Interpolation-Sequence Based Model Checking

Yakir Vizel and Orna Grumberg

FMCAD 2009

Verification course, June 2015

Inspired by:

- forward reachability analysis

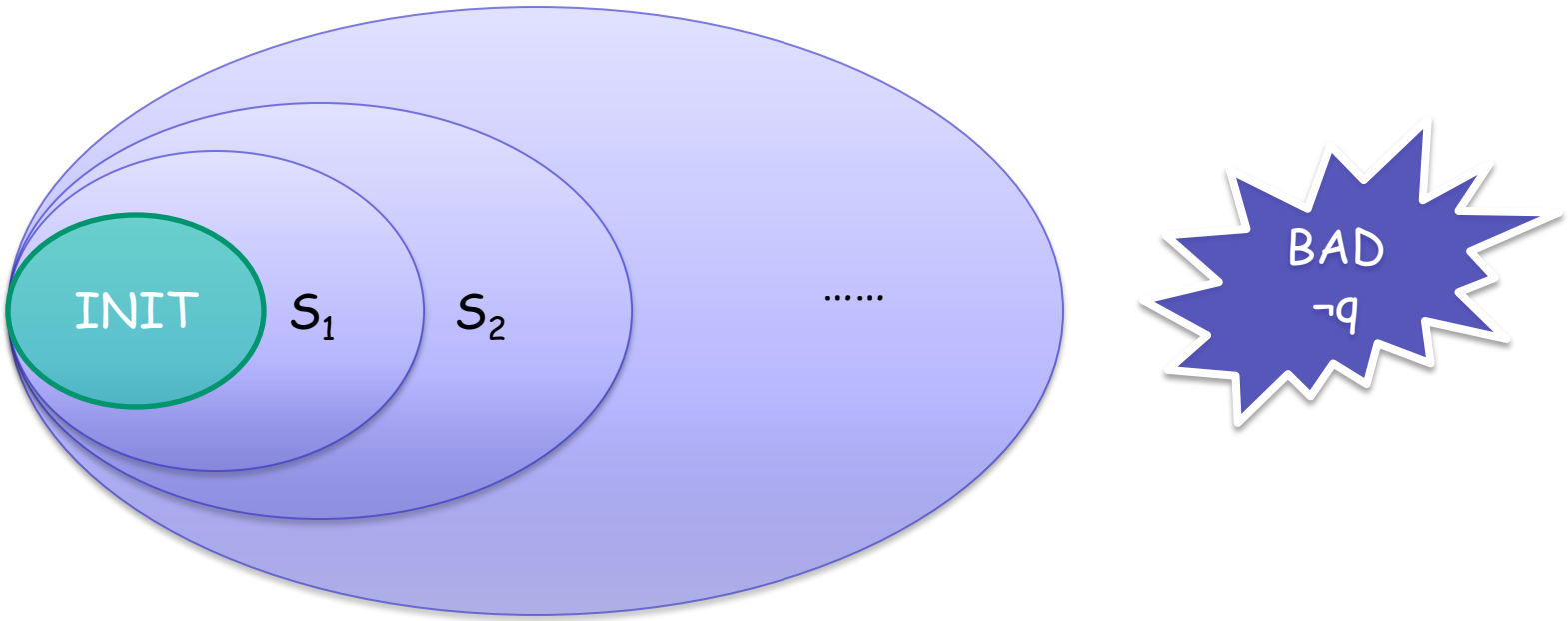
Combines:

- Bounded Model Checking
- Interpolation-sequence

Obtains:

- SAT-based model checking algorithm for full verification

Forward Reachability Analysis



Forward reachability analysis

- S_j is the set of states reachable from some initial state in j steps
- termination when
  - either a bad state satisfying $\neg q$ is found
  - or a fixpoint is reached:

    $S_j \subseteq \cup_{i=1,j-1} S_i$

SAT-based model checking:

A solution for the state explosion problem

Main idea

- Translate the model and the specification to propositional formulas
- Use efficient tools (SAT solvers) for solving the satisfiability problem

Bounded Model Checking (BMC) for checking AGp

- Unwind the model for k levels, i.e., construct all computations of length k
- If a state satisfying $\neg p$ is encountered, produce a counterexample;
  Otherwise, increase k

[BCCZ 99]

Bounded Model Checking

- Does the system have a counterexample of length k?

