Towards a Theory of Learning Inductive Invariants

Yotam Feldman
TEL AVIV UNIVERSITY אוניברסיטת תל אביב

Neil Immerman
University of Massachusetts Amherst 1863

Mooly Sagiv
TEL AVIV UNIVERSITY אוניברסיטת תל אביב

Sharon Shoham
TEL AVIV UNIVERSITY אוניברסיטת תל אביב

James R. Wilcox
CERTORA

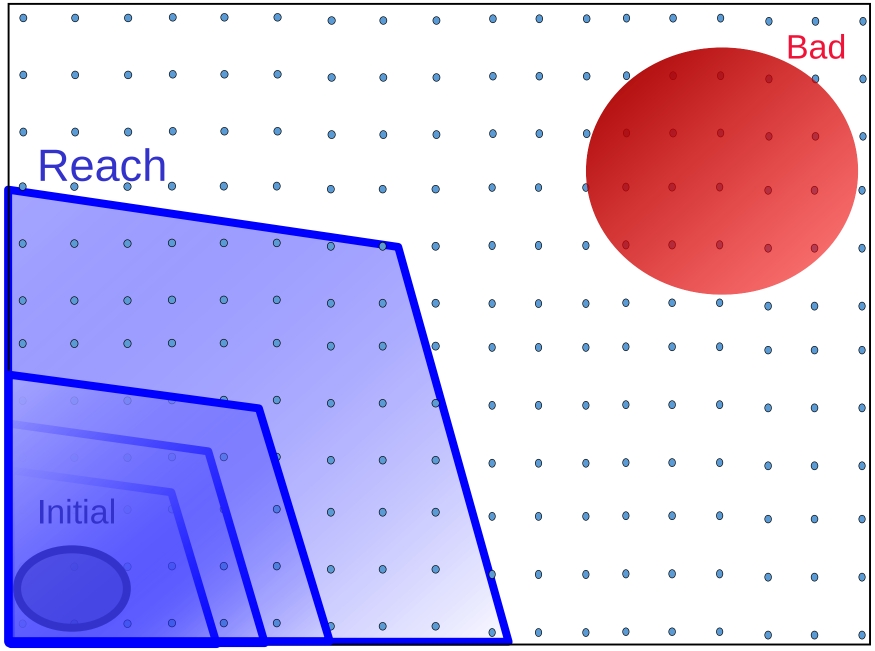
@yotamfe, @SagivMooly, @wilcoxjay

Safety of Transition Systems

<u>Init</u>:
$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

<u>Bad</u>:
$(x_1, \ldots, x_n) \;=\; 1 \ldots 1$

<u>δ</u>:
$y_1, \ldots, y_n \coloneqq *$
$x_1, \ldots, x_n \coloneqq (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

