Model Checking for Propositions

- Problem: Would like an efficient way to answer for a given proposition $P$:
  - Is $P$ a tautology?
  - Is $P$ satisfiable?
    - Note: And general algorithm to answer the first can be used to answer the second and vice versa.
  - Does a given valuation satisfy $P$?

- Difficulty: Answering if $P$ is satisfiable is NP-complete
- Algorithms exist with good performance in general practice
- BDDs are one such
Binary Decision Trees

- **Binary decision tree** is a (rooted, directed) edge and vertex labeled tree with two types of vertices – **internal nodes**, and **leaves** – such that:
  - Internal nodes have **exactly two** out edges
  - Leaves have **no** out edges
  - Internal nodes are labeled by **atomic propositions** (variables)
  - Leaves are labeled by **true** or **false**.
  - Left edges labeled **false** and right edges labeled **true**.
  - For each path (branch) in the tree, each atomic proposition may label at most one vertex of that path.
Binary Decision Trees

- Binary decision trees can record the set of all models (and non-models) of a proposition
  - Path records a valuation: out edge label gives value for variable labeling an internal
  - Any variable not on path can have any value
  - Leaf label says whether a valuation assigning those values to those variables
    - Is a model (true, the tree accepts the valuation)
    - Or not a model (false, the tree)
  - Each valuation matches exactly one branch
  - More than one valuation may (will) match a given branch
Example:

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

A

false

true

B

false

true

C

false

true

false

true

false

true

false

true

false

true

true
Example: Different Variable Ordering - Different Tree

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

```
  C
 /   \
false  true

  B
 /   \
false  true

  A
 /   \
false  true

  A
 /   \
false  true

  A
 /   \
false  true

  A
 /   \
false  true
```
Example: Many Logically Equivalent Trees

\[(A \land B) \lor (\neg C)\]
Alternate Syntax for Propositional Logic

- Still have constants \( \{T, F\} \)
- Still have countable set \( AP \) of propositional variables a.k.a. atomic propositions
- Only one ternary connective: the conditional `if_then_else`
  - First argument only a variable
  - Second and third arguments propositions
Translating Original Propositions into if\_then\_else

- Start with proposition $P_0$ with variables $v_1, \ldots, v_n$
- $P[c/v]$ is the proposition resulting from replacing all occurrences of variable $v$ with constant $c$
- Let $\overline{P}$ be the result of evaluating every subexpression of $P$ containing no variables
- Let $P_1 = \text{if } v_1 \text{ then } P_0[T/v_1] \text{ else } P_0[F/v_1]$
- Let $P_i = \text{if } v_i \text{ then } P_{i-1}[T/v_i] \text{ else } P_{i-1}[F/v_i]$
- $P_n$ is logically equivalent to $P$, but only uses \text{if\_then\_else}.
  - Valuation satisfies $P$ if and only if it satisfies $P_n$
  - $P_n$ depends on the order of variables $v_1, \ldots, v_n$
  - $P_n$ directly corresponds to a binary decision tree
Example:

\[ P = (A \land B) \lor (\neg C), \text{ variables } \{A, B, C\} \]

\[ P_0 = (A \land B) \lor (\neg C) \]
Example:

\[ P = (A \land B) \lor (\neg C) \], variables \{A, B, C\} \\
\[ P_0 = (A \land B) \lor (\neg C) \]
\[ P_1 = \text{if } A \text{ then } (T \land B) \lor (\neg C) \text{ else } (F \land B) \lor (\neg C) \]
Example:

\[ P = (A \land B) \lor (\neg C), \text{ variables } \{A, B, C\} \]

\[ P_0 = (A \land B) \lor (\neg C) \]

\[ P_1 = \text{if } A \text{ then } (T \land B) \lor (\neg C) \text{ else } (F \land B) \lor (\neg C) \]

\[ P'_2 = \text{if } B \text{ then } \text{if } A \text{ then } (T \land T) \lor (\neg C) \text{ else } (F \land T) \lor (\neg C) \]

\[ \text{else } \text{if } A \text{ then } (T \land F) \lor (\neg C) \text{ else } (F \land F) \lor (\neg C) \]
Example:

\[ P = (A \land B) \lor (\neg C), \text{ variables } \{A, B, C\} \]

\[ P_0 = (A \land B) \lor (\neg C) \]
\[ P_1 = \text{if } A \text{ then } (T \land B) \lor (\neg C) \text{ else } (F \land B) \lor (\neg C) \]
\[ P'_2 = \text{if } B \text{ then } (\text{if } A \text{ then } (T \land T) \lor (\neg C) \text{ else } (F \land T) \lor (\neg C)) \]
\[ \quad \text{else } (\text{if } A \text{ then } (T \land F) \lor (\neg C) \text{ else } (F \land F) \lor (\neg C)) \]
\[ P_2 = \text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg C) \text{ else } F \lor (\neg C)) \]
\[ \quad \text{else } (\text{if } A \text{ then } F \lor (\neg C) \text{ else } F \lor (\neg C)) \]
Example:

\[ P = (A \land B) \lor (\neg C), \text{ variables } \{A, B, C\} \]

