ORF307 – Optimization

21. Integer optimization algorithms

Bartolomeo Stellato — Spring 2023

Ed Forum

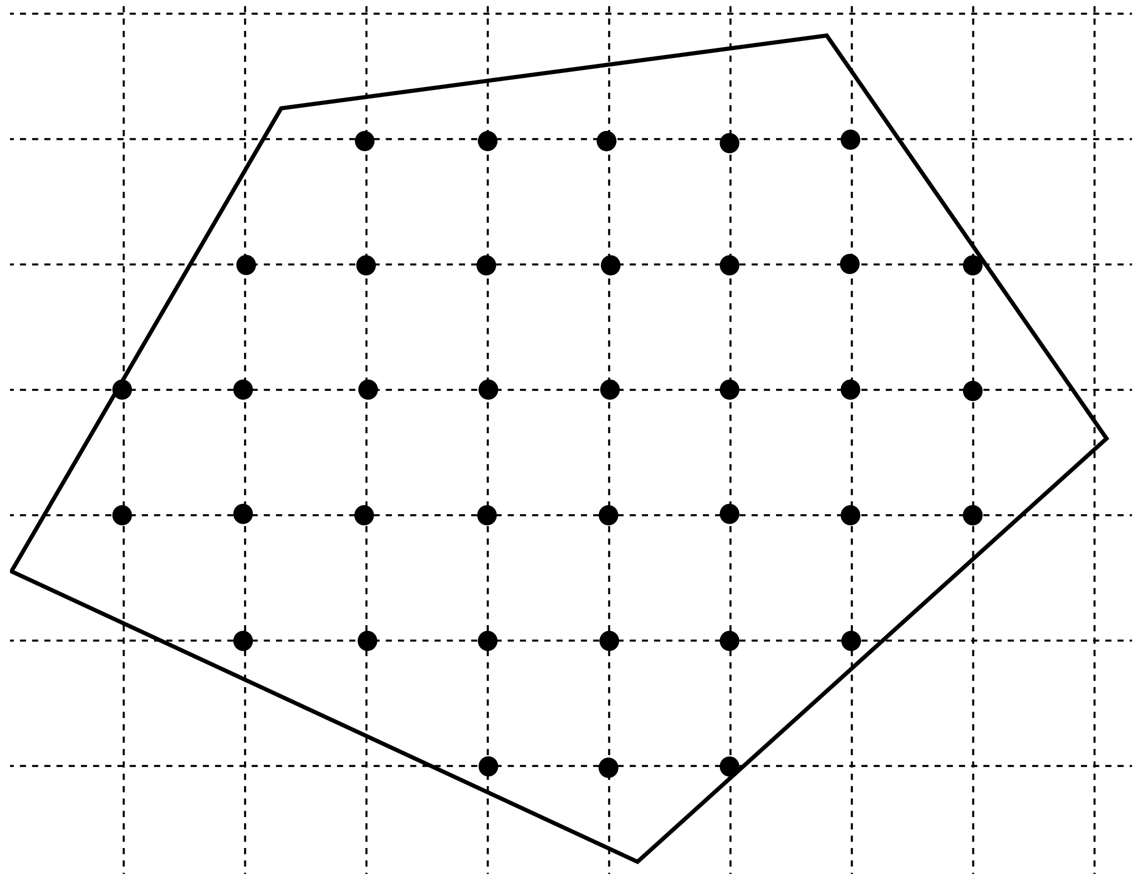
- In Lecture 16, we discussed the integrality theorem for flow networks, but I was still wondering how we know in general whether an integer valued solution exists?

- We also went over the facility location problem and different ways to formulate it (max bound vs avg bound). However, I was confused about the purpose of both formulations if one was said to be better than the other. Additionally, how different could an optimal integer solution be compared to a decimal solution (would rounding work)?

Recap

Relaxations

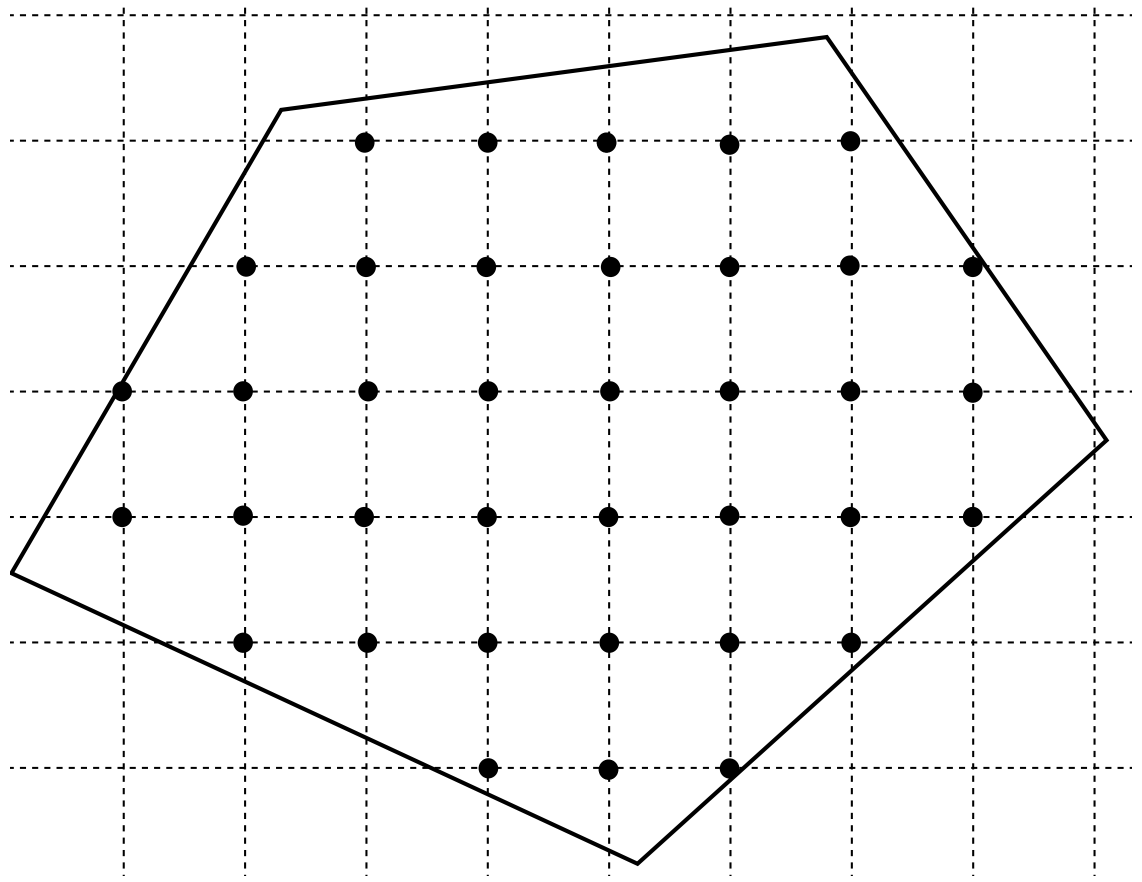
$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array}$$

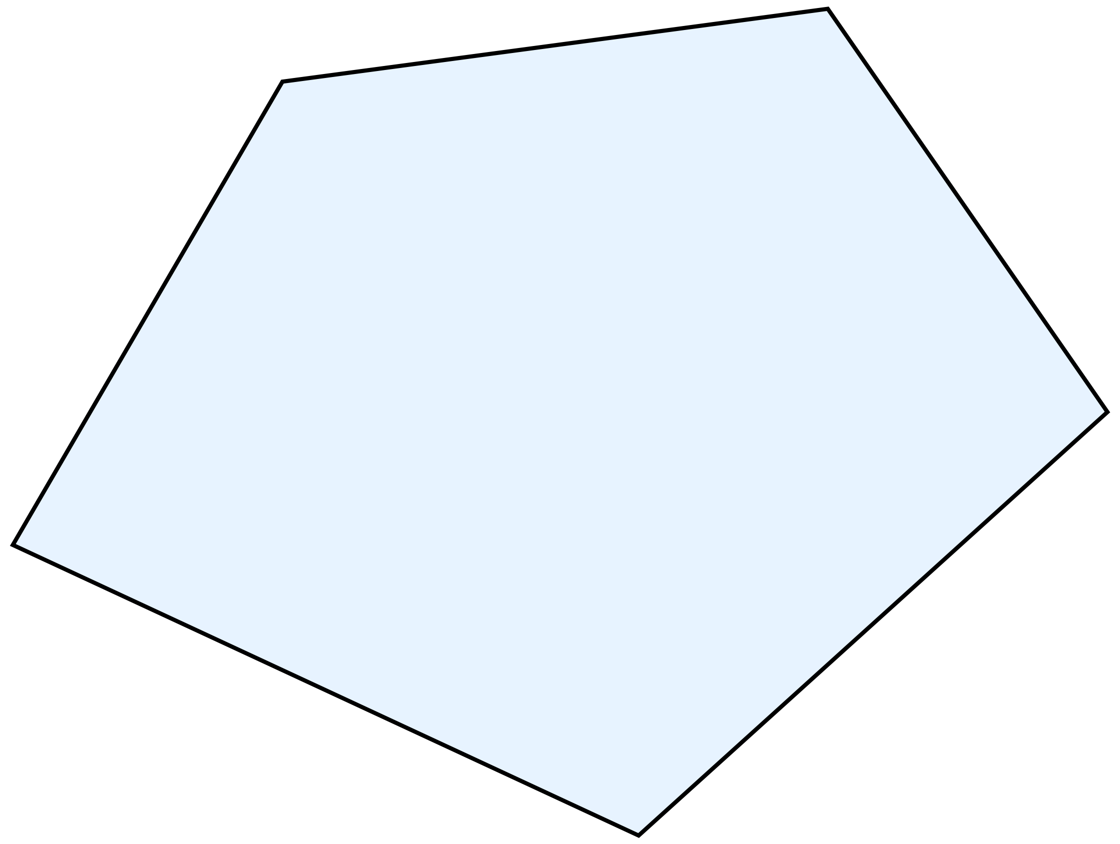


Relaxations

Remove integrality constraints

$$
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & Ax \leq b \\
& x_i \in \mathbf{Z}, \quad i \in \mathcal{I}
\end{array}
\longrightarrow
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & Ax \leq b
\end{array}
$$





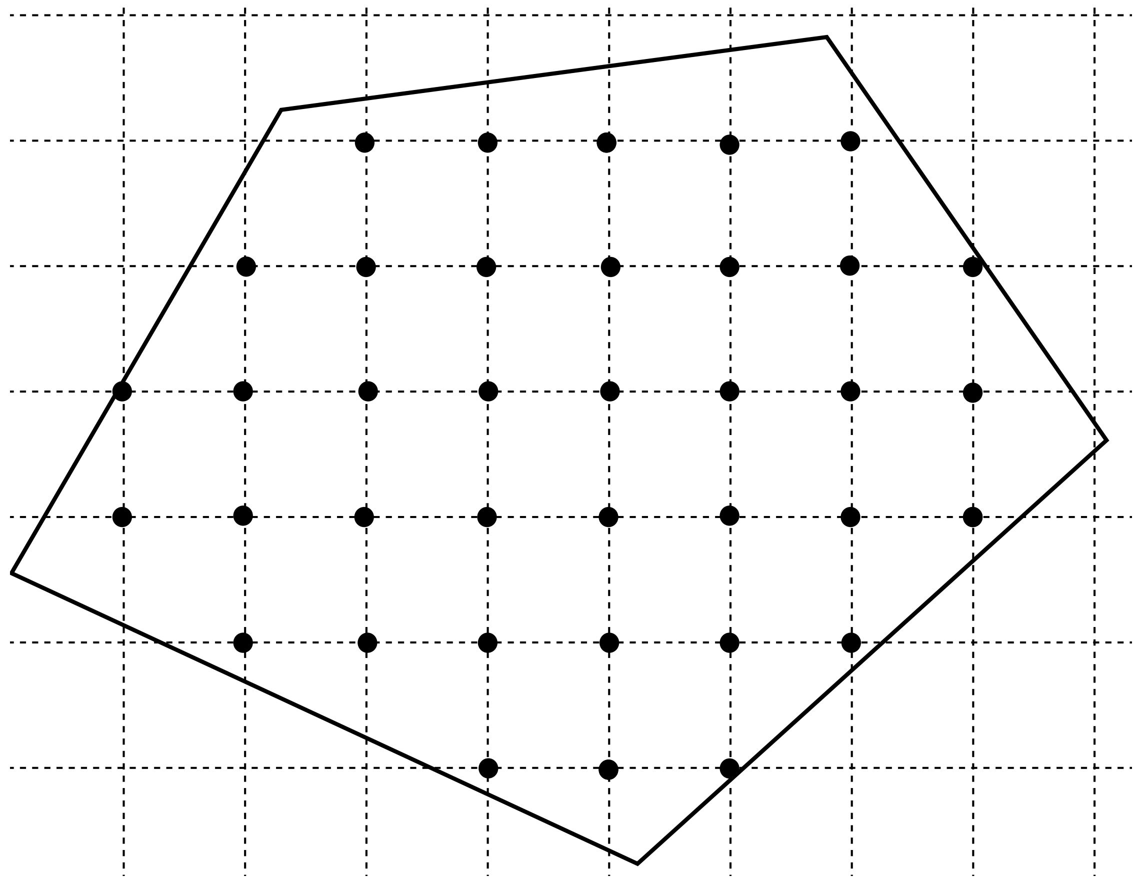
Relaxations

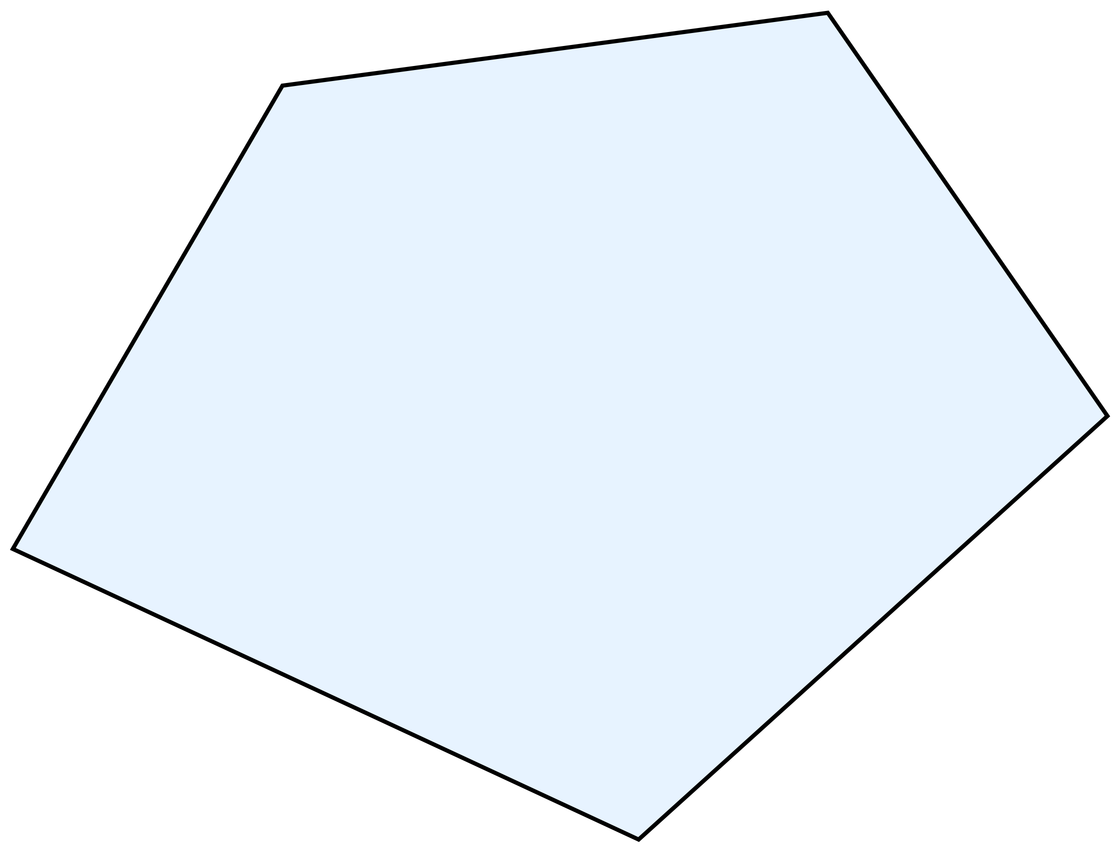
Remove integrality constraints

$$P_{\text{ip}} \longrightarrow \quad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array}$$

$$\longrightarrow$$

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \end{array} \quad \longleftarrow P_{\text{rel}}$$





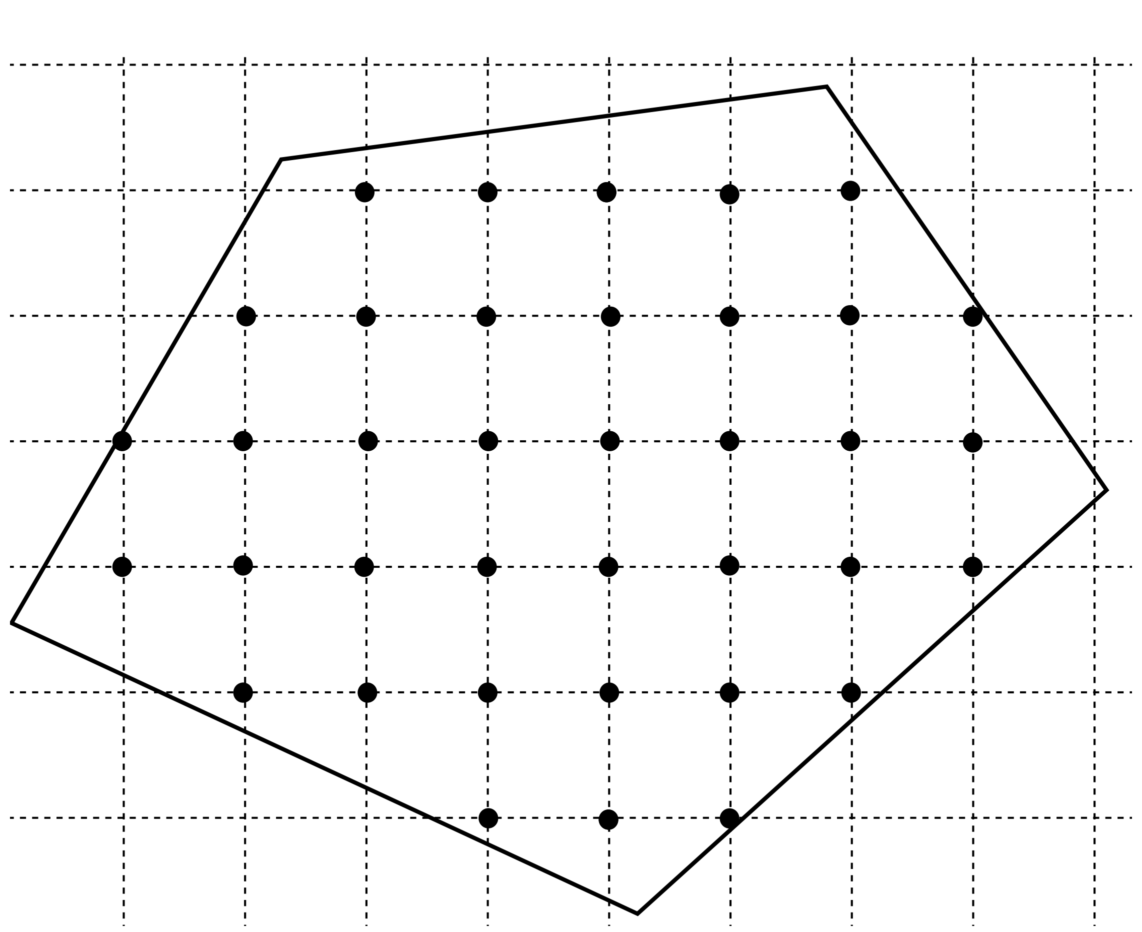
Relaxations

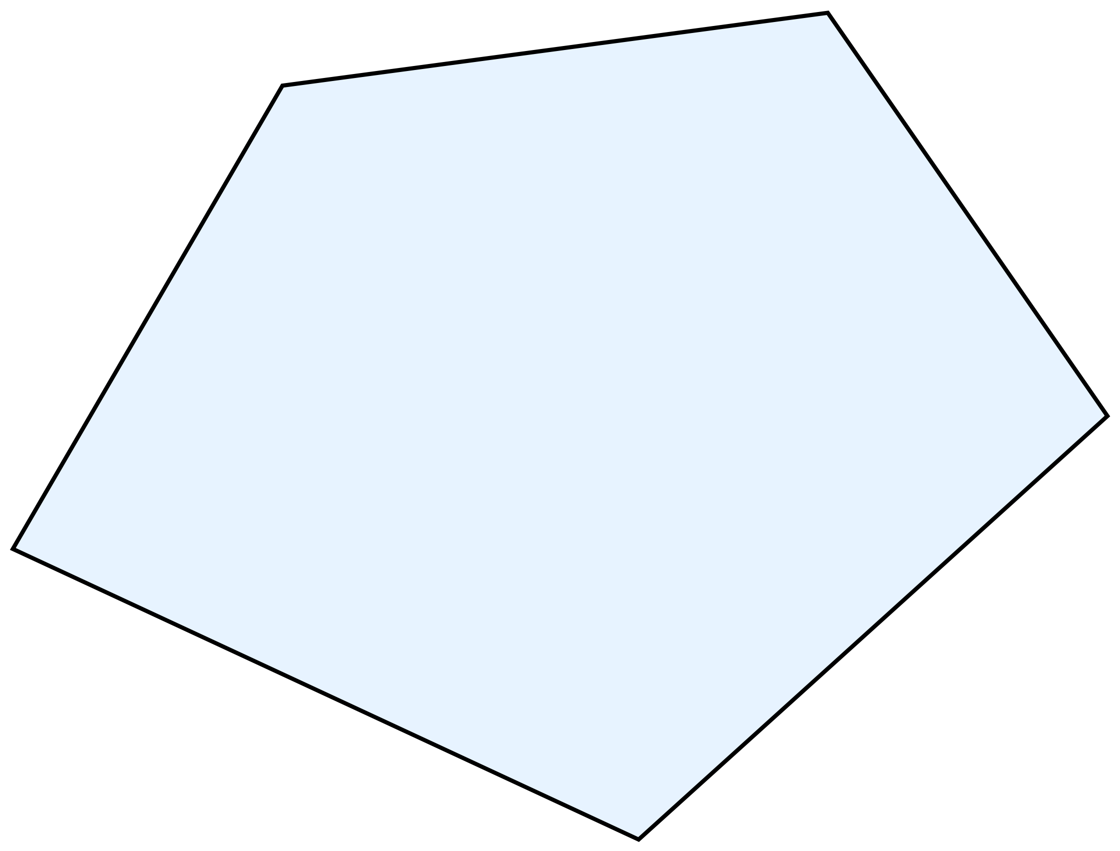
Remove integrality constraints

$$P_{\text{ip}} \longrightarrow \quad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array}$$

