Reduced Ordered Binary Decision Diagrams

Lecture #13 of Advanced Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

E-mail: katoen@cs.rwth-aachen.de

June 16, 2014

Basic approach

- let $\mathit{TS} = (S, \rightarrow, I, \mathit{AP}, L)$ be a "large" finite transition system
  - the set of actions is irrelevant here and has been omitted, i.e., $\rightarrow \subseteq S \times S$
- For $n \geqslant \lceil \log |S| \rceil$, let injective function $\mathit{enc} : S \rightarrow \{ 0, 1 \}^n$
  - note: $\mathit{enc}(S) = \{0, 1\}^n$ is no restriction, as all elements $\{ 0, 1 \}^n \setminus \mathit{enc}(S)$ can be treated as the encoding of pseudo states that are unreachable
- Identify the states $s \in S = \mathit{enc}^{-1}(\{ 0, 1 \}^n)$ with $\mathit{enc}(s) \in \{0, 1\}^n$
- And $T \subseteq S$ by its characteristic function $\chi_T : \{ 0, 1 \}^n \rightarrow \{ 0, 1 \}$
  - that is $\chi_T(\mathit{enc}(s)) = 1$ if and only if $s \in T$
- And $\rightarrow \subseteq S \times S$ by the Boolean function $\Delta : \{ 0, 1 \}^{2n} \rightarrow \{ 0, 1 \}$
  - such that $\Delta \left(\mathit{enc}(s), \mathit{enc}(s')\right) = 1$ if and only if $s \rightarrow s'$

Switching functions

- Let *Var* $= \{z_1, \ldots, z_m\}$ be a finite set of Boolean variables
- An evaluation is a function η : *Var* $\to \{0, 1\}$
  - let *Eval*$(z_1, \ldots, z_m)$ denote the set of evaluations for $z_1, \ldots, z_m$
  - shorthand $[z_1 = b_1, \ldots, z_m = b_m]$ for $\eta(z_1) = b_1, \ldots, \eta(z_m) = b_m$
- f : *Eval*(*Var*) $\to \{0, 1\}$ is a *switching function* for *Var* $= \{z_1, \ldots, z_m\}$
- Logical operations and quantification are defined by:

$$
\begin{array}{rcl}
f_1(\cdot) \wedge f_2(\cdot) & = & \min\{ f_1(\cdot), f_2(\cdot) \} \\
f_1(\cdot) \vee f_2(\cdot) & = & \max\{ f_1(\cdot), f_2(\cdot) \} \\
\exists z.\, f(\cdot) & = & f(\cdot)|_{z=0} \vee f(\cdot)|_{z=1}, \text{ and} \\
\forall z.\, f(\cdot) & = & f(\cdot)|_{z=0} \wedge f(\cdot)|_{z=1}
\end{array}
$$

Polynomial-size data structure impossible

- There is no poly-size data structure for all switching functions
  - $|\mathit{Eval}(z_1, \ldots, z_m)| = 2^m$, so #functions $\mathit{Eval}(z_1, \ldots, z_m) \to \{0, 1\}$ is 2^{2^m}
- Suppose there is a data structure that can represent K_m switching functions by at most 2^{m-1} bits
- Then $K_m \leqslant \sum_{i=0}^{2^{m-1}} 2^i = 2^{2^{m-1}+1} - 1 < 2^{2^{m-1}+1}$
- But then there are at least

$$2^{2^m} - 2^{2^{m-1}+1} = 2^{2^{m-1}+1} \cdot \left(2^{2^m - 2^{m-1} - 1} - 1\right) = 2^{2^{m-1}+1} \cdot \left(2^{2^{m-1}-1} - 1\right)$$

switching functions whose representation needs more than 2^{m-1} bits

Representing switching functions

- Truth tables
  - very space inefficient: 2^n entries for n variables
  - satisfiability and equivalence check: easy; boolean operations also easy
  - . . . but have to consider exponentially many lines (so are hard)
- . . . in Disjunctive Normal Form (DNF)
  - satisfiability is easy: find a disjunct that does have complementary literals
  - negation and conjunction complicated
  - equivalence checking ($f = g$?) is coNP-complete
- . . . in Conjunctive Normal Form (CNF)
  - satisfiability problem is NP-complete (Cook's theorem)
  - negation and disjunction complicated

Representing switching functions

representation	*compact?*	*sat*	*equi*	$\wedge$	$\vee$	$\neg$
propositional formula	often	hard	hard	easy	easy	easy
DNF	sometimes	easy	hard	hard	easy	hard
CNF	sometimes	hard	hard	easy	hard	hard
(ordered) truth table	never	hard	hard	hard	hard	hard

