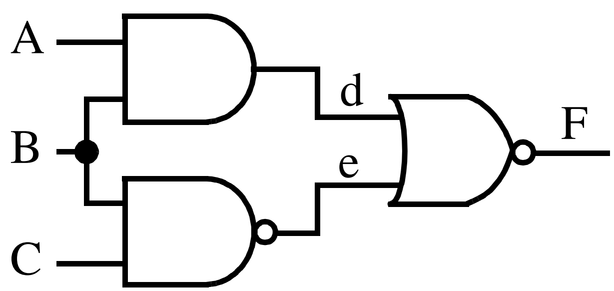
D-Algorithm -- Roth (IBM 1966)

Roth's D-Algorithm (D-ALG) defined the calculus and algorithms for ATPG using D-cubes.

Definitions

- ***Singular cover***: Defined to be the minimal set of input signal assignments needed to represent ***essential prime implicants*** in Karnaugh map.



AND	a	b	d	NOR	d	e	F
1	0	X	0	*4*	1	X	0
2	X	0	0	*5*	X	1	0
3	1	1	1	*6*	0	0	1

D-Calculus and D-Algorithm

- ***D-cube***: A collapsed truth table entry.

  For example, combine rows 3 and 1 of the AND gate singular cover, and express it in Roth's 5-valued algebra (row 3 is good machine).

  $D\ 1\ D$

  Rows 3 and 2 yield the ***propagation*** *D-cube*:

  $1\ D\ D$

  A third is $D\ D\ D$

  Inverting D to $\overline{D}$ in each of these yields the 6 ***D-cubes*** for the AND gate.

  3 of the NOR gate ***D-cubes*** are:

  $D\ 0\ \overline{D}$

  $0\ D\ \overline{D}$

  $D\ D\ \overline{D}$

D-Calculus and D-Algorithm

- ***D-intersection***: Define how different ***D-cubes*** can coexist for different gates in a logic circuit.

$$0 \cap 0 = 0 \cap X = X \cap 0 = 0$$

$$1 \cap 1 = 1 \cap X = X \cap 1 = 1$$

$$X \cap X = X$$

Rule: If one cube assigns a specific signal value, the other cubes must assign either the same signal or X

For example, ***0 X X*** intersect ***1 X X*** is the empty cube (incompatible).

D-intersection	*0*	*1*	*X*	*D*	$\overline{D}$
0	0	ϕ	0	ψ	ψ
1	ϕ	1	1	ψ	ψ
X	0	1	X	D	$\overline{D}$
D	ψ	ψ	D	μ	λ
$\overline{D}$	ψ	ψ	$\overline{D}$	λ	μ

D-Calculus and D-Algorithm

- ***D-intersection*** (contd.):

  The greek symbols ϕ and ψ represent incompatible assignments.

  If the values are incompatible during propagation or implications, the assignment is called ***inconsistent*** and ***backtracking*** is necessary.

  Greek symbols μ and λ indicate incompatibilities if *both* are present in ***D-cubes*** with multiple input D and $\overline{D}$.

  For example, if only 1 occurs, invert the Ds in the second cube and perform intersection.

- ***D-contains***: A cube ***A*** ***D-contains*** cube ***B*** if the set of ***A*** cube vertices contains (is a superset of) the ***B*** cube vertices.

D-Calculus and D-Algorithm

- ***Primitive D-cubes of failure (PDF)***: These model faults including:
  - ***SA0*** (represented by D)
  - ***SA1*** (represented by $\overline{D}$)
  - Bridging faults (short circuits)
  - Arbitrary change in logic gate function (e.g., from AND to OR).

For the AND gate, the ***PDF*** for output ***SA0*** is ***1 1 D***

Here the good machine generates a ***1*** when both inputs are ***1***, while the bad machine generates a ***0***.

The ***PDF***s for the AND gate output ***SA1*** are ***0 X*** $\overline{D}$ and ***X 0*** $\overline{D}$.

