Counter-Example Guided Abstraction Refinement

Testing & Verification

Dept. of Computer Science & Engg, IIT Kharagpur

Pallab Dasgupta

Professor, Dept. of Computer Science & Engg.,
Professor-in-charge, AVLSI Design Lab,
Indian Institute of Technology Kharagpur

Reference

This presentation is based on the work of E. Clarke, A. Gupta, J. Kukula, O. Strichman (CAV'02). Most of these slides are from A. Gupta's presentation.

Model Checking

- **Given:**
  - Finite transition system $M(S, I, R, L)$
  - A temporal property p

- **The model checking problem:**
  - Does M satisfy p?

$$M \overset{?}{\models} p$$

Model Checking

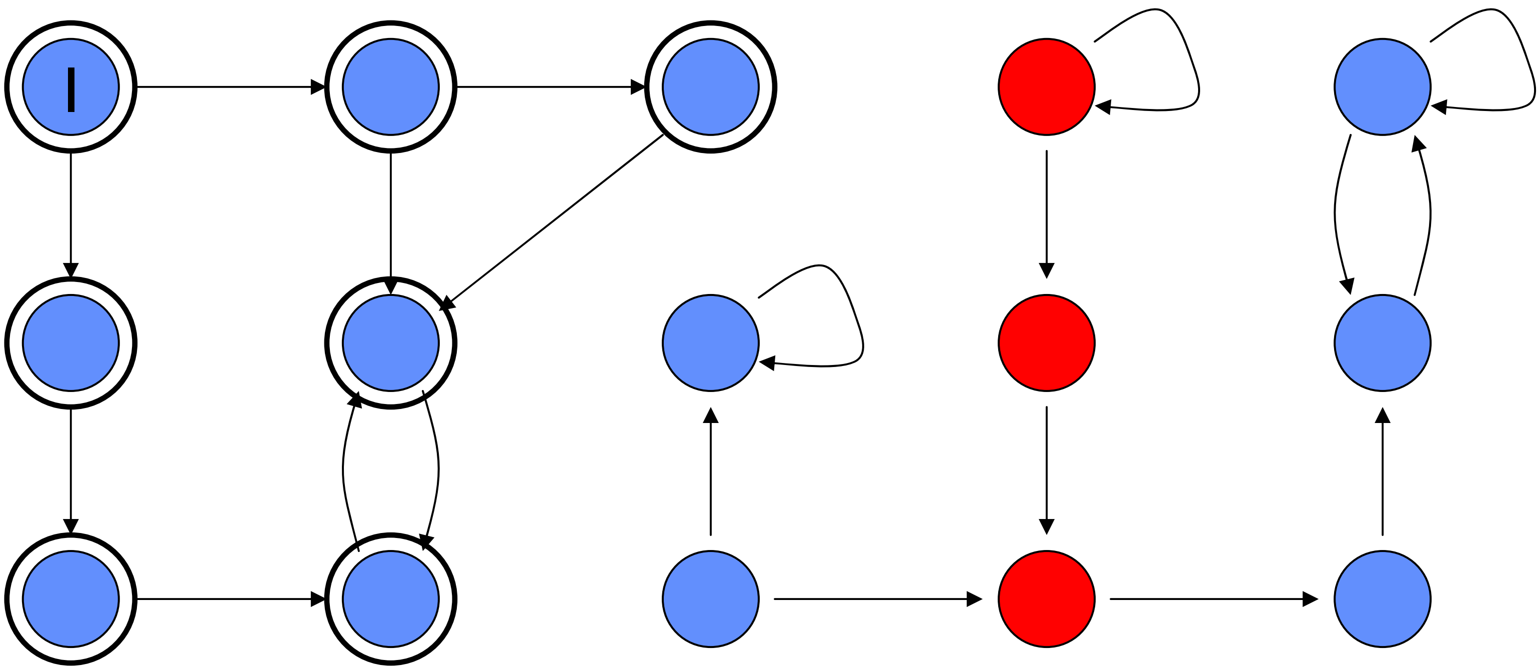
- **Temporal properties:**
  - **"Always x=y"**
    (G(x=y))
  - **"Every Send is followed immediately by Ack"**
    (G(Send → X Ack))

  "Safety" properties

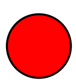
  - **"Reset can always be reached"**
    (GF Reset)
  - **"From some point on, always switch_on"**
    (FG switch_on)

