# Scheduling Home Health Care with Separating Benders Cuts in Decision Diagrams

André Ciré
University of Toronto

John Hooker
Carnegie Mellon University

INFORMS 2014

# Home Health Care

- Home health care delivery problem.
  - Assign **nurses** to homebound **patients**.
    - …subject to constraints on nurse qualifications.
  - **Route** each nurse through assigned patients, observing **time windows**.

# Home Health Care

- Home health care delivery problem.
  - Assign **nurses** to homebound **patients**.
    - …subject to constraints on nurse qualifications.
  - **Route** each nurse through assigned patients, observing **time windows**.
    - Additional constraints and work rules.
    - One patient may require a team of nurses.

# Home Health Care

- A **large industry**, and rapidly growing.
  - Roughly as large as all courier and delivery services.

### Relative Size of Two Industries

| | Home health care | Courier and delivery services |
|---|---|---|
| U.S. revenues, $ billions | 75 | 93 |
| U.S. workers, millions | 1.5 | 0.7 |
| World revenue, $ billions | 196 | 206 |

# Home Health Care

- A large industry, and **rapidly growing**.
  - Roughly as large as all courier and delivery services.

### Projected Growth of Home Health Care Industry

| | 2014 | 2018 |
|---|---|---|
| U.S. revenues, $ billions | 75 | 150 |
| World revenues, $ billions | 196 | 306 |

### Increase in U.S. Employment, 2010-2020

| | |
|---|---|
| Home health care industry | 70% |
| Entire economy | 14% |

# Home Health Care

- Advantages of home health care
  - Lower cost
    - Hospital space is very expensive.

# Home Health Care

- Advantages of home health care
  - Lower cost
    - Hospital space is very expensive.
  - No hospital-acquired infections
    - Less exposure to superbugs.

# Home Health Care

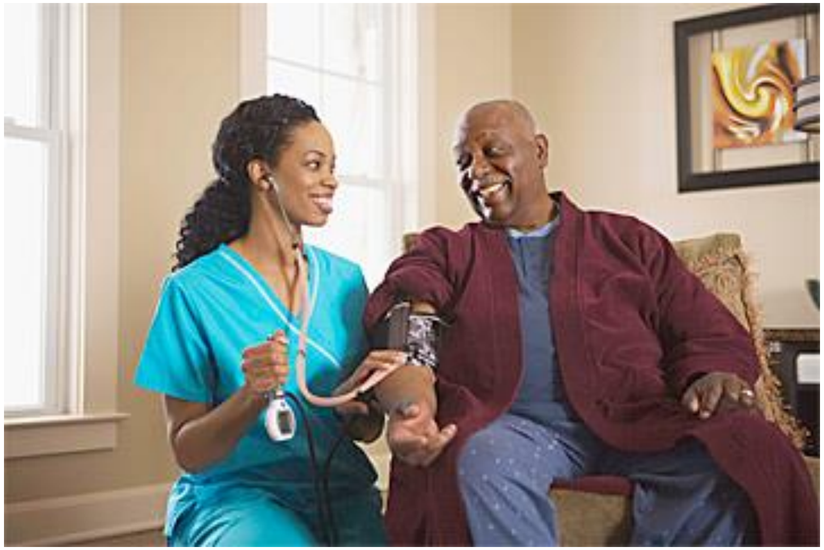
- Advantages of home health care
  - Lower cost
    - Hospital space is very expensive.
  - No hospital-acquired infections
    - Less exposure to superbugs.
  - Preferred by patients
    - Comfortable, familiar surroundings of home.
    - Sense of control over one’s life.

# Home Health Care

- Advantages of home health care
  - Lower cost
    - Hospital space is very expensive.
  - No hospital-acquired infections
    - Less exposure to superbugs.
  - Preferred by patients
    - Comfortable, familiar surroundings of home.
    - Sense of control over one's life.
  - Supported by new equipment & technology
    - IT integration with hospital systems.

# Home Health Care

- Critical factor to realize cost savings:
  - Nurses must be **efficiently** scheduled.
- This is our task.
  - Computational results very preliminary.



# Benders approach

- Solve the problem using **logic-based Benders decomposition**.
  - Master problem **assigns** nurses to patients.
  - Subproblem finds **routes** and **schedules** for nurses.

# Benders approach

- Solve the problem using **logic-based Benders decomposition**.
  - Master problem **assigns** nurses to patients.
    - This is the bottleneck.
    - Use a **decision diagram** to formulate the master.
    - Master problem becomes a **shortest path problem**.
  - Subproblem finds **routes** and **schedules** for nurses.

# Benders approach

- Solve the problem using **logic-based Benders decomposition**.
  - Master problem **assigns** nurses to patients.
    - This is the bottleneck.
    - Use a **decision diagram** to formulate the master.
    - Master problem becomes a **shortest path problem**.
  - Subproblem finds **routes** and **schedules** for nurses.
    - Can't use classical Benders.
    - Add logic-based Benders cuts to master.
    - This poses a **separation problem** for the decision diagram.

# Benders approach

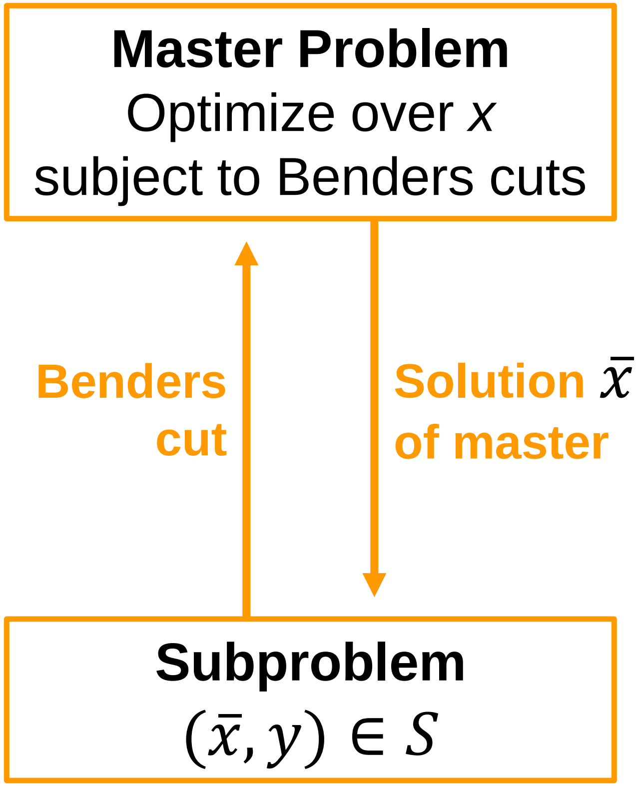
- Solve the problem using **logic-based Benders decomposition**.
  - Master problem **assigns** nurses to patients.
    - This is the bottleneck.
    - Use a **decision diagram** to formulate the master.
    - Master problem becomes a **shortest path problem**.
  - Subproblem finds **routes** and **schedules** for nurses.
    - Can’t use classical Benders.
    - Add logic-based Benders cuts to master.
    - This poses a **separation problem** for the decision diagram.
  - Focus on finding **feasible solutions**.

# Logic-based Benders

- **Logic-based Benders decomposition** is a generalization of classical Benders.
  - Consider a feasibility problem:

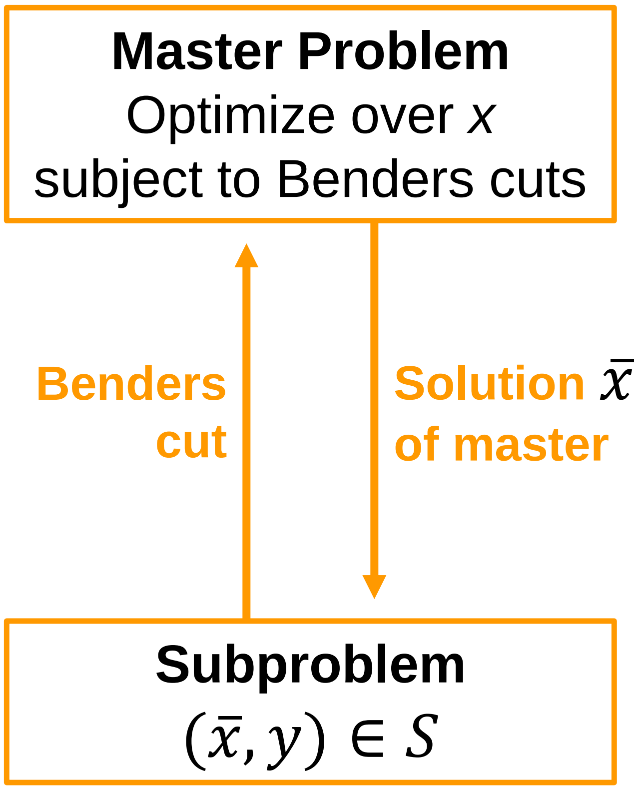
$$\min f(x)$$
$$(x, y) \in S$$

  - Benders cut excludes $\bar{x}$ (and perhaps similar solutions) if it is infeasible in the subproblem.
  - Benders cut based on **inference dual**
  - Algorithm terminates when $\bar{x}$ is feasible in the subproblem.



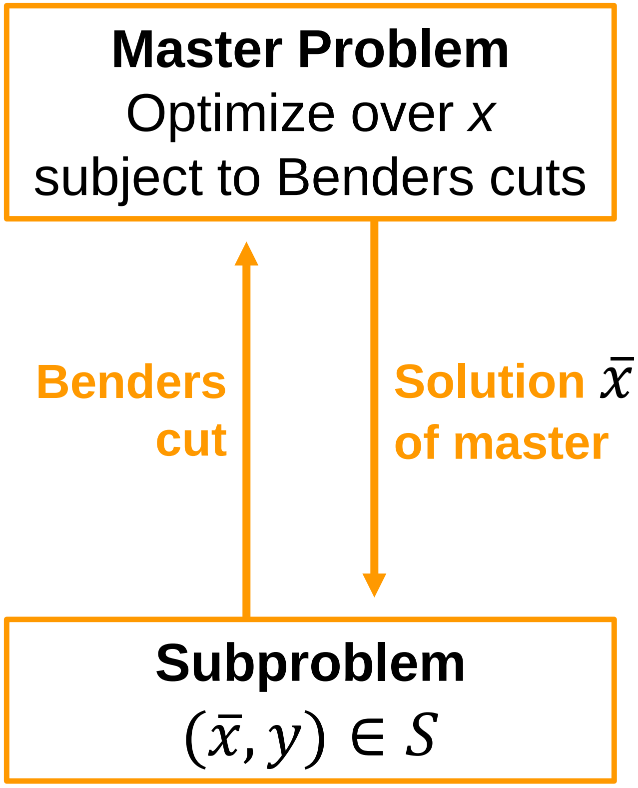
# Benders Decomposition

- Logic-based Benders decomposition is a generalization of classical Benders.
  - Master problem is initially a **relaxation** of the original problem over $x$ (warm start).
  - Relaxation becomes **tighter** as Benders cuts are added.



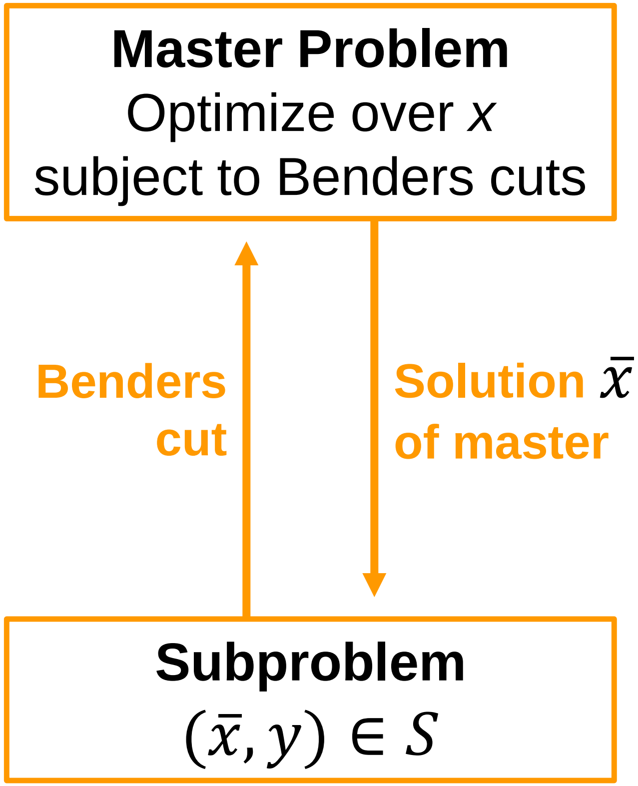
# Benders Decomposition

- Logic-based Benders decomposition is a generalization of classical Benders.
  - Master problem is initially a **relaxation** of the original problem over *x* (warm start).
  - Relaxation becomes **tighter** as Benders cuts are added.
  - We will use **relaxed decision diagram** to represent master problem.



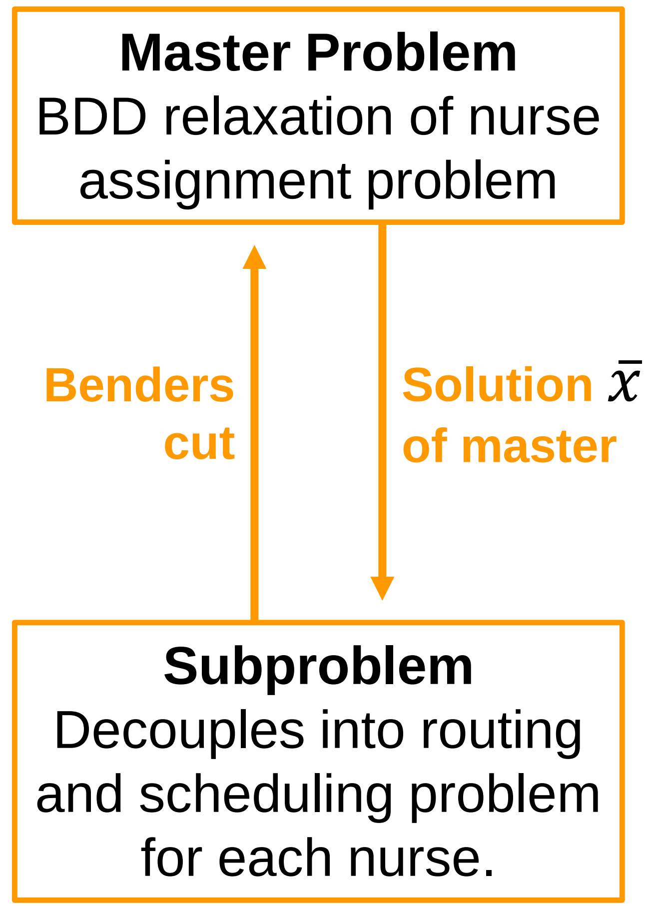
# Benders Decomposition

- Logic-based Benders decomposition is a generalization of classical Benders.
  - Master problem is initially a **relaxation** of the original problem over *x* (warm start).
  - Relaxation becomes **tighter** as Benders cuts are added.
  - We will use **relaxed decision diagram** to represent master problem.
  - Add a Benders cut by creating a **separating decision diagram**.



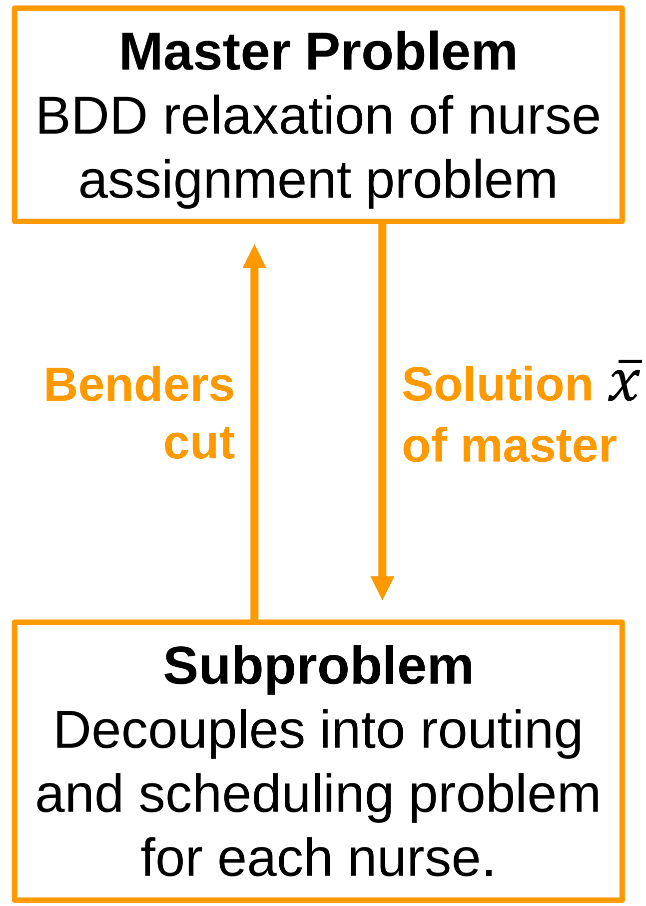
# Home Health Care

- Solve with Benders decomposition.
  - **Assignment problem** in master.
  - Subproblem generates Benders cuts when there is no feasible schedule for one or more nurses.
  - Each cut excludes a **partial assignment** of nurses to patients that **causes infeasibility**.



# Home Health Care

- Solve with Benders decomposition.
  - **Assignment problem** in master.
  - Subproblem generates Benders cuts when there is no feasible schedule for one or more nurses.
  - Each cut excludes a **partial assignment** of nurses to patients that **causes infeasibility**.
- How fast does separating decision diagram grow?



# Decision Diagrams

- **Binary decision diagrams (BDD**s**)** 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.
    - Hadzic & Hooker 2007.
  - Easy generalization to **multivalued** decision diagrams (MDDs) for finite domain variables.

# Decision Diagrams

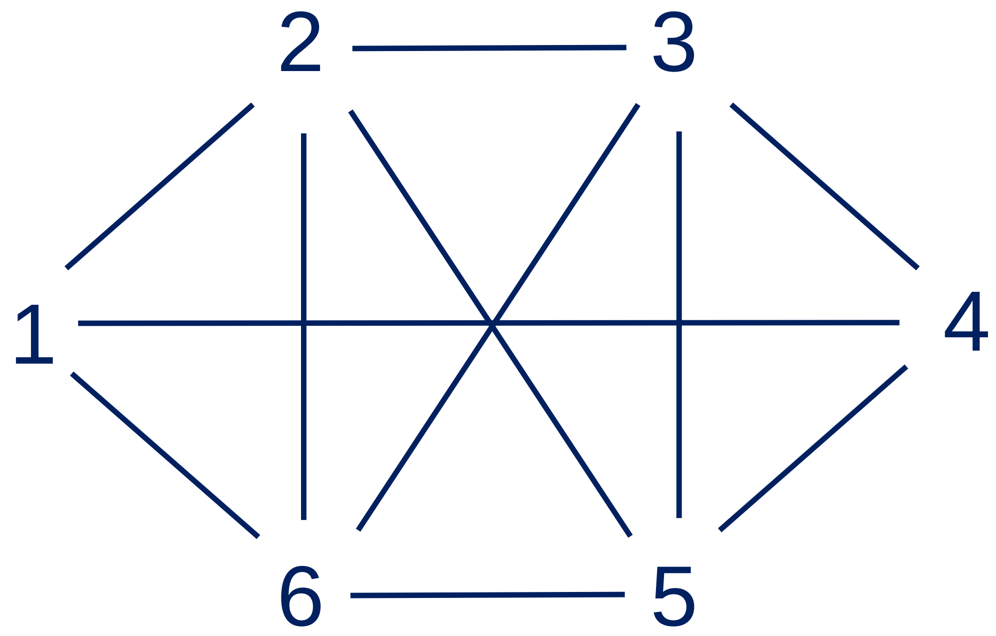
- Decision diagram can grow exponentially with problem size.
  - So we use a smaller, **relaxed** diagram that represents **superset** of feasible set.
    - Andersen, Hadzic, Hooker, Tiedemann 2007.
    - For graph coloring (alldiff systems), reduced CP search tree from >1 million nodes to 1 node.
- Example: **independent set problem** on a graph…

