Test Pattern Generation

Virendra Singh
Associate Professor
Computer Architecture and Dependable Systems Lab
Dept. of Electrical Engineering
Indian Institute of Technology Bombay
viren@ee.iitb.ac.in

EE 709: Testing & Verification of VLSI Circuits

Lecture – 11 (Jan 26, 2012)

Boolean Difference

- **Shannon's Expansion Theorem:**

$$F(X_1, X_2, \ldots, X_n) = X_2 \bullet F(X_1, 1, \ldots, X_n) + \overline{X_2} \bullet F(X_1, 0, \ldots, X_n)$$

- **Boolean Difference (partial derivative):**

$$\frac{\partial F_j}{\partial g} = F_j(1, X_1, X_2, \ldots, X_n) \oplus F_j(0, X_1, \ldots, X_n)$$

- **Fault Detection Requirements:**

$$G(X_1, X_2, \ldots, X_n) = 1$$

$$\frac{\partial F_j}{\partial g} = F_j(1, X_1, X_2, \ldots, X_n) \oplus F_j(0, X_1, \ldots, X_n) = 1$$

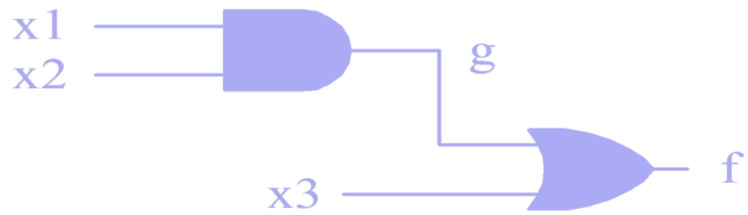
Boolean Difference

$$f(x_1,\ldots,x_i=0,\ldots,x_n)\oplus f(x_1,\ldots,x_i=1,\ldots,x_n)=1$$

- Represented by the symbol *df(x)/dx*
- *df(x)/dx*$_{i \text{ for } x=0}$ and *df(x)/dx*$_{i \text{ for } x=1}$ are called *the residues/co-factors* of the function for $x = xi$
- One of the residue is the good-circuit value and the other is the faulty-circuit value for x_i
- To detect the fault, the two residues should be complementary
- Solving the equation yield the values of the primary inputs to detect a stuck-at fault on x_i
- The test pattern is: $x_i\, df(x)/dx_i = 1$ & $x_i'\, df(x)/dx_i = 1$

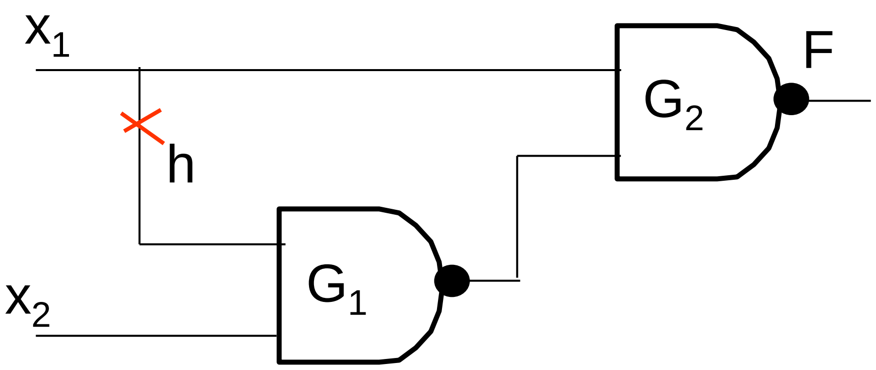
Fault Detection

- $x_i \, df(x)/dx_i = 1$ for s-a-0 at x_i
- $x_i' \, df(x)/dx_i = 1$ for s-a-1 at x_i
- As an example, let us consider the function



- $f(x) = x_1 x_2 + x_3$
- Thus $df(x)/dx_2 = x_3 \oplus (x_1 + x_3) = x_3' x_1 = 1$. Then
- $x_1 = 1$ and $x_3 = 0$.
- For the SA1 and SA0 faults on x_2, the patterns are then $x_1 x_2 x_3 = (100)$ and (110), respectively.

