Part I

- Reminders -
  - What is Logic
  - Proofs by deduction
  - Proofs by enumeration
  - Decidability, Soundness and Completeness
  - Some notes on Propositional Logic
- Deciding Propositional Logic
  - SAT tools
  - BDDs

Next: Binary Decision Diagrams

- SAT looked for a satisfying solution to CNF
- We will now examine a graph-based data structure called Binary Decision Diagrams.
- It has several advantages and disadvantages comparing to SAT
- Developed by Bryant [1986]. Next few slides are from the source …

Alternate Approach

- Generate complete representation of function

  \[(A \land C) \lor (C \land B)\]

  \[(A \lor B) \land (C)\]

- Canonicity: functions are equal iff representations are identical
Decision Structures

Truth Table

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Decision Tree

- Vertex represents decision
- Follow green (dashed) line for value 0
- Follow red (solid) line for value 1
- Function value determined by leaf value.

Variable Ordering

- Assign arbitrary total ordering to variables
- e.g., $x_1 < x_2 < x_3$
- Variables must appear in ascending (consistent) order along all paths

Reduction Rule #1

- Merge equivalent leaves

Reduction Rule #2

- Merge isomorphic nodes
Reduction Rule #3

Eliminate Redundant Tests

Example OBDD

Canonical representation of Boolean functions
For a given variable ordering
- Two functions are equivalent iff graphs are isomorphic
  - Can be tested in linear time
- Desirable property: simplest form is canonical.

Satisfiability etc.

Constants
- 0: Unique unsatisfiable function
- 1: Unique tautology

Conclusion: given a BDD it takes constant time to check:
- Validity
- Contradiction
- Satisfiability

Is this a free lunch? …

Effect of Variable Ordering

\((a_1 \land b_1) \lor (a_2 \land b_2) \lor (a_3 \land b_3)\)

Good Ordering

Bad Ordering

Linear Growth

Exponential Growth
Selecting Good Variable Ordering

- Intractable Problem
  - Even when problem represented as OBDD
  - i.e., to find optimum improvement to current ordering

- Application-Based Heuristics
  - Exploit characteristics of application
  - e.g., Ordering for functions of combinational circuit
    - Traverse circuit graph depth-first from outputs to inputs

Building BDDs ‘from below’

- Starting from a binary decision tree and reducing it is too hard for formulas with many variables.
- Goal: construct the BDD ‘from below’.

Building BDDs ‘from below’

- For this we will need a function called APPLY:
  - Given the BDDs for $f_1$ and $f_2$, and a binary connective, $\circ \in \{\land, \lor, \to, \leftrightarrow, \ldots\}$ (any of the 16 binary connectives),
  - Construct the BDD for $f_1 \circ f_2$

Restriction and BDDs

- Def: The restriction of boolean function $f$ to $x=d$, denoted $f_{x=d}$ where $x \in \text{vars}(f)$, $d \in \{0, 1\}$, is equal to $f$ after assigning $x=d$.
- Given the BDD of $f$, deriving the BDD of $f_{x=d}$ is simple:
Shannon Expansion Theorem and BDDs

- Shannon’s Theorem: For a boolean function \( f \) and a variable \( x \in \text{vars}(f) \),
  \[
  f = (\neg x \land f|_{x=0}) \lor (x \land f|_{x=1}).
  \]

- BDD interpretation, if \( x \) is first in the variable order:

Now, APPLY (1 of 3)

Let \( v_1, v_2 \) denote the root nodes of the BDDs for \( f_1, f_2 \), respectively:

Case 1: If \( v_1 \) and \( v_2 \) are leaves, the BDD for \( f_1 \circ f_2 \) is a leaf node with value \( \text{val}(v_1) \circ \text{val}(v_2) \)

    \[
    \begin{array}{cc}
    0 & 1 \\
    1 & 0
    \end{array}
    \]

Now, APPLY (2 of 3)

Case 2: If \( \text{var}(v_1) \) and \( \text{var}(v_2) \) are the same variable, \( x \), apply Shannon expansion:

    \[
    f_1 \circ f_2 = (\neg x \land (f_1 \circ f_2)|_{x=0}) \lor (x \land (f_1 \circ f_2)|_{x=1})
    = (\neg x \land (f_1|_{x=0} \circ f_2|_{x=0})) \lor (x \land (f_1|_{x=1} \circ f_2|_{x=1}))
    \]

Now, APPLY (3a)

Case 3a: If \( \text{var}(v_1) = x_1, \text{var}(v_1) = x_2 \), and \( x_1 < x_2 \) in the variable order, then \( f_2 = f_2|_{x_1=0} = f_2|_{x_1=1} \)

So, applying Shannon’s expansion:

    \[
    f_1 \circ f_2 = (\neg x_1 \land (f_1 \circ f_2)|_{x_1=0}) \lor (x_1 \land (f_1 \circ f_2)|_{x_1=1})
    = (\neg x_1 \land (f_1|_{x_1=0} \circ f_2)) \lor (x_1 \land (f_1|_{x_1=1} \circ f_2))
    \]
Finally, APPLY (3b)

Case 3b: If \( \text{var}(v_1) \) is \( x_2 \) and \( \text{var}(v_1) \) is \( x_1 \), and \( x_1 < x_2 \) in the variable order, then \( f_1 = f_1|_{x_1=0} = f_1|_{x_1=1} \).

So, applying Shannon’s expansion:
\[
f_1 \circ f_2 = (\neg x \land (f_1 \circ f_2)|_{x=0}) \lor (x \land (f_1 \circ f_2)|_{x=1})
\]
\[
= (\neg x_1 \land (f_1 \circ f_2)|_{x_1=0}) \lor (x_1 \land (f_1|_{x_1=0} \circ f_2))
\]

BDDs from below: example.

\[
f_1 \lor f_2: (x_1 \leftrightarrow x_2) \lor \neg x_2
\]
Comparing SAT to BDDs

- BDD is a canonical data structure that represents the semantic of a function, i.e. all its solutions
  - Some applications (e.g. symbolic model checking) need canonicity to detect when two sets are equivalent.
  - Can require exponential space & time (highly sensitive to variable ordering)

- SAT searches through CNF for a single solution
  - CNF is not canonical.
  - Poly-space algorithms exists. Time can be exponential.