

ORF307 – Optimization

21. Integer optimization algorithms

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)?

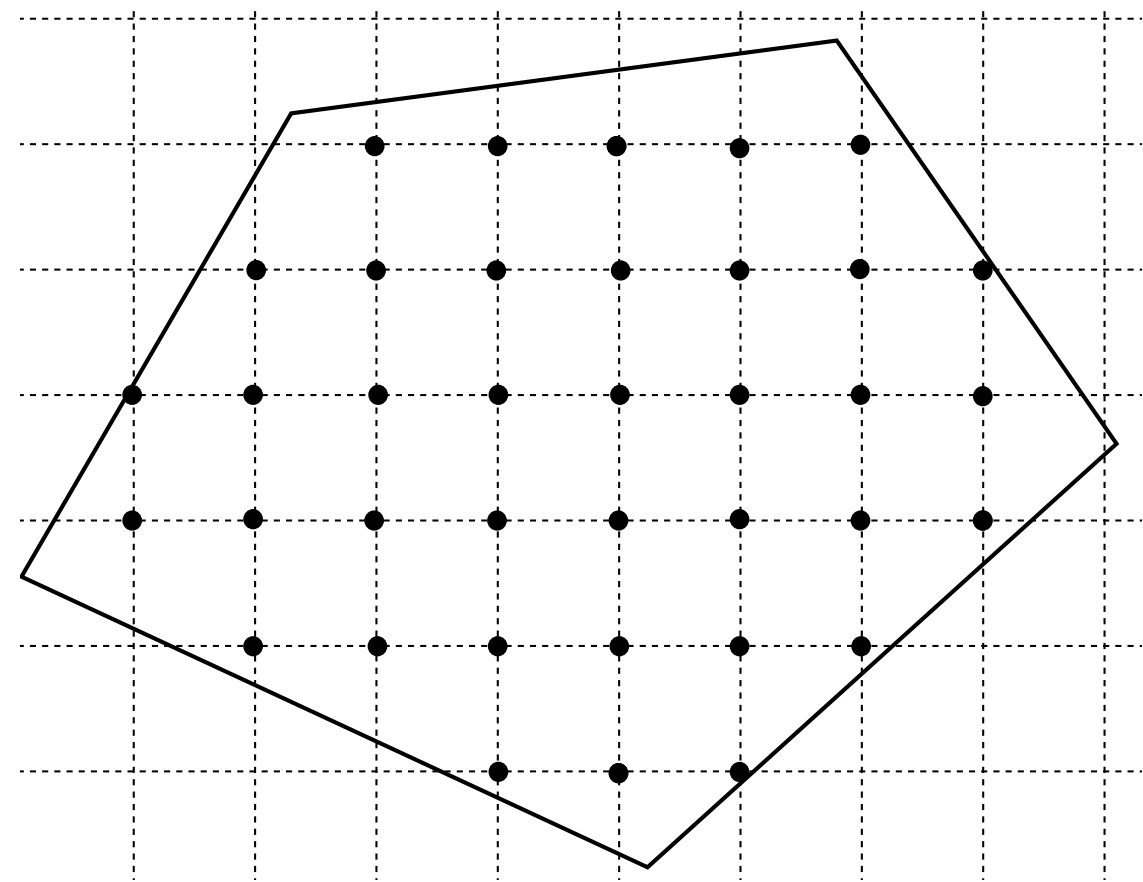
Recap

Relaxations

Remove integrality constraints

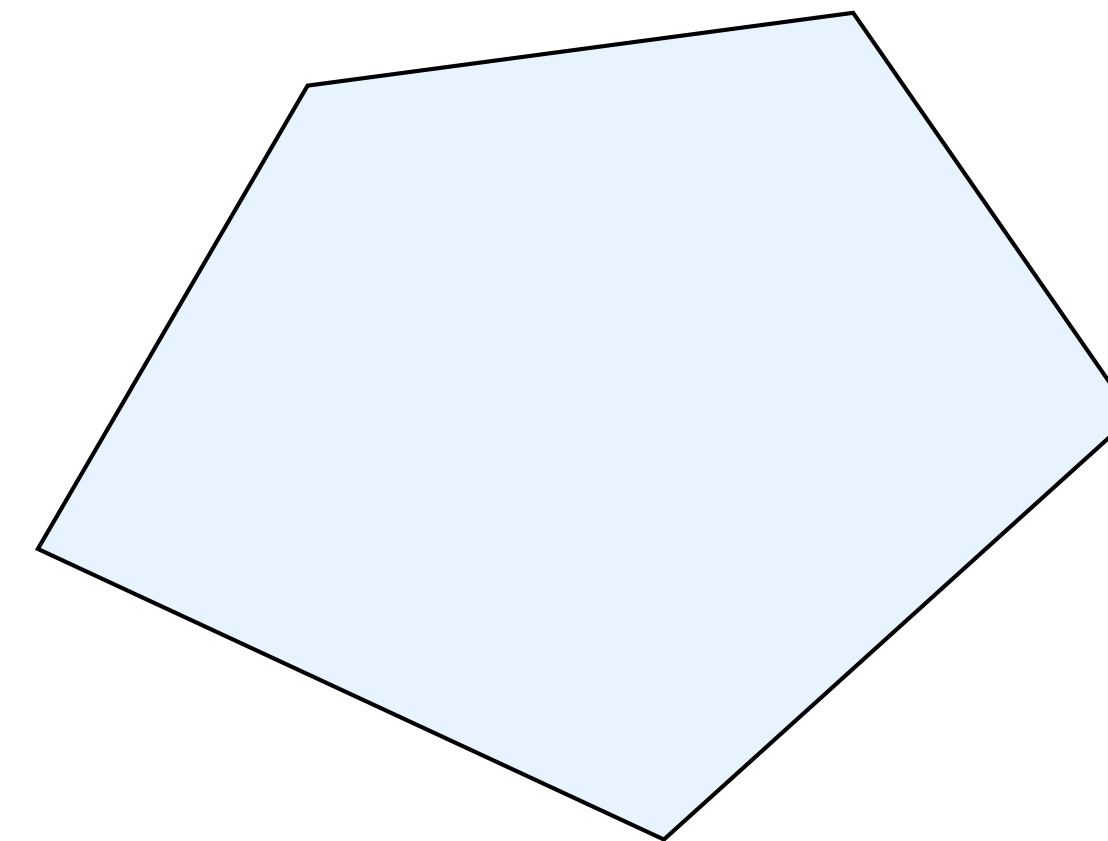
minimize $c^T x$
subject to $Ax \leq b$
 $x_i \in \mathbf{Z}, \quad i \in \mathcal{I}$

P_{ip} \longrightarrow



minimize $c^T x$
subject to $Ax \leq b$

P_{rel} \longleftarrow



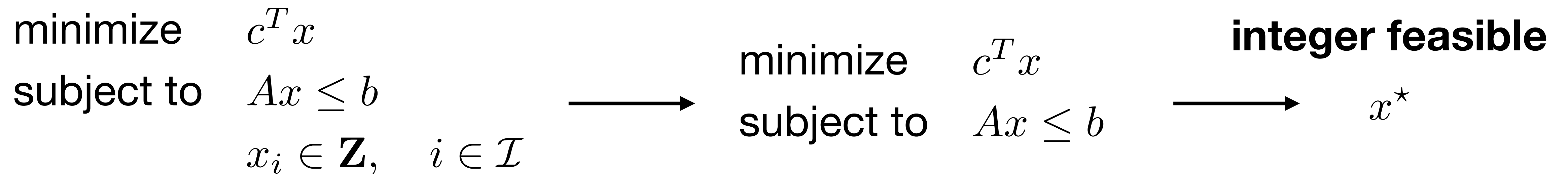
$P_{\text{ip}} \subset P_{\text{rel}}$



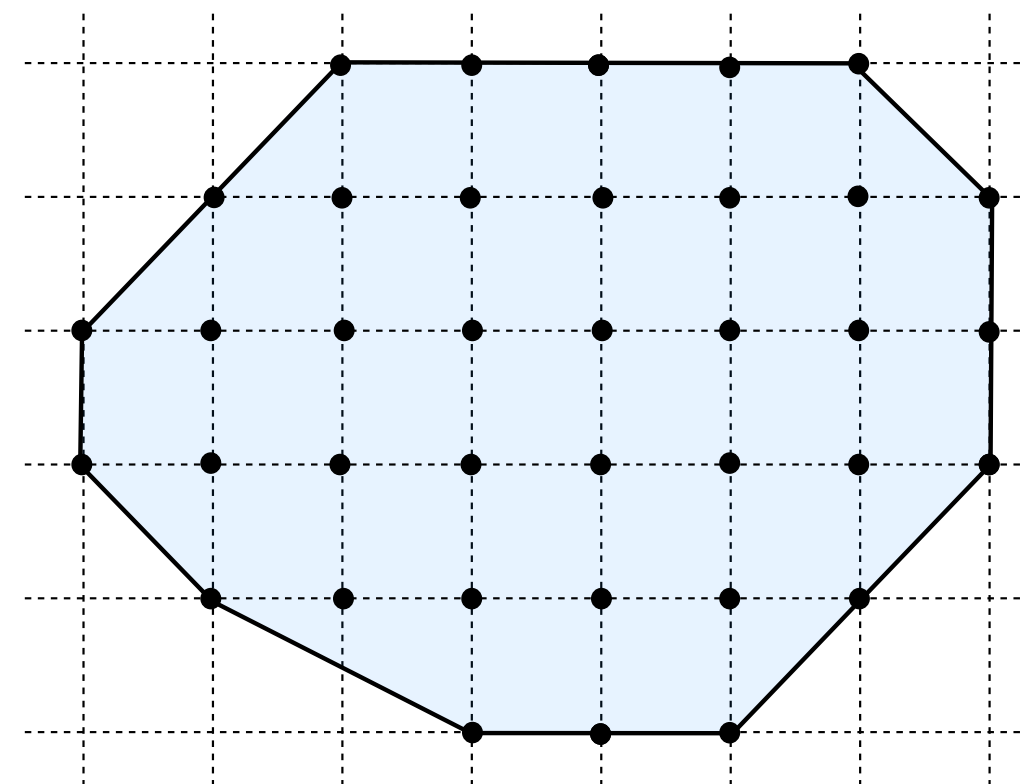
Relaxations provide
lower bounds to p_{ip}^*
 $p_{\text{rel}}^* \leq p_{\text{ip}}^*$