There is hope . . . . . perhaps

Nevertheless there are data structures which yield compact representations for many switching functions that appear in practical applications

for hardware circuits, ordered binary decision diagrams (OBDDs) are successful

Representing boolean functions

representation	*compact?*	*sat*	*equ*	$\wedge$	$\vee$	$\neg$
propositional formula	often	hard	hard	easy	easy	easy
DNF	sometimes	easy	hard	hard	easy	hard
CNF	sometimes	hard	hard	easy	hard	hard
(ordered) truth table	never	hard	hard	hard	hard	hard
reduced ordered binary decision diagram	often	easy	easy*	medium	medium	easy

* provided appropriate implementation techniques are used

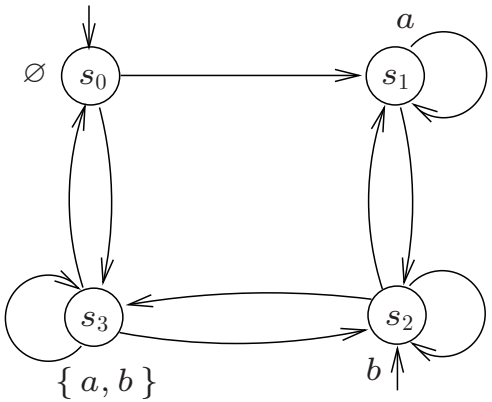
Binary decision tree

- The BDT for function f on *Var* $= \{ z_1, \ldots, z_m \}$ has depth m
  - outgoing edges for node at level i stand for $z_i = 0$ (dashed) and $z_i = 1$ (solid)
- For evaluation $s = [z_1 = b_1, \ldots, z_m = b_m]$, $f(s)$ is the value of the leaf
  - reached by traversing the BDT from the root using branch $z_i = b_i$ for at level i
- The subtree of node v at level i for variable ordering $z_1 < \ldots < z_m$ represents

$$f_v \;=\; f|_{z_1=b_1,\ldots,z_{i-1}=b_{i-1}}$$

  - which is a switching function over $\{ z_i, \ldots, z_m \}$ and
  - where $z_1 = b_1, \ldots, z_{i-1} = b_{i-1}$ is the sequence of decisions made along the path from the root to node v

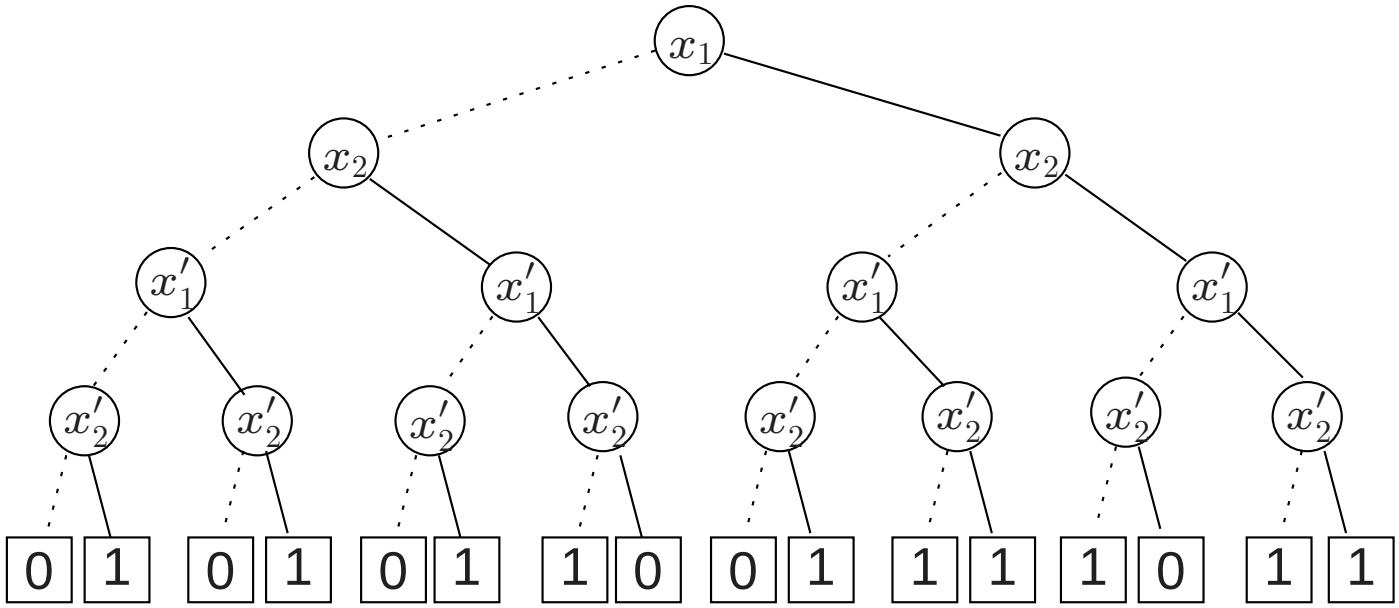
Symbolic representation of a transition system



