Model Checking with Craig Interpolants

Ken McMillan, 2003

2010 CAV Award: “has significantly influenced both academic research and industrial practice, and has dramatically changed the scale of systems that can be analyzed by model checking.”

Kenneth McMillan

Interpolants as Inductive Invariants

- BMC finds bugs (and absence of bugs up to k steps)
- How to Show Correctness?
  - k-induction
  - Interpolants

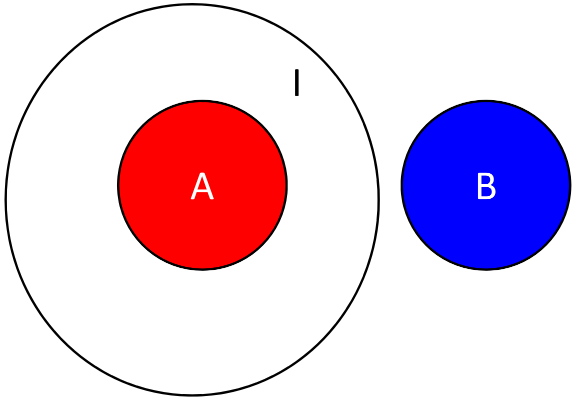
- Find Interpolants I such that
  - States reachable in k steps are in I
  - no bad states are in I
- Interpolants are (good) overapproximation of post-image computation

Interpolant

William Craig, 1957

Definition. Given formulas A, B such that $A \wedge B = \bot$, an interpolant is a formula I such that

1. $A \rightarrow I$
2. $I \wedge B \equiv \bot$
3. I only uses symbols that occur both in A and in B



Example. Let

$A = (a_1 \vee \neg a_2) \wedge (\neg a_1 \vee \neg a_3) \wedge a_2$,

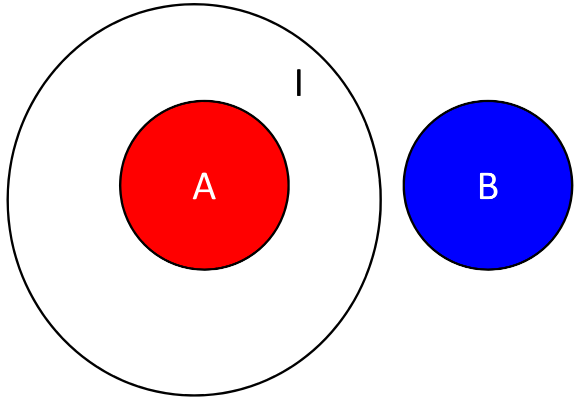
$\mathrm{B} = (\neg a_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \neg a_4$.

Interpolant

William Craig, 1957

Definition. Given formulas A, B such that $A \wedge B = \bot$, an interpolant is a formula I such that

1. $A \rightarrow I$
2. $I \wedge B \equiv \bot$
3. I only uses symbols that occur both in A and in B



Example. Let

$A = (a_1 \vee \neg a_2) \wedge (\neg a_1 \vee \neg a_3) \wedge a_2$,

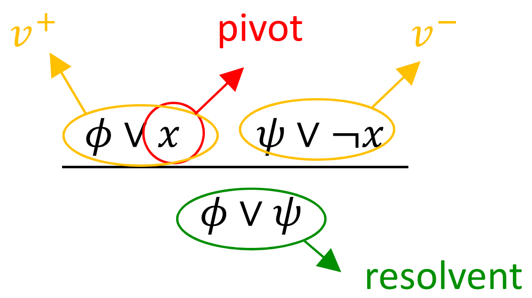
$\mathrm{B} = (\neg a_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \neg a_4$.

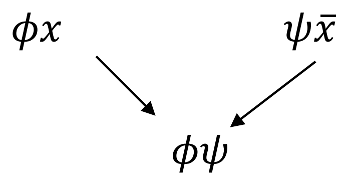
$A \wedge B$ is not satisfiable.

$\neg a_3 \wedge a_2$ is an interpolant:

1. $\big((a_1 \vee \neg a_2) \wedge (\neg a_1 \vee \neg a_3) \wedge a_2\big) \rightarrow (\neg a_3 \wedge a_2)$
2. $(\neg a_3 \wedge a_2) \wedge \big((\neg a_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \neg a_4\big) \equiv \bot$
3. a_2 and a_3 occur in A and in B

Resolution (Chap 9)