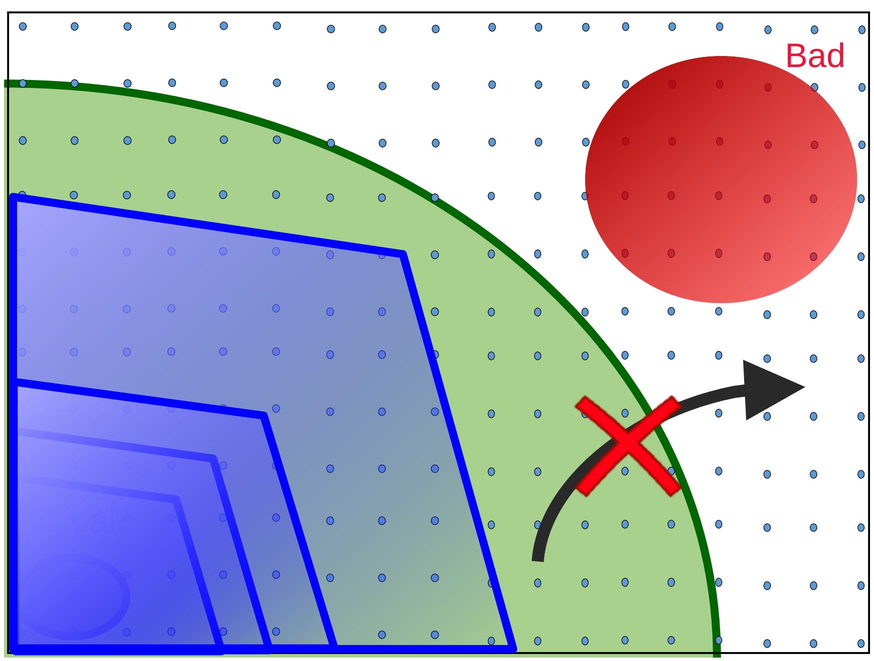


Inductive Invariants

<u>Init</u>:
$(x_1, \dots, x_n) \coloneqq 0 \dots 0$

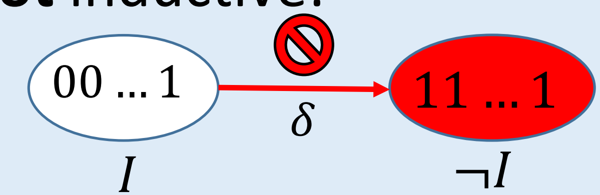
<u>Bad</u>:
$(x_1, \dots, x_n) = 1 \dots 1$

<u>δ</u>:
$y_1, \dots, y_n \coloneqq *$
$x_1, \dots, x_n \coloneqq (x_1, \dots, x_n) + 2 \cdot (y_1, \dots, y_n) \pmod{2^n}$



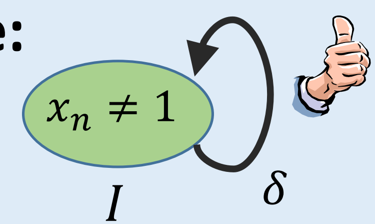
I: $(x_1, \dots, x_n) \neq 1 \dots 1$

Not inductive:



I: $x_n \neq 1$

Inductive:



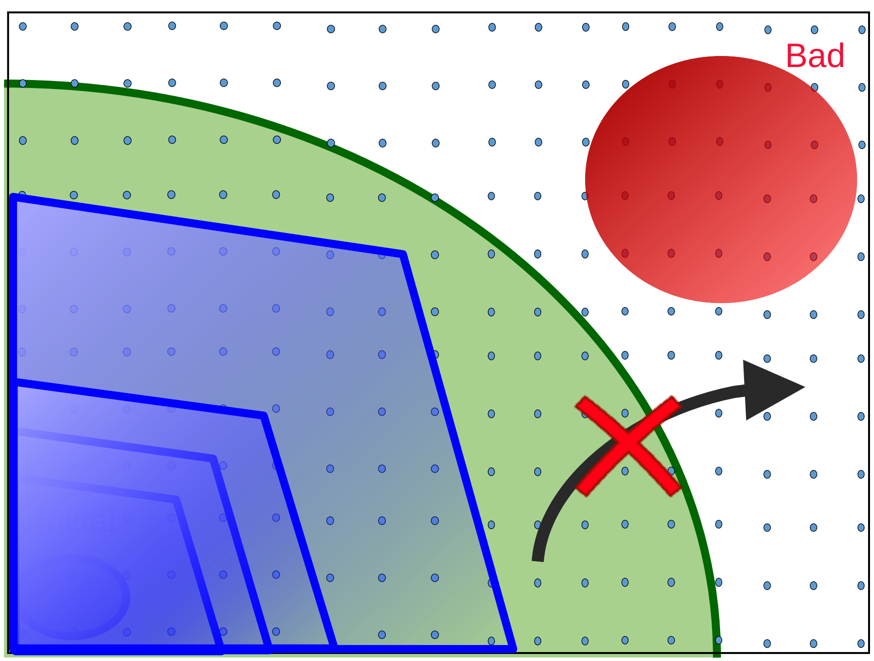
Inductive Invariants

Init: δ:

Goal:

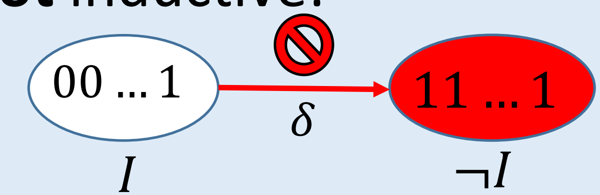
Find inductive invariants automatically

$(x_1, \dots, x_n) = 1 \dots 1$



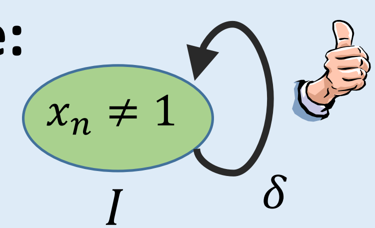
I: $(x_1, \dots, x_n) \neq 1 \dots 1$

Not inductive:



