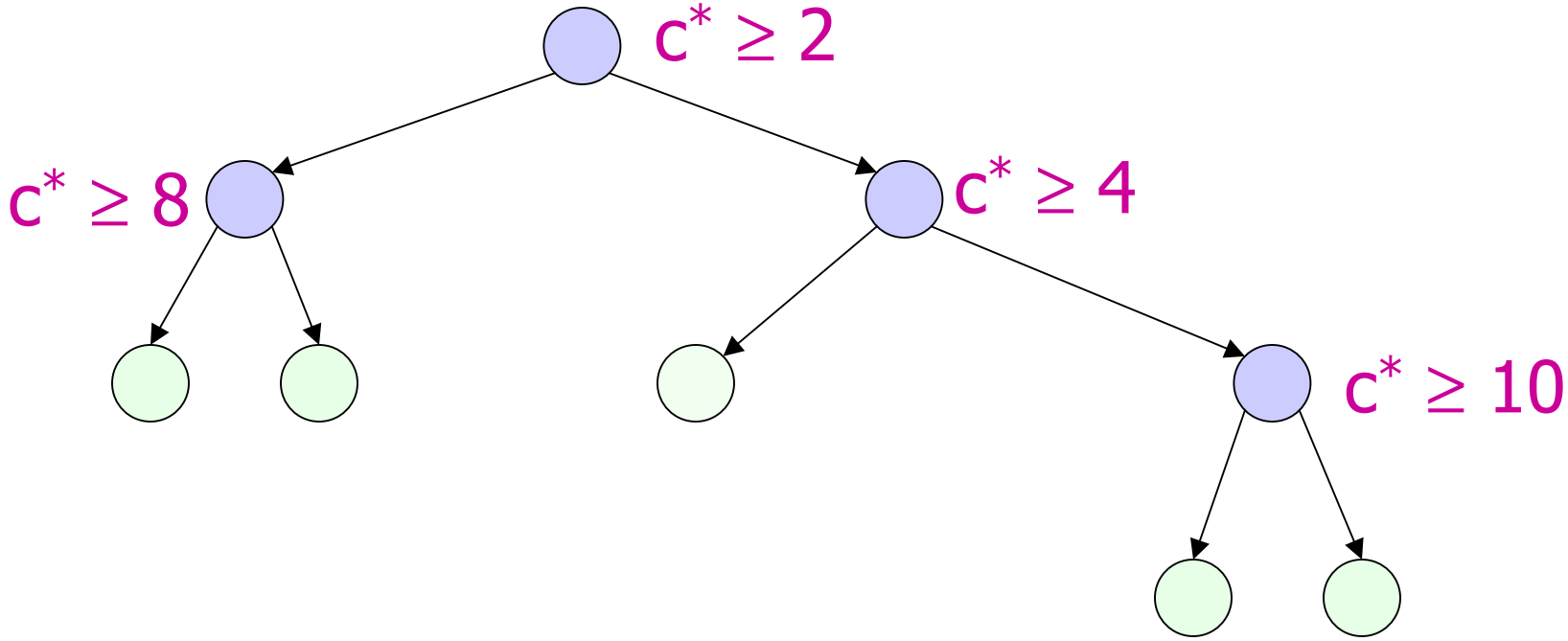
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



