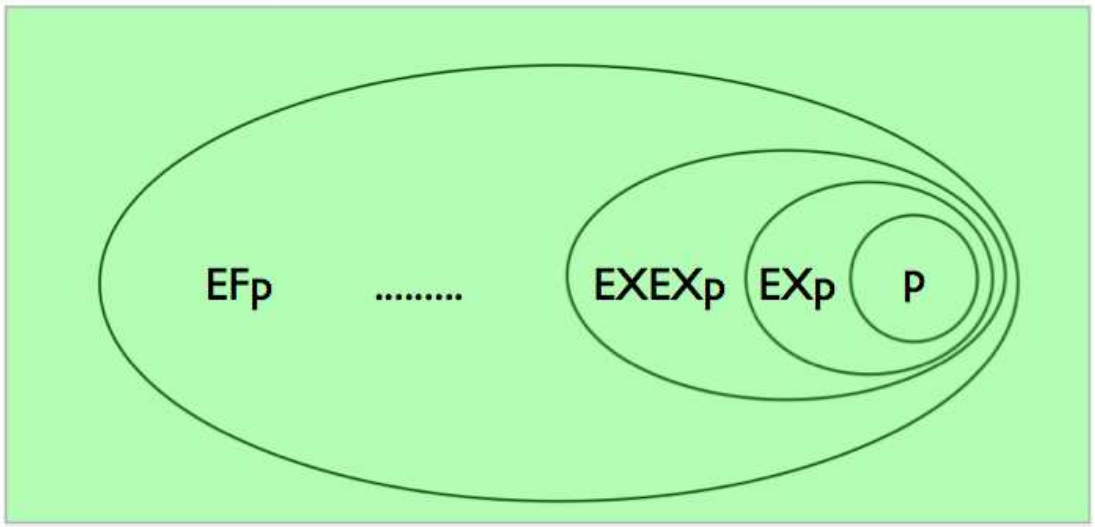
Computer aided verification

Lecture 6:

CTL symbolic model-checking

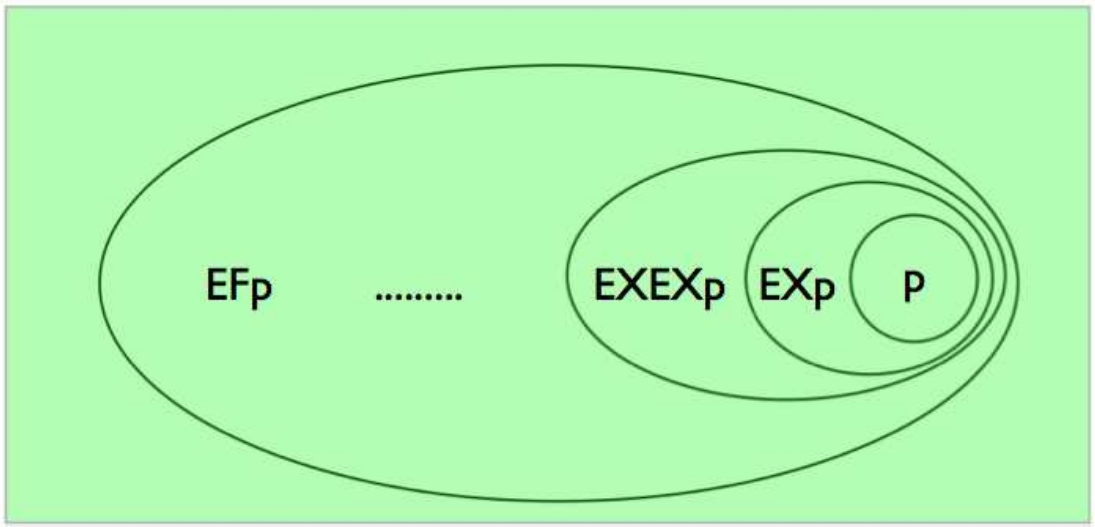
Sławomir Lasota
University of Warsaw

Idea of SMC: EF p



– Compute the set of all states satisfying EF p by iterating predecessor EX _.

Idea of SMC: EF p

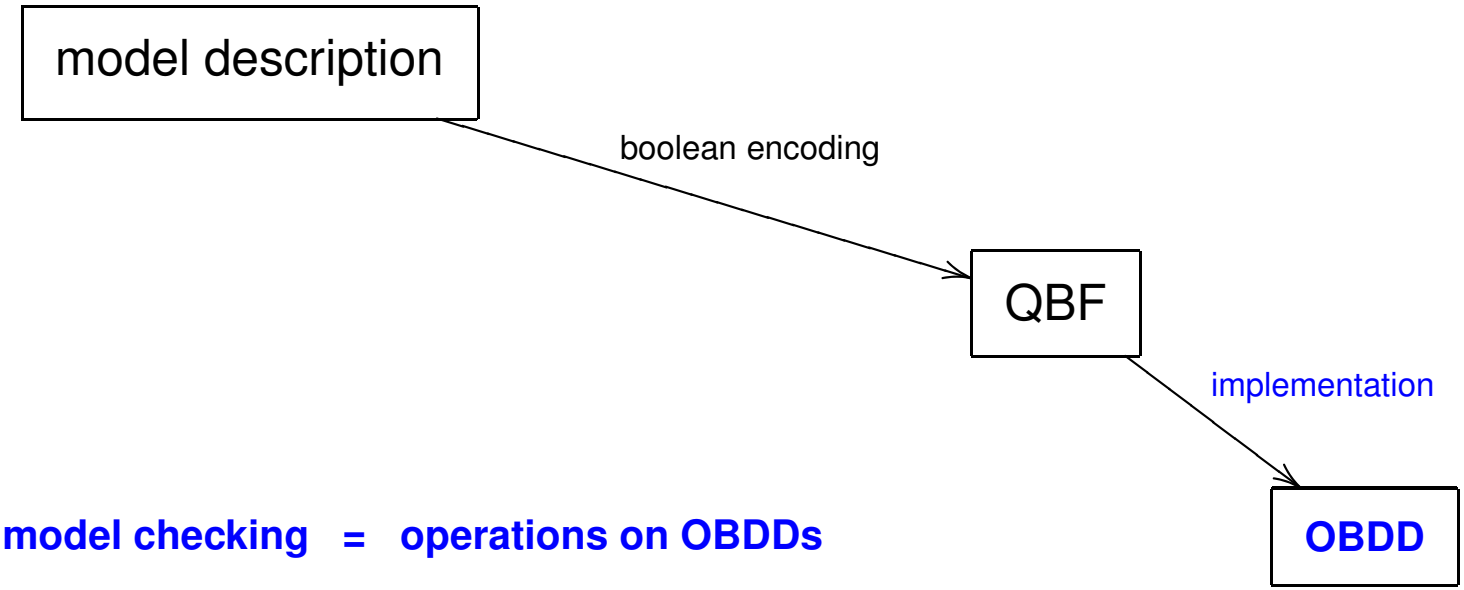


– Compute the set of all states satisfying EF p by iterating predecessor EX _.

– Instead of enumerating all states individually...

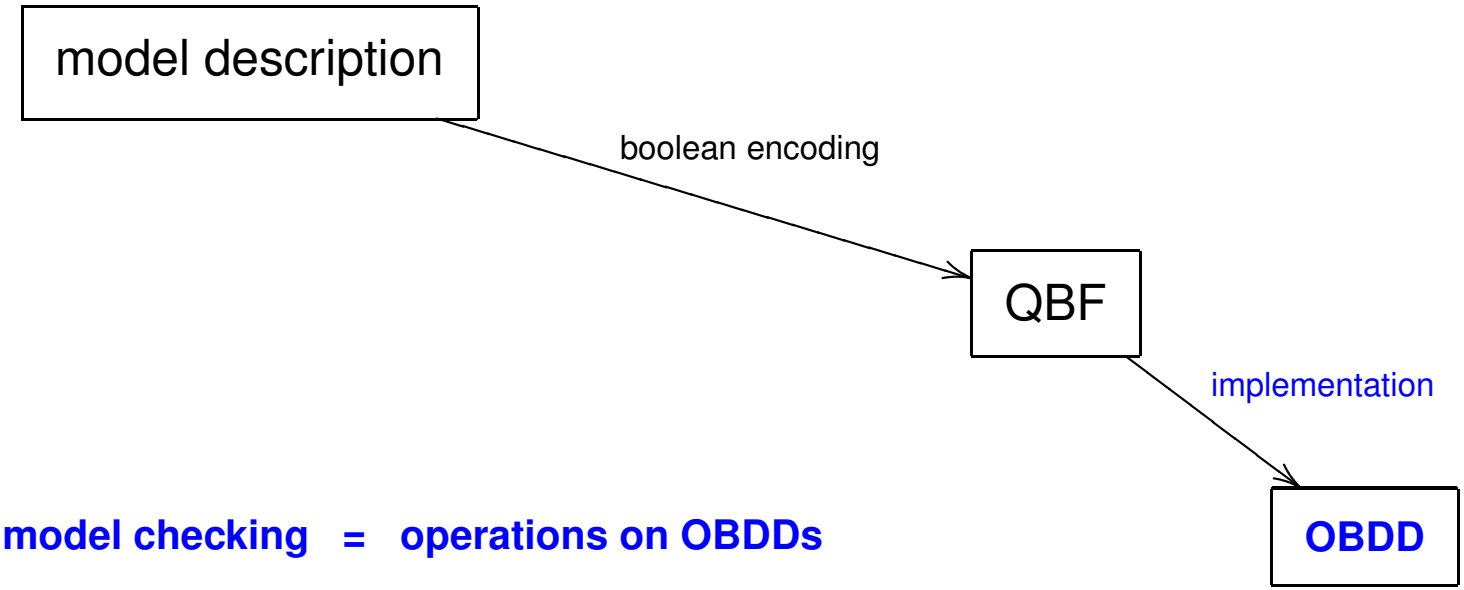
compute **symbolic description** of the set

Symbolic model checking



model checking = operations on OBDDs

Symbolic model checking

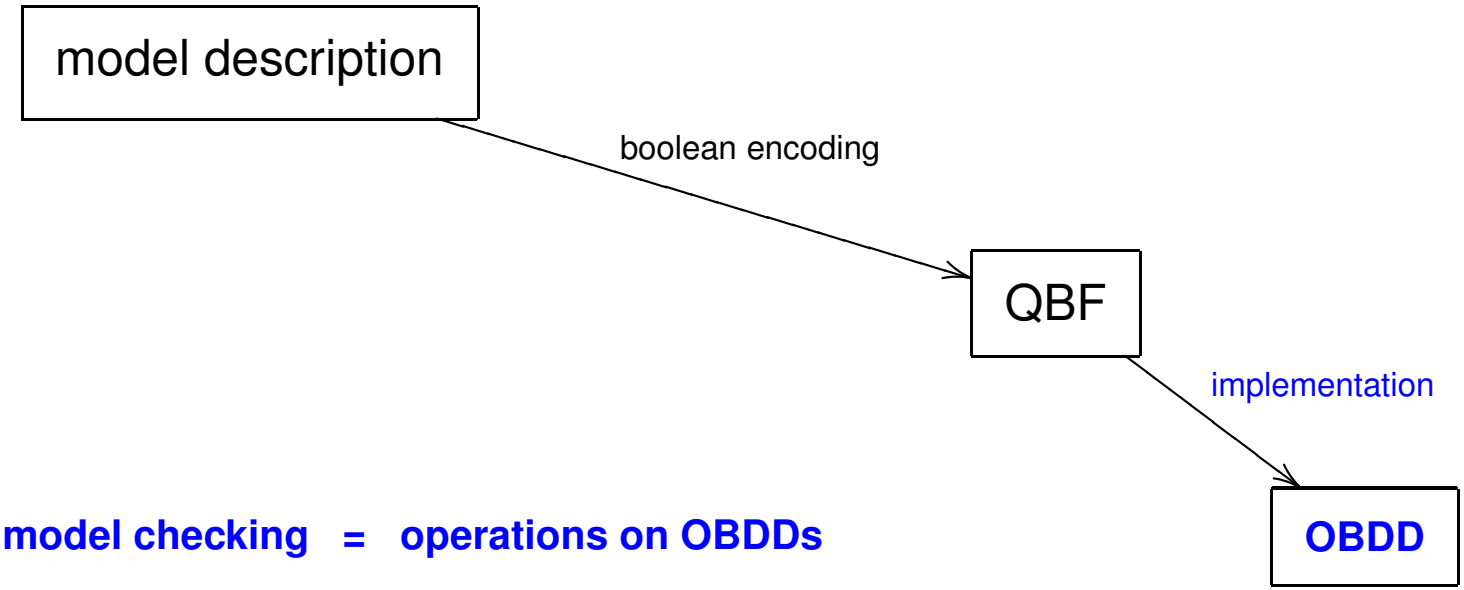


model checking = operations on OBDDs

Plan:

- OBDDs

Symbolic model checking

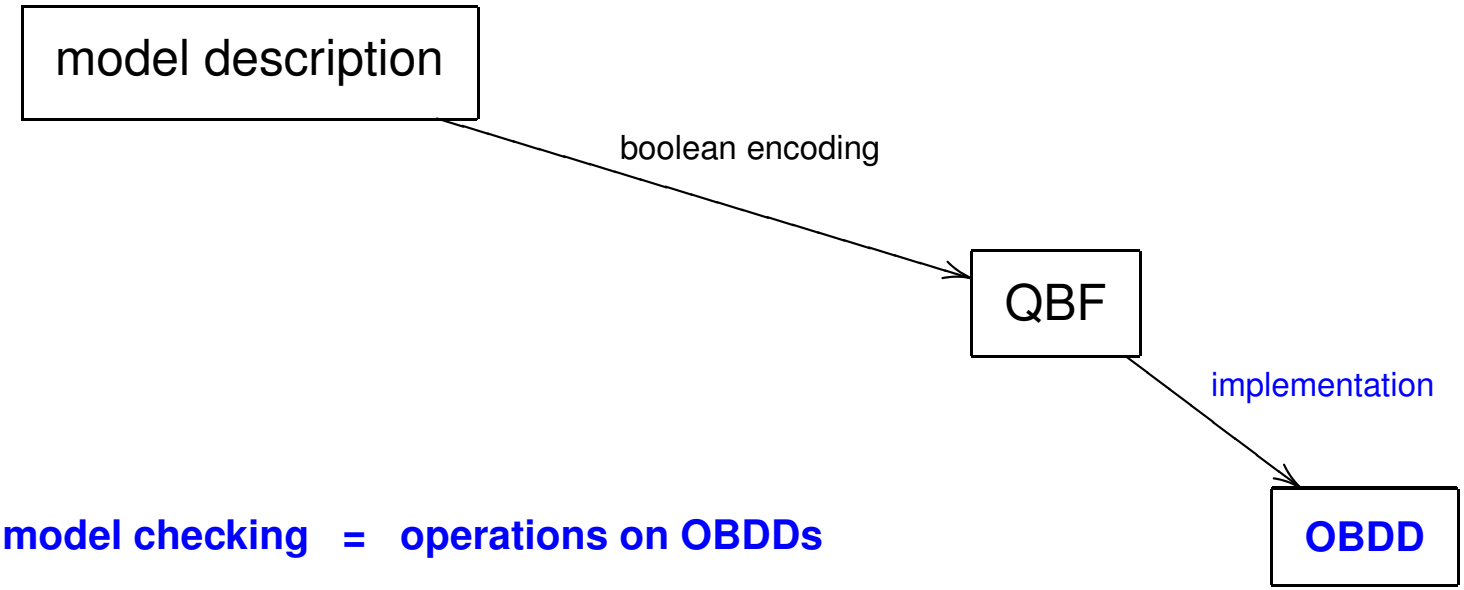


model checking = operations on OBDDs

Plan:

- OBDDs
- Boolean encoding

Symbolic model checking

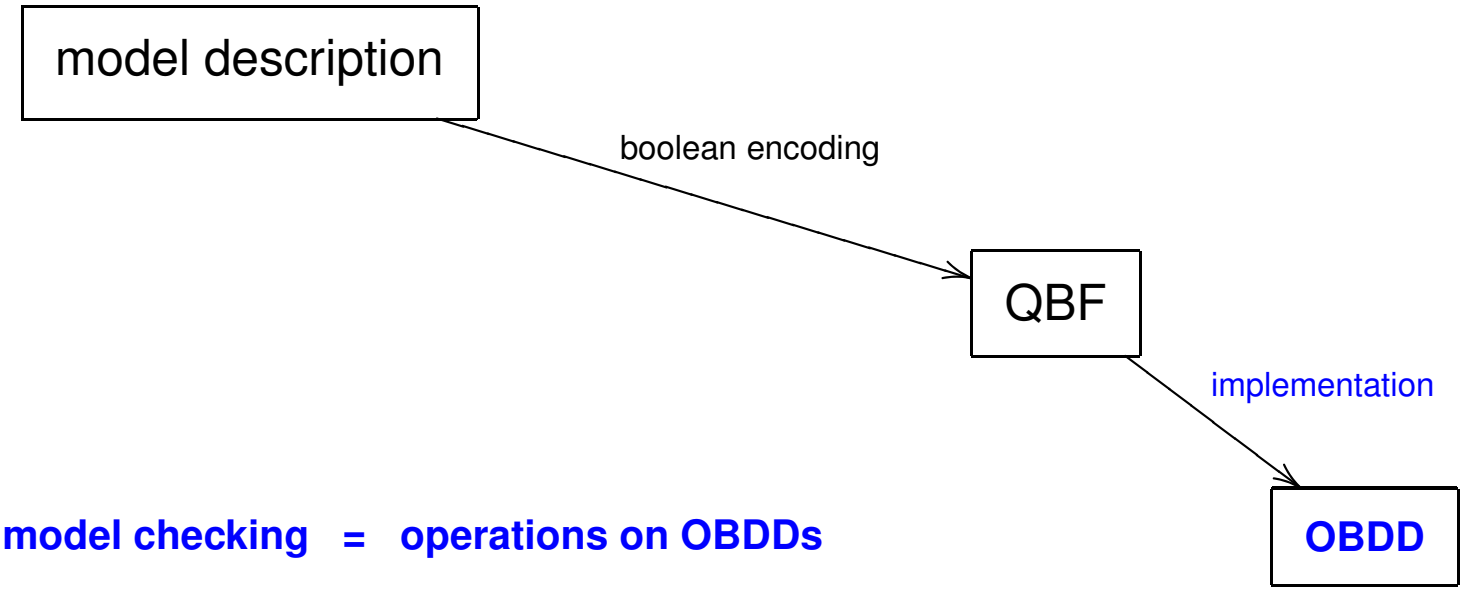


model checking = operations on OBDDs

Plan:

- OBDDs
- Boolean encoding
- CTL Symbolic model-checking

Symbolic model checking

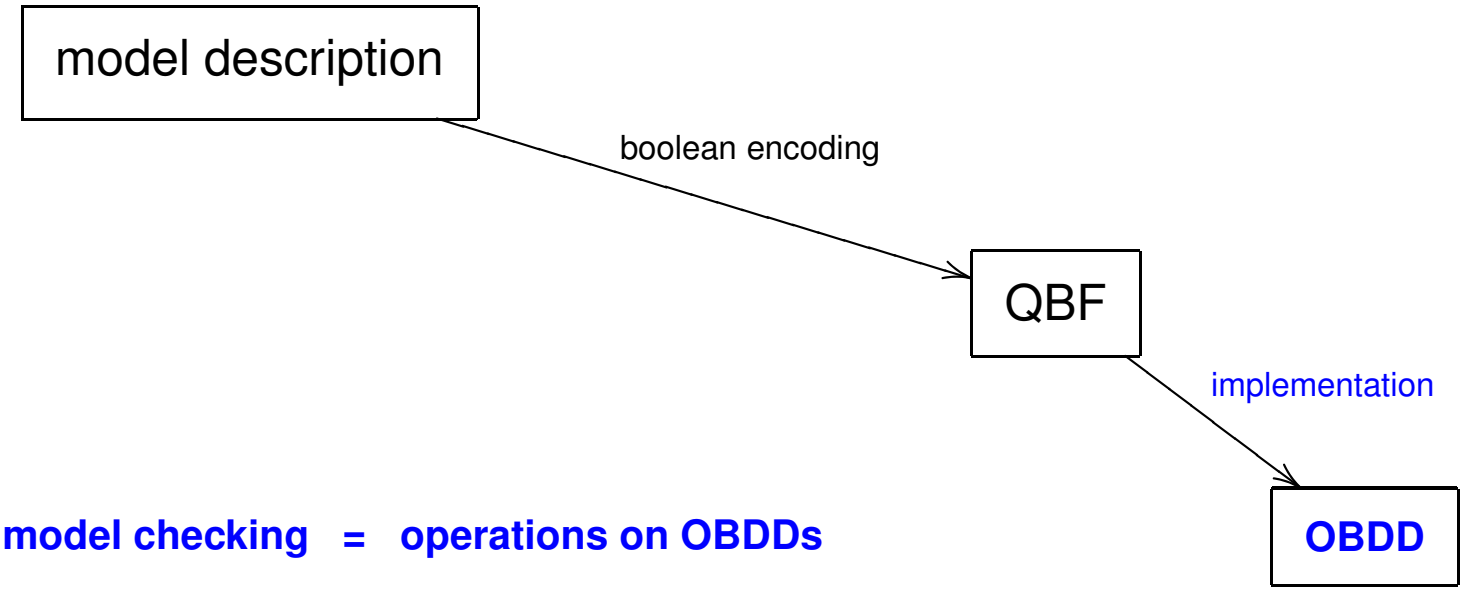


model checking = operations on OBDDs

Plan:

- OBDDs
- Boolean encoding
- CTL Symbolic model-checking
- Fairness

Symbolic model checking



model checking = operations on OBDDs

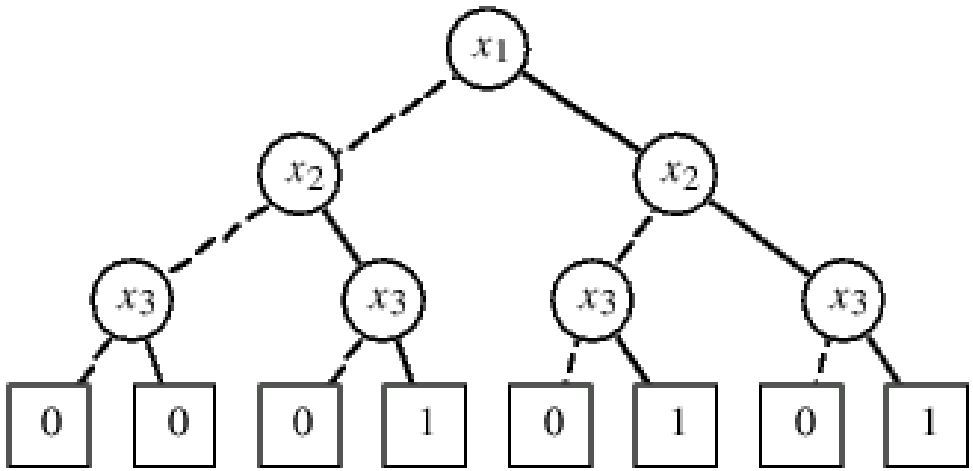
Plan:

- OBDDs
- Boolean encoding
- CTL Symbolic model-checking
- Fairness
- How to compute EX f ?

OBDDs

Ordered Binary Decision Trees

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]

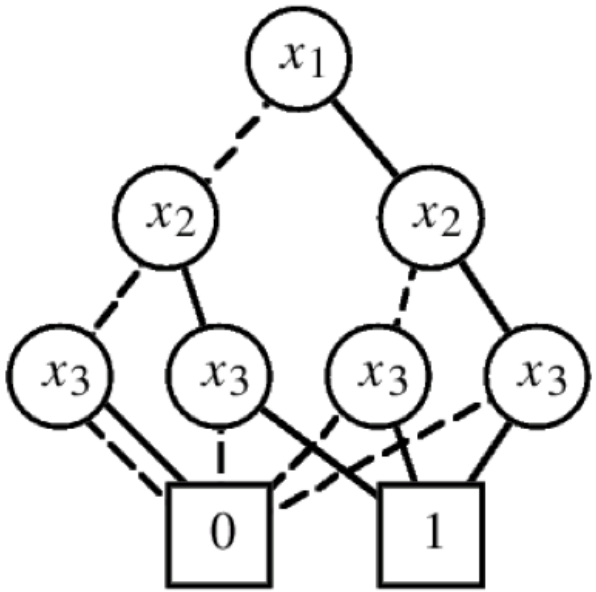
- $f : \{0,1\}^3 \rightarrow \{0,1\}$
- $Z \subseteq \{0,1\}^3$
- fixed order on variables: $x_1 < x_2 < x_3$

Ordered Binary Decision Diagrams

OBDD = rooted directed acyclic graph

Attributes of a node v:

- when v is a leaf
  - a value $\mathrm{val}(v) \in \{0, 1\}$
- when v is not a leaf
  - a variable $\mathrm{var}(v) \in \{x_1, x_2, \ldots\}$
  - 2 successor nodes $s_0(v), s_1(v)$



[Bryant 1992]

The order of variables must be obeyed on every path.

