Chapter 6 Testability Analysis

可測度分析法

Outline

- Introduction
- SCOAP
- COP
- High-level Testability

Testability Analysis

- Applications
  - To give early warnings about the test problems
    - Guide the selection of test points to improve testability.
    - Automate the “Design for Testability” problem
  - To provide guidance in ATPG
    - For example, determine the “hardest” & “easiest” inputs in *backtrace* of PODEM
- Complexity should be simpler than ATPG and fault simulation
    - Need to be linear or almost linear in terms of circuit size
- Topology analysis
  - Only the structure of the circuit is analyzed
  - No test vectors are involved
  - Only an approximation
    - reconvergent fanouts cause inaccuracy

Testability Measures

- Controllability
  - The difficulty of setting a particular logic signal to 0 or 1.
- Observability
  - The difficulty of observing the logic state of a signal.

SCOAP

Sandia Controllability/Observability Analysis Program

Goldstein, DAC 1980

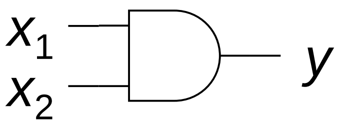
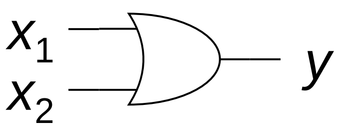
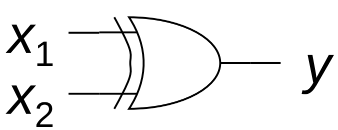
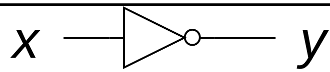
- SCOAP computes 6 numbers for each node *N*.

	0-controllability	1-controllability	Observability
Combinational	$CC^0(N)$	$CC^1(N)$	$CO(N)$
Sequential	$SC^0(N)$	$SC^1(N)$	$SO(N)$

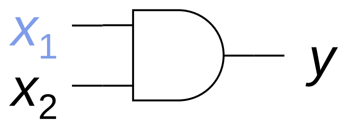
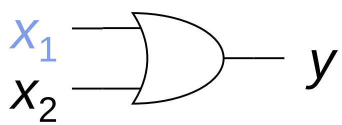
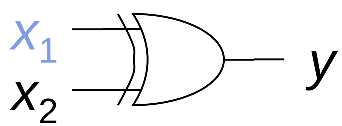
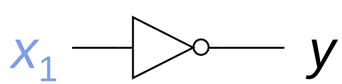
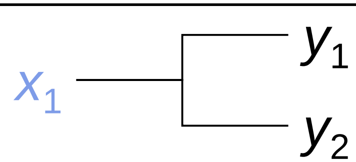
Combinational SCOAP Measures

- Combinational controllability
  - $CC^0(N)$, $CC^1(N)$
  - Related to the minimum number of combinational node (PI or gate output) assignments required to justify a 0 or 1 on a node *N*.
- Combinational observability
  - $CO(N)$
  - Related to the number of gates between *N* and PO's, ***and***
  - the minimum number of PI assignments required to propagate the logical value on node *N* to a primary output.

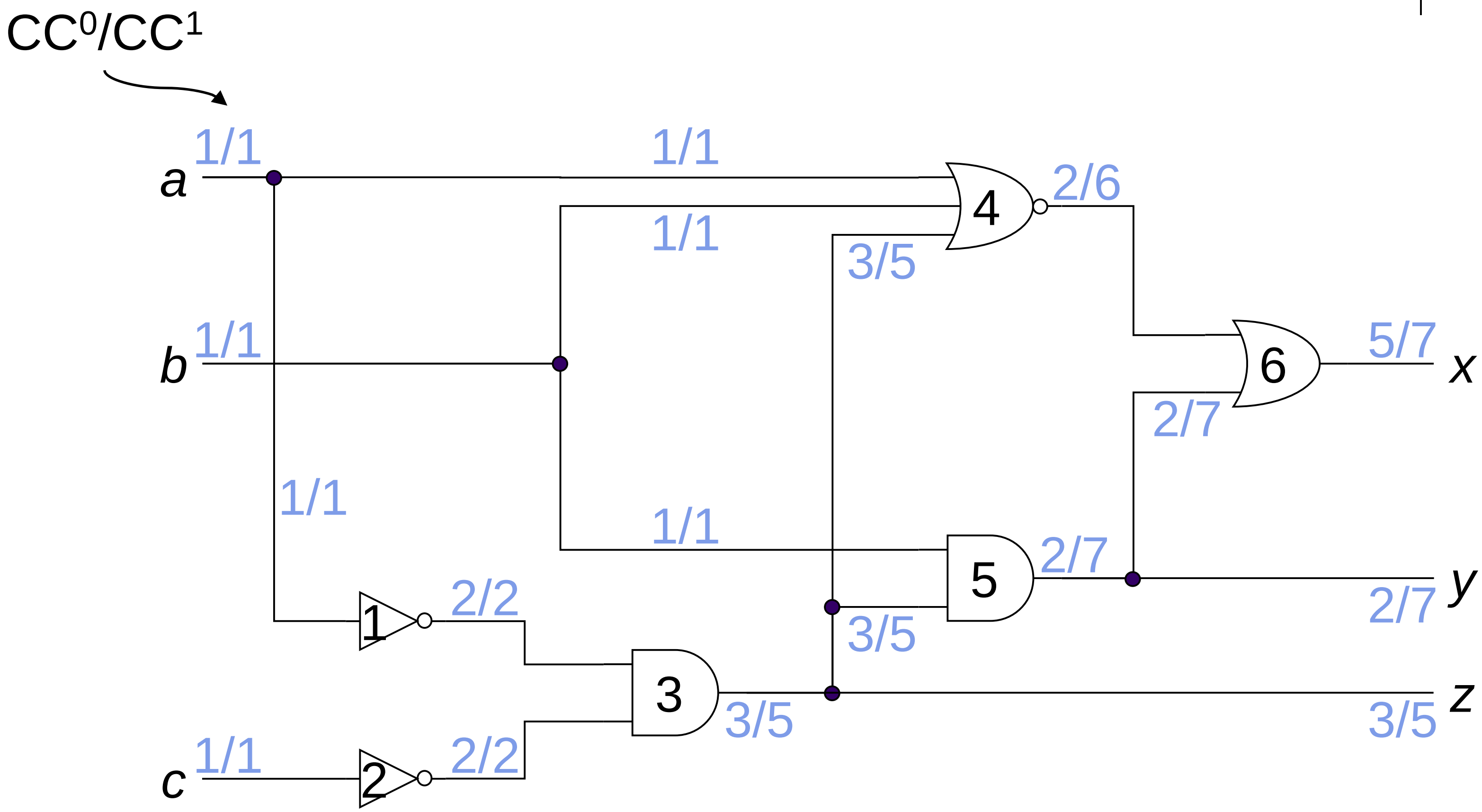
$CC^0(N)$ & $CC^1(N)$

	$CC^0(y)$	$CC^1(y)$
	$\min[CC^0(x_1), CC^0(x_2)] + 1$	$CC^1(x_1) + CC^1(x_2) + 1$
	$CC^0(x_1) + CC^0(x_2) + 1$	$\min[CC^1(x_1), CC^1(x_2)] + 1$
	$\min[CC^0(x_1) + CC^0(x_2), CC^1(x_1) + CC^1(x_2)] + 1$	$\min[CC^0(x_1) + CC^1(x_2), CC^1(x_1) + CC^0(x_2)] + 1$
	$CC^1(x) + 1$	$CC^0(x) + 1$
Primary inputs	1	1

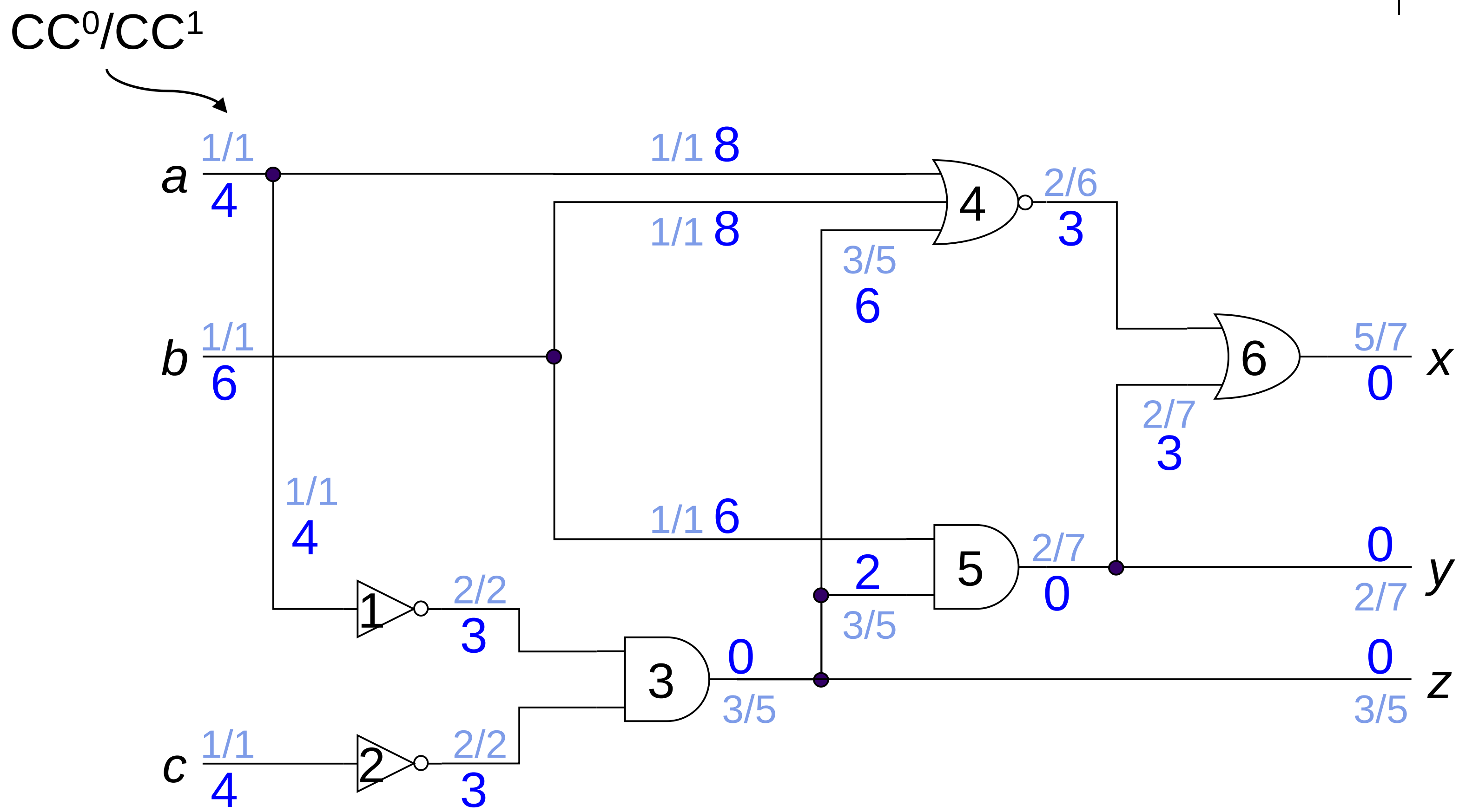
CO(N)

