Discrete Optimization with Decision Diagrams

J. N. Hooker

Joint work with
David Bergman, André Ciré, Willem van Hoeve
Carnegie Mellon University

Australian OR Society, May 2014

Goal

- Find an alternative research direction for **general-purpose** discrete optimization.
  - Given the maturity of IP technology.

Goal

- Find an alternative research direction for **general-purpose** discrete optimization.
  - Given the maturity of IP technology
- How? Explore **alternative problem representations**.
  - Use **knowledge representation** techniques from AI.
    - Decision diagrams, and/or graphs, join graphs, Bayesian networks, etc.
  - We focus on **decision diagrams**.

Goal

- Find an alternative research direction for **general-purpose** discrete optimization.
  - Given the maturity of IP technology
- How? Explore **alternative problem representations**.
  - Use **knowledge representation** techniques from AI.
    - Decision diagrams, and/or graphs, join graphs, Bayesian networks, etc.
  - We focus on **decision diagrams**.
- Use a **model** with constant size as instances scale up.
  - To solve instances **too large to load** in a MIP solver.
  - Distinguish model from relaxation (e.g. inequalities).

Optimization with Decision Diagrams

- Idea: let decision diagrams play the role of LP relaxation.
  - **Relaxed** decision diagrams provide **bounds**.
  - **Restricted** decision diagrams provide **primal heuristic**.
  - **Size** of decision diagram is **adjustable**.

Optimization with Decision Diagrams

- Idea: let decision diagrams play the role of LP relaxation.
  - **Relaxed** decision diagrams provide **bounds**.
  - **Restricted** decision diagrams provide **primal heuristic**.
  - **Size** of decision diagram is **adjustable**.
- Novel branching scheme
  - **Branch in decision diagram** rather than on variables.

Optimization with Decision Diagrams

- Idea: let decision diagrams play the role of LP relaxation.
  - **Relaxed** decision diagrams provide **bounds**.
  - **Restricted** decision diagrams provide **primal heuristic**.
  - **Size** of decision diagram is **adjustable**.
- Novel branching scheme
  - **Branch in decision diagram** rather than on variables.
- Constant size model
  - **DP-style** model of problem.
  - State space size is no concern.

Decision Diagrams

- **Binary decision diagrams (BDDs)** historically used for circuit design and verification.
  - Lee 1959, Akers 1978, Bryant 1986.

Decision Diagrams

- **Binary decision diagrams (BDD**s**)** historically used for circuit design and verification.
  - Lee 1959, Akers 1978, Bryant 1986.
- **Compact** graphical representation of **boolean** function.
  - Can also represent **feasible set** of problem with binary variables.
  - Slight generalization (MDDs) represents finite domain variables.
  - **Reduced** BDD is result of superimposing isomorphic subtrees in a search tree.

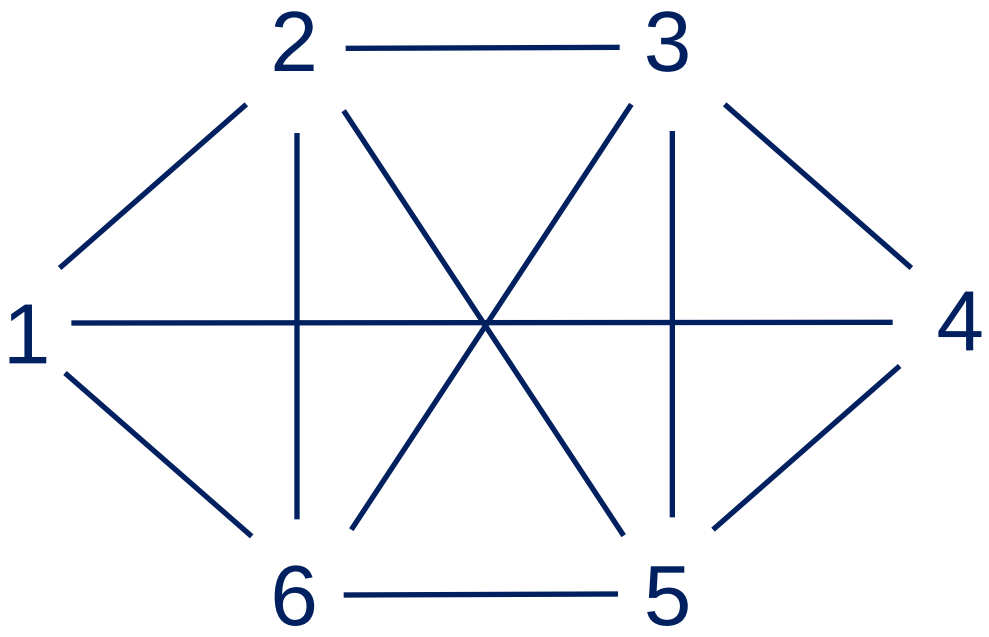
Brief History: BDDs in Optimization

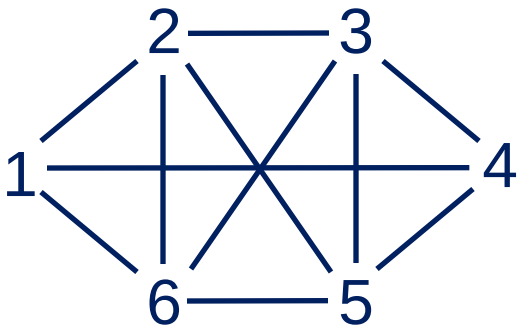
- BDDs for cut generation
  - Becker, Behle, Eisenbrand, Wimmer (2005)
- BDDs for 0-1 global optimization, postoptimality analysis
  - Hadžić & Hooker (2006, 2007)
- Relaxed BDDs for propagation, bounds
  - Andersen, Hadžić, Hooker, Tiedemann (2007)
  - Bergman, Ciré, van Hoeve, Hooker (2013)
- Restricted BDDs for primal heuristics
  - Bergman, Ciré, van Hoeve, Yunes (submitted)

Stable Set Problem

Let each vertex have weight w_i

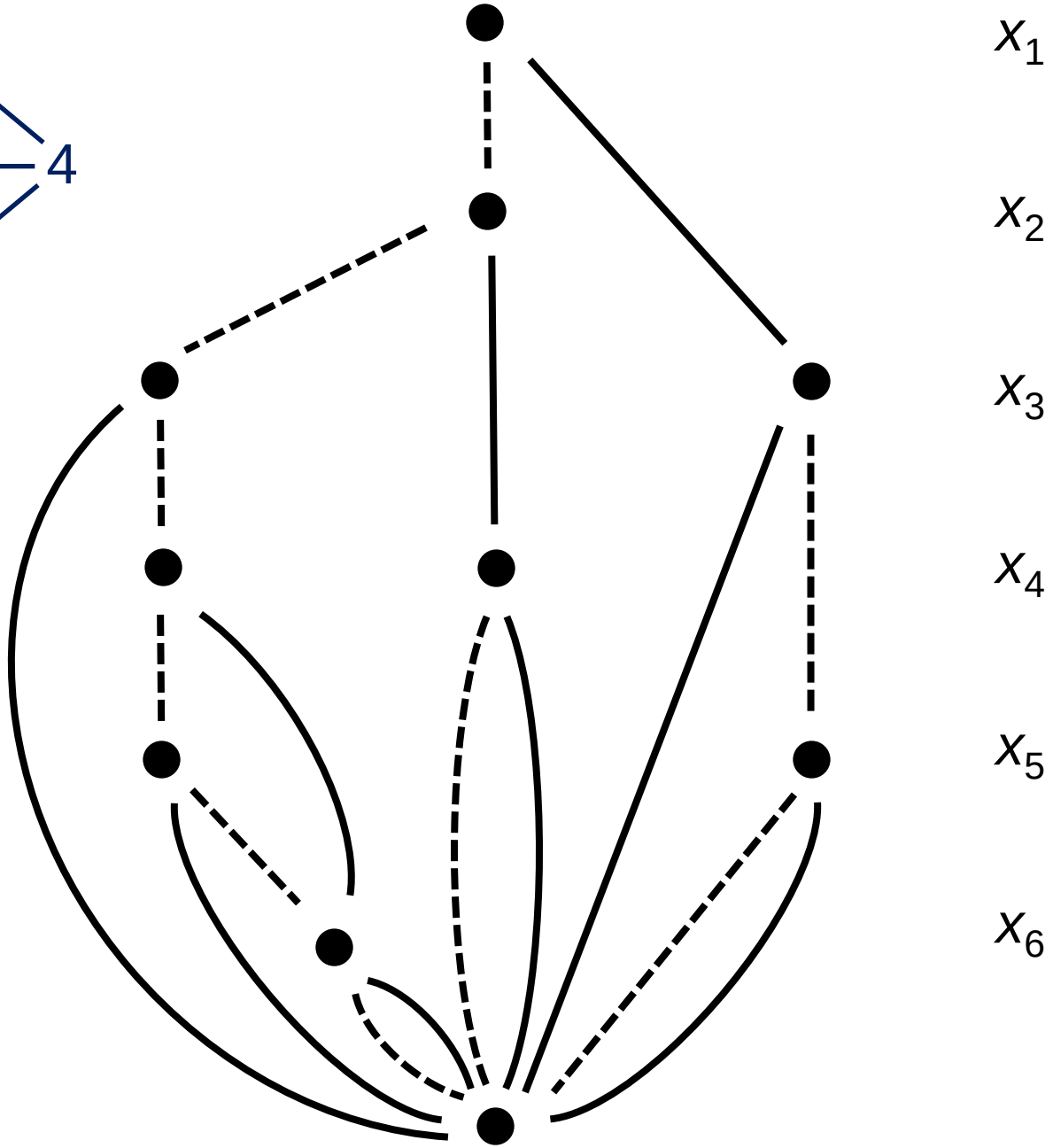
Select nonadjacent vertices to maximize $\sum_i w_i x_i$

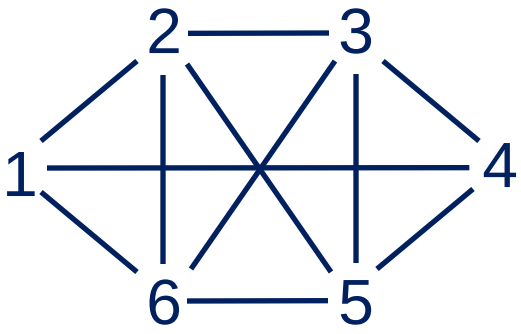




Exact BDD for stable set problem

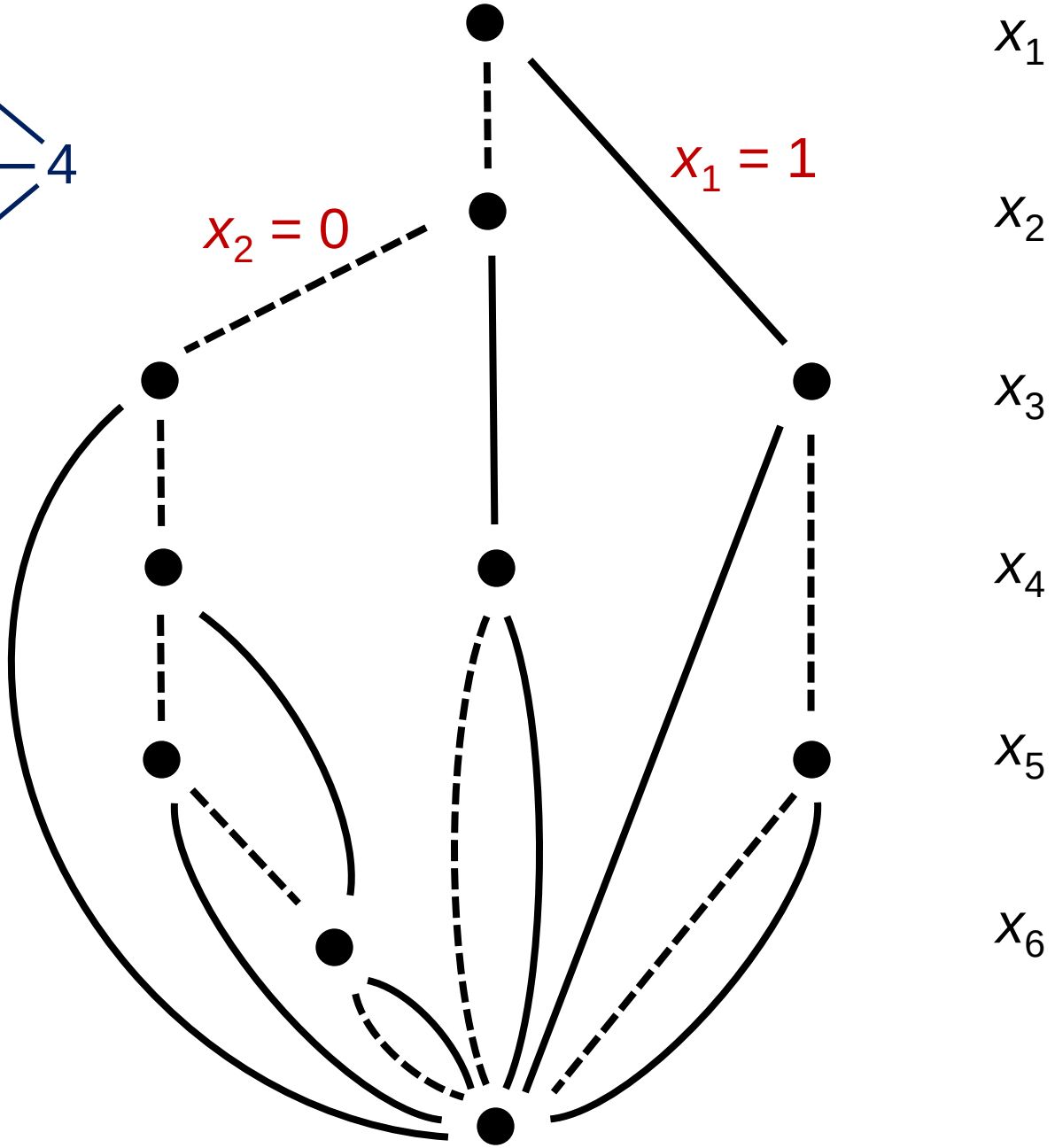
"zero-suppressed" BDD

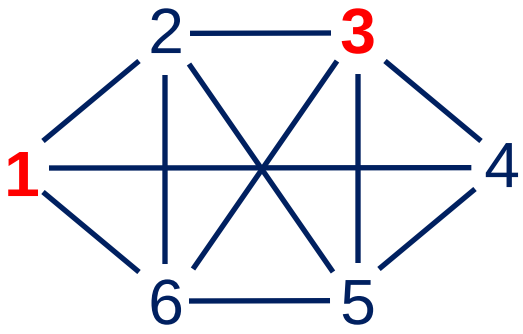




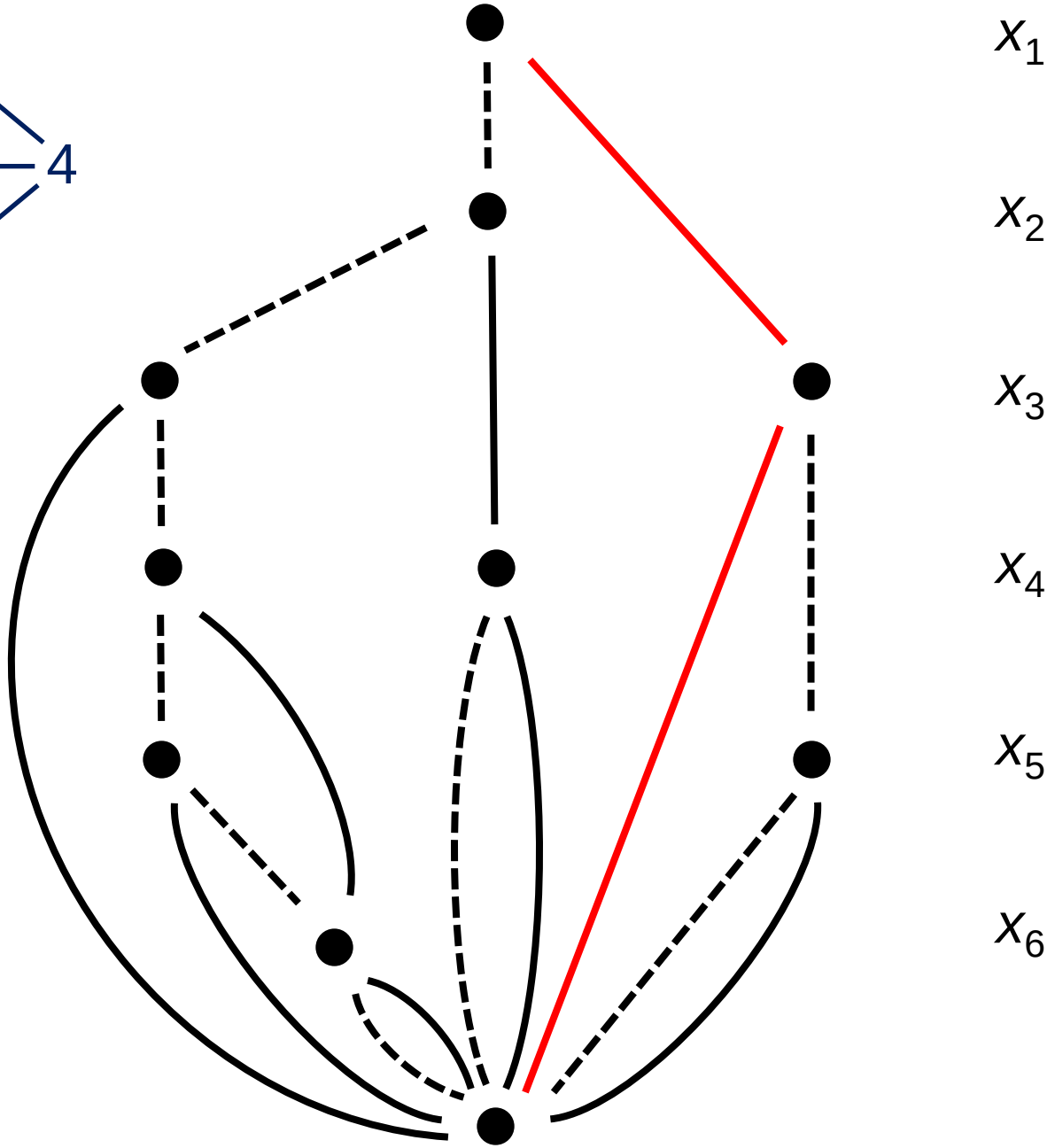
Exact BDD for stable set problem

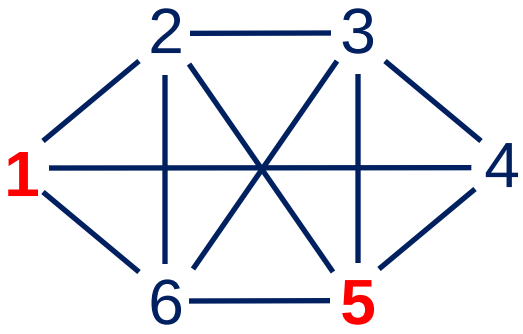
"zero-suppressed" BDD



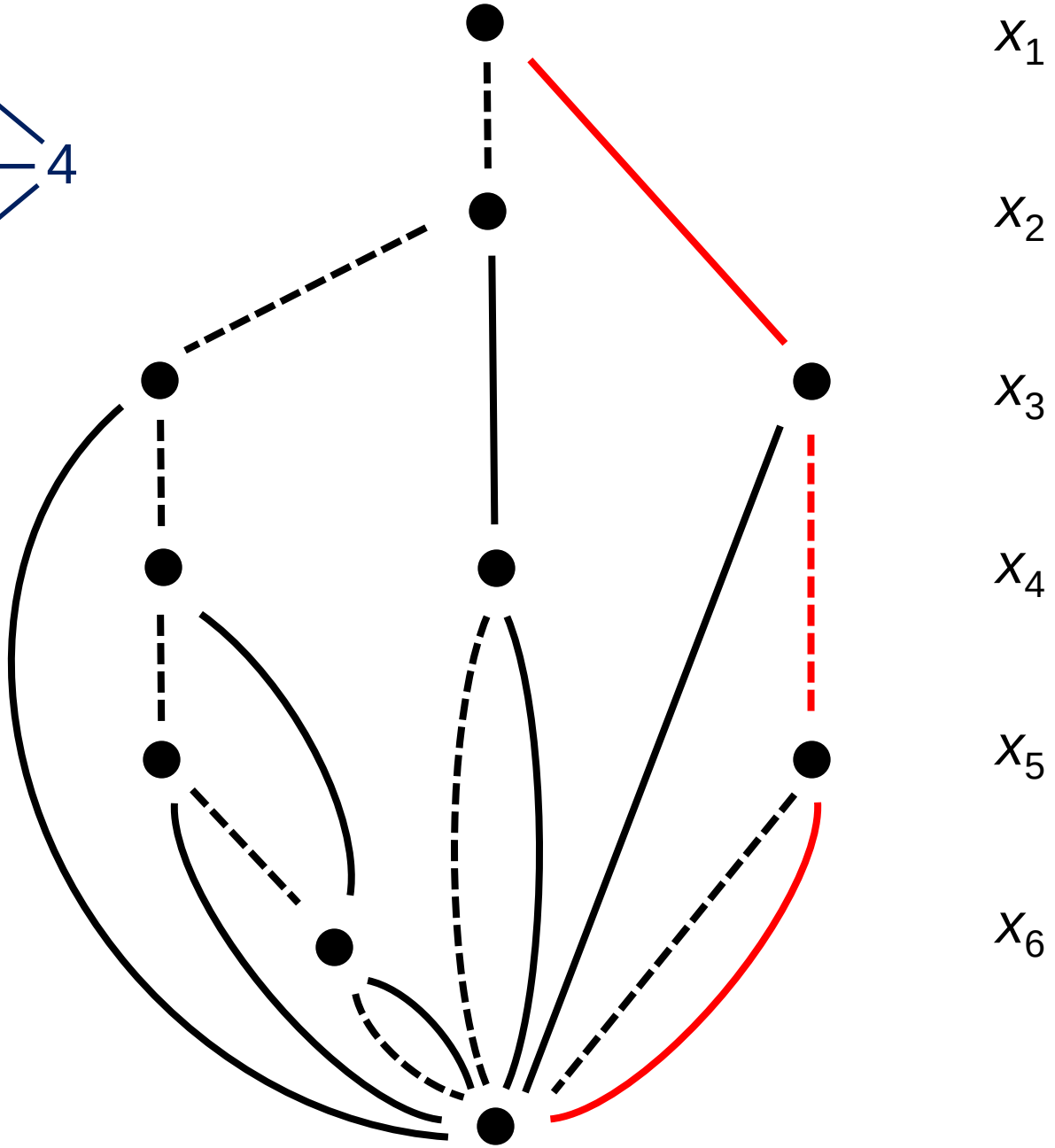


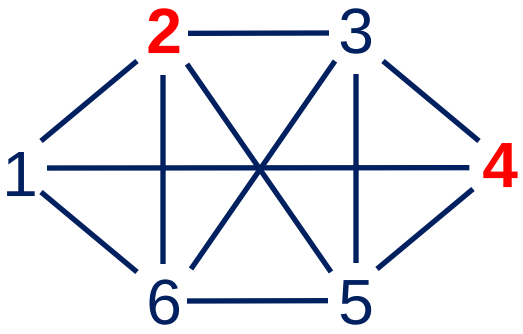
Paths from top to bottom correspond to the 11 feasible solutions



