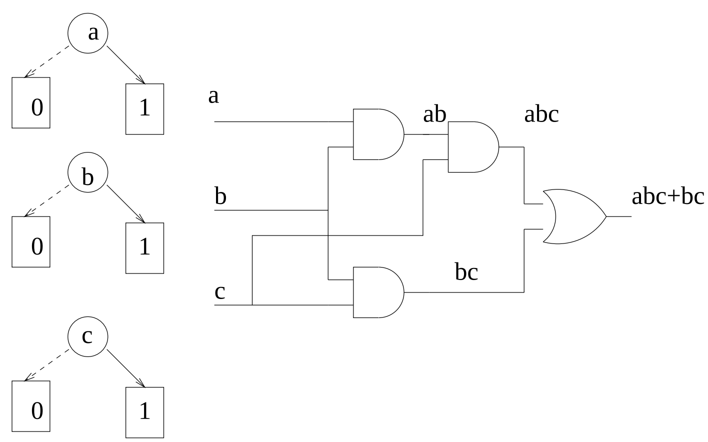
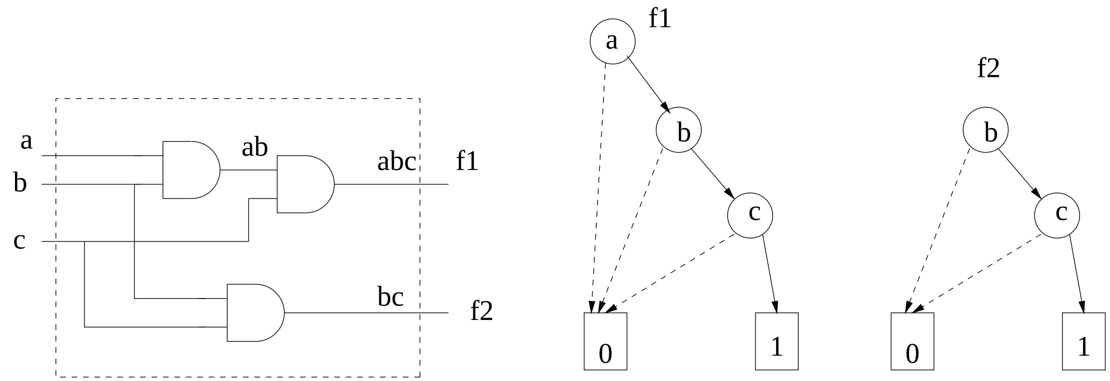
Build Reduced OBDDs Directly from a Circuit

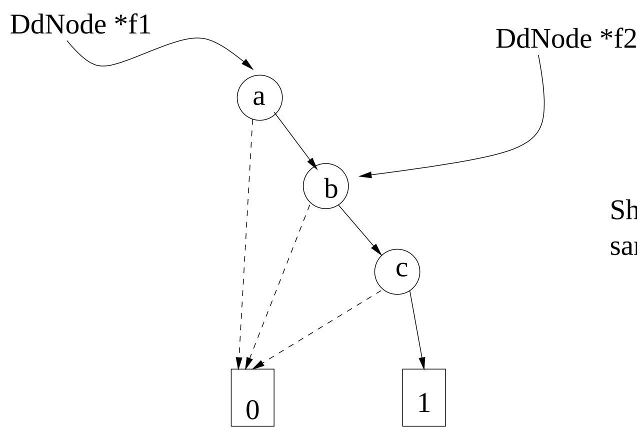
- $f = abc + bc$
- Build Trivial ROBDDs for a, b, c
- Build ROBDD for abc from ROBDDs for a, b and c
- Operate on GRAPHs of a, b and c and get ROBDD for abc!



BDDs as Multi-Rooted DAGs

- Ckt w/ 2 Outputs: $f1 = abc, f2 = bc$





The ROBDD Manager

- Objective: Create ROBDD package as a software library
- Create a "global" manager that:
  - Allows global access to any node in the ROBDD
  - Keeps statistical info of the number of BDD nodes
  - Keeps a pointers to terminal nodes **0** & **1**
  - Keeps a record of the number of BDD vars (via levels)
  - Maintains the uniform *variable order*
- Use the same manager for all ROBDDs (Multi-rooted BDDs)
- Maintain CANONICAL form using: ITE operator, OBDD Reduce operations, symbol table implemented as a *unique table*

First Learn the ITE Operator

- Let $Z = ITE(f, g, h) = f \cdot g + f' \cdot h$
- Let an ROBDD w/ top-node $= v$ compute a function $= Z$
- Apply Shannon's expansion on Z w.r.t. v