	$CO(x_1)$
	$CO(y) + CC^1(x_2) + 1$
	$CO(y) + CC^0(x_2) + 1$
	$CO(y) + \min[CC^0(x_2) ,CC^1(x_2)] + 1$
	$CO(y) + 1$
	$\min[CO(y_1),CO(y_2)]$
Primary outputs	0

An Example – Controllability



An Example – Observability



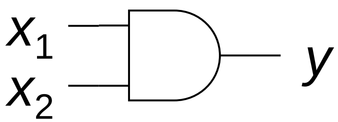
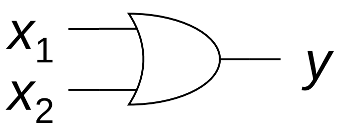
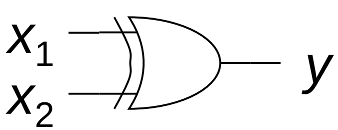
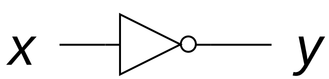
Sequential SCOAP Measures

- Sequential controllability
  - $SC^0(N)$, $SC^1(N)$
  - Estimate the minimum number of sequential node (FF output) assignments required to justify a 0 or 1 on a node *N*.
- Sequential observability
  - $SO(N)$
  - Related to the number of FF's between *N* and PO's, ***and***
  - the minimum number of FF assignments required to propagate the logical value on node *N* to a primary output.

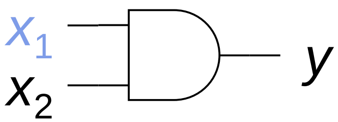
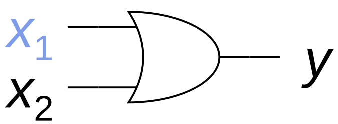
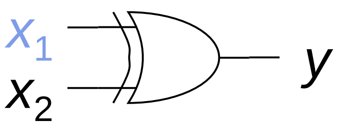
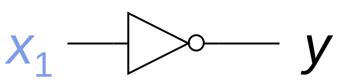
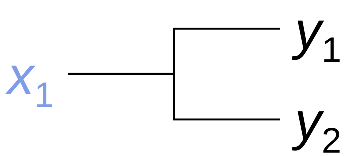
Computing the Sequential SCOAP Measures

- **Computation of $SC^0(N)$, $SC^1(N)$, and $SO(N)$ is similar to that of $CC^0(N)$, $CC^1(N)$, and $CO(N)$.**
- **The differences are**
  - One increments the sequential measures by 1 only when signals propagate from FF inputs to Q or Q', or backwards.
  - Several iterations may be required for the controllability numbers to converge.

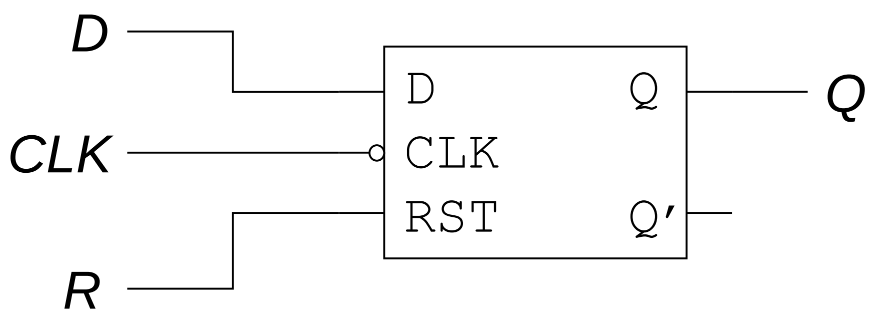
Computing $SC^0(N)$ and $SC^1(N)$

	$SC^0(y)$	$SC^1(y)$
	$\min[SC^0(x_1), SC^0(x_2)]$	$SC^1(x_1) + SC^1(x_2)$
	$SC^0(x_1) + SC^0(x_2)$	$\min[SC^1(x_1), SC^1(x_2)]$
	$\min[SC^0(x_1) + SC^0(x_2),$ $SC^1(x_1) + SC^1(x_2)]$	$\min[SC^0(x_1) + SC^1(x_2),$ $SC^1(x_1) + SC^0(x_2)]$
	$SC^1(x)$	$SC^0(x)$
Primary inputs	0	0

SO(N)

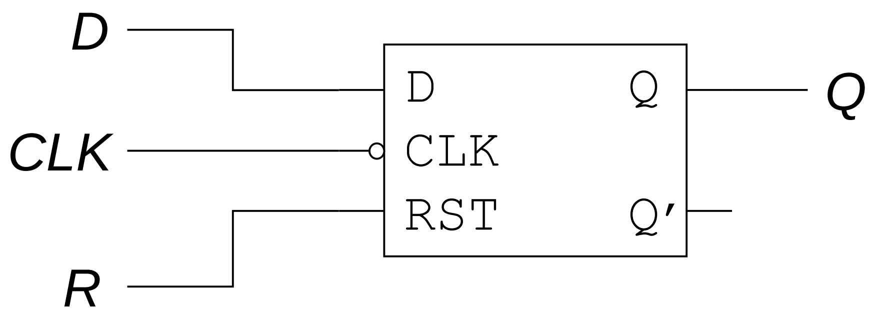
<table>
<tr><th></th><th>SO(x_1)</th></tr>
<tr><td></td><td>SO(y) + SC^1(x_2)</td></tr>
<tr><td></td><td>SO(y) + SC^0(x_2)</td></tr>
<tr><td></td><td>SO(y) + min[SC^0(x_2) ,SC^1(x_2)]</td></tr>
<tr><td></td><td>SO(y)</td></tr>
<tr><td></td><td>min[SO(y_1),SO(y_2)]</td></tr>
<tr><td>Primary outputs</td><td>0</td></tr>
</table>

Flip-Flop



$$CC^1(Q) = CC^1(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R)$$
$$SC^1(Q) = SC^1(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1$$

$$CC^0(Q) = \min[CC^1(R) + CC^0(CLK),\ CC^0(D) + CC^1(CLK) + CC^0(CLK) + CC^0(R)]$$
$$SC^0(Q) = \min[SC^1(R) + SC^0(CLK),\ SC^0(D) + SC^1(CLK) + SC^0(CLK) + SC^0(R)] + 1$$