Note the ***PDF*** are distinct from the ***propagation D-cubes***.

The former models a failure at the gate.

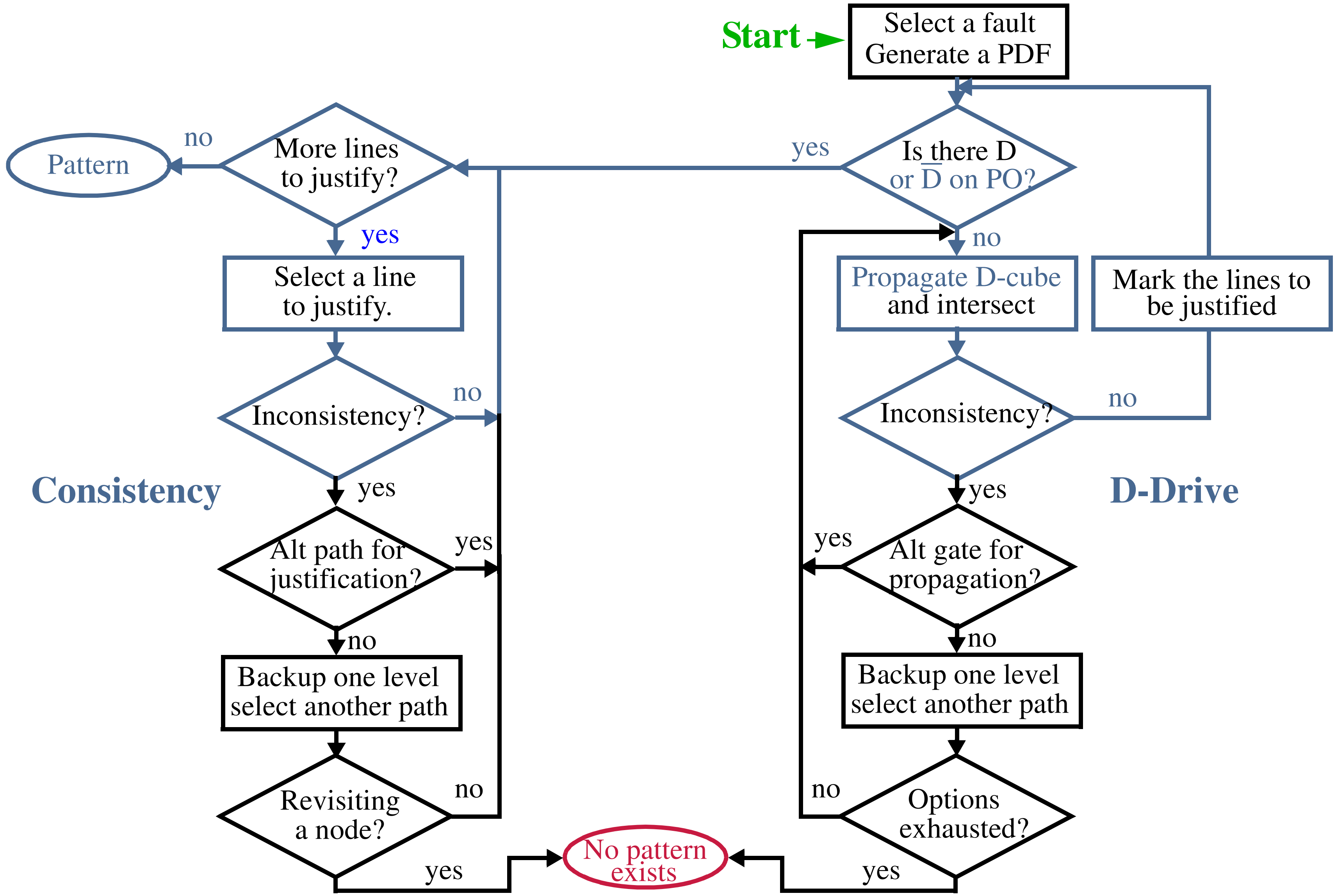
The latter models the conditions for fault effect propagation.