Interpolants from Resolution Proofs

For clause C, $C|B$ is obtained by removing literals not in B

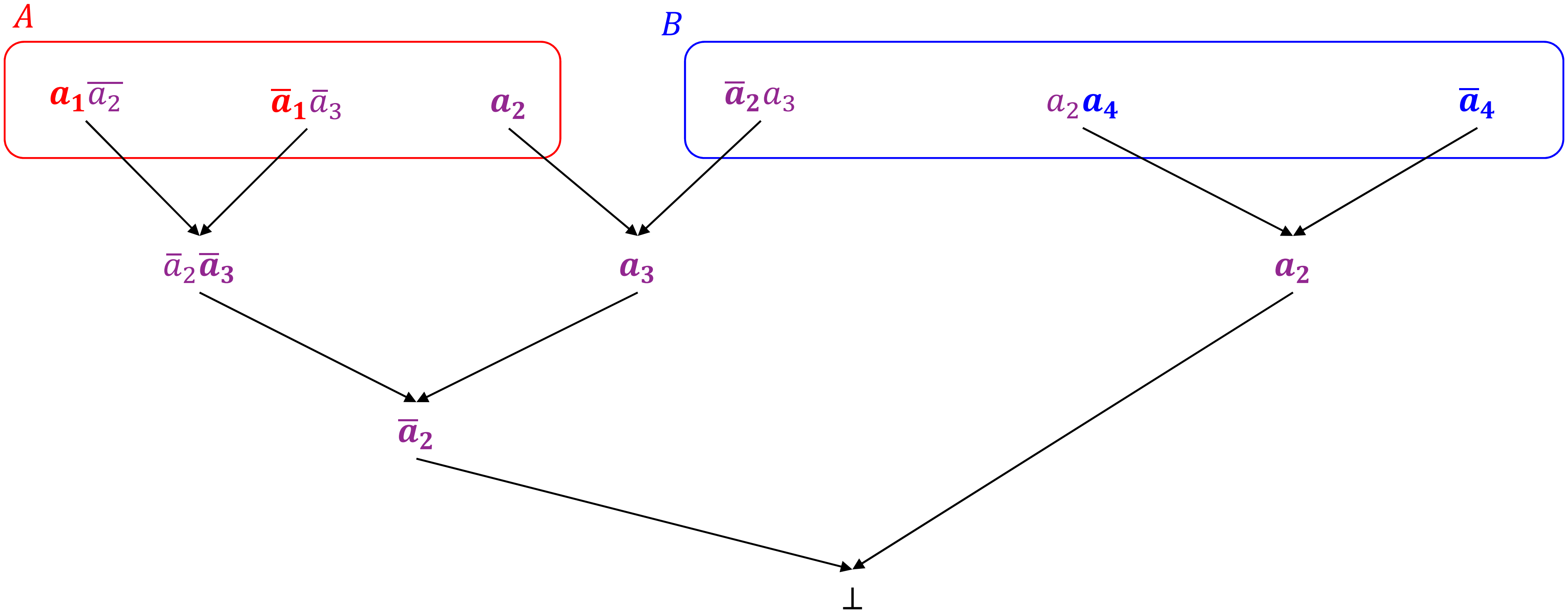
Algorithm. Go through resolution proof top-down.

1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$

Interpolation Example

Algorithm. Go through resolution proof top-down.

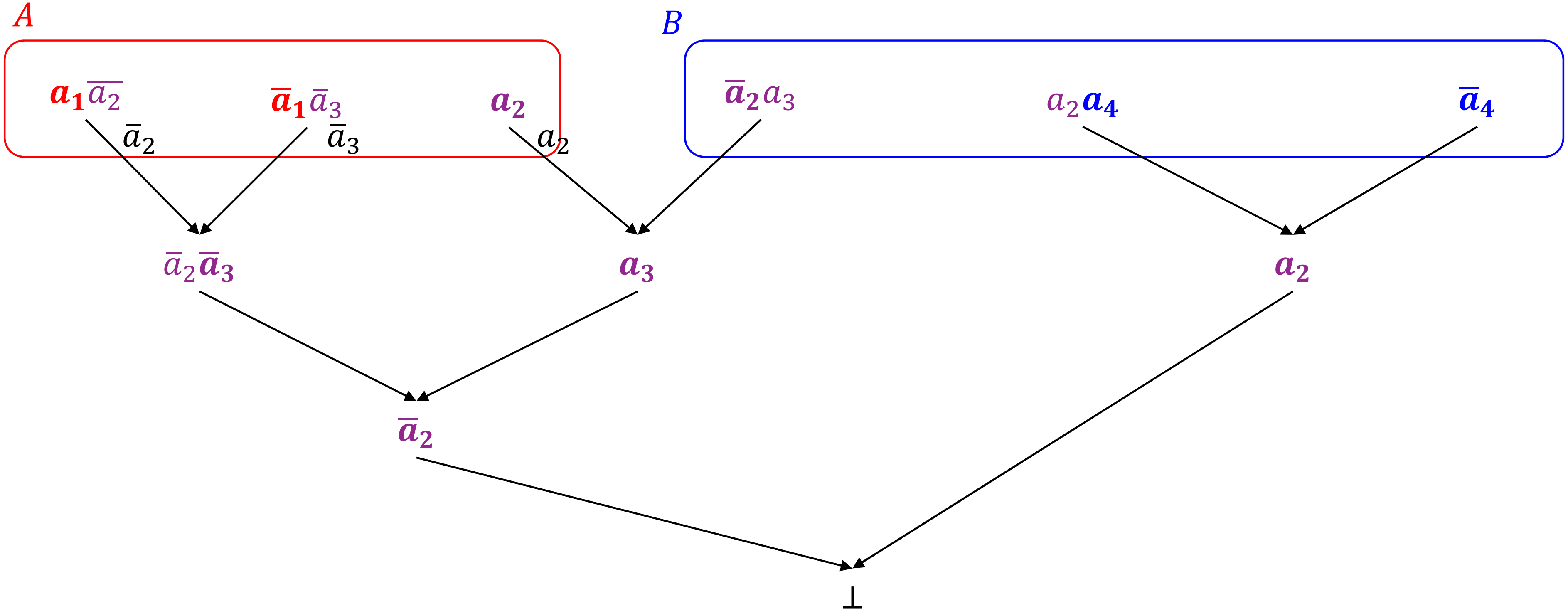
1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$



Interpolation Example

Algorithm. Go through resolution proof top-down.

