An efficient solver for L0-penalized problems

Theo Guyard with C. Herzet, A. N. Arslan and C. Elvira

INRIA and INSA Rennes

ROADEF | 21 Feb 2023

L0-penalized problems

Objectives

L0-penalized problems

Minimize a loss ⟷ **Objectives**

L0-penalized problems

Minimize a loss $\xleftrightarrow{\text{Objectives}}$ Sparse solution

L0-penalized problems

Minimize a loss $\longleftrightarrow$ **Sparse solution**

Objectives

Machine Learning
Signal processing
Network design
...

L0-penalized problems

Minimize a loss ⟷ **Objectives** ⟷ **Sparse solution**

Machine Learning
Signal processing
Network design
...

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

L0-penalized problems

Minimize a loss ⟷ **Objectives** ⟷ **Sparse solution**

Machine Learning
Signal processing
Network design
...

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

$F(Ax)$: loss

Minimize a loss ⟷ **Objectives** ⟷ **Sparse solution**

Machine Learning
Signal processing
Network design
...

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

$F(Ax)$: loss
$\|x\|_0$: sparsity

L0-penalized problems

Minimize a loss $\longleftrightarrow$ **Objectives** $\longleftrightarrow$ **Sparse solution**

Machine Learning
Signal processing
Network design
...

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda \|x\|_0 + G(x)$$

$F(Ax)$: loss
$\|x\|_0$: sparsity
λ : trade-off

L0-penalized problems

Minimize a loss ⟷ **Objectives** ⟷ **Sparse solution**

Machine Learning
Signal processing
Network design
...

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

$$\begin{aligned} F(Ax) &: \text{loss} \\ \|x\|_0 &: \text{sparsity} \\ \lambda &: \text{trade-off} \\ G(x) &: \text{modelling} \end{aligned}$$

Contributions

ℓ_0-penalized problem

$$\min_{x} \; F(Ax) + \lambda\|x\|_0 + G(x)$$

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

Contributions

ℓ_0-penalized problem

$$\min_{x} \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Contributions

ℓ_0-penalized problem

$$\min_{x} \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda \|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

ℓ_0-penalized problem

$$\min_{x} \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

- This talk (2023)
- Extended paper (202?)

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

- This talk (2023)
- Extended paper (202?)

✓ BnB algorithm

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

- This talk (2023)
- Extended paper (202?)

✓ BnB algorithm
✓ Efficient bounding

Contributions

ℓ_0-penalized problem

$$\min_x \; F(Ax) + \lambda \|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

- This talk (2023)
- Extended paper (202?)

✓ BnB algorithm
✓ Efficient bounding
✓ Generalize w.r.t. F and G

Contributions

ℓ_0-penalized problem

$$\min_{x} \; F(Ax) + \lambda\|x\|_0 + G(x)$$

NP-hard

Generic approach

- D. Bertsimas (2016)
- S. Bourguignon (2017)
- A. Atamtürk (2020)
- D. Bertsimas (2021)
- C. Kanzow (2022)

✓ MIP formulation
✓ Off-the-shelf solvers
✓ User friendly
✗ Slow

Tailored approach

- R. Ben Mhenni (2021)
- H. Hazimeh (2021)
- G. Samain (2022)
- A. Olama (2022)
- A. Atamtürk (2022)

✓ BnB algorithm
✓ Efficient bounding
✗ Restrictions on F and G
✗ Inappropriate exploration

Contributions

- This talk (2023)
- Extended paper (202?)

✓ BnB algorithm
✓ Efficient bounding
✓ Generalize w.r.t. F and G
✓ Tailored exploration

Branch-and-Bound


N^0

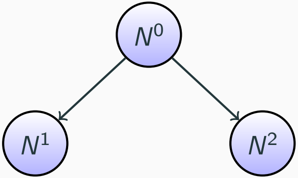
Branch-and-Bound

$\min_{\mathbf{x}} \;\; F(\mathbf{A}\mathbf{x}) + \lambda\|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$

N^0

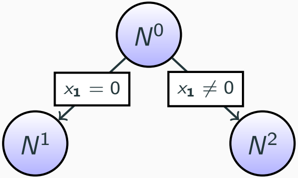
Branch-and-Bound

$$\min_{\mathbf{x}} \; F(\mathbf{A}\mathbf{x}) + \lambda\|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$$



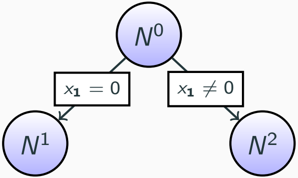
Branch-and-Bound

$\min_{\mathbf{x}} \;\; F(\mathbf{A}\mathbf{x}) + \lambda\|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$



Branch-and-Bound

$$\min_{\mathbf{x}} \; F(\mathbf{A}\mathbf{x}) + \lambda\|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$$

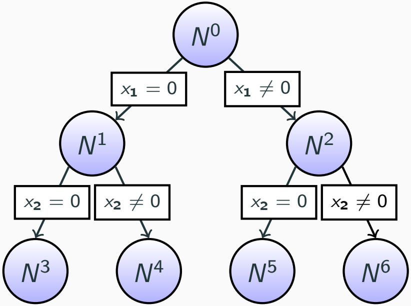


$$\begin{array}{ll} \min_{\mathbf{x}} & \text{Same obj.} \\ \text{s.t.} & x_{\mathbf{1}} = 0 \end{array}$$

$$\begin{array}{ll} \min_{\mathbf{x}} & \text{Same obj.} \\ \text{s.t.} & x_{\mathbf{1}} \neq 0 \end{array}$$

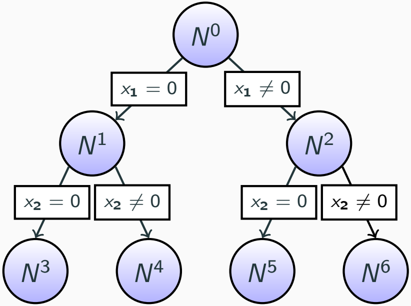
Branch-and-Bound

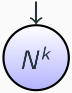
$$\min_{\mathbf{x}} \;\; F(\mathrm{A}\mathrm{x}) + \lambda \|\mathrm{x}\|_{\mathbf{0}} + G(\mathrm{x})$$



