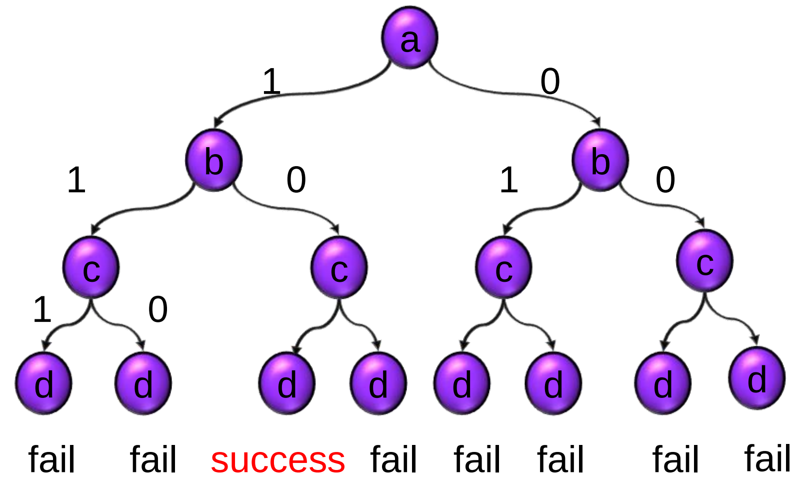
Combinational ATPG

- Introduction
- **Deterministic Test Pattern Generation**
  - Boolean difference approach*
  - Path sensitization method**
  - D-Algorithm [Roth 1966] **
  - PODEM [Goel 1981] **
  - **FAN [Fujiwara 1983]****
  - SAT-based [Larrabee 1992] *
- Acceleration Techniques
- Concluding Remarks

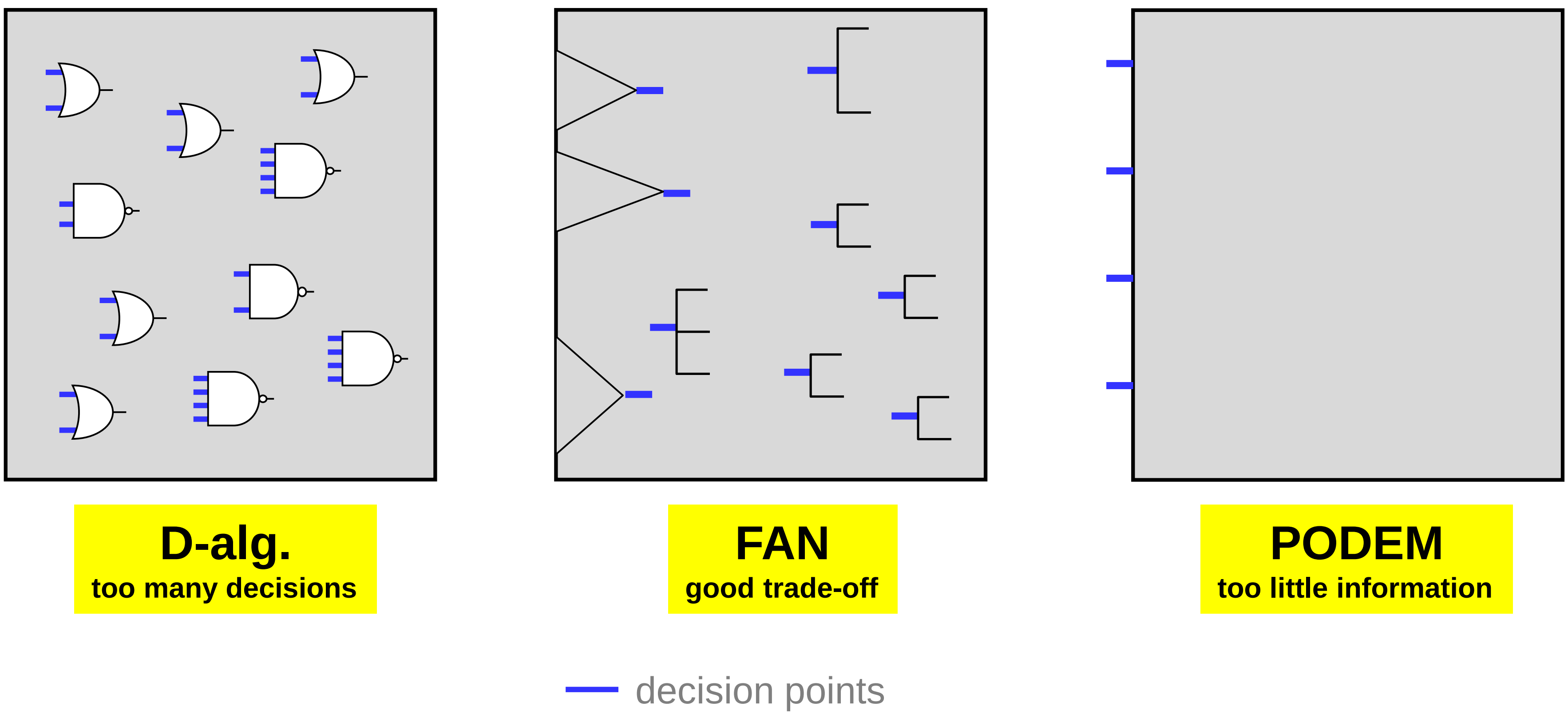
*Boolean-based methods

**path-based methods



D→PODEM →FAN

- D-algorithm decision at internal nodes → too many decisions, slow!
- PODEM decision at PI → too little information, mistake-prone!
- FAN decision at head lines and fanout stems → good trade-off



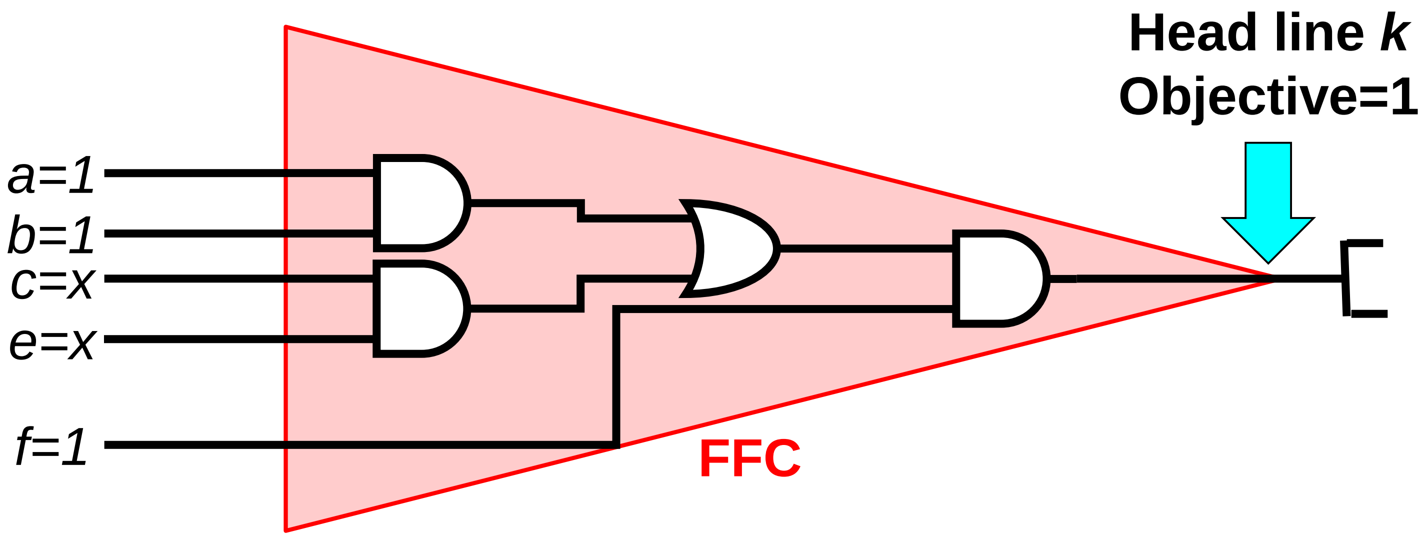
FAN [Fujiwara 1983]