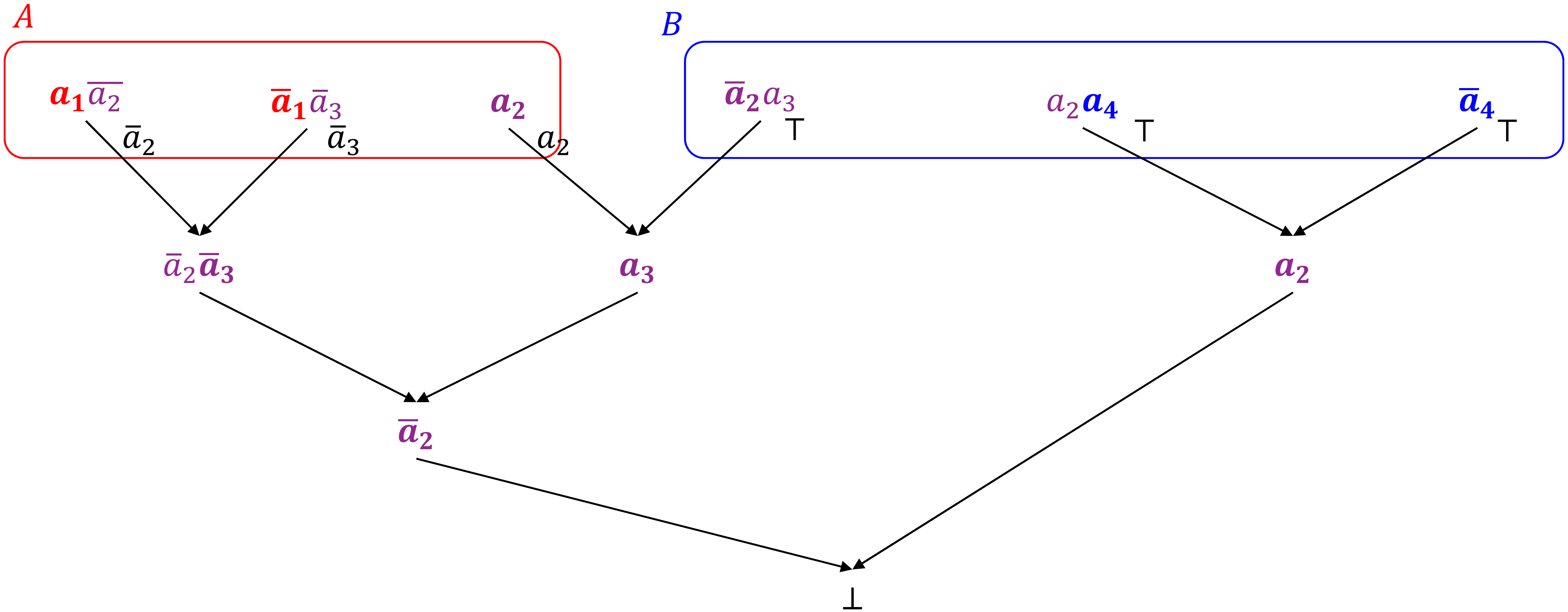
1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$



Interpolation Example

Algorithm. Go through resolution proof top-down.

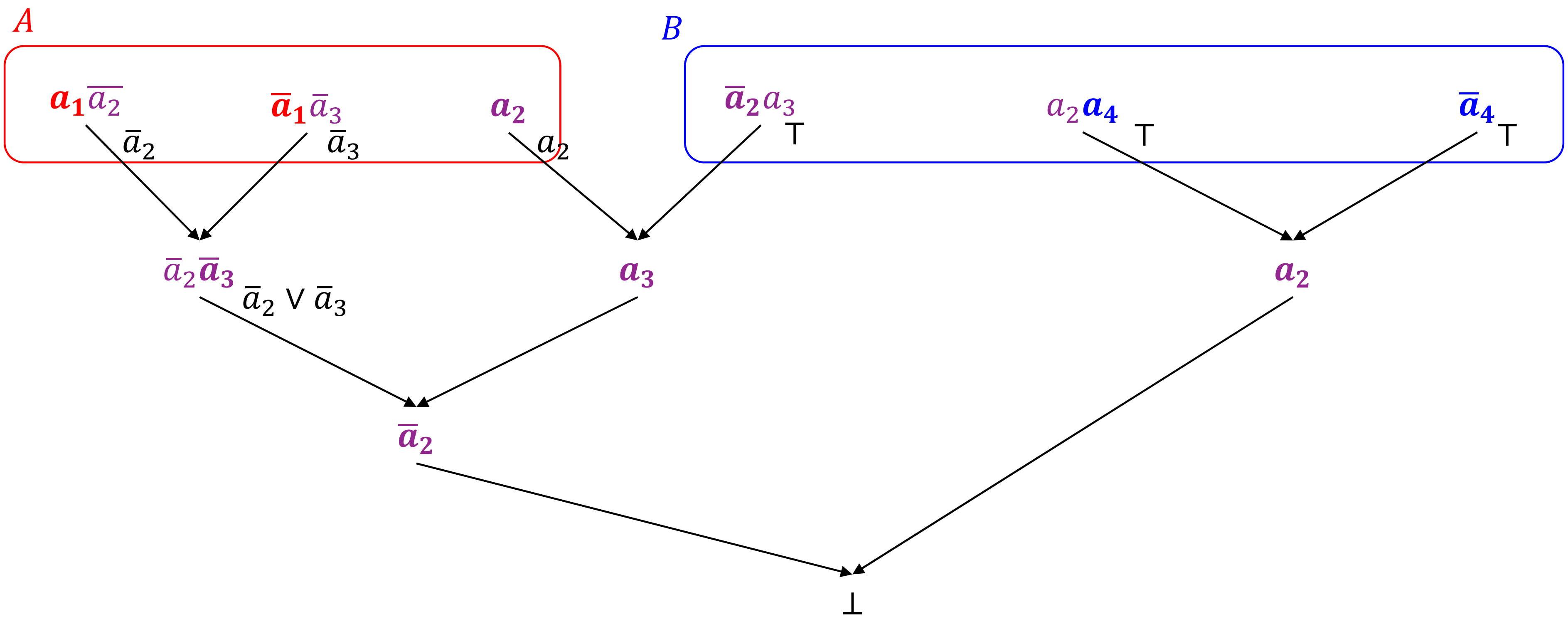
1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$



Interpolation Example

Algorithm. Go through resolution proof top-down.

