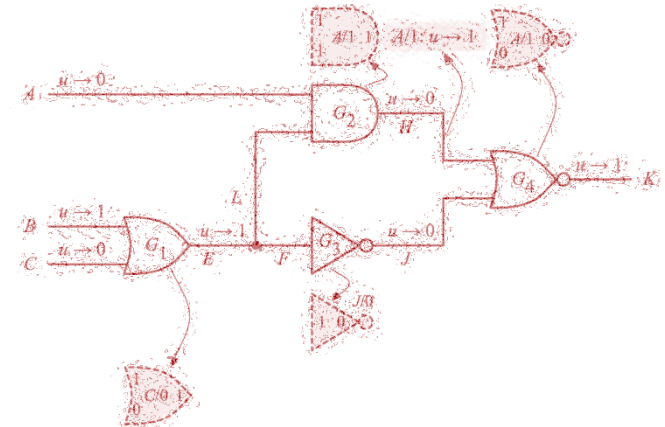
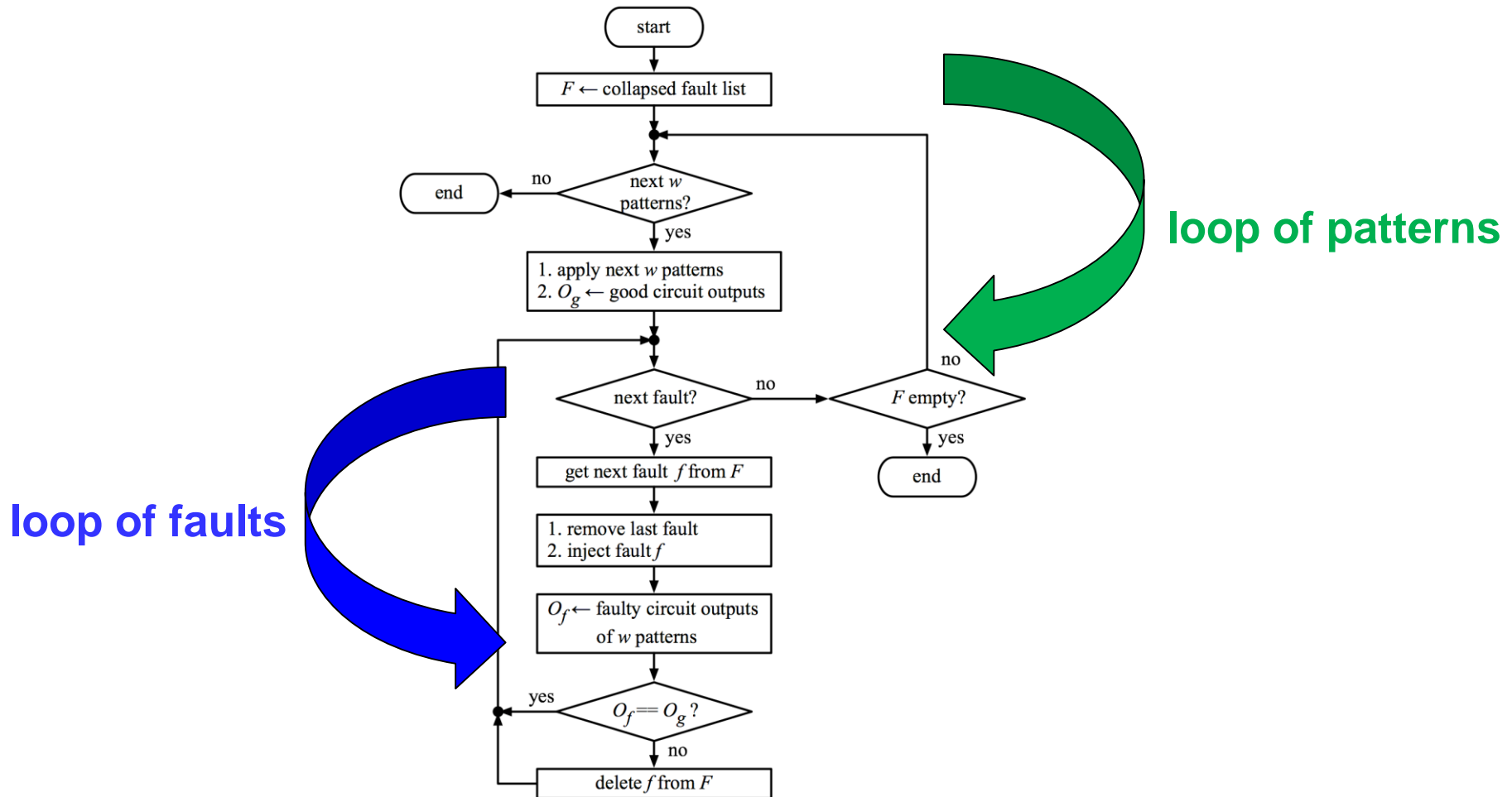