D-Calculus and D-Algorithm

- ***Implication procedure***: Consists of the following steps:
  - Model the fault with the appropriate ***PDF***.
  - Select ***propagation D-cubes*** to propagate fault-effect to PO(s) (***D-drive procedure***).
  - Select singular cover cubes to justify internal circuit signals (***consistency procedure***).

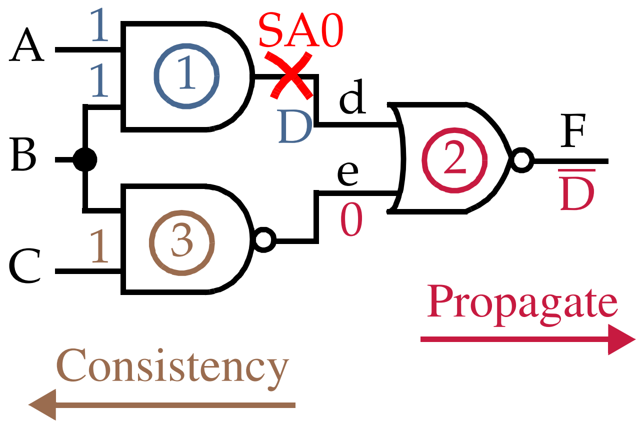
The D-algorithm's main problem is that it selects cubes and singular covers arbitrarily during test generation.

D-ALG


Start
Select a fault
Generate a PDF
Is there D
or $\overline{D}$ on PO?
yes
no
More lines
to justify?
Pattern
yes
Select a line
to justify.
Inconsistency?
no
yes
Alt path for
justification?
yes
no
Backup one level
select another path
Revisiting
a node?
no
yes
No pattern
exists
Consistency
Propagate D-cube
and intersect
Mark the lines to
be justified
Inconsistency?
no
yes
D-Drive
Alt gate for
propagation?
yes
no
Backup one level
select another path
Options
exhausted?
no
yes

D-ALG Examples

Assign PDF



Truth Table

A	*B*	*C*	*F*
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Singular Cover

A	*B*	*C*	*d*	*e*	*F*
1	1		1		
0			0		
	0		0		
	1	1		0	
	0			1	
		0		1	
				1	0
			1		0
			0	0	1

Propagation D-cubes

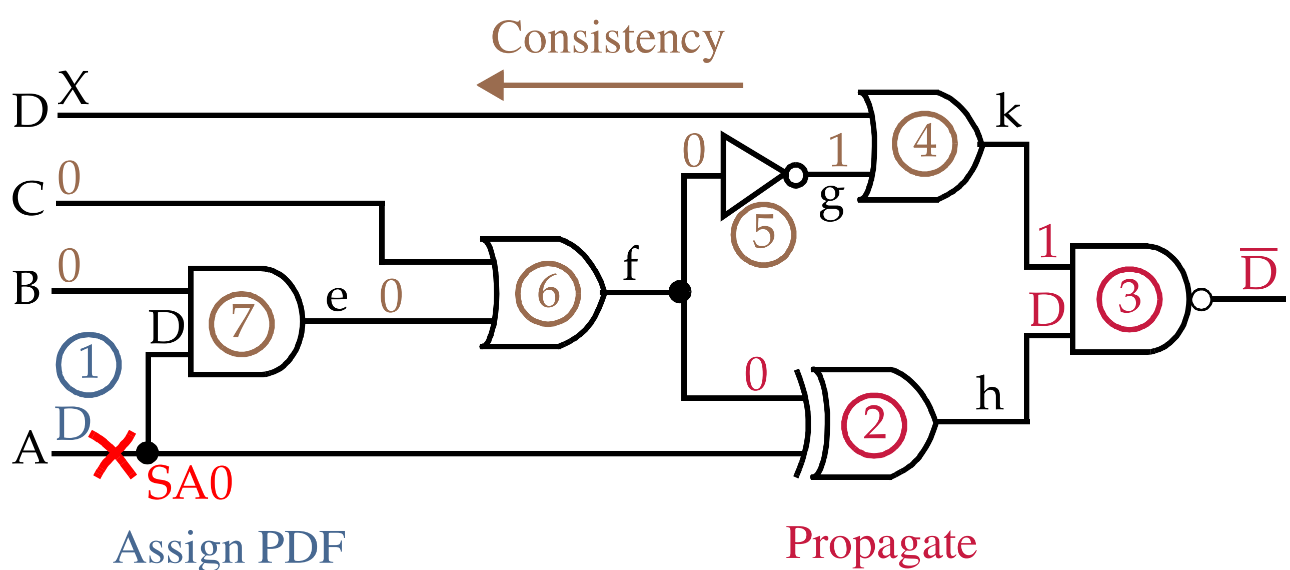
A	*B*	*C*	*d*	*e*	*F*
D	1		D		
1	D		D		
D	D		D		
	D	1		$\overline{D}$	
	1	D		$\overline{D}$	
	D	D		$\overline{D}$	
			D	0	$\overline{D}$
			0	D	$\overline{D}$
			D	D	$\overline{D}$