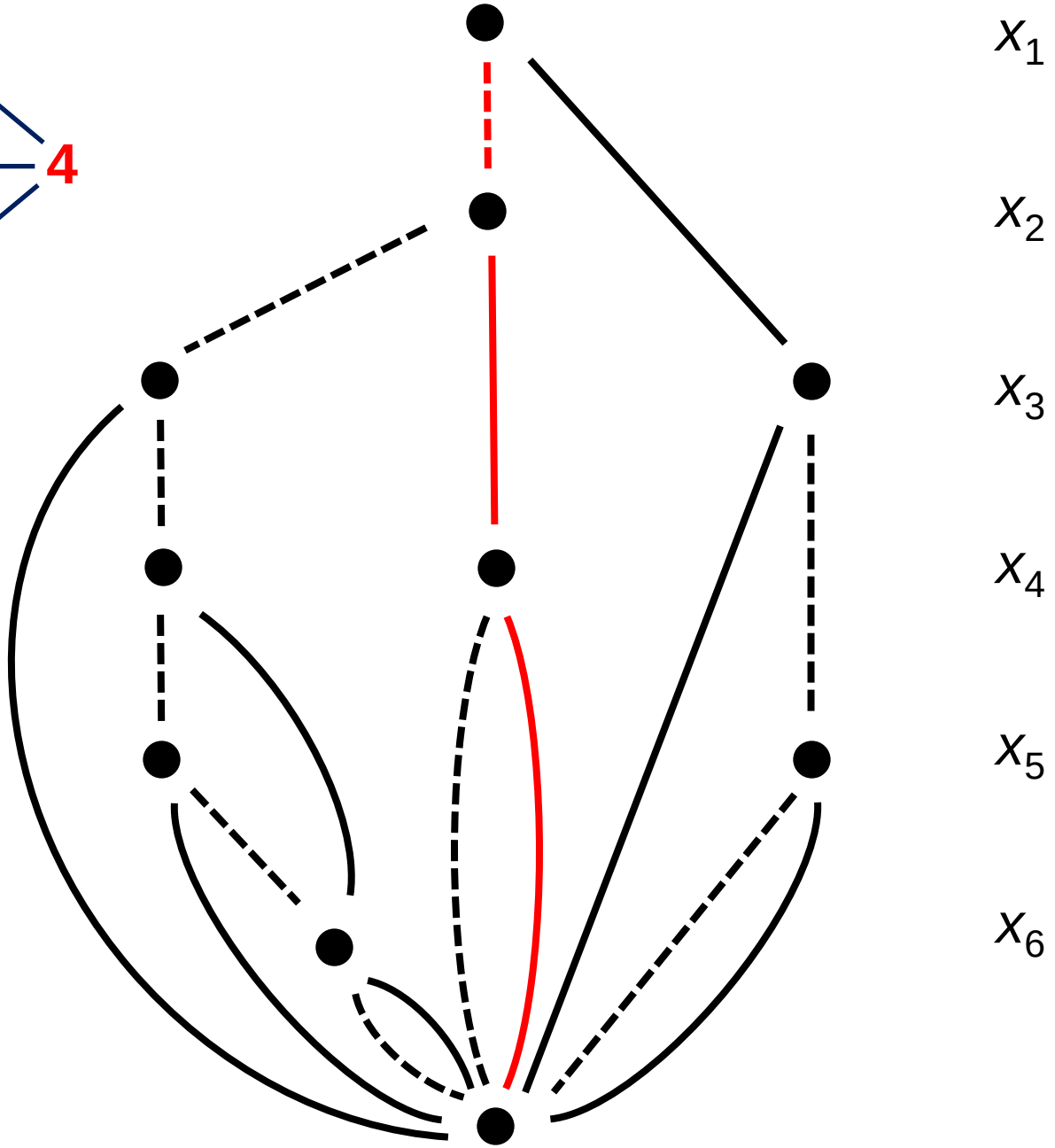


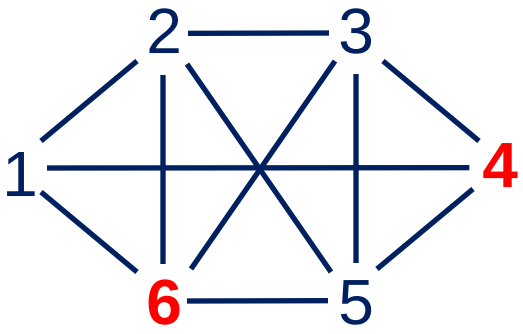
Paths from top to bottom correspond to the 11 feasible solutions



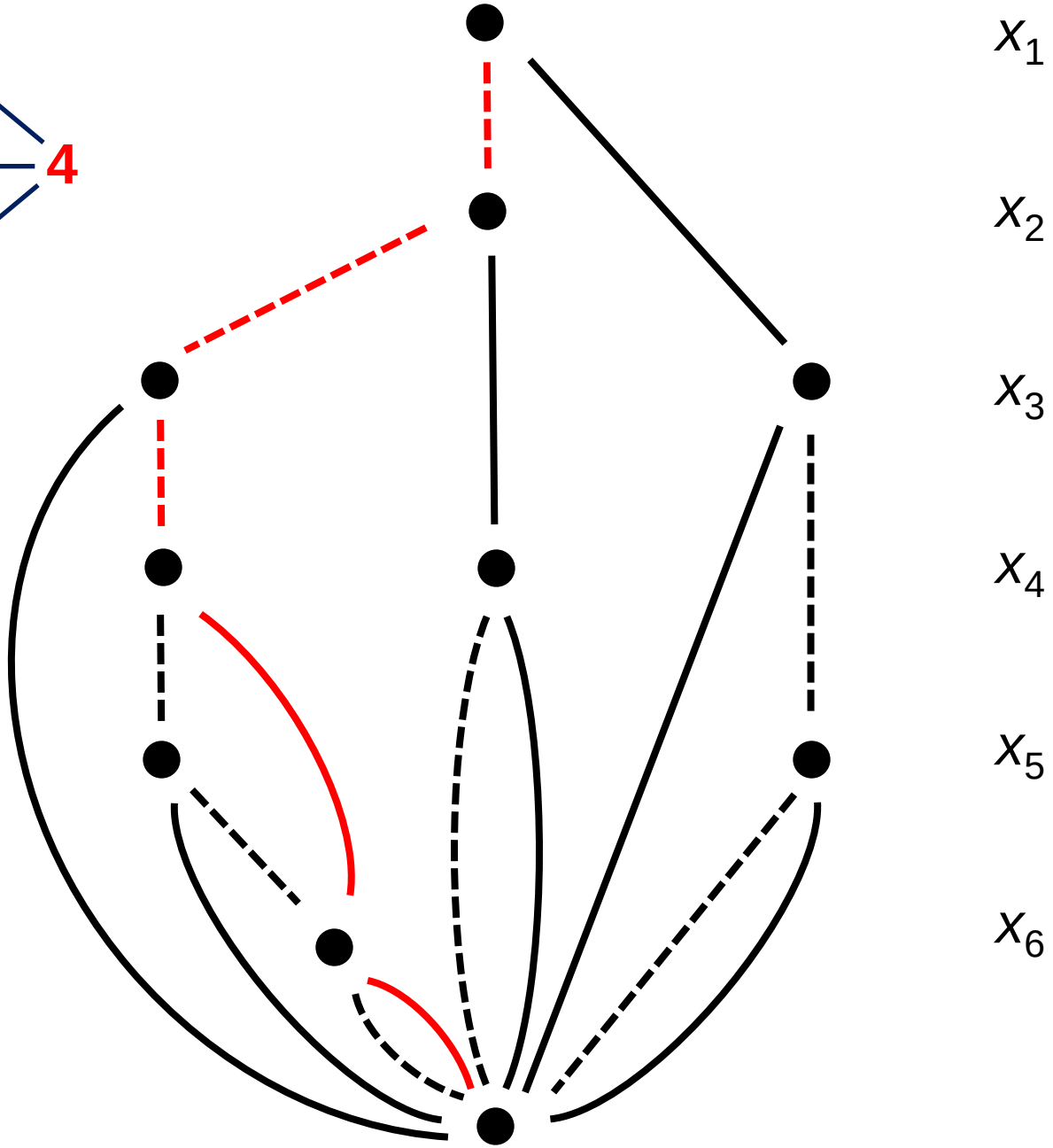


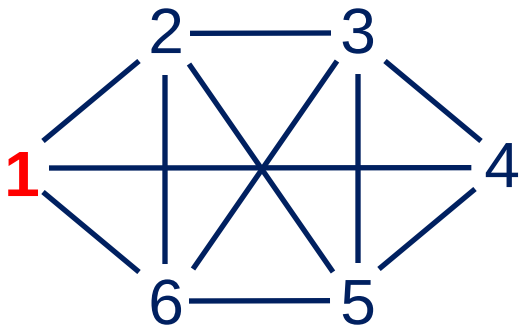
Paths from top to bottom correspond to the 11 feasible solutions





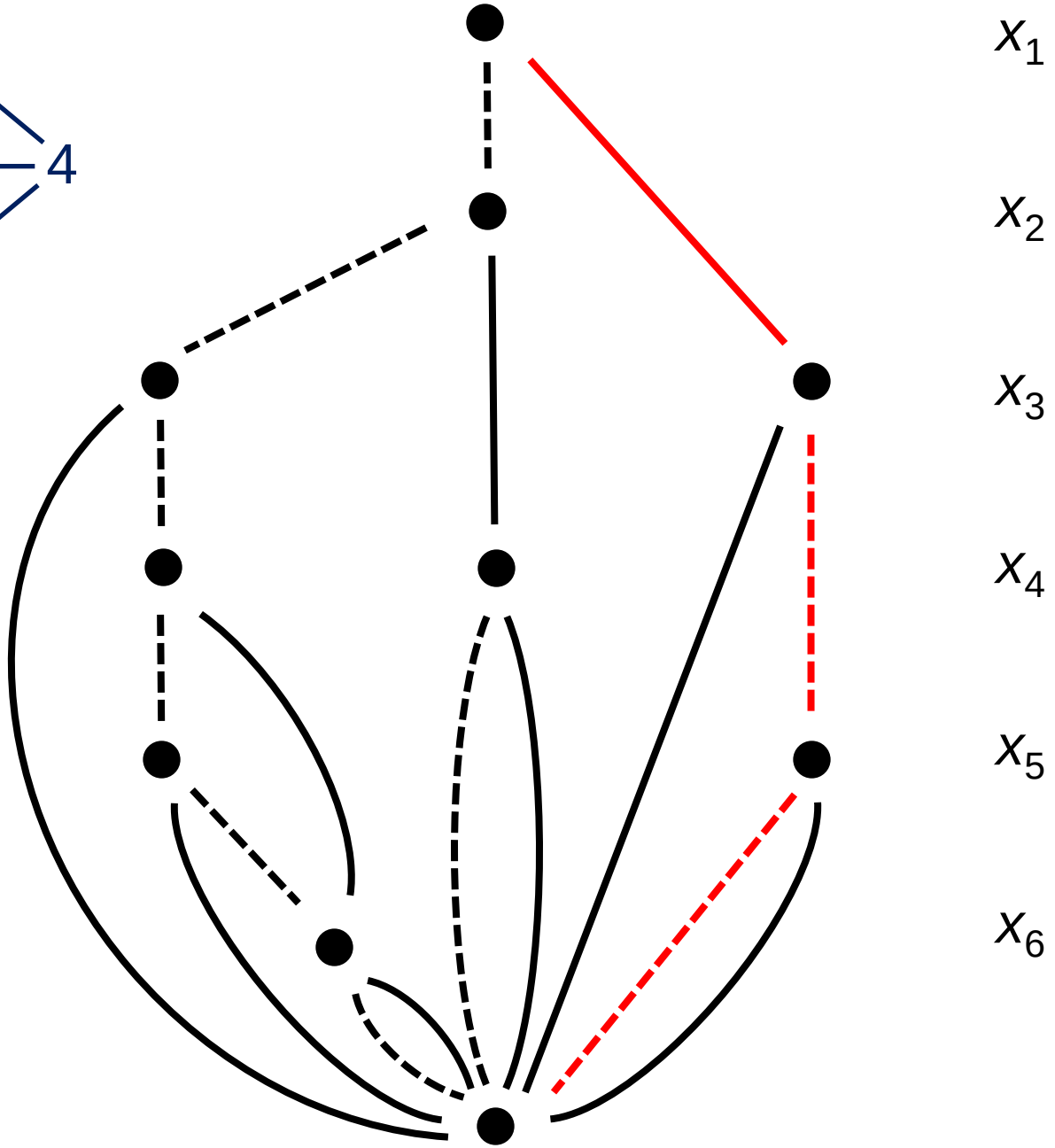
Paths from top to bottom correspond to the 11 feasible solutions

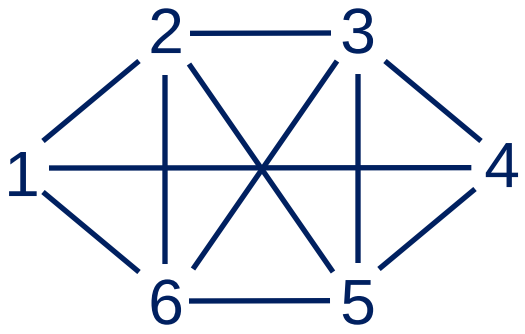




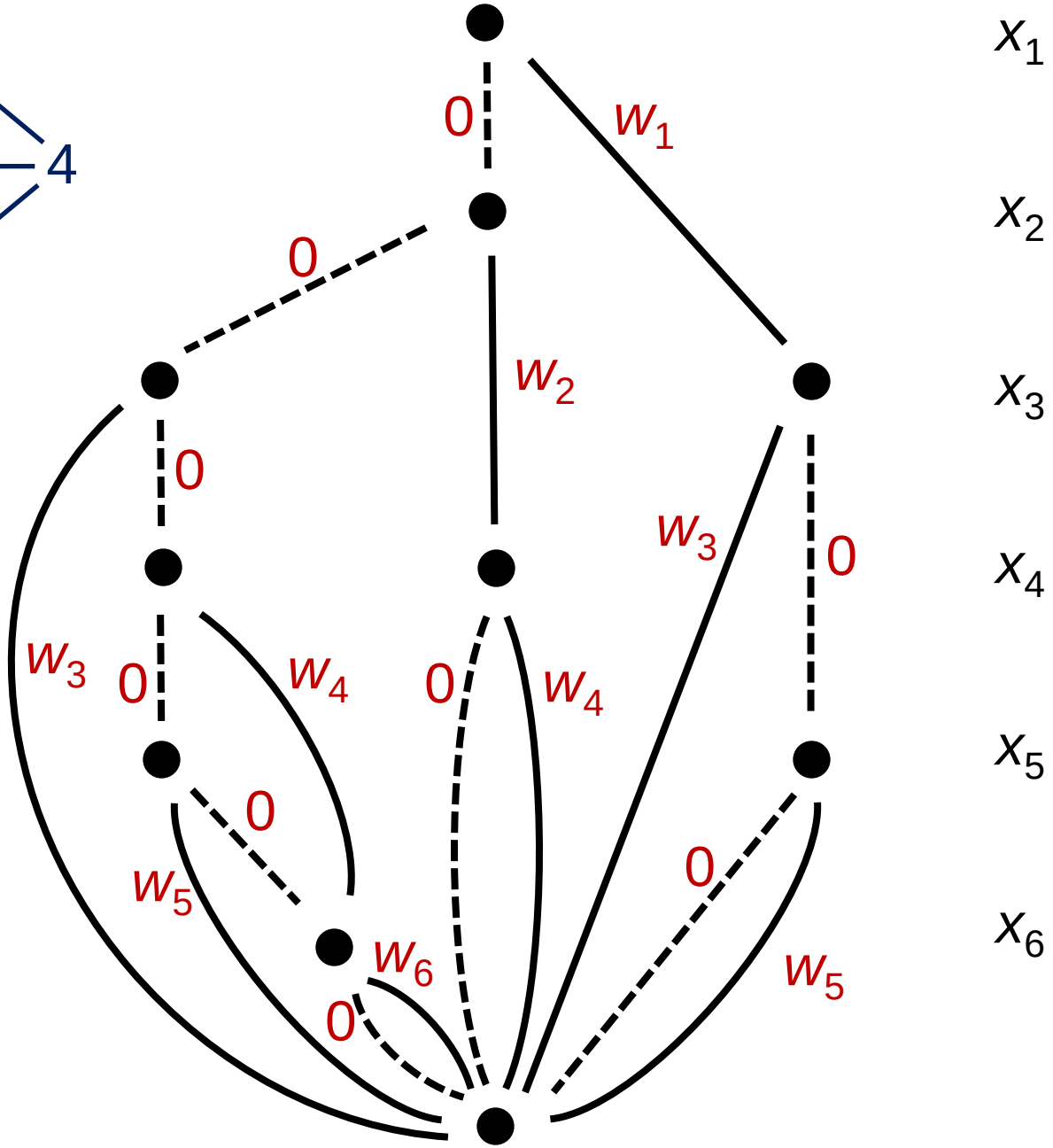
Paths from top to bottom correspond to the 11 feasible solutions

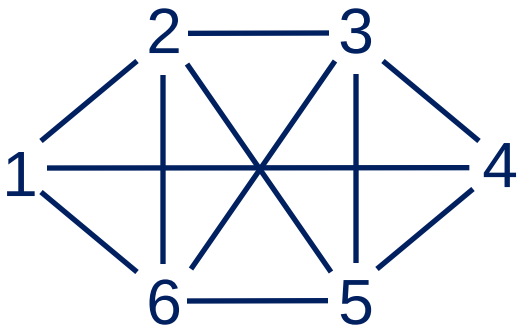
…and so forth





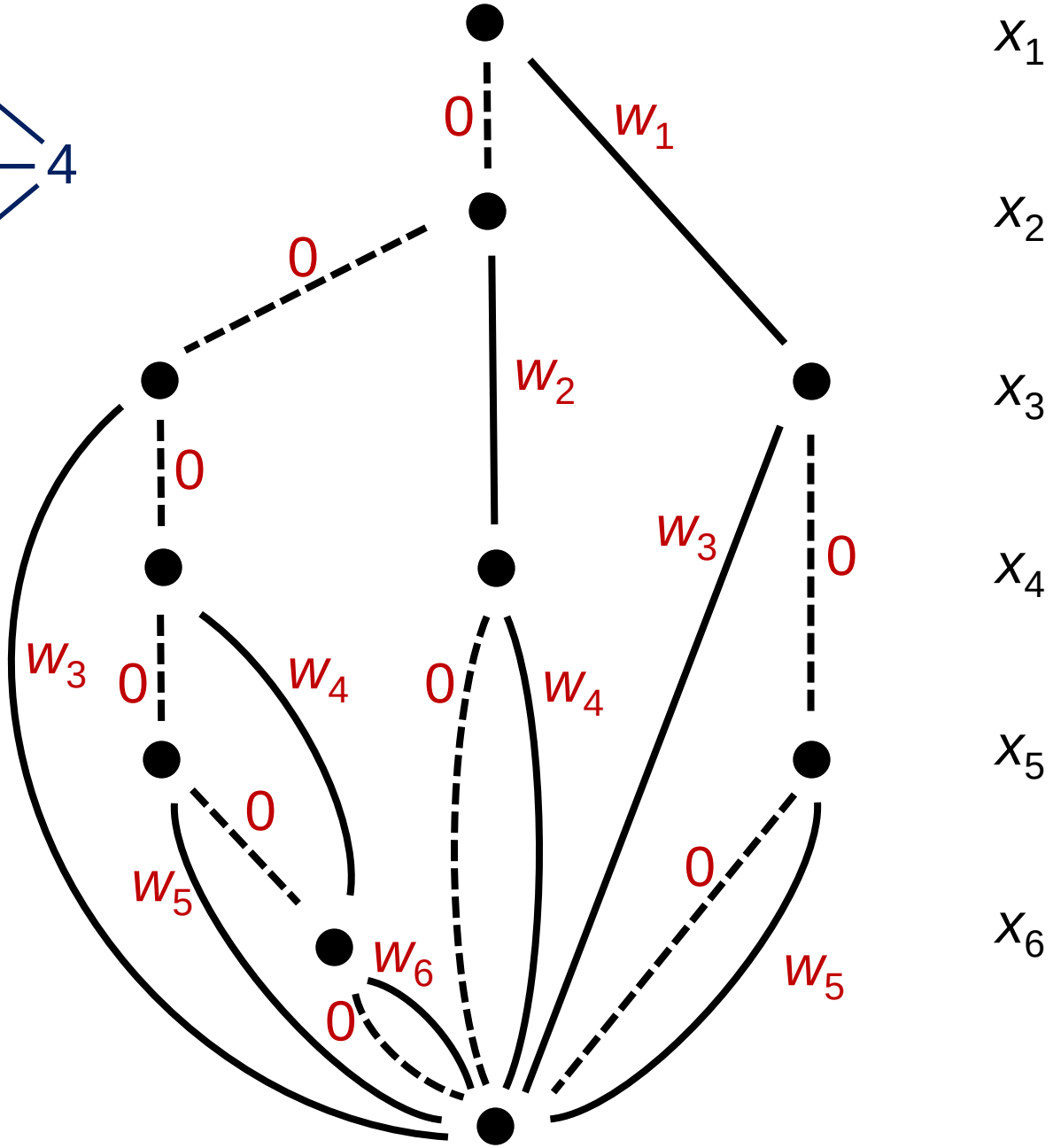
For objective function, associate weights with arcs





For objective function, associate weights with arcs

Optimal solution is **longest path**

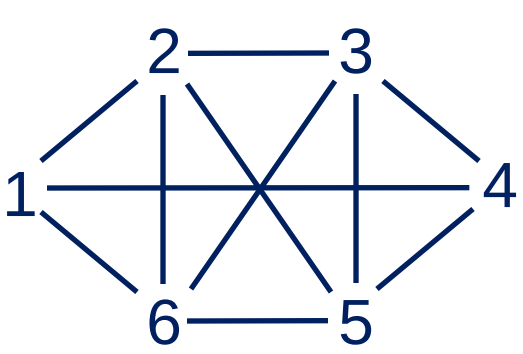


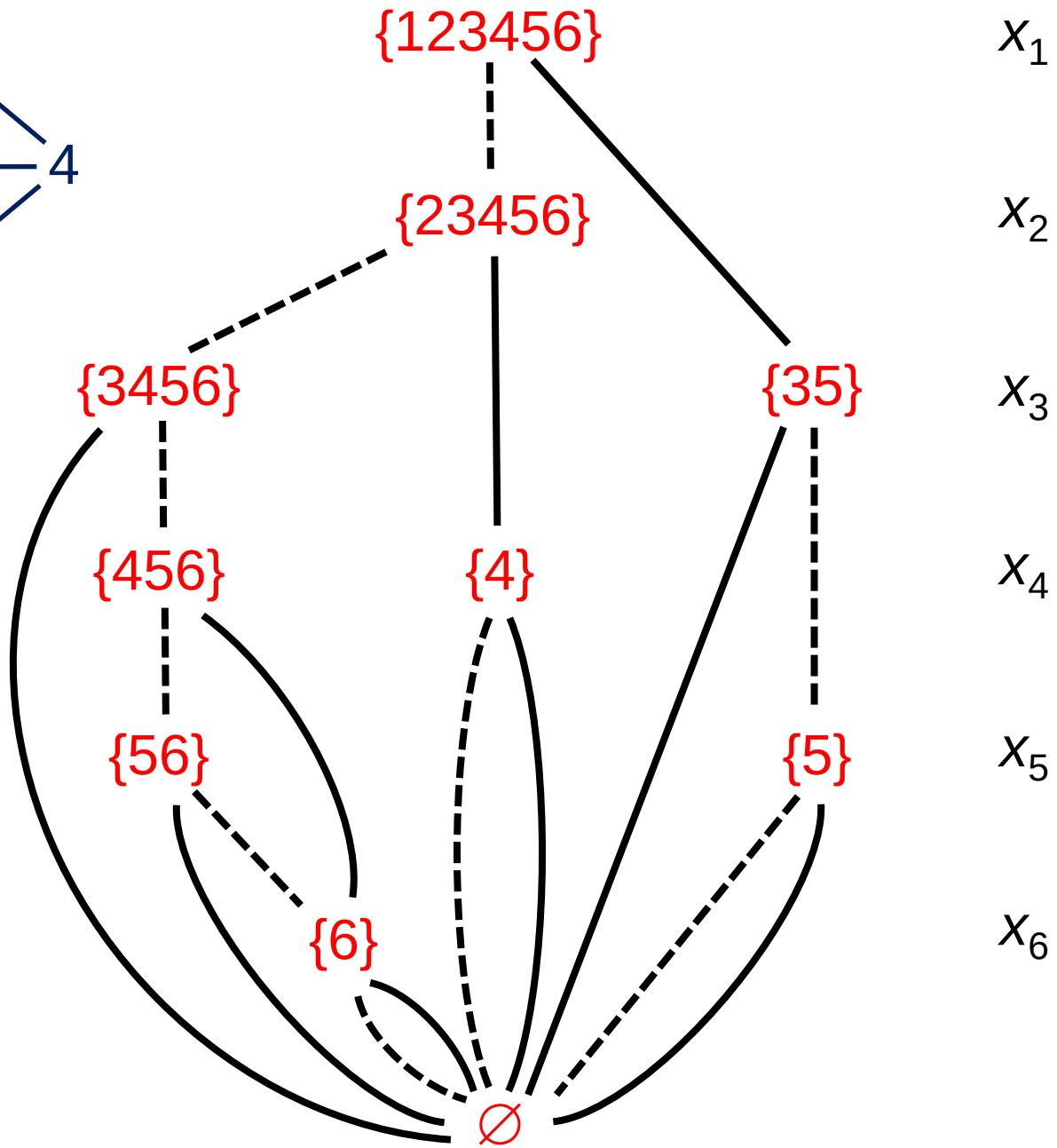
Objective Function

- In general, objective function can be any **separable function.**
  - Linear or nonlinear, convex or nonconvex
- BDDs can be generalized to **nonseparable** objective functions**.**
  - There is a unique reduced BDD with **canonical** edge costs.