$$INIT(V_0) \wedge \neg p(V_0)$$

$$INIT(V_0) \wedge T(V_0, V_1) \wedge \neg p(V_1)$$

$$INIT(V_0) \wedge T(V_0, V_1) \wedge T(V_1, V_2) \wedge \neg p(V_2)$$

$$\vdots$$

$$INIT(V_0) \wedge T(V_0, V_1) \wedge T(V_1, V_2) \wedge \ldots \wedge T(V_{k-1}, V_k) \wedge \neg p(V_k)$$

Bounded Model Checking

Terminates

- with a counterexample or
- with time- or memory-out

The method is suitable for **falsification**, not verification

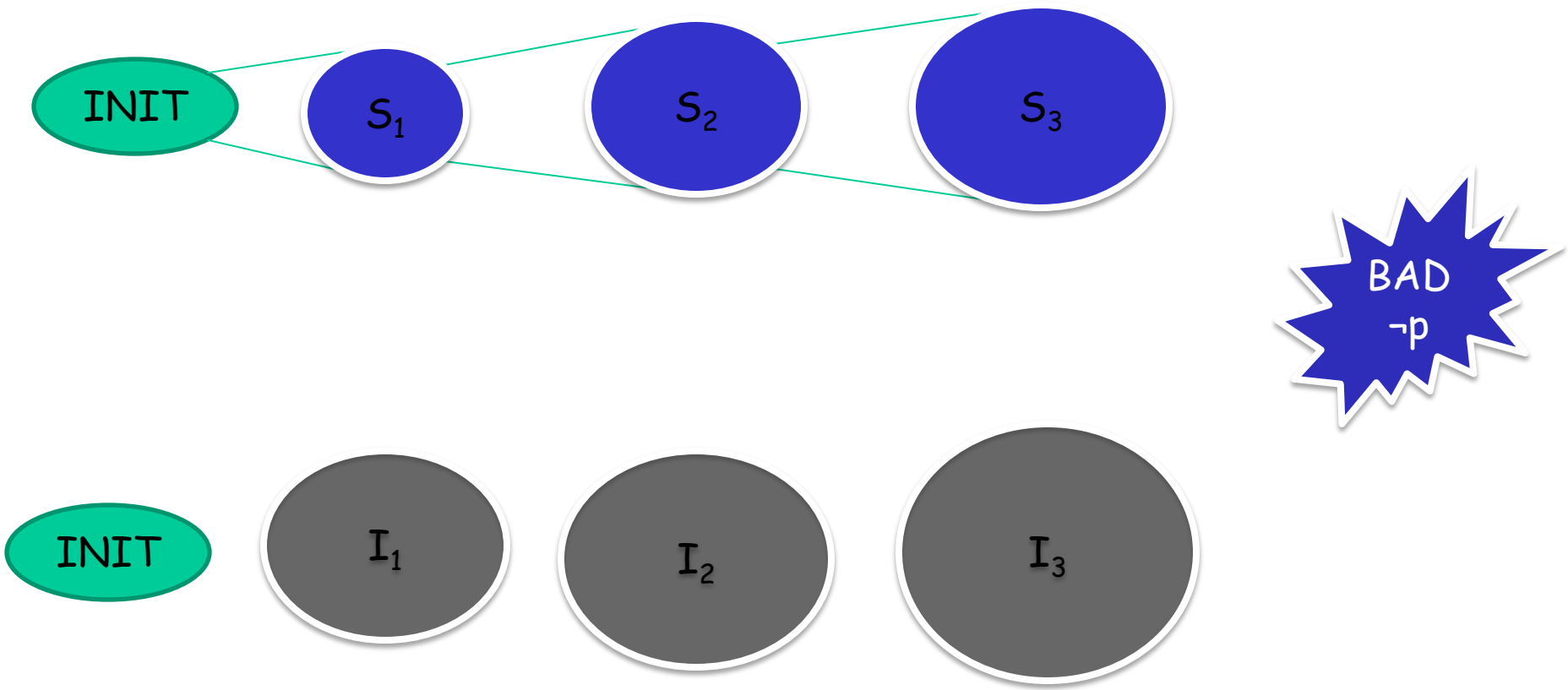
Verification with SAT solvers

Two successful methods for SAT-based verification are based on:

- Interpolation [McMillan 03]
- IC3 [Bradley 11]

we present two methods for enhancing interpolation and IC3 model checking

A Bit of Intuition



Interpolation

[Craig 57]

