Intelligent Systems Programming

Lecture 5: **Boolean Expression Representations & Binary Decision Diagrams (BDDs)**

Today's Program

- **[10:00-10:40] Classical representations**
  - Boolean expressions and Boolean functions
  - Desirable properties of representations of Boolean functions
  - Classical representations of Boolean expressions
    - Truth tables
    - CNF and DNF
- **[10:50-11:50] Binary Decision Diagrams**
  - If-then-else normal form (INF)
  - Decision trees
  - Ordered Binary Decision Diagrams (OBDDs)
  - Reduced Ordered Binary Decision Diagrams (ROBDDs / BDDs)

Boolean Expressions

- Boolean Expressions

  $t \ ::= x \mid 0 \mid 1 \mid \neg t \mid t \wedge t \mid t \vee t \mid t \Rightarrow t \mid t \Leftrightarrow t$

- Precedence

  $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

- Set of truth values

  $\boldsymbol{B} = \{0,1\}$

- Truth assignments

  e.g. $t\,[0/x,\, 1/y]$

Boolean Functions

- Boolean expression E defines **boolean function** $f : \boldsymbol{B}^n \rightarrow \boldsymbol{B}$, where

$$f(x_1,x_2,\ldots,x_n) = E(x_1,x_2,\ldots,x_n)$$

- Above, f is an n-ary function

- **Example**

$$f(x_1,x_2,x_3) = x_1 \Leftrightarrow \neg x_2$$

Properties of Boolean Functions

- Equality

  $f = g \quad \text{iff} \quad \forall \boldsymbol{x} \,.\, f(\boldsymbol{x}) = g(\boldsymbol{x})$

- Order of arguments matter

  $f(x,y) = x \Rightarrow y \qquad \neq \qquad g(y,x) = x \Rightarrow y$

- Several expressions may represent same function

  $f(x,y) = \; x \Rightarrow y \; = \neg x \vee y = (\neg x \vee y) \wedge (\neg x \vee x) = \ldots$

Number of Boolean Functions

Number of Boolean functions $f : \boldsymbol{B}^n \rightarrow \boldsymbol{B}$

x_1	...	x_n	f
0	...	0	$f(0,\ldots,0)$
0	...	1	$f(0,\ldots,1)$
0	...	0	$f(0,\ldots,0)$
0	...	1	$f(0,\ldots,1)$
...	...	...	...
1	...	0	$f(1,\ldots,0)$
1	...	1	$f(1,\ldots,1)$
1	...	0	$f(1,\ldots,0)$
1	...	1	$f(1,\ldots,1)$
...	...	...	...

$$2^{\left(2^n\right)}$$

Representation of Boolean functions

Desirable properties:

1. Compact
2. Equality check easy
3. Easy to evaluate the truth-value of an assignment
4. Boolean operations efficient
5. SAT check efficient
6. Tautology check efficient
7. Canonicity: exactly one representation of each Boolean function. - Solves 2, 5, and 6, why?

Compact representations are rare

- $2^{\left(2^n\right)}$ boolean functions in *n* variables…
  - How do we find a single compact representation for them all?
- The fraction of Boolean functions of *n* variables with a polynomial size in *n* → 0 for *n* → ∞

Curse of Boolean function representations:

This problem exists for all representations we know!

Classical Representations of Boolean Functions

Truth tables

- Compact ☹ table size 2^n
- Equality check easy ☺ canonical
- Easy to evaluate the truth-value of an assignment ☺ log m or constant
- Boolean operations efficient ☺ linear
- SAT check efficient ☺ linear
- Tautology check efficient ☺ linear

x	y	z	$x \wedge y \vee z$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

DNF and CNF

- Is there a DNF and CNF of every expression?
- Given a truth table representation of a Boolean formula, can we easily define a DNF and CNF of the formula?

x	*y*	*z*	*e*
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

DNF from on-tuples

- Example DNF of *e* – use *on-tuples*

x	y	z	e	
0	0	0	0	
0	0	1	1	$\neg x \wedge \neg y \wedge z \ \vee$
0	1	0	0	
0	1	1	1	$\neg x \wedge y \wedge z \ \vee$
1	0	0	0	
1	0	1	1	$x \wedge \neg y \wedge z \ \vee$
1	1	0	1	$x \wedge y \wedge \neg z \ \vee$
1	1	1	1	$x \wedge y \wedge z$

CNF from off-tuples

- Example CNF of *e* - use *off-tuples*

x	y	z	e	
0	0	0	0	$\neg(\neg x \wedge \neg y \wedge \neg z)\ \wedge$
0	0	1	1	
0	1	0	0	$\neg(\neg x \wedge y \wedge \neg z)\ \wedge$
0	1	1	1	
1	0	0	0	$\neg(x \wedge \neg y \wedge \neg z)\ \wedge$
1	0	1	1	
1	1	0	1	
1	1	1	1	

CNF from off-tuples

- Example CNF of *e* - use *off-tuples*

x	y	z	e	
0	0	0	0	$(x \vee y \vee z) \wedge$
0	0	1	1	
0	1	0	0	$(x \vee \neg y \vee z) \wedge$
0	1	1	1	
1	0	0	0	$(\neg x \vee y \vee z)$
1	0	1	1	
1	1	0	1	
1	1	1	1	

DNF and CNF

Every Boolean formula has a DNF and CNF representation

- The special version DNF and CNF representations produced from *on* and *off*-tuples are canonical and called cDNF and cCNF
- Are cDNF and cCNF minimum size DNF and CNF representations?