Switching function: $\Delta(\underbrace{x_1, x_2}_{s}, \underbrace{x_1', x_2'}_{s'}) = 1$ if and only if $s \rightarrow s'$

$$
\begin{aligned}
\Delta(x_1, x_2, x_1', x_2') = \quad & (\neg x_1 \wedge \neg x_2 \wedge \neg x_1' \wedge x_2') \\
\vee \quad & (\neg x_1 \wedge \neg x_2 \wedge x_1' \wedge x_2') \\
\vee \quad & (\neg x_1 \wedge x_2 \wedge x_1' \wedge \neg x_2') \\
\vee \quad & \ldots \\
\vee \quad & (x_1 \wedge x_2 \wedge x_1' \wedge x_2')
\end{aligned}
$$

Transition relation as a BDT



A BDT representing Δ for our example using ordering $x_1 < x_2 < x'_1 < x'_2$

Considerations on BDTs

- BDTs are not compact
  - a BDT for switching function f on n variables has 2^n leafs

  $\Rightarrow$ they are as space inefficient as truth tables!

$\Rightarrow$ BDTs contain quite some redundancy

  - all leafs with value one (zero) could be collapsed into a single leaf
  - a similar scheme could be adopted for isomorphic subtrees

- The size of a BDT does not change if the variable order changes

Ordered Binary Decision Diagram

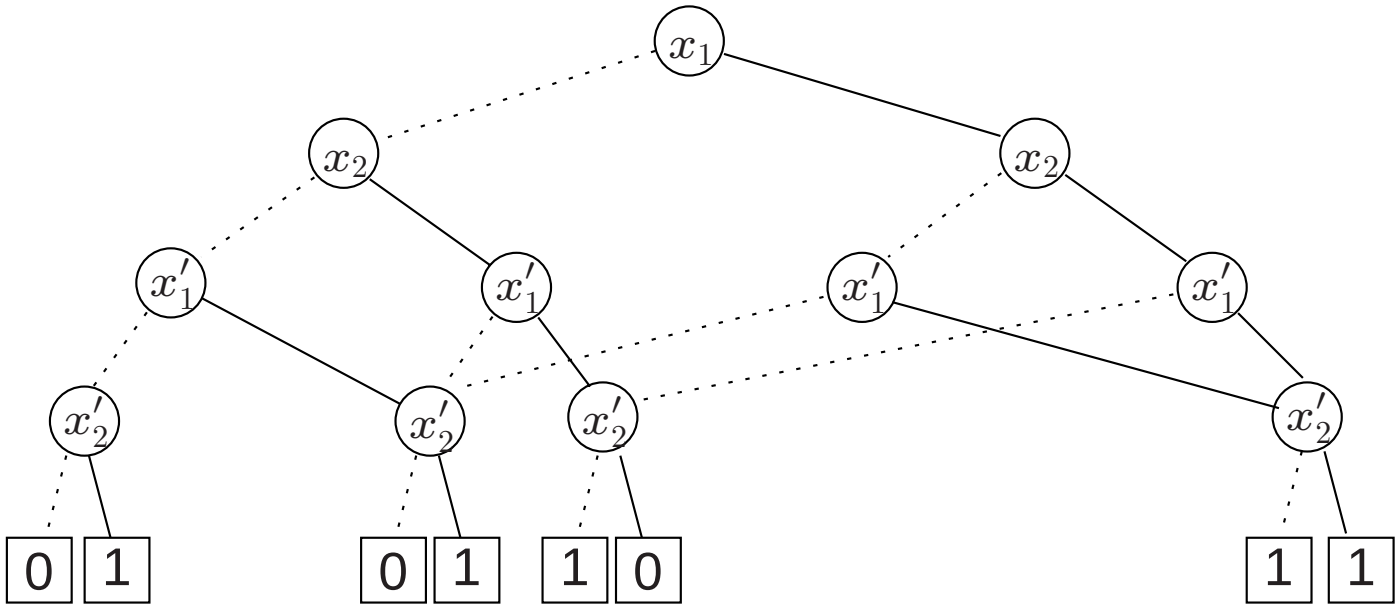
Let $\wp$ be a variable ordering for *Var* where $z_1 <_\wp \ldots <_\wp z_m$

An $\wp$-OBDD is a tuple $\mathfrak{B} = (V, V_I, V_T, \mathit{succ}_0, \mathit{succ}_1, \mathit{var}, \mathit{val}, v_0)$ with

- a finite set V of nodes, partitioned into V_I (inner) and V_T (terminals)
  - and a distinguished root $v_0 \in V$
- successor functions $\mathit{succ}_0, \mathit{succ}_1 : V_I \to V$
  - such that each node $v \in V \setminus \{v_0\}$ has at least one predecessor
- labeling functions $\mathit{var} : V_I \to \mathit{Var}$ and $\mathit{val} : V_T \to \{0, 1\}$ satisfying

$$v \in V_I \ \wedge \ w \in \{\mathit{succ}_0(v), \mathit{succ}_1(v)\} \cap V_I \ \Rightarrow \ \mathit{var}(v) <_\wp \mathit{var}(w)$$