Branch-and-Bound

$\min_{\mathbf{x}} \; F(\mathbf{A}\mathbf{x}) + \lambda \|\mathbf{x}\|_0 + G(\mathbf{x})$

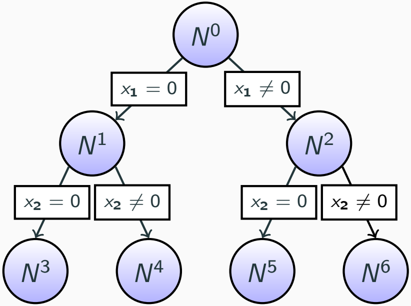


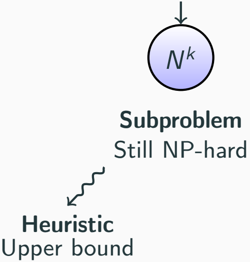


Subproblem
Still NP-hard

Branch-and-Bound

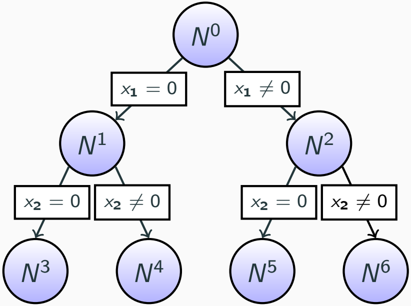
$\min_{\mathbf{x}} \;\; F(\mathbf{A}\mathbf{x}) + \lambda\|\mathbf{x}\|_0 + G(\mathbf{x})$

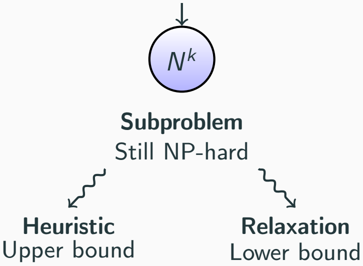




Branch-and-Bound

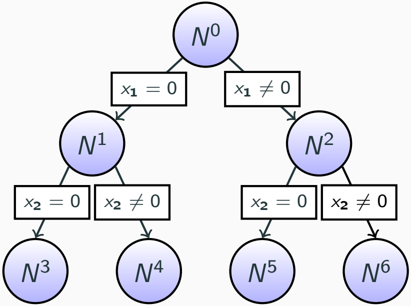
$$\min_{\mathbf{x}} \; F(\mathbf{A}\mathbf{x}) + \lambda \|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$$

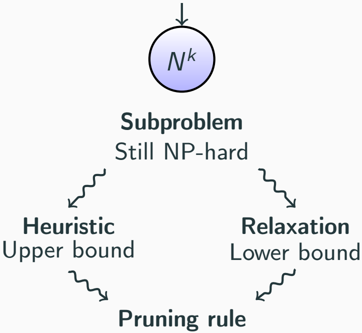




Branch-and-Bound

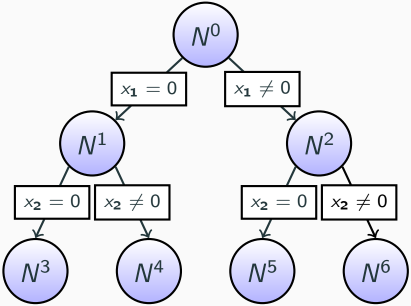
$$\min_{\mathbf{x}} \; F(\mathrm{A}\mathrm{x}) + \lambda \|\mathrm{x}\|_{\mathbf{0}} + G(\mathrm{x})$$

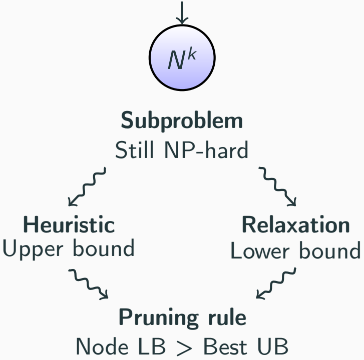




Branch-and-Bound

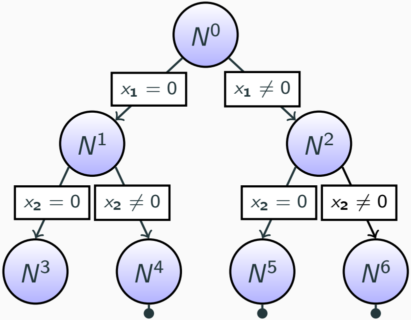
$\min_{\mathbf{x}} \;\; F(\mathbf{A}\mathbf{x}) + \lambda \|\mathbf{x}\|_0 + G(\mathbf{x})$





Branch-and-Bound

$\min_{\mathbf{x}} \; F(\mathbf{A}\mathbf{x}) + \lambda \|\mathbf{x}\|_{\mathbf{0}} + G(\mathbf{x})$



