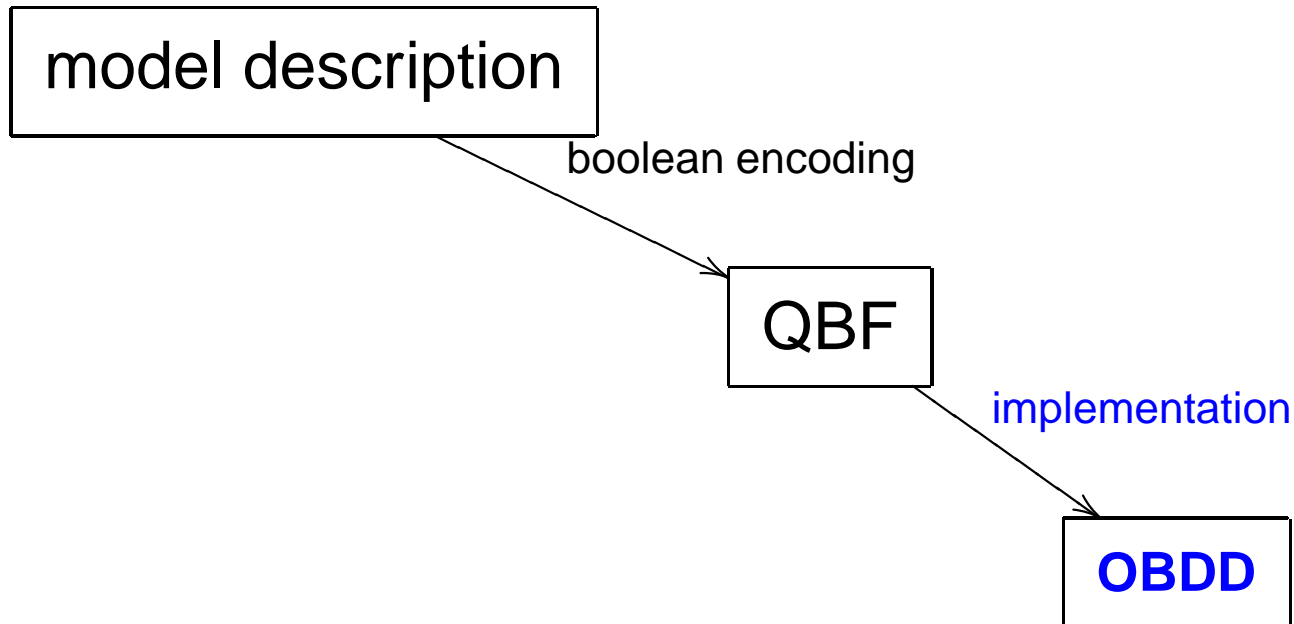


Computer aided verification

Lecture 6: Symbolic verification I: OBDD

Symbolic model checking

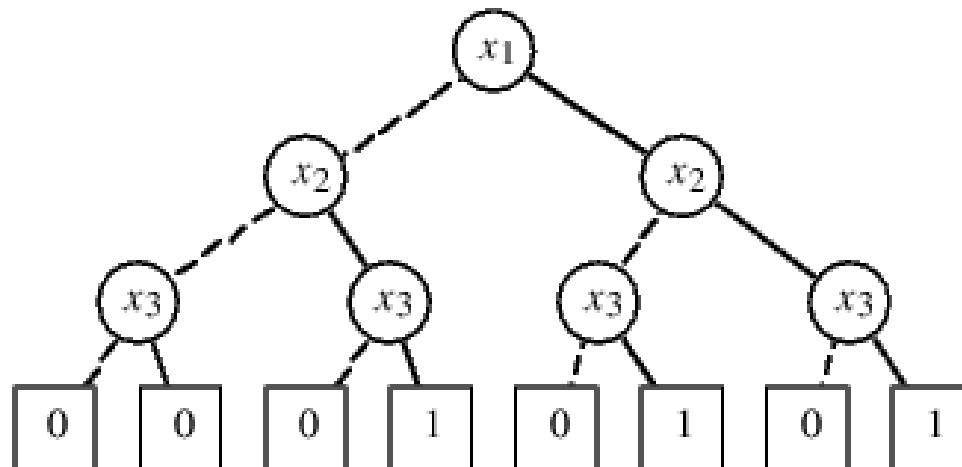


model checking = operations on OBDDs

I. OBDD

Ordered Binary Decision Diagrams

x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



[Bryant 1992]

- $f : \{0, 1\}^3 \rightarrow \{0, 1\}$
- fixed order on variables: $x_1 < x_2 < x_3$

OBDD = rooted acyclic graph

Attributes of a node v :