Ideal formulations

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



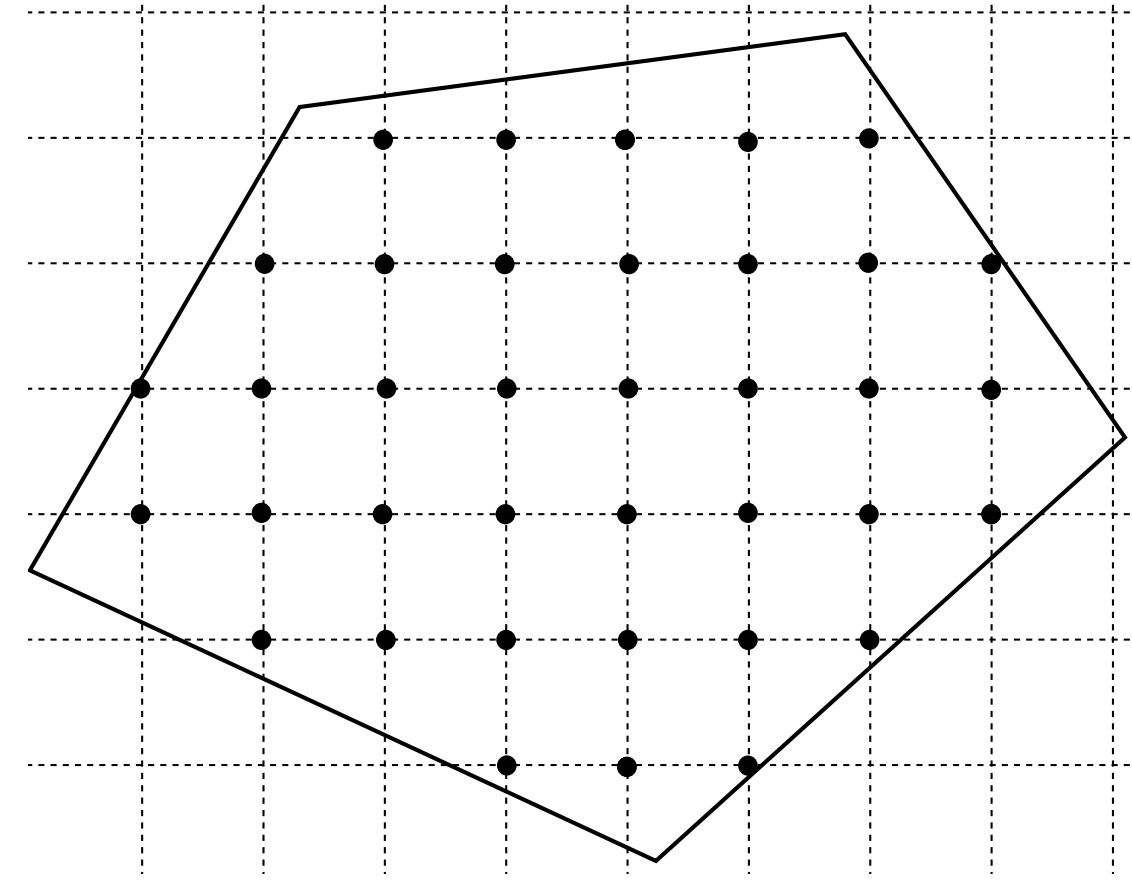
This happens if
 $\text{conv } P = \{Ax \leq b\}$



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

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

How do you solve it?

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

- 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

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

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

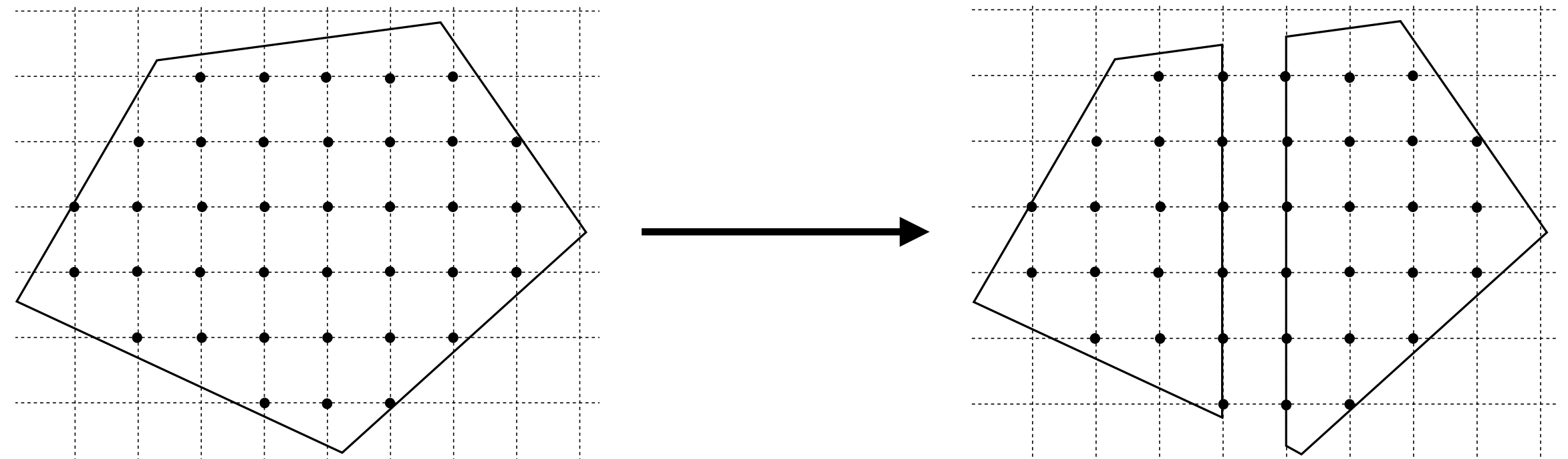
Divide and conquer

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

- **Solve subproblems**

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

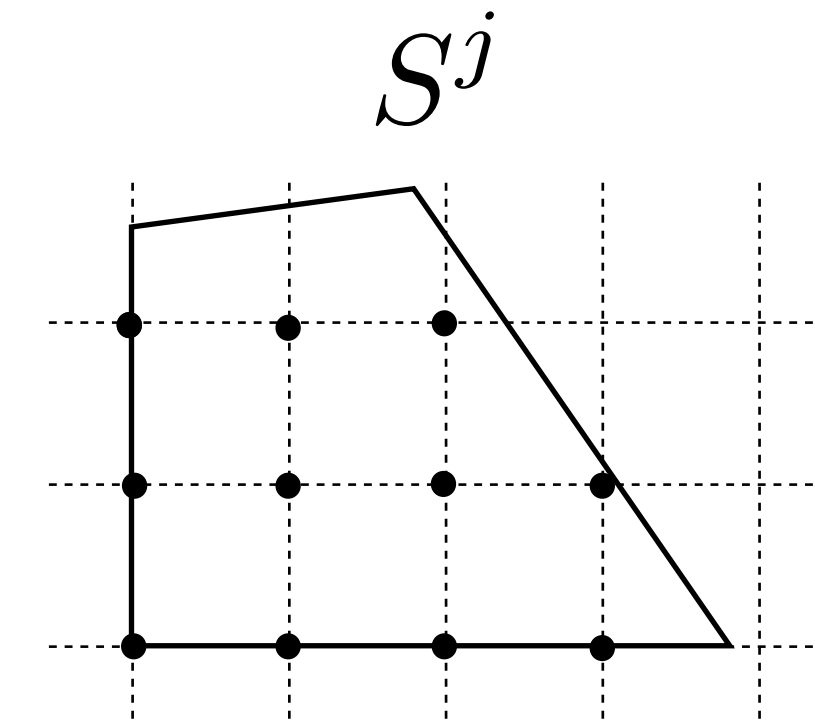
- Until **optimal** x^* **is found**



Two efficient subroutines

For every region S^j

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



Lower bound
(relaxation)



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

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



Upper bound
(evaluate any point)

Bounds guide us in the search!

Branch and bound algorithm

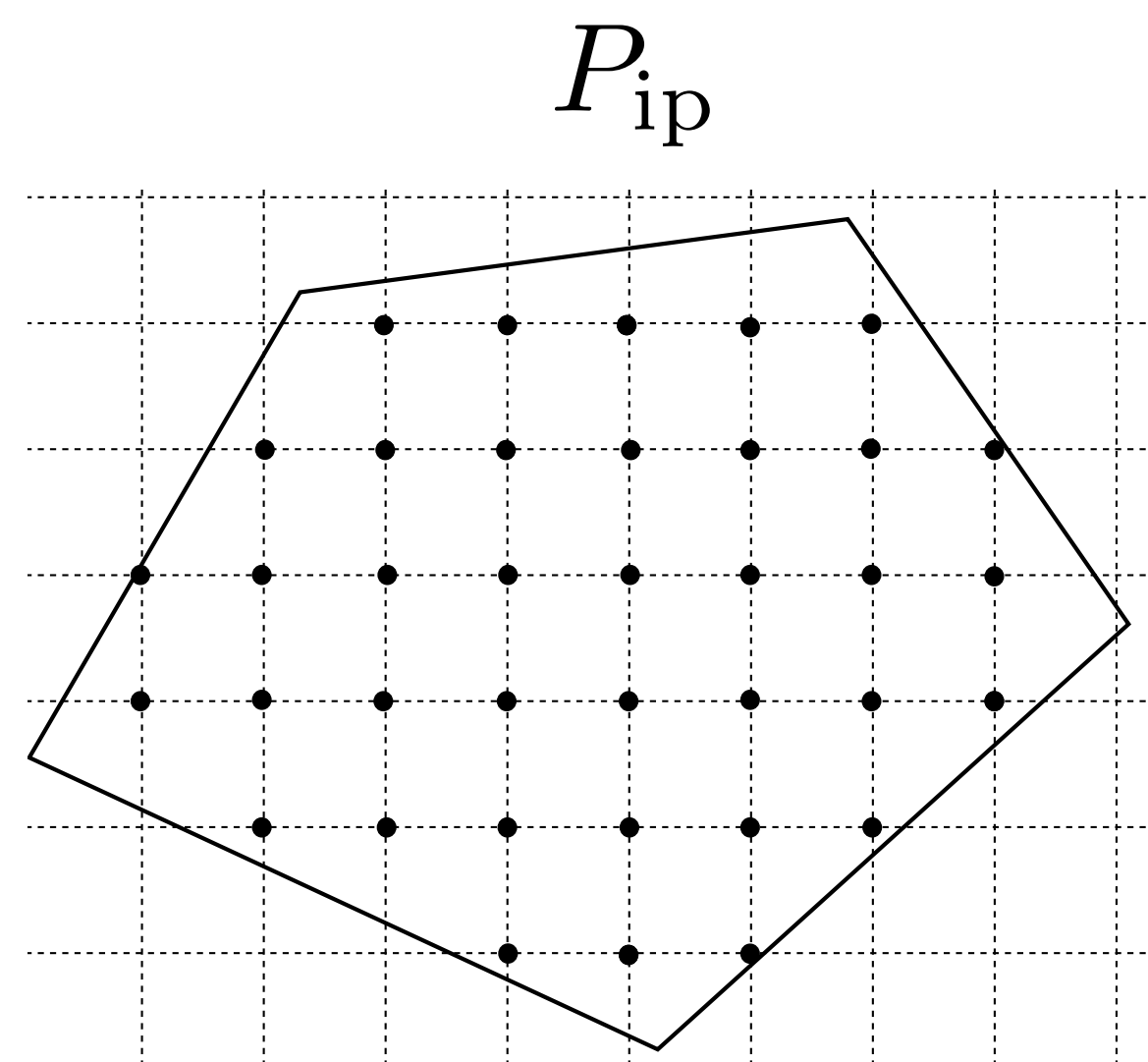
$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x \in P_{\text{ip}} \end{array} \quad P_{\text{ip}} = \{x \mid Ax \leq b, \quad x_i \in \mathbf{Z}, \quad i \in \mathcal{I}\}$$