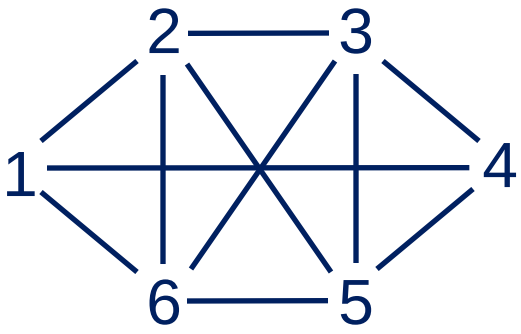
DP-Style Modeling

- Model has two components**.**
  - **DP model** of problem, using **state variables**.
    - Analogous to inequality model in IP.
  - Rule for **merging states** to create relaxed BDD.
    - Analogous to adding valid inequalities in IP.





To build BDD, associate **state** with each node



{123456}

x_1

x_2

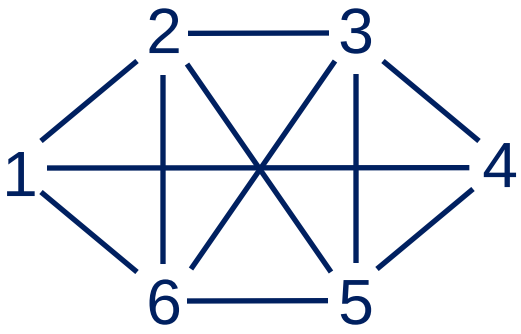
x_3

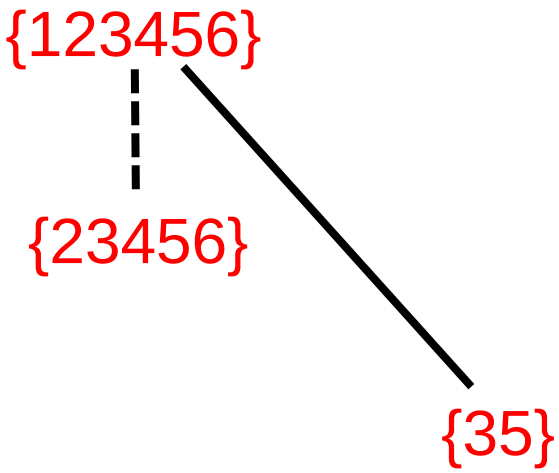
x_4

x_5

x_6

To build BDD, associate **state** with each node





x_1

x_2

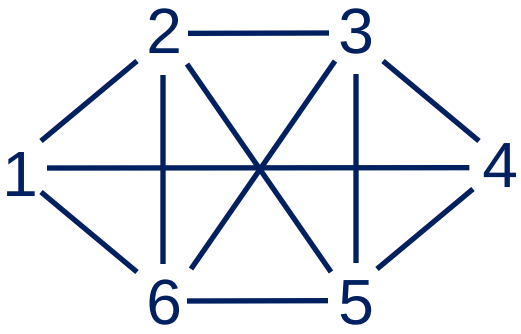
x_3

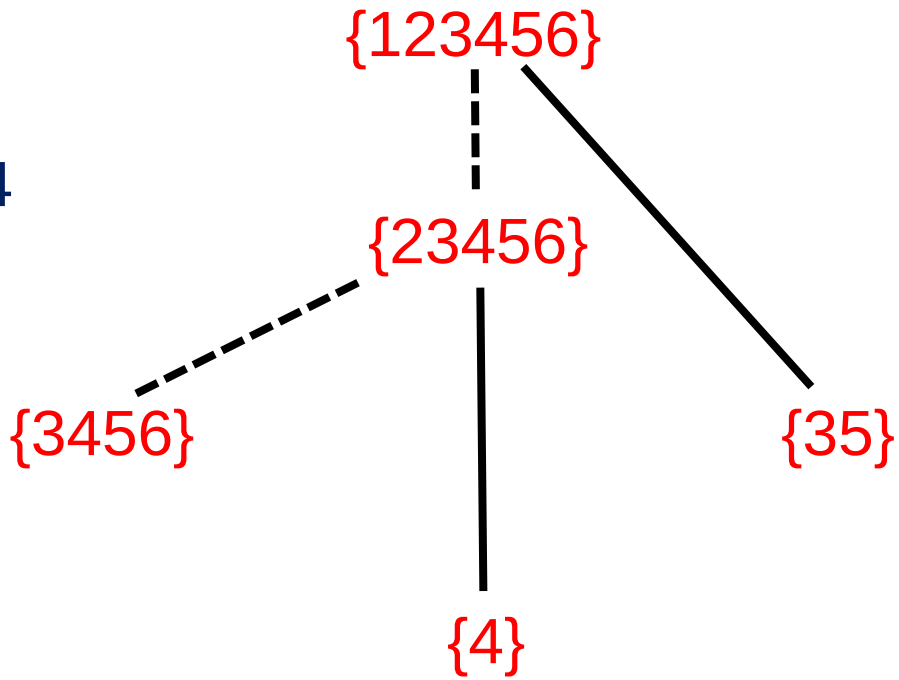
x_4

x_5

x_6

To build BDD, associate **state** with each node





x_1

x_2

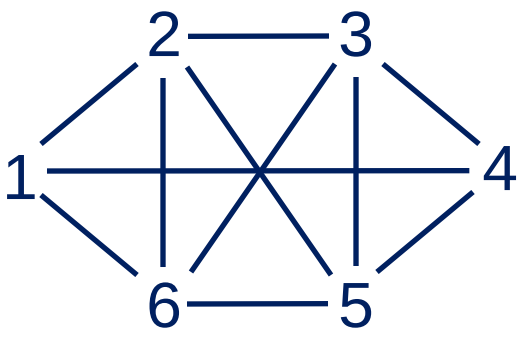
x_3

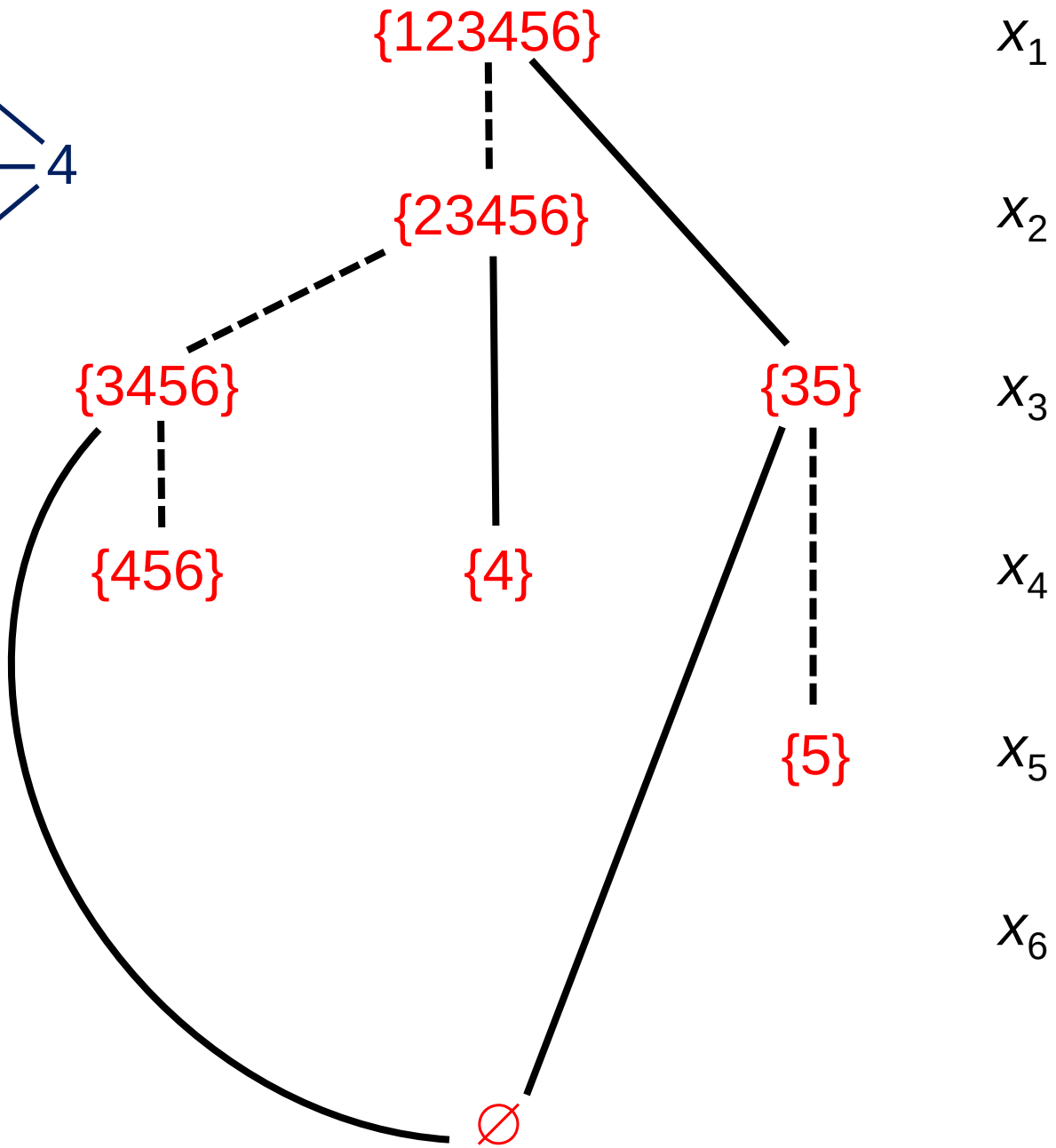
x_4

x_5

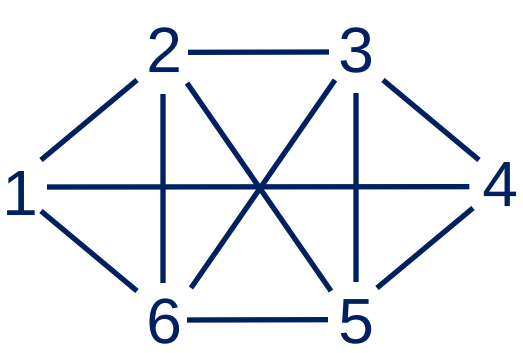
x_6

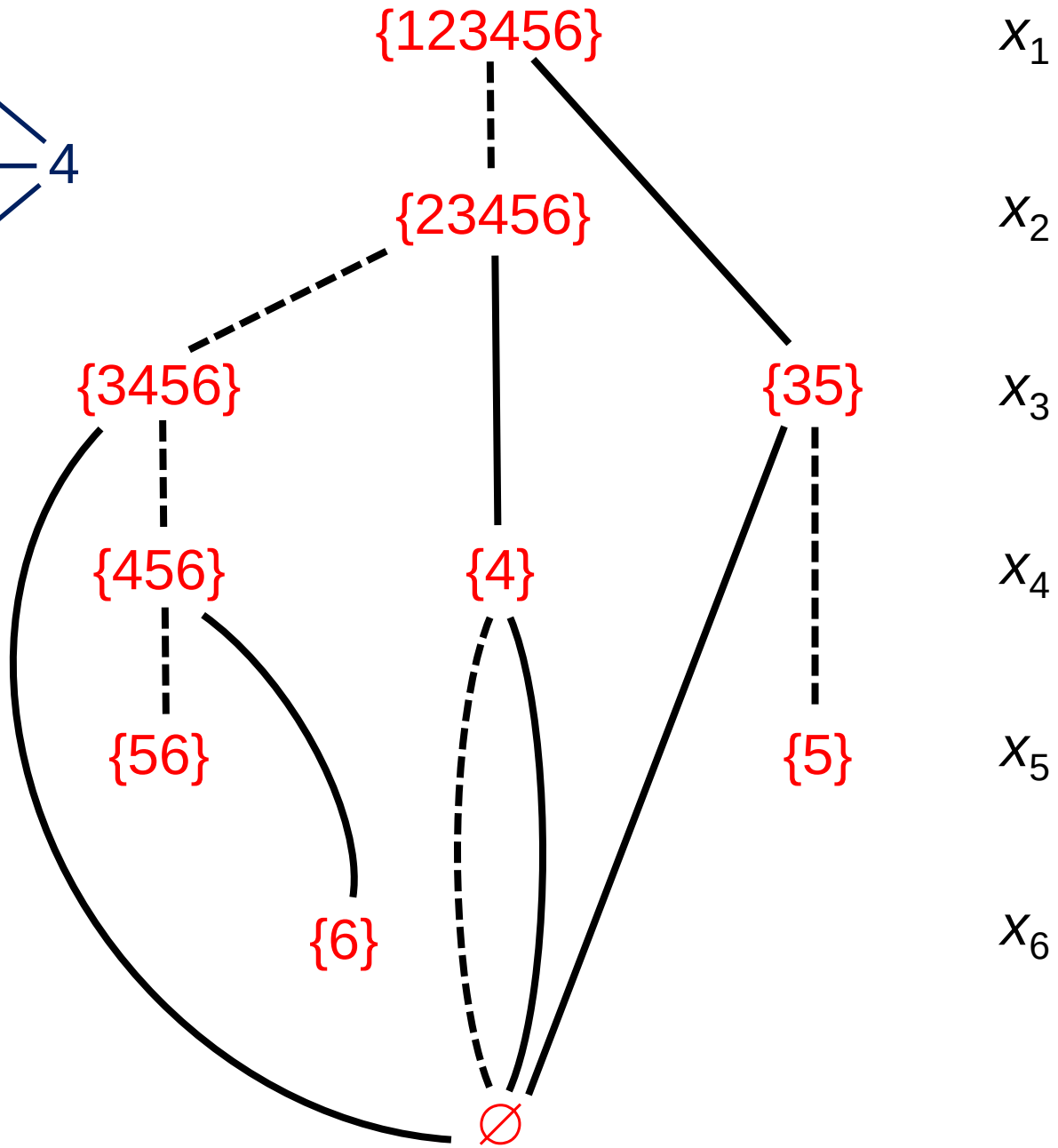
To build BDD, associate **state** with each node



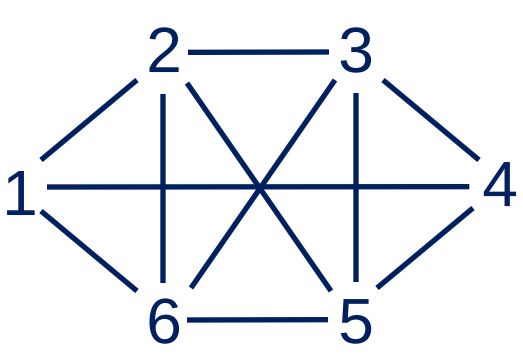


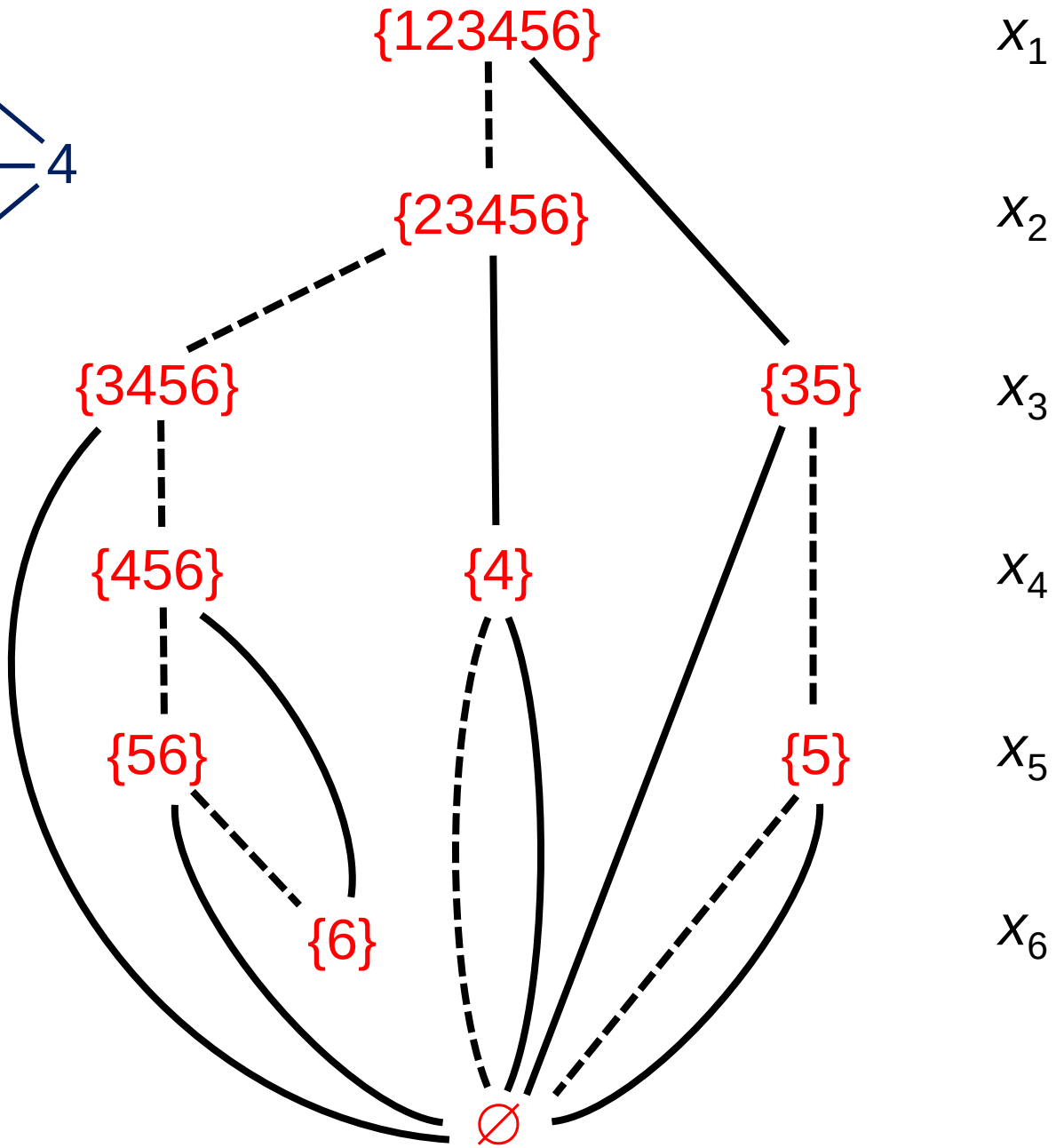
Merge nodes that correspond to the same state



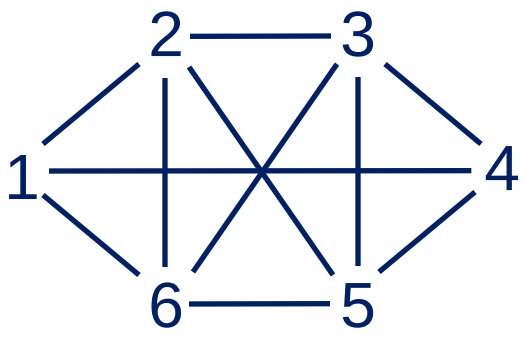


Merge nodes that correspond to the same state



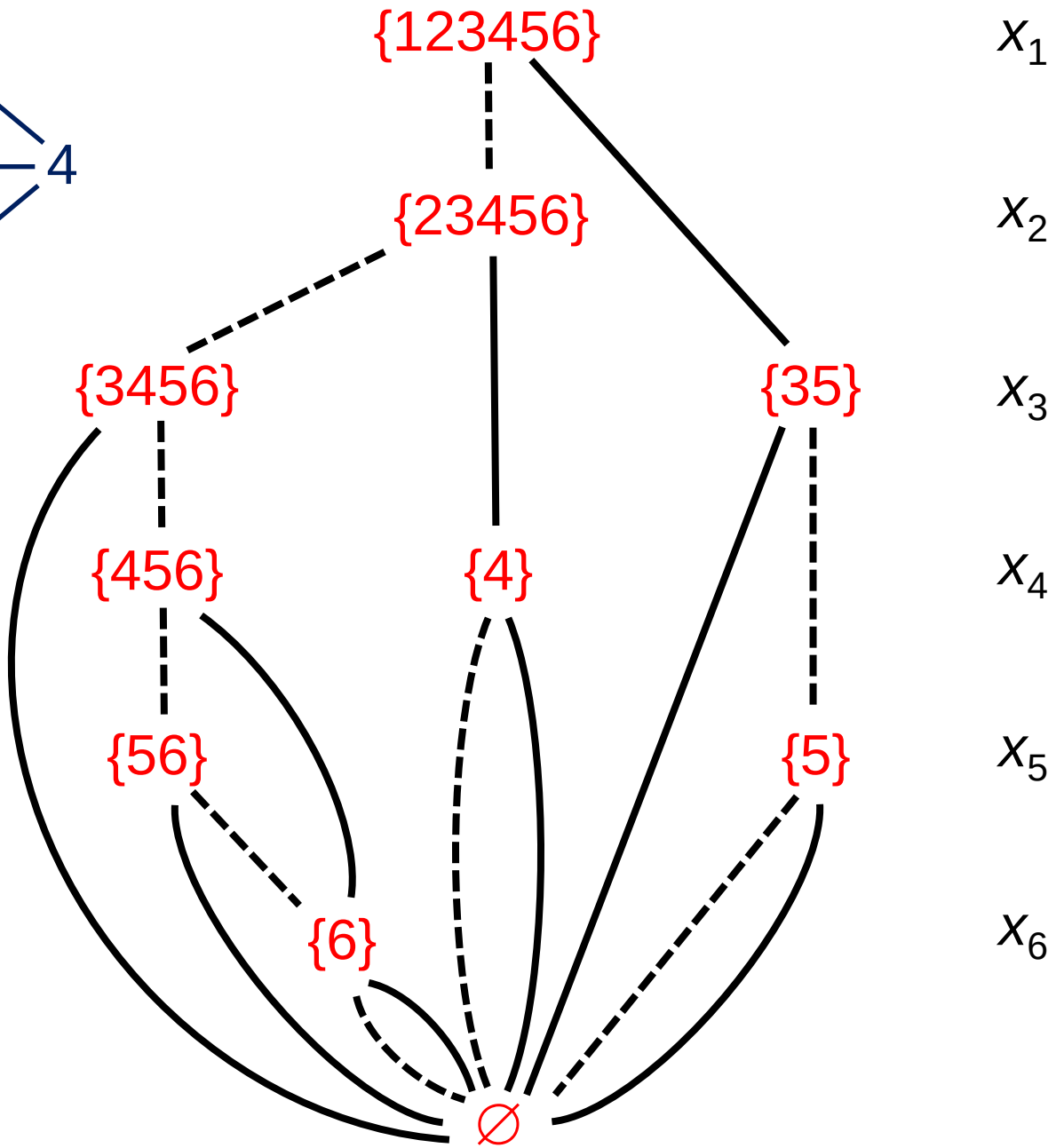


Merge nodes that correspond to the same state



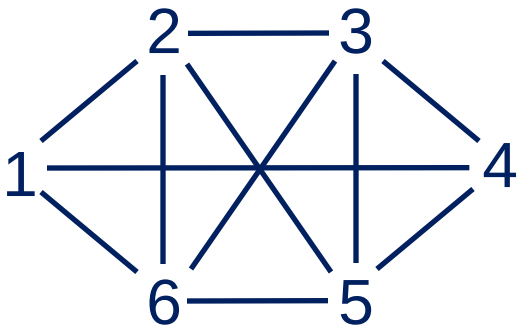
Width = 2

Merge nodes that correspond to the same state



Relaxation Bounding

- To obtain a bound on the objective function:
  - Use a **relaxed** BDD
  - Analogous to LP relaxation in IP
  - This relaxation is **discrete**.
  - Doesn't require the linear inequality formulation of IP.



