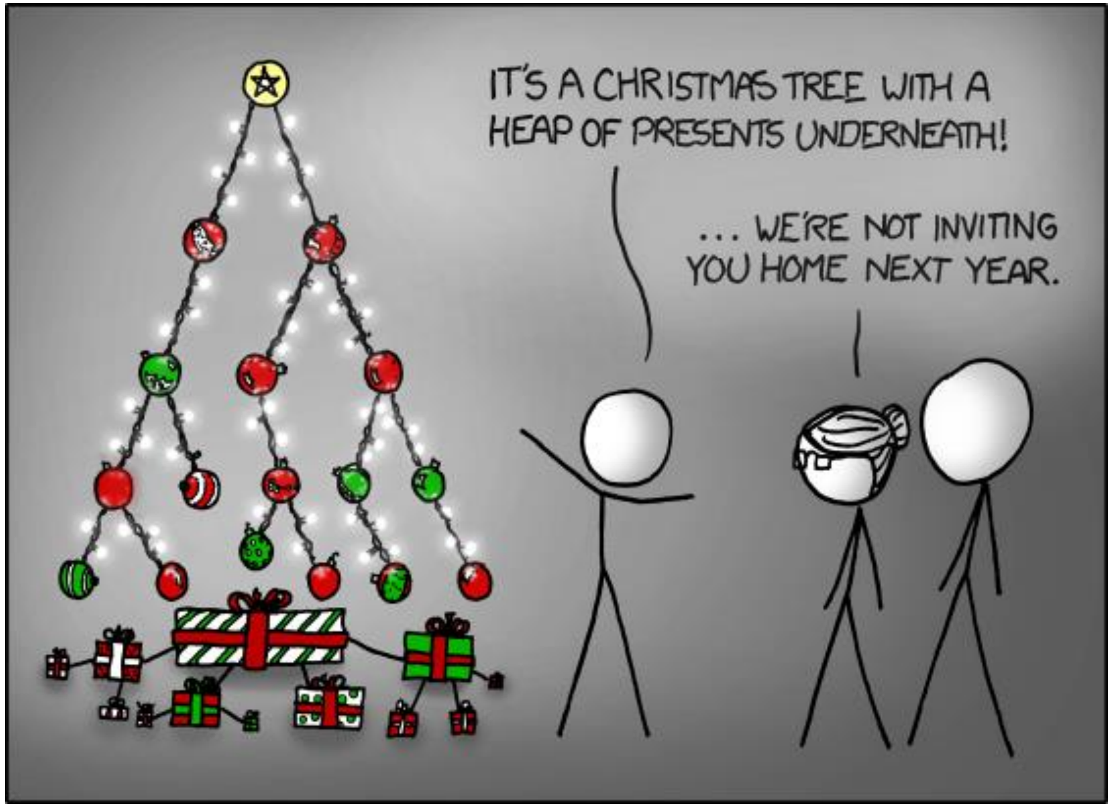
Logic and Computability

Binary Decision Diagrams (BDDs)

Bettina Könighofer
bettina.koenighofer@lamarr.at

Stefan Pranger
stefan.pranger@iaik.tugraz.at



https://xkcd.com/835/

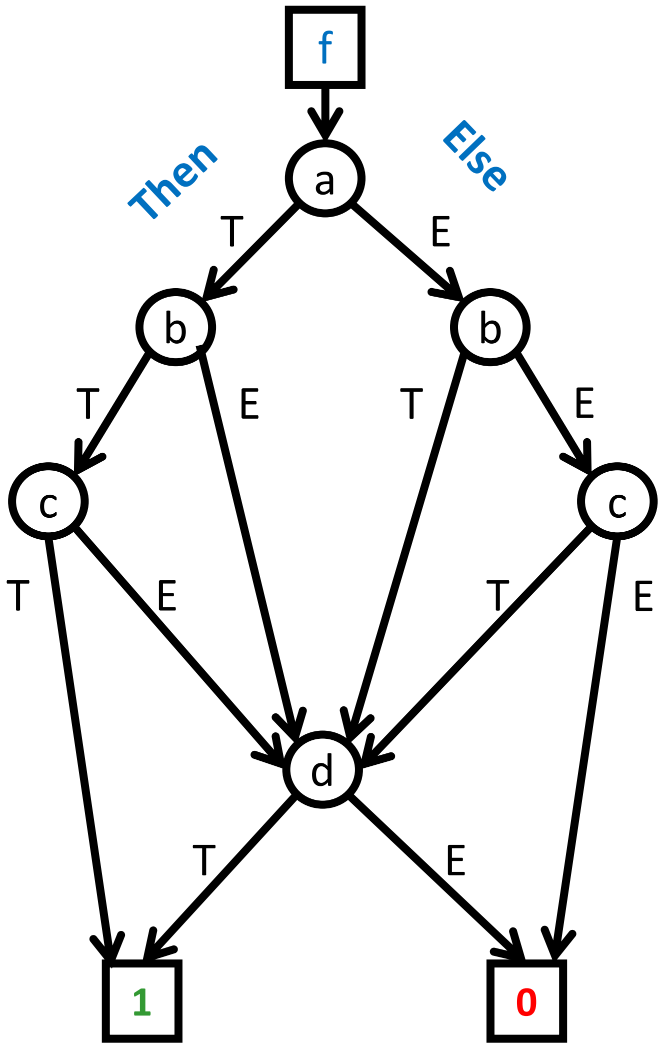
Motivation – BDDs

- Efficient Representation of Boolean Formulas
  - Small for many practical cases
  - Efficient Manipulation
    - Boolean Operations

Outline

- What are Binary Decision Diagrams (BDDs)?
  - Intuitive Explanation
  - Formal Definition
- From BDDs to Reduced Ordered BDDs (ROBDDs)
- Construct Formula from ROBDD
- Construct ROBDD from Formula
- Pros and Cons of BDDs