- *FANout-oriented test generation*
- **Four improvements over PODEM**
  - **#1. Make decision at head lines or fanout stem**
  - **#2. Forward/backward Implications**
  - **#3. Unique sensitization**
  - **#4. Multiple backtraces**

https://insights.ubuntu.com

Justify Head Line Is Easy

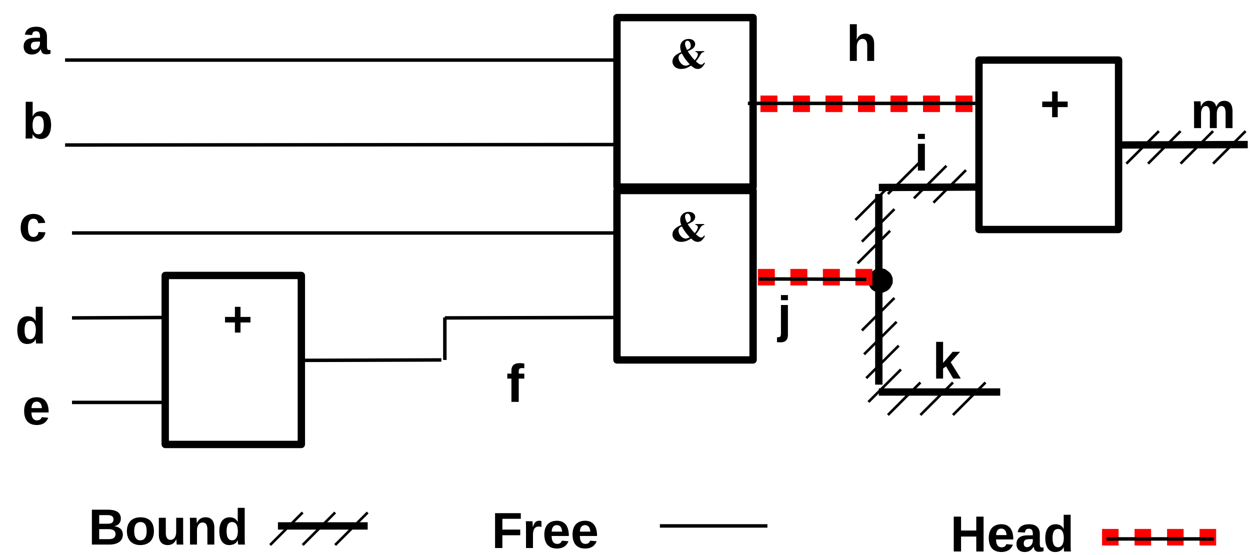
- **Justification inside FFC (*fanout-free cone*) is linear time**
  - **Guaranteed to find an answer**
- **Example: *a,b,c,e,f* are PI, *k* is *head line***
  - **objective *k* =1**



Can Make Decision at Head Line
Instead of PI

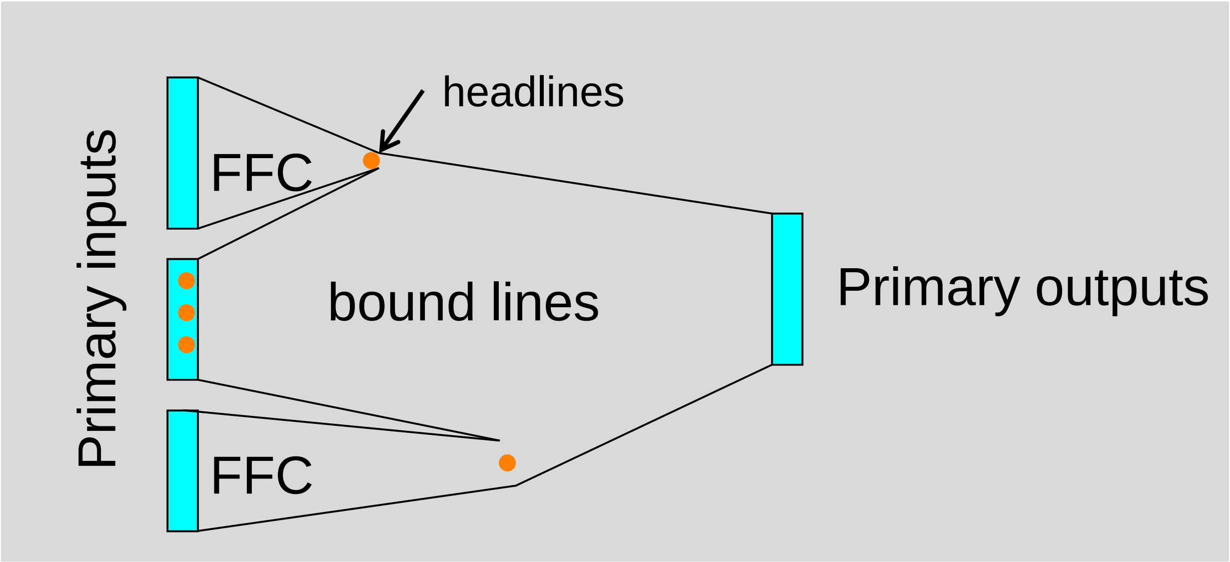
Head Line

- ***Bound line***: line fed directly (*i, k*) or indirectly (*m*) by fanout stem
- ***Free Line***: line that is not Bound (*a~f, h, j*)
- ***Head Line***: free line that is either
  - Fanout stem (*j*), or
  - Input to a gate with bound output (*h*)



#1. Make Decision at Head Lines

- FFC can be isolated from rest of circuit by cutting head lines
- Assignment of PI's that feed head lines are
  - deferred until other objectives have been achieved
    - Reduce search space

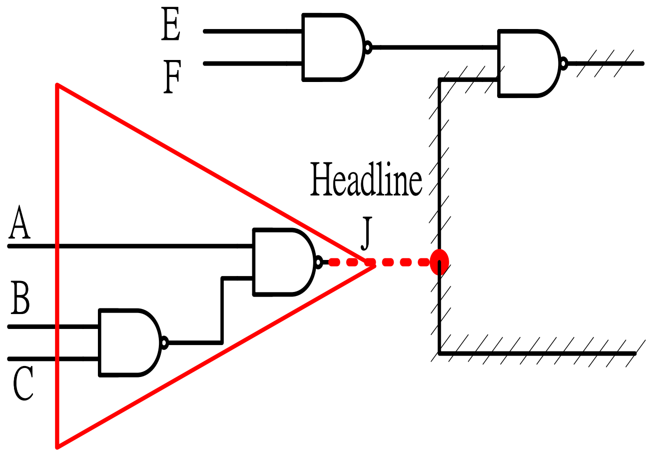


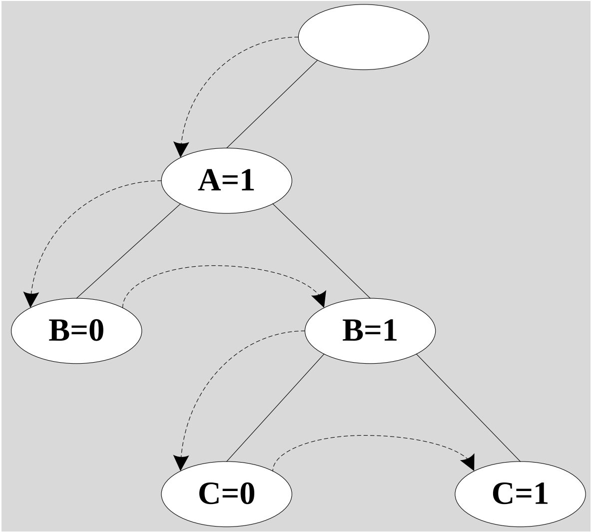
Head Lines Reduces Search Space

Example

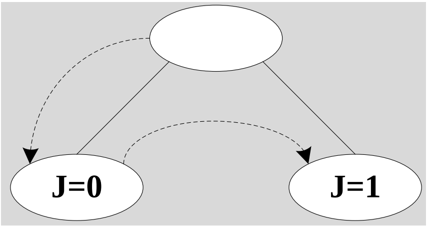
- PODEM decision tree is big
  - A=1,B=0
    - No test, backtrack
  - B=1, C=0
    - No test, backtrack
  - C=1