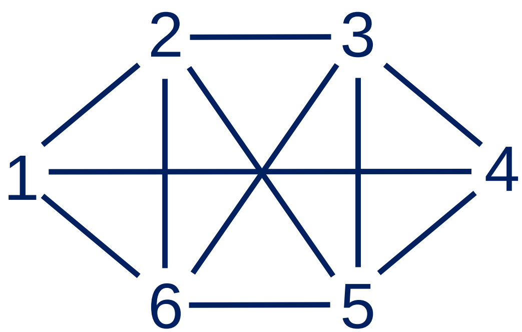
# Independent Set Problem

Let each vertex have weight $w_i$

Select nonadjacent vertices to maximize $\sum_i w_i x_i$

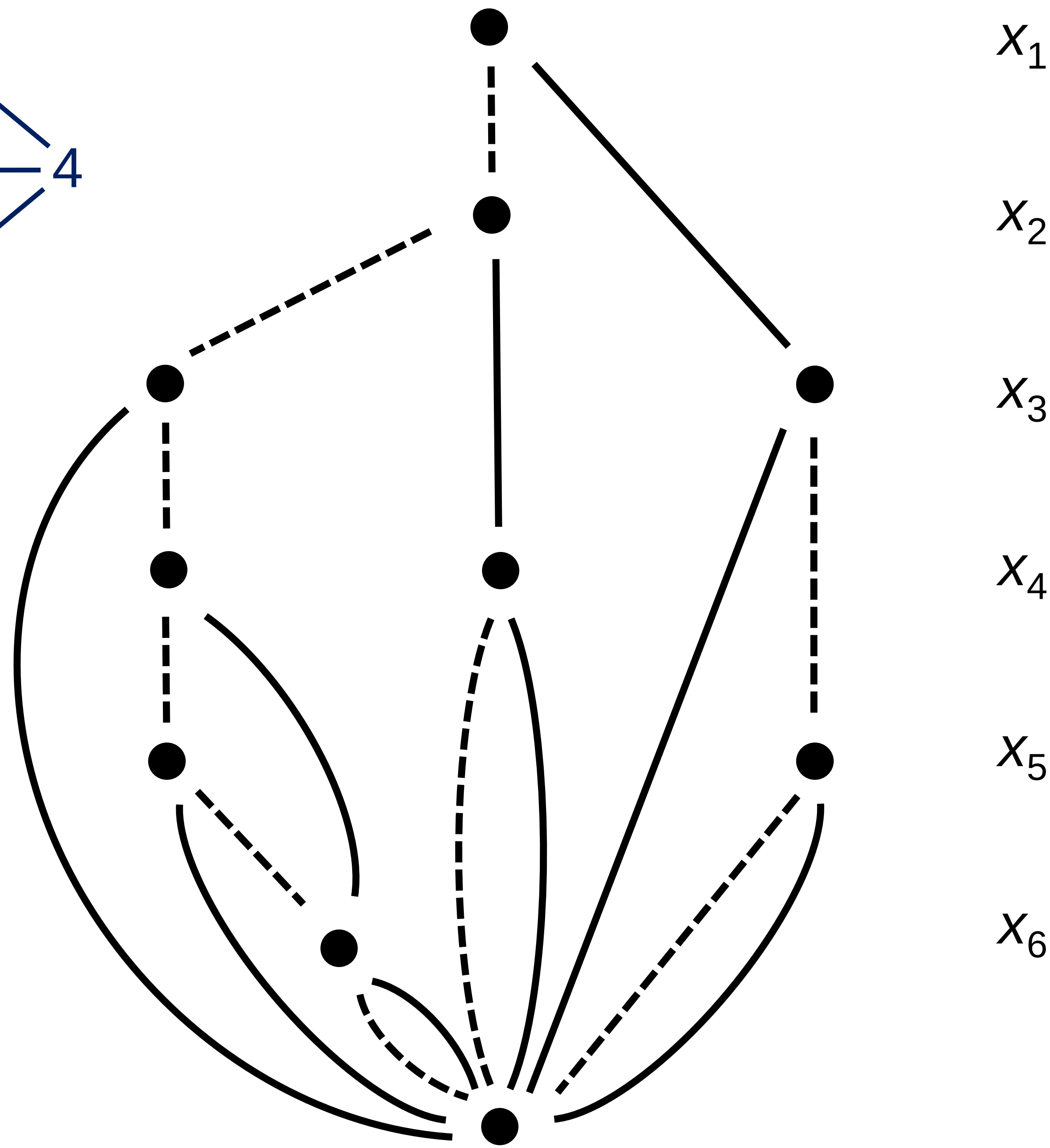
$x_i$ =1 if vertex i selected

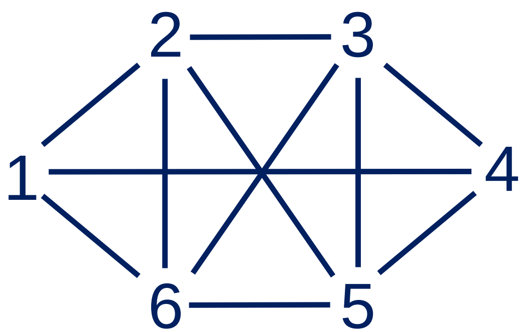





Exact BDD for independent set problem

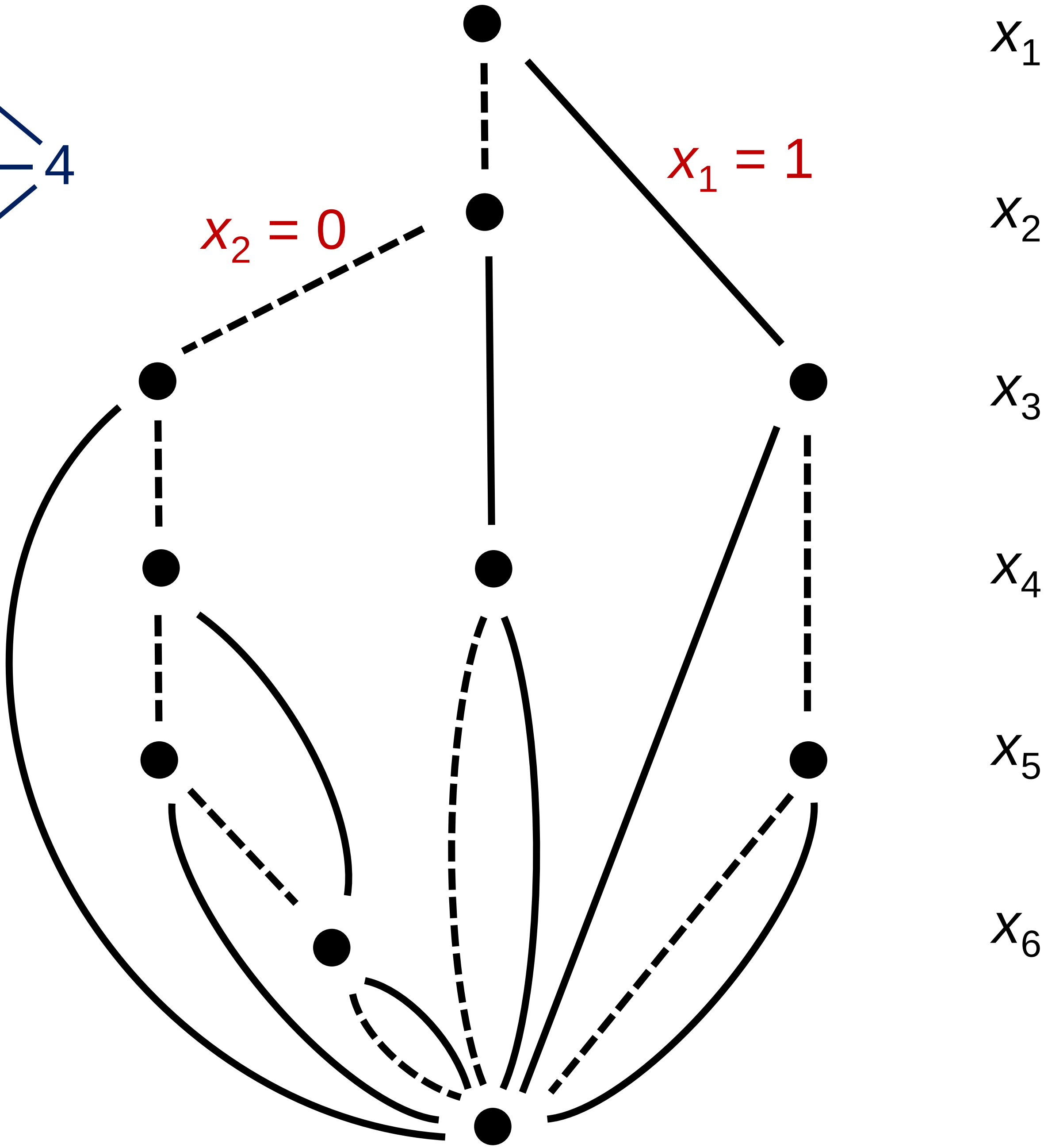
“zero-suppressed” BDD

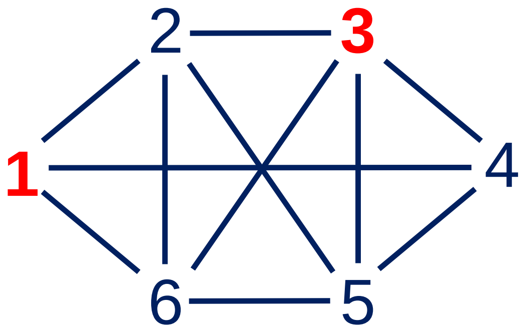




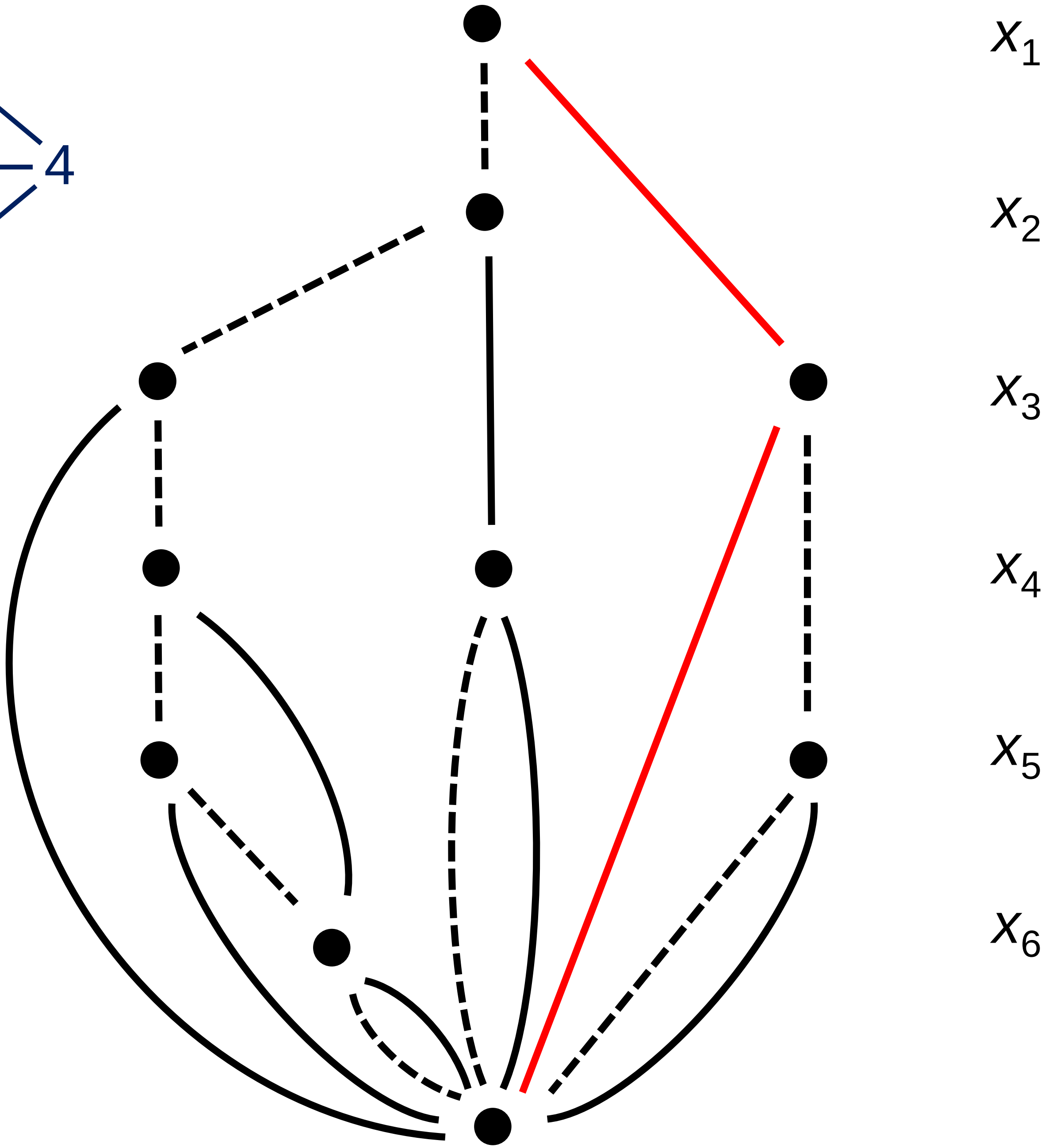

Exact BDD for independent set problem

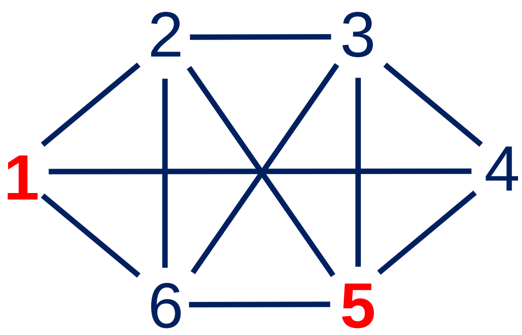
“zero-suppressed” BDD



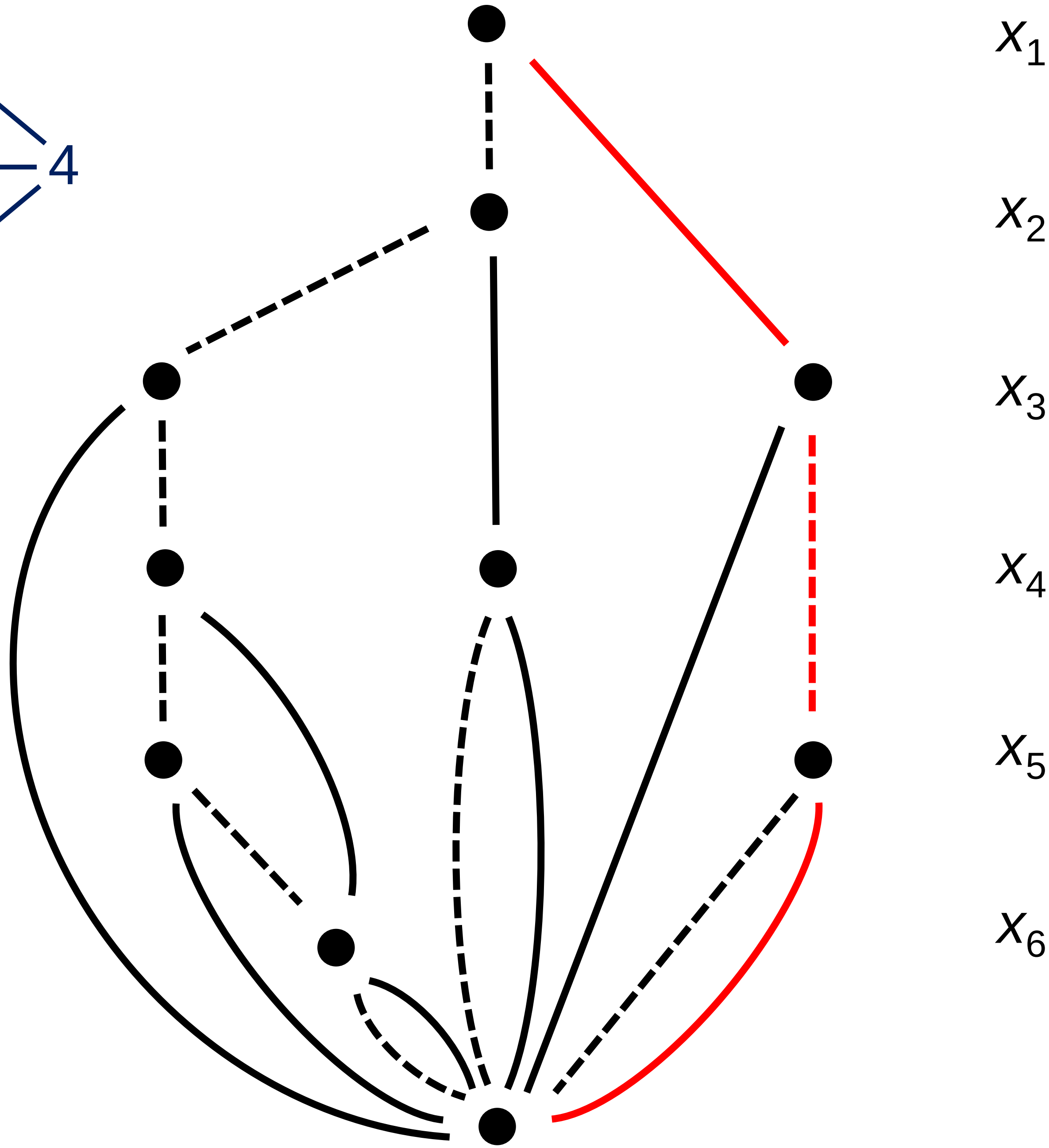


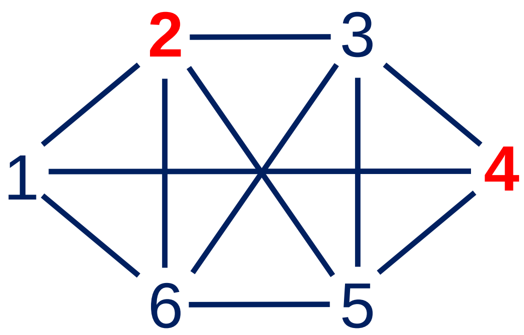

Paths from top
to bottom
correspond to
the 11 feasible
solutions



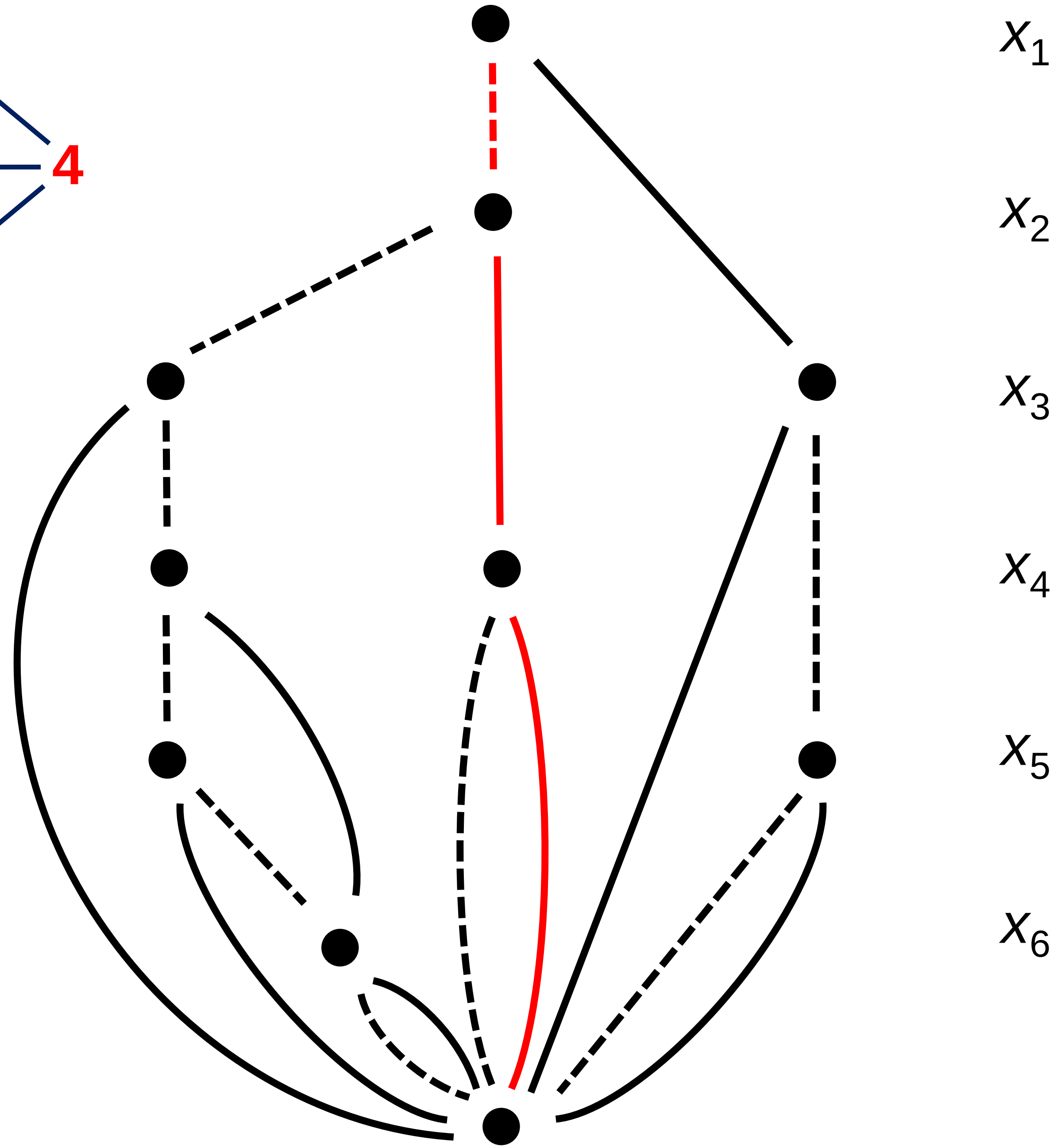


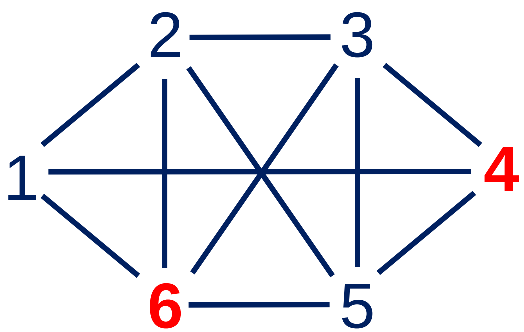

Paths from top to bottom correspond to the 11 feasible solutions



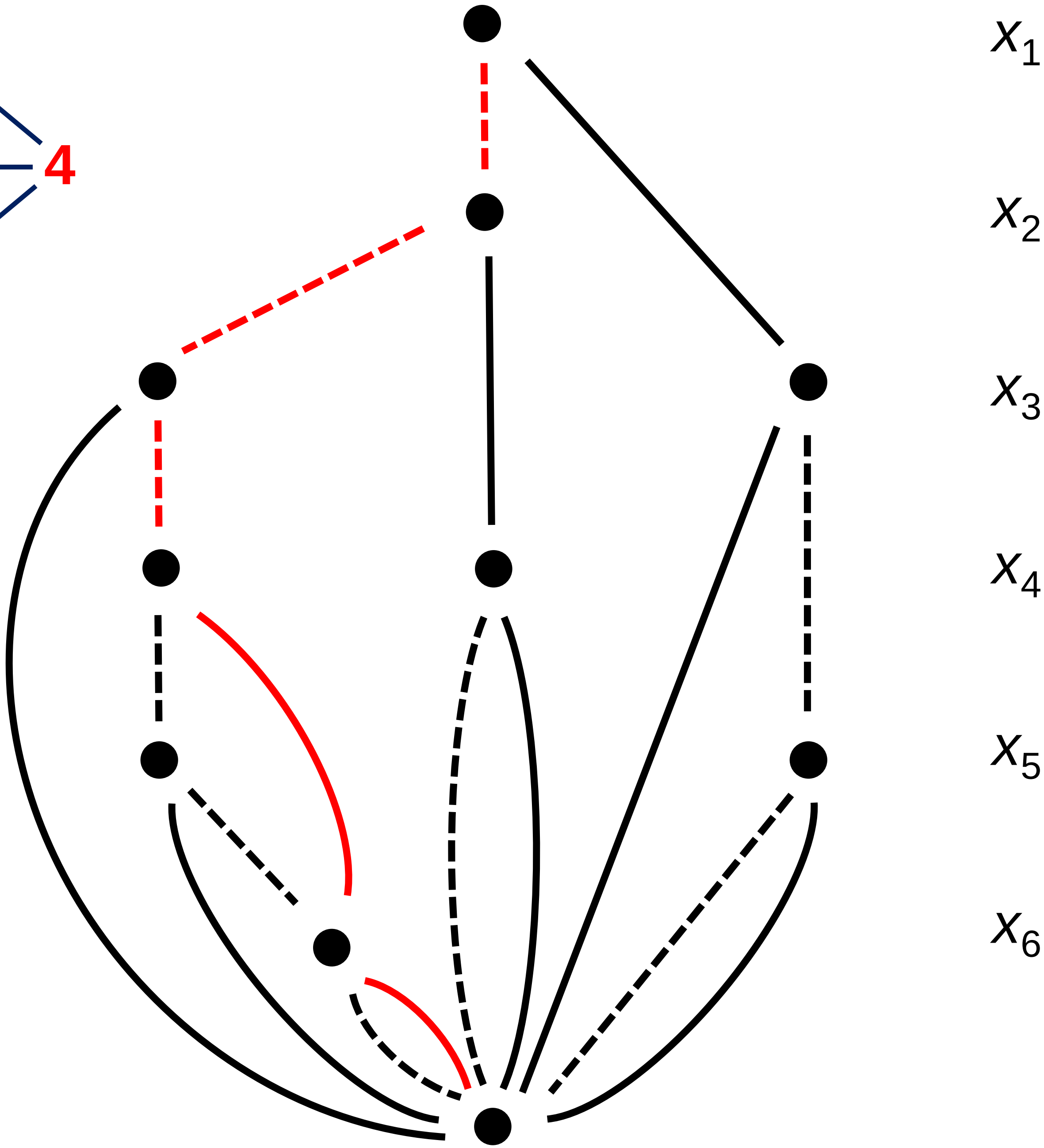


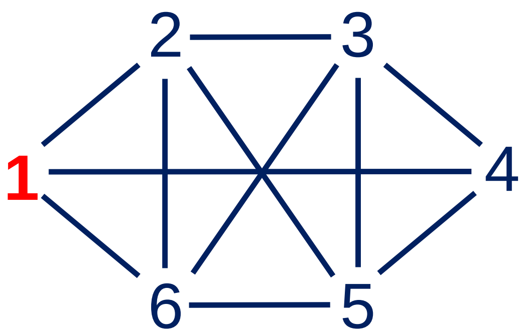

Paths from top to bottom correspond to the 11 feasible solutions



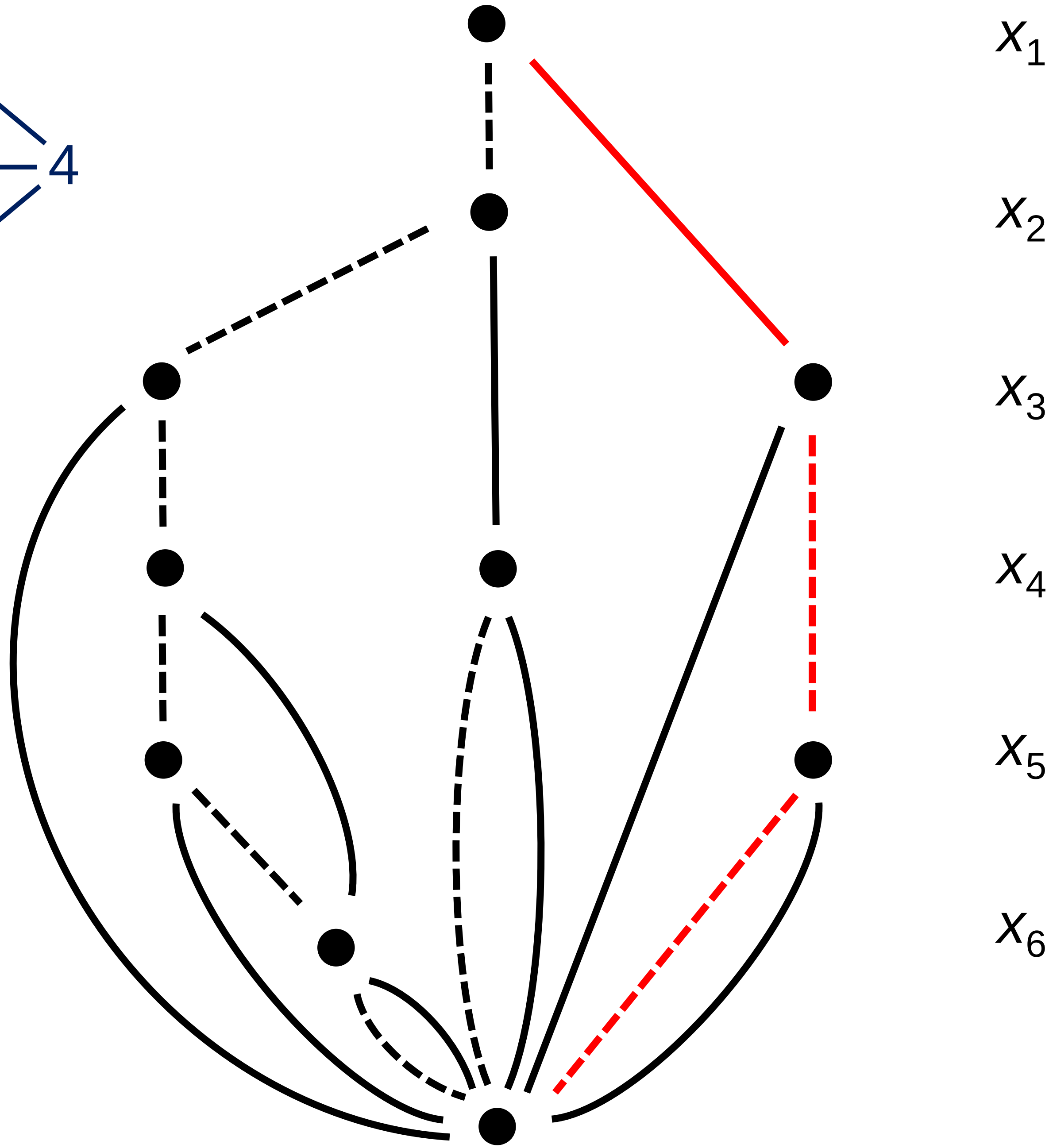



Paths from top to bottom correspond to the 11 feasible solutions

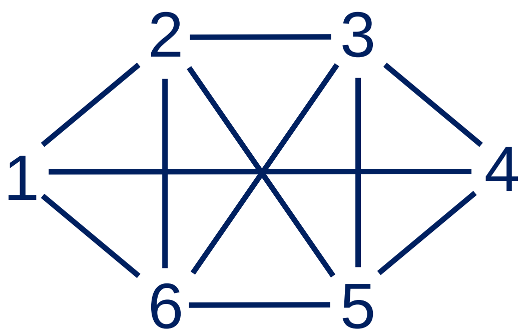




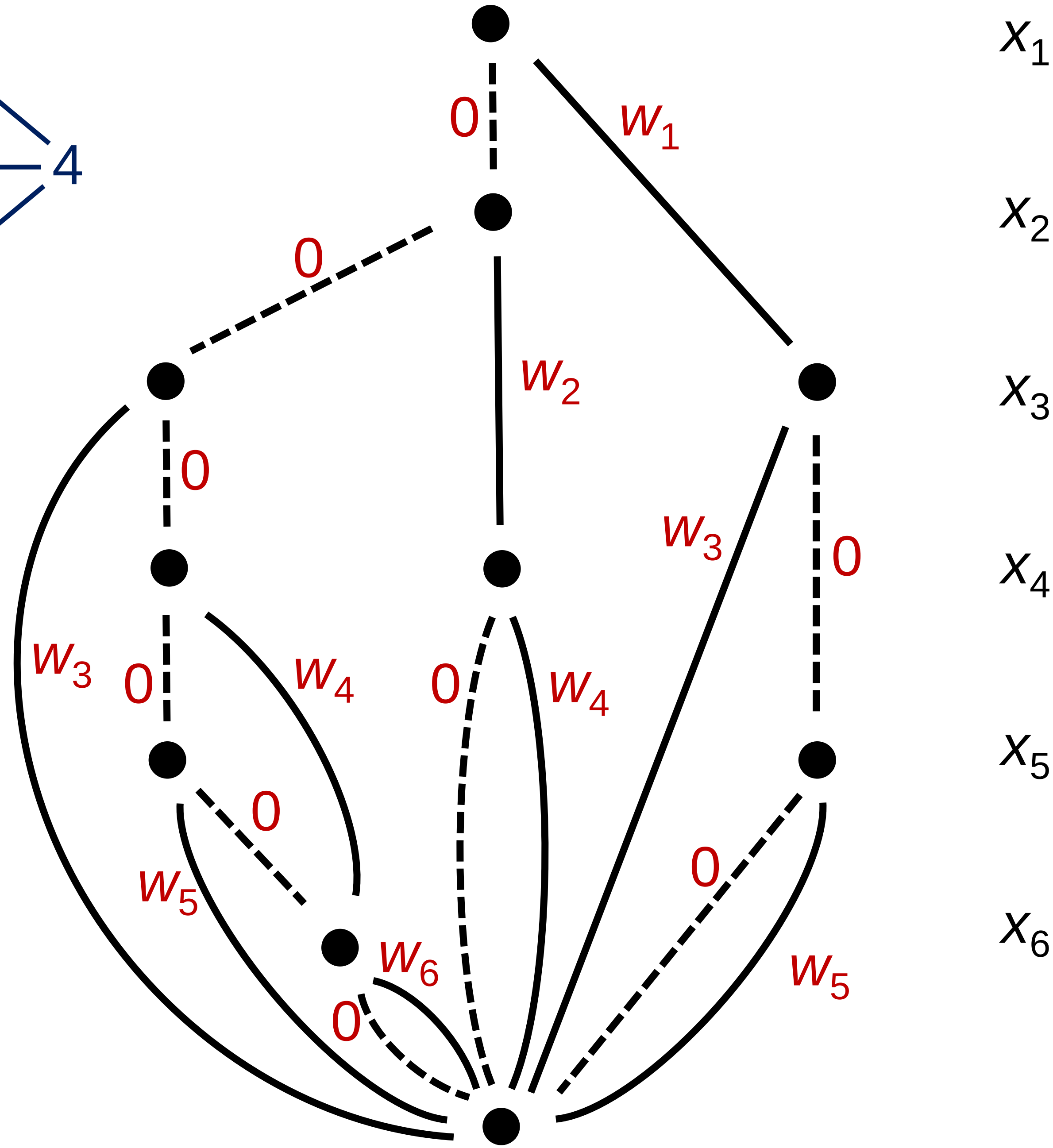

Paths from top to bottom correspond to the 11 feasible solutions

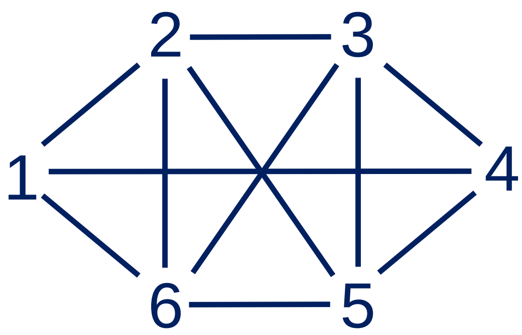


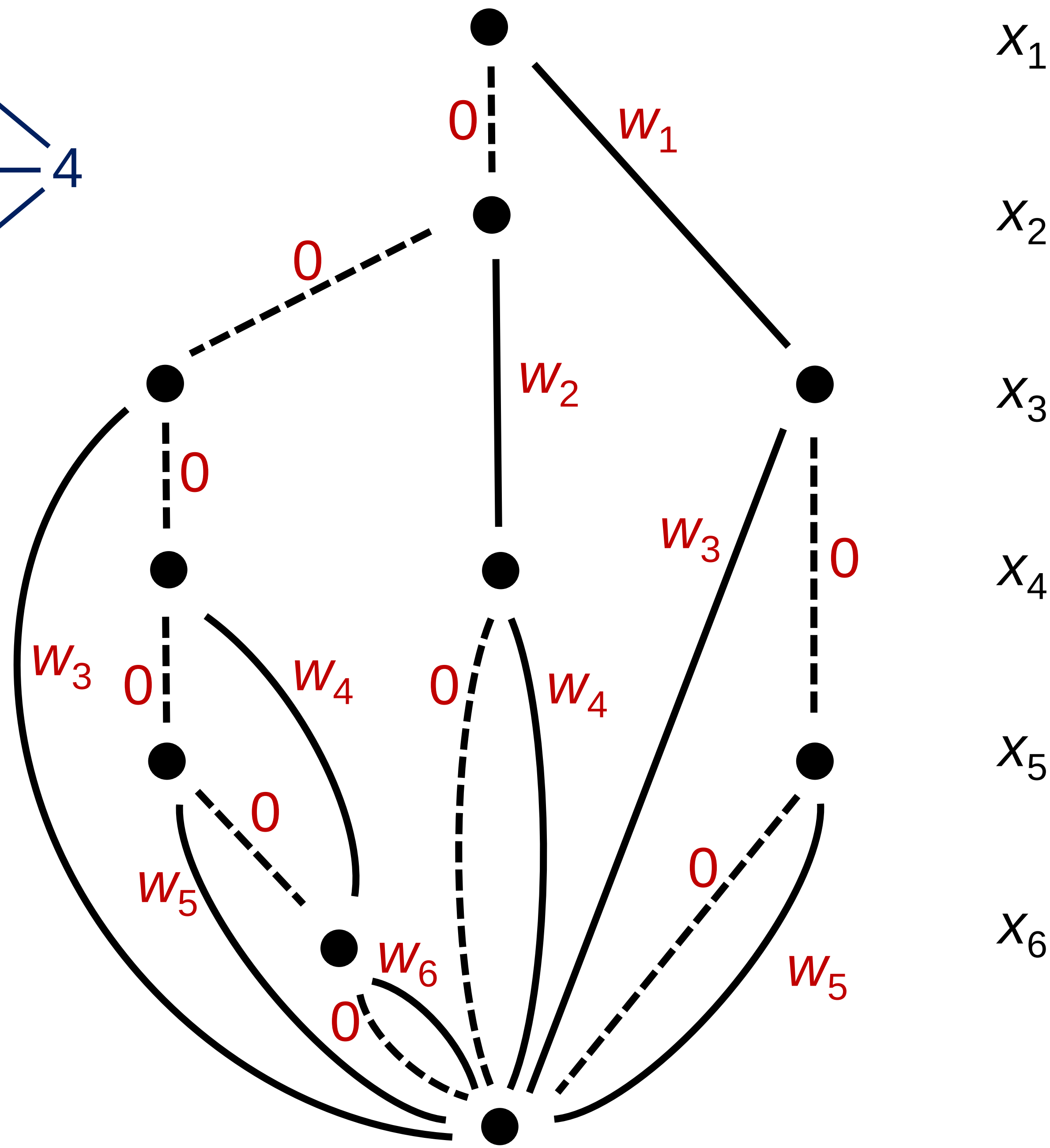

…and so forth




For objective function, associate weights with arcs