Checking DNF and CNF-formulas

- Symmetry properties of DNF and CNF

	SAT	Tautology
CNF	NP complete	Polynomial (exercise)
DNF	Polynomial (exercise)	Co-NP complete

- Idea: Solve CNF-SAT by conversion to DNF-SAT
  – Problem: conversion between CNF and DNF may be exponential

Converting CNF to DNF

- Example
  - CNF $\left(x_0^1 \vee x_1^1\right) \wedge \left(x_0^2 \vee x_1^2\right) \wedge \cdots \wedge \left(x_0^n \vee x_1^n\right)$
  - Corresponding DNF blows up

$$
\begin{gathered}
\left(x_0^1 \wedge x_0^2 \wedge \cdots \wedge x_0^{n-1} \wedge x_0^n\right) \vee \\
\left(x_0^1 \wedge x_0^2 \wedge \cdots \wedge x_0^{n-1} \wedge x_1^n\right) \vee \\
\vdots \\
\left(x_1^1 \wedge x_1^2 \wedge \cdots \wedge x_1^{n-1} \wedge x_0^n\right) \vee \\
\left(x_1^1 \wedge x_1^2 \wedge \cdots \wedge x_1^{n-1} \wedge x_1^n\right)
\end{gathered}
$$

Binary Decision Diagrams

If-then-else operator

- The *if-then-else* Boolean operator is defined by

$$x \to y_1 , y_0 \;\equiv\; (x \wedge y_1) \vee (\neg x \wedge y_0)$$

- We have

$$(x \to y_1 , y_0)\,[1/x] \;\equiv\; (1 \wedge y_1) \vee (0 \wedge y_0) \equiv y_1$$

$$(x \to y_1 , y_0)\,[0/x] \;\equiv\; (0 \wedge y_1) \vee (1 \wedge y_0) \equiv y_0$$

- What is $x \to 1 , 0$ equivalent to? And $x \to 0 , 1$?

If-then-else operator

- All operators in propositional logic can be expressed using only $\rightarrow$ operators with:
  - $\rightarrow$ expressions and 0 and 1 for y_1 and y_0
  - tests on un-negated variables
  - Variables only as tests
- What are *if-then-else* expressions for
  - x, $\neg x$
  - $x \wedge y$
  - $x \vee y$
  - $x \Rightarrow y$

If-then-else Normal Form (INF)

An *if-then-else* Normal Form (INF) is a Boolean expression build entirely from the if-then-else operator and the constants 0 and 1 such that all test are performed only on un-negated variables

- **Proposition**: any Boolean expression t is equivalent to an expression in INF

  Proof:

  $t \equiv x \rightarrow t[1/x], t[0/x]$ (Shannon expansion of t)

  Apply the Shannon expansion recursively on t. The recursion must terminate in 0 or 1, since the number of variables is finite

Shannon Expansion

Expression t with 4 variables: t x_1,x_2,x_3,x_4

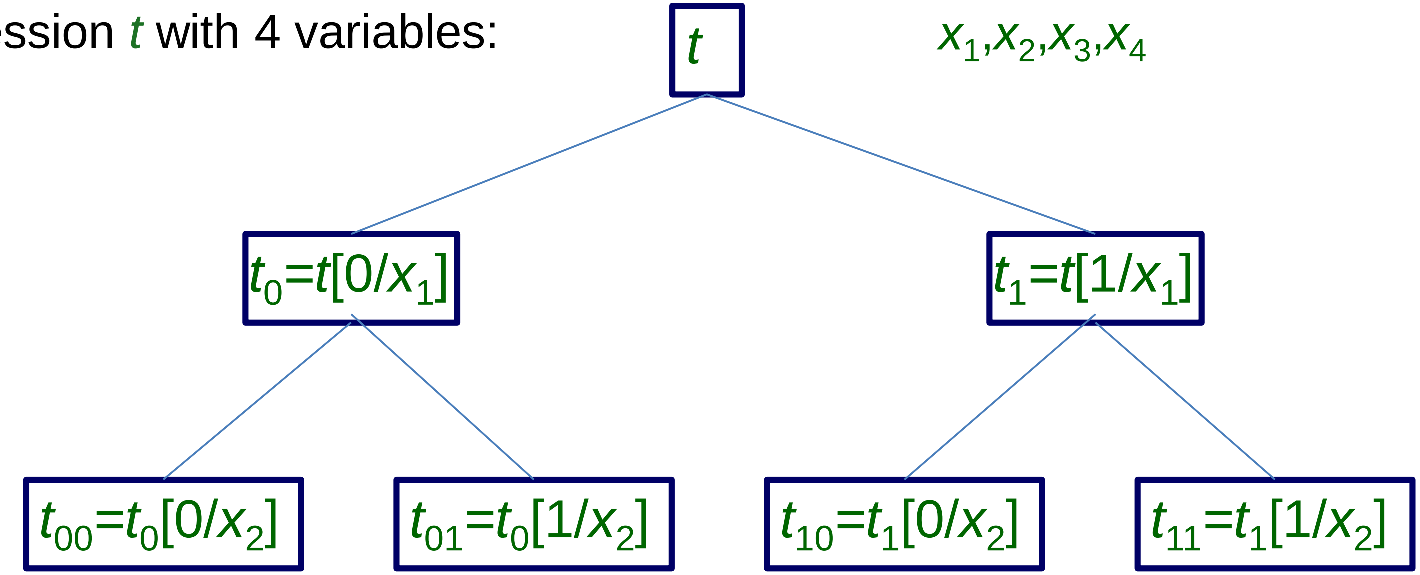
$t_0=t[0/x_1]$ $t_1=t[1/x_1]$

$$t \equiv x_1 \rightarrow t_1,\ t_0$$

Shannon Expansion

Expression t with 4 variables:

x_1, x_2, x_3, x_4

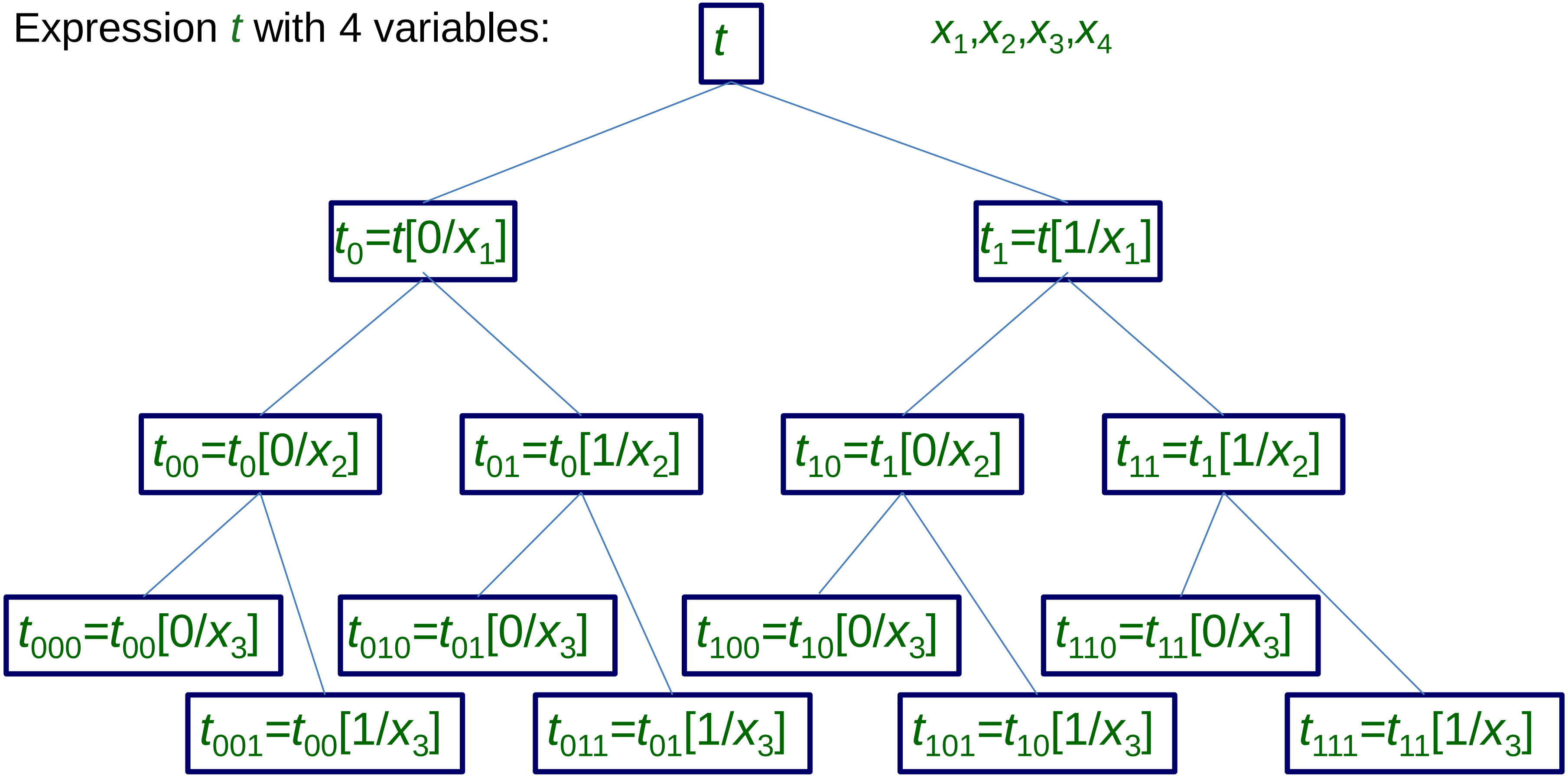


$$t \equiv x_1 \rightarrow t_1,\ t_0$$
$$t_0 \equiv x_2 \rightarrow t_{01},\ t_{00}$$
$$t_1 \equiv x_2 \rightarrow t_{11},\ t_{10}$$

Shannon Expansion

Expression t with 4 variables:

x_1, x_2, x_3, x_4



$t \equiv x_1 \to t_1, t_0$

$t_0 \equiv x_2 \to t_{01}, t_{00}$

$t_1 \equiv x_2 \to t_{11}, t_{10}$

$t_{00} \equiv x_3 \to t_{001}, t_{000}$

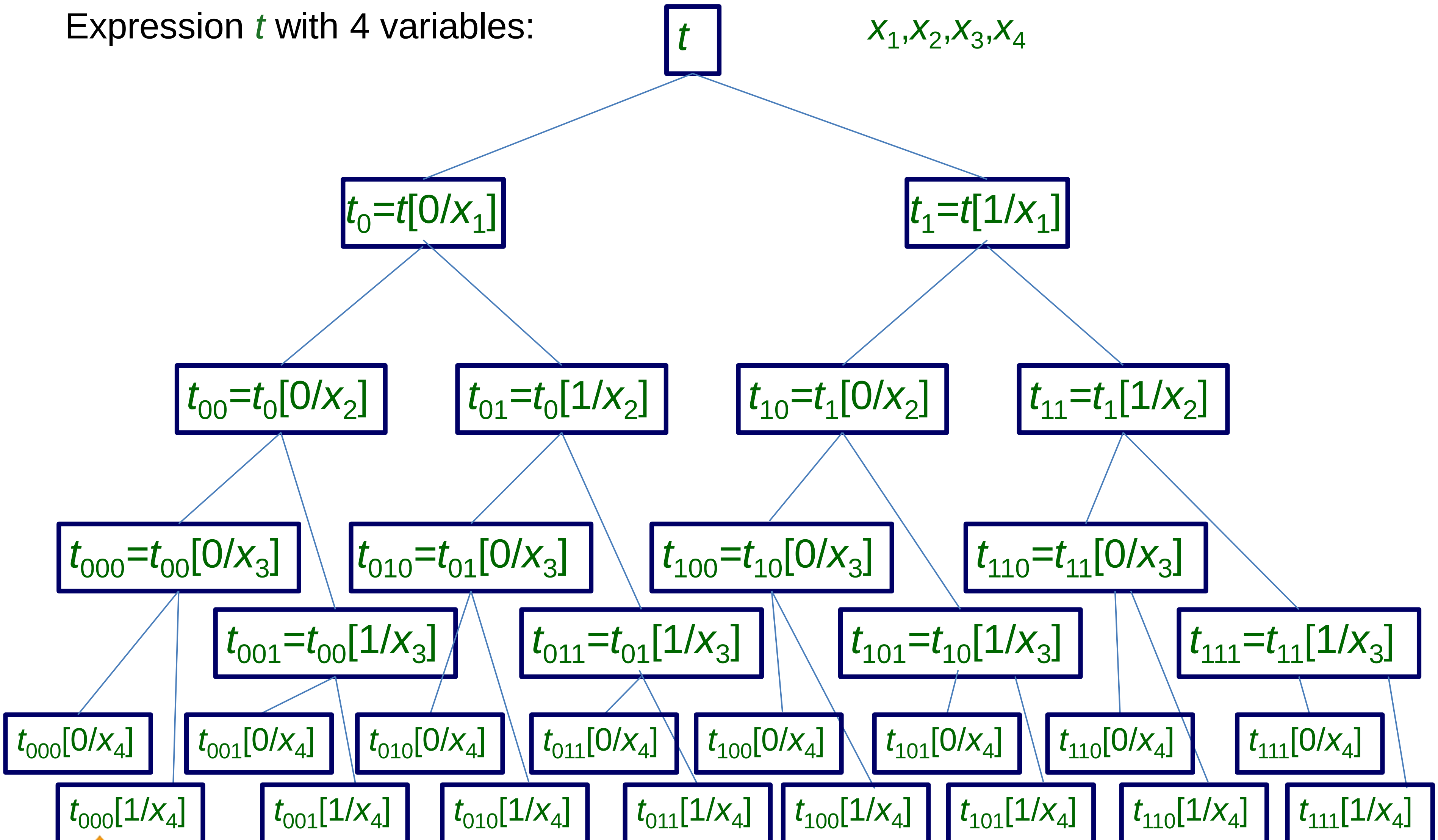
$t_{01} \equiv x_3 \to t_{011}, t_{010}$