- FAN decision tree is small
  - J=0
    - No test, backtrack
  - J=1





PODEM

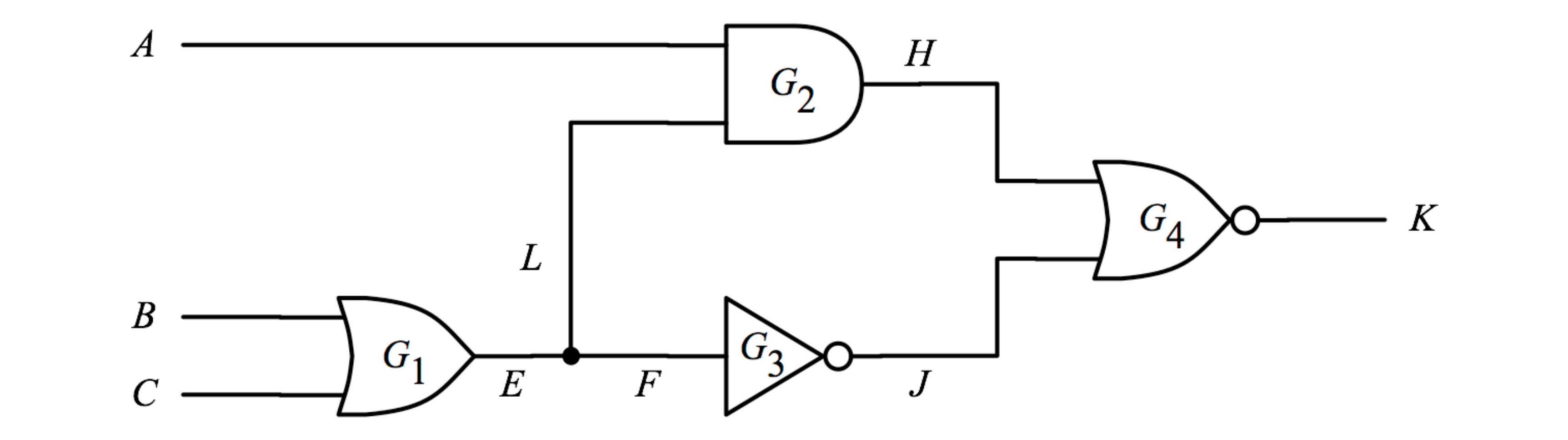


FAN

Quiz

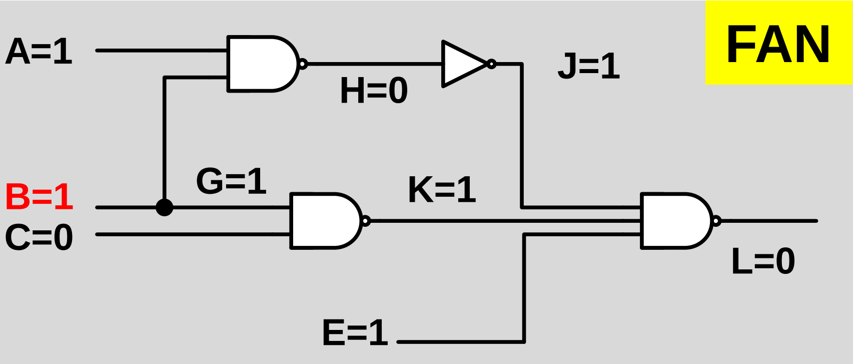
Q: Which are bound lines? free lines? head lines?

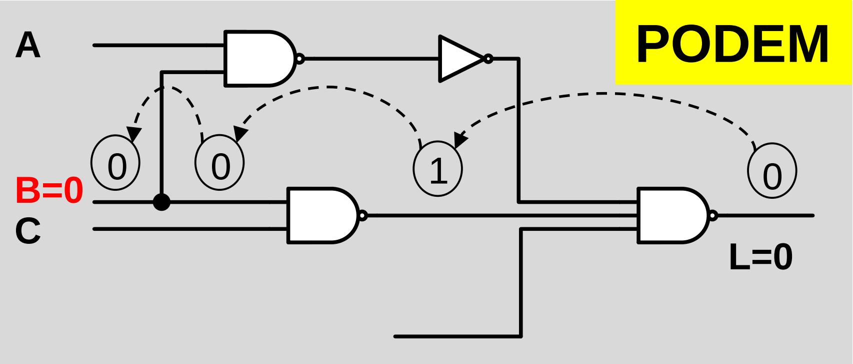
A:



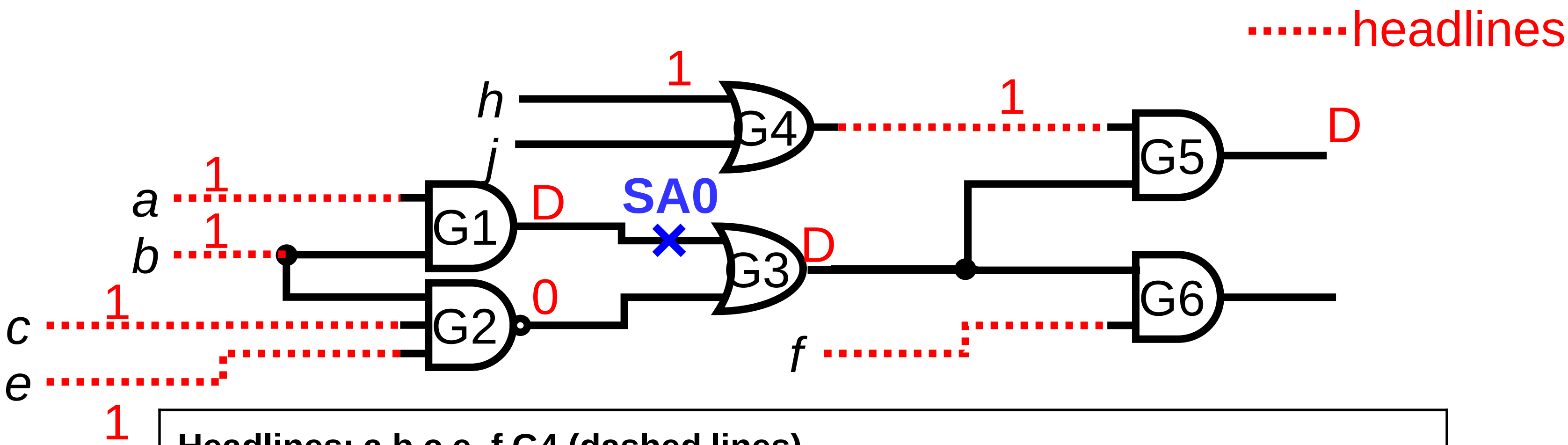
#2. Forward/backward Implication

- PODEM does not assign internal values
  - Only forward implication, no backward implication
- FAN assigns internal values when they are uniquely implied
  - Both forward and backward implication
- Example: *L* SA1 fault
- FAN
  - *Bwd: JKE*=1, *H*=0, *A*=1, *B*=1
  - *Fwd*: *G*=1
  - *Bwd: K=1, C*=0
  - no backtrack needed
- PODEM
  - Backtrace to *B*=0
  - Forward implication
  - Wrong! backtrack





FAN Example



Headlines: a b c e f G4 (dashed lines)
Initial objective: G1 output =1
Implication: assign *a* = 1; *b*=1
Objective: propagate through G3, objective G2=0
Implication: assign *c* = 1; e=1
Propagate through G5, objective G4=1
Assign headline G4 = 1
G5=D, Objective achieved.
Justify head line G4 = 1 → h=1
Test generated abcehjf = 11111xx