$\longrightarrow$

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \end{array} \quad \longleftarrow P_{\text{rel}}$$





$P_{\text{ip}} \subset P_{\text{rel}}$ $\longrightarrow$ Relaxations provide **lower bounds** to $p^\star_{\text{ip}}$

$$p^\star_{\text{rel}} \le p^\star_{\text{ip}}$$

Ideal formulations

A formulation is ideal if solving its relaxation gives an integer feasible point

$$
\begin{array}{ll}
\text{minimize} & c^T x \\
\text{subject to} & Ax \leq b \\
& x_i \in \mathbf{Z}, \quad i \in \mathcal{I}
\end{array}
$$

Ideal formulations

A formulation is ideal if solving its relaxation gives an integer feasible point

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array} \longrightarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \end{array} \longrightarrow x^\star$$

integer feasible

Ideal formulations

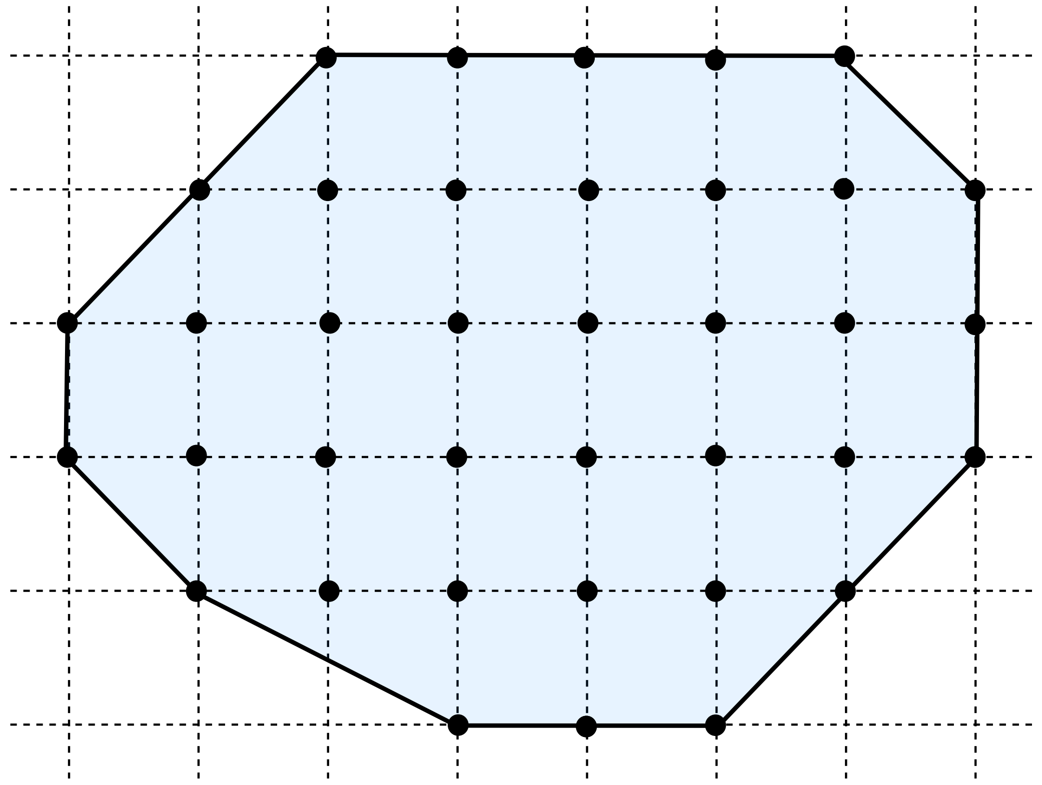
A formulation is ideal if solving its relaxation gives an integer feasible point

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array} \longrightarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \end{array} \longrightarrow x^\star$$

integer feasible

This happens if

$$\mathbf{conv}\, P = \{Ax \le b\}$$



Ideal formulations

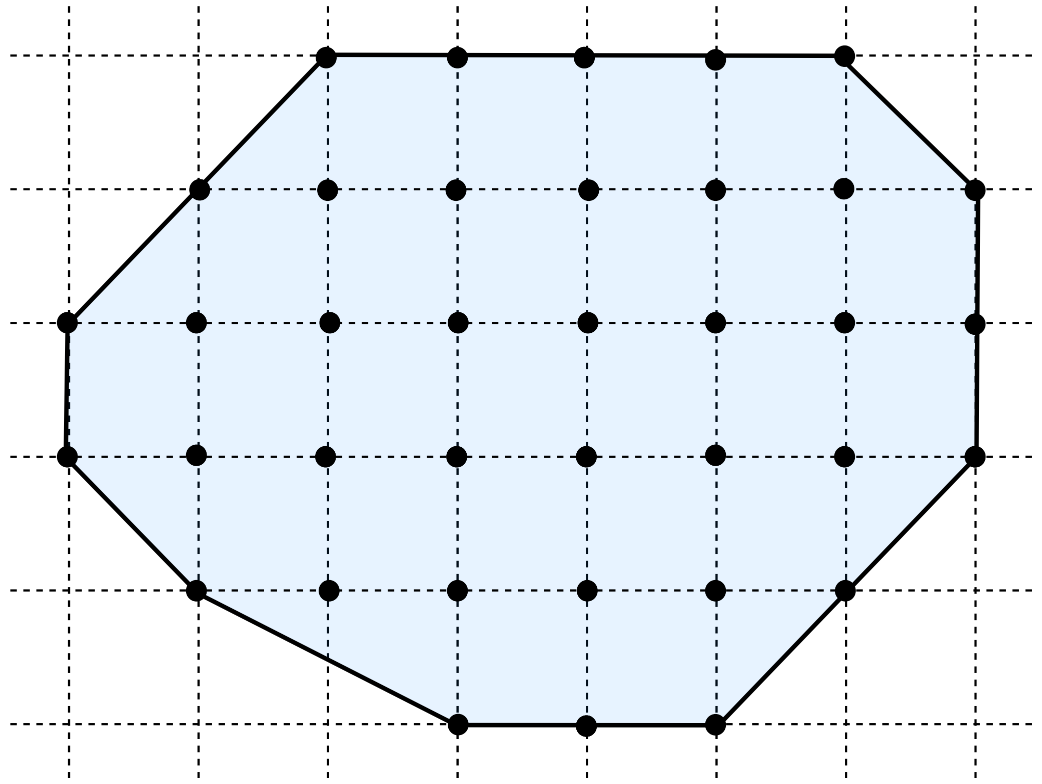
A formulation is ideal if solving its relaxation gives an integer feasible point

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array} \longrightarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \end{array} \longrightarrow x^\star$$

integer feasible

This happens if

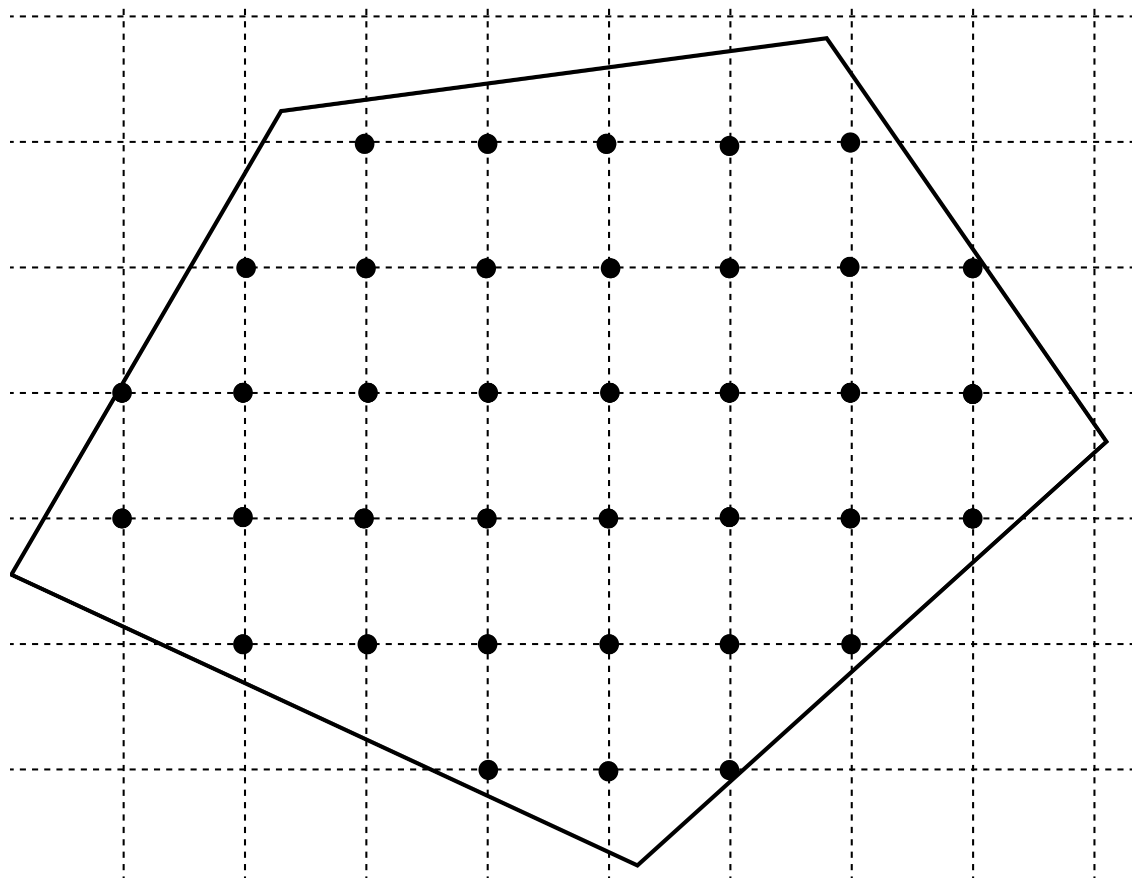
$$\mathbf{conv}\, P = \{Ax \le b\}$$



It is very hard to construct ideal formulations!

How do we solve integer optimization problems?

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \mathbf{Z}, \quad i \in \mathcal{I} \end{array}$$



Main idea

Refine the feasible set until the relaxation gives integer feasible solutions!

Today's lecture

Integer optimization algorithms

- Branch and bound algorithm
- Branch and bound rules
- Examples
- Cardinality minimization

Branch-and-bound algorithm

Example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \le b \\ & x_1 \in \{0, 1\} \end{array}$$

How do you solve it?

Example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_1 \in \{0, 1\} \end{array}$$

How do you solve it?

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \{0, 1\}, \quad i = 1, \ldots, 10 \end{array}$$

Example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_1 \in \{0, 1\} \end{array}$$

How do you solve it?

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \{0, 1\}, \quad i = 1, \ldots, 10 \end{array}$$

- Solve $2^{10} = 1024$ LPs
- Parallelize solutions
- Warm-start: similar problems

Example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_1 \in \{0, 1\} \end{array}$$

How do you solve it?

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \{0, 1\}, \quad i = 1, \ldots, 10 \end{array}$$

- Solve $2^{10} = 1024$ LPs
- Parallelize solutions
- Warm-start: similar problems

It can quickly explode: $2^{30} \approx 1$ bln

Example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_1 \in \{0, 1\} \end{array}$$

How do you solve it?

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x_i \in \{0, 1\}, \quad i = 1, \dots, 10 \end{array}$$

- Solve $2^{10} = 1024$ LPs
- Parallelize solutions
- Warm-start: similar problems

It can quickly explode: $2^{30} \approx 1$ bln

Branch and bound works more systematically
and
(hopefully) decreases the number of subproblems

Branch and bound algorithm

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in P_{\text{ip}} \end{array}$$

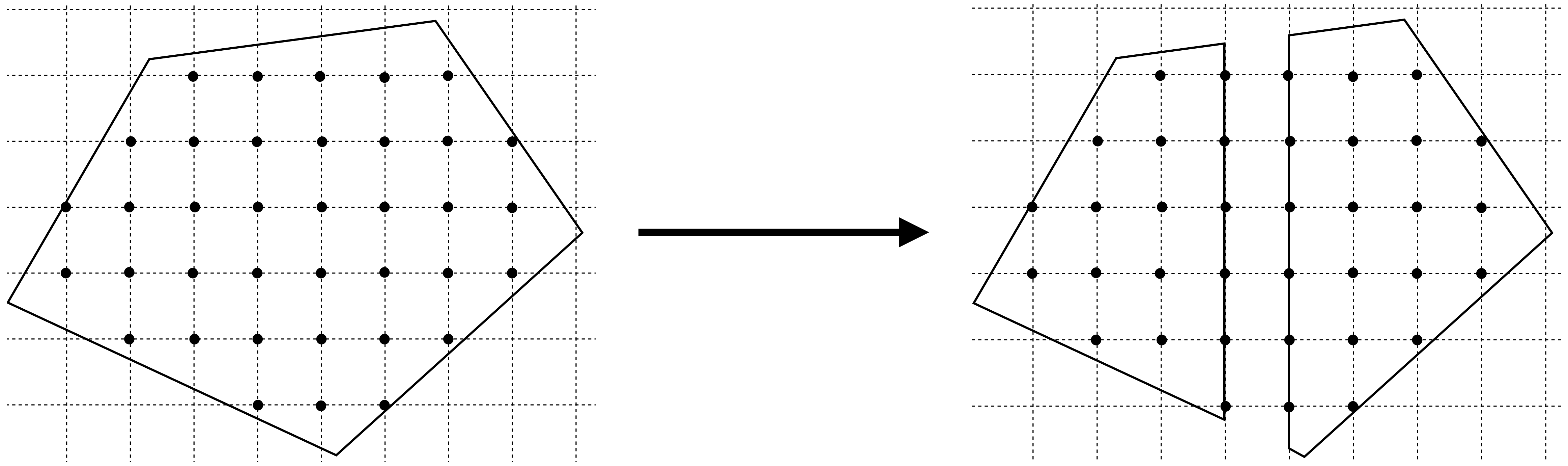
$$P_{\text{ip}} = \{x \mid Ax \leq b, \quad x_i \in \mathbf{Z},\ i \in \mathcal{I}\}$$

Divide and conquer

- **Partition** P_{ip} in smaller sets S^j
- **Solve subproblems**

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j \end{array}$$

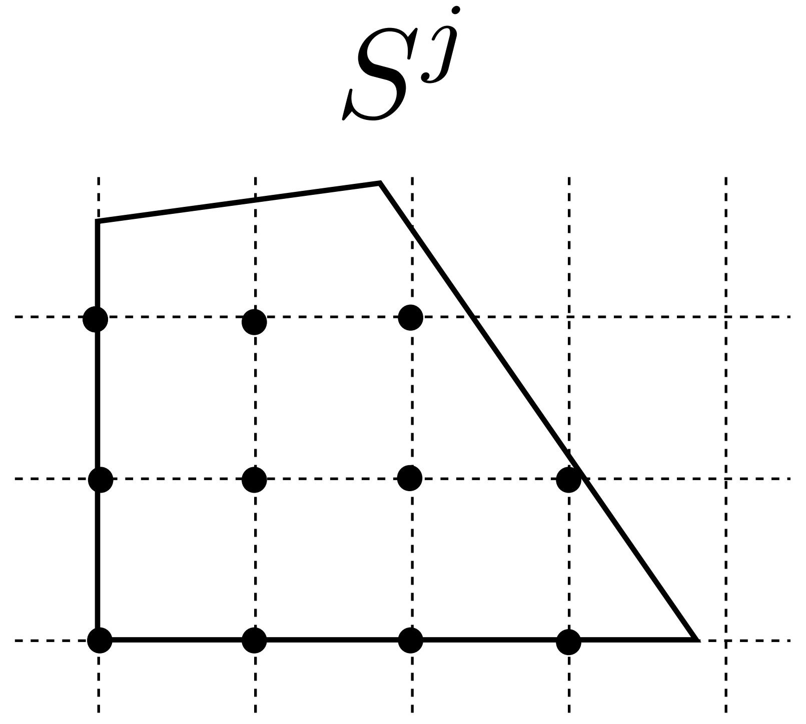
- Until **optimal** $x^\star$ **is found**