Fault Detection



$F(X,h) = x_1' + hx_2$

$h(X) = x_1$

$dF^*(X,h)/dh = x'_1 \oplus (x_1' + x_2)$

$= x_1x_2$

S-a-0 fault at h

Test Vector

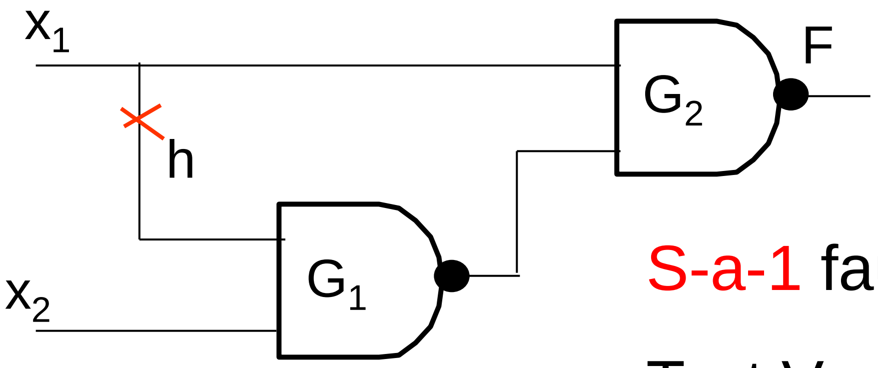
$h(X)\ dF^*(X,h)/dh = 1$

$x_1 .\ x_1x_2 = x_1x_2 = 1$

$x_1 = 1$

$x_2 = 1$

Fault Detection



S-a-1 fault at h

Test Vector

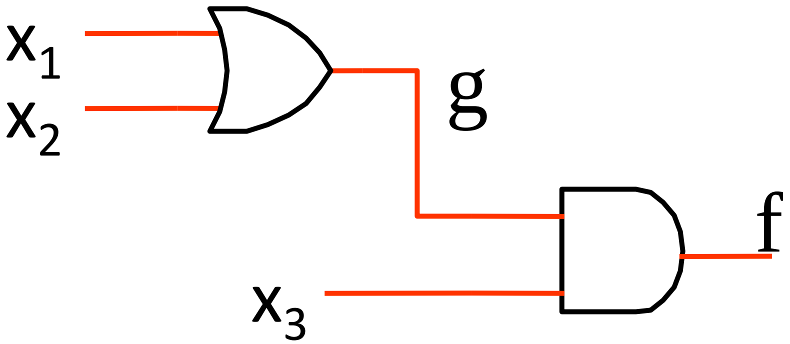
$h(X)' \, dF^*(X,h)/dh = 1$

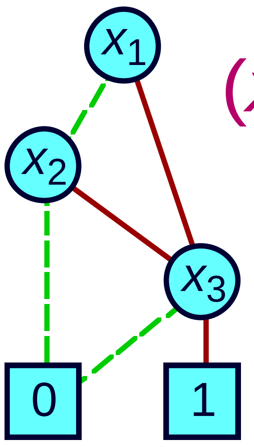
$x'_1x_1x_2 = x_1x_2 = 0$

It cannot satisfy required condition

Fault - Redundant

TG using BDDs (1/4)

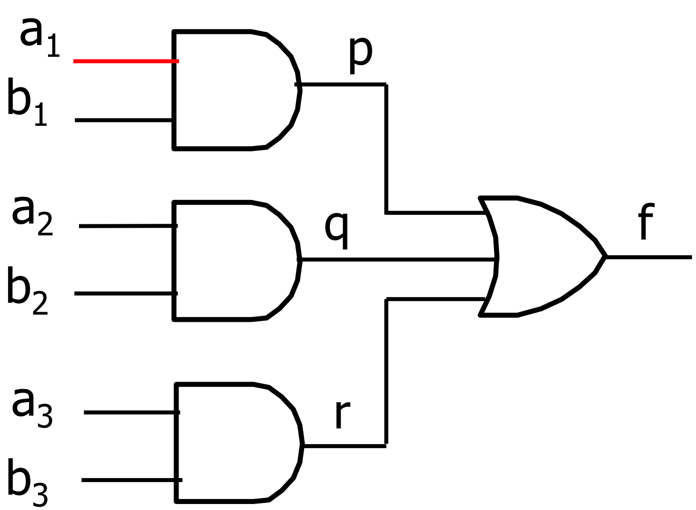


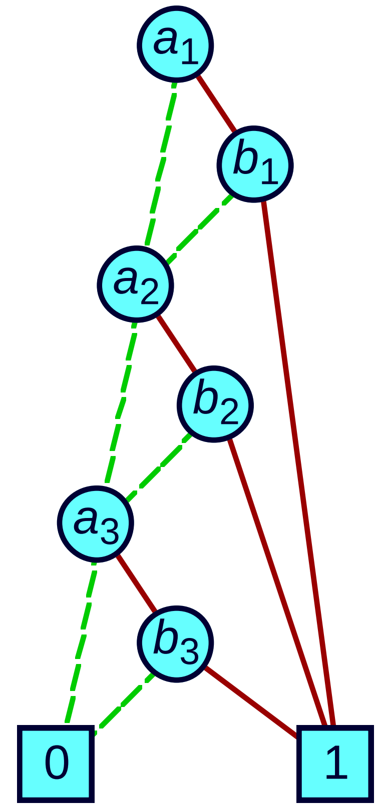


- Trace a path from the root to 0 and 1
- Value of the variables other than fault should have same value
- TP for s-a-0 fault at x_1 is $x_1x_2x_3$ = 101
- TP for s-a-1 fault at x_1 is $x_1x_2x_3$ = 001

TG using BDDs (2/4)