Fault Simulation

- Introduction
- Fault simulation techniques
 - ◆ Serial fault simulation
 - ◆ Parallel fault simulation (1965)
 - ◆ PPSFP (1985)
 - ◆ Deductive fault simulation (1972)
 - ◆ Concurrent fault simulation (1974)
 - ◆ Differential fault simulation (1989)
- Alternatives to fault simulation
- Issues of fault simulation
- Concluding remarks



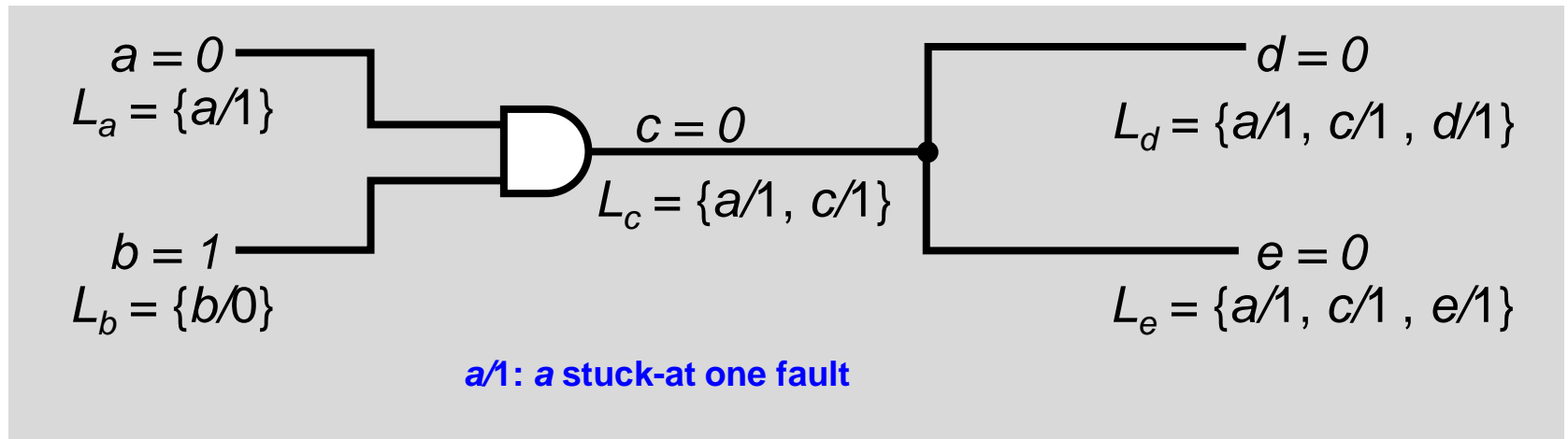
Two Loops Is Slow!



Can we Do Better?