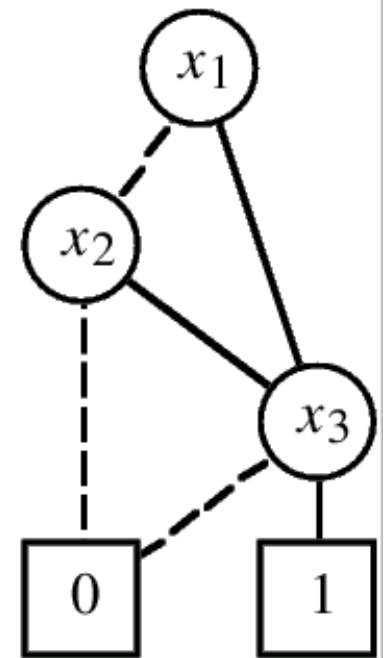
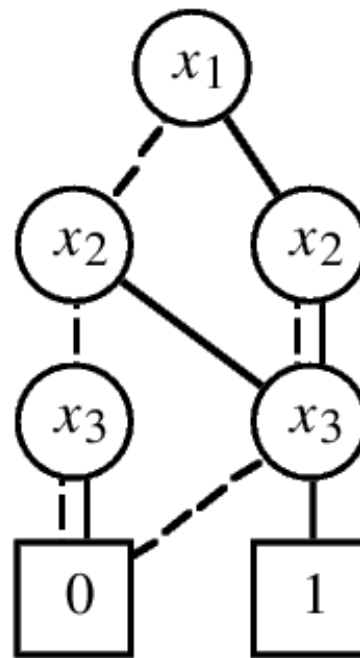
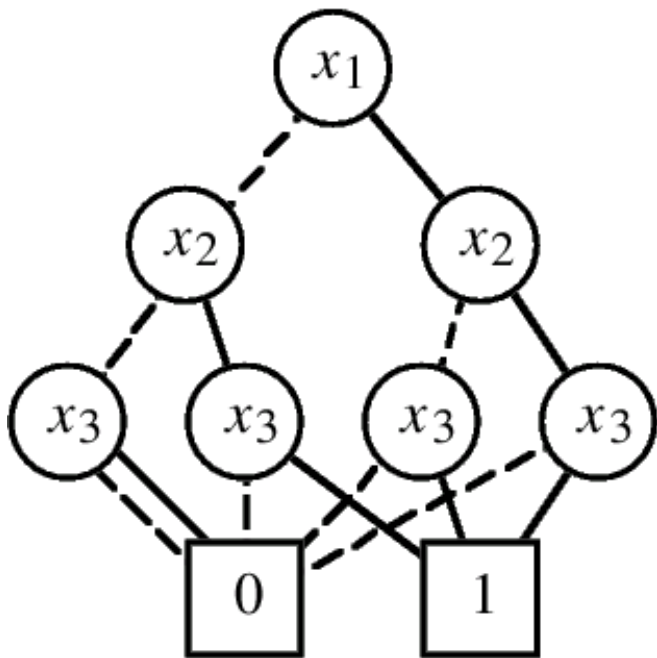
- when v is a leaf
 - $\text{val}(v) \in \{0, 1\}$
- when v is not a leaf
 - $\text{var}(v) \in \{x_1, x_2, \dots\}$
 - $\text{lo}(v), \text{hi}(v)$ – 2 nodes

The order of variables must be obeyed on every path.

Simplifying OBDD

- remove redundant leaves
- remove redundant non-leaves
- remove redundant tests

OBDD \mapsto ROBDD



[Bryant 1992]

Canonical form: reduced OBDD

Canonical form for a **boolean function**:

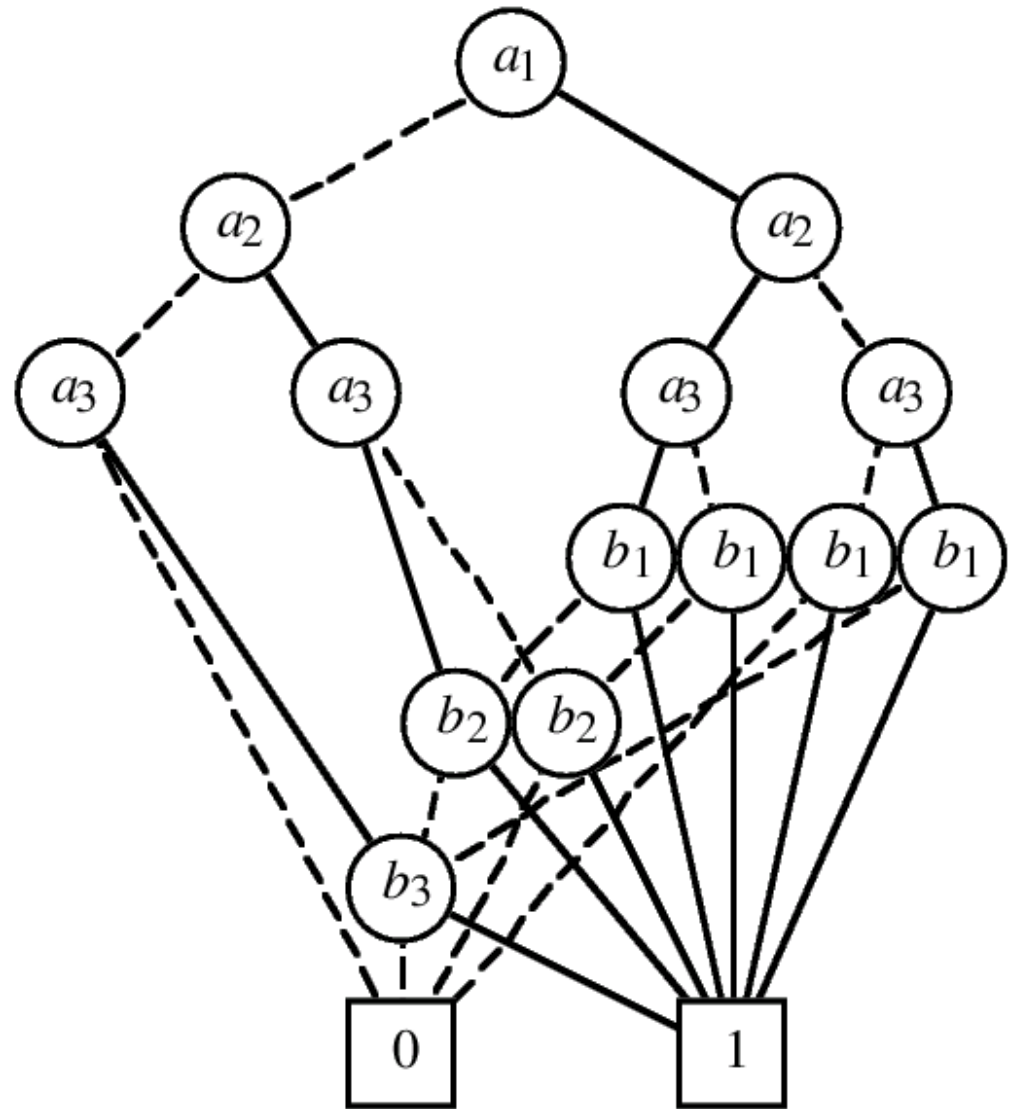
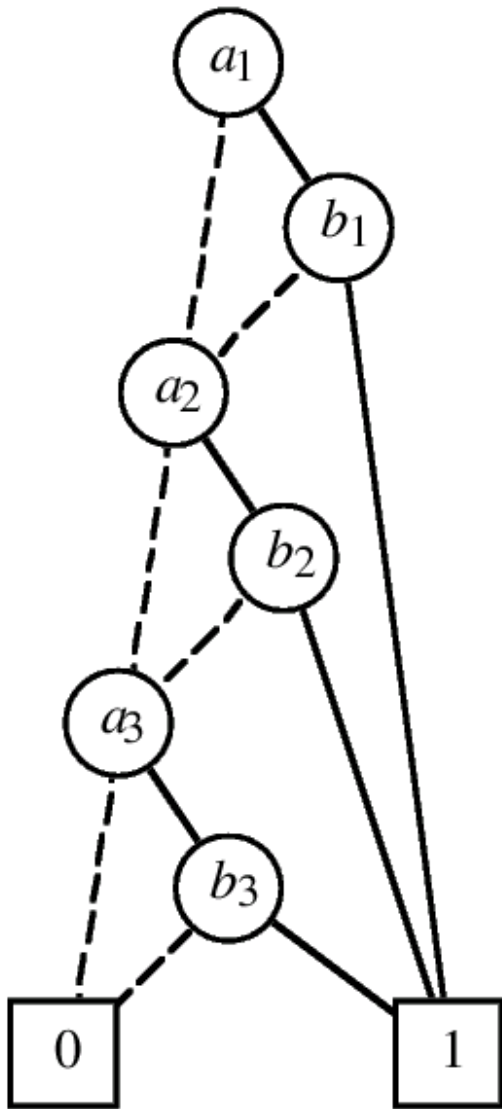
- independent from the initial OBDD
- (**strongly**) dependent on the order of variables