- If $A \wedge B$ = false, there exists an *interpolant* I for (A,B) such that:

$$A \Rightarrow I$$

$$I \wedge B = \text{false}$$

I refers only to common variables of A,B

Interpolation in the context of model checking

- Given the following BMC formula φ^k

$$\overbrace{INIT(V_0) \wedge T(V_0, V_1)}^{A} \wedge \overbrace{T(V_1, V_2) \wedge \ldots \wedge T(V_{k-1}, V_k) \wedge \neg p(V_k)}^{B}$$

$$\Downarrow$$

$$\mathbf{I}$$

$$A \Rightarrow I$$

$$I \wedge B \equiv false$$

I is over the common variables of A and B, i.e V_1

Interpolation in the context of model checking

- I is over V_1
- $A \Rightarrow I$
  - I over-approximates the set S_1
- $I \wedge B \equiv$ false
  - States in I cannot reach a bug in k-1 steps

Interpolation-Sequence

- The same BMC formula partitioned in a different manner:

$$\overbrace{INIT(V_0) \wedge T(V_0,V_1)}^{A_1} \wedge \overbrace{T(V_1,V_2)}^{A_2} \wedge \overbrace{T(V_2,V_3)}^{A_3} \wedge \ldots \wedge \overbrace{T(V_{k-1},V_k)}^{A_k} \wedge \overbrace{\neg p(V_k)}^{A_{k+1}}$$

$$\Downarrow \qquad \Downarrow \qquad \Downarrow \qquad \Downarrow \qquad \Downarrow$$

$$I_1 \qquad I_2 \qquad I_3 \quad I_{k-1} \qquad I_k$$

$$I_0 = true, I_{k+1} = false$$

$$I_{j-1} \wedge A_j \Rightarrow I_j$$

I_j is over the common variables of $A_1,\ldots,A_j$ and $A_{j+1},\ldots,A_{k+1}$, i.e V_j

Interpolation-Sequence

- I_j – over-approximation of the set of states reachable in j steps

- $I_k \wedge A_{k+1} \Rightarrow$ false
  the states in I_k do not violate p

Interpolation-Sequence

- Can easily be computed in the same way a single interpolation is computed:

- For $1 \leq j < n$
  - $A(j) = A_1 \wedge \ldots \wedge A_j$
  - $B(j) = A_{j+1} \wedge \ldots \wedge A_n$
  - I_j is the interpolant for the pair (A(j), B(j))

Combining Interpolation-Sequence and BMC

- Uses BMC for bug finding
- Uses Interpolation-sequence for computing over-approximation of sets S_j of reachable states

Combining Interpolation-Sequence and BMC

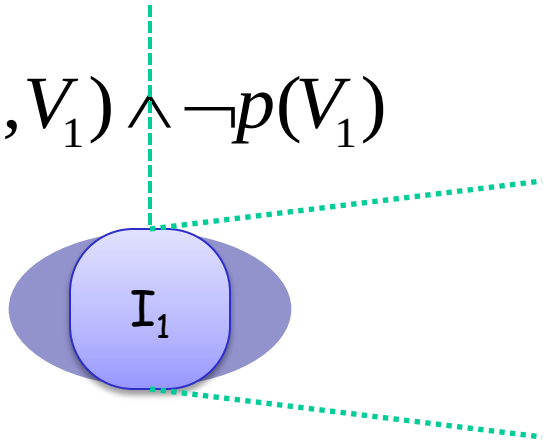
Always terminates

- either when BMC finds a bug:
  $M \not\models AGp$

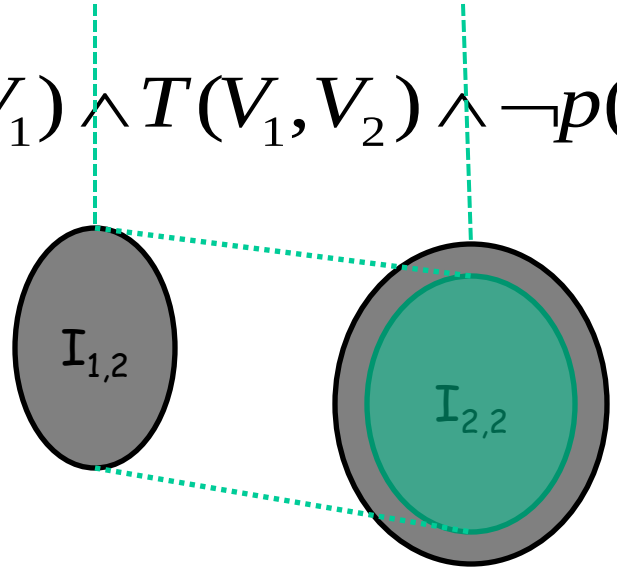
- or when all reachable states has been found:
  $M \models AGp$