\[ P_0 = (A \land B) \lor (\neg C) \]

\[ P_1 = \text{if } A \text{ then } (T \land B) \lor (\neg C) \text{ else } (F \land B) \lor (\neg C) \]

\[ P'_2 = \text{if } B \text{ then } (\text{if } A \text{ then } (T \land T) \lor (\neg C) \text{ else } (F \land T) \lor (\neg C)) \]

\[ \quad \text{else } (\text{if } A \text{ then } (T \land F) \lor (\neg C) \text{ else } (F \land F) \lor (\neg C)) \]

\[ P_2 = \text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg C) \text{ else } F \lor (\neg C)) \]

\[ \quad \text{else } (\text{if } A \text{ then } F \lor (\neg C) \text{ else } F \lor (\neg C)) \]

\[ P'_3 = \text{if } C \text{ then } (\text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg T) \text{ else } F \lor (\neg T)) \]

\[ \quad \text{else } (\text{if } A \text{ then } F \lor (\neg T) \text{ else } F \lor (\neg T))) \]

\[ \quad \text{else } (\text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg F) \text{ else } F \lor (\neg F)) \]

\[ \quad \text{else } (\text{if } A \text{ then } F \lor (\neg F) \text{ else } F \lor (\neg F))) \]
Example:

\[ P = (A \land B) \lor (\neg C), \text{ variables } \{A, B, C\} \]

\[
P_0 = (A \land B) \lor (\neg C)
\]

\[
P_1 = \text{if } A \text{ then } (T \land B) \lor (\neg C) \text{ else } (F \land B) \lor (\neg C)
\]

\[
P'_2 = \text{if } B \text{ then } (\text{if } A \text{ then } (T \land T) \lor (\neg C) \text{ else } (F \land T) \lor (\neg C))
\]

\[\text{ else } (\text{if } A \text{ then } (T \land F) \lor (\neg C) \text{ else } (F \land F) \lor (\neg C))\]

\[
P_2 = \text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg C) \text{ else } F \lor (\neg C))
\]

\[\text{ else } (\text{if } A \text{ then } F \lor (\neg C) \text{ else } F \lor (\neg C))\]

\[
P'_3 = \text{if } C \text{ then } (\text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg T) \text{ else } F \lor (\neg T)))
\]

\[\text{ else } (\text{if } A \text{ then } F \lor (\neg T) \text{ else } F \lor (\neg T)))\]

\[\text{ else } (\text{if } B \text{ then } (\text{if } A \text{ then } T \lor (\neg F) \text{ else } F \lor (\neg F)))
\]

\[\text{ else } (\text{if } A \text{ then } F \lor (\neg F) \text{ else } F \lor (\neg F)))\]

\[
P_3 = \text{if } C \text{ then } (\text{if } B \text{ then } (\text{if } A \text{ then } T \text{ else } F)
\]

\[\text{ else } (\text{if } A \text{ then } F \text{ else } F))\]

\[\text{ else } (\text{if } B \text{ then } (\text{if } A \text{ then } T \text{ else } T)
\]

\[\text{ else } (\text{if } A \text{ then } T \text{ else } T))\]
Example, cont.

\[ P_3 = \text{if } C \text{ then (if } B \text{ then (if } A \text{ then } \text{T} \text{ else } \text{F}) \text{ else (if } A \text{ then } \text{F} \text{ else } \text{F})} \text{ else (if } B \text{ then (if } A \text{ then } \text{T} \text{ else } \text{T}) \text{ else (if } A \text{ then } \text{T} \text{ else } \text{T})} \]

\( P_3 \) corresponds to second binary decision tree given earlier

- Any proposition is strict \text{if\_then\_else\_} form corresponds directly to a binary decision tree that accepts exactly the valuations that satisfy (model) the proposition.
Binary Decision Diagram

- Binary decision trees may contain (much) redundancy
- Binary Decision Diagram (BDD): Replace trees by (rooted) directed acyclic graphs
- Require all other conditions still hold
- Generalization of binary decision trees
- Allows for sharing of common subtrees.
- Accepts / rejects valuations as with binary decision trees.
Problem: given proposition may correspond to many different BDDs

How to create a (compact) canonical BDD for a proposition such that two different propositions are logically equivalent if and only if they have the same (isomorphic) canonical BDD

Start: order propositional variables $v_i < v_j$.

Bryant showed you can obtain such a canonical BDD by requiring
- Variables should appear in order on each path for root to leaf
- No distinct duplicate (isomorphic) subtrees (including leaves)
Achieving Canonical Form

- Start with an Ordered BDD (all edges in correct order)
- Repeat following until none apply
  - **Remove duplicate leaves:** Eliminate all but one leaf with a given label and redirect all edges to the eliminated leaves to the remaining one
  - **Remove duplicate nonterminals:** If node $n$ and $m$ have the same variable label, their left edges point to the same node and their right edges point to the same node, remove one and redirect edges that pointed to it to the other
  - **Remove redundant tests:** If both out edges node $n$ point to node $m$, eliminate $n$ and redirect all edges coming in to $n$ to $m$
- Bryant gave procedure to do the above that terminates in linear time