$$(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee (a_3 \wedge b_3)$$



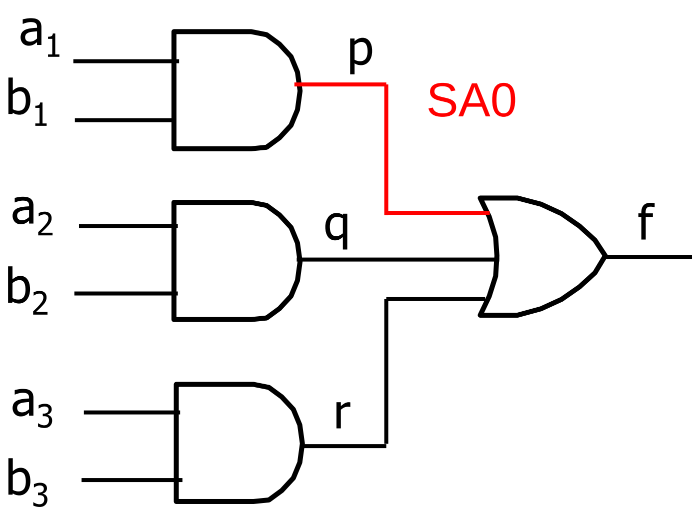


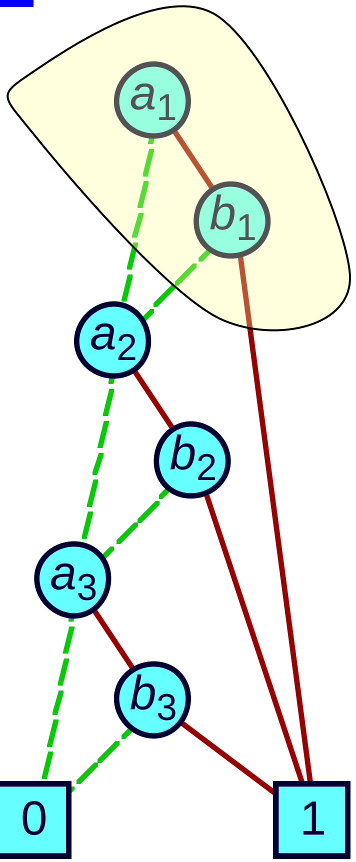
s-a-0 at a_1

$a_1=1$, $b_1=1$, $a_2=0$, $a_3=0$

TG using BDDs (3/4)

$$(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee (a_3 \wedge b_3)$$



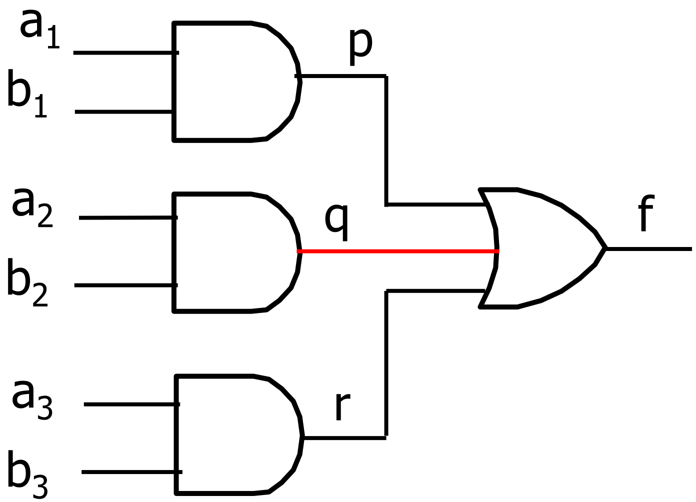


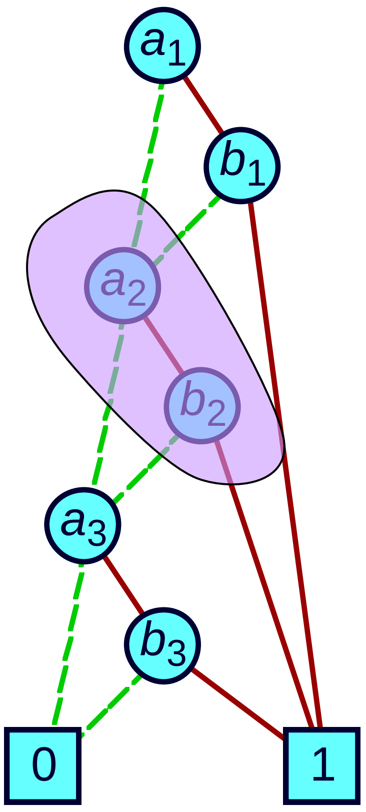
s-a-0 at p

a_1=1, b_1=1, a_2=0, a_3=0

TG using BDDs (4/4)

$$(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee (a_3 \wedge b_3)$$





s-a-0 at q

$a_2=1$, $b_2=1$, $a_1=0$, $a_3=0$

ATPG - Algorithmic

- Path Sensitization Method
  - Fault Sensitization
  - Fault Propagation
  - Line Justification

- Path Sensitization Algorithms
  - D- Algorithm (Roth)
  - PODEM (P. Goel)
  - FAN (Fujiwara)
  - SOCRATES (Schultz)
  - SPIRIT (Emil & Fujiwara)

Path Sensitization

General Structure of TG Algorithm

begin

set all values to x

Justify (l, v)