N^k

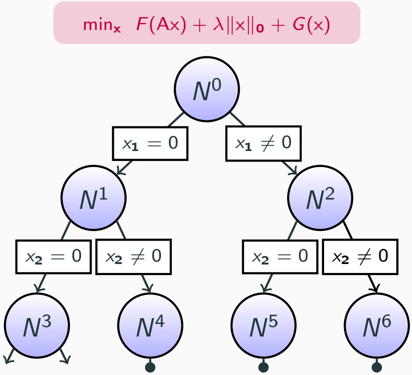
Subproblem
Still NP-hard

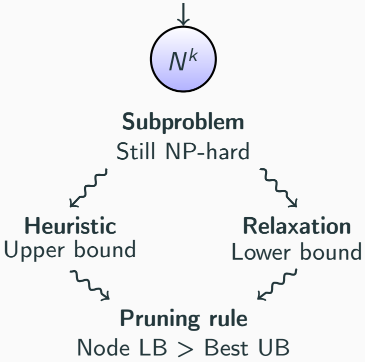
Heuristic
Upper bound

Relaxation
Lower bound

Pruning rule
Node LB > Best UB

Branch-and-Bound

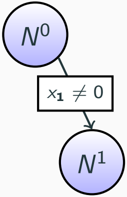




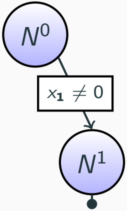
Structural bottleneck

Structural bottleneck

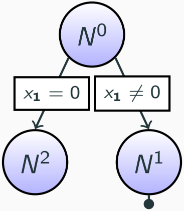
Structural bottleneck



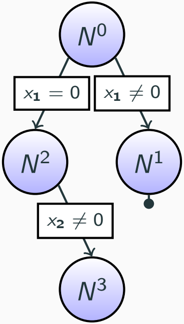
Structural bottleneck



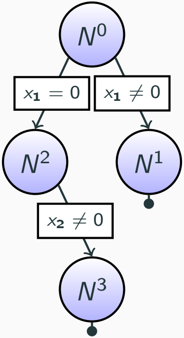
Structural bottleneck



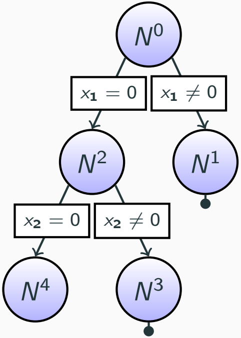
Structural bottleneck



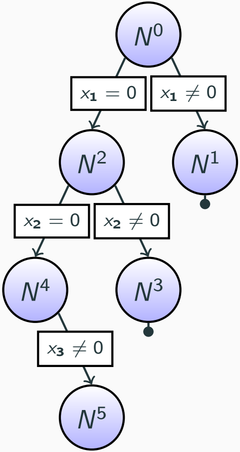
Structural bottleneck



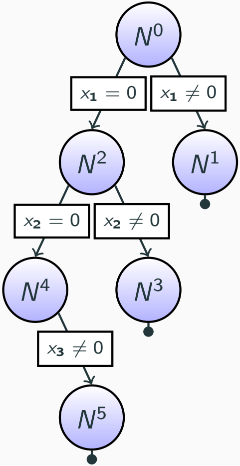
Structural bottleneck



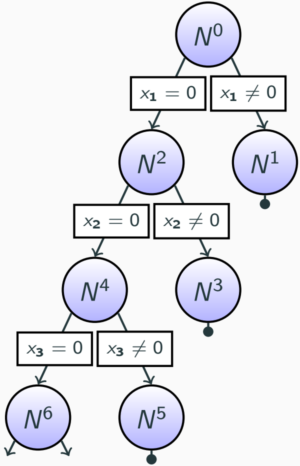
Structural bottleneck



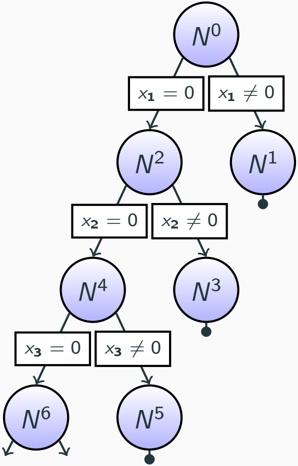
Structural bottleneck



Structural bottleneck

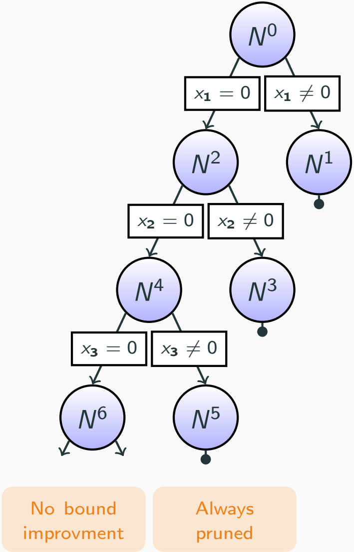


Structural bottleneck

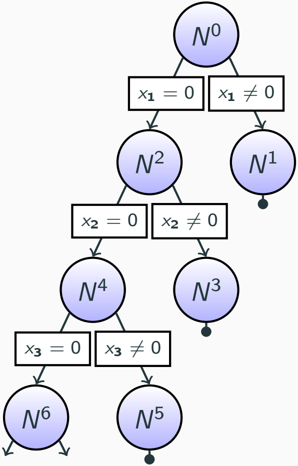


Always pruned

Structural bottleneck



Structural bottleneck

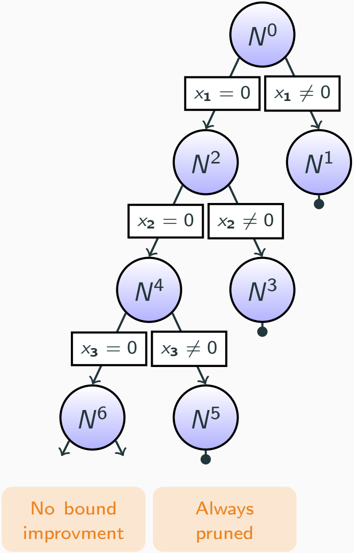


No bound improvment

Always pruned

Waste of computational power !

Structural bottleneck

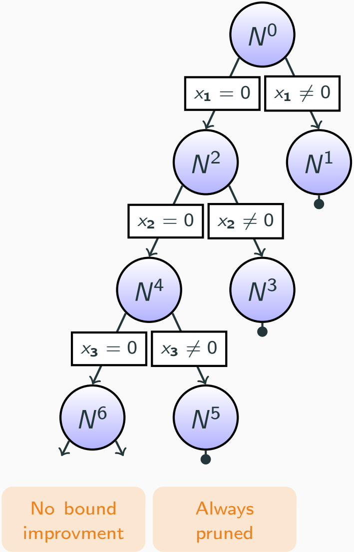


Waste of computational power !

