Intelligent Systems Programming

Lecture 6: **Representations of Boolean Expressions & Binary Decision Diagrams (BDDs)**

IT University of Copenhagen

Today's Program

- **Representations**
  - Boolean expressions and Boolean functions
  - Desirable properties of representations of Boolean functions
  - Examples: Truth tables, cCNF.
- **Binary Decision Diagrams**
  - If-then-else normal form (INF)

  BREAK
  - Decision trees
  - Ordered Binary Decision Diagrams (OBDDs)
  - Reduced Ordered Binary Decision Diagrams (ROBDDs / BDDs)
  - Unique Table representation

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

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

- Equivalence: $\equiv$
- Set of truth values: $\boldsymbol{B} = \{0,1\}$
- Truth assignments

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

Boolean Functions

- A **boolean (n-ary) function**

$$f : \mathbf{B}^n \rightarrow \mathbf{B}$$

- Boolean expression E defines Boolean function

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

- **Examples**

$$f(x_1,x_2) = x_1 \vee x_2$$

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

Properties of Boolean Functions

- Equality

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

- Order of arguments matter

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

- Several expressions may represent same function

$$\begin{aligned} f(x,y) &= x \Rightarrow y \\ &= \neg x \vee y \\ &= (\neg x \vee y) \wedge (\neg x \vee x) \\ &= \ldots \end{aligned}$$

Number of Boolean Functions

Number of Boolean functions $f : \mathbf{B}^n \to \mathbf{B}$

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. Evaluating truth-value of an assignment easy
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!

Truth tables

- Compact table size $O(2^n)$
- 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*	*f(x,y,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

CNF-representation

- There exists a CNF of every expression
- Given a truth table representation of a Boolean formula, can we easily define a 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

© Rune Møller Jensen

CNF from off-tuples

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

x	y	z	e	$e \equiv$
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
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

$e \equiv$

$(\neg\neg x \lor \neg\neg y \lor \neg\neg z) \ \land$

$(\neg\neg x \lor \neg y \lor \neg\neg z) \ \land$

$(\neg x \lor \neg\neg y \lor \neg\neg z) \ \land$

CNF from off-tuples

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

x	y	z	e	$e \equiv$
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	

© Rune Møller Jensen

cCNF

- The special CNF-representations produced from *off*-tuples are canonical and called cCNF
- Are cCNF minimum **size** CNF representations?
  - Hint: look at representation of False
- Easy accessibility?

Binary Decision Diagrams

If-then-else operator

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

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

- We have

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

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

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

© Rune Møller Jensen

If-then-else operator

- All operators in propositional logic can be expressed using only $\rightarrow$ operators with
  - $\rightarrow$ expressions, 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:

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

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

Shannon Expansion

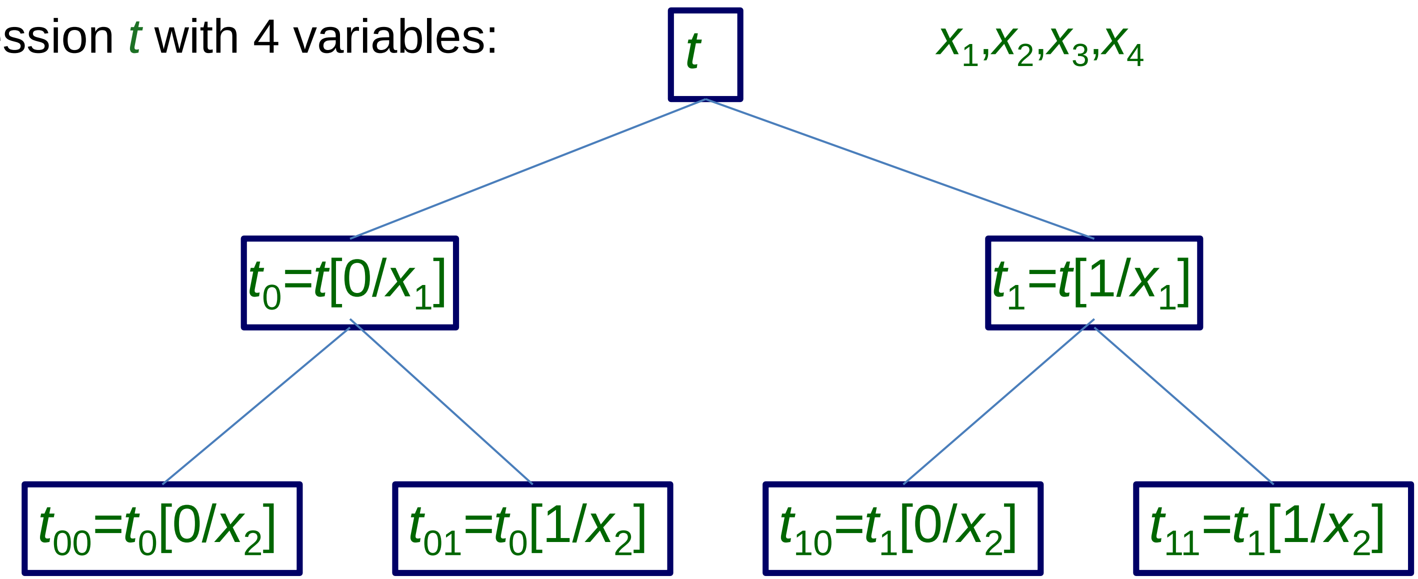
Expression t with 4 variables: $\boxed{t}$ x_1, x_2, x_3, x_4

