# Logic Synthesis & Verification, Fall 2023

**National Taiwan University**

## Reference Solution to Problem Set 2

Due on 2023/10/17 23:59 on NTU Cool.

## 1 [Cofactor]

(a) Let $f = f(x_1, \ldots, x_n)$ be an $n$-variable Boolean function and $v \in \{x_k, \neg x_k\}$ be the literal corresponding to the $k^{\text{th}}$ variable. By definition, $\neg f$ is defined by $(\neg f)^1 = f^0$ and $(\neg f)^0 = f^1$. In other words, $(\neg f)(m_1, \ldots, m_n) = (f(m_1, \ldots, m_n))'$ for any input assignment $m_1 \ldots m_n$, where $m_i \in \{0, 1\}$. Similarly, $(\neg(f_v))(m_1, \ldots, m_n) = (f_v(m_1, \ldots, m_n))'$ for any input assignment $m_1 \ldots m_n$. Now consider an arbitrary input assignment $m_1 \ldots m_n$. Then we have

$$\begin{aligned}((\neg f)_v)(m_1, \ldots, m_n) &= (\neg f)(m_1, \ldots, m_{k-1}, V(v), m_{k+1}, \ldots, m_n) \\ &= (f(m_1, \ldots, m_{k-1}, V(v), m_{k+1}, \ldots, m_n))' \\ &= (f_v(m_1, \ldots, m_n))' \\ &= (\neg(f_v))(m_1, \ldots, m_n),\end{aligned}$$

where

$$V(v) = \begin{cases} 0, & \text{if } v = \neg x_k \\ 1, & \text{otherwise} \end{cases}.$$

Since $(\neg f)_v$ and $\neg(f_v)$ have the same values for all input assignments, they have the same onset and offset. Therefore, $(\neg f)_v = \neg(f_v)$.

(b) Let $f = f(x_1, \ldots, x_n)$ and $g = g(x_1, \ldots, x_n)$ be two $n$-variable Boolean functions and $v \in \{x_k, \neg x_k\}$ be the literal corresponding to the $k^{\text{th}}$ variable. Then $f \to g$ is defined by $(f \to g)^1 = f^0 \cup g^1$ and $(f \to g)^0 = f^1 \cap g^0$. In other words, $(f \to g)(m_1, \ldots, m_n) = (f(m_1, \ldots, m_n))' + g(m_1, \ldots, m_n)$ for any input assignment $m_1 \ldots m_n$, where $m_i \in \{x_i, \neg x_i\}$. Similarly, $(f_v \to g_v)(m_1, \ldots, m_n) = (f_v(m_1, \ldots, m_n))' + g_v(m_1, \ldots, m_n)$ for any input assignment $m_1 \ldots m_n$. Now consider an arbitrary input assignment $m_1 \ldots m_n$. Then we have

$$\begin{aligned}((f \to g)_v)(m_1, \ldots, m_n) &= (f \to g)(m_1, \ldots, m_{k-1}, V(v), m_{k+1}, \ldots, m_n) \\ &= (f(m_1, \ldots, m_{k-1}, V(v), m_{k+1}, \ldots, m_n))' \\ &\qquad + g(m_1, \ldots, m_{k-1}, V(v), m_{k+1}, \ldots, m_n) \\ &= (f_v(m_1, \ldots, m_n))' + (g_v(m_1, \ldots, m_n)) \\ &= (f_v \to g_v)(m_1, \ldots, m_n),\end{aligned}$$

where

$$V(v) = \begin{cases} 0, & \text{if } v = \neg x_k \\ 1, & \text{otherwise} \end{cases}.$$

Since $(f \to g)_v$ and $f_v \to g_v$ have the same values for all input assignments, they have the same onset and offset. Therefore, $(f \to g)_v = f_v \to g_v$.

## 2 [Quantification]

(a)

$$\begin{aligned}
&F_2 \to F_1, F_3, F_4, F_5, F_6, F_7, F_8 \\
&F_3 \to F_1, F_4, F_5 \\
&F_4 \to F_1, F_5 \\
&F_5 \to F_1, F_4 \\
&F_6 \to F_1 \\
&F_7 \to F_1, F_2, F_3, F_4, F_5, F_6, F_8 \\
&F_8 \to F_1, F_2, F_3, F_4, F_5, F_6, F_7
\end{aligned}$$

(b) True. The proof is as follows.

$$\begin{aligned}
\exists x.(f(x,y) \vee g(x,y)) &= (f(0,y) \vee g(0,y)) \vee (f(1,y) \vee g(1,y)) \\
&= (f(0,y) \vee f(1,y)) \vee (g(0,y) \vee g(1,y)) \\
&= (\exists x.f(x,y) \vee \exists x.g(x,y)).
\end{aligned}$$

