Modeling Disjunctive Constraints with a Logarithmic Number of Binary Variables and Constraints

Juan Pablo Vielma

IBM Watson Research Center

Shabbir Ahmed and George L. Nemhauser

H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

International Symposium for Mathematical Programming
August, 2009 – Chicago, IL

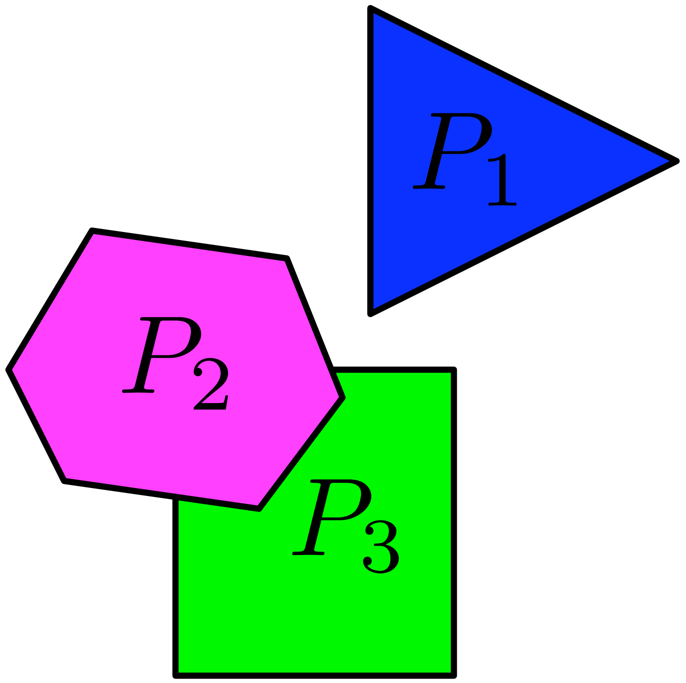
Outline

- Introduction: MIP models for disjunctive constraints.
- Smaller MIPs for SOS1, SOS2, piecewise linear functions
- Computational Results.

MIPs for Disjunctive Constraints/Set

$$x \in \bigcup_{i=1}^{m} P_i \subset \mathbb{R}^n$$

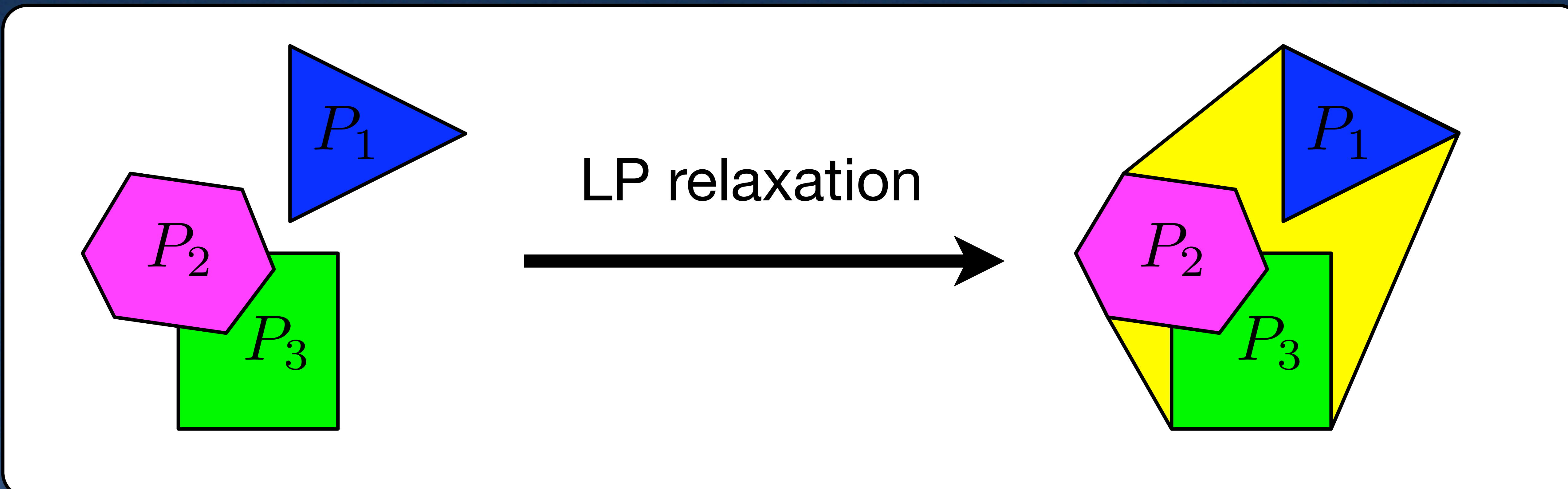
$$P_i := \text{polytope}$$



- 0-1 Mixed Integer Programming (MIP) formulation.
- Can use auxiliary variables besides 0-1 variables.
- Want strong but small formulations.

Strong MIPs : Two Levels

- ***Sharpness***: Projection of LP relaxation onto original variables equals $\operatorname{conv}\left(\bigcup_{i=1}^{m} P_i\right)$.



- ***Locally Ideal*** (stronger): LP has integral extreme points.

A Standard MIP Formulation

$$x \in \bigcup_{i=1}^{m} P_i \subset \mathbb{R}^n$$

$$P_i := \{x \in \mathbb{R}^n \ : \ A^i x \leq b^i\}$$

$$A^i \in \mathbb{R}^{r \times n},\ b^i \in \mathbb{R}^{r_i}$$

$$\longrightarrow$$

$$A^i x^i \leq b^i y_i \quad \forall i$$

$$\sum_{i=1}^{m} x^i = x$$

$$\sum_{i=1}^{m} y_i = 1$$

$$y \in \{0,1\}^M$$

- Sharp and Locally Ideal.
- $\Theta(nm)$ extra vars and $\Theta\left(n + rm\right)$ constraints.

Special Disjunctive Constraint

$$x \in \bigcup_{i=1}^{m} P(F_i) \qquad P(F_i) := \{x \in \Delta^n \,:\, x_j \leq 0 \quad \forall j \in F_i\}$$

$$\Delta^n := \left\{x \in [0,1]^n \,:\, \sum\nolimits_{i=1}^{n} x_i = 1\right\}$$

- SOS1: $m = n, \quad F_i = \{1, \ldots, n\} \setminus \{i\}$.
- SOS2: $m = n - 1, \quad F_i = \{1, \ldots, n\} \setminus \{i, i+1\}$.
- Continuous Piecewise Linear Functions.
- Standard formulation has $\Theta(nm)$ extra vars and constraints $\Theta\left(n + m\right)$.