if (v = 0) then *Propagate (l, D)*

else *Propagate (l, D')*

end

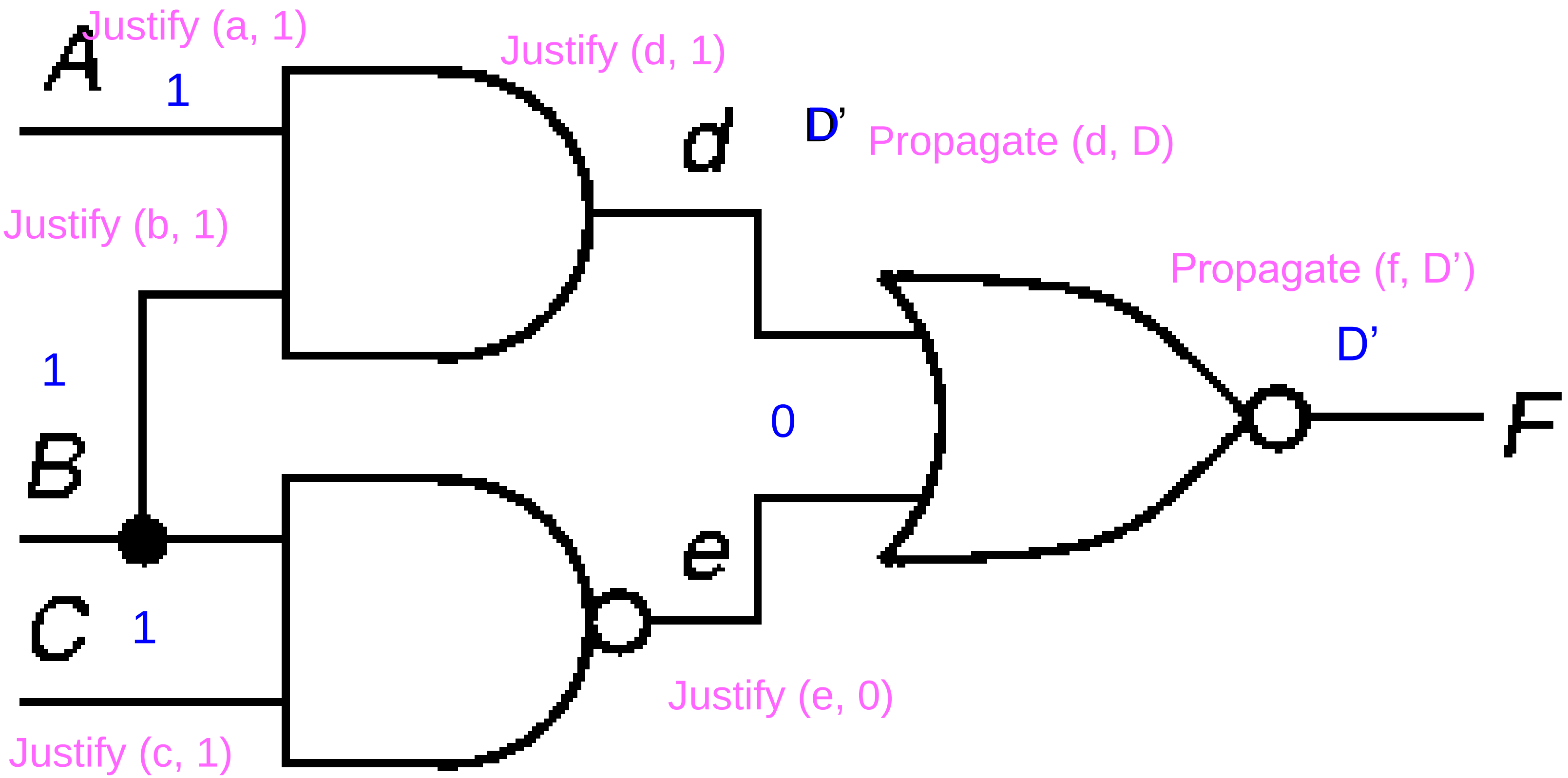
Path Sensitization

```
Justify( l, val)
begin
    set l to val
    if l is a PI then return
    /* l is a gate output */
    c = controlling value of l
    i = inversion of l
    inval = val ⊕ i
    if (inval = c')
        then for every input j of l
                Justify (j, inval)
        else
                select one input ( j ) of l
                Justify (j, inval)
end
```

Path Sensitization

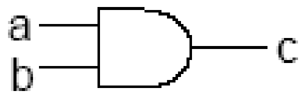
```
Propagate (l, err)
/* err is D or D' */
begin
    set l to err
    if l is PO then RETURN
    k = fanout of l
    c = controlling value of k
    i = inversion of k
    for every input of j of k other than l
        Justify ( j, c' )
    Propagate ( k, err ⊕ i )
end
```

Path Sensitization



Common Concept

- Fault Activation problem → a LJ Problem
- The Fault Propagation problem →
  1. Select a FP path to PO → Decision
  2. Once the path is selected → a set of LJ problems
- The LJ Problems → Decisions or Implications



To justify c = 1 → a = 1, b = 1 (Implication)

To justify c = 0 → a = 0 or b = 0 (Decision)

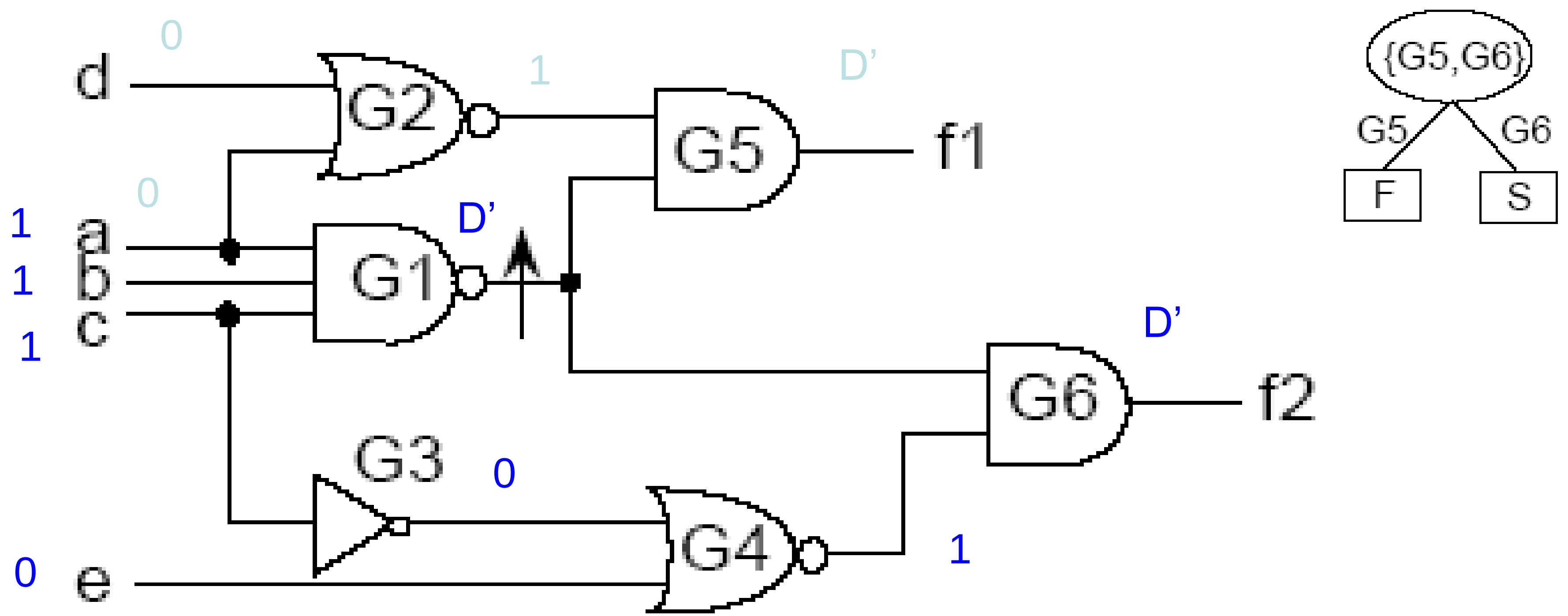
- Incorrect decision → Backtrack → Another decision

