IA169 Model Checking

Symbolic model checking for CTL

Jan Strejček

Faculty of Informatics
Masaryk University

Motivation

- system states typically correspond to variable assignments
- for finite systems, the number of variables is finite and their domains are finite
- we can assume that state is an assignment $s : V \to \{0, 1\}$, where V is a finite set of state variables

Motivation

- system states typically correspond to variable assignments
- for finite systems, the number of variables is finite and their domains are finite
- we can assume that state is an assignment $s : V \to \{0, 1\}$, where V is a finite set of state variables
- a set of states can be described by propositional formulas
- for example, a set of states where the values of two bitvectors of length 2 agree (i.e. $x_1 x_2 = y_1 y_2$) can be represented by

Motivation

- system states typically correspond to variable assignments
- for finite systems, the number of variables is finite and their domains are finite
- we can assume that state is an assignment $s : V \to \{0, 1\}$, where V is a finite set of state variables
- a set of states can be described by propositional formulas
- for example, a set of states where the values of two bitvectors of length 2 agree (i.e. $x_1 x_2 = y_1 y_2$) can be represented by

$$(x_1 \wedge y_1 \wedge x_2 \wedge y_2) \vee (x_1 \wedge y_1 \wedge \neg x_2 \wedge \neg y_2) \vee$$
$$\vee (\neg x_1 \wedge \neg y_1 \wedge x_2 \wedge y_2) \vee (\neg x_1 \wedge \neg y_1 \wedge \neg x_2 \wedge \neg y_2)$$

or by $x_1 \Leftrightarrow y_1 \wedge x_2 \Leftrightarrow y_2$

- such a formula can be equivalently seen as a Boolean funcion
- aternatively, a set can be described by a binary decision diagram (BDD)

Agenda and sources

agenda

- binary decision diagrams (BDDs) and their properties
- Kripke structures represented by BDDs
- CTL model checking algorithm based on BDDs

source

- Chapter 8 of E. M. Clarke, O. Grumberg, D. Kroening, D. Peled, and H. Veith: *Model Checking*, Second Edition, MIT, 2018.

Binary decision diagrams (BDDs) and their properties

Binary decision diagrams (BDDs)

- *"one of the only really fundamental data structures that came out in the last twenty-five years"* [Donald Knuth, 2008]
- investigated by Randal Bryant in 1986
- can represent an arbitrary Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ or the set of models of a propositional formula φ

Binary decision diagrams (BDDs)

- *"one of the only really fundamental data structures that came out in the last twenty-five years"* [Donald Knuth, 2008]
- investigated by Randal Bryant in 1986
- can represent an arbitrary Boolean function $f : \{0, 1\}^n \to \{0, 1\}$ or the set of models of a propositional formula φ

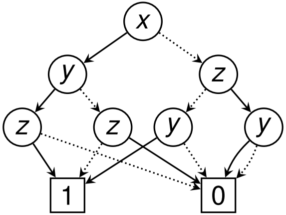
Definition (binary decision diagram, BDD)

A binary decision diagram (BDD) is a finite rooted directed acyclic graph with two kinds of nodes and two kinds of edges:

- each terminal (i.e., a node without any successor) is labeled with 0 or 1,
- each nonterminal node v is labeled with a variable $var(v)$ and has a low successor $low(v)$ and a high successor $high(v)$.

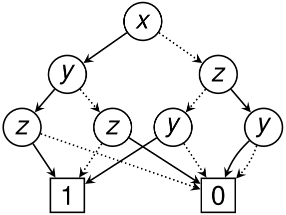
Binary decision diagrams (BDDs)

- $low(v) = w$ is depicted by a dashed/dotted edge from v to w
- $high(v) = w$ is depicted by a solid edge from v to w
- nodes are directly labeled with $var(v)$, terminal nodes with 0 or 1

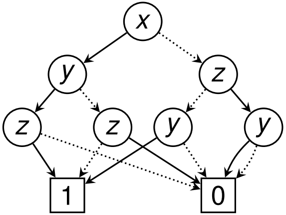


Semantics of BDDs

- a BDD with variables $x_1, \ldots, x_n$ describes a Boolean function $f(x_1, \ldots, x_n)$
- for $b_1, \ldots, b_n \in \{0, 1\}$, the value of $f(b_1, \ldots, b_n)$ is the value of the terminal node reached from the root by following
  - $low(v)$ whenever $var(v) = x_i$ and $b_i = 0$
  - $high(v)$ whenever $var(v) = x_i$ and $b_i = 1$