Using Interpolation-Sequence

$$INIT(V_0) \wedge T(V_0, V_1) \wedge \neg p(V_1)$$



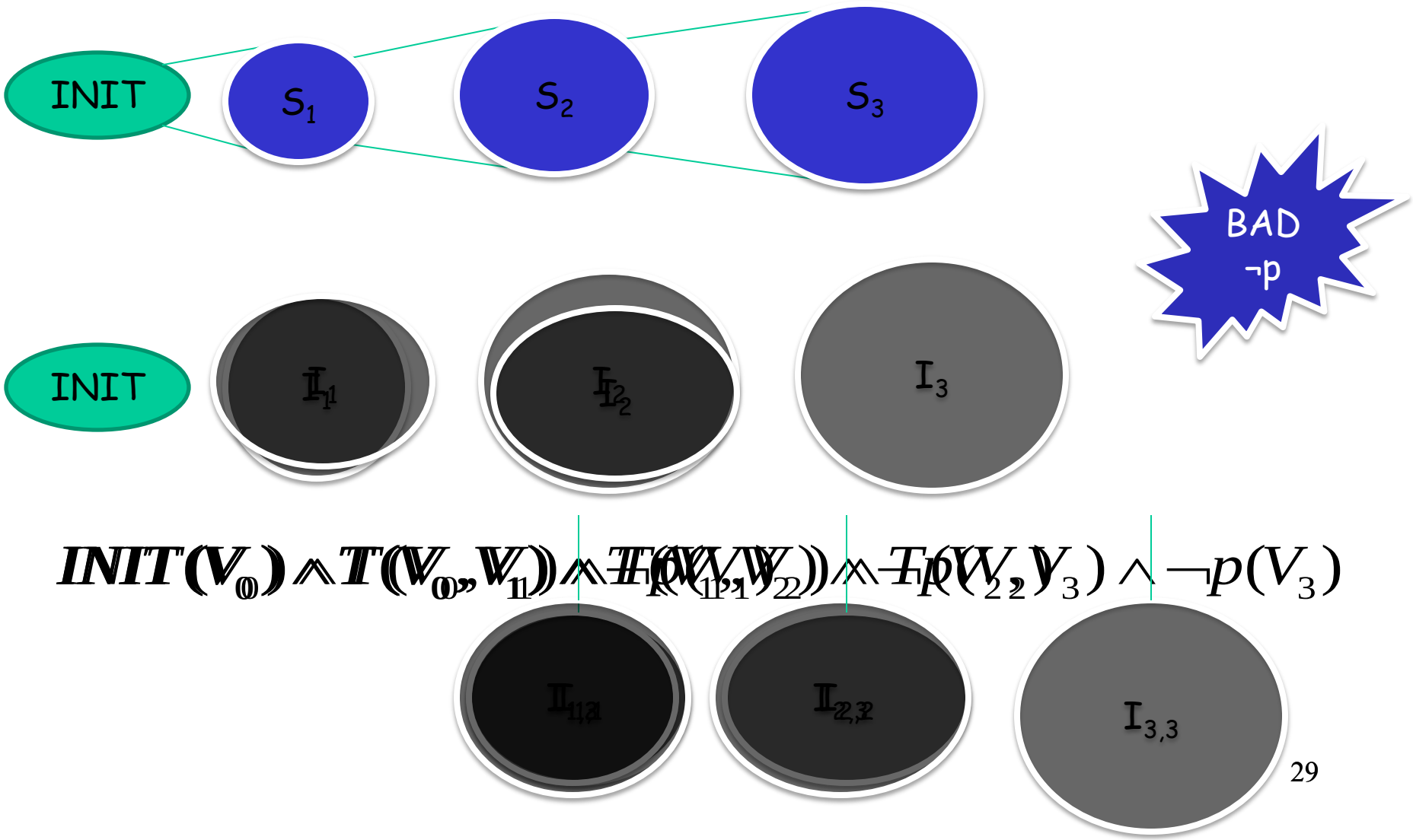
$$INIT(V_0) \wedge T(V_0, V_1) \wedge T(V_1, V_2) \wedge \neg p(V_2)$$



Checking if a "fixpoint" has been reached

- $I_j \Rightarrow \bigvee_{k=1,j-1} I_k$
- Similar to checking fixpoint in forward reachability analysis :
  $S_j \subseteq \bigcup_{k=1,j-1} S_k$
- But here we check inclusion for every $2 \leq j \leq N$
  - No monotonicity because of the approximation
- "Fixpoint" is checked with a SAT solver