$t_{10} \equiv x_3 \to t_{101}, t_{110}$

$t_{11} \equiv x_3 \to t_{111}, t_{110}$

Shannon Expansion

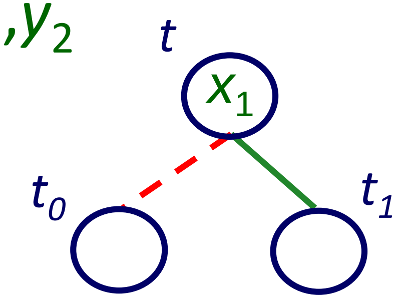
Expression t with 4 variables:



Example

- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1,y_1,x_2,y_2

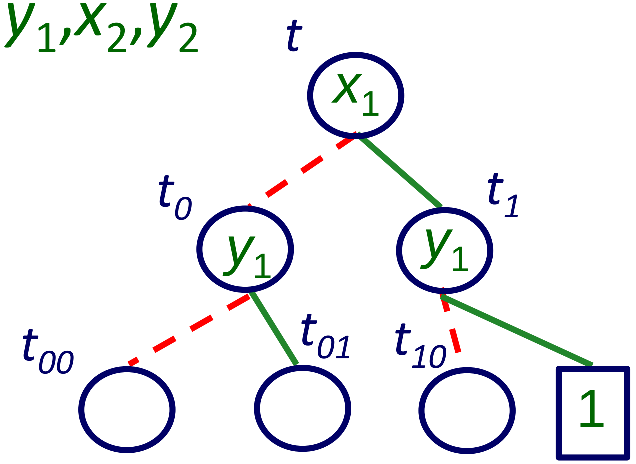
$t \quad \equiv x_1 \rightarrow t_1, t_0$



Example

- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1, y_1, x_2, y_2

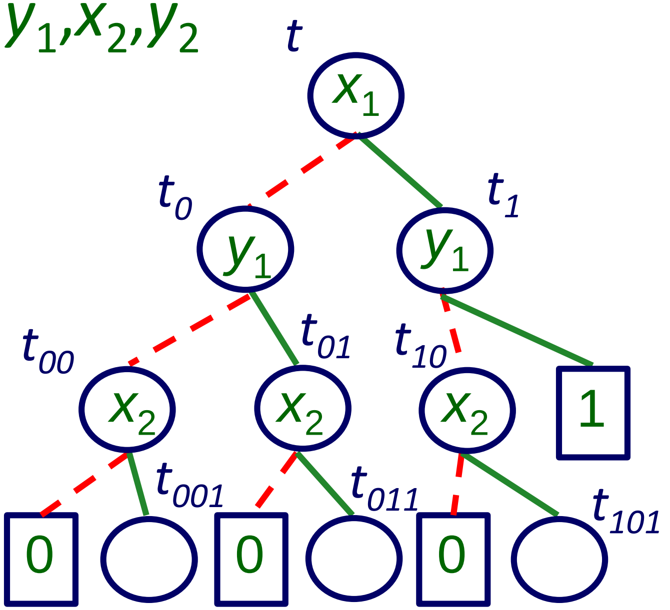
$$t \equiv x_1 \rightarrow t_1, t_0$$
$$t_0 \equiv y_1 \rightarrow t_{01}, t_{00}$$
$$t_1 \equiv y_1 \rightarrow 1, t_{10}$$