(c) False. Here is a counterexample. Let $f(x,y) = 0$, $g(x,y) = xy$. Then

$$\begin{aligned}
\exists x.(f(x,y) \vee g(x,y)) &= \exists x.(0 \vee xy) \\
&= \exists x.(xy) \\
&= y,
\end{aligned}$$

while

$$\begin{aligned}
\exists x.f(x,y) \vee \forall x.g(x,y) &= (\exists x.(0) \vee \forall x.(xy)) \\
&= 0 \vee 0 \\
&= 0.
\end{aligned}$$

Since $y \neq 0$ when we assign 1 to $y$, $\exists x.(f(x,y) \vee g(x,y)) \neq (\exists x.f(x,y) \vee \forall x.g(x,y))$.

(d) True. The proof is as follows.

$$\begin{aligned}
\forall x.(f(x,y) \vee g(y)) &= (f(0,y) \vee g(y)) \wedge (f(1,y) \vee g(y)) \\
&= (f(0,y) \wedge f(1,y)) \vee g(y) \\
&= (\forall x.f(x,y)) \vee g(y).
\end{aligned}$$

(e) False. Here is a counterexample. Let $f(x, y) = x$, $g(x, y) = 0$. Then

$$\begin{aligned}\exists x.(f(x,y) \to g(x,y)) &= \exists x.(x \to 0) \\ &= \exists x.(\neg x) \\ &= 1,\end{aligned}$$

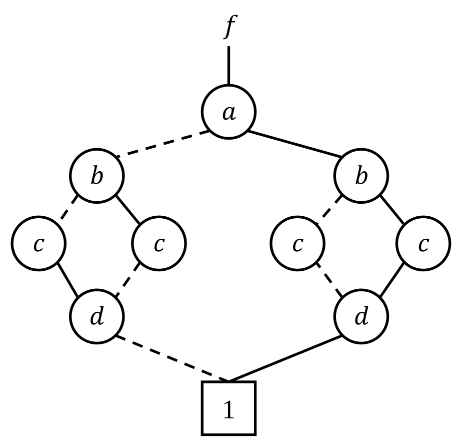
while

$$\begin{aligned}(\exists x.f(x,y)) \to (\exists x.g(x,y)) &= (\exists x.(x)) \to (\exists x.(0)) \\ &= 1 \to 0 \\ &= 0.\end{aligned}$$

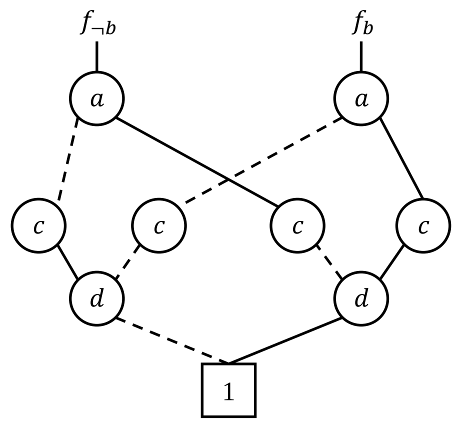
Since $1 \neq 0$ , $\exists x.(f(x,y) \to g(x,y)) \neq (\exists x.f(x,y)) \to (\exists x.g(x,y))$.

## 3 [BDD and ITE]

(a)



(b)




(c)

$$
\begin{aligned}
\frac{\partial f}{\partial b} &= f_b \oplus f_{\neg b} \\
&= ITE(f_b, ITE(f_{\neg b}, 0, 1), f_{\neg b}) \\
&= ITE(f_b, ITE(a, ITE(C, 0, 1), ITE(A, 0, 1)), f_{\neg b}),
\end{aligned}
$$