D-Algorithm

Roth (IBM) - 1966

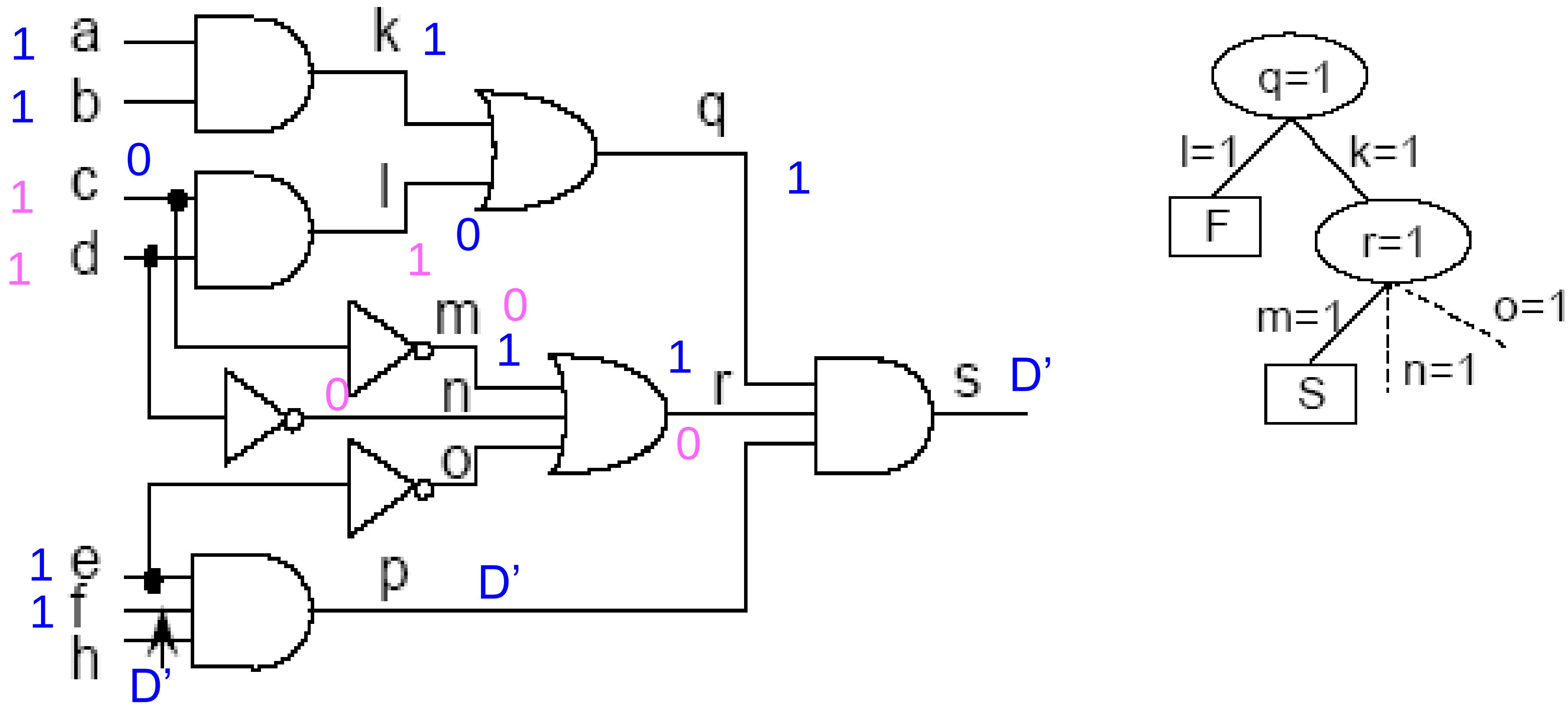
- Fundamental concepts invented:
  - First complete ATPG algorithm
  - D-Calculus (5 valued logic)
  - Implications – forward and backward
  - Implication stack
  - Backtrack
  - Test Search Space