Eliminate copies of x and stay Sharp?

$$\sum_{i=1}^{m} x^i = x, \quad Ax^i \le b^i y_i \quad \forall i \qquad \longrightarrow \qquad Ax \le \sum_{i=1}^{m} b^i y_i$$

$$\sum_{i=1}^{m} y_i = 1, \quad y \in \{0,1\}^m \qquad\qquad \sum_{i=1}^{m} y_i = 1, \quad y \in \{0,1\}^m$$

- Works for special case (Balas, Blair and Jeroslow).
- $\Theta(m)$ extra vars and $\Theta(n)$ constraints.

$$\sum_{j=1}^{n} x_j = 1, \; x \ge 0, \; x_j \le \sum_{i : j \notin F_i} y_i, \; \sum_{i=1}^{m} y_j = 1, \quad y \in \{0,1\}^m$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k) \right) \qquad F_1^k \cap F_2^k = \emptyset$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k) \right) \qquad F_1^k \cap F_2^k = \emptyset$$

$$\begin{array}{llll} \lambda \in \Delta^n & \lambda_j \le y_1^k & \forall j \in F_2^k & \\ & \lambda_j \le y_2^k & \forall j \in F_1^k & y_1^k + y_2^k = 1 \\ & \lambda_j \le y_1^k + y_2^k & \forall j \notin F_1^k \cup F_2^k & y^k \in \{0,1\}^2 \end{array}$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k) \right) \qquad F_1^k \cap F_2^k = \emptyset$$

$$\lambda \in \Delta^n$$

$$\begin{aligned} &\lambda_j \le y_1^k && \forall j \in F_2^k && \\ &\lambda_j \le y_2^k && \forall j \in F_1^k && y_1^k + y_2^k = 1 \\ &\cancel{\lambda_j \le y_1^k + y_2^k} && \cancel{\forall j \notin F_1^k \cup F_2^k} && y^k \in \{0,1\}^2 \end{aligned}$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k) \right) \qquad F_1^k \cap F_2^k = \emptyset$$

$\lambda \in \Delta^n$

$$\sum_{j \in F_2^k} \lambda_j \leq y_1^k$$

$$\sum_{j \in F_1^k} \lambda_j \leq y_2^k$$

$$y_1^k + y_2^k = 1$$

$$y^k \in \{0,1\}^2$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k) \right) \qquad F_1^k \cap F_2^k = \emptyset$$

$$\lambda \in \Delta^n$$

$$\sum\nolimits_{j \in F_2^k} \lambda_j \leq y_1^k \qquad \forall k \in \{1, \ldots, \lceil \log_2 m \rceil\}$$

$$\sum\nolimits_{j \in F_1^k} \lambda_j \leq y_2^k \qquad y_1^k + y_2^k = 1$$

$$y^k \in \{0,1\}^2$$

Rewrite disjunction = reduce binaries.

$$\lambda \in \bigcup_{i=1}^{m} P(F_i) = \bigcap_{k=1}^{\lceil \log_2 m \rceil} \left(P(F_1^k) \cup P(F_2^k)\right) \qquad F_1^k \cap F_2^k = \emptyset$$

$$\lambda \in \Delta^n \qquad \sum_{j \in F_2^k} \lambda_j \le y_1^k \qquad \forall k \in \{1, \ldots, \lceil \log_2 m \rceil\}$$

$$y_1^k + y_2^k = 1$$

$$\sum_{j \in F_1^k} \lambda_j \le y_2^k \qquad y^k \in \{0,1\}^2$$

- $O(\log_2 m)$ extra vars/constraints and locally ideal!
- Vielma and Nemhauser 08/09, Vielma et al. 09.

Rewrite = Independent Branching

- Special Branching Scheme (e.g. SOS2 branch):
  - Both sides implemented by fixing vars to zero.
  - Levels are independent.
- Formulation: 1 binary for each dichotomy.
- For SOS1/SOS2 and Univariate/Multivariate Continuous/Discontinuous Piecewise Linear Functions.

SOS2: Non-zero = two adjacent vars.

- Standard Branching: $\bigvee$ $x_j = 0 \; \forall j < k$ / $x_j = 0 \; \forall j > k$

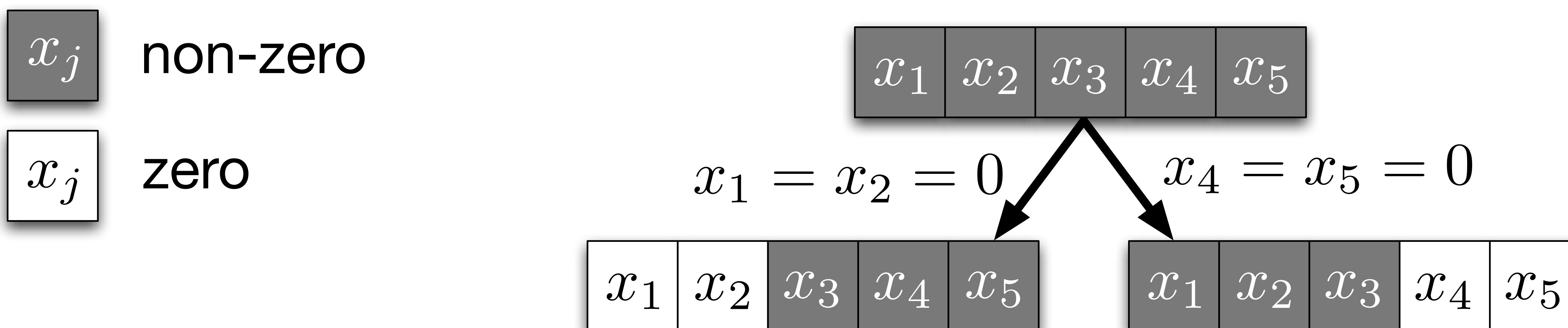
$\boxed{x_j}$ non-zero

$\boxed{x_j}$ zero

x_1	x_2	x_3	x_4	x_5

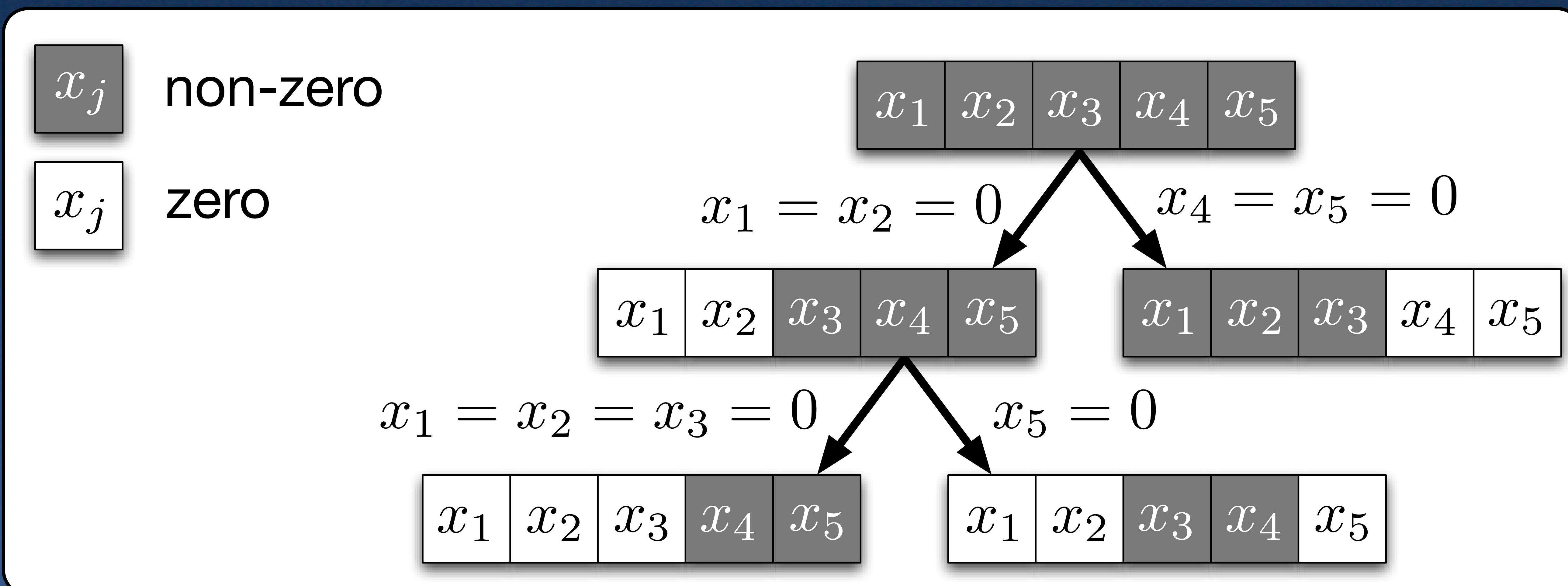
SOS2: Non-zero = two adjacent vars.