where

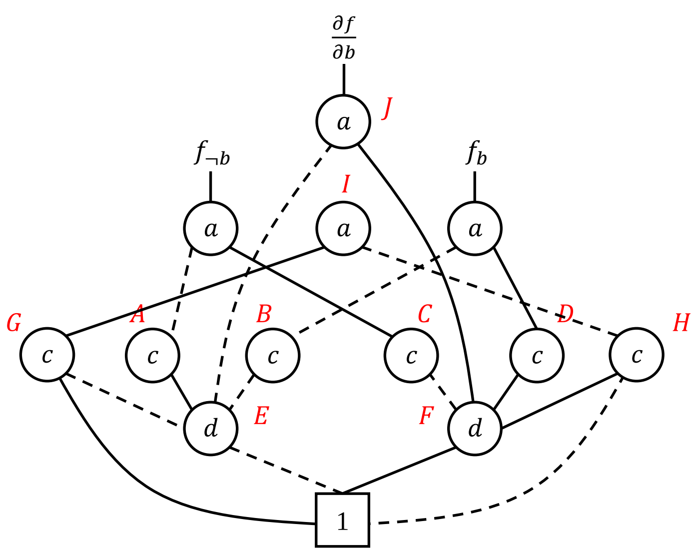
$$
\begin{aligned}
ITE(C, 0, 1) &= ITE(c, ITE(0, 0, 1), ITE(F, 0, 1)) \\
&= ITE(c, 1, ITE(d, ITE(1, 0, 1), ITE(0, 0, 1)) \\
&= ITE(c, 1, ITE(d, 0, 1)) \\
&= ITE(c, 1, E) \\
&= G
\end{aligned}
$$

and

$$
\begin{aligned}
ITE(A, 0, 1) &= ITE(c, ITE(E, 0, 1), ITE(0, 0, 1)) \\
&= ITE(c, ITE(d, ITE(0, 0, 1), ITE(1, 0, 1), 1) \\
&= ITE(c, ITE(d, 1, 0), 1) \\
&= ITE(c, F, 1) \\
&= H.
\end{aligned}
$$

Therefore,

$$\begin{aligned}
\frac{\partial f}{\partial b} &= ITE(f_b, ITE(a, ITE(C,0,1), ITE(A,0,1)), f_{\neg b}) \\
&= ITE(f_b, ITE(a, G, H), f_{\neg b}) \\
&= ITE(f_b, I, f_{\neg b}) \\
&= ITE(a, ITE(D, G, C), ITE(B, H, A)) \\
&= ITE(a, ITE(c, ITE(F,1,0), ITE(0,E,F)), ITE(c, ITE(0,F,E), ITE(E,1,0))) \\
&= ITE(a, ITE(c, F, F), ITE(c, E, E)) \\
&= ITE(a, F, E) \\
&= J
\end{aligned}$$




## 4 [BDD Onset Counting]

The algorithm is shown below. Since each node is traversed only twice (including initialization), this process is in linear time.

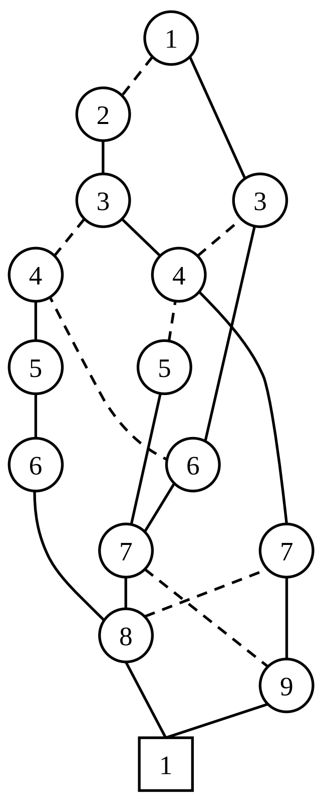
**Algorithm 1:** ROBDD Onset Counting

**Input :** $G$: an ROBDD;
$n$: the number of variables
**Output:** $G.const_1.count$: The number of onset minterms of $G$

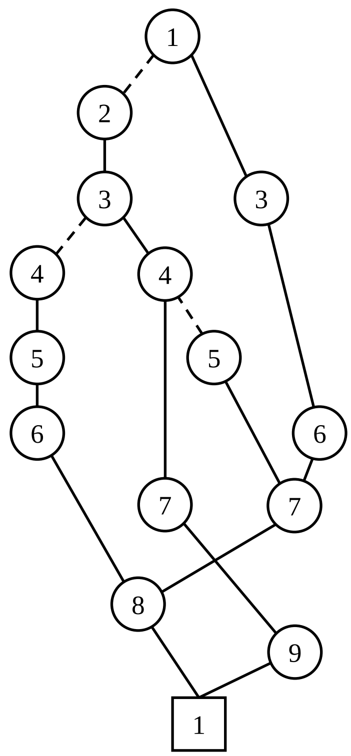
**for** each node $v$ **do**
  $v.count \leftarrow 0$
**end**
$G.root.count \leftarrow 2^n$
**for** each node $v$ in the top-down order **do**
  $v.left.count \leftarrow v.left.count + v.count/2$
  $v.right.count \leftarrow v.right.count + v.count/2$
**end**
**return** $G.const_1.count$

## 5 [ZDD]

(a) There are 13 simple paths, including {148, 1479, 1578, 159, 13678, 1369, 2348, 23479, 23578, 2359, 26548, 2678, 269}. The ZDD is as follows.