Transition relation as an OBDD



An example OBDD representing $f_{\rightarrow}$ for our example using $x_1 < x_2 < x'_1 < x'_2$

Bottom-up characterization of $f_{\mathfrak{B}}$

Let $\mathfrak{B}$ be a $\wp$-OBDD. Switching function f_v for node $v \in V$:

- If $v \in V_T$, then f_v is the constant switching function with value $\mathit{val}(v)$
- If $v \in V_I$ with $\mathit{var}(v) = z$, then $f_v = \underbrace{\left(\neg z \wedge f_{\mathit{succ}_0(v)}\right) \vee \left(z \wedge f_{\mathit{succ}_1(v)}\right)}_{\text{Shannon expansion}}$

Furthermore, $f_{\mathfrak{B}} = f_{v_0}$ for the root v_0 of $\mathfrak{B}$

Consistent co-factors in OBDDs

- Let f be a switching function for *Var*
- Let $\wp = (z_1, \ldots, z_m)$ a variable ordering for *Var*, i.e., $z_1 <_\wp \ldots <_\wp z_m$
- Switching function g is a *$\wp$-consistent cofactor* of f if

$$g = f|_{z_1=b_1,\ldots,z_i=b_i} \quad \text{for some } i \in \{0, 1, \ldots, m\}$$

- Then it holds that:
  1. for each node v of an $\wp$-OBDD $\mathfrak{B}$, f_v is a $\wp$-consistent cofactor of $f_\mathfrak{B}$
  2. for each $\wp$-consistent cofactor g of $f_\mathfrak{B}$ there is a node $v \in \mathfrak{B}$ with $f_v = g$

Reduced OBDDs

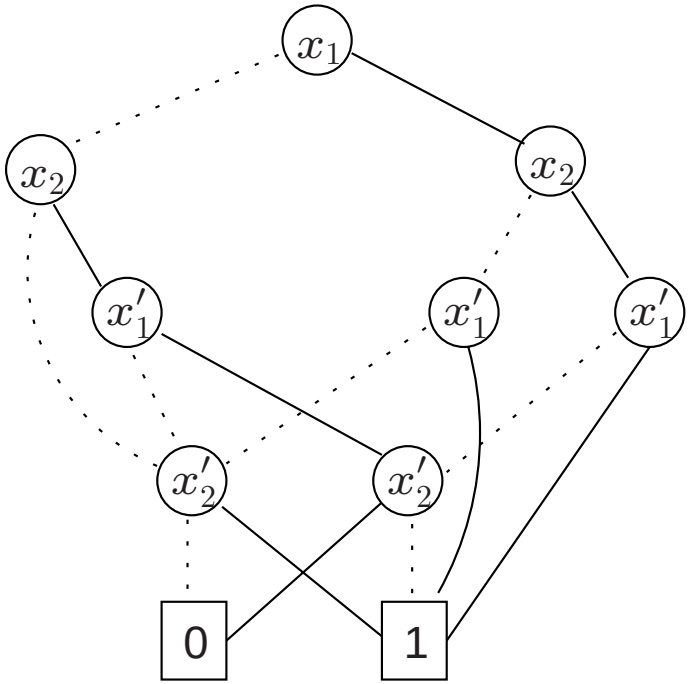
A $\wp$-OBDD $\mathfrak{B}$ is *reduced* if for every pair (v, w) of nodes in $\mathfrak{B}$:

$v \neq w$ implies $f_v \neq f_w$

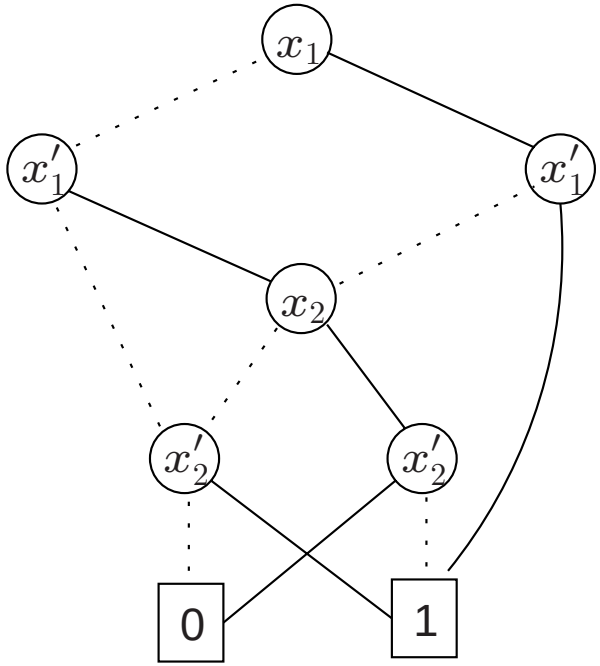
(A reduced $\wp$-OBDD is abbreviated as $\wp$-ROBDD)