  "Liveness" properties

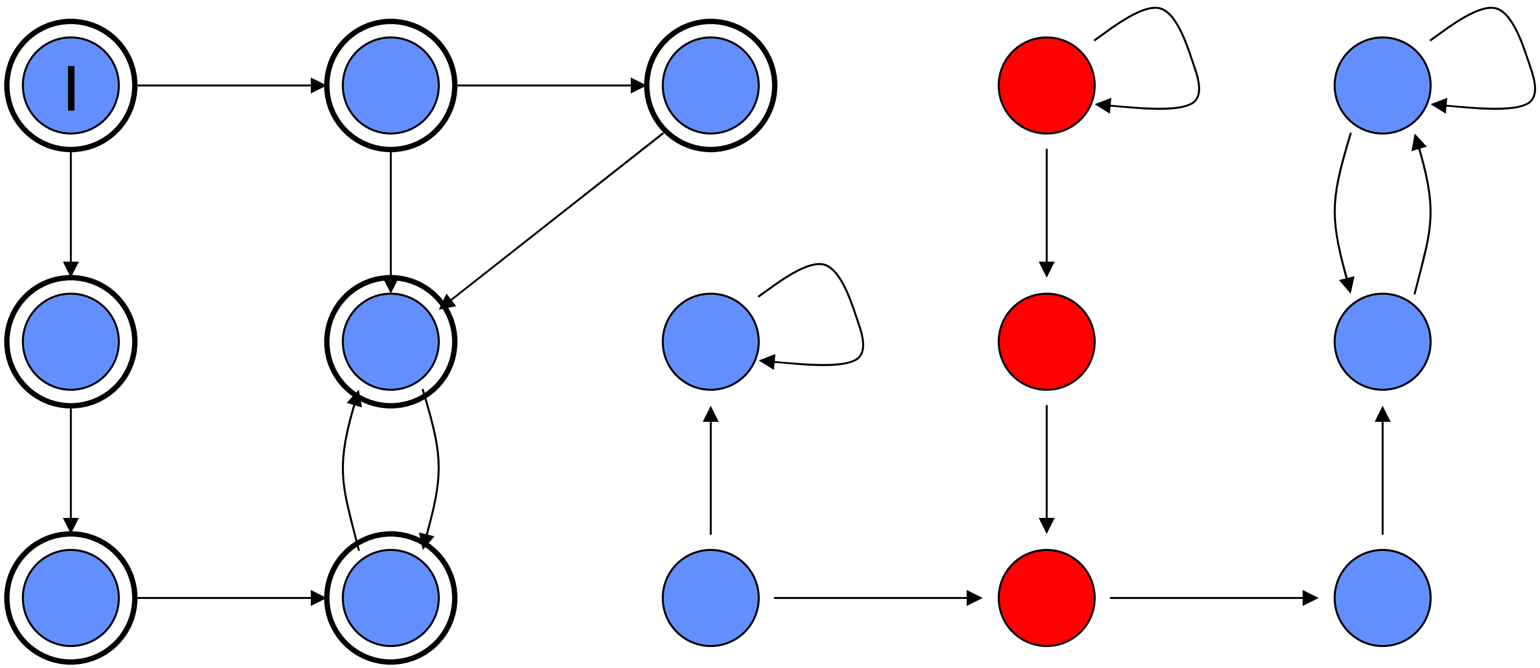
Model Checking (safety)



Add reachable states until reaching a fixed-point

 = bad state

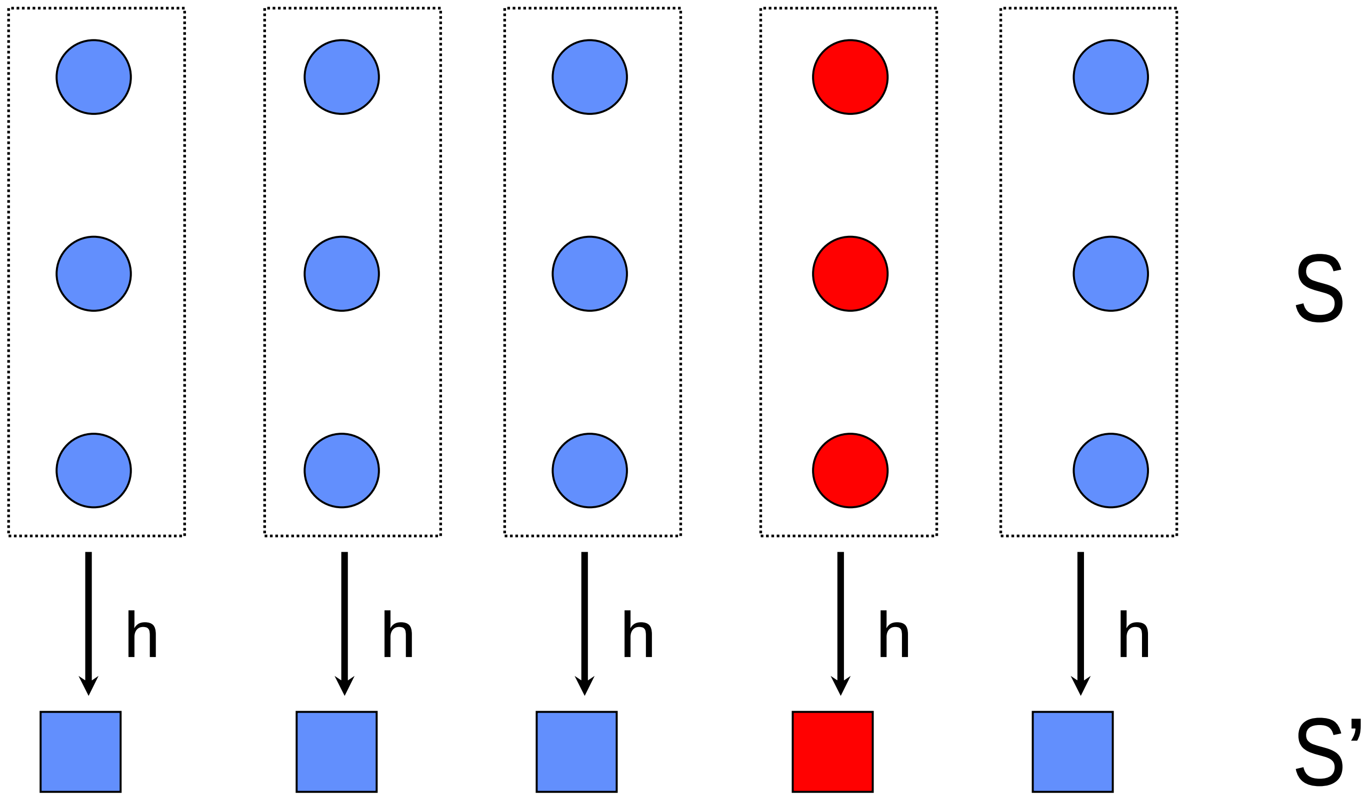
Model Checking (safety)