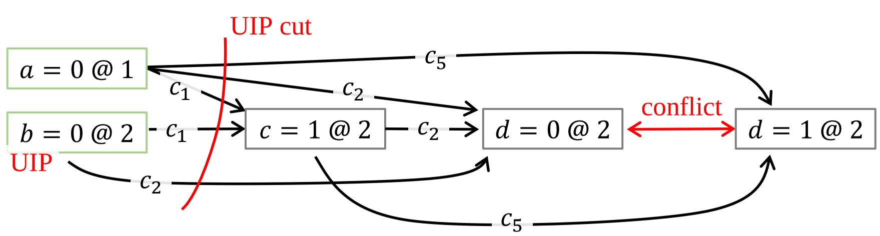


(b) There are 4 Hamiltonian paths, including {13678, 23479, 23578, 26548}. The ZDD is as follows.

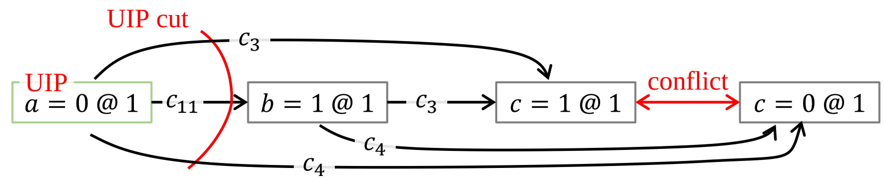



## 6 [SAT Solving]

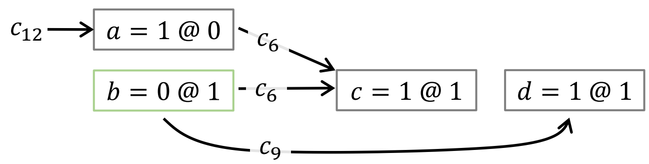
(a)




There is only one possible learned clause, $C_{11} = a + b$.




There is only one possible learned clause, $C_{12} = a$.




A satisfying assignment $(a, b, c, d) = (1, 0, 1, 1)$ is found.

(b) Sources of clause $C_{11} = a + b$:

$$C_1 = (a + b + c), C_2 = (a + b + c' + d'), C_5 = (a + c' + d)$$

Resolution:

$$\cfrac{\cfrac{C_2 \qquad C_5}{a + b + c'} \qquad C_1}{a + b}$$

Sources of clause $C_{12} = a$:

$$C_3 = (a + b' + c), C_4 = (a + b' + c'), C_{11} = (a + b)$$

Resolution:

$$\cfrac{\cfrac{C_3 \quad C_4}{a + b'} \qquad C_{11}}{a}$$

## 7 [SAT Solving]

(a) We introduce variables $P_{i,j}$ for $1 \leq i \leq m$, $1 \leq j \leq n$. Let $P_{i,j} = 1$ if and only if the pigeon $i$ is in the hole $j$. Then the CNF contains two parts:

$$\bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} P_{i,j} \tag{1}$$

poses the constraint that each pigeon must be in some hole;

$$\bigwedge_{j=1}^{n} \bigwedge_{i=1}^{m-1} \bigwedge_{k=i+1}^{m} (\neg P_{i,j} \vee \neg P_{k,j}) \tag{2}$$

poses the constraint that each hole contains at most one pigeon. The formula size is $m + 0.5nm(m-1)$ (in terms of number of clauses).
Some may additionally pose the constraint that one pigeon can only be in at most one hole:

$$\bigwedge_{i=1}^{m} \bigwedge_{j=1}^{n-1} \bigwedge_{k=j+1}^{n} (\neg P_{i,j} \vee \neg P_{i,k}) \tag{3}$$

Then the formula size would become $m + 0.5nm(m-1) + 0.5mn(n-1)$ (in terms of number of clauses).

(b) Yes, the solver is expected to be scalable on this problem because the CNF for the case $n = m$ is satisfiable. The solver can evoke some implications to lead to a satisfying assignment.

(c) No, the solver is not expected to be scalable on this problem because the CNF for the case $m = n + 1$ is unsatisfiable, and the number of backtracks in solver grows exponentially.

Note: The scalability trend may not seem clear for $m = 4, 5, 6$. However, if you experiment with values from $m = 7$ to $m = 11$, you will find that when $m = n$, the required number of decisions is approximately $O(m^2)$. In contrast, the required number of decisions grows exponentially when $m = n + 1$.