Implication

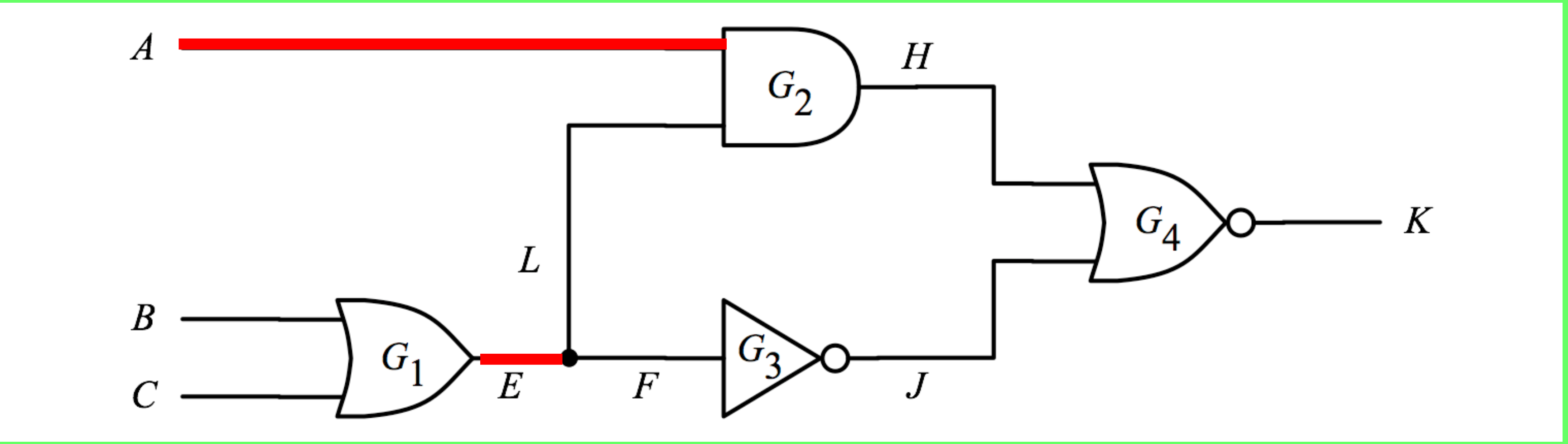
Make decision at head line

Justify head lines at end

Quiz

Q: If we want *K*=1, apply implication to determine head lines *A*=? *E*=?

ANS:



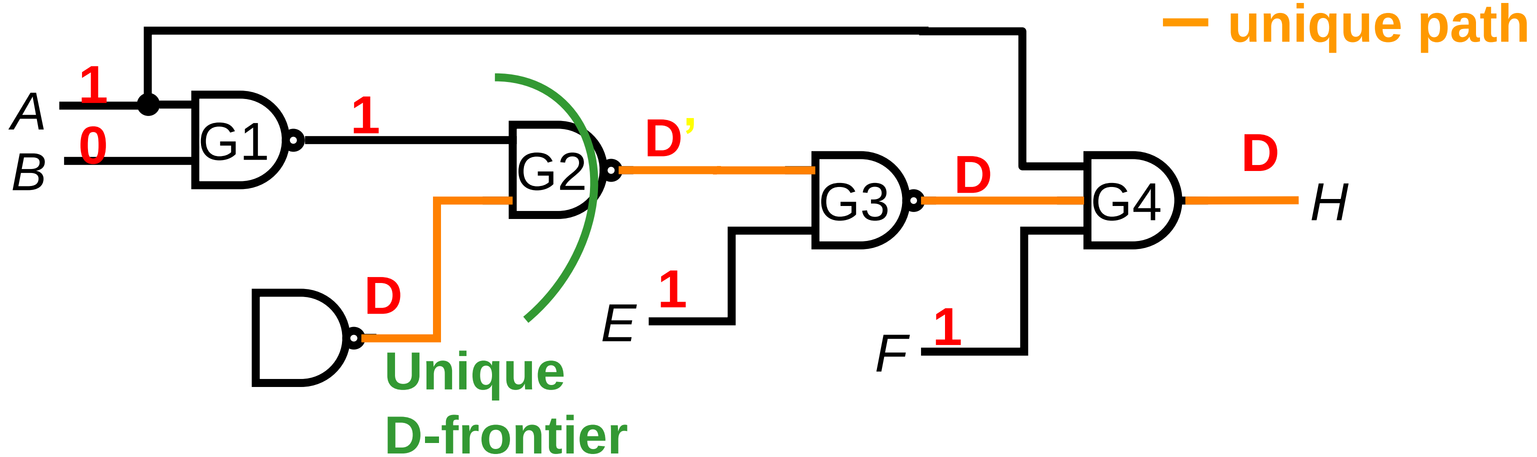
FAN

- **Four improvements over PODEM**
  - **#1. Make decision at head lines or fanout stem**
  - **#2. Forward/backward Implications**
  - **<u>#3. Unique sensitization</u>**
  - **<u>#4. Multiple backtraces</u>**

www.walmart.com

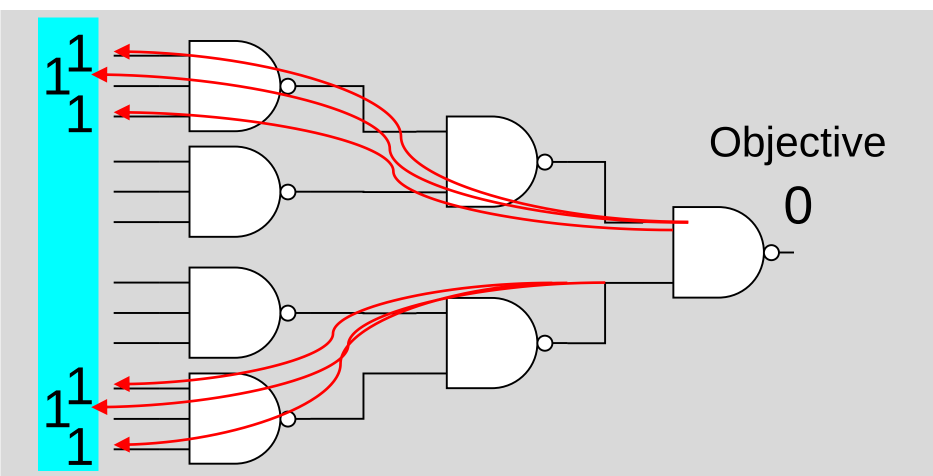
#3. Unique Sensitization

- When there is **only one gate in D-frontier**
  - if unique path exists, set side inputs to non-controlling values
- Example:
  - FAN
    - *G2* is unique D-frontier, only one path to *H*
    - *G1* = 1, *E* = 1, *F* = 1, *A* = 1 → *B* = 0 → success!
  - PODEM
    - Initial objective: *G1* = 1 →
    - backtrace to *A* = 0 → X-path disappear!