Deductive Fault Simulation [Armstrong 72]

- Idea: Use **logic deduction** instead of circuit simulation
- **Fault list of signal a (L_a)**
 - ♦ Fault $\alpha \in L_a$, if a 's value changes in the presence of α
- **Fault list propagation**
 - ♦ $a/1$ in L_a is propagated to L_c because c is changed when $a/1$
 - ♦ $b/0$ in L_b is **NOT** propagated to L_c because c is unchanged when $b/0$

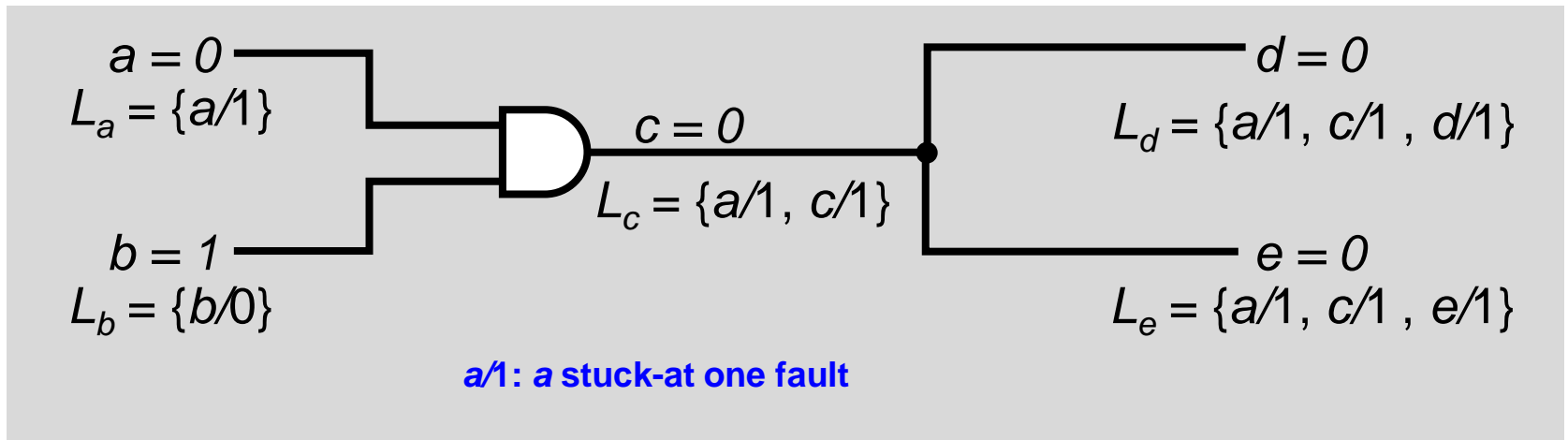


No Loop of Faults !

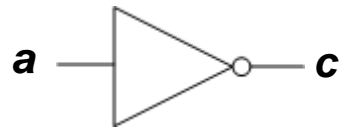
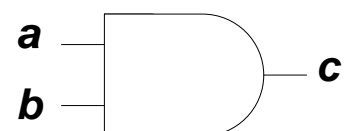
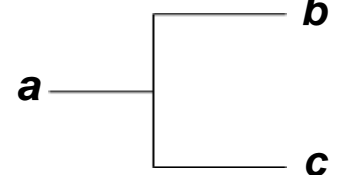
Deductive Fault Simulation

- Procedure

- ① Apply a test pattern, perform fault-free simulation
- ② A fault list is attached to each PI
 - Stuck-at value **opposite to its good value**
- ③ Fault lists propagated from PI to PO
 - Rule depends on **gate type** and **gate inputs** (see next slide)
- ④ A fault is **detected** if it is in fault list of a **PO**