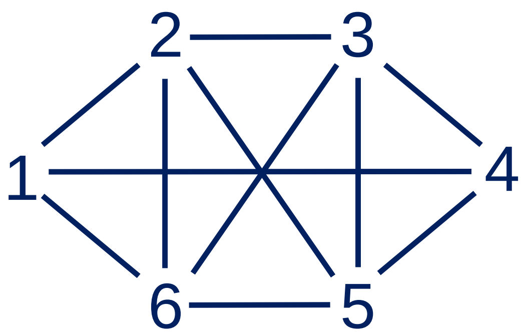


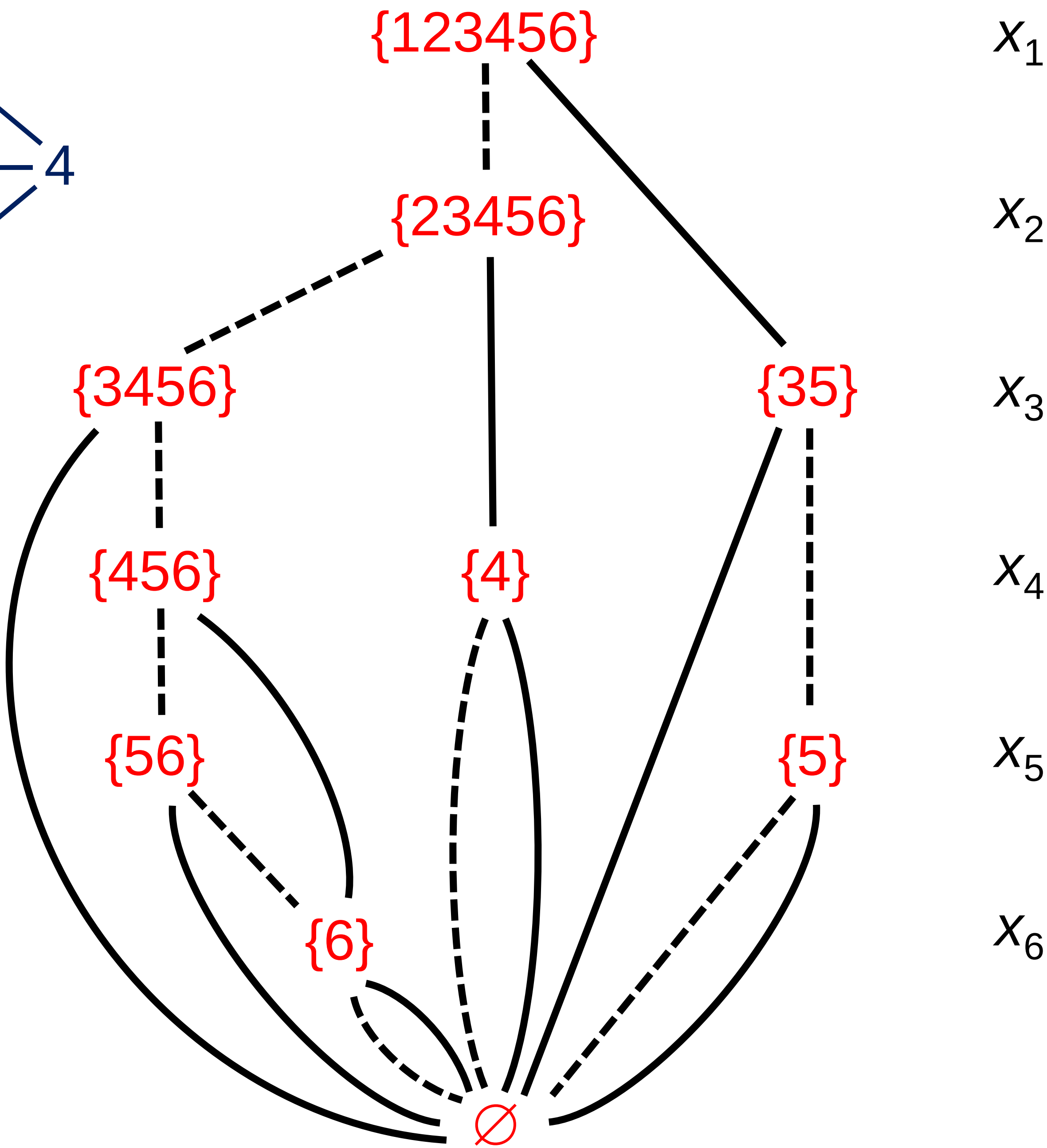


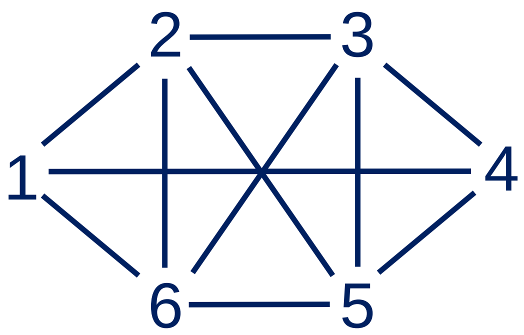

For objective function, associate weights with arcs

Optimal solution is **longest path**




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




{123456}

$x_1$

$x_2$

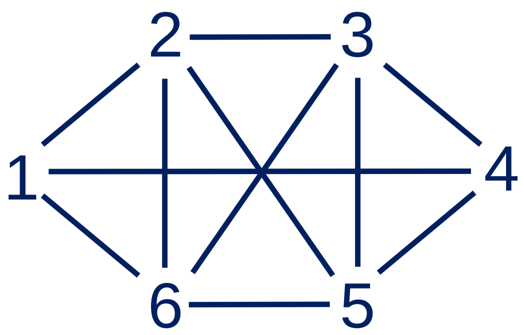
$x_3$

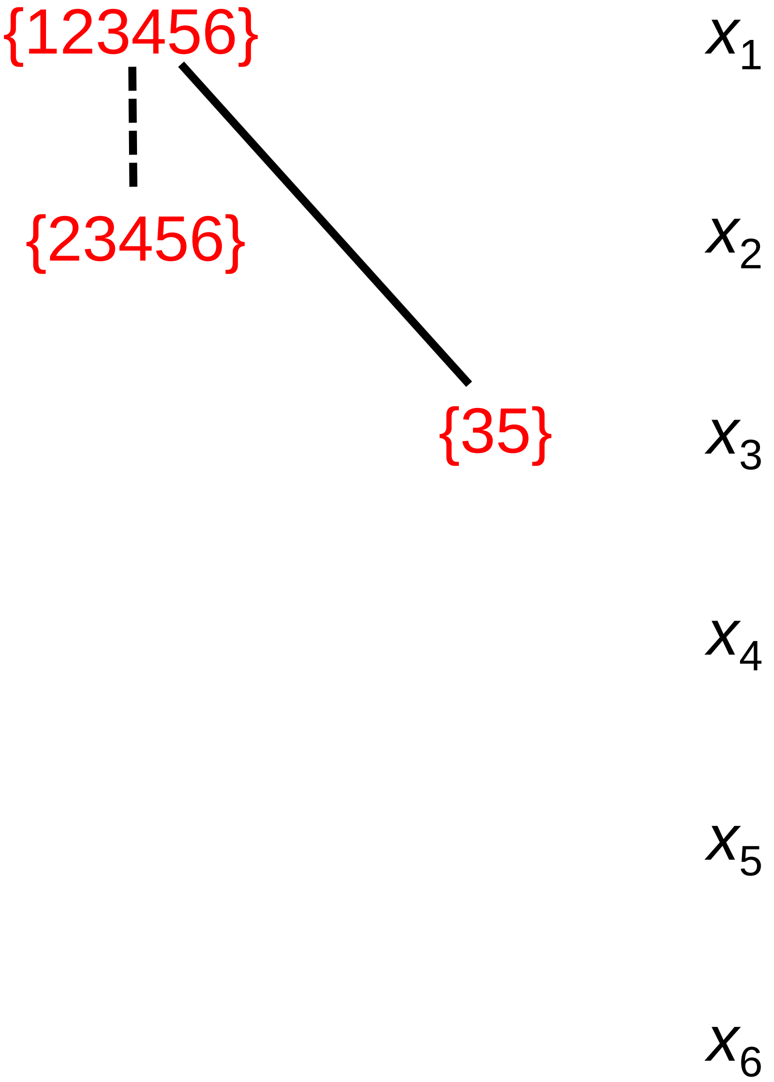
$x_4$

$x_5$

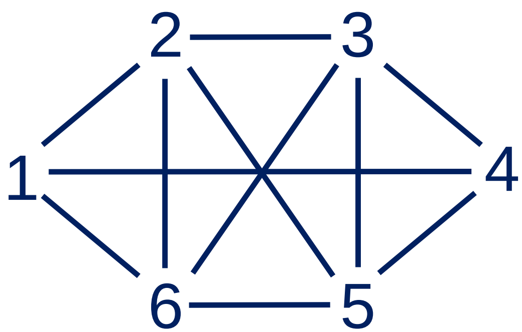
$x_6$

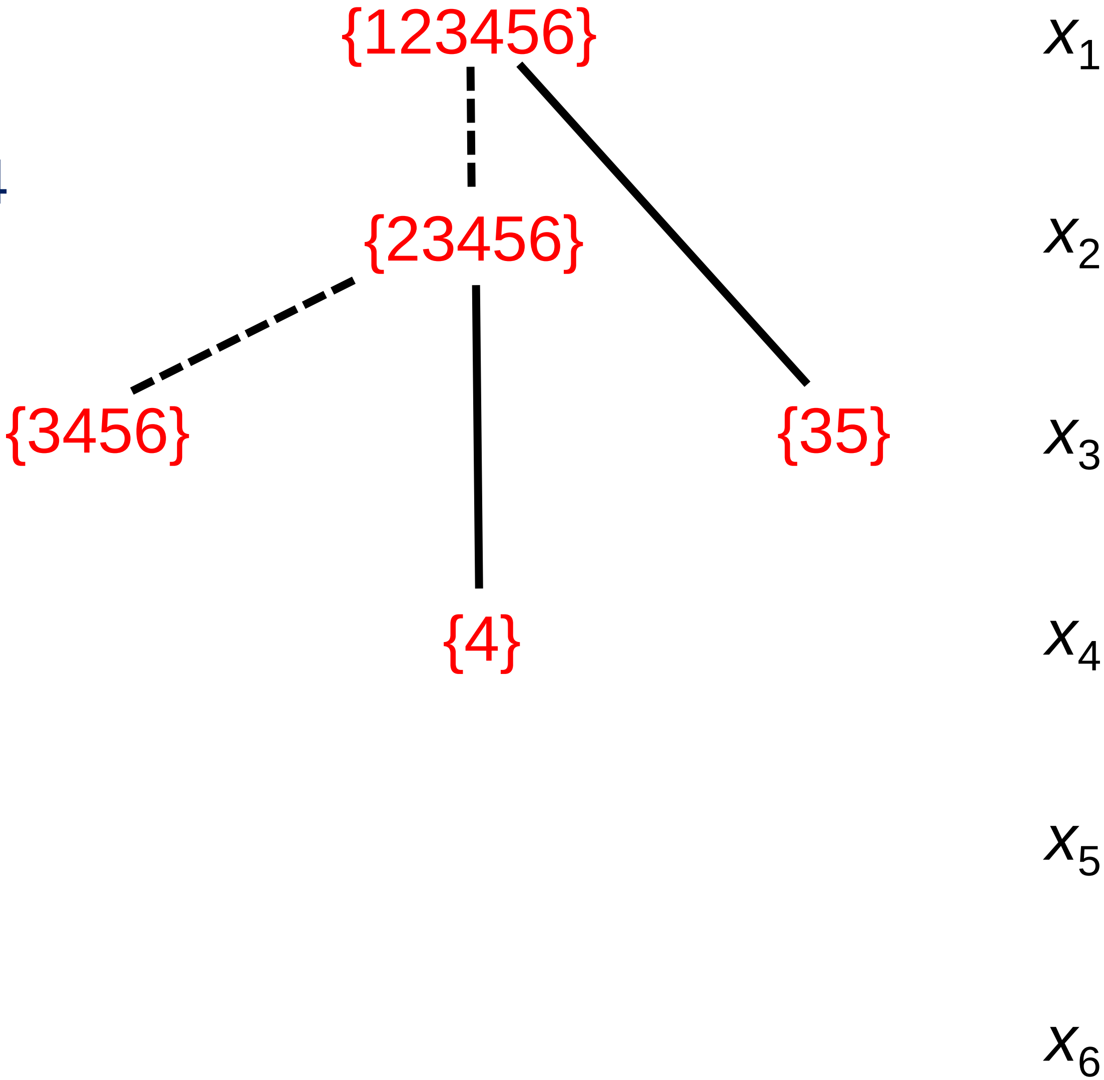
To build BDD, associate **state** with each node



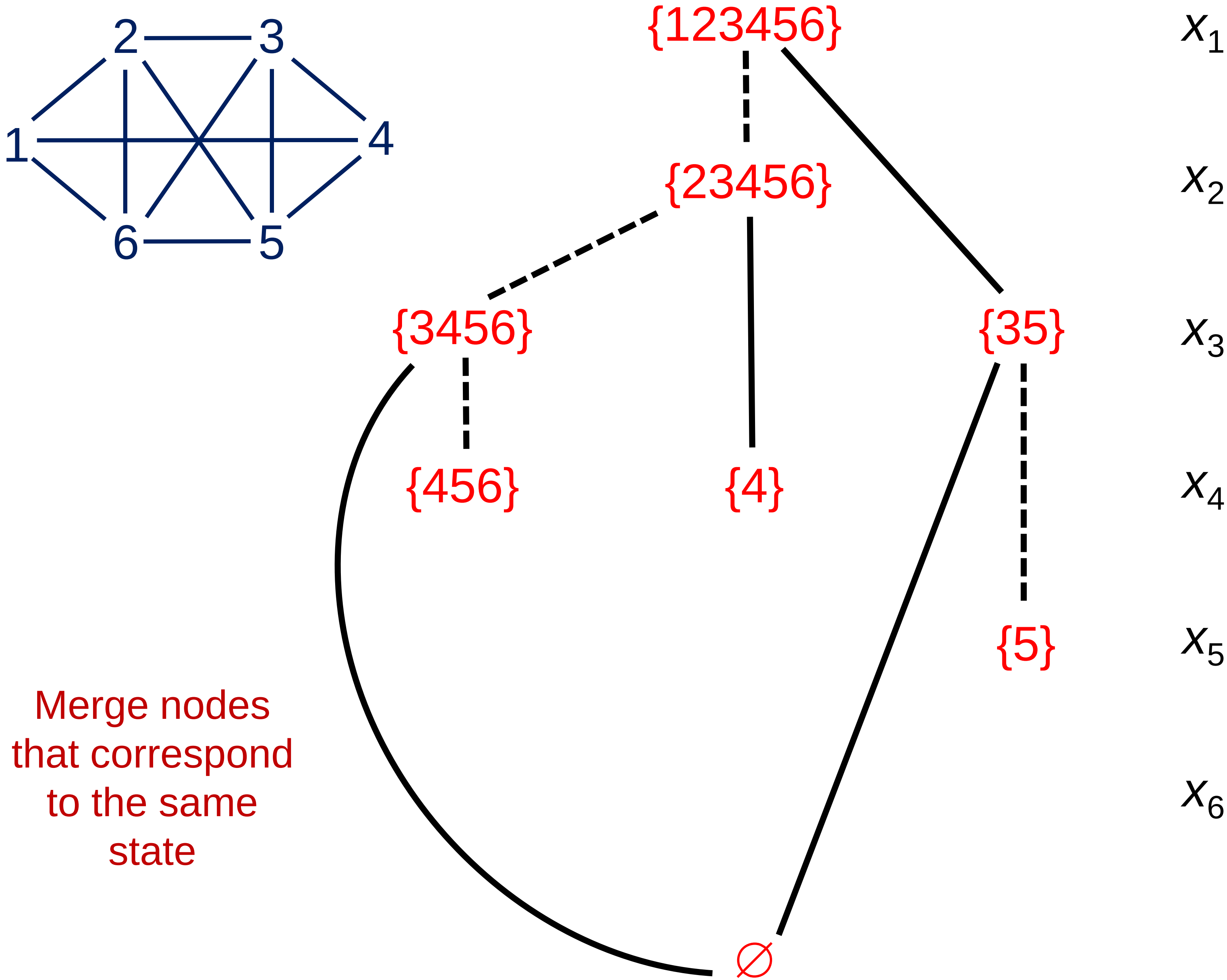




To build BDD,
associate **state**
with each node

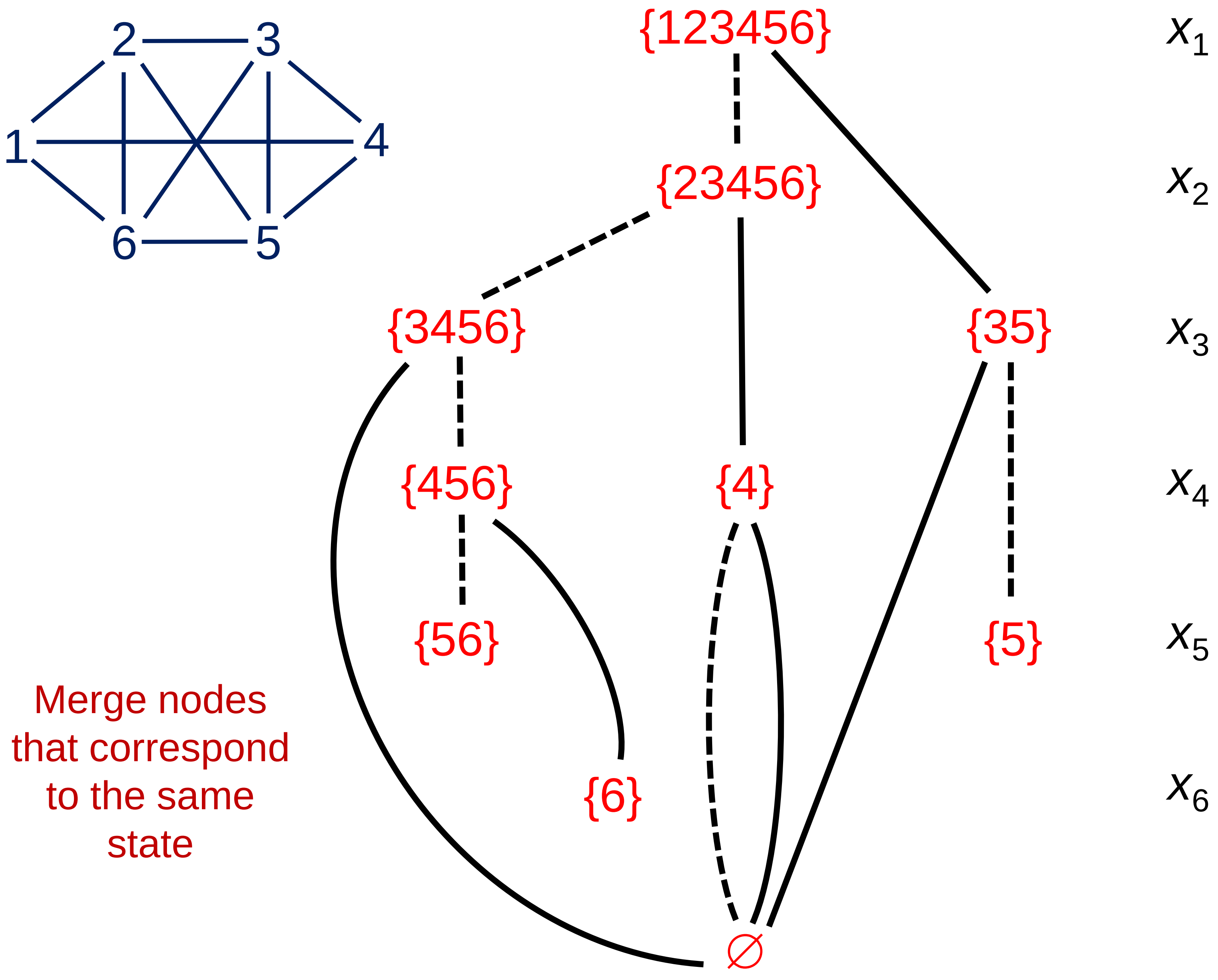


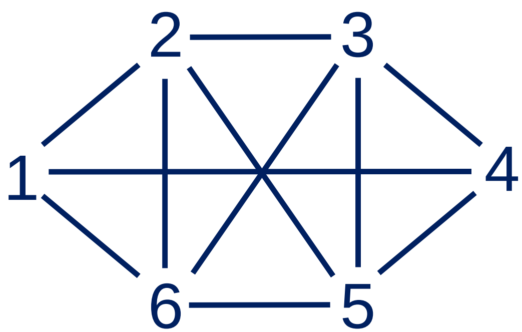

To build BDD, associate **state** with each node

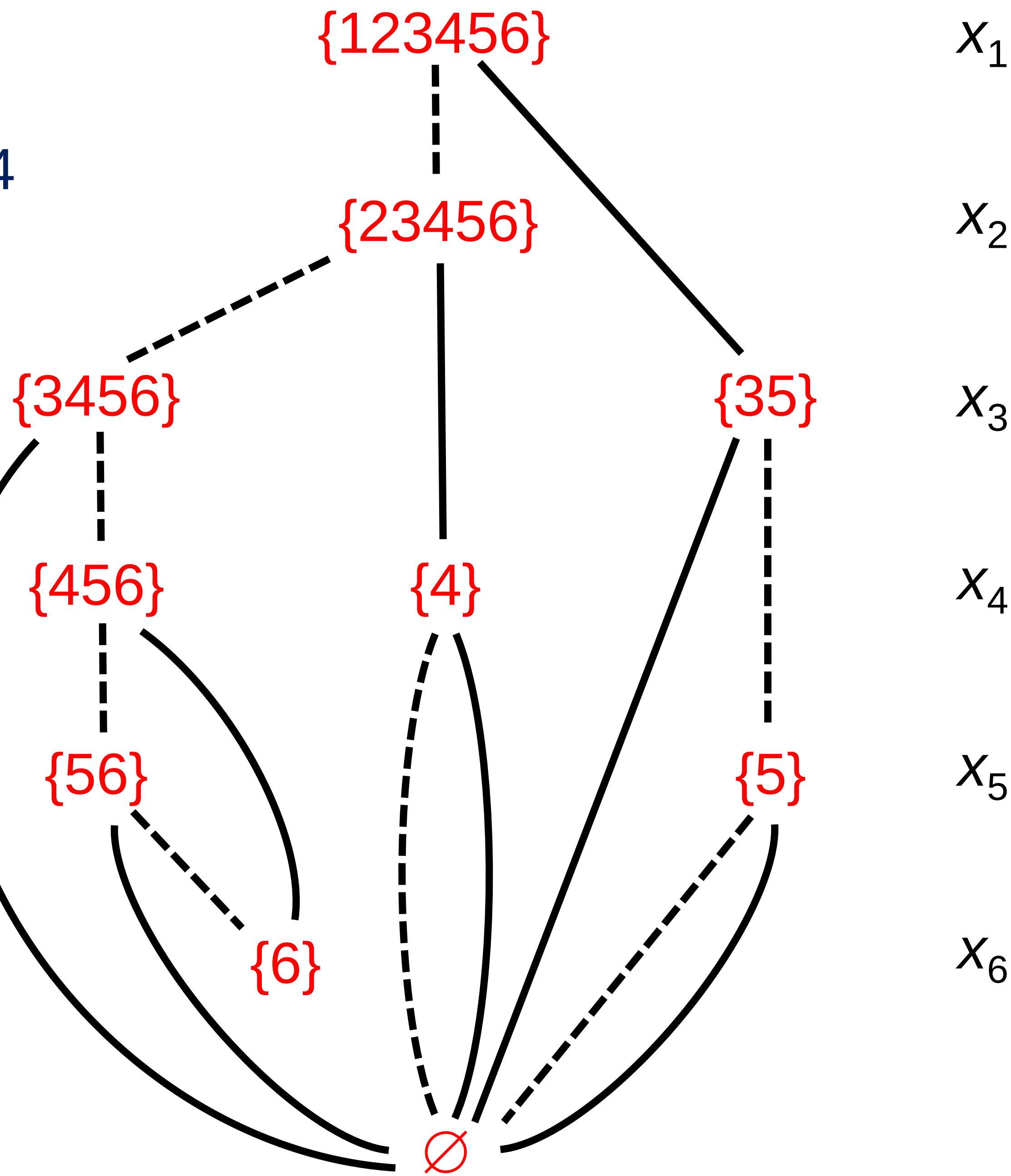



Merge nodes that correspond to the same state

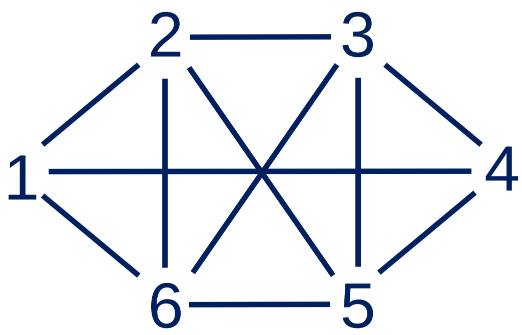


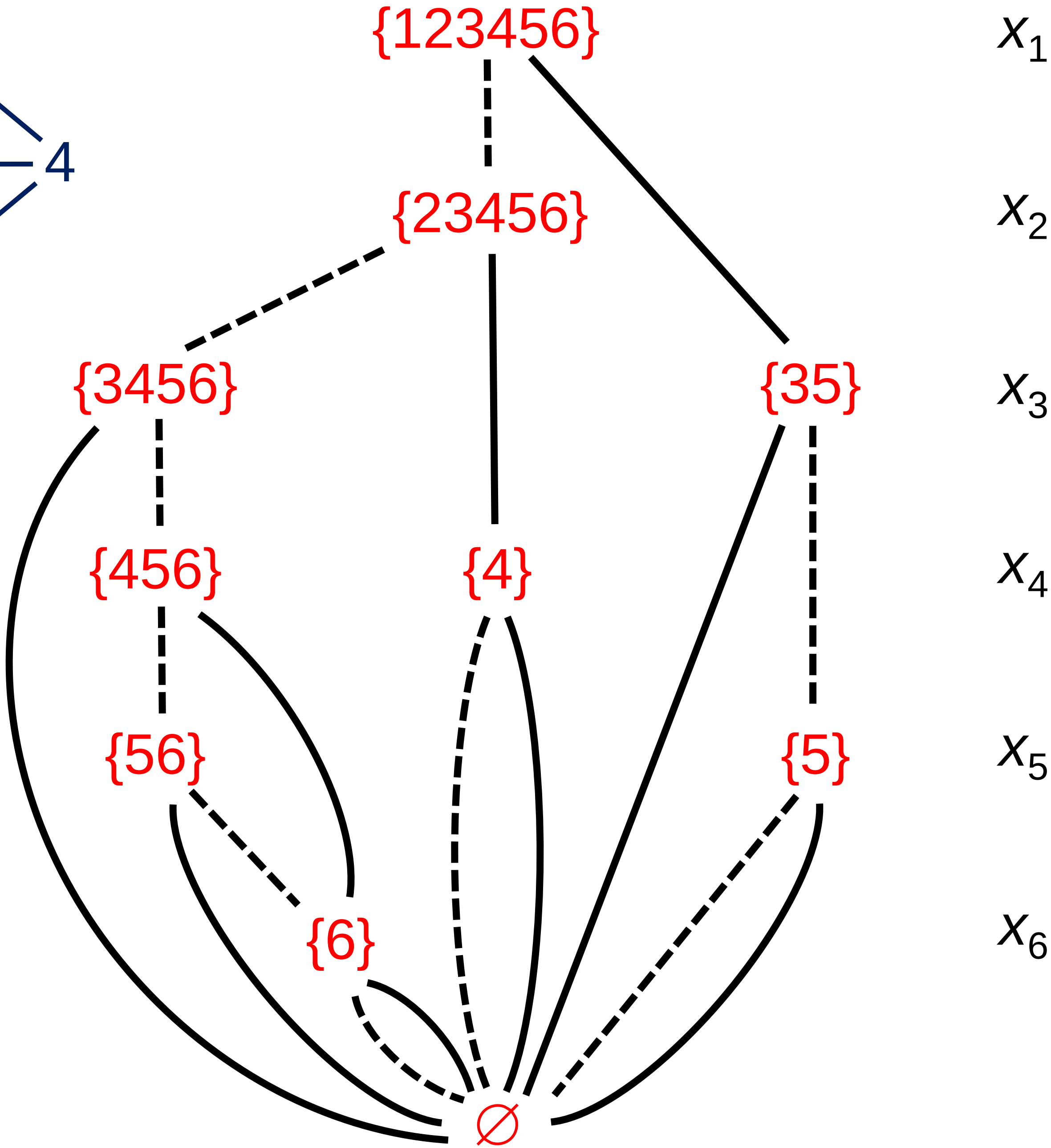

Merge nodes that correspond to the same state




Merge nodes that correspond to the same state



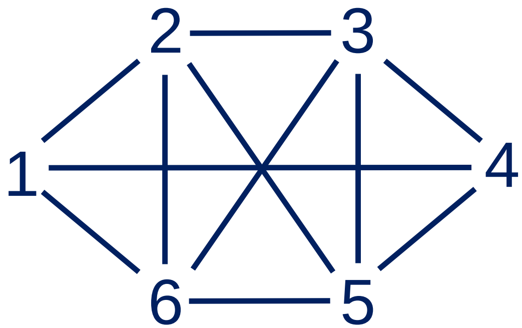




**Width** = 2

Merge nodes
that correspond
to the same
state

# Relaxation Bounding

- To obtain a bound on the objective function:
  - Use a **relaxed** decision diagram
  - Analogous to linear programming relaxation in MIP
  - This relaxation is **discrete**.
  - Doesn’t require the linear inequality formulation of MIP.