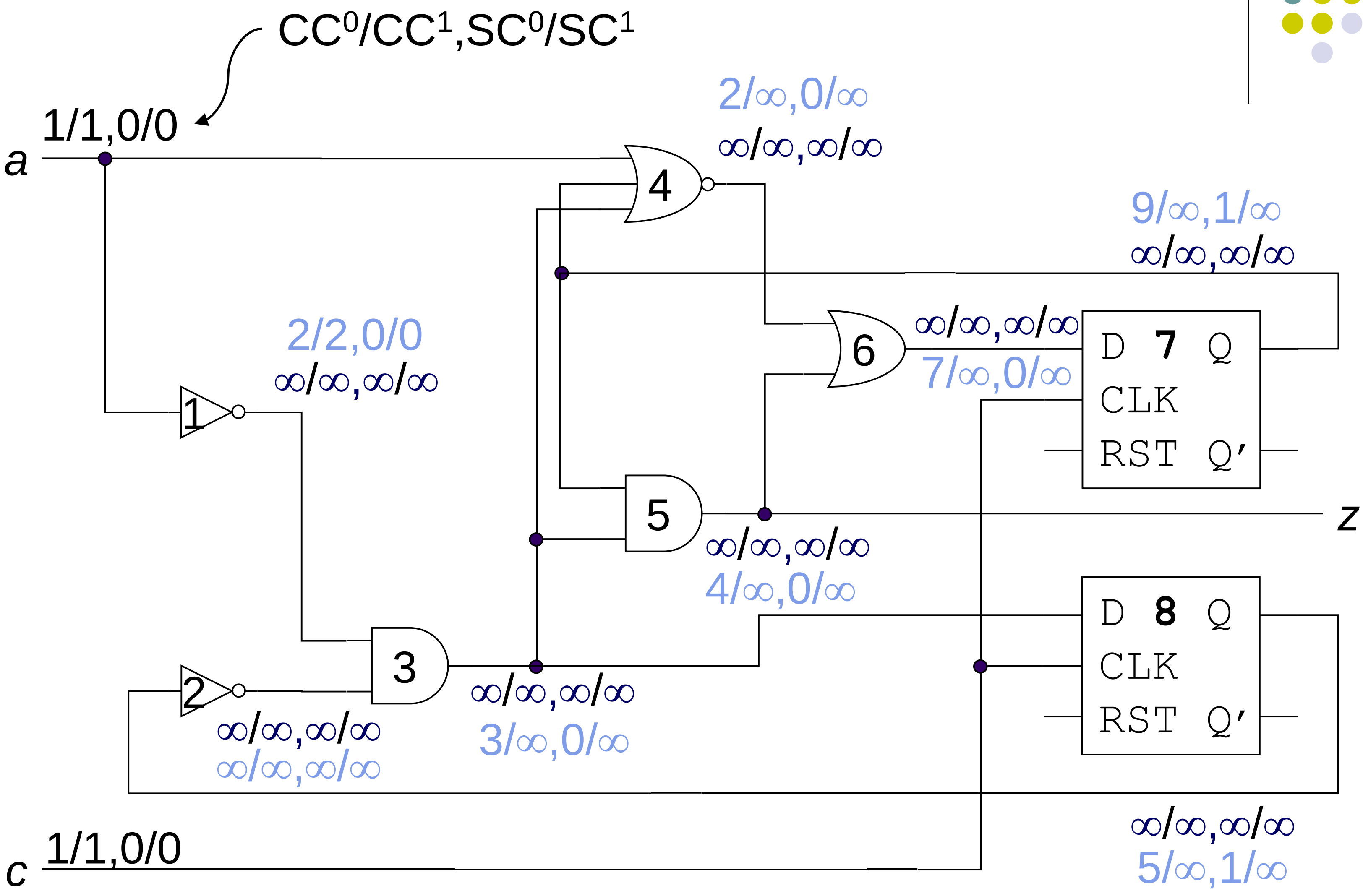


$$CO(D) = CO(Q) + CC^1(CLK) + CC^0(CLK) + CC^0(R)$$
$$SO(D) = SO(Q) + SC^1(CLK) + SC^0(CLK) + SC^0(R) + 1$$

Computing Testability Measures for Sequential Circuits

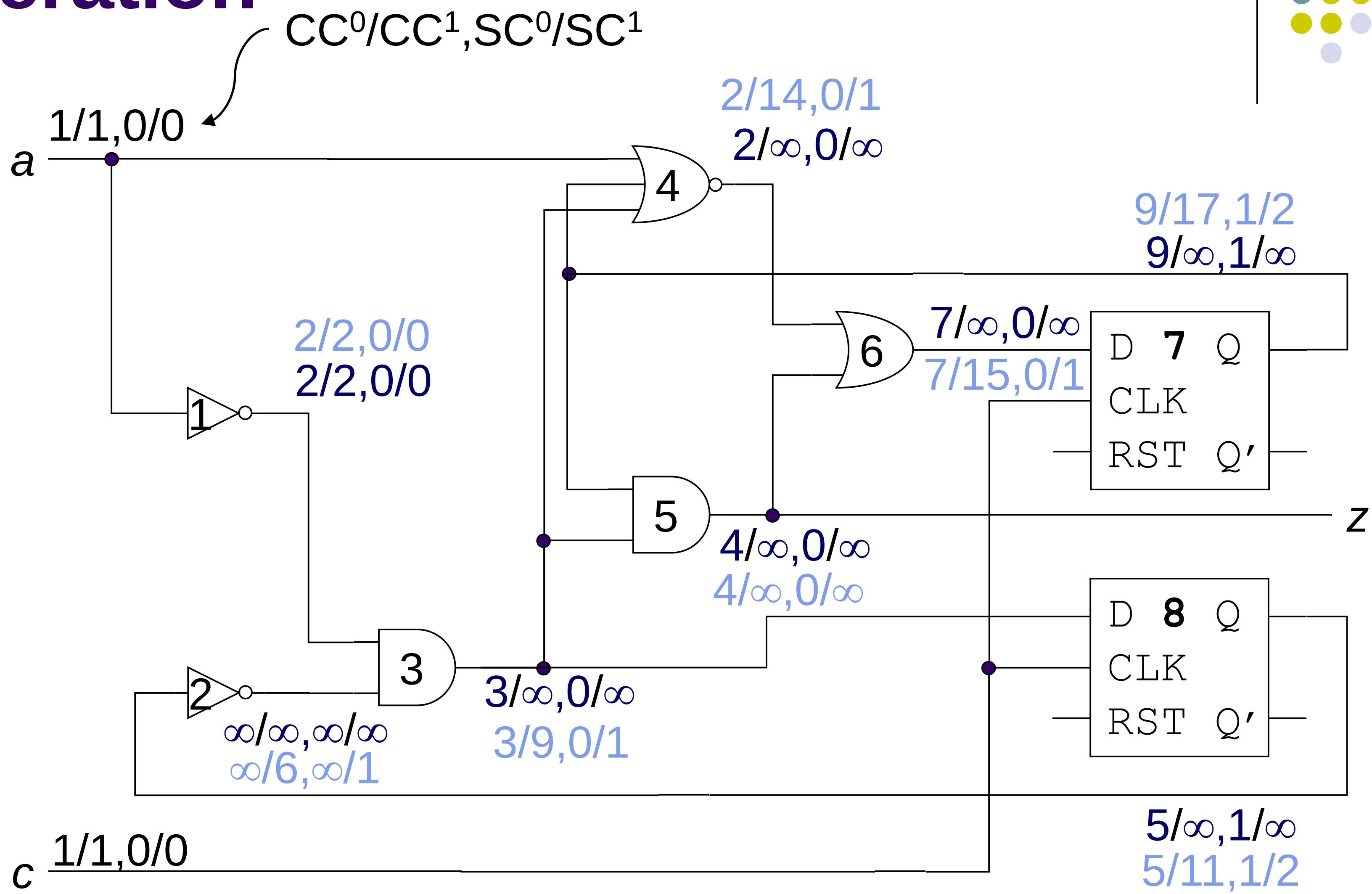
1. For all PI's, set $CC^0 = CC^1 = 1$ and $SC^0 = SC^1 = 0$.
2. For all other nodes, set $CC^0 = CC^1 = \infty$ and $SC^0 = SC^1 = \infty$.
3. Propagate controllability measures from PI's to PO's.
   Iterate until the controllability numbers stabilize.
4. For all PO's, set CO = SO = 0.
5. For all other nodes, set CO = SO = ∞.
6. Propagate observability from PO's to PI's.