- Standard Branching: $\bigvee \begin{cases} x_j = 0 \ \forall j < k \\ x_j = 0 \ \forall j > k \end{cases}$



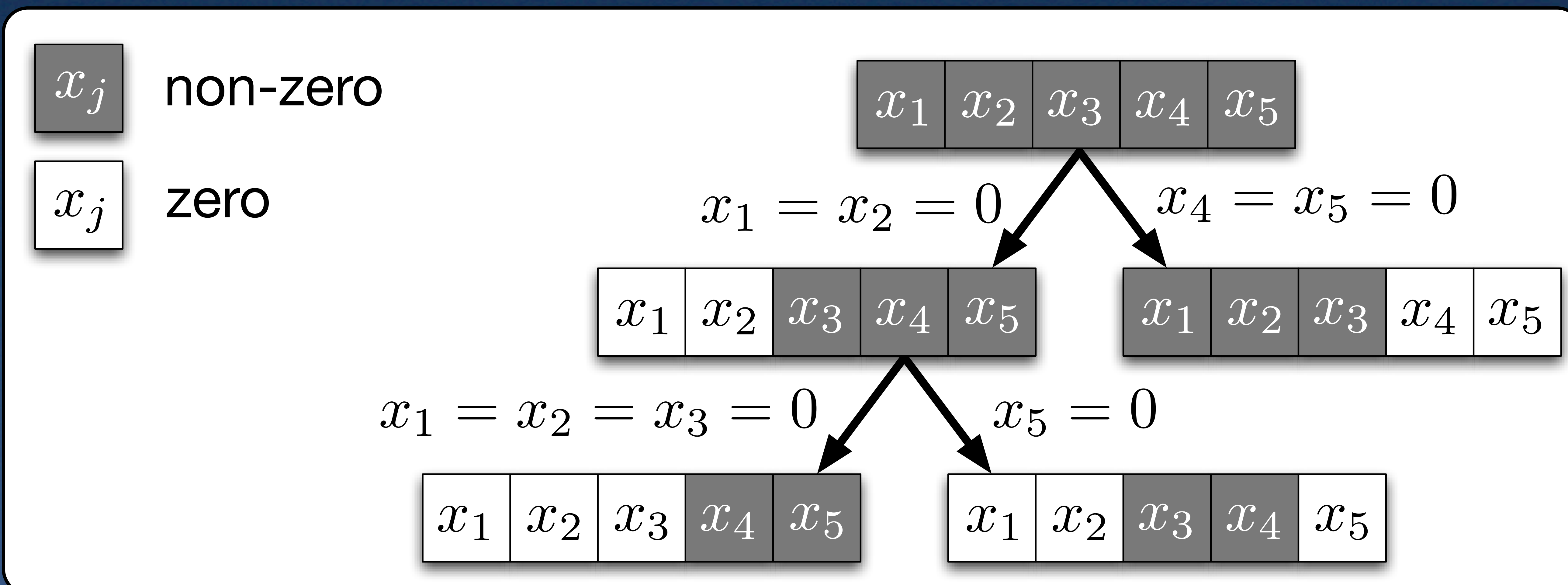
SOS2: Non-zero = two adjacent vars.

- Standard Branching: $\bigvee \begin{cases} x_j = 0 \ \forall j < k \\ x_j = 0 \ \forall j > k \end{cases}$



SOS2: Non-zero = two adjacent vars.

- Standard Branching: $\bigvee$
  - $x_j = 0 \; \forall j < k$
  - $x_j = 0 \; \forall j > k$

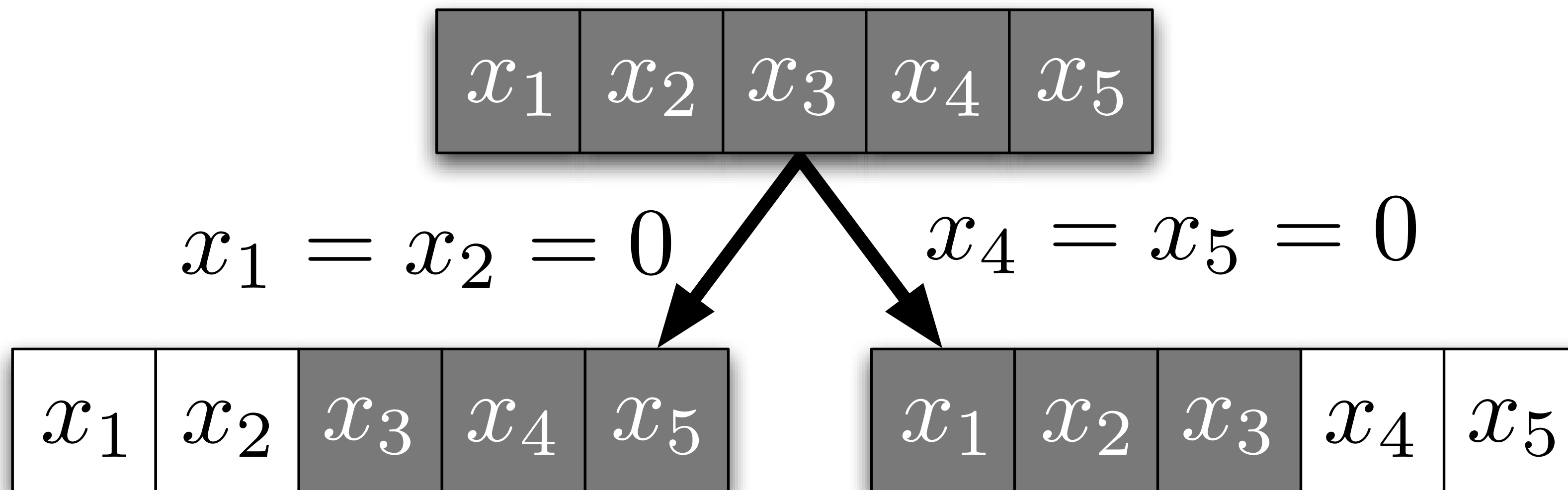


- Total independent dichotomies = m := # vars.