$\boxed{t_0 = t[0/x_1]}$ $\boxed{t_1 = t[1/x_1]}$

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

Shannon Expansion

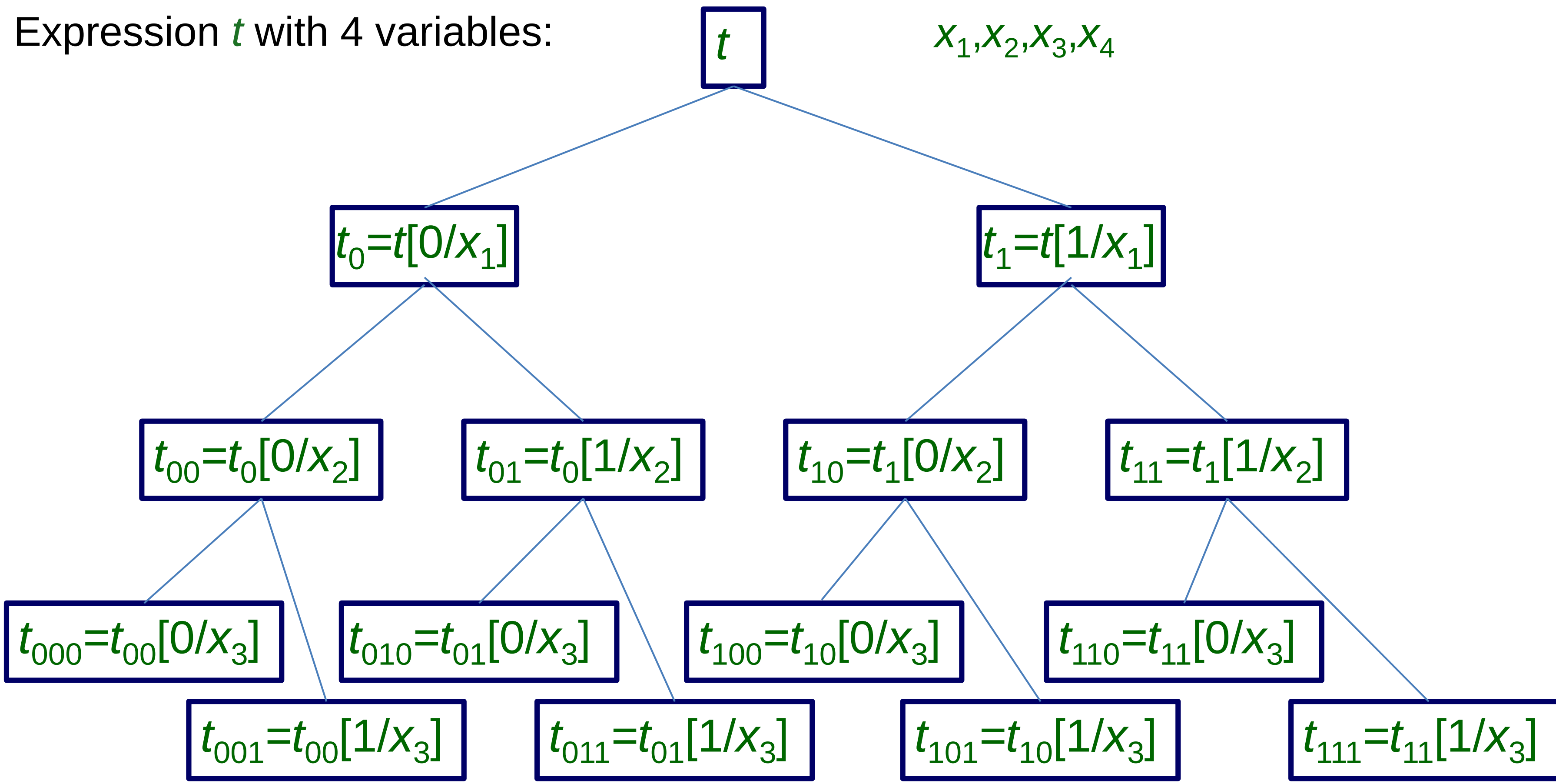
Expression *t* with 4 variables:



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

Shannon Expansion

Expression t with 4 variables:



$t \equiv x_1 \to t_1, t_0$

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

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

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

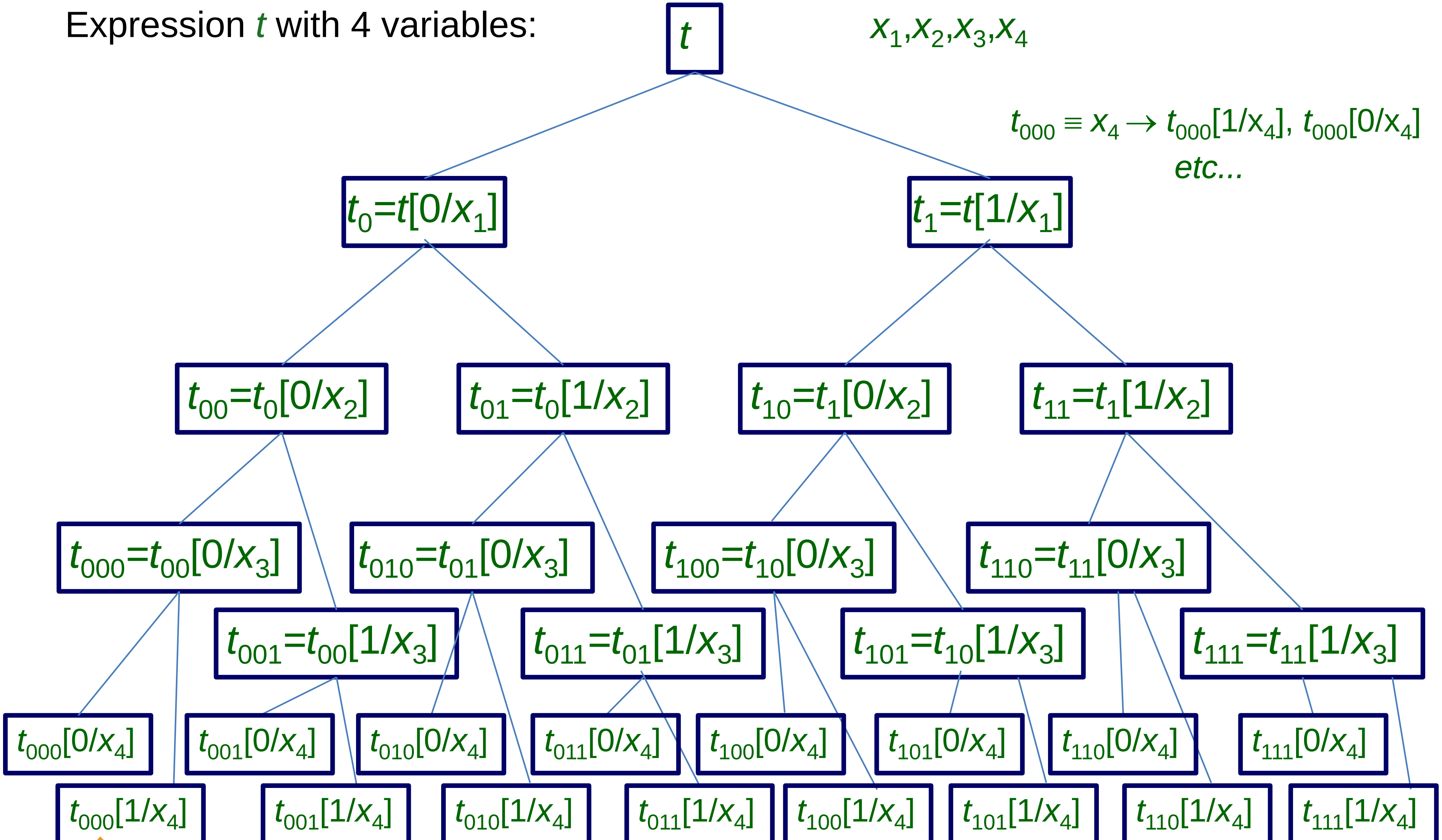
$t_{01} \equiv x_3 \to t_{011}, t_{010}$