Controllability Computation

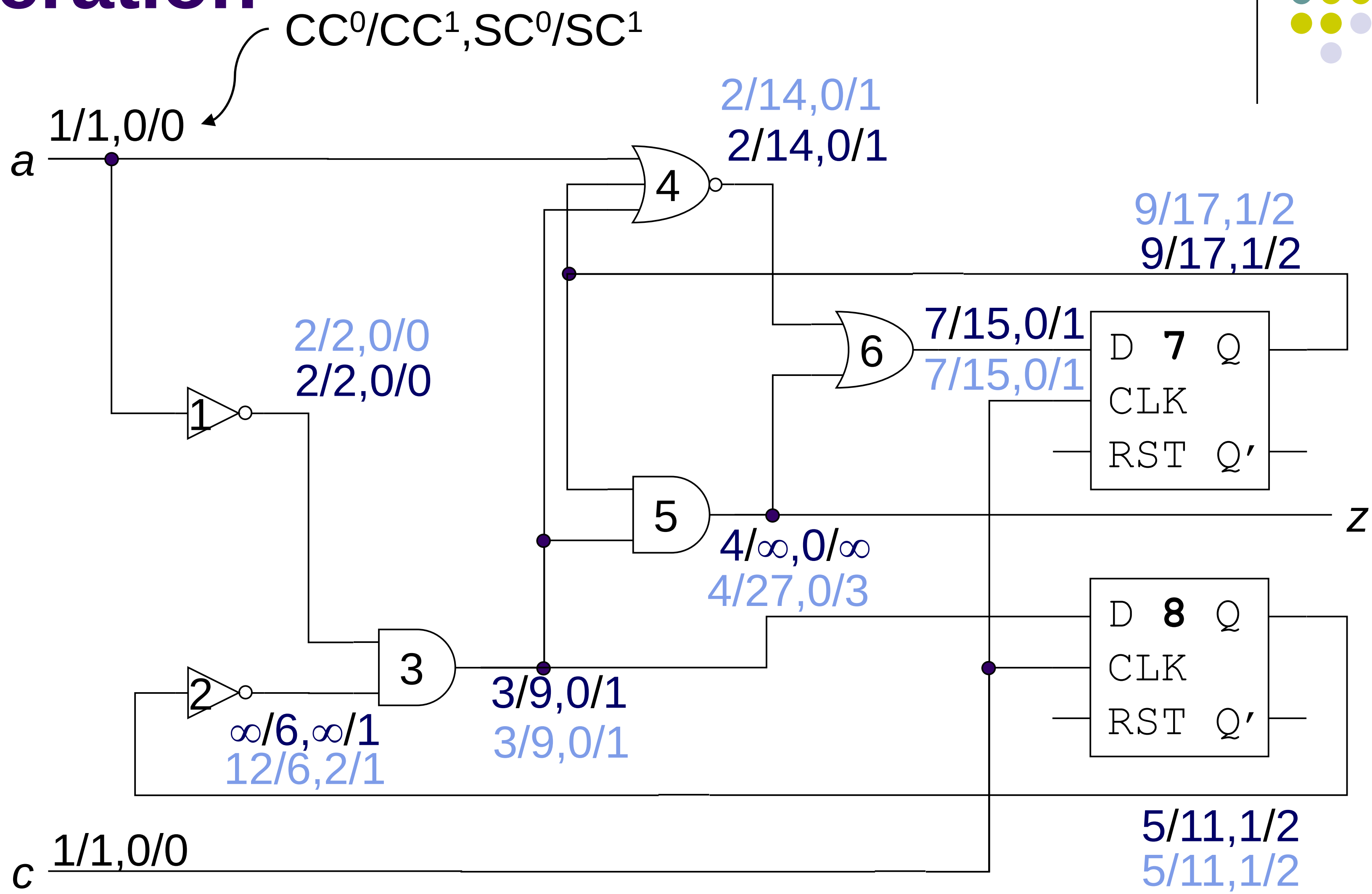


Assuming no RST can occur

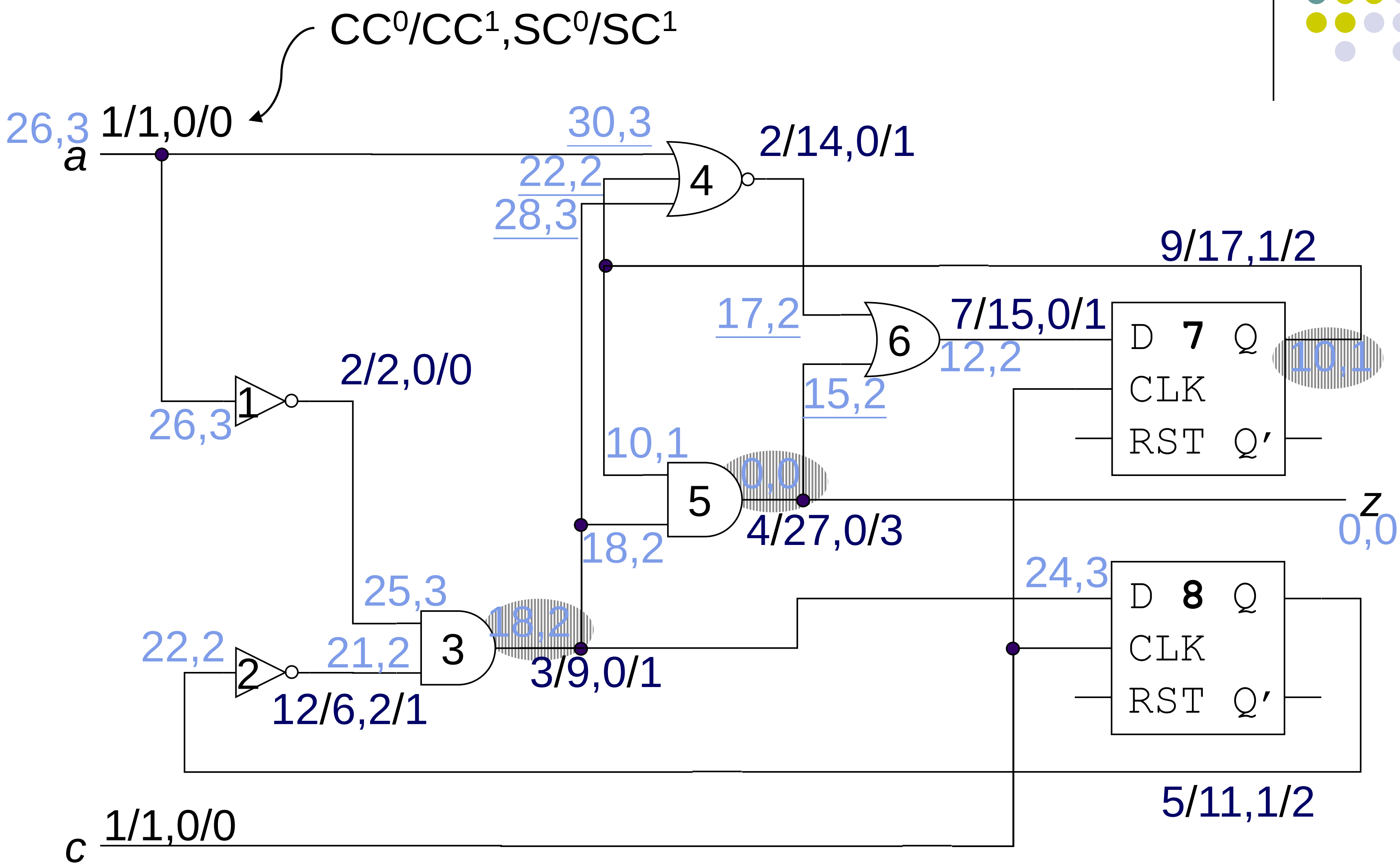
Controllability Computation – 2nd Iteration



Controllability Computation – 3rd iteration

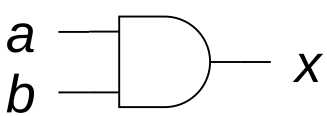
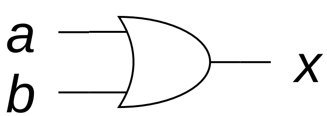
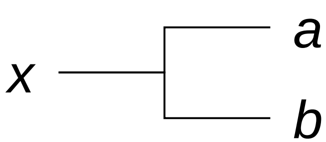


Observability Computation



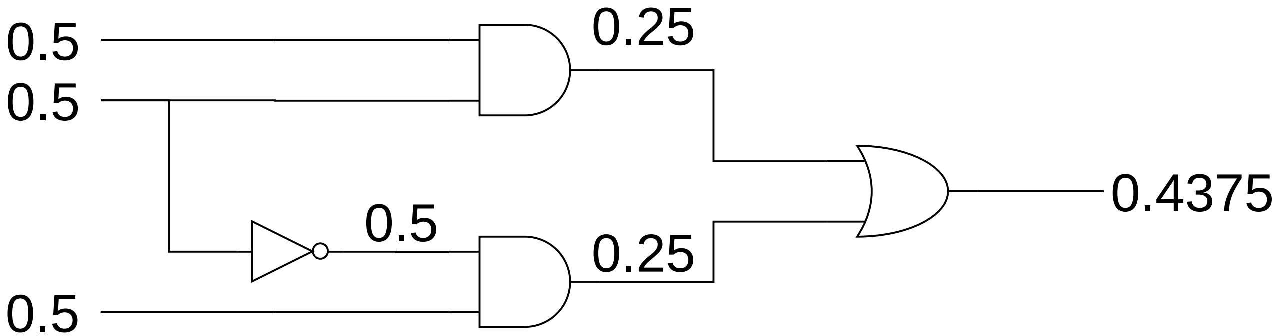
COP *[F. Brglez, '84]*

- C_x: the probability of *x* being 1.
- O_x: the probability of *x* being observed at a PO.

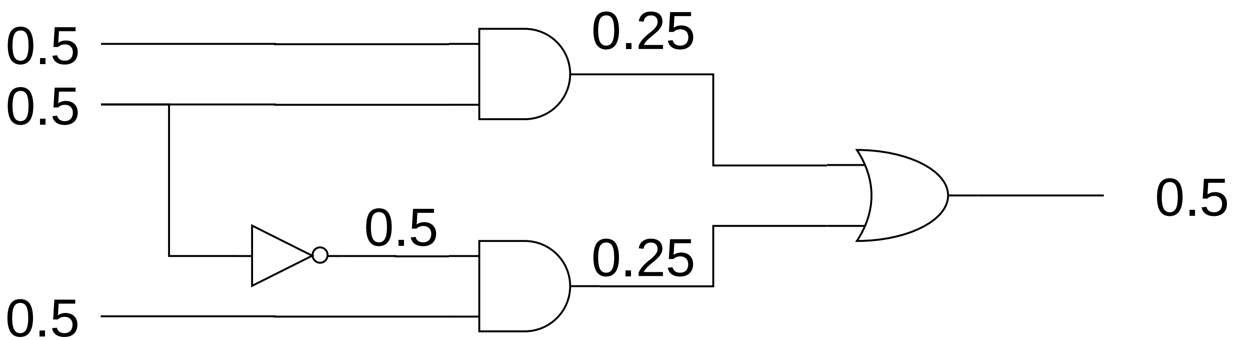
	C_x	O_a
	$C_x = C_a \times C_b$	$O_a = O_x \times C_b$
	$C_x = 1 - (1- C_a) \times (1- C_b)$	$O_a = O_x \times (1- C_b)$
	$C_x = C_a = C_b$	$O_x = 1 - (1- O_a) \times (1- O_b)$