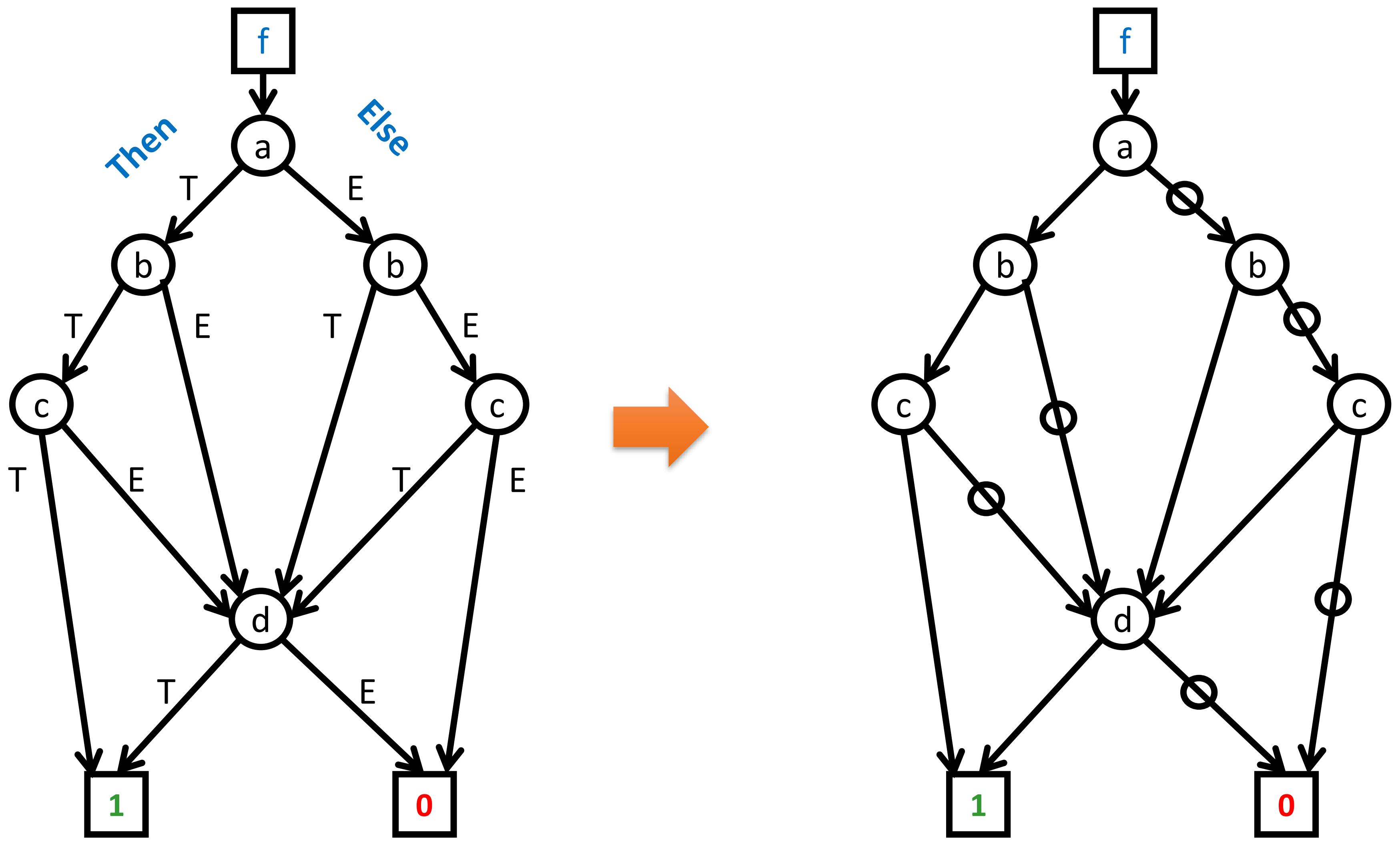
Binary Decision Diagram (BDD)