Fault List Propagation Rules

	Gate	<i>a</i>	<i>b</i>	<i>c</i>	Output fault list
	NOT	0		1	$L_a \cup c/0$
		1		0	$L_a \cup c/1$
		0	0	0	$\{L_a \cap L_b\} \cup c/1$
		0	1	0	$\{L_a - L_b\} \cup c/1$
	AND	1	0	0	$\{L_b - L_a\} \cup c/1$
		1	1	1	$\{L_a \cup L_b\} \cup c/0$
		0	0	0	$L_c = L_a \cup c/1$
		0	0	0	$L_b = L_a \cup b/1$
	Fanout	1	1	1	$L_c = L_a \cup c/0$
		1	1	1	$L_b = L_a \cup b/0$

\cap = set intersection; \cup = set union ; $A-B$ = remove $(A \cap B)$ from A

Quiz

Q: Fault list propagation rule for OR gate ?

A:

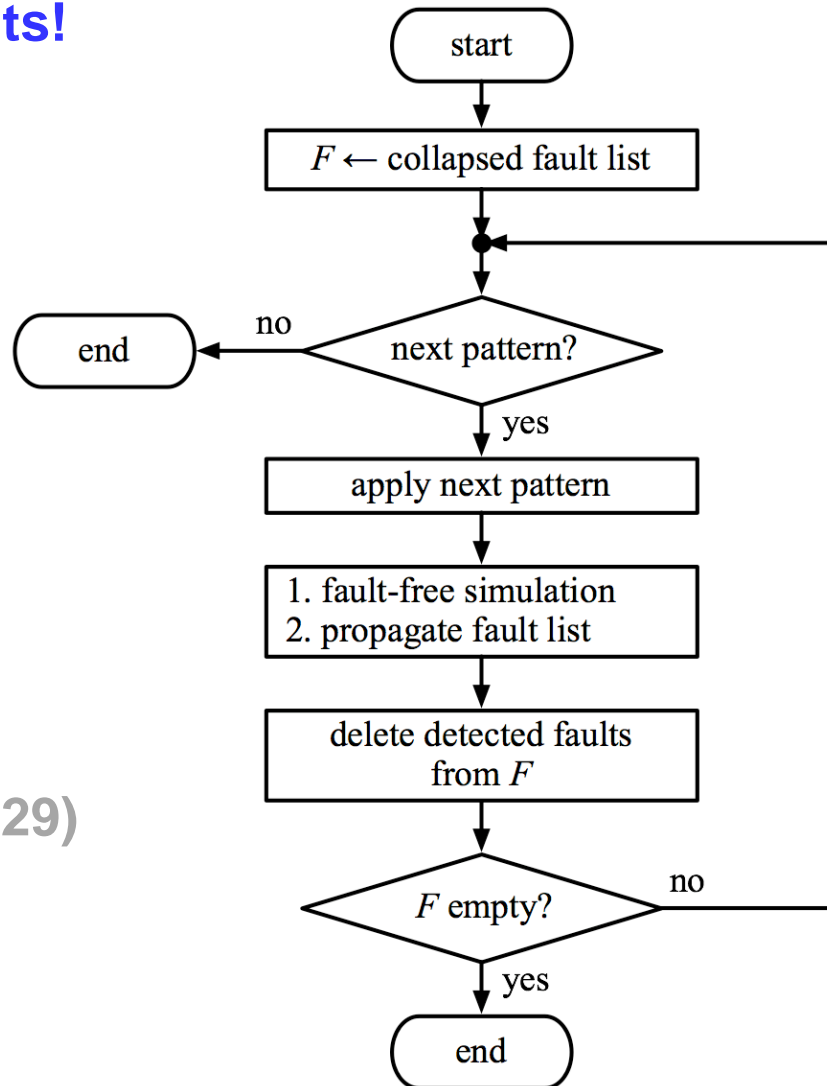


Gate	<i>a</i>	<i>b</i>	<i>c</i>	Output fault list, L_c
OR	0	0	0	
	0	1	1	
	1	0	1	
	1	1	1	

AND	0	0	0	$\{L_a \cap L_b\} \cup c/1$
	0	1	0	$\{L_a - L_b\} \cup c/1$
	1	0	0	$\{L_b - L_a\} \cup c/1$
	1	1	1	$\{L_a \cup L_b\} \cup c/0$