{123456}

$x_1$

$x_2$

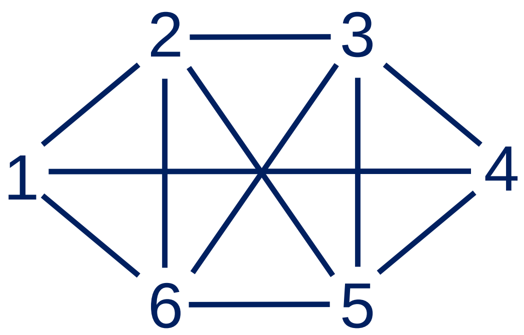
$x_3$

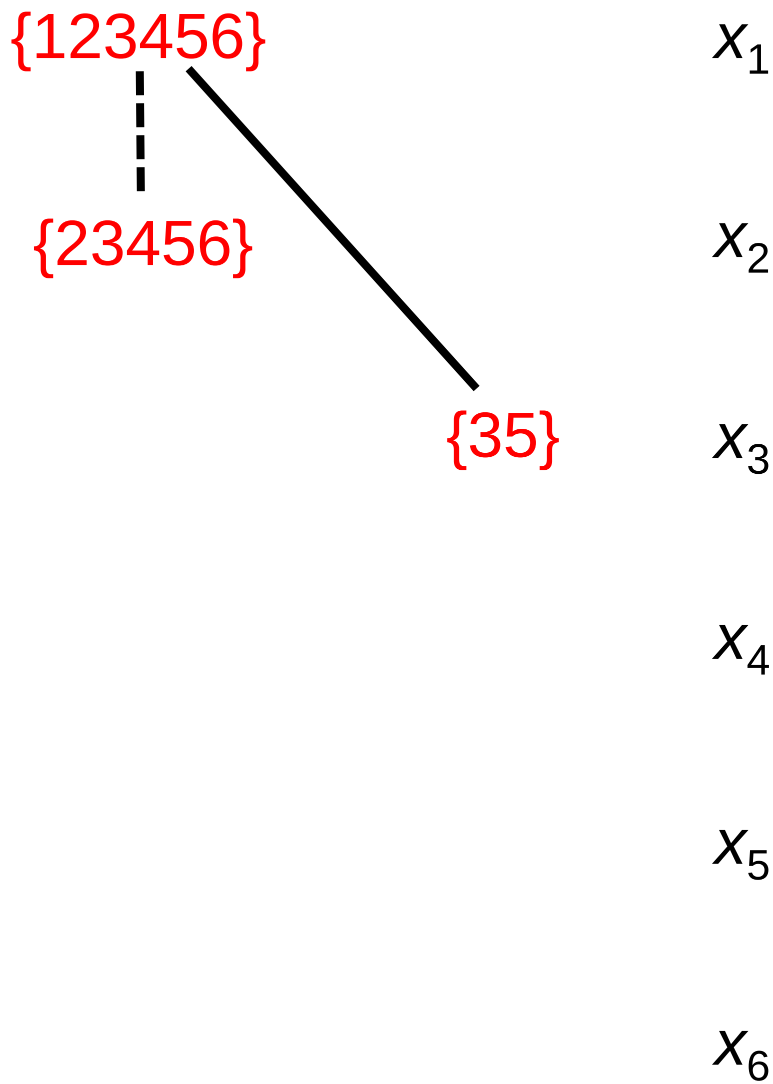
$x_4$

$x_5$

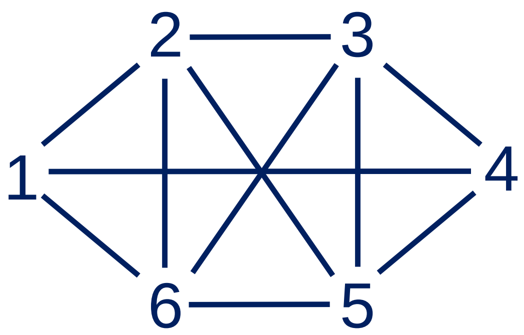
$x_6$

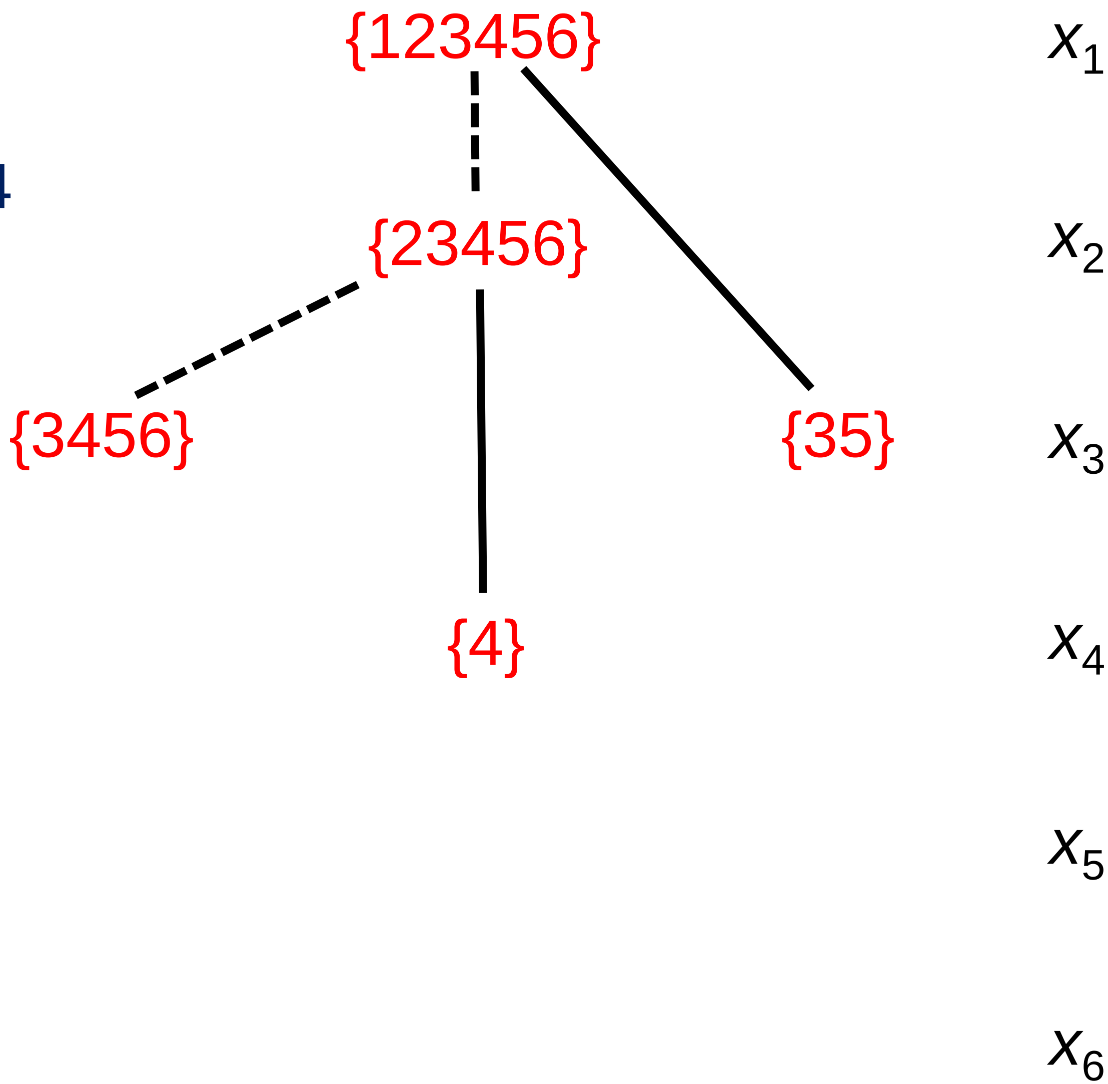
To build **relaxed** BDD, merge some additional nodes as we go along