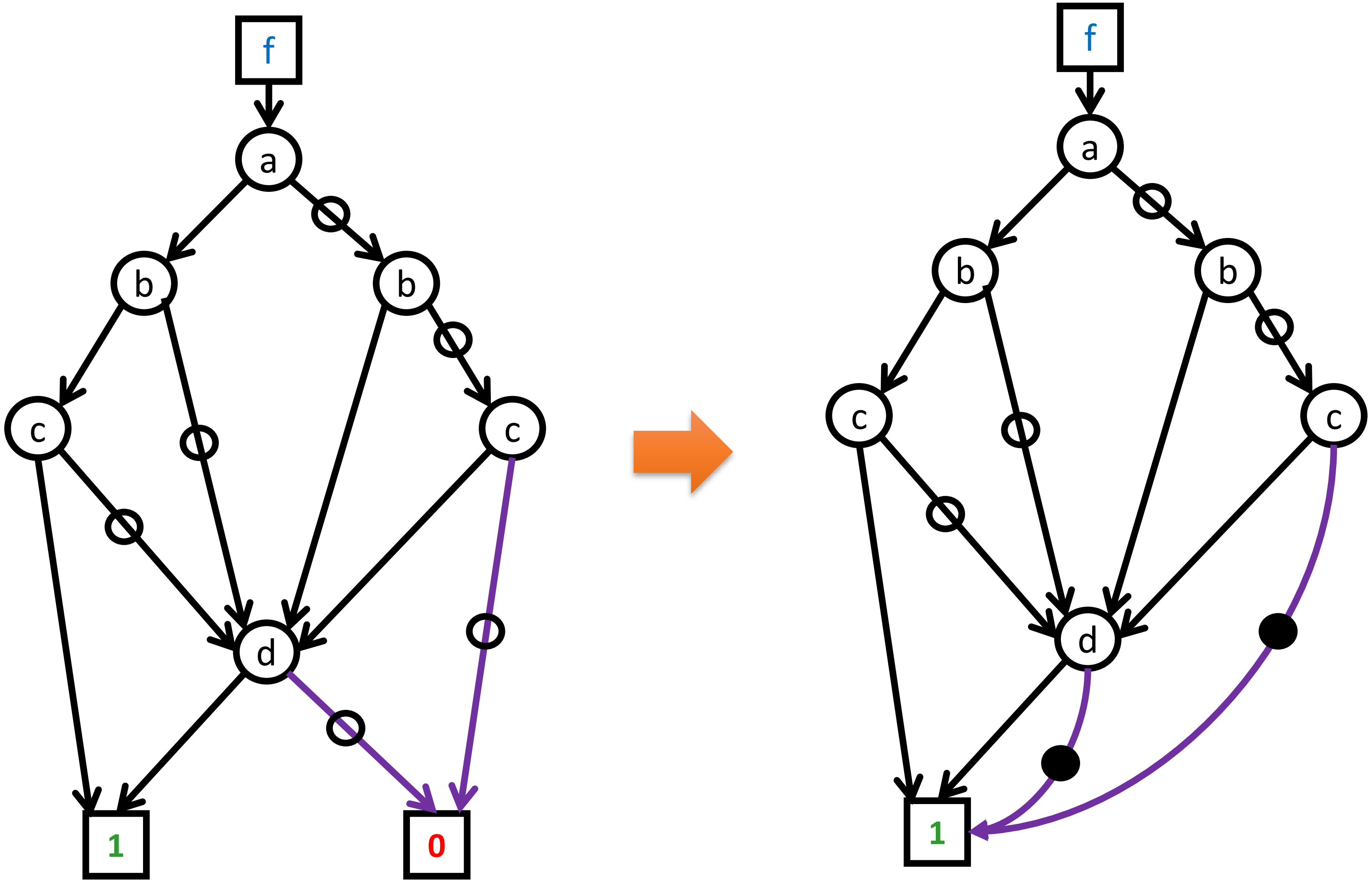
$$M := \{a = T, b = T, c = T, d = T\}$$

M is a satisfying assignment

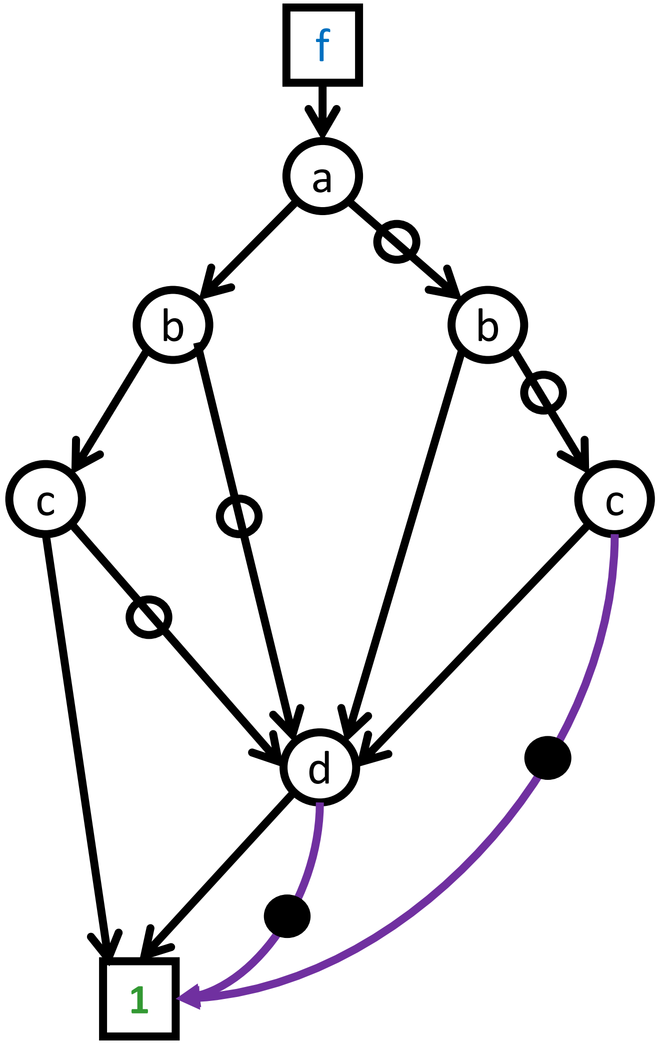
Binary Decision Diagram (BDD)