Two efficient subroutines

For every region S^j

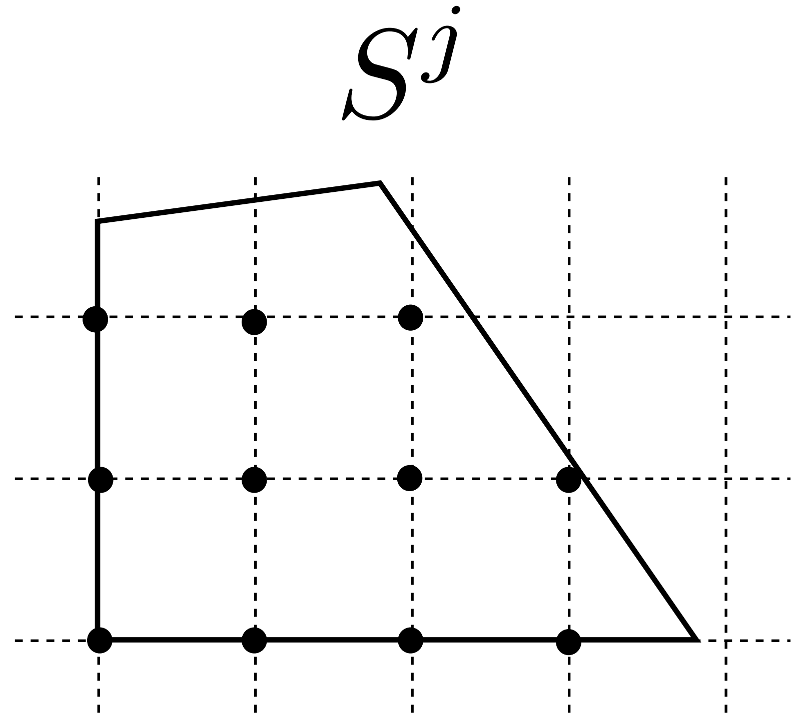
$$\Phi(S^j) = \min_{x \in S^j} c^T x$$



Two efficient subroutines

For every region S^j

$$\Phi(S^j) = \min_{x \in S^j} c^T x$$



Lower and upper bounds

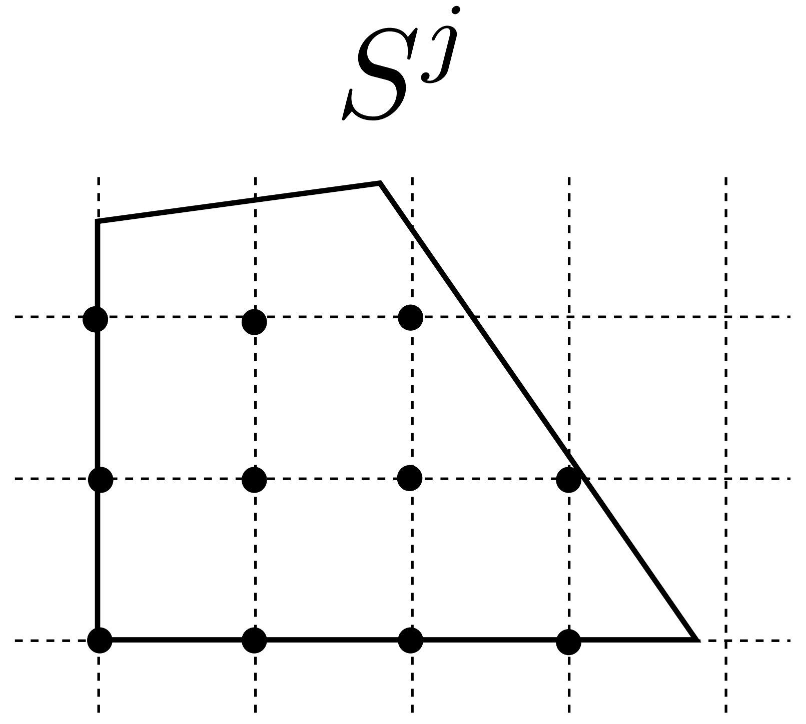
(they must be cheap to compute)

$$\Phi_{\mathrm{lb}}(S^j) \leq \Phi(S^j) \leq \Phi_{\mathrm{ub}}(S^j)$$

Two efficient subroutines

For every region S^j

$$\Phi(S^j) = \min_{x \in S^j} c^T x$$



Lower and upper bounds

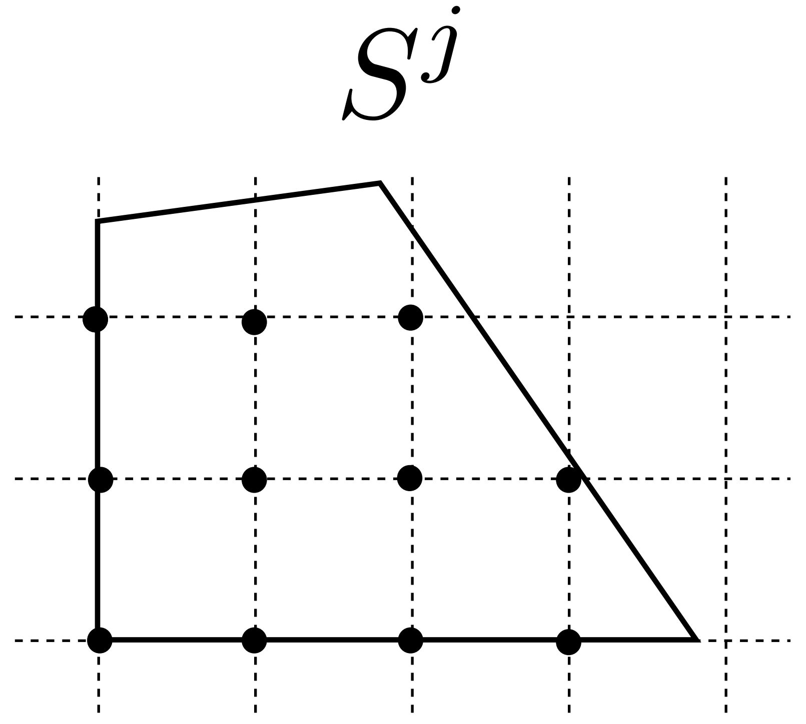
(they must be cheap to compute)

$$\Phi_{\mathrm{lb}}(S^j) \le \Phi(S^j) \le \Phi_{\mathrm{ub}}(S^j)$$

Lower bound

(relaxation)

Two efficient subroutines



For every region S^j

$$\Phi(S^j) = \min_{x \in S^j} c^T x$$

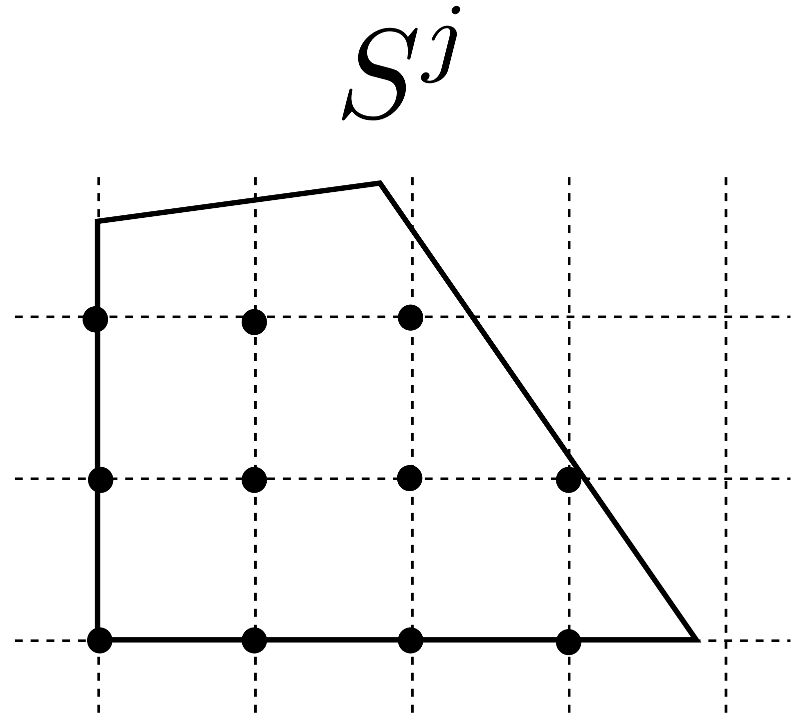
Lower and upper bounds
(they must be cheap to compute)

Lower bound
(relaxation)

$$\Phi_{\mathrm{lb}}(S^j) \leq \Phi(S^j) \leq \Phi_{\mathrm{ub}}(S^j)$$

Upper bound
(evaluate any point)

Two efficient subroutines



For every region S^j

$$\Phi(S^j) = \min_{x \in S^j} c^T x$$

Lower and upper bounds
(they must be cheap to compute)

Lower bound (relaxation) → $\Phi_{\mathrm{lb}}(S^j) \leq \Phi(S^j) \leq \Phi_{\mathrm{ub}}(S^j)$ ← **Upper bound** (evaluate any point)

Bounds guide us in the search!

Branch and bound algorithm

minimize $\quad c^T x$

subject to $\quad x \in P_{\mathrm{ip}}$

$$P_{\mathrm{ip}} = \{x \mid Ax \le b, \quad x_i \in \mathbf{Z},\ i \in \mathcal{I}\}$$

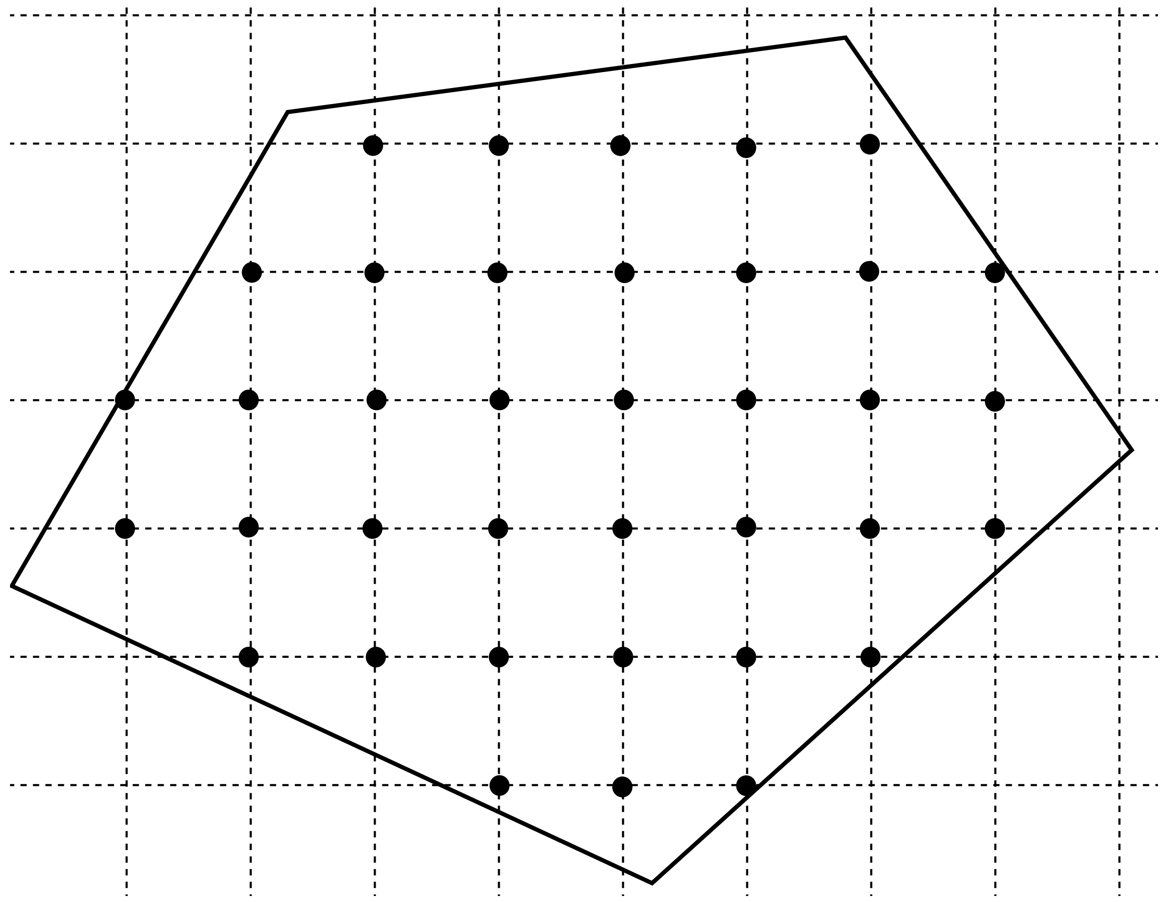
Iterations

1. **Branch**: create/refine the partition if P_{ip} and get S^j
2. **Bound:**
   - Compute **lower** and **upper bounds**
     $$L_j = \Phi_{\mathrm{lb}}(S^j), \quad U_j = \Phi_{\mathrm{ub}}(S^j), \quad \forall j$$
   - Update **global bounds** on $c^T x^\star$
     $$L = \min_j \{L_j\}, \quad U = \min_j \{U_j\}$$
3. If $U - L \le \epsilon$, **break**