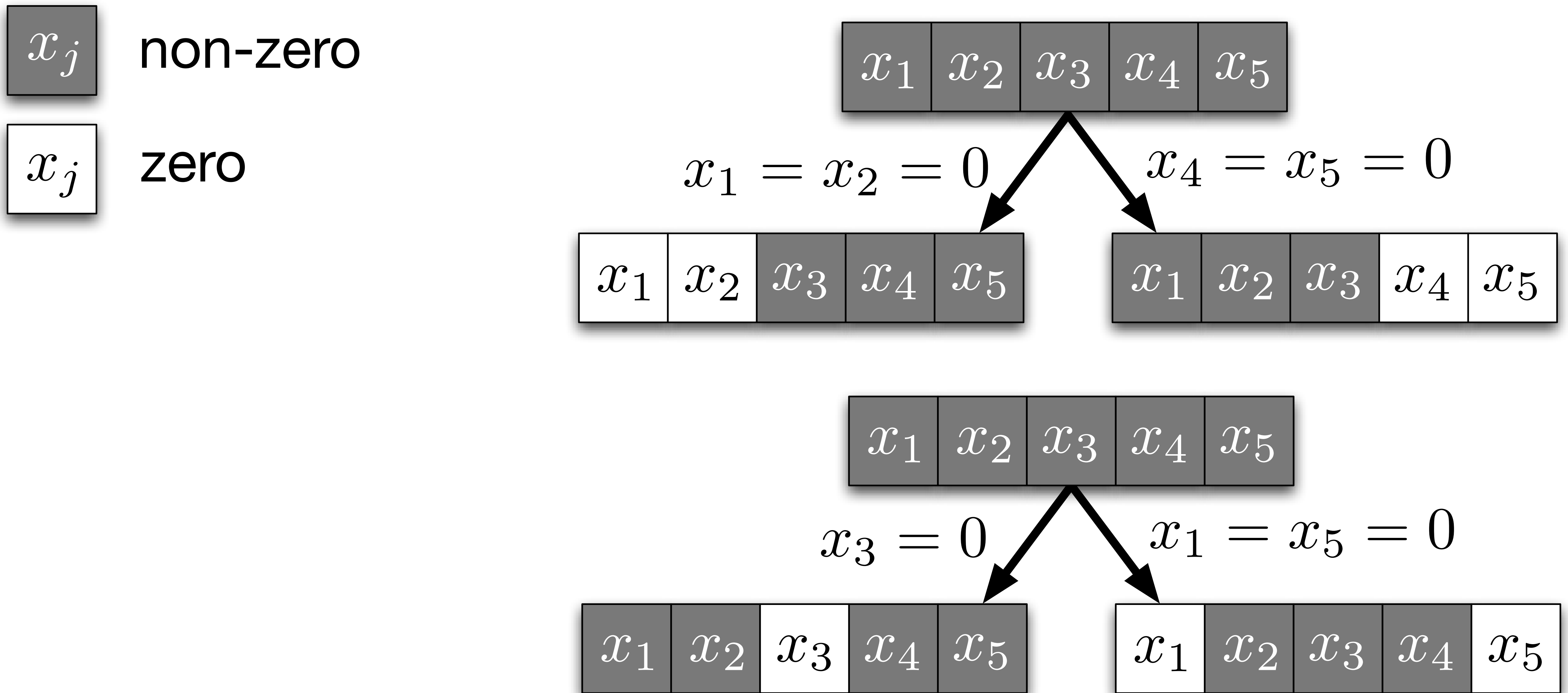
Independent Branching for SOS2

x_j non-zero

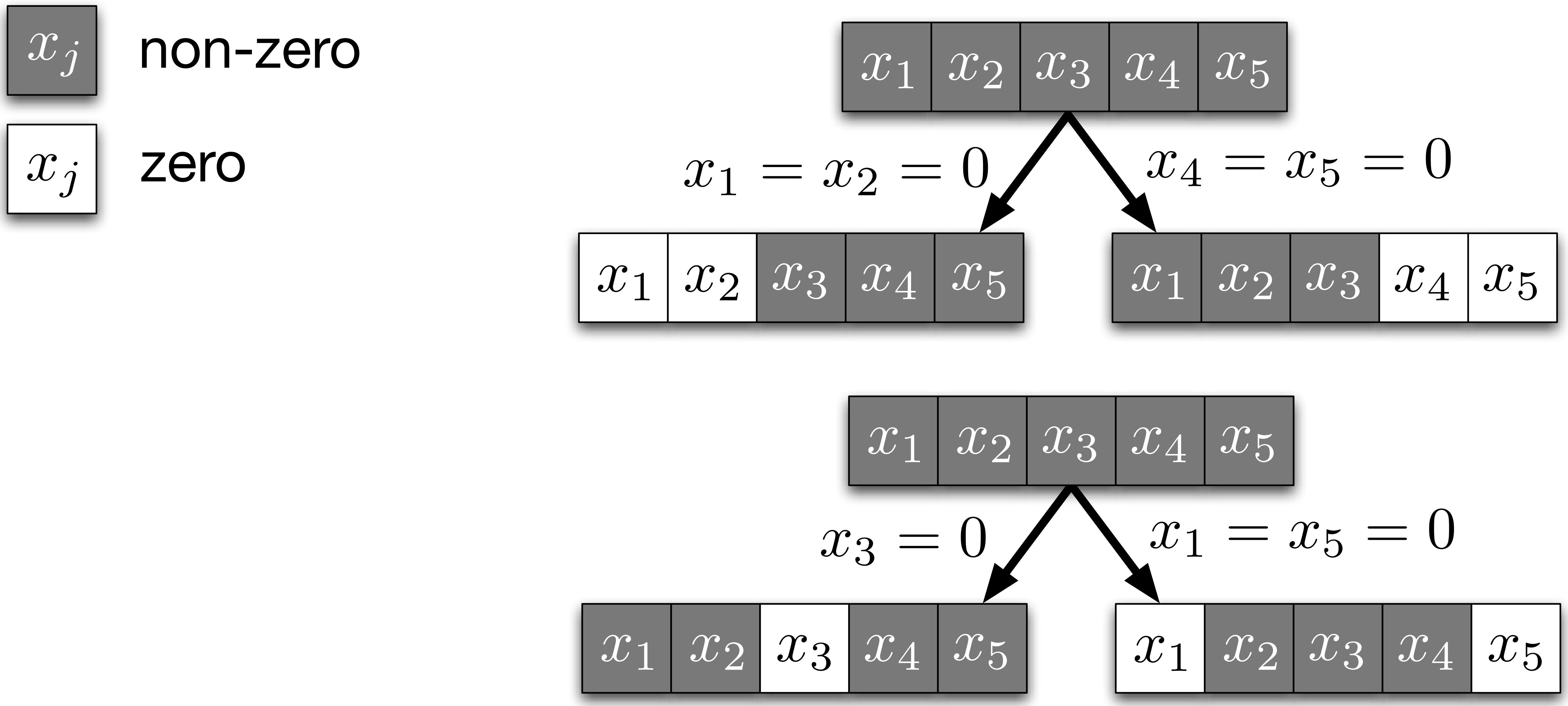
x_j zero



Independent Branching for SOS2



Independent Branching for SOS2

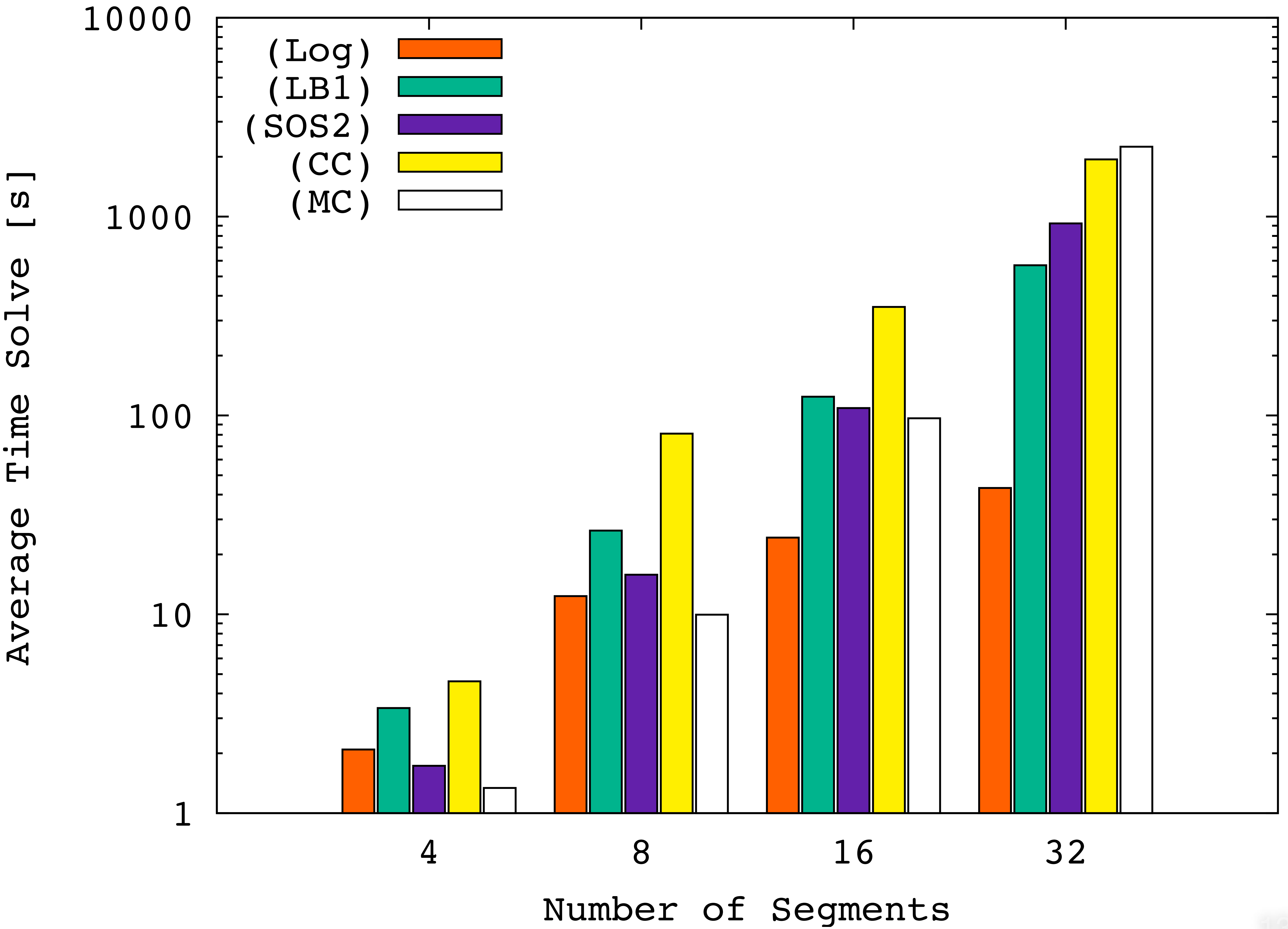


- Based on Gray Codes = More than one choice.

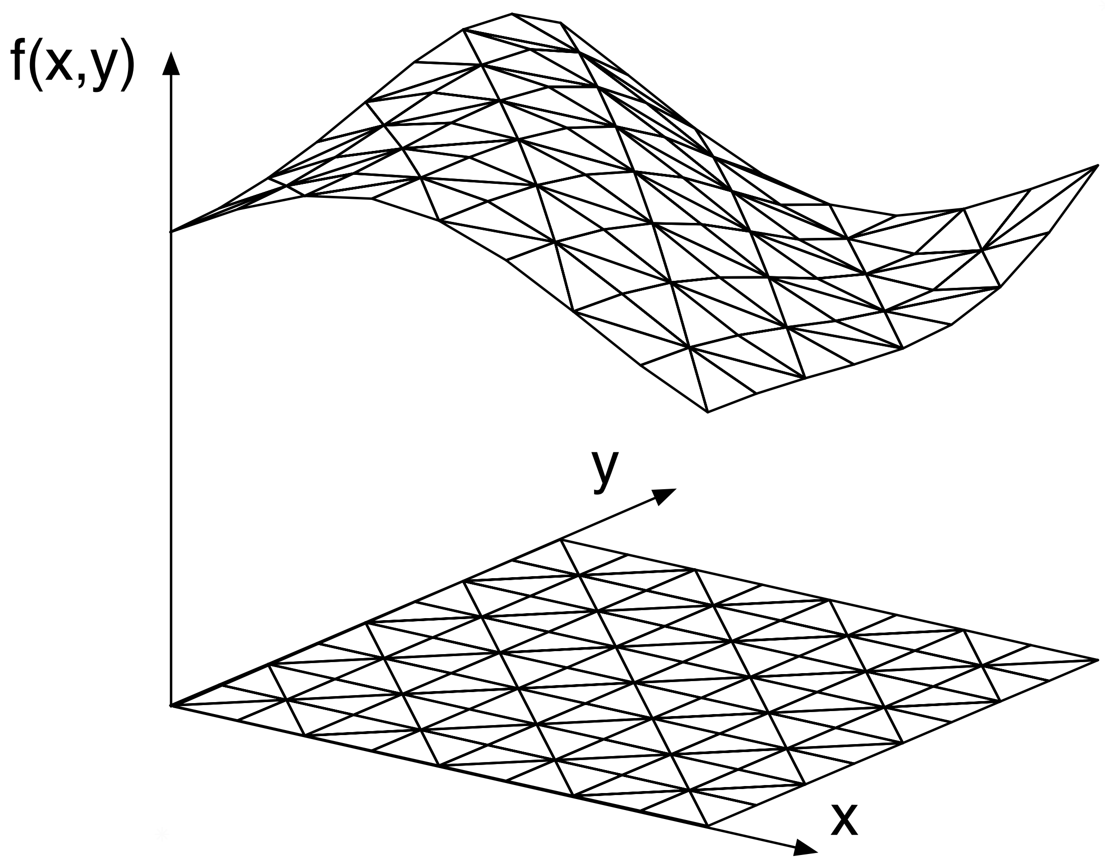
Computational Experiments