$\Rightarrow$ in $\wp$-ROBDDs any $\wp$-consistent cofactor is represented by exactly one node

Transition relation as an ROBDD



(a) ordering $x_1 < x_2 < x_1' < x_2'$



(b) ordering $x_1 <' x_1' <' x_2 <' x_2'$

Universality and canonicity theorem

[Fortune, Hopcroft & Schmidt, 1978]

Let *Var* be a finite set of Boolean variables and $\wp$ a variable ordering for *Var*. Then:

(a) For each switching function f for *Var* there exists a $\wp$-ROBDD $\mathfrak{B}$ with $f_{\mathfrak{B}} = f$

(b) Any $\wp$-ROBDDs $\mathfrak{B}$ and $\mathfrak{C}$ with $f_{\mathfrak{B}} = f_{\mathfrak{C}}$ are isomorphic

Any $\wp$-OBDD $\mathfrak{B}$ for f is reduced iff $\mathit{size}(\mathfrak{B}) \leqslant \mathit{size}(\mathfrak{C})$ for each $\wp$-OBDD $\mathfrak{C}$ for f

Proofs

The importance of canonicity

- Absence of redundant vertices
  - if $f_{\mathfrak{B}}$ does not depend on z_i, ROBDD $\mathfrak{B}$ does not contain an x_i node
- Test for equivalence: $f(x_1, \ldots, x_n) \equiv g(x_1, \ldots, x_n)$?
  - generate ROBDDs $\mathfrak{B}_f$ and $\mathfrak{B}_g$, and check isomorphism
- Test for validity: $f(x_1, \ldots, x_n) = 1$?
  - generate ROBDD $\mathfrak{B}_f$ and check whether it only consists of a 1-leaf
- Test for implication: $f(x_1, \ldots, x_n) \rightarrow g(x_1, \ldots, x_n)$?
  - generate ROBDD $\mathfrak{B}_{f \wedge \neg g}$ and check if it just consists of a 0-leaf
- Test for satisfiability
  - f is satisfiable if and only if $\mathfrak{B}_f$ has a reachable 1-leaf

Minimality of ROBDDs

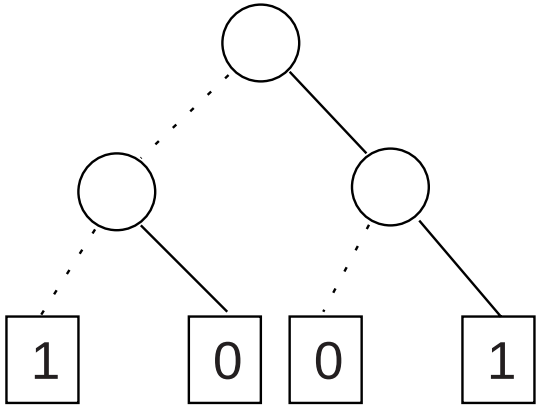
For any $\wp$-OBDD $\mathfrak{B}$ for f $\mathfrak{B}$ is reduced iff $\mathit{size}(\mathfrak{B}) \leqslant \mathit{size}(\mathfrak{C})$ for each $\wp$-OBDD $\mathfrak{C}$ for f

Reducing OBDDs

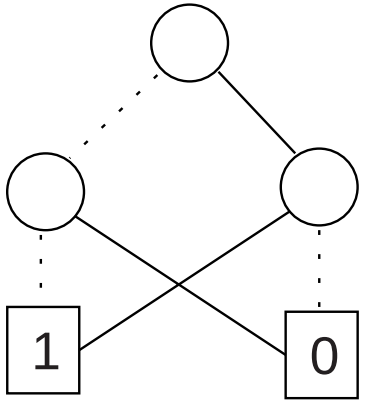
- Generate an OBDD (or BDT) for a switching function, then reduce
  - by means of a recursive descent over the OBDD
- Elimination of duplicate leafs
  - for a duplicate 0-leaf (or 1-leaf), redirect all incoming edges to just one of them
- Elimination of “don’t care” (non-leaf) vertices
  - if $\mathit{succ}_0(v) = \mathit{succ}_1(v) = w$, delete v and redirect all its incoming edges to w
- Elimination of isomorphic subtrees
  - if $v \neq w$ are roots of isomorphic subtrees, remove w and redirect all incoming edges to w to v

note that the first reduction is a special case of the latter

How to reduce an OBDD?