Naive construction of an OBDD for a boolean formula ϕ :

$\phi \longmapsto$ decision tree \longmapsto canonical OBDD

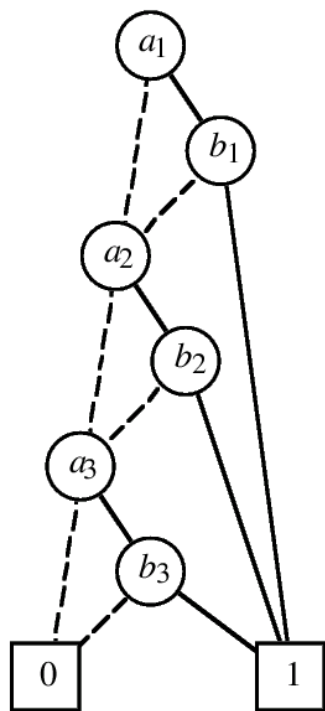
Finding an appropriate order of variables is **crucial!**

$$(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee (a_3 \wedge b_3)$$

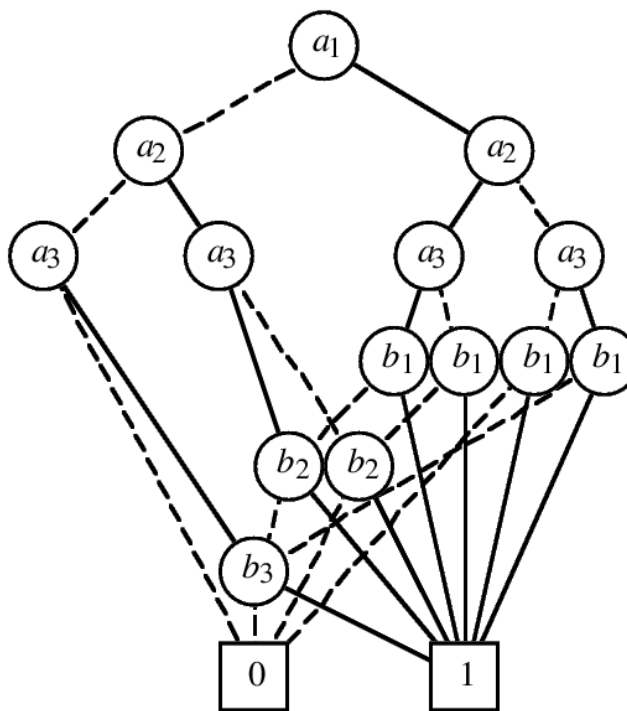


[Bryant 1992]

$$f(a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n) = (a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee \dots \vee (a_n \wedge b_n)$$



$$2 \cdot n$$



$$2 \cdot (2^n - 1)$$

Heuristic: closely related variables should be close in the order

example of a boolean function	lower bound	upper bound
symmetric functions	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
addition (any bit)	$\mathcal{O}(n)$	$\mathcal{O}(2^n)$
multiplication (middle bit)	$\mathcal{O}(2^n)$	$\mathcal{O}(2^n)$