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

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

Shannon Expansion

Expression t with 4 variables:

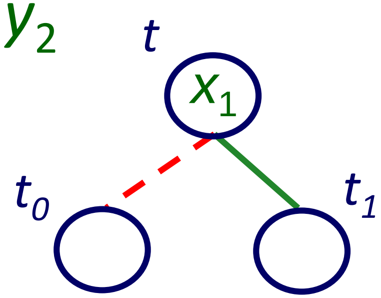


© Rune Møller Jensen

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

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



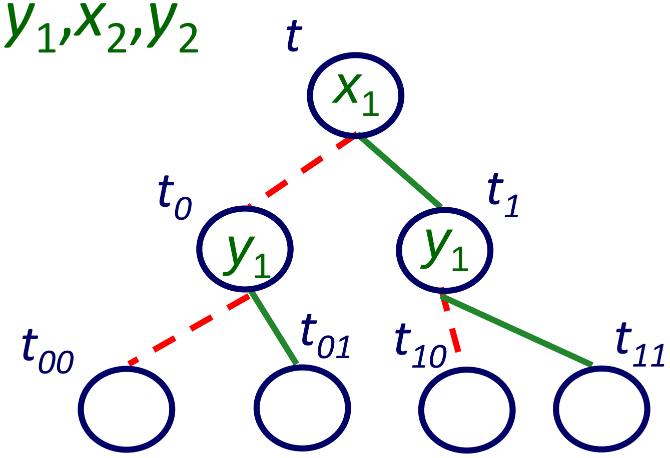
Example

- 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

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

$$t_0 \equiv y_1 \rightarrow t_{01}, t_{00}$$

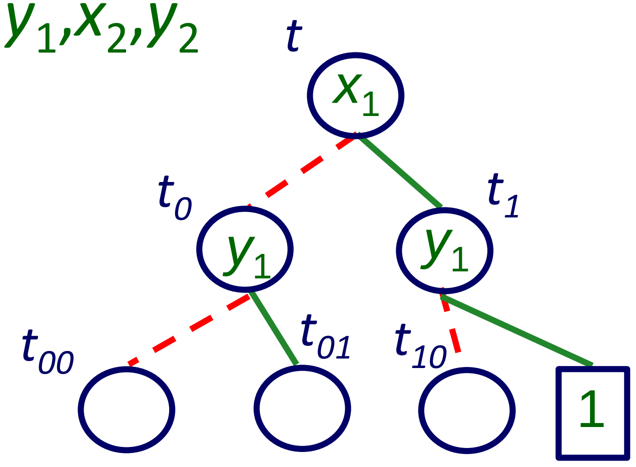
$$t_1 \equiv y_1 \rightarrow t_{11}, t_{10}$$



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

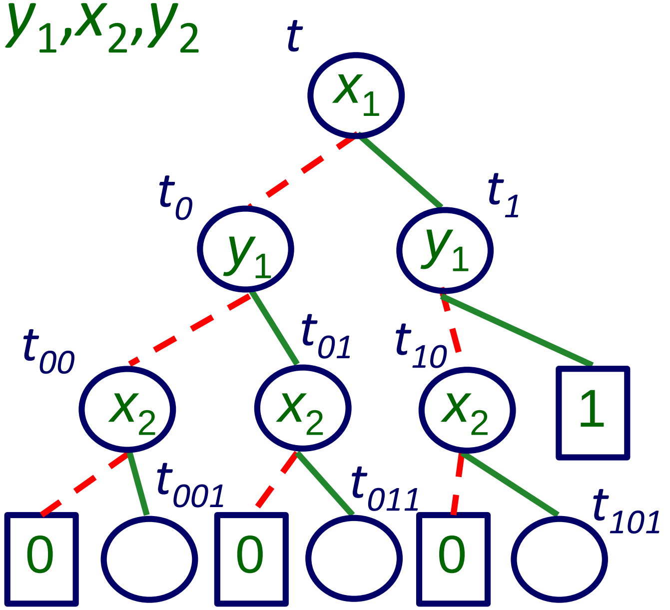
$$t \equiv x_1 \rightarrow t_1, t_0$$
$$t_0 \equiv y_1 \rightarrow t_{01}, t_{00}$$
$$t_1 \equiv y_1 \rightarrow 1, t_{10}$$