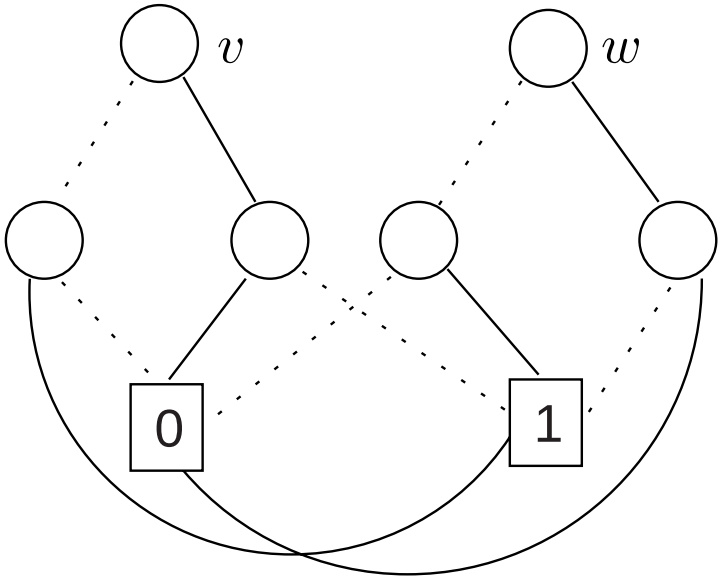


becomes

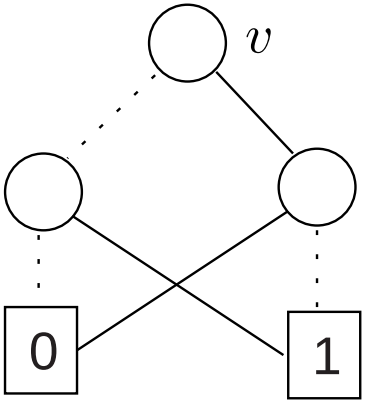


elimination of duplicated leaves

How to reduce an OBDD?

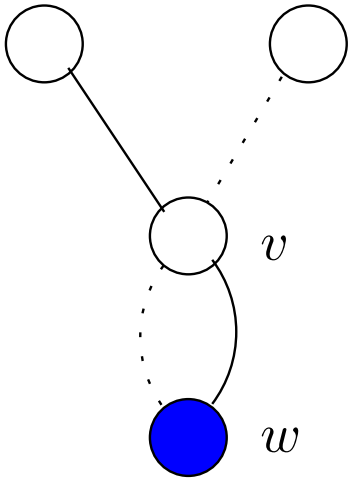


becomes

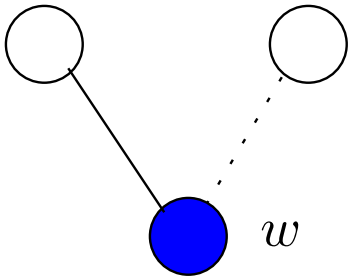


isomorphism rule

How to reduce an OBDD?



becomes



elimination rule

Soundness and completeness

if $\mathfrak{C}$ arises from a $\wp$-OBDD $\mathfrak{B}$ by applying the elimination or isomorphism rule, then:

$\mathfrak{C}$ is a $\wp$-OBDD with $f_{\mathfrak{B}} = f_{\mathfrak{C}}$

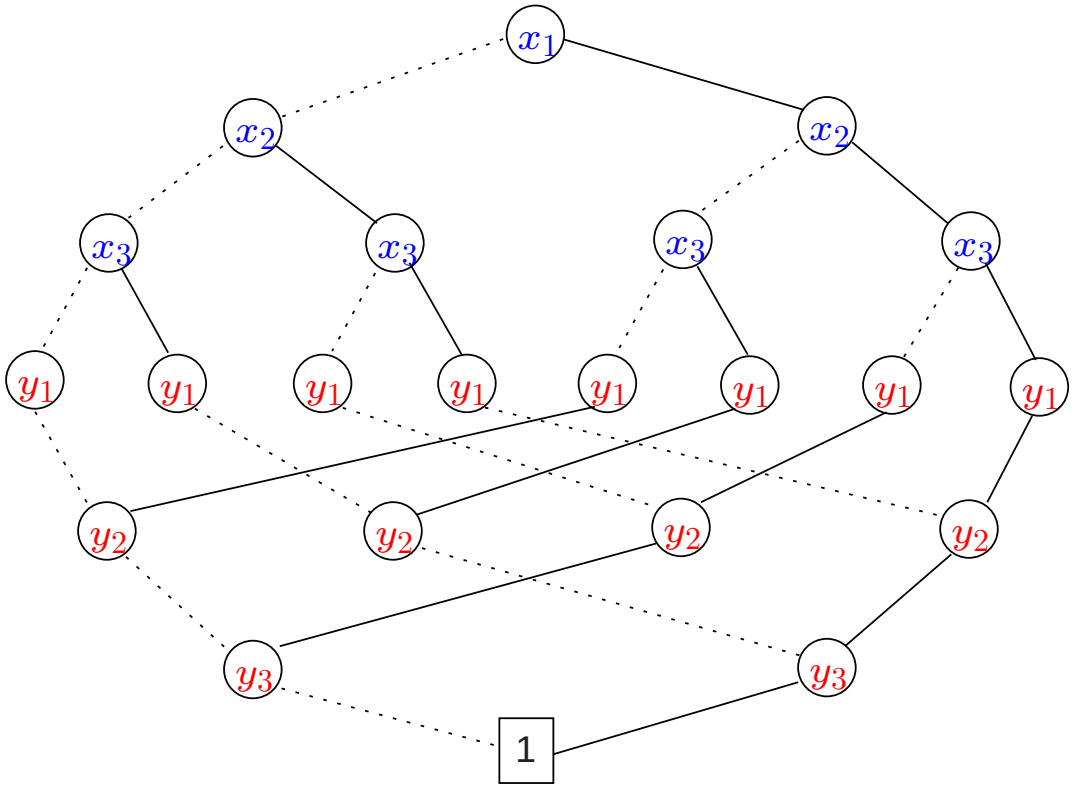
$\wp$-OBDD $\mathfrak{B}$ is reduced if and only if no reduction rule is applicable to $\mathfrak{B}$