Shannon's expansion

$$f = \neg x \wedge f|_{x \leftarrow 0} \vee x \wedge f|_{x \leftarrow 1}$$

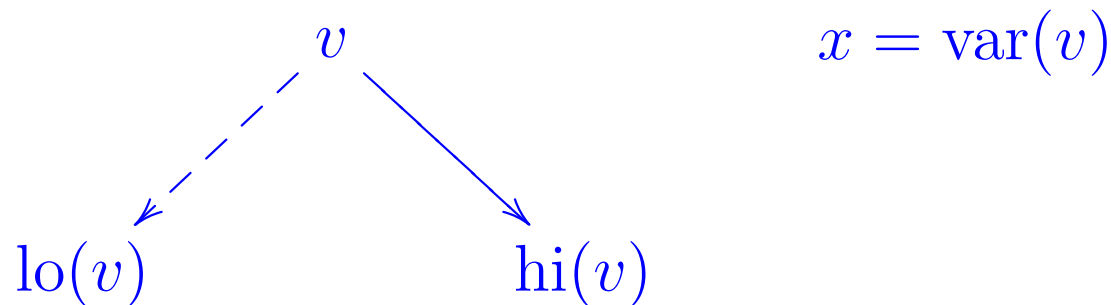
$$f|_{x_i \leftarrow b}(x_1, \dots, x_n) = f(x_1, \dots, x_{i-1}, b, x_{i+1}, \dots, x_n)$$

Shannon's expansion

$$f = \neg x \wedge f|_{x \leftarrow 0} \vee x \wedge f|_{x \leftarrow 1}$$

$$f|_{x_i \leftarrow b}(x_1, \dots, x_n) = f(x_1, \dots, x_{i-1}, b, x_{i+1}, \dots, x_n)$$

$$v = \neg x \wedge \text{lo}(v) \vee x \wedge \text{hi}(v)$$



OBDDs as an abstract data type

the order of variables the same in all OBDDs

Operations:

$f \vee g, f \wedge g, \neg f, \text{false}, \text{true}$

$BF \mapsto OBDD$

$f|_{x \leftarrow 0}, f|_{x \leftarrow 1}$

$\exists x. f, \forall x. f$

$QBF \mapsto OBDD$

$f = g$

Note: operations on **functions**, not on values $\{0, 1\}$.

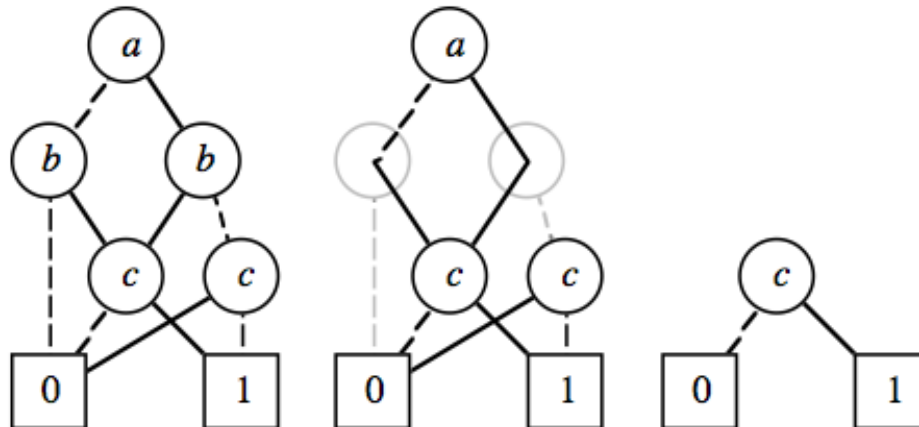
Implementation of unary operations

– $f|_{x \leftarrow b}$

traverse nodes n of OBDD representing f :

- if $x > \text{var}(n)$ then recursively traverse $\text{lo}(n)$ and $\text{hi}(n)$;
- if $x < \text{var}(n)$ then stop;
- otherwise replace n by:
$$\begin{cases} \text{lo}(n) & \text{if } b = 0 \\ \text{hi}(n) & \text{if } b = 1 \end{cases}$$

Example: OBDD for $f|_{b \leftarrow 1}$



[Bryant 1992]

Implementation of unary operations (cont.)

$$- \exists x. f = f|_{x \leftarrow 0} \vee f|_{x \leftarrow 1}$$

the order of variables

remains the same!

$$- \neg f \quad ?$$

Implementation of binary operations

the order of variables the same in all OBDDs

$$\bullet : \{0, 1\}^2 \rightarrow \{0, 1\}$$

$$\begin{aligned} f \bullet g &= \neg x \wedge (f \bullet g)|_{x \leftarrow 0} \quad \vee \quad x \wedge (f \bullet g)|_{x \leftarrow 1} \\ f \bullet g &= \neg x \wedge (f|_{x \leftarrow 0} \bullet g|_{x \leftarrow 0}) \quad \vee \quad x \wedge (f|_{x \leftarrow 1} \bullet g|_{x \leftarrow 1}) \end{aligned}$$

Apply(f, g, \bullet)

(think of f, g as roots of OBDDs)

– f, g leaves: $\text{val}(f \bullet g) = \text{val}(f) \bullet \text{val}(g)$

– f leaf, g not: $f \bullet g = \text{op}(g)$

– $\text{var}(f) = \text{var}(g) = x$:

$$\text{lo}(f \bullet g) = \text{lo}(f) \bullet \text{lo}(g)$$

$$\text{hi}(f \bullet g) = \text{hi}(f) \bullet \text{hi}(g)$$

– $\text{var}(f) = x < y = \text{var}(g)$:

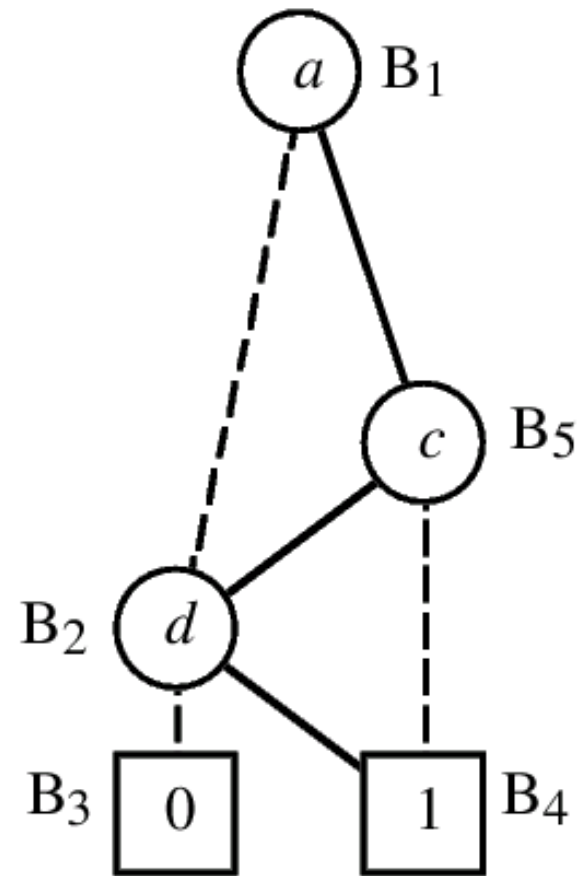
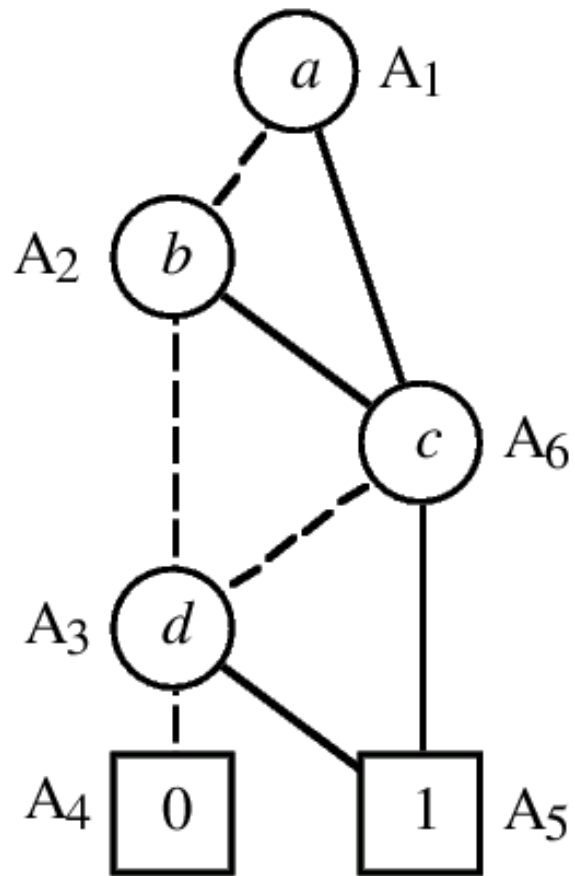
$$\text{lo}(f \bullet g) = \text{lo}(f) \bullet g$$

$$\text{hi}(f \bullet g) = \text{hi}(f) \bullet g$$

$$f \bullet g = \neg x \wedge (f \bullet g)|_{x \leftarrow 0} \quad \vee \quad x \wedge (f \bullet g)|_{x \leftarrow 1}$$