Too many states to handle !

= bad state

Abstraction



Abstraction Function h : S ! S’

Abstraction Function

- Partition variables into visible(V) and invisible(I) variables.

  - The abstract model consists of V variables. I variables are made inputs.

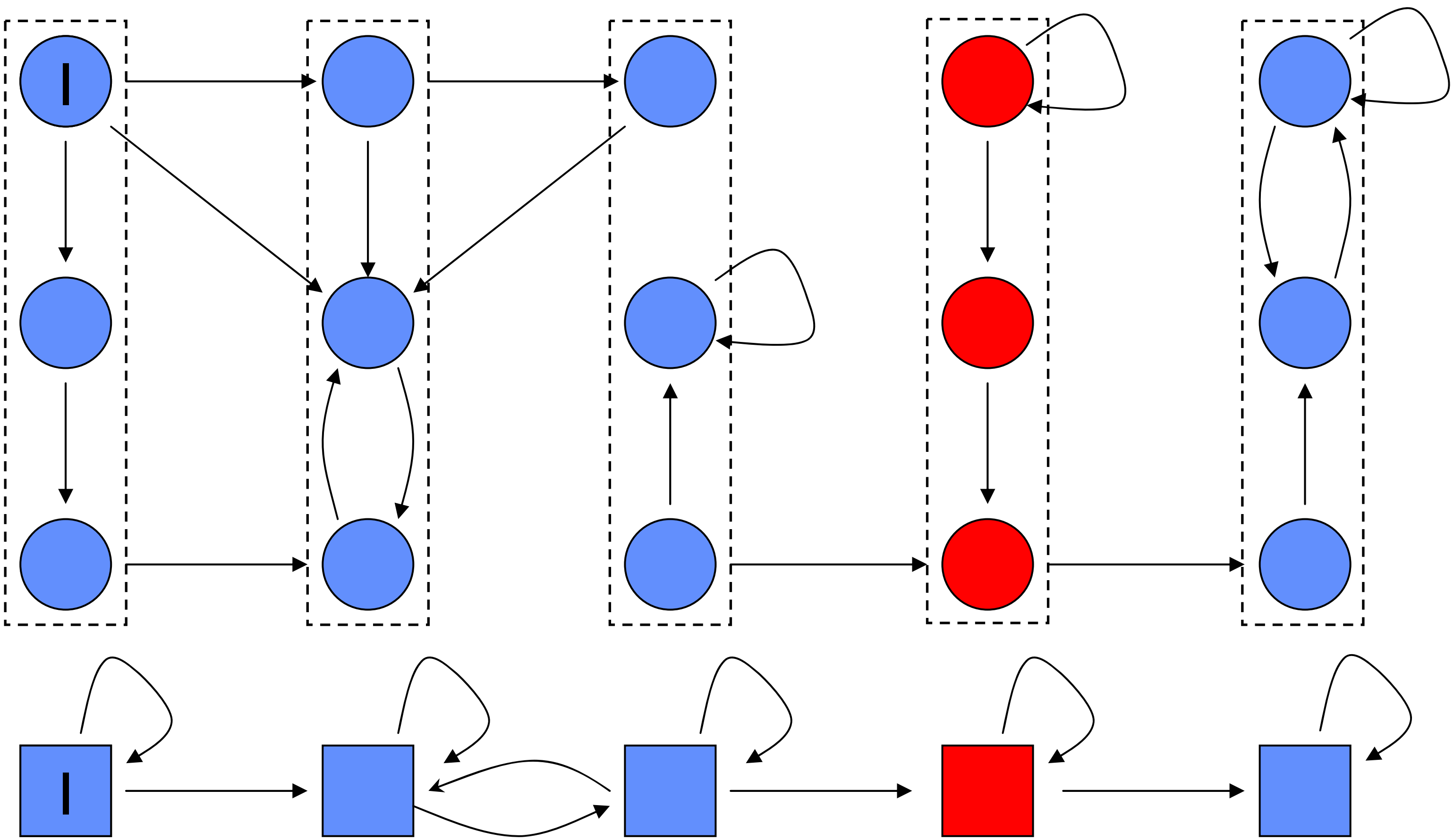
  - The abstraction function maps each state to its projection over V.

Abstraction Function

$$\begin{array}{cccc} x1 & x2 & x3 & x4 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array} \xrightarrow{\ \mathrm{h}\ } \begin{array}{cc} x1 & x2 \\ \hline 0 & 0 \end{array}$$

Group concrete states with identical visible part to a single abstract state.