{123456}

x_1

x_2

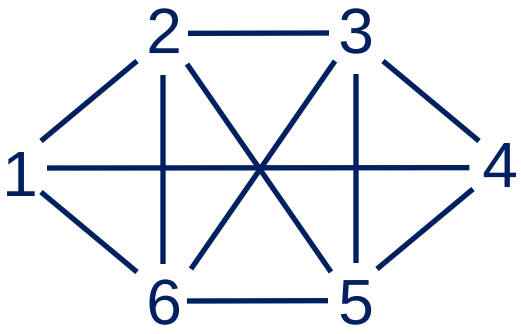
x_3

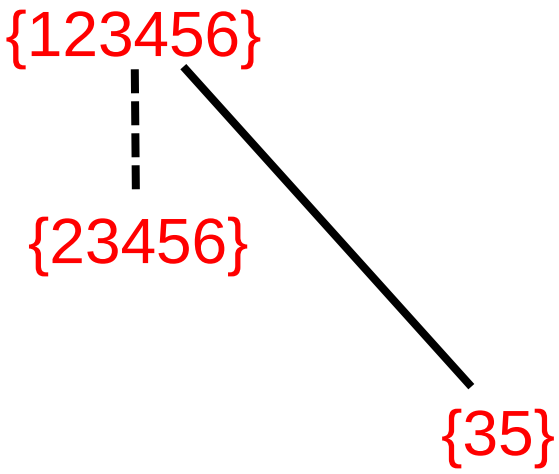
x_4

x_5

x_6

To build **relaxed** BDD, merge some additional nodes as we go along





x_1

x_2

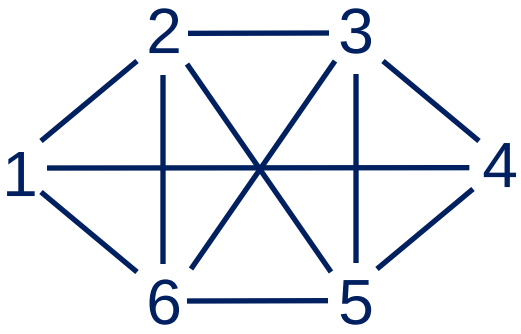
x_3

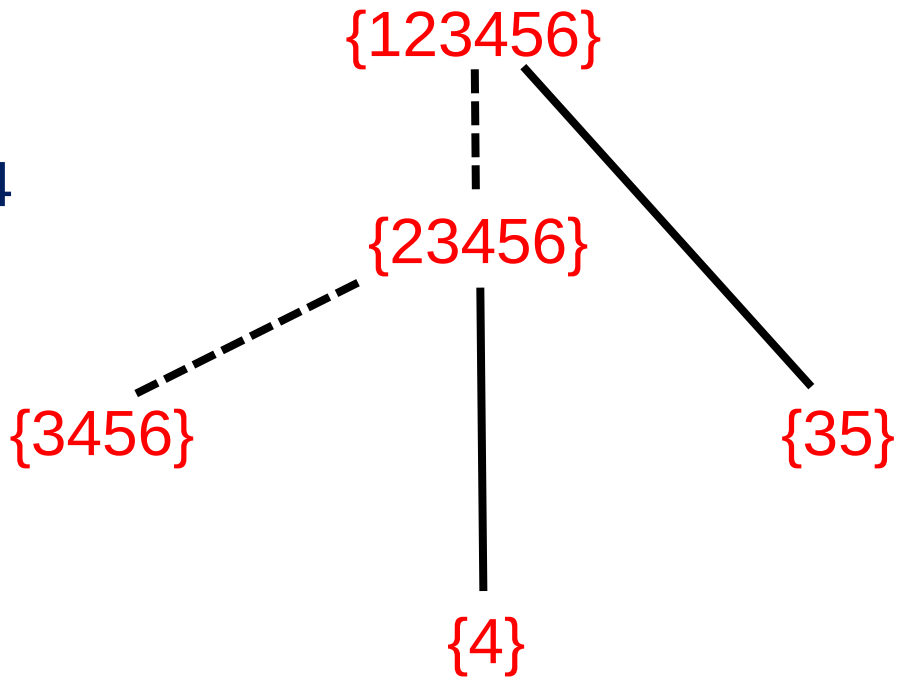
x_4

x_5

x_6

To build **relaxed** BDD, merge some additional nodes as we go along





x_1

x_2

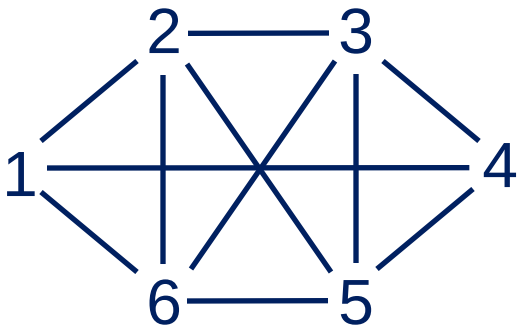
x_3

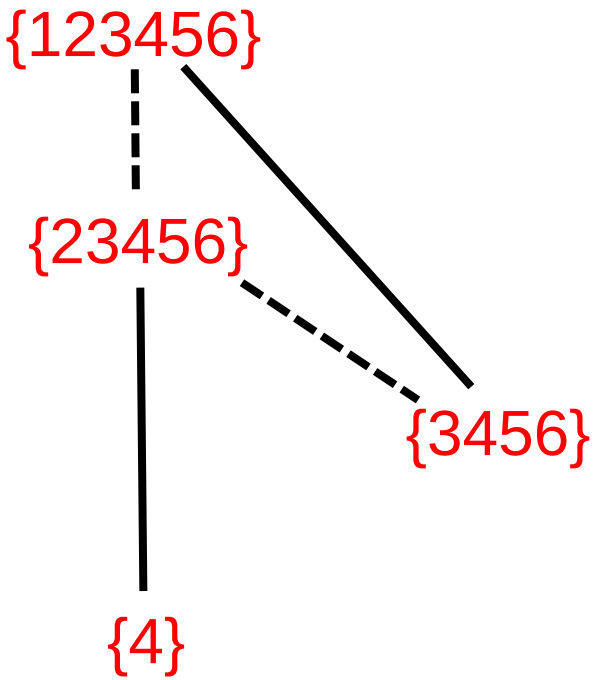
x_4

x_5

x_6

To build **relaxed** BDD, merge some additional nodes as we go along





x_1

x_2

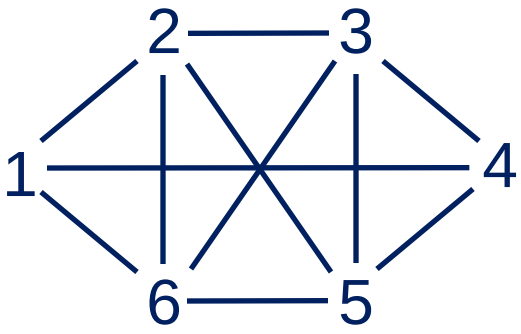
x_3

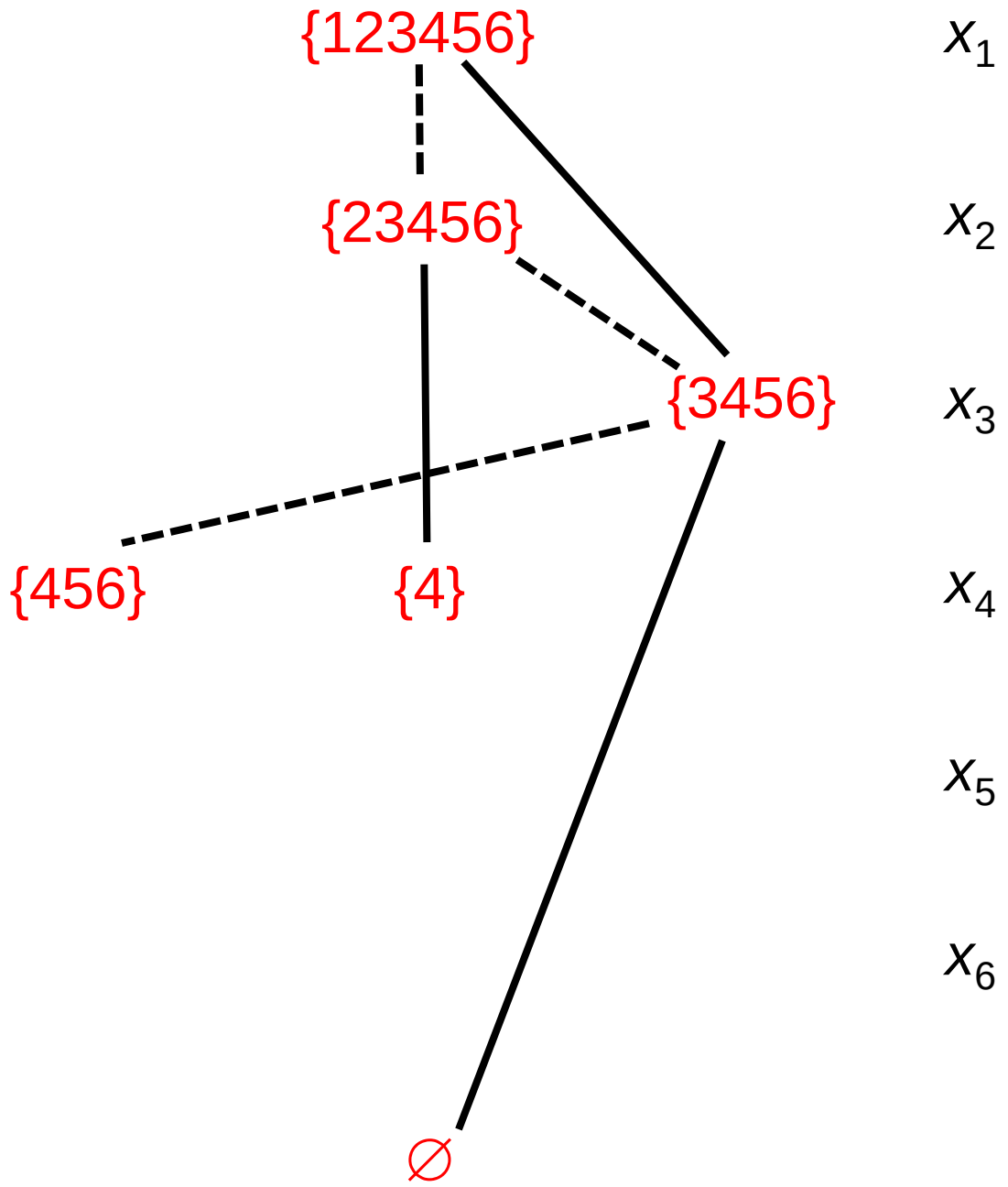
x_4

x_5

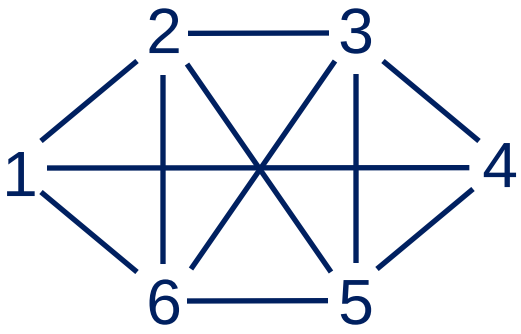
x_6

To build **relaxed** BDD, merge some additional nodes as we go along

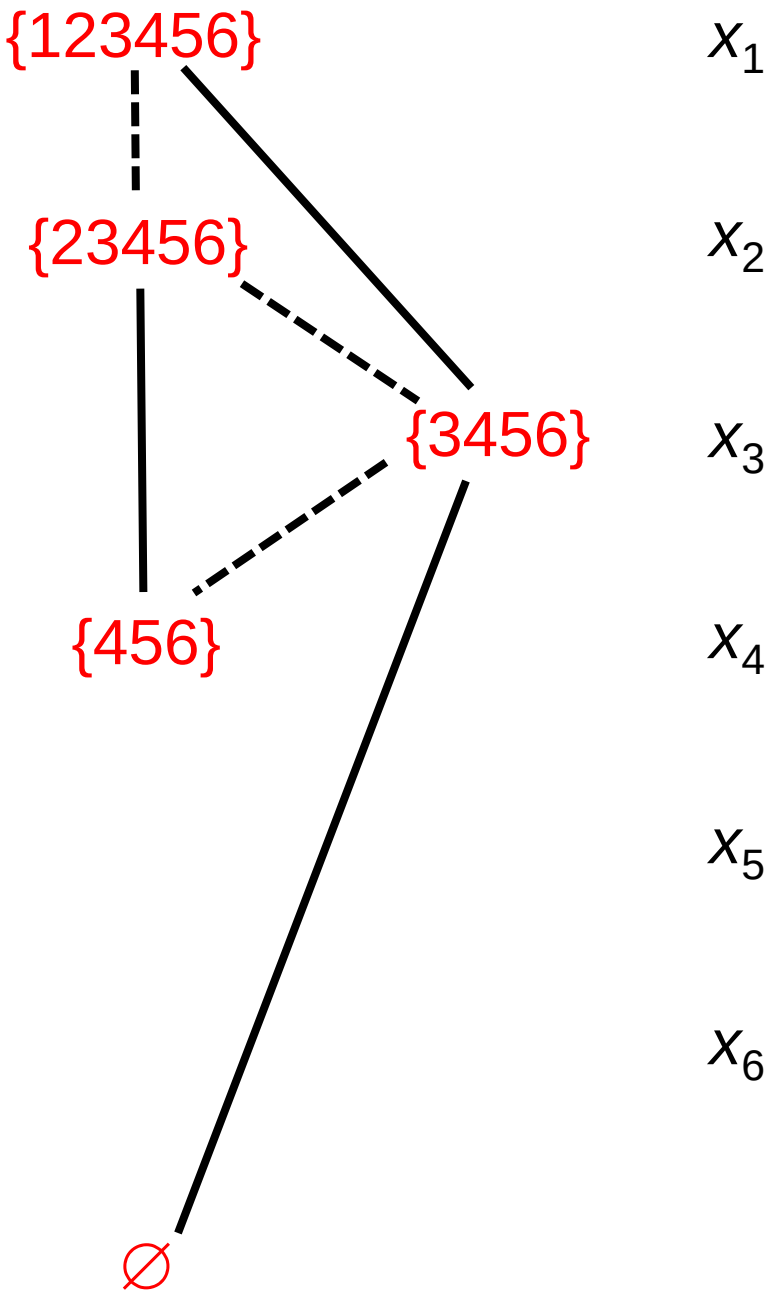


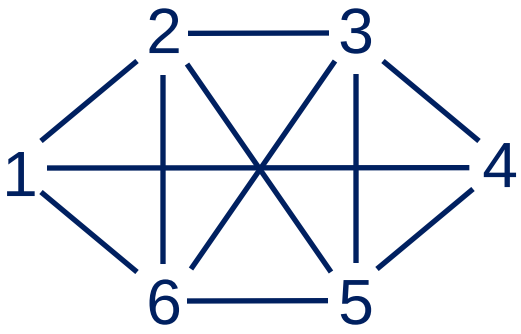


To build **relaxed** BDD, merge some additional nodes as we go along

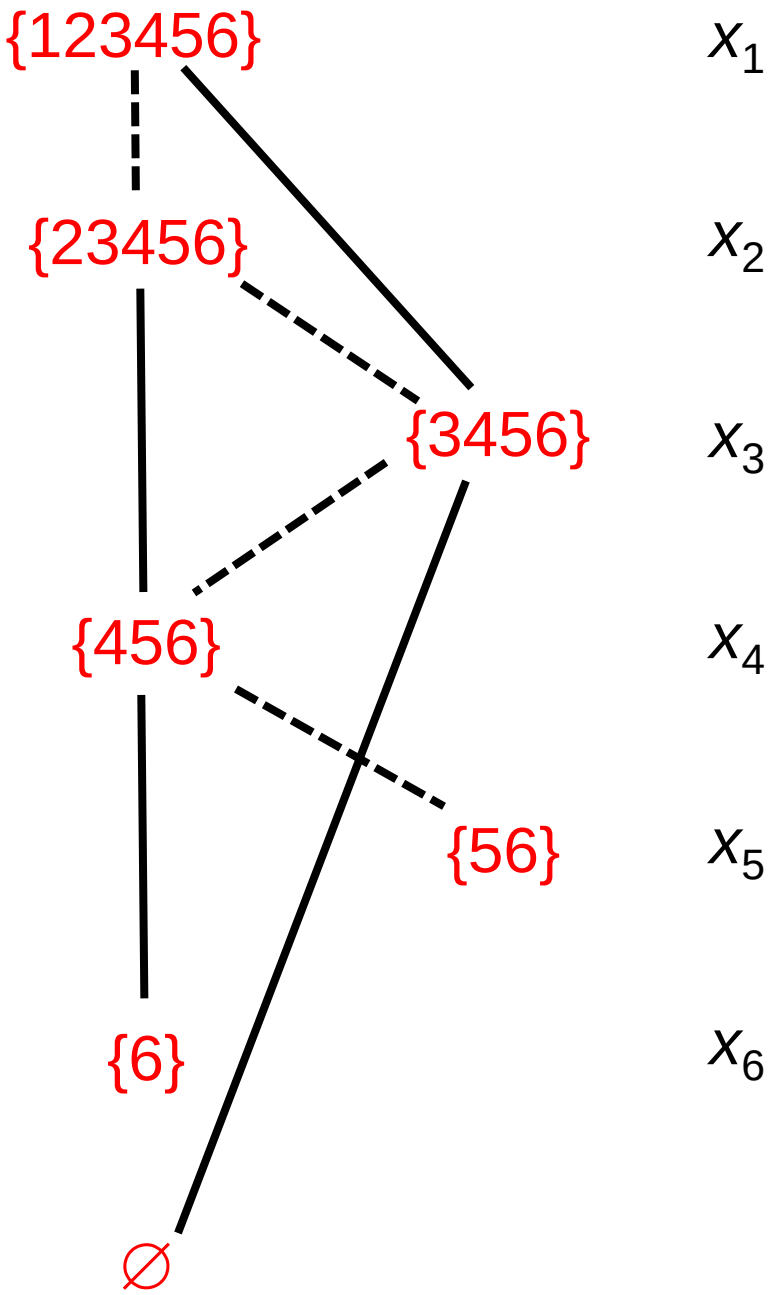


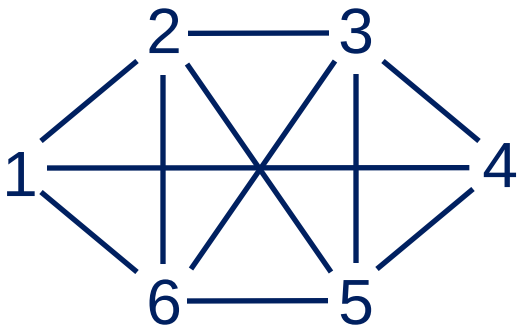
To build **relaxed** BDD, merge some additional nodes as we go along



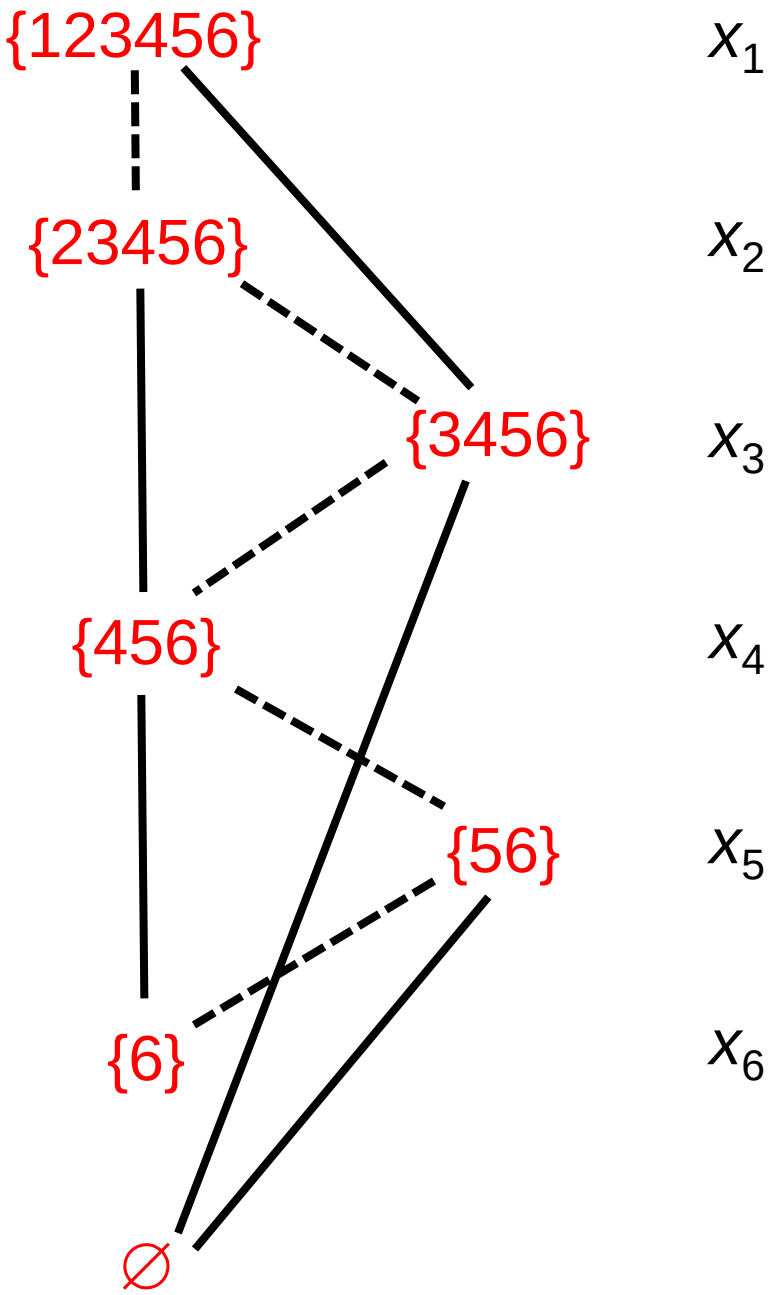


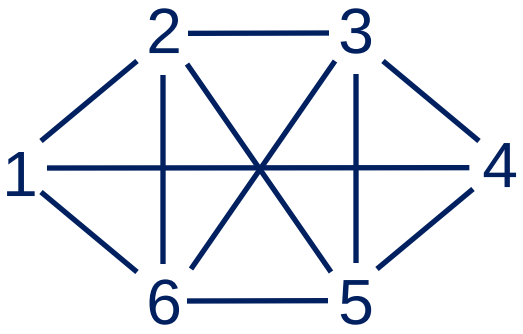
To build **relaxed** BDD, merge some additional nodes as we go along