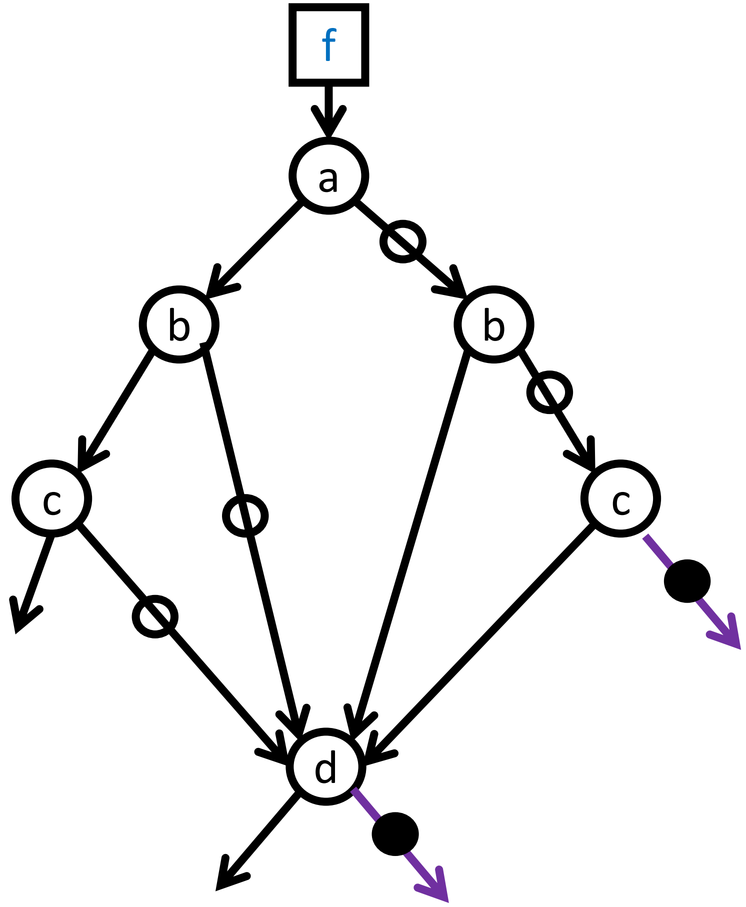


BDD with Complimented Edges

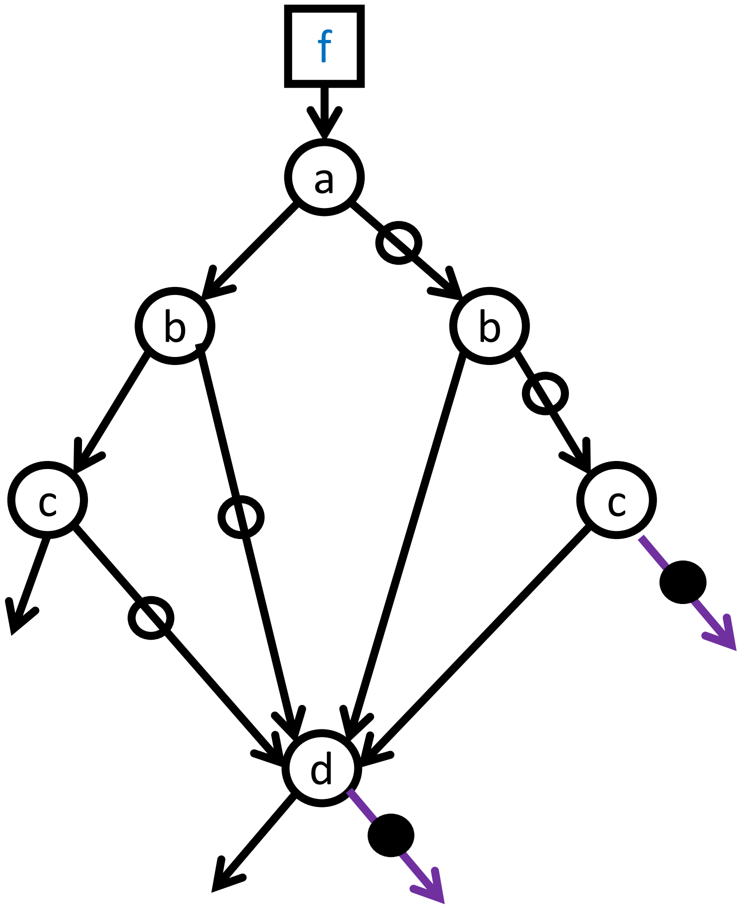


BDD with Complimented and dangling Edges



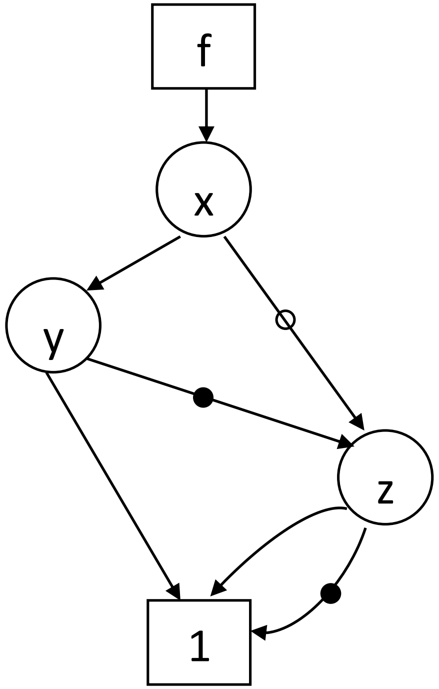


From now on....



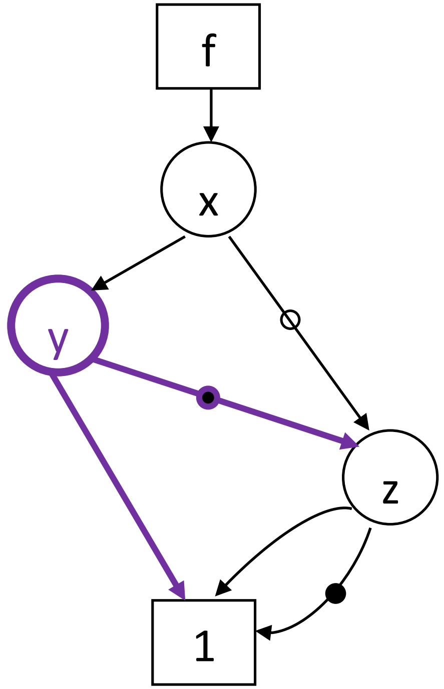
Definition of BDDs

- Directed Acyclic Graph
- $(V \cup \Phi \cup \{\mathbf{1}\},\ E)$
  - Internal Nodes $v \in V$
  - Function Nodes $f_i \in \Phi$
  - Terminal Node $\mathbf{1}$
  - Edges E
    - “Complement” attribute



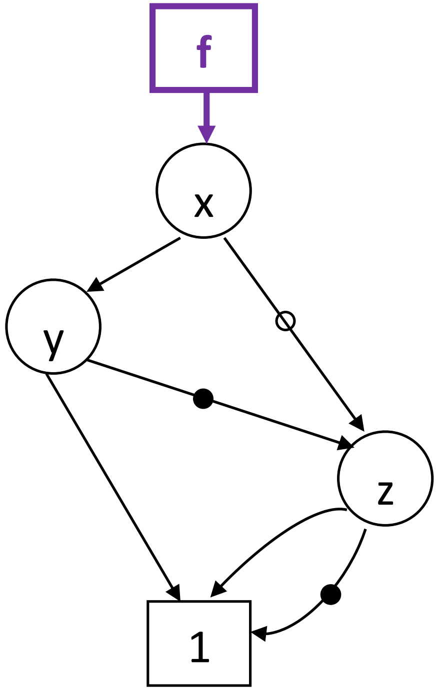
Definition of BDDs: Internal Node

- Label $l(v) \in \{x_1, \dots, x_n\}$
  - Variables of f
- Out–degree: 2
  - Then-Edge T
  - Else-Edge E
    - Marked with (empty) circle
    - Can have complement attribute (full cycle)



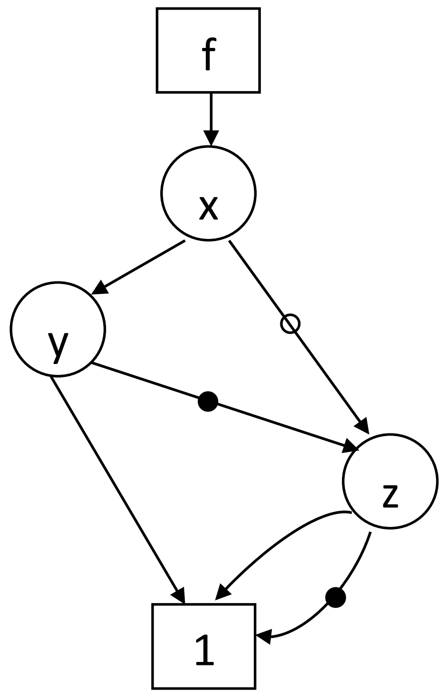
Definition of BDDs: Function Node

- Represents Boolean Formula f_i
- In-degree: 0
- Out-degree: 1
  - Edge can have complement attribute