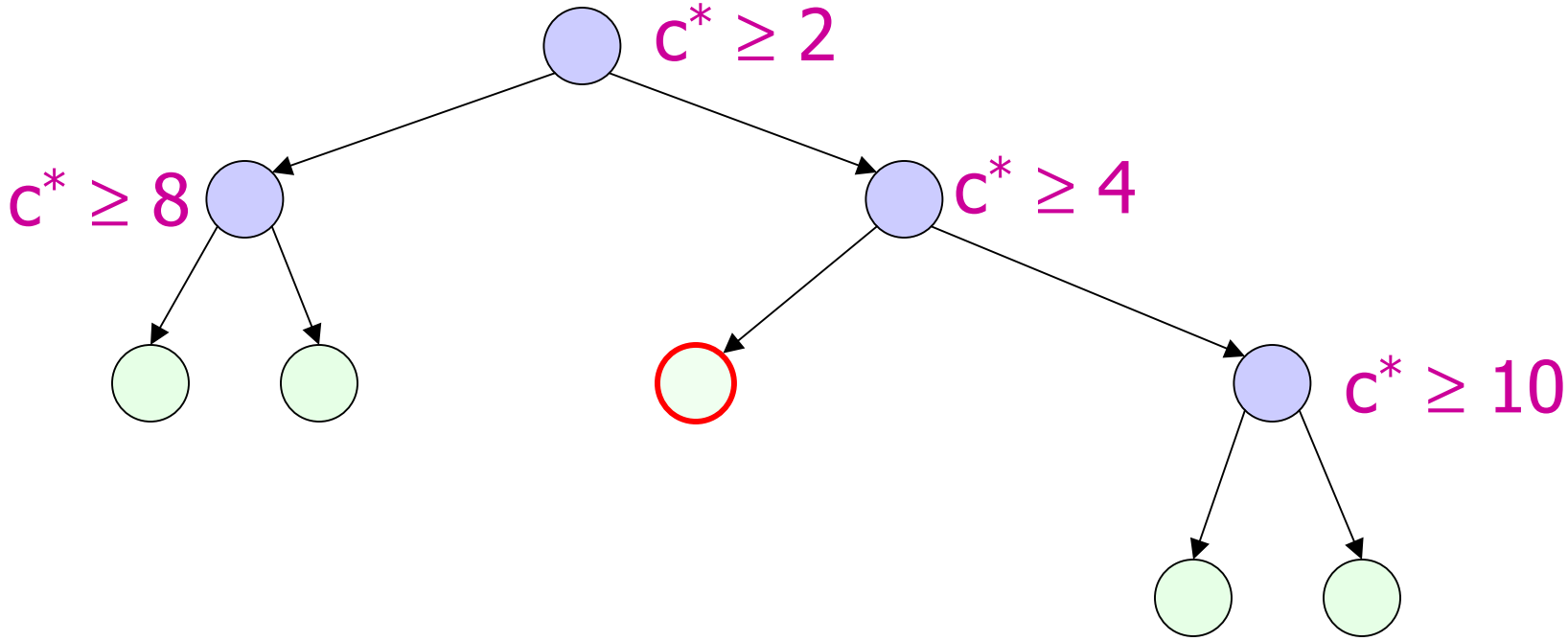
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



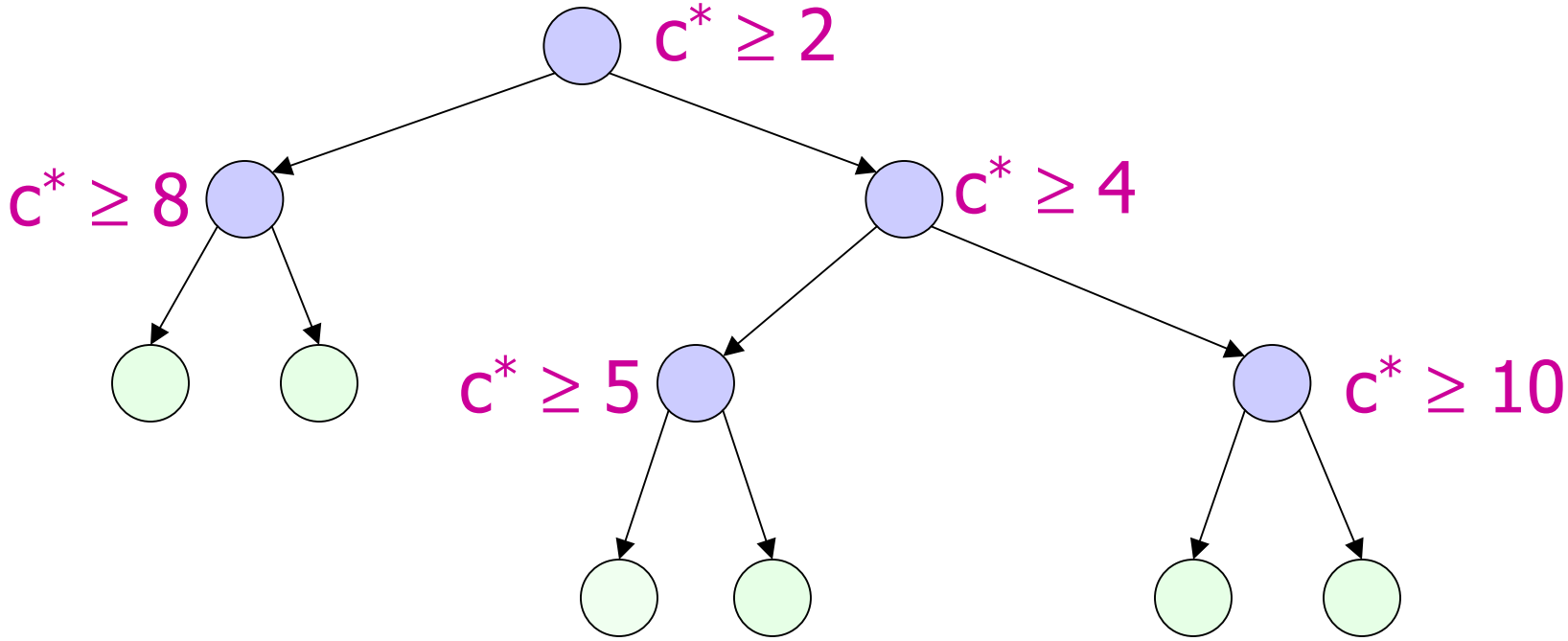
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