An Example – Controllability

COP values

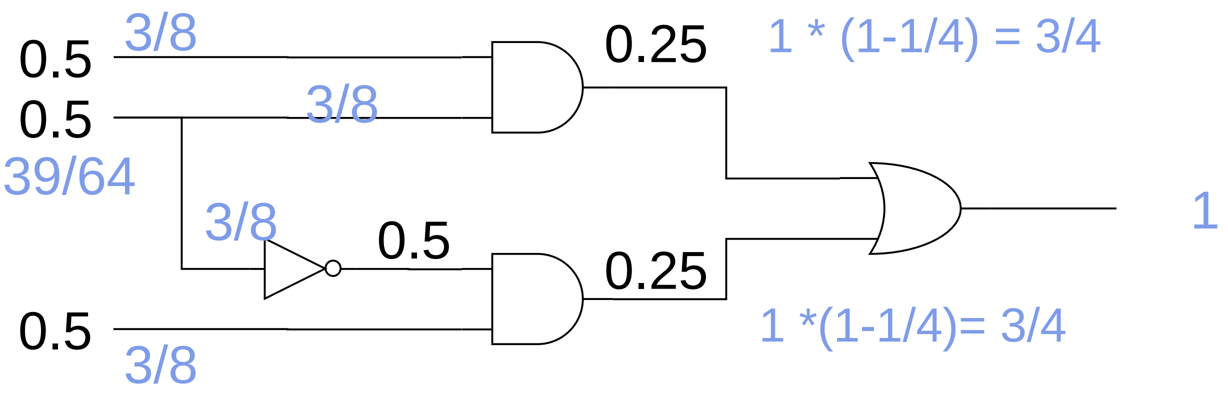


Actual contrallabilities

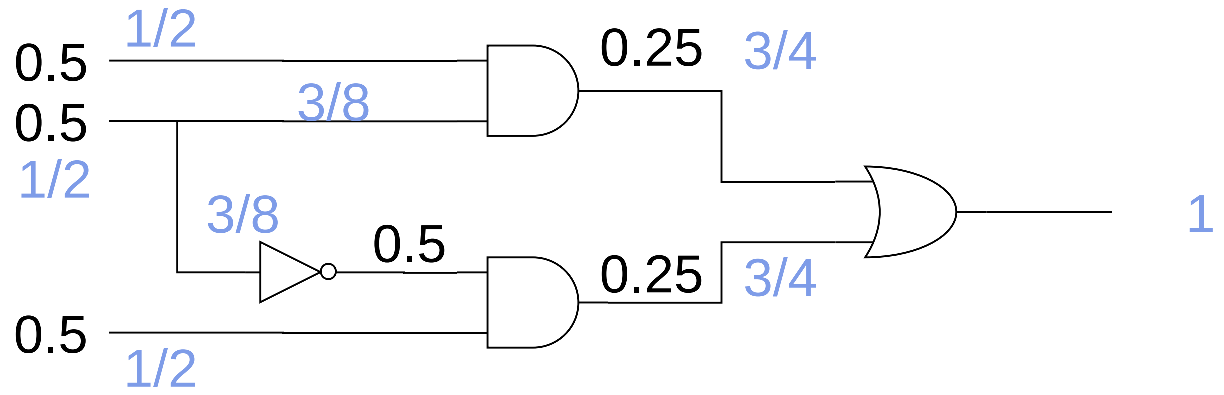


An Example – Observability

COP values



Actual observabilities



PODEM: Example (1/4)

Input: $C_A = 0.5$, $C_B = 0.5$, $C_C = 0.5$ → $C_d = 1 - 0.5 = 0.5$, $C_e = 1 - 0.5 = 0.5$

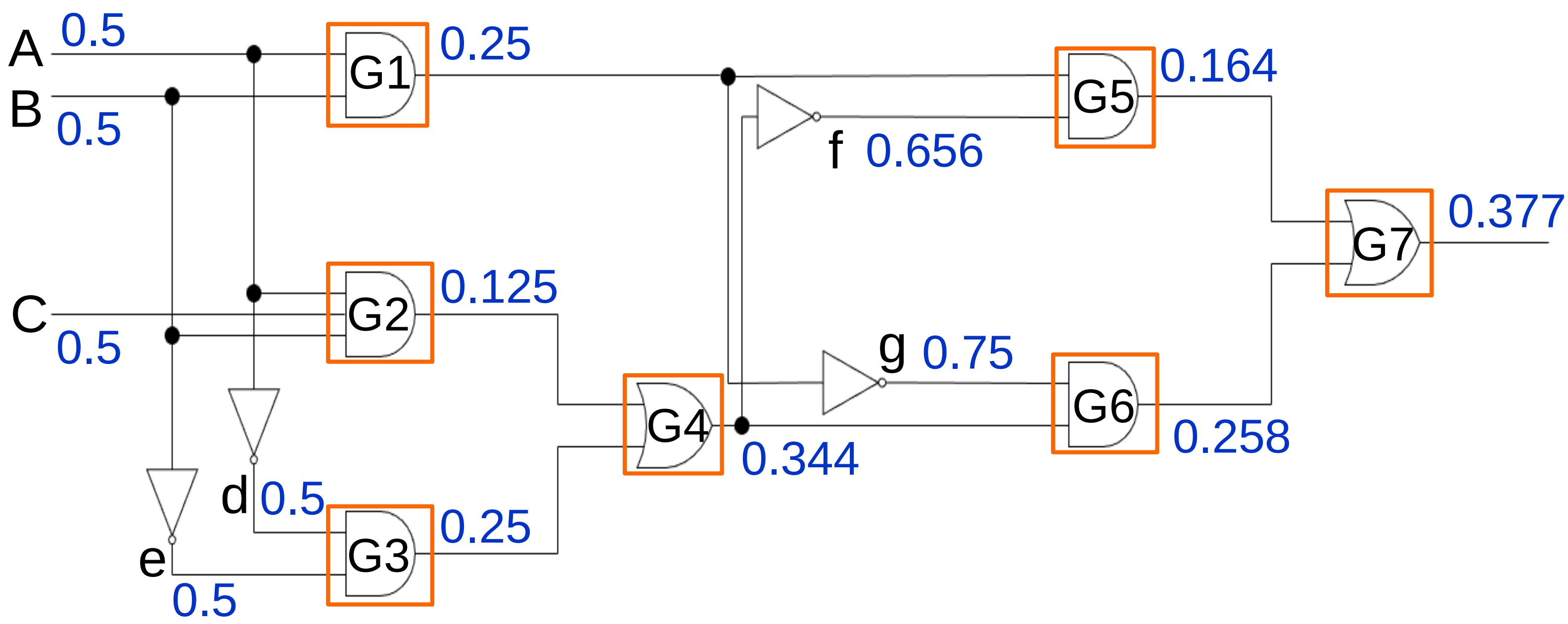
→ $C_{G1} = 0.5 * 0.5 = 0.25$, $C_{G2} = 0.5 * 0.5 * 0.5 = 0.125$, $C_{G3} = 0.5 * 0.5 = 0.25$

→ $C_{G4} = 1 – (1 – 0.125) * (1 - 0.25) = 0.344$

→ $C_f = 1 – 0.344 = 0.656$, $C_g = 1 - 0.25 = 0.75$

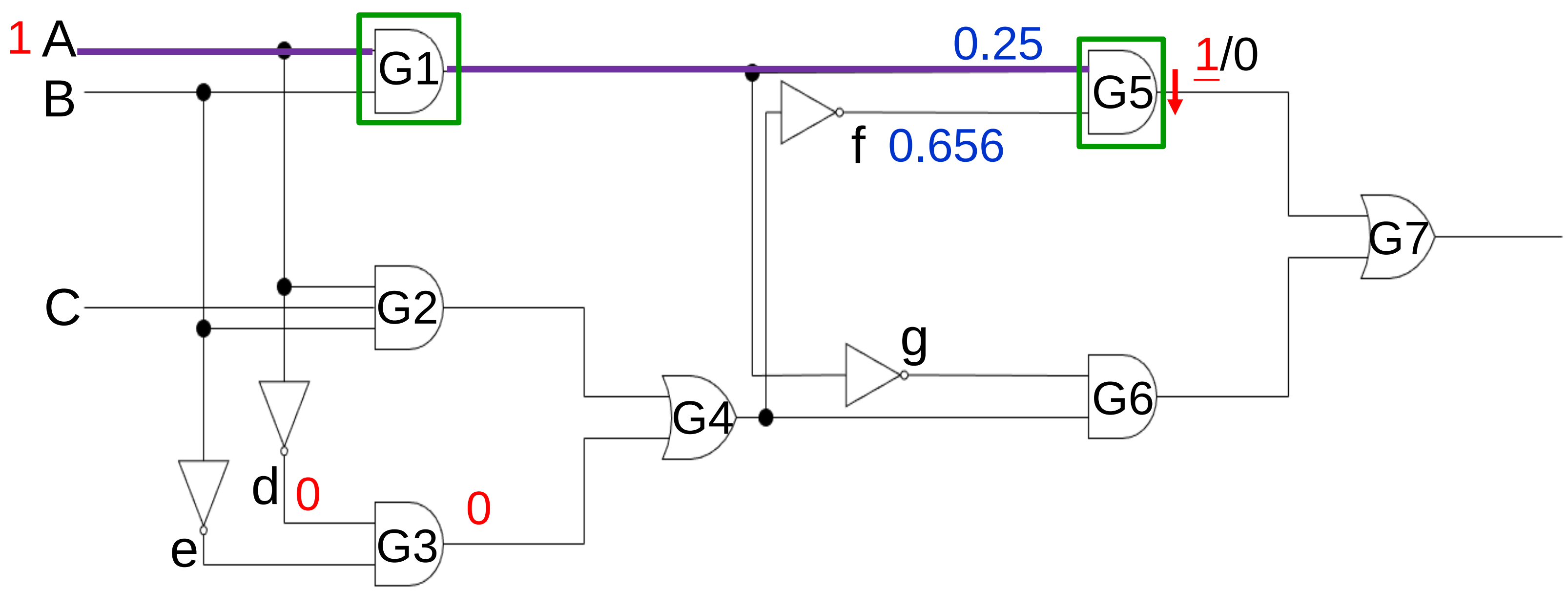
→ $C_{G5} = 0.25 * 0.656 = 0.164$, $C_{G6} = 0.75 * 0.344 = 0.258$

→ $C_{G7} = 1 – (1 – 0.164) * (1 – 0.258) = 0.377$



PODEM: Example (2/4)

Initial objective=(G5,1).
G5 is an AND gate → Choose the hardest-1
→ Back-trace to (G1,1).
G1 is an AND gate → Choose the hardest-1
→ Arbitrarily, back-trace to (A,1). A is a PI → Implication → G3=0.