Decisions during FP



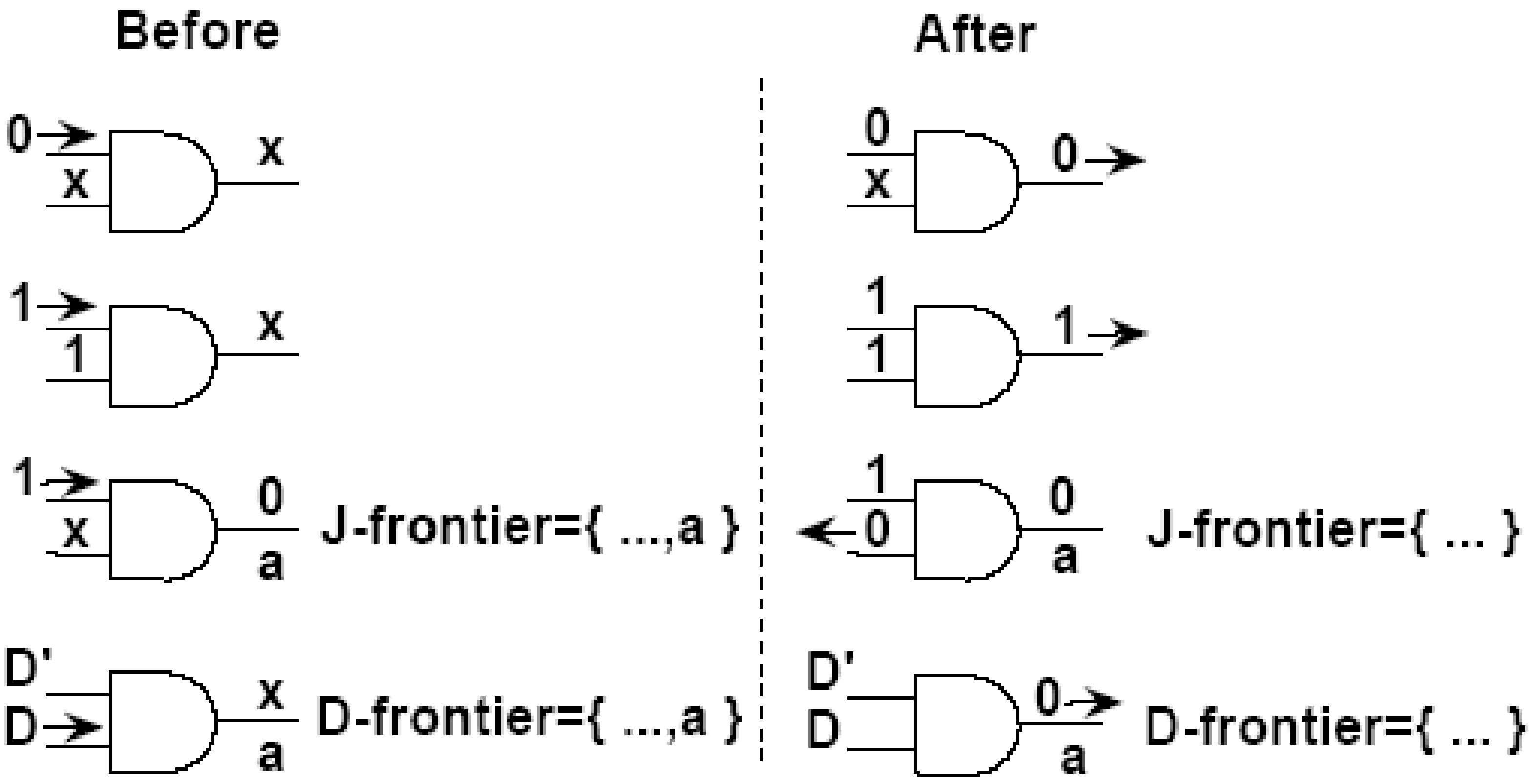
D – frontier: The set of all gates whose output value is currently x but have one or more fault signals on their inputs