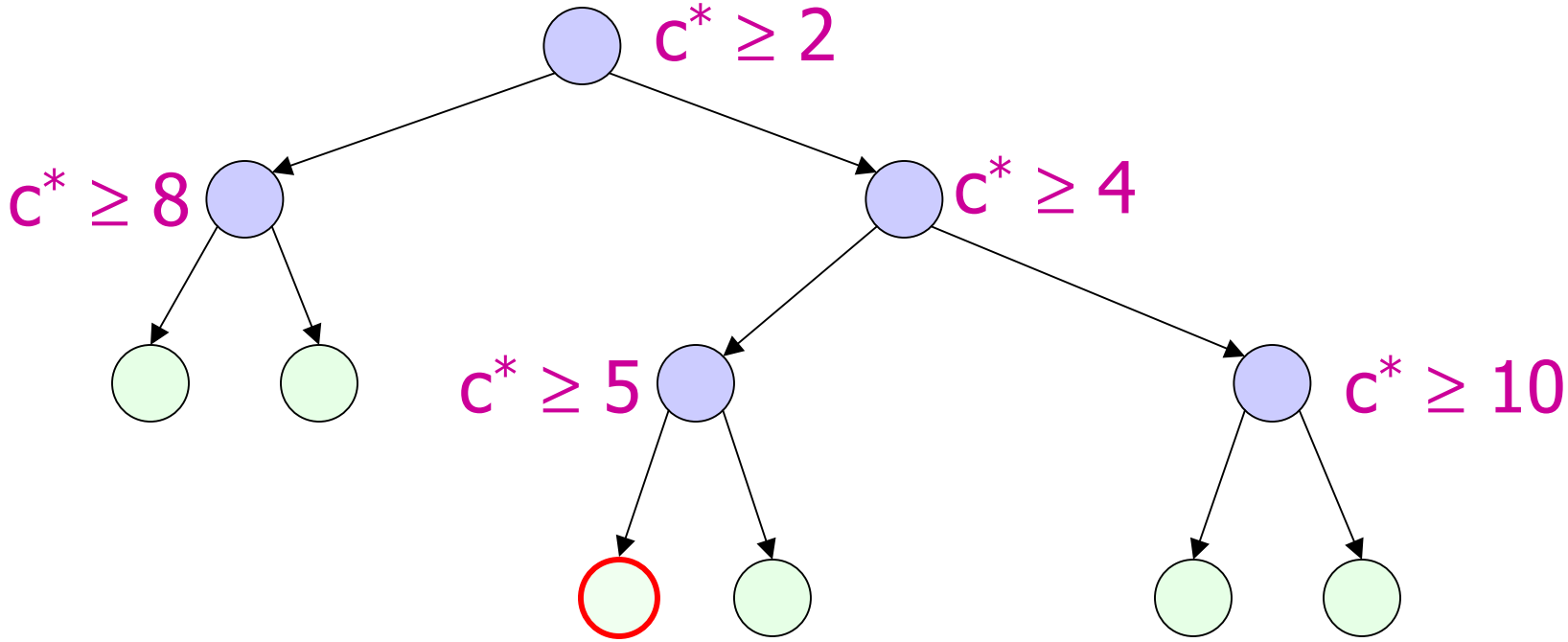
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