Branch and bound

Example in 2D

P_{ip}



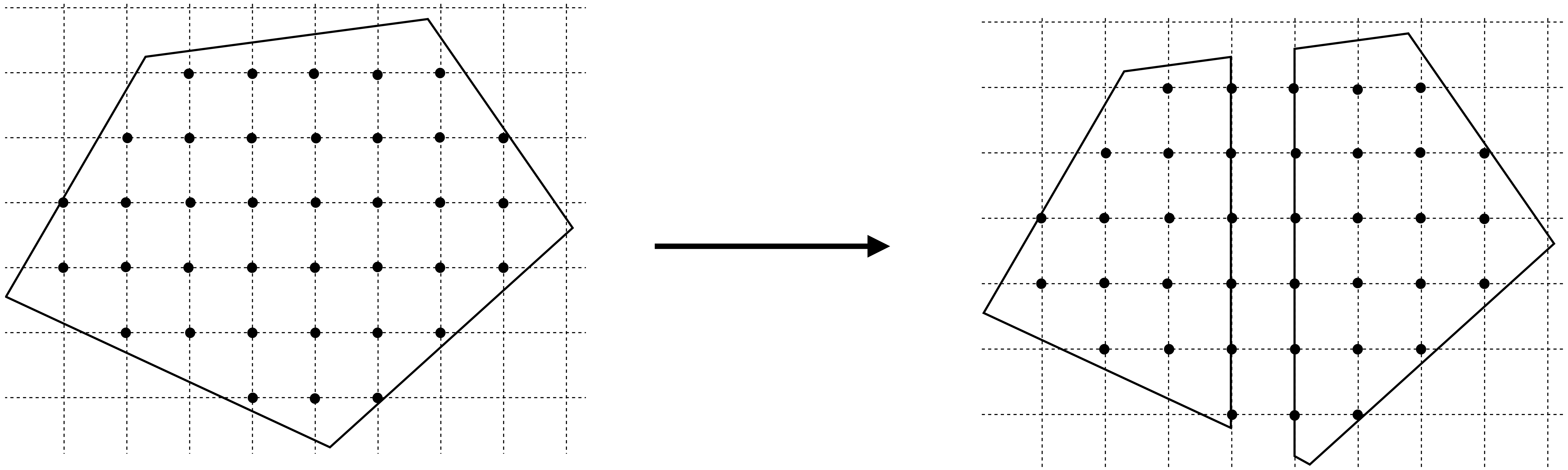
$U = 8.5$

$L = 2$

Branch and bound

Example in 2D

P_{ip} S_1 S_2



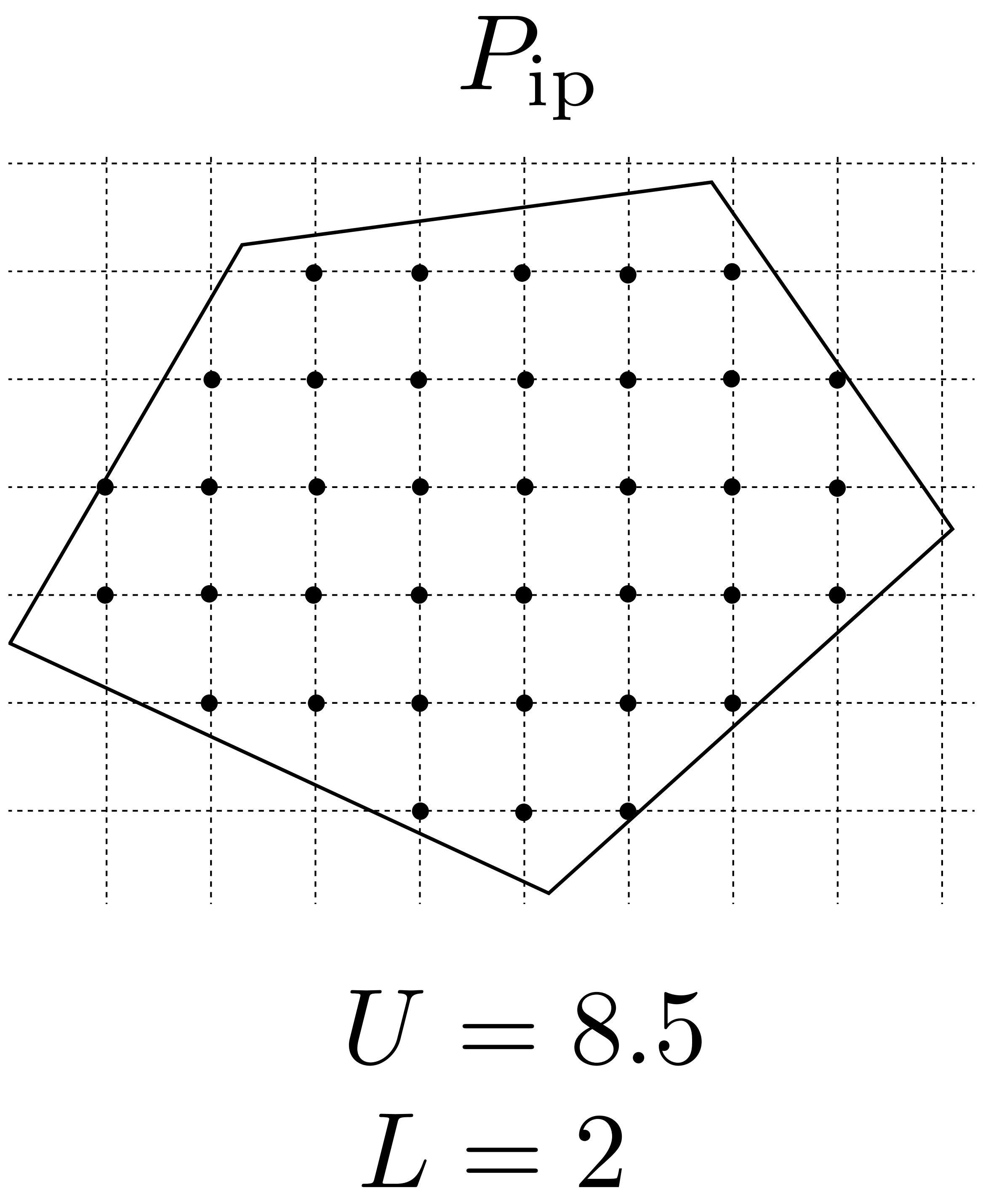
$U = 8.5$
$L = 2$

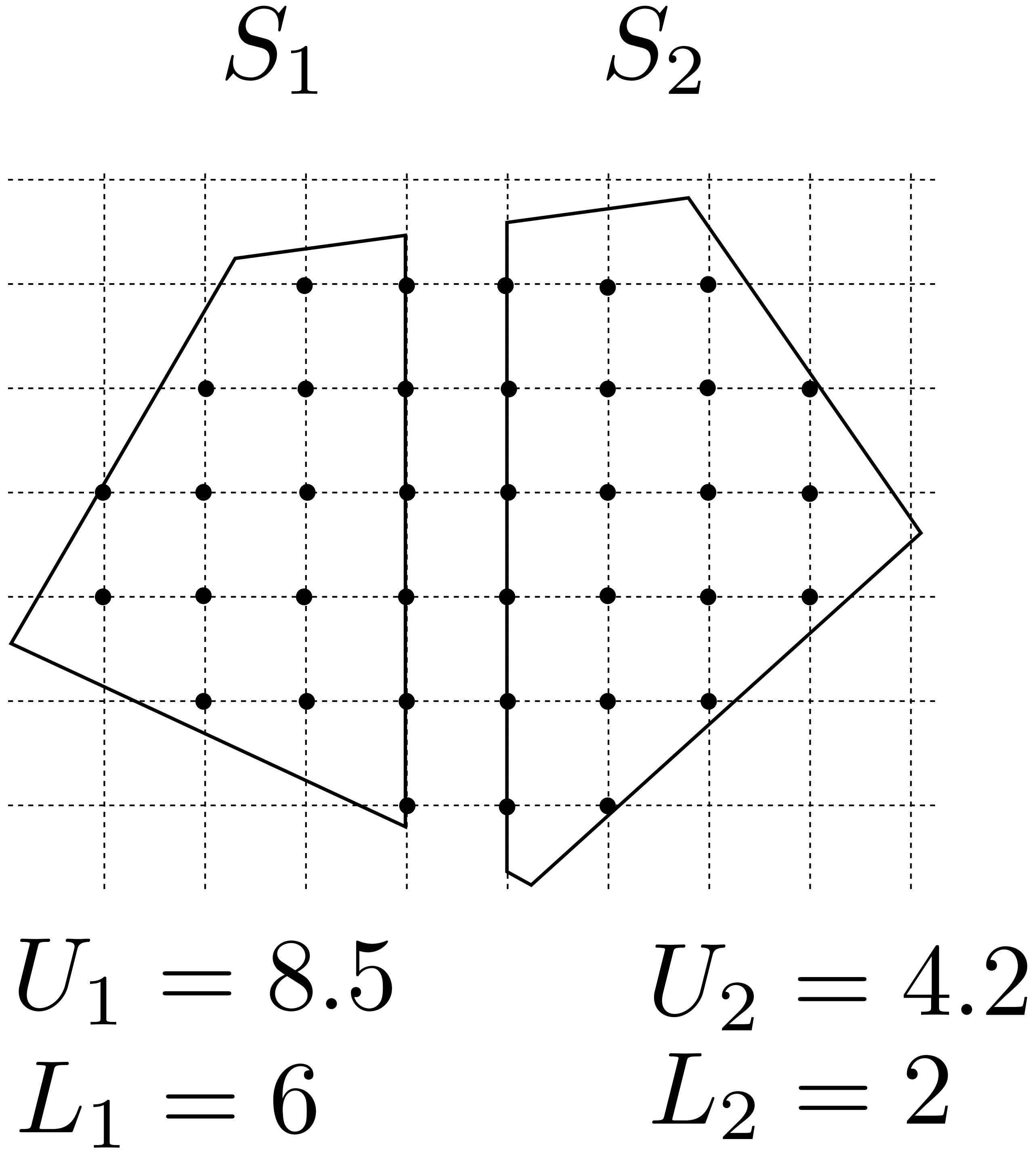
$U_1 = 8.5$
$L_1 = 6$

$U_2 = 4.2$
$L_2 = 2$

Branch and bound

Example in 2D



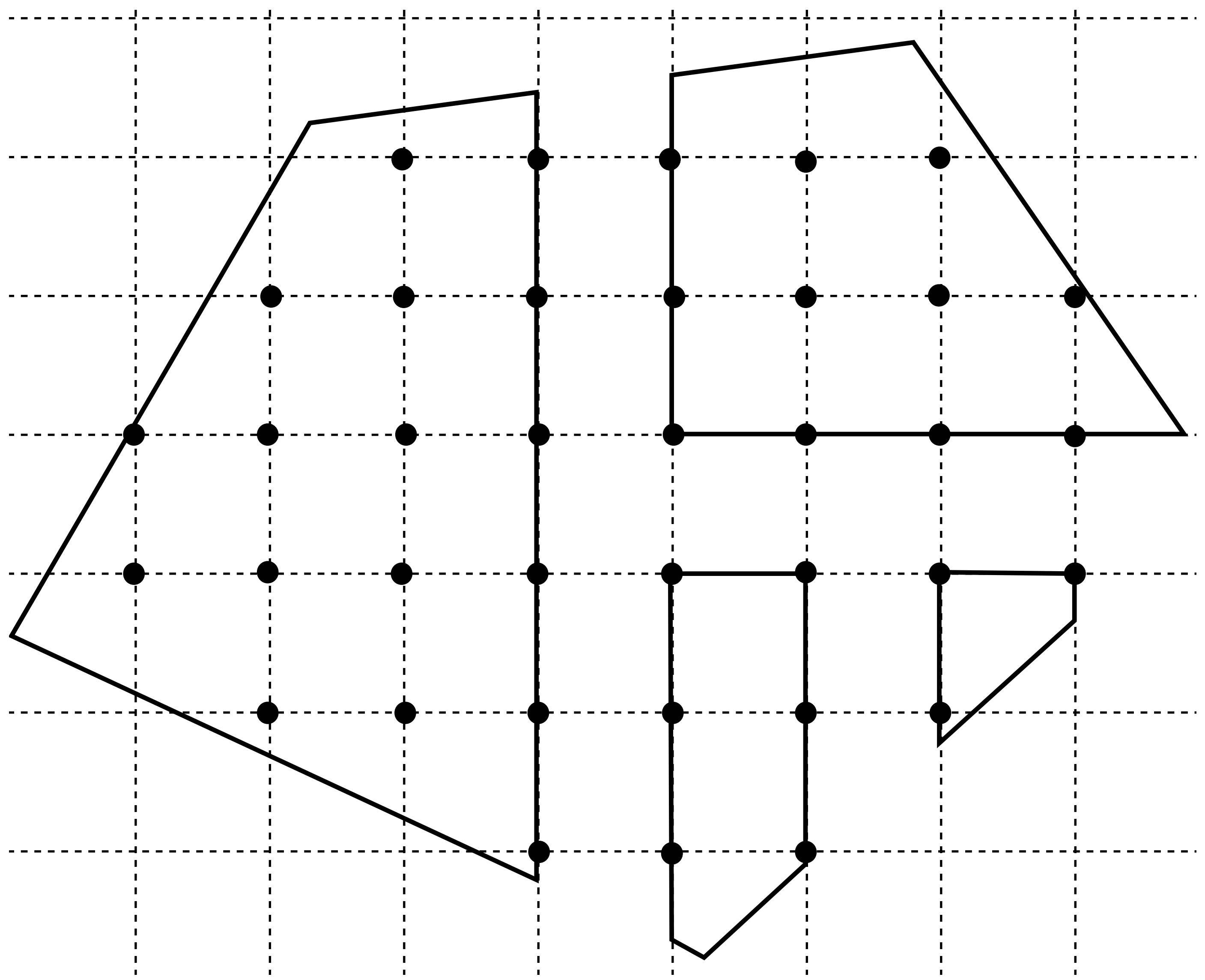


What does this say about global bounds L and U?

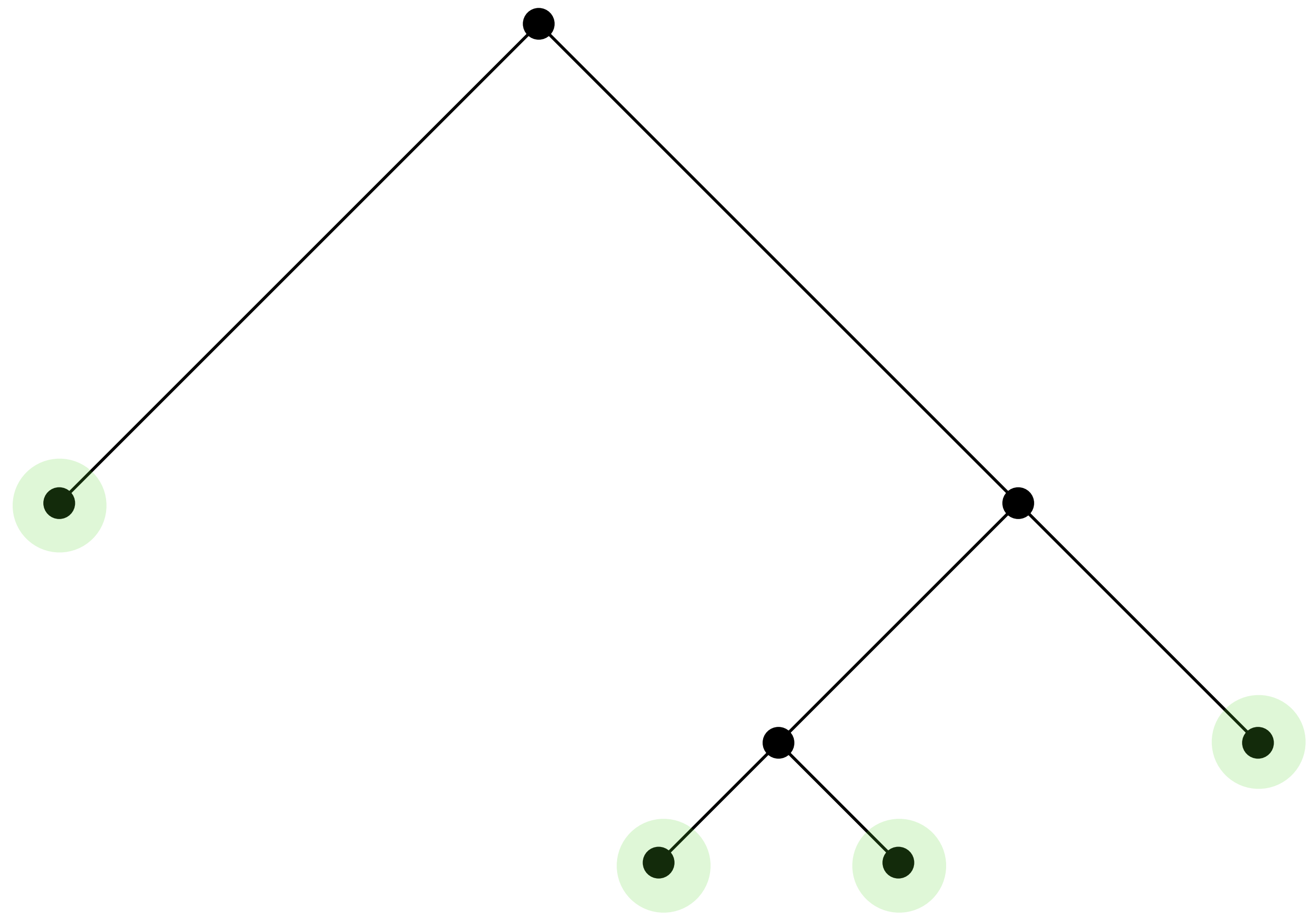
Partition as a binary tree

Example in 2D

Partition

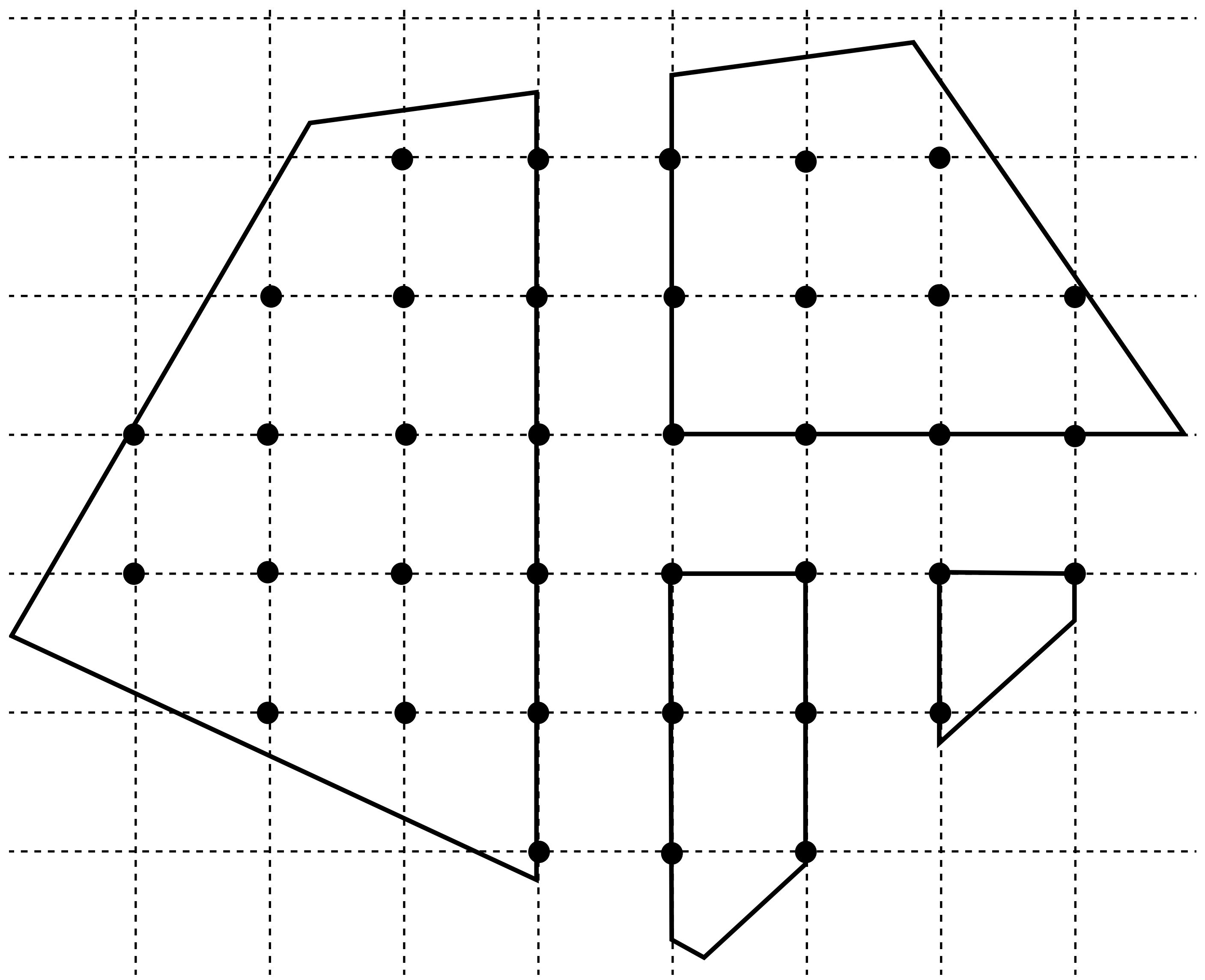


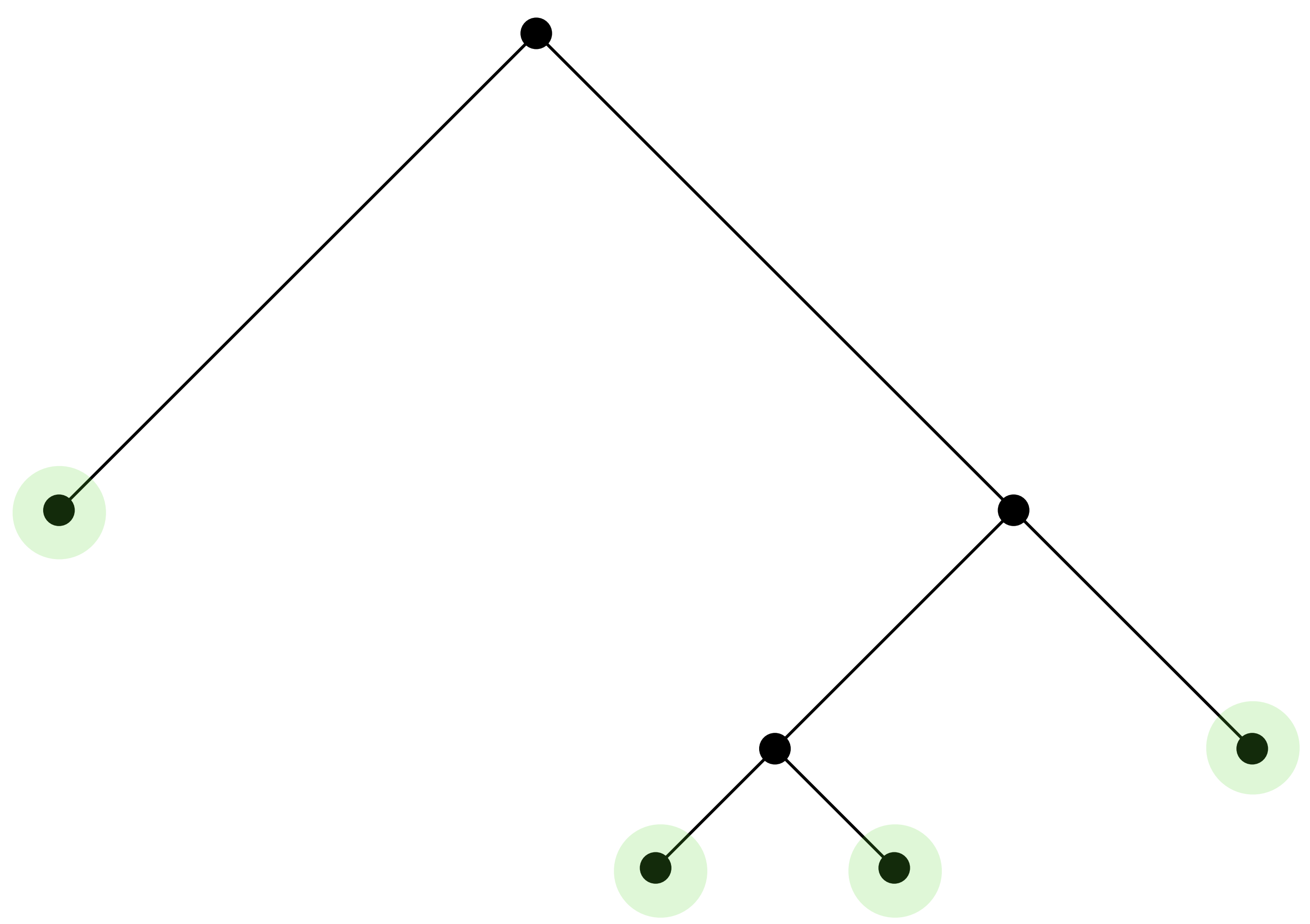
Binary tree



Partition as a binary tree

Example in 2D





At each step we have a **binary tree**

Children correspond to subregions formed by splitting parents

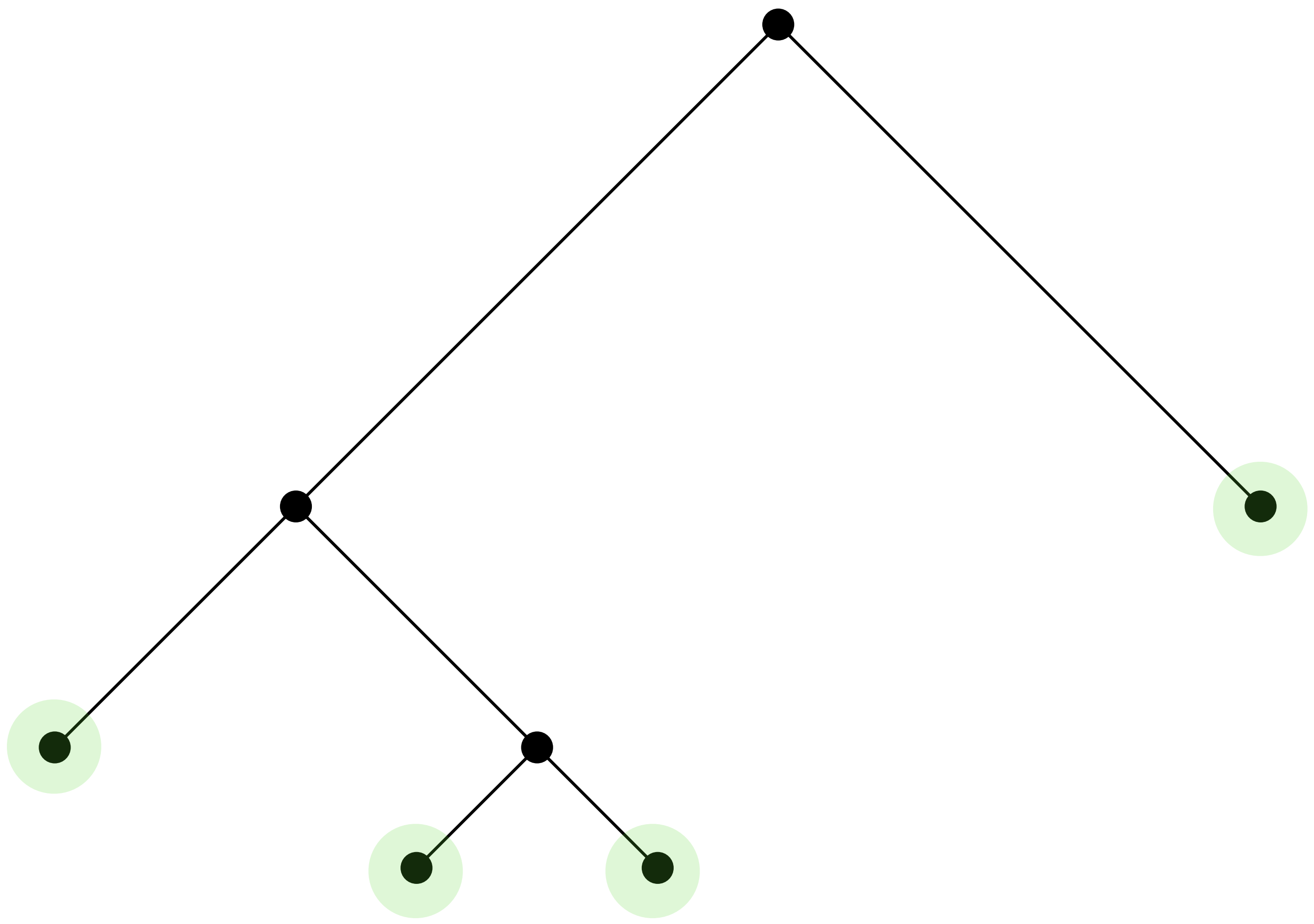
Certifying optimality

certify optimality ⟶ $L \leq c^T x^\star \leq U$ ⟵ return feasible point "incumbent"

Certifying optimality

certify optimality → $L \leq c^T x^\star \leq U$ ← return feasible point "incumbent"

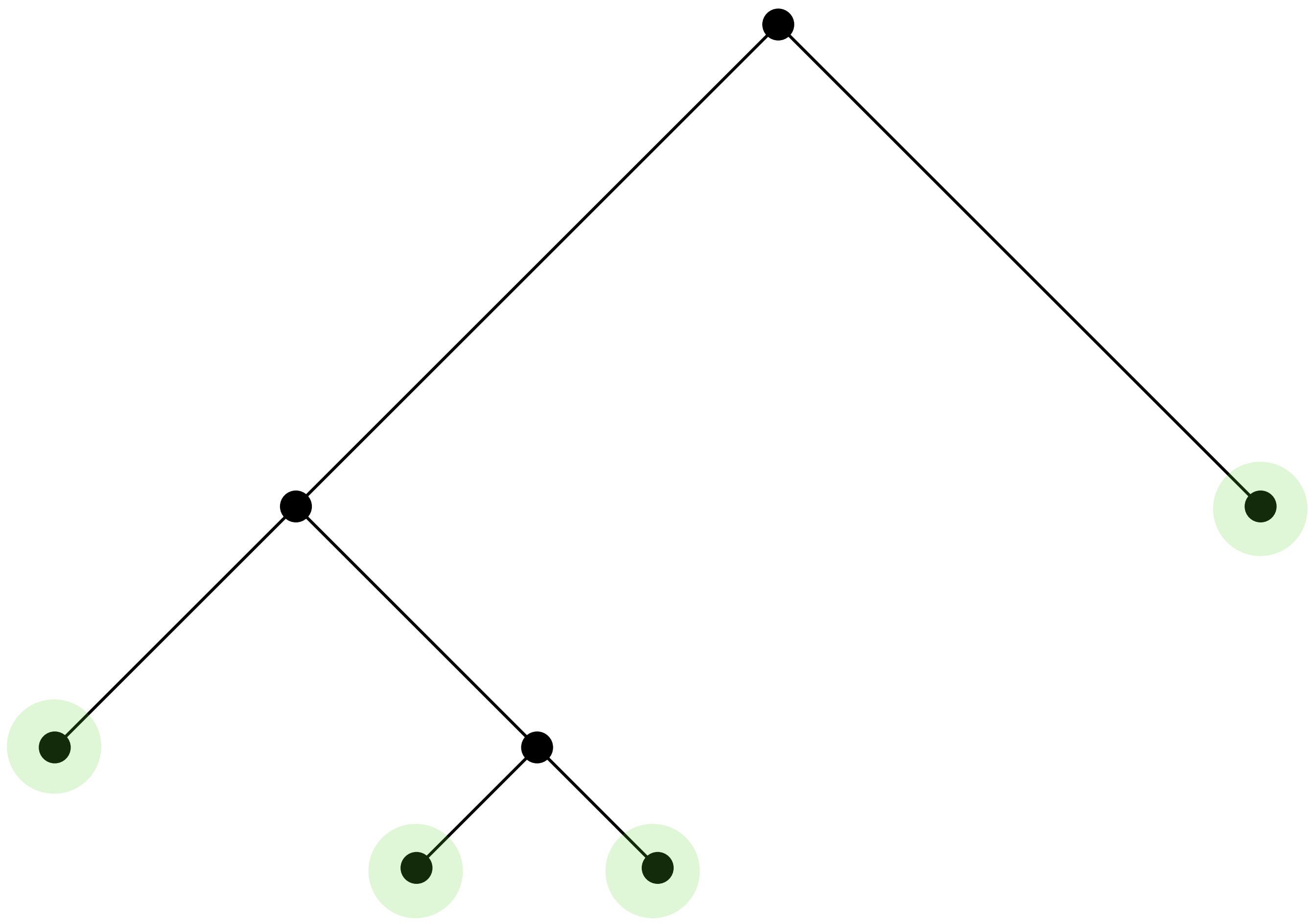
Partition = Leaves



Certifying optimality

certify optimality → $L \leq c^T x^\star \leq U$ ← return feasible point "incumbent"