Iterations

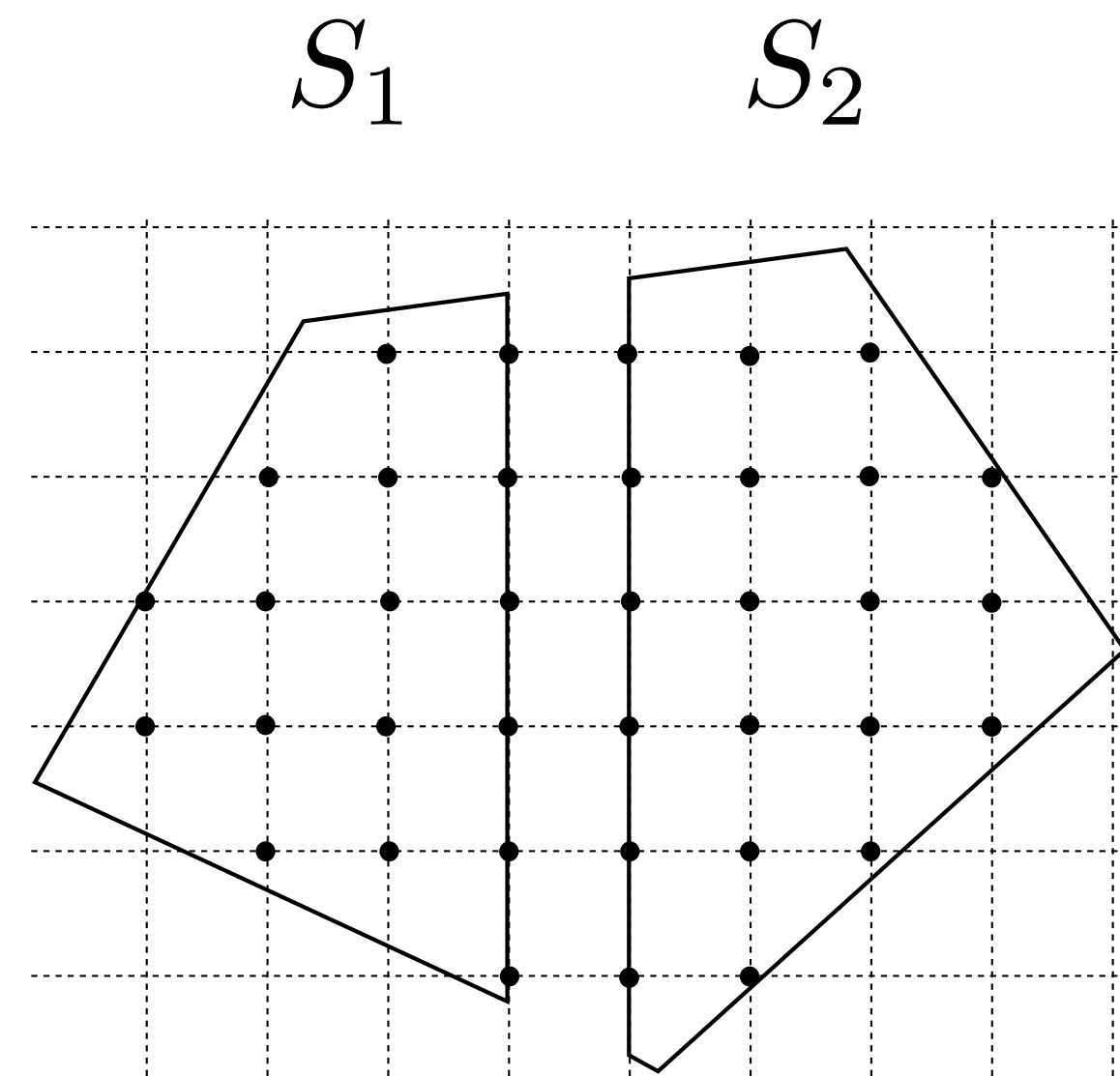
1. **Branch:** create/refine the partition if P_{ip} and get S^j
2. **Bound:**
 - Compute **lower and upper bounds**
$$L_j = \Phi_{\text{lb}}(S^j), \quad U_j = \Phi_{\text{ub}}(S^j), \quad \forall j$$
 - Update **global bounds** on $c^T x^*$
$$L = \min_j \{L_j\}, \quad U = \min_j \{U_j\}$$
3. If $U -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

Branch and bound

Example in 2D



$$U = 8.5$$
$$L = 2$$



$$U_1 = 8.5$$
$$L_1 = 6$$

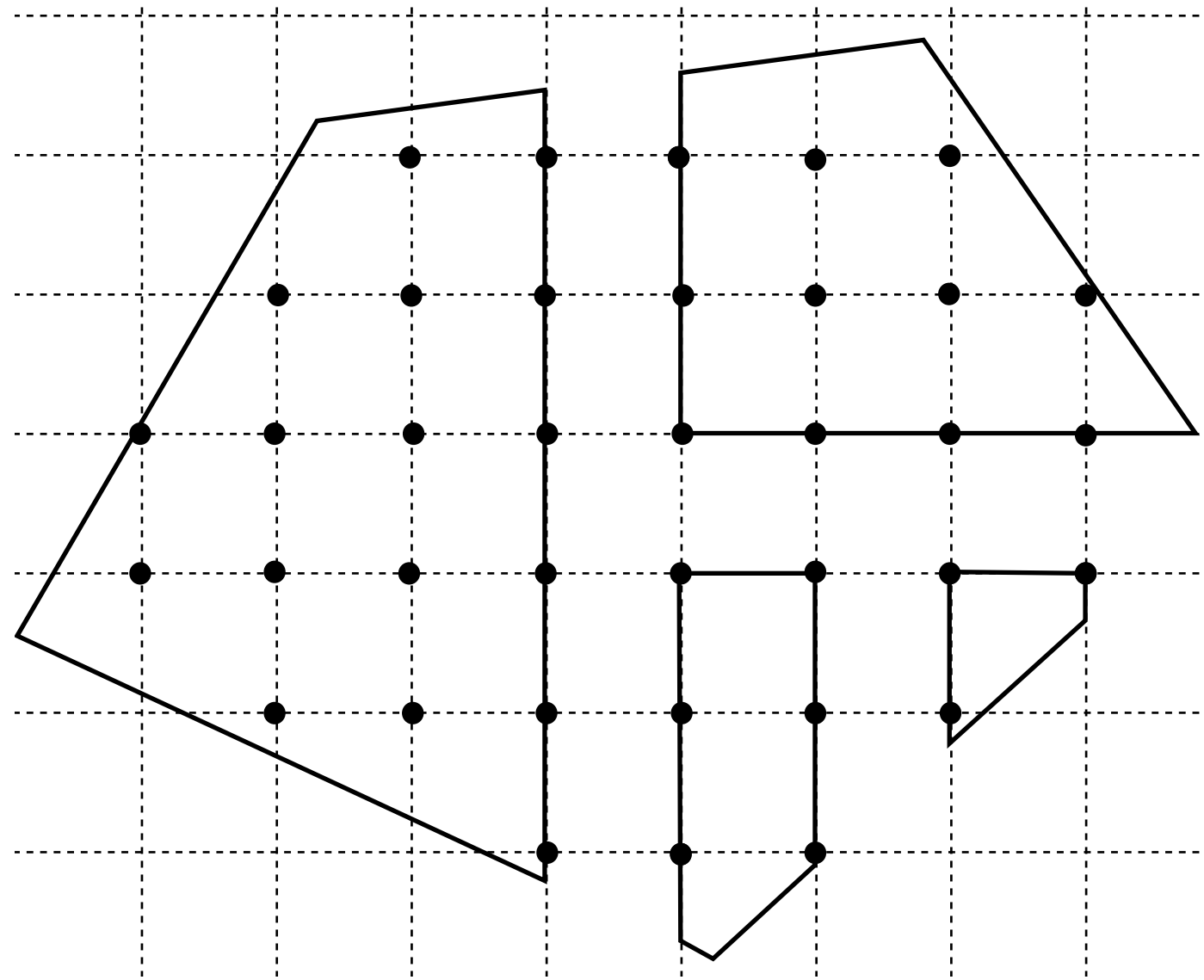
$$U_2 = 4.2$$
$$L_2 = 2$$

What does this say
about global bounds
 L and U ?