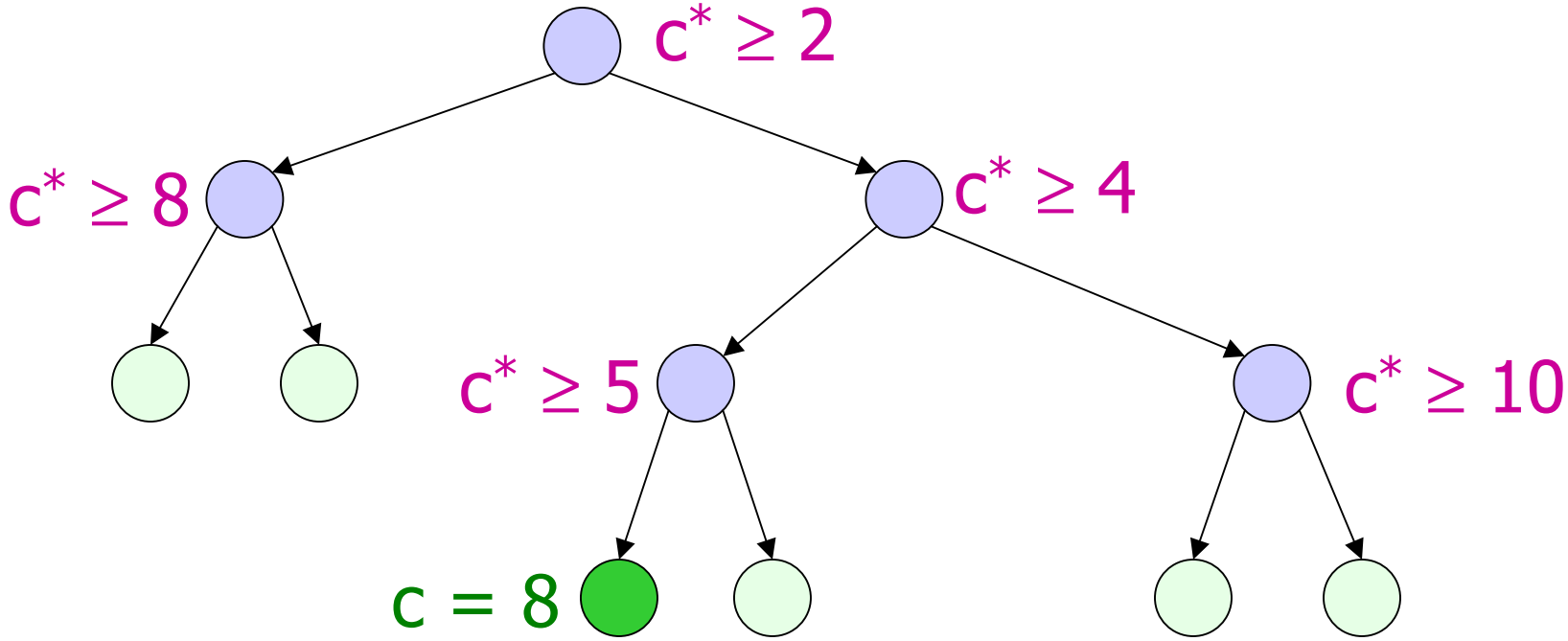
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