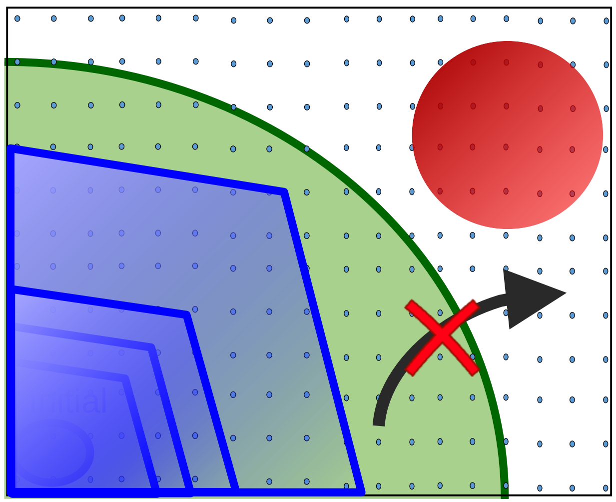
I: $x_n \neq 1$

Inductive:



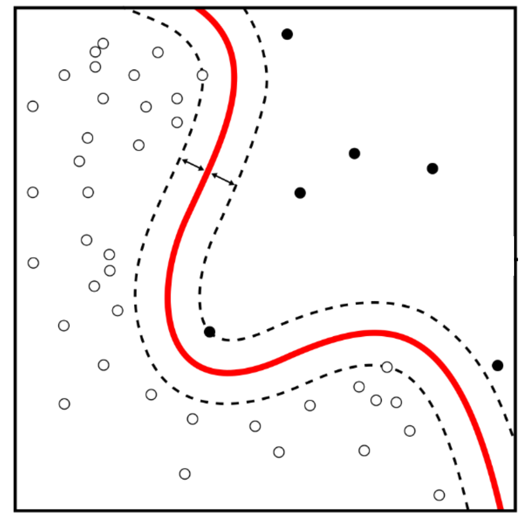
This Work

Invariant Inference



vs.

Exact Concept Learning



- Query-based learning models for invariant inference
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from classification algorithms

Problem Setting: Polynomial-Length Inference

Boolean transition systems, $\Sigma = \{p_1, \ldots, p_n\}$

Given a transition system from a class $\mathcal{P}$ (over Σ),

Find an inductive invariant

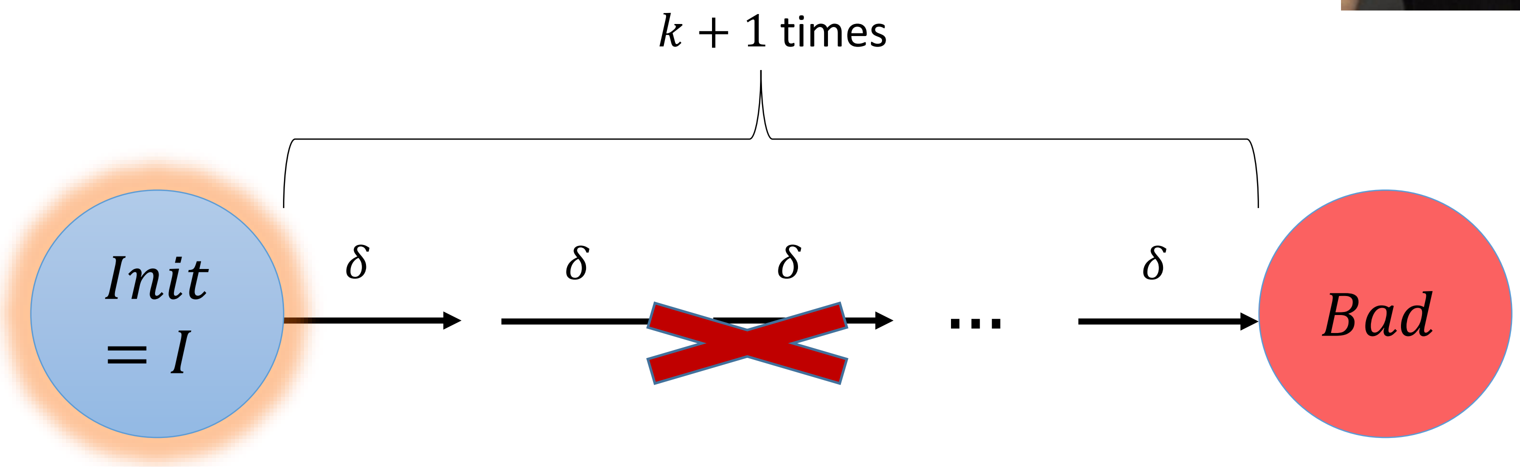
$I \in \mathrm{DNF}$ $|I| \leq \mathrm{poly}(n)$

(Decision problem is Σ_2^P-complete.)