Partition = Leaves



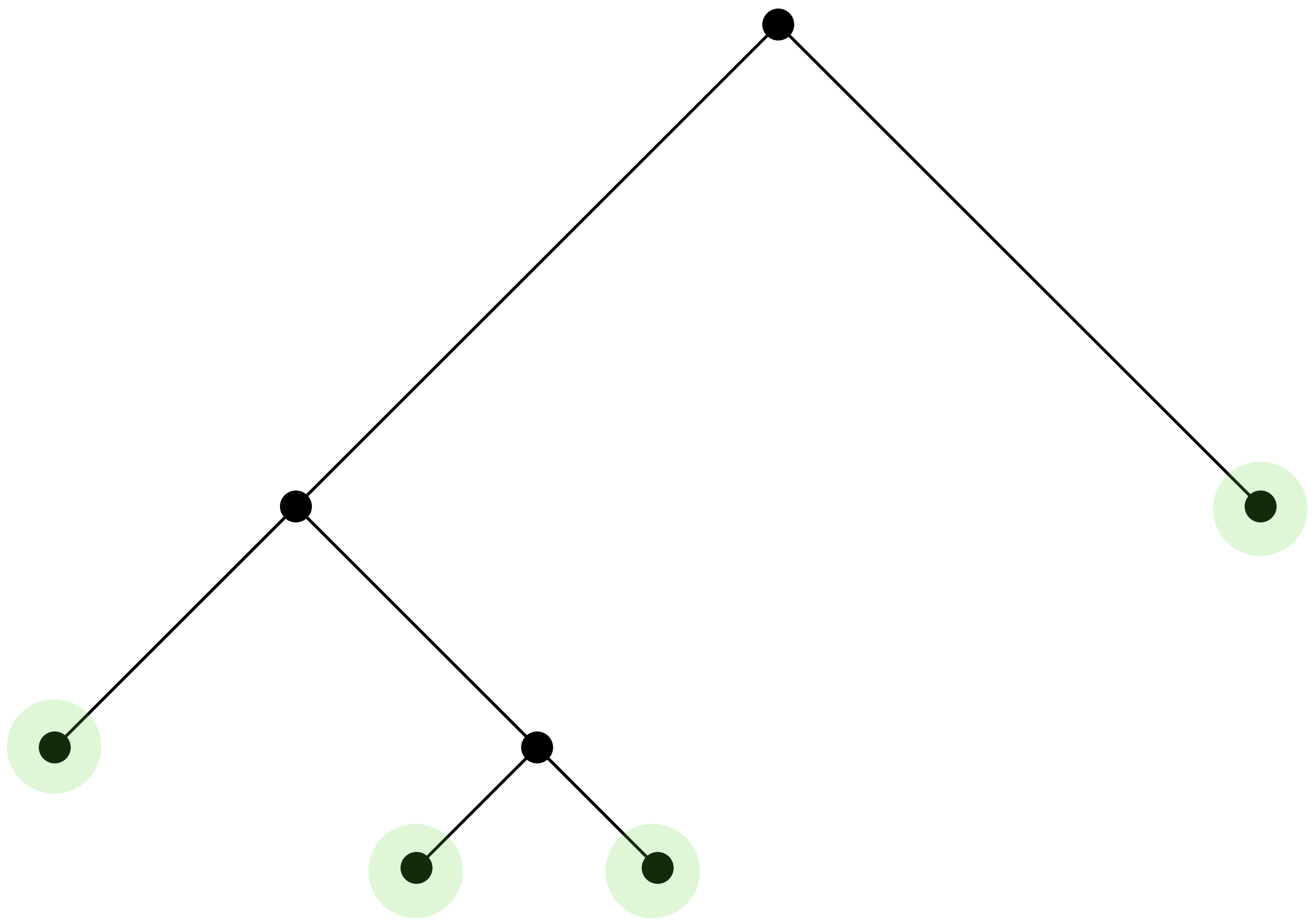
Optimality certificate in integer optimization

- Partition S^j
- Bounds $(L_j, U_j) \quad \forall j$

Certifying optimality

certify optimality → $L \leq c^T x^\star \leq U$ ← return feasible point "incumbent"

Partition = Leaves



Optimality certificate in integer optimization

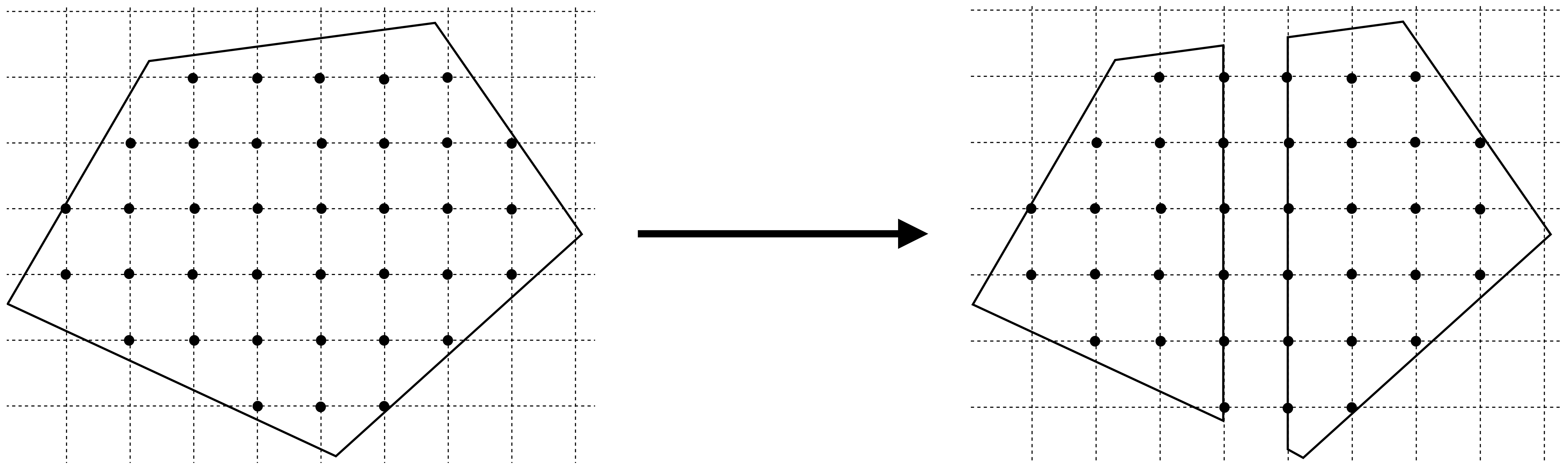
- Partition S^j
- Bounds $(L_j, U_j) \quad \forall j$

Optimality certificate in linear optimization

Dual variables and cost

Branch and bound rules

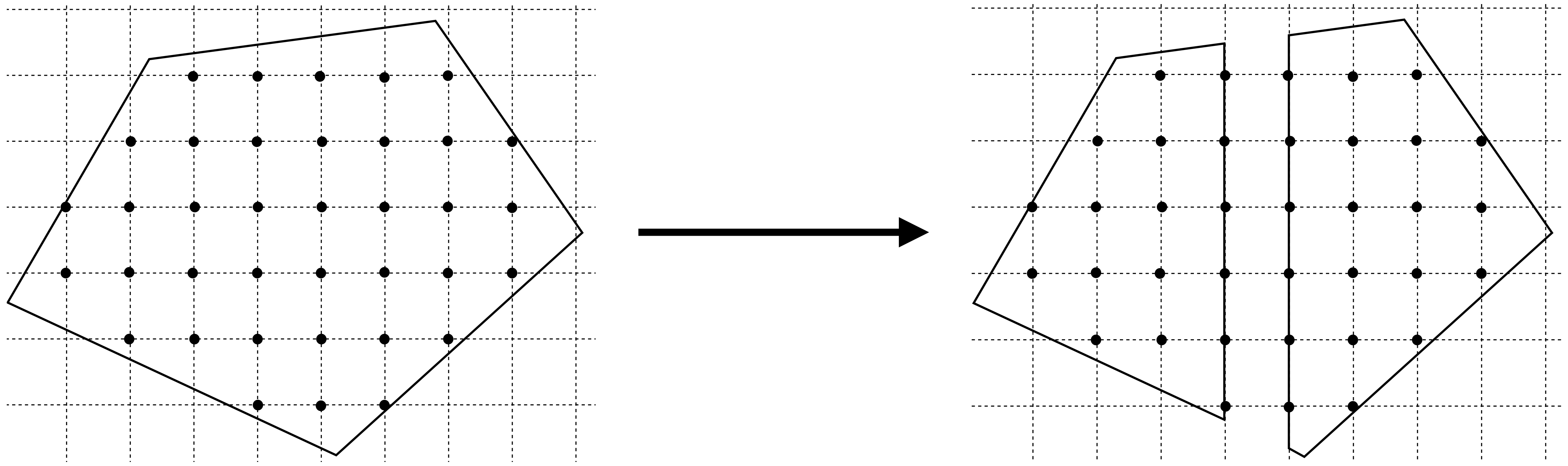
Partitioning



Pick one subproblem and solve its relaxation

$$\bar{x} \leftarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j_{\text{rel}} \end{array}$$

Partitioning



Pick one subproblem and solve its relaxation

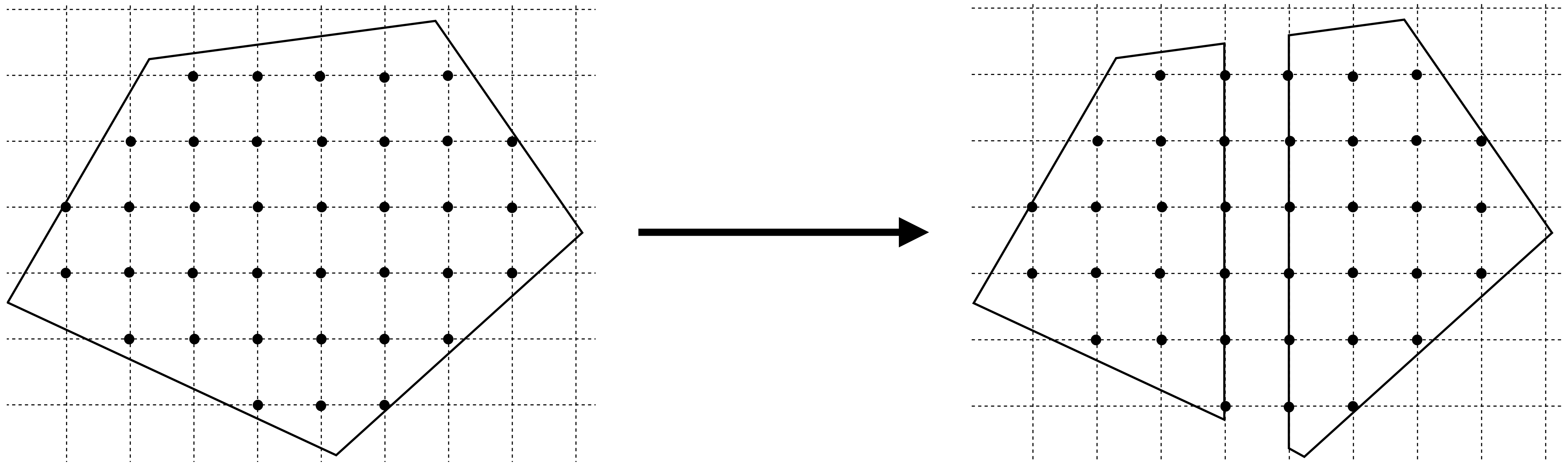
$$\bar{x} \leftarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j_{\text{rel}} \end{array}$$

→

Two possible outcomes

- If $\bar{x}$ integral ($\bar{x} \in S^j$), then $\bar{x}$ is the optimal solution to the subproblem
- If $\bar{x}$ is not integral, then there is an $\bar{x}_i$, $i \in \mathcal{I}$ that is fractional and **we partition**

Partitioning



Pick one subproblem and solve its relaxation

$$\bar{x} \leftarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j_{\text{rel}} \end{array}$$

Two possible outcomes

- If $\bar{x}$ integral ($\bar{x} \in S^j$), then $\bar{x}$ is the optimal solution to the subproblem
- If $\bar{x}$ is not integral, then there is an $\bar{x}_i$, $i \in \mathcal{I}$ that is fractional and **we partition**

Create two subproblems

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j \\ & x_i \leq \lfloor \bar{x}_i \rfloor \end{array} \qquad \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j \\ & x_i \geq \lceil \bar{x}_i \rceil \end{array}$$

Branching rules

Branching decisions

- Which region S^j to split
- Which fractional variable $\bar{x}_i$

Goal

Get tight global bounds as quickly as possible

They can **dramatically affect performance**

Branching rules

Branching decisions

- Which region S^j to split
- Which fractional variable $\bar{x}_i$

Goal

Get tight global bounds as quickly as possible

They can **dramatically affect performance**

Example heuristic (best-bound search)

- **Optimism**: split S^j with lowest L_j
- **Greed**: split most fractional $\bar{x}_i$

3RD COMPONENT

$\bar{x} = (0.1, 3, 0.45)$

Pruning

Key performance component

$$L = \min_j L_j \leq c^T x^\star \leq \min_j U_j = U$$

S^j is **active** if $L_j \leq \min_j U_j$

Otherwise it is **inactive** ($x^\star \notin S^j$)
and we can **prune** it

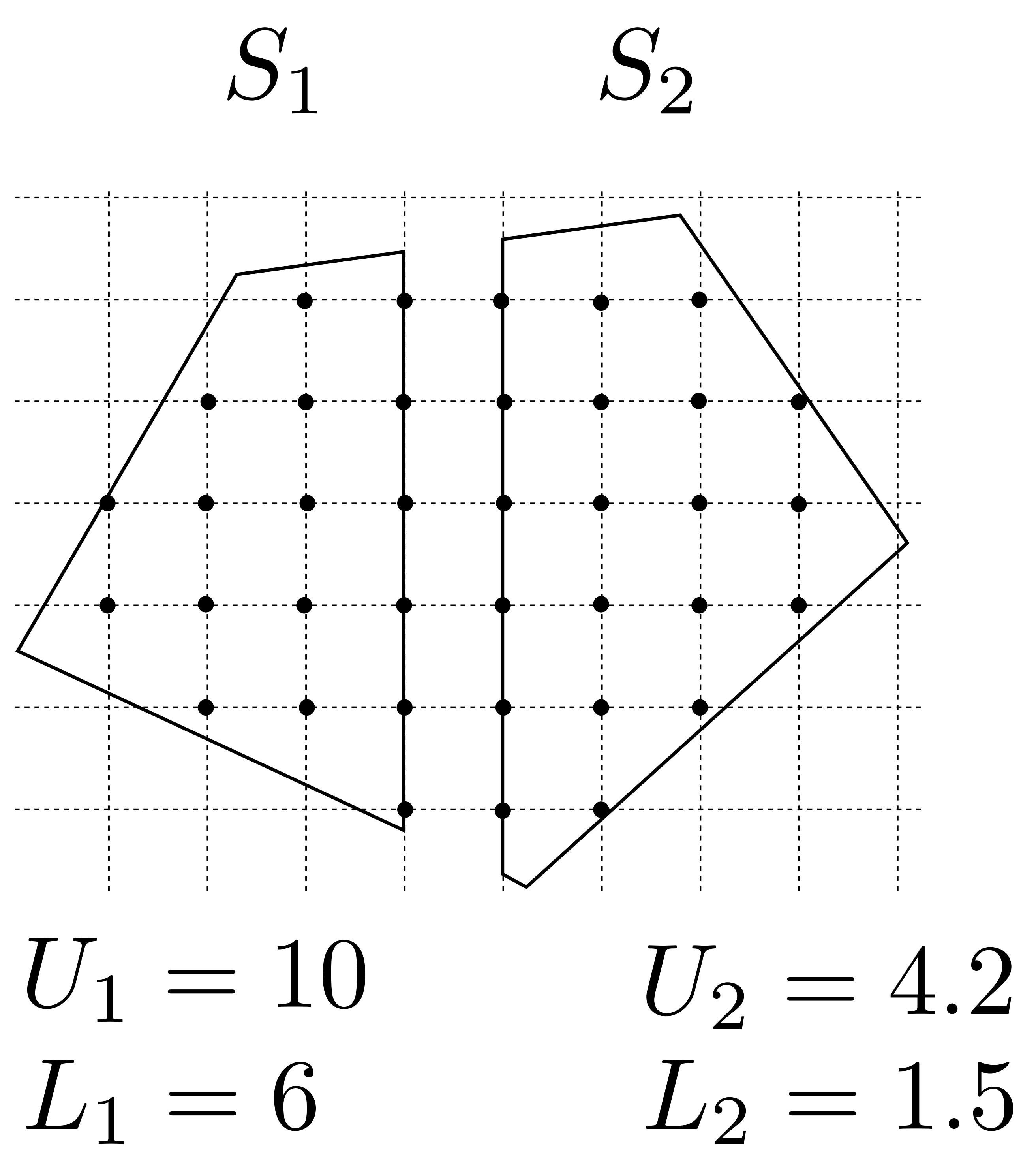
Pruning

Key performance component

$$L = \min_j L_j \leq c^T x^\star \leq \min_j U_j = U$$

S^j is **active** if $L_j \leq \min_j U_j$

Otherwise it is **inactive** ($x^\star \notin S^j$)
and we can **prune** it



Questions

What is S^1? active/inactive

What is S_2? active/inactive

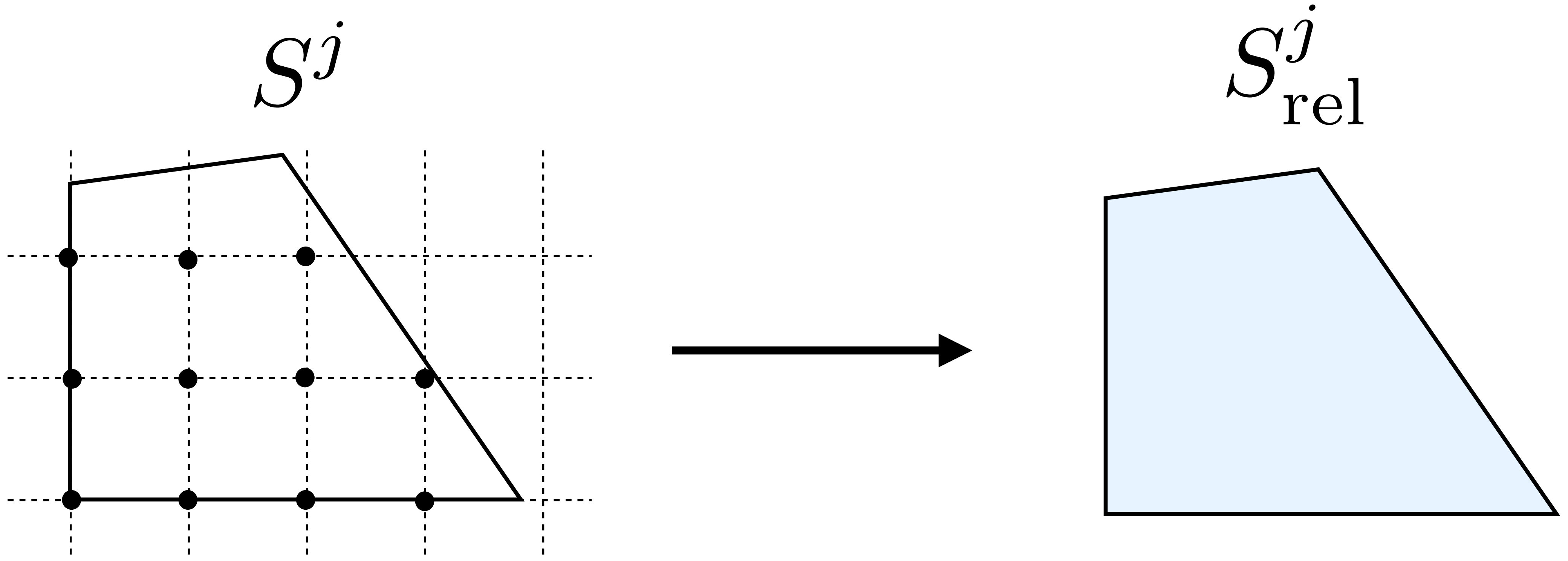
$L = L_2$

$U = U_2$

Bounding rules

Lower bounds. Solve relaxation

$$\bar{x} \leftarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j_{\text{rel}} \end{array}$$