Example

- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1, y_1, x_2, y_2

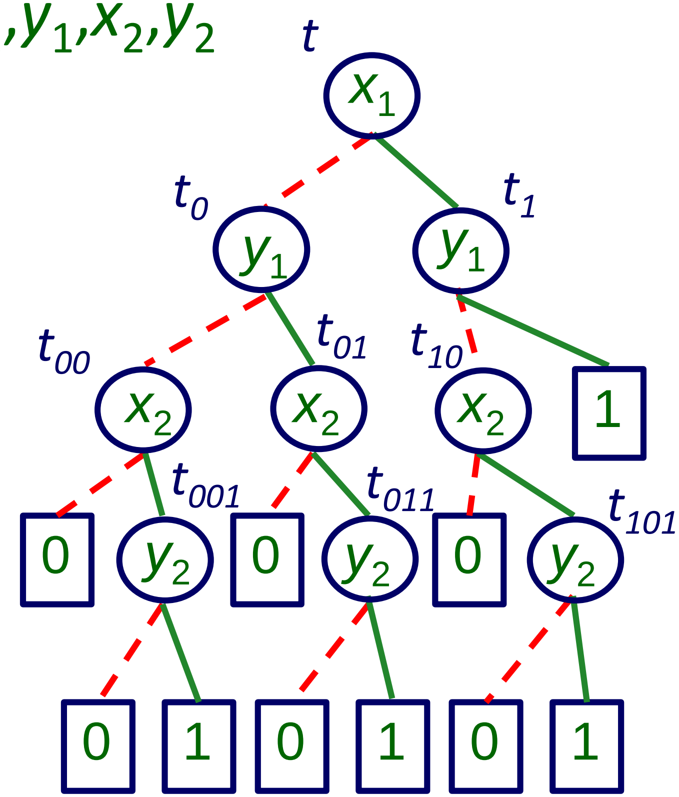
$$
\begin{aligned}
t &\equiv x_1 \to t_1, t_0 \\
t_0 &\equiv y_1 \to t_{01}, t_{00} \\
t_1 &\equiv y_1 \to 1, t_{10} \\
t_{01} &\equiv x_2 \to t_{011}, 0 \\
t_{00} &\equiv x_2 \to t_{001}, 0 \\
t_{10} &\equiv x_2 \to t_{101}, 0
\end{aligned}
$$



Decision Tree

- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1, y_1, x_2, y_2

$$
\begin{aligned}
t &\equiv x_1 \to t_1, t_0 \\
t_0 &\equiv y_1 \to t_{01}, t_{00} \\
t_1 &\equiv y_1 \to 1, t_{10} \\
t_{01} &\equiv x_2 \to t_{011}, 0 \\
t_{00} &\equiv x_2 \to t_{001}, 0 \\
t_{10} &\equiv x_2 \to t_{101}, 0 \\
t_{011} &\equiv y_2 \to 1,0 \\
t_{001} &\equiv y_2 \to 1,0 \\
t_{101} &\equiv y_2 \to 1,0
\end{aligned}
$$

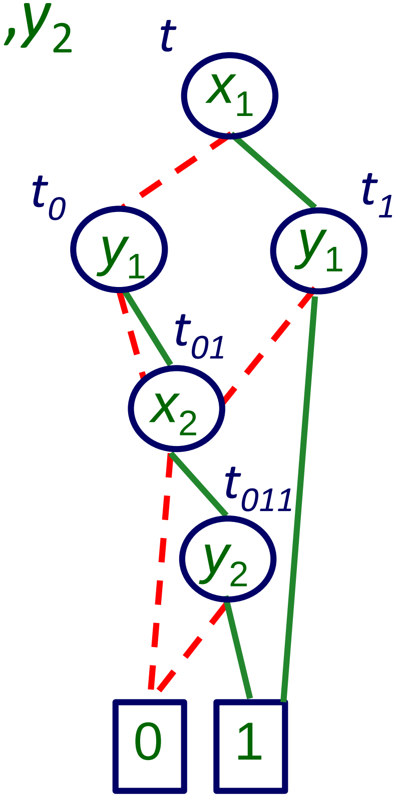


$$
\begin{aligned}
t = {} & x_1 \to (y_1 \to 1, (x_2 \to (y_2 \to 1,0), 0)), \\
& (y_1 \to (x_2 \to (y_2 \to 1,0), 0), (x_2 \to (y_2 \to 1,0), 0))
\end{aligned}
$$

Reduction I: substitute identical subtrees

- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1, y_1, x_2, y_2

$$t = x_1 \rightarrow t_1, t_0$$
$$t_0 = y_1 \rightarrow t_{01}, t_{01}$$
$$t_1 = y_1 \rightarrow 1, t_{01}$$
$$t_{01} = x_2 \rightarrow t_{011}, 0$$
$$t_{011} = y_2 \rightarrow 1,0$$

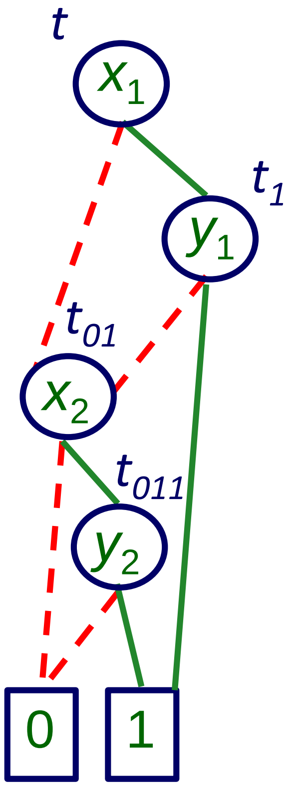


Result: an Ordered Binary Decision Diagram (OBDD)

Reduction II: remove redundant tests

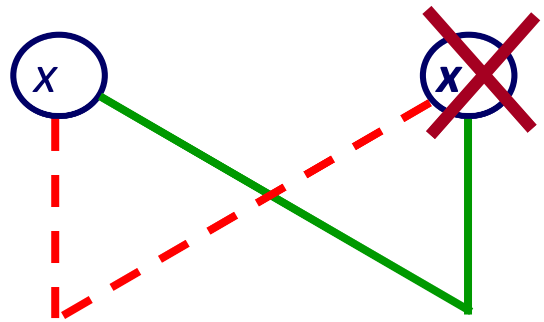
- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1,y_1,x_2,y_2

$$
\begin{aligned}
t &= x_1 \to t_1,\ t_{01} \\
t_1 &= y_1 \to 1,\ t_{01} \\
t_{01} &= x_2 \to t_{011},\ 0 \\
t_{011} &= y_2 \to 1,0
\end{aligned}
$$

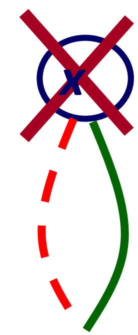


Result: a Reduced Ordered Binary Decision Diagram (ROBDD)
[often called a BDD]