[CADE'09] Complexity and Algorithms for Monomial and Clausal Predicate Abstraction. Lahiri, Qadeer
[POPL'20] Complexity and Information in Invariant Inference. Feldman, Immerman, Shoham, Sagiv

Interpolation-Based Inference

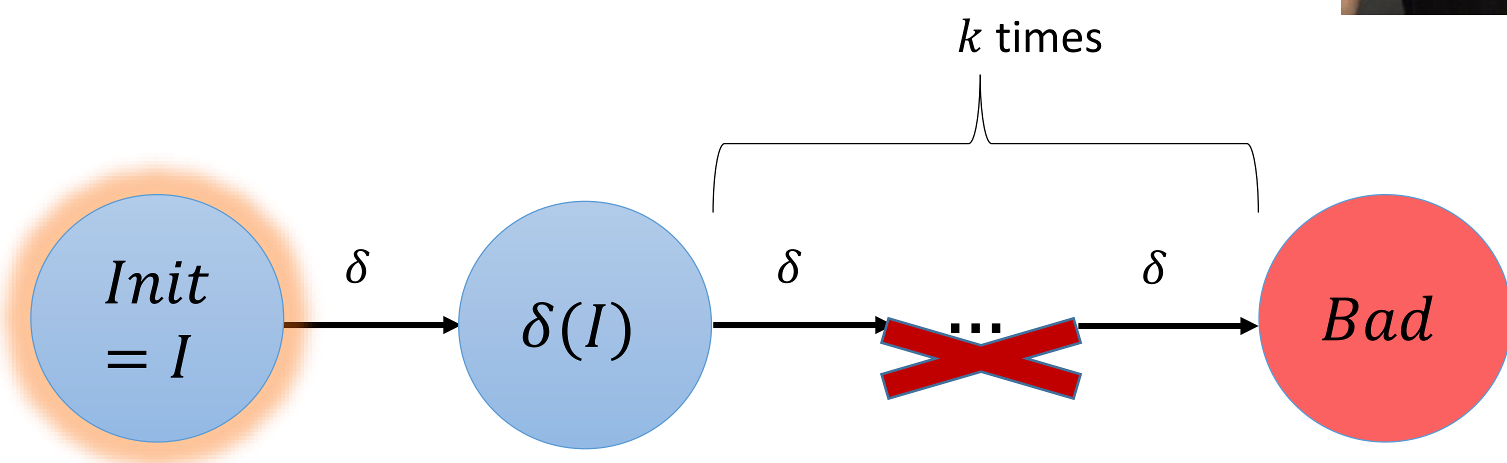
$I = Init$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Interpolation-Based Inference