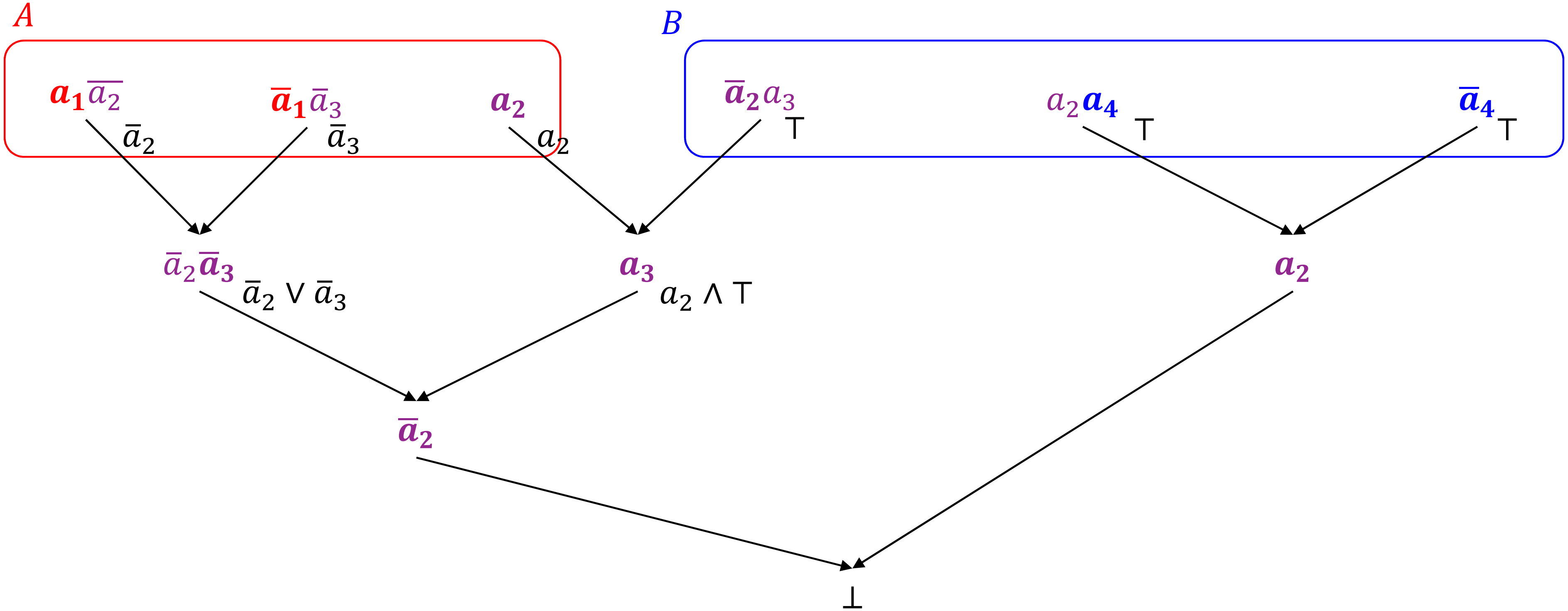
1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^{+}) \wedge Itp(v^{-})$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^{+}) \vee Itp(v^{-})$



Interpolation Example

Algorithm. Go through resolution proof top-down.

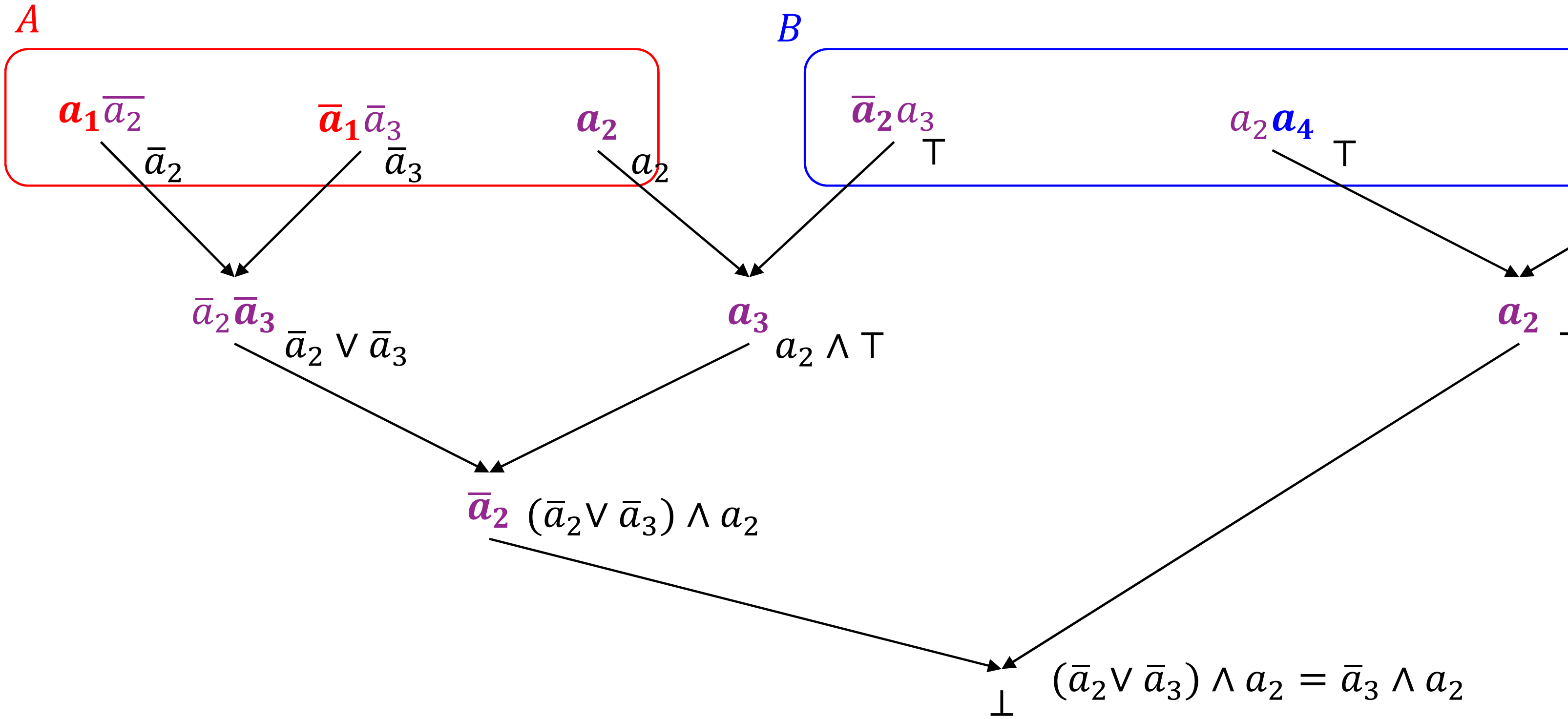
1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$