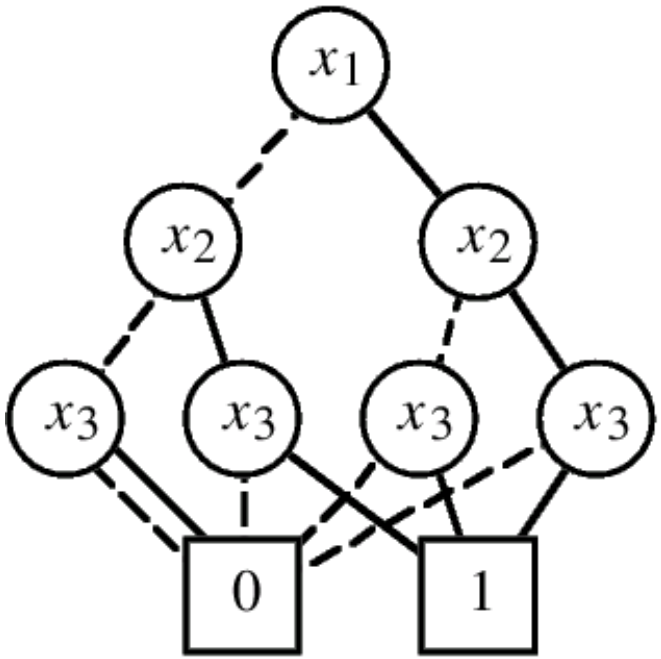
Reduced OBDD

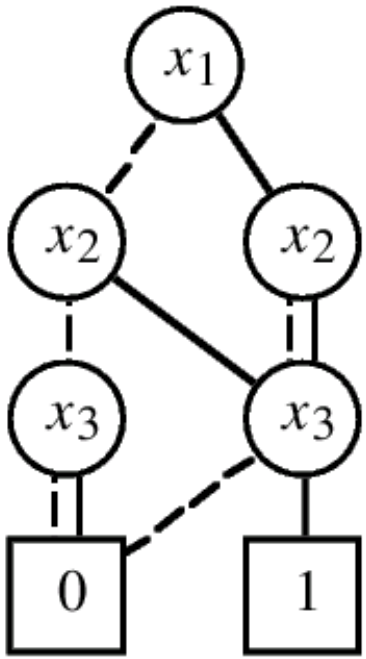
– remove redundant leaves

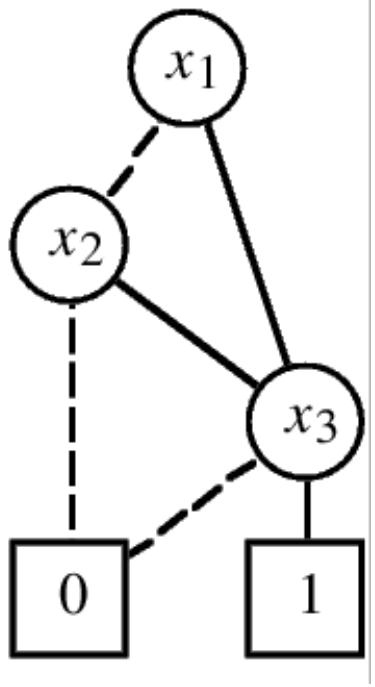
– remove redundant non-leaves

– remove redundant tests

OBDD $\longmapsto$ ROBDD







[Bryant 1992]

Reduced OBDD = canonical form

Canonical form for a boolean function:

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

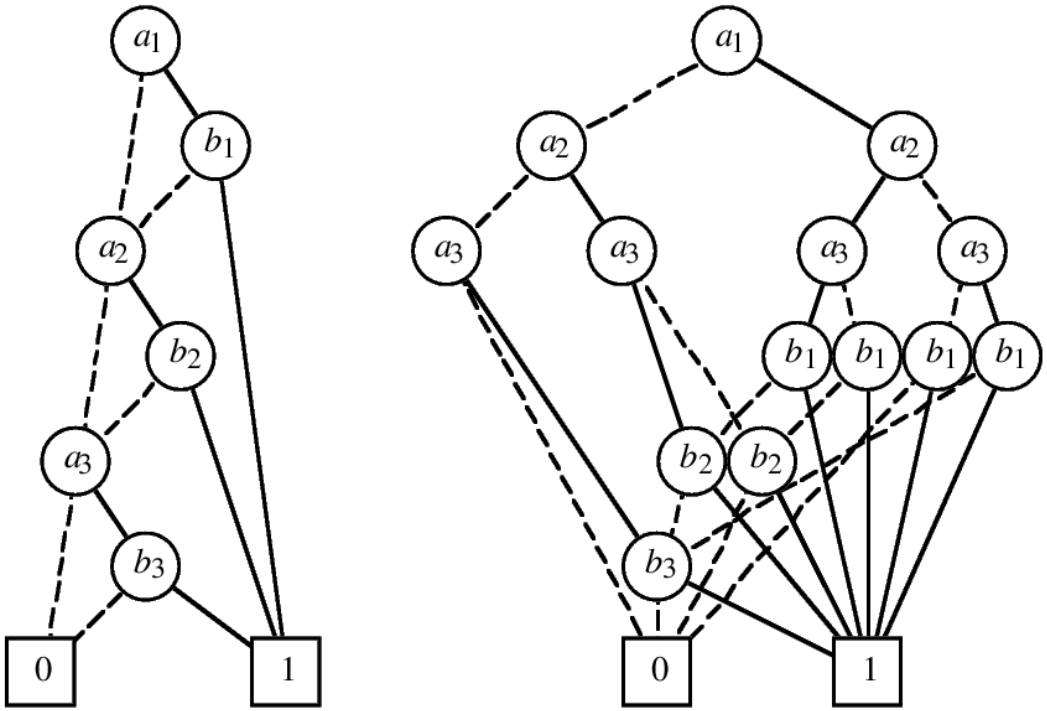
Naive construction of an OBDD for a boolean formula ϕ:

$$\phi \longmapsto \text{decision tree} \longmapsto \text{canonical OBDD}$$

Finding an appropriate order of variables is crucial!

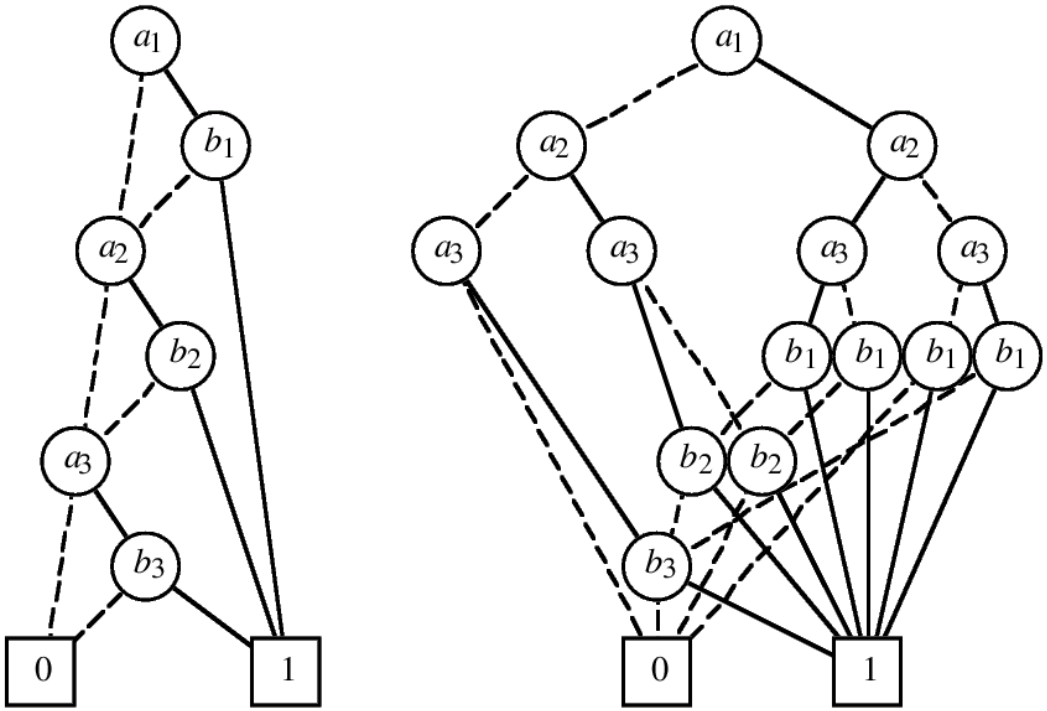
Order of variables is crucial

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



Order of variables is crucial

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



$$a_1 \wedge b_1 \ \vee \ a_2 \wedge b_2 \ \vee \ \ldots \ \vee \ a_n \wedge b_n$$

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

OBDD size

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 (oldest bits)	$\mathcal{O}(n)$	$\mathcal{O}(2^n)$
multiplication (middle bits)	$\mathcal{O}(2^n)$	$\mathcal{O}(2^n)$

Shannon's expansion

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

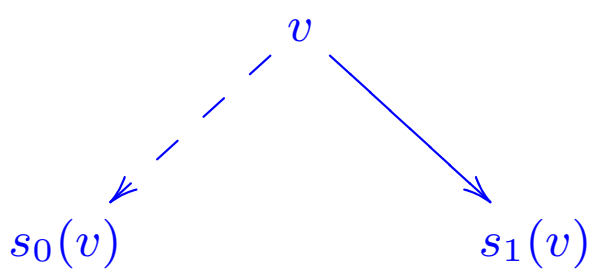
$$f\big|_{x_i \leftarrow b}(x_1, \ldots, x_n) \;=\; f(x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n)$$

Shannon's expansion

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

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

$$v \;=\; \neg x \wedge s_0(v) \;\vee\; x \wedge s_1(v)$$



OBDDs as an abstract data type

the order of variables the same in all OBDDs

OBDDs as an abstract data type

the order of variables the same in all OBDDs

Operations:

$f \vee g$, $f \wedge g$, $\neg f$, `false`, `true` $\qquad BF \mapsto OBDD$

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

OBDDs as an abstract data type

the order of variables the same in all OBDDs

Operations:

$f \vee g$, $f \wedge g$, $\neg f$, `false`, `true` $\quad BF \mapsto OBDD$

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

$\exists x.\ f$, $\forall x.\ f$ $\quad QBF \mapsto OBDD$

OBDDs as an abstract data type

the order of variables the same in all OBDDs

Operations:

$f \vee g$, $f \wedge g$, $\neg f$, `false`, `true` $\quad BF \mapsto OBDD$

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

$\exists x.\ f$, $\forall x.\ f$ $\quad QBF \mapsto OBDD$

$f = g$

OBDDs as an abstract data type

the order of variables the same in all OBDDs

Operations:

$f \vee g$, $f \wedge g$, $\neg f$, `false`, `true` $\qquad BF \mapsto OBDD$

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

$\exists x.\ f$, $\forall x.\ f$ $\qquad QBF \mapsto OBDD$

$f = g$

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

Implementation of unary operations

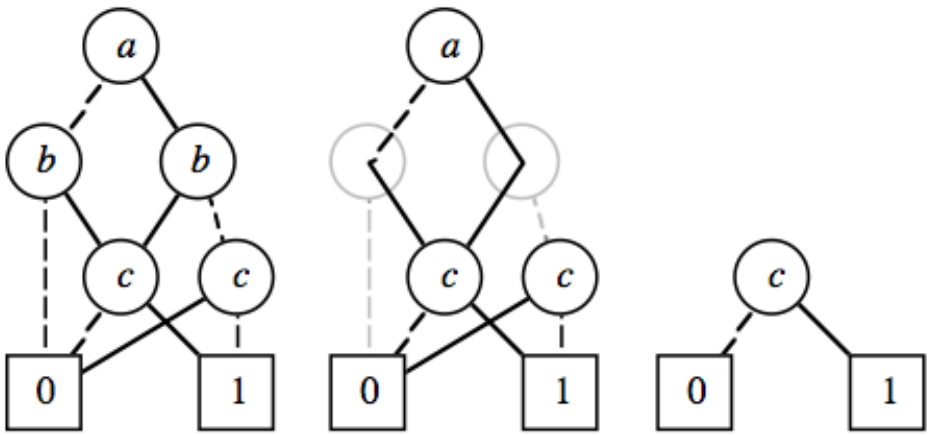
– $f|_{x \leftarrow b}$

traverse nodes n of OBDD representing f:

– if $x > \mathrm{var}(n)$ then recursively traverse $s_0(n)$ and $s_1(n)$;

– if $x < \mathrm{var}(n)$ then stop;

– otherwise replace n by: $\begin{cases} s_0(n) & \text{if } b = 0 \\ s_1(n) & \text{if } b = 1 \end{cases}$



[Bryant 1992]

Implementation of unary operations (cont.)

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

the order of variables

remains the same!