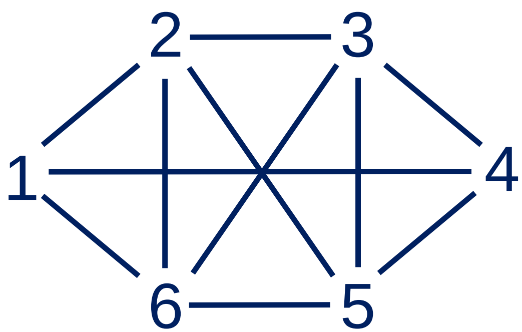



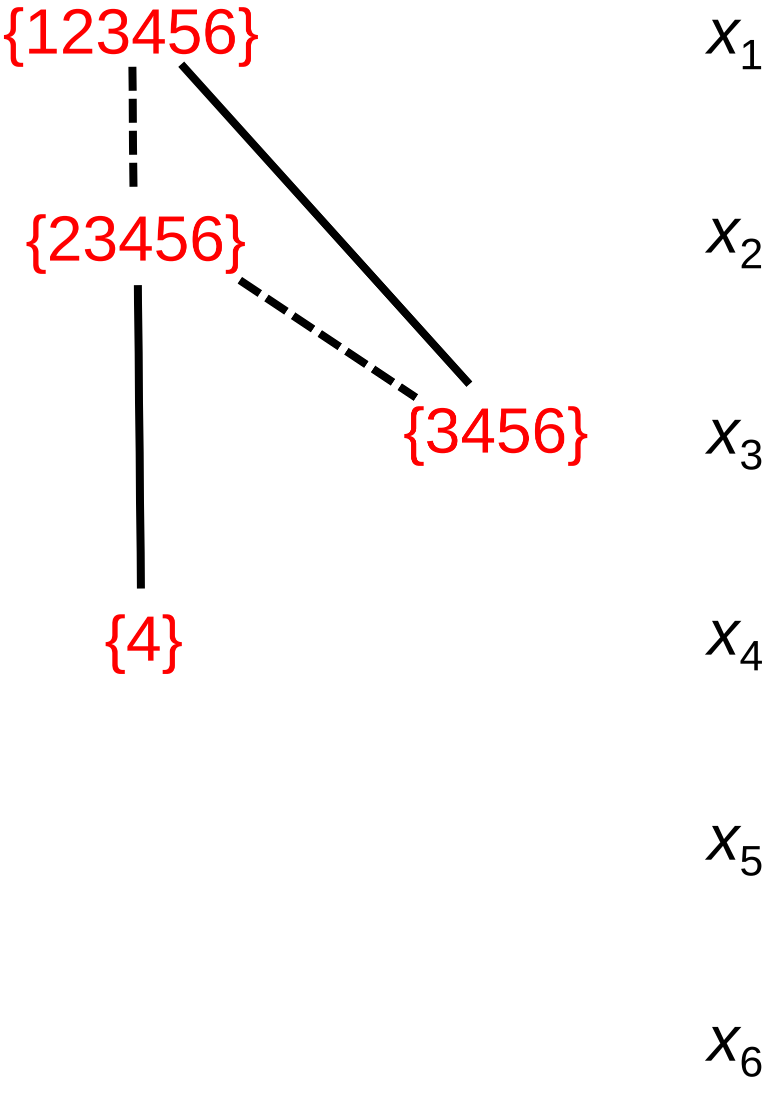

To build **relaxed** BDD, merge some additional nodes as we go along



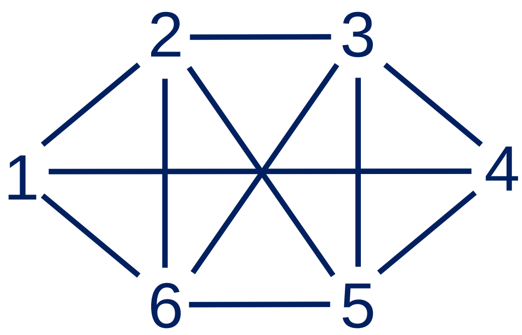



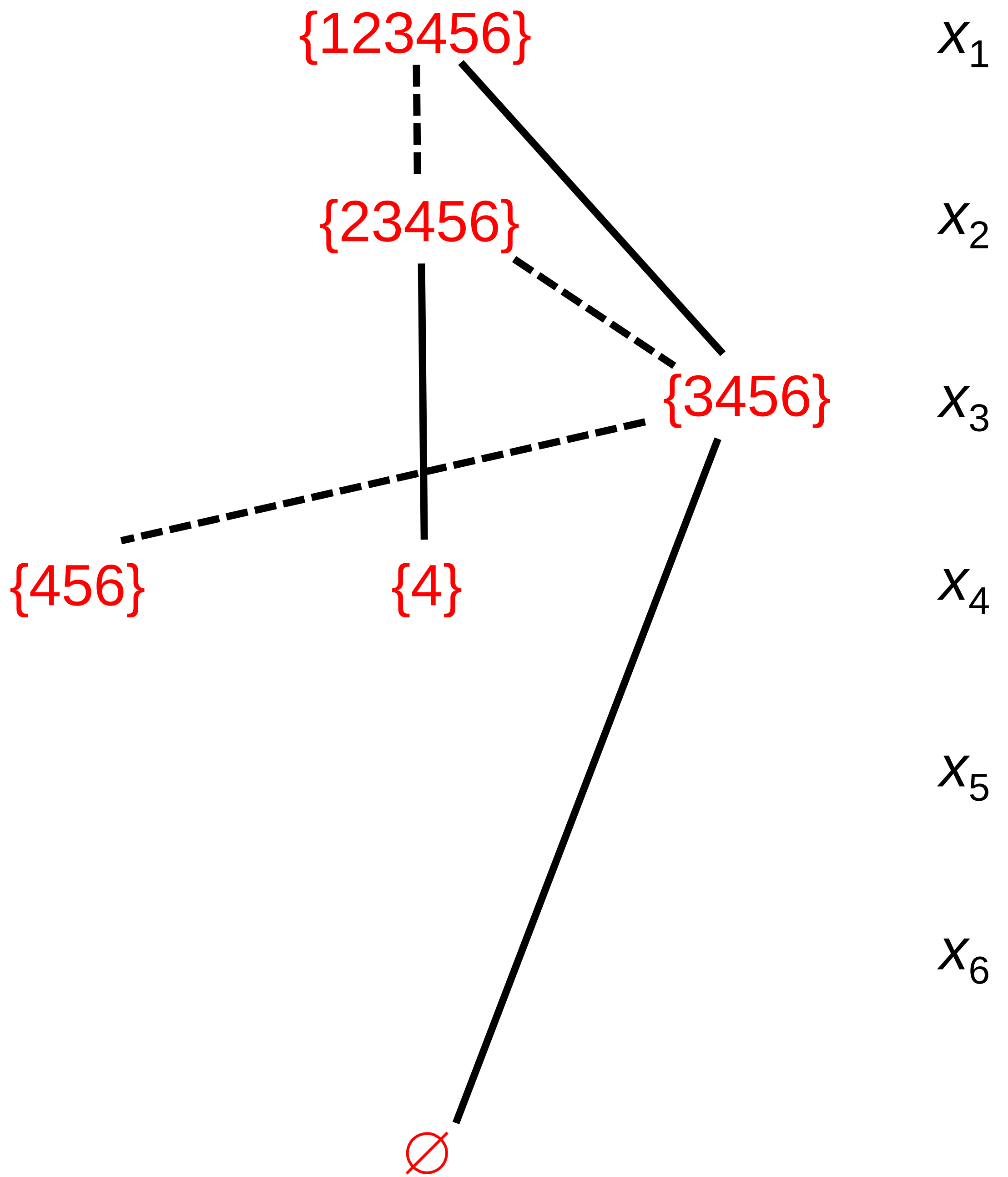

To build **relaxed** BDD, merge some additional nodes as we go along



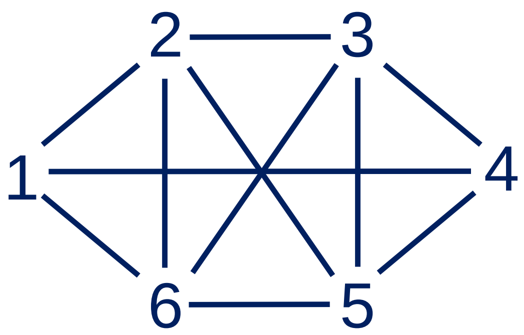




To build **relaxed** BDD, merge some additional nodes as we go along

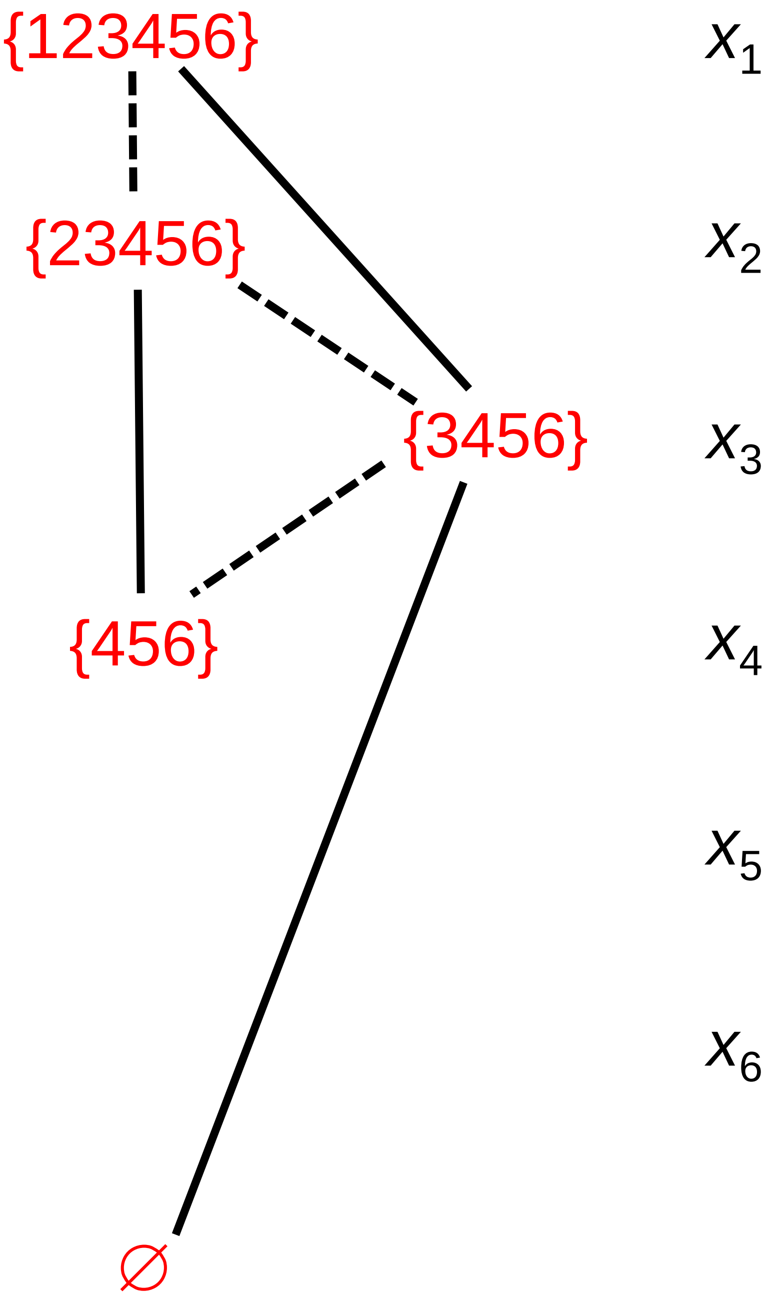


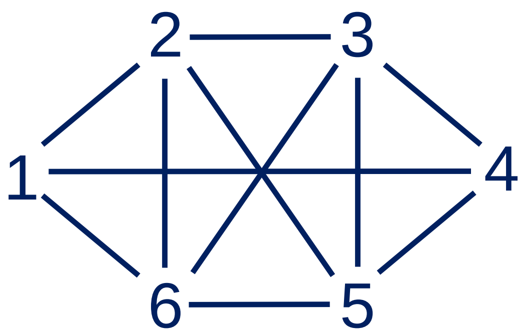




To build **relaxed** BDD, merge some additional nodes as we go along

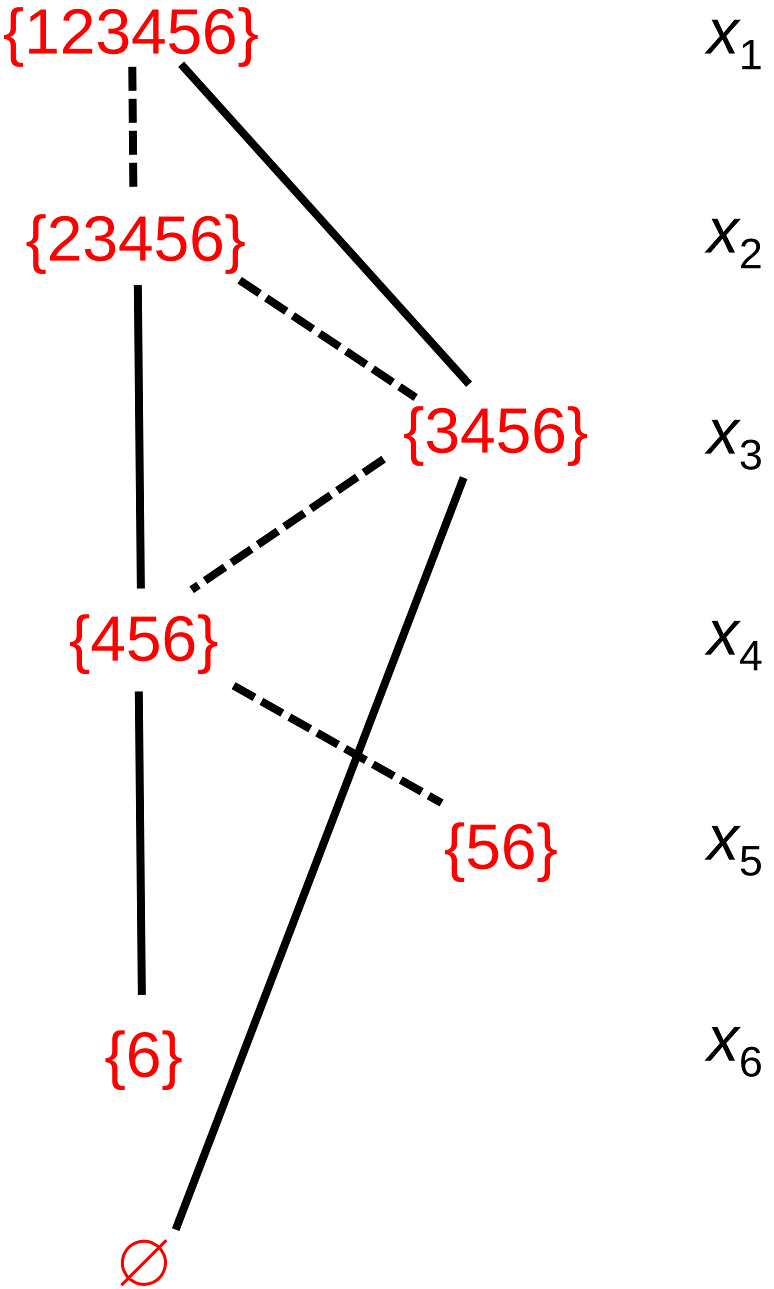


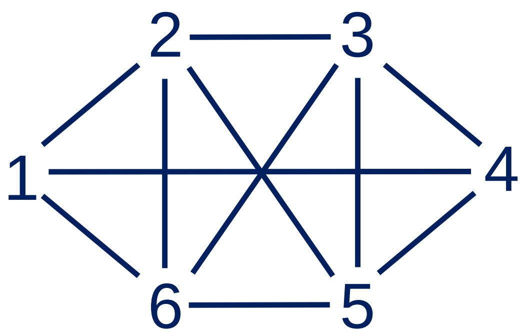

To build **relaxed** BDD, merge some additional nodes as we go along



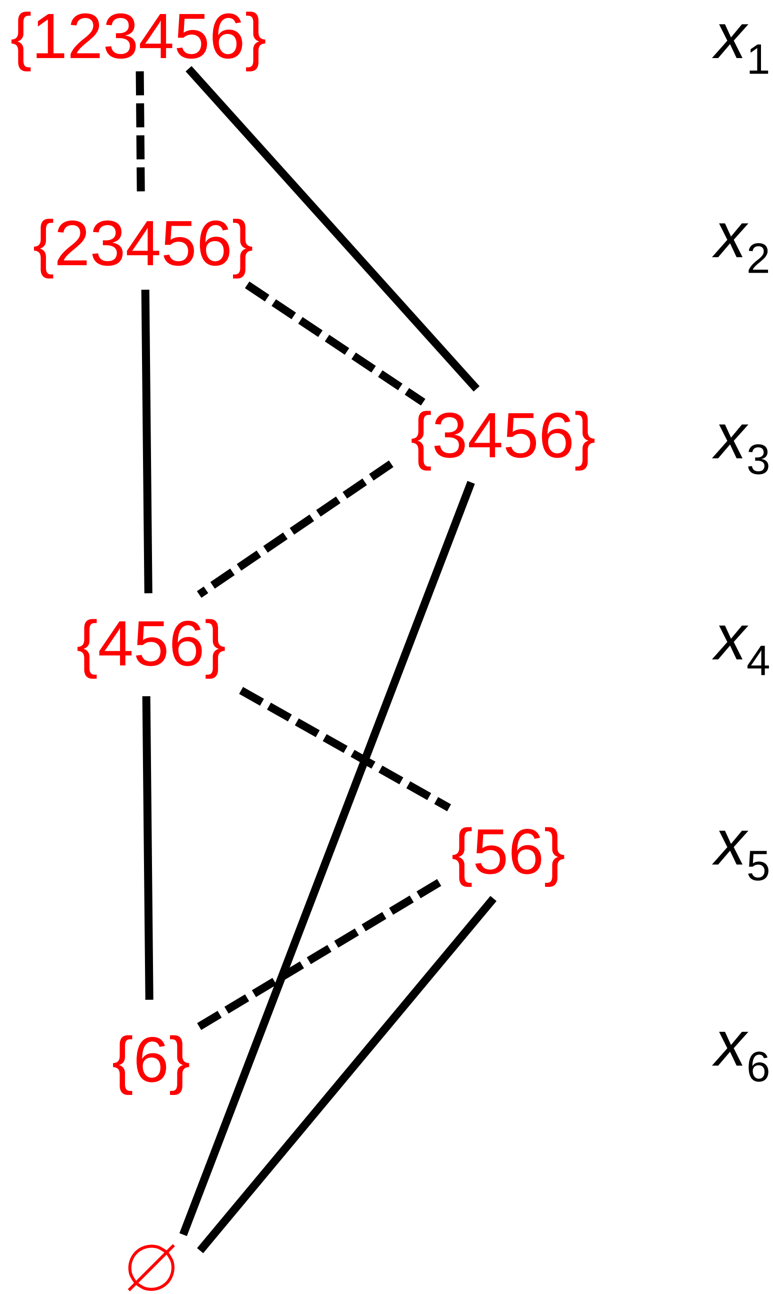


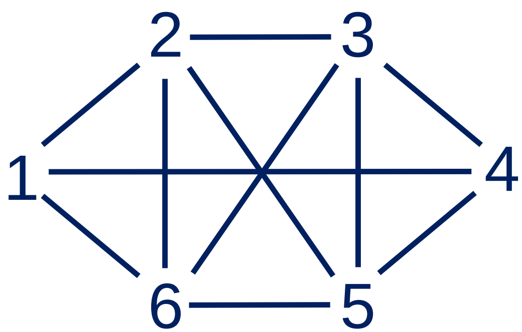

To build **relaxed** BDD, merge some additional nodes as we go along





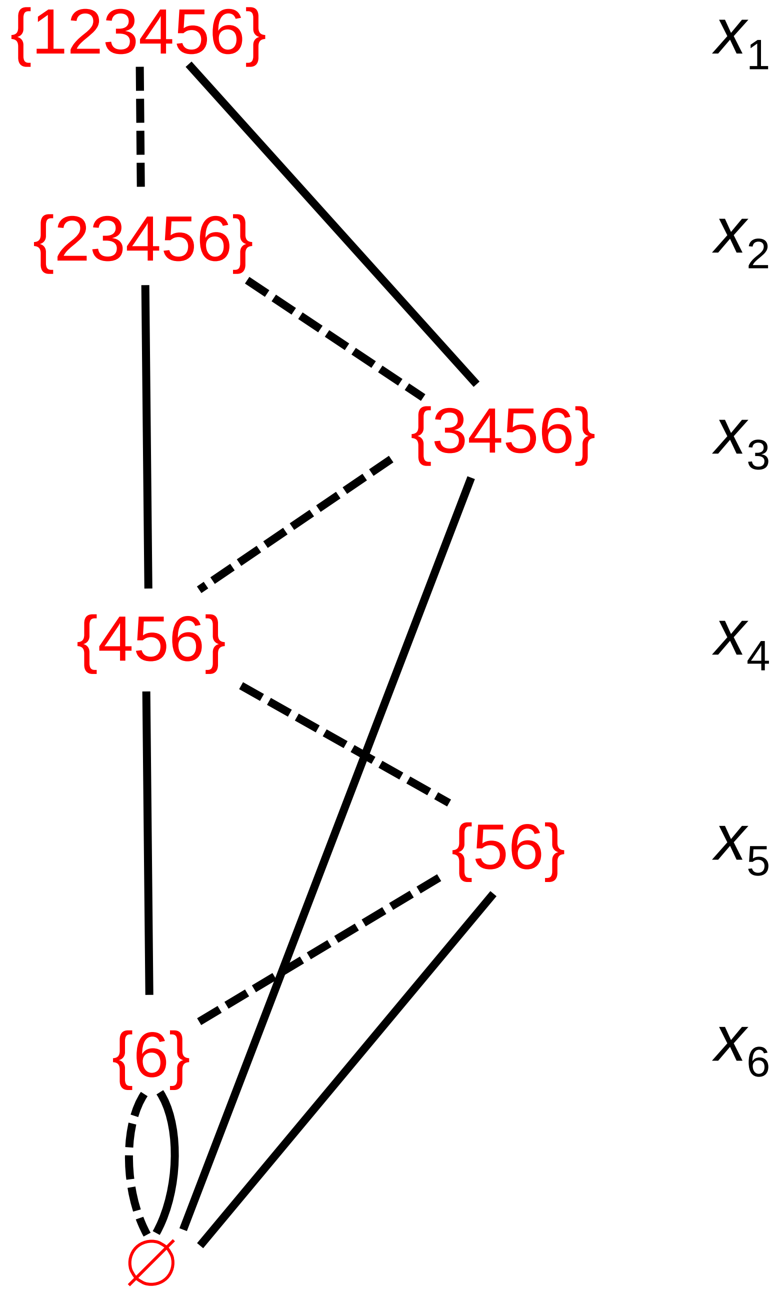

To build **relaxed** BDD, merge some additional nodes as we go along

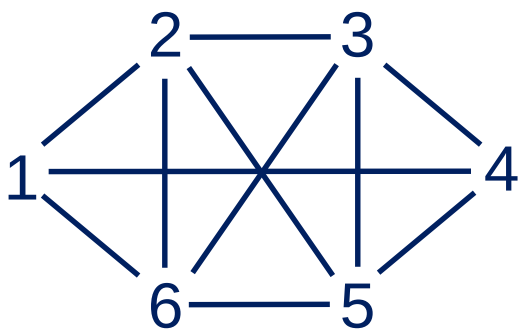





**Width** = 1

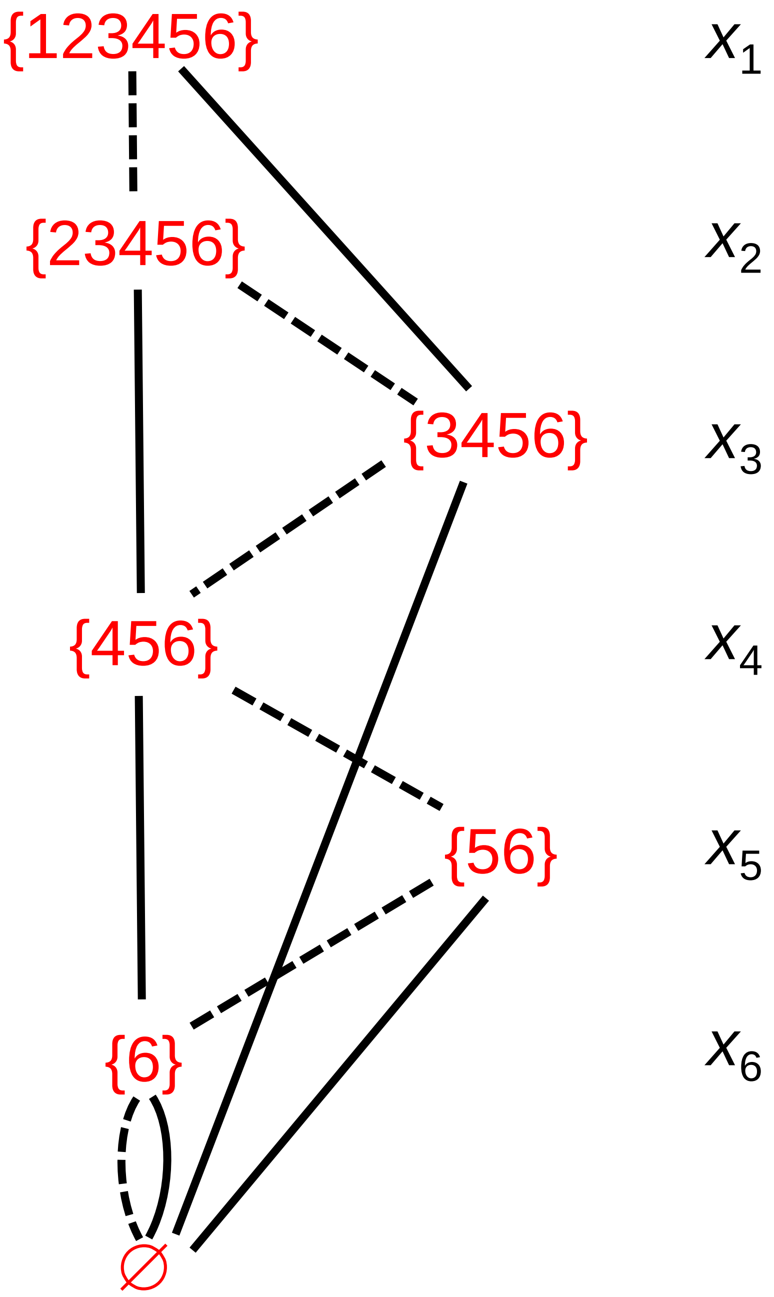
To build **relaxed** BDD, merge some additional nodes as we go along

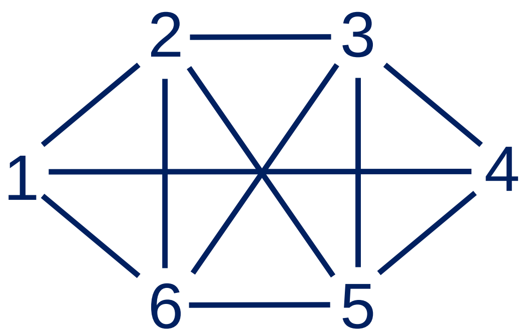




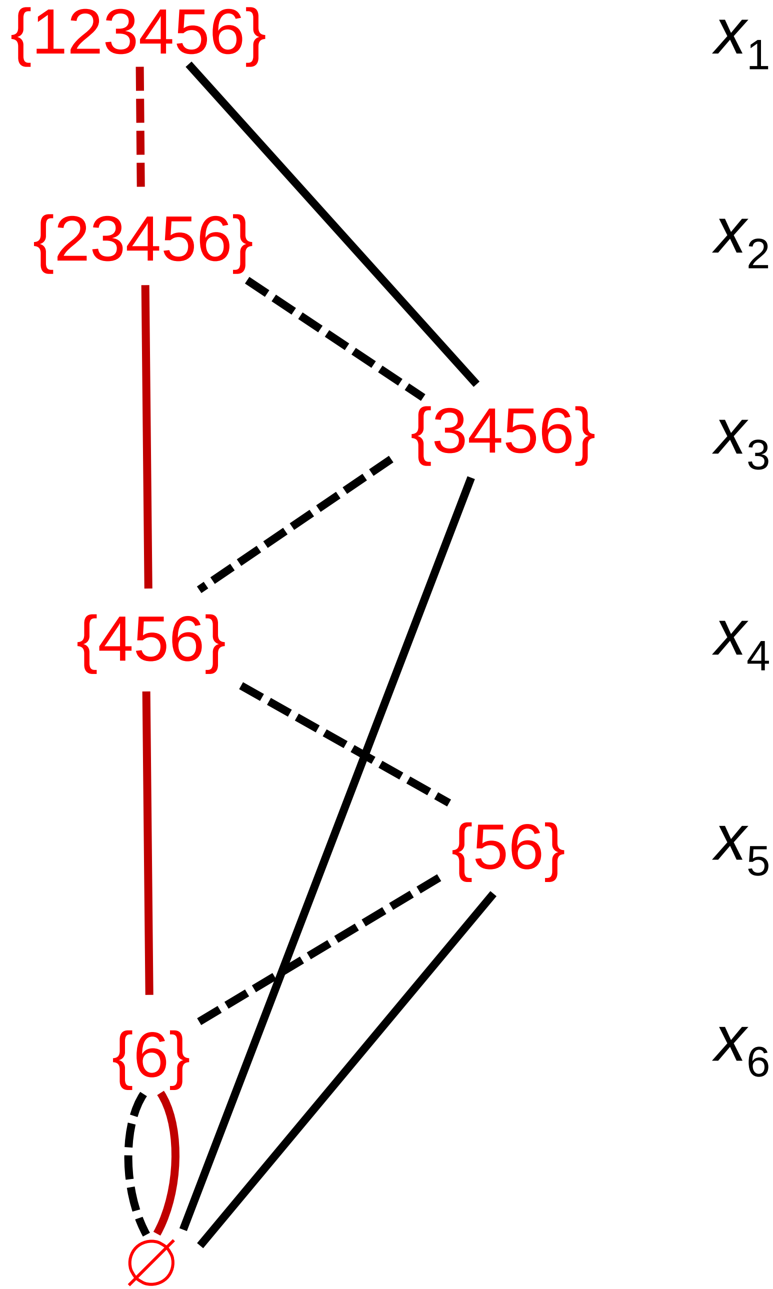

**Width** = 1

Represents 18 solutions, including 11 feasible solutions





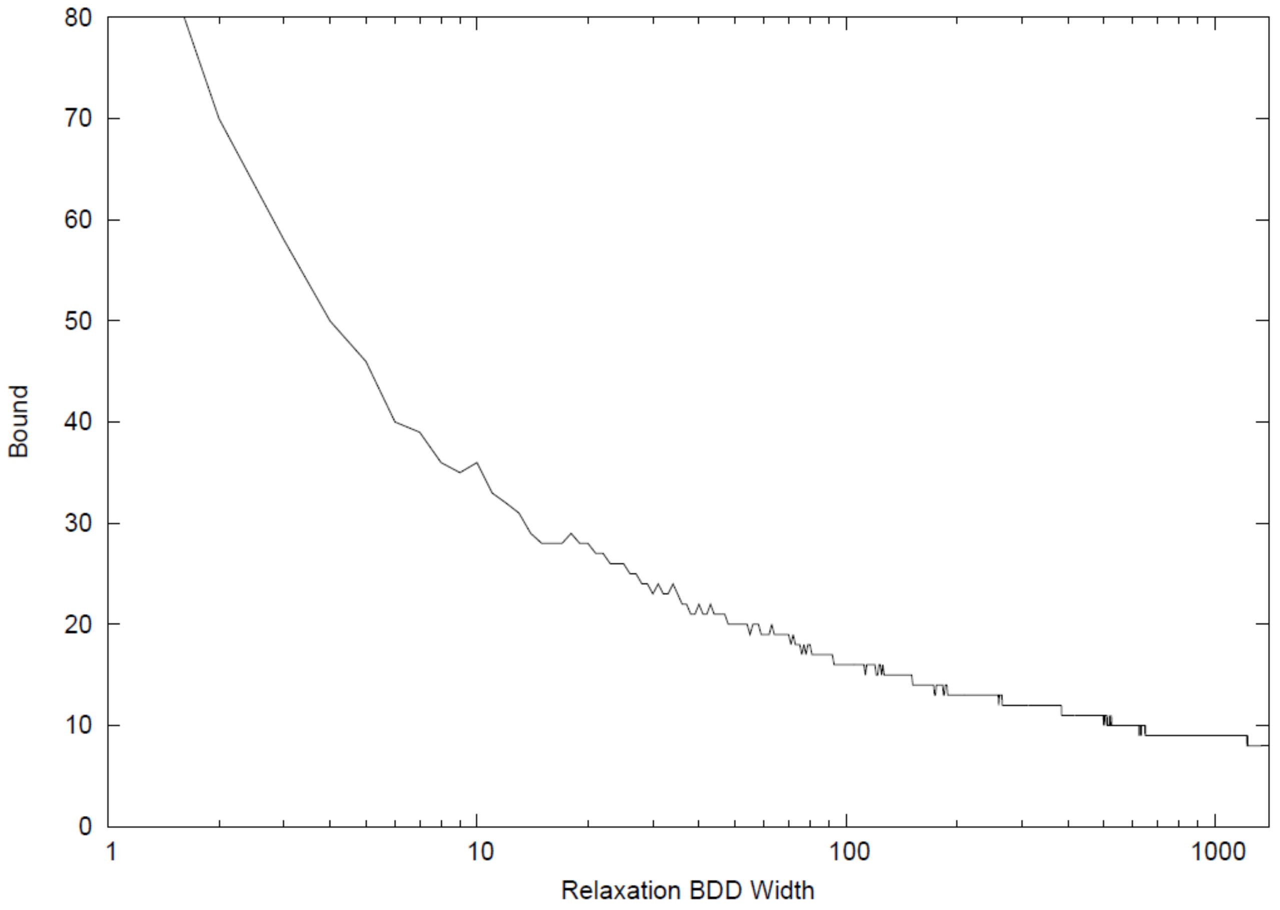

**Width** = 1




**Longest path**
gives bound
of 3 on optimal
value of 2

# Bound vs. Width

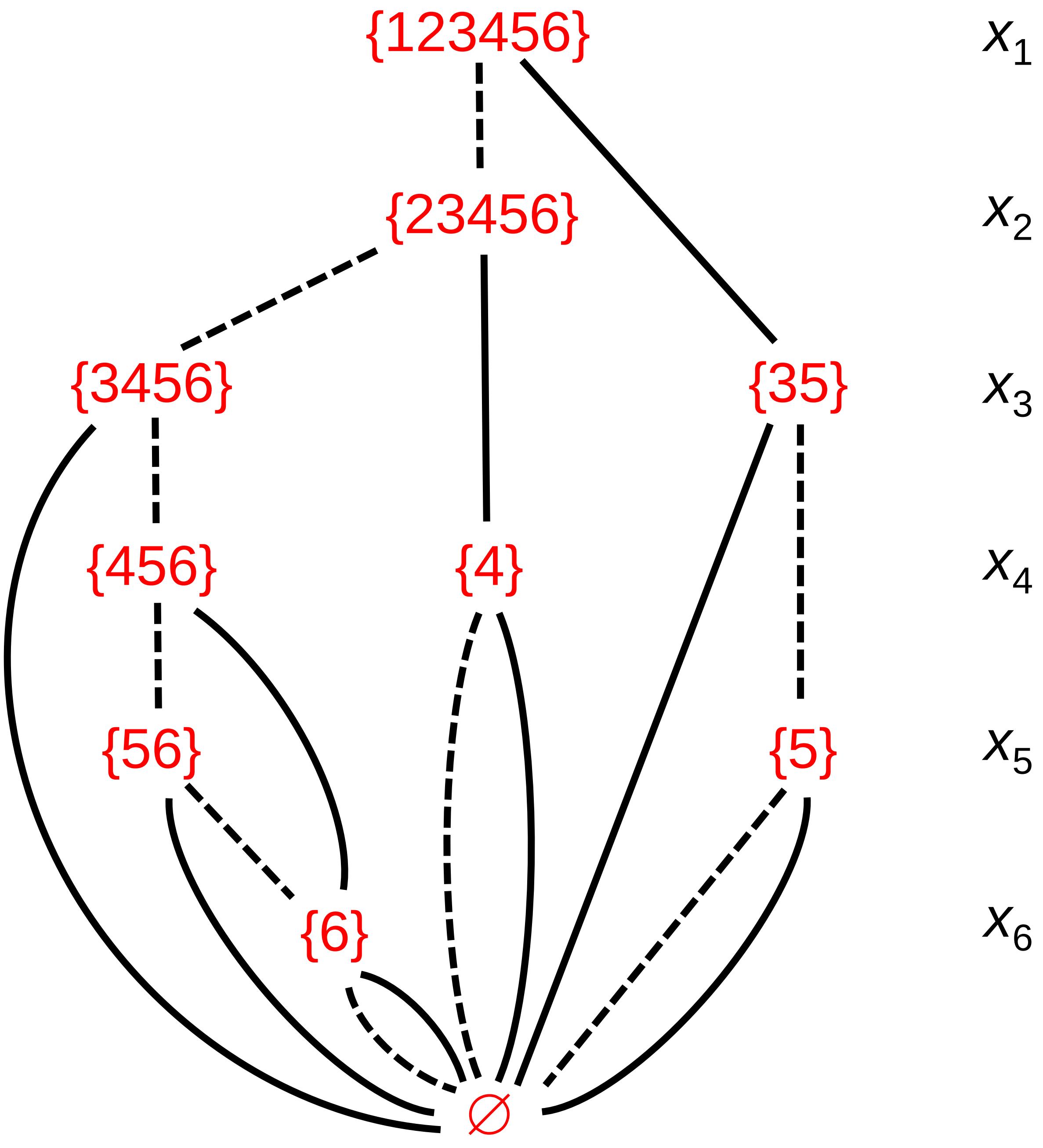
- Wider BDDs yield tighter bounds.
  - But take longer to build.



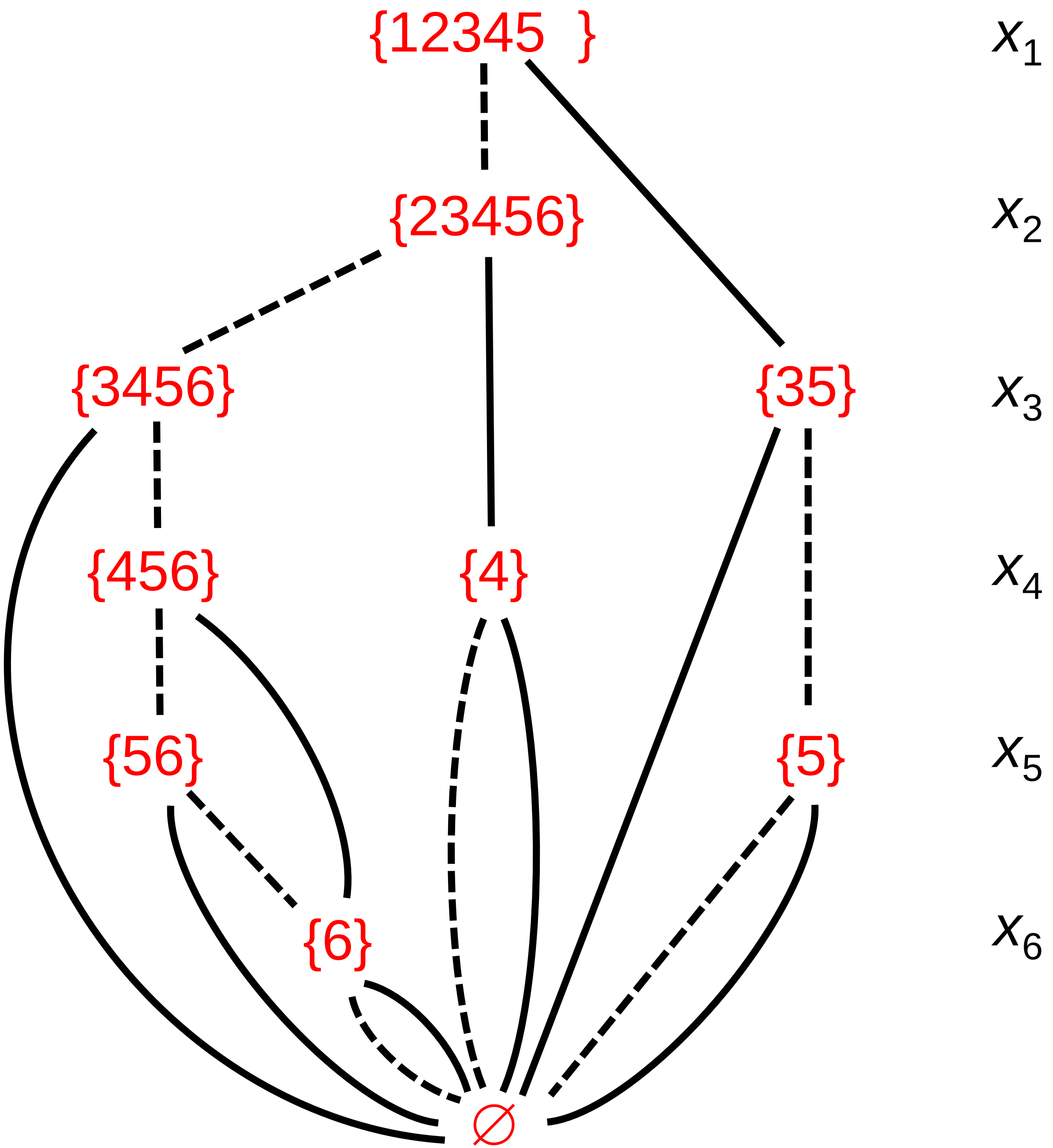
# Propagation

- We can propagate by removing arcs from the decision diagram.
  - Rather than removing elements from variable domains.
  - More effective than traditional domain filtering.
  - More information propagated from one constraint to the next.

Suppose this is the relaxed decision diagram

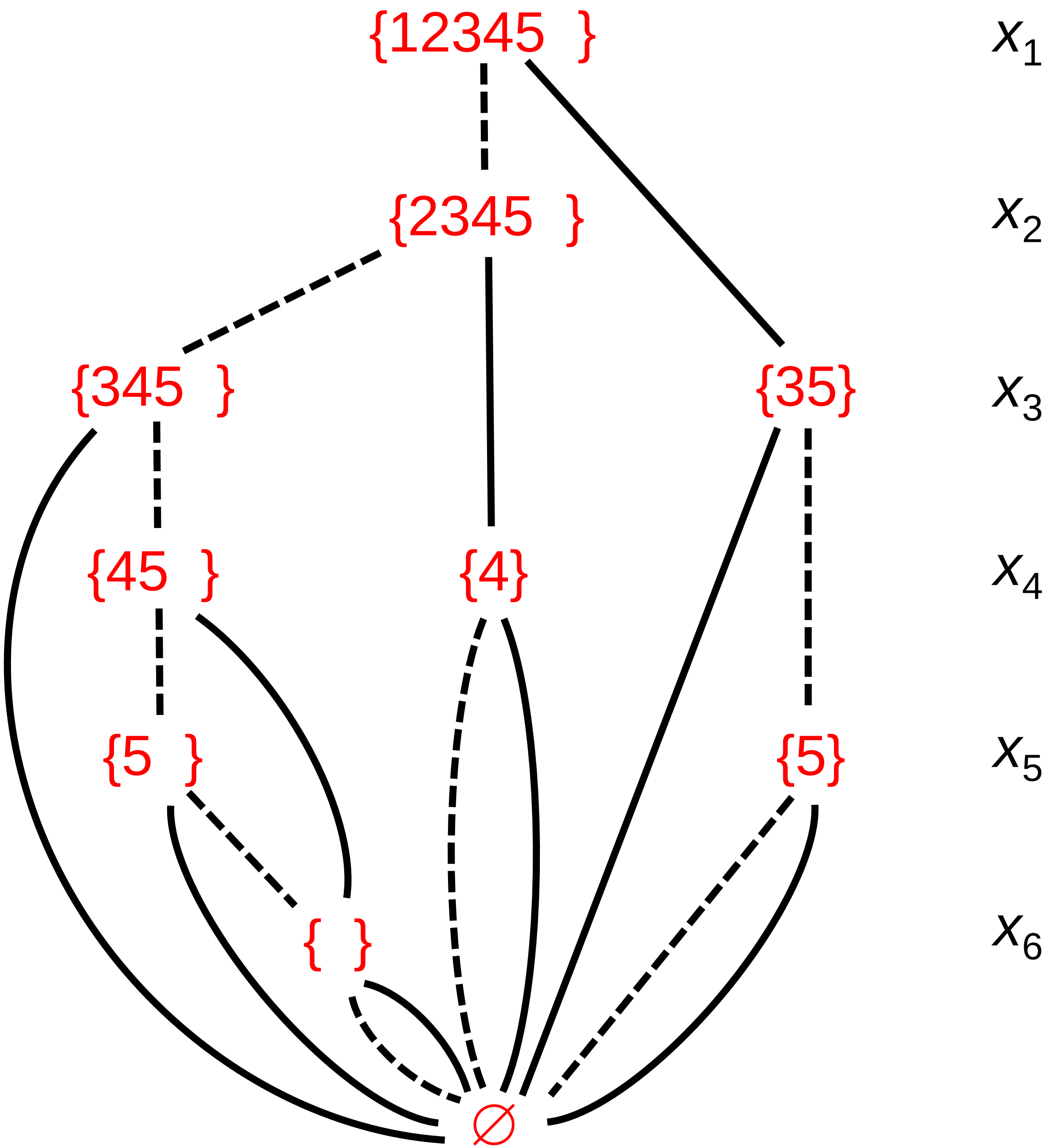


**Suppose this is the relaxed decision diagram**



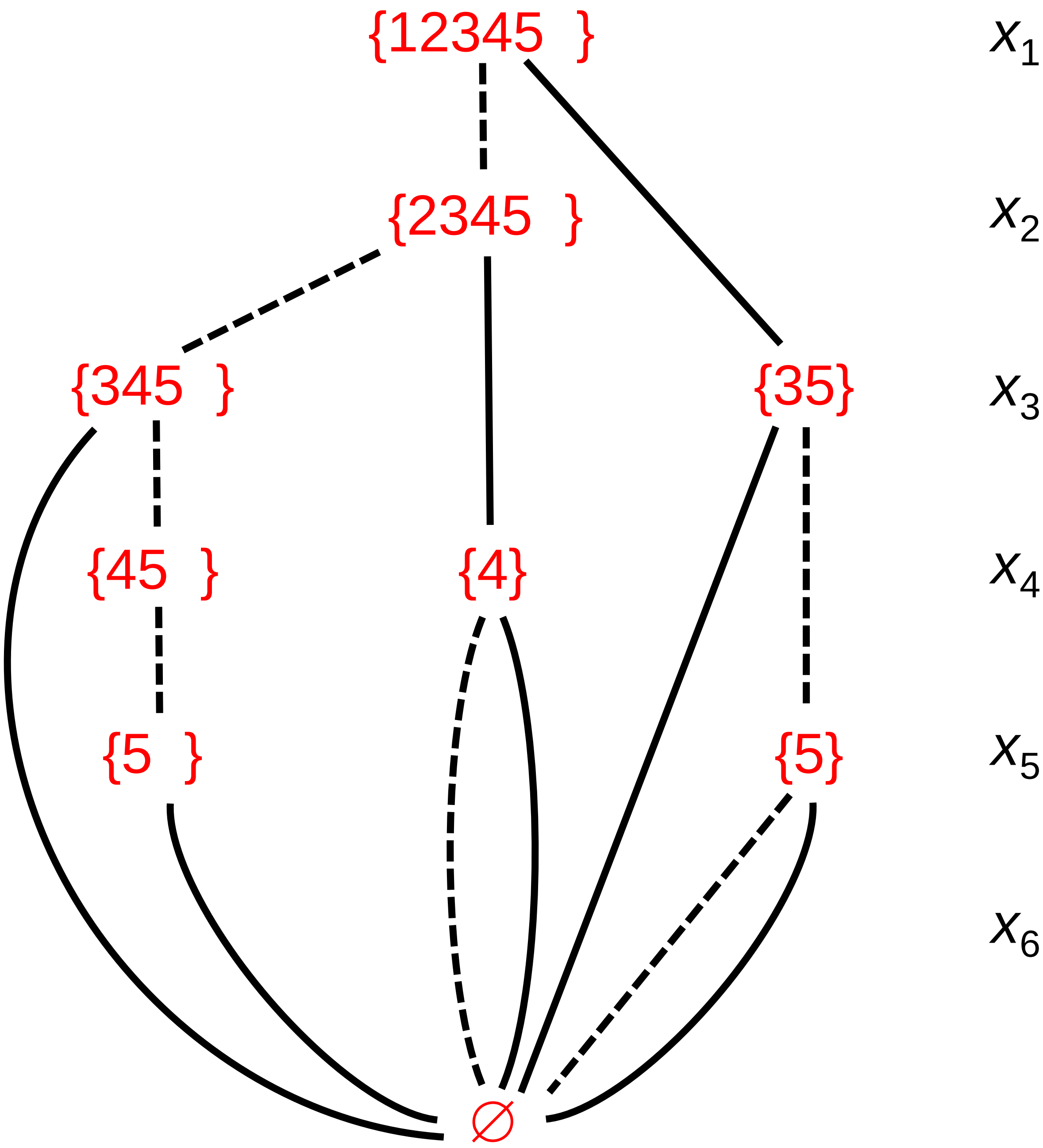

Suppose other constraints remove 6 from domain of $x_1$

Suppose this is the relaxed decision diagram




This propagates through the states and removes some arcs.