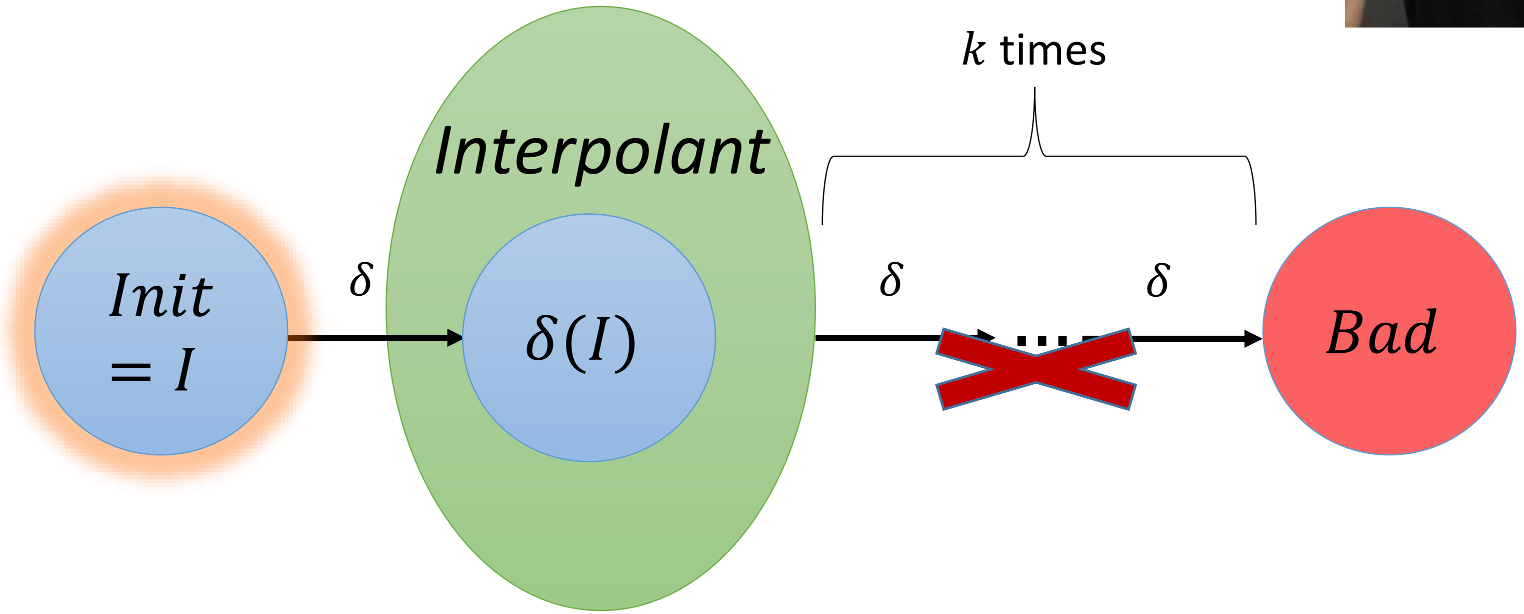
$I = Init$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Interpolation-Based Inference

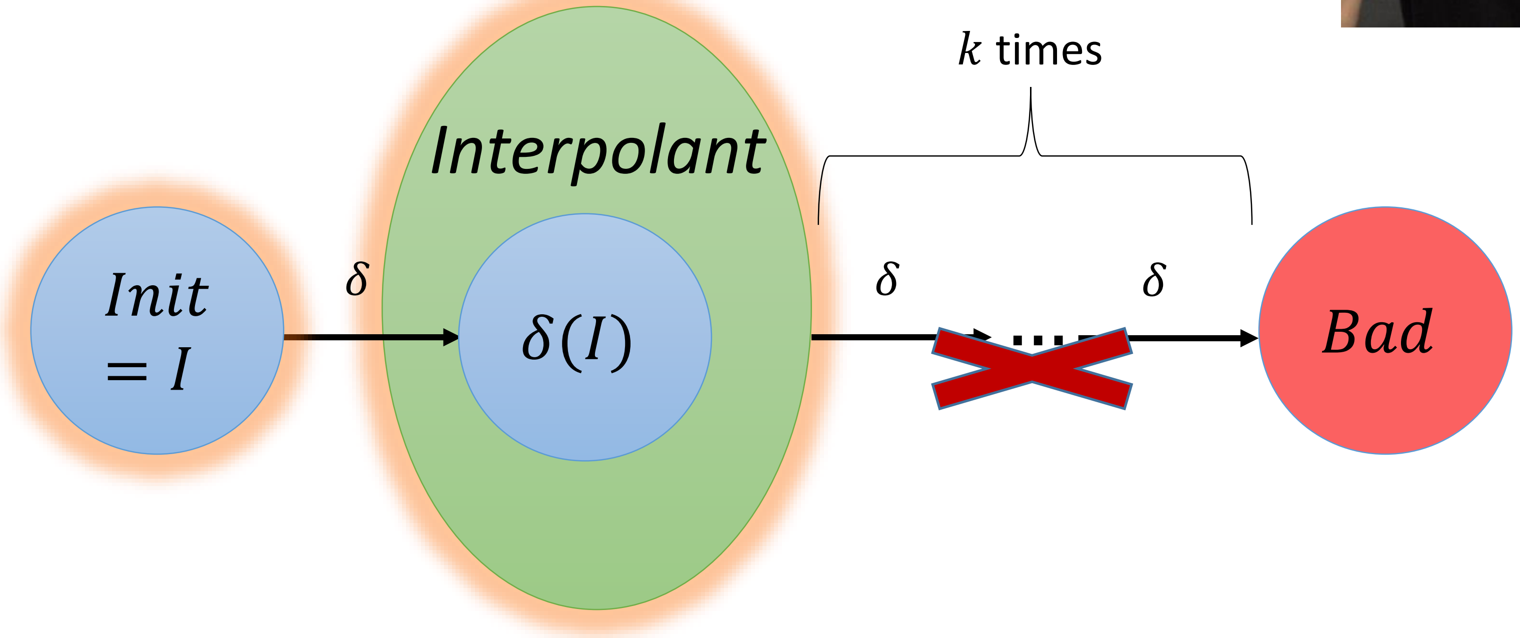
$I = Init$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Interpolation-Based Inference