- a BDD with variables $x_1, \ldots, x_n$ describes a Boolean function $f(x_1, \ldots, x_n)$
- for $b_1, \ldots, b_n \in \{0, 1\}$, the value of $f(b_1, \ldots, b_n)$ is the value of the terminal node reached from the root by following
  - $low(v)$ whenever $var(v) = x_i$ and $b_i = 0$
  - $high(v)$ whenever $var(v) = x_i$ and $b_i = 1$



$$f(x, y, z) = \begin{cases} 1 & \text{for } x = 1,\ y = 1,\ z = 1 \\ & \text{or } x = 1,\ y = 0,\ z = 0 \\ & \text{or } x = 0,\ z = 0,\ y = 1 \\ 0 & \text{otherwise} \end{cases}$$

Semantics of BDDs

Definition

Consider a BDD labeled with (some of) variables $x_1, \ldots, x_n$. Every node v of the BDD describes a Boolean function $f_v(x_1, \ldots, x_n)$ defined inductively as follows.

- if v is a terminal node labeled with 0, then $f_v(x_1, \ldots, x_n) = 0$
- if v is a terminal node labeled with 1, then $f_v(x_1, \ldots, x_n) = 1$
- if v is a nonterminal node labeled with a variable x_i, then

$$f_v(x_1, \ldots, x_n) = (\neg x_i \wedge f_{low(v)}(x_1, \ldots, x_n)) \vee (x_i \wedge f_{high(v)}(x_1, \ldots, x_n))$$

The BDD represents the Boolean function corresponding to its root.

Semantics of BDDs

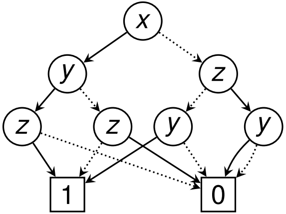
Definition

Consider a BDD labeled with (some of) variables $x_1, \ldots, x_n$. Every node v of the BDD describes a Boolean function $f_v(x_1, \ldots, x_n)$ defined inductively as follows.

- if v is a terminal node labeled with 0, then $f_v(x_1, \ldots, x_n) = 0$
- if v is a terminal node labeled with 1, then $f_v(x_1, \ldots, x_n) = 1$
- if v is a nonterminal node labeled with a variable x_i, then

$$f_v(x_1, \ldots, x_n) = (\neg x_i \wedge f_{low(v)}(x_1, \ldots, x_n)) \vee (x_i \wedge f_{high(v)}(x_1, \ldots, x_n))$$

The BDD represents the Boolean function corresponding to its root.



Semantics of BDDs

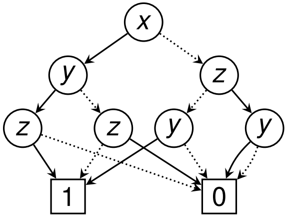
Definition

Consider a BDD labeled with (some of) variables $x_1, \ldots, x_n$. Every node v of the BDD describes a Boolean function $f_v(x_1, \ldots, x_n)$ defined inductively as follows.

- if v is a terminal node labeled with 0, then $f_v(x_1, \ldots, x_n) = 0$
- if v is a terminal node labeled with 1, then $f_v(x_1, \ldots, x_n) = 1$
- if v is a nonterminal node labeled with a variable x_i, then

$$f_v(x_1, \ldots, x_n) = (\neg x_i \wedge f_{low(v)}(x_1, \ldots, x_n)) \vee (x_i \wedge f_{high(v)}(x_1, \ldots, x_n))$$

The BDD represents the Boolean function corresponding to its root.

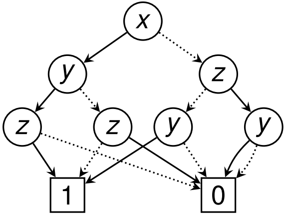


$$f(x, y, z) =$$
$$= (x \wedge (y \iff z)) \vee (\neg x \wedge \neg z \wedge y)$$

Ordered BDD

Definition (ordered BDD)

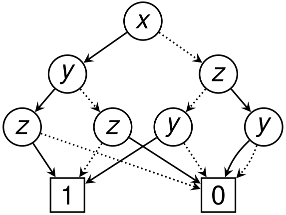
A BDD is ordered if there exists a linear ordering $<$ on its variables such that for every node v with a nonterminal successor w it holds $var(v) < var(w)$.

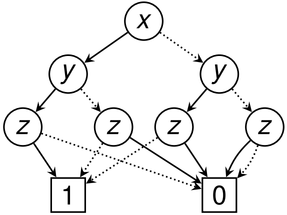


Ordered BDD

Definition (ordered BDD)

A BDD is ordered if there exists a linear ordering $<$ on its variables such that for every node v with a nonterminal successor w it holds $var(v) < var(w)$.

