ORF522 – Linear and Nonlinear Optimization

21. Branch and bound algorithms

Bartolomeo Stellato — Fall 2021

Final exam length poll

- 24 hours
- 12 hours

Please complete the poll at

https://forms.gle/FmniL2dVmQWFHgv17

Ed Forum

- What is the proof that convex-concave procedure converges to a stationary point? Sketch

  1) show always feasible: $f_i(x) - g_i(x) \leq f_i(x) - \hat{g}_i(x) \leq 0$
  2) show descent method:

$$f_0(x^k) - g_0(x^k) \geq f_0(x^k) - \hat{g}_0(x^k; x^k)$$
$$\geq \min_x f_0(x) - \hat{g}_0(x; x^k) = f_0(x^{k+1}) - \hat{g}_0(x^{k+1}; x^k)$$

Therefore $f_0(x^k) - \hat{g}_0(x^k)$ is nonincreasing and it converges (possibly to $-\infty$)

[More at "Variations and extension of the convex–concave procedure", Lipp, Boyd]

Today's lecture
[MINLO][ee364b]

Branch and bound algorithms

- Main concepts
- Spacial branch and bound
- Convergence analysis
- Mixed-boolean convex optimization
- Cardinality minimization example

Main concepts

Methods for nonconvex optimization

Convex optimization algorithms: global and typically **fast**

Nonconvex optimization algorithms: must give up one, global or fast

- **Local methods: fast** but **not global**
  Need not find a global (or even feasible) solution.
  They cannot certify global optimality because
  KKT conditions are not sufficient.

- **Global methods: global** but often **slow**
  They find a global solution and certify it.

Branch and bound algorithms

Methods for **global optimization** for nonconvex problems

Not a heuristic

- Provable lower and upper bounds on global objective value
- Terminate with **certificate** of ϵ-suboptimality
- Always return global optimum

Often very slow

Exponential worst-case performance
(sometimes it works well)

The problem and its relaxation

Problem

$$x^\star = \begin{array}{ll} \text{argmin} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- f can be nonconvex
- $\mathcal{X}$ can be nonconvex

The problem and its relaxation

Problem

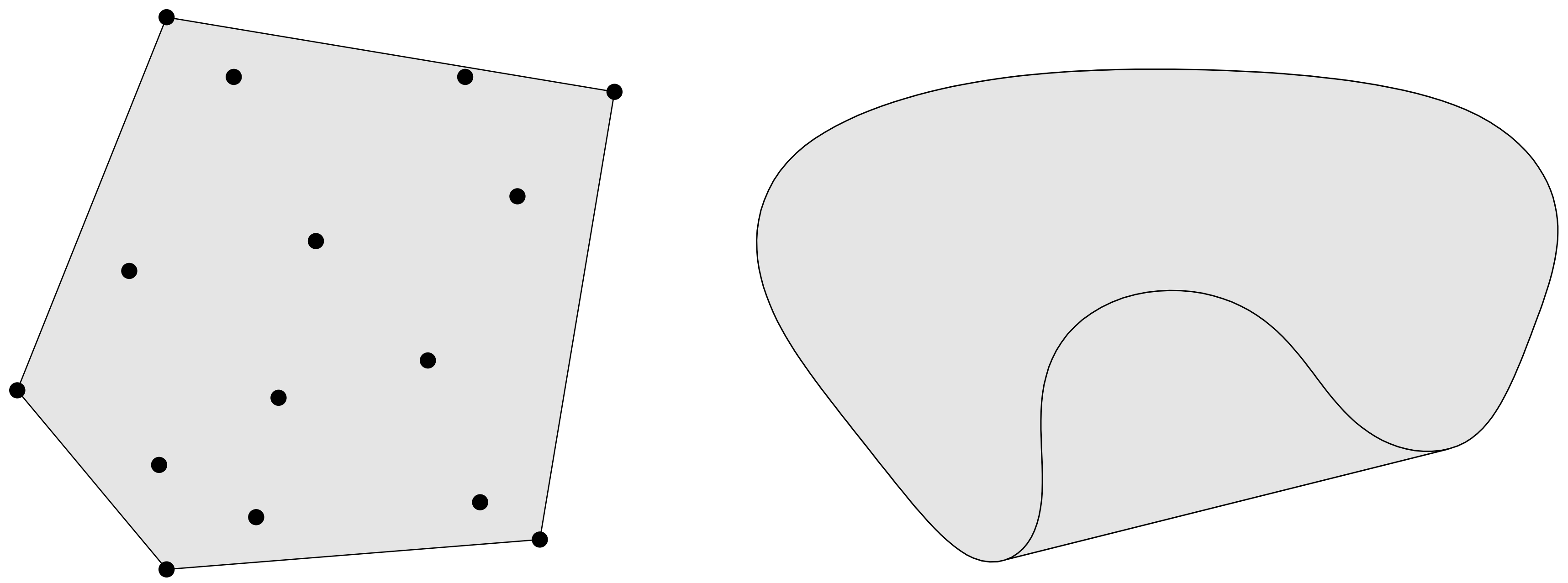
$$x^\star = \begin{array}{ll} \text{argmin} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- f can be nonconvex
- $\mathcal{X}$ can be nonconvex

Relaxation

$$\hat{x}^\star = \begin{array}{ll} \text{argmin} & \hat{f}(x) \\ \text{subject to} & x \in \mathbf{conv}\,\mathcal{X} \end{array}$$

- $\hat{f}(x) \leq f(x)$: convex underestimator
- $\mathbf{conv}\,\mathcal{X}$: convex hull



The problem and its relaxation

Problem

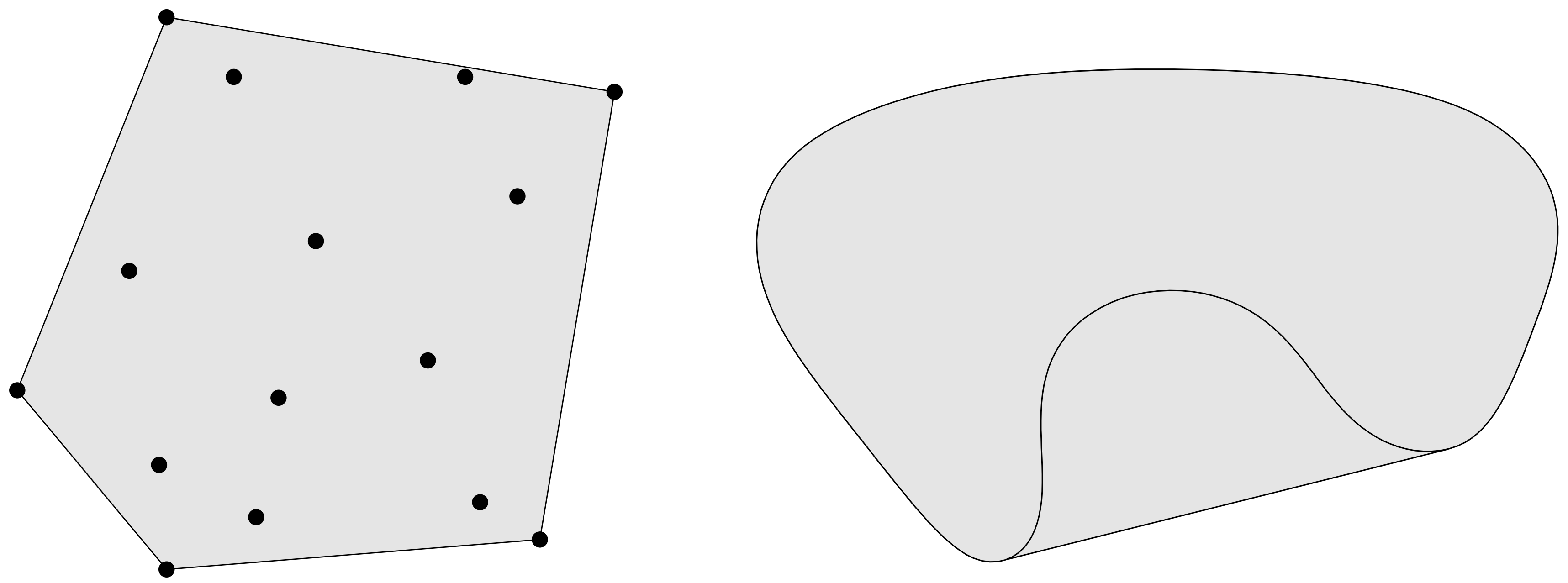
$$x^\star = \begin{array}{ll} \text{argmin} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- f can be nonconvex
- $\mathcal{X}$ can be nonconvex

Relaxation

$$\hat{x}^\star = \begin{array}{ll} \text{argmin} & \hat{f}(x) \\ \text{subject to} & x \in \mathbf{conv}\,\mathcal{X} \end{array}$$

- $\hat{f}(x) \leq f(x)$: convex underestimator
- $\mathbf{conv}\,\mathcal{X}$: convex hull



Properties

- Lower bound: $\hat{f}(\hat{x}^\star) \leq f(x^\star)$
- Larger feasible set: $\mathcal{X} \subseteq \mathbf{conv}\,\mathcal{X}$

Example

minimize $c^T x$

subject to $Ax \leq b$

$x_1 \in \{0, 1\}$

Is it convex?

How do you solve it?

Example

minimize $c^T x$

subject to $Ax \leq b$

$x_1 \in \{0, 1\}$

Is it convex?

How do you solve it?

minimize $c^T x$

subject to $Ax \leq b$

$x_i \in \{0, 1\}, \quad i = 1, \ldots, 10$

Example

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

Is it convex?

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

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

Is it convex?

How do you solve it?

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

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

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

Example

minimize $c^T x$
subject to $Ax \leq b$
$x_1 \in \{0, 1\}$

Is it convex?

How do you solve it?

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

- 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

Main idea

Two efficient subroutines
(for every region)

Lower bound: can be sophisticated

- Relaxed problem
- Lagrange dual
- Other bounds…

Upper bound: evaluate any point in the region

- Local optimization
- Evaluate function at the center

Main idea

Two efficient subroutines
(for every region)

Lower bound: can be sophisticated

- Relaxed problem
- Lagrange dual
- Other bounds…

Upper bound: evaluate any point in the region

- Local optimization
- Evaluate function at the center

Iterations

1. **Partition feasible set** into convex sets and compute lower and upper bounds
2. **Form global lower and upper bounds**. If they are close, **break**
3. **Refine partitions** and repeat

Spacial branch and bound

Problem setup

minimize $\quad f(x)$
subject to $\quad x \in \mathcal{Q}_{\text{init}}$

- f can be nonconvex
- $\mathcal{Q}_{\text{init}}$ is a n-dimensional rectangle

For any rectangle $\mathcal{Q} \subseteq \mathcal{Q}_{\text{init}}$ we define

$$\Phi(\mathcal{Q}) = \min_{x \in \mathcal{Q}} f(x)$$

Global optimal value

$$f(x^\star) = \Phi(\mathcal{Q}_{\text{init}})$$

Lower and upper bounds

Lower and upper bound functions
(they must be cheap to compute)

$$\Phi_{\text{lb}}(\mathcal{Q}) \leq \Phi(\mathcal{Q}) \leq \Phi_{\text{ub}}(\mathcal{Q})$$

Lower and upper bounds

Lower and upper bound functions

(they must be cheap to compute)

$$\Phi_{\mathrm{lb}}(\mathcal{Q}) \leq \Phi(\mathcal{Q}) \leq \Phi_{\mathrm{ub}}(\mathcal{Q})$$

Assumption

bounds must become tight as rectangles shrink

$\forall \epsilon > 0, \exists \delta > 0$ such that $\forall \mathcal{Q} \subseteq \mathcal{Q}_{\mathrm{init}}$

$$\mathrm{size}(\mathcal{Q}) \leq \delta \quad \Longrightarrow \quad \Phi_{\mathrm{ub}}(\mathcal{Q}) - \Phi_{\mathrm{lb}}(\mathcal{Q}) \leq \epsilon$$

where $\mathrm{size}(\mathcal{Q})$ is the longest edge of $\mathcal{Q}$

Branch and bound algorithm

Iterations

1. **Branch**: create/refine the partition

$$\mathcal{Q}_{\text{init}} = \cup_i \mathcal{Q}_i, \quad \cap_i \mathcal{Q}_i = \emptyset$$

2. **Bound:**
   - Compute **lower** and **upper bounds**

$$L_i = \Phi_{\text{lb}}(\mathcal{Q}_i), \quad U_i = \Phi_{\text{ub}}(\mathcal{Q}_i), \quad \forall i$$

   - Update global lower bounds on $f(x^\star)$

$$L = \min_i \{L_i\}, \quad U = \min_i \{U_i\}$$

3. If $U - L \leq \epsilon$, **break**

Branch and bound algorithm

Iterations

1. **Branch**: create/refine the partition

$$\mathcal{Q}_{\text{init}} = \cup_i \mathcal{Q}_i, \quad \cap_i \mathcal{Q}_i = \emptyset$$