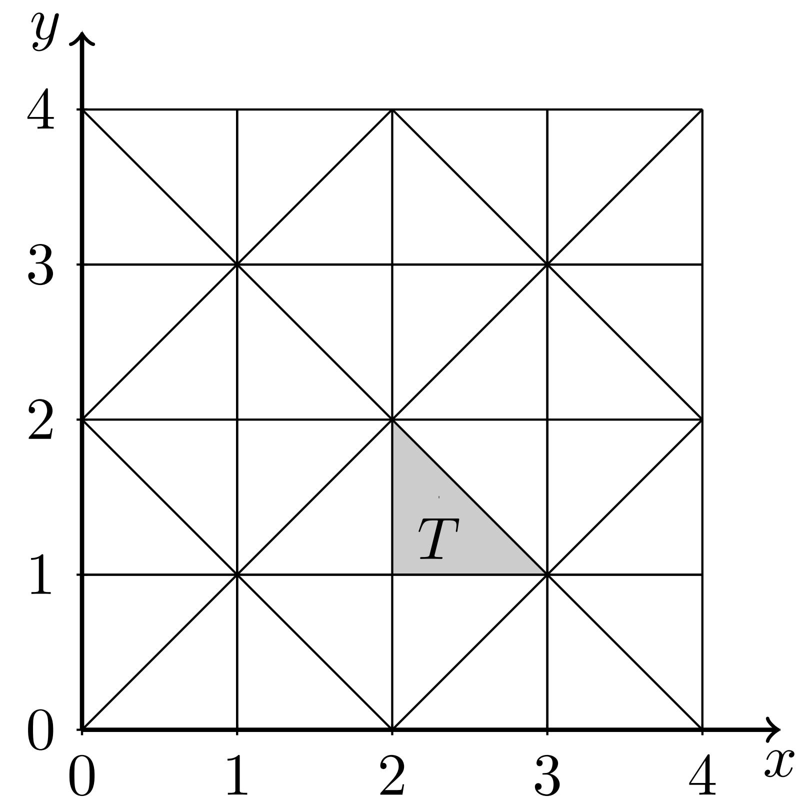
- Instances: Transp. probs. w. piecewise linear cost.
- CPLEX 11, 2.4GHz Xeon with 2GB of RAM.
  - Log: Log size Ind. Branch. for SOS2.
  - LB1: Linear size Ind. Br. for SOS2 (Shields, 07)
  - SOS2: CPLEX 11 specialized SOS2 branch.
  - CC: Standard formulation for SOS2.
  - MC: Non-SOS2 formulation for piecewise linear.