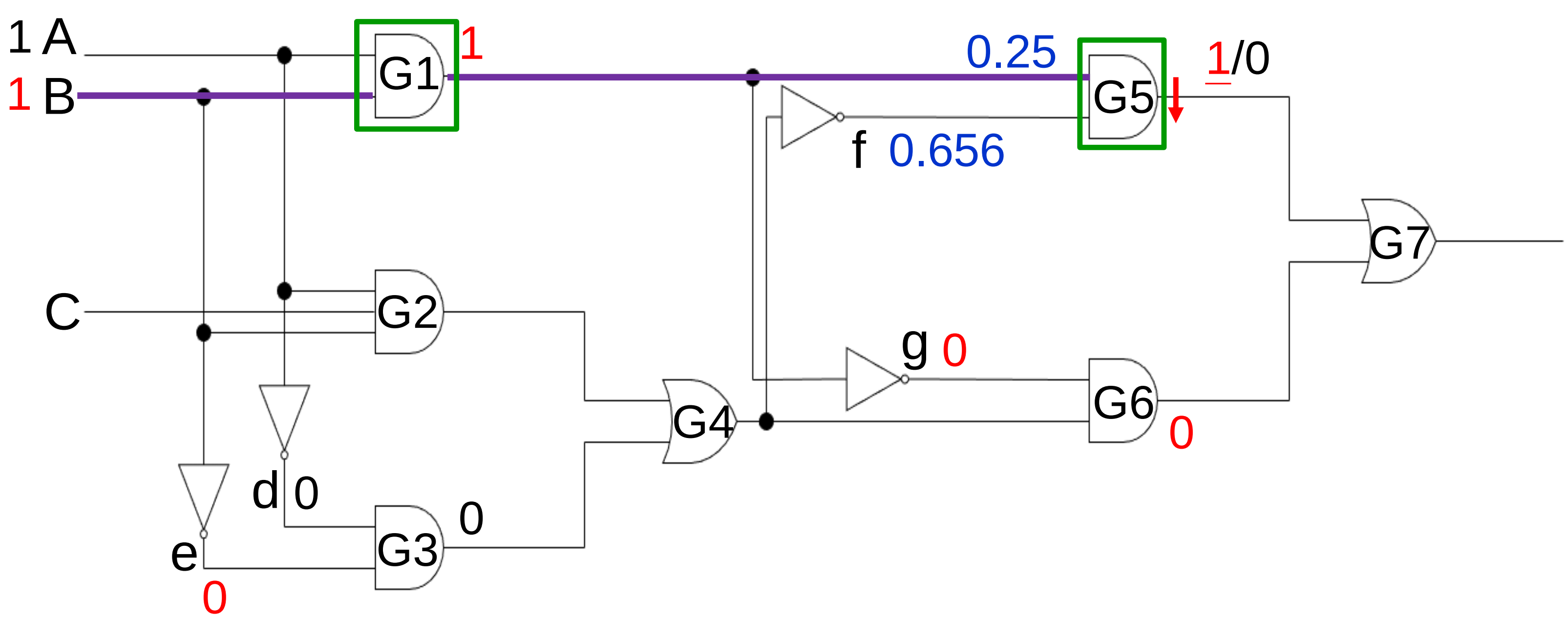
PODEM: Example (3/4)

The initial objective satisfied? No! → Current objective=(G5,1).

G5 is an AND gate → Choose the hardest-1 → Back-trace to (G1,1).

G1 is an AND gate → Choose the hardest-1

→ Arbitrarily, back-trace to (B,1). B is a PI → Implication → G1=1, G6=0.



PODEM: Example (4/4)

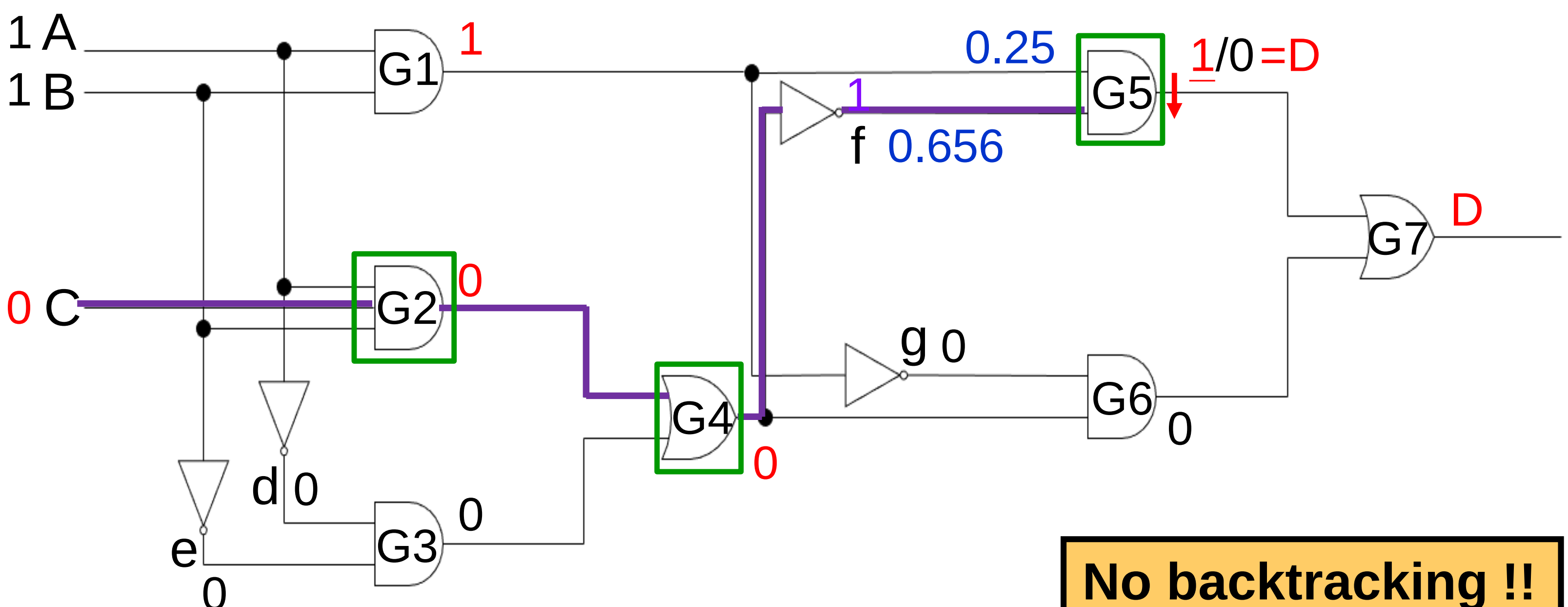
The initial objective satisfied? No! → Current objective=(G5,1).

The value of G1 is known → Back-trace to (G4,0).

The value of G3 is known → Back-trace to (G2,0).

A, B is known → Back-trace to (C,0).

C is a PI → Implication → G2=0, G4=0, G5=D, G7=D.



No backtracking !!

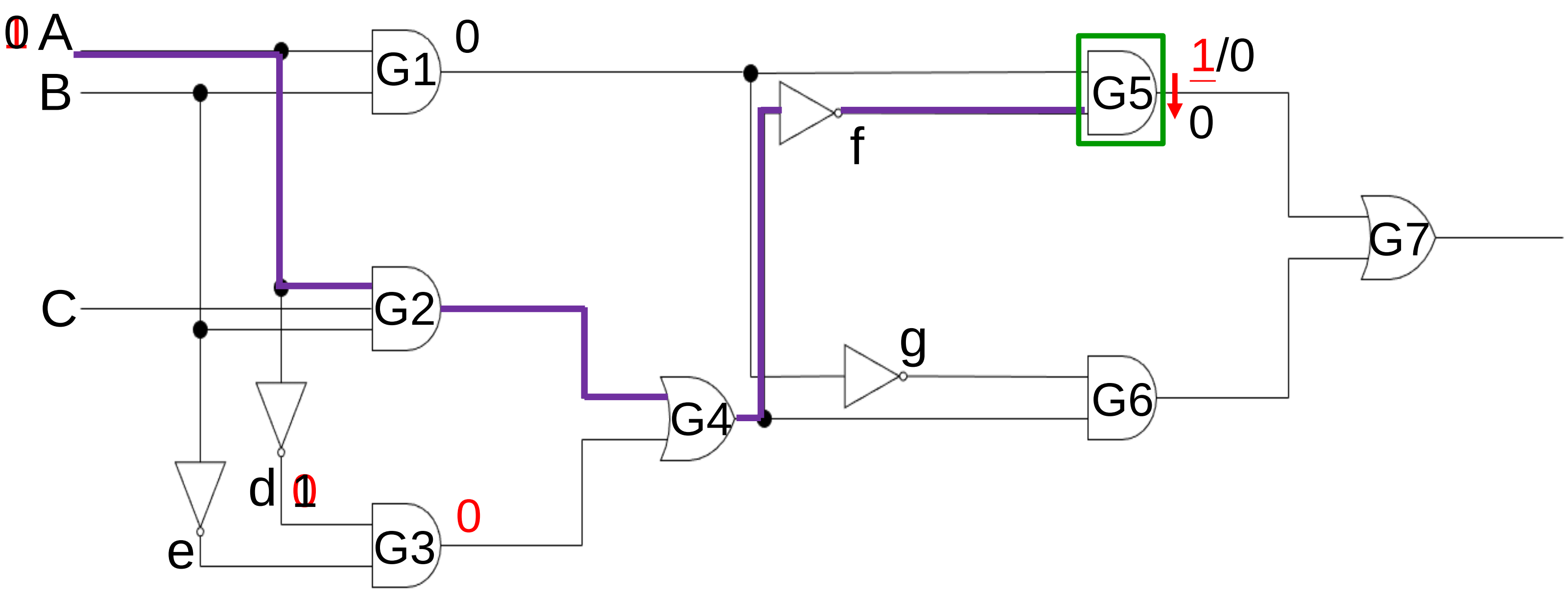
If The Backtracing Is Not Guided (1/3)

Initial objective=(G5,1).