Existential Abstraction



Model Checking Abstract Model

- **Preservation Theorem**

$$M' \models p \rightarrow M \models p$$

- **Converse does not hold**

$$M' \not\models p \not\rightarrow M \not\models p$$

- **The counterexample may be spurious**

Checking the Counterexample

- Counterexample : $(c_1, \ldots, c_m)$
  - Each c_i is an assignment to V.

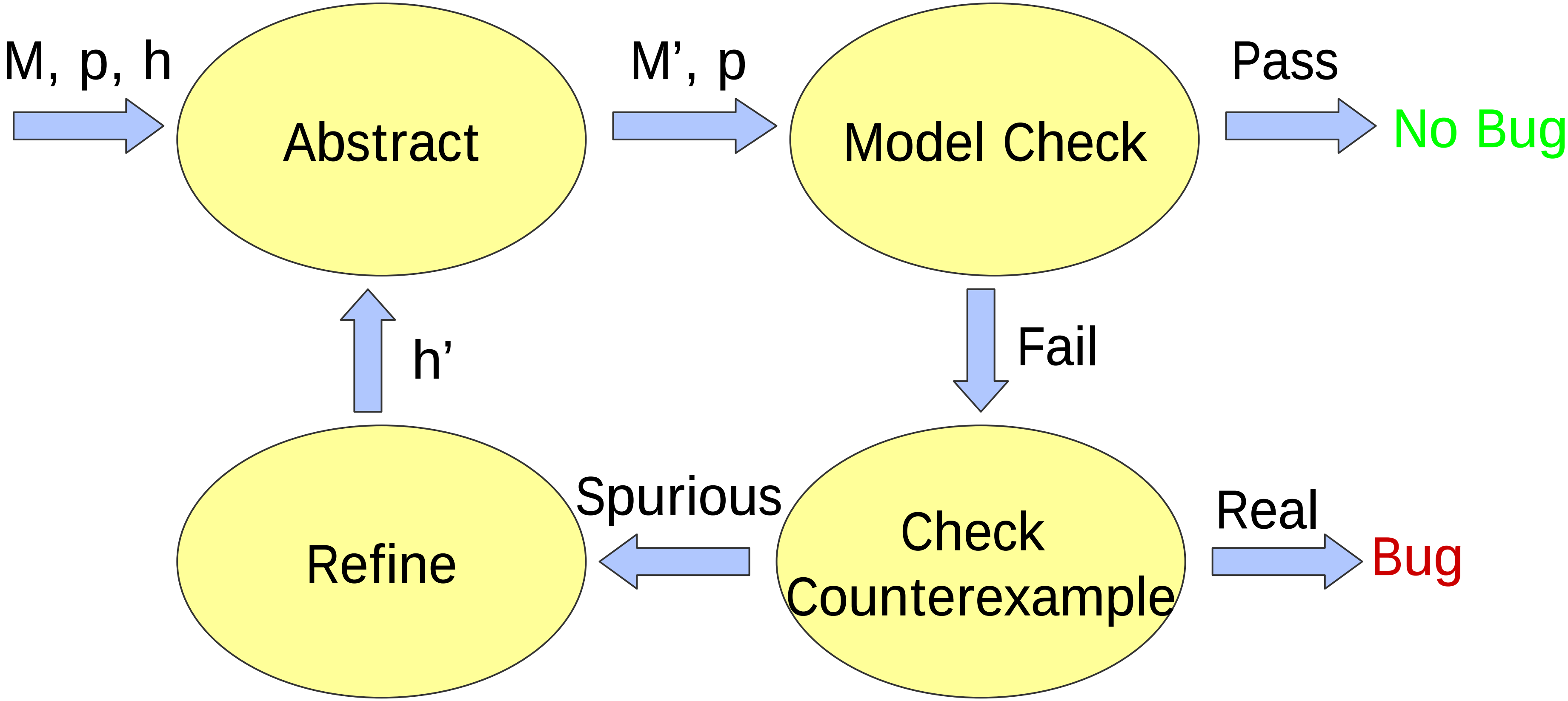
- Simulate the counterexample on the concrete model.

Checking the Counterexample

Concrete traces corresponding to the counterexample:

$\phi \;=\; I(s_1) \;\wedge$	**(Initial State)**
$\bigwedge_{i=1}^{m-1} R(s_i, s_{i+1}) \;\wedge$	**(Unrolled Transition Relation)**
$\bigwedge_{i=1}^{m} \mathsf{visible}(s_i) = c_i$	**(Restriction of V to Counterexample)**