Definition of BDDs: Terminal Node

- Constant Function **True**
- Out-degree: 0



Outline

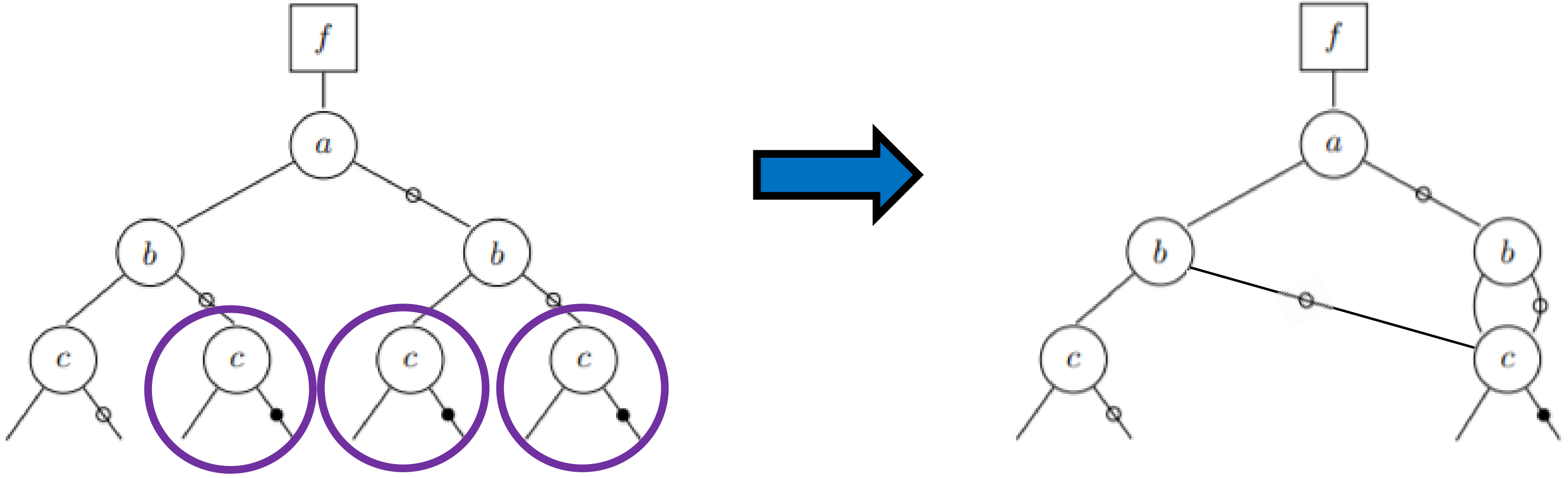
- What are Binary Decision Diagrams (BDDs)?
  - Intuitive Explanation
  - Formal Definition
- **From BDDs to Reduced Ordered BDDs (ROBDDs)**

- Construct Formula from ROBDD
- Construct ROBDD from Formula

- Pros and Cons of BDDs

From BDD to Reduced BDD

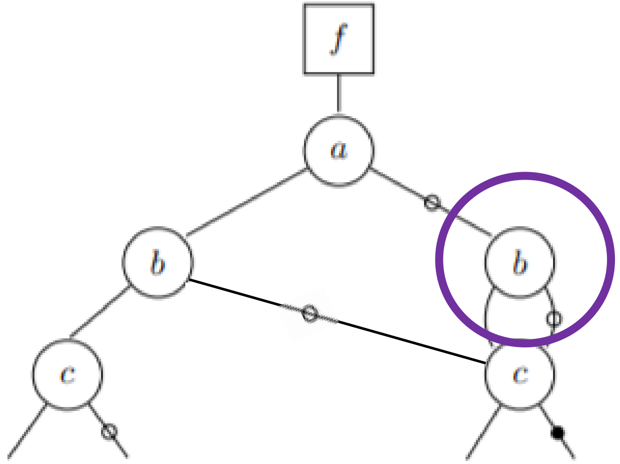
1. No duplicate sub-BDDs

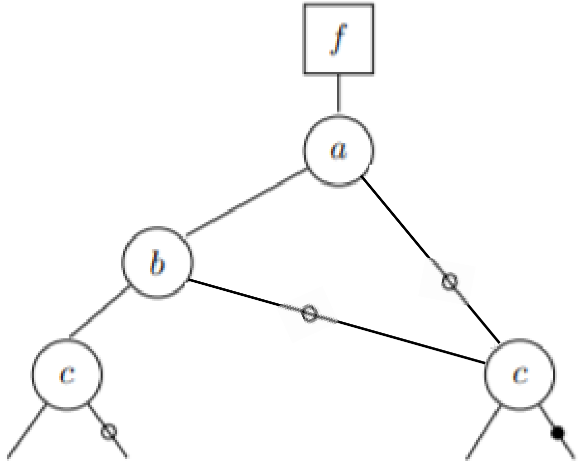


From BDD to Reduced BDD

1. No duplicate sub-BDDs
2. No redundant nodes

} Reduced BDD



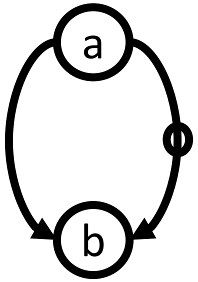


From BDD to Reduced BDD

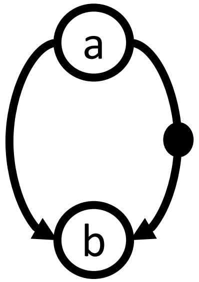
1. No duplicate sub-BDDs
2. No redundant nodes

} Reduced BDD

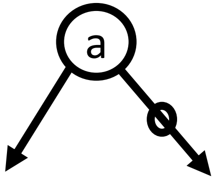
Redundant



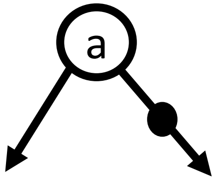
NOT Redundant



Redundant
(special case)

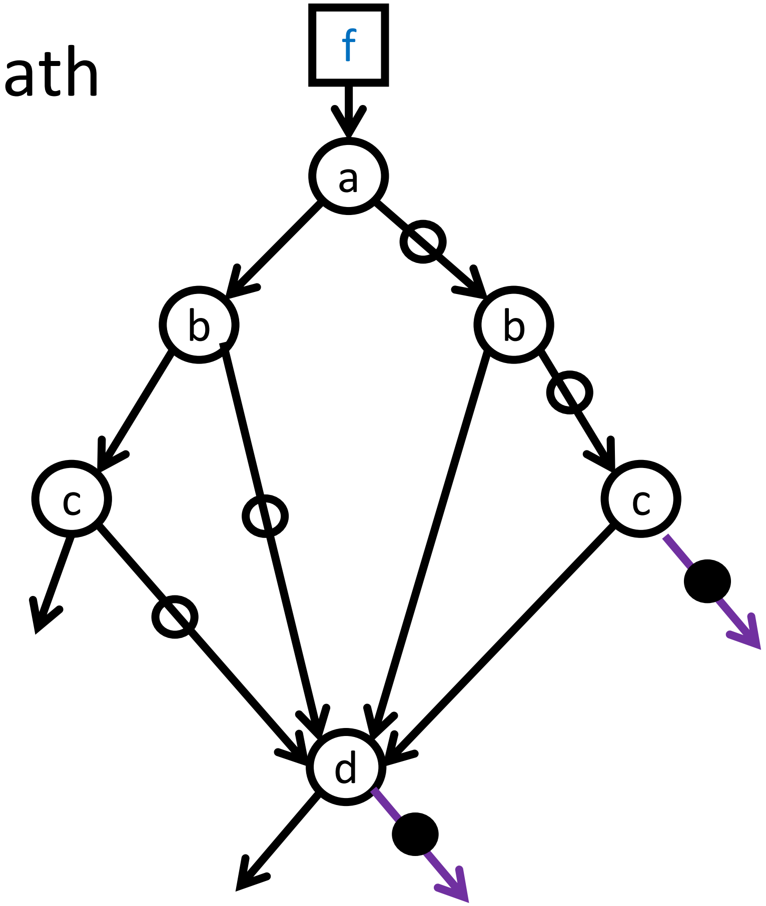


NOT Redundant
(special case)