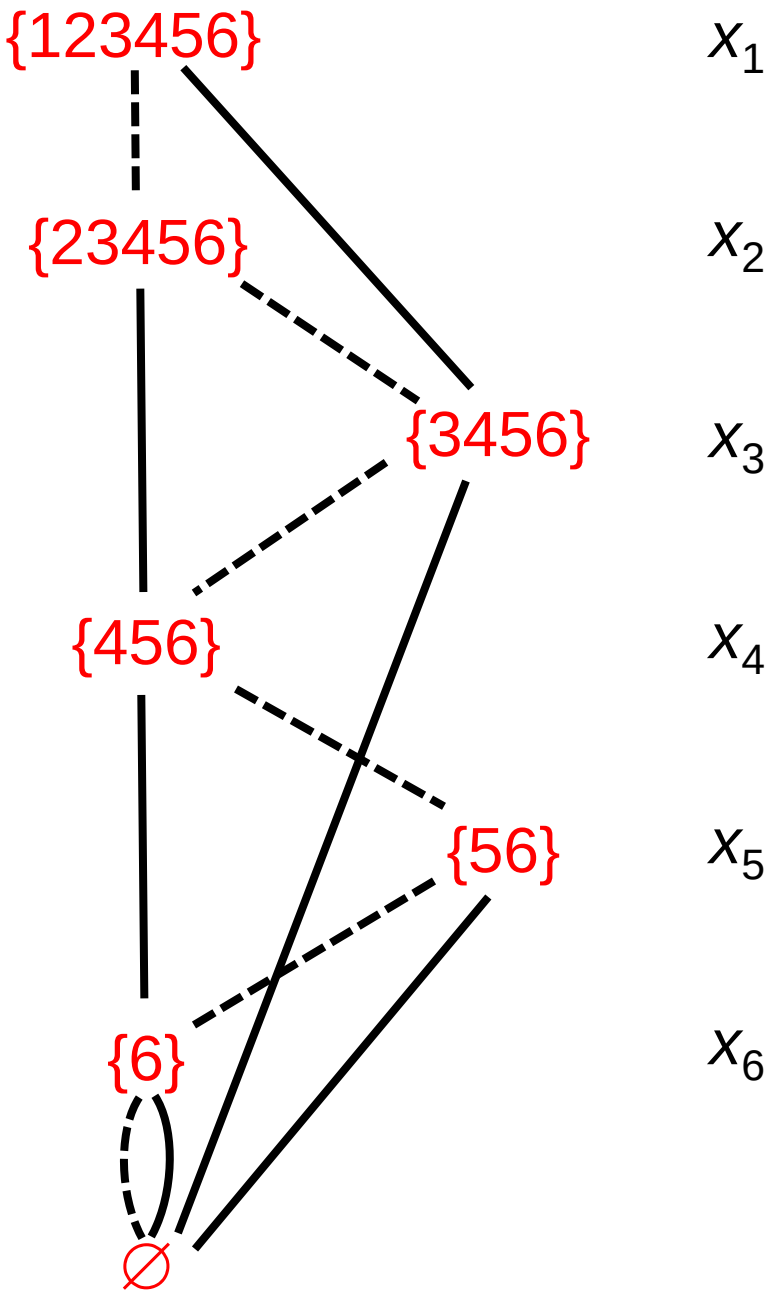
To build **relaxed** BDD, merge some additional nodes as we go along

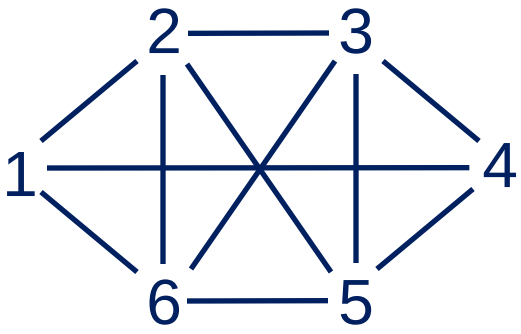




Width = 1

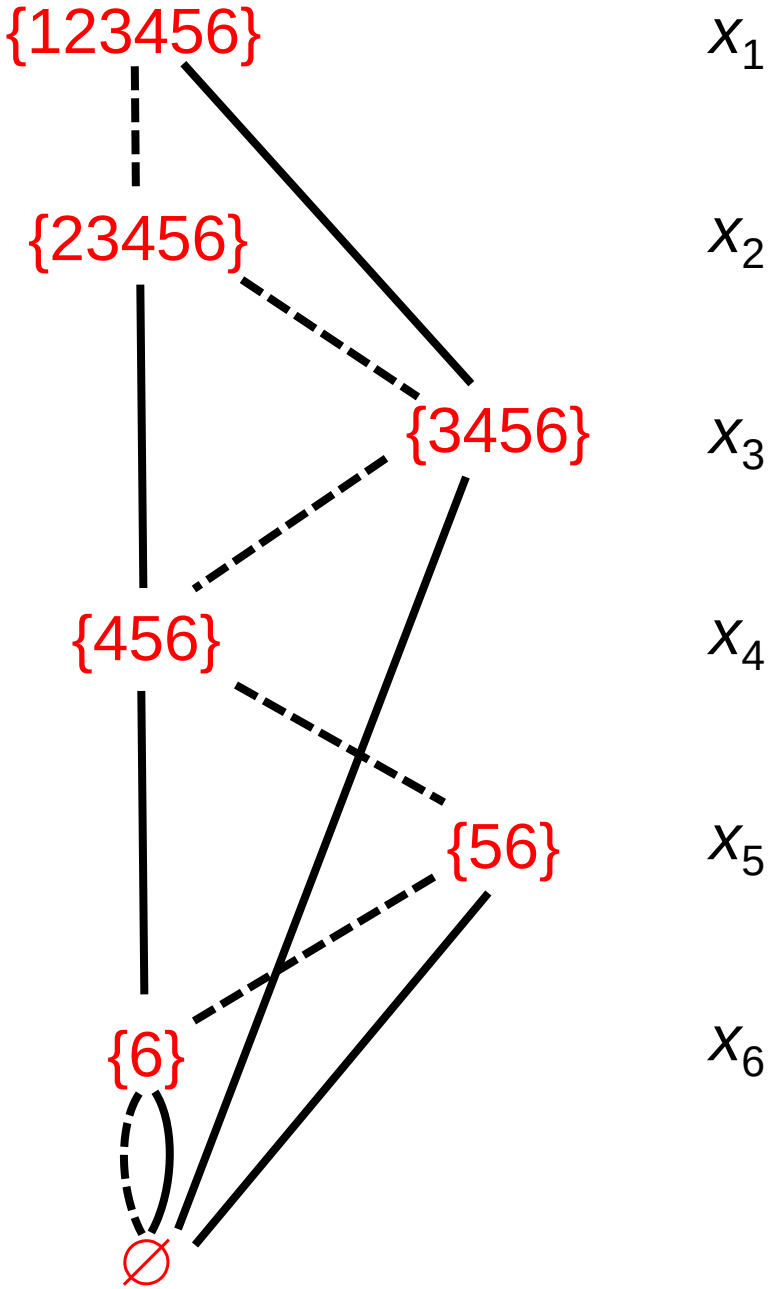
To build **relaxed** BDD, merge some additional nodes as we go along

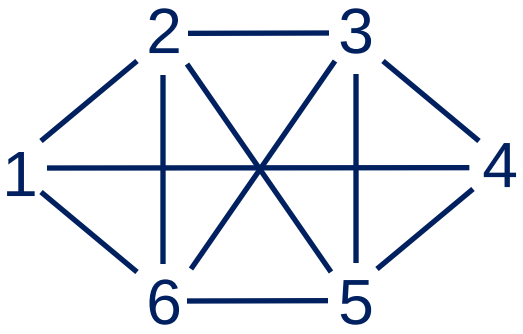




Width = 1

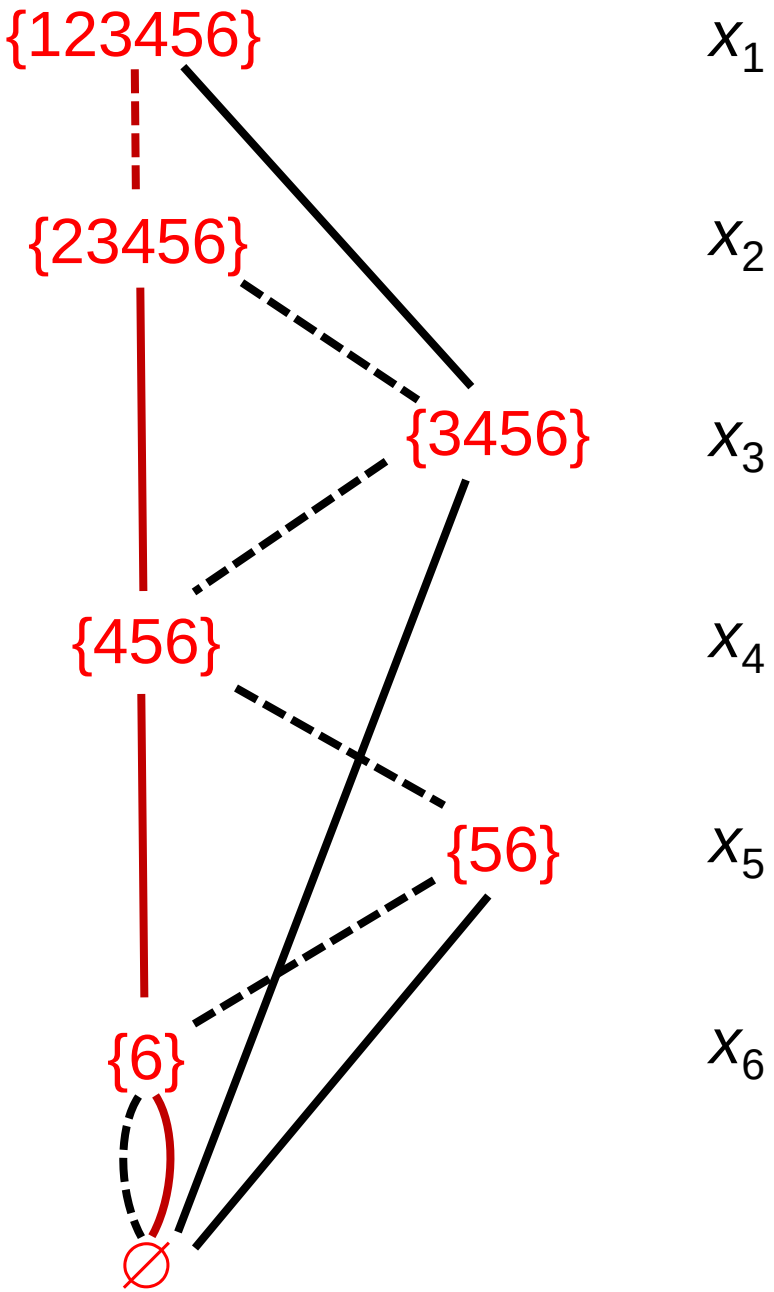
Represents 18 solutions, including 11 feasible solutions





Width = 1

Longest path
gives bound
of 3 on optimal
value of 2



Variable Ordering

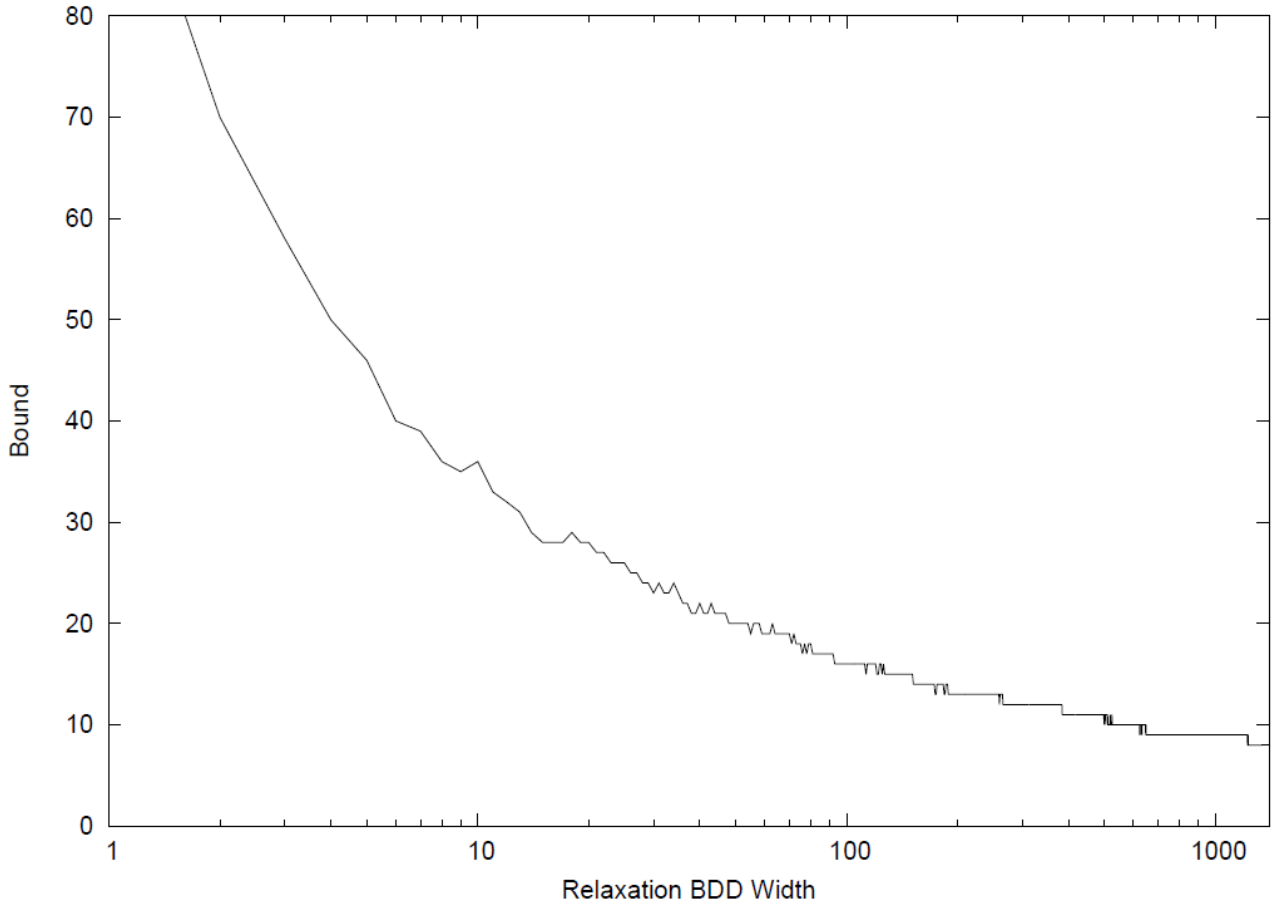
- Variable ordering is key.
  - Just as in branching methods.
  - Some orderings provide much tighter bounds.
- Dynamic ordering heuristic.
  - Next variable is the one appearing in the smallest number of states on the current level.

Node Merging Heuristic

- Merge nodes with worst node values.
  - Continue merging until desired width is achieved.
  - **Node value** is defined as part of model.
    - Related to quality of paths through the node.
    - For example, longest path length to node from top of BDD.

Width of Decision Diagram

- Wider BDDs yield tighter bounds.
  - But take longer to build.
  - Adjust width dynamically.



Incrementality

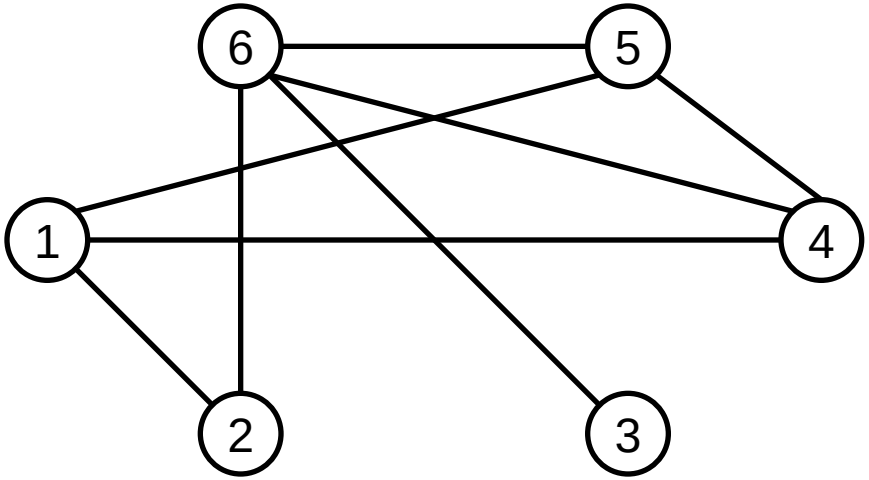
- Fast incremental calculation of bound.
  - Modify relaxed BDD after branching (fast).
  - Recompute longest path (fast).

Primal Heuristic

- Use **restricted** decision diagram**.**
  - All paths are feasible.
  - Longest path is a good feasible solution
  - Reduce width by removing nodes with worst longest-path values.

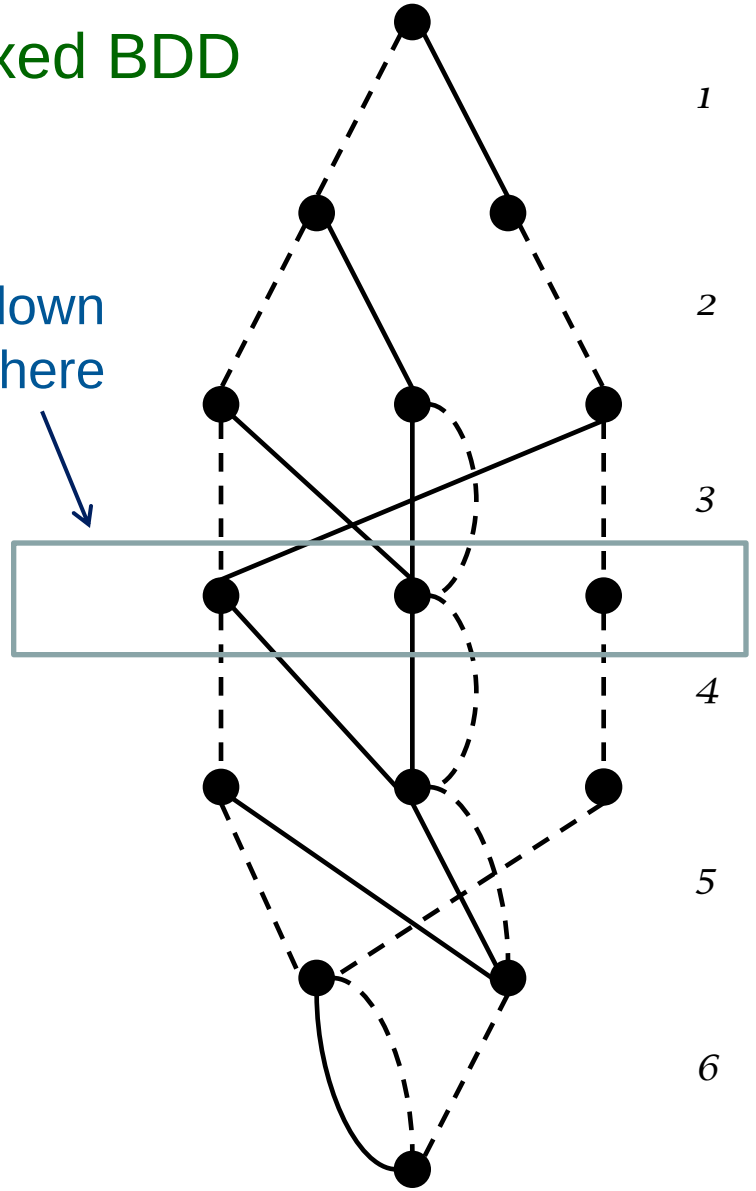
Branching

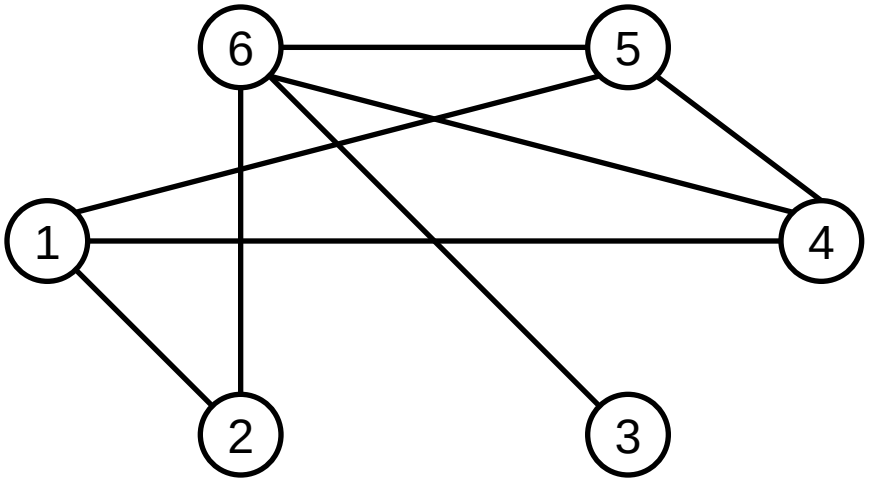
- Novel branching scheme
  - Branch in relaxed BDD.
  - Branch on pools of partial solutions.