Reductions

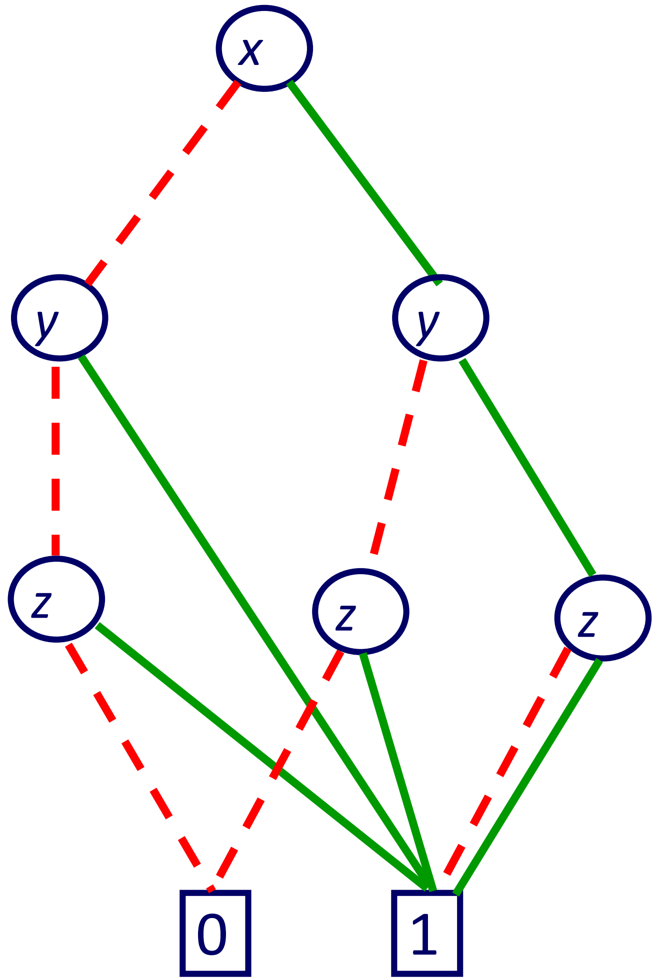


Uniqueness requirement

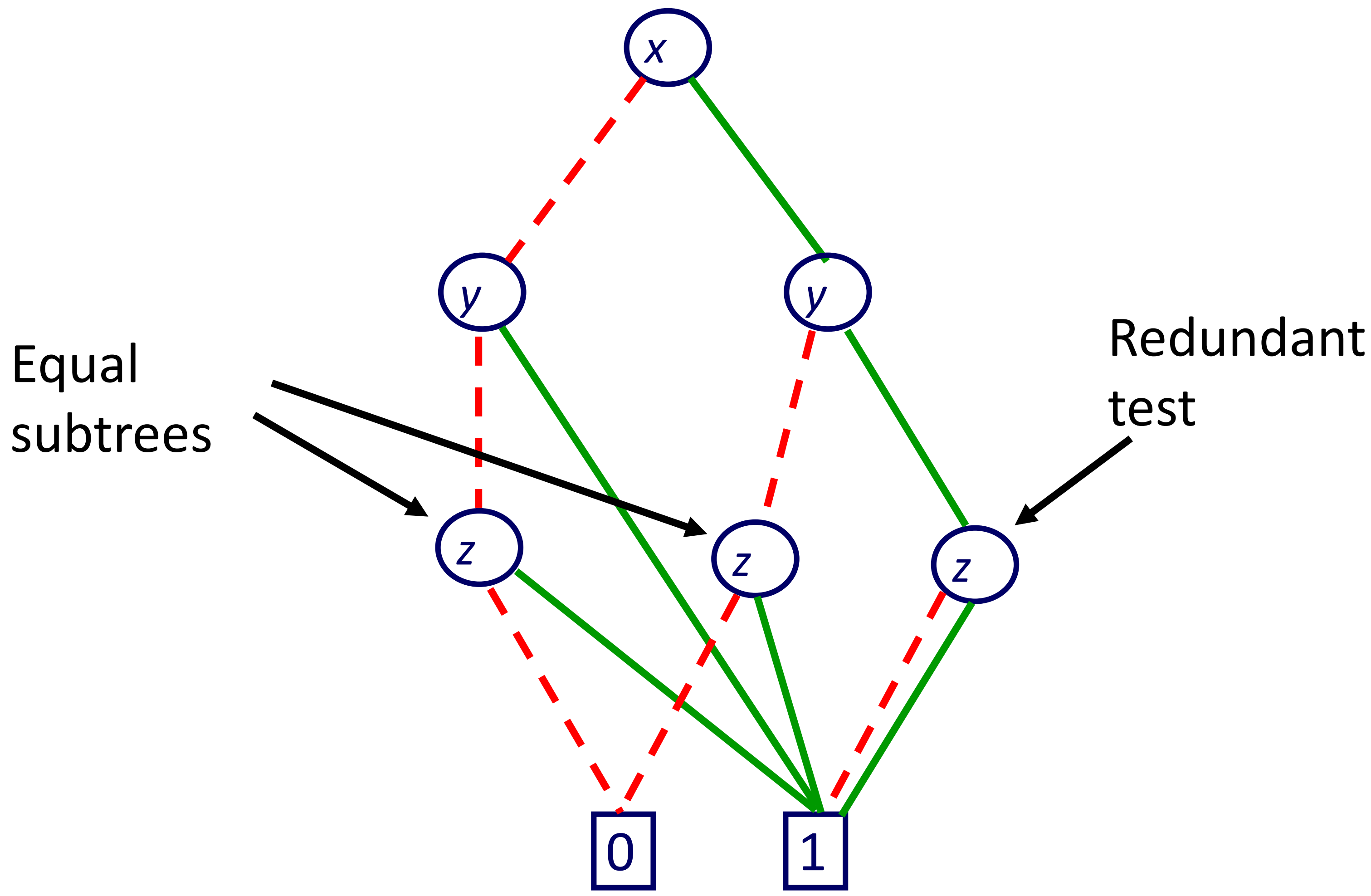


Non-redundant tests requirement

Another reduction example

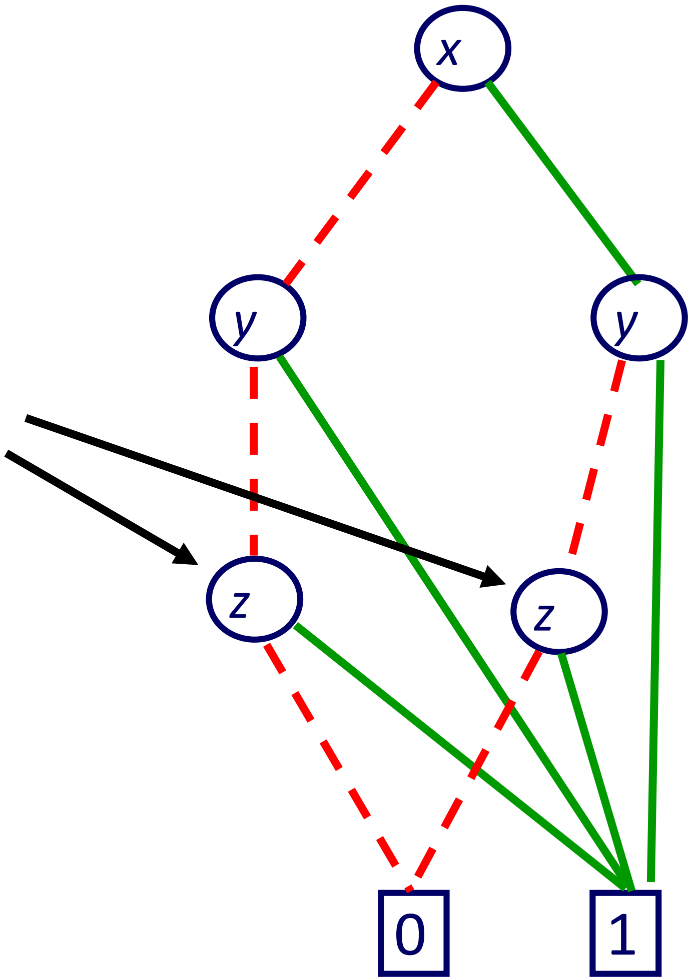


Another reduction example

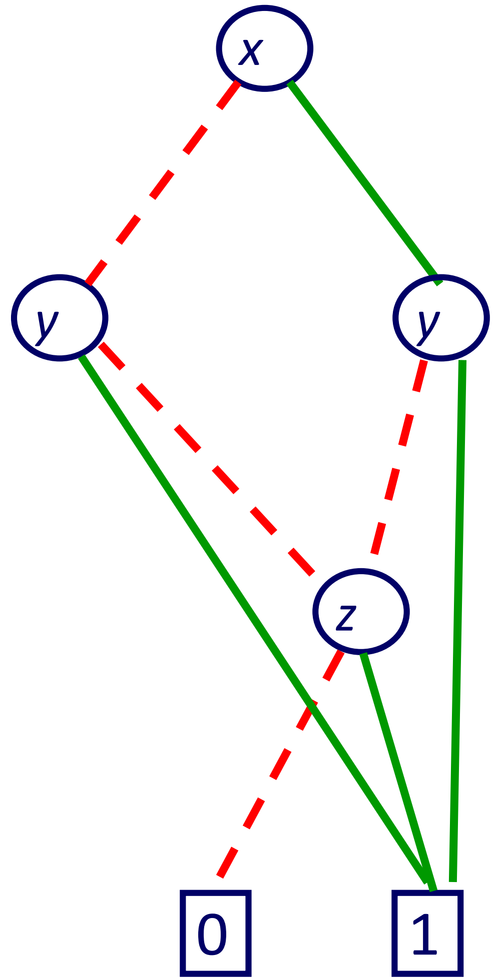


Another reduction example

Equal subtrees

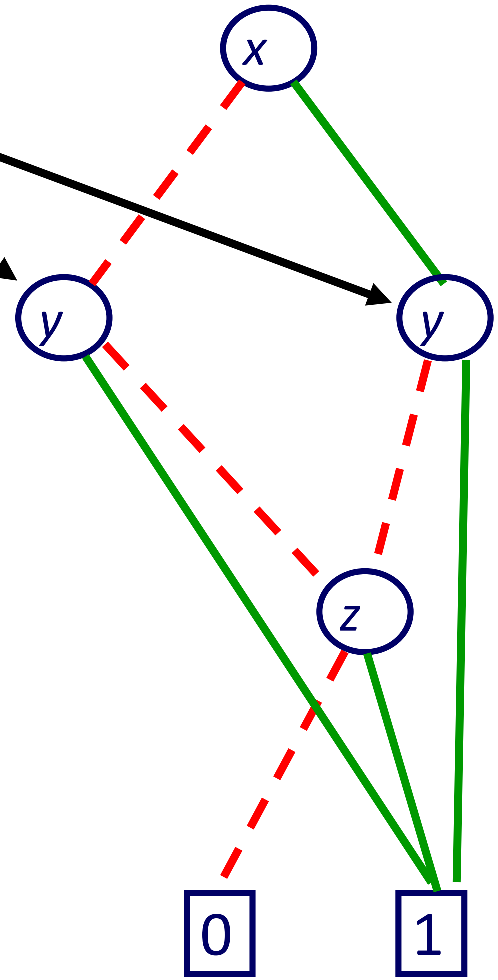


Another reduction example

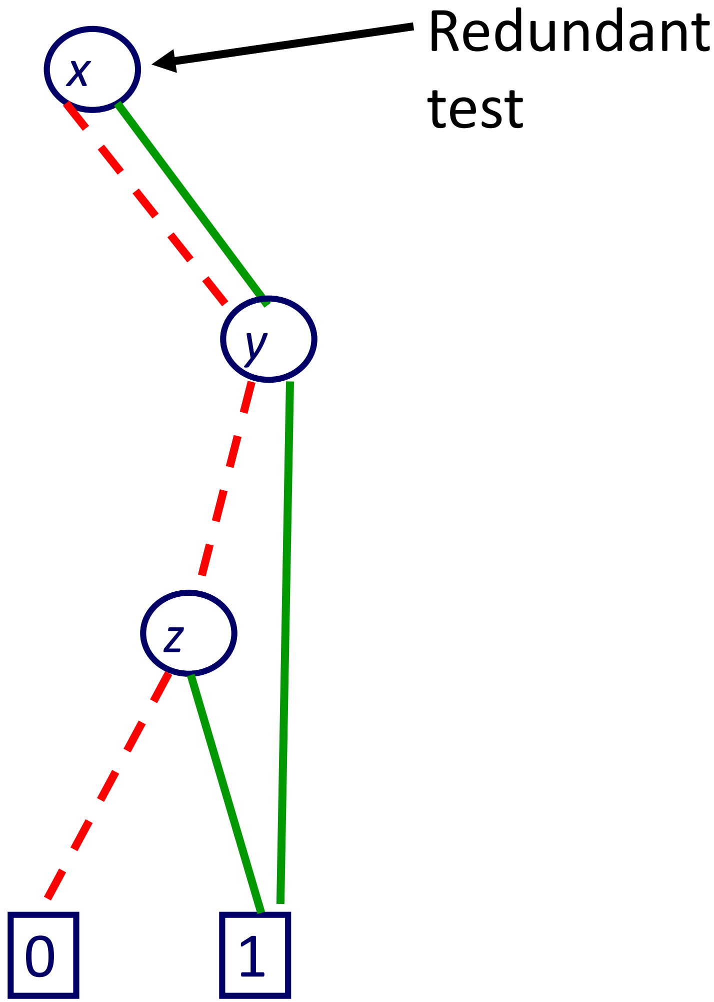


Another reduction example

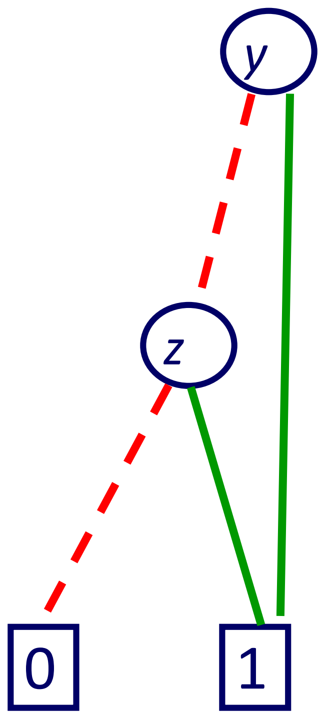
Equal subtrees



Another reduction example