D-ALG Examples

The following procedure is carried out for ***d SA0*** in the previous circuit:

Step	*A*	*B*	*C*	*d*	*e*	*F*	*Type of cube*
1	1	1		D			PDF for AND gate
2				D	0	$\overline{D}$	*Propagation D-cube* for NOR gate
3		1	1		0		Singular cover of NAND gate

Example #2:



D-ALG Examples

Steps followed to generate ***test cube (tc)***:

	Step		*A*	*B*	*C*	*D*	*e*	*f*	*g*	*h*	*k*	*L*
D-drive	1		D									
	2		**D**					**0**		**D**		
	3		D					0		**D**	**1**	$\mathbf{\overline{D}}$
Consistency	4	or							1		1	
	5	not						0	1			
	6	or			0		0	0				
	7	and		0			0					
		tc	D	0	0		0	0	1	D	1	$\overline{D}$
D-chain dies												

This example and table is given in Roth's paper.

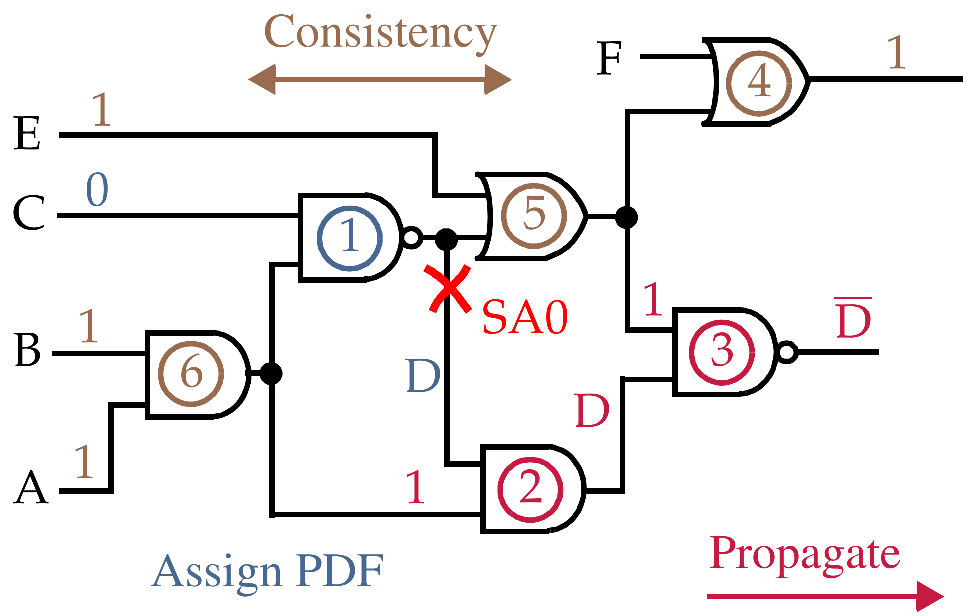
Several other examples are covered in the paper.