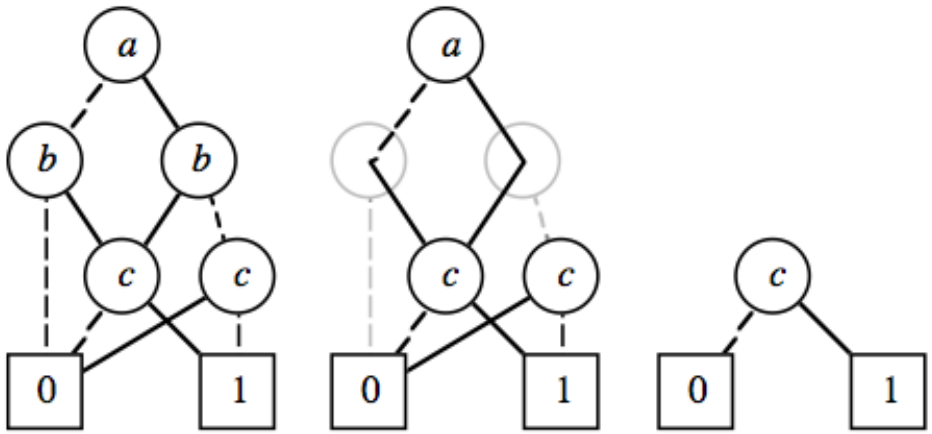
Implementation of unary operations (cont.)

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

the order of variables

remains the same!

– $\neg f$ **?**



[Bryant 1992]

Implementation of binary operations

the order of variables the same in all OBDDs

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

for two given OBDDs representing f and g,
compute an OBDD representing $f \bullet g$

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

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

Implementation of binary operations

$\mathrm{Apply}(\ f,\ g,\ \bullet\ ) \rightsquigarrow\ f \bullet g$ (think of f, g as roots of OBDDs)

- f, g leaves: $\mathrm{val}(f \bullet g) := \mathrm{val}(f) \bullet \mathrm{val}(g)$
- f leaf, g not: $f \bullet g := \mathit{op}(g)$
- $\mathrm{var}(f) = \mathrm{var}(g) = x$:

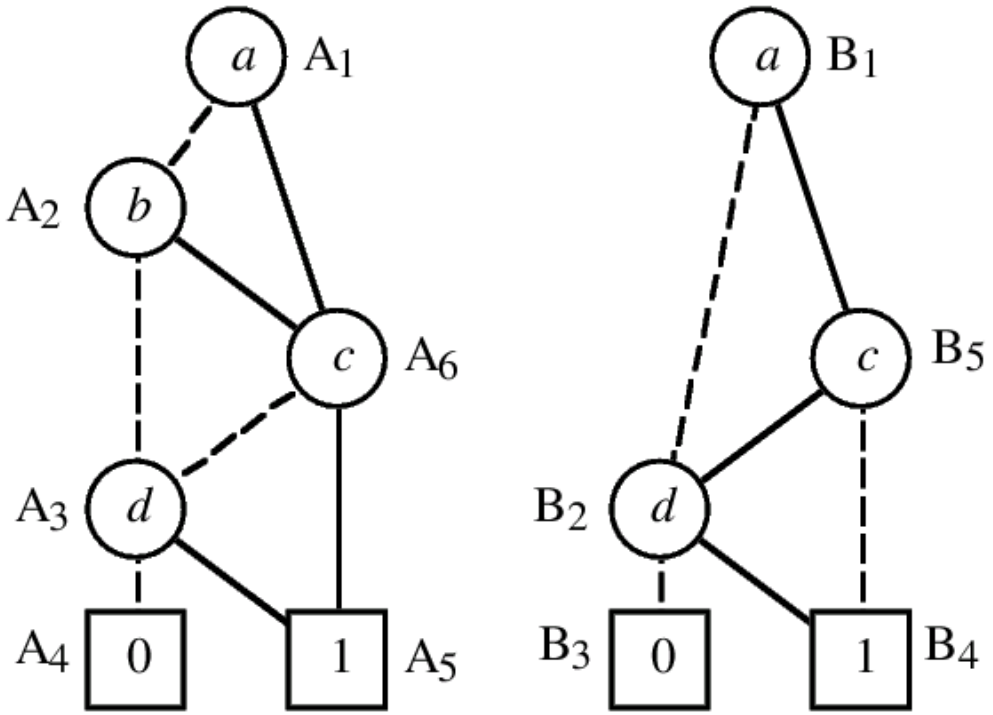
  $\mathrm{var}(f \bullet g) := x \quad s_0(f \bullet g) := s_0(f) \bullet s_0(g) \quad s_1(f \bullet g) := s_1(f) \bullet s_1(g)$
- $\mathrm{var}(f) = x < y = \mathrm{var}(g)$:

  $\mathrm{var}(f \bullet g) := x \qquad s_0(f \bullet g) := s_0(f) \bullet g \qquad s_1(f \bullet g) := s_1(f) \bullet g$

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

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

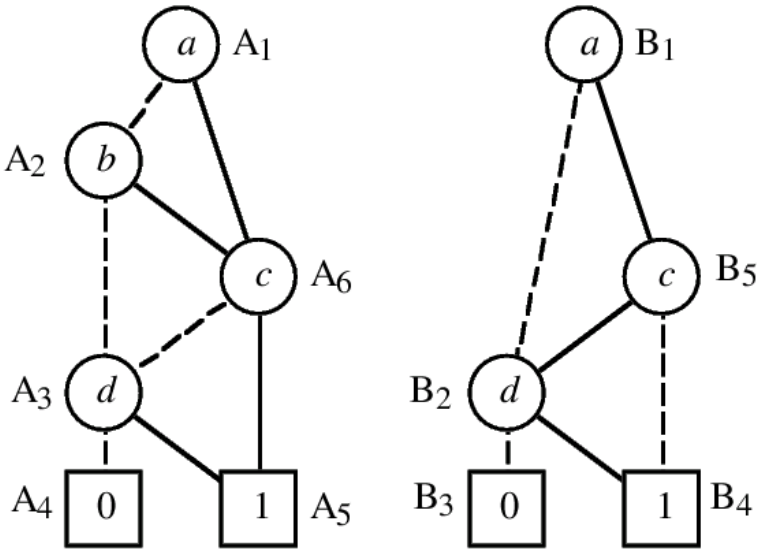
Example: input OBDDs

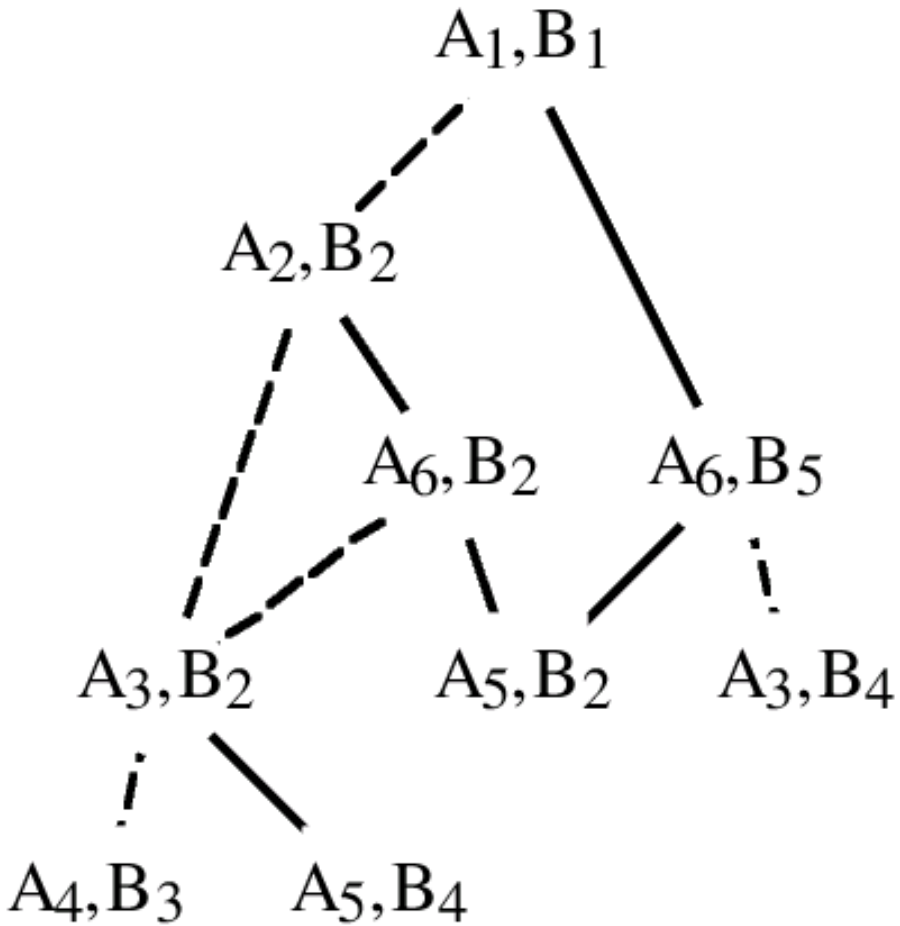


[Bryant 1992]

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

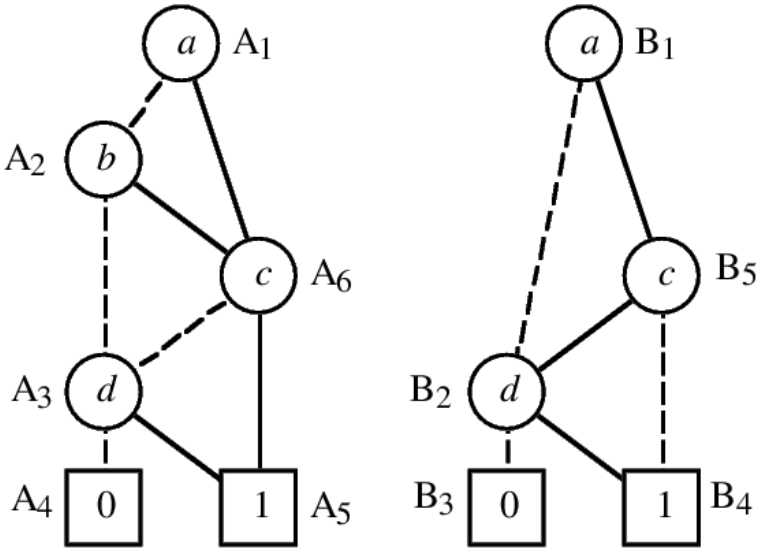
Example: recursive calls

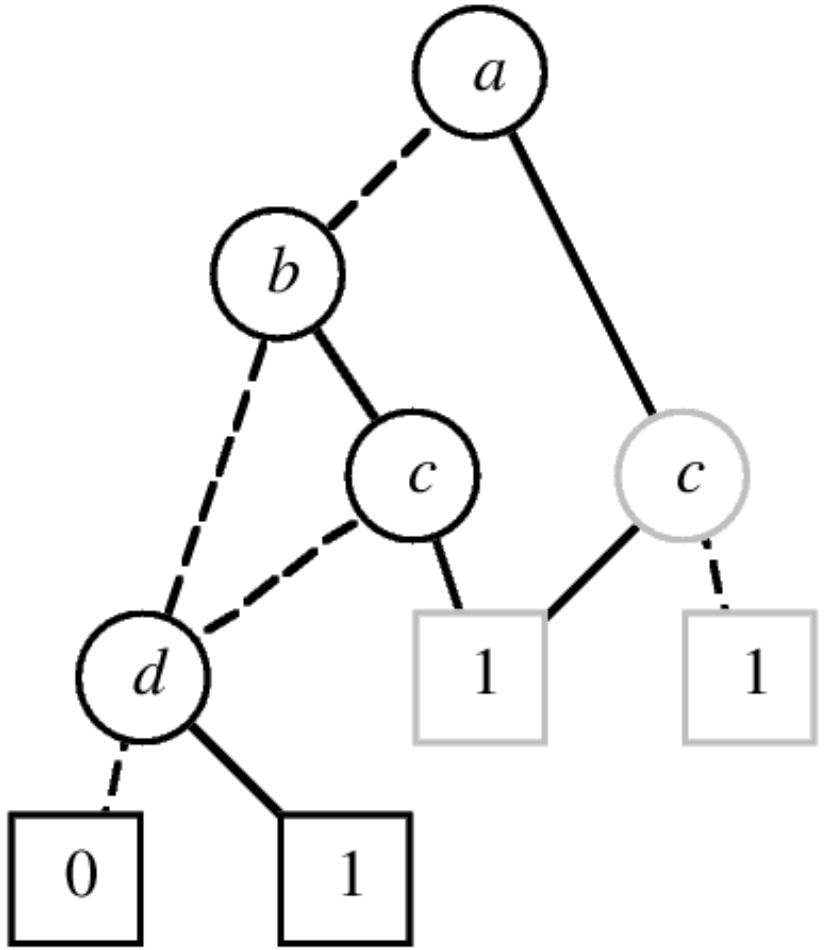




[Bryant 1992]