Interpolation Example

Algorithm. Go through resolution proof top-down.

1. If leaf v is labeled $C \in A$, then $Itp(v) = C|B$
2. If leaf v is labeled $C \in B$, then $Itp(v) = \top$
3. If node v has pivot variable $x \in B$ then $Itp(v) = Itp(v^+) \wedge Itp(v^-)$
4. If node v has pivot variable $x \notin B$ then $Itp(v) = Itp(v^+) \vee Itp(v^-)$



Computation with Overapproximations

BMC to prove p: $S_0(s_0) \wedge \bigwedge_{i=0}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$.

Suppose $\mathrm{R} \rightarrow R'$

What does this do?

$$S_0(s_0) \wedge \bigwedge_{i=0}^{k-1} R'(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$$

Reachability Checking with Interpolation

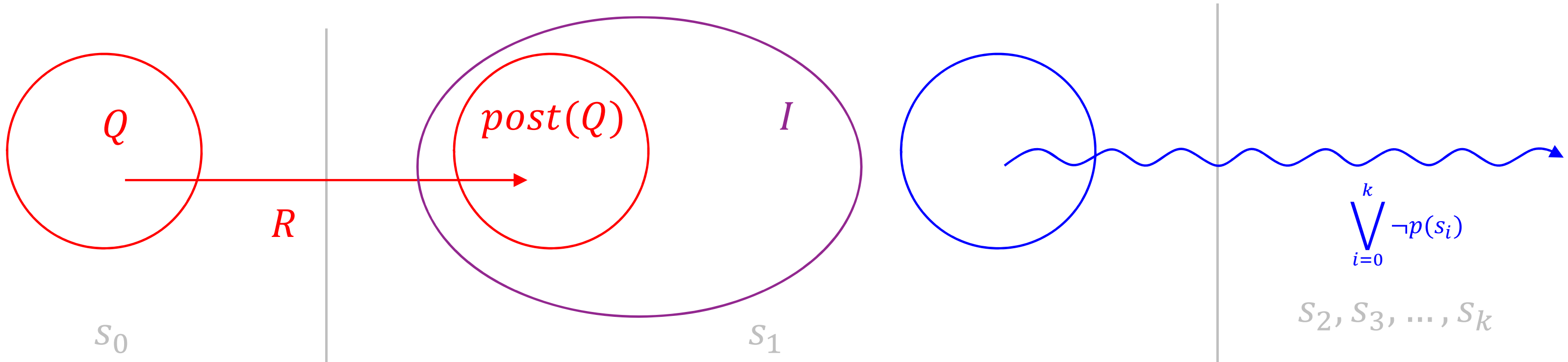
Recall BMC check for $\neg\mathbf{AG}p$:

$S_0(s_0) \wedge \bigwedge_{i=0}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$.

Start from Q such that $Q \vDash p$

$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge \bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i)$.

Suppose ϕ unsatisfiable, $I(s_1)$ is an interpolant



Reachability Checking with Interpolation

Recall BMC check for $\neg \mathbf{AG}p$:

$S_0(s_0) \wedge \bigwedge_{i=0}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$.

Start from Q such that $Q \vDash p$

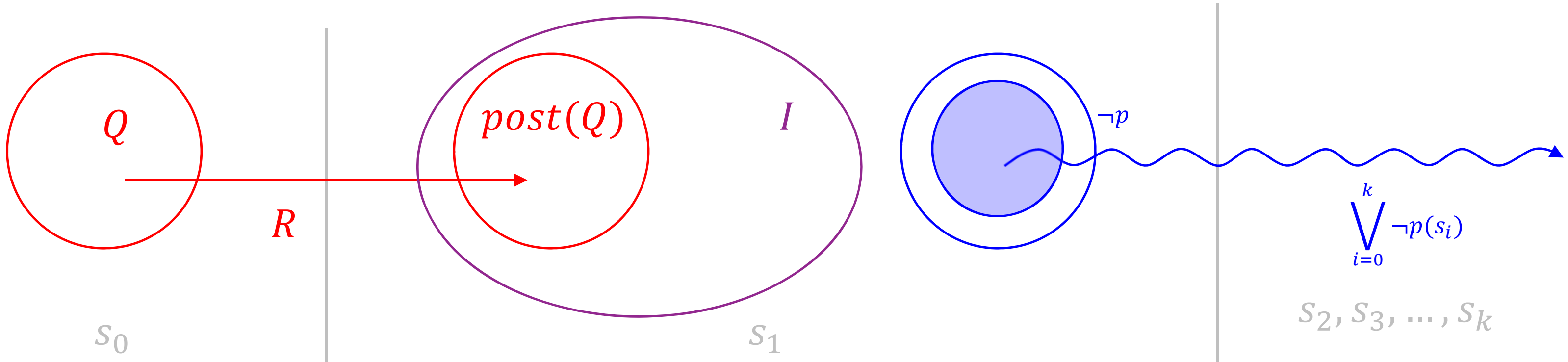
$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge \bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i)$.