Relaxed BDD

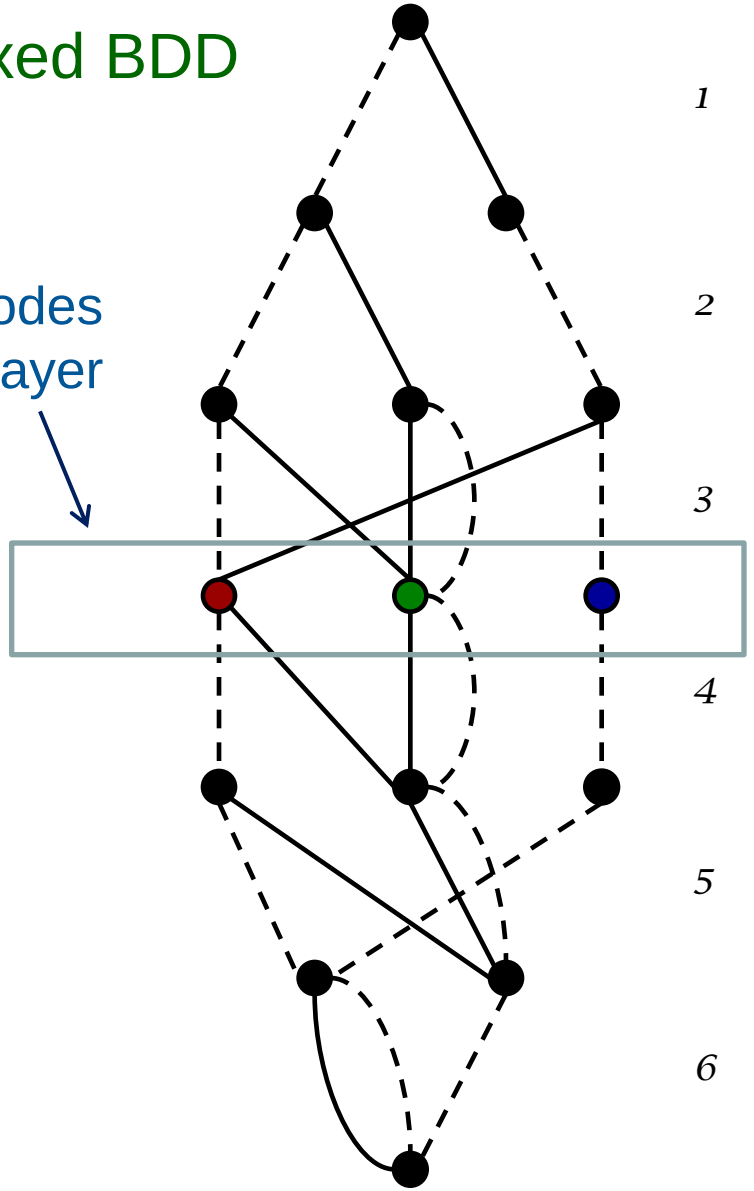
Exact down to here

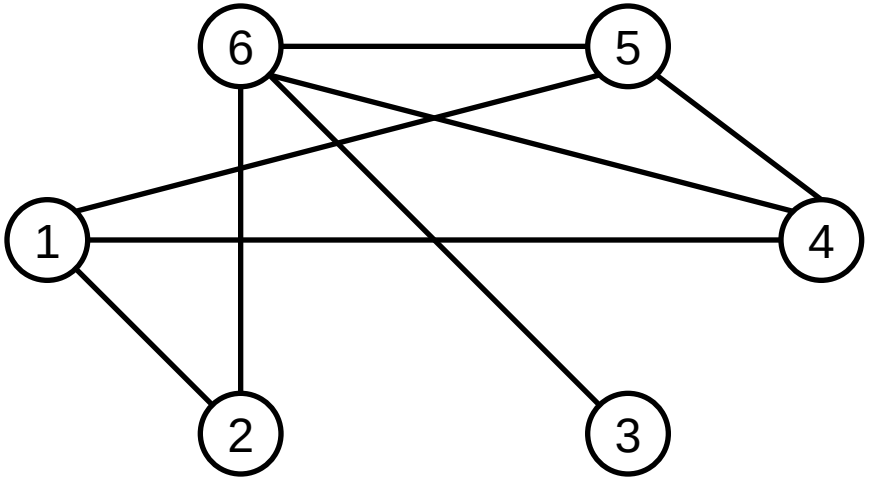




Relaxed BDD

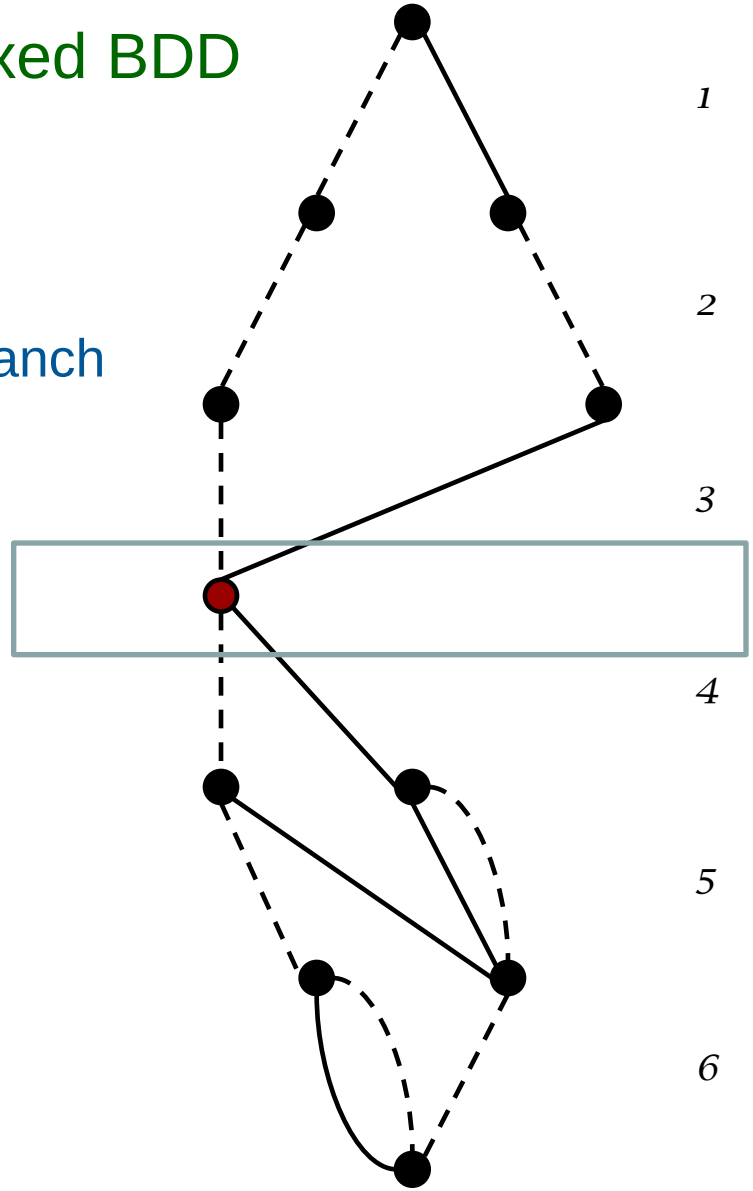
Branch on nodes
in this layer



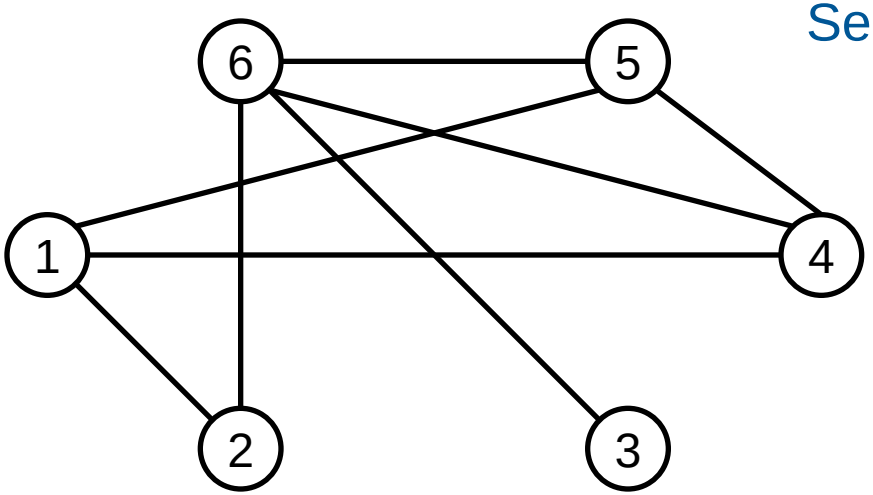


Relaxed BDD

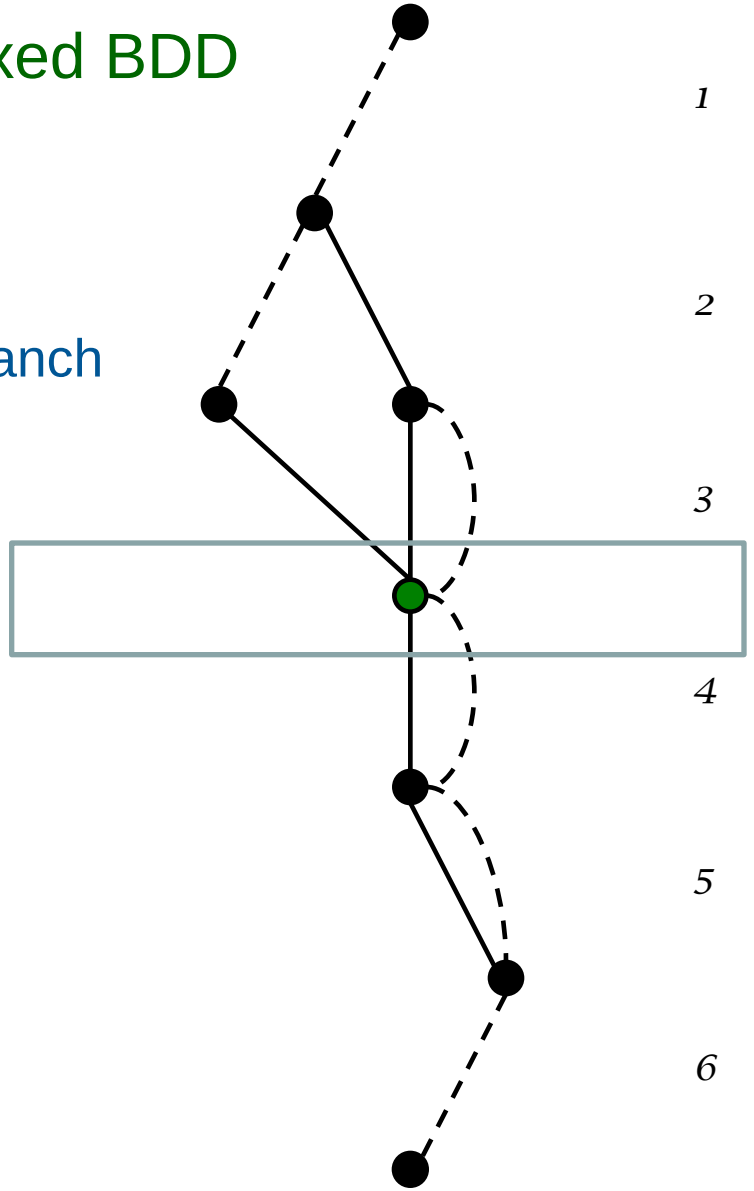
First branch



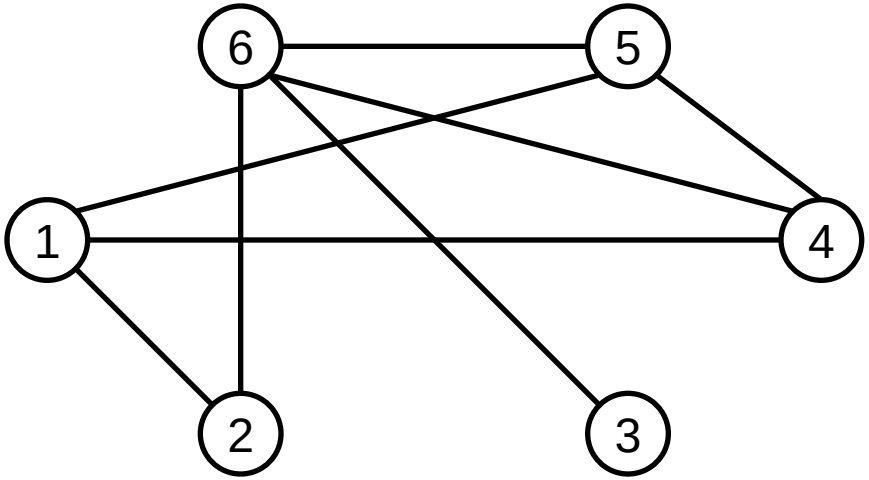
Relaxed BDD



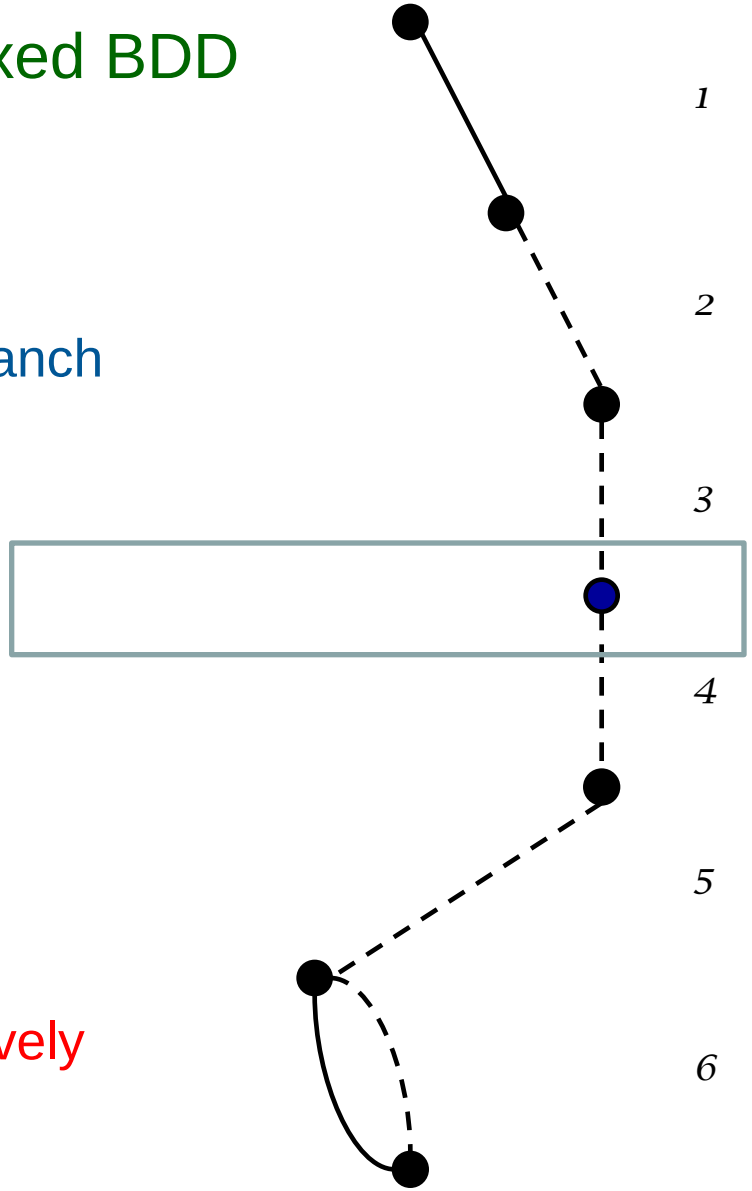
Second branch



Relaxed BDD



Third branch



Continue recursively

Put it All Together

- **General-purpose** discrete solver
  - **Bounds** from relaxed BDD
  - **Primal heuristic** from restricted BDD
  - **Formulation** with state variables, node merger rule
  - **Branching** in relaxed BDD

Put it All Together

- **General-purpose** discrete solver
  - **Bounds** from relaxed BDD
  - **Primal heuristic** from restricted BDD
  - **Formulation** with state variables, node merger rule
  - **Branching** in relaxed BDD
- Compare with CPLEX
  - Select problems with **good IP model**.
    - Stable set problem
    - Max cut problem
    - Max 2-SAT

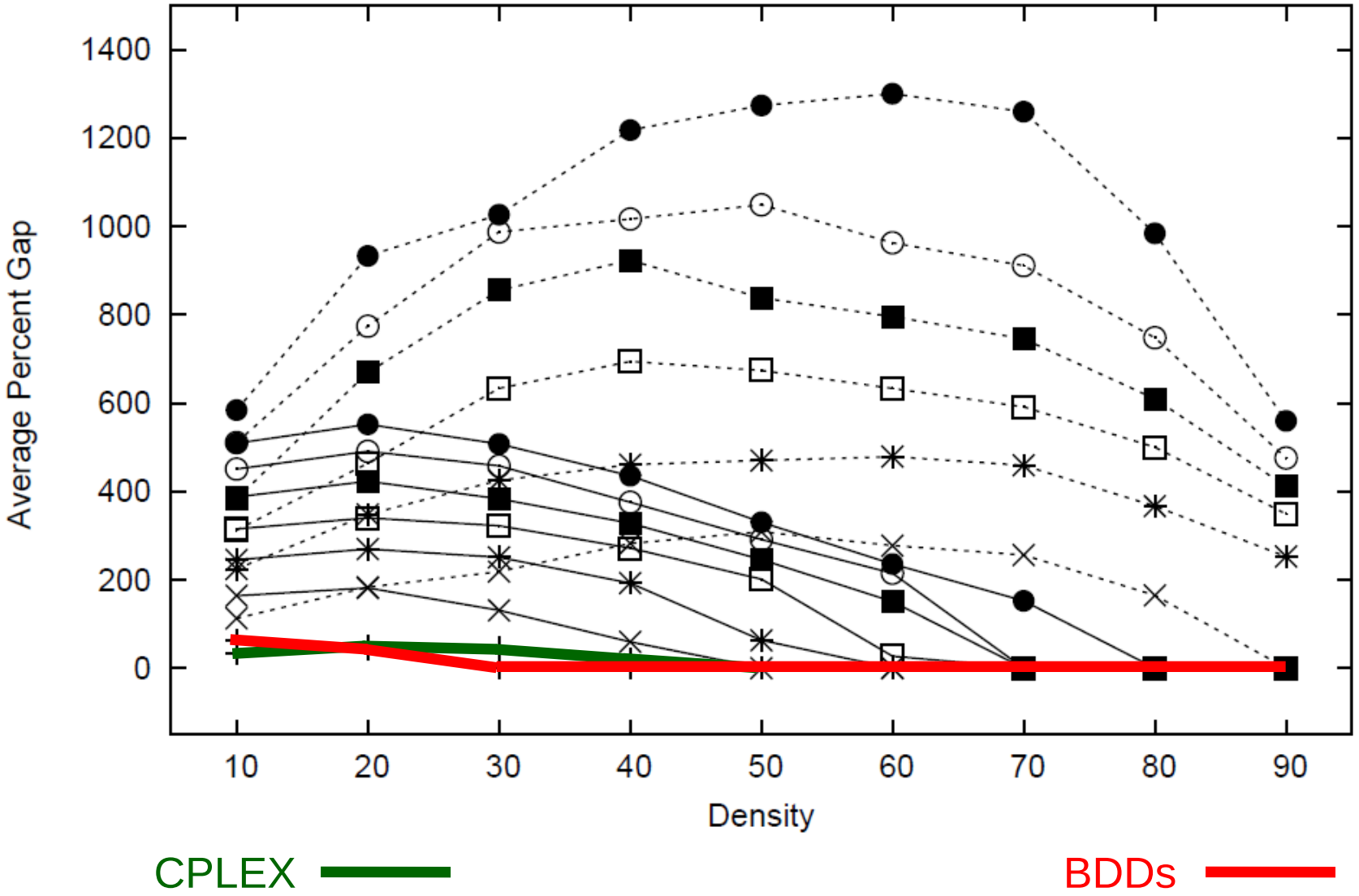
Stable Set Problem

- BDD Model:
  - **State** = Set of available vertices to add to stable set.
  - **Arc cost** = weight of added vertex
  - **Node value** = longest path length to node
  - **Node merger** = union of states
- MIP model:
  - Clique cover model.
  - Clique cover generated by greedy heuristic based on vertex degree.

Random stable set instances, 10 per data point

Gap after 30 minutes

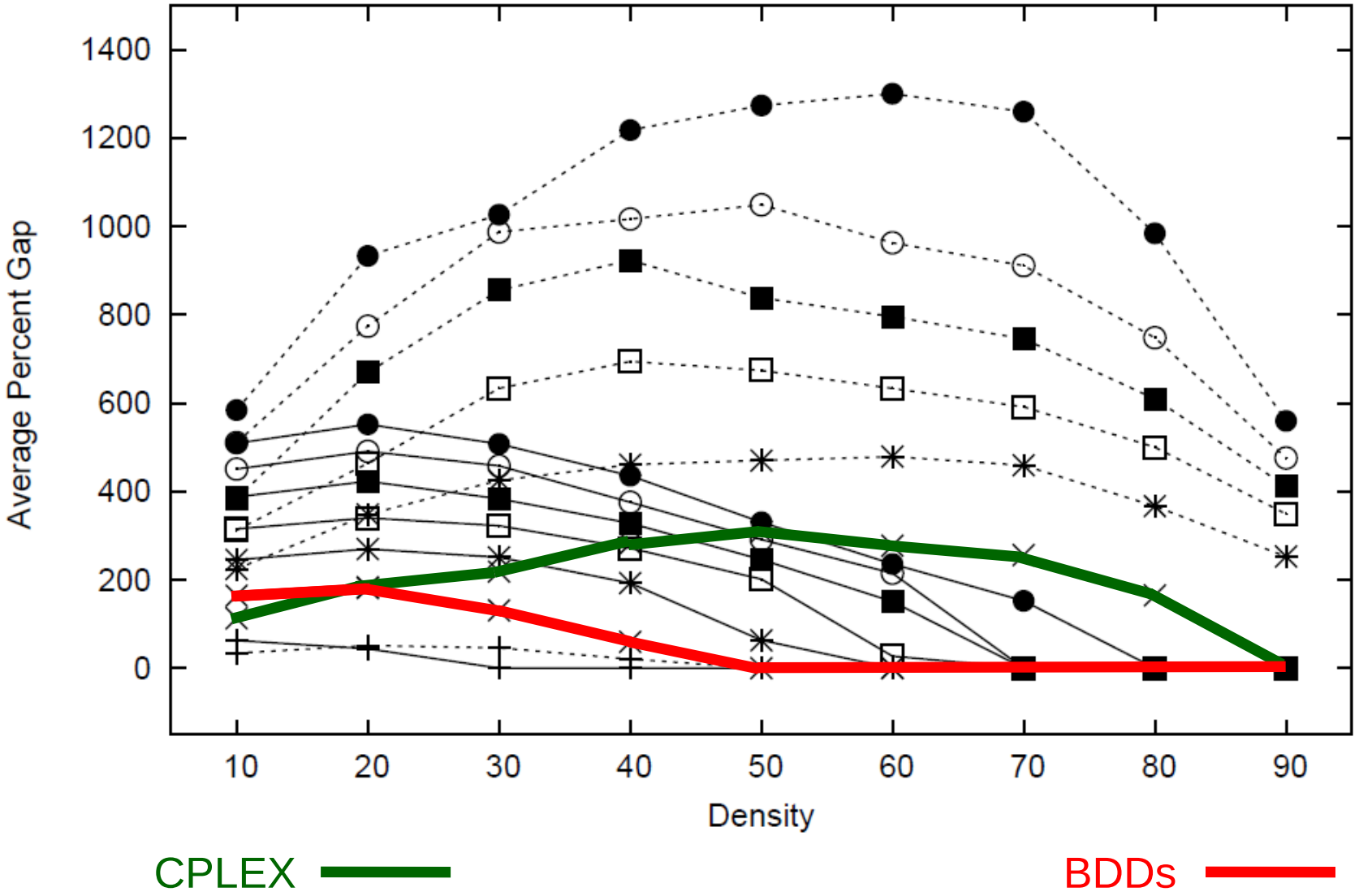
250 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