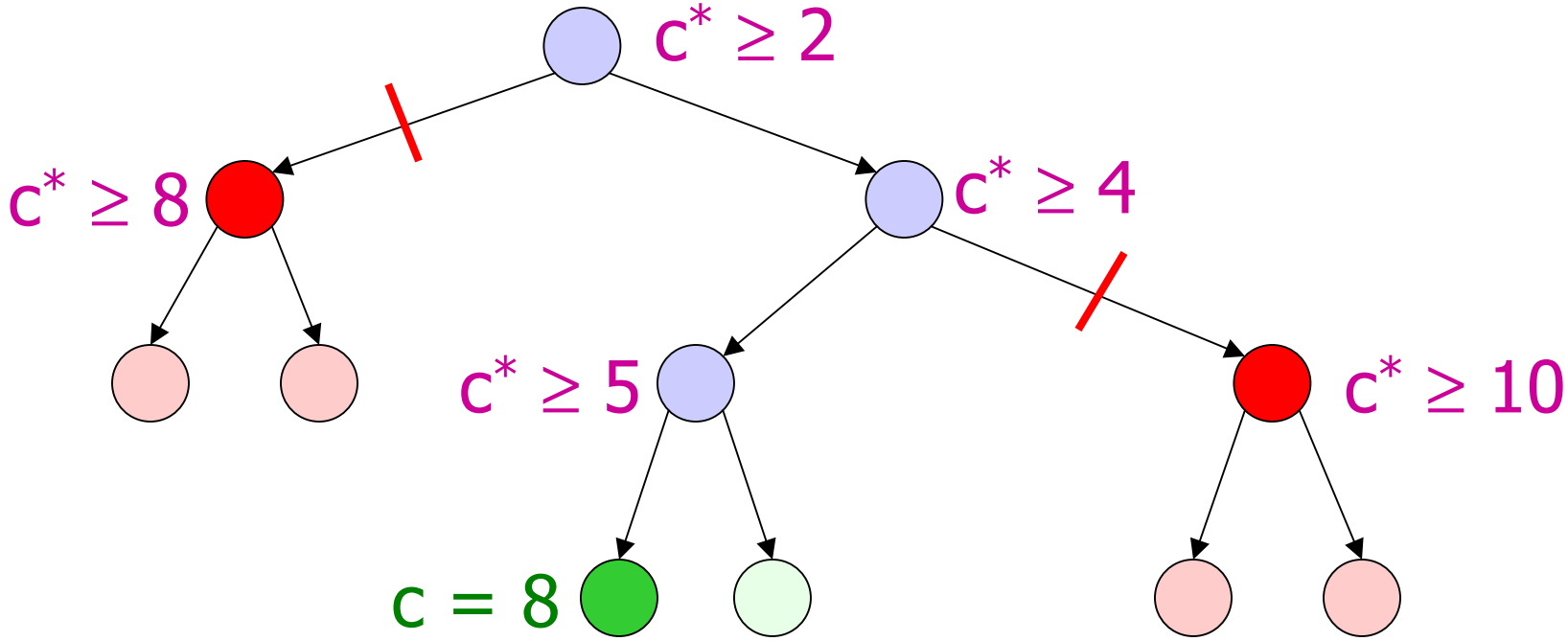
- relaxation yields lower bounds

Branch and Bound (MIP + CP)



- relaxation yields lower bounds
- primal solution yields upper bound

Branch and Bound (MIP + CP)



- relaxation yields lower bounds
- primal solution yields upper bound
- subproblems cannot contain better solution

SCIP as standalone MIP Solver

- reads MPS file format (e.g., generated by ZIMPL)
- built-in MIP specific components:
  - branching rules (reliability, strong, most infeasible, ...)
  - primal heuristics (rounding, diving, feas. pump, ...)
  - node selectors (depth-first, best-first with plunging)
  - presolving (dual fixing, probing, ...)
  - cut separators (clique, impl. bounds, c-MIR, Gomory, strong CG, lifted knapsack cover)
- Approx. 25% slower than CPLEX 9.03

SCIP as CIP framework

- C-Interface with C++ wrapper classes
- infrastructure to support user plugins
  - subproblem and branching tree management
  - global cut pool
  - event mechanism (bound changes, new solutions, ...)
  - lots of statistical data of the solving process
  - efficient memory management
- all existing MIP components are implemented as user plugins $\Rightarrow$ interface is powerful enough for most applications

TSP with SCIP

main program to invoke SCIP	196 lines
TSP file reader	407 lines
Graph storage class	80 lines
Subtour constraint handler	793 lines
Gomory-Hu Tree code	658 lines
farthest insert heuristic	354 lines
2-opt heuristic	304 lines
	2792 lines

SCIP Website

http://scip.zib.de

(download of SCIP is not yet possible)