Choose path G5-G4-G2-A → A=0.

Implication for A=0 → G1=0, G5=0 → Backtracking to A=1.

Implication for A=1 → G3=0.



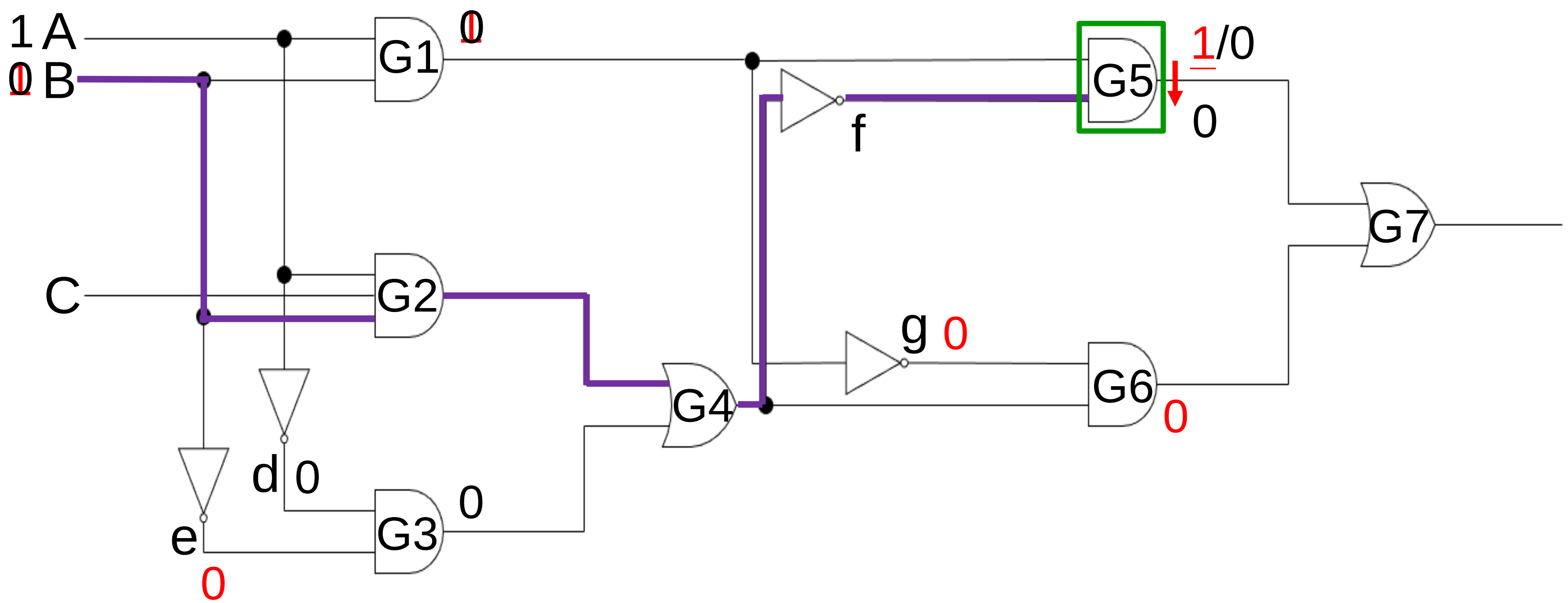
If The Backtracing Is Not Guided (2/3)

The initial objective satisfied? No! → Current objective=(G5,1).

Choose path G5-G4-G2-B → B=0.

Implication for B=0 → G1=0, G5=0 → Backtracking to B=1.

Implication for B=1 → G1=1, G6=0.

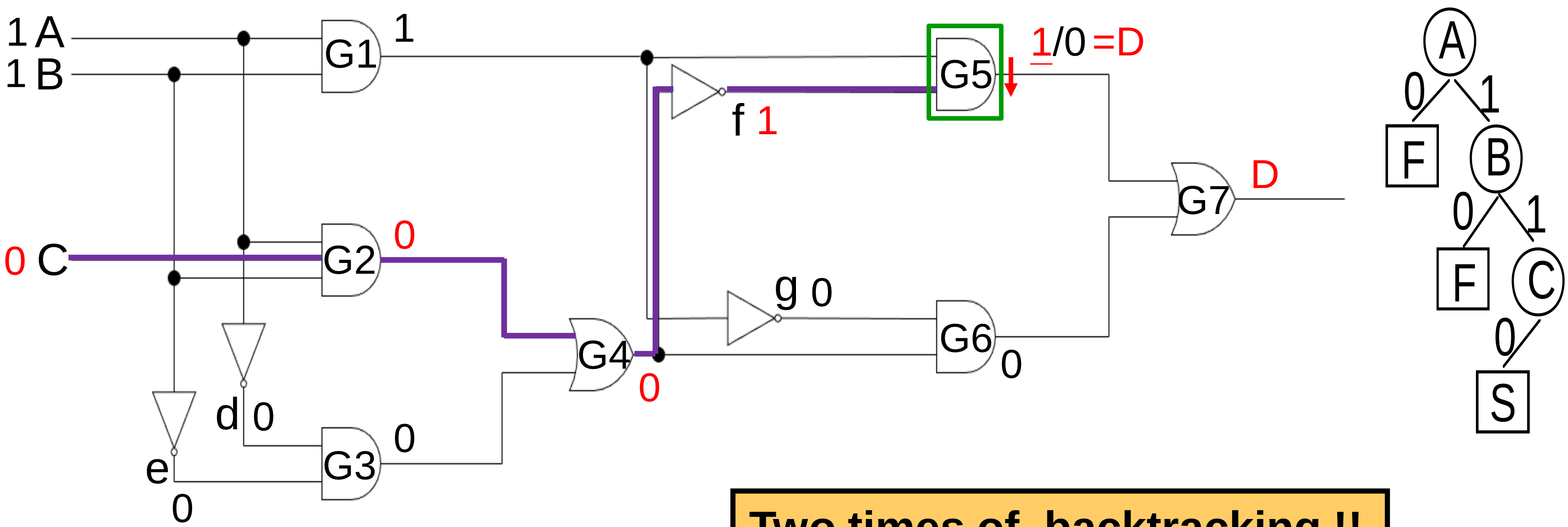


If The Backtracing Is Not Guided (3/3)

The initial objective satisfied? No! → Current objective=(G5,1).

Choose path G5-G4-G2-C → C=0.

Implication for C=0 → G2=0, G4=0, G5=D, G7=D.



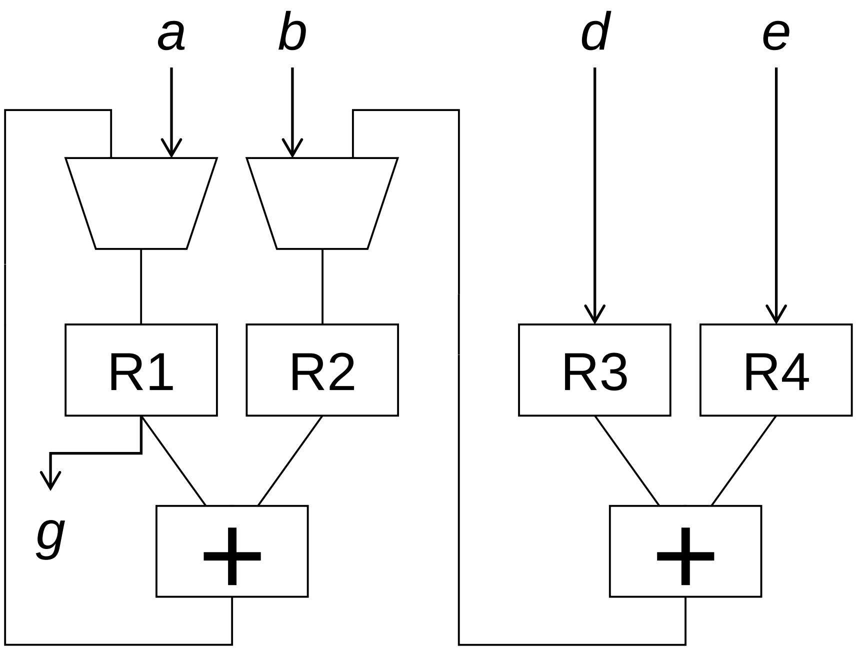
Two times of backtracking !!