Suppose ϕ unsatisfiable, $I(s_1)$ is an interpolant.

Note 1: $\neg p(s_1) \rightarrow B$

so $I(s_1) \wedge \neg p(s_1) = \bot$

Note 2: $I \supseteq post(Q)$



Interpolant Reachability Idea

$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge \bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg p(s_i)$.

1. Start with $Q = S_0$
2. If ϕ not satisfiable, set Q to Q ∪ I
3. If Q remains unchanged, p **is not reachable** *(Interpolants are approximation to post-image),* otherwise goto 2
4. If ϕ is satisfiable and $Q = S_0$, $\neg p$ is **reachable**
5. If ϕ is satisfiable and $Q \neq S_0$, increase k to increase precision of approximation, goto 1.

Procedure terminates when k is diameter of system (or earlier!)

Algorithm

```
procedure CraigReachability(model M, p ∈ AP)
   if S_0 ∧ ¬p is SAT return "M ⊭ AG p ";
   k := 1;
   Q := S_0(s_0);
   while true do
      A := Q(s_0) ∧ R(s_0, s_1);
      B := ⋁_{i=1}^{k-1} R(s_i, s_{i+1}) ∧ ⋁_{i=1}^{k} ¬p(s_i);
         if A ∧ B is SAT then                          // ¬p can be reached from Q
               if Q = S_0 then return "M ⊭ AG p";      // ¬p can be reached from S_0
               Increase k                              // Not sure if path to ¬p is real. Increase precision
               Q := S_0(s_0);
            else
               compute interpolant I for A and B;
               If I(s_0) == Q then return "M ⊨ AG p";  // Reached the fixpoint of overapproximated reachability?
               Q := Q ∨ I(s_0);                         // Another step of overapproximated reachability?
            end if
   end while
end procedure
```

if $A \wedge B$ is SAT **then**

if $Q = S_0$ then return "$M \nvDash \mathrm{AG}\, p$";

increase k

Q := $S_0(s_0)$;

else

compute interpolant I for A and B;

if $I(s_0) \rightarrow Q$ then return "$M \vDash \mathrm{AG}\, p$";

Q:= $Q \vee I(s_0)$;

end if

10.4.4 Correctness

If CraigReachability returns "$M \vDash AG\ p$" then $M \vDash AG\ p$

Let Q_i denote Q at iteration i. For all i, $Q_i \leftarrow postimage^i(Q_0)$. If $I \rightarrow Q_i$, we have reached a fixed point $Q^* = Q_i$ so $Q^* \leftarrow postimage^*(Q_0)$. Now because $Q_i \wedge \neg p = \bot$, we have $postimage^*(Q_0) \wedge \neg p = \bot$.

If CraigReachability returns "$M \nvDash AG\ p$" then $M \nvDash AG\ p$

$A \wedge B$ encodes a path from Q_0 to $\neg p$.

CraigReachability terminates

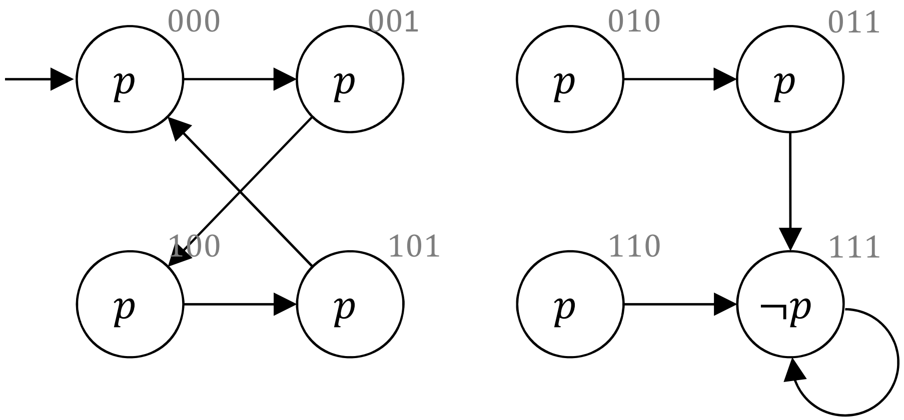
Note that k increases.

If $M \nvDash AG\ p$, there is a path of length l to $\neg p$ and we will find it when $l = k$.

Suppose $M \vDash AG\ p$. If k is the diameter of the graph, no I and thus no Q_i can contain a state that reaches $\neg p$. Thus, $A \wedge B$ is never SAT and the algorithm terminates because the Q_i cannot grow forever.

$x_1 x_2 x_3$

Example $\mathbf{AG}\ \boldsymbol{p}$



$$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge$$
$$\bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i).$$

if $A \wedge B$ is SAT **then**

if $Q = S_0$ then return "$M \nvDash \mathrm{AG}\, p$";

increase k

$Q := S_0(s_0)$;