Decisions during LJ

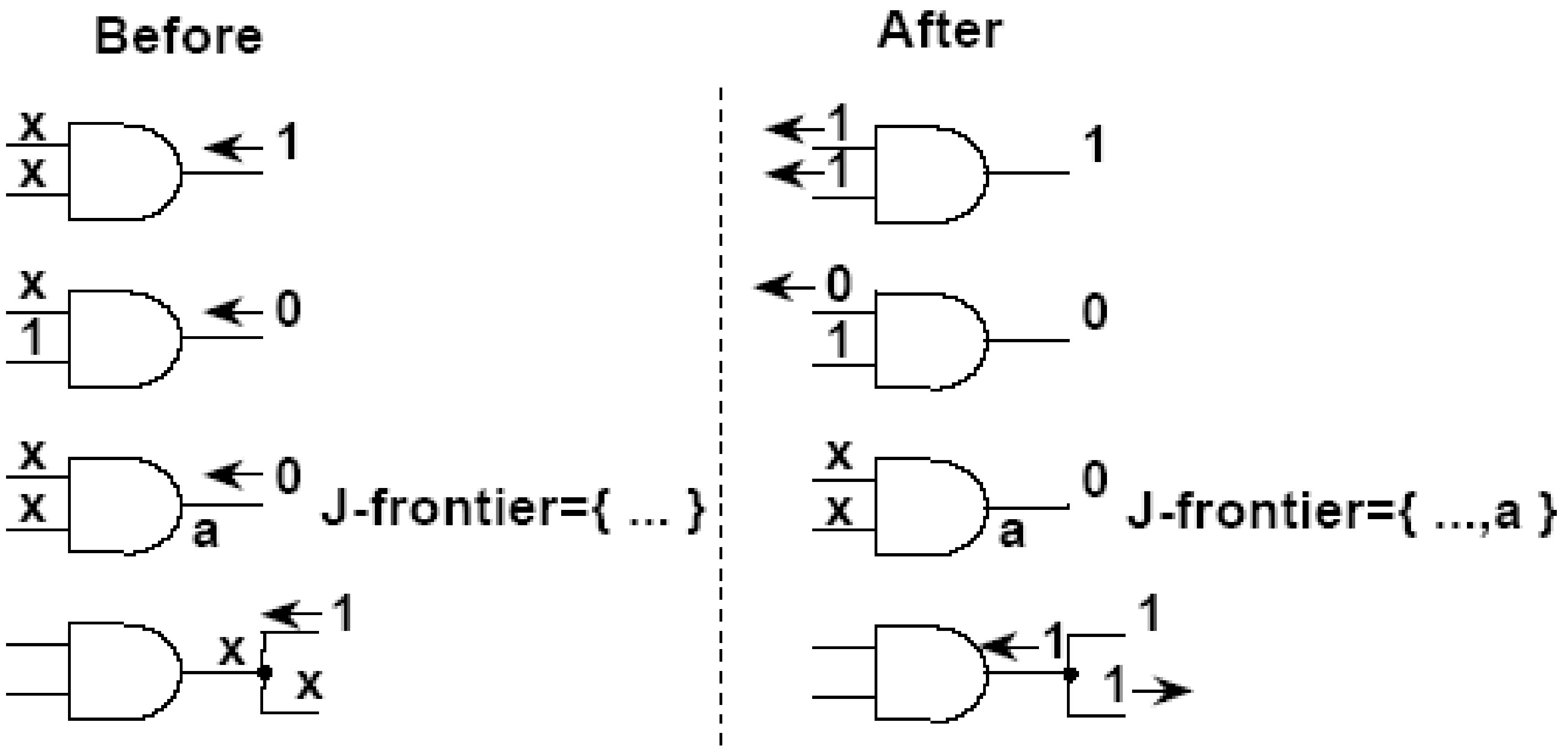


J – Frontier : A set of all gates whose output value is known but not implied by its input value

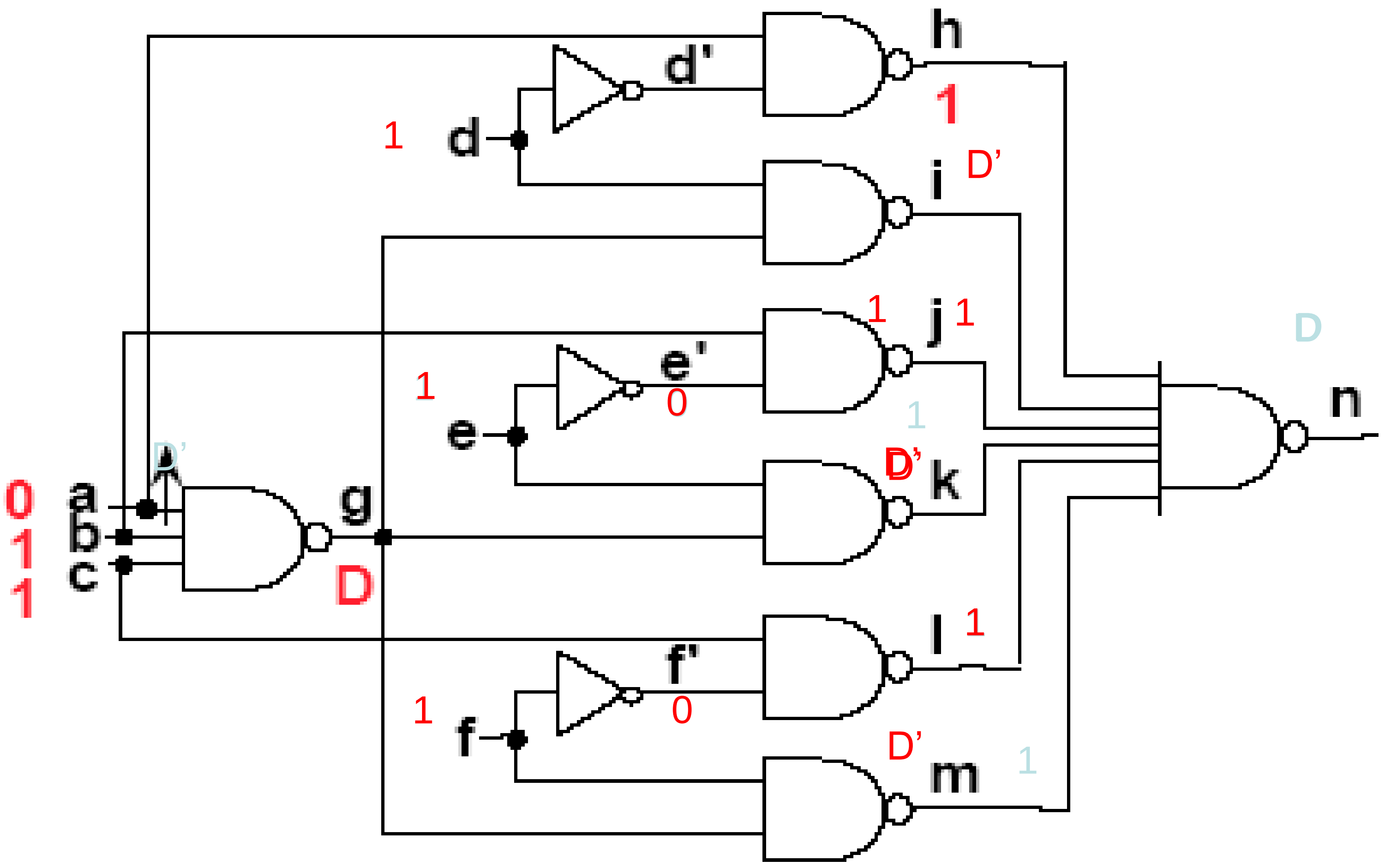
Implications (Forward)