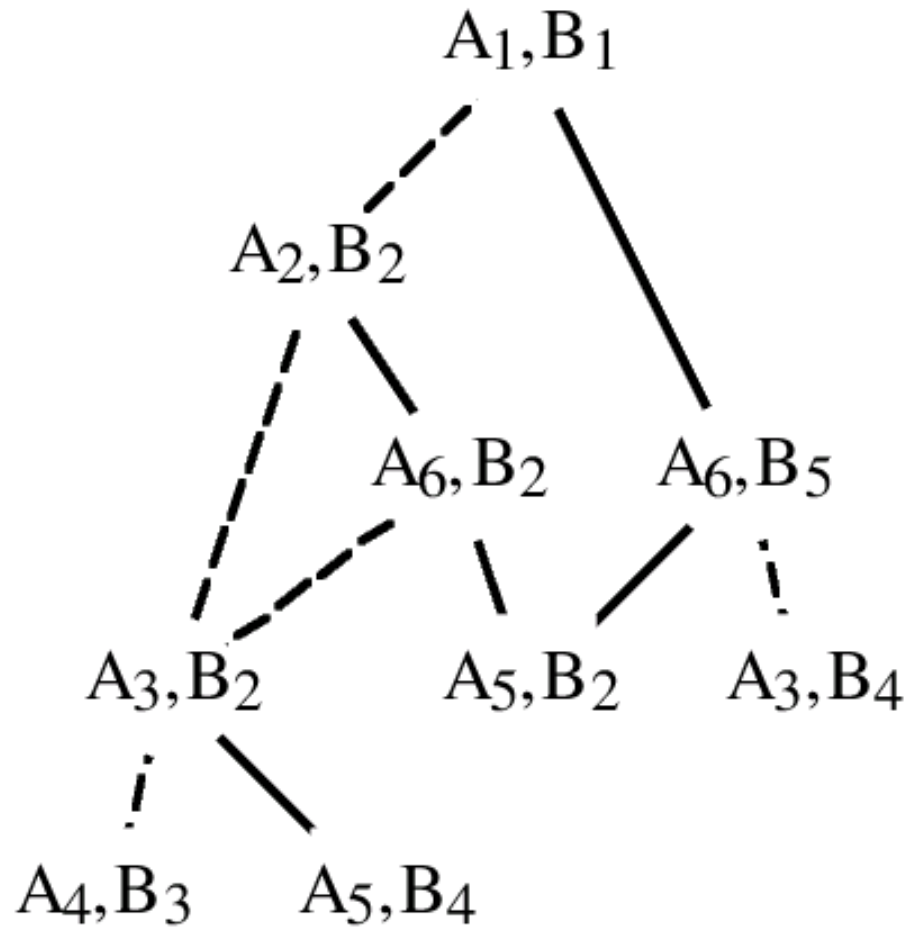
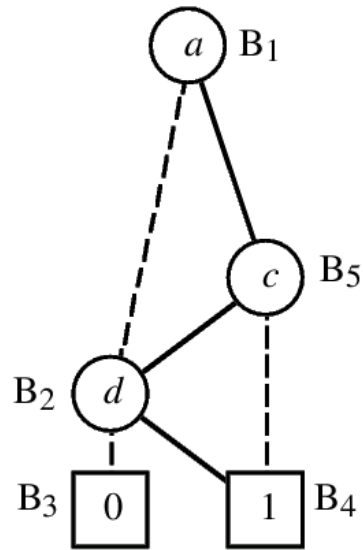
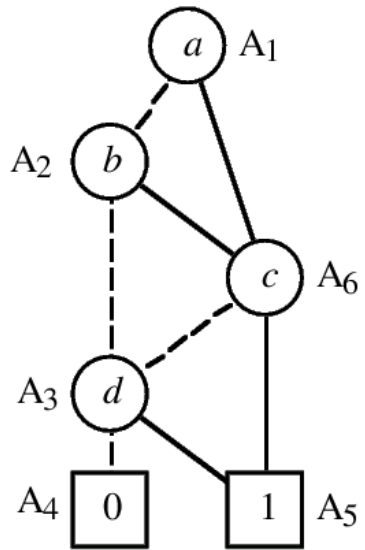
$$f \bullet g = \neg x \wedge (f|_{x \leftarrow 0} \bullet g|_{x \leftarrow 0}) \quad \vee \quad x \wedge (f|_{x \leftarrow 1} \bullet g|_{x \leftarrow 1})$$

Example: input OBDDs



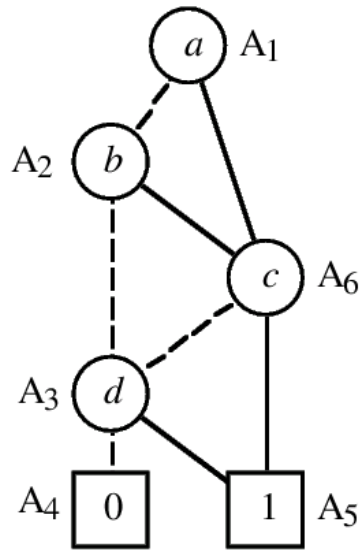
[Bryant 1992]

Example: recursive calls

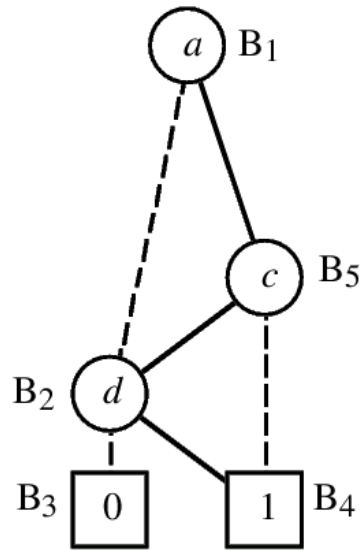


[Bryant 1992]

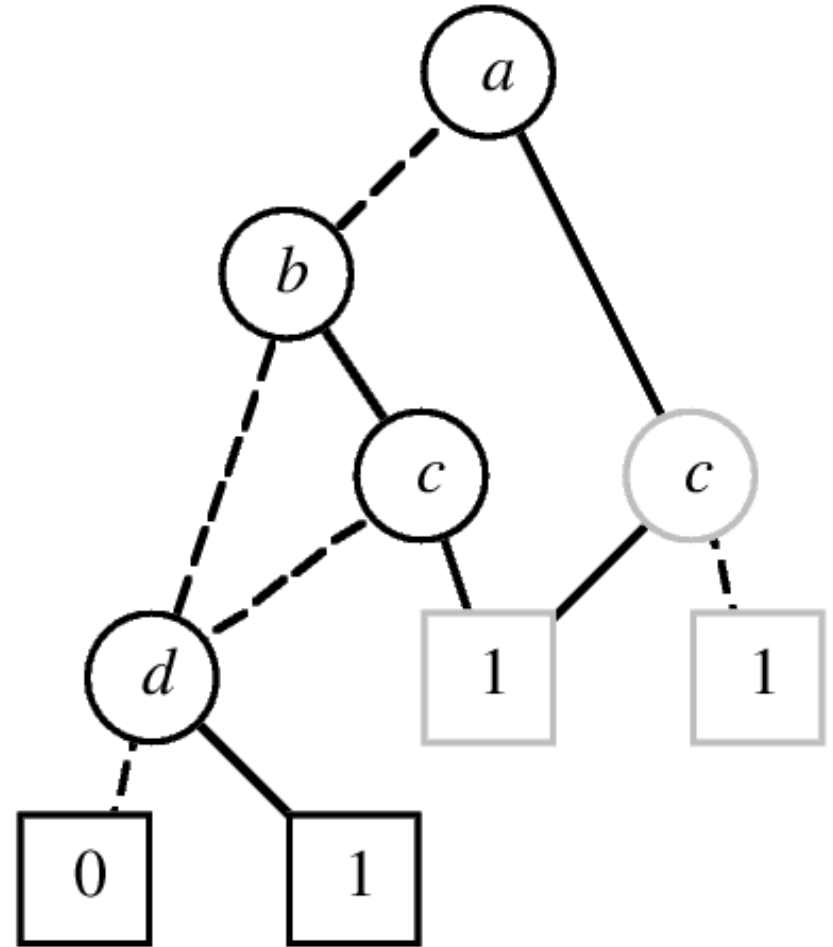
Example: result = $a \vee (b \wedge c) \vee d$