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

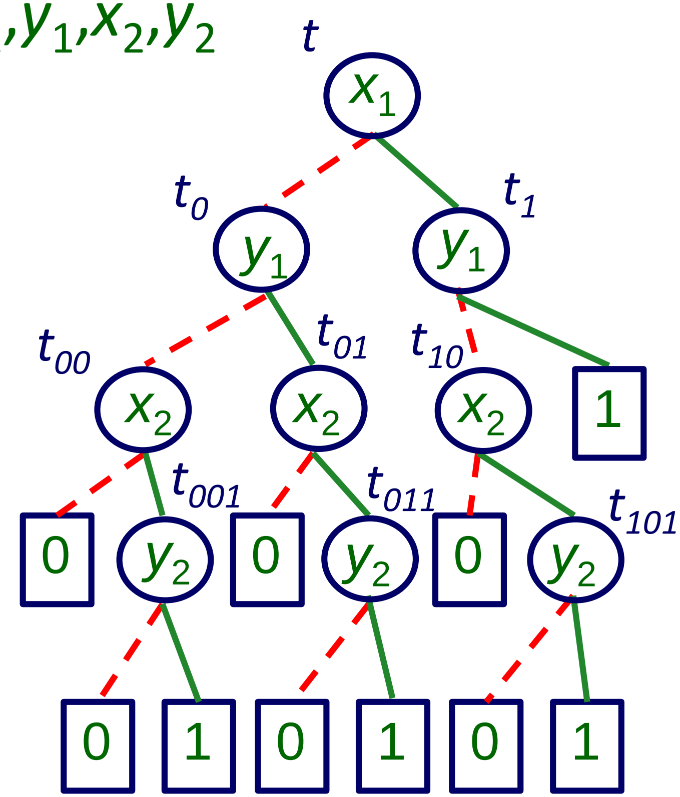
$$
\begin{aligned}
t &\equiv x_1 \rightarrow t_1, t_0 \\
t_0 &\equiv y_1 \rightarrow t_{01}, t_{00} \\
t_1 &\equiv y_1 \rightarrow 1, t_{10} \\
t_{01} &\equiv x_2 \rightarrow t_{011}, 0 \\
t_{00} &\equiv x_2 \rightarrow t_{001}, 0 \\
t_{10} &\equiv x_2 \rightarrow t_{101}, 0
\end{aligned}
$$



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

$$t \equiv x_1 \rightarrow t_1, t_0$$
$$t_0 \equiv y_1 \rightarrow t_{01}, t_{00}$$
$$t_1 \equiv y_1 \rightarrow 1, t_{10}$$
$$t_{01} \equiv x_2 \rightarrow t_{011}, 0$$
$$t_{00} \equiv x_2 \rightarrow t_{001}, 0$$
$$t_{10} \equiv x_2 \rightarrow t_{101}, 0$$
$$t_{011} \equiv y_2 \rightarrow 1,0$$
$$t_{001} \equiv y_2 \rightarrow 1,0$$
$$t_{101} \equiv y_2 \rightarrow 1,0$$

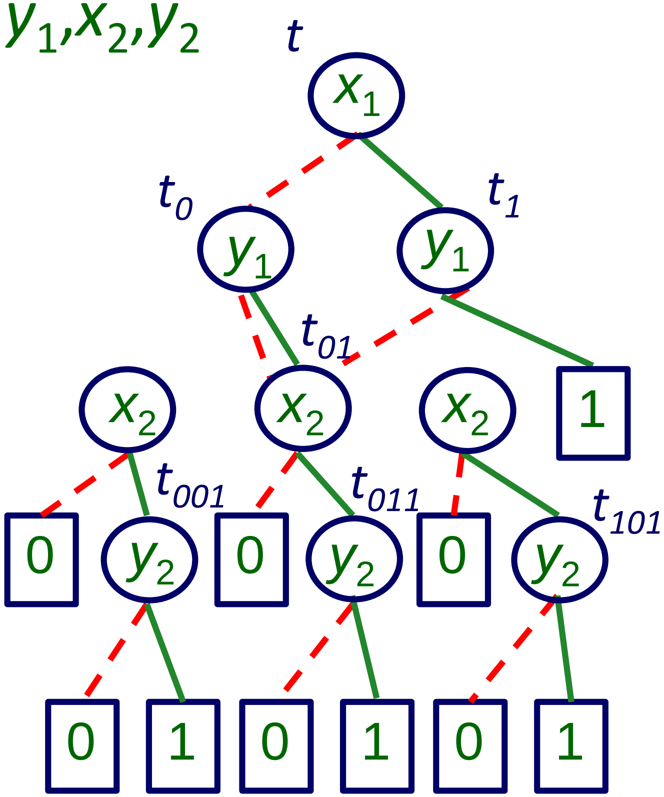


$$t = x_1 \rightarrow (y_1 \rightarrow 1, (x_2 \rightarrow (y_2 \rightarrow 1,0), 0)), (y_1 \rightarrow (x_2 \rightarrow (y_2 \rightarrow 1,0), 0), (x_2 \rightarrow (y_2 \rightarrow 1,0), 0))$$

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

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

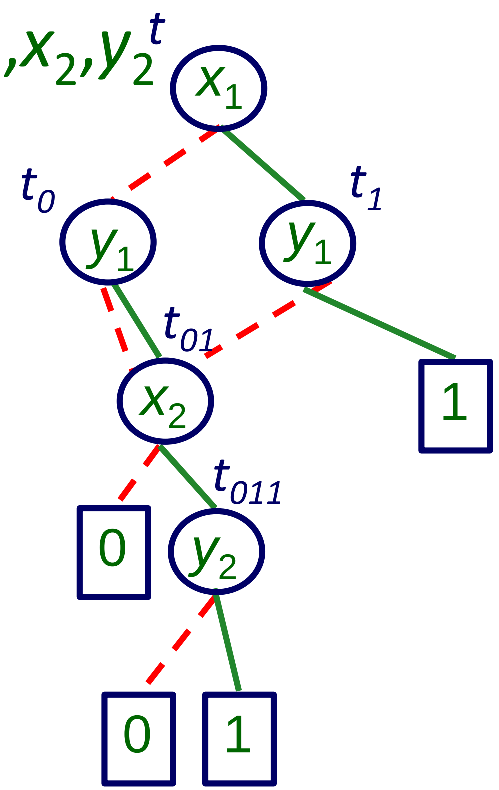


$$
\begin{aligned}
t = {} & x_1 \rightarrow (y_1 \rightarrow 1, (x_2 \rightarrow (y_2 \rightarrow 1,0), 0)), \\
& (y_1 \rightarrow (x_2 \rightarrow (y_2 \rightarrow 1,0), 0), (x_2 \rightarrow (y_2 \rightarrow 1,0), 0))
\end{aligned}
$$