$\downarrow$

How to avoid such situations ?

Structural bottleneck



Waste of computational power !

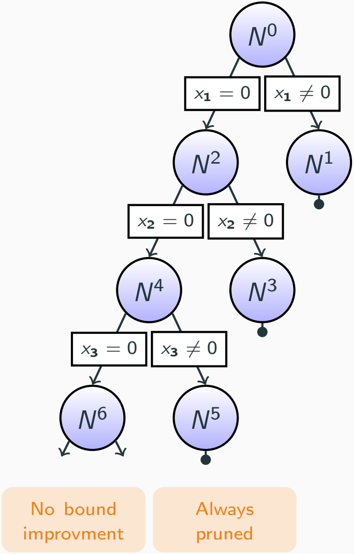
↓

How to avoid such situations ?

↓

Leverage duality link between nodes

Structural bottleneck



Waste of computational power !

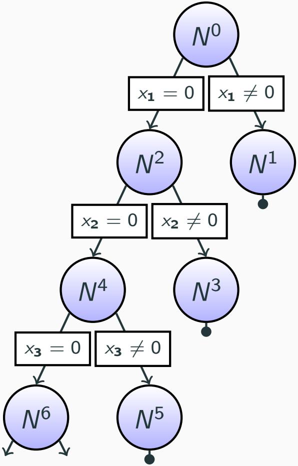
↓

How to avoid such situations ?

↓

Leverage duality link between nodes

Structural bottleneck



No bound improvment

Always pruned

Waste of computational power !

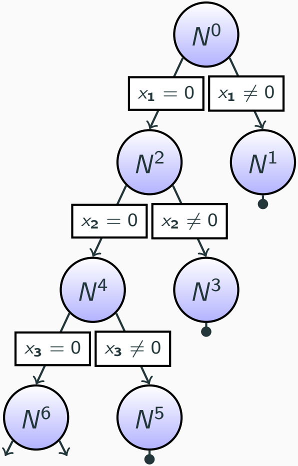
↓

How to avoid such situations ?

↓

Leverage duality link between nodes

Structural bottleneck



No bound improvment

Always pruned

Waste of computational power !

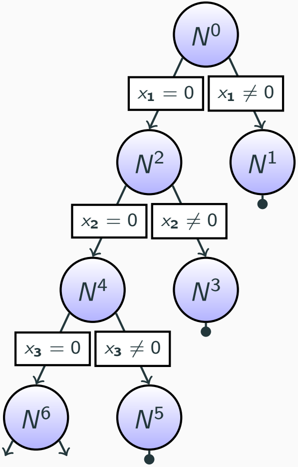
↓

How to avoid such situations ?

↓

Leverage duality link between nodes

Structural bottleneck



No bound improvment

Always pruned

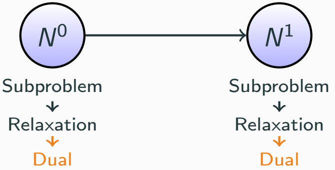
Waste of computational power !

↓

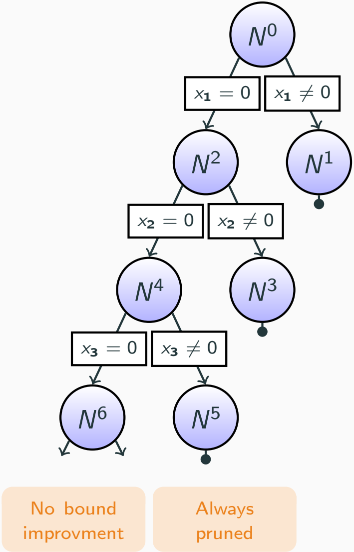
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



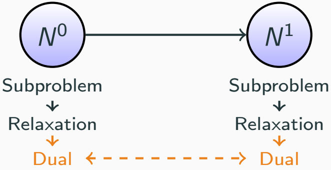
Waste of computational power !

↓

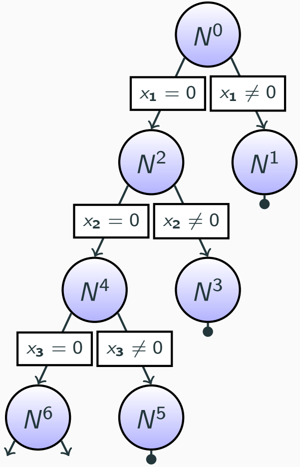
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



No bound improvment

Always pruned

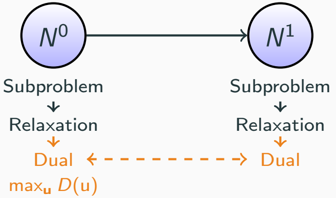
Waste of computational power !

↓

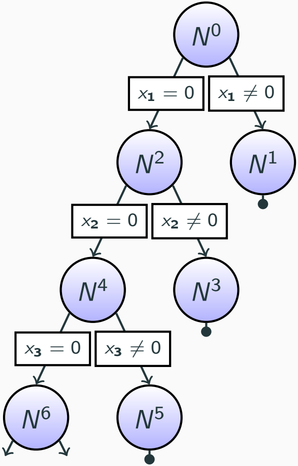
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



No bound improvment

Always pruned

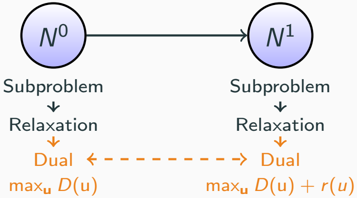
Waste of computational power !