2. **Bound:**
   - Compute **lower** and **upper bounds**

$$L_i = \Phi_{\text{lb}}(\mathcal{Q}_i), \quad U_i = \Phi_{\text{ub}}(\mathcal{Q}_i), \quad \forall i$$

   - Update global lower bounds on $f(x^\star)$

$$L = \min_i \{L_i\}, \quad U = \min_i \{U_i\}$$

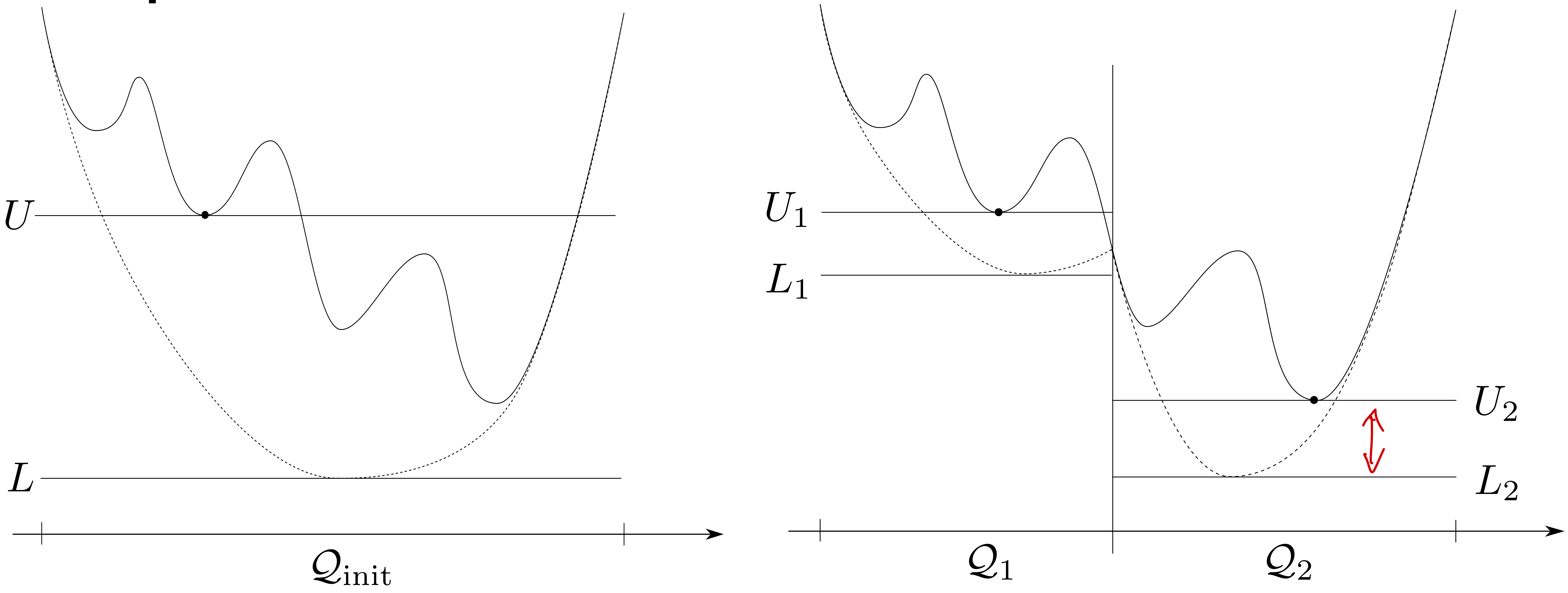
3. If $U - L \leq \epsilon$, **break**

Remarks

- No need to make progress at every iterations
- Partitioning can be uneven

Branch and bound

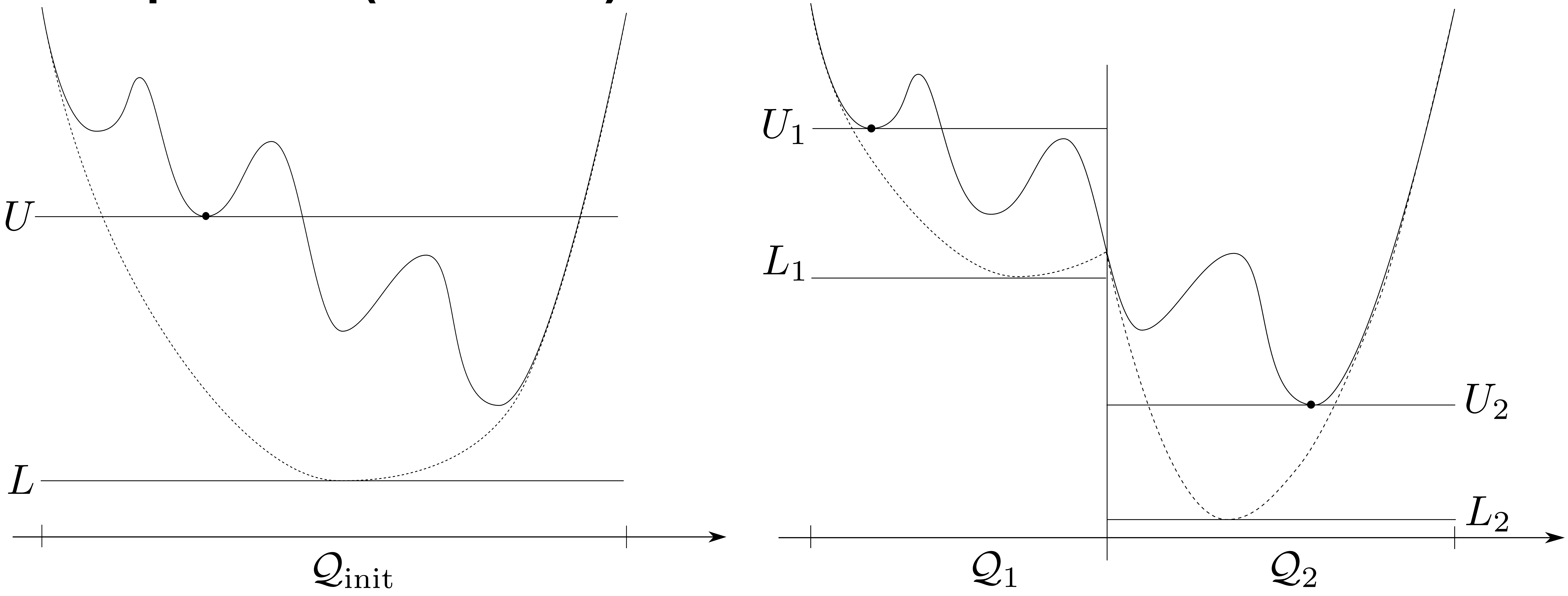
Example in 1D



What does it say about L? And about U?

Branch and bound

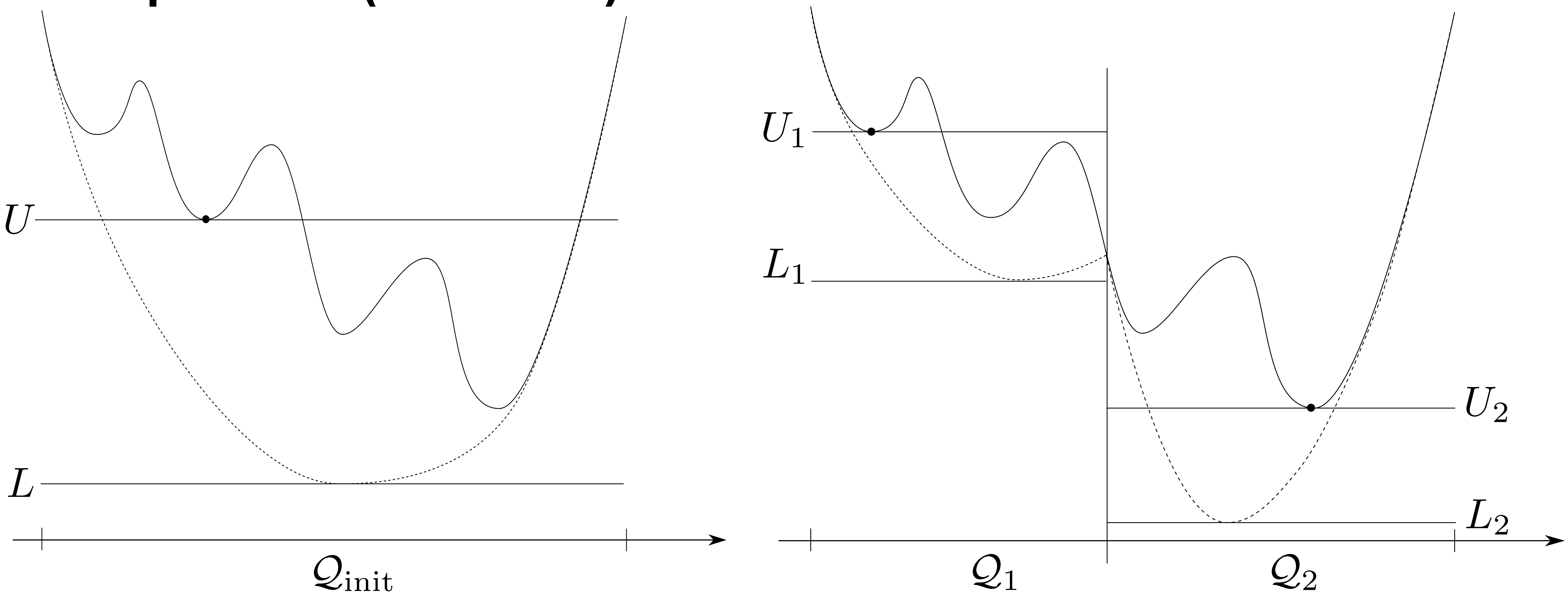
Example in 1D (continued)



What does it say about L? And about U?

Branch and bound

Example in 1D (continued)

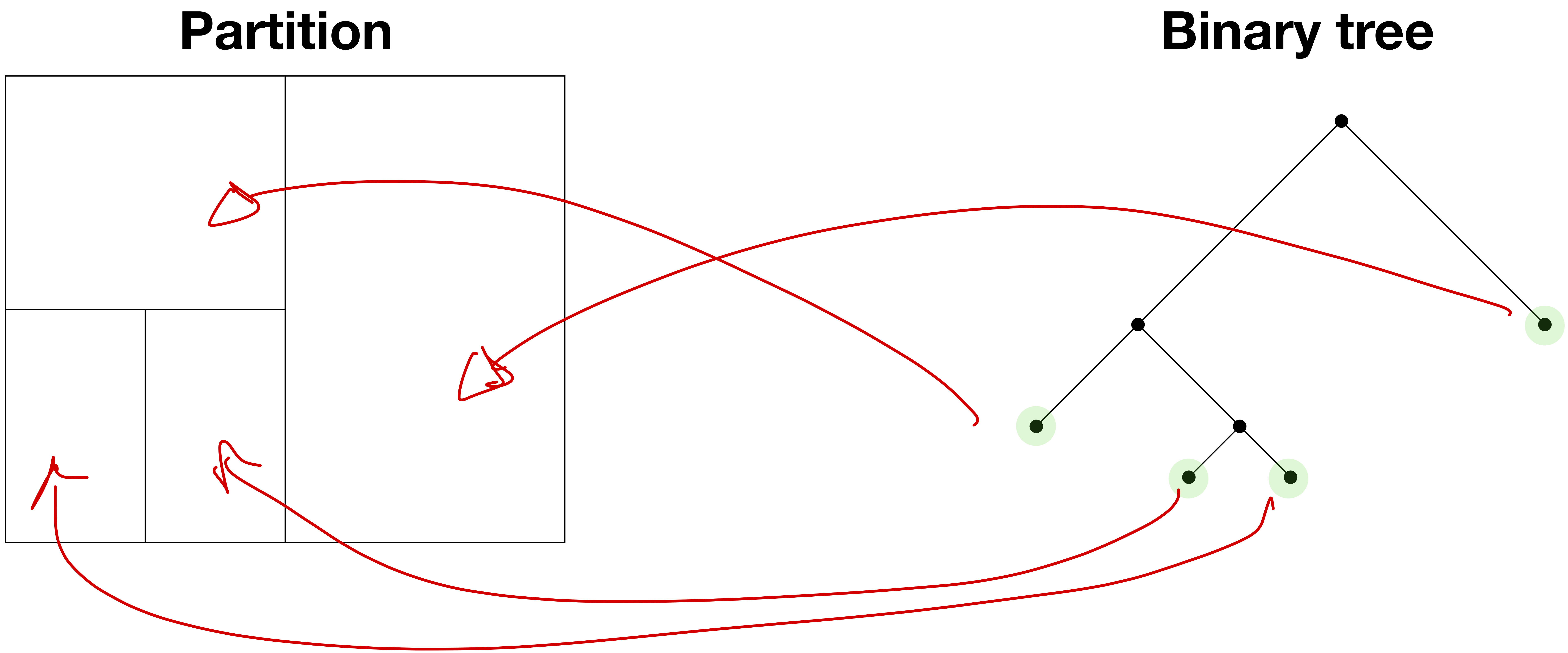


What does it say about L? And about U?