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

$$t \equiv x_1 \rightarrow t_1, t_0$$
$$t_0 \equiv y_1 \rightarrow t_{01}, t_{00}$$
$$t_1 \equiv y_1 \rightarrow 1, t_{10}$$
$$t_{01} \equiv x_2 \rightarrow t_{011}, 0$$
$$t_{011} \equiv y_2 \rightarrow 1,0$$

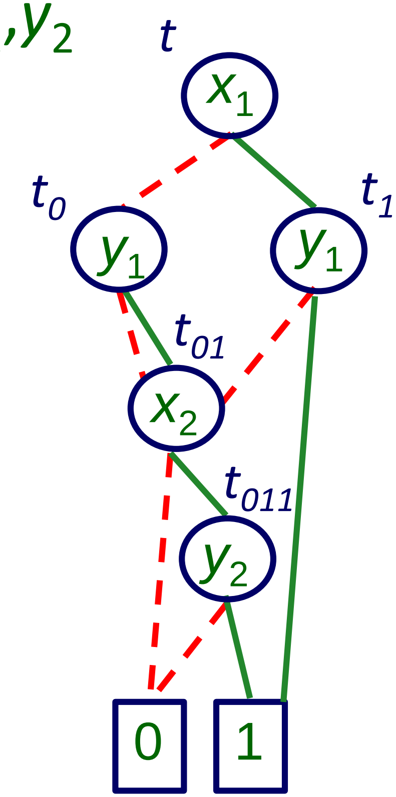


$$t = x_1 \rightarrow (y_1 \rightarrow 1,(x_2 \rightarrow (y_2 \rightarrow 1,0), 0)), (y_1 \rightarrow (x_2 \rightarrow (y_2 \rightarrow 1,0), 0), (x_2 \rightarrow (y_2 \rightarrow 1,0), 0))$$

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

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

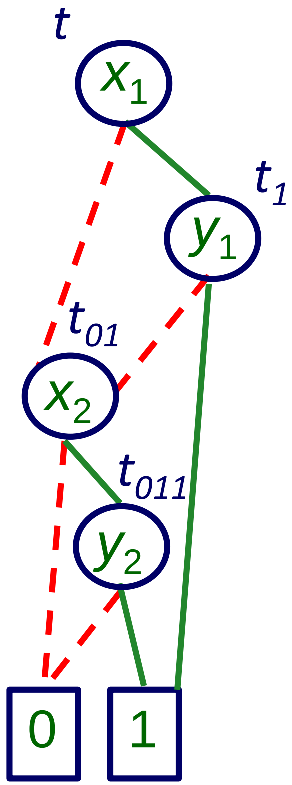


Result: an Ordered Binary Decision Diagram (OBDD)

Reduction II: remove redundant tests

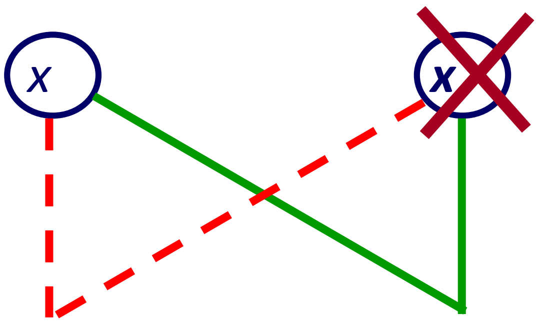
- Example: $t = (x_1 \wedge y_1) \vee (x_2 \wedge y_2)$
- Shannon expansion of t in order x_1, y_1, x_2, y_2

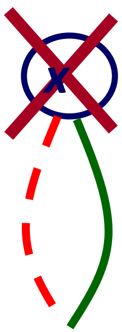
$$t = x_1 \rightarrow t_1, t_{01}$$
$$t_1 = y_1 \rightarrow 1, t_{01}$$
$$t_{01} = x_2 \rightarrow t_{011}, 0$$
$$t_{011} = y_2 \rightarrow 1,0$$