↓

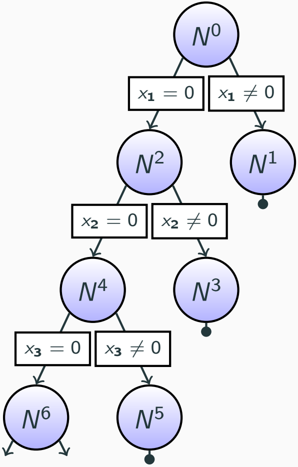
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



No bound improvment

Always pruned

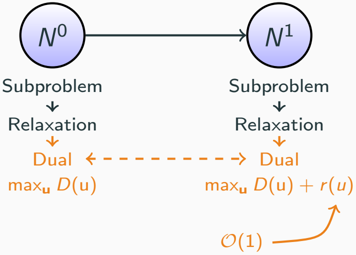
Waste of computational power !

↓

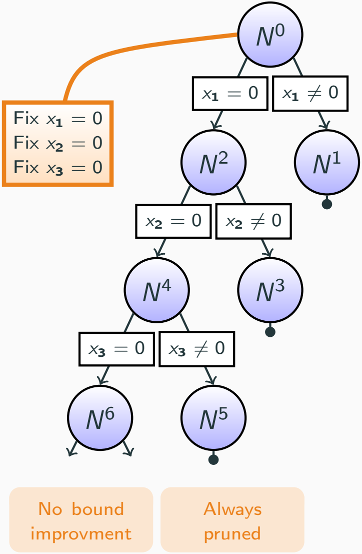
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



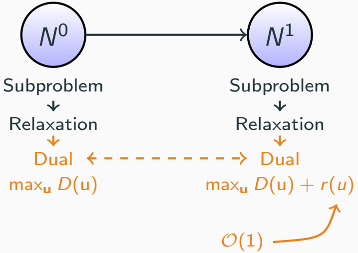
Waste of computational power !

↓

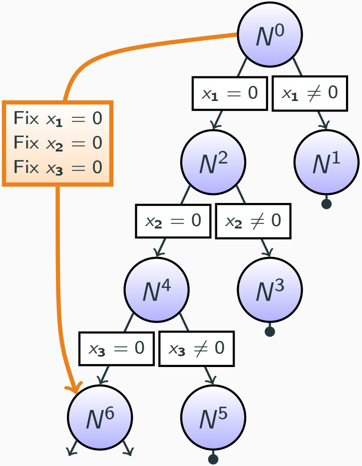
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



No bound improvment

Always pruned

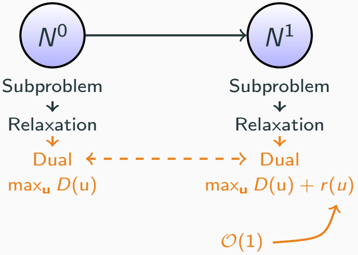
Waste of computational power !

↓

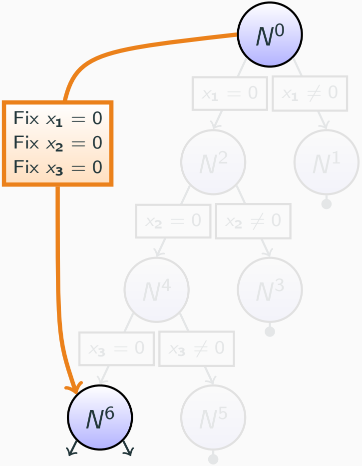
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



No bound improvment

Always pruned

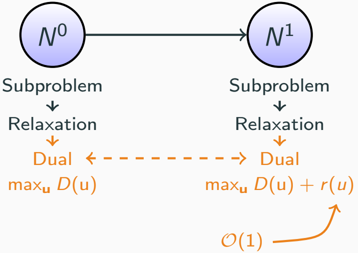
Waste of computational power !

↓

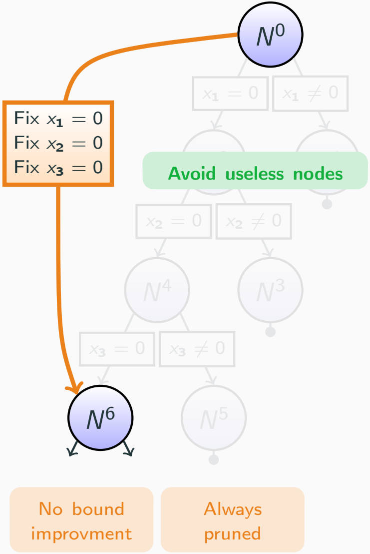
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



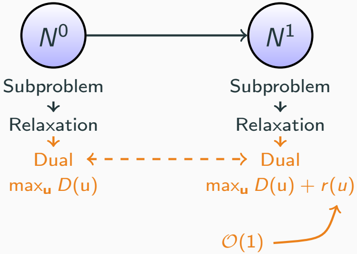
Waste of computational power !

↓

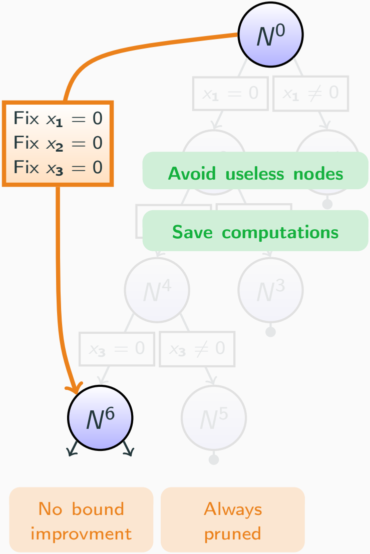
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



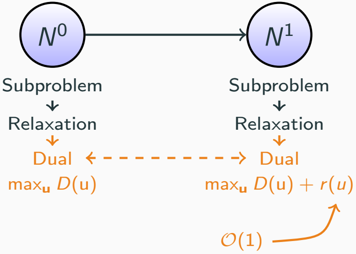
Waste of computational power !