The Analogy to Forward Reachability Analysis



Notation:

If no counterexample of length N or less exists in M, then:

- I_j^k is the j-th element in the interpolation-sequence extracted from the BMC-partition of φ^k
- $I_j = \Lambda_{k=j,N} I_j^k \; [V^j \leftarrow V]$
- The reachability vector is:
  $\hat{I} = (I_1, I_2, \ldots , I_N)$

explanation

$I_j = \Lambda_{k=j,N} I_j^k \; [V^j \leftarrow V]$

- Each I_j^k over-approximates S_j
  - Their conjunction results in a more accurate over-approximation
- Only I_j^j is guaranteed to satisfy p
  - I_j satisfies p

function UpdateReachable($\hat{I}$, $\hat{I}^k$)

j=1

while (j < k) do

$I_j = I_j \wedge I_j^k$

$\hat{I}[j] = I_j$

end while

$\hat{I}[k] = I_k^k$

end function

```
function FixpointReached (Î)
   j=2
   while (j ≤ Î.length) do
        R = V_{k=1,j-1} I_k
        α = I_j ∧ ¬R  // negation of I_j⇒R
        if (SAT(α)==false) then return true
        end if
        j = j+1
   end while
   return false
end function
```

```
Function ISB(M, f)    // f = AGq
    k = 0
    result = BMC (M, f, 0)
    if (result == cex) then return cex
     Î = ϕ  // the reachability vector
    while (true) do
          k = k+1
          result = BMC (M, f, k)
          if (result==cex) then return cex
          Î^k = ( T, I_1^k, … , I_k^k, F )
          UpdateReachable (Î , Î^k )
          if ( FixpointReached (Î ) == true) then
                    return true
          end if
    end while
end function
```

Interpolation-Based Model Checking [McM03]

Interpolation In The Context of Model Checking

- We can check several bounds with one formula
- Given a BMC formula with possibly several bad states

$$\overbrace{INIT(V_0) \wedge T(V_0,V_1)}^{A} \wedge \overbrace{T(V_1,V_2) \wedge \ldots \wedge T(V_{k-1},V_k) \wedge (\neg q(V_1) \vee \ldots \vee \neg q(V_k))}^{B}$$

$$\Downarrow$$

$$\mathbf{I}$$

$$A \Rightarrow I$$

$$I \wedge B \equiv F$$

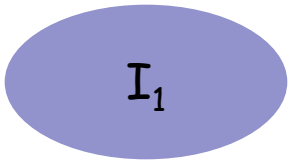
I is over the common variables of A and B, i.e V_1

Interpolation In The Context of Model Checking

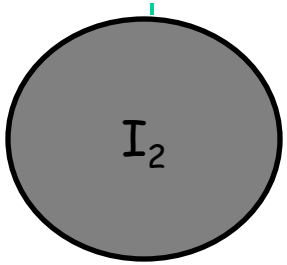
- The interpolant represents an over-approximation of reachable states after one transition.
- Also, there is no path of length *k-1* or less that can reach a bad state.

Using Interpolation

$$INIT(V_0) \wedge T(V_0, V_1) \wedge \neg q(V_1)$$



$$I_1(V_0) \wedge T(V_0, V_1) \wedge \neg q(V_1)$$



$$I_2(V_0) \wedge T(V_0, V_1) \wedge \neg q(V_1)$$

Using Interpolation

$$INIT(V_0) \wedge T(V_0,V_1) \wedge T(V_1,V_2) \wedge (\neg q(V_1) \vee \neg q(V_2))$$

I'_1

$$I_1'(V_0) \wedge T(V_0,V_1) \wedge T(V_1,V_2) \wedge (\neg q(V_1) \vee \neg q(V_2))$$