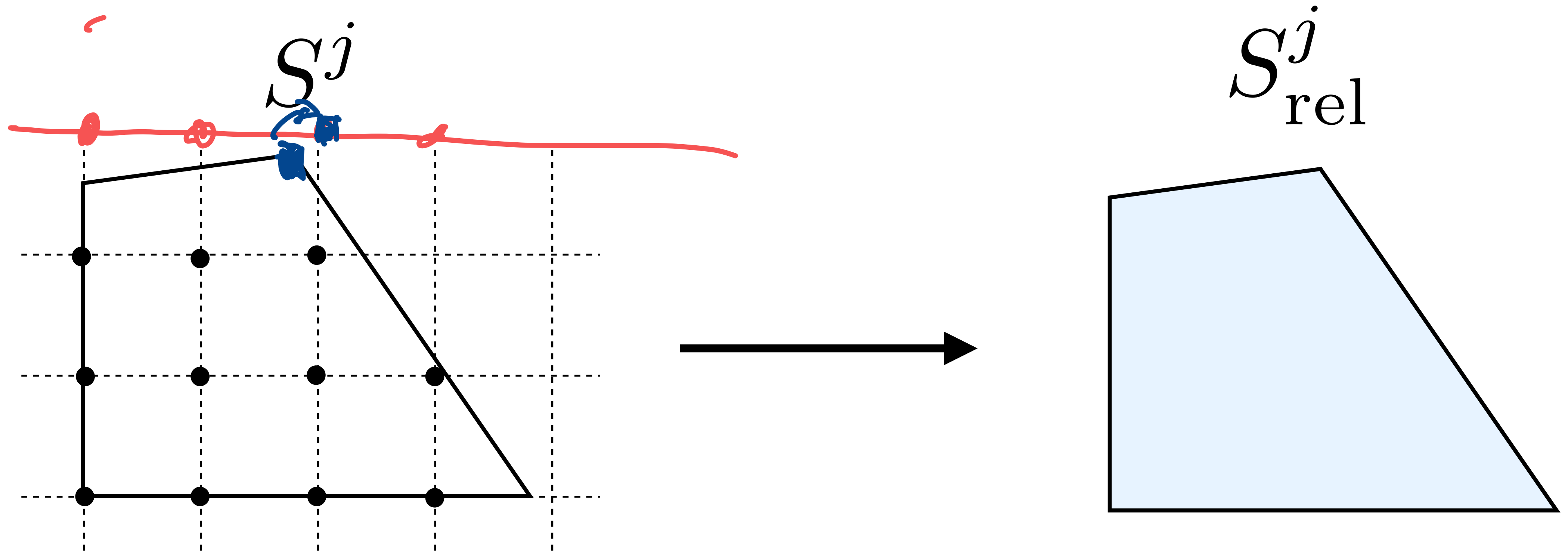
- $L_j = c^T \bar{x}$
- $L_j = \infty$ if infeasible

Bounding rules

Lower bounds. Solve relaxation

$$\bar{x} \leftarrow \begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in S^j_{\text{rel}} \end{array}$$

- $L_j = c^T \bar{x}$
- $L_j = \infty$ if infeasible



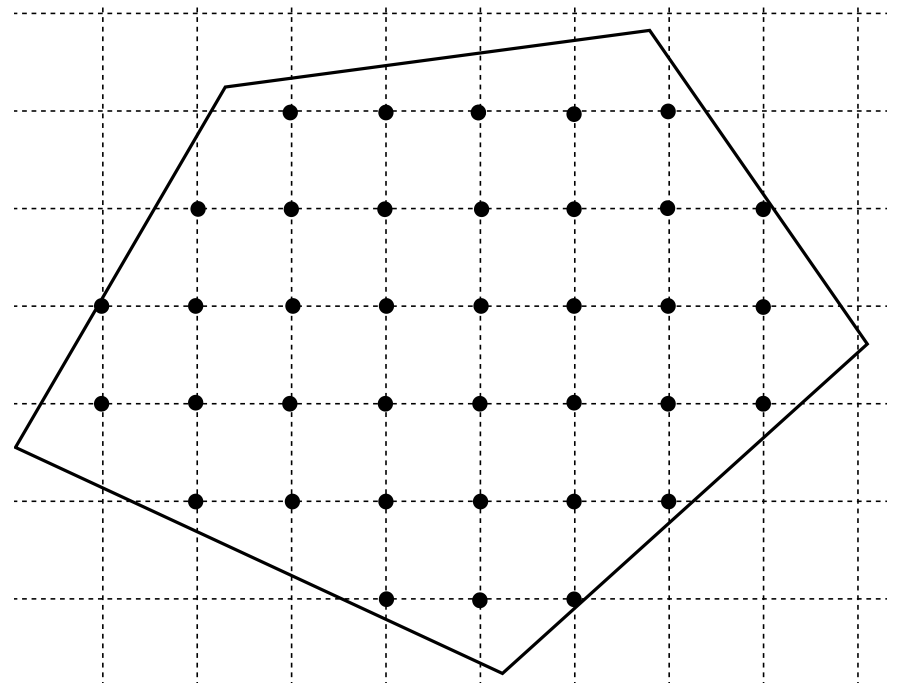
Upper bounds. Try to get feasible point

- $[\bar{x}] \leftarrow$ Round $\bar{x}$ (relaxation solution)
- $U_j = c^T [\bar{x}]$
- $U_j = \infty$ if $[\bar{x}] \notin S^j$ (not feasible)

Branch and bound convergence

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \in \{0,1\}^n \end{array}$$

Example $x \in \mathbf{Z}^n$



Branch and bound

worst-case: we end up partitioning all 2^n points

hope: it works better for our problem

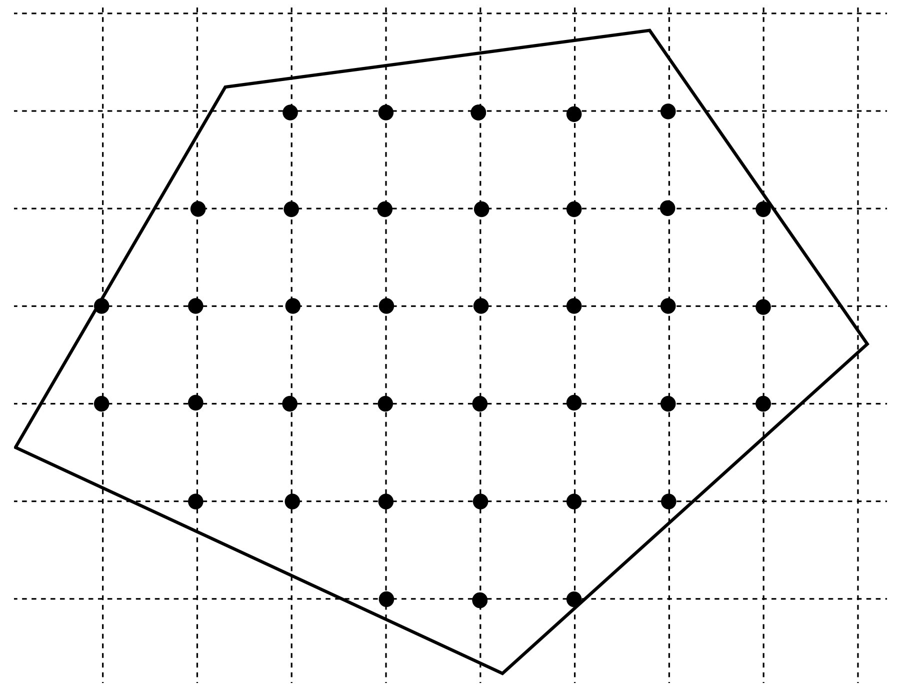
Branch and bound convergence

minimize $c^T x$

subject to $Ax \leq b$

$x \in \{0, 1\}^n$

Example $x \in \mathbf{Z}^d$



Branch and bound

worst-case: we end up partitioning all 2^n points

hope: it works better for our problem

Brute force

Solve problem for all 2^n possible values of $x \in \{0, 1\}^n$

(it blows up for $n \geq 20$)

Practical considerations

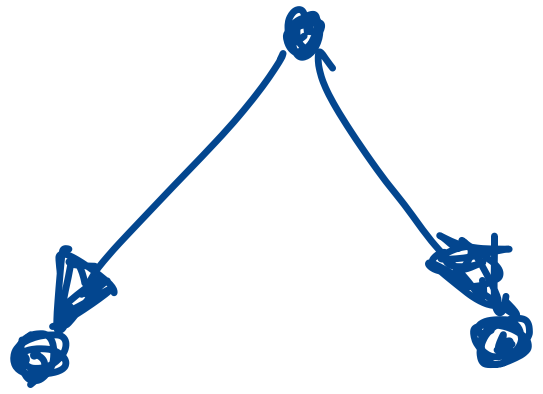
Subproblem solutions are independent

We can solve them in parallel on multiple cores or computing nodes

Practical considerations

Subproblem solutions are independent

We can solve them in parallel on multiple cores or computing nodes



Subproblems can be very similar

(feasible region with added constraints)

We can warm start the subproblem algorithm

Practical considerations

Subproblem solutions are independent

We can solve them in parallel on multiple cores or computing nodes

Subproblems can be very similar

(feasible region with added constraints)

We can warm start the subproblem algorithm

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b \\ & x_i \le \lfloor \bar{x}_i \rfloor \end{aligned}$$

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \le b \\ & x_i \ge \lceil \bar{x}_i \rceil \end{aligned}$$

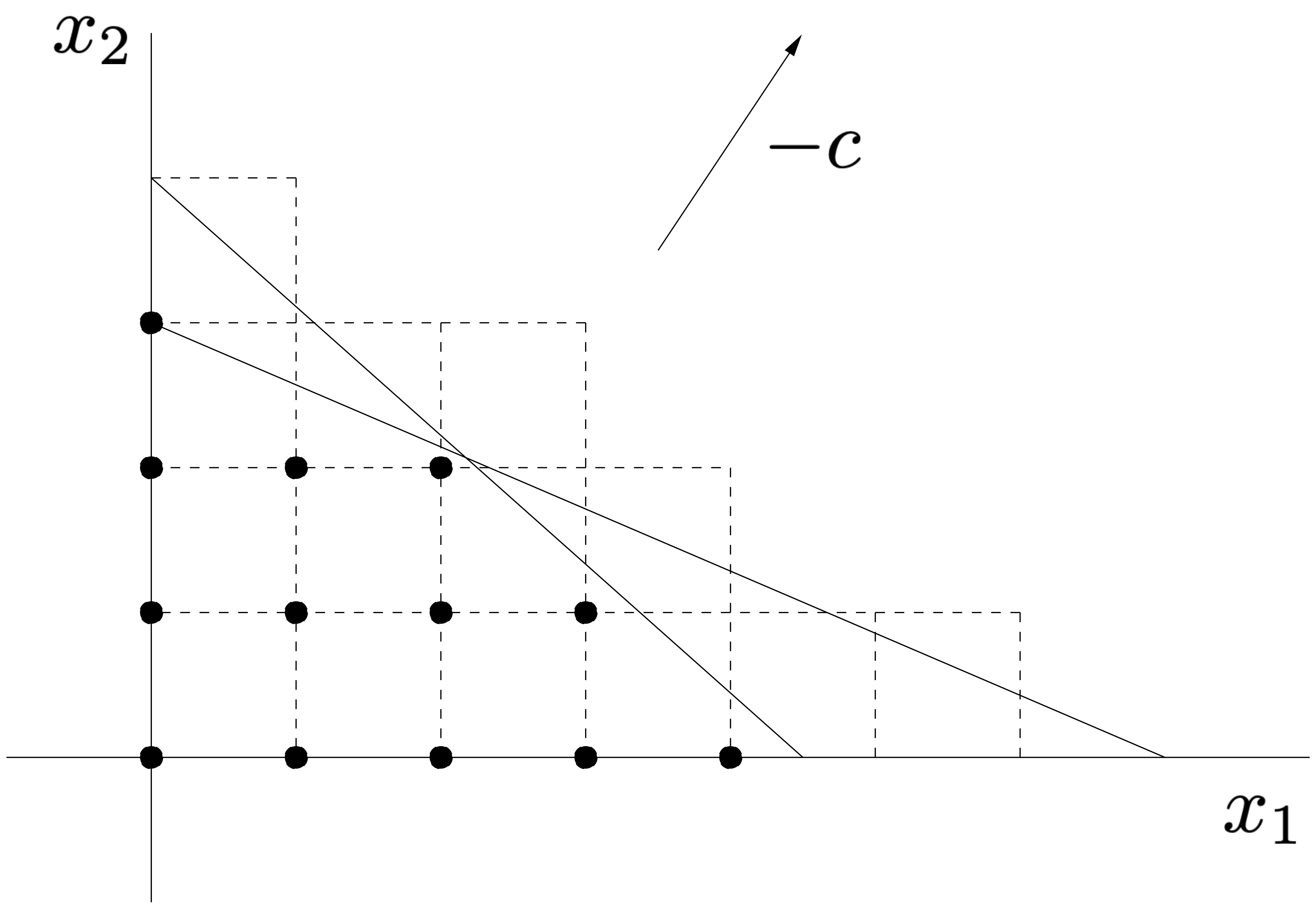
Which algorithm would you use LP subproblems?

DUAL SIMPLEX

Small examples

Branch and bound example

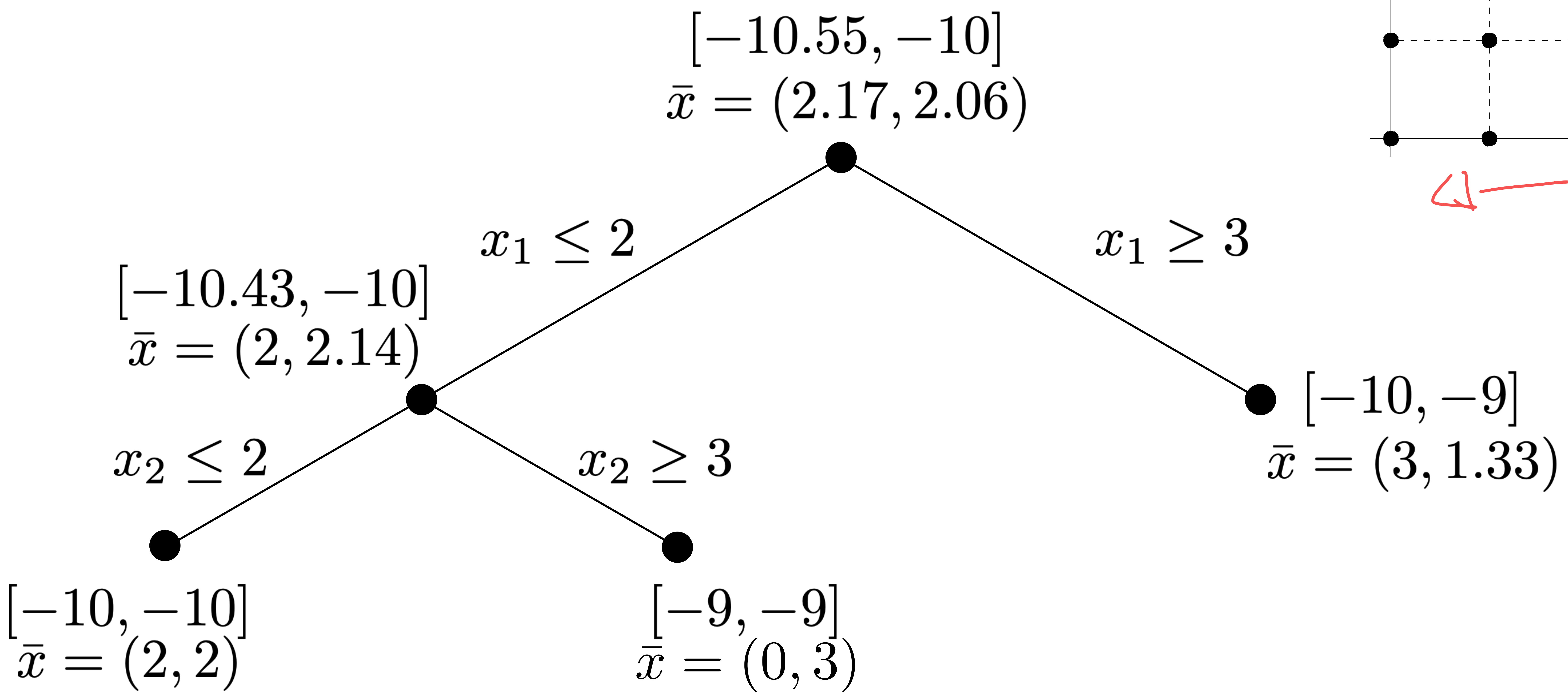
$$\begin{array}{ll} \text{minimize} & -2x_1 - 3x_2 \\ \text{subject to} & (2/9)x_1 + (1/4)x_2 \leq 1 \\ & (1/7)x_1 + (1/3)x_2 \leq 1 \\ & x_1, x_2 \geq 0 \\ & x_1, x_2 \in \mathbf{Z} \end{array}$$