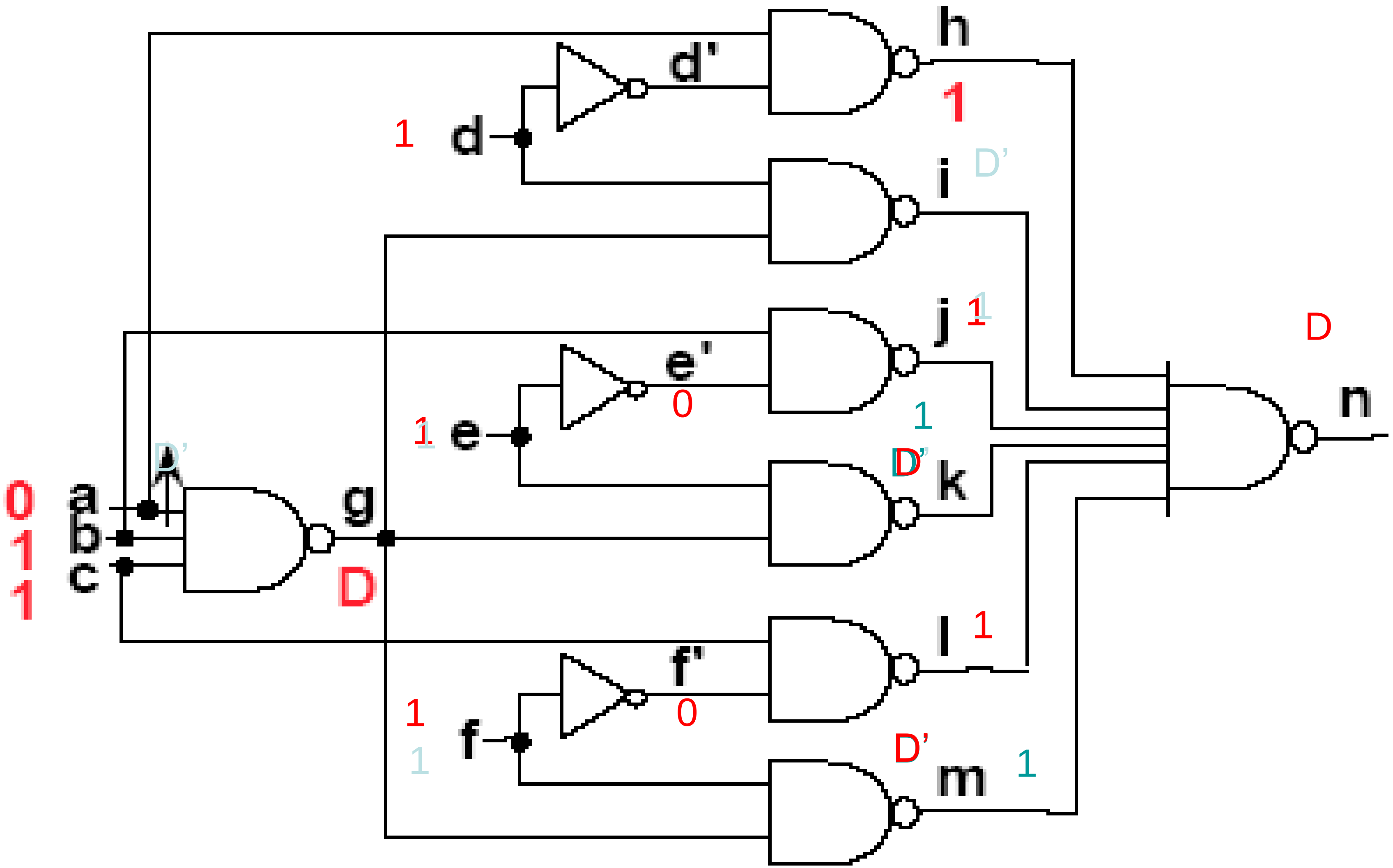
Implications (Backward)



D-Algorithm : Example



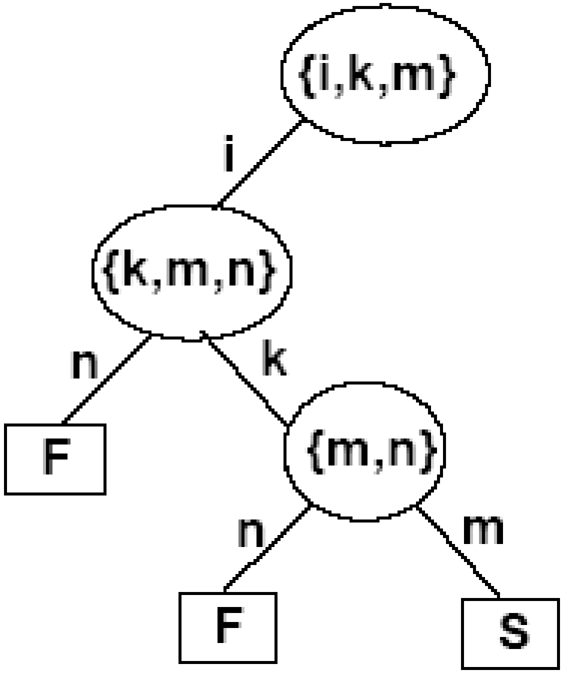
D-Algorithm : Example



Value Computation

Decision	Implication	Comments
	a=0 h=1 b=1 c=1 g=D	Active the fault Unique D-drive
d=1	i=D d?0	Propagate via i
j=1 k=1 l=1 m=1	n=D e?0 e=1 k=D	Propagate via n Contradiction
e=1	k=D e?0 j=1	Propagate via k
l=1 m=1	n=D f?0 f=1 m=D	Propagate via n Contradiction
f=1	m=D f?0 l=1 n=D	Propagate via m

Decision Tree



Two times of backtracking!!

Thank You