$$((a \vee b) \wedge c) \vee d$$

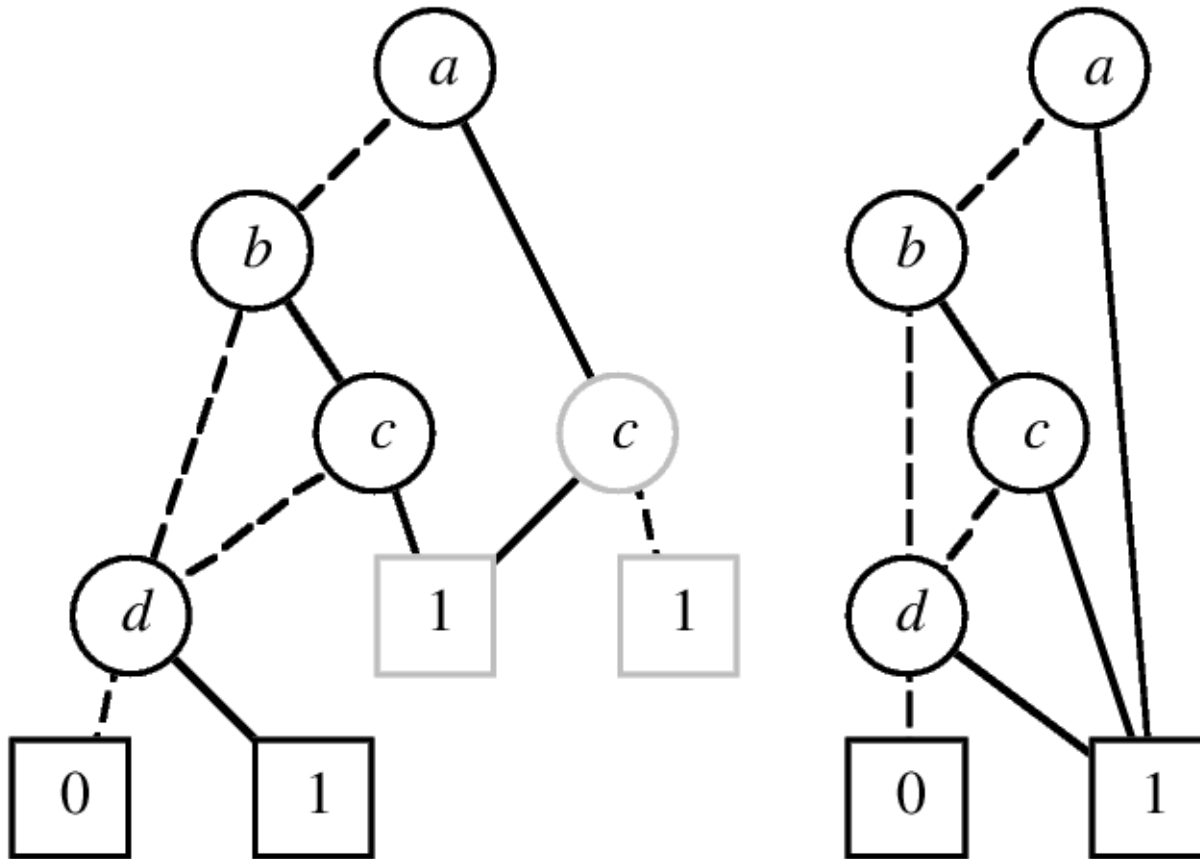


$$(a \wedge \neg c) \vee d$$



[Bryant 1992]

Example: reduced result = $a \vee (b \wedge c) \vee d$



[Bryant 1992]