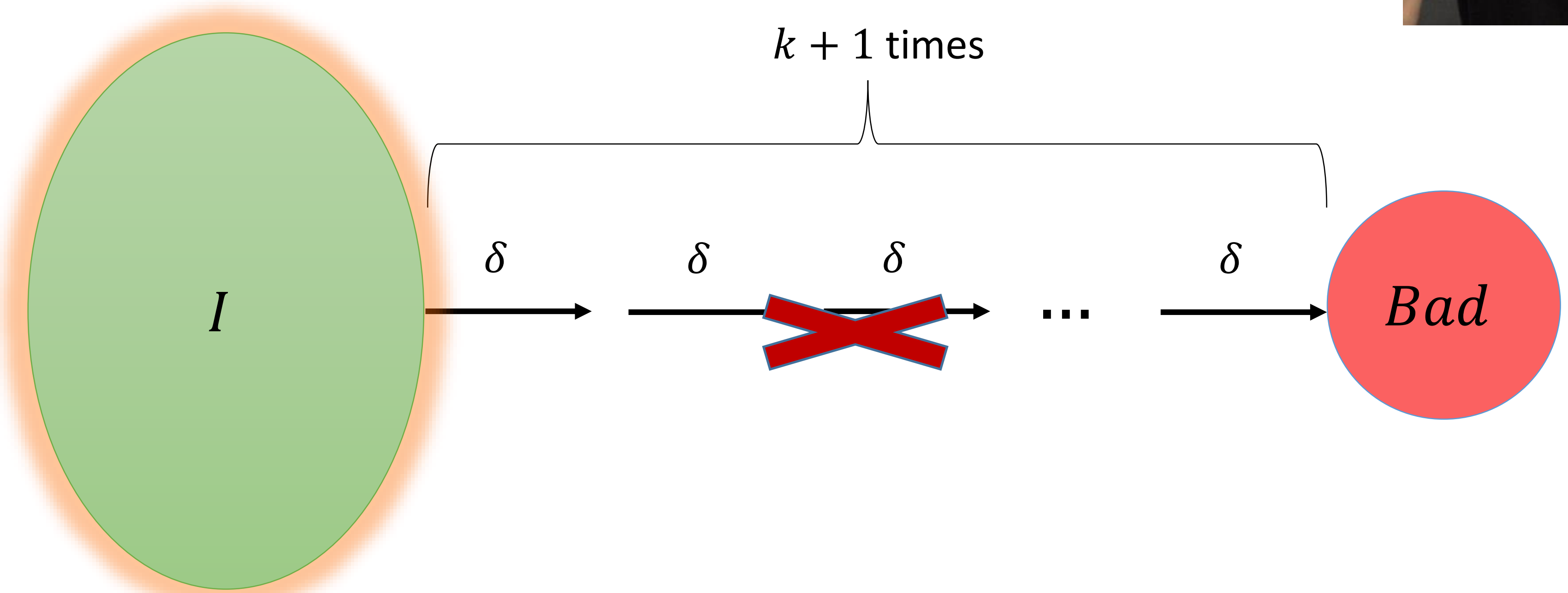
$$I = Init \lor Interpolant$$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Interpolation-Based Inference

