Overview

- Representations of boolean functions
- Canonical and non-canonical forms
- Decision trees and reduction rules
- Building already reduced diagrams
- Zero-Suppressed BDDs
- BDDs with complement edges
- Dynamic variable reordering

Examples

- Truth table, SoP and PoS
- Decision tree and truth table

Known Representations

- Truth table
- SoP (DNF) and PoS (CNF)
- Decision tree
- Reduced decision tree
- Zero-Suppressed BDD
- BDD with complement edges
Reduction of Decision Tree

**Rule 1: Merging Rule:**
Nodes must be unique

**Rule 2: Elimination Rule:**
Redundant tests should not be present

Example of Decision Tree Reduction

---

**Shannon Expansion**

- A boolean function can be expanded with respect to any variable
  \[ F(x, y, z) = x' \land F_x' \lor x \land F_x \]
  where \( F_x' \) and \( F_x \) are positive (negative) cofactors
  \[ F_x' = F(0, y, z), \quad F_x = F(1, y, z) \]

---

**Building Already Reduced Tree**

Build a reduced tree for \( F \) by calling \( \text{Build}(F, 1) \), if variable ordering \( x_1 < x_2 < x_3 < \ldots < x_n \) is given

function \( \text{Build}(F, i) \)
  
  if ( \( i > n \) )
    if ( \( F == 0 \) ) return Node0;
    else return Node1;
  else
    \( v_0 = \text{Build}(F_x', i+1) \);
    \( v_1 = \text{Build}(F_x, i+1) \);
    return \( \text{CreateNode}(i, v_0, v_1) \);

---

**Creating a Unique Node**

assuming there is a node table with functions CheckExists() and Insert()

function \( \text{CreateNode}(\text{Var}, F_x', F_x) \)
  
  if ( \( F_x' == F_x \) )
    return \( F_x' \);
  else if ( \( \text{CheckExists}(\text{Var}, F_x', F_x) \) )
    return the existing node;
  else \( \text{Insert}(\text{Var}, F_x', F_x) \);
    return the new node;

---

**Zero-Suppressed BDDs**

**Rule 2: Elimination Rule:**
Redundant tests should not be present

Rule 2: Elimination Rule:
Remove nodes with 1-edge pointing to 0-terminal
Complement Edges

- BDD edges can carry attributes
- Using complementation as an attribute for edges saves BDD nodes and makes NOT a constant time operation
- To maintain canonicity, the use of attribute edges should be constrained

Equivalent Pairs of Functions