↓

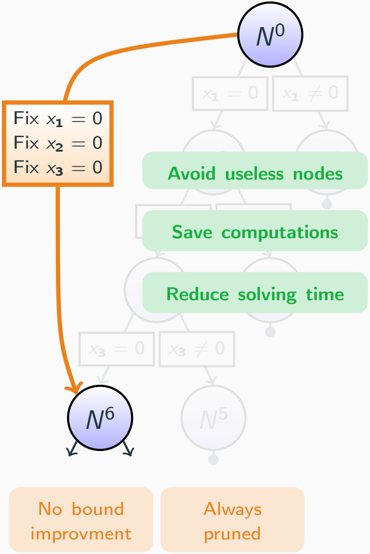
How to avoid such situations ?

↓

Leverage duality link between nodes



Structural bottleneck



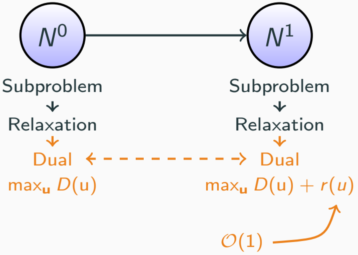
Waste of computational power !

↓

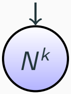
How to avoid such situations ?

↓

Leverage duality link between nodes



Variable fixing

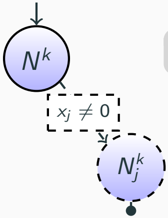


Variable fixing

N^k

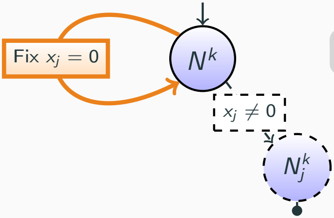
Identify an index j

Variable fixing



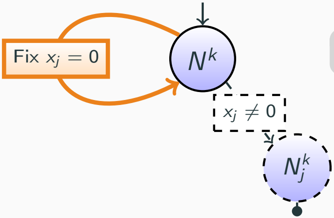
Identify an index j

Variable fixing

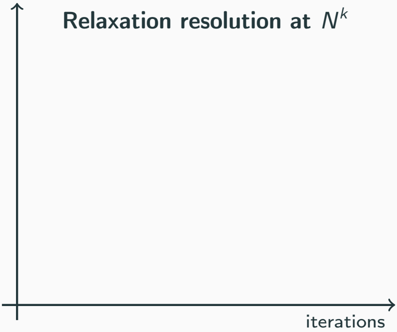


Identify an index j

Variable fixing

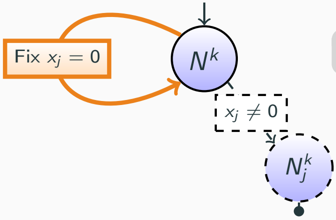


Identify an index j



Ingredients

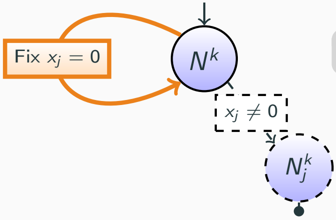
Variable fixing



Identify an index j

Ingredients

Variable fixing

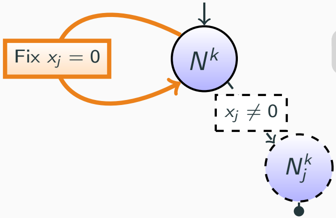


Identify an index j

Ingredients

- Already-computed quantities

Variable fixing

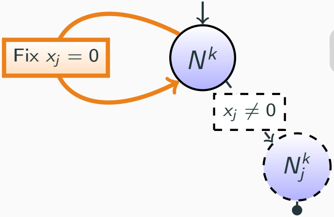


Identify an index j

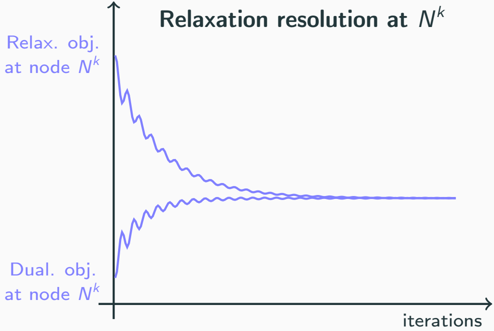
Ingredients

- Already-computed quantities
- Cost-free dual evaluation

Variable fixing



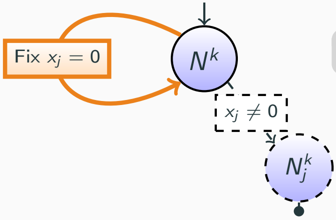
Identify an index j



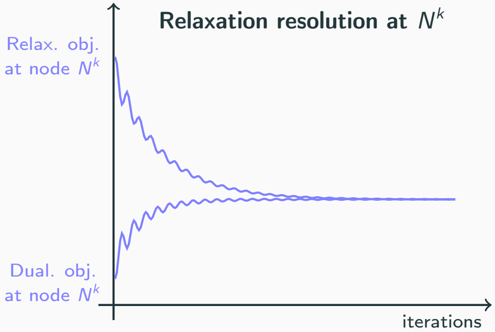
Ingredients

- Already-computed quantities
- Cost-free dual evaluation

Variable fixing



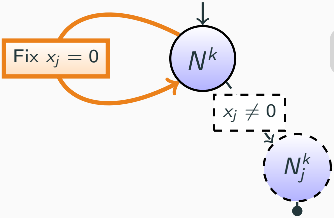
Identify an index j



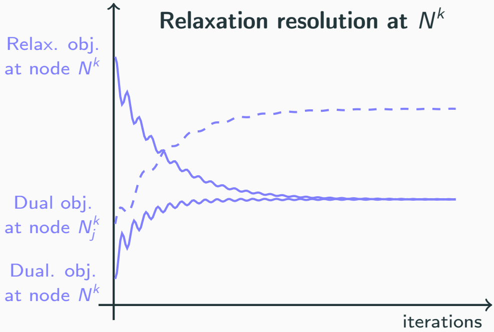
Ingredients

- Already-computed quantities
- Cost-free dual evaluation
- Dual link between nodes