**Suppose this is the relaxed decision diagram**



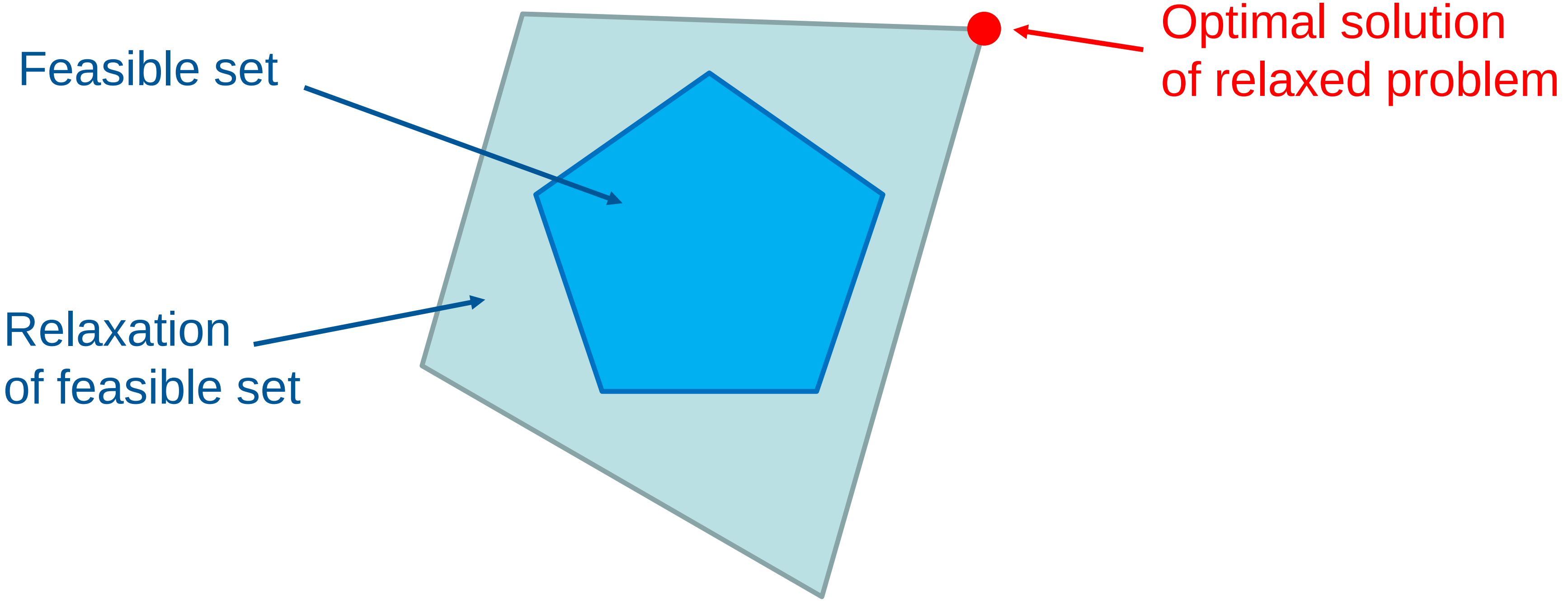

This propagates through the states and removes some arcs.

# Optimization with Decision Diagrams

- A relaxed decision diagram can provide framework for **branch-and-bound search**.
  - Bergman, Cire, van Hoeve, Hooker 2014
- Here, we introduce decision diagrams into **Benders methods**.
  - Must solve **separation problem** to implement Benders cuts**.**

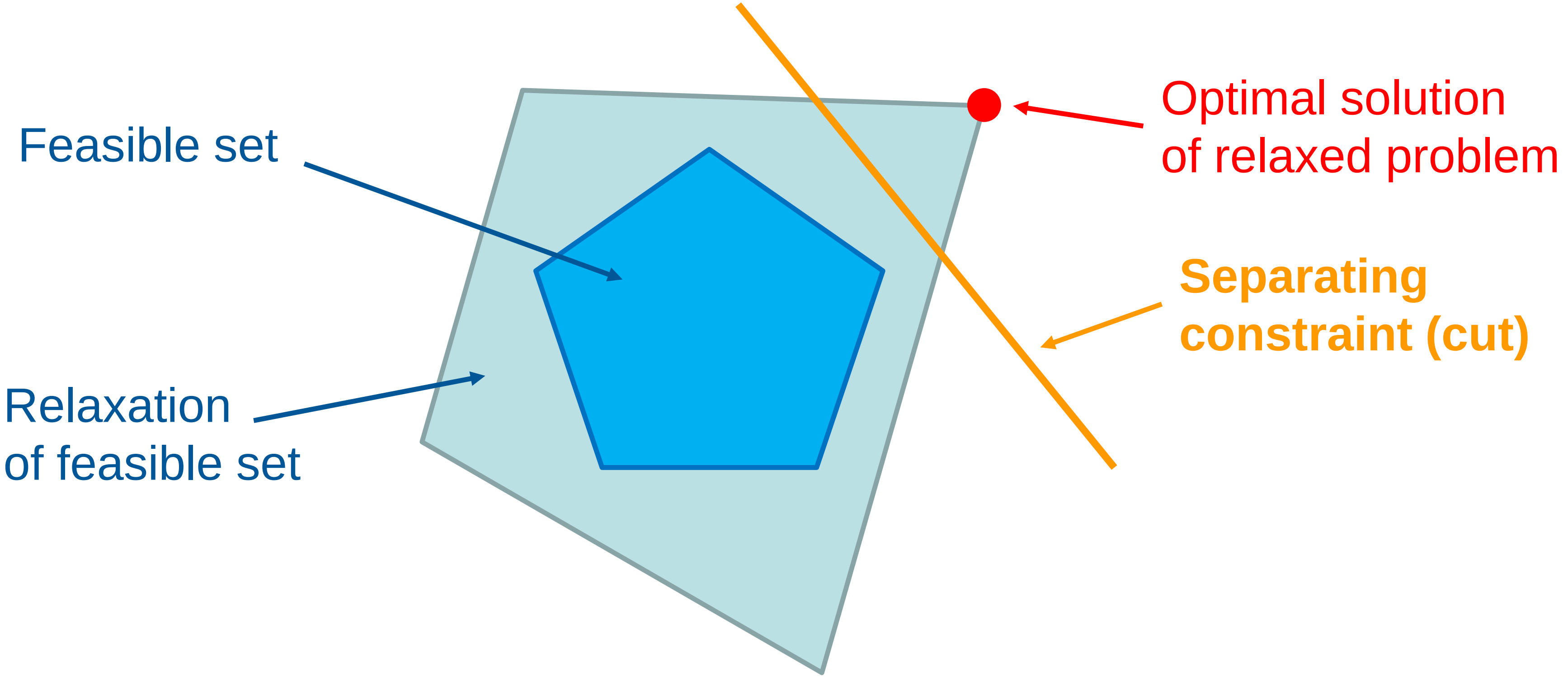
# Separation Problem in Optimization

- Given a relaxation of an optimization problem…
- Find a constraint that **separates** solution of the relaxation from the feasible set



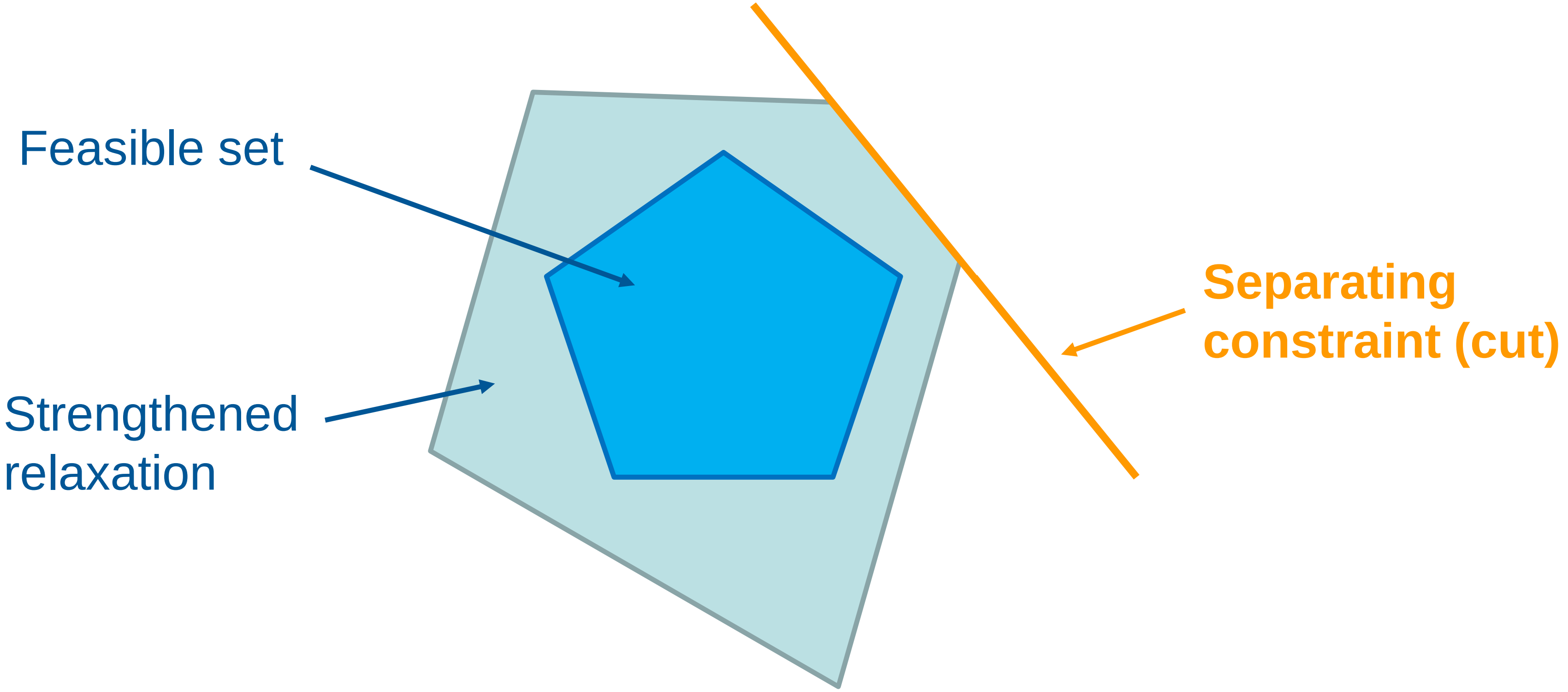
# Separation Problem in Optimization

- Given a relaxation of an optimization problem…
- Find a constraint that **separates** solution of the relaxation from the feasible set



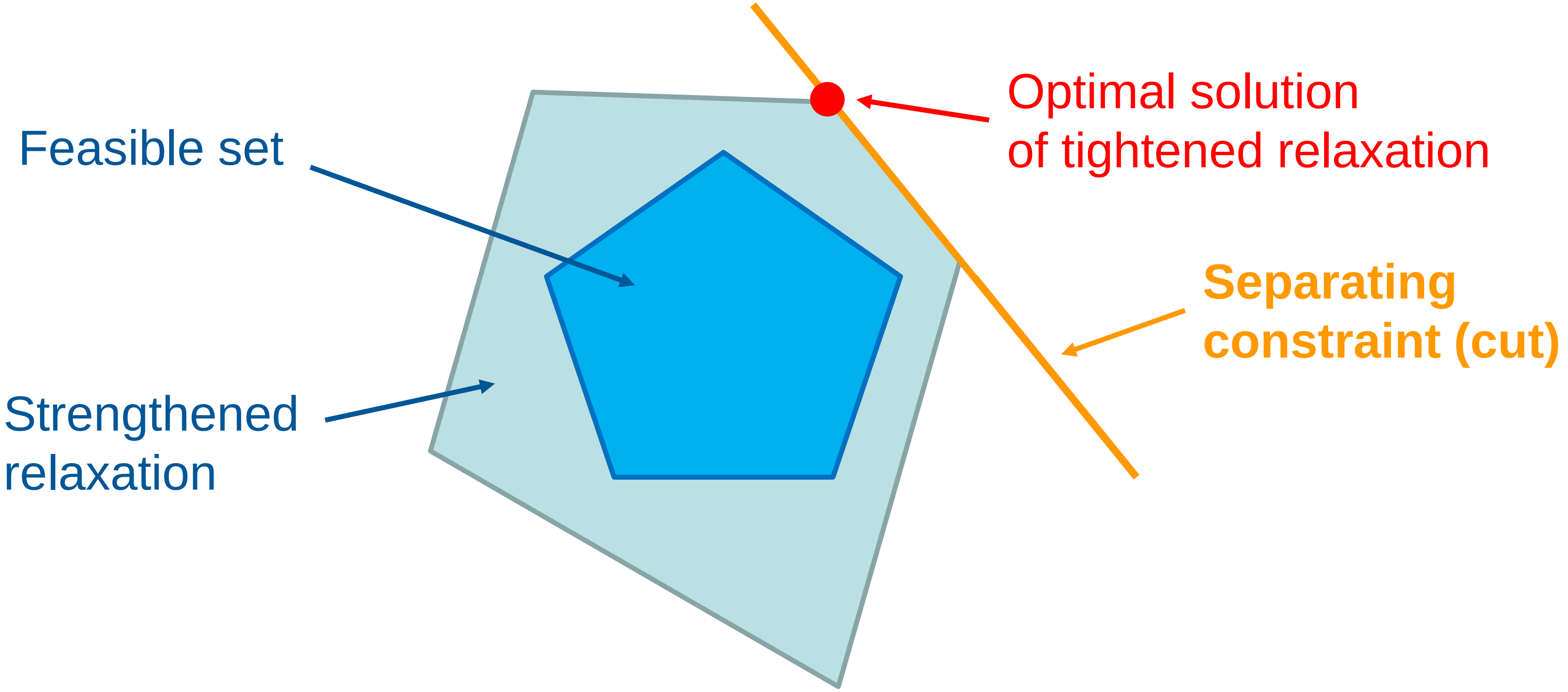
# Separation Problem in Optimization

- Now **strengthen** the relaxation with the separating cut.
  - Cuts are usually linear inequalities.



# Separation Problem in Optimization

- Now **strengthen** the relaxation with the separating cut.
  - Cuts are usually linear inequalities.
  - Re-solve relaxation and repeat.



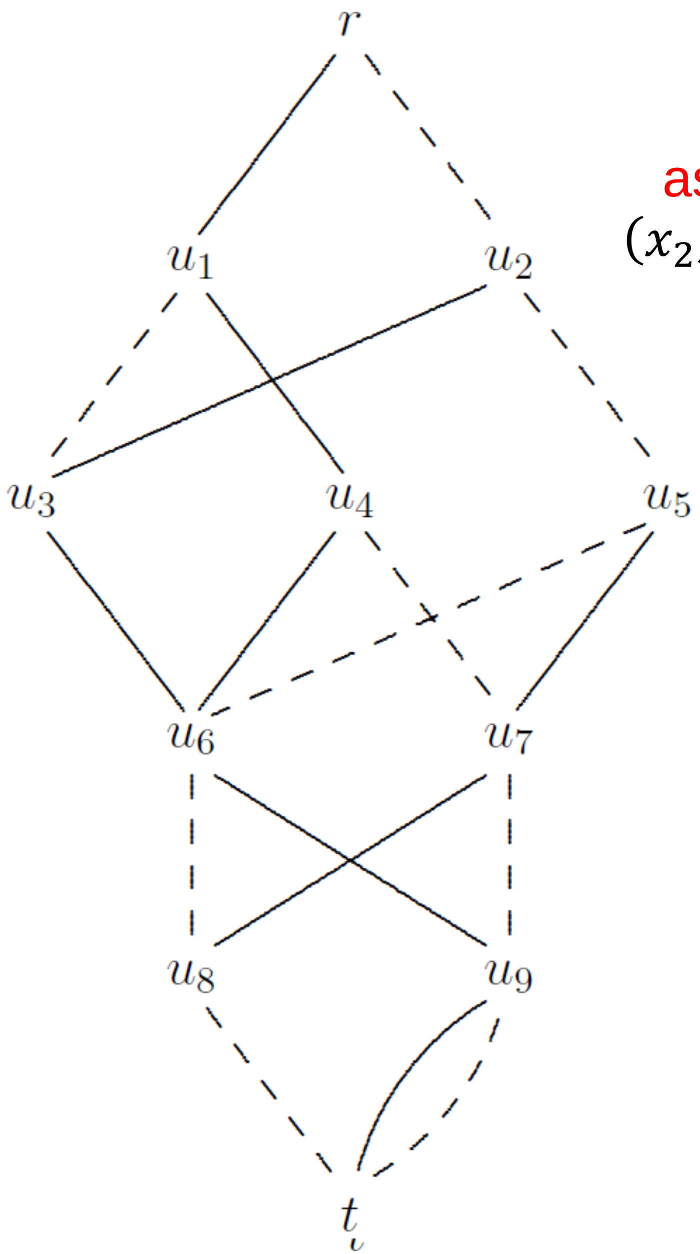
# Separation Problem in Optimization

- Separation is a **workhorse** in integer and nonlinear programming.
  - **Gomory** cuts
  - **Mixed integer rounding** cuts
  - Separating **knapsack** cuts
  - Separating **cover** inequalities
  - Separating cuts in special families
    - Subtour elimination, combs for TSP
    - Separating flow cuts for fixed-charge network flow
    - etc. (**huge** literature)

# Separation Problem for Decision Diagrams

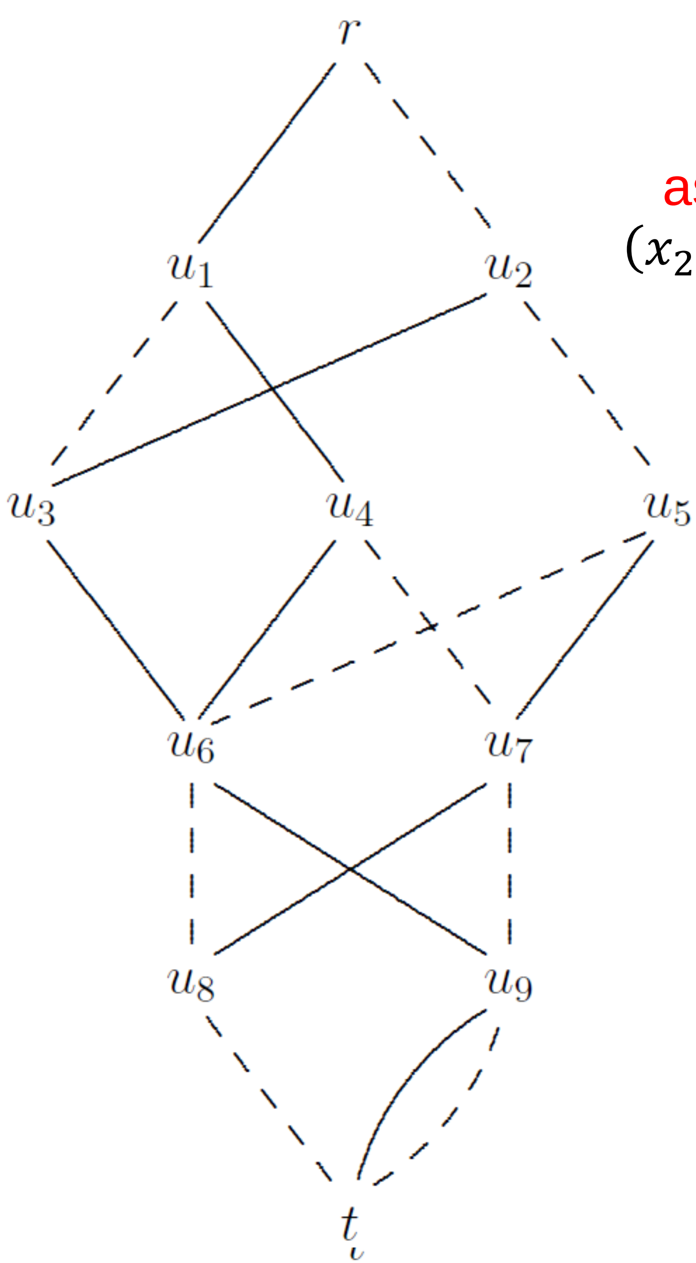
- Exclude a given partial assignment $x_i = \bar{x}_i$ for $i \in I$.
  - That is, remove all paths in which $x_i = \bar{x}_i$ for $i \in I$.
- Example…

Original Diagram



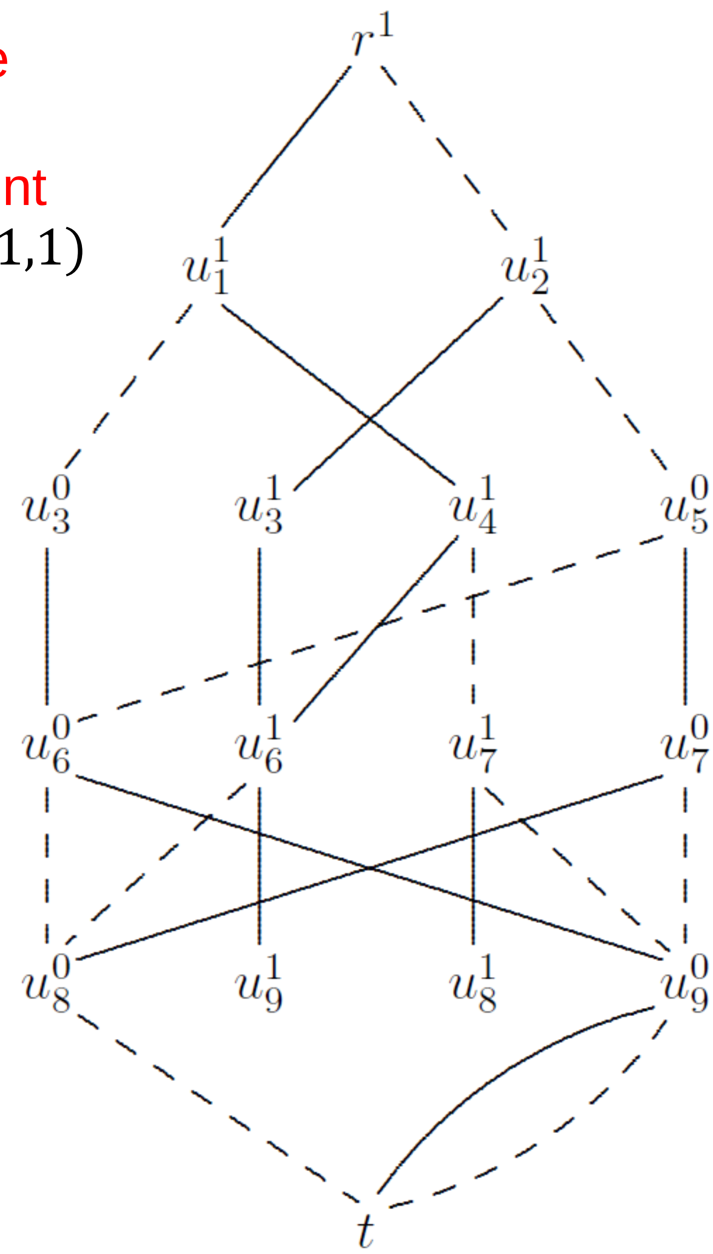

Remove partial assignment $(x_2, x_4) = (1,1)$

Original Diagram




Remove partial assignment $(x_2, x_4) = (1,1)$

Separating Diagram



# Separation Algorithm

- In principle, a partial assignment can be separated by conjoining two BDDs.

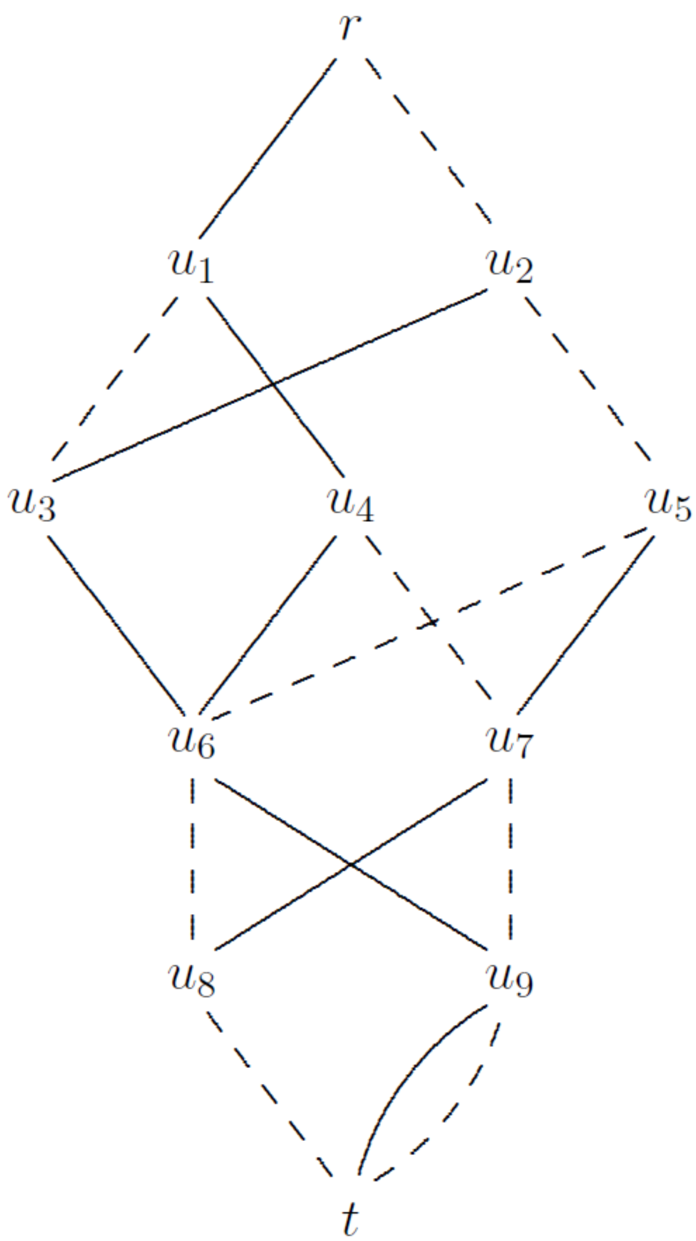
| | | |
|---|---|---|
| **Original Diagram** | $\wedge$ | **Width-2 diagram** representing negation of partial assignment |

  - However, this introduces an unnecessary data structure.