We can assume w.l.o.g. that U is **nonincreasing** and L **nondecreasing**

Branch and bound

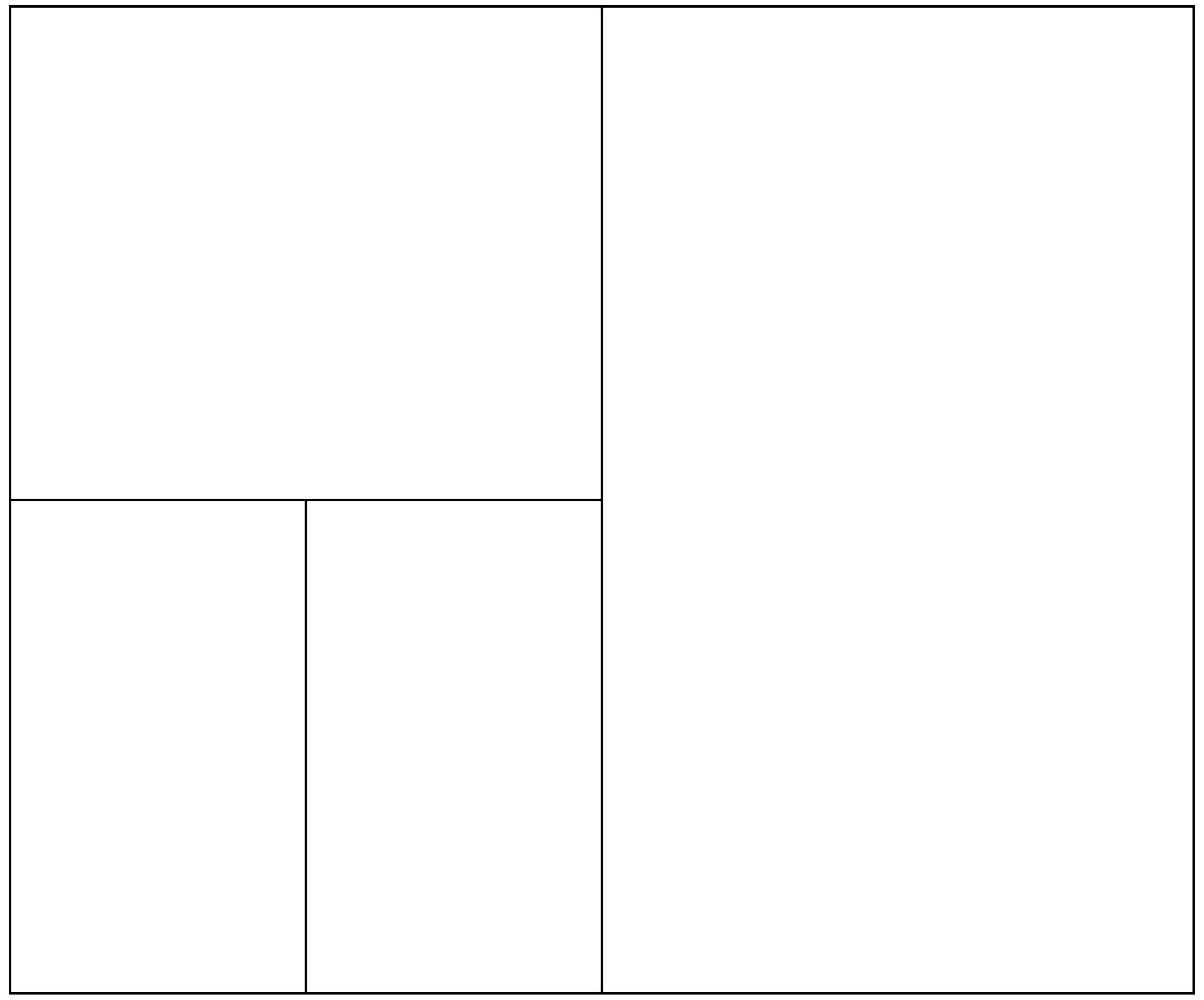
Example in 2D



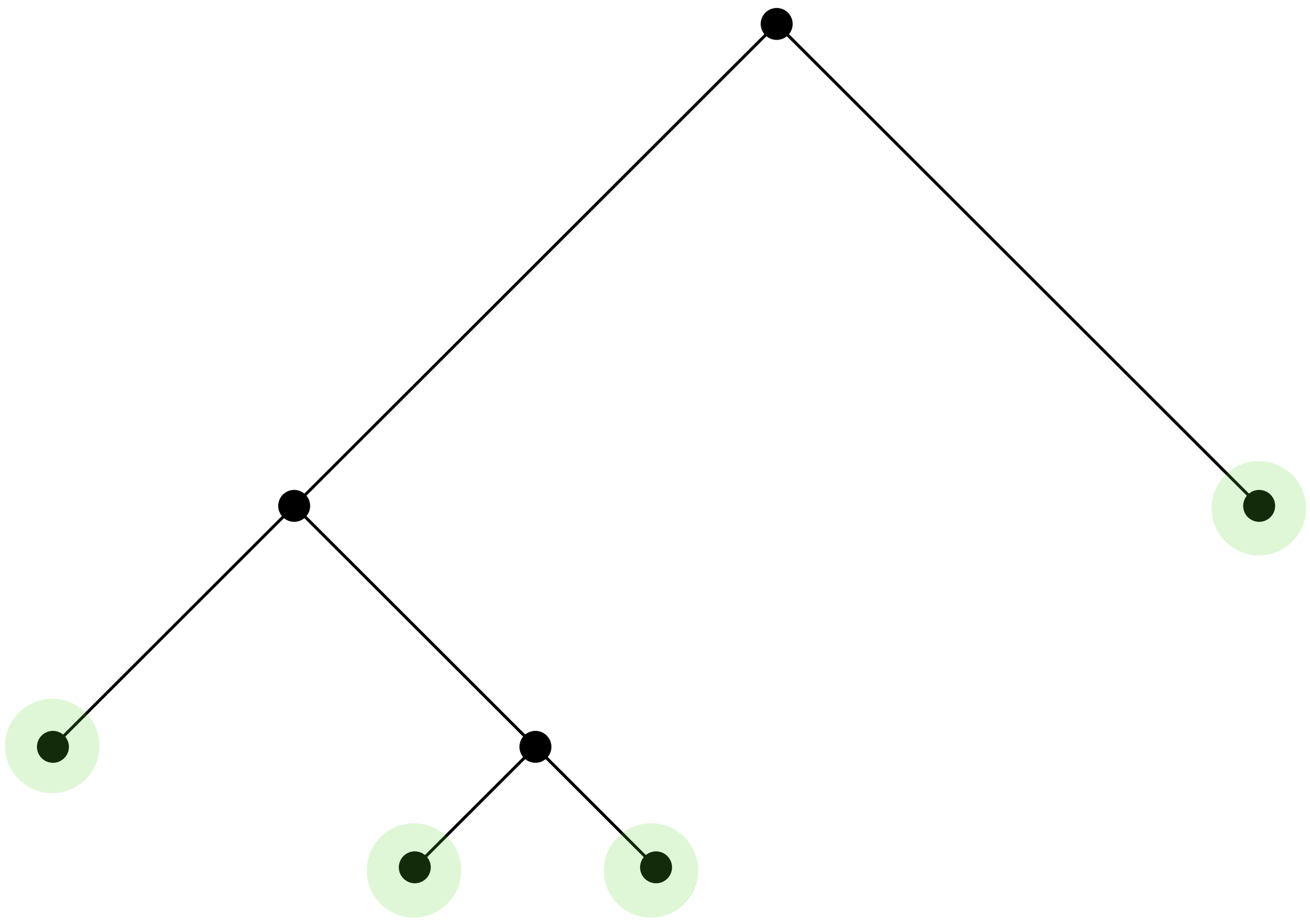
Branch and bound

Example in 2D

Partition



Binary tree



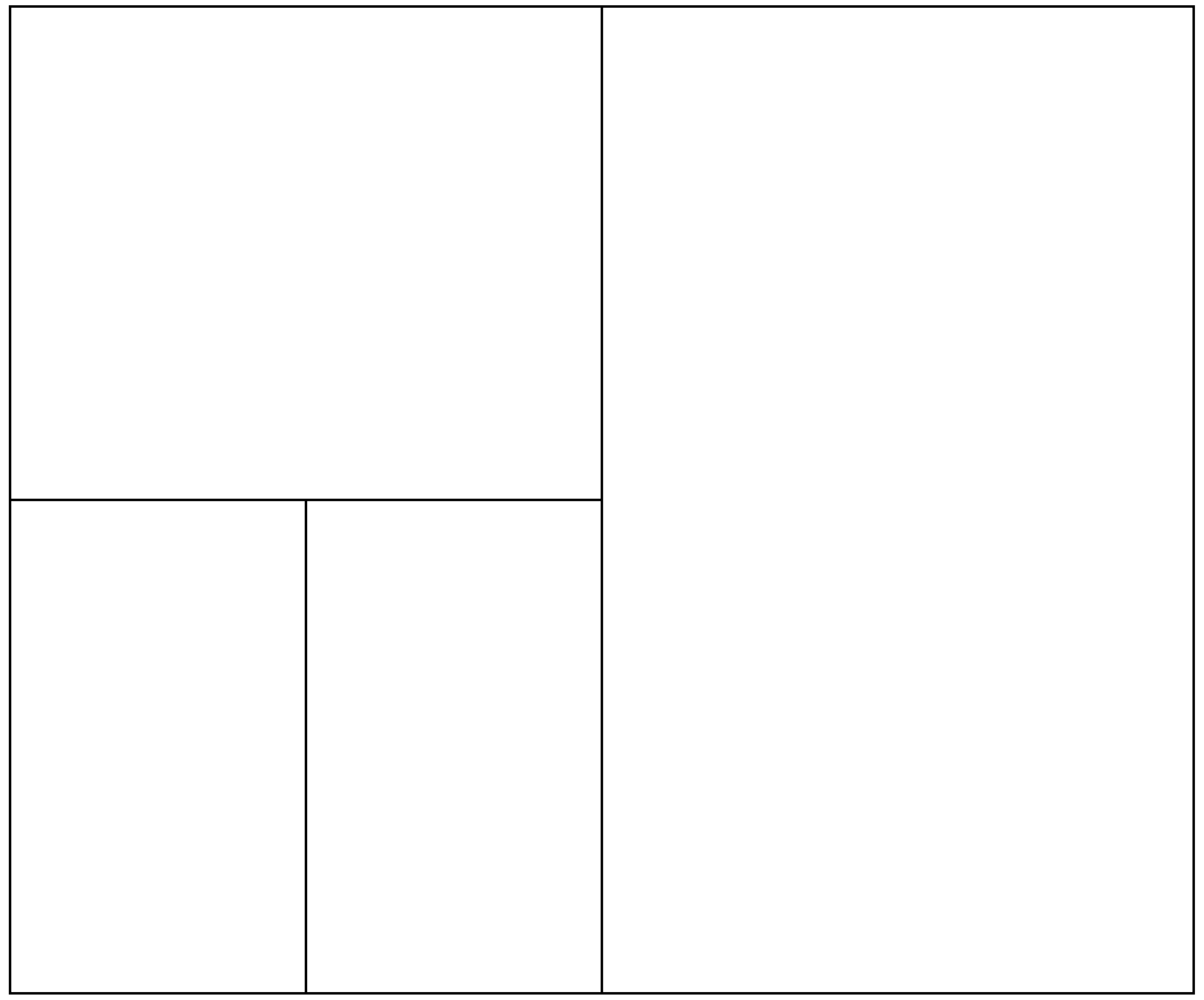
At each step we have a **binary tree**

Children correspond to subregions formed by splitting parents

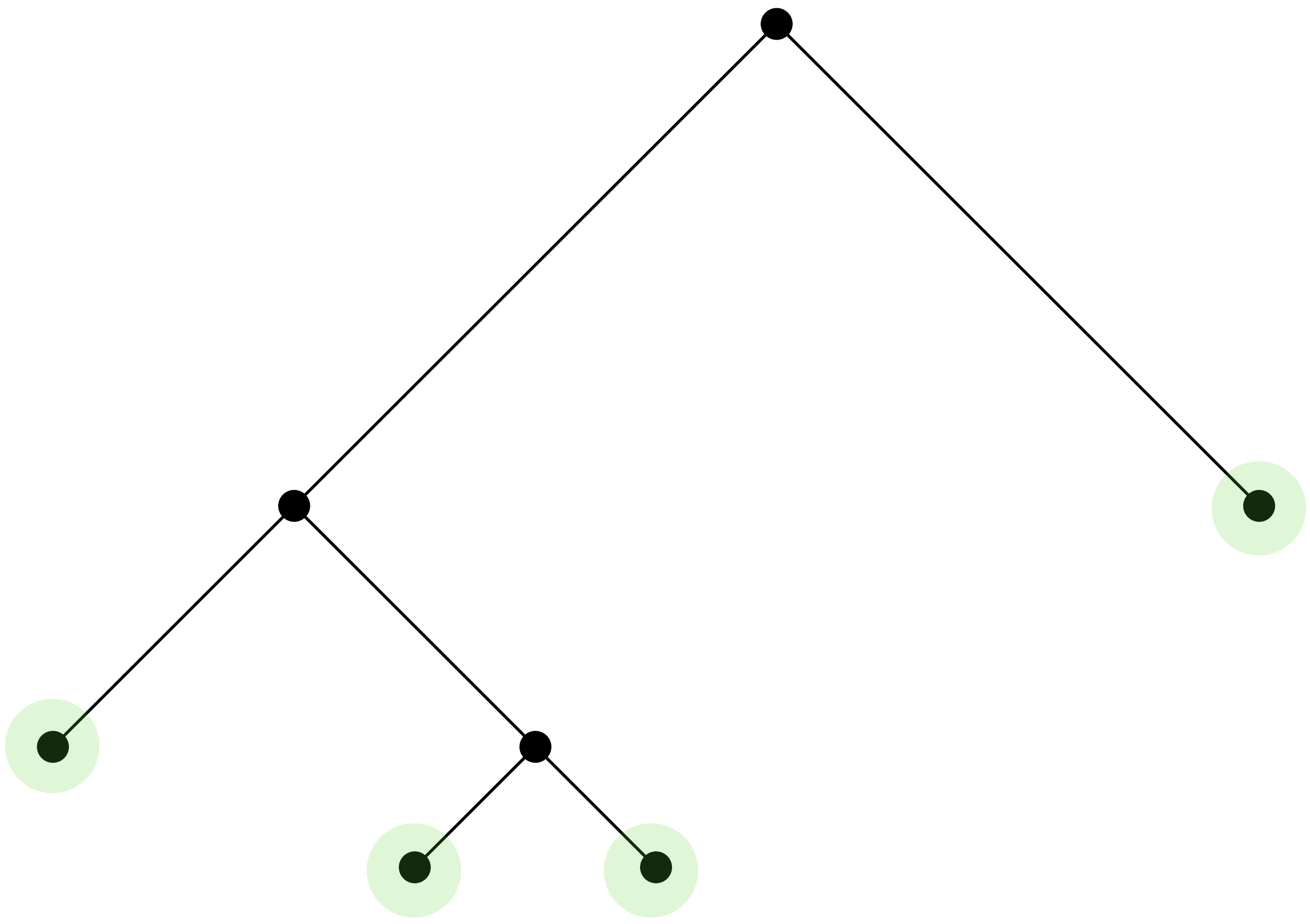
Branch and bound

Example in 2D

Partition



Binary tree



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

Children correspond to subregions formed by splitting parents