Variable fixing



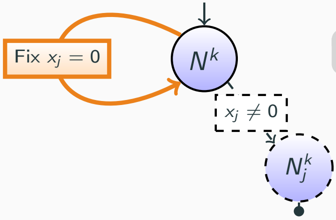
Identify an index j



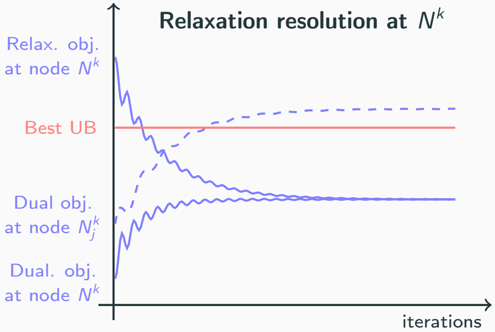
Ingredients

- Already-computed quantities
- Cost-free dual evaluation
- Dual link between nodes

Variable fixing



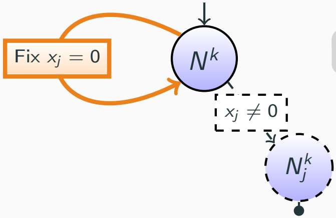
Identify an index j



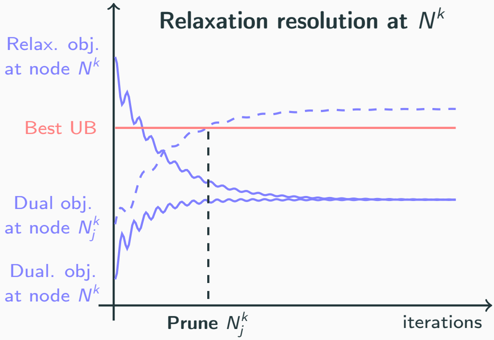
Ingredients

- Already-computed quantities
- Cost-free dual evaluation
- Dual link between nodes

Variable fixing



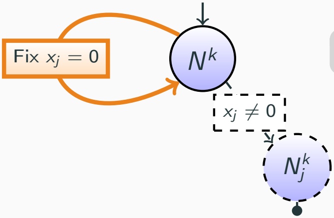
Identify an index j



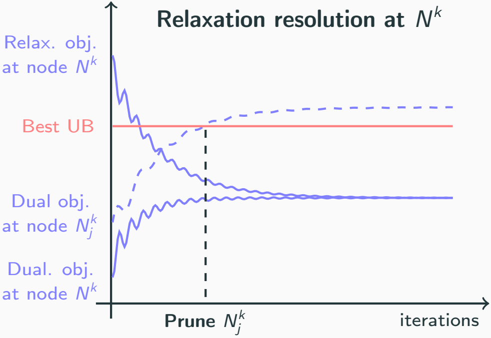
Ingredients

- Already-computed quantities
- Cost-free dual evaluation
- Dual link between nodes

Variable fixing



Identify an index j



Ingredients

- Already-computed quantities
- Cost-free dual evaluation
- Dual link between nodes
- Can fix **multiple** variables

Numerical illustrations

$$\min_{x} \; F(Ax) + \lambda \|x\|_0 + G(x)$$

Numerical illustrations

$$\min_{x}\ F(Ax) + \lambda\|x\|_0 + G(x)$$

Dataset : Sparse linear regression

Numerical illustrations

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

Dataset : Sparse linear regression

F(Ax) : Mean squared error

Numerical illustrations

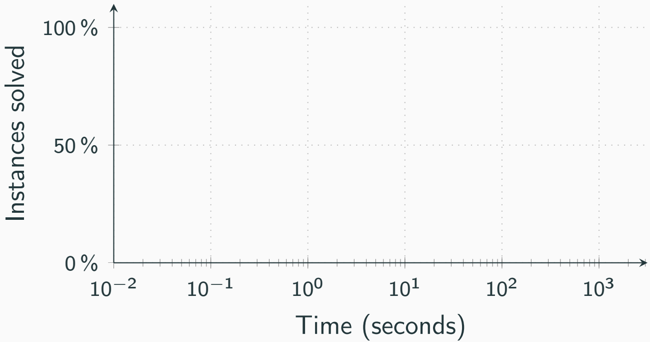
$$\min_x \; F(Ax) + \lambda \|x\|_0 + G(x)$$

Dataset : Sparse linear regression
F(Ax) : Mean squared error
G(x) : Constraint $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$

Numerical illustrations

$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$

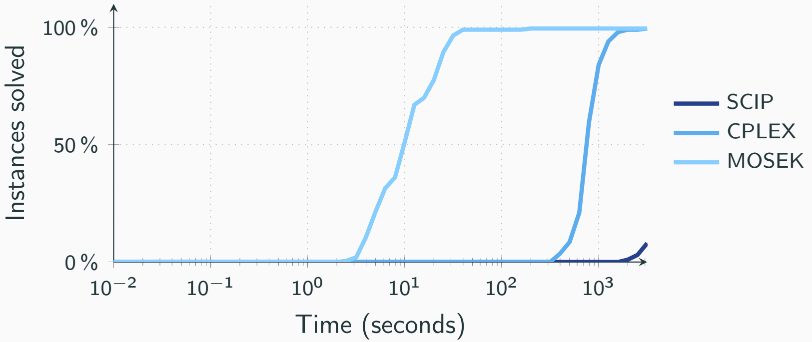
Dataset : Sparse linear regression
F(Ax) : Mean squared error
G(x) : Constraint $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$



Numerical illustrations

$$\min_{\mathrm{x}} \; F(\mathrm{Ax}) + \lambda\|\mathrm{x}\|_0 + G(\mathrm{x})$$