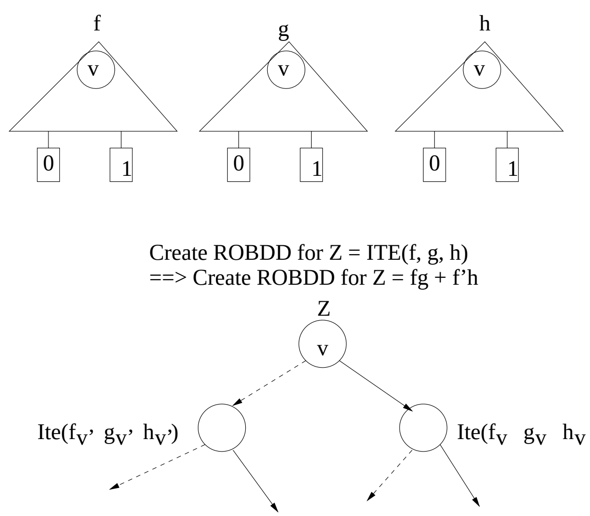
$$\begin{aligned}
Z &= vZ_v + v'Z_{v'} && (1)\\
&= v(ite(f, g, h))_v + v'(ite(f, g, h))_{v'} && (2)\\
&= v(fg + f'h)_v + v'(fg + f'h)_{v'} && (3)\\
&= v(f_v g_v + f'_v h_v) + v'(f_{v'} g_{v'} + f'_{v'} h_{v'}) && (4)\\
&= ite(v, (f_v g_v + f'_v h_v), (f_{v'} g_{v'} + f'_{v'} h_{v'})) && (5)\\
&= ite(v, ite(f_v, g_v, h_v), ite(f_{v'}, g_{v'}, h_{v'})) && (6)\\
&= v \cdot ite(f_v, g_v, h_v) + v' \cdot ite(f_{v'}, g_{v'}, h_{v'}) && (7)
\end{aligned}$$

- Apply ITE at top node $\rightarrow$ Apply ITE to its co-factors!

Significance of the ITE Operator

- Let $Z = ITE(f, g, h) = f \cdot g + f' \cdot h$
- Apply ITE at top node $\rightarrow$ Apply ITE to its co-factors!

$$
\begin{aligned}
Z &= vZ_v + v'Z_{v'} &(8)\\
&= v(ite(f,g,h))_v + v'(ite(f,g,h))_{v'} &(9)\\
&= v \cdot ite(f_v, g_v, h_v) + v' \cdot ite(f_{v'}, g_{v'}, h_{v'}) &(10)
\end{aligned}
$$

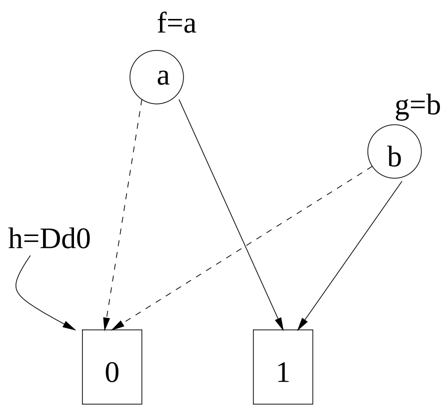


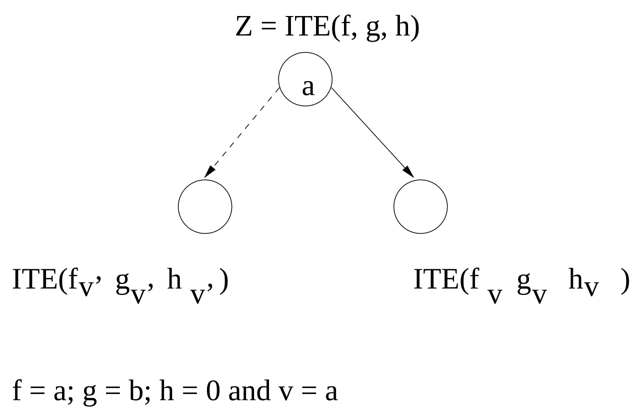
Boolean Computations and ITE

- Compute $f \cdot g$ using ITE operation
- $ITE(f, g, h) = f \cdot g + f' \cdot h$
- $ITE(f, g, 0) = f \cdot g + 0 = f \cdot g$
- Compute $f + g$: $ITE(\_\_, \_\_, \_\_) = f + g$
- Compute $f \oplus g = ITE(\_\_, \_\_, \_\_) = f \cdot g' + f' \cdot g$
- Compute any and all functions using ITE