Another reduction example



Canonicity of ROBDDs

- **Canonicity Lemma**: for any function $f : \mathbf{B}^n \rightarrow \mathbf{B}$ there is exactly one ROBDD u with a variable ordering $x_1 < x_2 < \ldots < x_n$ such that $f_u = f(x_1,\ldots,x_n)$

Proof (by induction on n)

Read on your own!

Practice

- What are the ROBDDs of
  - x
  - 1
  - 0
  - $x \wedge y$ order x, y
  - $(x \Rightarrow y) \wedge z$ order x, y, z

Size of ROBDDs

- ROBDDs of many practically important Boolean functions are small
- Do all functions have polynomial ROBDD size? NO
  - ROBDDs do not escape the curse of Boolean function representation

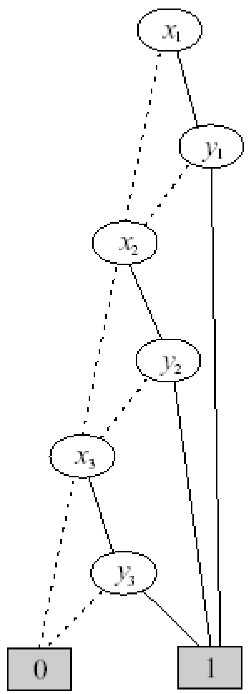
Size of ROBDDs

- The size of an ROBDD depends heavily on the variable ordering
- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Build ROBDD of t in order x_1, x_2, y_1, y_2

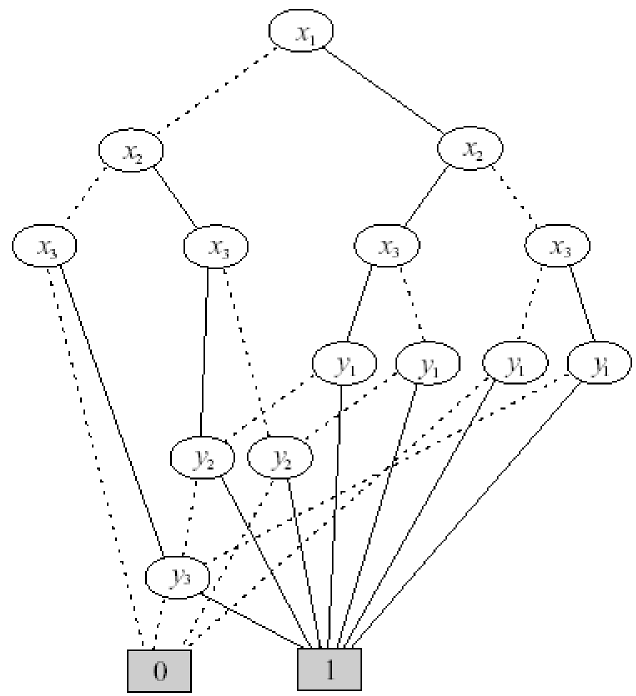
Size of ROBDDs

- The size of an ROBDD depends heavily on the variable ordering

$(x_1 \wedge y_1) \vee (x_2 \wedge y_2) \vee \ldots \vee (x_n \wedge y_n)$



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



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