Abstraction-Refinement Loop



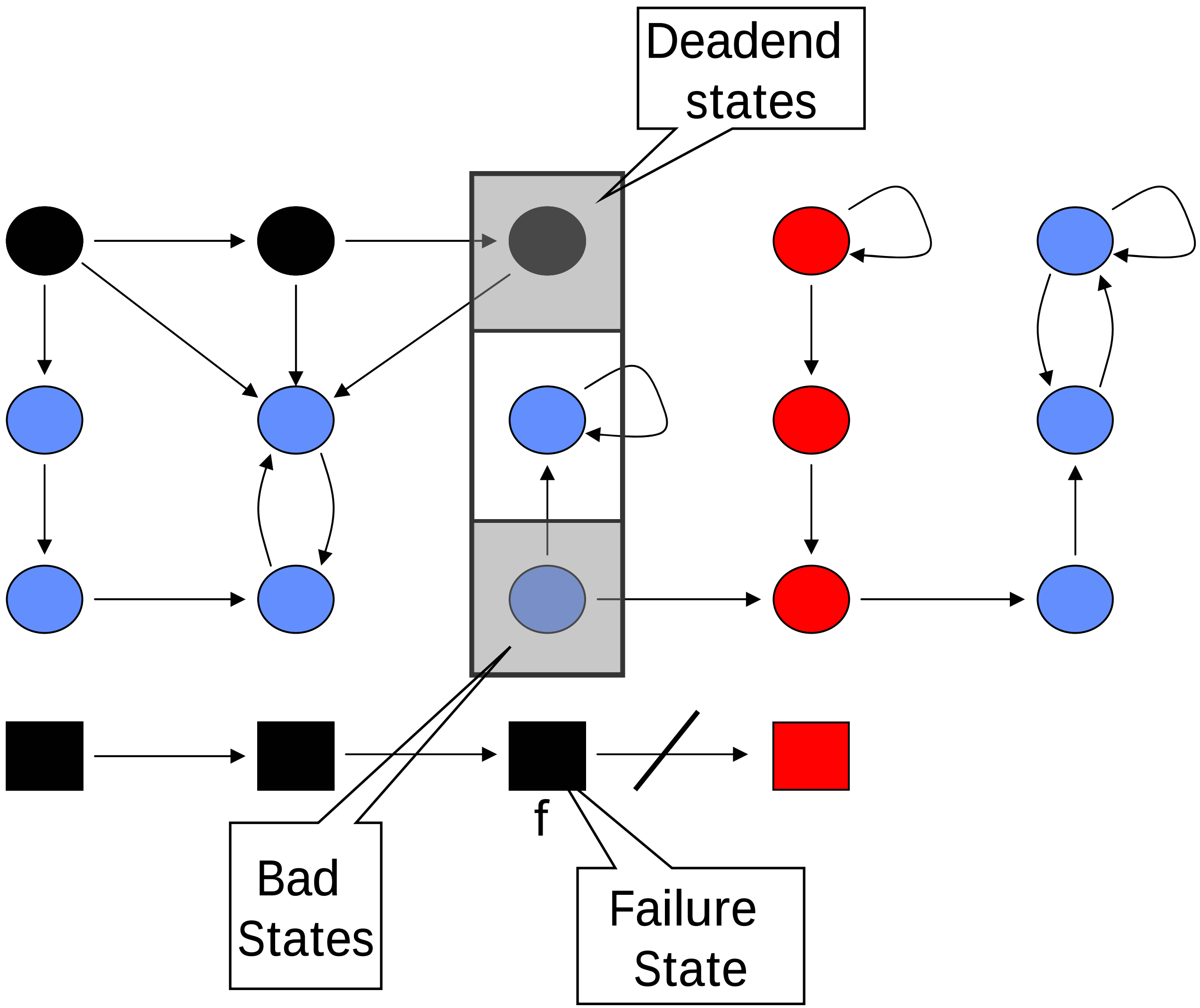
Refinement

Abstraction/refinement with conflict analysis

(Chauhan, Clarke, Kukula, Sapra, Veith, Wang, FMCAD 2002)

- **Simulate counterexample on concrete model with SAT**
- **If the instance is unsatisfiable, analyze conflict**
- **Make visible one of the variables in the clauses that lead to the conflict**

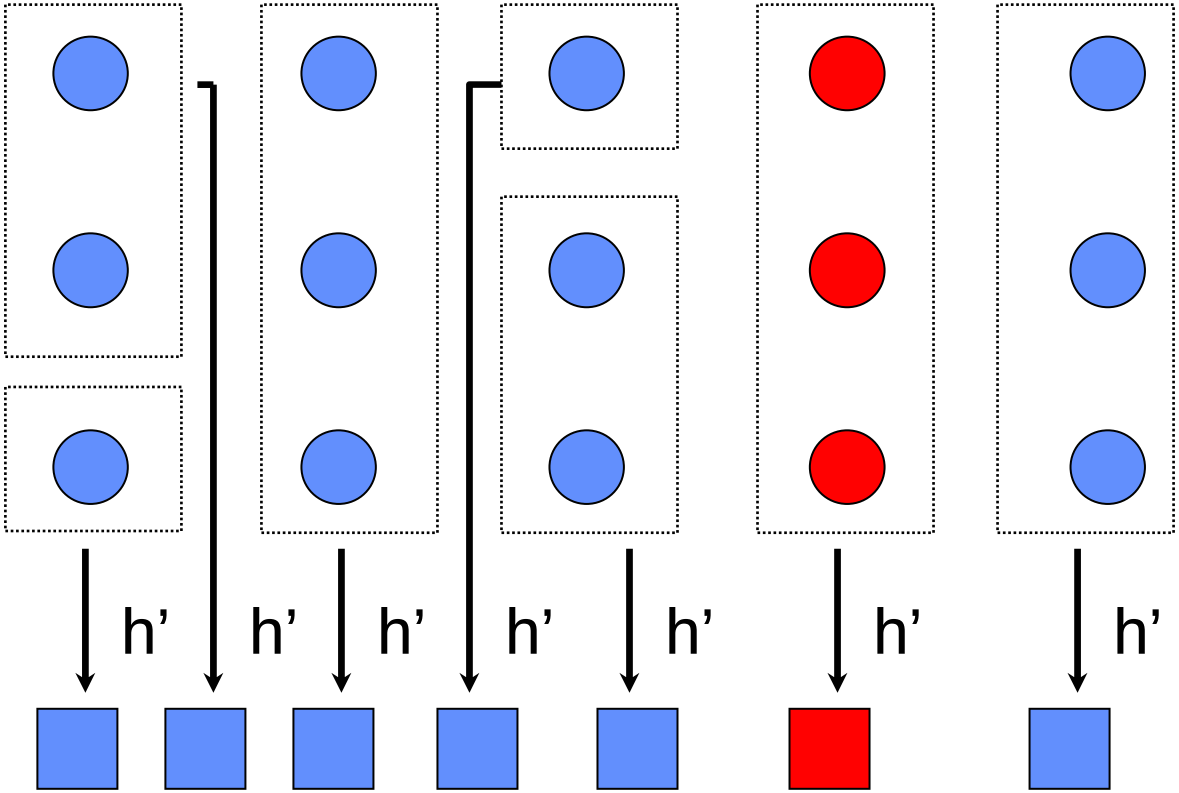
Why spurious counterexample?