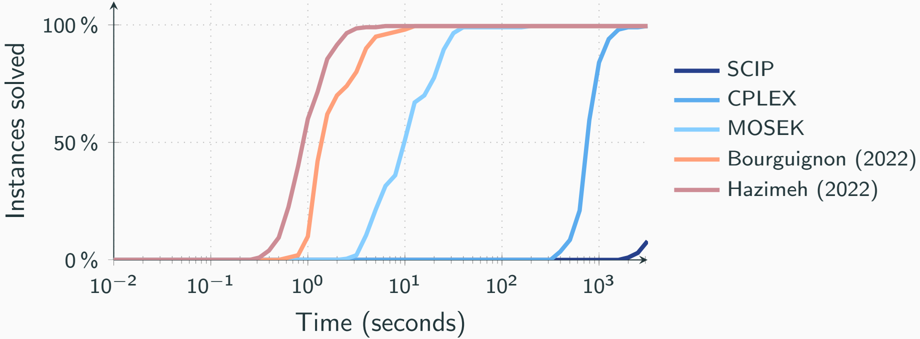
Dataset : Sparse linear regression
F(Ax) : Mean squared error
G(x) : Constraint $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$



Numerical illustrations

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

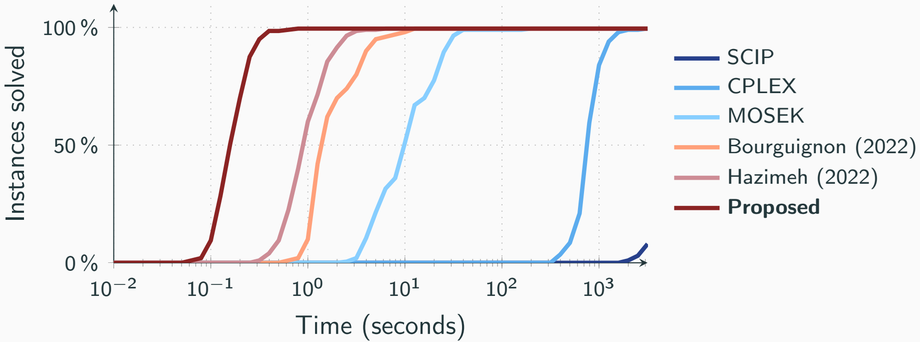
Dataset : Sparse linear regression
F(Ax) : Mean squared error
G(x) : Constraint $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$



Numerical illustrations

$$\min_x \; F(Ax) + \lambda\|x\|_0 + G(x)$$

Dataset : Sparse linear regression
F(Ax) : Mean squared error
G(x) : Constraint $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$



Numerical illustrations

Highlights

Highlights

- Gains up to $\times 10^5$ against MIP solvers

Numerical illustrations

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB

Numerical illustrations

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB
- Scales up to dimension $\approx 10^5$ in minutes

Numerical illustrations

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB
- Scales up to dimension $\approx 10^5$ in minutes

TheoGuyard/**El0ps.jl**

An Exact L0-penalized Problem Solver.

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB
- Scales up to dimension $\approx 10^5$ in minutes

Efficient BnB

💪

TheoGuyard/**El0ps.jl**

An Exact L0-penalized Problem Solver.

Numerical illustrations

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB
- Scales up to dimension $\approx 10^5$ in minutes

Efficient BnB 💪

Generic framework 💪

TheoGuyard/El0ps.jl

An Exact L0-penalized Problem Solver.

Numerical illustrations

Highlights

- Gains up to $\times 10^5$ against MIP solvers
- Gains up to $\times 10^2$ against tailored BnB
- Scales up to dimension $\approx 10^5$ in minutes

Efficient BnB 💪 → **TheoGuyard/El0ps.jl** ← Generic framework 💪

An Exact L0-penalized Problem Solver.

↑ 🥳

Works with any F and G
under mild assumptions !

Question time !

Working hypotheses

Loss function

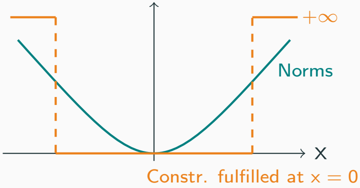
Assumptions on F :

- F is proper, convex, lower-semicontinuous
- F is differentiable with ∇F being L-Lipschitz

Modelling term

Assumptions on G :

- G is proper, convex, lower-semicontinuous
- $G = \sum_i G_i$ with $G_i \geq G_i(0) = 0$



Node problems

N^k

$\mathcal{S}_0 = \{i \mid x_i = 0\}$

$\mathcal{S}_1 = \{i \mid x_i \neq 0\}$

$\mathcal{S}_\bullet = \{i \mid x_i \text{ free}\}$

Subproblem

$$\begin{cases} \min_{\mathrm{x}} & F(\mathrm{Ax}) + \lambda\|\mathrm{x}\|_0 + G(\mathrm{x}) \\ \text{s.t.} & \mathrm{x}_{\mathcal{S}_0} = 0,\ \mathrm{x}_{\mathcal{S}_1} \neq 0 \end{cases}$$

NP-hard unless $\mathcal{S}_\bullet = \emptyset$.

Relaxation

$$\min_{\mathrm{x}} \left\{ F(\mathrm{Ax}) + H(\mathrm{x}) \right\}$$

with $H(\mathrm{x})$ being the convex envelope of $\lambda\|\mathrm{x}\|_0 + G(\mathrm{x}) + \mathbb{I}(\mathrm{x}_{\mathcal{S}_0} = 0, \mathrm{x}_{\mathcal{S}_1} \neq 0)$.

Dual

$$\max_{\mathrm{u}} \left\{ -F^\star(-\mathrm{u}) - H^\star(\mathrm{A}^{\mathrm{T}}\mathrm{u}) \right\}$$

with $H^\star(\mathrm{A}^{\mathrm{T}}\mathrm{u}) = \sum_{i \in \mathcal{S}_\bullet} [G^\star(\mathrm{a}_i^{\mathrm{T}}\mathrm{u}) - 1]_+ + \sum_{i \in \mathcal{S}_1} (G^\star(\mathrm{a}_i^{\mathrm{T}}\mathrm{u}) - 1)$