From RBDD to Reduced Ordered BDD (ROBDD)

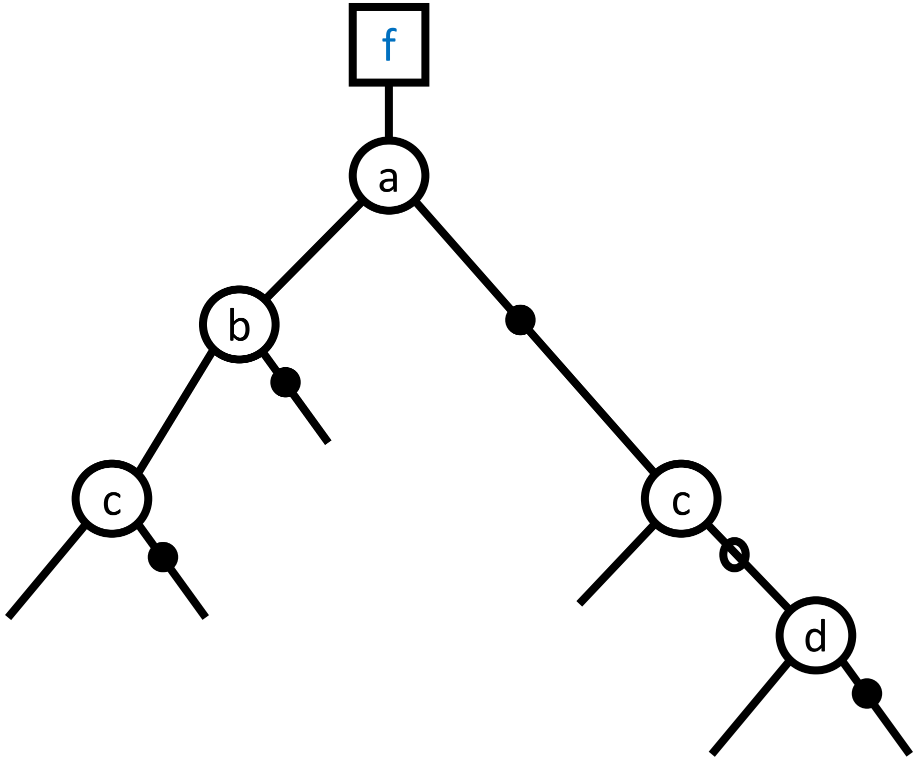
- Ordering on the variables along any path
  - E.g., $a < b < c < \mathrm{d}$
- A ROBDD gives a **canonical** representation of a formula
  - For given variable ordering
  - Meaning:
    - If two formulas are semantically equivalent, they will be represented by the exact same ROBDD
  - Allows Equivalence Checking in constant time



Outline

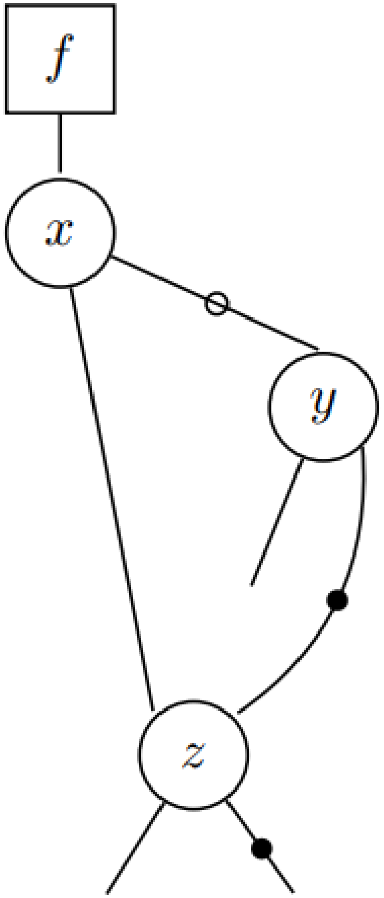
- What are Binary Decision Diagrams (BDDs)?
  - Intuitive Explanation
  - Formal Definition
- From BDDs to Reduced Ordered BDDs (ROBDDs)
- Construct Formula from ROBDD
- Construct ROBDD from Formula
- Pros and Cons of BDDs

From ROBDD to Formula, Example 1



$$f = (a \wedge b \wedge c) \vee (\neg a \wedge \neg c \wedge \neg d)$$

From ROBDD to Formula, Example 2



$$f := (\neg x \wedge y) \vee (\neg x \wedge \neg y \wedge \neg z) \vee (x \wedge z).$$

Outline

- What are Binary Decision Diagrams (BDDs)?
  - Intuitive Explanation
  - Formal Definition
- From BDDs to Reduced Ordered BDDs (ROBDDs)
- Construct Formula from ROBDD
- Construct ROBDD from Formula
- Pros and Cons of BDDs

From Formula to BDD

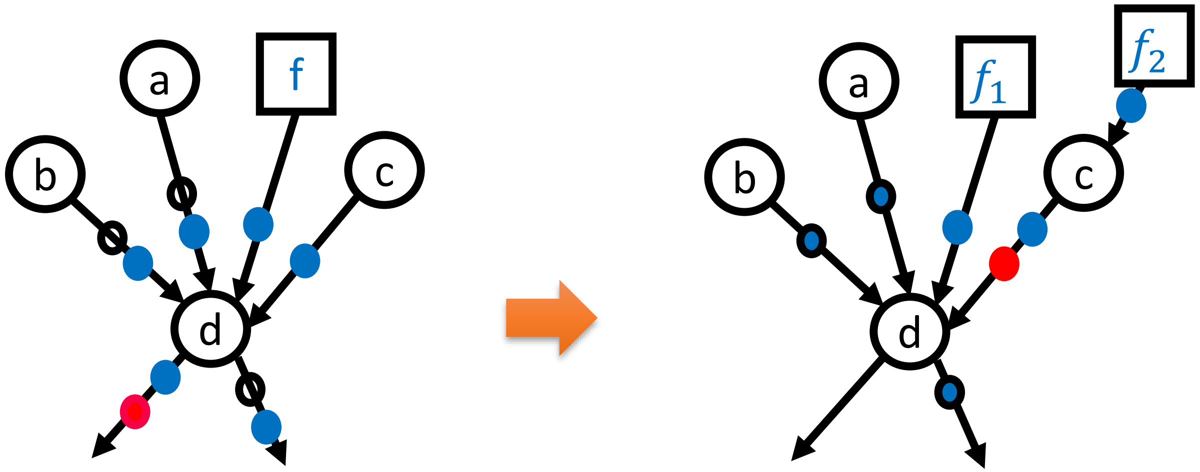
1. Compute all Cofactors
2. Draw ROBDD from Cofactors
3. Shift Negations Upwards