Implementation of binary operations

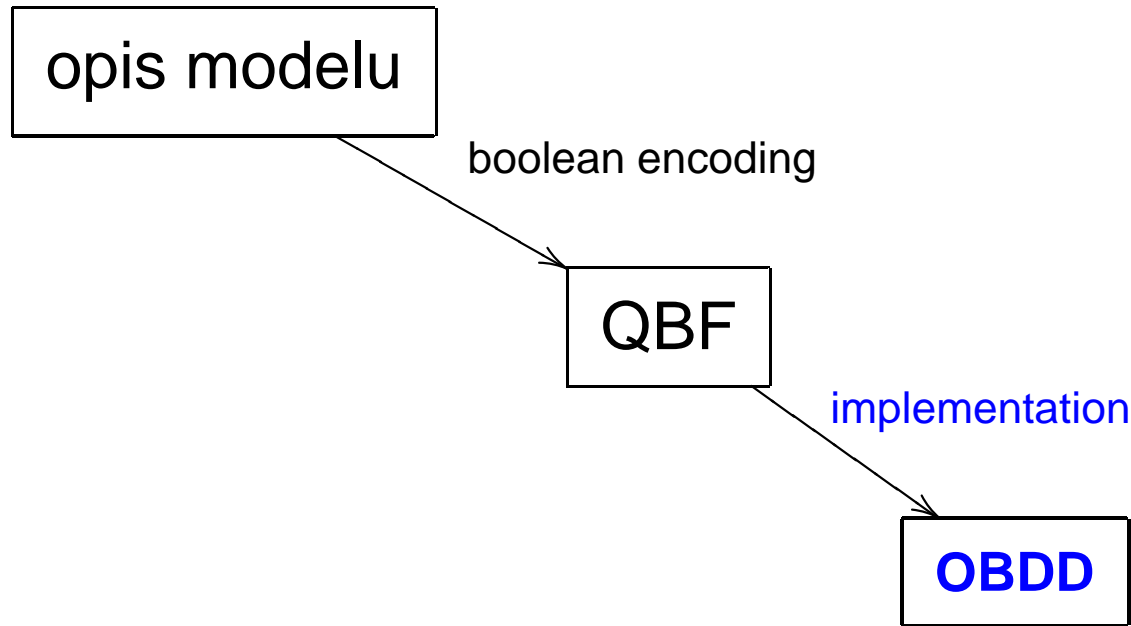
Apply(f, g, \bullet)

- running time: $\mathcal{O}(|f| \cdot |g|)$
- result in the canonical form

Question: $f \iff g$? $f = g$?

- one OBDD shared by all functions
 - = in constant time
- edges representing \neg
- ...

II. Boolean encoding



model checking = operations on OBDDs

– S described by m boolean variables: $S \equiv \{0, 1\}^m$

– transition relation $R : \{0, 1\}^m \times \{0, 1\}^m \rightarrow \{0, 1\}$

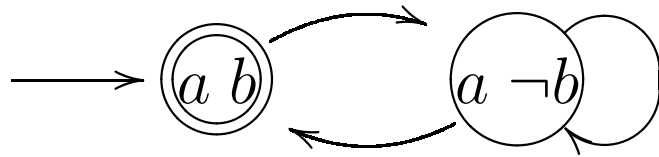
$$R(x_1, \dots, x_m, x'_1, \dots, x'_m) \in \{0, 1\}$$

– initial states $S_0 : \{0, 1\}^m \rightarrow \{0, 1\}$

$$S_0(x_1, \dots, x_m) \in \{0, 1\}$$

– atomic properties $L_p = \{s \mid p \in L(s)\} : \{0, 1\}^m \rightarrow \{0, 1\}$

$$L_p(x_1, \dots, x_m) \in \{0, 1\}$$



(reachable states)

$$R = (x \wedge \neg x') \vee (\neg x \wedge \neg x') \vee (\neg x \wedge x')$$

$$S_0 = x$$

$$L_b = x$$

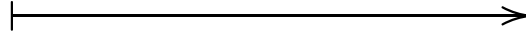
$$L_a = 1$$

From a model to OBDD

description of a Kripke structure



Kripke structure



OBDD

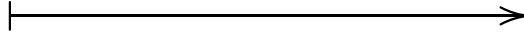
NO!

From a model to OBDD

description of a Kripke structure



Kripke structure



OBDD

NO!

description of a Kripke structure



OBDD

YES!

Compositional model description

Synchronous processes:

$$R = R_1 \wedge R_2 \wedge \dots \wedge R_n$$

Asynchronous processes (interleaving):

$$R = R'_1 \vee R'_2 \vee \dots \vee R'_n$$

$$R'_i = R_i \wedge (\bigwedge_{j \neq i} \text{Id}_j)$$

Asynchronous processes (simultaneous execution):

$$R = R'_1 \wedge R'_2 \wedge \dots \wedge R'_n$$

$$R'_i = R_i \vee \text{Id}_i$$

Restriction to reachable states

$$\widehat{R}(x_1, \dots, x_m, x'_1, \dots, x'_m) = 1$$



$$R(x_1, \dots, x_m, x'_1, \dots, x'_m) \wedge$$

$(x_1, \dots, x_m), (x'_1, \dots, x'_m)$ **reachable**

Restriction to reachable states

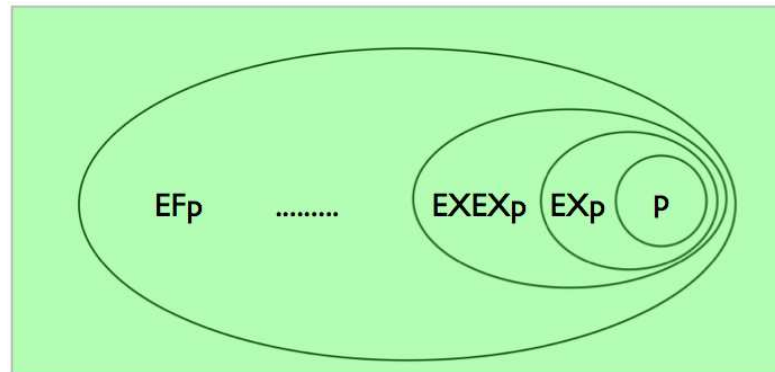
$$\widehat{R}(x_1, \dots, x_m, x'_1, \dots, x'_m) = 1$$



$$R(x_1, \dots, x_m, x'_1, \dots, x'_m) \wedge$$

(x_1, \dots, x_m) reachable

III. Symbolic verification



Symbolic model checking

CTL (\neg , \wedge , **EX**, **E_U_**, **EG**) (these connectives are sufficient)

Check : CTL \mapsto OBDD

Check(ϕ) represents $\{s \mid s \models \phi\}$

Example: Check(p) represents L_p