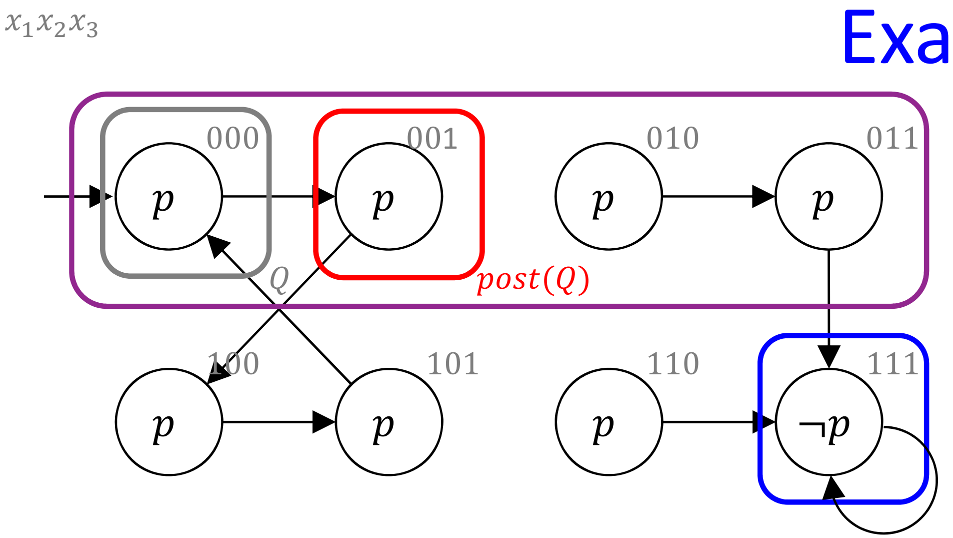
else

compute interpolant I for A and B;

if $I(s_0) \rightarrow Q$ then return "$M \vDash \mathrm{AG}\, p$";

$Q := Q \vee I(s_0)$;

Example $\mathbf{AG}\ \boldsymbol{p}$



$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge$
$\bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i)$.

$\boldsymbol{k}$ **=1.**

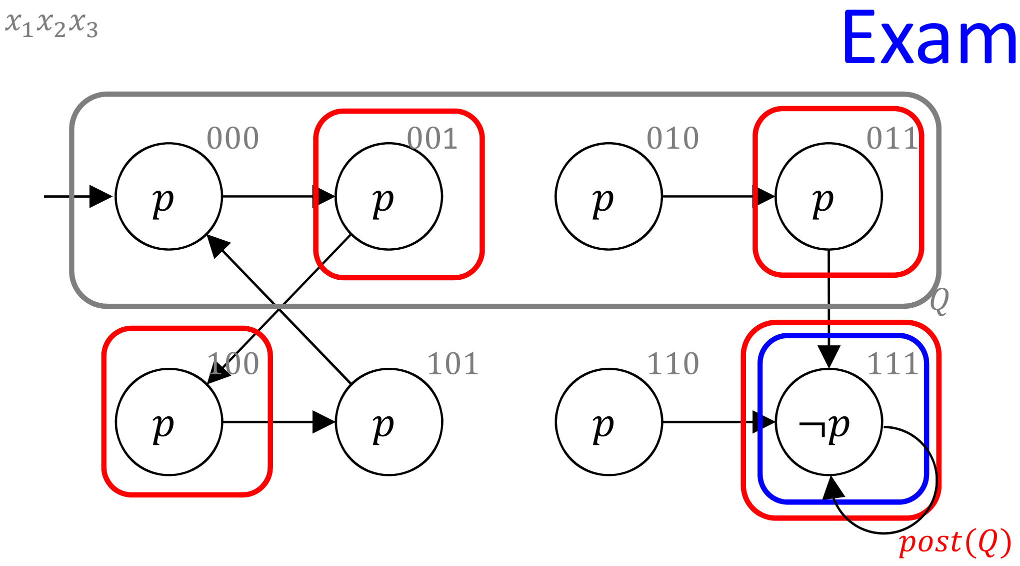
$Q = \neg x_1 \wedge \neg x_2 \wedge \neg x_3 = \{000\}$.

ϕ is UNSAT

Invariant checks first bit: $I = \neg x_1$

if $A \wedge B$ is SAT **then**
  if $Q = S_0$ then return "$M \not\models$ AG p";
  increase k
  $Q := S_0(s_0)$;
else
  compute interpolant I for A and B;
  if $I(s_0) \rightarrow Q$ then return "$M \models$ AG p";
  $Q := Q \vee I(s_0)$;

Example AG p



$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge$
$\bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i)$.

***k* =1.**

$Q = \neg x_1 = \{000,001,010,011\}$.

ϕ is SAT

if $A \wedge B$ is SAT **then**

if $Q = S_0$ then return "$M \not\models$ AG p";

increase k

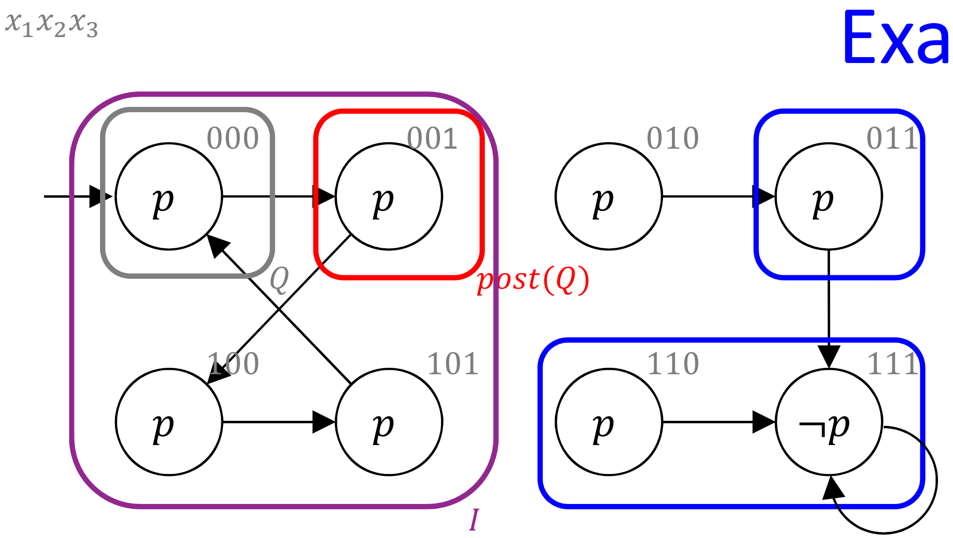
$Q := S_0(s_0)$;

else

compute interpolant I for A and B;

if $I(s_0) \rightarrow Q$ then return "$M \models$ AG p";

$Q := Q \vee I(s_0)$;



Example $\mathbf{AG}\ \boldsymbol{p}$

$\phi = Q(s_0) \wedge R(s_0, s_1) \wedge$
$\bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \wedge \bigvee_{i=1}^{k} \neg p(s_i)$.