Refinement

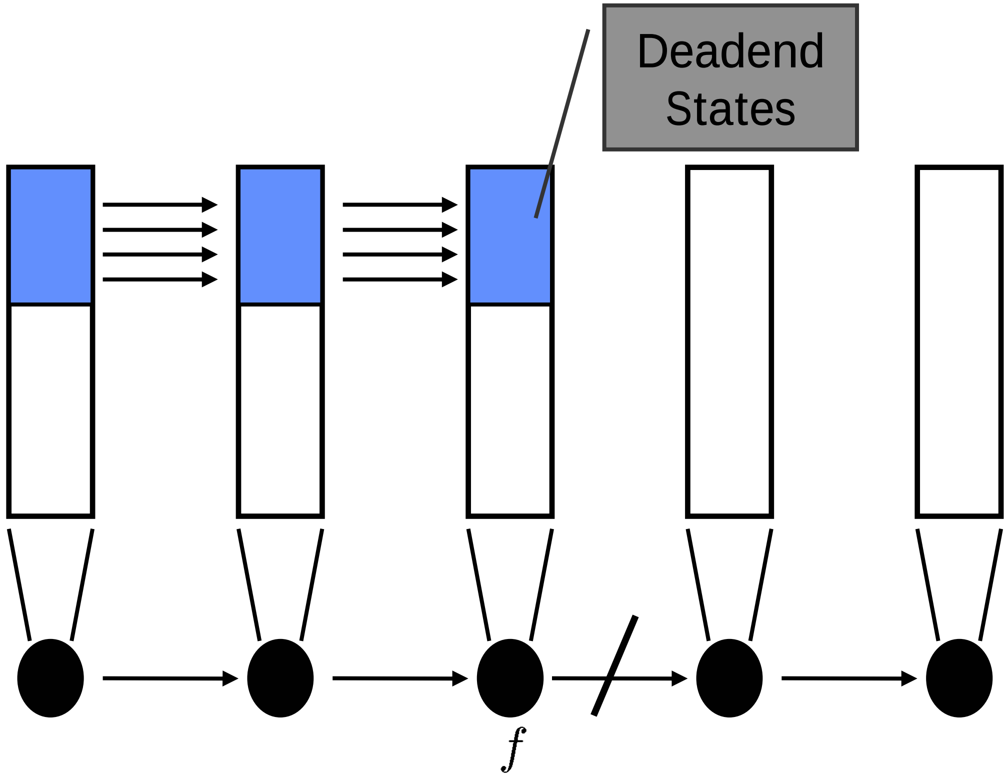
- Problem: Deadend and Bad States are in the same abstract state.
- Solution: Refine abstraction function.
- The sets of Deadend and Bad states should be separated into different abstract states.