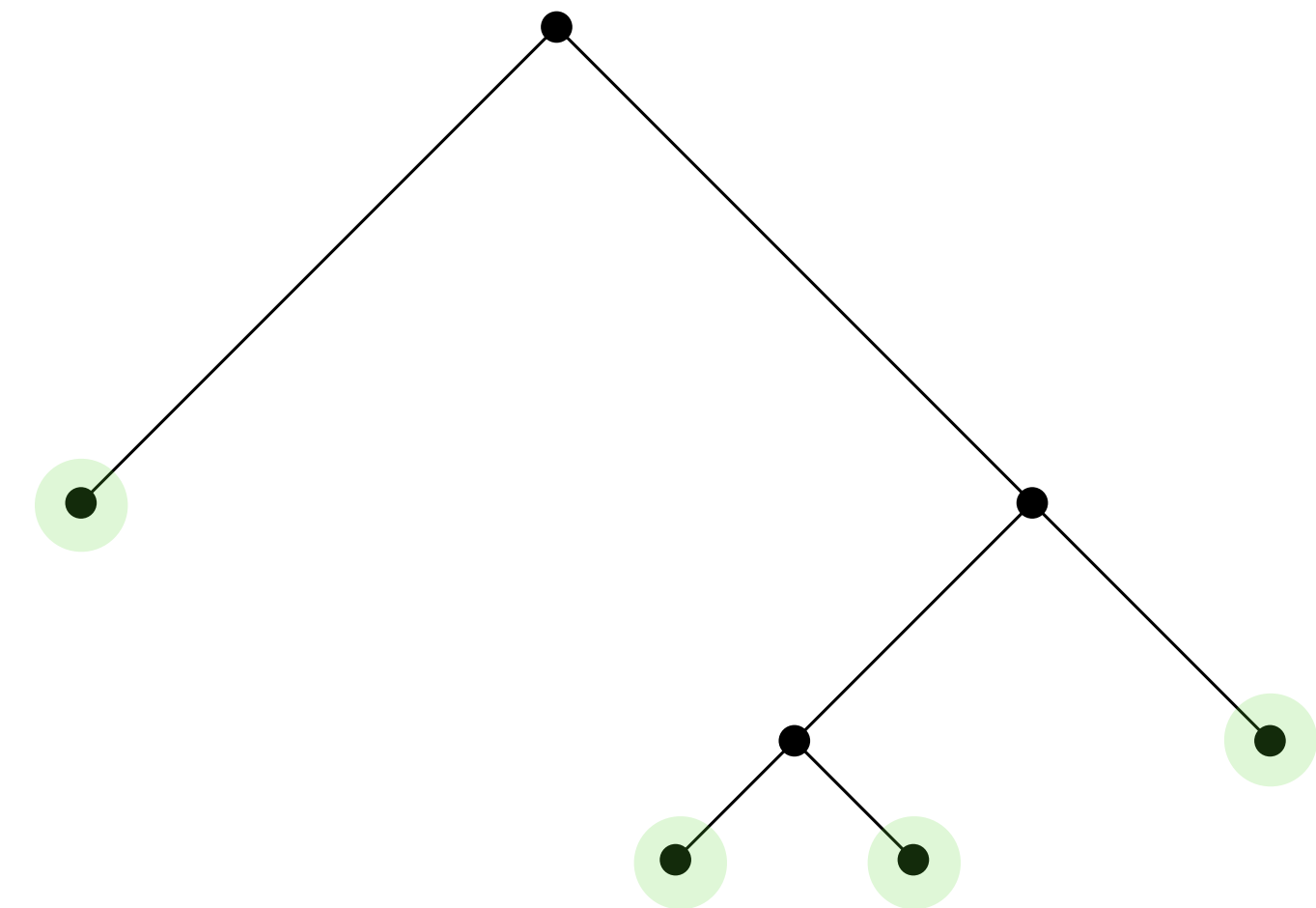
Partition as a binary tree

Example in 2D

Partition



Binary tree



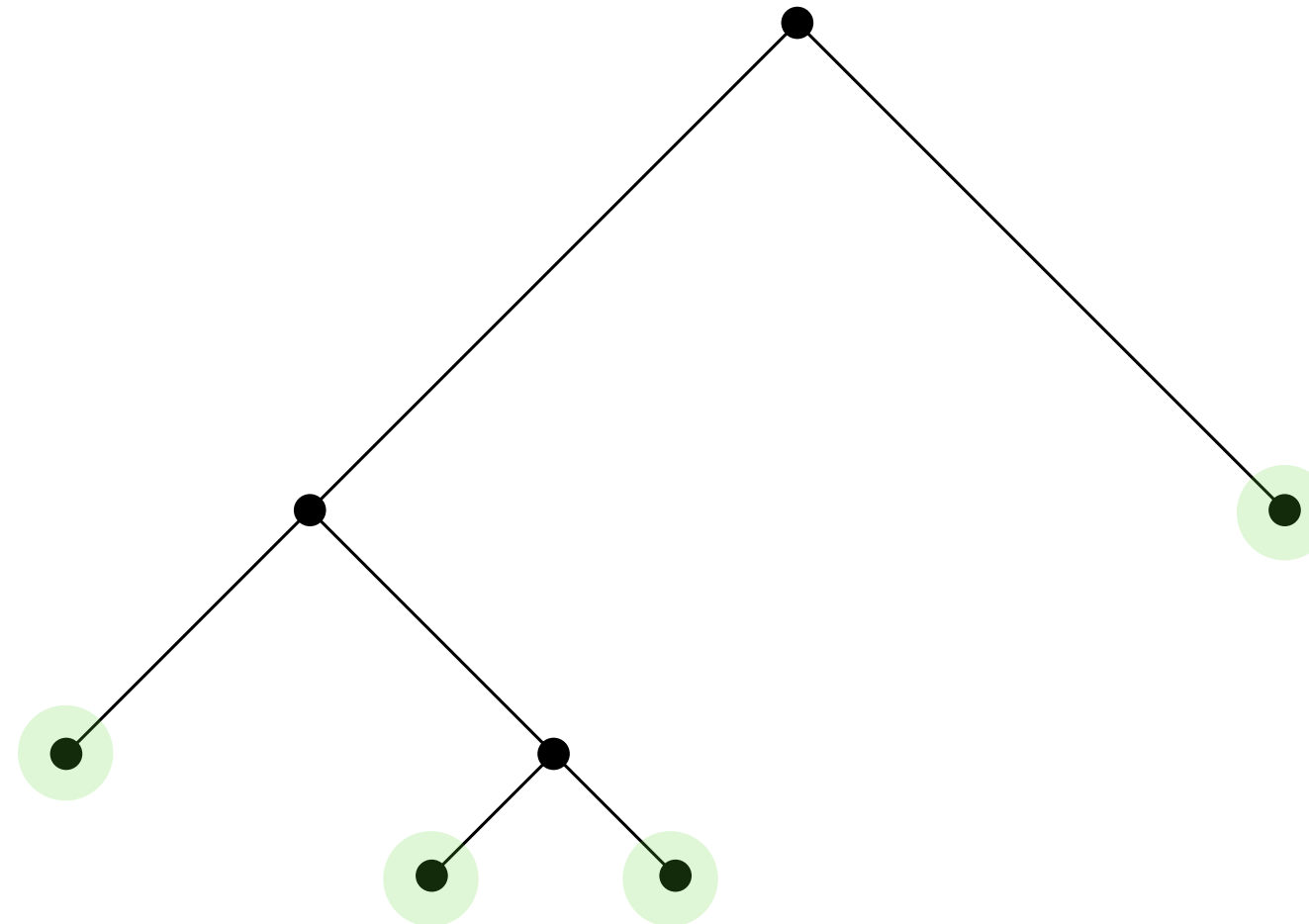
At each step we have a **binary tree**

Children correspond to subregions formed by splitting parents

Certifying optimality

certify optimality \longrightarrow $L \leq c^T x^* \leq U$ \longleftarrow return feasible point
"incumbent"

Partition = Leaves



Optimality certificate in integer optimization

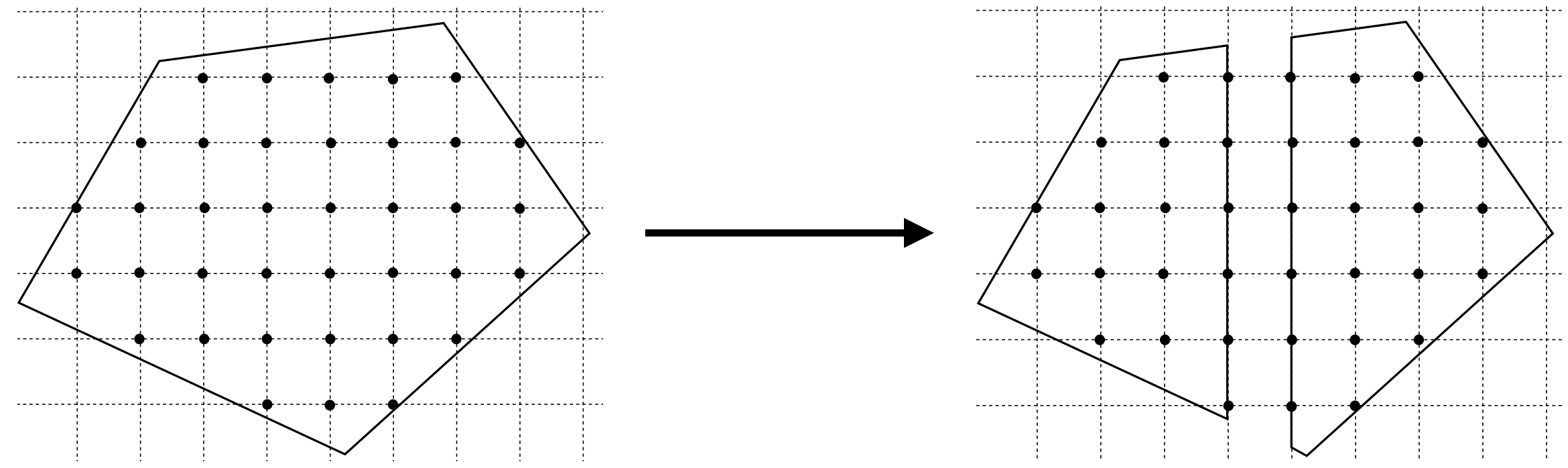
- Partition S^j
- Bounds $(L_j, U_j) \quad \forall j$

Optimality certificate in linear optimization

Dual variables and cost

Branch and bound rules

Partitioning



**Pick one subproblem
and solve its relaxation**

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

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

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

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

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**

Example heuristic (best-bound search)

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

Pruning

Key performance component

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

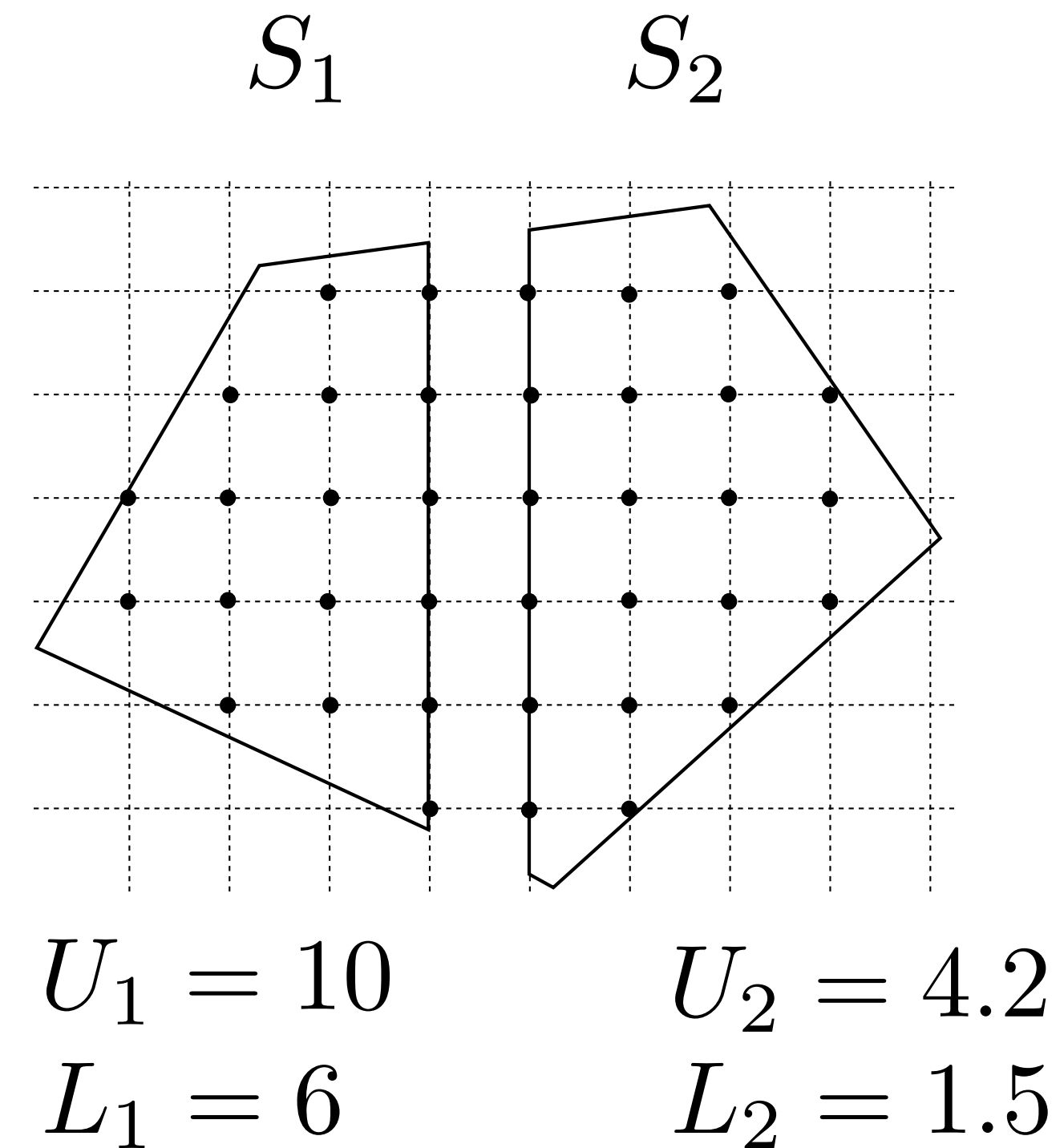
S^j is **active** if $L_j \leq \min_j U_j$