From Formula to BDD – Step 1: Cofactors

- Boolean formula f w.r.t. a variable x
  - Positive Cofactor f_x: f with x set to ⊤
  - Negative Cofactor $f_{\neg x}$: f with x set to ⊥

- Example:
  - $f = (x \wedge y) \vee (\neg x \wedge z)$
    - $f_x = y$
    - $f_{\neg x} = z$

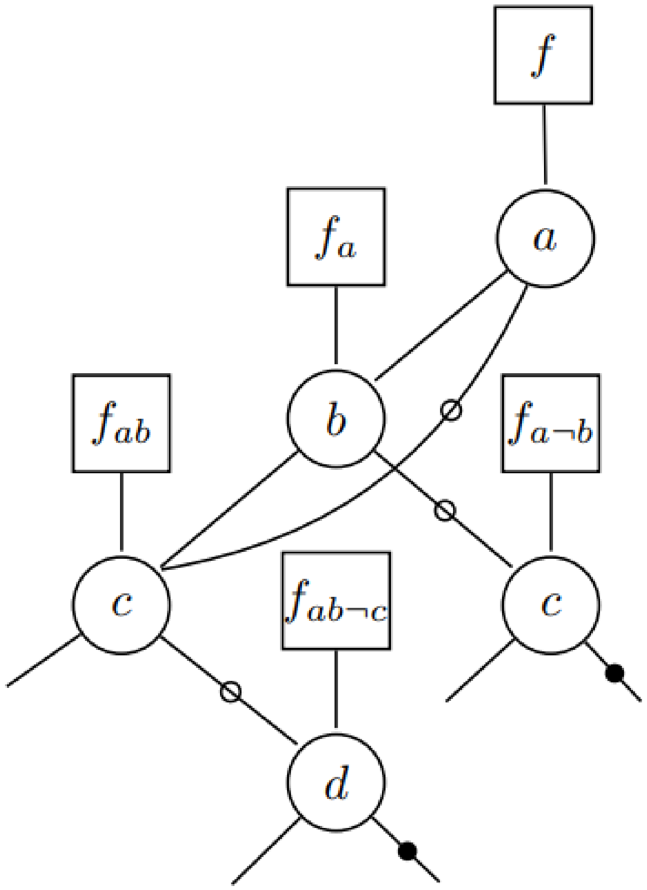
From Formula to BDD – Step 3: Shift Negations Upwards



From Formula to BDD – Example 1

$$f = (a \wedge b \vee \neg a) \wedge \neg c \wedge d \vee c$$

$$
\begin{aligned}
f &= (a \wedge b \vee \neg a) \wedge \neg c \wedge d \vee c \\
f_a &= b \wedge \neg c \wedge d \vee c \\
f_{ab} &= \neg c \wedge d \vee c \\
f_{abc} &= \top \\
f_{ab\neg c} &= d \\
f_{ab\neg cd} &= \top \\
f_{ab\neg c\neg d} &= \bot \\
f_{a\neg b} &= c \\
f_{a\neg bc} &= \top \\
f_{a\neg b\neg c} &= \bot \\
f_{\neg a} &= \neg c \wedge d \vee c = f_{ab}
\end{aligned}
$$



From Formula to BDD – Example 2

$$f \;=\; (a \wedge \neg c) \vee \left(\neg a \wedge (b \vee (\neg b \wedge c))\right)$$

$$f = (a \wedge \neg c) \vee \left(\neg a \wedge (b \vee (\neg b \wedge c))\right)$$