Leaves give the current partition of $\mathcal{Q}_{\text{init}}$

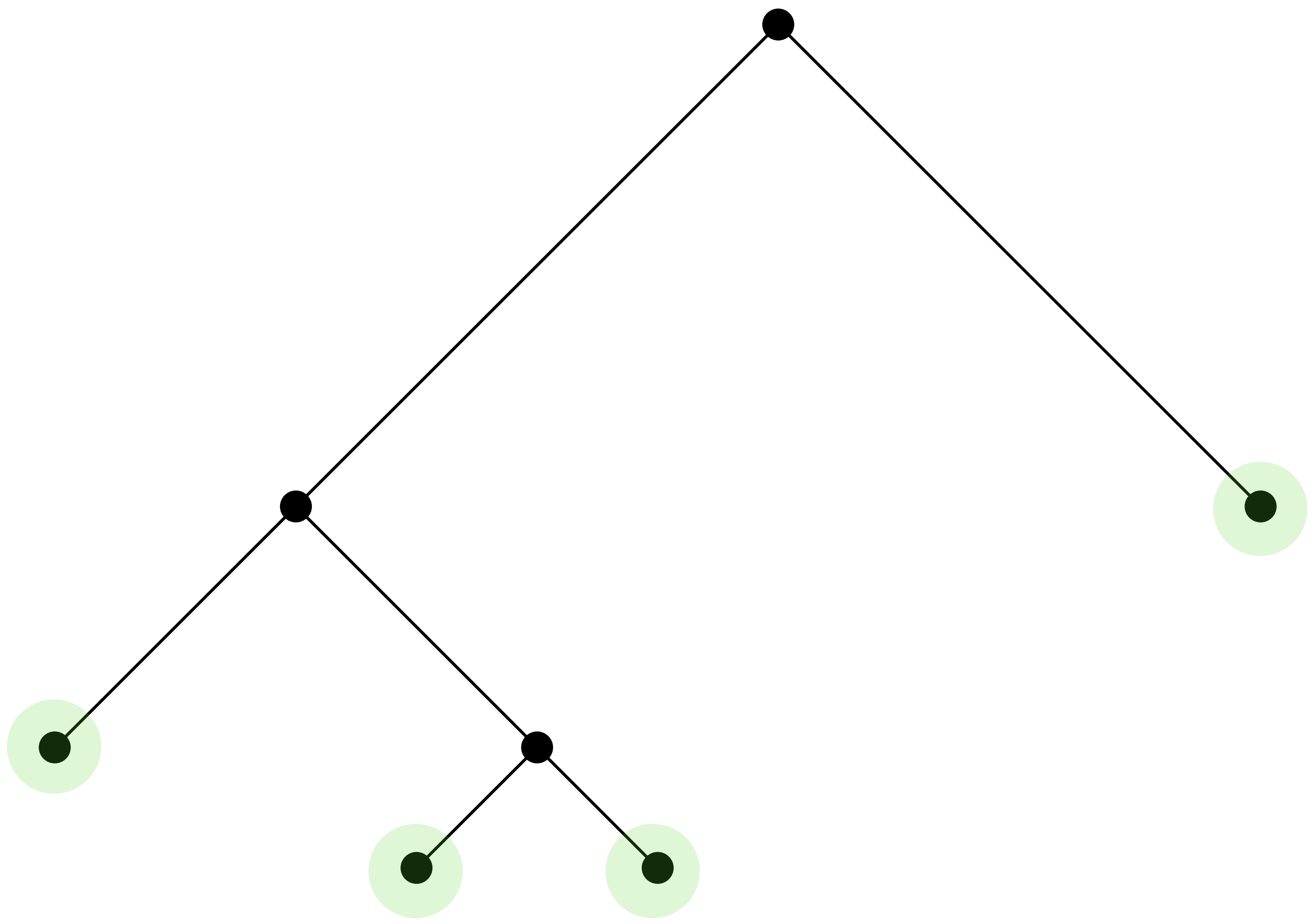
Branch and bound

certify optimality ⟶ $L \leq f(x^\star) \leq U$ ⟵ return point "incumbent"

Branch and bound

certify optimality → $L \leq f(x^\star) \leq U$ ← return point "incumbent"

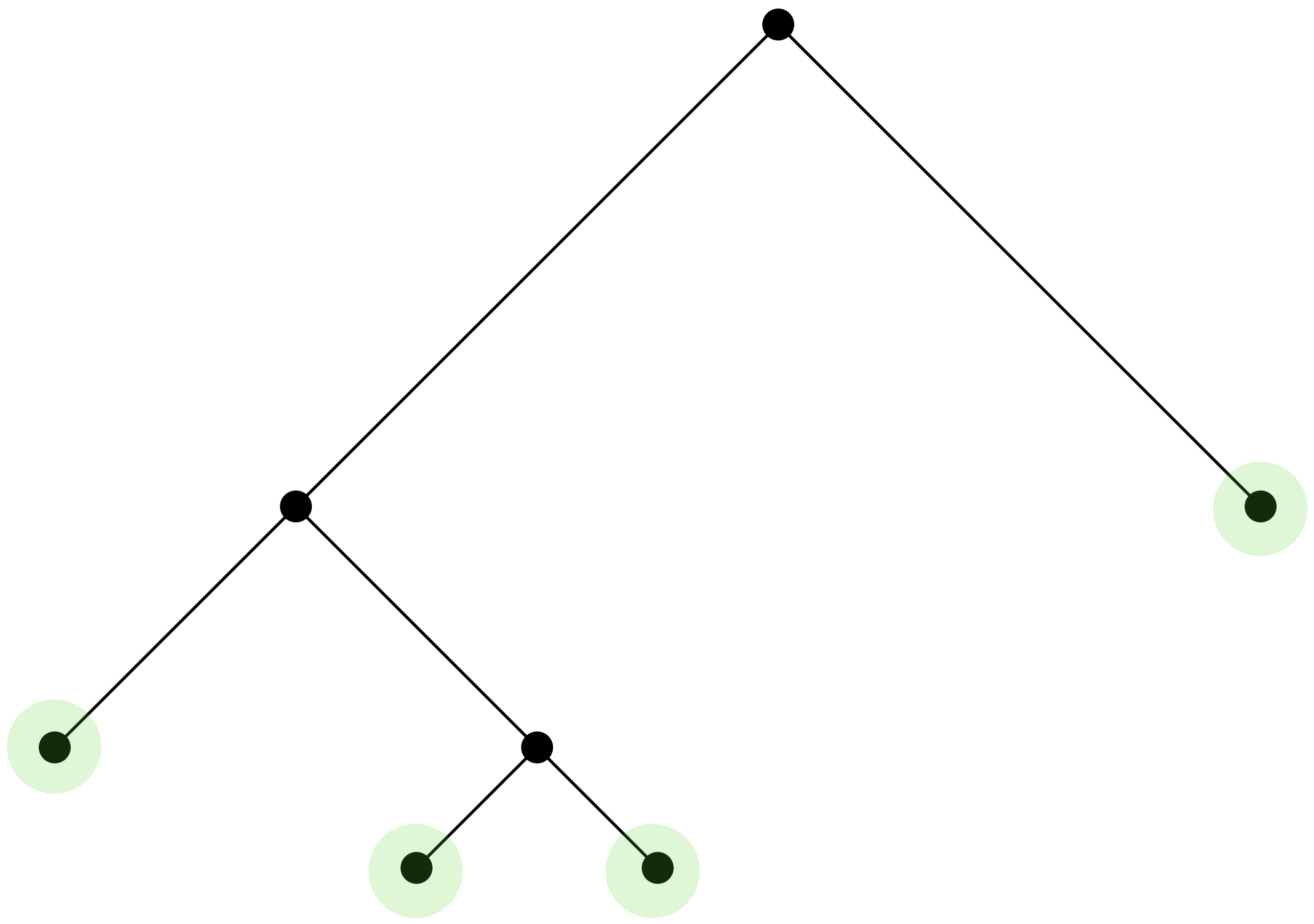
Partition = Leaves



Branch and bound

certify optimality ⟶ $L \leq f(x^\star) \leq U$ ⟵ return point "incumbent"

Partition = Leaves



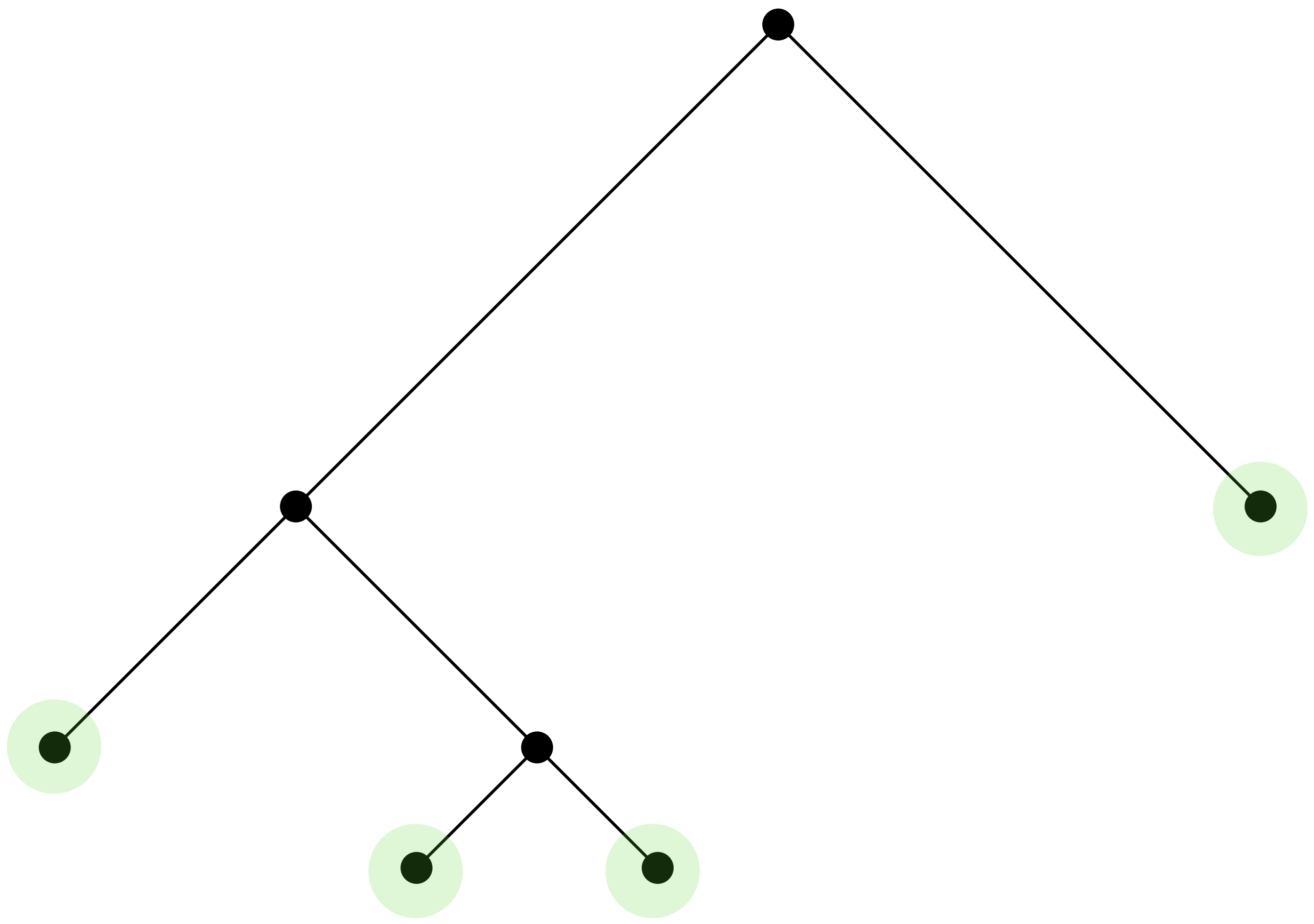
Optimality certificate in nonconvex optimization