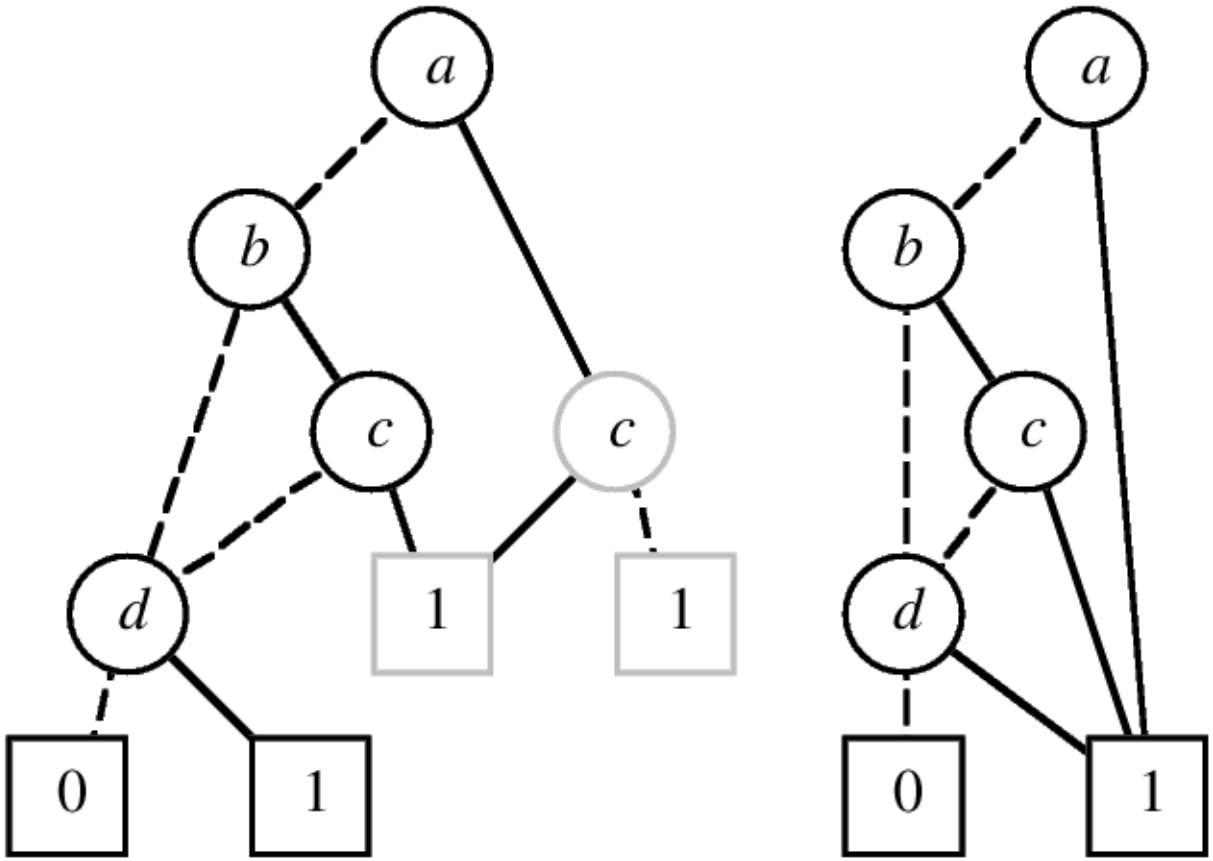
Example: result = $a \vee b \wedge c \vee d$





[Bryant 1992]

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



[Bryant 1992]

Implementation of binary operations

$\mathrm{Apply}(\ f,\ g,\ \bullet\ )$

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

Implementation of binary operations

$\mathrm{Apply}(\ f,\ g,\ \bullet\ )$

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

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

Implementation of binary operations

$\mathrm{Apply}(\ f,\ g,\ \bullet\ )$

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

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

Question: $f = g$ **?**

Variations and extensions

– one OBDD shared by all functions $\mapsto$ $f = g$ in constant time

Variations and extensions

– one OBDD shared by all functions $\mapsto$ $f = g$ in constant time

– edges representing $\neg$

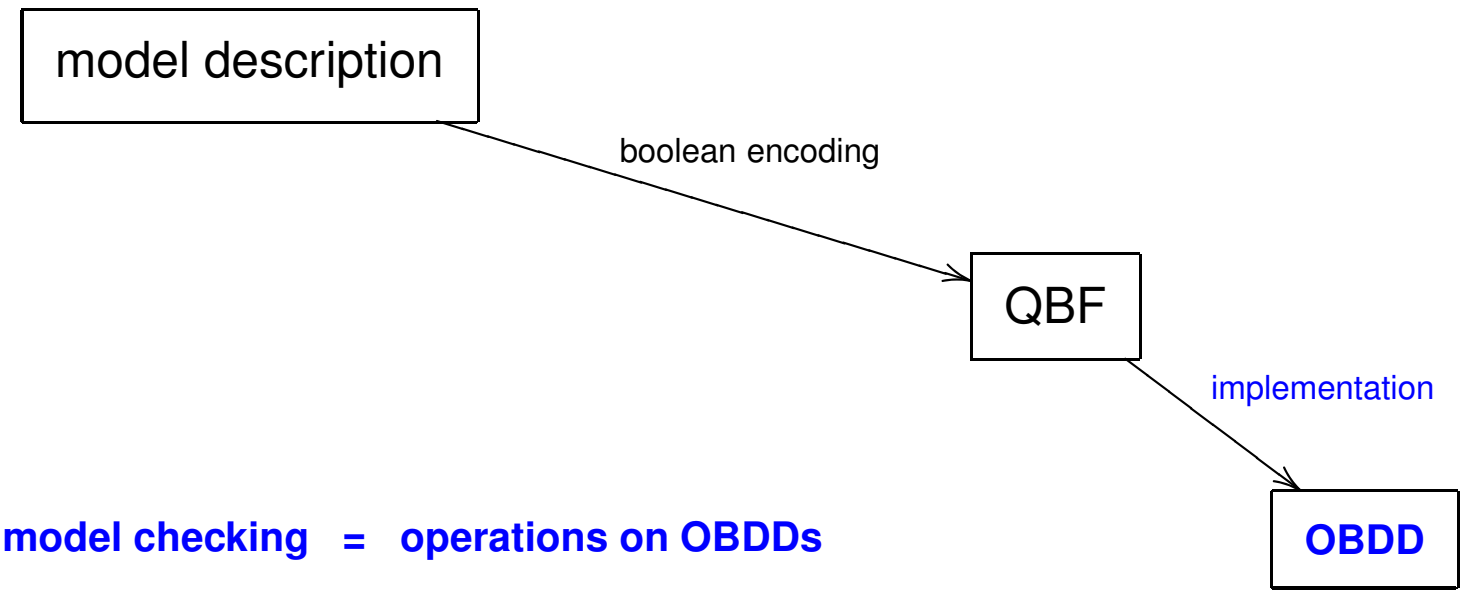
Variations and extensions

- one OBDD shared by all functions $\mapsto$ $f = g$ in constant time
- edges representing $\neg$
- OZBDDs (Ordered Zero-supressed BDDs)

Variations and extensions

- one OBDD shared by all functions $\mapsto$ $f = g$ in constant time
- edges representing $\neg$
- OZBDDs (Ordered Zero-supressed BDDs)
- ...

Boolean encoding



model checking = operations on OBDDs

Model description

– Kripke structure $M = (S, S_0, R, L)$

Model description

– Kripke structure $M = (S, S_0, R, L)$

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

Model description

– Kripke structure $M = (S, S_0, R, L)$

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

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

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

Model description

- Kripke structure $M = (S, S_0, R, L)$
- S described by m boolean variables: $S \equiv \{0,1\}^m$
- transition relation $R : \{0,1\}^m \times \{0,1\}^m \to \{0,1\}$

  $R(x_1, \ldots, x_m, x'_1, \ldots, x'_m) \in \{0,1\}$
- initial states $S_0 : \{0,1\}^m \to \{0,1\}$

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

Model description

– Kripke structure $M = (S, S_0, R, L)$

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

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

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

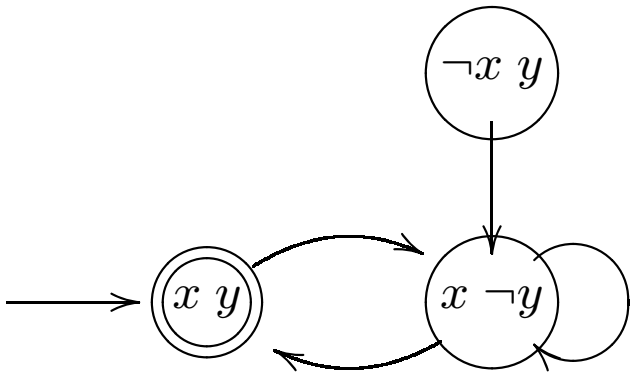
– initial states $S_0 : \{0,1\}^m \to \{0,1\}$

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

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

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

Example



(reachable states)

$$
\begin{aligned}
\text{variables} &= \{x, y\} \\
R &= (x \wedge y \wedge x' \wedge \neg y') \vee (x \wedge \neg y \wedge x' \wedge \neg y') \vee (x \wedge \neg y \wedge x' \wedge y') \\
S_0 &= x \wedge y \\
L_p &= y
\end{aligned}
$$

SMV model – example

```
MODULE main
VAR
  request : {Tr, Fa};
  state : {ready, busy};
ASSIGN
  init(state) := ready;
  next(state) := case
                   state = ready & (request = Tr): busy;
                   TRUE : {ready,busy};
                 esac;
SPEC
  AG((request = Tr) -> AF state = busy)
```

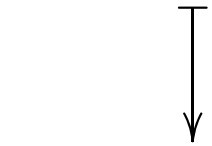
From a model to OBDD

description of a Kripke structure

NO!

From a model to OBDD

description of a Kripke structure



Kripke structure $\longmapsto$ OBDD

NO!

description of a Kripke structure $\longmapsto$ OBDD

YES!

Compositional model desciption

Synchronous processes:

$$R \;=\; R_1 \;\wedge\; R_2 \wedge \;\ldots\; \wedge\; R_n$$

Compositional model desciption

Synchronous processes:

$$R \;=\; R_1 \;\wedge\; R_2 \wedge \;\ldots\; \wedge\; R_n$$

Asynchronous processes (interleaving):

$$R \;=\; R'_1 \;\vee\; R'_2 \vee \;\ldots\; \vee\; R'_n$$

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

Compositional model desciption

Synchronous processes:

$$R \;=\; R_1 \;\wedge\; R_2 \wedge \;\ldots\; \wedge\; R_n$$

Asynchronous processes (interleaving):

$$R \;=\; R'_1 \;\vee\; R'_2 \vee \;\ldots\; \vee\; R'_n$$

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

Asynchronous processes (simultaneous execution):

$$R \;=\; R'_1 \;\wedge\; R'_2 \wedge \;\ldots\; \wedge\; R'_n$$

$$R'_i \;=\; R_i \vee \mathrm{Id}_i$$

Restriction to reachable states

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

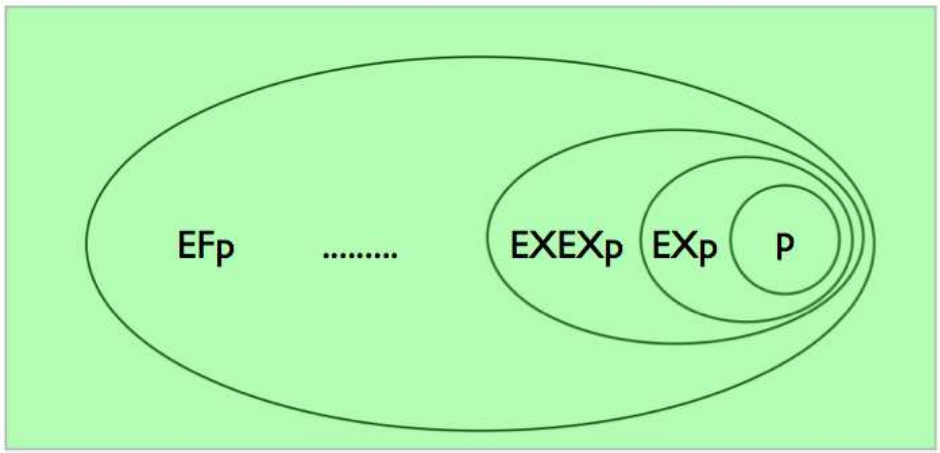
$$\Updownarrow$$

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

$$(x_1, \ldots, x_m) \text{ reachable}$$

Should we restrict to reachable states?