Computational Experiments

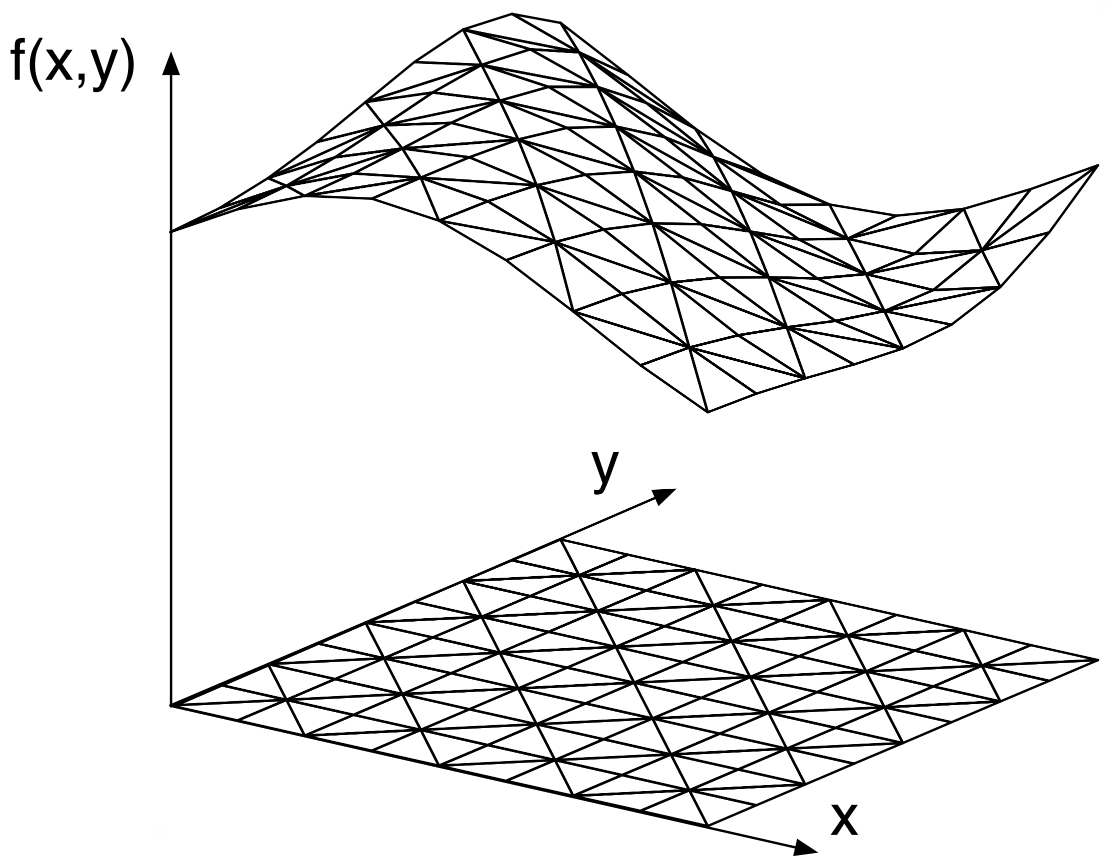


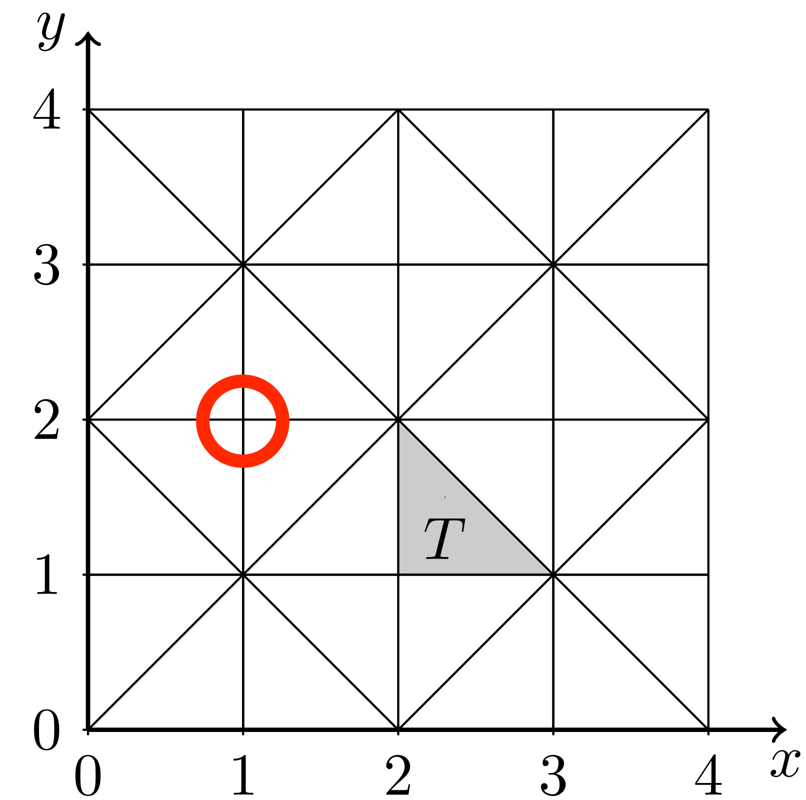
Multivariate Piecewise Linear Functions





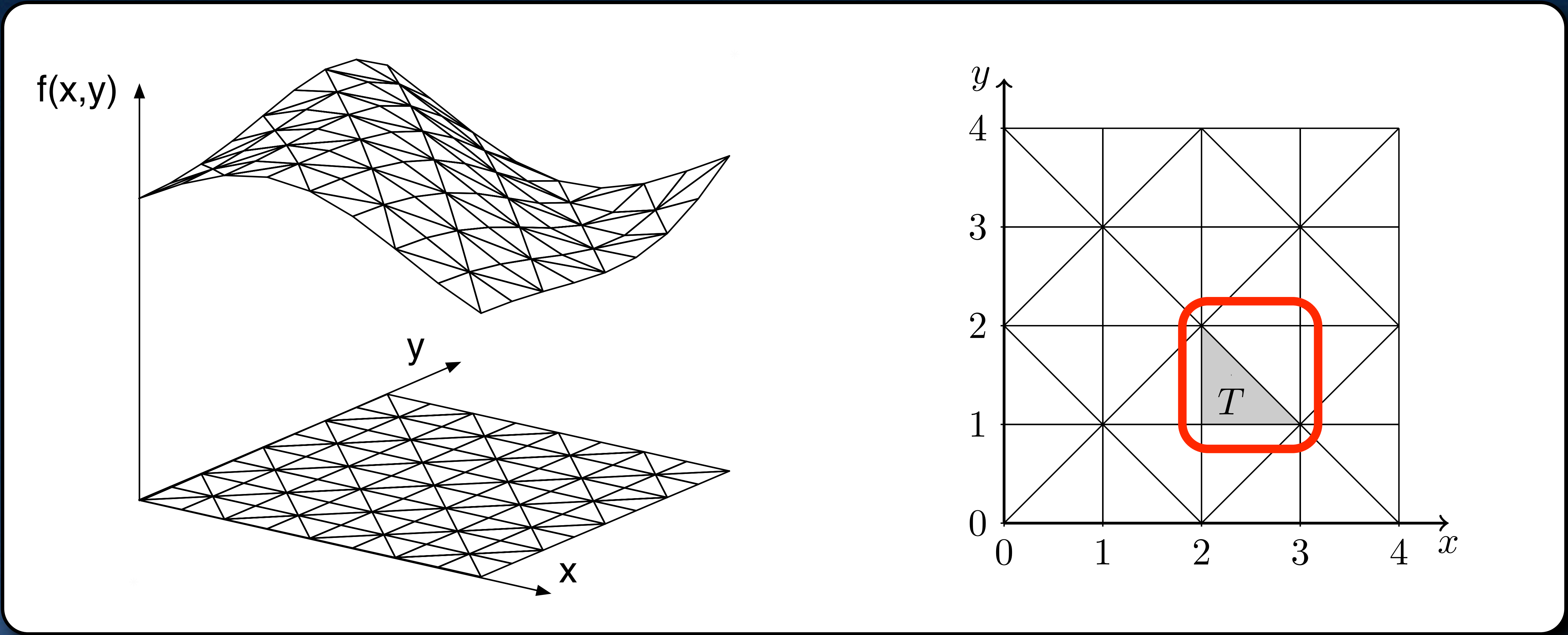
Multivariate Piecewise Linear Functions





- Variables = Vertices.