- Partition $\mathcal{Q}_{\mathrm{init}} = \cup_i \mathcal{Q}_i$
- Bounds $(L_i, U_i) \quad \forall i$

Branch and bound

certify optimality → $L \leq f(x^\star) \leq U$ ← return point "incumbent"

Partition = Leaves



Optimality certificate in nonconvex optimization

- Partition $\mathcal{Q}_{\rm init} = \cup_i \mathcal{Q}_i$
- Bounds $(L_i, U_i) \quad \forall i$

Optimality certificate in convex optimization

Dual variables and cost

Branching rules

Branching decisions

- Which rectangle $\mathcal{Q}_i$ to split
- Which edge (variable) to split
- Where to split (what value of the variable)

Goal

Get tight bounds as quickly as possible

They can **dramatically affect performance**

Branching rules

Branching decisions

- Which rectangle $\mathcal{Q}_i$ to split
- Which edge (variable) to split
- Where to split (what value of the variable)

Goal

Get tight bounds as quickly as possible

They can **dramatically affect performance**

Example heuristic (best-bound search)

- **Optimism**: split $\mathcal{Q}_i$ with lowest L_i
- **Greed**: split along coordinate i with greatest uncertainty (along longest endge)
- **Hope**: split at value x_i where $f(x_i) = U_i$

Pruning

Key performance component

$$\min_i L_i \leq f(x^\star) \leq \min_i U_i$$

$\mathcal{Q}_i$ is **active** if $L_i \leq \min_i U_i$

Otherwise it is **inactive** ($x^\star \notin \mathcal{Q}_i$)
and we can **prune** it

Pruning

Key performance component

$$\min_i L_i \leq f(x^\star) \leq \min_i U_i$$

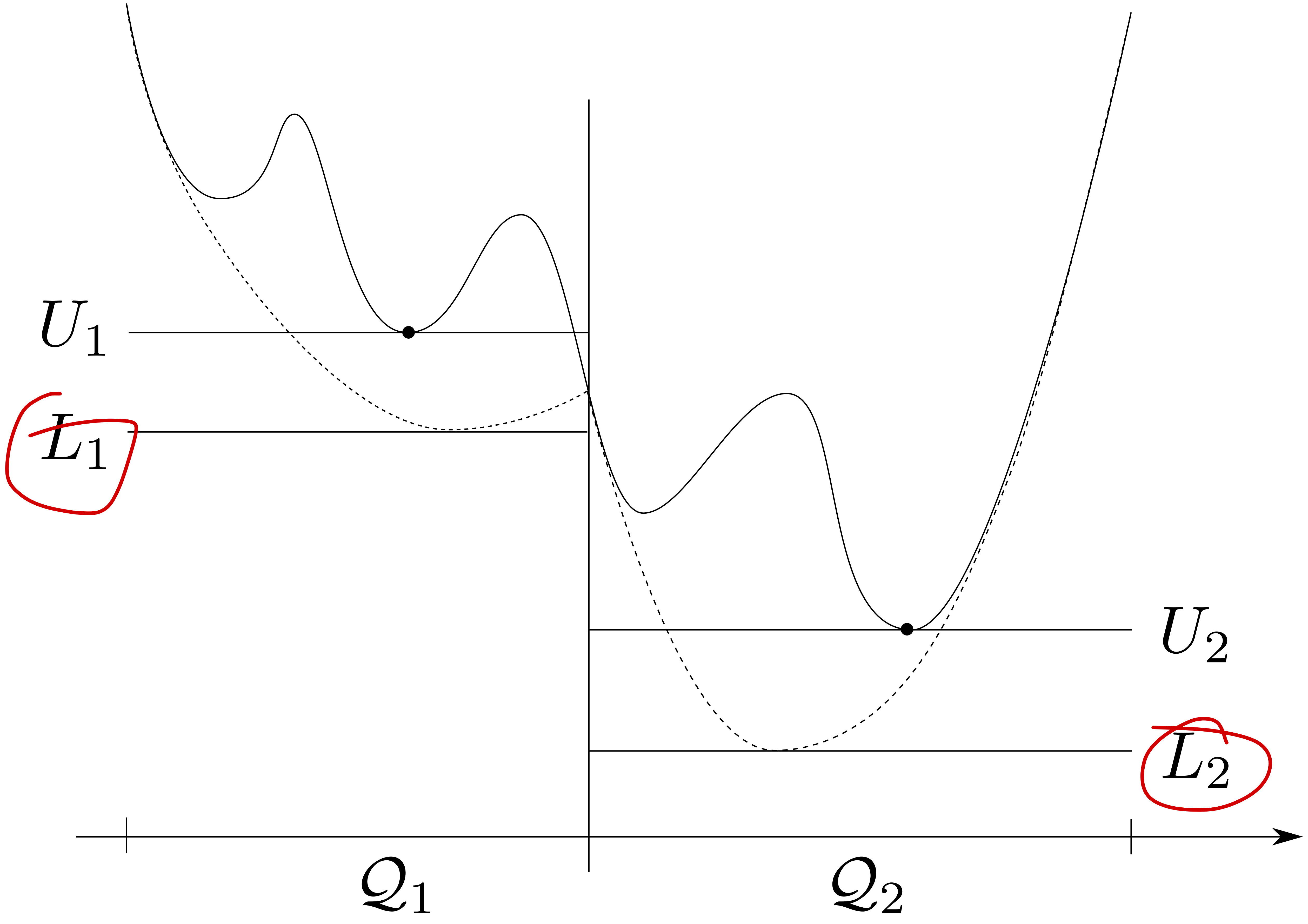
$\mathcal{Q}_i$ is **active** if $L_i \leq \min_i U_i$

Otherwise it is **inactive** ($x^\star \notin \mathcal{Q}_i$) and we can **prune** it

Questions

What is $\mathcal{Q}_1$? active/inactive

What is $\mathcal{Q}_2$? active/inactive



$L = L_2$

$U = U_2$

Convergence analysis

Bounds and volume decrease

Assumption
bounds become tight as rectangles shrink

$\forall \epsilon > 0, \exists \delta > 0$ such that $\forall \mathcal{Q} \subseteq \mathcal{Q}_{\text{init}}$

$$\text{size}(\mathcal{Q}) \leq \delta \quad \Longrightarrow \quad \Phi_{\text{ub}}(\mathcal{Q}) - \Phi_{\text{lb}}(\mathcal{Q}) \leq \epsilon$$

where $\text{size}(\mathcal{Q})$ is the diameter (longest edge of $\mathcal{Q}$)

Bounds and volume decrease

Assumption
bounds become tight as rectangles shrink

$\forall \epsilon > 0, \exists \delta > 0$ such that $\forall \mathcal{Q} \subseteq \mathcal{Q}_{\text{init}}$

$$\text{size}(\mathcal{Q}) \leq \delta \quad \implies \quad \Phi_{\text{ub}}(\mathcal{Q}) - \Phi_{\text{lb}}(\mathcal{Q}) \leq \epsilon$$

where $\text{size}(\mathcal{Q})$ is the diameter (longest edge of $\mathcal{Q}$)

Volume decrease

At iteration k we have the partition $\mathcal{L}_k = \{\mathcal{Q}_1, \ldots, \mathcal{Q}_k\}$

$$\min_{\mathcal{Q} \in \mathcal{L}_k} \text{vol}(\mathcal{Q}) \leq \frac{\text{vol}(\mathcal{Q}_{\text{init}})}{k}$$

Bounding the condition number

Condition number

For a rectancle $\mathcal{Q} = [l_1, u_1] \times \cdots \times [l_n, u_n]$

$$\mathrm{cond}(\mathcal{Q}) = \frac{\max_i (u_i - l_i)}{\min_i (u_i - l_i)}$$

Bounding the condition number

Condition number

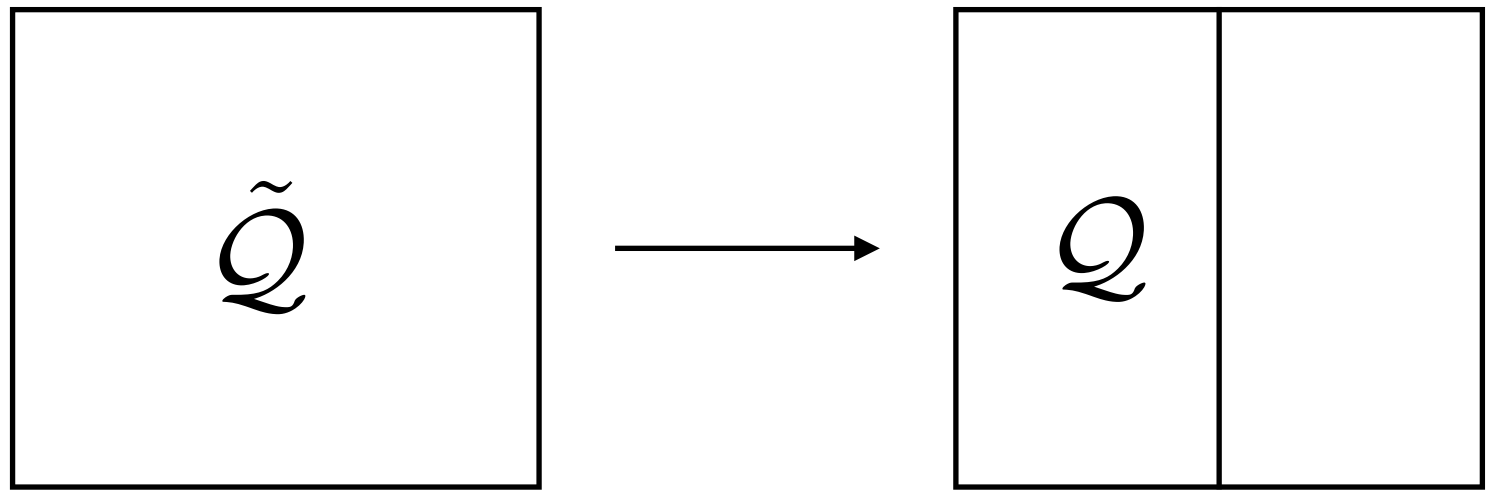
For a rectancle $\mathcal{Q} = [l_1, u_1] \times \cdots \times [l_n, u_n]$

$$\mathrm{cond}(\mathcal{Q}) = \frac{\max_i (u_i - l_i)}{\min_i (u_i - l_i)}$$

Worst-case

If we split $\tilde{Q}$ along longest edge in half, we have

$$\mathrm{cond}(\mathcal{Q}) \leq \max\{\mathrm{cond}(\tilde{Q}), 2\}$$



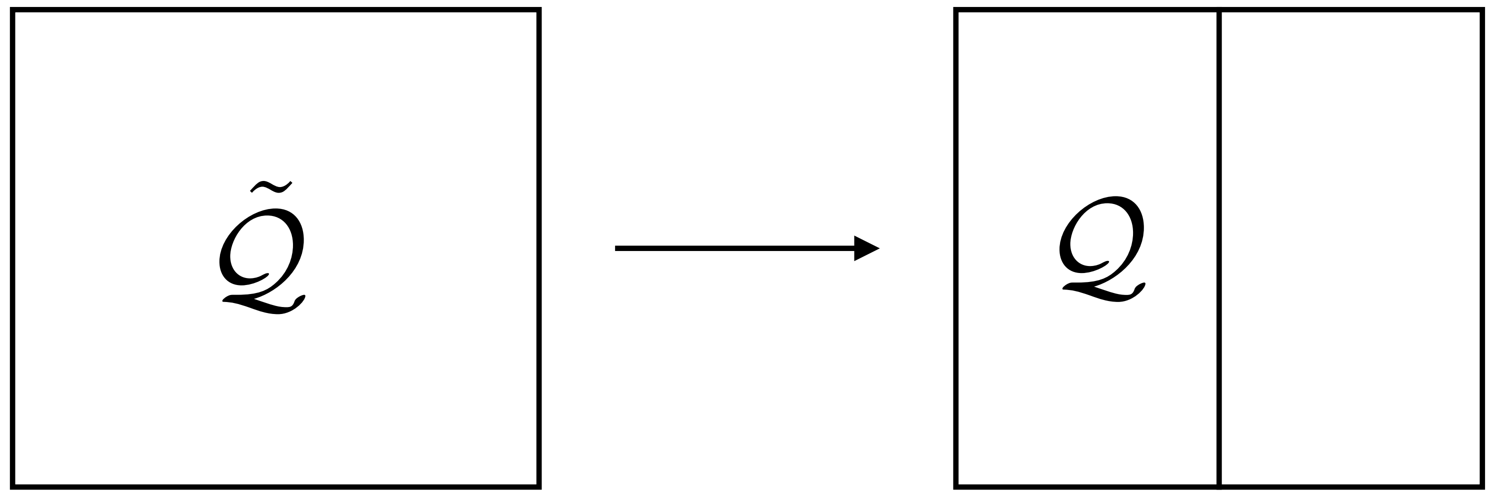
Bounding the condition number

Condition number

For a rectancle $\mathcal{Q} = [l_1, u_1] \times \cdots \times [l_n, u_n]$

$$\operatorname{cond}(\mathcal{Q}) = \frac{\max_i(u_i - l_i)}{\min_i(u_i - l_i)}$$