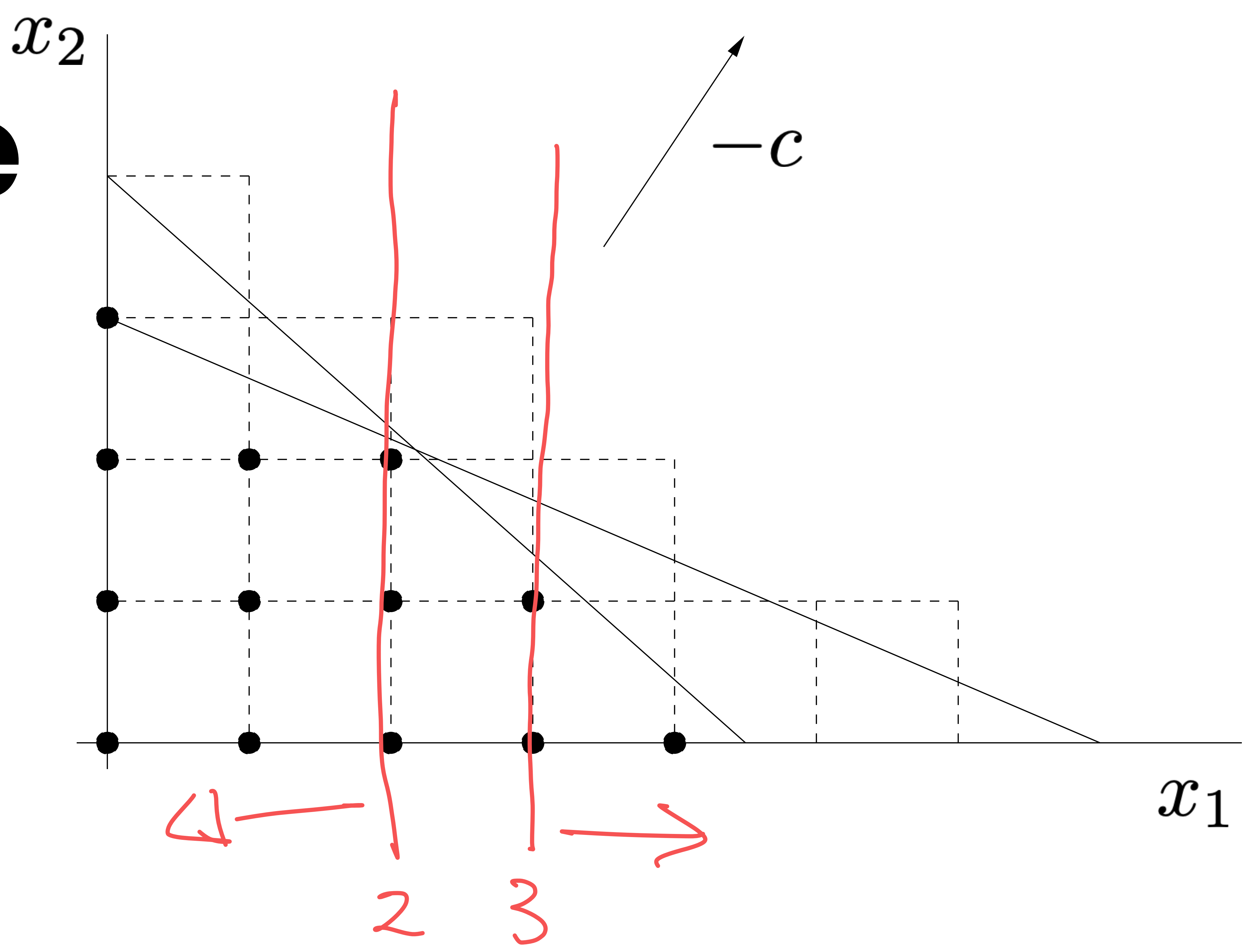


Optimal solution

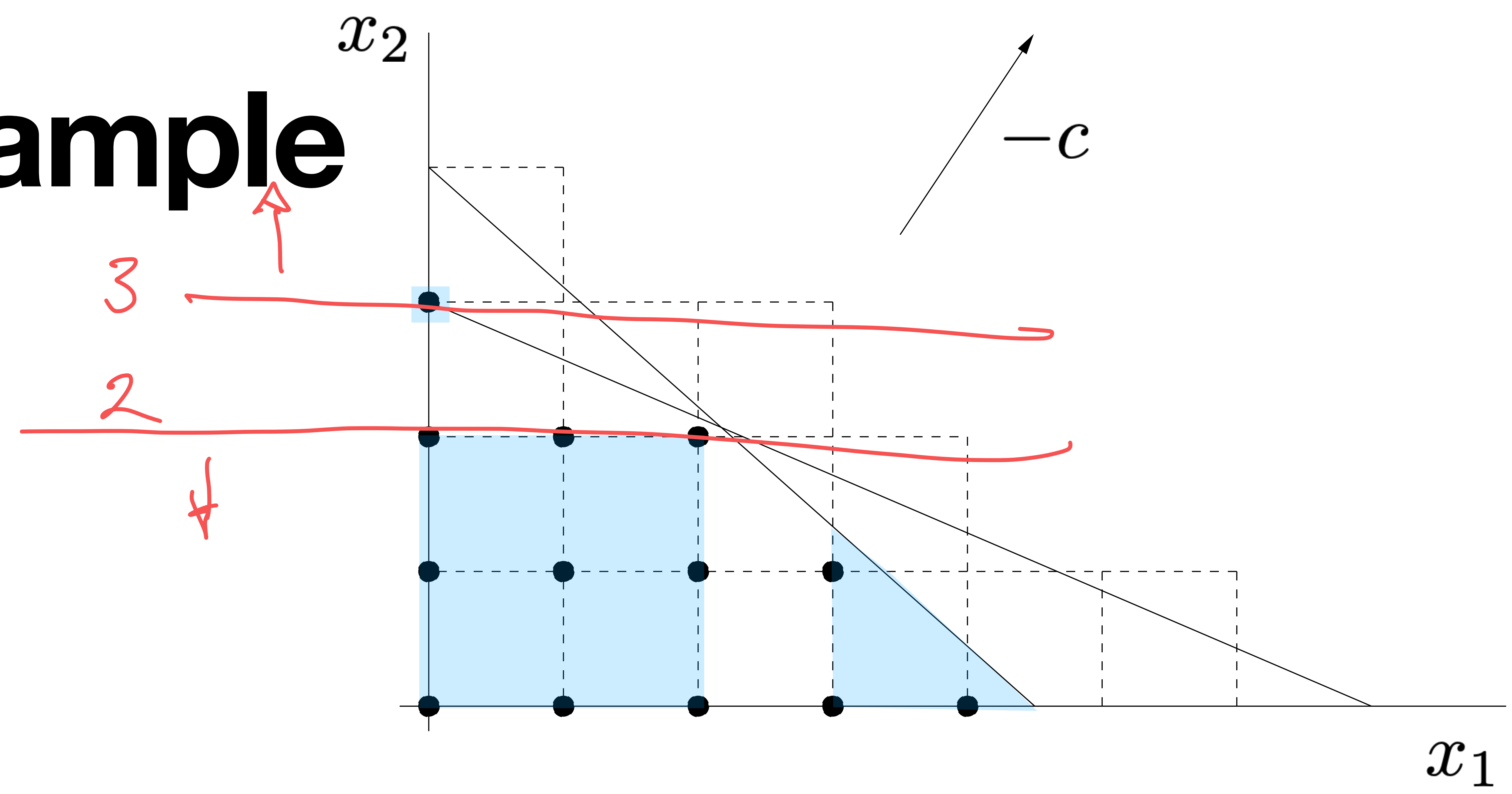
$$x^\star = (2, 2)$$

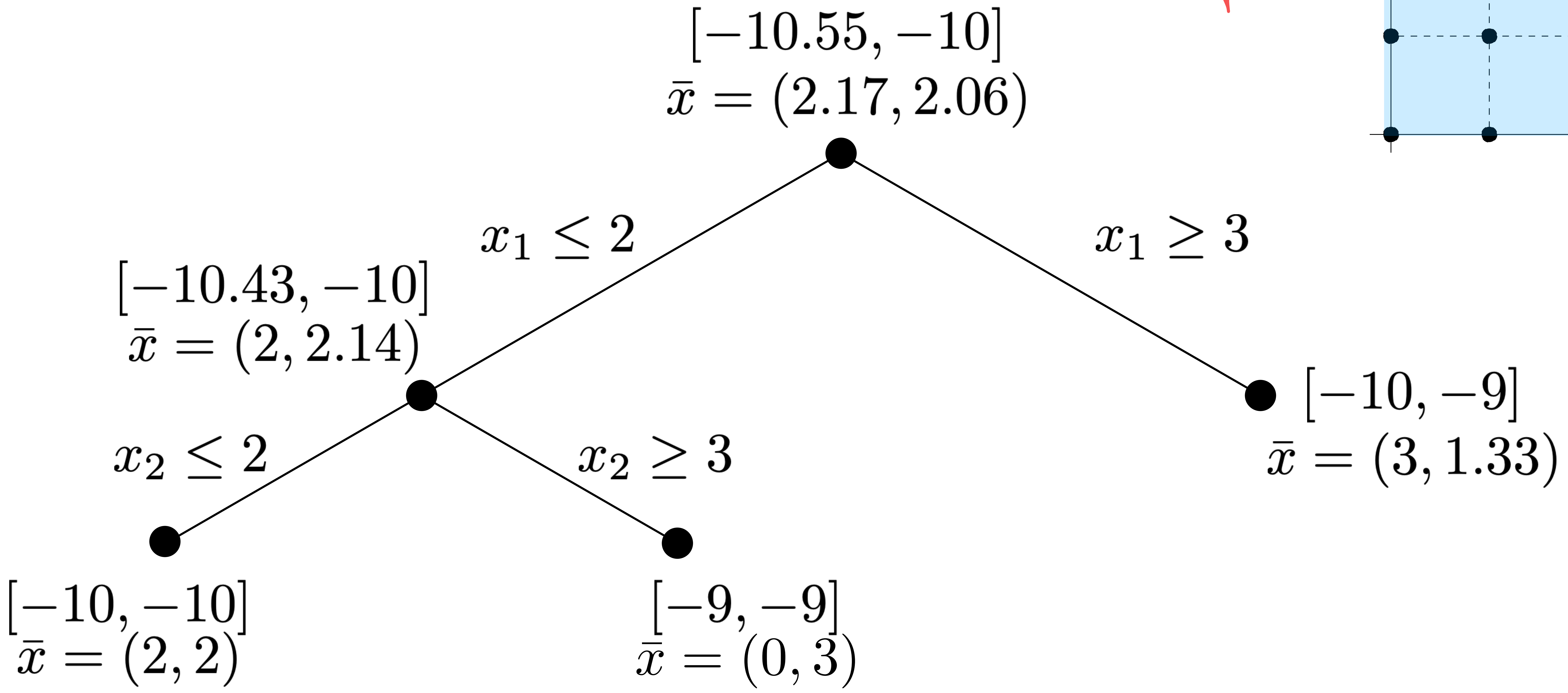
Branch and bound example





Branch and bound example





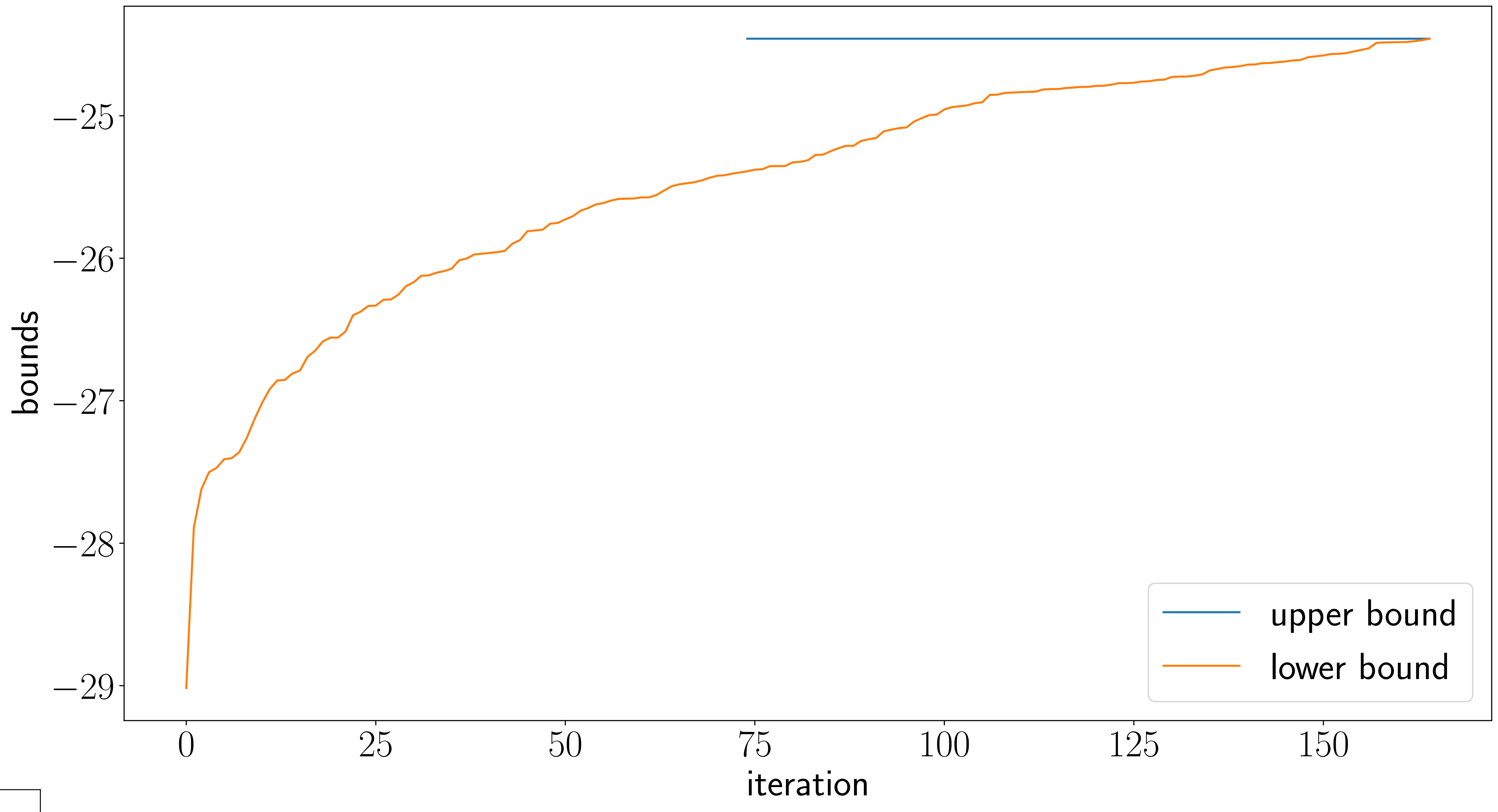
$x \in \{0,1\}$

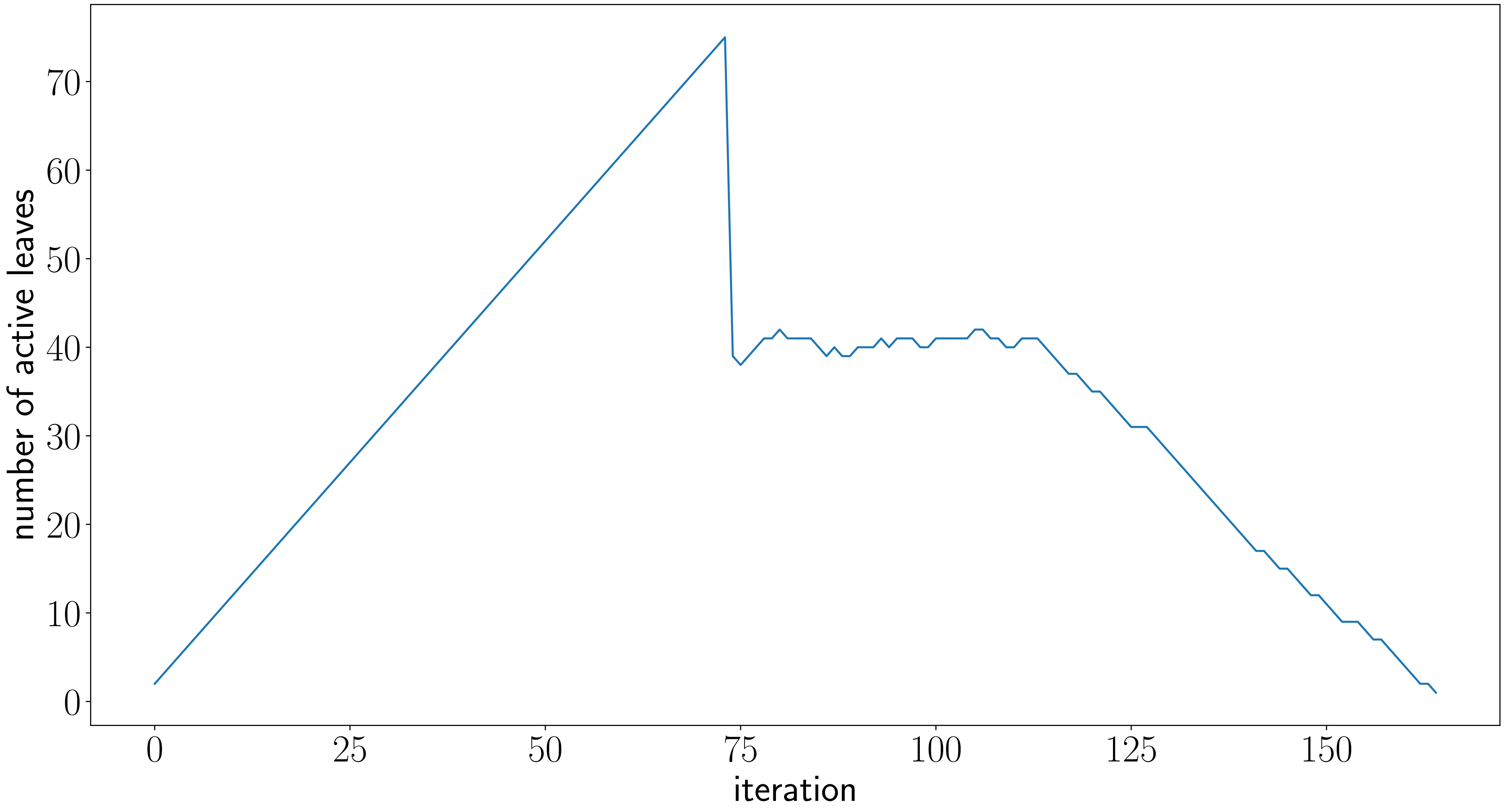
$\hookrightarrow x \in [0,1]$

$x \leq 0 \quad x \geq 1$

A larger example

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \in \mathbf{Z}^n \end{array} \qquad \begin{array}{l} m = 20 \\ n = 10 \end{array}$$





Cardinality minimization

Minimum cardinality example

Find sparsest x satisfying linear inequalities

$$\begin{array}{ll} \text{minimize} & \mathrm{card}(x) \\ \text{subject to} & Ax \leq b \end{array}$$

Minimum cardinality example

Find sparsest x satisfying linear inequalities

$$\begin{array}{ll} \text{minimize} & \mathrm{card}(x) \\ \text{subject to} & Ax \leq b \end{array}$$

Equivalent mixed-boolean LP

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T z \\ \text{subject to} & l_i z_i \leq x_i \leq u_i z_i, \quad i = 1, \ldots, n \\ & Ax \leq b \\ & z \in \{0, 1\}^n \end{array}$$

Big-M formulation

Minimum cardinality example

Find sparsest x satisfying linear inequalities

$$\begin{array}{ll} \text{minimize} & \mathrm{card}(x) \\ \text{subject to} & Ax \le b \end{array}$$

Equivalent mixed-boolean LP

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T z \\ \text{subject to} & l_i z_i \le x_i \le u_i z_i, \quad i = 1, \dots, n \\ & Ax \le b \\ & z \in \{0,1\}^n \end{array}$$

Big-M formulation

- l_i, u_i are lower/upper bounds on x_i
- The tightness of l_i, u_i can greatly influence convergence

Computing big-M constants

l_i is the optimal value of

$$\begin{array}{ll} \text{minimize} & x_i \\ \text{subject to} & Ax \leq b \end{array}$$

u_i is the optimal value of

$$\begin{array}{ll} \text{maximize} & x_i \\ \text{subject to} & Ax \leq b \end{array}$$

Total
$2n$ LPs

Computing big-M constants

l_i is the optimal value of

$$\begin{array}{ll} \text{minimize} & x_i \\ \text{subject to} & Ax \leq b \end{array}$$

u_i is the optimal value of

$$\begin{array}{ll} \text{maximize} & x_i \\ \text{subject to} & Ax \leq b \end{array}$$

Total
$2n$ LPs

Remarks

- If $l_i > 0$ or $u_i < 0$ we can just set $z_i = 1$
  (we cannot have $x_i = 0$)
- This procedure, called "bound tightening", is very common in the pre-processing step of modern solvers

Cardinality problem relaxation

$$
\begin{array}{ll}
\text{minimize} & \mathbf{1}^T z \\
\text{subject to} & l_i z_i \leq x_i \leq u_i z_i, \quad i = 1, \ldots, n \\
& Ax \leq b \\
& 0 \leq z \leq 1
\end{array}
$$

Cardinality problem relaxation

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T z \\ \text{subject to} & l_i z_i \leq x_i \leq u_i z_i, \quad i = 1, \ldots, n \\ & Ax \leq b \\ & 0 \leq z \leq 1 \end{array}$$

If $u_i = -l_i = M$, then

$$-M z_i \leq x_i \leq M z_i \quad \Rightarrow \quad (1/M)|x_i| \leq z_i$$

→

1-norm minimization

$$\begin{array}{ll} \text{minimize} & (1/M)\|x\|_1 \\ \text{subject to} & Ax \leq b \end{array}$$

Cardinality problem relaxation

$$
\begin{array}{ll}
\text{minimize} & \mathbf{1}^T z \\
\text{subject to} & l_i z_i \leq x_i \leq u_i z_i, \quad i = 1, \dots, n \\
& Ax \leq b \\
& 0 \leq z \leq 1
\end{array}
$$

If $u_i = -l_i = M$, then

$$-M z_i \leq x_i \leq M z_i \quad \Rightarrow \quad (1/M)|x_i| \leq z_i$$

→

1-norm minimization

$$
\begin{array}{ll}
\text{minimize} & (1/M)\|x\|_1 \\
\text{subject to} & Ax \leq b
\end{array}
$$

Relaxation is fancier version of 1-norm minimization
(induces sparsity)

Implementation details

Upper bound $\mathrm{card}(\bar{x})$ with $\bar{x}$ from the relaxation (1-norm induces sparsity)

Implementation details

Upper bound $\mathrm{card}(\bar{x})$ with $\bar{x}$ from the relaxation (1-norm induces sparsity)

Lower bound we can replace L with $\lceil L \rceil$ since card is integer valued

Implementation details

Upper bound $\mathrm{card}(\bar{x})$ with $\bar{x}$ from the relaxation (1-norm induces sparsity)

Lower bound we can replace L with $\lceil L \rceil$ since card is integer valued

Best-bound search split node with lowest L

Implementation details

Upper bound $\mathrm{card}(\bar{x})$ with $\bar{x}$ from the relaxation (1-norm induces sparsity)