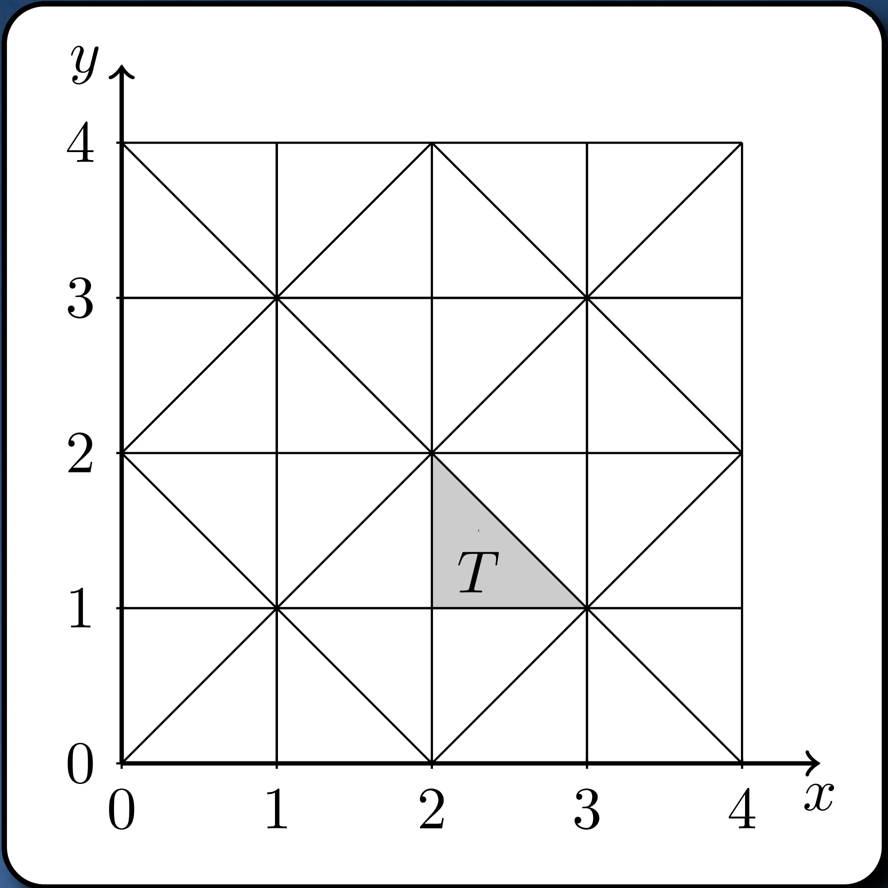
Multivariate Piecewise Linear Functions



- Variables = Vertices.
- Allowed non-zero variables = Vertices of a triangle.

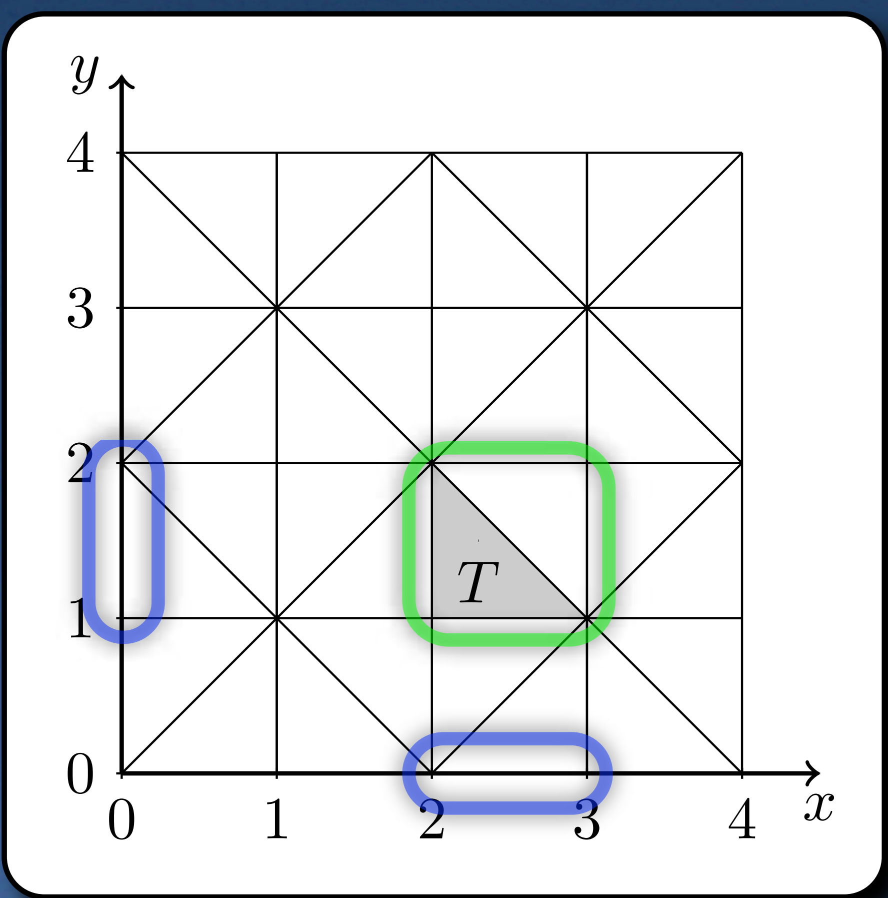
Independent Branching PWL Function

- Select Triangle by forbidding vertices.
- 2 stages:
  - Select Square by SOS2 on each variable.
  - Select 1 triangle from each square.