Worst-case



If we split $\tilde{Q}$ along longest edge in half, we have

$$\operatorname{cond}(\mathcal{Q}) \leq \max\{\operatorname{cond}(\tilde{Q}), 2\}$$

Note: we can bound $\operatorname{cond}(\mathcal{Q})$ also if we do not split in half, by using other rules (e.g., cycling over the variables)

Small volume implies small size

$$\mathrm{vol}(\mathcal{Q}) = \Pi_i(u_i - l_i)$$

$$\geq \max_i(u_i - l_i)\left(\min_i(u_i - l_i)\right)^{n-1}$$

$$= \frac{\mathrm{size}(\mathcal{Q})^n}{\mathrm{cond}(\mathcal{Q})^{n-1}}$$ (multiply/divide by $(\max_i(u_i - l_i))^{n-1}$)

$$\geq \left(\frac{\mathrm{size}(\mathcal{Q})}{\mathrm{cond}(\mathcal{Q})}\right)^n$$ ($\mathrm{cond}(\mathcal{Q}) \geq 1$)

Small volume implies small size

$$\text{vol}(\mathcal{Q}) = \Pi_i(u_i - l_i)$$

$$\geq \max_i(u_i - l_i)\left(\min_i(u_i - l_i)\right)^{n-1}$$

$$= \frac{\text{size}(\mathcal{Q})^n}{\text{cond}(\mathcal{Q})^{n-1}}$$ (multiply/divide by $(\max_i(u_i - l_i))^{n-1}$)

$$\geq \left(\frac{\text{size}(\mathcal{Q})}{\text{cond}(\mathcal{Q})}\right)^n$$ ($\text{cond}(\mathcal{Q}) \geq 1$)

Therefore, $\text{size}(\mathcal{Q}) \leq \text{vol}(\mathcal{Q})^{1/n}\,\text{cond}(\mathcal{Q})$

Small volume implies small size

$$\begin{aligned}
\mathrm{vol}(\mathcal{Q}) &= \Pi_i (u_i - l_i) \\
&\geq \max_i (u_i - l_i) \left(\min_i (u_i - l_i) \right)^{n-1} \\
&= \frac{\mathrm{size}(\mathcal{Q})^n}{\mathrm{cond}(\mathcal{Q})^{n-1}} && \text{(multiply/divide by } (\max_i (u_i - l_i))^{n-1}) \\
&\geq \left(\frac{\mathrm{size}(\mathcal{Q})}{\mathrm{cond}(\mathcal{Q})} \right)^n && (\mathrm{cond}(\mathcal{Q}) \geq 1)
\end{aligned}$$

Therefore, $\mathrm{size}(\mathcal{Q}) \leq \mathrm{vol}(\mathcal{Q})^{1/n} \, \mathrm{cond}(\mathcal{Q})$

Since $\mathrm{cond}(\mathcal{Q})$ is bounded, then we have

$$\mathrm{vol}(Q) \leq \gamma \quad \implies \quad \mathrm{size}(\mathcal{Q}) \leq \delta$$

Upper and lower bounds convergence

Small volume implies small size

$\forall \delta > 0, \exists \gamma > 0$ such that $\forall \mathcal{Q} \subseteq \mathcal{Q}_{\text{init}}$

$$\text{vol}(Q) \leq \gamma \quad \Longrightarrow \quad \text{size}(\mathcal{Q}) \leq \delta$$

Upper and lower bounds convergence

Small volume implies small size

$\forall \delta > 0, \exists \gamma > 0$ such that $\forall \mathcal{Q} \subseteq \mathcal{Q}_{\text{init}}$

$$\text{vol}(Q) \leq \gamma \quad \implies \quad \text{size}(\mathcal{Q}) \leq \delta$$

Hence, (roughly)

k large $\implies \exists \mathcal{Q} \in \mathcal{L}_k, \quad \text{vol}(\mathcal{Q}) \leq \gamma \quad \implies \quad \text{size}(\mathcal{Q}) \leq \delta = \eta/2$

$\implies \text{size}(\tilde{Q}) \leq \eta,$ (parent)

$\implies \Phi_{\text{ub}}(\tilde{\mathcal{Q}}) - \Phi_{\text{lb}}(\tilde{\mathcal{Q}}) \leq \epsilon$

$\implies U - L \leq \epsilon$ ⟶

When $\mathcal{Q}$ was added to $\mathcal{L}_k$ ($\tilde{\mathcal{Q}}$ was split), the algorithm should have terminated (best-bound heuristic) ■

Branch and bound convergence

It converges but we can show
all worst-case rates are exponential

We cannot hope to have non-exponential
worst-case performance

(unless $\mathcal{P} = \mathcal{NP}$)

Mixed-boolean convex optimization

Mixed-boolean convex optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x, z) \leq 0 \\ & z \in \{0, 1\}^n \end{array}$$

$x \in \mathbf{R}^p$ is the *continuous variable*

$z \in \{0, 1\}^n$ is the *boolean variable*

f and g are convex in x and z

For each fixed z, the reduced problem in x is **convex**

Global solution methods

Brute force

Solve problem for all 2^n possible values of $z \in \{0, 1\}^n$ (it blows up for $n \geq 20$)

Branch and bound

worst-case: we end up solving all 2^n convex problems

hope: it works better for our problem

Lower bounds via convex relaxations

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & 0 \leq z \leq 1 \end{array}$$

- Convex problem in x, z (easy)
- Optimal value is $L \leq f(x^\star)$ (lower bound)
- L can be $+\infty$ (original problem infeasible)

Upper bounds

Round (simplest): round each relaxed boolean variable $z_i^\star$ to 0 or 1

Round and polish: round each relaxed boolean variable and solve resulting problem in x

Upper bounds

Round (simplest): round each relaxed boolean variable $z_i^\star$ to 0 or 1

Round and polish: round each relaxed boolean variable and solve resulting problem in x

Randomization
- Generate random $z_i \in \{0, 1\}$ with $\mathbf{prob}(z_i = 1) = z_i^\star$
- Solve for x
- Take best result after some samples

Upper bounds

Round (simplest): round each relaxed boolean variable $z_i^\star$ to 0 or 1

Round and polish: round each relaxed boolean variable and solve resulting problem in x

Randomization

- Generate random $z_i \in \{0, 1\}$ with $\mathbf{prob}(z_i = 1) = z_i^\star$
- Solve for x
- Take best result after some samples

Neighborhood search (after rounding)

- Pick z_i and flip its value $0/1$
- Solve for x to polish and get bound
- Iterate over all components of z and take best result

Upper bounds

Round (simplest): round each relaxed boolean variable $z_i^\star$ to 0 or 1

Round and polish: round each relaxed boolean variable and solve resulting problem in x

Randomization

- Generate random $z_i \in \{0, 1\}$ with $\mathbf{prob}(z_i = 1) = z_i^\star$
- Solve for x
- Take best result after some samples

Neighborhood search (after rounding)

- Pick z_i and flip its value $0/1$
- Solve for x to polish and get bound
- Iterate over all components of z and take best result

Remarks

U can be $+\infty$
(we can fail to find a feasible point)

If $U - L \leq \epsilon$
we can quit

Boolean variables branching

Pick and index k and form two subproblems

$$
\begin{aligned}
f_0^\star = \text{minimize} \quad & f(x) \\
\text{subject to} \quad & g(x, z) \leq 0 \\
& z \in \{0, 1\}^n \\
& z_k = 0
\end{aligned}
$$

$$
\begin{aligned}
f_1^\star = \text{minimize} \quad & f(x) \\
\text{subject to} \quad & g(x, z) \leq 0 \\
& z \in \{0, 1\}^n \\
& z_k = 1
\end{aligned}
$$

Boolean variables branching

Pick and index k and form two subproblems

$$f_0^\star = \text{minimize} \quad f(x)$$
$$\text{subject to} \quad g(x,z) \le 0$$
$$z \in \{0,1\}^n$$
$$z_k = 0$$

$$f_1^\star = \text{minimize} \quad f(x)$$
$$\text{subject to} \quad g(x,z) \le 0$$
$$z \in \{0,1\}^n$$
$$z_k = 1$$

Remarks

- Each problem has $n-1$ boolean variables
- Optimal value $f(x^\star) = \min\{f_0^\star, f_1^\star\}$
- We can relax the two problems to obtain lower bounds

Bounds from subproblems

$$f_0^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 0 \end{array}$$

$$f_1^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 1 \end{array}$$

L_q, U_q are the lower, upper bounds for $z_k = q$ with $q = 0, 1$

$$L = \min\{L_0, L_1\} \leq f(x^\star) \leq \min\{U_0, U_1\} = U$$

Bounds from subproblems

$$f_0^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 0 \end{array}$$

$$f_1^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 1 \end{array}$$

L_q, U_q are the lower, upper bounds for $z_k = q$ with $q = 0, 1$

$$L = \min\{L_0, L_1\} \leq f(x^\star) \leq \min\{U_0, U_1\} = U$$

$\geq$ previous lower bound

Bounds from subproblems

$$f_0^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 0 \end{array}$$

$$f_1^\star = \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x,z) \leq 0 \\ & z \in \{0,1\}^n \\ & z_k = 1 \end{array}$$

L_q, U_q are the lower, upper bounds for $z_k = q$ with $q = 0, 1$

$$L = \min\{L_0, L_1\} \leq f(x^\star) \leq \min\{U_0, U_1\} = U$$

$\geq$ previous lower bound

$\leq$ previous upper bound

Boolean branch and bound iterations

1. **Branch**: pick node i and index k
   form subproblems for $z_k = 0$ and $z_k = 1$
2. **Bound:**
   - Compute **lower** and **upper bounds** for $z_k = 0$ and $z_k = 1$
   - Update global lower ~~lower~~ UPPER bounds on $f(x^\star)$
     $$L = \min_i \{L_i\}, \quad U = \min_i \{U_i\}$$
3. If $U - L \leq \epsilon$, **break**

Convergence

(trivial) worst-case
2^n iterations
before $U = L$

Boolean branch and bound iterations

1. **Branch**: pick node i and index k
   form subproblems for $z_k = 0$ and $z_k = 1$
2. **Bound:**
   - Compute **lower** and **upper bounds**
     for $z_k = 0$ and $z_k = 1$
   - Update global lower bounds on $f(x^\star)$
     $$L = \min_i \{L_i\}, \quad U = \min_i \{U_i\}$$
3. If $U - L \leq \epsilon$, **break**