# Separation Algorithm

- We will propose an algorithm specifically for separation.
  - Exposes **essential logic** of separation.
  - Operates on **original data structure**.
  - Allows proof of **tighter bounds** on growth of the separating diagram as cuts are added.

## Original BDD



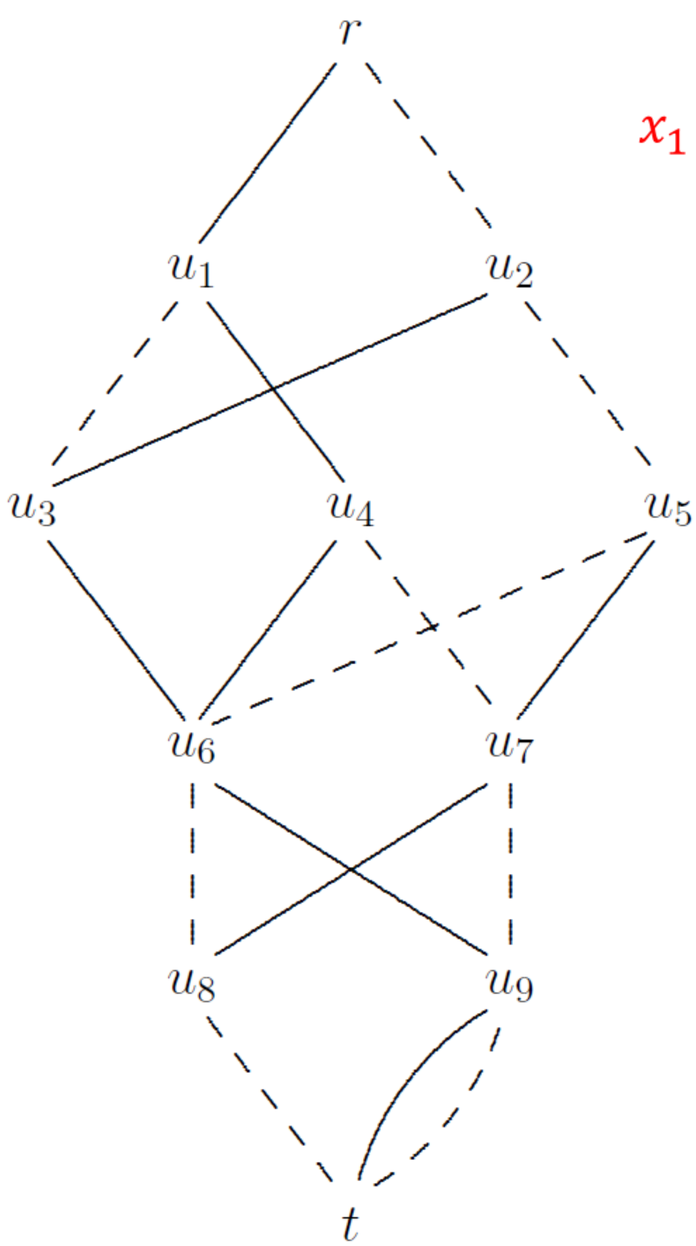

## Separating BDD


A node has **state 1** when all incoming paths are excluded.

Otherwise **state 0**.

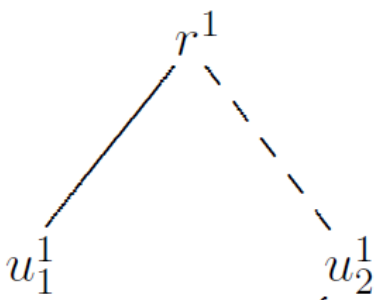
Assign state 1 to root node.

Original BDD




$x_1$ unrestricted

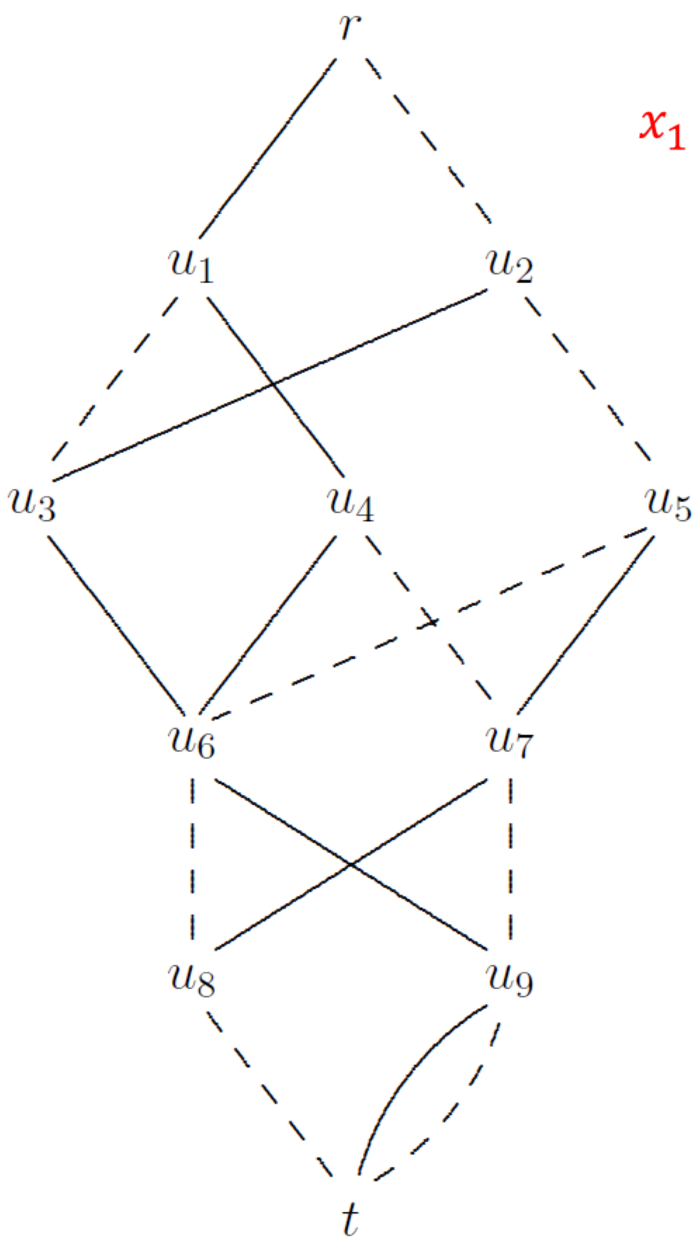
Separating BDD




Duplicate arcs leaving *r* in original BDD.

Child nodes inherit state of parent node.

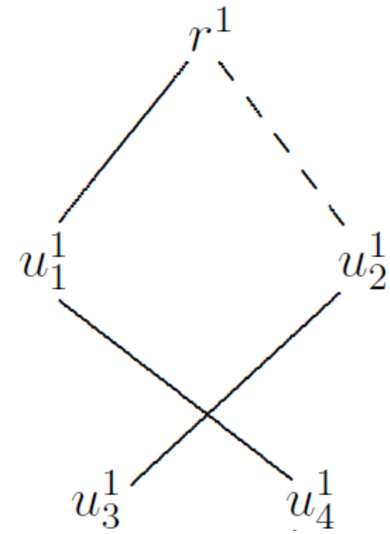
Original BDD




$x_1$ unrestricted

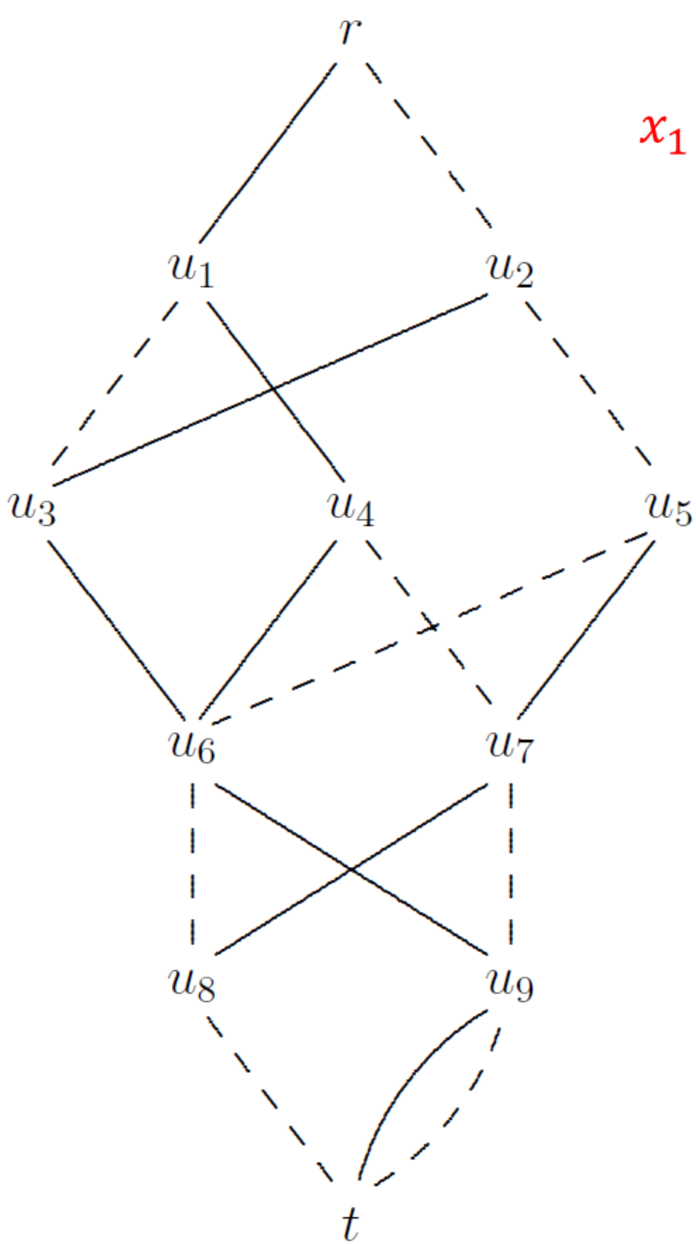
$x_2 \neq 1$

Separating BDD




1-arcs from state 1 nodes
preserve state 1

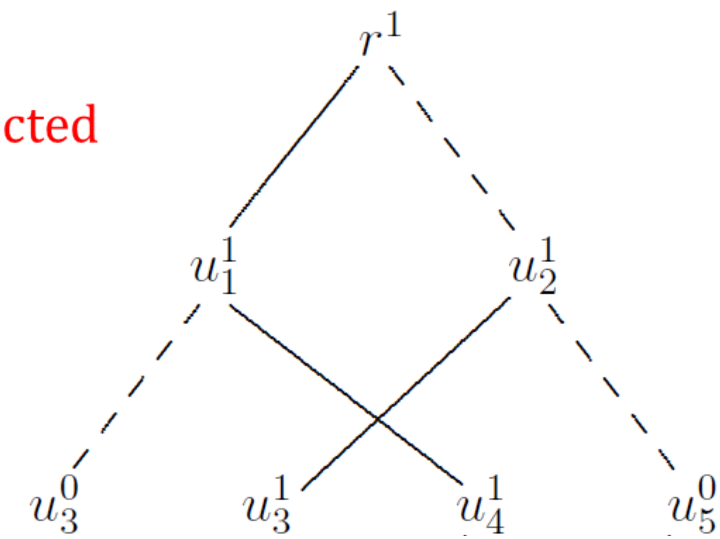
## Original BDD




$x_1$ unrestricted

$x_2 \neq 1$

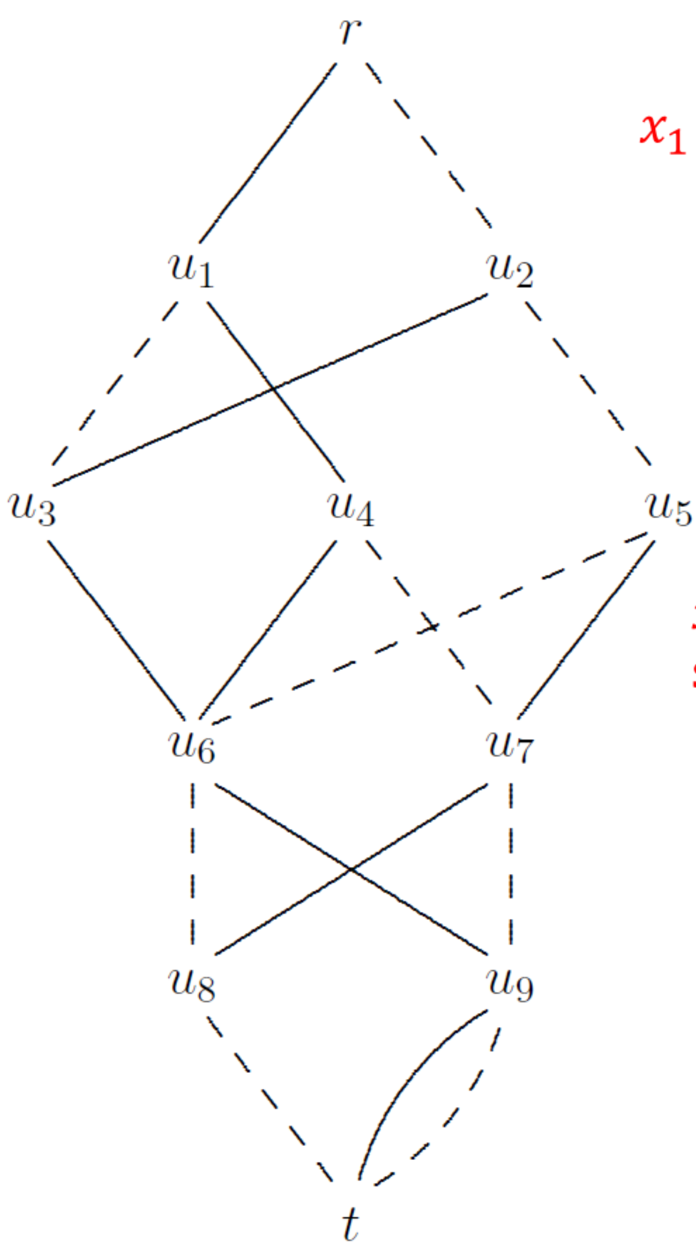
## Separating BDD




1-arcs from state 1 nodes preserve state 1

0-arcs from state 1 nodes switch to state 0

Original BDD

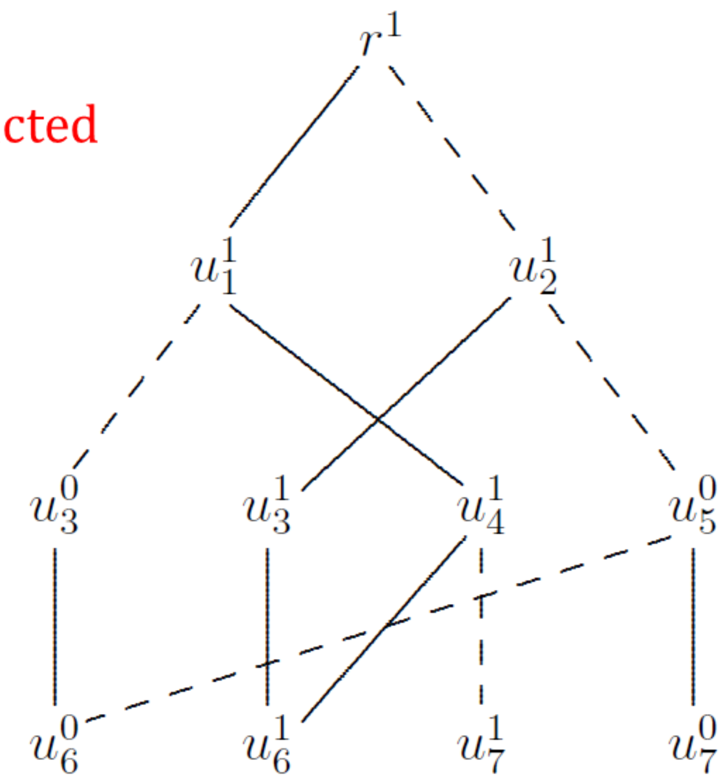



$x_1$ unrestricted

$x_2 \neq 1$

$x_3$ unre-
stricted

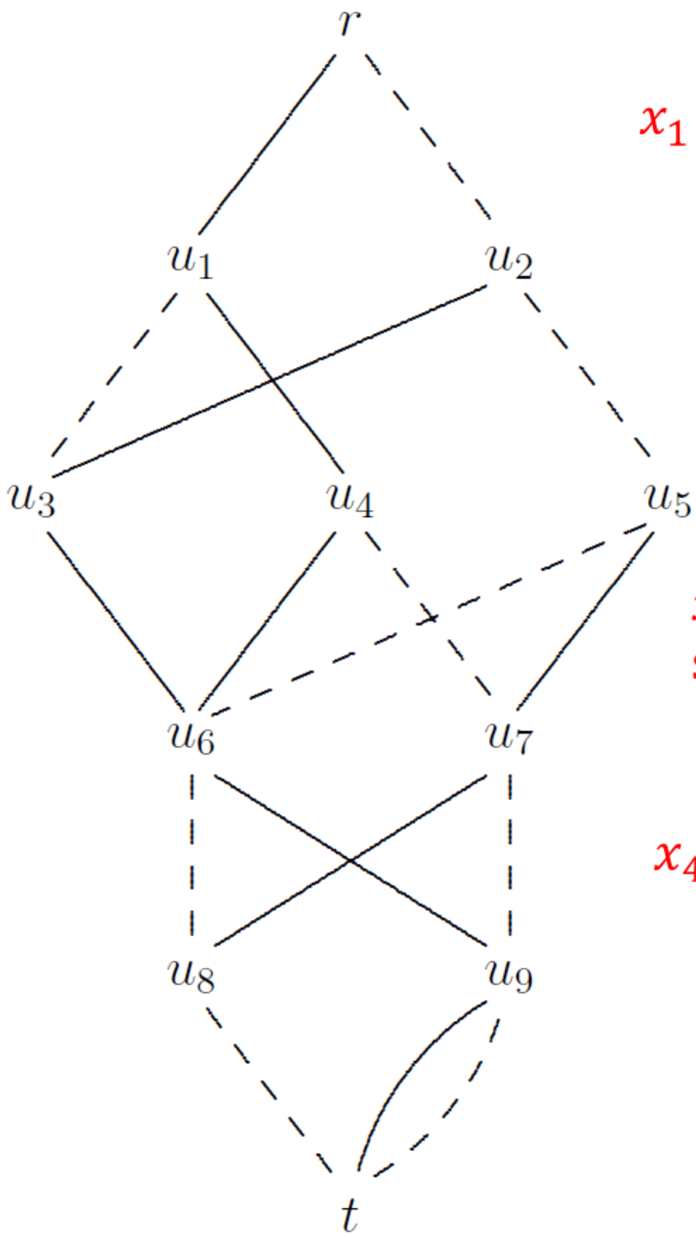
Separating BDD




Duplicate arcs in original BDD.

Child nodes inherit state of parent node.

Original BDD



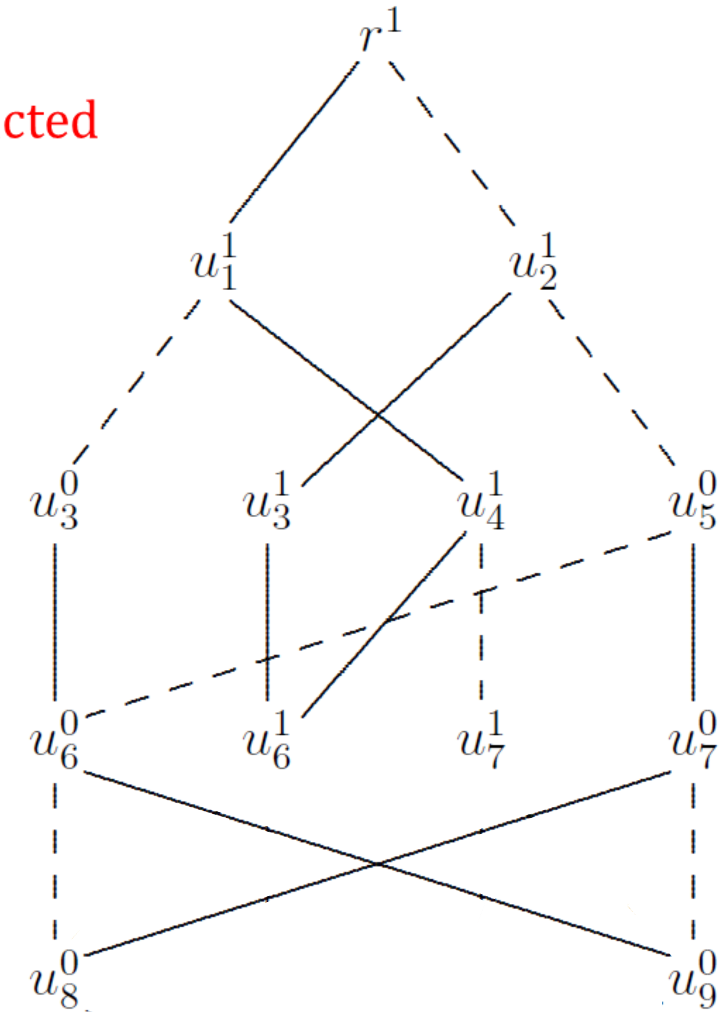

$x_1$ unrestricted

$x_2 \neq 1$

$x_3$ unre-stricted

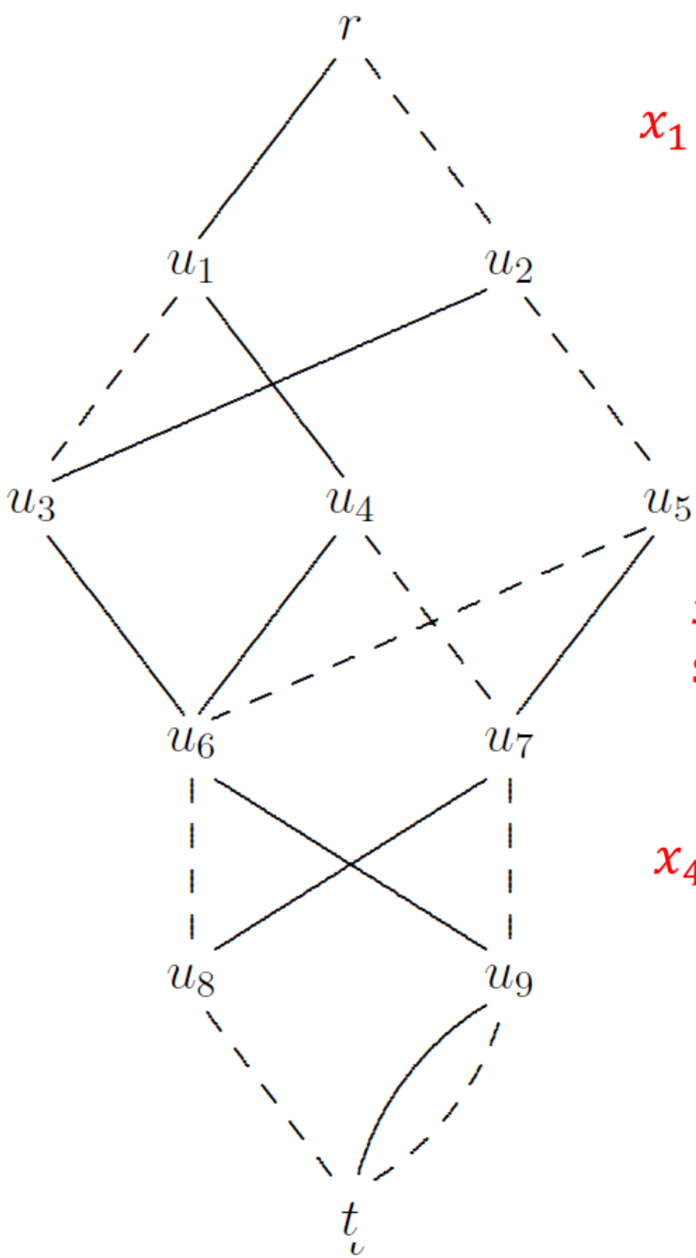
$x_4 \neq 1$

Separating BDD

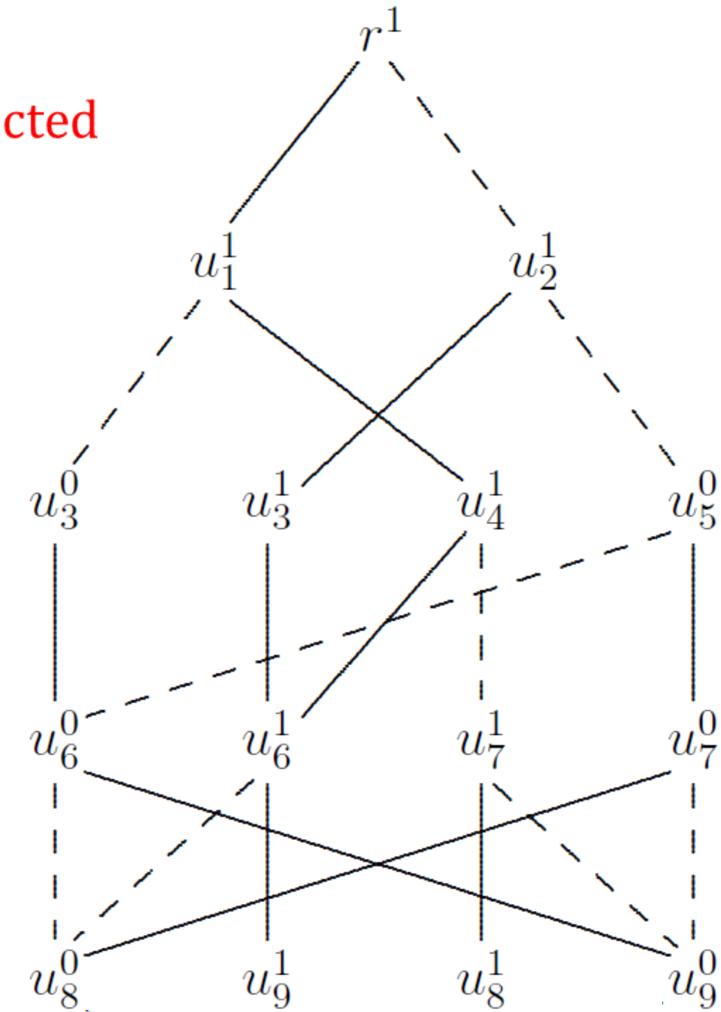



Duplicate arcs from nodes with state 0, preserving state.

Original BDD




Separating BDD



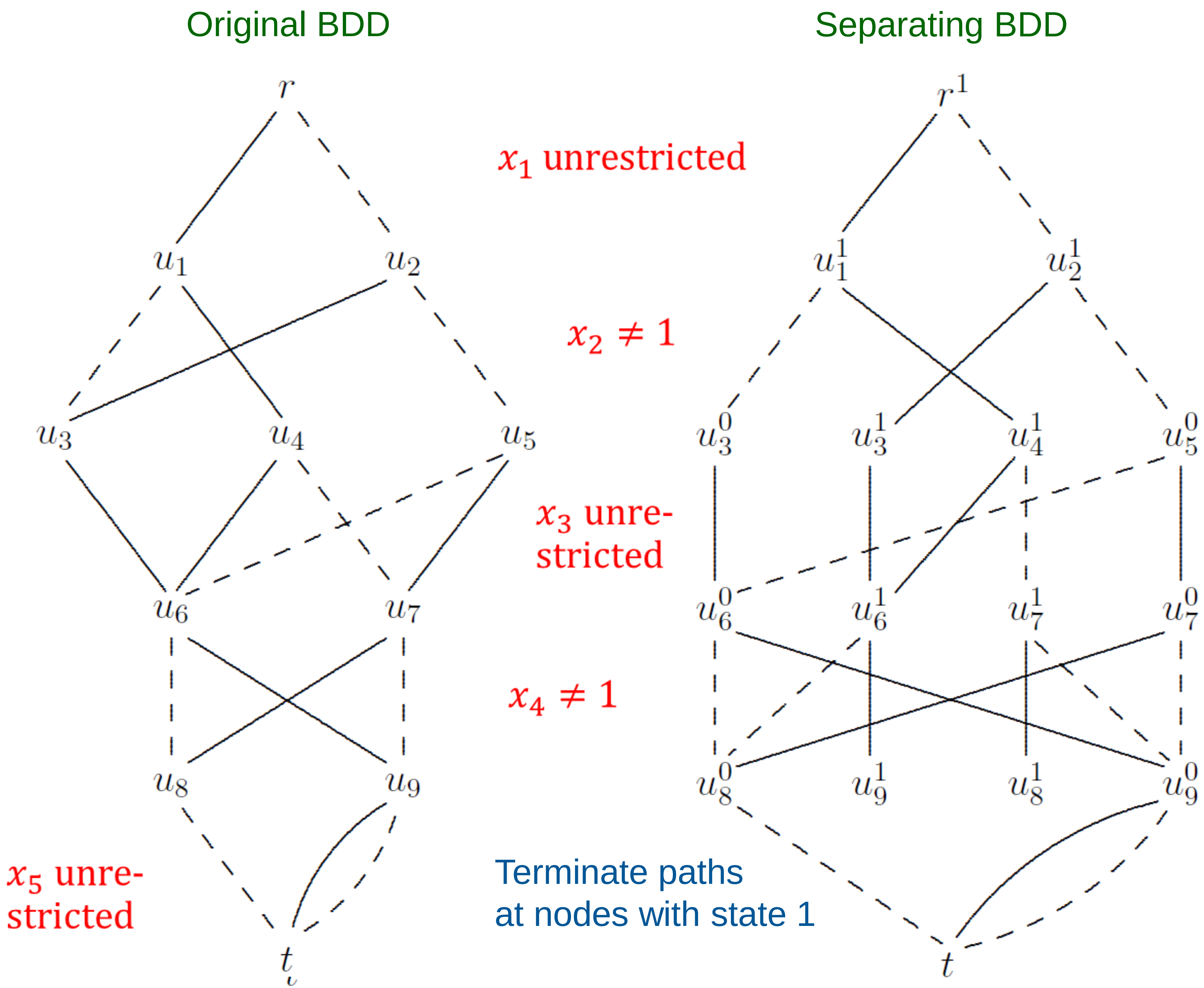

$x_1$ unrestricted

$x_2 \neq 1$

$x_3$ unre-
stricted

$x_4 \neq 1$

1-arcs from state 1 nodes
preserve state
0-arcs switch state to 0.