Refinement



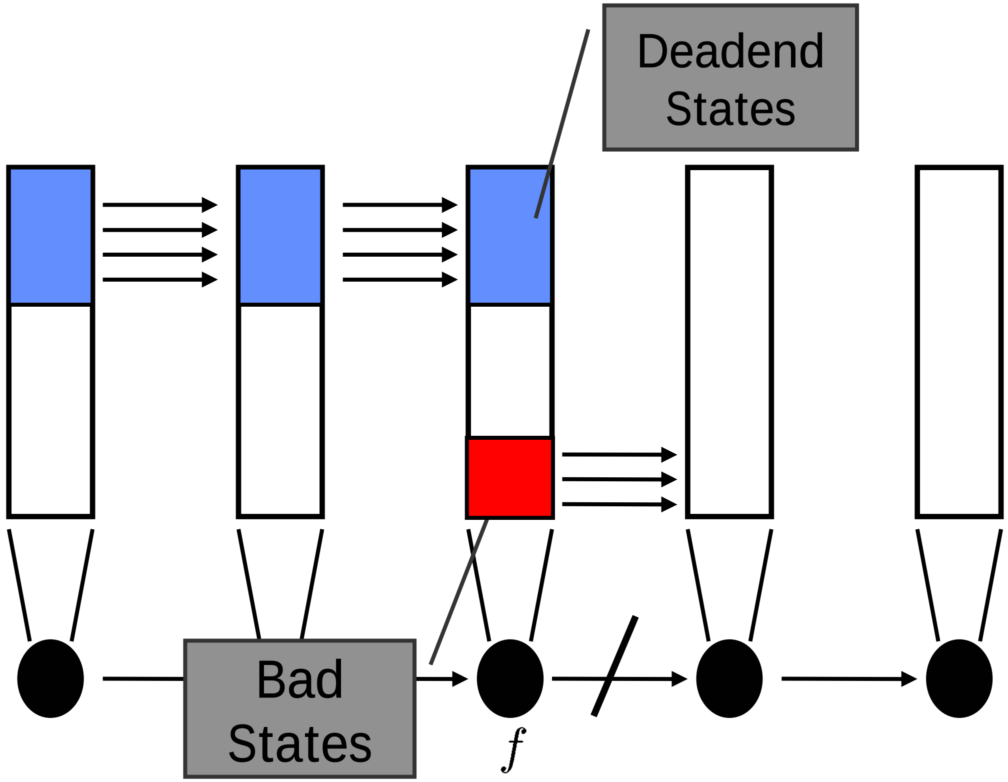
Refinement : h’

Refinement



$$\phi_D = I(s_1) \wedge \bigwedge_{i=1}^{f-1} R(s_i, s_{i+1}) \wedge \bigwedge_{i=1}^{f} \text{visible}(s_i) = c_i$$

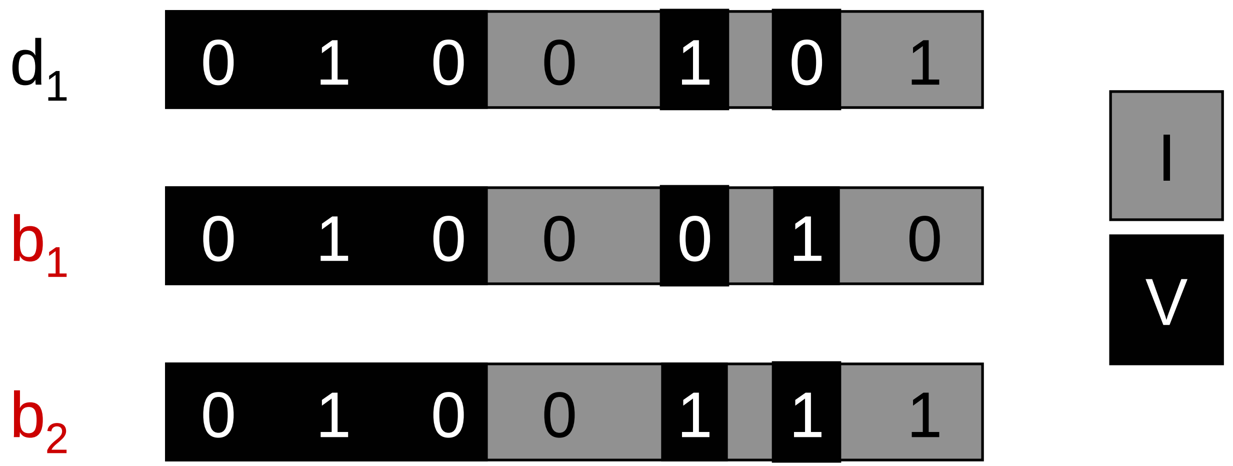
Refinement



$$\phi_B = R(s_f, s_{f+1}) \ \wedge$$

$$\text{visible}(s_f) = c_f \wedge \text{visible}(s_{f+1}) = c_{f+1}$$

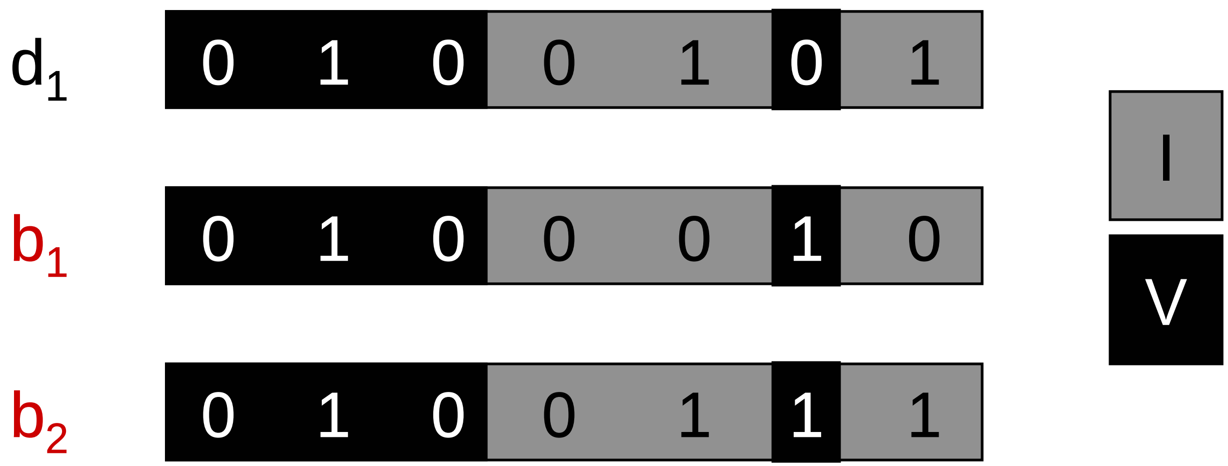
Refinement as Separation



Refinement : Find subset U of I that separates between all pairs of deadend and bad states. Make them visible.