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

Questions

What is S^1 ? active/inactive

What is S^2 ? active/inactive



$$L = L_2$$

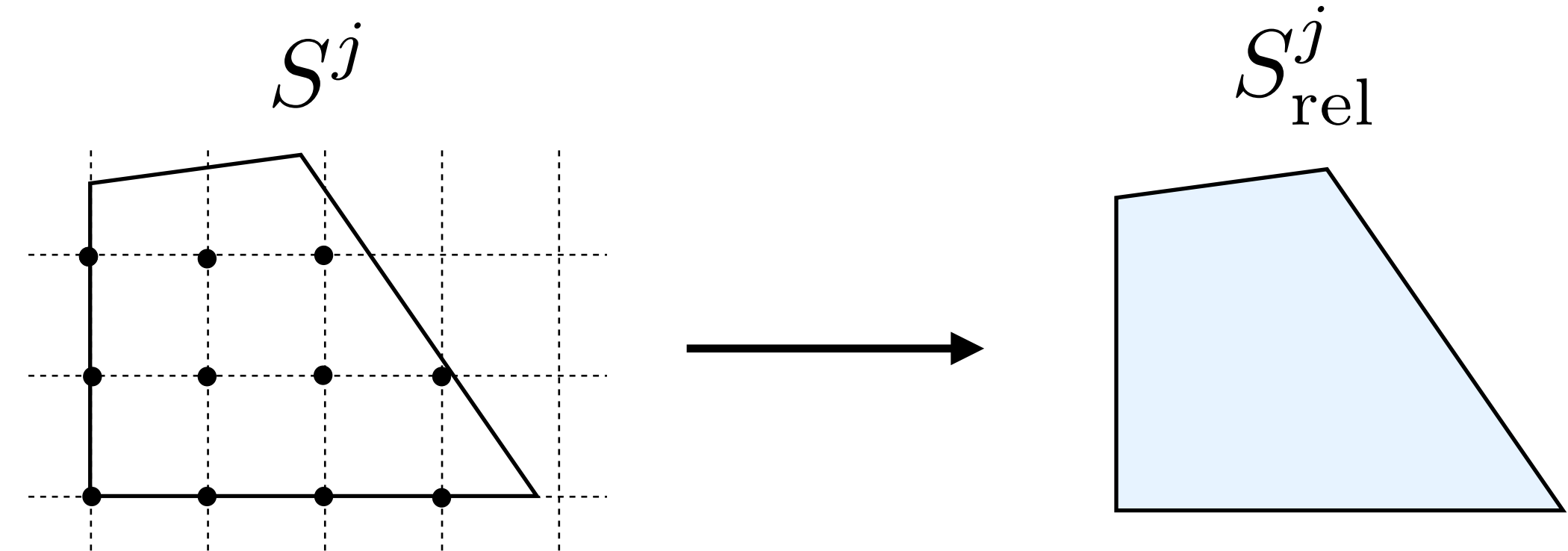
$$U = U_2$$

Bounding rules

Lower bounds. Solve relaxation

$$\begin{aligned} \bar{x} \leftarrow & \text{minimize} && c^T x \\ & \text{subject to} && x \in S_{\text{rel}}^j \end{aligned}$$

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

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

Branch and bound

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

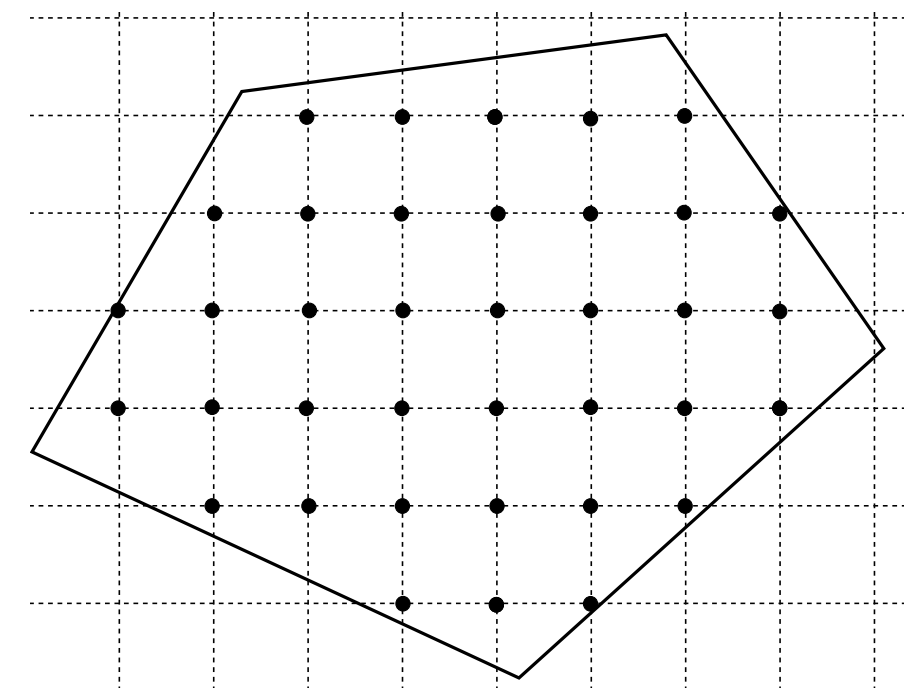
hope: it works better for our problem

Brute force

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

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

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



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

Which algorithm would you use LP subproblems?