$\boldsymbol{k = 2}$.

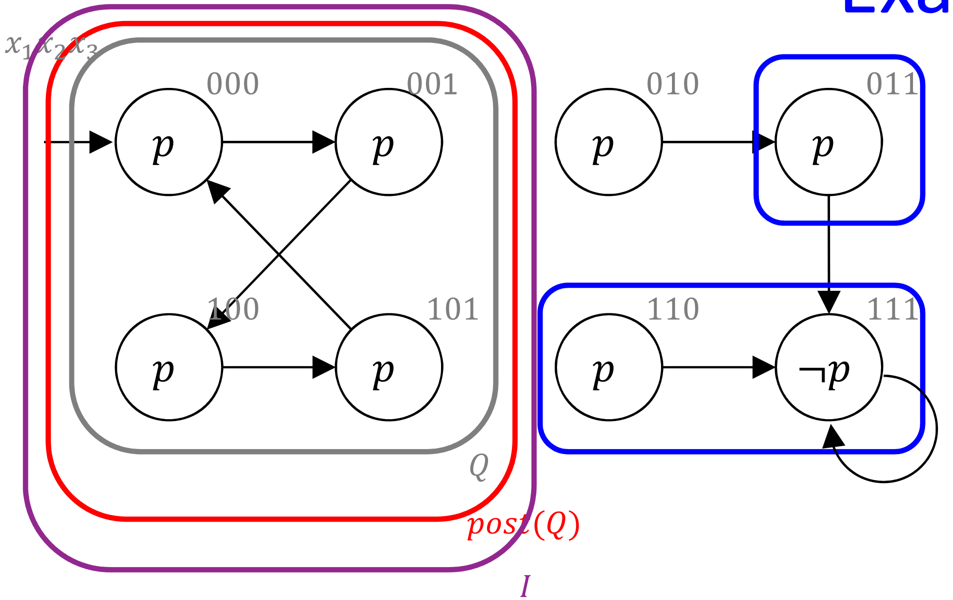
$Q = \neg x_1 \wedge \neg x_2 \wedge \neg x_3 = \{000\}$.

ϕ is UNSAT

Invariant checks 2nd bit: $I = \neg x_1$

if $A \wedge B$ is SAT **then**
  if $Q = S_0$ then return "$M \nvDash$ AG p";
  increase k
  $Q := S_0(s_0)$;
else
  compute interpolant I for A and B;
  if $I(s_0) \rightarrow Q$ then return "$M \vDash$ AG p";
  $Q := Q \vee I(s_0)$;

Example $\mathbf{AG}\ \boldsymbol{p}$



if $A \land B$ is SAT **then**

if $Q = S_0$ then return "$M \nvDash \text{AG}\ p$";

increase k

$Q := S_0(s_0)$;

else

compute interpolant I for A and B;

if $I(s_0) \to Q$ then return "$M \vDash \text{AG}\ p$";

$Q := Q \lor I(s_0)$;

$\phi = Q(s_0) \land R(s_0, s_1) \land \bigwedge_{i=1}^{k-1} R(s_i, s_{i+1}) \land \bigvee_{i=1}^{k} \neg p(s_i)$.

$\boldsymbol{k = 2.}$

$Q = \neg x_2 = \{000, 001, 100, 101\}$

ϕ is UNSAT

$\boldsymbol{I = \neg x_2 = Q.}$

Algorithm terminates.

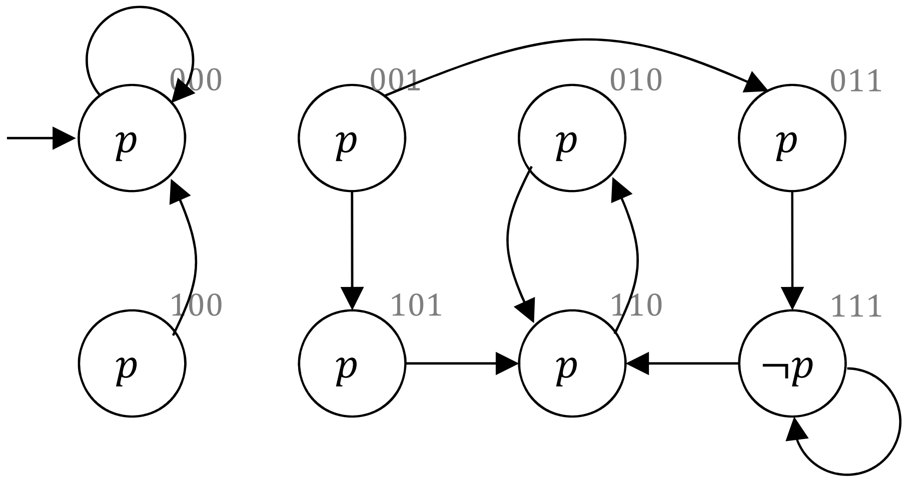
How Did I Pick the Interpolants?

What I did

- Start with $\mathrm{A} = \ postimg(Q)$
- Perform each of the following steps
    1. Can I throw away x_3? (Is $(\exists x_3 . A) \cap B = \emptyset$?) If yes, $A \ \coloneqq \exists x_3 . A$
    2. Can I throw away x_2? If yes, $A \ \coloneqq \exists x_2 . A$
    3. Can I throw away x_1? If yes, $A \ \coloneqq \exists x_1 . A$

(This hack only works because the postimg(Q) is a state or a cube!)

Homework Draft



Next Week

2PM i13: Property-Directed Reachability

Homework!