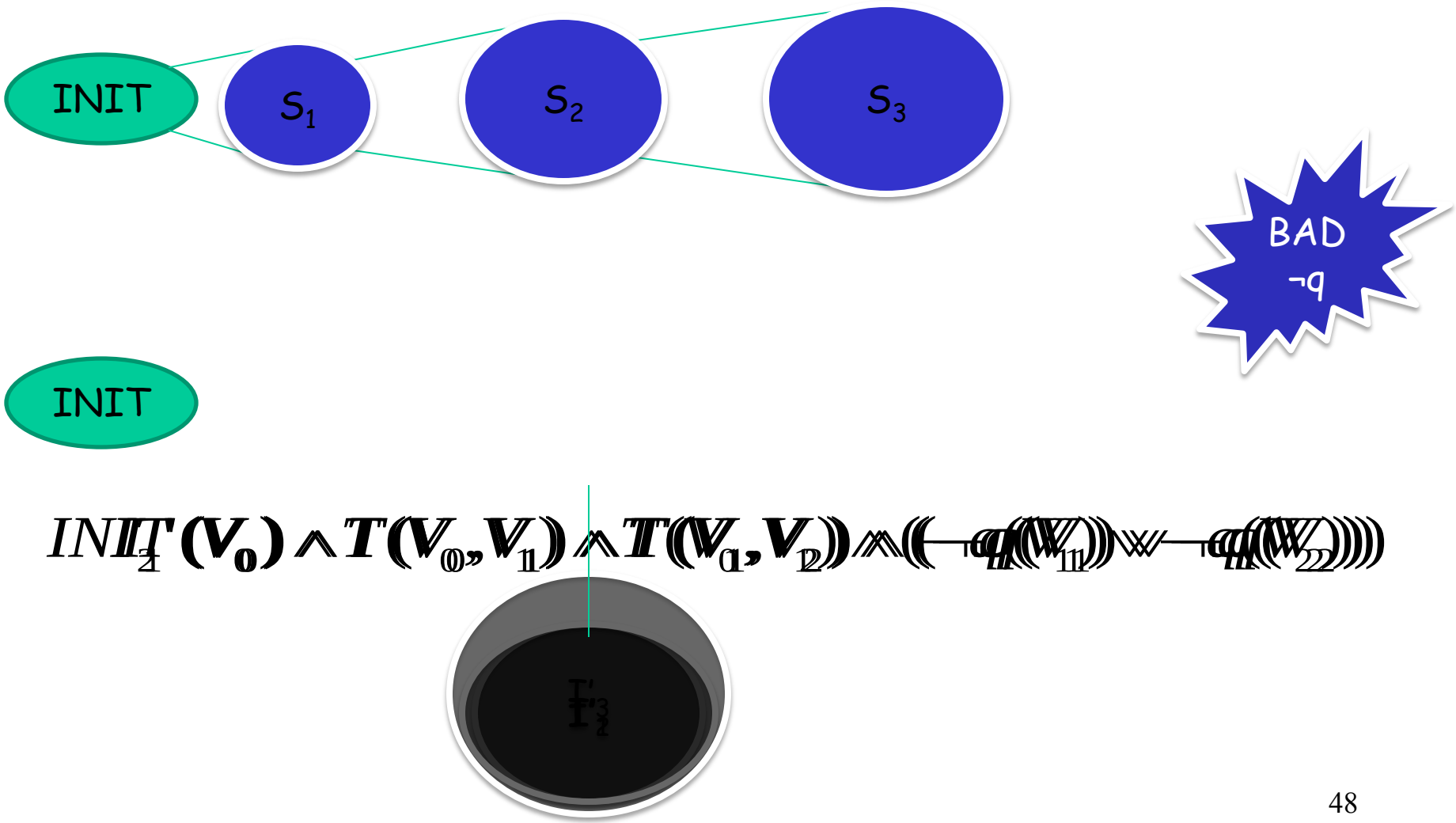
.

.

.

$$I_k'(V_0) \wedge T(V_0,V_1) \wedge T(V_1,V_2) \wedge (\neg q(V_1) \vee \neg q(V_2))$$

The Analogy to Forward Reachability Analysis



$$INIT(V_0) \wedge T(V_0, V_1) \wedge T(V_1, V_2) \wedge ((\neg q(V_1) \vee \neg q(V_2)))$$

- If BMC finds a satisfying assignment the counterexample might be spurious
  - The set of initial states is over-approximated
- Increase k and start with the original INIT

Charectaristics

- When calculating the interpolant for the i-th iteration, for bound *k* the following holds:
  - The interpolant represents an over-approximation of reachable states after *i* transitions
  - Also, it cannot reach a bad state in *k-1+i* steps or less
    - It is similar to I_i calculated in ISB after *k+i* iterations