Lower bound we can replace L with $\lceil L \rceil$ since card is integer valued

Best-bound search split node with lowest L

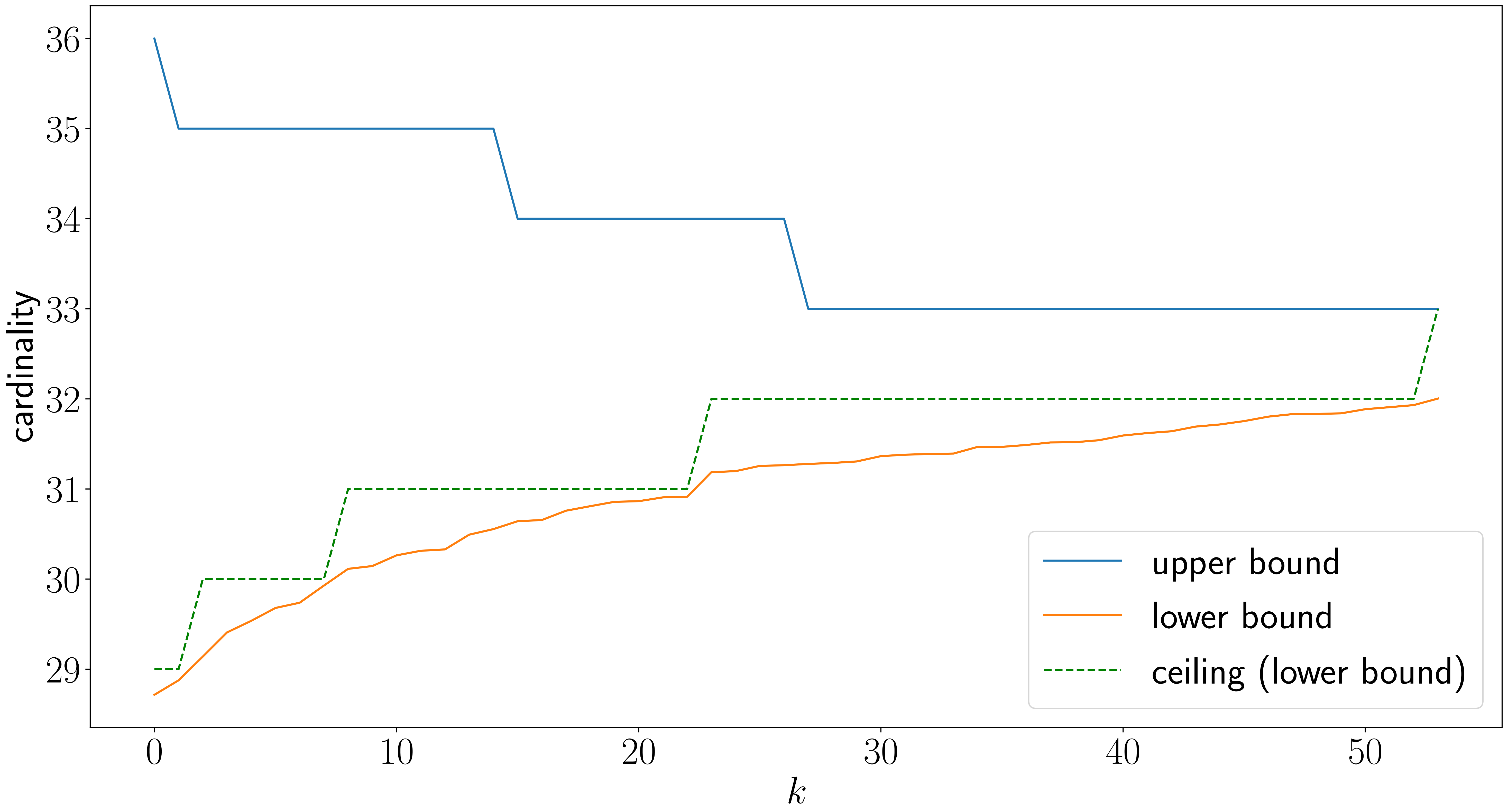
Most ambivalent variable the closest z_j to $1/2$

Small example

Data

40 variables, 200 constraints

$2^{40} \approx 1$ trillion combinations



Results

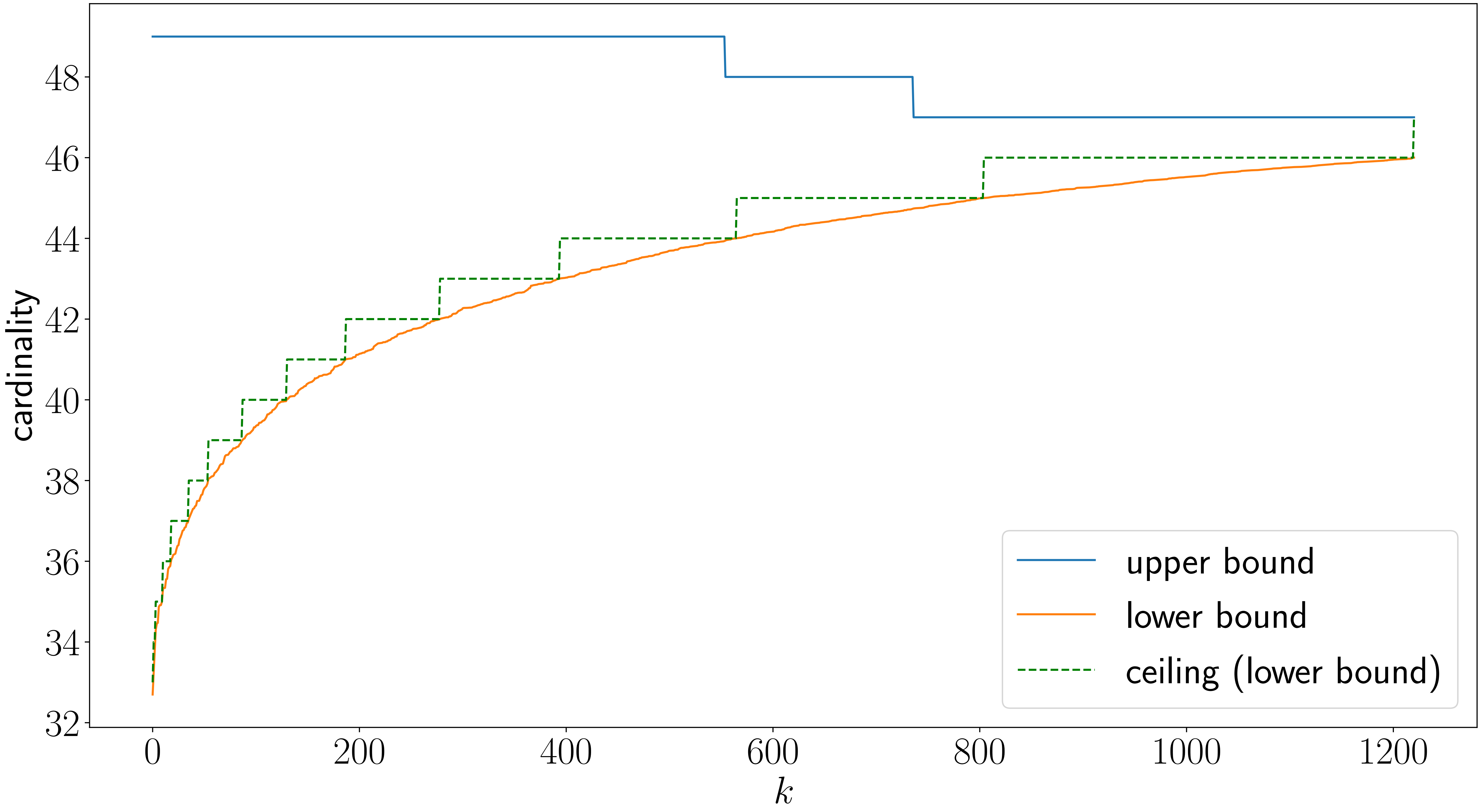
- Finds good solution very quikcly
- Weighted 1-norm heuristic works very well
- Terminates in 54 iterations

Medium example

Data

60 variables, 200 constraints

$2^{60} \approx 1.15 \cdot 10^{18}$ combinations



Results

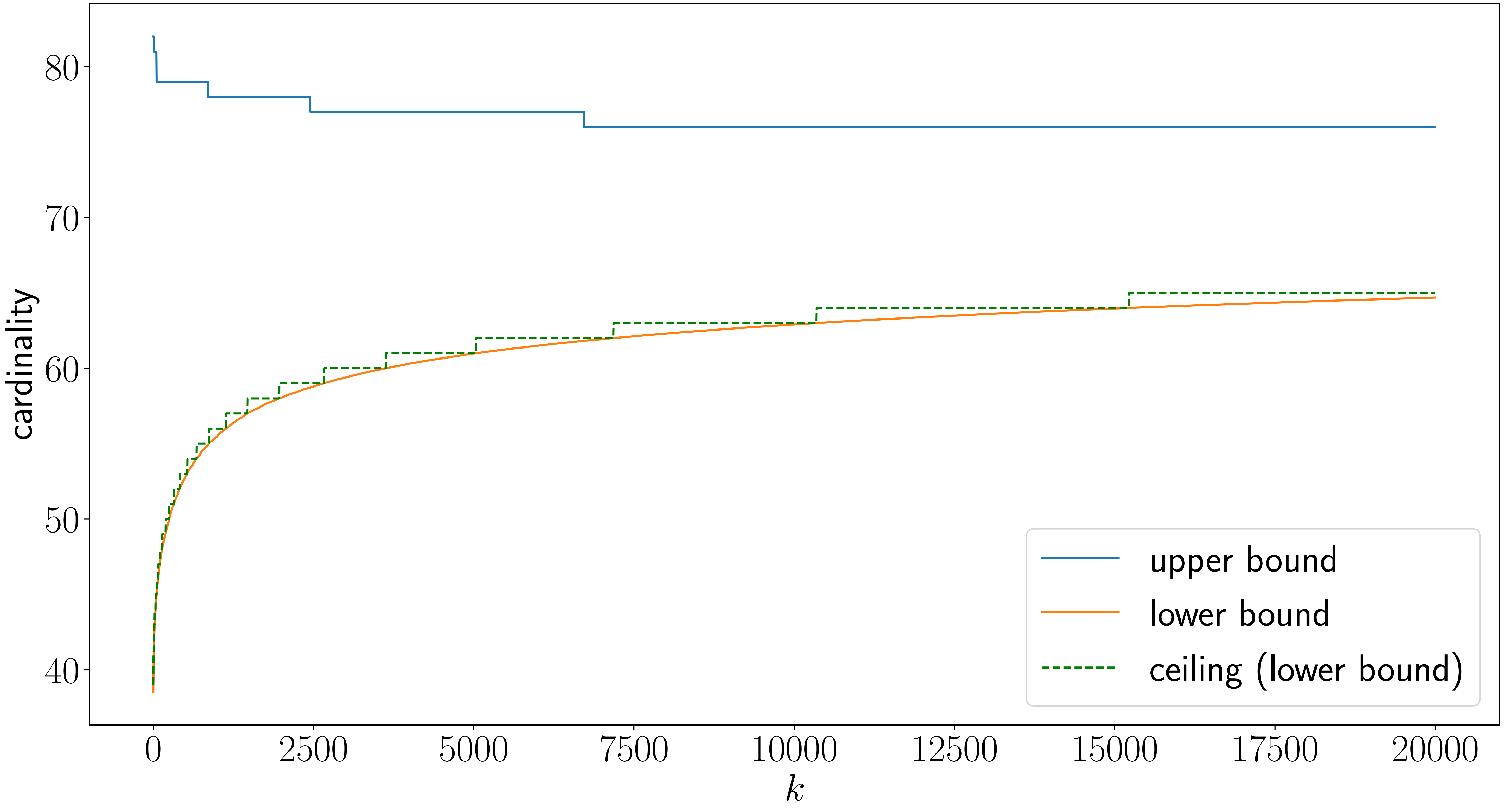
- Finds good solution very quikcly
- Weighted 1-norm heuristic works very well
- Terminates in ≈ 1200 iterations

Larger example

Data

100 variables, 300 constraints

$2^{100} \approx 1.26 \cdot 10^{30}$ combinations



Results

- Finds good solution very quikcly
- 6 hours run, no termination
- Only optimality gap $U - L$ in the end

Larger example with commercial solver

Data

100 variables, 300 constraints

$2^{100} \approx 1.26 \cdot 10^{30}$ combinations

Results

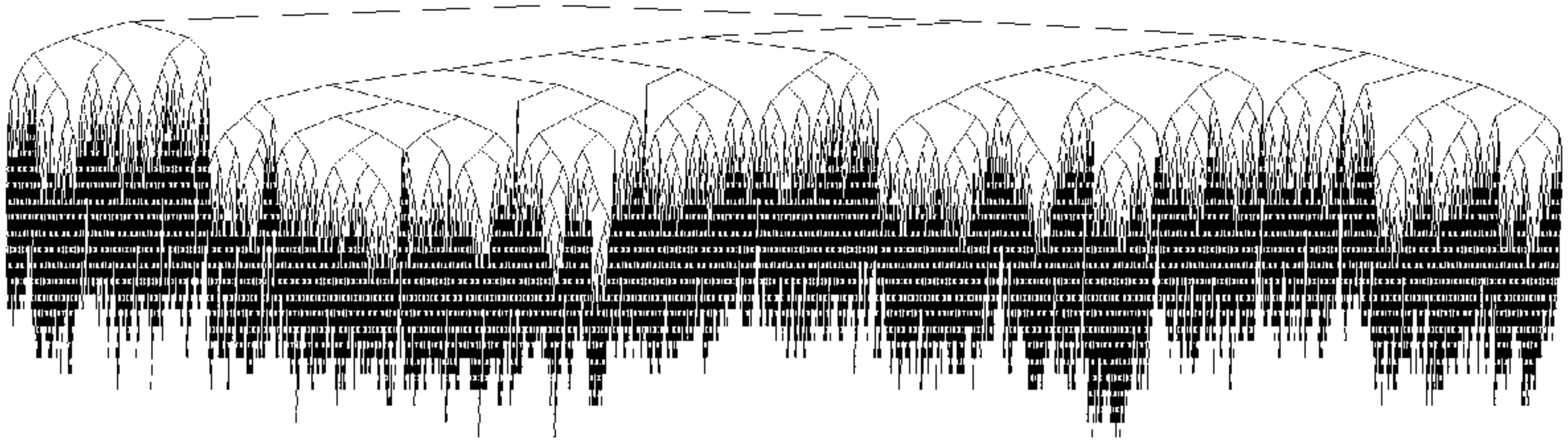
- Optimal cardinality 72
- Much more sophisticated method
- 1888 seconds (31 minutes) run (very slow!)

Gurobi output

```
Gurobi Optimizer version 9.0.3 build v9.0.3rc0 (mac64)
Optimize a model with 500 rows, 200 columns and 30400 nonzeros
Variable types: 100 continuous, 100 integer (100 binary)
Coefficient statistics:
  Matrix range     [4e-05, 5e+00]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [4e-03, 3e+01]
Presolve time: 0.05s
Presolved: 500 rows, 200 columns, 30400 nonzeros
Variable types: 100 continuous, 100 integer (100 binary)

Root relaxation: objective 2.933185e+01, 735 iterations, 0.18 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   29.33185    0   85          -   29.33185      -     -    0s
H    0     0                      85.0000000   29.33185  65.5%     -    0s
     0     0   30.18570    0   83   85.00000   30.18570  64.5%     -    1s
H    0     0                      83.0000000   30.18570  63.6%     -    1s
     0     0   31.35255    0   86   83.00000   31.35255  62.2%     -    2s
     0     2   31.81240    0   86   83.00000   31.81240  61.7%     -    3s
H  271    73                      82.0000000   35.05009  57.3%  58.6    4s
   376   104   47.90892   36   47   82.00000   35.05009  57.3%  54.1    5s
                                        …
                                        …
 2887987 13108     cutoff   88        72.00000   70.70801  1.79%  34.1 1880s
 2897345  4880     cutoff   87        72.00000   70.86531  1.58%  34.1 1885s

Explored 2903463 nodes (98760290 simplex iterations) in 1888.42 seconds
Thread count was 16 (of 16 available processors)

Optimal solution found (tolerance 1.00e-04)
Best objective 7.200000000000e+01, best bound 7.200000000000e+01, gap 0.0000%
```

Tree size can grow dramatically

Example for 360 seconds on CPU…

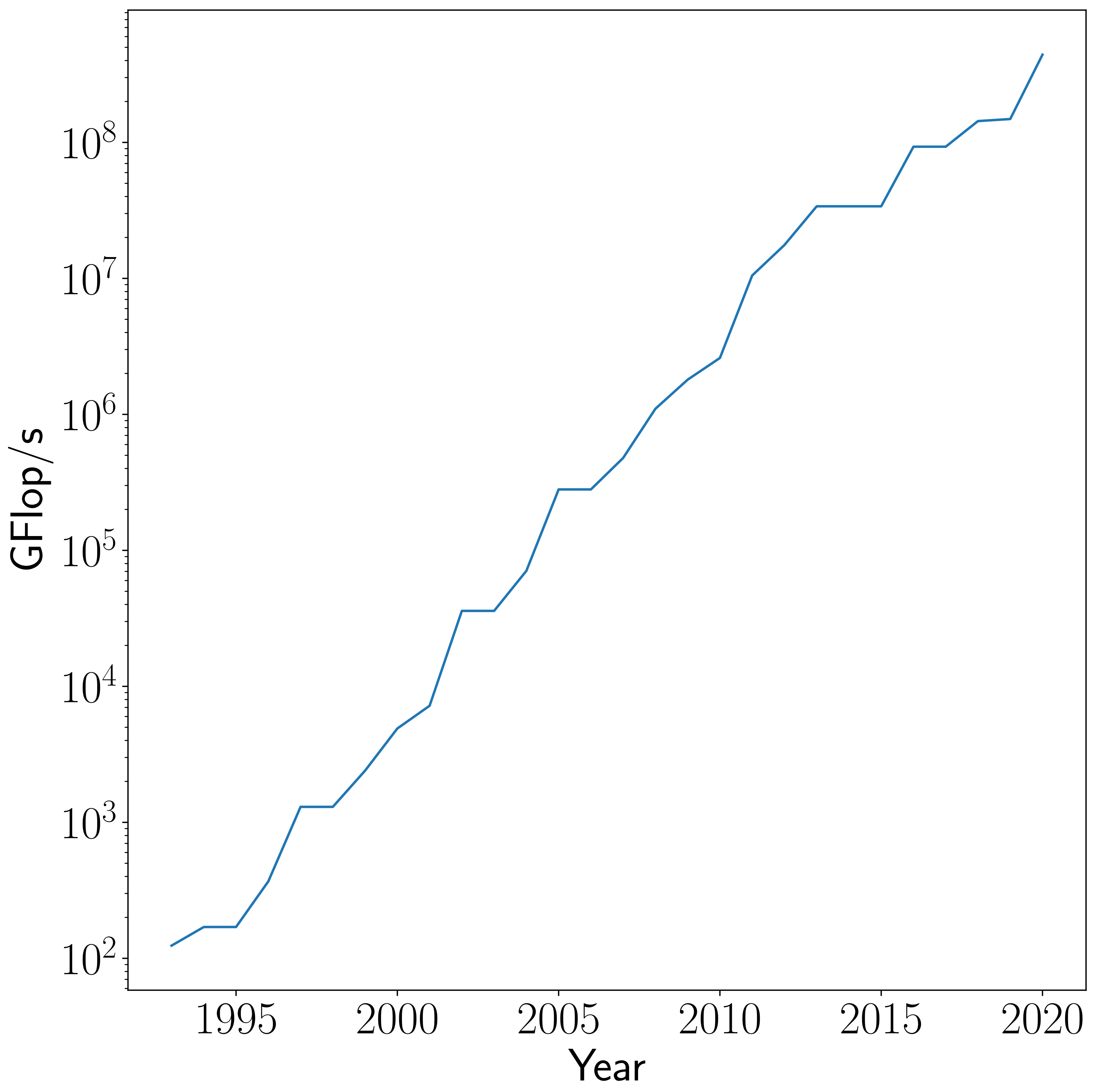


10,000 nodes

[Mixed-integer nonlinear optimization, Belotti, Kirches, Leyffer, Linderoth, Luedtke, Ashutosh]

Progress of mixed-integer optimization

Top500 peak CPU power



Hardware speedups

4 mln x

Software speedups

100,000 x

400 billion times speedups!

400,000 years → **30 seconds**

[Our World in Data, https://github.com/owid/owid-datasets]

Branch and bound algorithms

Today, we learned to:

- **Develop** branch and bound iterations to solve mixed-integer optimization
- **Understand** the rules and the practical implications in branch and bound
- **Solve** small numerical examples
- **Apply** branch and bound to a cardinality-constrained optimization

Next lecture

- The role of optimization