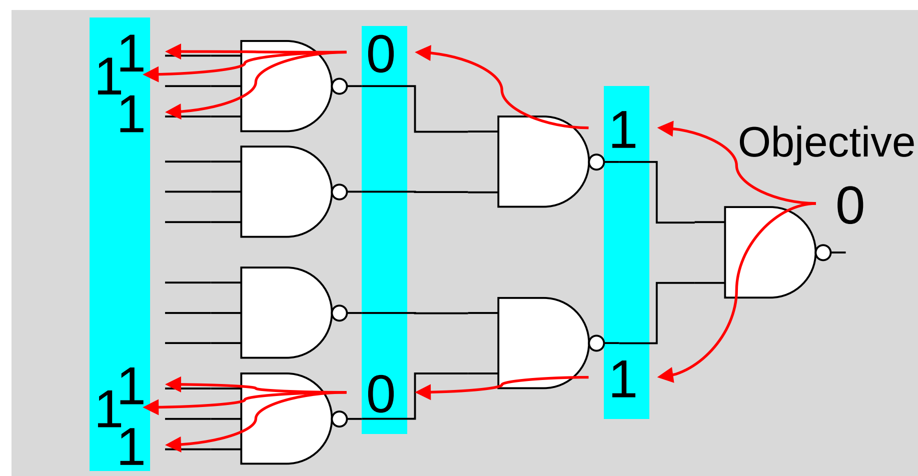


#4. Multiple Backtraces

- PODEM uses **depth-first search (DFS)**
  - One single backtrace at a time
- FAN uses **breadth first search (BFS)**
  - Multiple parallel search at a time
- Example
  - PODEM needs **6 backtraces**
  - FAN needs only **1 multiple backtrace**



PODEM

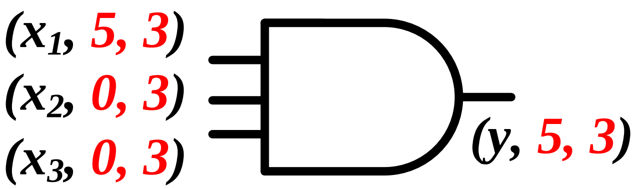


FAN

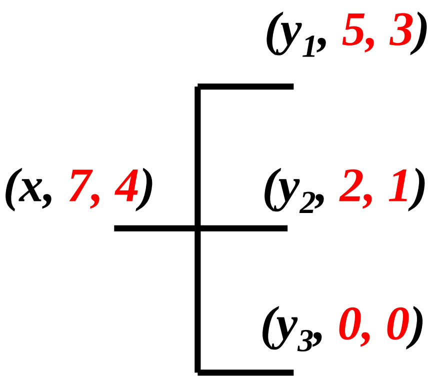
Rules for Multiple Backtraces

- Objective: (x, n_0, n_1)
  - ♦ number of backtraced zeros (n_0) and ones (n_1) on signal x

- For AND gate
  - ♦ Easiest unspecified input x_1
    - * $(x_1, n_0, n_1) = (y, n_0, n_1)$
  - ♦ Other inputs x_2, x_3
    - * $(x_2, n_0) = 0$
    - * $(x_2, n_1) = (y, n_1)$

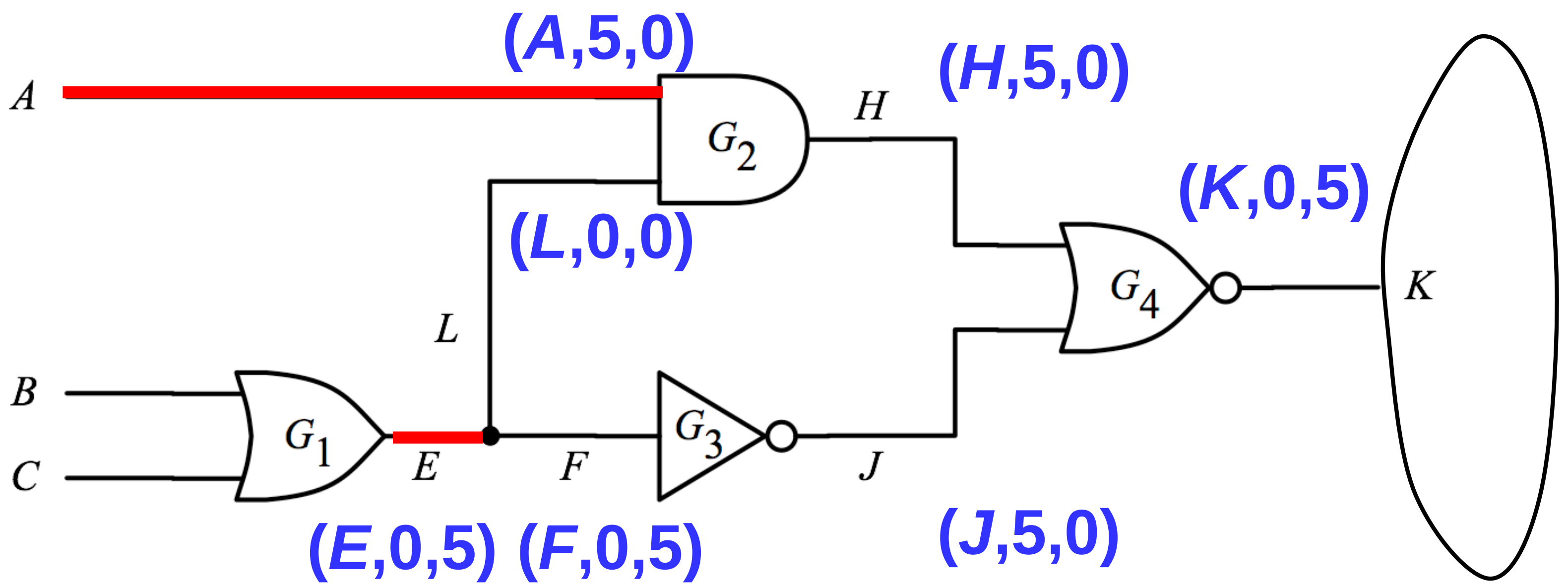


- For fanout Stem
    - * (x, n_0) = sum of (y_i, n_0)
    - * (x, n_1) = sum of (y_i, n_1)



Example

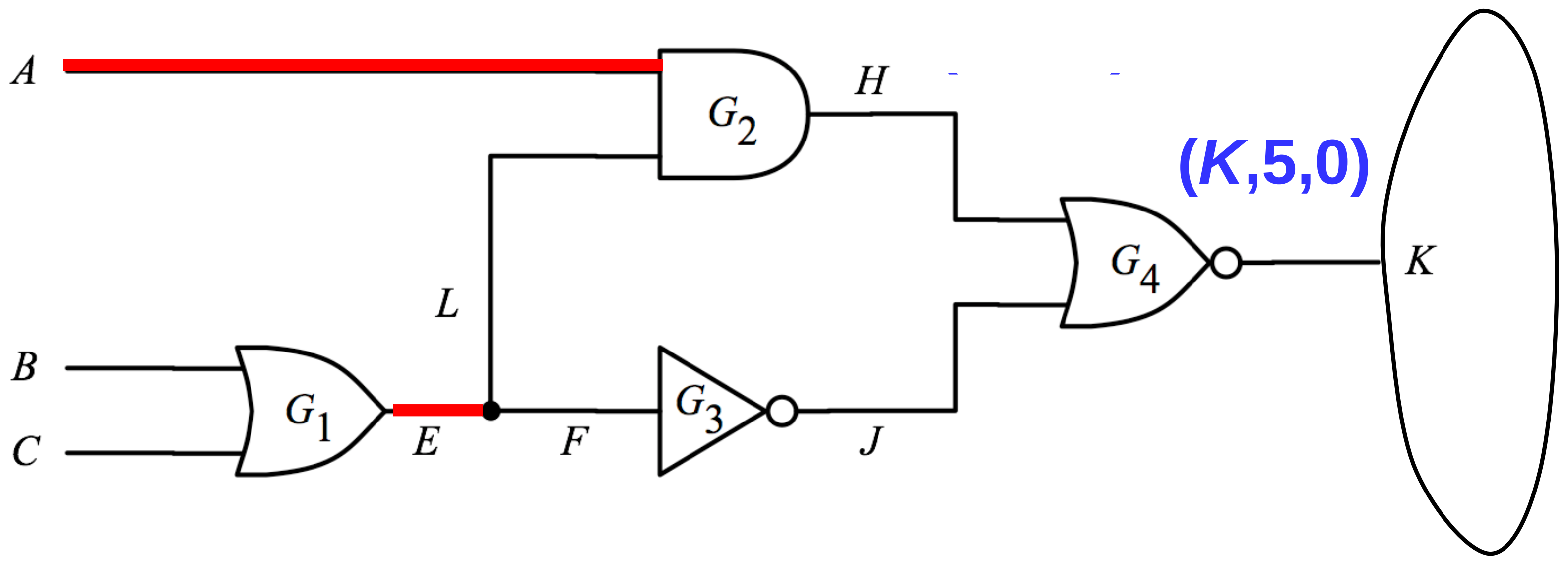
- Starting from (*K*,0,5), multiple backtrace to head lines *A* and *E*
  - So we get two assignments *A*=0, *E*=1