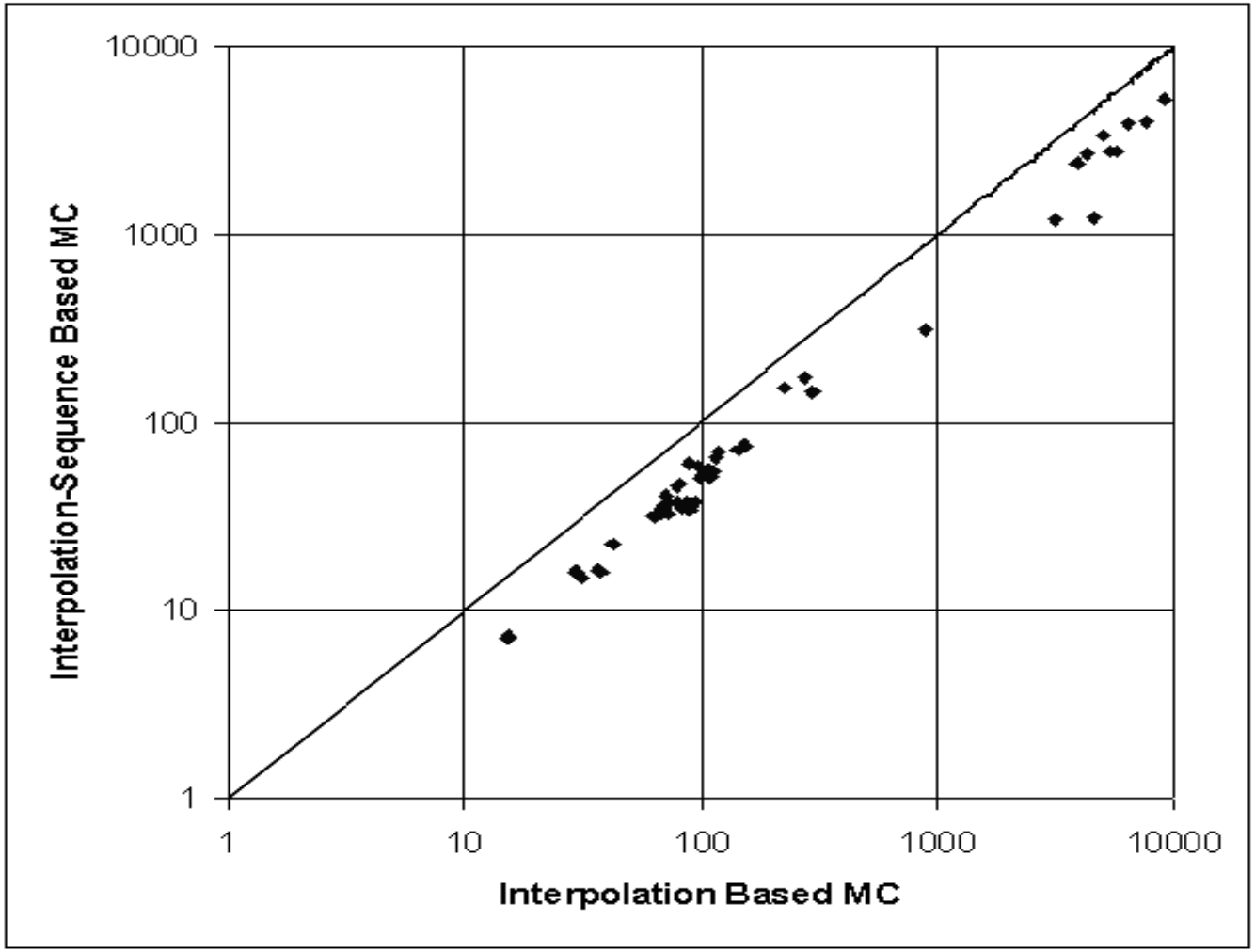
Interpolation-Based Model Checking [McM03]

- The computation itself is different
  - Uses interpolation, not interpolation sequence
  - Based on nested loops
  - Not incremental
- The computed over-approximated sets are different.

