Binary Decision Diagrams (BDDs)

An Efficient and Canonical Representation of Boolean Functions

Georg Hofferek
IAIK – Graz University of Technology
georg.hofferek@iaik.tugraz.at
Motivation

- Well-suited for Symbolic Algorithms
  - Verification
  - Synthesis

- Efficient Representation
  - Small, for many practical cases
  - Efficient Manipulation
    - Boolean Operations
  - Canonical
Outline

- Intuitive Examples
- Formal Definition
  - Structure
  - Semantics
- Construction
  - Via Cofactors
- Properties
  - Advantages, Disadvantages
- Software Packages
  - Common Concepts
Learning Targets

- Explain BDDs (and components) and their properties, advantages, and disadvantages
  - Based on example

- Determine the function of a given BDD

- Construct a BDD for a given function
  - By computing cofactors
  - Using a given variable order

- Convert a BDD into a multiplexer circuit
Formal Definition of a BDD

- **Directed Acyclic Graph**
  - Representing \( f: \mathbb{B}^n \mapsto \mathbb{B}^m \)

- \((V \cup \Phi \cup \{1\}, E)\)
  - Internal Nodes \( v \in V \)
  - Function Nodes \( f_i \in \Phi \)
  - Terminal Node \( 1 \)

- Edges \( E \)
  - “Complement” attribute
Internal Node

- **Label** \( l(v) \in \{x_1, \ldots, x_n\} \)
  - Variables of \( f \)

- **Out-degree**: 2
  - Then-Edge \( T \)
  - Else-Edge \( E \)
    - Marked with (empty) circle
    - Can have complement attribute (full circle)

- **Ordered**
  - \( v_i \) is ancestor of \( v_j \)  \( \iff \)  \( l(v_i) < l(v_j) \)
Function Node

- Single Boolean Function $f_i$
- In-degree: 0
- Out-degree: 1
  - Edge can have complement attribute
Terminal Node

- **Constant Function** \textbf{TRUE}

- Out-degree: 0
Semantics of BDDs

Function of

- Terminal Node:
  - TRUE

- Edge:
  - Target node
  - Negation, if complement attribute present

- Internal Node $v$:
  - $(l(v) \land f_T) \lor (\neg l(v) \land f_E)$

- Function node:
  - Outgoing edge
Reduced Ordered BDDs (ROBDD)

- No redundant nodes
  - Edges have different functions

- Unique nodes
  - Different nodes $\iff$ Different functions

- Canonical
  - For given variable ordering

Henceforth: BDD = ROBDD (unless explicitly stated otherwise)
Cofactors

- Boolean Function $f$
- with respect to a variable $x$
  - Positive Cofactor $f_x : f$ with $x$ set to $\top$
  - Negative Cofactor $f_{\neg x} : f$ with $x$ set to $\bot$

Example:

- $f = (x \land y) \lor (\neg x \land z)$
  - $f_x = y$
  - $f_{\neg x} = z$
Constructing BDDs

\[ f_x \quad X \quad f_{\neg x} \]
Advantages of BDDs

- **Size-Efficiency**
  - Worst case: exponential
  - Efficient for many practical examples

- **Efficient Operations**
  - e.g. AND, OR: Polynomial time
  - Equivalence Check: Constant time
  - Satisfiability and Validity Check: Constant Time
Disadvantages of BDDs

- Variable order
  - Big impact
  - Hard to optimize

- Circuit implementation
  - CNF or DNF more natural
BDD Software Packages

- Construction
- Manipulation
- Conversion

- E.g.: CUDD – Colorado University Decision Diagram Package
  - http://vlsi.colorado.edu/~fabio/CUDD/
  - BSD-style license
BDD Manager

- $f : \mathbb{B}^n \rightarrow \mathbb{B}^m$
- $m$ function nodes
- **Reuse** and **Sharing** of common internal nodes
Unique Table

- **Hash-Table**
  - **Entries:**
    - Variable
    - Then-Pointer
    - Else-Pointer

- Look-Up before node creation
  - No duplicate nodes constructed
Computed Table

- **Cache**
  - Operator
  - List of Parameters
  - Result

- **Example:**
  - \((\text{AND}, \text{Node } x, \text{Node } y) \rightarrow \text{Node } z\)

- Prevents costly re-computations

- **Recursion**
  - Intermediate results!
Dynamic Reordering

- Change variable order on-the-fly
  - E.g. when size > threshold

- Based on swap of adjacent layers

- Many different heuristics
  - When?
  - How?

- Time-intensive
  - Often worth the effort, or even necessary
Summary

- Semantics
  - Conversion to Formulas

- Construction
  - From Formula
  - Via Cofactors

- Properties
  - Canonicity
  - No redundancy

- Software Packages
  - Unique Table
  - Computed Table
  - Dynamic Reordering