Build ROBDD using ITE

- $f = a, g = b, h = 0$ and $a \cdot b = ITE(f, g, 0)$
- Var order (in the same BDD Manager) $\equiv$ a=0, b=1, c=2
- $Z = v \cdot ite(f_v, g_v, h_v) + v' \cdot ite(f_{v'}, g_{v'}, h_{v'})$
- v = variable associated w/ topmost nodes among f, g, h $(= a)$

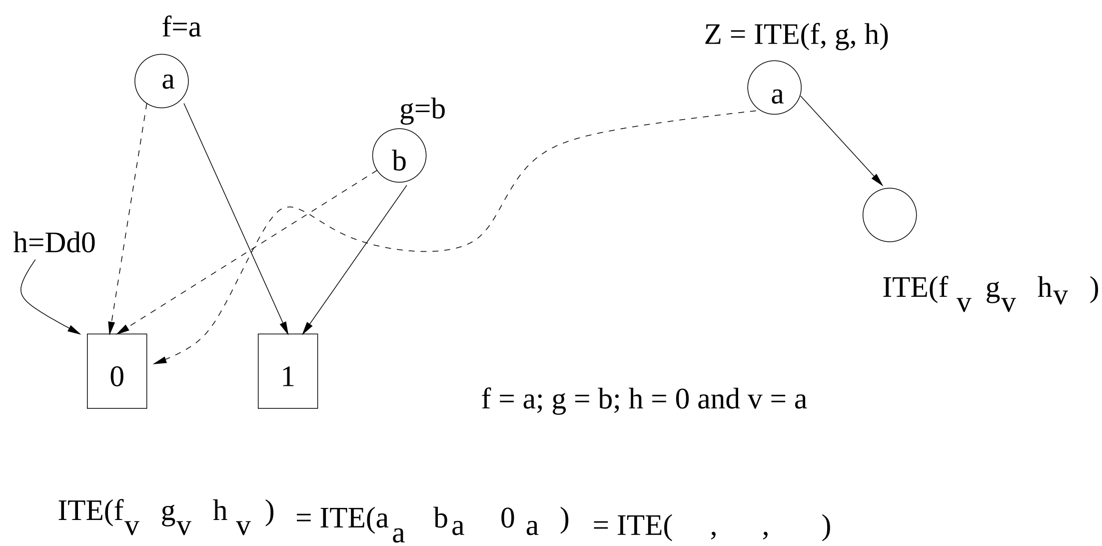




ITE($f_{v'}$, $g_{v'}$, $h_{v'}$) = ITE($a_{a'}$, $b_{a'}$, $0_{a'}$) = ITE(, ,)

Build ROBDD using ITE (*Contd...*)

- $f = a, g = b, h = 0$ and $a \cdot b = ITE(f, g, 0)$
- Var order (in the same BDD Manager) ≡ a=0, b=1, c=2
- $Z = v \cdot ite(f_v, g_v, h_v) + v' \cdot ite(f_{v'}, g_{v'}, h_{v'})$
- v = variable associated w/ topmost nodes among f, g, h $(= a)$

