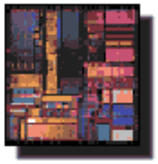
Digital Systems Design Automation

Unit 2: Advanced Boolean Algebra

Lecture 2.3: Boolean Function Representations



Anand Raghunathan
raghunathan@purdue.edu

Outline

Representations of Boolean Functions

- Truth Table
- Hypercube
- Boolean Formula
  - Sum of Products (SOP) / Disjunctive Normal Form (DNF) / List of Cubes
  - Product of Sums / Conjunctive Normal Form (CNF) / List of Conjuncts
- Network (graph) of Boolean primitives
- Binary Decision Tree, Binary Decision Diagram (BDD)

Representations of Boolean Functions

- Important questions to ask of any representation
  - Scalable (can it represent large functions)?
  - Canonical?
    - If two functions are equivalent, then are their representations isomorphic (structurally identical)?
  - Efficient to manipulate?

Truth Table

- Truth table of a function f: $B^n \rightarrow B$ is a tabulation of its values at each of the 2^n vertices of B^n
- The truth table representation is
  - \+ Canonical
  - \- Not scalable (very large for large *n*)

$n \cdot 2^n$

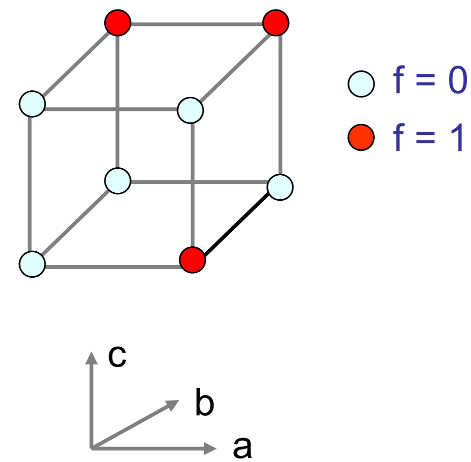
Example: f = b c + a b' c'

a b c	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Hypercube

- A function f: $B^n \rightarrow B$ can be represented by a coloring of the vertices of an n-dimensional hypercube
- The hypercube representation is
  - \+ Canonical
  - \- Not scalable (very large for large *n*)

Example: f = b c + a b' c'



Representations of Boolean Functions

- Truth Table
- Hypercube
- Boolean Formula
  - Sum of Products (SOP) / Disjunctive Normal Form (DNF) / List of Cubes
  - Product of Sums / Conjunctive Normal Form (CNF) / List of Conjuncts
- Network (graph) of Boolean primitives
- Binary Decision Tree, Binary Decision Diagram (BDD)

Boolean Formula

- Boolean functions can be represented by formulae defined as well-formed sequences of
  - Literals: x_1, x_1'
  - Boolean operators: + (OR), . (AND), ' (NOT)
    - NOT: $f' = h$ such that $h^1 = f^0$
    - AND: (f AND g) = h such that $h^1 = \{x \mid f(x) = 1$ and $g(x) = 1\}$
    - OR: (f OR g) = h such that $h^1 = \{x \mid f(x) = 1$ or $g(x) = 1\}$
  - Parentheses: ( )

Examples:

$f = x_1.x_2' + x_1'.x_2$

$\quad = (x_1 + x_2).(x_1' + x_2')$

$h = x_1 + x_2.x_3$

$\quad = (x_1'.(x_2' + x_3'))'$

Notation: Often the "." symbol for AND is omitted

e.g., $x_1x_2 + x_3$

Questions

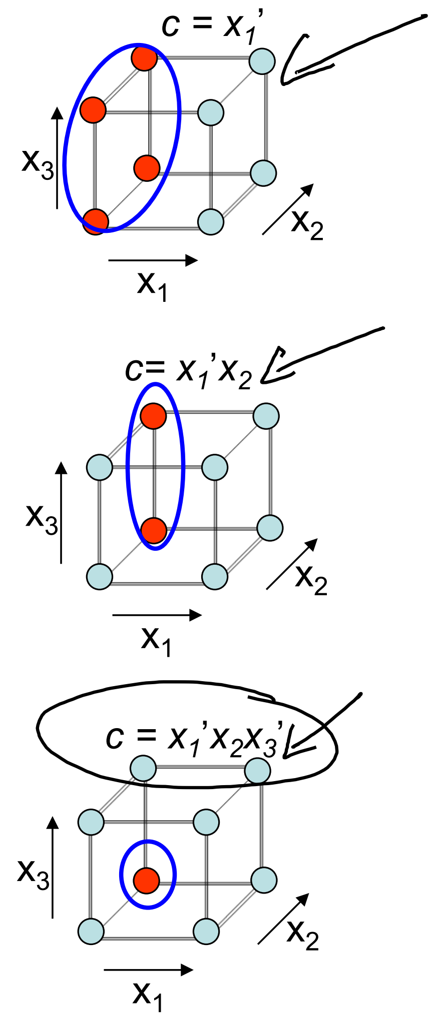
- How many Boolean formulae can be constructed with n variables?