CTL Symbolic model-checking



Reminder

Fixed points in a complete lattice $\langle A, \leq \rangle$. Assume A is finite.

Let $f : A \to A$ monotonic.

Reminder

Fixed points in a complete lattice $\langle A, \leq \rangle$. Assume A is finite.

Let $f : A \to A$ monotonic.

- the least f.p.: $\bot \leq f(\bot) \leq f^2(\bot) \leq \ldots \rightsquigarrow \mu Z.\, f(Z)$
- the greatest f.p.: $\top \geq f(\top) \geq f^2(\top) \geq \ldots \rightsquigarrow \nu Z.\, f(Z)$

When A finite, the fixed points are reached after $\leq |A|$ iterations.

Reminder

Fixed points in a complete lattice $\langle A, \leq \rangle$. Assume A is finite.

Let $f : A \to A$ monotonic.

- the least f.p.: $\bot \leq f(\bot) \leq f^2(\bot) \leq \ldots \rightsquigarrow \mu Z.\, f(Z)$
- the greatest f.p.: $\top \geq f(\top) \geq f^2(\top) \geq \ldots \rightsquigarrow \nu Z.\, f(Z)$

When A finite, the fixed points are reached after $\leq |A|$ iterations.

Example: $\langle A, \leq \rangle = \langle \mathcal{P}(S), \subseteq \rangle$ EX $Z = \{s : R(s, s') \text{ for some } s' \in Z\}$

Reminder

Fixed points in a complete lattice $\langle A, \leq \rangle$. Assume A is finite.

Let $f : A \to A$ monotonic.

- the least f.p.: $\bot \leq f(\bot) \leq f^2(\bot) \leq \ldots \rightsquigarrow \mu Z.\, f(Z)$
- the greatest f.p.: $\top \geq f(\top) \geq f^2(\top) \geq \ldots \rightsquigarrow \nu Z.\, f(Z)$

When A finite, the fixed points are reached after $\leq |A|$ iterations.

Example: $\langle A, \leq \rangle = \langle \mathcal{P}(S), \subseteq \rangle$ $\quad$ EX $Z = \{s : R(s, s') \text{ for some } s' \in Z\}$

$Z \mapsto$ EX Z $\quad$ $\mu Z.$ EX $Z = \bot = \emptyset$ $\quad$ $\nu Z.$ EX Z = **?**

Reminder

Fixed points in a complete lattice $\langle A, \leq \rangle$. Assume A is finite.

Let $f : A \to A$ monotonic.

- the least f.p.: $\bot \leq f(\bot) \leq f^2(\bot) \leq \ldots \rightsquigarrow \mu Z.\, f(Z)$
- the greatest f.p.: $\top \geq f(\top) \geq f^2(\top) \geq \ldots \rightsquigarrow \nu Z.\, f(Z)$

When A finite, the fixed points are reached after $\leq |A|$ iterations.

Example: $\langle A, \leq \rangle = \langle \mathcal{P}(S), \subseteq \rangle$ $\quad$ EX $Z = \{s : R(s, s') \text{ for some } s' \in Z\}$

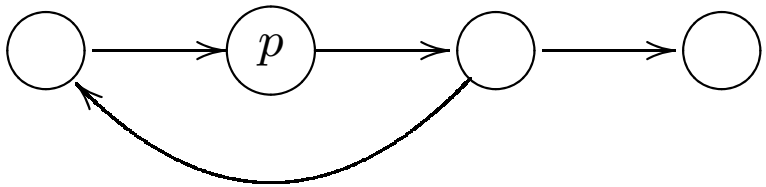
$Z \mapsto$ EX Z $\quad$ $\mu Z.$ EX $Z = \bot = \emptyset$ $\quad$ $\nu Z.$ EX Z = ?

$Z \mapsto p \vee$ EX Z $\quad$ $\mu Z.\, p \vee$ EX Z = ?

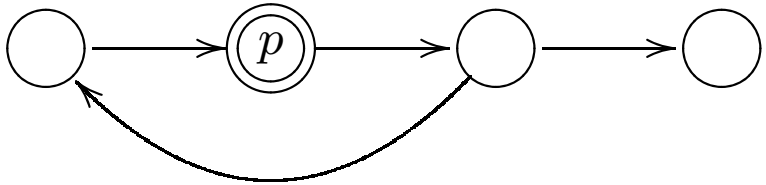
Example

$\mathsf{EF}\, p \;=\; \mu Z.\, p \vee \mathsf{EX}\, Z$ $\qquad\qquad$ $Z \mapsto p \vee \mathsf{EX}\, Z$

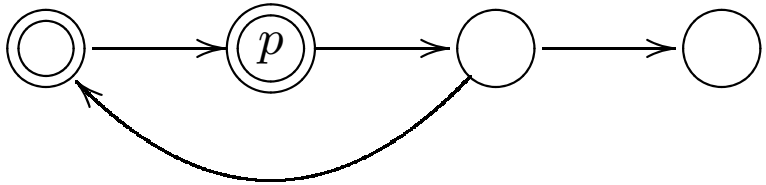
$\mathtt{false}$



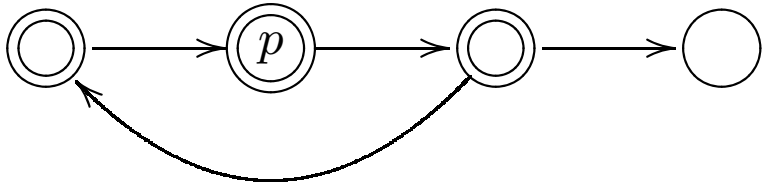
$p \vee \mathsf{EX}\, \mathtt{false} \equiv p$



$p \vee \mathsf{EX}\, p$



$p \vee \mathsf{EX}\, (p \vee \mathsf{EX}\, p)$



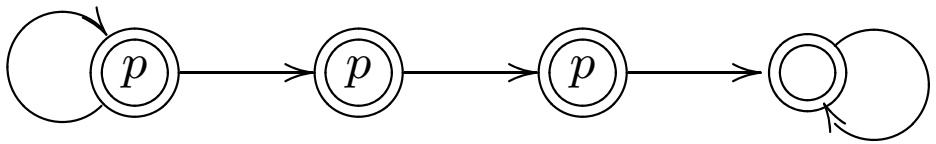
CTL via fixed points

- $\mathsf{EF}\,\phi \;=\; \mu Z.\, \phi \vee \mathsf{EX}\, Z$ $\qquad Z \mapsto \phi \vee \mathsf{EX}\, Z$
- $\mathsf{AF}\,\phi \;=\; \mu Z.\, \phi \vee \mathsf{AX}\, Z$ $\qquad Z \mapsto \phi \vee \mathsf{AX}\, Z$
- $\mathsf{EG}\,\phi \;=\; \nu Z.\, \phi \wedge \mathsf{EX}\, Z$ $\qquad Z \mapsto \phi \wedge \mathsf{EX}\, Z$
- $\mathsf{AG}\,\phi \;=\; \nu Z.\, \phi \wedge \mathsf{AX}\, Z$ $\qquad Z \mapsto \phi \wedge \mathsf{AX}\, Z$
- $\mathsf{E}\,\phi\,\mathsf{U}\,\psi \;=\; \mu Z.\, \psi \vee (\phi \wedge \mathsf{EX}\, Z)$ $\qquad Z \mapsto \psi \vee (\phi \wedge \mathsf{EX}\, Z)$
- $\mathsf{A}\,\phi\,\mathsf{U}\,\psi \;=\; \mu Z.\, \psi \vee (\phi \wedge \mathsf{AX}\, Z)$ $\qquad Z \mapsto \psi \vee (\phi \wedge \mathsf{AX}\, Z)$
- ...

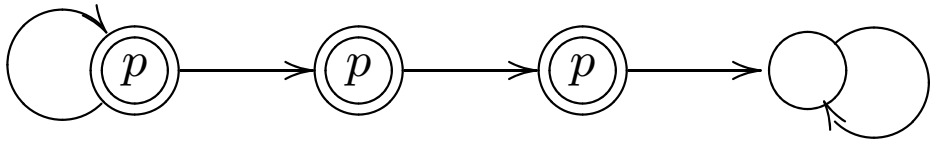
Example

$\mathsf{EG}\, p \;=\; \nu Z.\, p \wedge \mathsf{EX}\, Z \qquad\qquad Z \mapsto p \wedge \mathsf{EX}\, Z$

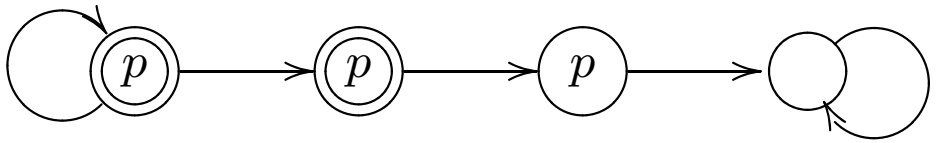
$\texttt{true}$



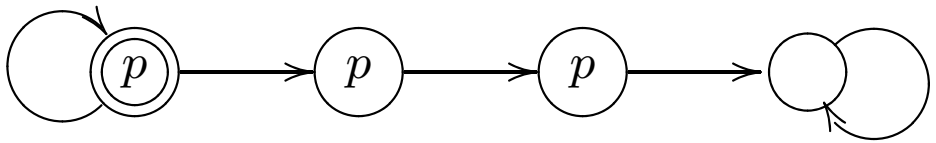
$p \wedge \mathsf{EX}\, \texttt{true} \equiv p$



$p \wedge \mathsf{EX}\, p$



$p \wedge \mathsf{EX}\, (p \wedge \mathsf{EX}\, p)$



CTL Symbolic model checking

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

$\mathrm{OBDD} : \mathrm{CTL} \mapsto \mathrm{OBDD}$

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

CTL Symbolic model checking

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

OBDD : CTL $\mapsto$ OBDD

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

– $\mathrm{OBDD}(p)$:= OBDD representing L_p

CTL Symbolic model checking

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

OBDD : CTL $\mapsto$ OBDD

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

- $\mathrm{OBDD}(p) :=$ OBDD representing L_p
- $\mathrm{OBDD}(\neg\phi) := \neg\mathrm{OBDD}(\phi)$

CTL Symbolic model checking

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

$\mathrm{OBDD} : \mathrm{CTL} \mapsto \mathrm{OBDD}$

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

- $\mathrm{OBDD}(p) := $ OBDD representing L_p
- $\mathrm{OBDD}(\neg\phi) := \neg\mathrm{OBDD}(\phi)$
- $\mathrm{OBDD}(\phi \wedge \psi) := \mathrm{OBDD}(\phi) \wedge \mathrm{OBDD}(\psi)$

Symbolic model checking (EX _)

$\mathrm{OBDD} : \mathrm{CTL} \to \mathrm{OBDD}$

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

$\mathrm{OBDD}(\mathsf{EX}\,\phi) \;:=\; \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge f[\vec{x}'/\vec{x}]$ where $f = \mathrm{OBDD}(\phi)$

$\mathrm{OBDD}(\mathsf{EX}\,\phi) \;:=\; \mathsf{EX}\, f \;=\; \mathsf{EX}\,\mathrm{OBDD}(\phi)$

$\mathsf{EX}\,\phi$

$\mathsf{EX}\, Z$

$\mathsf{EX}\, f$

Order of variables

$$\exists \vec{x}'.\ R(\vec{x}, \vec{x}') \wedge f[\vec{x}'/\vec{x}])$$

$$\vec{x} = x_1, x_2, \ldots, x_m$$

$$x_1 < x_2 < \ldots < x_m$$

$$x_1 < x_1' < x_2 < x_2' < \ldots < x_m < x_m'$$

$x_i < x_j$ if and only if $x_i' < x_j'$

Symbolic model checking (E _ U _)

$\mathrm{OBDD} : \mathrm{CTL} \to \mathrm{OBDD}$

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

$\mathrm{OBDD}(\mathsf{E}\,\phi\,\mathsf{U}\,\psi) \;:=\; \mu Z.\; g \vee (f \wedge \mathsf{EX}\, Z)$

where $f = \mathrm{OBDD}(\phi)$
$g = \mathrm{OBDD}(\psi)$

$h \mapsto g \vee (f \wedge \mathsf{EX}\, h)$

$h \mapsto g \vee (f \wedge \exists \vec{x}'.\; R(\vec{x}, \vec{x}') \wedge h[\vec{x}'/\vec{x}])$

Symbolic model checking (E _ U _)

OBDD : CTL $\to$ OBDD

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \models \phi\}$

$\mathrm{OBDD}(\mathsf{E}\,\phi\,\mathsf{U}\,\psi) \;:=\; \mu Z.\, g \vee (f \wedge \mathsf{EX}\, Z)$

where $f = \mathrm{OBDD}(\phi)$
$g = \mathrm{OBDD}(\psi)$

$$h \mapsto g \vee (f \wedge \mathsf{EX}\, h)$$

$$h \mapsto g \vee (f \wedge \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge h[\vec{x}'/\vec{x}])$$