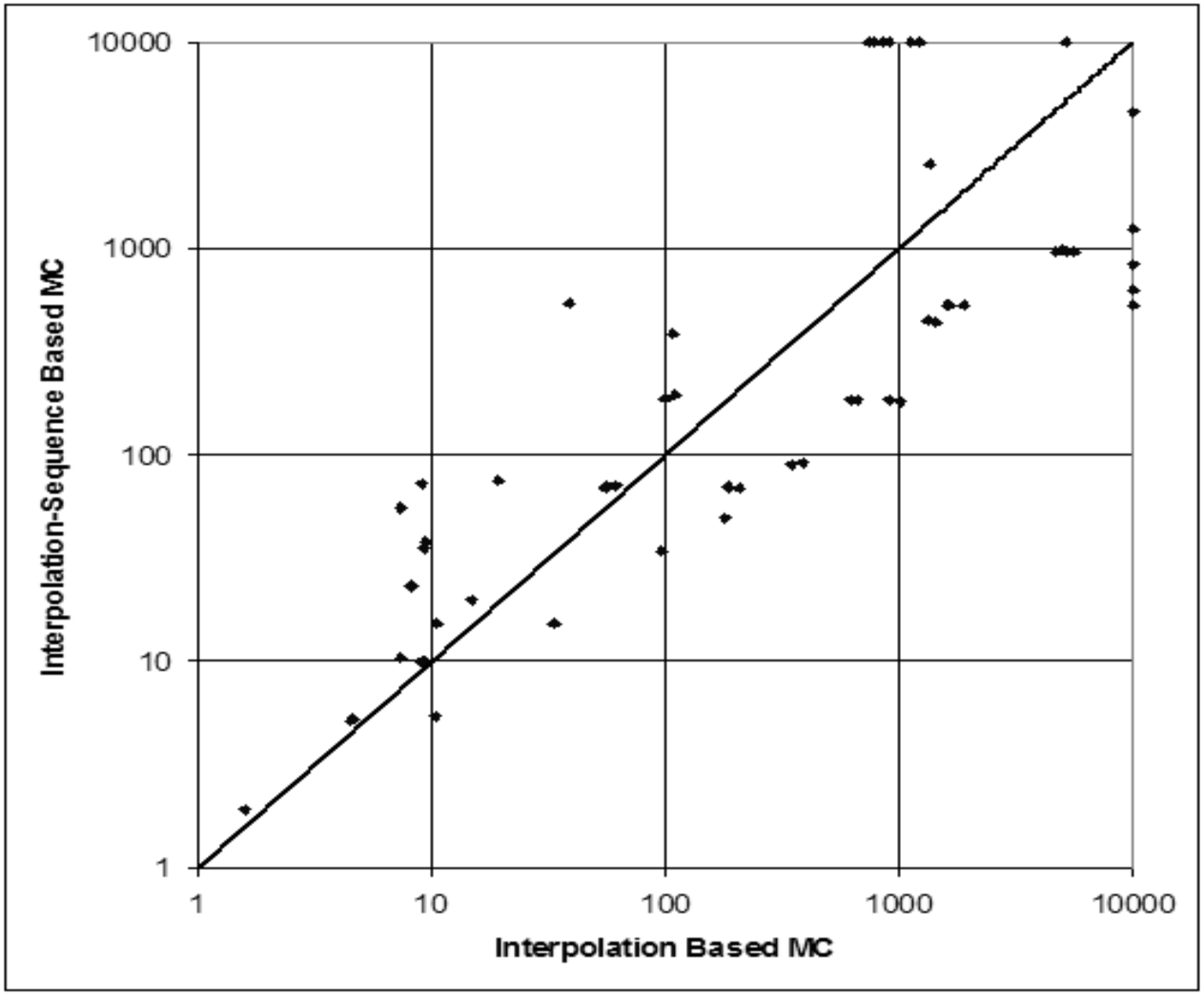
Experimental Results

- Experiments were conducted on two future CPU designs from Intel (two different architectures)

Experimental Results - Falsification



Experimental Results - Verification



Experiments Results - Analysis

Spec	#Vars	Bound (Ours)	Bound (M)	#Int (Ours)	#Int (M)	#BMC (Ours)	#BMC (M)	Time [s] (Ours)	Time [s] (M)
F_1	3406	**16**	15	**136**	80	**16**	80	**970**	5518
F_2	1753	**9**	8	**45**	40	**9**	40	**91**	388
F_3	1753	**16**	15	**136**	94	**16**	94	**473**	1901
F_4	3406	**6**	5	**21**	13	**6**	13	**68**	208
F_5	1761	**2**	1	**3**	2	**2**	2	**5**	4
F_6	3972	**3**	1	**6**	3	**3**	3	**19**	14
F_7	2197	**3**	1	**6**	3	**3**	3	**2544**	1340
F_8	4894	**5**	1	**15**	3	**5**	3	**635**	101

Analysis

- False properties is always faster.
- True properties – results vary. Heavier properties favor ISB where the easier favor IB.
- Some properties cannot be verified by one method but can be verified by the other and vise-versa.

Conclusions

- A new SAT-based method for **unbounded** model checking.
  - BMC is used for falsification.
  - Simulating forward reachability analysis for verification.
- Method was successfully applied to industrial sized systems.

Additional comments:

- Interpolation and interpolation sequence: defined for additional logics, not just propositional logic

- Interpolation sequence was suggested in "Lazy abstraction with interpolation", McMillan, CAV 2006

Thank you!