Quiz

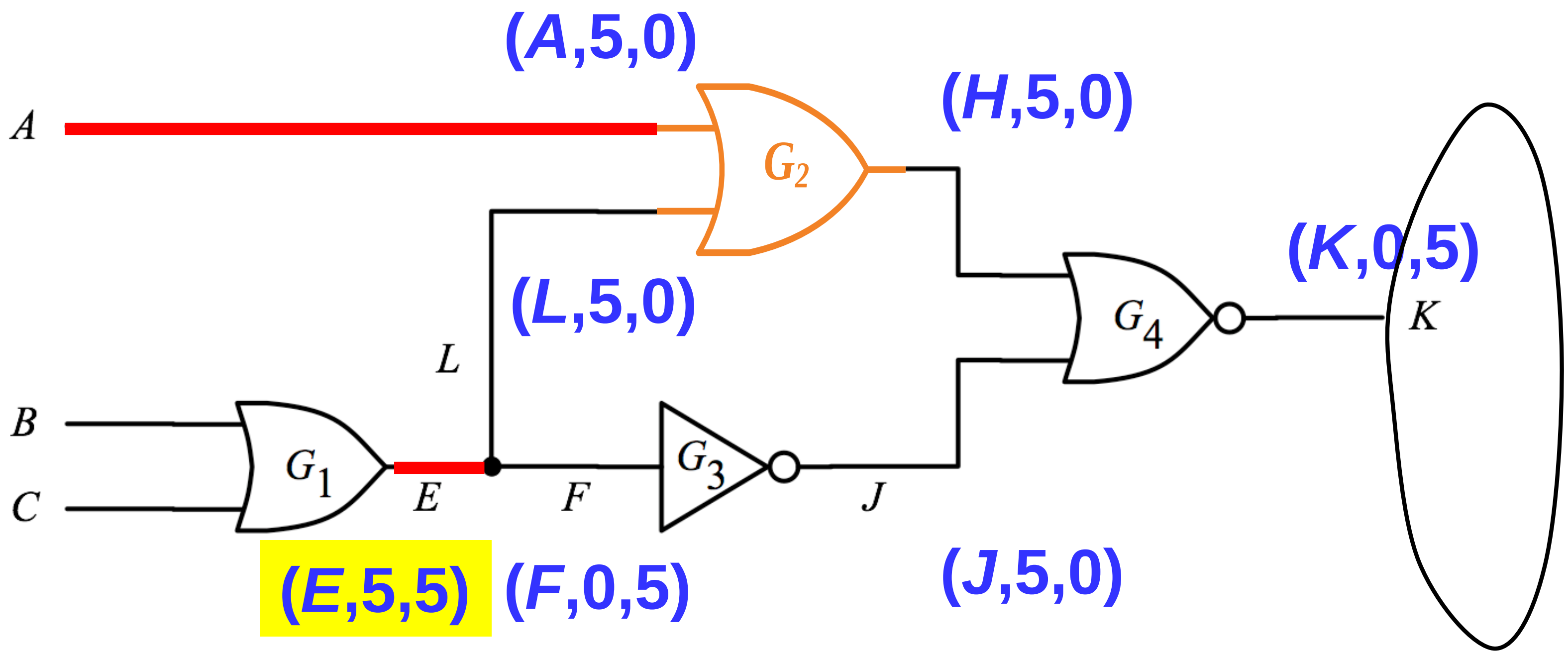
Q: Starting from (*K*, 5, 0), multiple backtrace to head lines (*E*, n_0, n_1)=? (*A*, n_0, n_1)=? Suppose *H* is chosen over *J*

A:



Multiple Backtrace Conflict

- What if change G_2 to OR gate?
- Want K=1, perform multiple backtrace to headlines A and E
  - conflicting values at E !



- How to handle a conflict ?
  - assign a value **most requested**, then start next backtrace
  - will backtrack if it is wrong

Multiple Backtrace (1/2)

```
MultipleBacktrace (Initial_objectives, Fanout_objectives) {
  Current_objectives = Initial_objectives
  while (Current_objectives ≠ ϕ  or Fanout_objectives ≠ ϕ) {
      dequeue entry (k, v_k) from Current or Fanout_objectives
      switch (type of entry) {    //   (k, v_k ) = want v_k on signal k
         1. HEAD_LINE:
            add (k, v_k) to Headline_objectives
         2. FANOUT_BRANCH:
            j = stem(k);
            increment n_0 or n_1 at j for v_k;   //sum of n_0, n_1
            add j to Fanout_objectives
         3. GATE:                                  //page 15
            i = inversion of k; c = controlling value of k;
            if ((v_k ⊕ i) == c) {
                  select easiest input  j with unknown value
                  add( j, c) to Current_objectives;}
            else {
                  for every input  j of k with value
                  add(j, c') to Current_objectives; }
      }  // switch
```

Multiple Backtrace (2/2)

(cont'd from previous page) *simplified from Fig. 8 of FAN paper

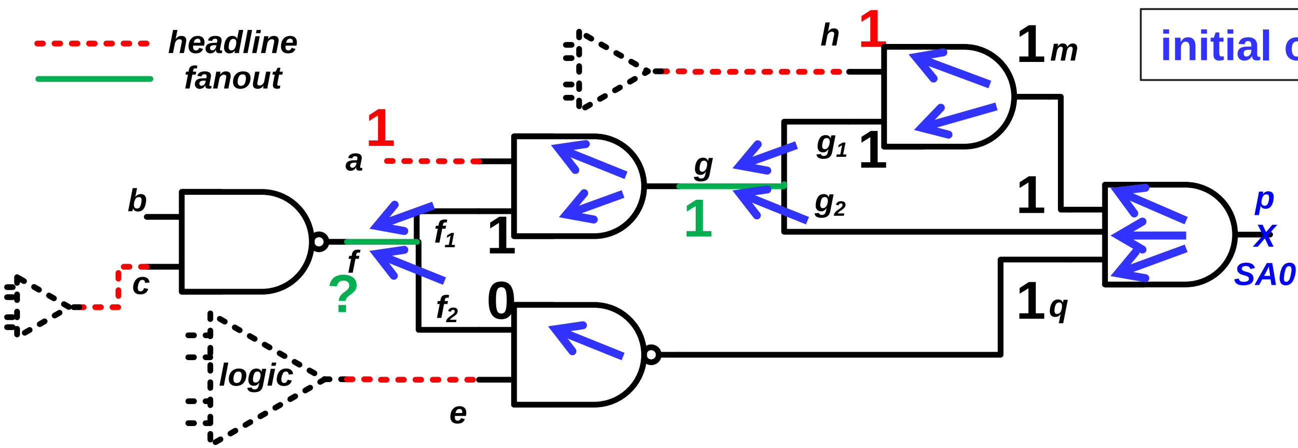
if(Fanout_objectives $\neq \phi$) {
  dequeue highest-level stem (k) from Fanout_Objectives
  v_k = 0 or 1, depends on which of (n_0 , n_1) is larger
  if there is no conflict on k {
    add (k, v_k) to Current_objectives} // continue backtrace
  else { return (k, v_k) as the Final_objective} // stop backtrace
else { // no fanout objective
  dequeue (k, v_k) from Headline_objectives
  return (k, v_k) as the Final_objective }
} // while
} // MultipleBacktrace