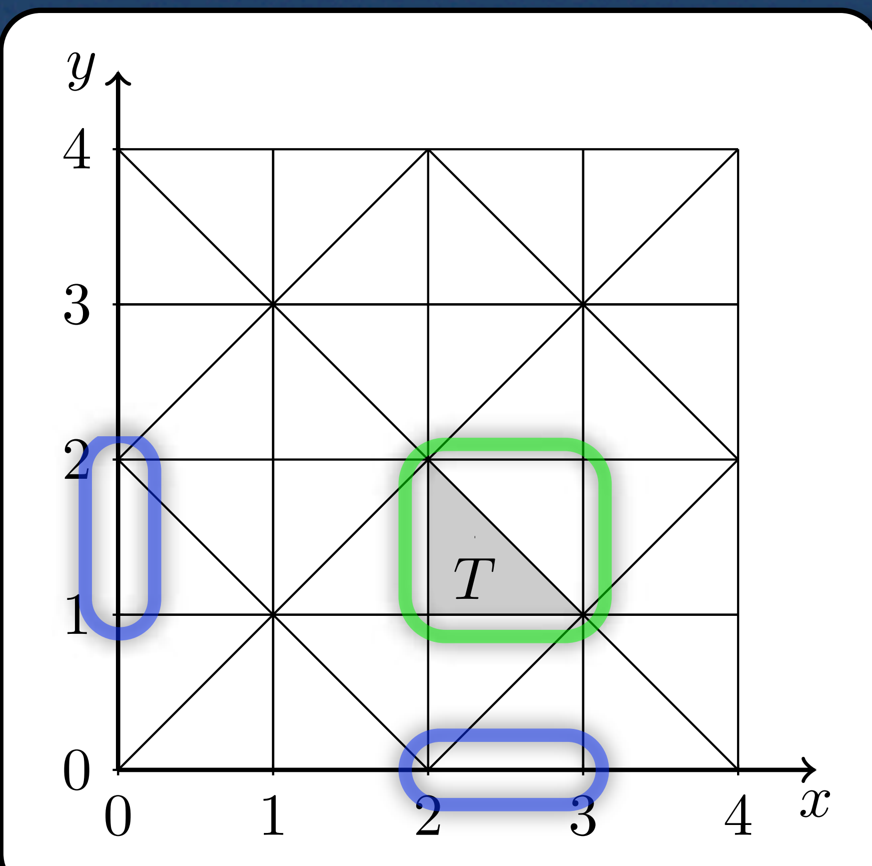
Independent Branching PWL Function

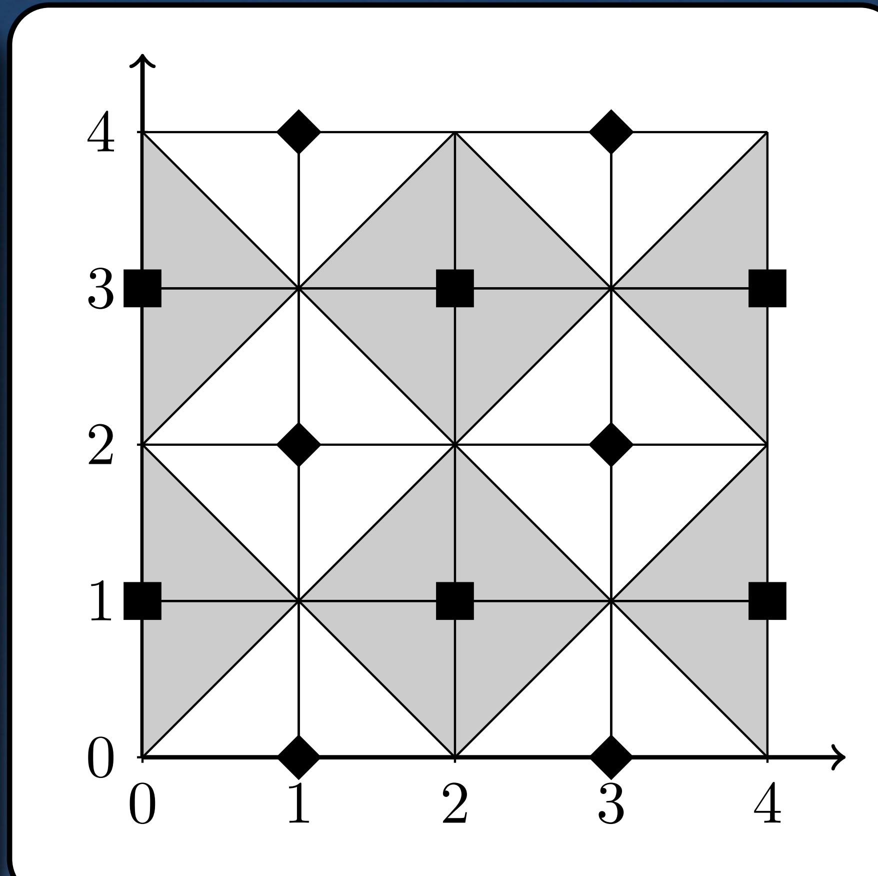
- Select Triangle by forbidding vertices.
- 2 stages:
  - Select Square by SOS2 on each variable.
  - Select 1 triangle from each square.



Independent Branching PWL Function

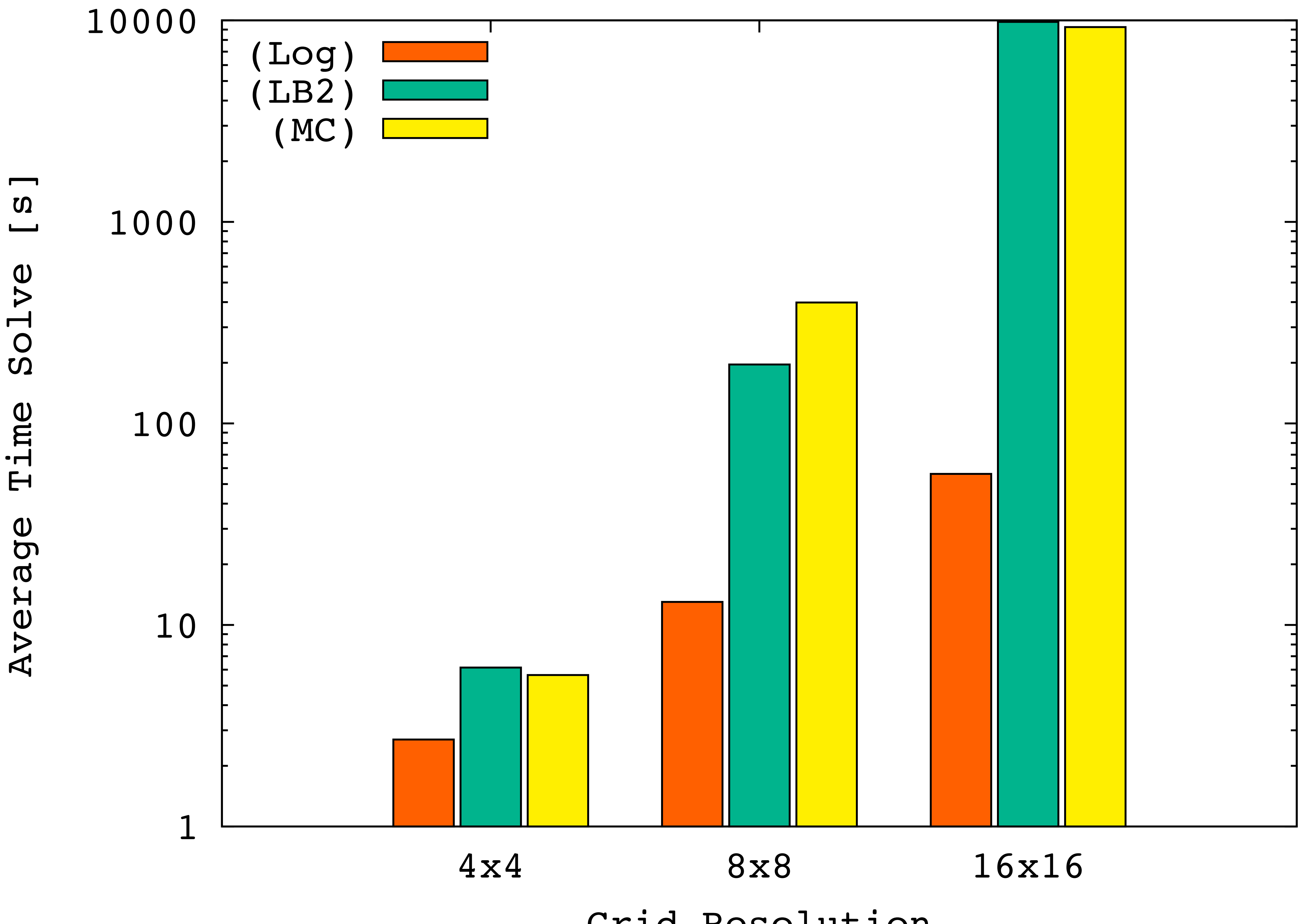
- Select Triangle by forbidding vertices.
- 2 stages:
  - Select Square by SOS2 on each variable.
  - Select 1 triangle from each square.





$$\bar{L} = \{(r,s) \in J : \; r \text{ even and } s \text{ odd}\} = \{\text{square vertices}\}$$

$$\bar{R} = \{(r,s) \in J : \; r \text{ odd and } s \text{ even}\} = \{\text{diamond vertices}\}$$


Average Time Solve [s]
10000
1000
100
10
1
(Log)
(LB2)
(MC)
4x4
8x8
16x16
Grid Resolution