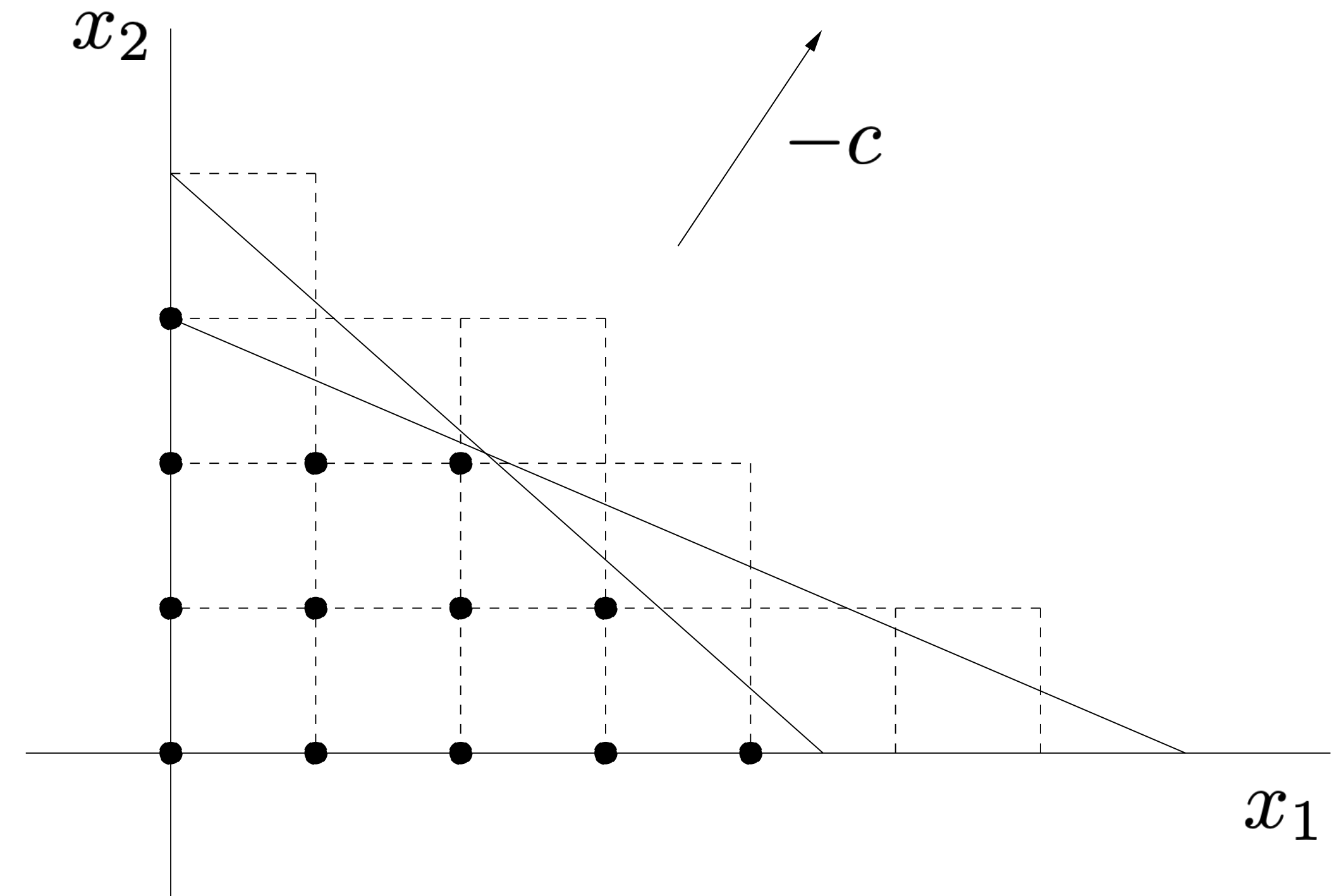
Small examples

Branch and bound example

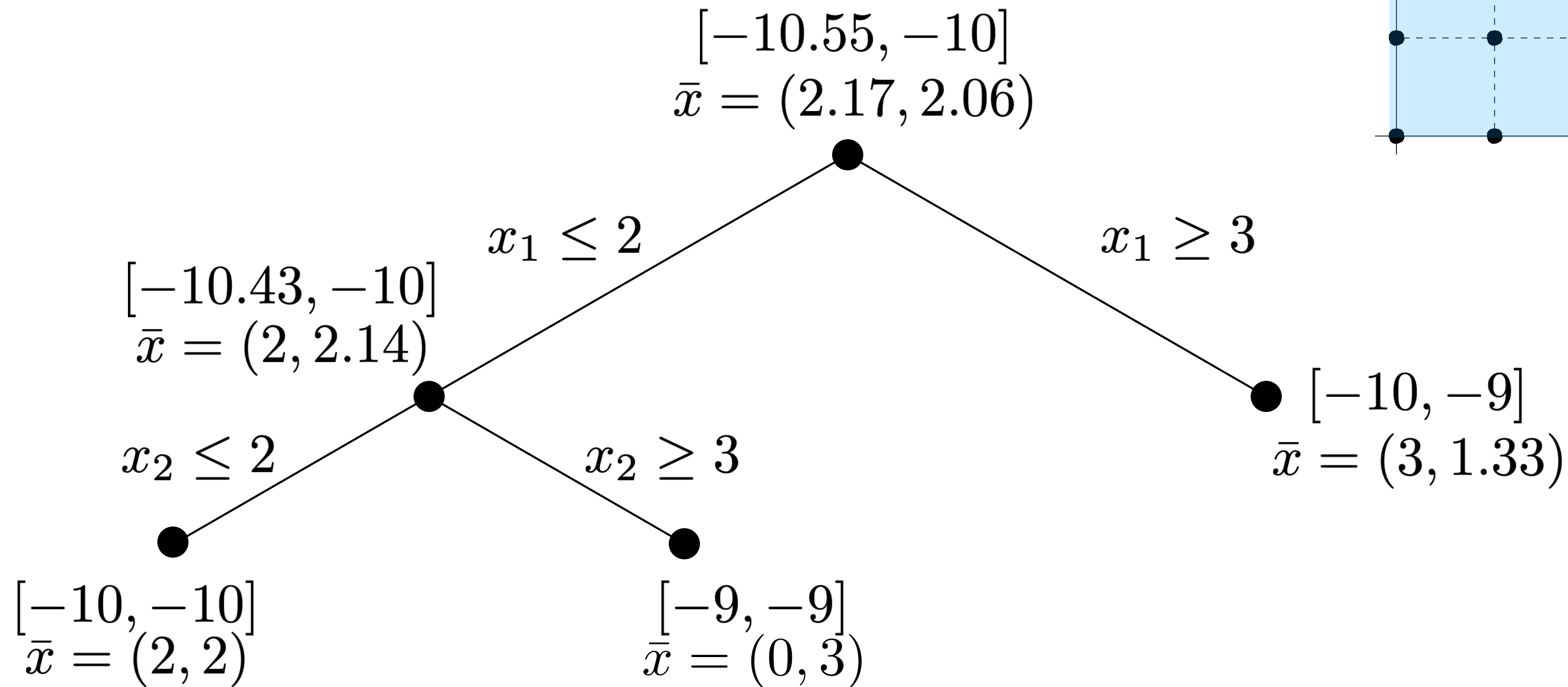
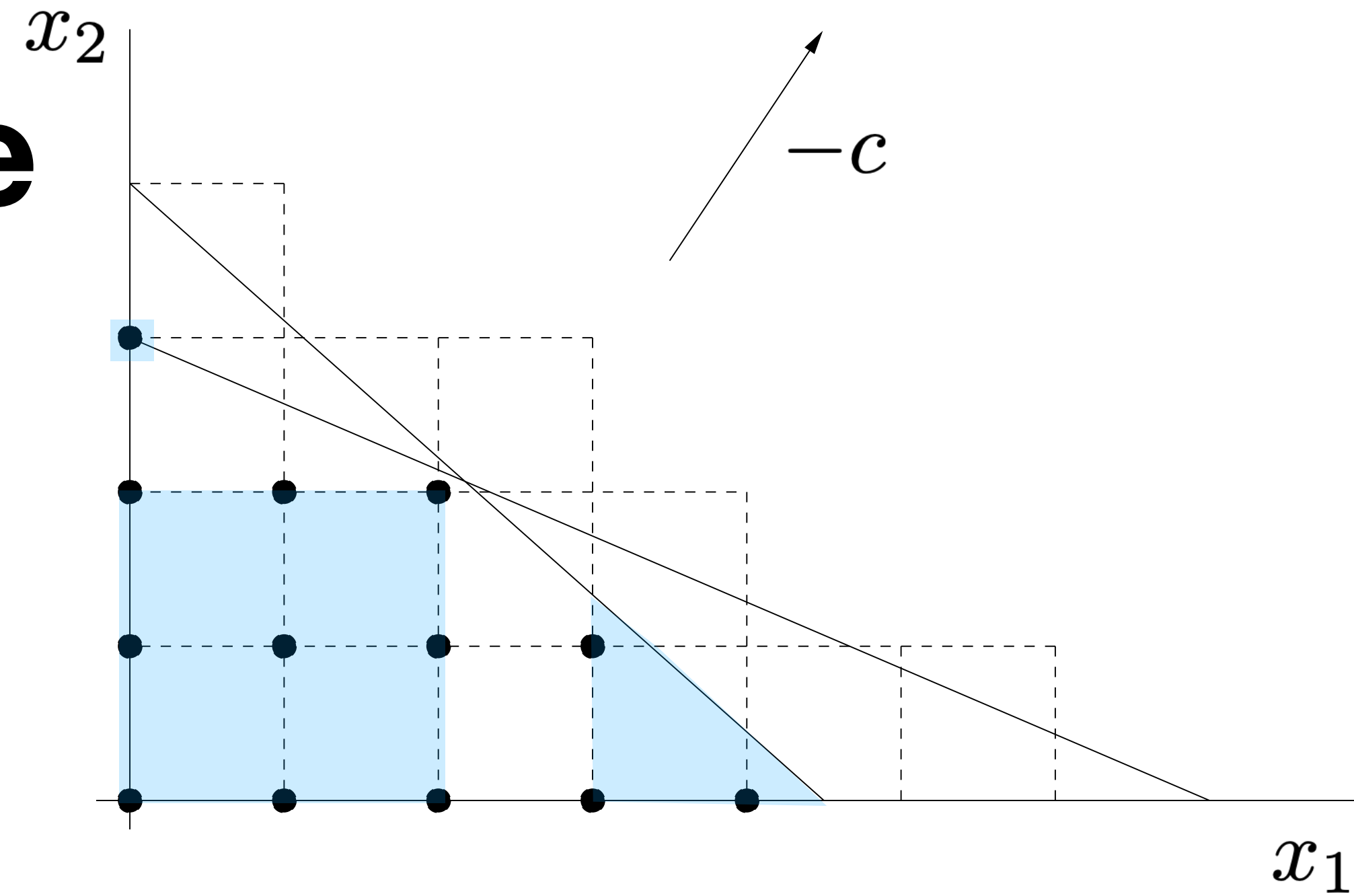
minimize $-2x_1 - 3x_2$
subject to $(2/9)x_1 + (1/4)x_2 \leq 1$