MultipleBacktrace

final_objective
(either a fanout or a headline)

initial_objectives
fanout_objectives

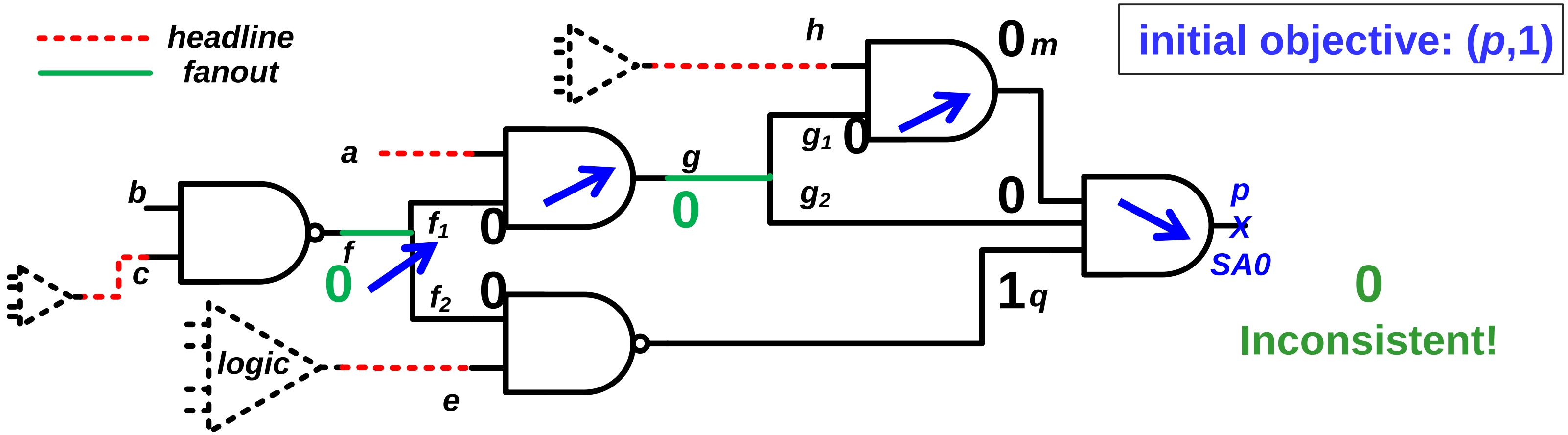
Multiple Backtrace Example (1/3)



Current Obj.	processed entry	Fanout Obj.	Headline Obj.
$(m,1)(g_2,1)(q,1)$	$(m,1)$	-	$(h,1)$
$(g_2,1)(q,1)(g_1,1)$	$(g_2,1)$	$(g,n_1=1)$	$(h,1)$
$(q,1)(g_1,1)$	$(q,1)$	$(g,n_1=1)$	$(h,1)$
$(g_1,1)(f_2,0)$	$(g_1,1)$	$(g,n_1=2)$	$(h,1)$
$(f_2,0)$	$(f_2,0)$	$(g,n_1=2)$ $(f,n_0=1)$	$(h,1)$
-	$(g,1)$ *consistent*	$(f,n_0=1)$	$(h,1)$
$(g,1)$	$(g,1)$	$(f,n_0=1)$	$(h,1)(a,1)$
$(f_1,1)$	$(f_1,1)$	$(f,n_0=1\ n_1=1)$	$(h,1)(a,1)$
-	*f* *conflict!*	-	$(h,1)(a,1)$

Final Objective: *f=?*

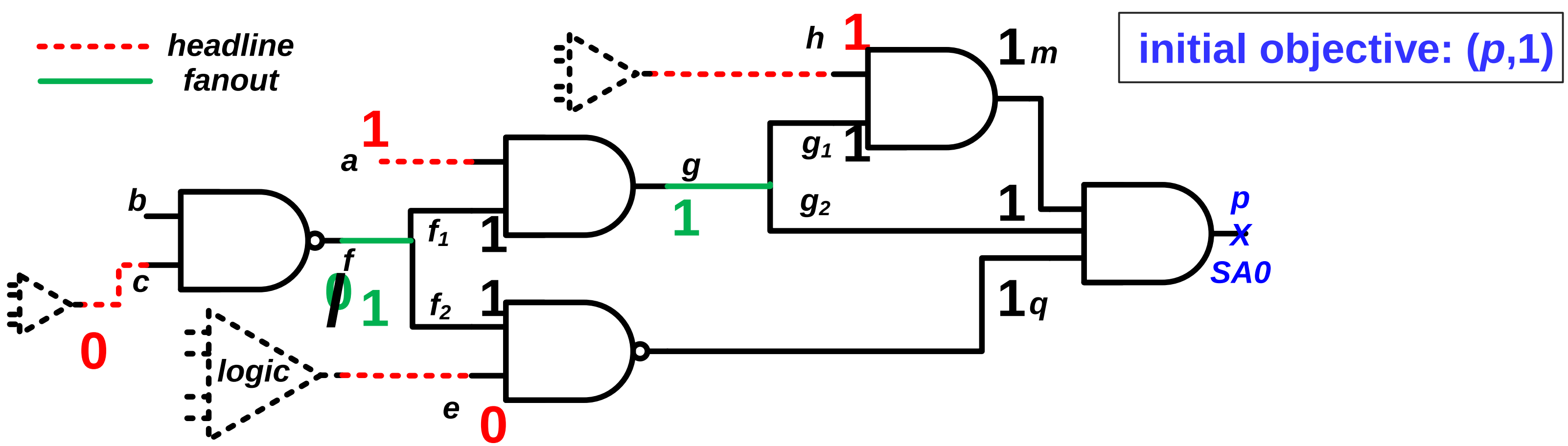
Multiple Backtrace Example (2/3)



because n_0=n_1=1, just choose randomly
suppose we assign *f* = 0
forward implication
inconsistent with initial objectives!
backtrack to *f*=1

Decision at Fanout Stem
Detects Inconsistency Earlier

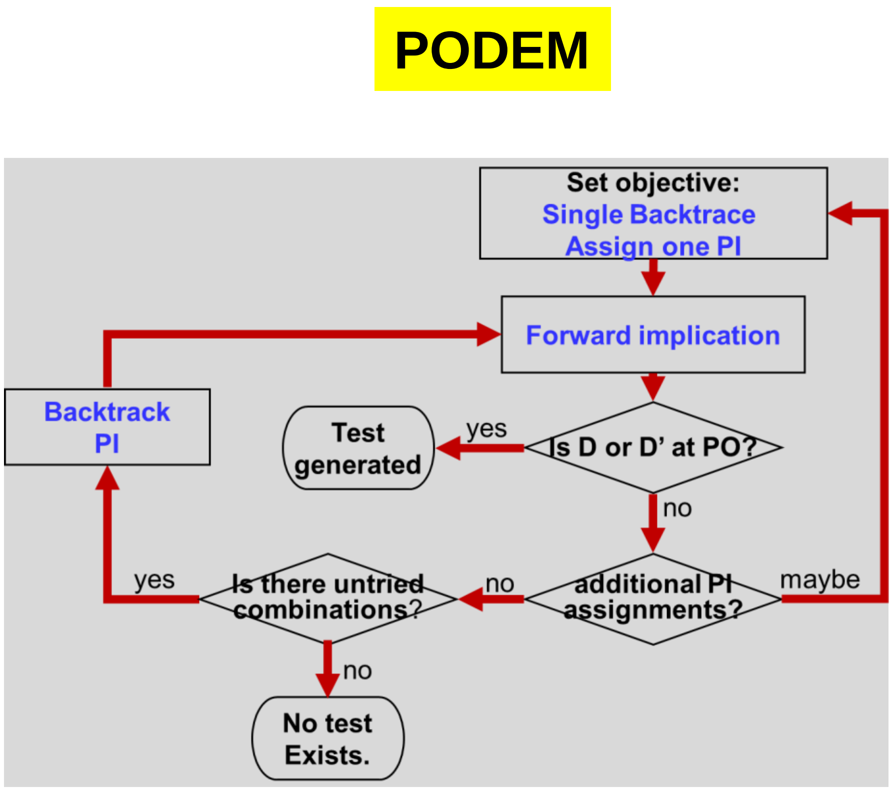
Multiple Backtrace Example (3/3)