Convergence

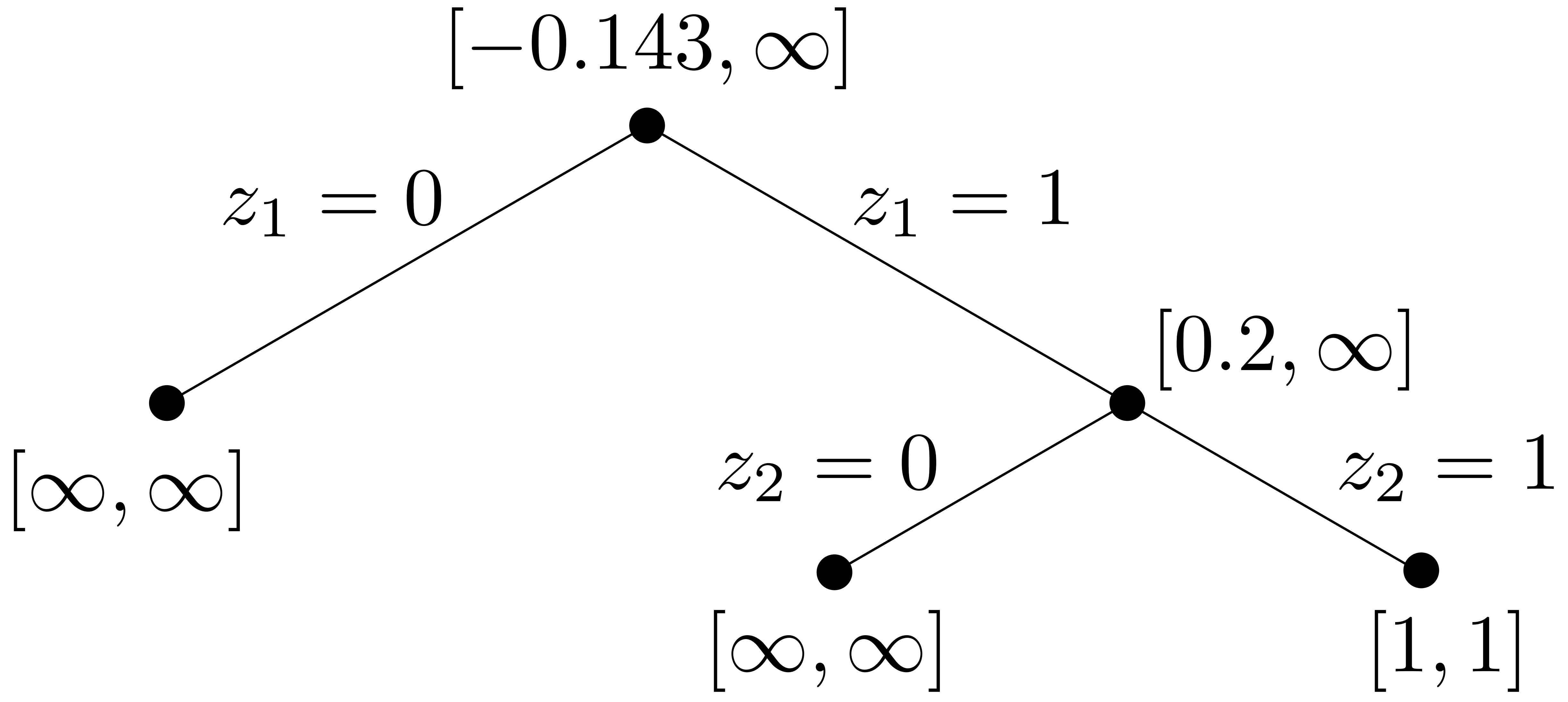
(trivial) worst-case
2^n iterations
before $U = L$

Remarks

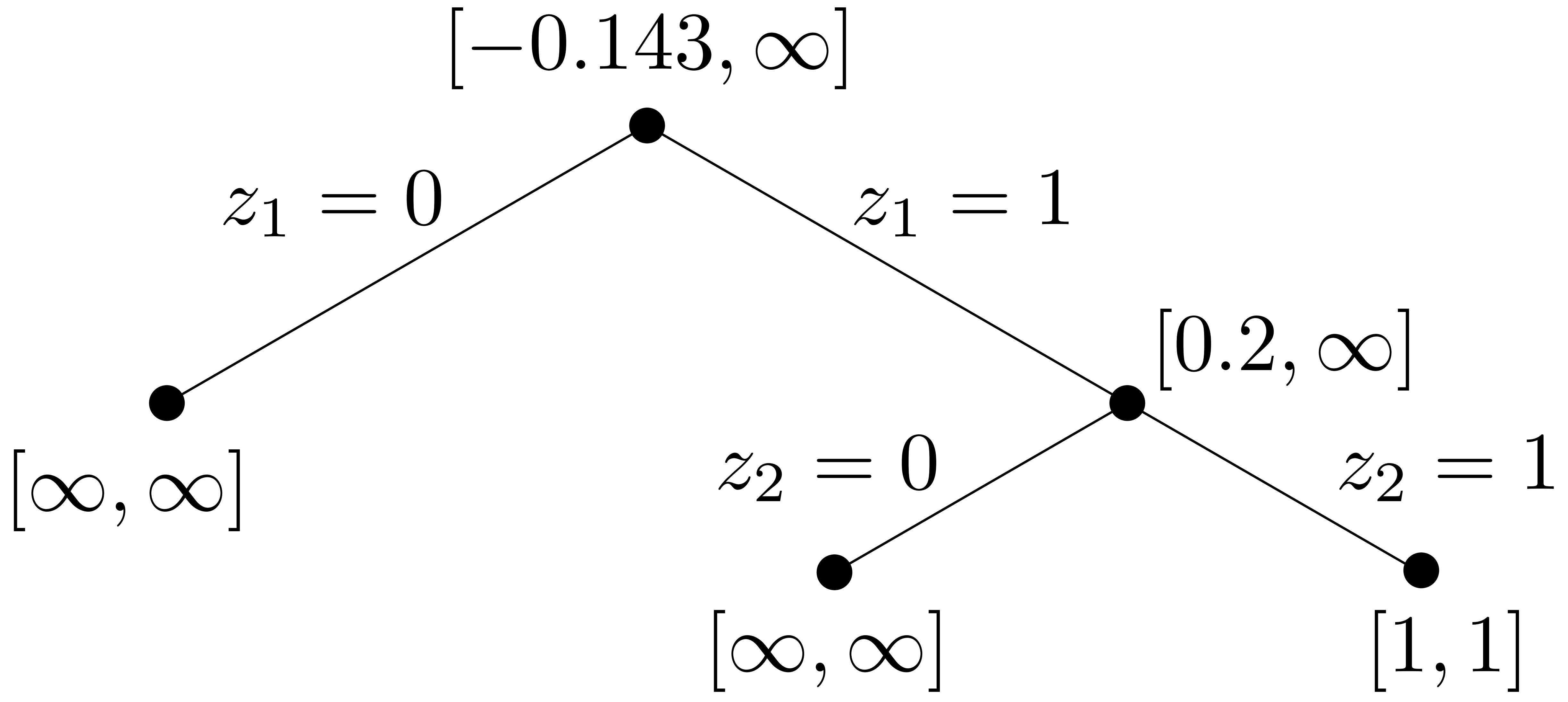
- Pruning works in the same way, i.e., if $L_i > U$
- Best-bound heuristic very common
- Variable k selection examples:
  - "least ambivalent": $z_k^\star = 0$ or 1 and largest Lagrange multiplier
  - "most ambivalent": $|z_k^\star - 1/2|$ is minimum

Boolean toy example

$$\begin{array}{ll} \text{minimize} & c^T z \\ \text{subject to} & Az \leq b \\ & z \in \{0,1\}^3 \end{array}$$



Boolean toy example



$$
\begin{array}{ll}
\text{minimize} & c^T z \\
\text{subject to} & Az \leq b \\
 & z \in \{0, 1\}^3
\end{array}
$$

Questions

- How much work (LPs) have we saved? $2^3 - 5$
- What happens if $L_i = \infty$? INFEAS
- What happens if $U_i = \infty$? CAN'T GET FEASIBLE POINT
- What if you get to a node where the relaxed $z^\star \in \{0, 1\}^3$? OPTIMAL SOL

Practical considerations

Subproblem solutions are independent

We can exploit parallelism on multiple cores or computing nodes

Practical considerations

Subproblem solutions are independent

We can exploit parallelism on multiple cores or computing nodes

Subproblems can be very similar

(feasible region variations)

We can warm start the subproblem solver

Practical considerations

Subproblem solutions are independent

We can exploit parallelism on multiple cores or computing nodes

Subproblems can be very similar

(feasible region variations)

We can warm start the subproblem solver

Which algorithms would you choose for convex subproblems?

What if you have LP subproblems? DUAL SIMPLEX

Practical considerations

Subproblem solutions are independent
We can exploit parallelism on multiple cores or computing nodes

Subproblems can be very similar
(feasible region variations)
We can warm start the subproblem solver

Which algorithms would you choose for convex subproblems?
What if you have LP subproblems?

Integer linear programs are much easier than integer convex
Tailored software can greatly speedup the solution

Cardinality minimization

Minimum cardinality example

Find sparsest x satisfying linear inequalities

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

Minimum cardinality example

Find sparsest x satisfying linear inequalities

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

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

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

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

- 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

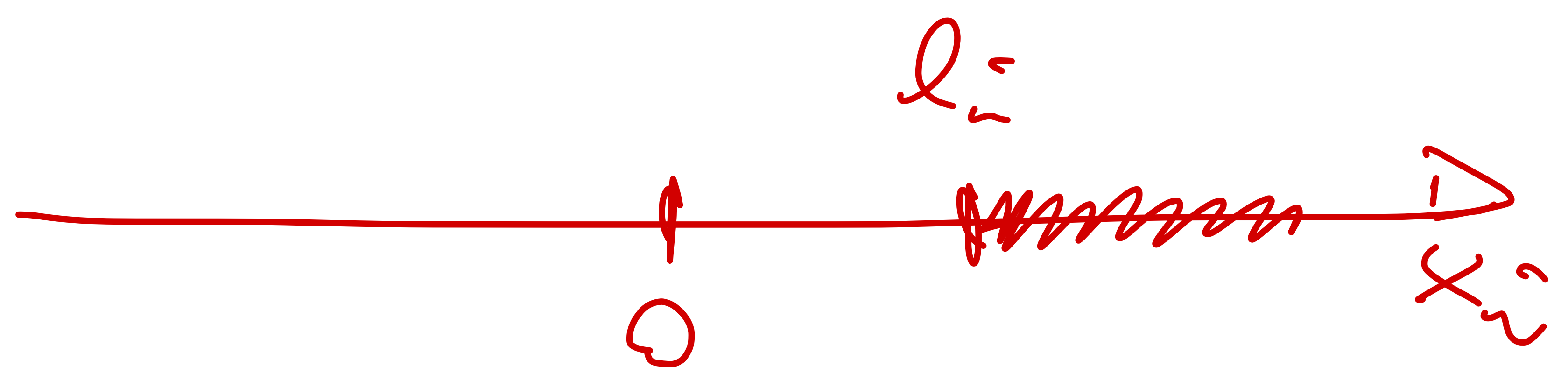
$$\begin{array}{ll} \text{minimize} & x_i \\ \text{subject to} & Ax \leq b \end{array}$$

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

Relaxed problem

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

Cardinality problem relaxation

Relaxed problem

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

Assuming $l_i < 0$ and $u_i > 0$, it is equivalent to

minimize	$\sum_{i=1}^{n} (1/u_i)(x_i)_+ + (-1/l_i)(x_i)_-$
subject to	$Ax \leq b$

Asymmetric weighted 1-norm objective

Cardinality problem relaxation

Relaxed problem

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

Assuming $l_i < 0$ and $u_i > 0$, it is equivalent to

$$
\begin{array}{ll}
\text{minimize} & \sum_{i=1}^n (1/u_i)(x_i)_+ + (-1/l_i)(x_i)_- \\
\text{subject to} & Ax \leq b
\end{array}
$$

Asymmetric weighted 1-norm objective

If $u_i = \bar{u} = \bar{l} = -l_i, \; \forall i$, we recover the 1-norm penalty

Implementation details

Upper bound $\mathrm{card}(x^\star)$ ($x^\star$ relaxed) relaxation seeks for sparse solutions

Implementation details

Upper bound $\mathrm{card}(x^\star)$ ($x^\star$ relaxed) relaxation seeks for sparse solutions

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

Implementation details

Upper bound $\mathrm{card}(x^\star)$ ($x^\star$ relaxed) relaxation seeks for sparse solutions

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 $\mathrm{card}(x^\star)$ ($x^\star$ relaxed) relaxation seeks for sparse solutions