$$f_a = \neg c$$

$$f_{ab} = \neg c = f_a$$

$$f_{abc} = \bot$$

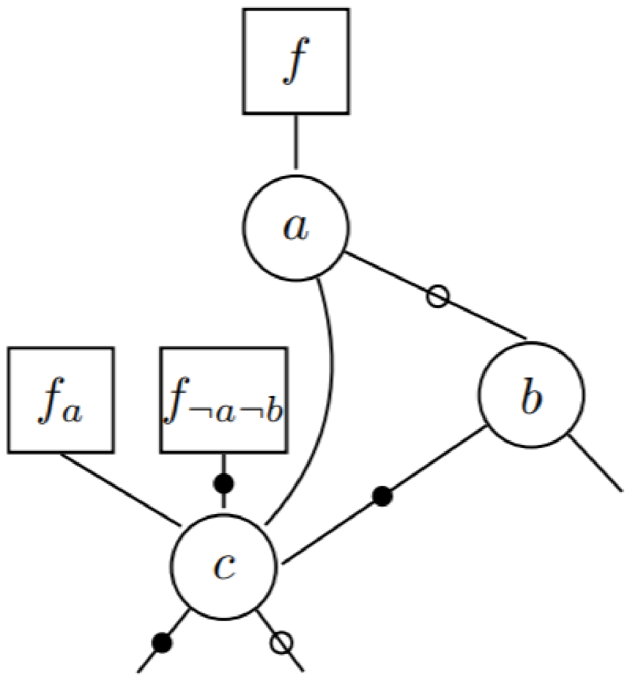
$$f_{ab\neg c} = \top$$

$$f_{a\neg b} = \neg c = f_a$$

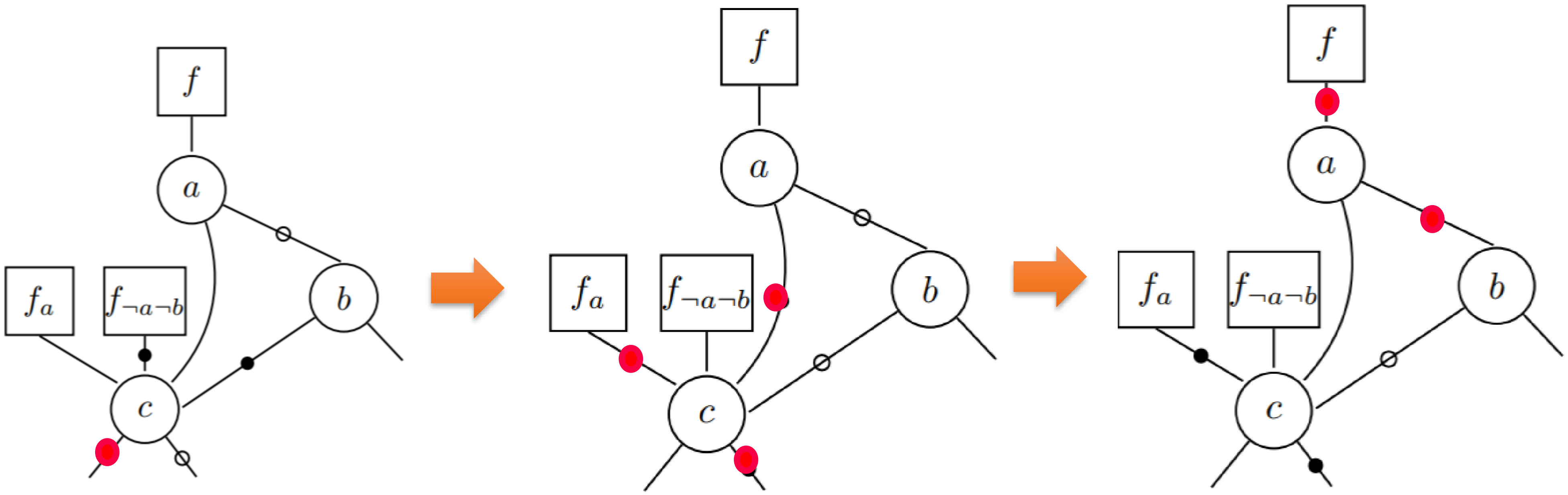
$$f_{\neg a} = b \vee (\neg b \wedge c)$$

$$f_{\neg ab} = \top$$

$$f_{\neg a \neg b} = c = \neg f_{ab}$$



From Formula to BDD – Example 2



Outline

- What are Binary Decision Diagrams (BDDs)?
  - Intuitive Explanation
  - Formal Definition
- From BDDs to Reduced Ordered BDDs (ROBDDs)
- Construct Formula from ROBDD
- Construct ROBDD from Formula
- Pros and Cons of BDDs

Advantages / Disadvantages of BDDs

\+ Size-Efficiency

- Worst case: exponential
- Often: BDDs contain a lot of redundancy. Eliminating redundancy results in small BDD

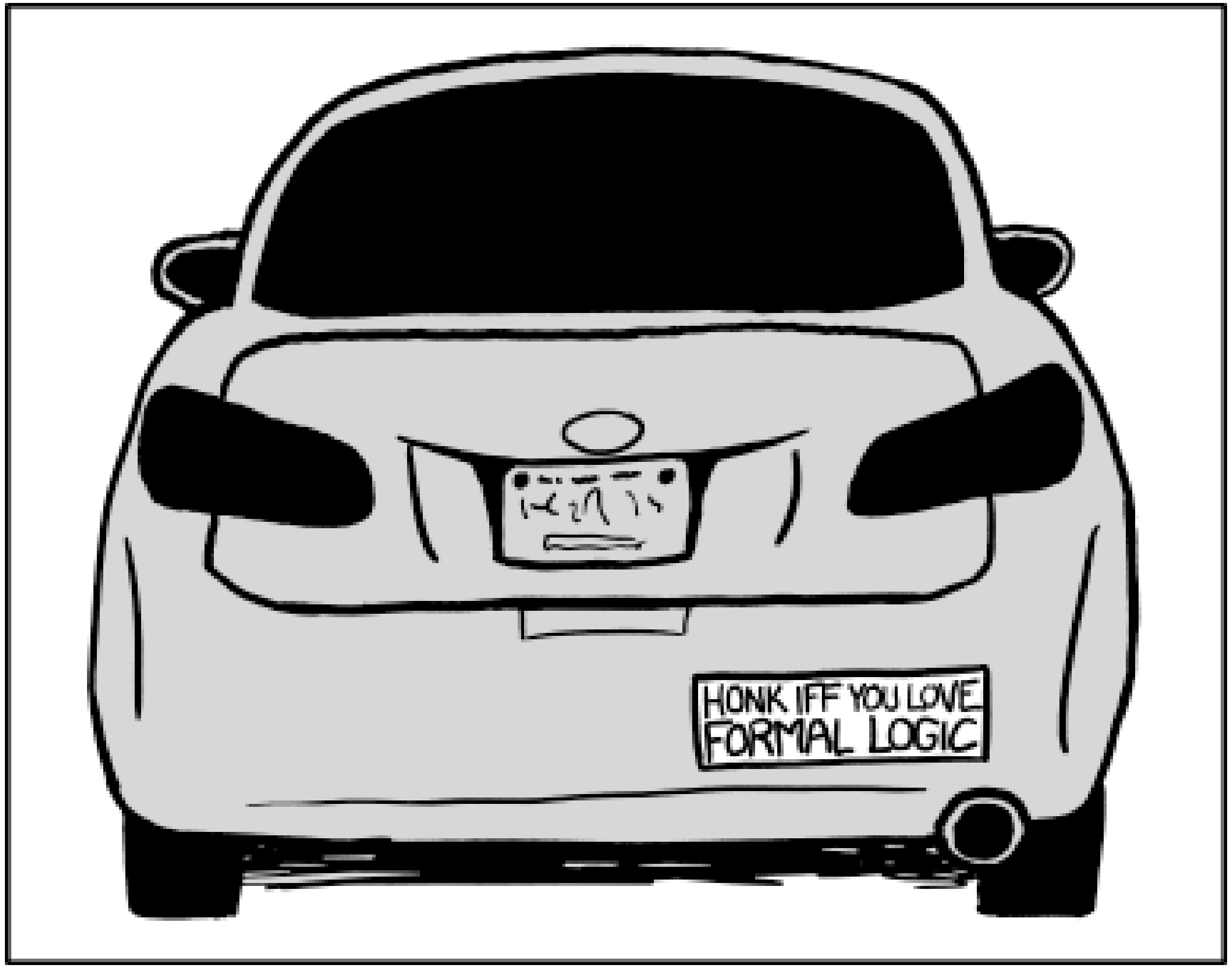
\+ Efficient Operations

- e.g. AND, OR: Polynomial time
- Equivalence Check: Constant time
- Satisfiability and Validity Check: Constant Time

– Variable order

- Big impact
- Hard to optimize

Thank You



https://xkcd.com/1033/