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

Reductions

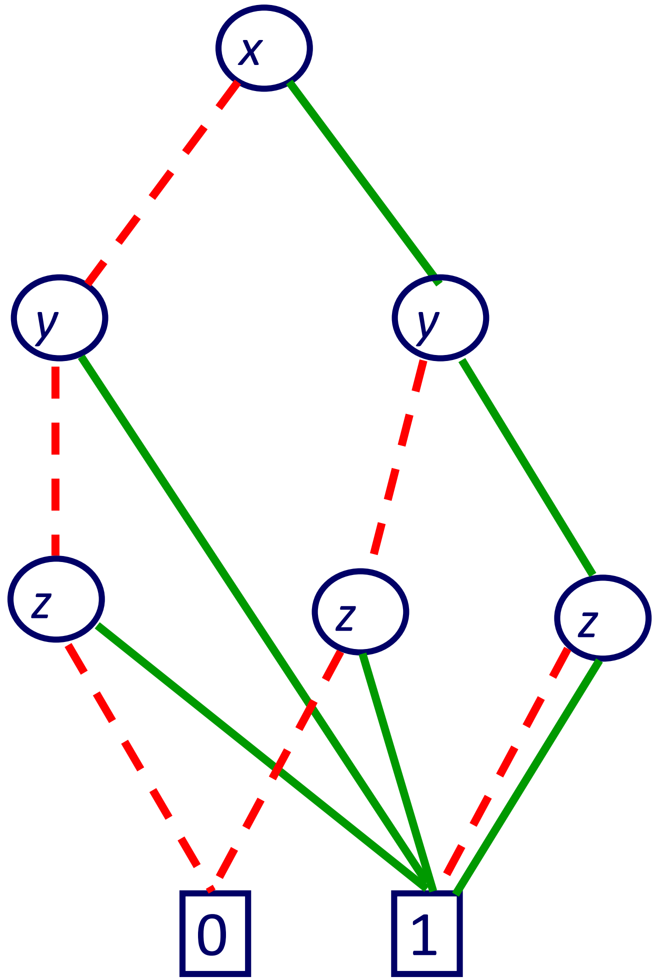




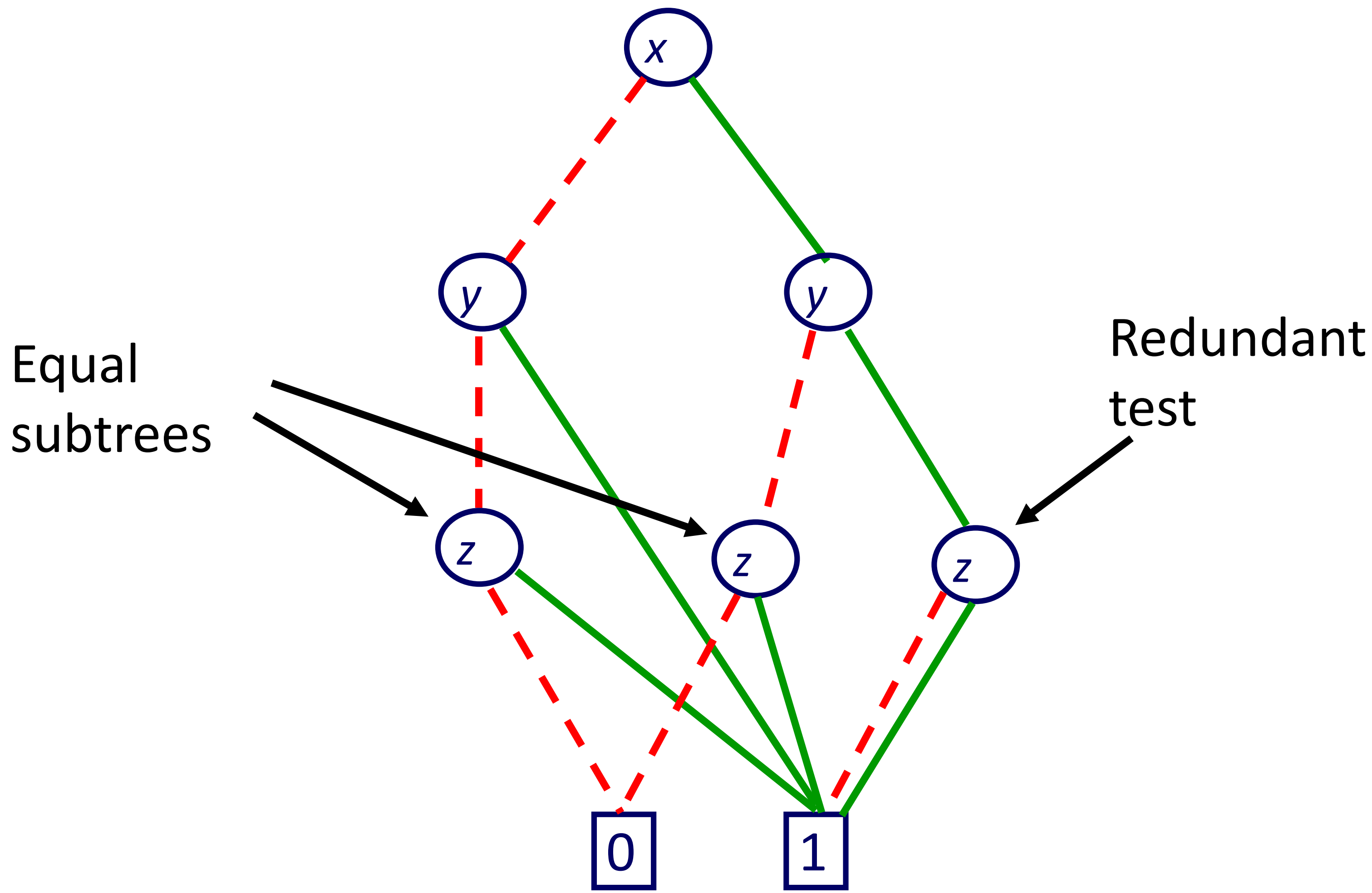
Uniqueness requirement

Non-redundant tests requirement

Another reduction example

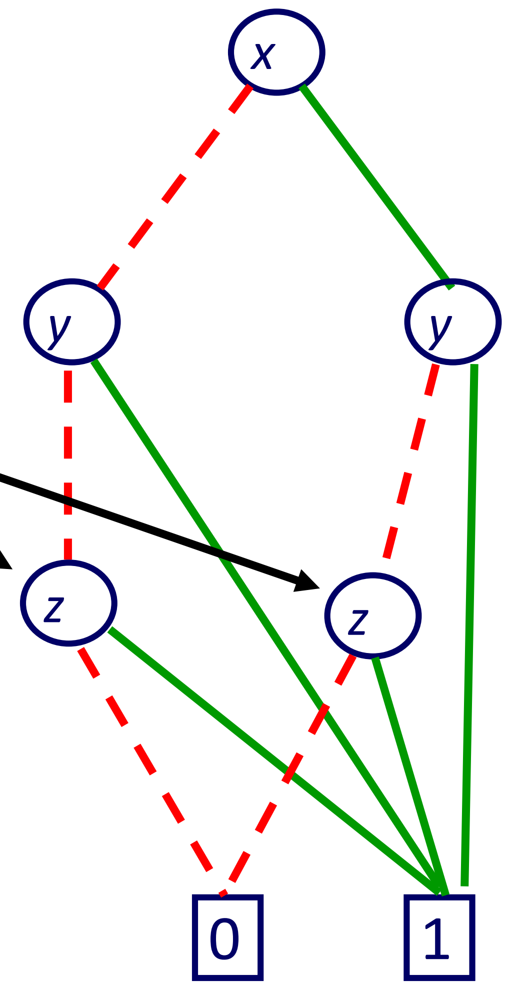


Another reduction example

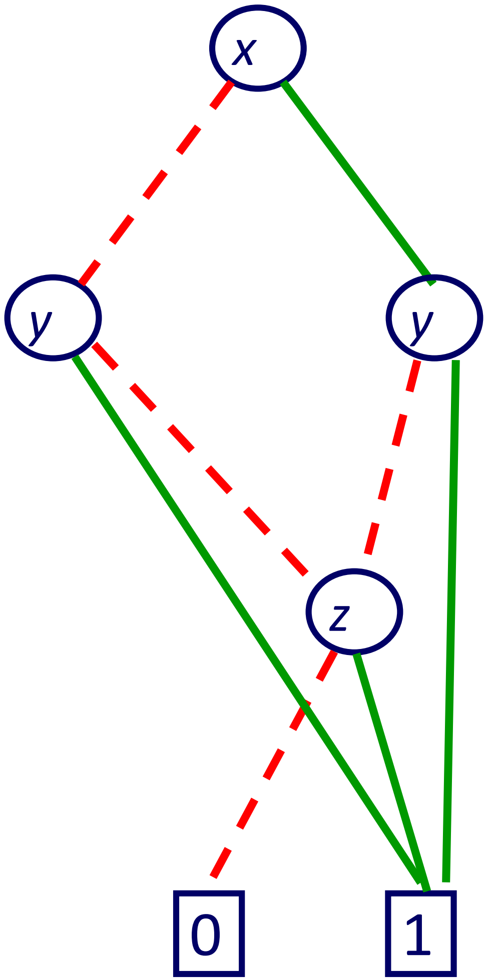


Another reduction example

Equal
subtrees

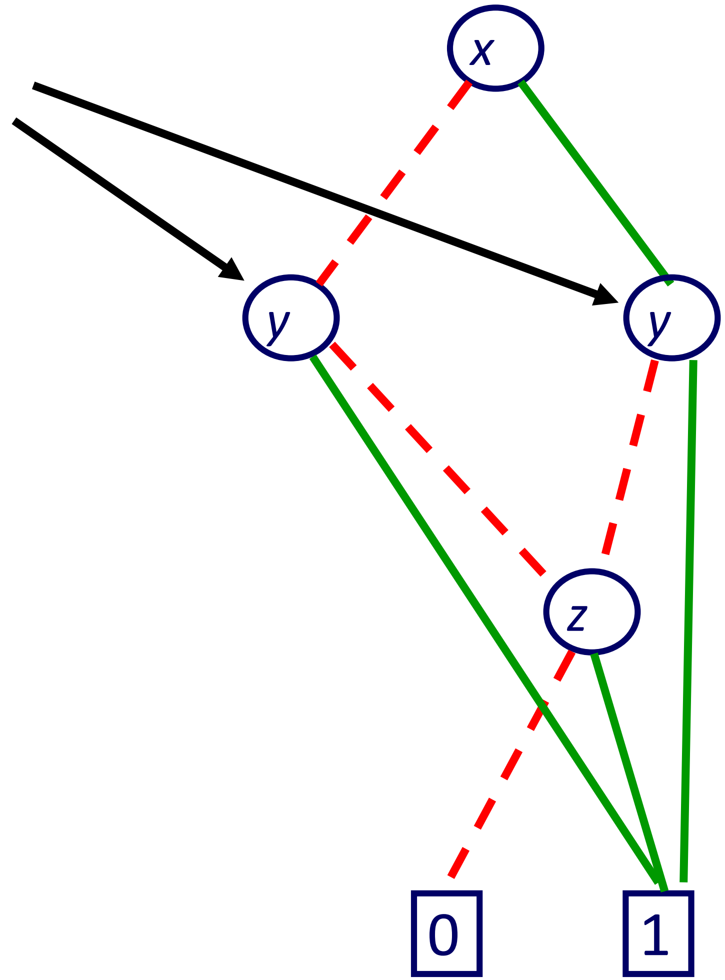


Another reduction example

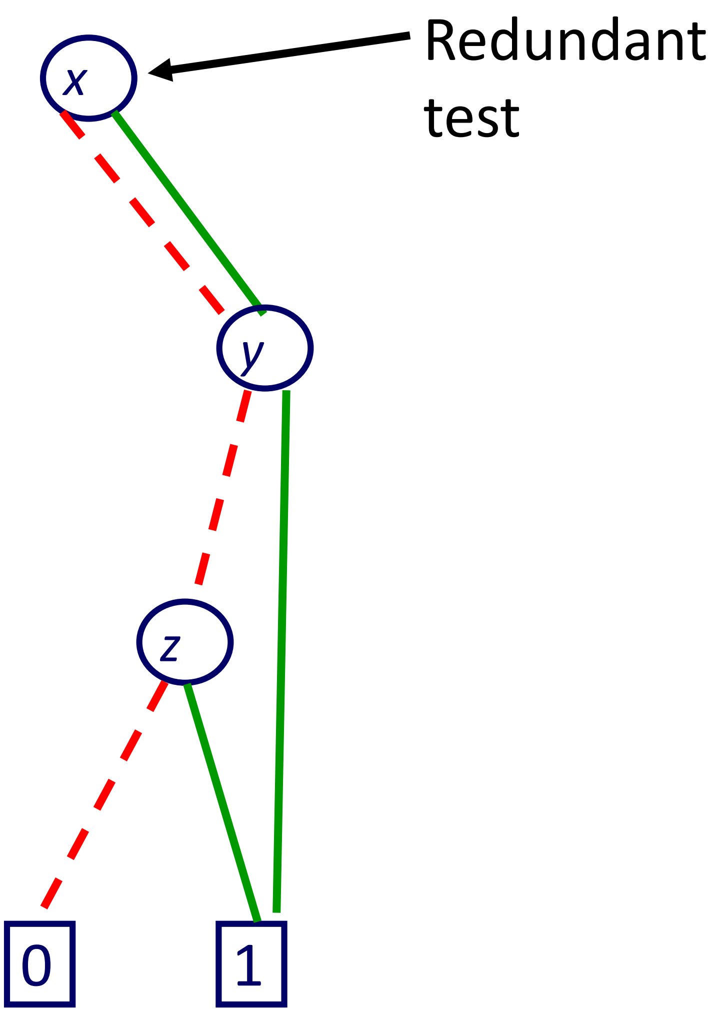


Another reduction example

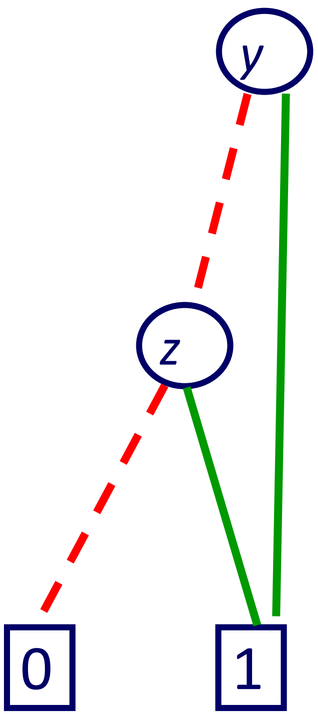
Equal subtrees



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

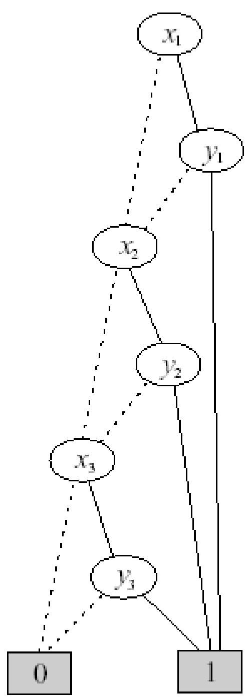
Size of ROBDDs

- 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