Proof

Variable ordering

- ROBDDs are canonical for a fixed variable ordering
  - the size of the ROBDD crucially depends on the variable ordering
  - $\#$ nodes in ROBDD $\mathfrak{B}$ = $\#$ of $\wp$-consistent co-factors of f
- Some switching functions have linear and exponential ROBDDs
  - e.g., the addition function, or the stable function
- Some switching functions only have polynomial ROBDDs
  - this holds, e.g., for symmetric functions (see next)
  - examples $f(\ldots) = x_1 \oplus \ldots \oplus x_n$, or $f(\ldots) = 1$ iff $\geqslant k$ variables x_i are true
- Some switching functions only have exponential ROBDDs
  - this holds, e.g., for the middle bit of the multiplication function

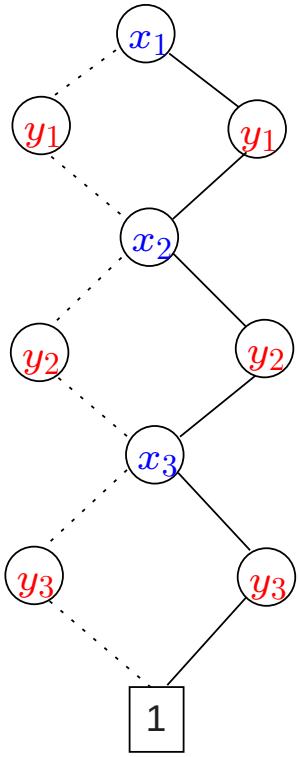
The function stable with exponential ROBDD



The ROBDD of $f_{stab}(\overline{x}, \overline{y}) = (x_1 \leftrightarrow y_1) \ \wedge \ \ldots \ \wedge \ (x_n \leftrightarrow y_n)$

has $3 \cdot 2^n - 1$ vertices under ordering $x_1 < \ldots < x_n < y_1 < \ldots < y_n$

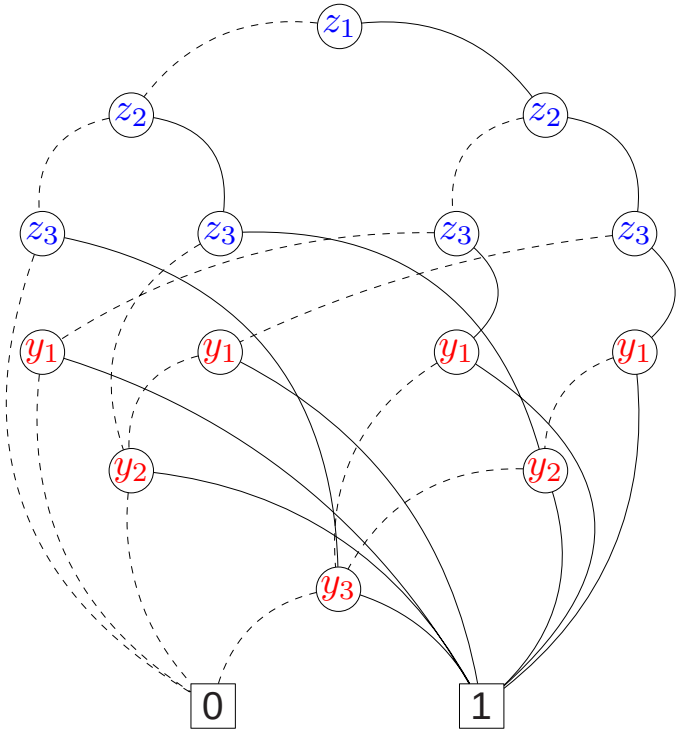
The function stable with linear ROBDD



The ROBDD of $f_{stab}(\overline{x}, \overline{y}) = (x_1 \leftrightarrow y_1) \ \wedge \ \ldots \ \wedge \ (x_n \leftrightarrow y_n)$