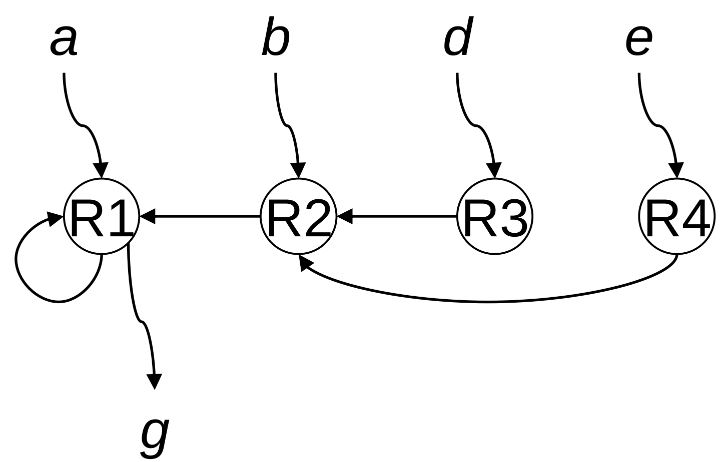
High-Level Testability Analysis

- **Based on behavioral level circuit model.**
- **Usually part of the behavior synthesis program.**
- **To improve the testability at earlier design stage.**

Data Flow Graph (DFG)

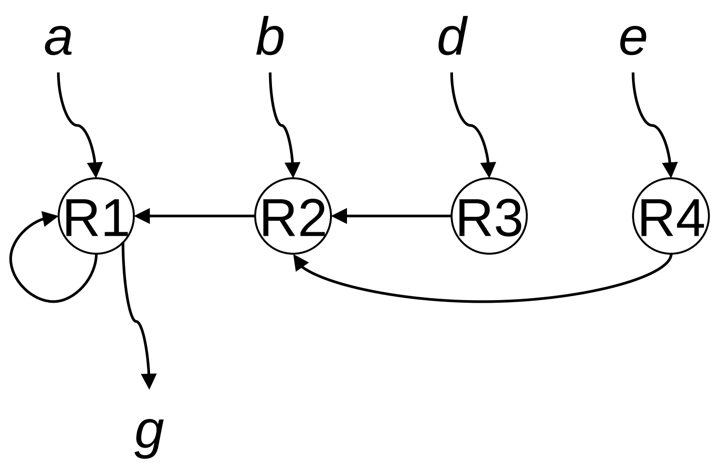
- **Each node corresponds to a register.**
- **Each arc represents a combinational path between two registers.**





A High-Level Testability Measure – Sequential Depth

- **The length of a sequential path between two nodes is the number of arcs along the path.**
- **The sequential depth between a pair of registers is the length of the shortest path between them.**

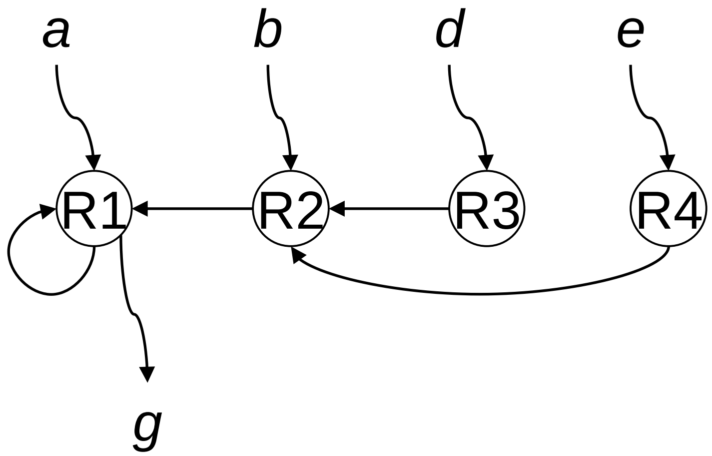


R1 → R1 : 0
R2 → R1 : 1
R3 → R1 : 2
R4 → R1 : 2

a → *g* : 2
b → *g* : 3
d → *g* : 4
e → *g* : 4

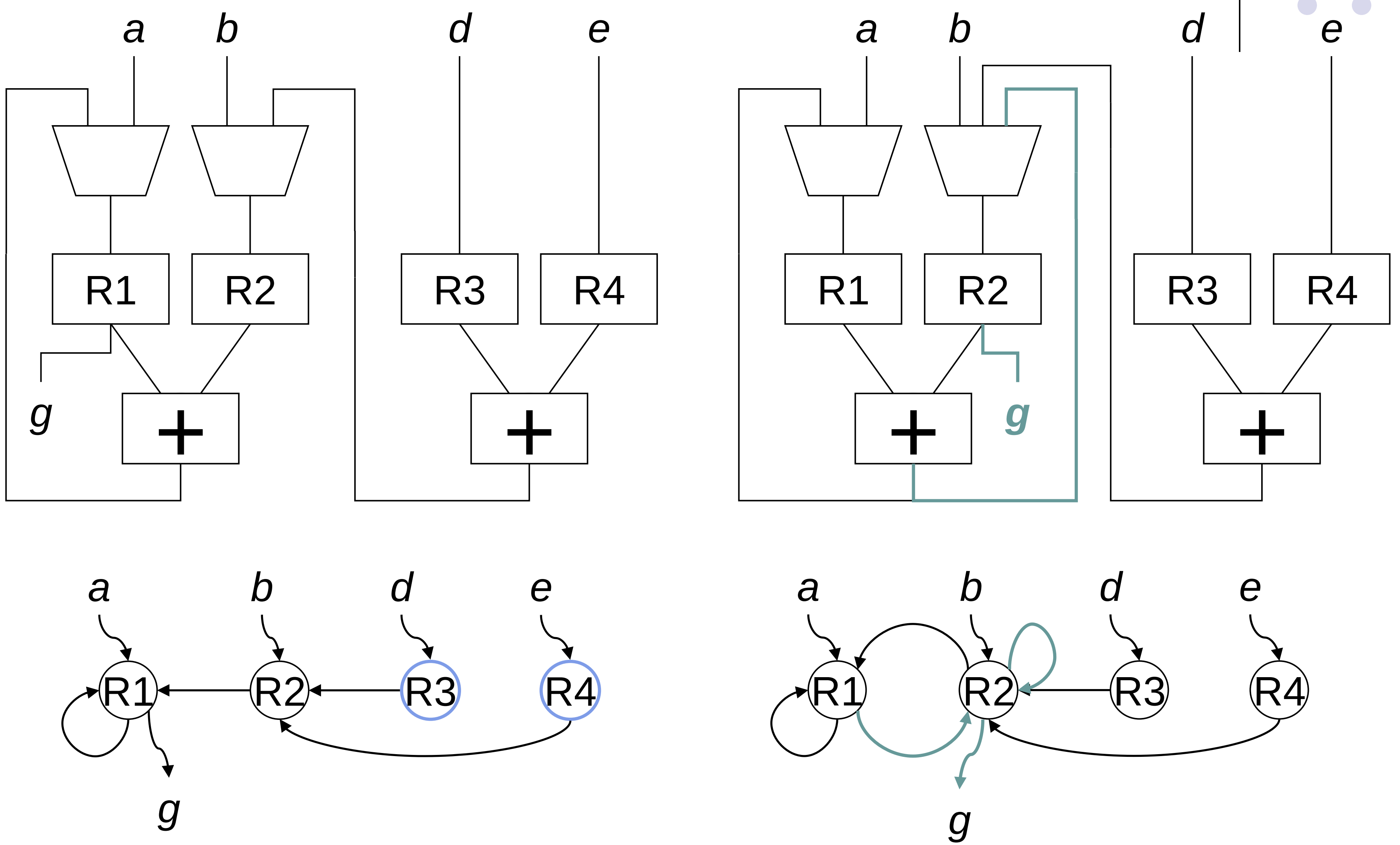
Testability Enhancement

- Improve controllability and observability of registers.
  - Whenever possible, allocate a register to at least one PI or PO.
- Reduce the sequential depth between a controllable and an observable registers.

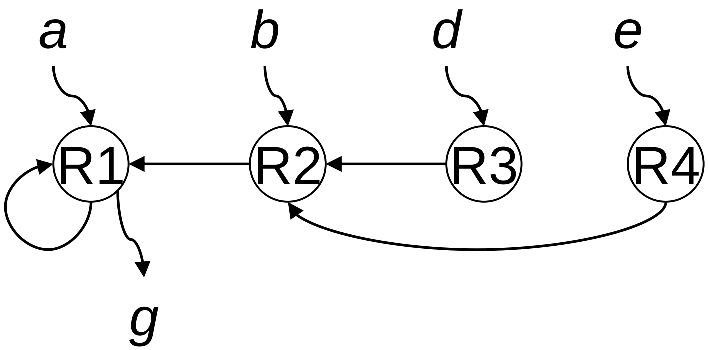


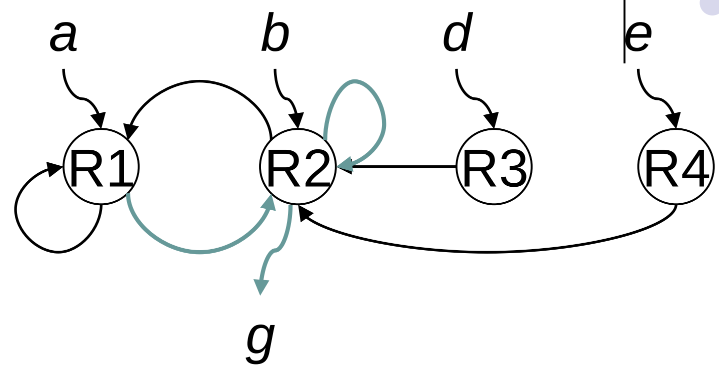
R1 → R1 : 0
R2 → R1 : 1
R3 → R1 : 2
R4 → R1 : 2

An Example



a
b
d
e
R1
R2
R3
R4
g
+
+
a
b
d
e
R1
R2
R3
R4
g
+
+
a
b
d
e
R1
R2
R3
R4
g
a
b
d
e
R1
R2
R3
R4
g





R1 → R1 : 0
R2 → R1 : 1
R3 → R1 : 2
R4 → R1 : 2

R1 → R2 : 1
R2 → R2 : 0
R3 → R2 : 1
R4 → R2 : 1