Implement computation of the approximations of lfp by OBDDs:

$$\begin{array}{lcll} h_0 & \equiv & \texttt{false} & \\ h_1 & \equiv & g \vee (f \wedge \mathsf{EX}\, h_0) & \equiv \quad g \\ h_2 & \equiv & g \vee (f \wedge \mathsf{EX}\, h_1) & \equiv \quad g \vee (f \wedge \mathsf{EX}\, g) \\ h_3 & \equiv & \ldots & \equiv \quad g \vee (f \wedge \mathsf{EX}\, (g \vee (f \wedge \mathsf{EX}\, g))) \end{array}$$

$$\mu Z.\, g \vee (f \wedge \mathsf{EX}\, Z) \qquad (h_i = h_{i+1})$$

Symbolic model checking (EG _)

OBDD : CTL $\to$ OBDD

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

$\mathrm{OBDD}(\mathsf{EG}\,\phi) \;:=\; \nu Z.\, f \wedge \mathsf{EX}\, Z$ where $f = \mathrm{OBDD}(\phi)$

$h \mapsto f \wedge \mathsf{EX}\, h$

$h \mapsto f \wedge \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge h[\vec{x}'/\vec{x}]$

Symbolic model checking (EG _)

OBDD : CTL $\to$ OBDD

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \vDash \phi\}$

$\mathrm{OBDD}(\mathsf{EG}\,\phi) \quad := \quad \nu Z.\, f \wedge \mathsf{EX}\, Z$ where $f = \mathrm{OBDD}(\phi)$

$$h \mapsto f \wedge \mathsf{EX}\, h$$

$$h \mapsto f \wedge \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge h[\vec{x}'/\vec{x}]$$

Implement computation of the approximations of gfp by OBDDs, as bofore.

EX ϕ	E ϕ U ψ	EG ϕ
EX Z	E Z U Z'	EG Z
EX f	E f U g	EG f

Graph diameter

Efficiency of SMC depends on the diameter of M.

Fairness

Fair CTL

$\mathbf{F} = \{\psi_1, \ldots, \psi_n\}, \quad \psi_i \in \mathsf{CTL} \qquad \mapsto \quad F = \{Z_1, \ldots, Z_n\}$

$$
\begin{array}{lcl}
s \models_{\mathbf{F}} p & \iff & p \in L(s) \ \wedge \ \exists \text{ fair } \Pi \text{ from } s \\
s \models_{\mathbf{F}} \mathsf{A}\,\phi\,\mathsf{U}\,\psi & \iff & \forall \text{ fair } \Pi \text{ from } s \,.\ \Pi \models \phi\,\mathsf{U}\,\psi \\
s \models_{\mathbf{F}} \mathsf{E}\,\phi\,\mathsf{U}\,\psi & \iff & \exists \text{ fair } \Pi \text{ from } s \,.\ \Pi \models \phi\,\mathsf{U}\,\psi \\
s \models_{\mathbf{F}} \mathsf{AX}\,\phi & \iff & \forall \text{ fair } \Pi \text{ from } s \,.\ \Pi \models \mathsf{X}\,\phi \\
s \models_{\mathbf{F}} \mathsf{EX}\,\phi & \iff & \exists \text{ fair } \Pi \text{ from } s \,.\ \Pi \models \mathsf{X}\,\phi
\end{array}
$$

$\mathbf{F} = \{h_1, \ldots, h_n\}, \quad h_i \in \mathsf{OBDD}$

Fair EG

$\mathbf{F} = \{\psi_1, \ldots, \psi_n\}, \quad \psi_i \in \text{CTL} \qquad \mapsto \quad F = \{Z_1, \ldots, Z_n\}$

$\text{EG}\,\phi \;=\; \{s \mid s \vDash_{\mathbf{F}} \text{EG}\,\phi\} \;=$ the greatest Z s.t. if $s \in Z$ then

- $s \vDash \phi$
- $\forall i \leq n \,.\; \exists s' \,.\; s \rightarrow \ldots \rightarrow s' \in Z_i \cap Z, \quad s' \neq s,$ all intermediate states satisfy ϕ

Fair EG

$\mathbf{F} = \{\psi_1, \ldots, \psi_n\}, \quad \psi_i \in \mathsf{CTL} \qquad \mapsto \quad F = \{Z_1, \ldots, Z_n\}$

$\mathsf{EG}\,\phi \;=\; \{s \mid s \vDash_{\mathbf{F}} \mathsf{EG}\,\phi\} \;=$ the greatest Z s.t. if $s \in Z$ then

- $s \vDash \phi$
- $\forall i \leq n \,.\; \exists s' \,.\; s \to \ldots \to s' \in Z_i \cap Z, \quad s' \neq s,$ all intermediate states satisfy ϕ

$\mathsf{EG}\,\phi \;=\; \nu Z.\, \phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\,\phi\,\mathsf{U}\,(\psi_i \wedge Z)$

Fair EG

$\mathbf{F} = \{\psi_1, \ldots, \psi_n\}, \quad \psi_i \in \mathsf{CTL} \qquad \mapsto \quad F = \{Z_1, \ldots, Z_n\}$

$\mathsf{EG}\,\phi \;=\; \{s \mid s \vDash_{\mathbf{F}} \mathsf{EG}\,\phi\} \;=$ the greatest Z s.t. if $s \in Z$ then

- $s \vDash \phi$
- $\forall i \leq n \,.\; \exists s' \,.\; s \rightarrow \ldots \rightarrow s' \in Z_i \cap Z, \quad s' \neq s,$ all intermediate states satisfy ϕ

$\mathsf{EG}\,\phi \;=\; \nu Z.\, \phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\,\phi\,\mathsf{U}\,(\psi_i \wedge Z)$

$\mathsf{EG}\,\phi \;=\; \nu Z.\, \phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mu Y.(\psi_i \wedge Z) \vee (\phi \wedge \mathsf{EX}\,Y)$ alternation!

Fair EG

Thm:

$$\mathsf{EG}\,\phi \;=\; \nu Z.\;\phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\,\phi\,\mathsf{U}\,(\psi_i \wedge Z)$$

Proof:

$$\mathsf{EG}\,\phi \;=\; \phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\,\phi\,\mathsf{U}\,(\psi_i \wedge \mathsf{EG}\,\phi)$$

$$Z \;=\; \phi \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\,\phi\,\mathsf{U}\,(\psi_i \wedge Z) \implies Z \subseteq \mathsf{EG}\,\phi$$

Fair symbolic model checking (EG _)

$\mathrm{OBDD} : \mathrm{CTL} \to \mathrm{OBDD}$

$\mathrm{OBDD}(\phi)$ represents $\{s \mid s \models_{\mathbf{F}} \phi\}$

$\mathbf{F} = \{\psi_1, \ldots, \psi_n\}, \psi_i \in \mathrm{CTL} \quad \mapsto \quad F = \{h_1, \ldots, h_n\}, h_i \in \mathrm{OBDD}$

$\mathrm{OBDD}(\mathsf{EG}\,\phi) \;:=\; \nu Z.\, f \;\wedge\; \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\, f\, \mathsf{U}\, (h_i \wedge Z)$

where $f = \mathrm{OBDD}(\phi)$

$$Z \mapsto f \wedge \bigwedge_{i=1}^{n} \mathsf{EX}\,\mathsf{E}\, f\, \mathsf{U}\, (h_i \wedge Z)$$

Fair CTL symbolic model checking

$$\texttt{fair} \;:=\; \mathrm{OBDD}(\,\mathsf{EG}\,\texttt{true})$$

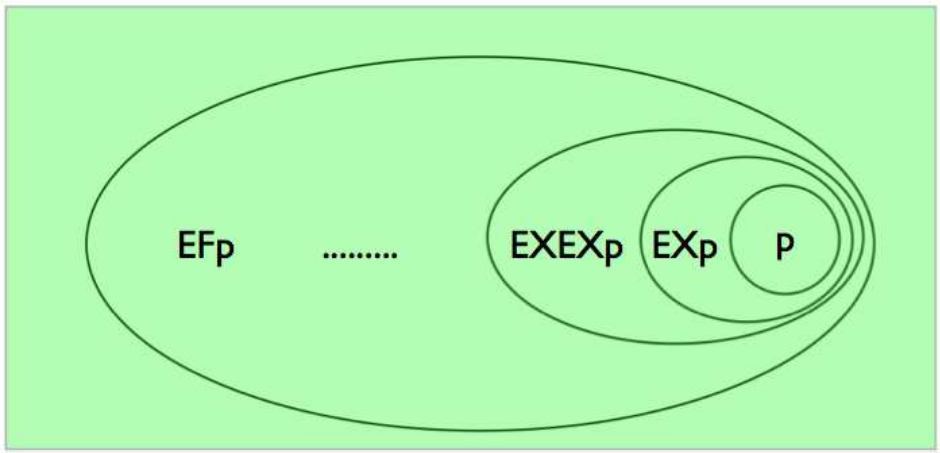
$$\mathrm{OBDD}(\,\mathsf{EX}\,\phi) \;:=\; \exists \vec{x}'.\; R(\vec{x},\vec{x}') \wedge f(\vec{x}') \wedge \texttt{fair}(\vec{x}')$$

where $f = \mathrm{OBDD}(\phi)$

$$\mathrm{OBDD}(\,\mathsf{E}\,\phi\,\mathsf{U}\,\psi) \;:=\; \mu Z.\; (g \wedge \texttt{fair}) \vee (f \wedge \mathsf{EX}\, Z)$$

where $f = \mathrm{OBDD}(\phi)$
$g = \mathrm{OBDD}(\psi)$

How to compute EX f ?



EX f by AndExist

$$\mathsf{EX}\, f \;:=\; \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge f(\vec{x}')$$

operation $\exists\wedge(g, h, V) \;:=\; \exists V.\, g \wedge h$ (V – set of variables)

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

$f(x_1, \ldots, x_m) \;\mapsto\; f'(x'_1, \ldots, x'_m)$ $\qquad x_i \leq x_j \iff x'_i \leq x'_j$

$$\mathsf{EX}\, f \;=\; \exists\wedge(R, f', \{x'_1, \ldots, x'_m\})$$

AndExist

$\exists\wedge(f,\ g, V)$ $\qquad\qquad$ $(\ \exists V.\ f \wedge g\ )$

- f, g leaves: $\mathrm{val}(\exists\wedge(f,g,V)) := \mathrm{val}(f) \wedge \mathrm{val}(g)$
- f a leaf, g not: $\exists\wedge(f,g,V) :=$ `false` or $\exists V.\ g$
- $x = \mathrm{var}(f) = \mathrm{var}(g)$:

  $l := \exists\wedge(s_0(f), s_0(g), V), \quad h := \exists\wedge(s_1(f), s_1(g), V)$

  - $x \in V$: $\exists\wedge(f,g,V) := l \vee h$
  - $x \notin V$: $s_0(\exists\wedge(f,g,V)) := l$ $\qquad\qquad$ $s_1(\exists\wedge(f,g,V)) := h$
- $x = \mathrm{var}(f) < \mathrm{var}(g)$: ...

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

R is not monolytic

$$\mathsf{EX}\, f \;:=\; \exists \vec{x}'.\, R(\vec{x}, \vec{x}') \wedge f(\vec{x}')$$

Asynchronous model: $R \;=\; R'_1 \;\vee\; R'_2 \vee \;\ldots\; \vee\; R'_n$

$$R'_i \;=\; R_i \wedge \textstyle\bigwedge_{j \neq i} \mathrm{Id}_j$$

Synchronous model: $R \;=\; R_1 \;\wedge\; R_2 \wedge \;\ldots\; \wedge\; R_n$

Can one profit from this additional structure ?