 $(1/7)x_1 + (1/3)x_2 \leq 1$
 $x_1, x_2 \geq 0$
 $x_1, x_2 \in \mathbf{Z}$

Optimal solution

$$x^* = (2, 2)$$

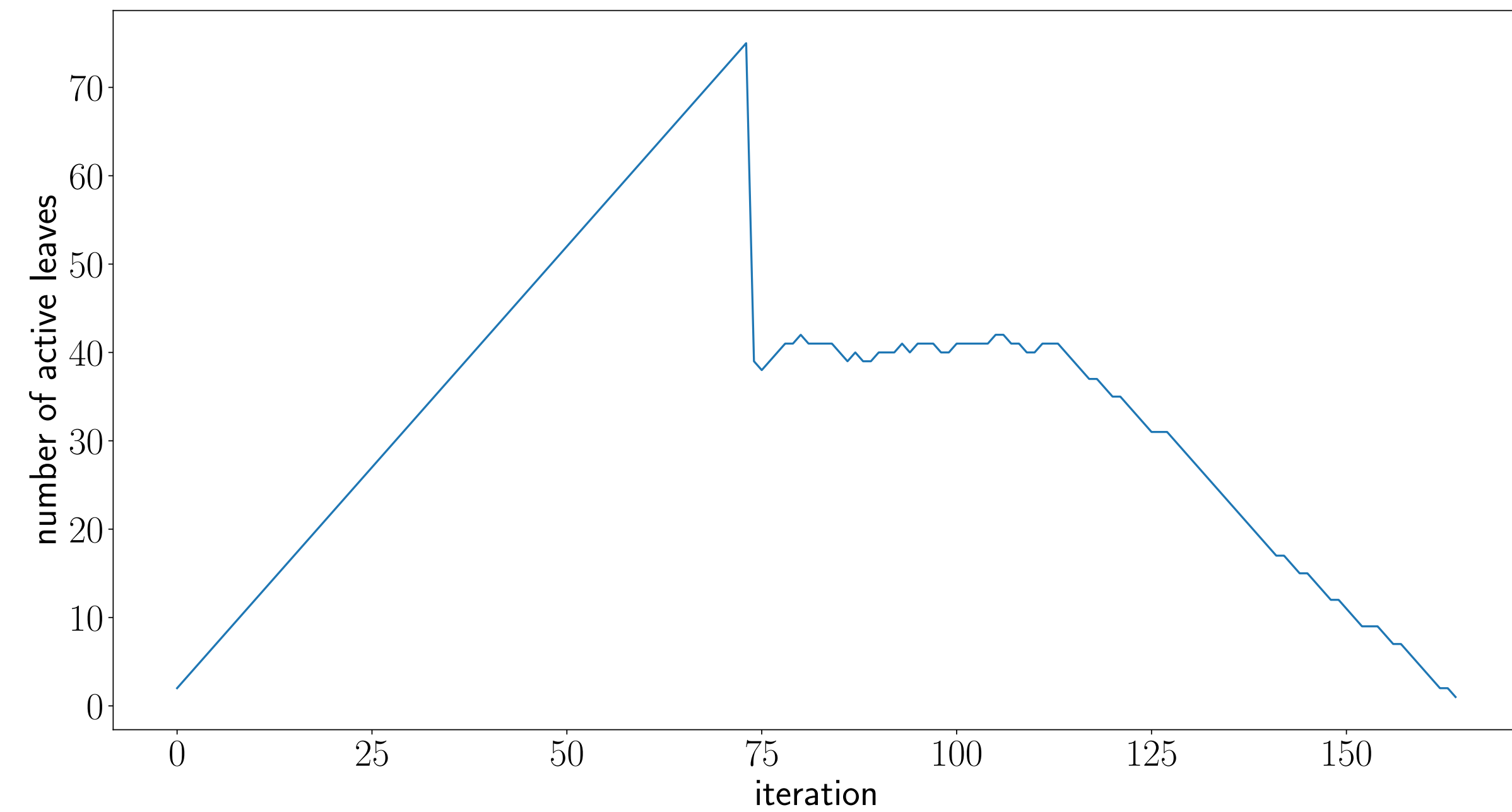
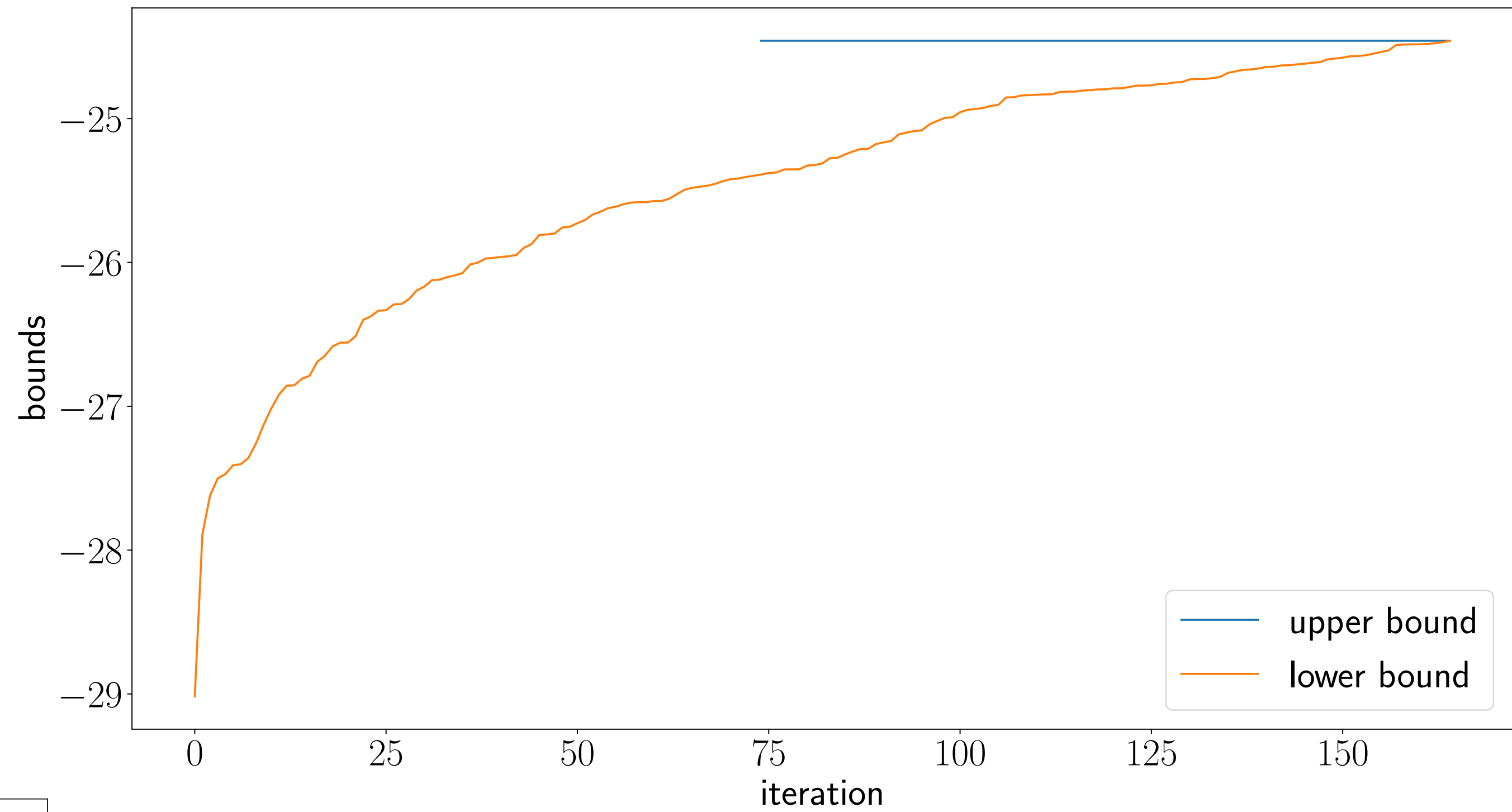