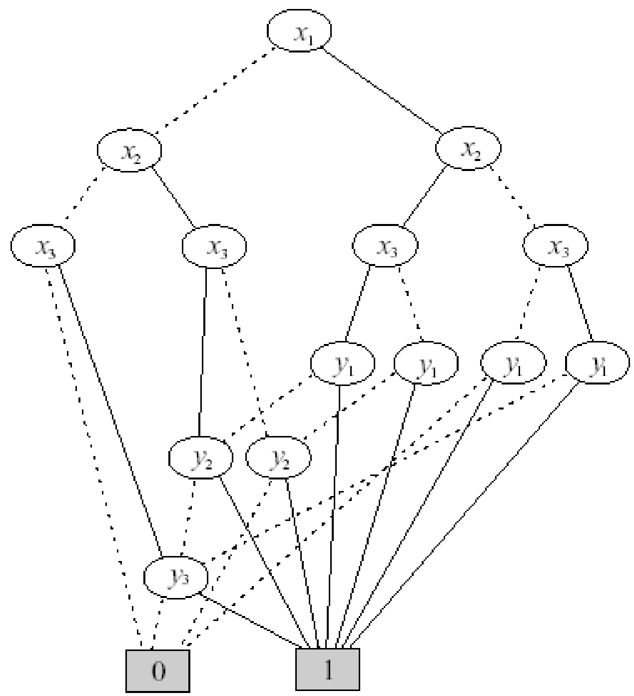
Size of ROBDDs

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

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



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

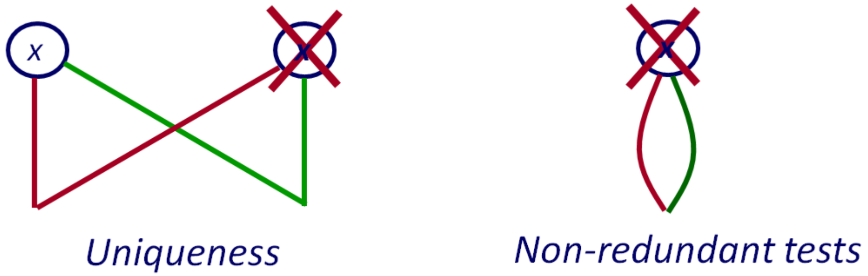


$x_1 < x_2 < \ldots < x_n < y_1 < x_2 < \ldots < y_n$

BDD construction

What we just saw:

1. Make a Decision Tree of the Boolean expression
2. Keep reducing it until no further reductions are possible



Next week:

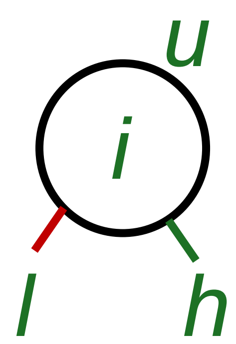
- Reduce the decision tree to a BDD while building it

© Rune Møller Jensen

Unique Table Representation

Node Attributes

u	unique node identifier $\{\mathbf{0},\mathbf{1},2,3,\ldots\}$
i	variable index $\{1,2,\ldots,n,\mathbf{n+1}\}$
l	node identifier of low
h	node identifier of high



Represent Unique Table by two tables *T* and *H*

$T : u \rightarrow (i,l,h)$

$H: (i,l,h) \rightarrow u$

H is the inverse of T:

$T(u) = (i,l,h) \;\Leftrightarrow\; H(i,l,h) = u$

Primitive Operations on T and H

$T : u \mapsto (i, l, h)$

$init(T)$	initialize T to contain only 0 and 1
$u \leftarrow add(T, i, l, h)$	allocate a new node u with attributes (i, l, h)
$var(u), low(u), high(u)$	lookup the attributes of u in T

$H : (i, l, h) \mapsto u$

$init(H)$	initialize H to be empty
$b \leftarrow member(H, i, l, h)$	check if (i, l, h) is in H
$u \leftarrow lookup(H, i, l, h)$	find $H(i, l, h)$
$insert(H, i, l, h, u)$	make (i, l, h) map to u in H