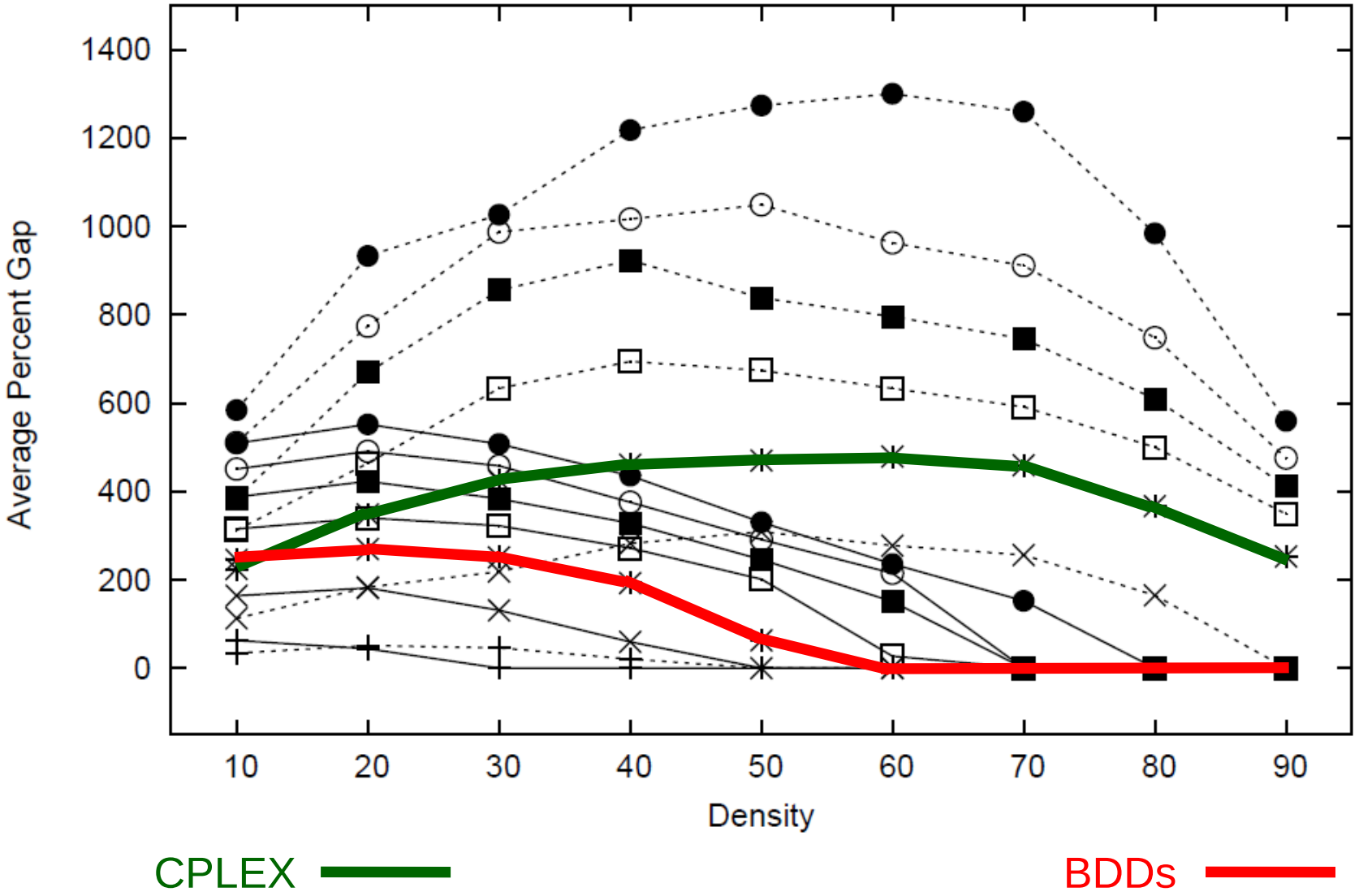
500 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

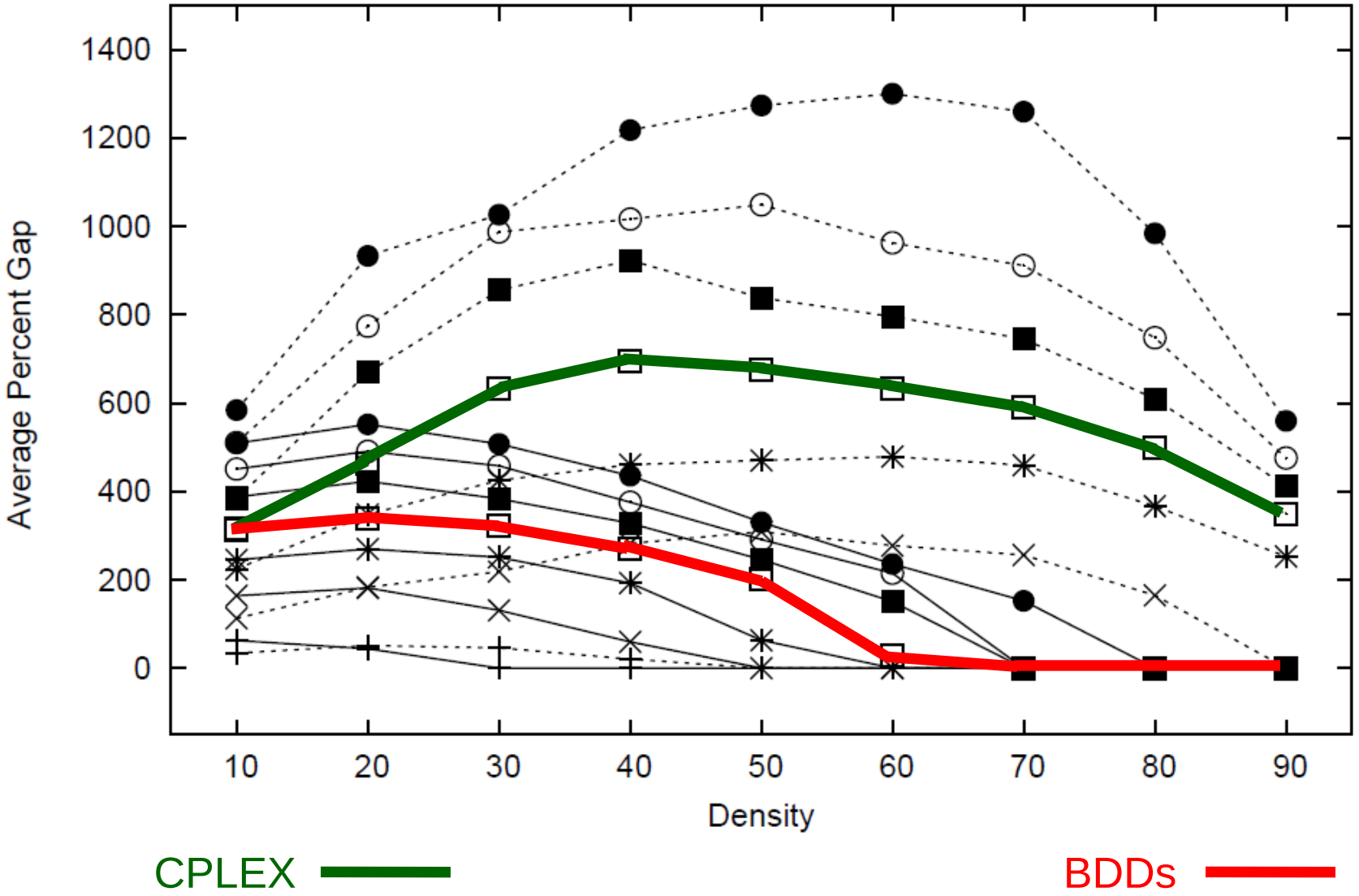
750 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

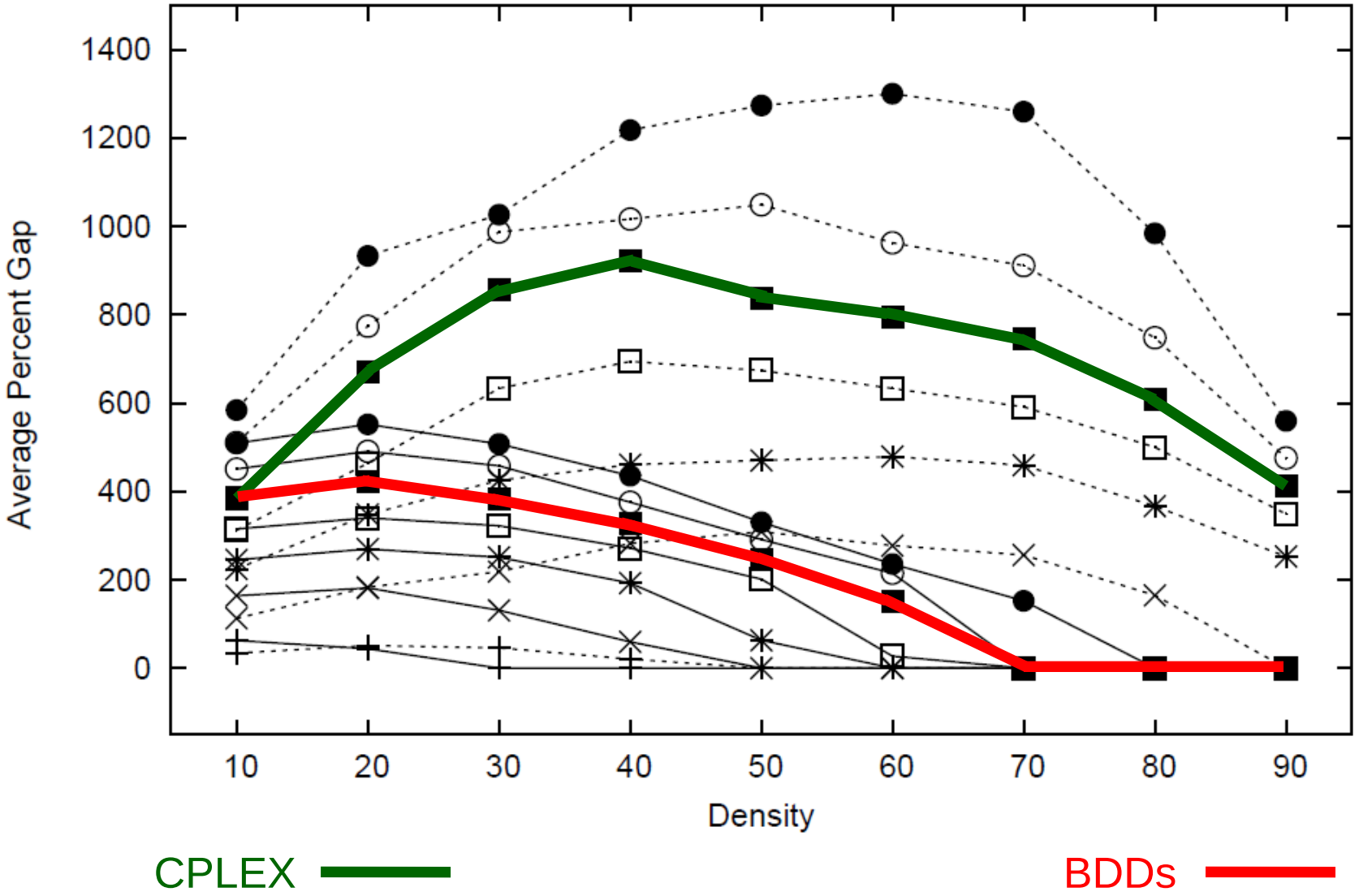
1000 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

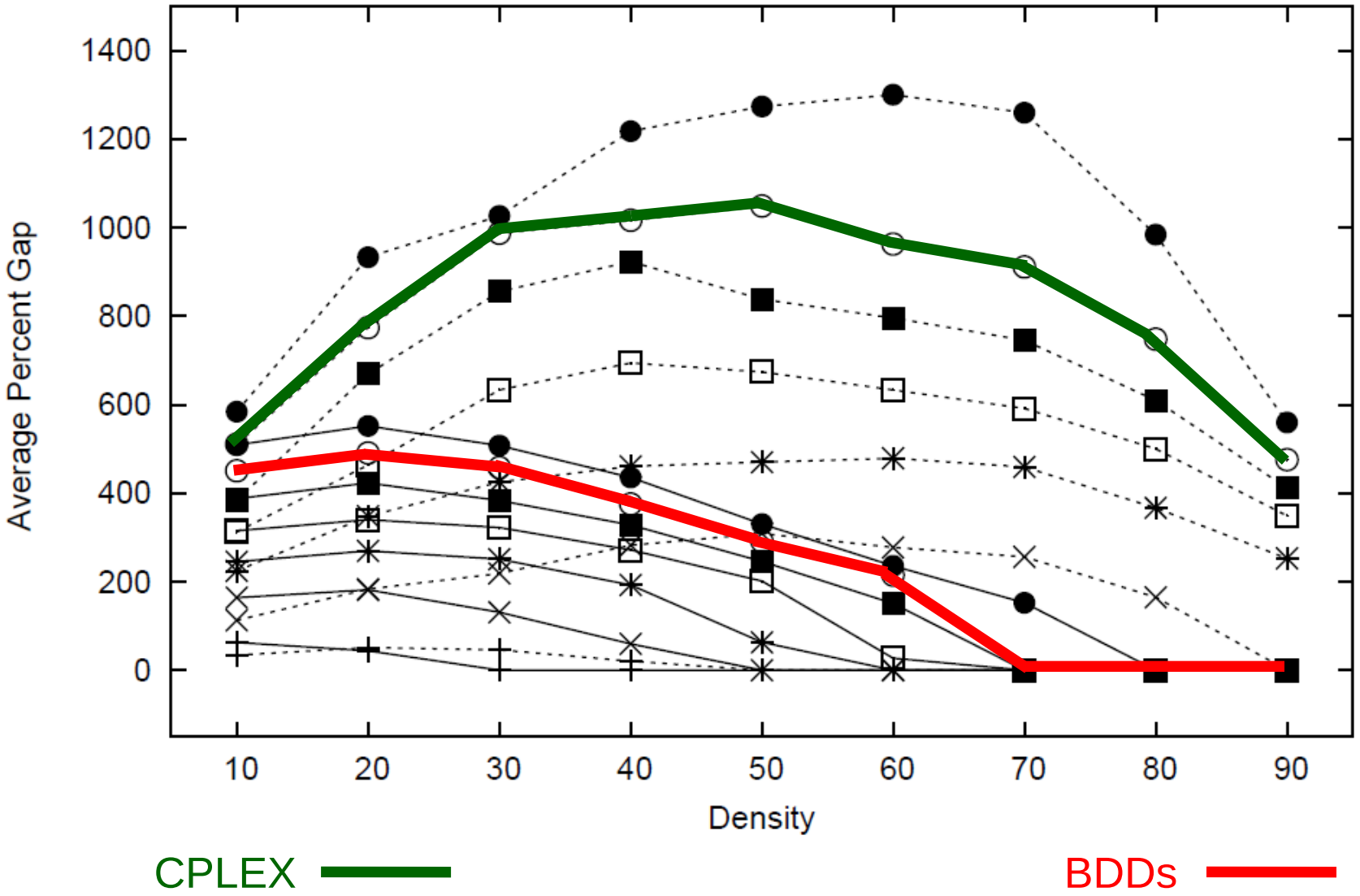
1250 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

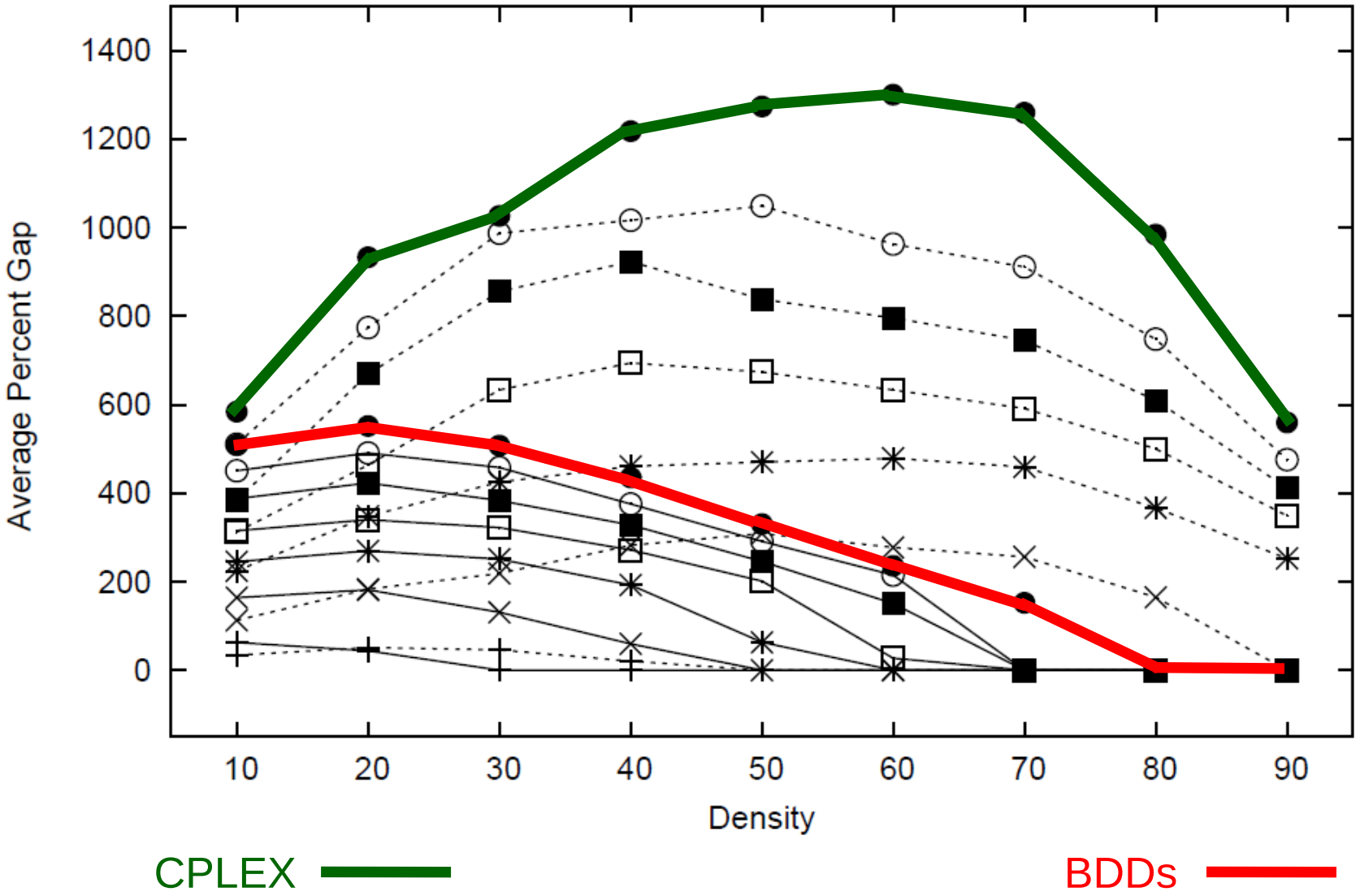
1500 vertices



Random stable set instances, 10 per data point

Gap after 30 minutes

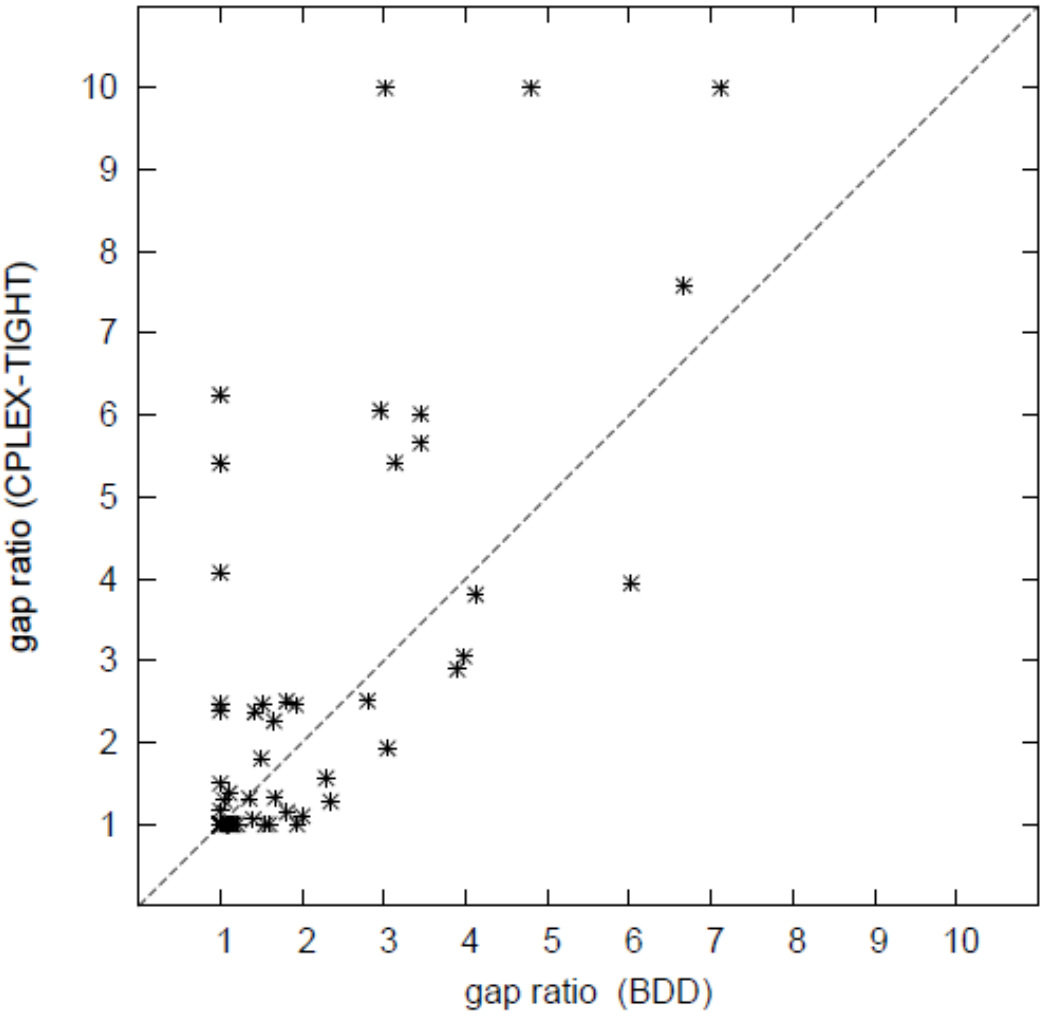
1750 vertices



DIMACS instances

Graph complement of 87 max clique instances

Relative gap after 30 minutes



Max Cut Problem

- Problem:
  - In a graph, find a cut that maximizes weight of edges crossing the cut.
- BDD Model:
  - Choose a DP model in which state reflects the solution quality.
  - **State:** $\boldsymbol{s} = (s_1, \ldots, s_n)$, where s_i is net marginal benefit of placing vertex i on the left side of the cut.
  - **Arc cost** = (roughly) net change in $\sum_i |s_i|$.
  - **Node value** = longest path length to node + $\sum_i |s_i|$.
  - **Node merger:** Merge states $\boldsymbol{s}'$ and $\boldsymbol{s}''$ to obtain $\boldsymbol{s}$, where

$$s_i = \begin{Bmatrix} \min\{s_i', s_i''\} \text{ if } s_i', s_i'' \geq 0 \\ \max\{s_i', s_i''\} \text{ if } s_i', s_i'' < 0 \\ 0 \text{ if } s_i' \cdot s_i'' < 0 \end{Bmatrix}$$

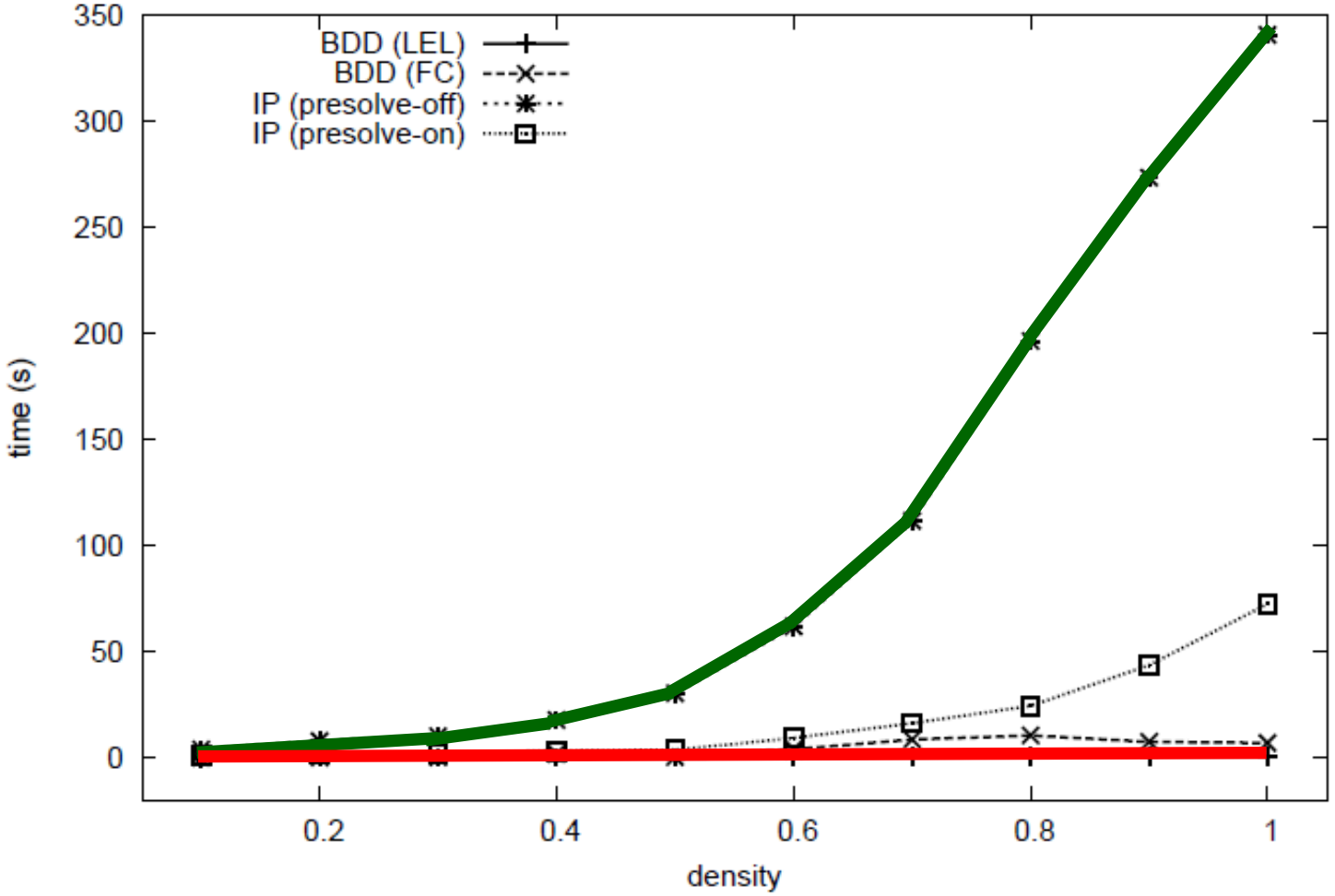
MIP model:

- Standard.