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

Best-bound search split node with lowest L

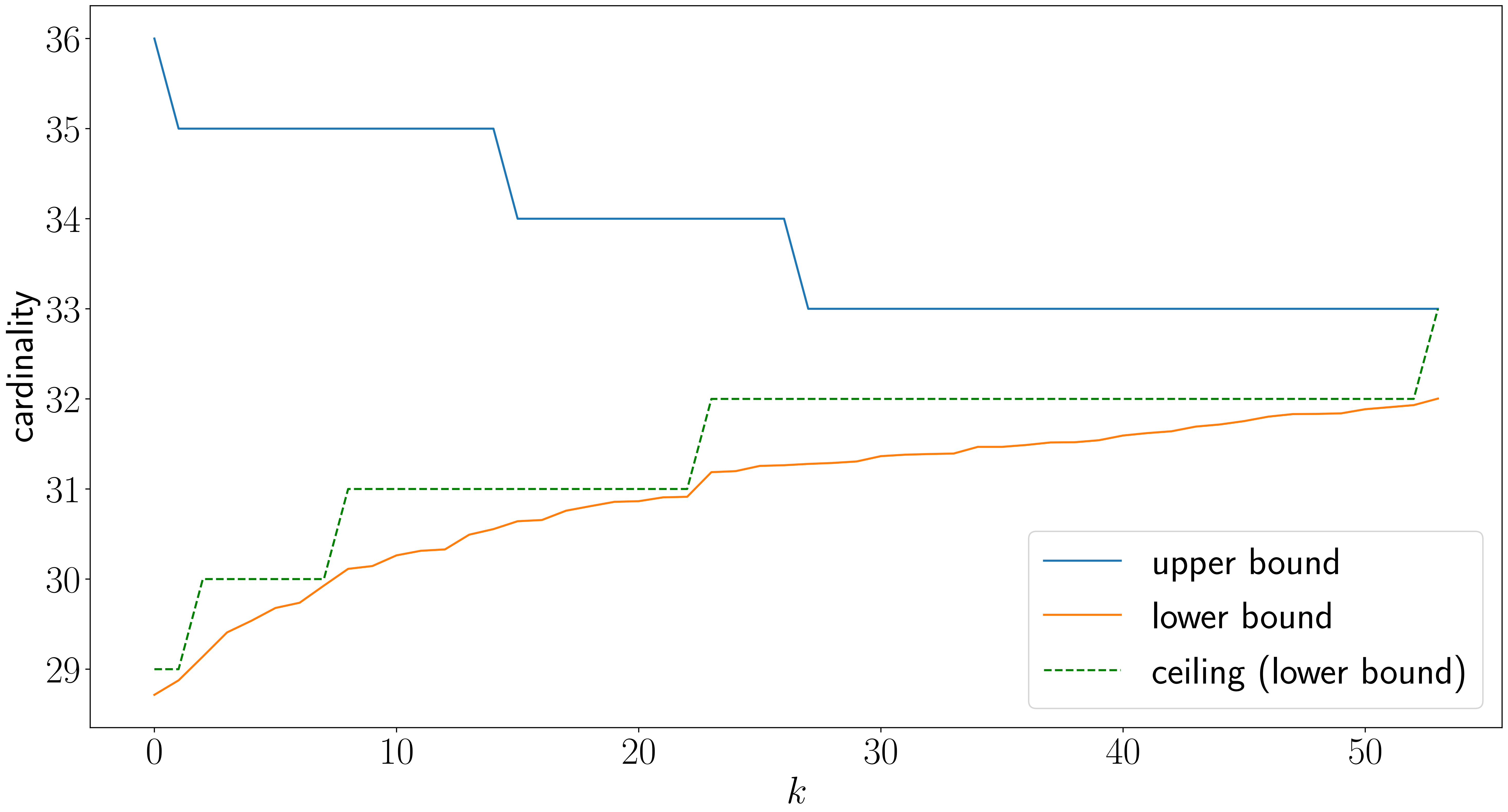
Most ambivalent variable the closest z_k to $1/2$

Small example

Data

40 variables, 200 constraints

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



Results

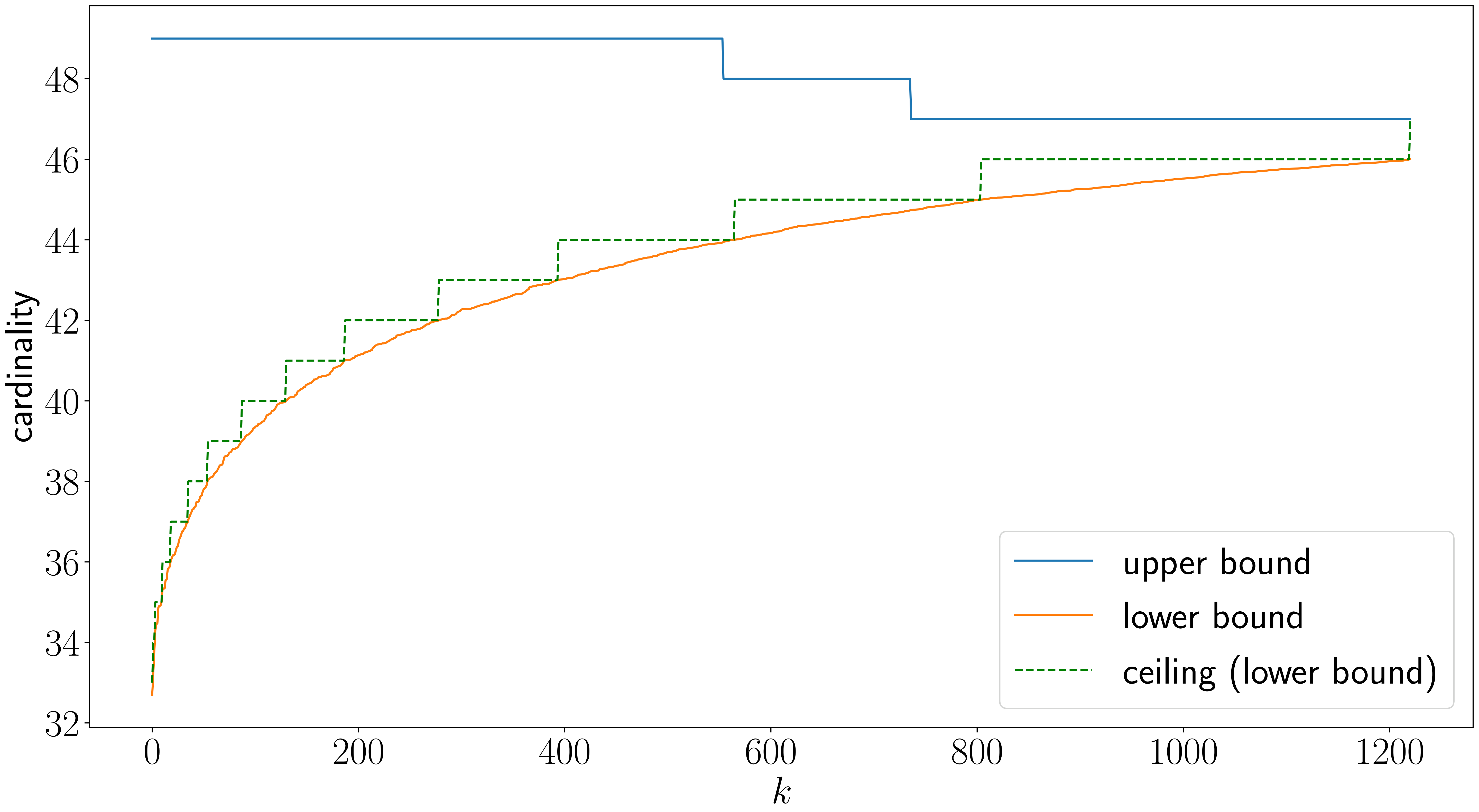
- Finds good solution very quikcly
- Weighted 1-norm heuristic works very well
- Terminates in 54 iterations

Medium example

Data

60 variables, 200 constraints

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



Results

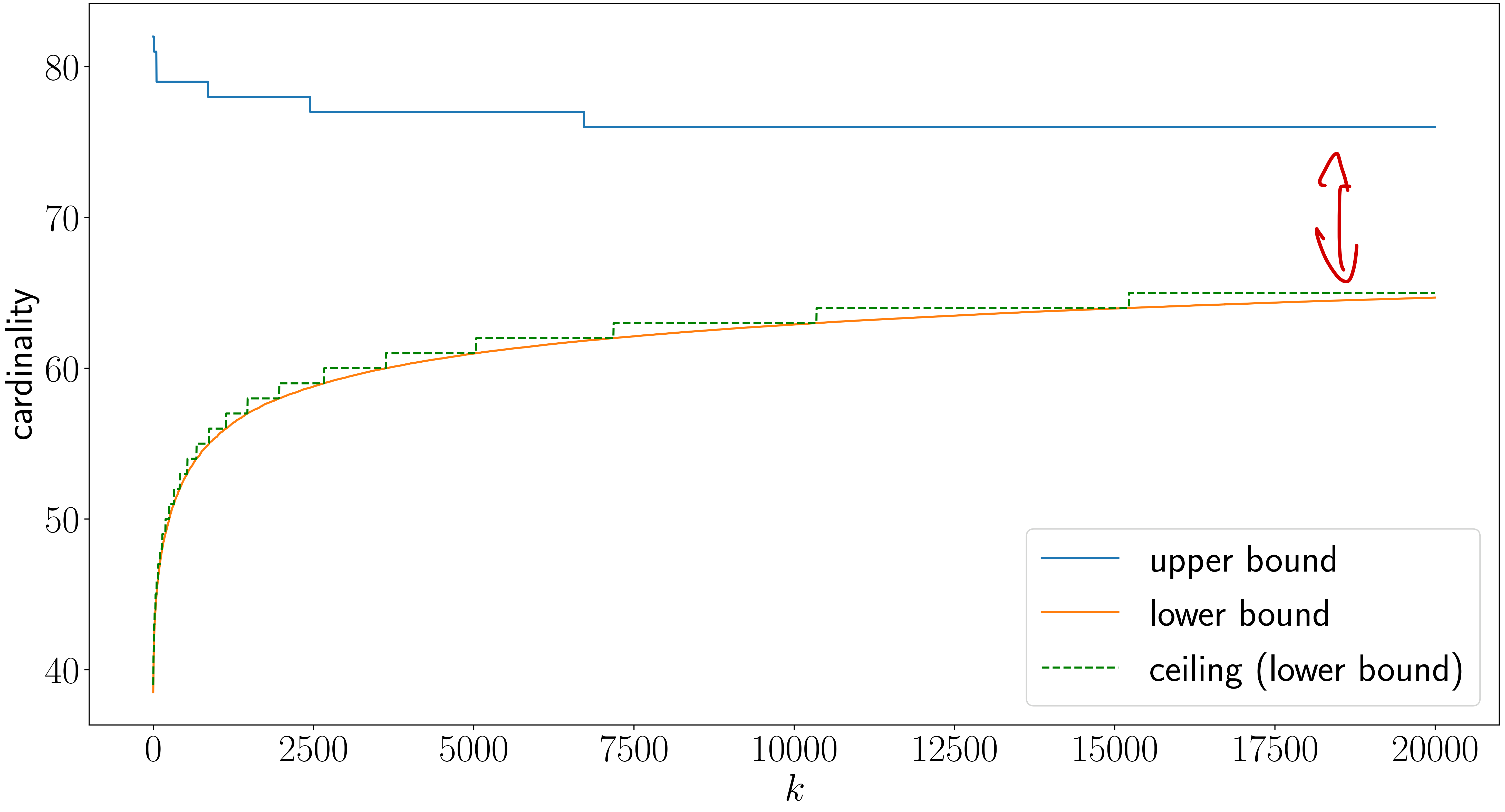
- Finds good solution very quikcly
- Weighted 1-norm heuristic works very well
- Terminates in ≈ 1200 iterations

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 gap certificate in the end

Larger example with commercial solver

Data

100 variables, 300 constraints

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

Results

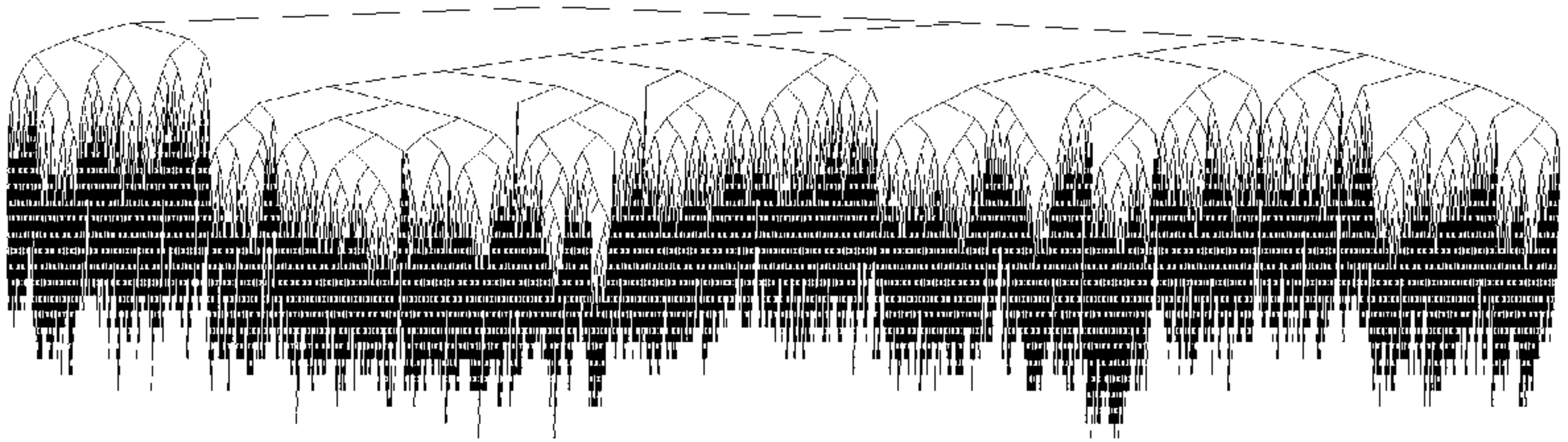
- Optimal cardinality 72
- Much more sophisticated method
- 1888 seconds (31 minutes) run (very slow!)

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 360s on CPU…



10,000 nodes

The cost of building a certificate

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

Branch and bound algorithms

Today, we learned to:

- **Understand and apply** branch and bound ideas for nonconvex optimization
- **Analyze** branch and bound convergence
- **Implement** branch and bound to mixed-integer convex optimization
- **Recognize** the current limitations of branch and bound schemes

Next lecture

- Conclusions