Original BDD
Separating BDD
$x_1$ unrestricted
$x_2 \neq 1$
$x_3$ unrestricted
$x_4 \neq 1$
$x_5$ unrestricted
Terminate paths at nodes with state 1
$r$
$u_1$
$u_2$
$u_3$
$u_4$
$u_5$
$u_6$
$u_7$
$u_8$
$u_9$
$t$
$r^1$
$u_1^1$
$u_2^1$
$u_3^0$
$u_3^1$
$u_4^1$
$u_5^0$
$u_6^0$
$u_6^1$
$u_7^1$
$u_7^0$
$u_8^0$
$u_9^1$
$u_8^1$
$u_9^0$
$t$

# Size of Separating Diagram

- We wish to separate from a given diagram all solutions *x* in which $x_i = \bar{x}_i$ for $i \in I$.

**Theorem** (easy). The separating diagram is at most *twice as large* as the original BDD.

If only one solution is separated, the separating diagram has at most *one additional node per layer*.

# Size of Separating Diagram

- We wish to separate from a given diagram all solutions *x* in which $x_i = \bar{x}_i$ for $i \in I$.

**Theorem** (easy). The separating diagram is at most *twice as large* as the original BDD.

If only one solution is separated, the separating diagram has at most *one additional node per layer*.

- This refers to separating diagram *created by the algorithm*
  – Not necessarily a **reduced** (minimal) diagram.

# Size of Separating BDD

- We wish to separate from a given BDD all solutions *x* in which $x_i = \bar{x}_i$ for $i \in I$.
- Let $n_i$ be size of layer *i* of original BDD.
- Let *j*,*k* be smallest, largest indices in $I$.

**Theorem** (not so easy).

Size of layer *i* of separating BDD $\leq \begin{Bmatrix} n_i + \varphi_i \text{ if } j \leq i \leq k \\ n_i \;\text{ otherwise} \end{Bmatrix}$

where $\varphi_i = \begin{Bmatrix} \min\{n_i, \varphi_{i-1}\} & \text{if } i-1 \in I \\ \min\{n_i, 2\varphi_{i-1}\} & \text{otherwise} \end{Bmatrix}$

# Size of Separating BDD

- We wish to separate from a given BDD all solutions x in which $x_i = \bar{x}_i$ for $i \in I$.
- Let $n_i$ be size of layer *i* of original BDD.
- Let *j*,*k* be smallest, largest indices in $I$.

**Corollary** Portion of diagram outside the range of indices in $I$ is *unaffected by separation*.

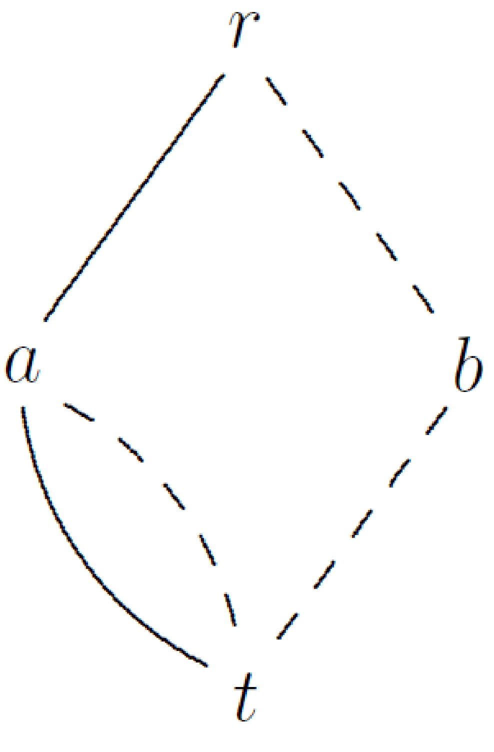
- This will be useful in decomposition methods.

# Reduced Separating Diagram

- Separating diagram generated by the algorithm need not be **reduced**.
  - The reduced diagram for a Boolean function is the smallest diagram that represents the function.
  - It is unique.

- For example…

Original diagram



Original diagram

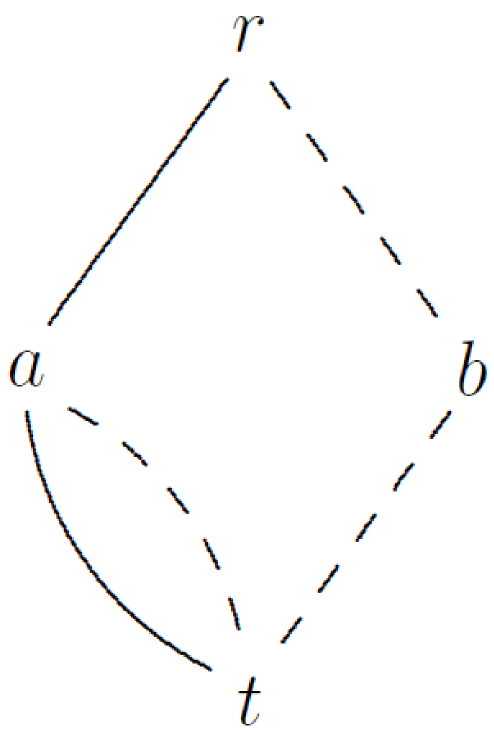
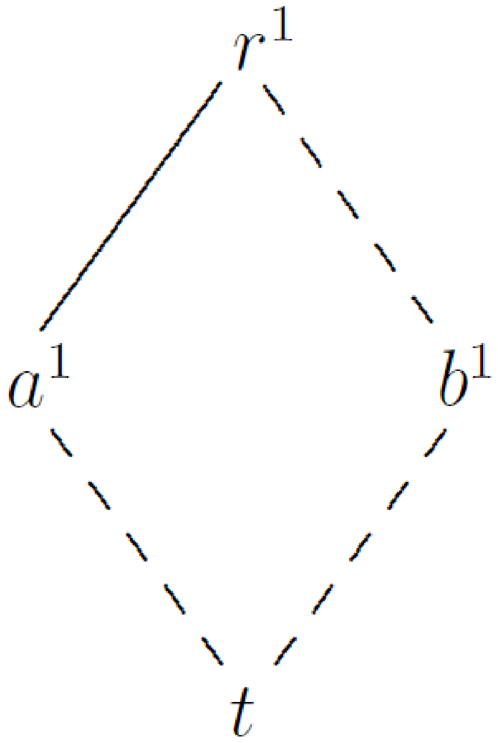



Diagram that separates $x_2 = 1$ as generated by algorithm



Original diagram

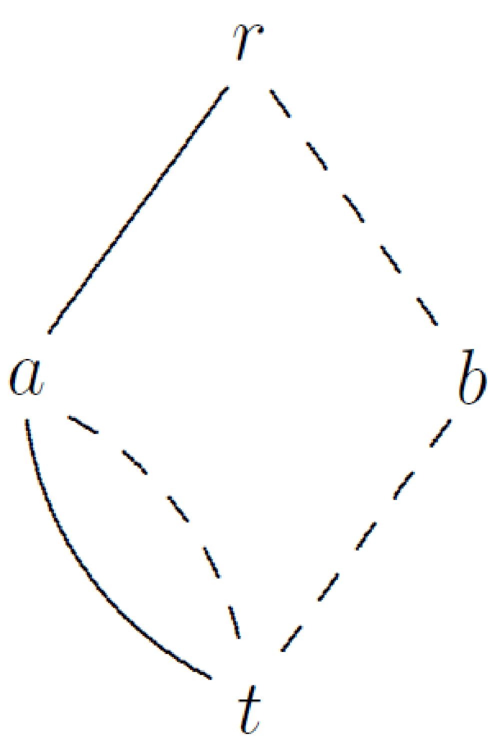
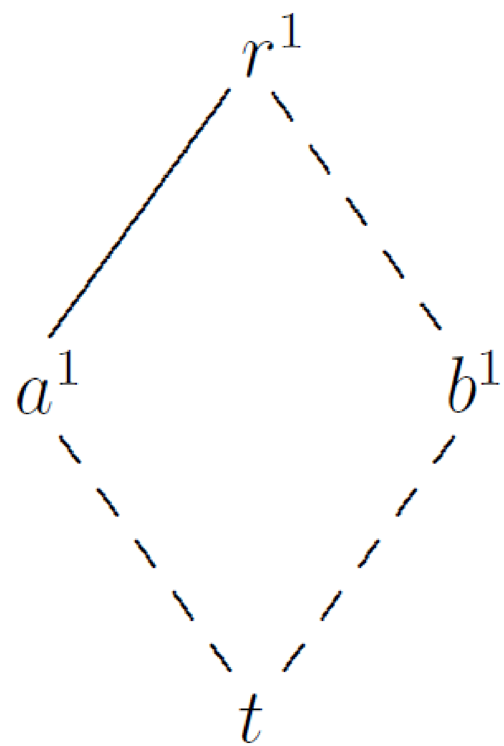
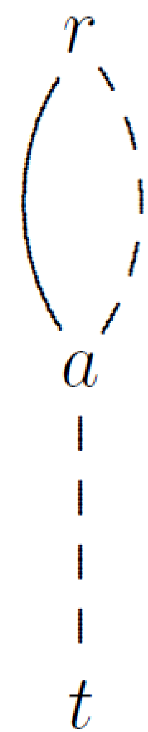



Diagram that separates $x_2 = 1$ as generated by algorithm




Reduced form of separating diagram



# Growth of Separating Diagram

- Key question: How fast does the separating diagram grow when a sequence of partial solutions are separated?
  - Traditional LP relaxation grows **linearly**.
  - One inequality constraint added per solution separated.

# Worst-Case Growth

- Can **reduced** separating diagram grow exponentially?
  - Yes

- Example
  - Start with diagram that represents all Boolean vectors (width 1).
  - Separate: $(1,*,*,\ldots,*,*,1)$

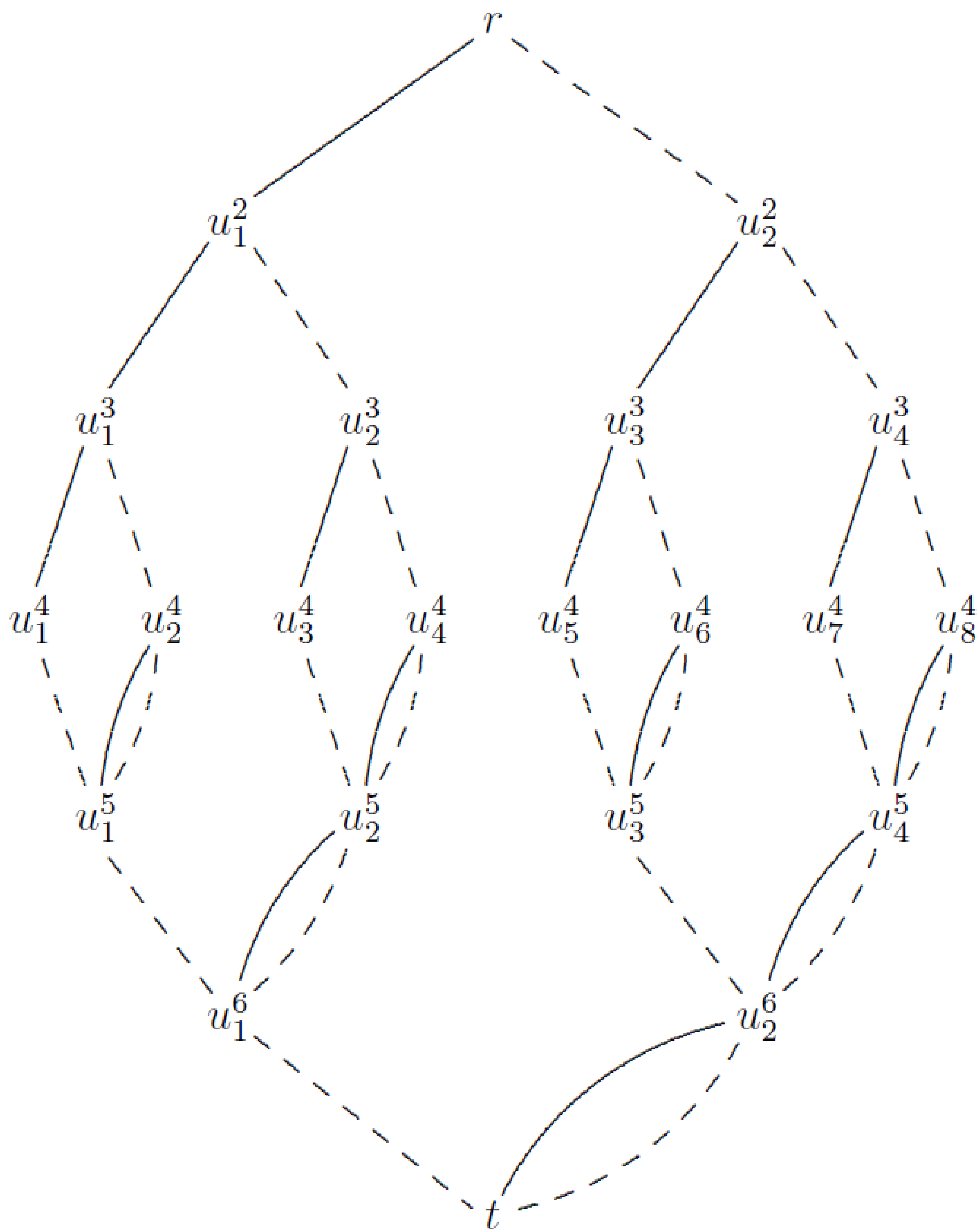
    $(*,1,*,\ldots,*,1,*)$

    $(*,*,1,\ldots,1,*,*)$

    etc.

Reduced diagram for $n = 6$ variables.

It has width $2^{n/2}$



# Empirical Growth

- How fast does the separating diagram grow in Benders method for home health care ?

# Home Health Care

- Reminder…home health care delivery problem.
  - Assign nurses to homebound patients.
  - …subject to constraints on nurse qualifications.
  - Route each nurse through assigned patients, observing time windows.
  - Nurse must take a break if day is long enough.
- Termination.
  - Terminate with **feasible solution** when all nurse scheduling subproblems are feasible.

# Home Health Care

- Reminder…home health care delivery problem.
  - Assign nurses to homebound patients.
  - …subject to constraints on nurse qualifications.
  - Route each nurse through assigned patients, observing time windows.
  - Nurse must take a break if day is long enough.
- Termination.
  - Terminate with **feasible solution** when all nurse scheduling subproblems are feasible.

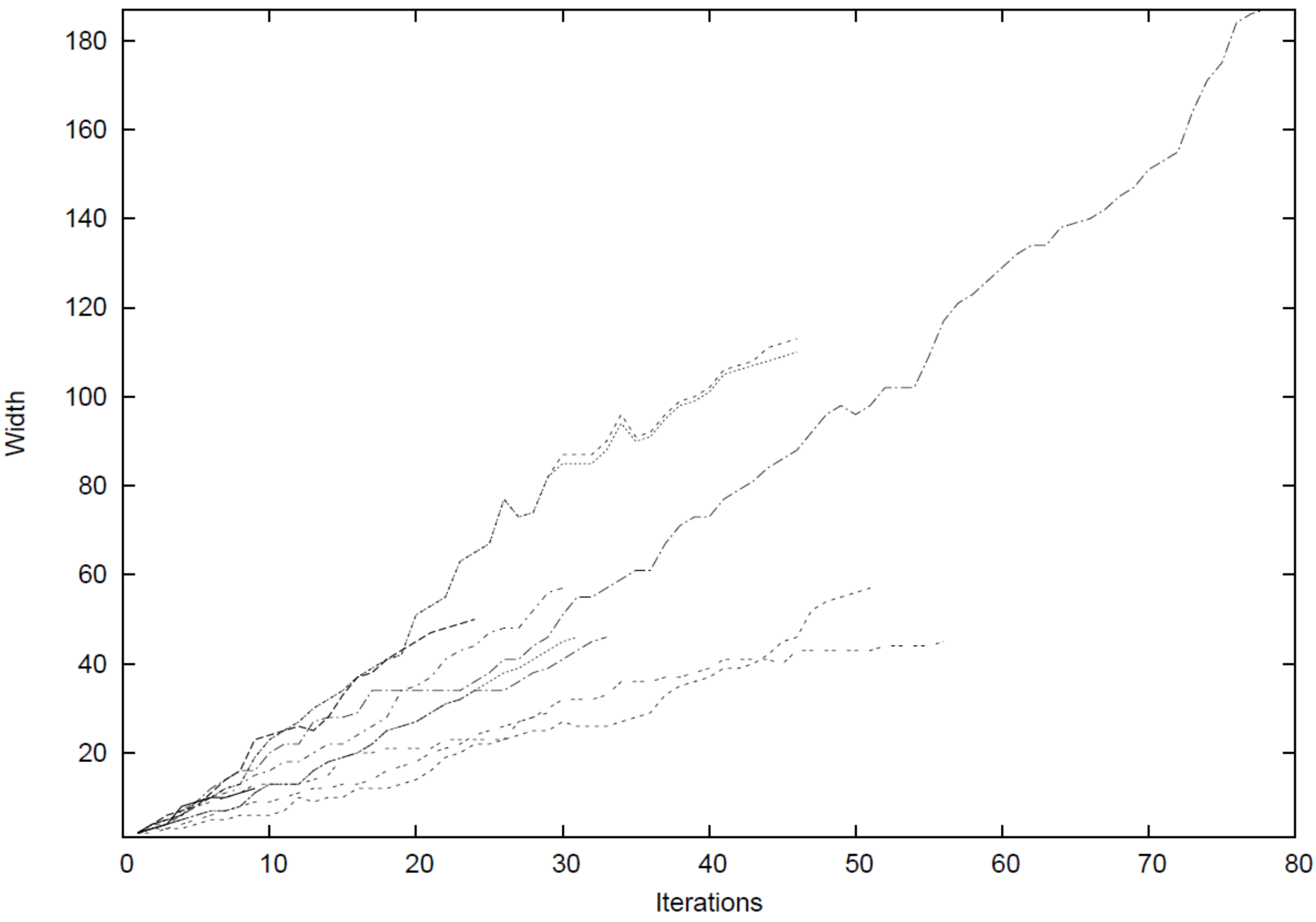
# Home Health Care

- Instances.
  - Scaled-down instances of **real-world problem** obtained from German firm.
  - Assign 6 nurses to 30 patients, one-day horizon.

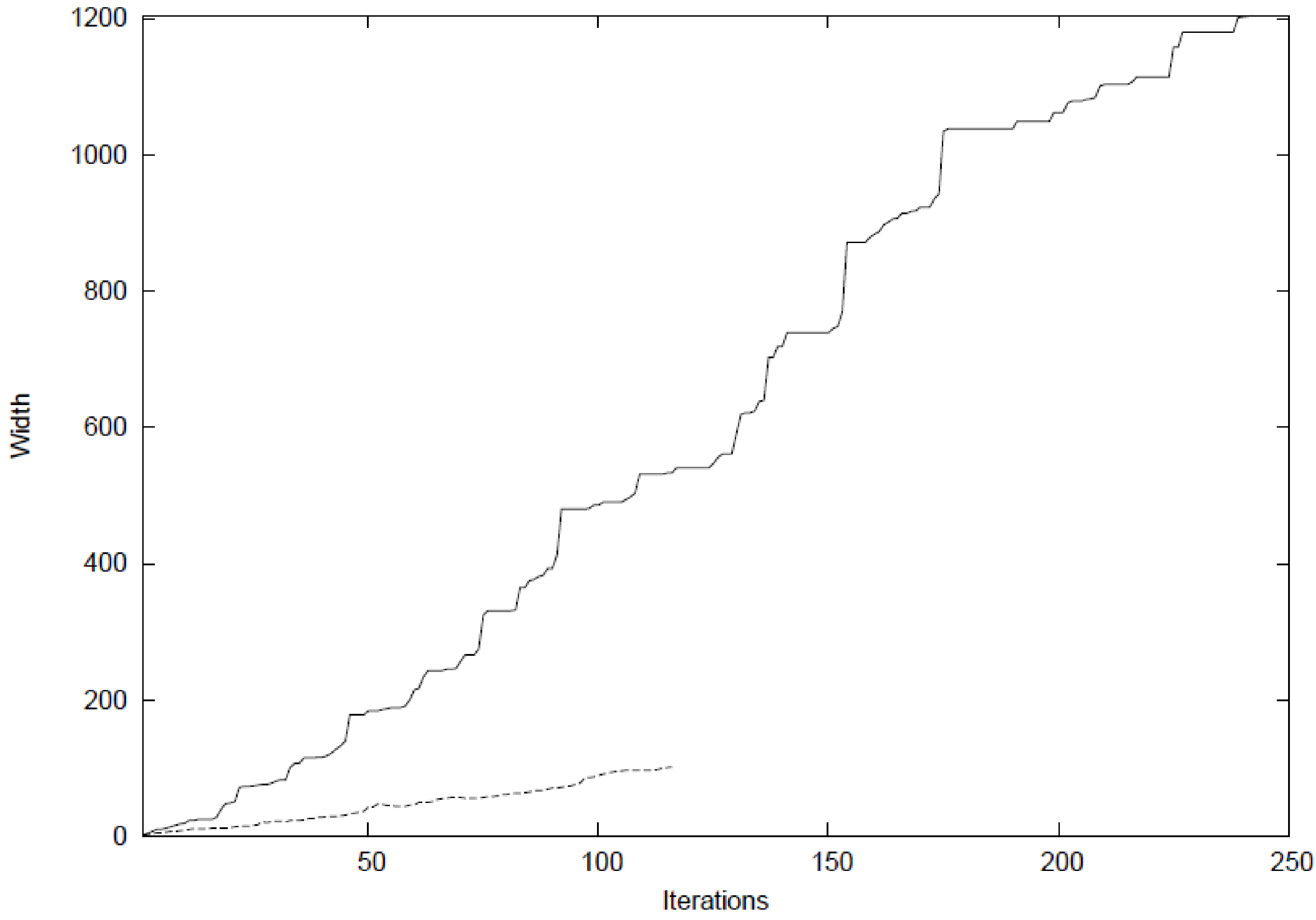
Results for 20 instances

| Instance | Iterations | Time (sec) |
|---|---|---|
| set1-n30r0 | 9 | 7.5 |
| set1-n30r1 | 24 | 24.4 |
| set1-n30r2 | 116 | 69.7 |
| set1-n30r3 | 46 | 40.1 |
| set1-n30r4 | 31 | 19.3 |
| set1-n30r5 | 78 | 64.3 |
| set1-n30r6 | 30 | 29.6 |
| set1-n30r7 | 29 | 18.0 |
| set1-n30r8 | 1 | 10.2 |
| set1-n30r9 | 2 | 11.3 |
| set2-n30r0 | 9 | 8.5 |
| set2-n30r1 | 24 | 23.8 |
| set2-n30r2 | 51 | 31.7 |
| set2-n30r3 | 46 | 39.4 |
| set2-n30r4 | 33 | 22.1 |
| set3-n30r0 | 8 | 3.1 |
| set3-n30r1 | 56 | 9.4 |
| set3-n30r2 | 4 | 0.7 |
| set3-n30r3 | 242 | 80.3 |
| set3-n30r4 | 820 | 568.6 |

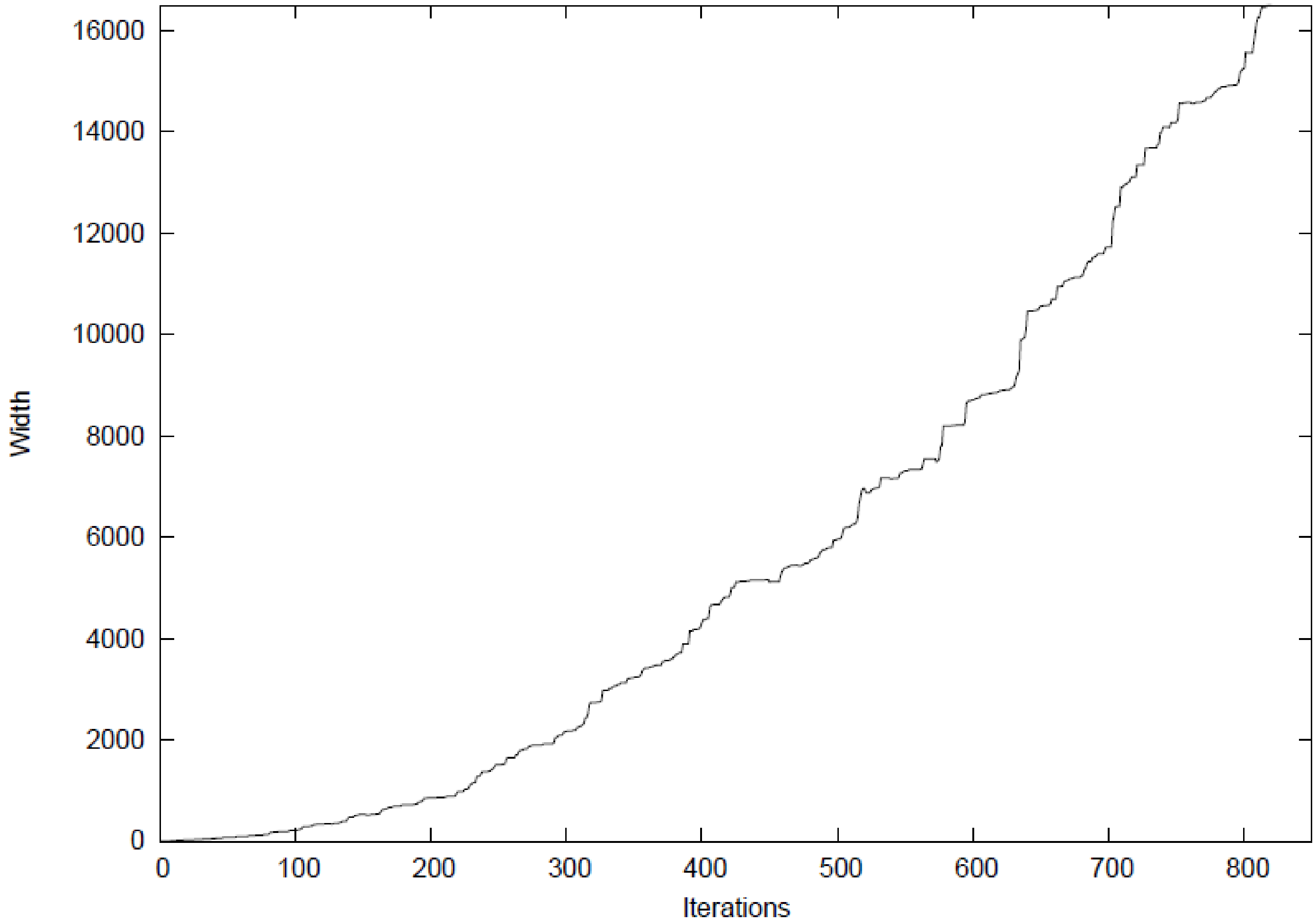
# Growth of Separating Diagram for All but 3 Instances



# Growth of Separating Diagram for 2 Harder Instances



# Growth of Separating Diagram for Hardest Instance



# Empirical Growth

- Separating diagram grows more or less **linearly** in all but one instance.
  - Somewhat superlinear in hardest instance.
  - Most diagrams never exceeded width of 100.
  - A width-1000 diagram can be processed in small fraction of a second.
- Hardest instance:
  - Width 16,496.
  - 820 iterations.
  - Final iteration processed in 2.9 seconds, including solution of subproblem.

# Conclusions

- Benders + decision diagrams may have promise for the home health care delivery problem.
  - Master problem can be solved **quickly** as shortest-path problem in decision diagram.
  - Diagram tends to grow **linearly** as Benders cuts are separated.