$2^{2^n} \longrightarrow \infty$

- How does this compare with the number of unique Boolean functions in n variables?
- Are Boolean formulae canonical? Are they scalable?

Cubes

- A cube (a.k.a. product term) is the conjunction (AND) of a set of literals
  - Also, a collection of vertices that forms a hypercube of lower dimension
- If $C \subseteq B^n$, and C has k literals, then $|C|$ covers 2^{n-k} vertices
- In an n-dimensional Boolean space B^n, a cube with n literals is called a minterm
- If a cube $C \subseteq f^1$ (f is a Boolean function), then C is an implicant of f



Sum of Products

Sum of Products (SOP)

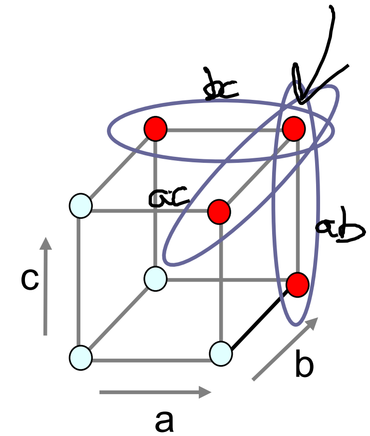
- A disjunction of product terms

$$f = ab + ac + bc$$

- Can also be thought of as a set of cubes

$$F = \{ab, ac, bc\}$$

- Any Boolean function can be represented by a sum of products



- A set of cubes that correctly represents *f* is called a ***cover*** of *f*
- A function may have several different SOP representations or covers
- Example:

  $F_1=\{ab, ac, bc\}$ and

  $F_2=\{abc, a'bc, ab'c, abc'\}$

  are possible covers of the Boolean function

  $$f = ab + ac + bc$$

Properties of a cover

- Each on-set vertex should be included in least one cube.
- No cube should include any off-set vertex.

Minterm Canonical Form

- A Sum of Products representation for a function where each product is a minterm
  - A minterm is a product term that has n literals representing all variables

Example:

a b c	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Minterm canonical form:

$a'bc + ab'c' + abc$

Question: Is the minterm canonical form a scalable representation?

Product of Sums

- Product (conjunction) of terms, each of which is a sum (disjunction) of literals
  - E.g., f = (a + b + c)(a + b + c') (a' + b + c')(a' + b' + c)
- One-to-one transformation from SOP representation for f to POS representation for f' (complement of f)
  - Follows from DeMorgan's law
- From truth table, use off-set to construct POS representation

Example:

a b c	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

POS representation:

$f = (a'b'c' + a'b'c + a'bc' + ab'c + abc')'$

$= (a+b+c)(a+b+c')(a+b'+c)(a'+b+c')(a'+b'+c)$

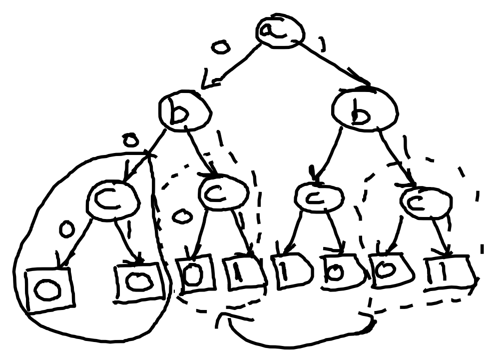
Binary Decision Tree

- Represent the function as a decision tree
- At each node, pick an input variable and branch based on it's value
- Leaves of the tree contain constants (0,1)

Example:

a b c	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Binary Decision Tree:



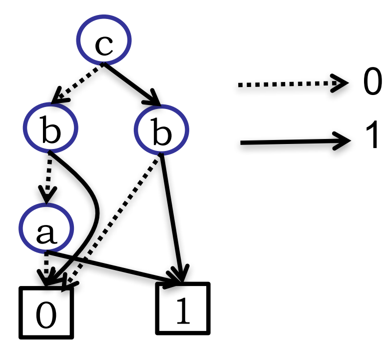
Binary Decision Diagram (BDD)

- Binary Decision Tree has large number of nodes
- Key idea: Share sub-trees and eliminate redundancy to reduce size
- More about BDDs later in the class

Example:

a b c	f
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Binary Decision Diagram:



Characteristic Functions

- Any sub-set of a Boolean space B^n can be represented as a characteristic function

Suppose $A \subseteq B^n$

$\chi_A(x) = 1$ if $x \in A$

Example:

$B^2 = \{00, 01, 10, 11\}$

$A = \{01, 10\}$

$\chi_A(01) = 1$

$\chi_A(11) = 0$

Characteristic Functions

- Given a Boolean function f : $B^n \rightarrow B^m$
- The characteristic function of f is a function $\chi_f : B^{n+m} \rightarrow B$

$\chi_f(x,y) = 1$ iff $f(x) = y$

Example:

$y = f(x_1,x_2) = x_1 + x_2$

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

$\chi_f(x_1,x_2,y) =$

x_1	x_2	y	χ_f
0	0	0	1
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