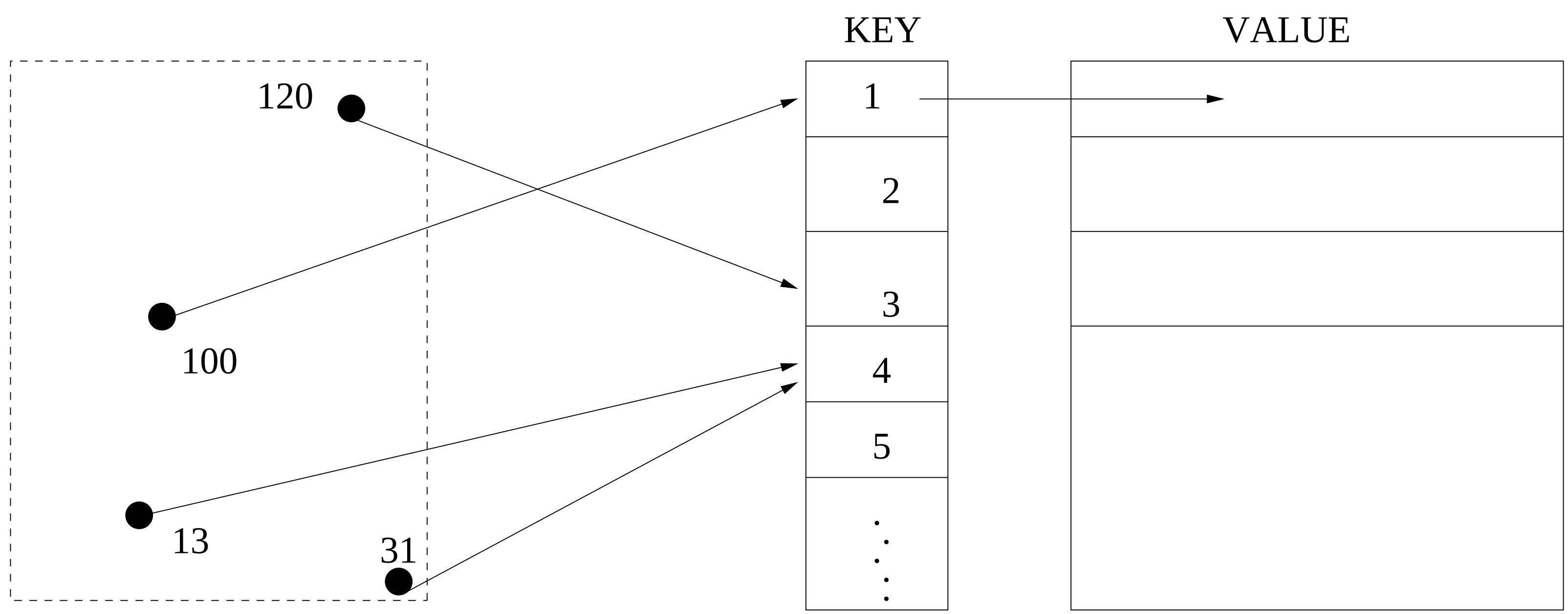


Implementing a Unique Symbol Table

- Reduce OBDD to create ROBDD:
  - Remove redundant nodes (easy)
  - Merge isomorphic subgraphs (difficult)
- How to identify which subgraph is isomorphic to which other subgraph?
- Check for Graph isomorphism is not easy (remember?)
- Solution: Obviate the need to perform isomorphism check!
- How? Using a data structure called "Symbol Table"

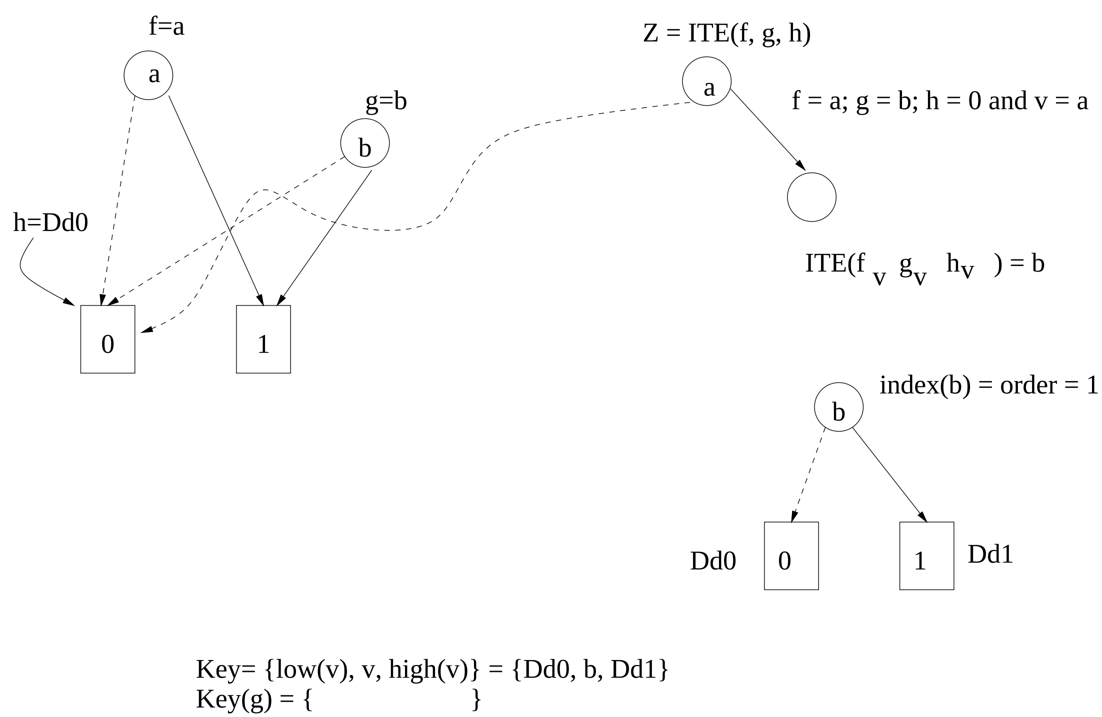
Implementing a Unique Symbol Table

- Pick-up any Data-structures book, and read about symbol tables.



Implementing a Unique Symbol Table

- For a BDD node, Key(node) = {low(v), v, high(v)}
- This Key is unique: equivalent nodes have the same Key!



Implementing a Unique Symbol Table

•

Implementing a Unique Symbol Table

-