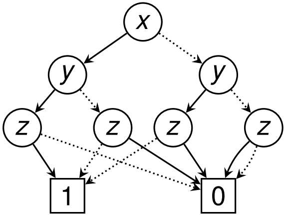




Reduced BDD

Definition (reduced BDD)

A BDD is reduced if it does not contain any nonterminal node with identical low and high child and any isomorphic subgraphs.



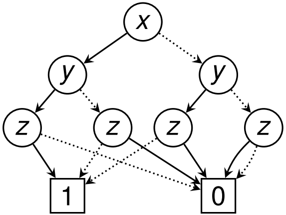
Reduced BDD

Definition (reduced BDD)

A BDD is reduced if it does not contain any nonterminal node with identical low and high child and any isomorphic subgraphs.

a BDD can be reduced by repeated applications of the following steps

1. merge all terminal nodes with the same label
2. remove each nonterminal node v with $low(v) = high(v)$ and redirect all incomming edges to $low(v)$
3. merge each pair v, w of nonterminal nodes satisfying $var(v) = var(w)$, $low(v) = low(w)$, and $high(v) = high(w)$



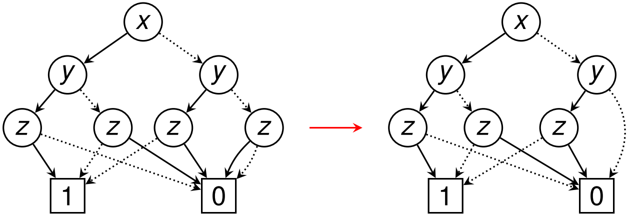
Reduced BDD

Definition (reduced BDD)

A BDD is reduced if it does not contain any nonterminal node with identical low and high child and any isomorphic subgraphs.

a BDD can be reduced by repeated applications of the following steps

1. merge all terminal nodes with the same label
2. remove each nonterminal node v with $low(v) = high(v)$ and redirect all incomming edges to $low(v)$
3. merge each pair v, w of nonterminal nodes satisfying $var(v) = var(w)$, $low(v) = low(w)$, and $high(v) = high(w)$



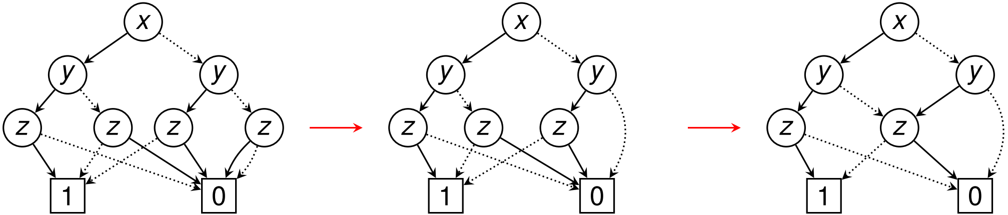
Reduced BDD

Definition (reduced BDD)

A BDD is reduced if it does not contain any nonterminal node with identical low and high child and any isomorphic subgraphs.

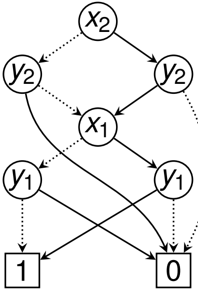
a BDD can be reduced by repeated applications of the following steps

1. merge all terminal nodes with the same label
2. remove each nonterminal node v with $low(v) = high(v)$ and redirect all incomming edges to $low(v)$
3. merge each pair v, w of nonterminal nodes satisfying $var(v) = var(w)$, $low(v) = low(w)$, and $high(v) = high(w)$



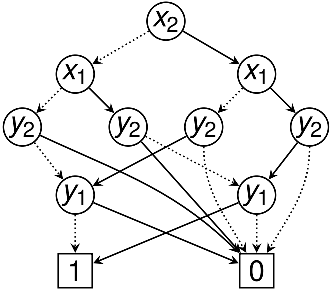
Properties of BDDs

- we assume that all BDDs are reduced and ordered
- for a fixed variable order, BDDs are a canonical representation of Boolean functions, i.e., two Boolean functions are equivalent (regardless their description) iff the corresponding BDDs are isomorphic
- BDD size heavily depends on considered variable order



$x_2 < y_2 < x_1 < y_1$

"$x_1 x_2 = y_1 y_2$"



$x_2 < x_1 < y_2 < y_1$

- some BDDs are exponential in the number of variables regardless their order

Operations on BDDs

variable instantiation

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

operation on the corresponding BDD

1. if the root r is labeled with x_i then the new BDD will have root
   - $low(r)$ if $b = 0$
   - $high(r)$ if $b = 1$
2. going from top to bottom, any edge leading to a nonterminal node v labeled with x_i is reconnected to
   - $low(v)$ if $b = 0$
   - $high(v)$ if $b = 1$
3. unreachable nodes are removed and BDD is reduced