Keep U small !

Refinement as Separation



Refinement : Find subset U of I that separates between all pairs of deadend and bad states. Make them visible.

Keep U small !

Refinement as Separation

The state separation problem

Input: Sets D, B

Output: Minimal U $\in$ I s.t.:

$$\forall\, d \in D,\ \forall\, b \in B,\ \exists u \in U.\ \ d(u) \neq b(u)$$

The refinement h' is obtained by adding U to V.

Two separation methods

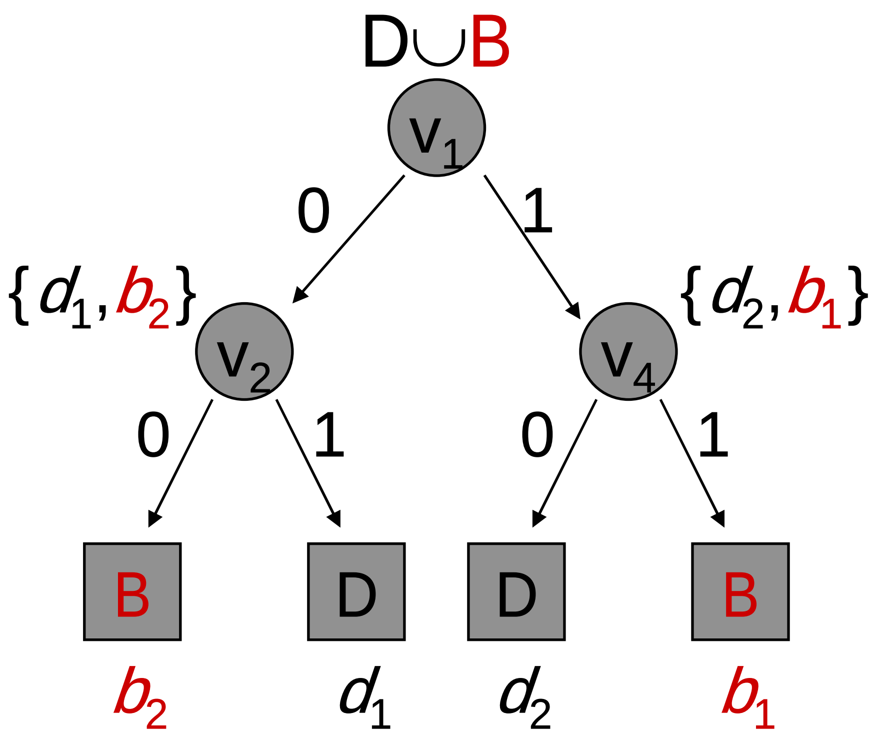
- **ILP-based separation**
  - Minimal separating set.
  - Computationally expensive.

- **Decision Tree Learning based separation.**
  - Not optimal.
  - Polynomial.

Separation with Decision Tree Learning (Example)

Classification:

D	B
$d_1 = (0,1,0,1)$	$b_1 = (1,1,1,1)$
$d_2 = (1,1,1,0)$	$b_2 = (0,0,0,1)$



Separating Set : $\{v_1, v_2, v_4\}$

Separation with 0-1 ILP (Example)

$$d_1 = (0,1,0,1) \qquad b_1 = (1,1,1,1)$$
$$d_2 = (1,1,1,0) \qquad b_2 = (0,0,0,1)$$

Min $\sum_{i=1}^{4} v_i$

subject to:

$$\begin{array}{lll} v_1 + v_3 & \geq 1 & /* \text{ Separating } d_1 \text{ from } b_1 */ \\ v_2 & \geq 1 & /* \text{ Separating } d_1 \text{ from } b_2 */ \\ v_4 & \geq 1 & /* \text{ Separating } d_2 \text{ from } b_1 */ \\ v_1 + v_2 + v_3 + v_4 & \geq 1 & /* \text{ Separating } d_2 \text{ from } b_2 */ \end{array}$$

Separation with 0-1 ILP

$$\text{Min } \sum_{i=1}^{|\mathcal{I}|} v_i$$

$$\text{subject to:} \quad (\forall d \in D)\ (\forall b \in B) \sum_{\substack{1 \le i \le |\mathcal{I}|, \\ d,b \text{ differ at } v_i}} v_i \ge 1$$

- One constraint per pair of states.
- $v_i = 1$ iff v_i is in the separating set.

Refinement as Learning

- **For systems of realistic size**
  - **Not possible to generate D and B.**
  - **Expensive to separate D and B.**

- **Solution:**
  - **Sample D and B**
  - **Infer separating variables from the samples.**

- **The method is still complete:**
  - **Counterexample will eventually be eliminated.**

The CMU CEGAR Tool