Note that all implications are performed in the ***consistency*** procedure here.

A later example by our authors indicates ***the implications*** are carried out after each *D* ***propagation*** step in the ***D-drive****?*

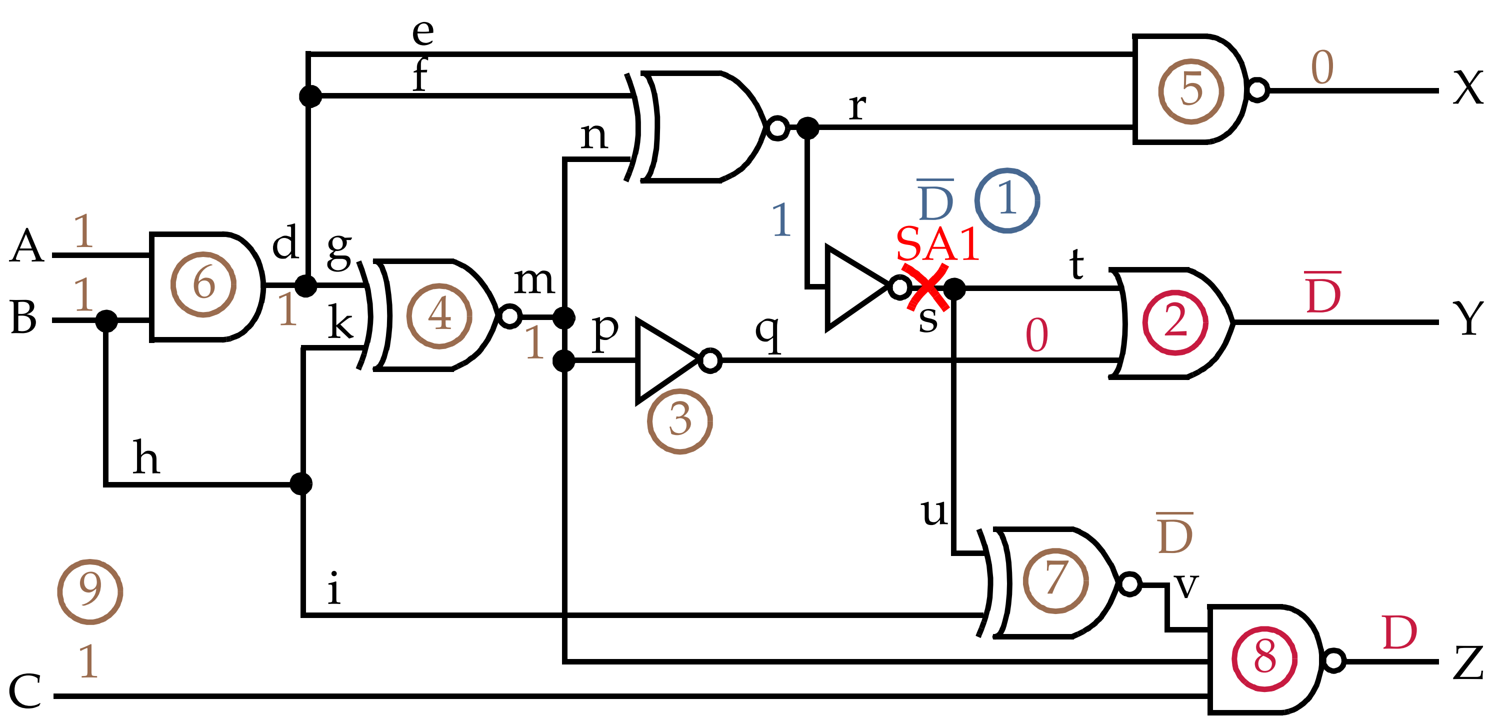
D-ALG Examples

Example #3



D-ALG Examples

Example #4



Assign PDF Propagate Consistency