Deductive Fault Sim. Flow

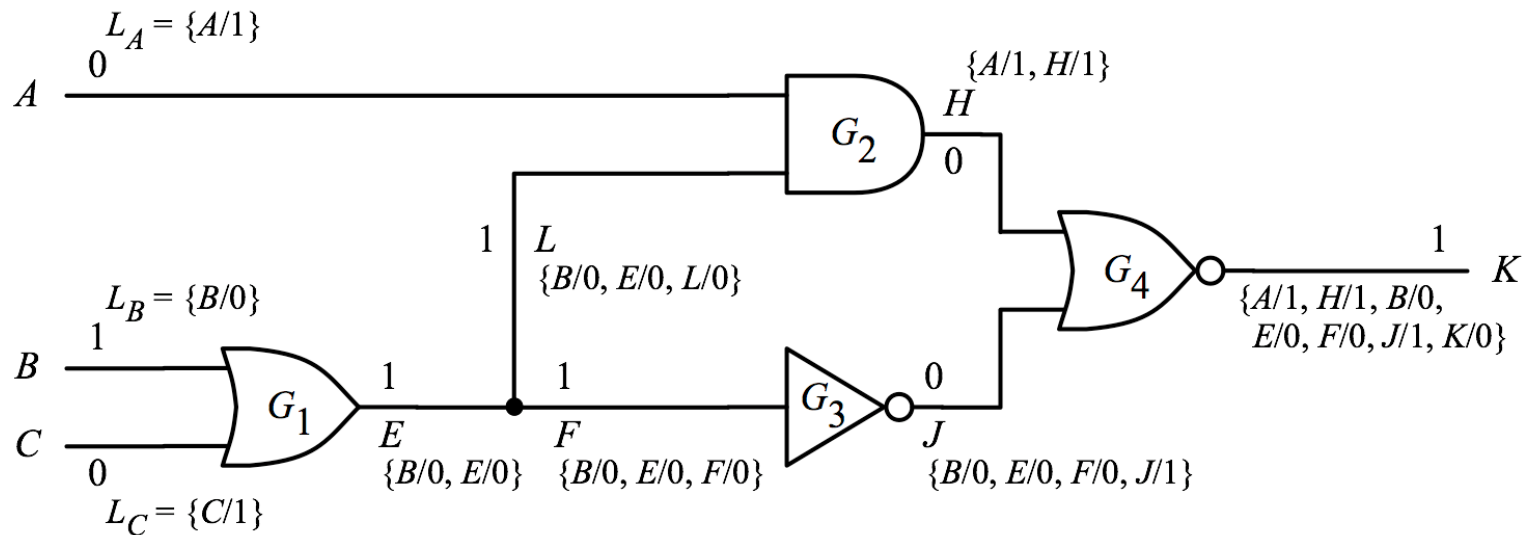
- Only loop of patterns
- No loop of faults!



(WWW Fig 3.29)

Example (P_1)