Asynchronous model: $R = R'_1 \vee R'_2 \vee \ldots \vee R'_n$

$$R'_i = R_i \wedge \bigwedge_{j \neq i} x_j = x'_j$$

$$\begin{aligned}
\exists \vec{x}'.\, R \wedge f(\vec{x}') &\equiv \exists \vec{x}'.\, (R'_1 \wedge f(\vec{x}')) \vee \ldots \vee (R'_n \wedge f(\vec{x}')) \\
&\equiv \big(\exists \vec{x}'.\, R'_1 \wedge f(\vec{x}')\big) \vee \ldots \vee \big(\exists \vec{x}'.\, R'_n \wedge f(\vec{x}')\big)
\end{aligned}$$

$$\begin{aligned}
\exists \vec{x}'.\, R'_i \wedge f(\vec{x}') &\equiv \exists \vec{x}'.\, R_i \wedge \big(\bigwedge_{j \neq i} x_j = x'_j\big) \wedge f(\vec{x}') \\
&\equiv \exists x'_i.\, R_i(\vec{x}, x'_i) \wedge f(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_m)
\end{aligned}$$

Synchronous model: $R = R_1 \wedge R_2 \wedge \ldots \wedge R_n$

$$\exists \vec{x}' .\ R_1(\vec{x}, \vec{x}') \wedge \ldots \wedge R_n(\vec{x}, \vec{x}') \wedge f(\vec{x}')$$

- relations R_i are local
- „early” quantification
- heuristics

Example: 3-bit counter

$$
\begin{array}{lcl}
R_0(\vec{x}, x_0') & = & (x_0' = \neg x_0) \\
R_1(\vec{x}, x_1') & = & (x_1' = x_0 \text{ xor } x_1) \\
R_2(\vec{x}, x_2') & = & (x_2' = (x_0 \wedge x_1) \text{ xor } x_2)
\end{array}
$$

$$\exists x_2' \exists x_1' \exists x_0'.\ f(x_0', x_1', x_2') \ \wedge\ R_0(\vec{x}, x_0') \ \wedge\ R_1(\vec{x}, x_1') \ \wedge\ R_2(\vec{x}, x_2')$$

$$\exists x_2' \big(\exists x_1' \exists x_0'.\ f(x_0', x_1', x_2') \ \wedge\ R_0(\vec{x}, x_0') \ \wedge\ R_1(\vec{x}, x_1')\big) \ \wedge\ R_2(\vec{x}, x_2')$$

$$\exists x_2' \big(\exists x_1' \big(\exists x_0'.\ f(x_0', x_1', x_2') \ \wedge\ R_0(\vec{x}, x_0')\big) \ \wedge\ R_1(\vec{x}, x_1')\big) \ \wedge\ R_2(\vec{x}, x_2')$$

$$\exists x_2' \big(\exists x_1' \big(\exists x_0'.\ f(x_0', x_1', x_2') \ \wedge\ R_0(x_0, x_0')\big) \ \wedge\ R_1(x_0, x_1, x_1')\big) \ \wedge\ R_2(x_0, x_1, x_2, x_2')$$

Example: 3-bit counter

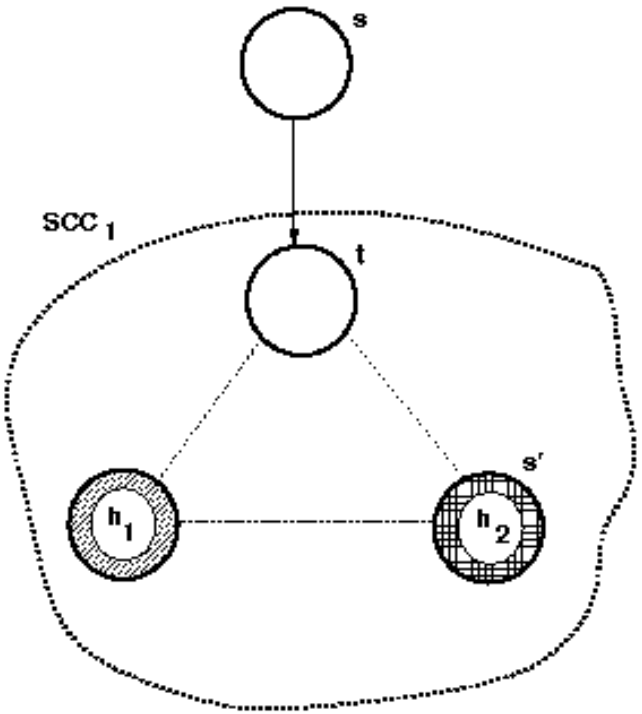
$$\exists x'_2 \quad (\exists x'_1 \quad (\exists x'_0 \quad f(x'_0, x'_1, x'_2) \ \wedge \ R_0(x_0, x'_0)) \\ \wedge \quad R_1(x_0, x_1, x'_1)) \\ \wedge \quad R_2(x_0, x_1, x_2, x'_2)$$

- sequence of $\exists\wedge$ operations
- optimal order of processes (not variables this time):
  - early elimination of variables ($\exists$)
  - late introducing of variables

(Counter)examples

Counterexample

counterexample for $\mathrm{AF}\,\phi$ = example for $\mathrm{EG}\,\neg\phi$



[Clarke, Grumberg, Long 1994]

Counterexample

counterexample for $\mathrm{AF}\,\phi$ = example for $\mathrm{EG}\,\neg\phi$

counterexample for $\mathrm{AG}\,\phi$ = example for $\mathrm{EF}\,\neg\phi$

(fair counterexample is always an infinite path)

Counterexample

counterexample for $\mathrm{AF}\,\phi$ = example for $\mathrm{EG}\,\neg\phi$

counterexample for $\mathrm{AG}\,\phi$ = example for $\mathrm{EF}\,\neg\phi$

(fair counterexample is always an infinite path)

counterexample for $\mathrm{EF}\,\phi$ = **?**

counterexample for $\mathrm{EG}\,\phi$ = **?**

Symbolic counterexample

How to compute an example for:

- EG ϕ
- E ϕ U ψ
- EX ϕ

symbolically ?

Example for E_U_

Computation of E f U g:

$$Q_0 \subseteq Q_1 \subseteq \ldots \quad (1 \leq i \leq n)$$

$s \in Q_j \iff g$ may be reached from s "via f" by $\leq j$ transitions

Computing an example for $s \models$ E f U g:

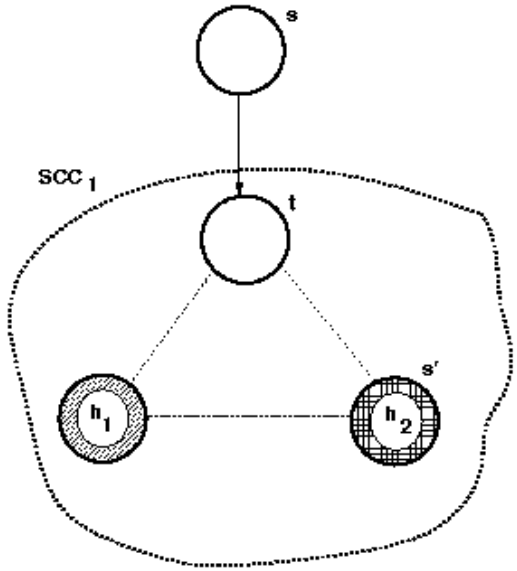
- let j minimal s.t. $s \in Q_j$
- reconstruct $s = s_j \rightarrow s_{j-1} \rightarrow \ldots \rightarrow s_0 \in g$

Symbolic counterexample

How to compute a fair example for:

- EG ϕ
- E ϕ U ψ
- EX ϕ

symbolically ?



[Clarke, Grumberg, Long 1994]

Fair example for EG _

$$\text{EG}\, f \;=\; \nu Z.\; f \;\wedge\; \bigwedge_{i=1}^{n} \text{EX E}\, f\, \text{U}\, (h_i \wedge Z)$$

last iteration $Z \;\mapsto\; f \;\wedge\; \bigwedge_{i=1}^{n} \text{EX E}\, f\, \text{U}\, (h_i \wedge Z)$:

computation of $\text{E}\, f\, \text{U}\, (h_i \wedge Z)$: $\qquad Z = \text{EG}\, f$

$$Q_0^i \;\subseteq\; Q_1^i \;\subseteq\; \ldots \qquad (1 \leq i \leq n)$$

$s \in Q_j^i \iff (h_i \wedge \text{EG}\, f)$ may be reached from s "via f"

by $\leq j$ transitions

Fair example for EG _

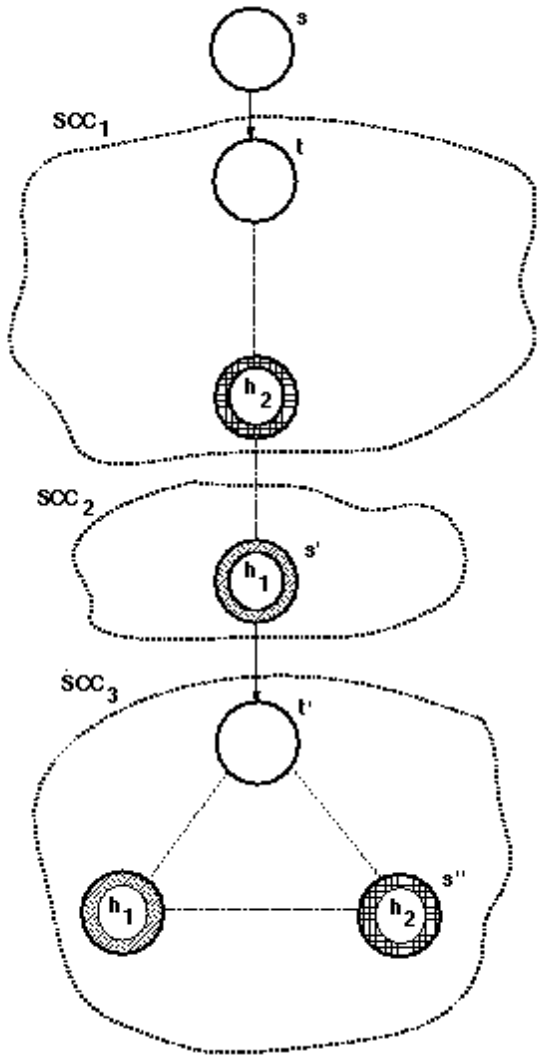
$$\mathsf{EG}\, f \;=\; \nu Z.\; f \;\wedge\; \bigwedge_{i=1}^{n} \mathsf{EX}\, \mathsf{E}\, f\, \mathsf{U}\, (h_i \wedge Z)$$

$s := s_0$ initial state
$I := \{1, \ldots, n\}$
repeat
  find t s.t. $s \to t$, $t \in Q_j^i$, $i \in I$, j minimal
  reconstruct $t = t_j \to t_{j-1} \to \ldots \to t_0 \in (h_i \wedge \mathsf{EG}\, f)$
  $I := I \setminus \{i \mid t_0 \in h_i\}$
  $s := t_0$
until $I = \emptyset$
$s' := s$

$I := I \setminus \{i \mid t \in Q_j^i\}$

$\mapsto$ path $s_0 \to \ldots \to s'$

Fair example for EG _



[Clarke, Grumberg, Long 1994]

Fair example for EG _

$$\mathsf{EG}\, f \;=\; \nu Z.\; f \;\wedge\; \bigwedge_{i=1}^{n} \mathsf{EX}\, \mathsf{E}\, f\, \mathsf{U}\, (h_i \wedge Z)$$

we have a path $s_0 \to \ldots \to s'$ let $t =$ the first t_0

(a) if $s' \vDash \mathsf{EX}\, \mathsf{E}\, f\, \mathsf{U}\, \{t\}$ stop

otherwise restart: $s_0 := s', \quad I := \{1, \ldots, n\}$

improvement:

(b) compute $\mathsf{E}\, f\, \mathsf{U}\, \{t\}$

as long as $\neg(s \vDash \mathsf{E}\, f\, \mathsf{U}\, \{t\})$, restart: $s_0 := s$, $I := \{1, \ldots, n\}$

Fair example for E_U_, EX_

Example for $\mathsf{E}\,\phi\,\mathsf{U}\,(\psi \wedge \texttt{fair})$ or $\mathsf{EX}\,(\phi \wedge \texttt{fair})$ extend with a fair example for $\mathsf{EG}\,\texttt{true}$.

What else can be computed using OBDDs ?

- $L_\omega(\mathcal{A}) \neq \emptyset$ — fair EG $\mathtt{true}$
- LTL model checking
- $L_\omega(\mathcal{A}_1) \subseteq L_\omega(\mathcal{A}_2)$ — $\mathcal{A}_1 \times \mathcal{A}_2 \models \mathsf{A}(\mathsf{G}\,\mathsf{F}\,q_1 \implies \mathsf{G}\,\mathsf{F}\,q_2)$
- μ-calculus model checking
- reachable states
- deadlocks
- (bi)simulation equivalence
- ...

What else can be computed using OBDDs ?

- $L_\omega(\mathcal{A}) \neq \emptyset$ — fair EG `true`
- LTL model checking
- $L_\omega(\mathcal{A}_1) \subseteq L_\omega(\mathcal{A}_2)$ — $\mathcal{A}_1 \times \mathcal{A}_2 \models \mathsf{A}(\mathsf{G}\,\mathsf{F}\,q_1 \implies \mathsf{G}\,\mathsf{F}\,q_2)$
- μ-calculus model checking
- reachable states
- deadlocks
- (bi)simulation equivalence
- ...

OBDDs are routinely used in hardware industry nowadays