Random max cut instances

30 vertices, 10 instances per data point

Average solve time



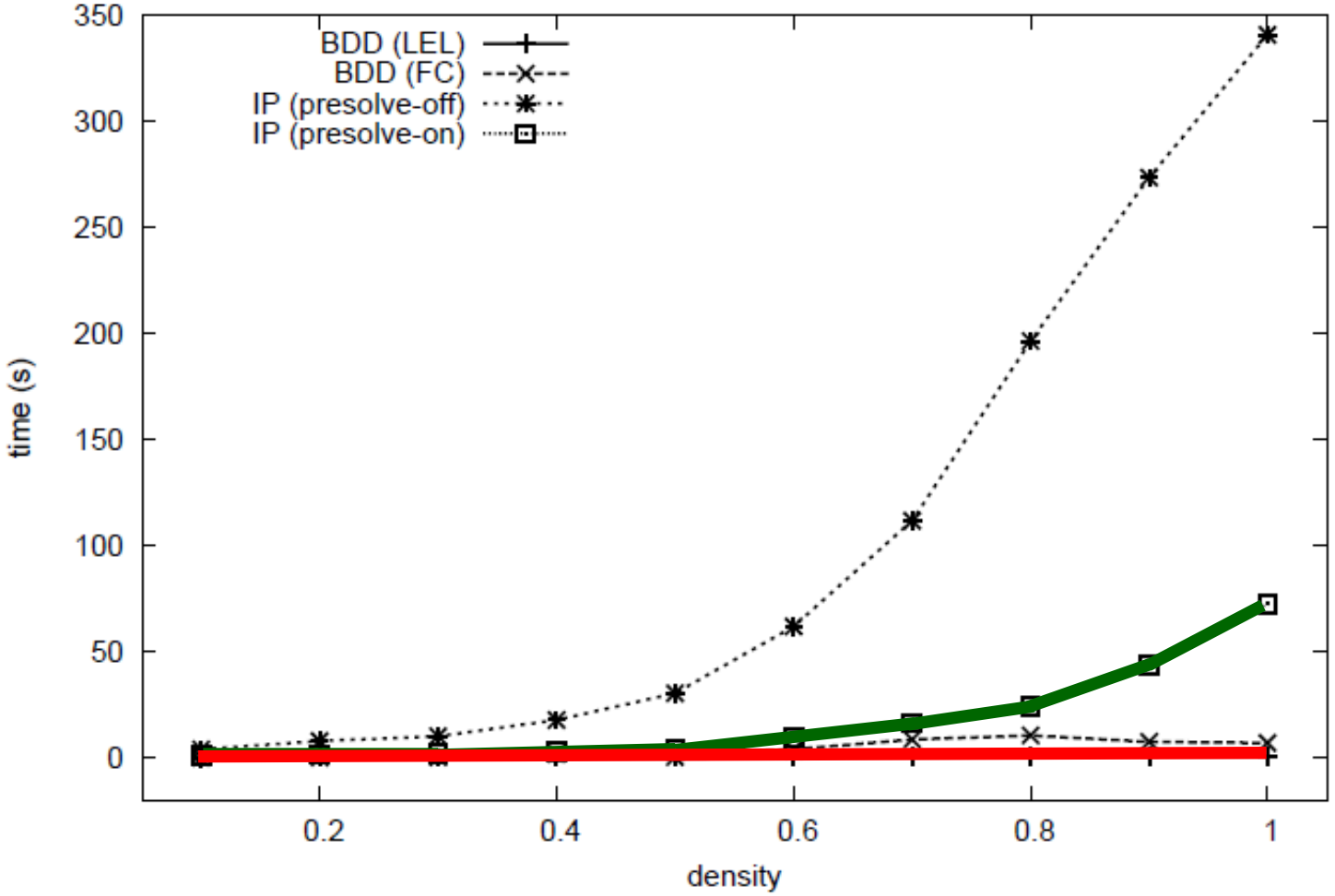
CPLEX without presolve

BDDs

Random max cut instances

30 vertices, 10 instances per data point

Average solve time



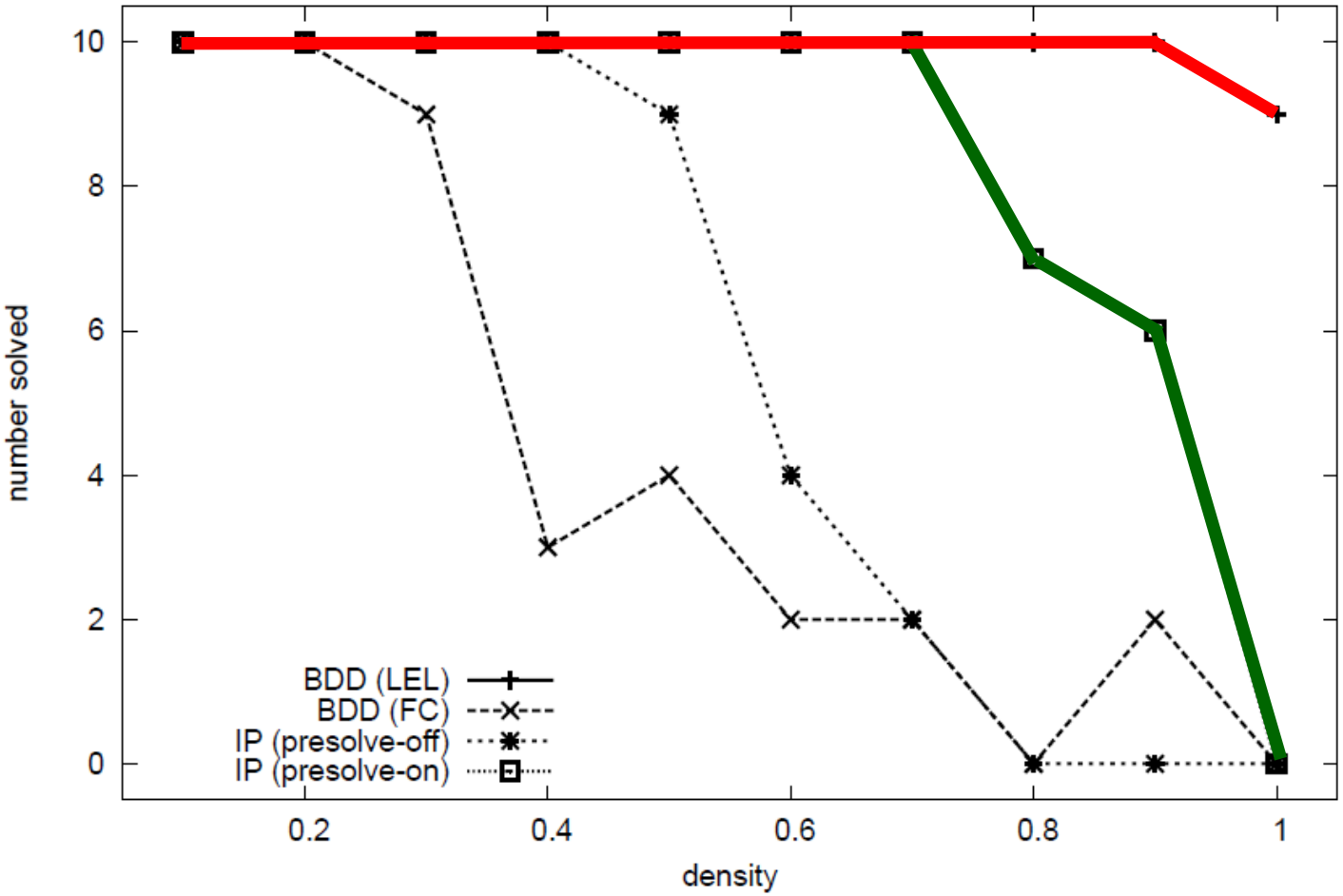
CPLEX with presolve

BDDs

Random max cut instances

40 vertices, 10 instances

Number solved after 30 minutes



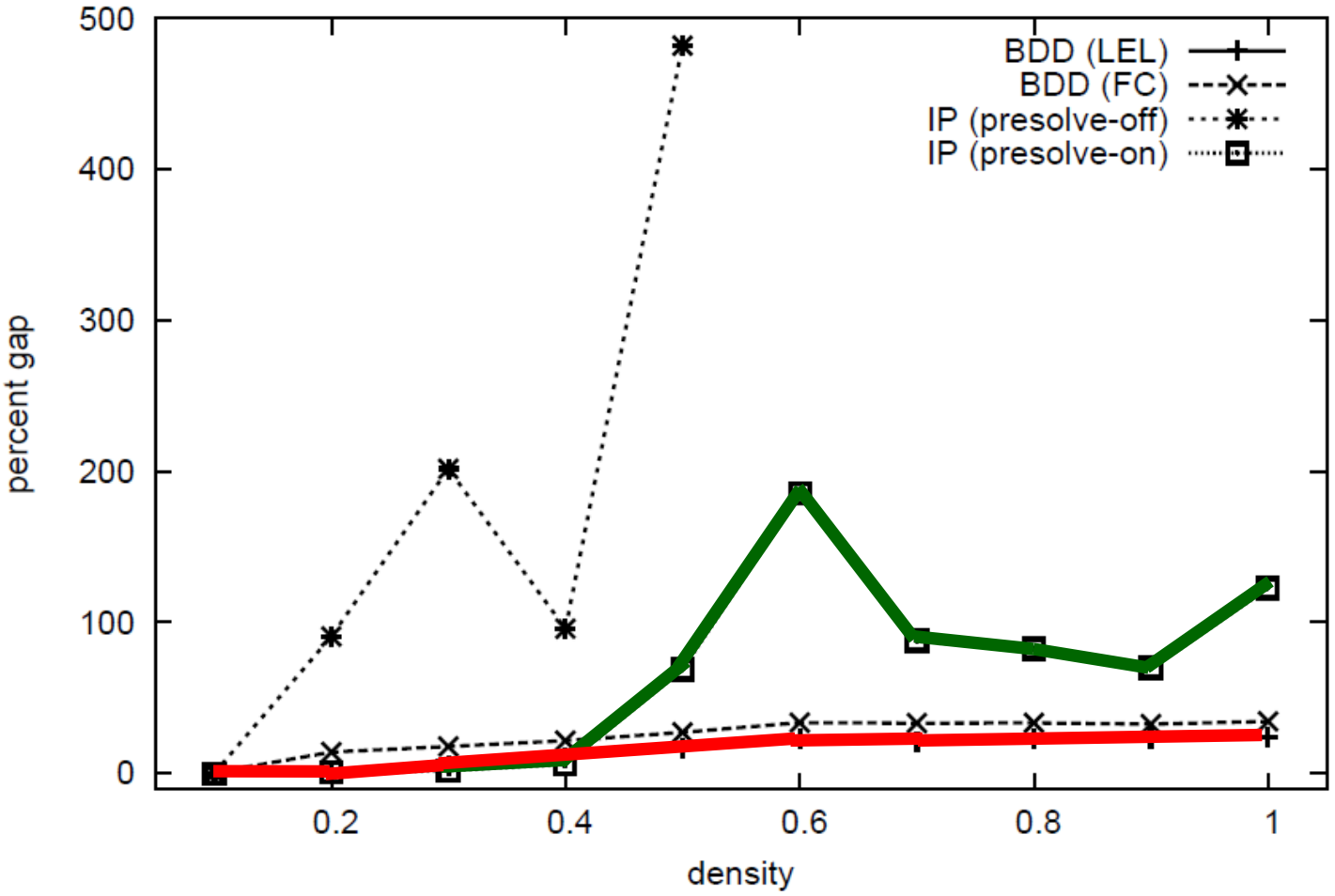
CPLEX with presolve

BDDs

Random max cut instances

50 vertices, 10 instances per data point

Average percent gap after 30 minutes



CPLEX with presolve

BDDs

Benchmark max cut instances (g-set)

Better than **specialized algorithms** for sparse instances.
Worse for dense instances.

Obtained **best known solution** for 6 instances:

Instance	%ReductionInGap
g50	**82.44**
g32	**8.20**
g33	**3.39**
g34	**2.65**
g11	**95.24**
g12	**7.69**

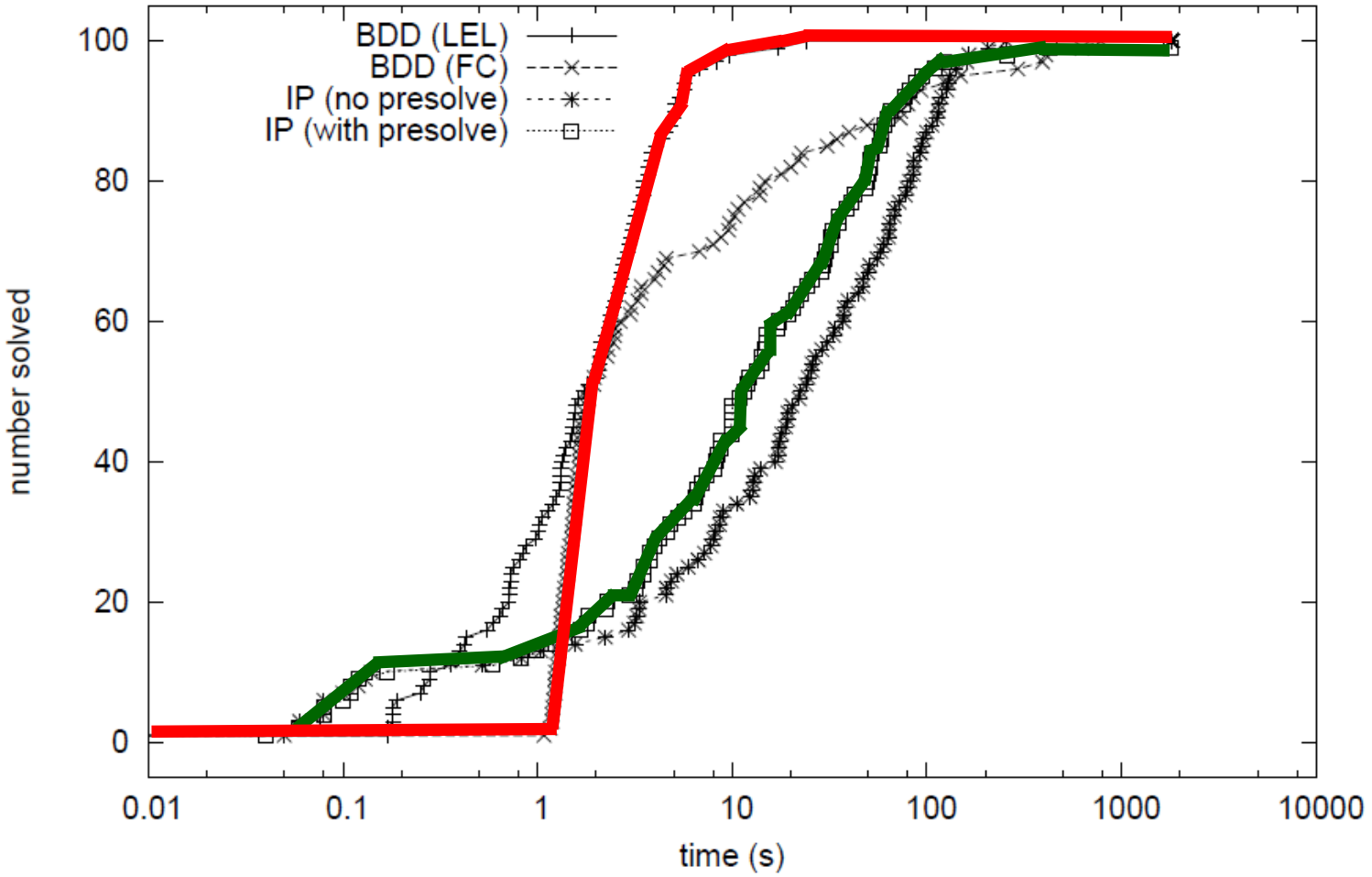
Max 2-SAT

- Problem:
  - Find truth assignment for a 2-SAT problem that maximizes total weight of satisfied clauses.
- BDD Model:
  - Very similar to max cut model.
- MIP model:
  - Standard model used in MaxSAT literature.
  - IP is often competitive with best special-purpose algorithms.

Random max 2-SAT instances

30 variables, 100 instances

Number of instances solved



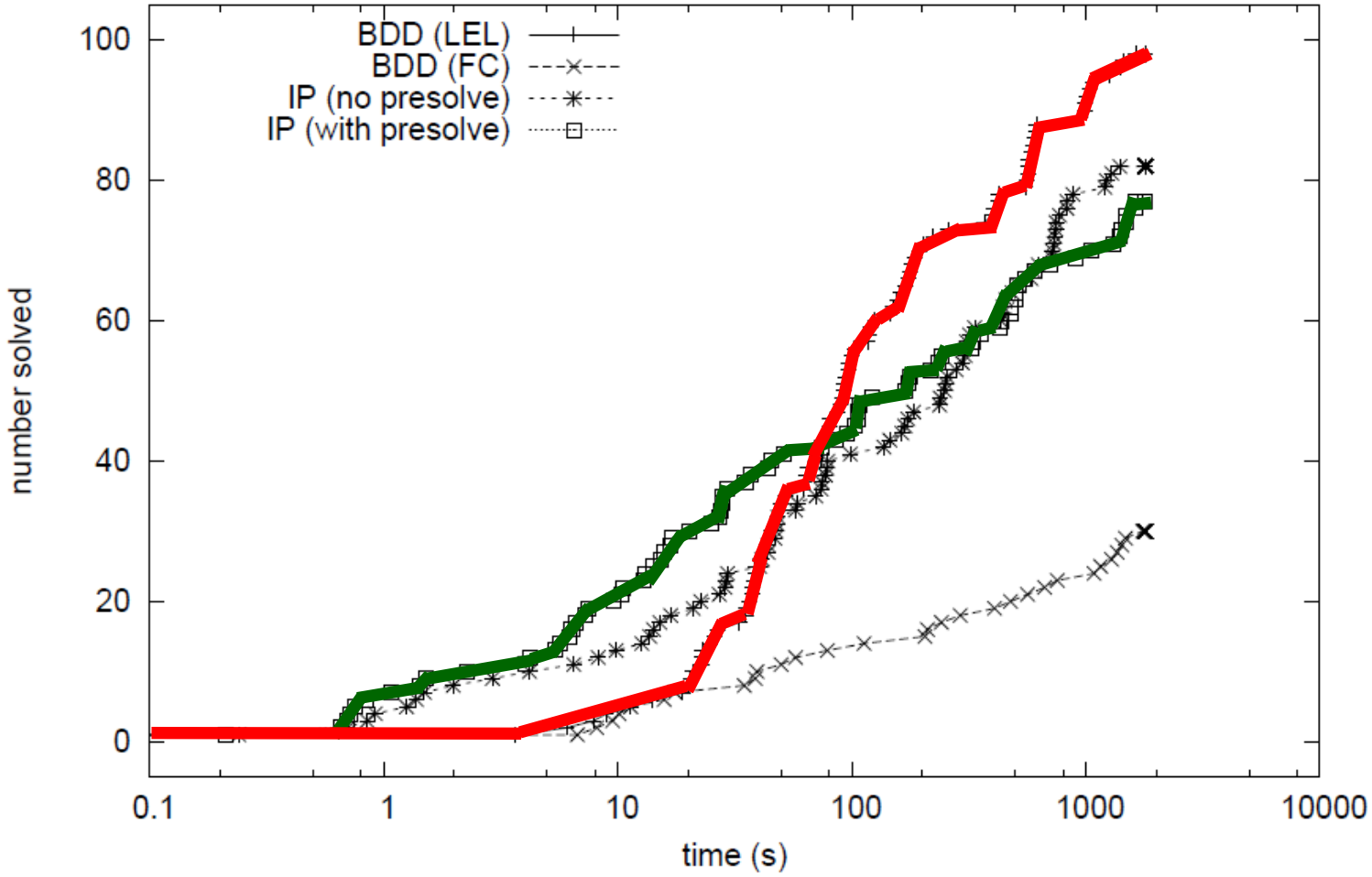
CPLEX

BDDs

Random max 2-SAT instances

40 variables, 100 instances

Number of instances solved



CPLEX

BDDs

IP Applications

- Tighter bounds
  - Use BDDs to improve IP bound.
  - Often better than LP + all cuts generated at root node.
  - Bound obtained in much less time.
- More effective primal heuristic
  - Often better than IP primal heuristics

Conclusions

- Hard to evaluate a general-purpose solver.
  - Needs extensive testing on wide range of problems.
  - Solver development takes time.
    - MIP improved many orders of magnitude in 50 years.

Conclusions

- Hard to evaluate a general-purpose solver
  - Needs extensive testing on wide range of problems.
  - Solver development takes time.
    - MIP improved many orders of magnitude in 50 years.
- BDD-based optimization could be promising.
  - Simple, rudimentary algorithm.
  - But very limited computational experience.

Ongoing Research

- Additional problems
  - Problems that are hard to formulate as IP
    - Min bandwidth, linear arrangement
    - Quadratic assignment
    - Nonlinear objective functions

Ongoing Research

- Additional problems
  - Problems that are hard to formulate as IP
    - Min bandwidth, linear arrangement
    - Quadratic assignment
    - Nonlinear objective functions
- Parallel computation
  - Very close to linear average speedup.
  - Frequently superlinear.
  - IP very hard to parallelize.

Ongoing Research

- Additional problems
  - Problems that are hard to formulate as IP
    - Min bandwidth, linear arrangement
    - Quadratic assignment
    - Nonlinear objective functions
- Parallel computation
  - Very close to linear average speedup.
  - Frequently superlinear.
  - IP very hard to parallelize.
- Finding all (near) optimal solutions
  - Build BDD of optimal solutions after one optimal solution is obtained…
    - perhaps by another solver.

Future Research

- Stochastic optimization
  - Use BDDs with chance nodes.
- Continuous variables
  - We have some ideas…
- Nonserial BDDs
  - Model is nonserial DP.
- Other types of knowledge representation
  - And/or graphs, join graphs, etc.