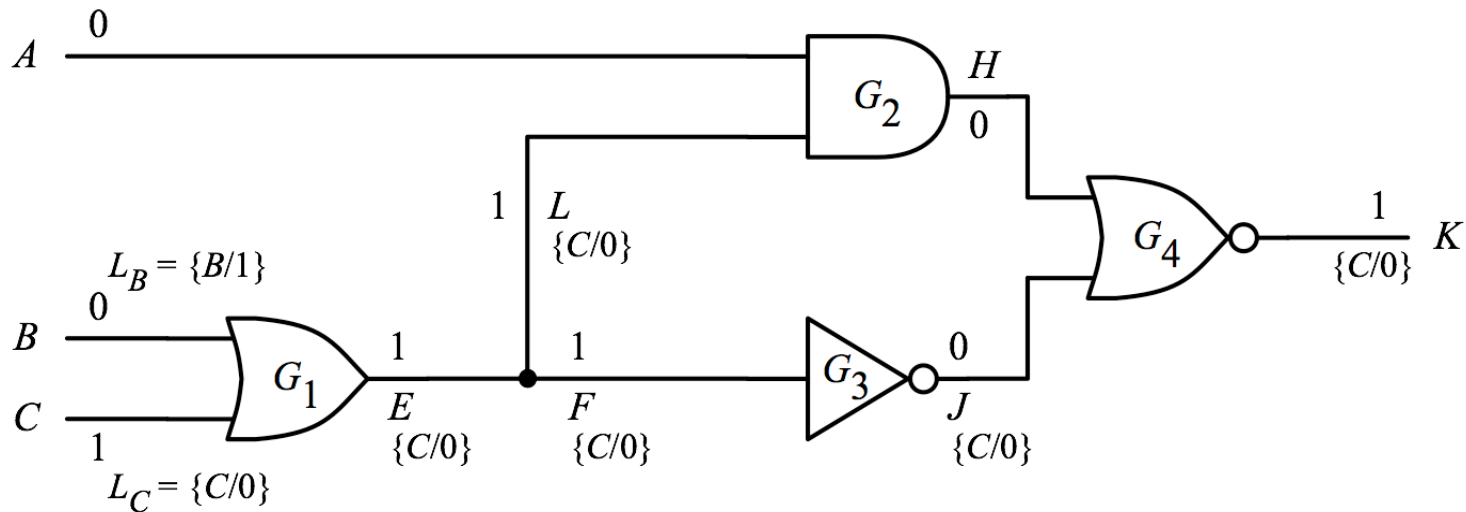
- Total **18** uncollapsed faults (for explanation purpose)
- **7** faults are detected by P_1
 - ♦ Detected faults are *dropped*
 - ♦ $18-7=11$ faults remains



(WWW Fig 3.26)

Example (P_2)

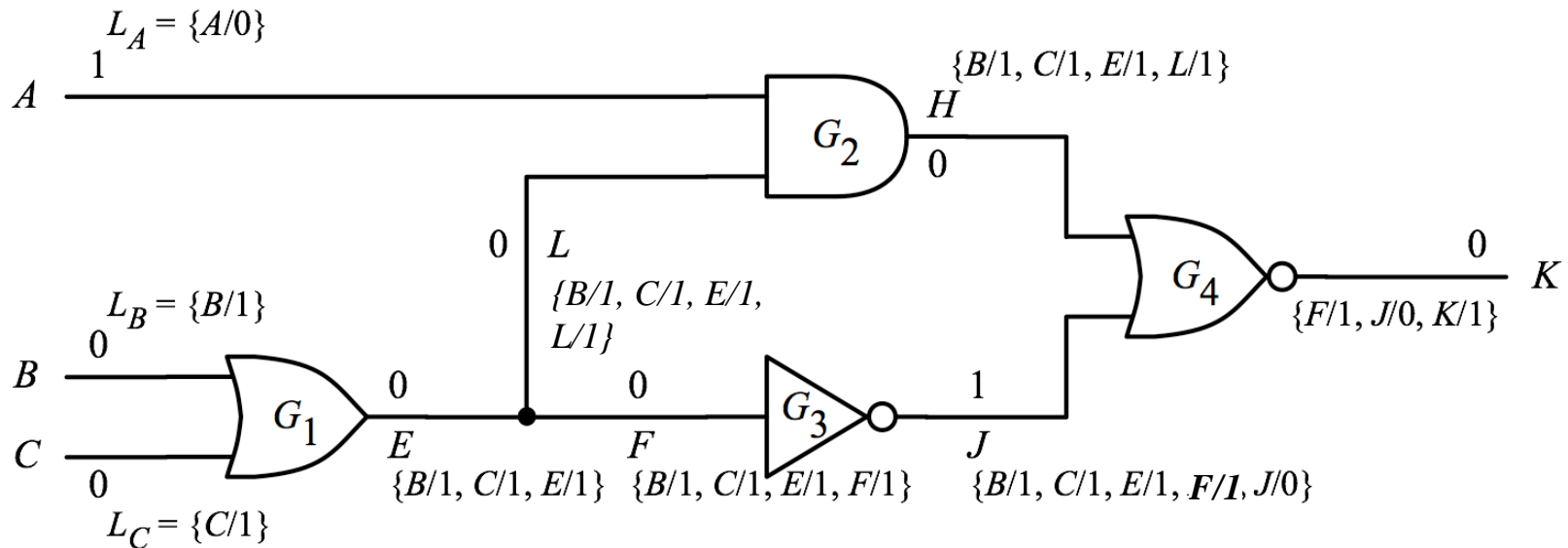
- **One** more fault is detected by P_2
 - ♦ Q: why no $A/1$?
- 11-1=10 faults remains