Branch and bound example



A larger example

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



Cardinality minimization

Minimum cardinality example

Find sparsest x satisfying linear inequalities

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

Equivalent mixed-boolean LP

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

**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

minimize x_i
subject to $Ax \leq b$

u_i is the optimal value of

maximize x_i
subject to $Ax \leq b$

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

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

If $u_i = -l_i = M$, then

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

1-norm minimization

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

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

Implementation details

Upper bound $\text{card}(\bar{x})$ 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

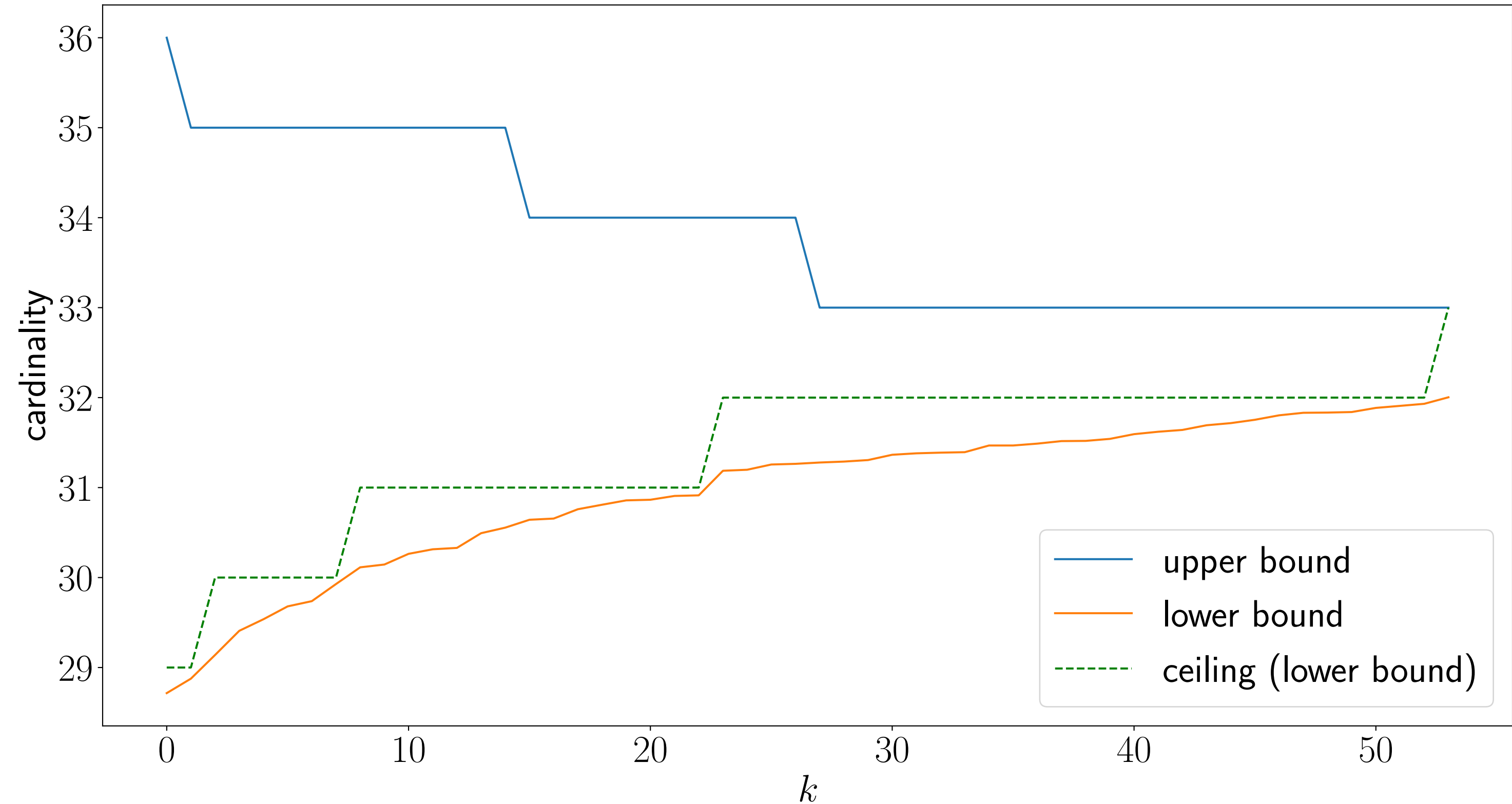
Data

40 variables, 200 constraints

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

Results

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



Medium example

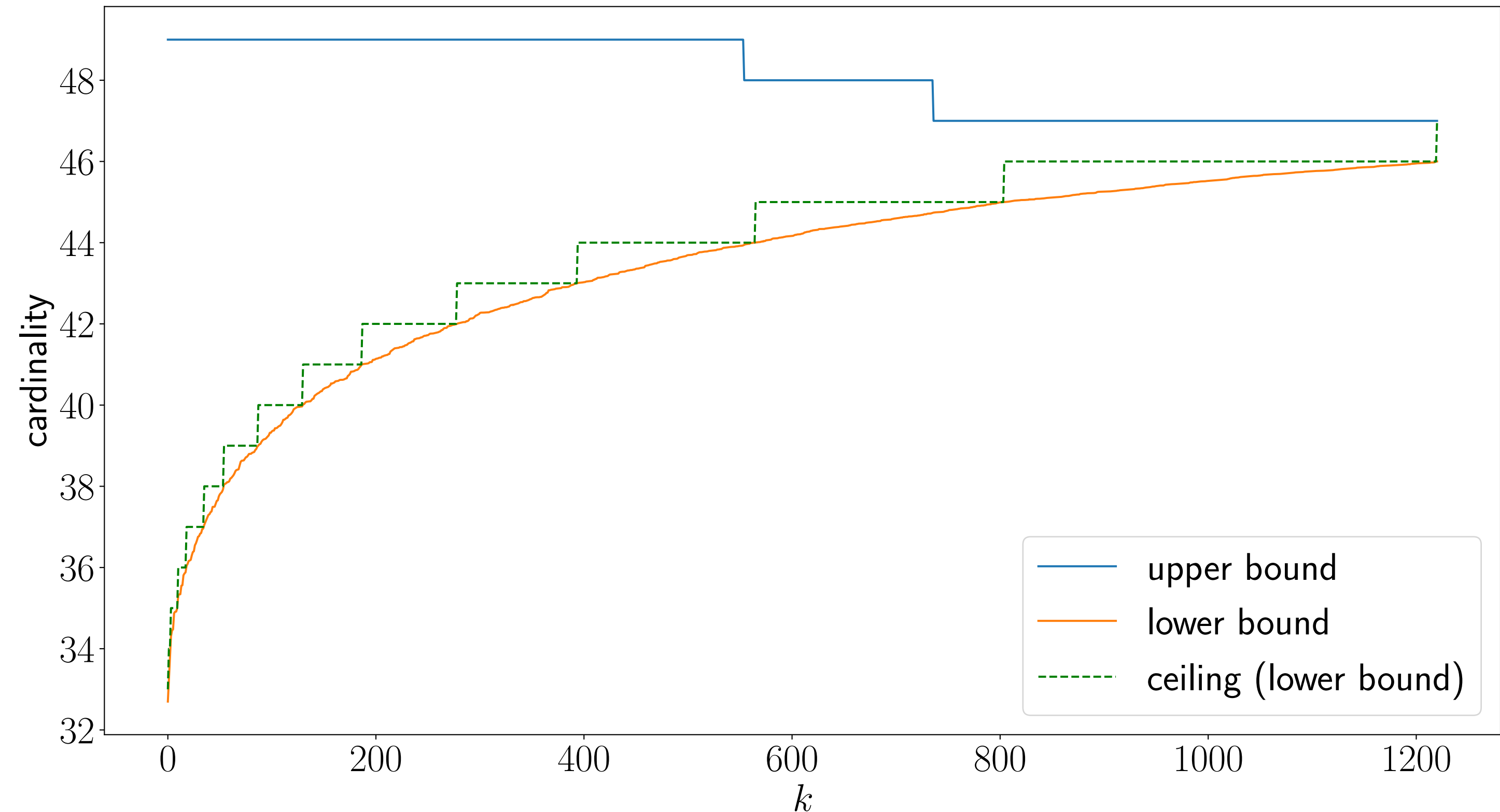
Data

60 variables, 200 constraints

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

Results

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



Larger example

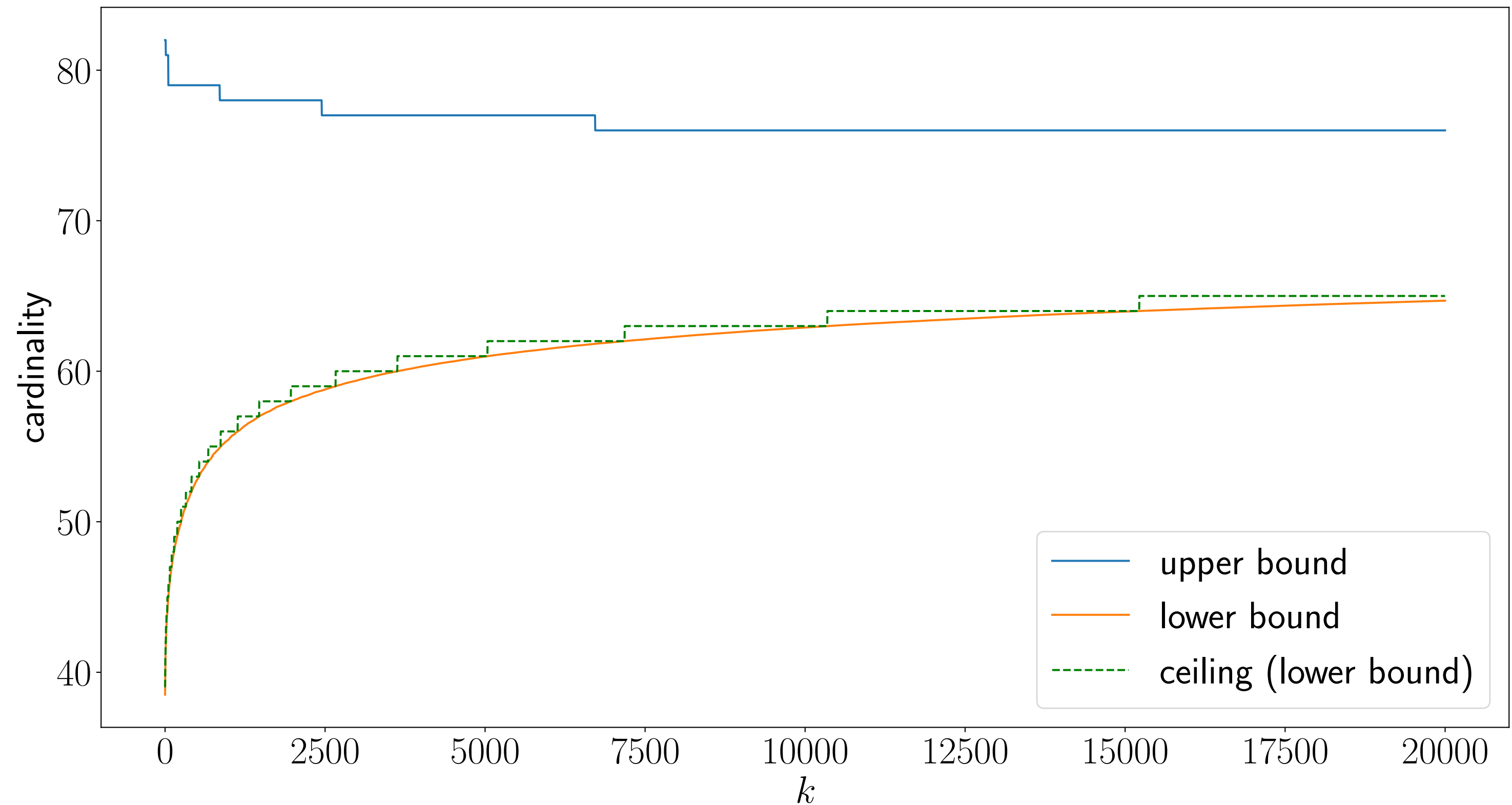
Data

100 variables, 300 constraints

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

Results

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



Larger example with commercial solver

Gurobi output

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 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

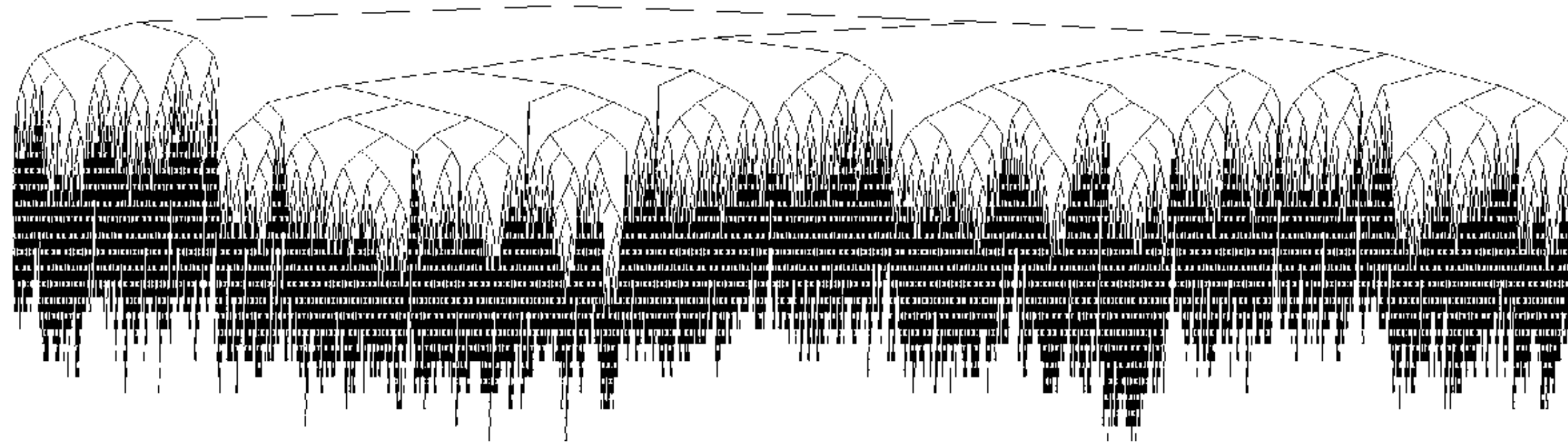
   Nodes |      Current Node |      Objective Bounds |      Work
  Expl Unexpl |  Obj  Depth IntInf | Incumbent  BestBd  Gap | It/Node Time
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----
    0     0   29.33185    0  85         -   29.33185    -   -    0s
H    0     0         -         -         -   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

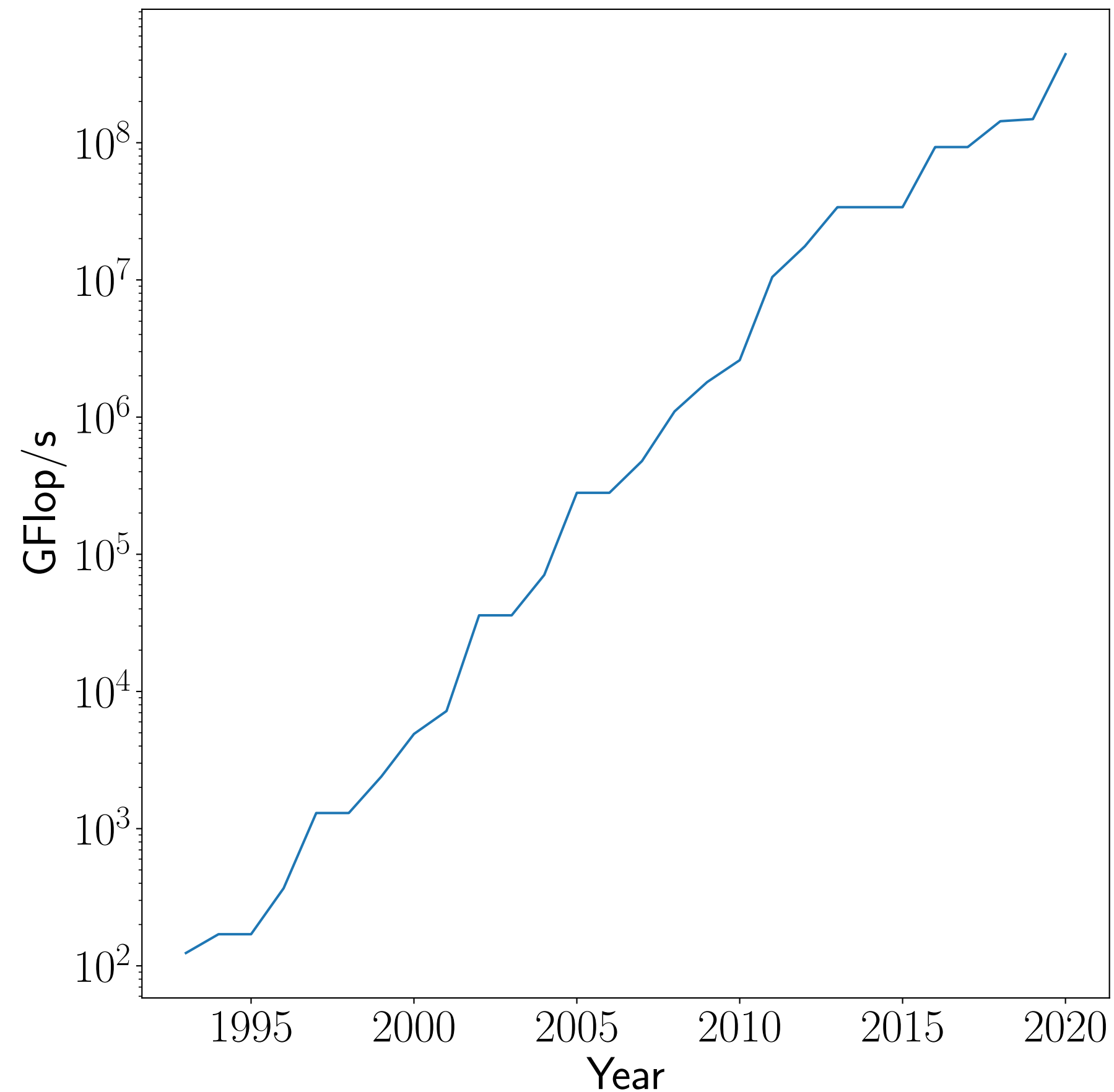
Example for 360 seconds on CPU...



10,000 nodes

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

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