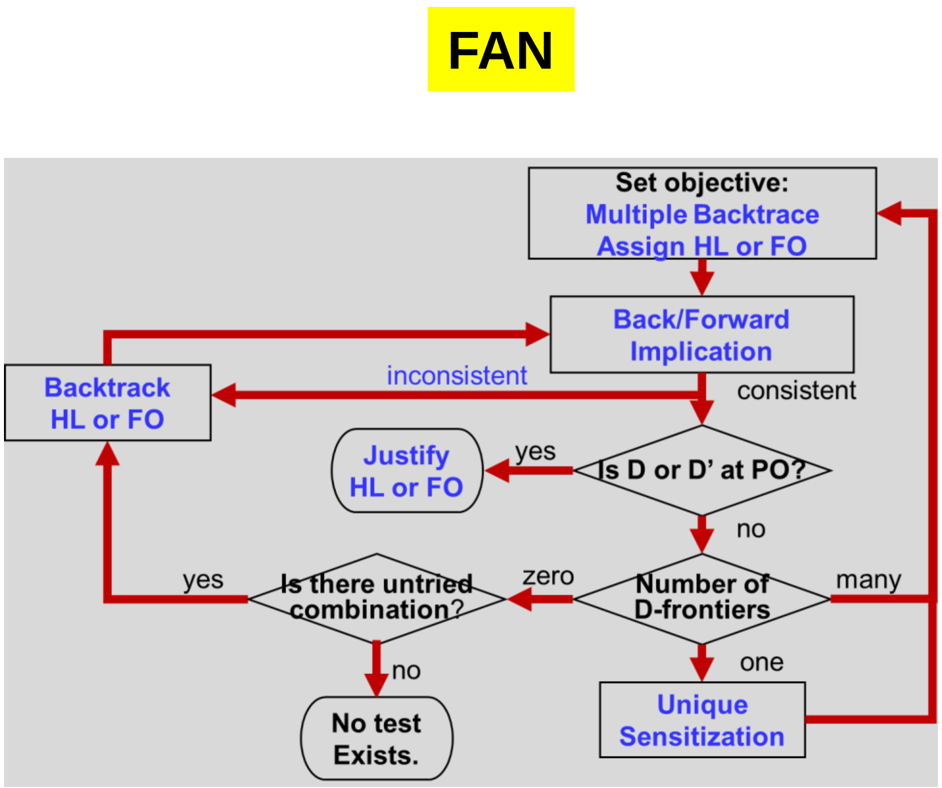


Assign ***f*=1**
Forward implication, consistent.
Multiple_Backtrace again
This time, headline objectives: ***h=1, a=1, e=0, c=0***
Forward implication
Initial objective achieved!

Multiple Backtrace is Fast

PODEM v.s. FAN

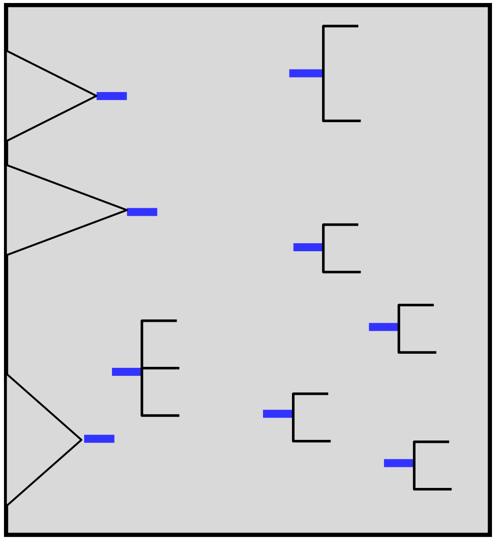




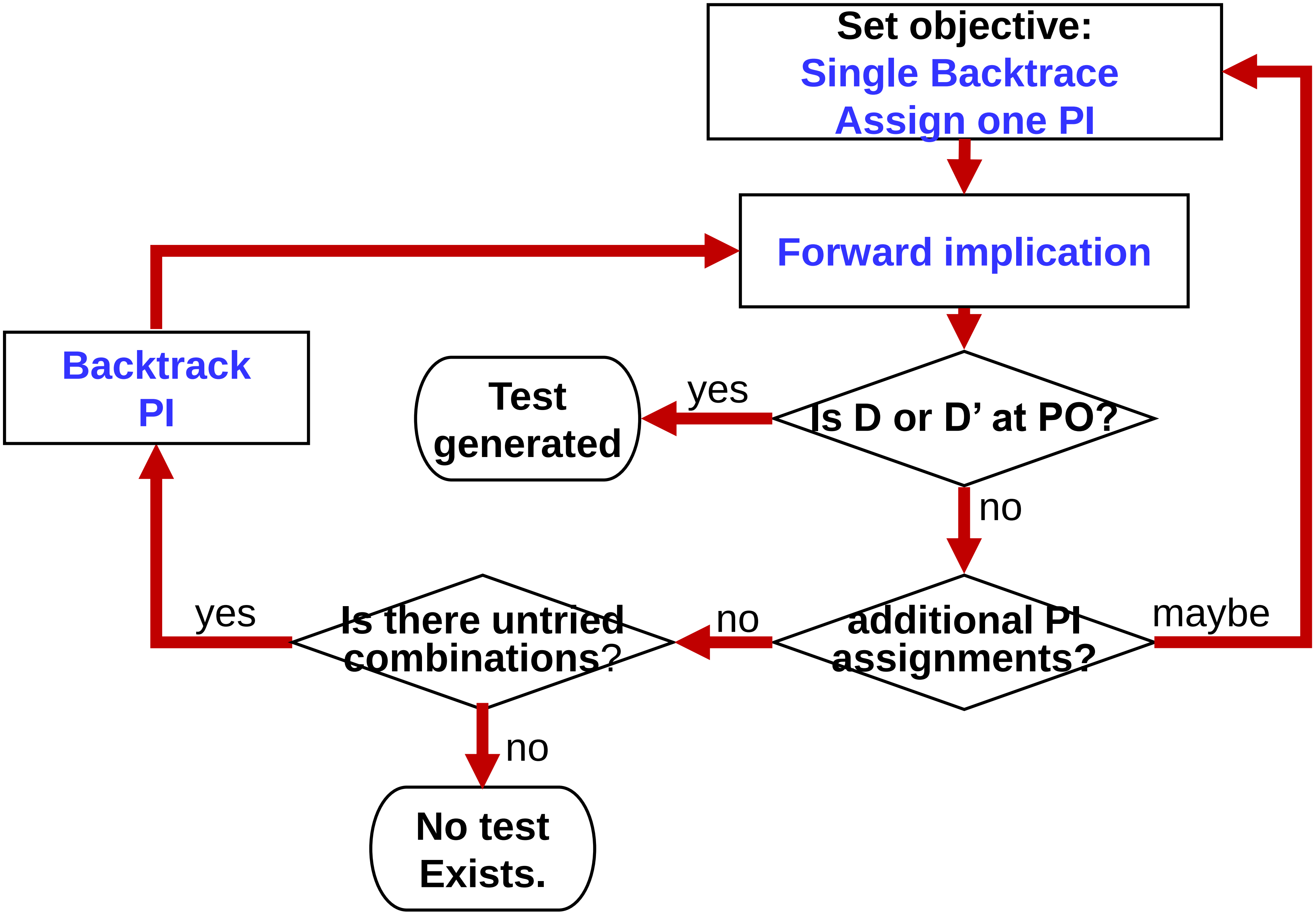
Difference highlighted in Blue

Summary

- 1. Make decision at head lines and fanout stem
  - Reduce search space
- 2. Forward/backward Implications
  - More information to make correct decision
- 3. Unique sensitization
  - Unique path to output
- 4. Multiple backtraces
  - BFS to search many paths together



Flowchart of PODEM



Flowchart of FAN

*simplified from Fig. 9 of FAN paper