(WWW Fig 3.27)

Example (P_3)

- **Three** more faults are detected by P_3
- **10-3=7** faults remains undetected



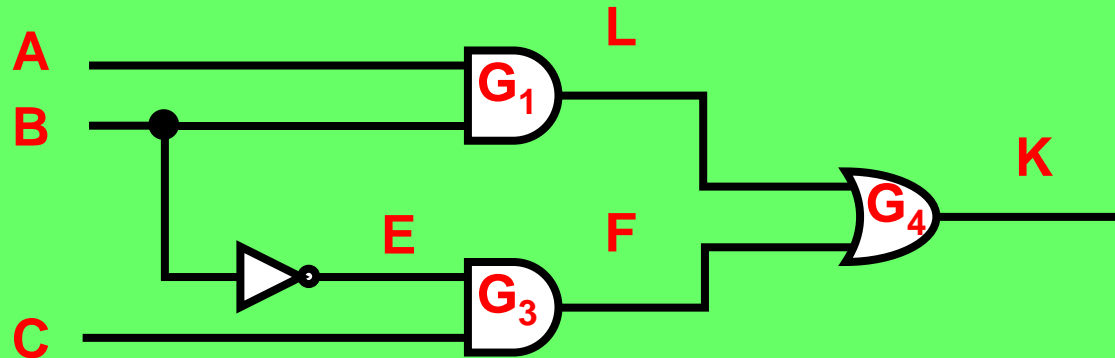
(WWW Fig 3.28)

Fault Coverage = 11/18 = 61%

Quiz

Q: Apply test pattern ABC=101.
Perform deductive fault simulation.
Consider only three faults {A/0, B/1, C/0}.
Which fault are detected?

A:



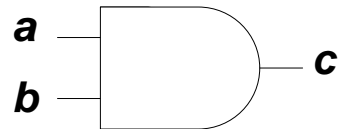
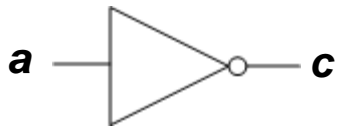
Summary

- Idea: Use logic deduction instead of simulation
- Advantages
 - ♦ No loop of faults
- Disadvantages
 - ① Problem with **multi-valued logic** (U, Z)
 - * Very complicated fault list propagation rules
 - ② Difficult **memory management**
 - * Memory requirement **not predictable**
 - * Many **\cap \cup operations** difficult for large designs
 - ③ Not suitable for **delay fault**

**Deductive Fault Sim. Interesting in Theory
But Not Useful in Practice**

FFT

- Q1: What **data structure** to implement fault list?
 - ♦ What is **time complexity** of set union/intersection?
- Q2: How to propagate fault list if there are **unknowns**?



Gate	<i>a</i>	<i>b</i>	<i>c</i>	Output fault list L_c
NOT	u		u	?
AND	0	u	0	?
	u	1	u	?
	u	u	u	?