has $3{\cdot}n + 2$ vertices under ordering $x_1 < y_1 < \ldots < x_n < y_n$

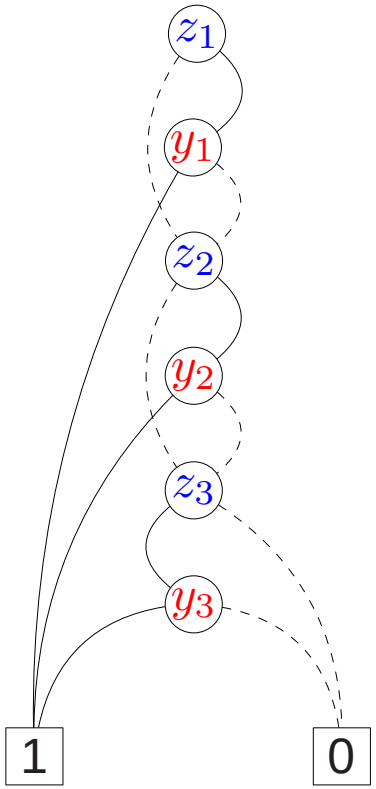
Another function with an exponential ROBDD



ROBDD for $f_3(\overline{z}, \overline{y}) \;=\; (z_1 \wedge y_1) \;\vee\; (z_2 \wedge y_2) \;\vee\; (z_3 \wedge y_3)$

for the variable ordering $z_1 < z_2 < z_3 < y_1 < y_2 < y_3$

And an optimal linear ROBDD



- ROBDD for $f_3(\cdot) = (z_1 \wedge y_1) \vee (z_2 \wedge y_2) \vee (z_3 \wedge y_3)$
- for ordering $z_1 < y_1 < z_2 < y_2 < z_3 < y_3$
- as all variables are essential for f, this ROBDD is optimal
- that is, for no variable ordering a smaller ROBDD exists

Symmetric functions

$f \in \mathit{Eval}(z_1, \ldots, z_m)$ is symmetric if and only if

$$f([z_1 = b_1, \ldots, z_m = b_m]) \;=\; f([z_1 = b_{i_1}, \ldots, z_m = b_{i_m}])$$

for each permutation $(i_1, \ldots, i_m)$ of $(1, \ldots, m)$

E.g.: $z_1 \vee z_2 \vee \ldots \vee z_m$, $z_1 \wedge z_2 \wedge \ldots \wedge z_m$, the parity function, and the majority function

If f is a symmetric function with m essential variables, then
for each variable ordering $\wp$ the $\wp$-ROBDD has size $\mathcal{O}(m^2)$

The even parity function

$f_{even}(x_1, \ldots, x_n) = 1$ iff the number of variables x_i with value 1 is even

truth table or propositional formula for f_{even} has exponential size

but an ROBDD of linear size is possible

The multiplication function

- Consider two n-bit integers
  - let $b_{n-1}b_{n-2}\ldots b_0$ and $c_{n-1}c_{n-2}\ldots c_0$
  - where b_{n-1} is the most significant bit, and b_0 the least significant bit
- Multiplication yields a $2n$-bit integer
  - the ROBDD $\mathfrak{B}_{f_{n-1}}$ has at least 1.09^n vertices
  - where f_{n-1} denotes the $(n-1)$-st output bit of the multiplication

Optimal variable ordering

- The size of ROBDDs is dependent on the variable ordering
- Is it possible to determine $\wp$ such that the ROBDD has minimal size?
  - to check whether a variable ordering is optimal is NP-hard
  - polynomial reduction from the 3SAT problem [Bollig & Wegener, 1996]
- There are many switching functions with large ROBDDs
  - for almost all switching functions the minimal size is in $\Omega(\frac{2^n}{n})$
- How to deal with this problem in practice?
  - guess a variable ordering in advance
  - rearrange the variable ordering during the ROBDD manipulations
  - not necessary to test all $n!$ orderings, best known algorithm in $\mathcal{O}(3^n \cdot n^2)$

Variable swapping

Sifting algorithm

[Rudell, 1993]

Dynamic variable ordering using variable swapping:

1. Select a variable x_i in OBDD at hand
2. Successively swap x_i to determine *size*$(\mathfrak{B})$ at any position for x_i
3. Shift x_i to position for which *size*$(\mathfrak{B})$ is minimal
4. Go back to the first step until no improvement is made

- Characteristics:
  - a variable may change position several times during a single sifting iteration
  - often yields a local optimum, but works well in practice

Interleaved variable ordering

- Which variable ordering to use for transition relations?
- The interleaved variable ordering:
  - for encodings $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ of state s and t respectively:

$$x_1 < y_1 < x_2 < y_2 < \ldots < x_n < y_n$$

- This variable ordering yields compact ROBDDs for binary relations
  - for transition relation with $z_1 \ldots z_m$ be the encoding of action α, take:

$$\underbrace{z_1 < z_2 < \ldots < z_m}_{\text{encoding of } \alpha} < \underbrace{x_1 < y_1 < x_2 < y_2 < \ldots < x_n < y_n}_{\text{interleaved order of states}}$$