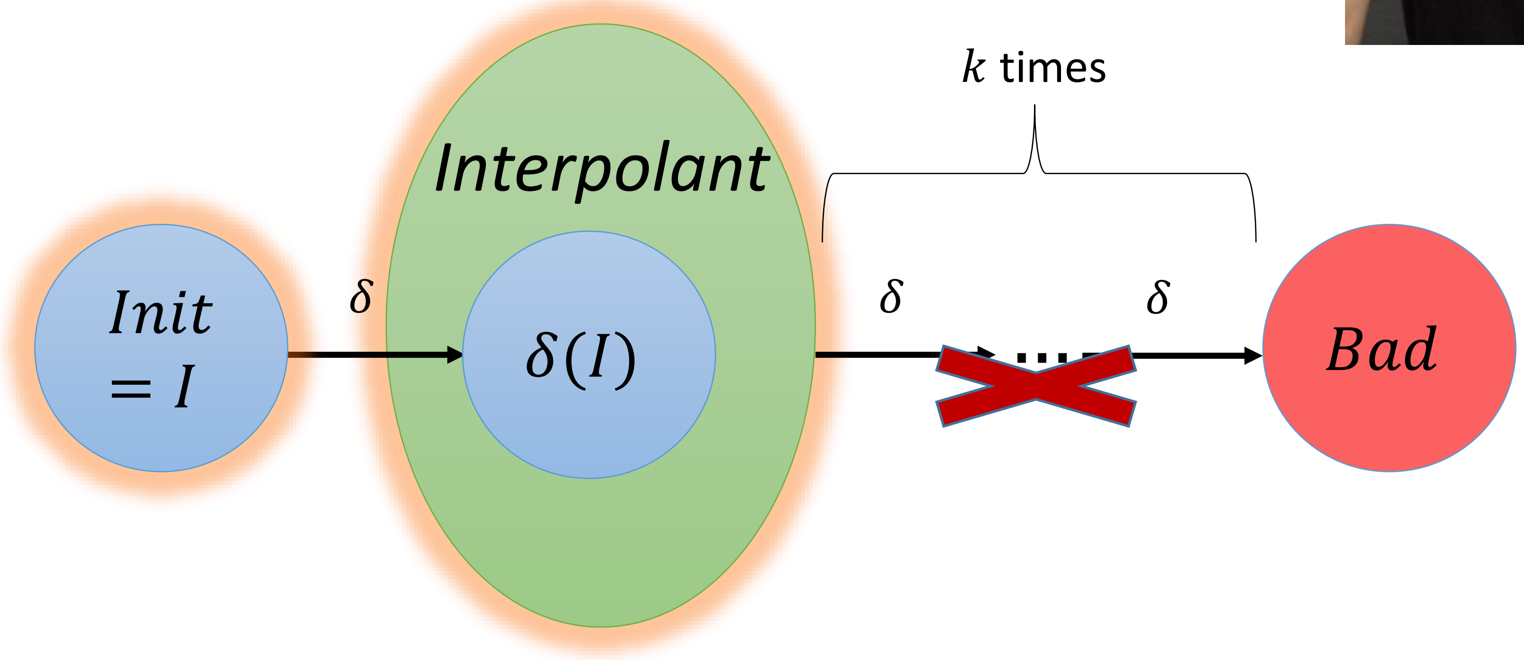
$$I = Init \vee Interpolant$$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Interpolation-Based Inference

$$I = Init \lor Interpolant$$



[CAV'03] Interpolation and SAT-Based Model Checking, McMillan

Model-Based Interpolation

Init:

$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

Bad:

$(x_1, \ldots, x_n) = 1 \ldots 1$

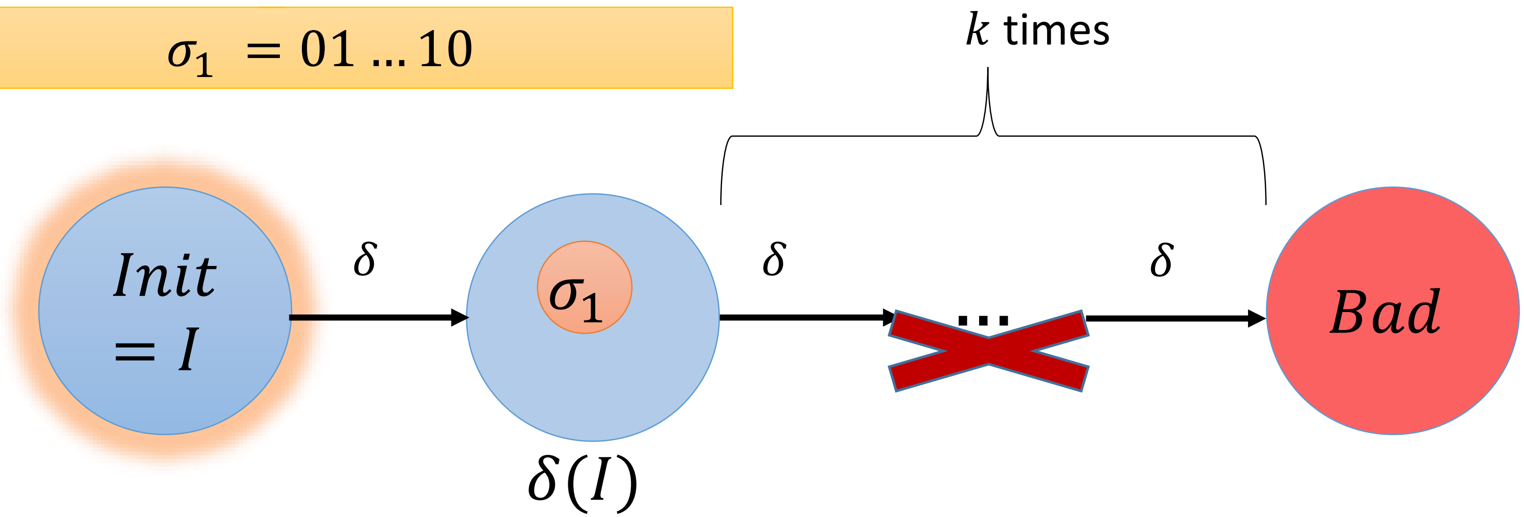
$\underline{\delta}$:

$y_1, \ldots, y_n \coloneqq *$

$x_1, \ldots, x_n \coloneqq (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_n = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \ldots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

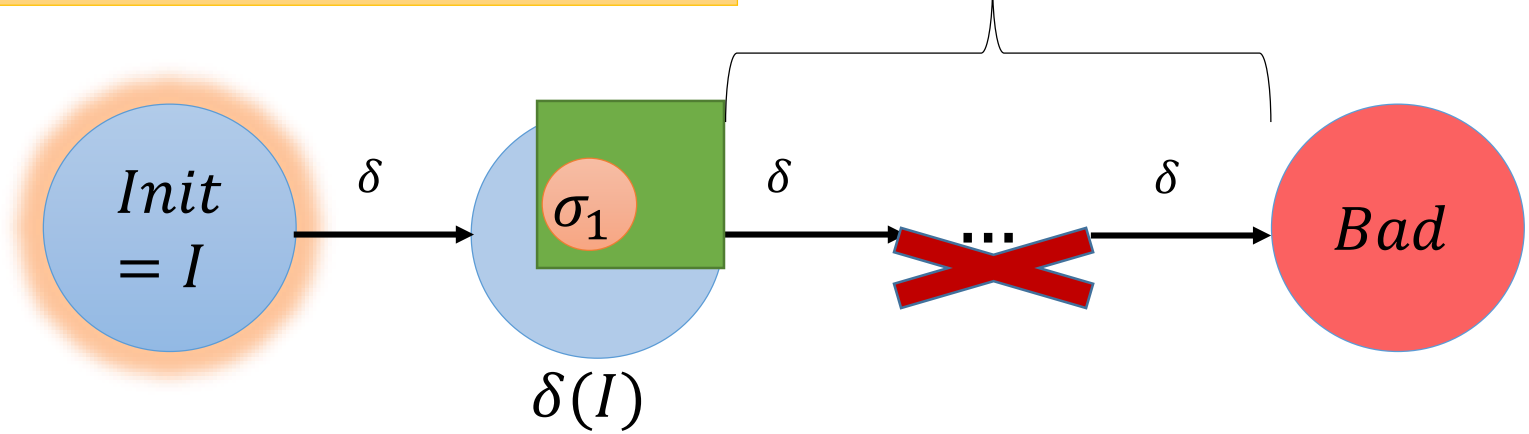
Init:
$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

Bad:
$(x_1, \ldots, x_n) = 1 \ldots 1$

δ:
$y_1, \ldots, y_n \coloneqq *$
$x_1, \ldots, x_n \coloneqq (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_n = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \ldots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

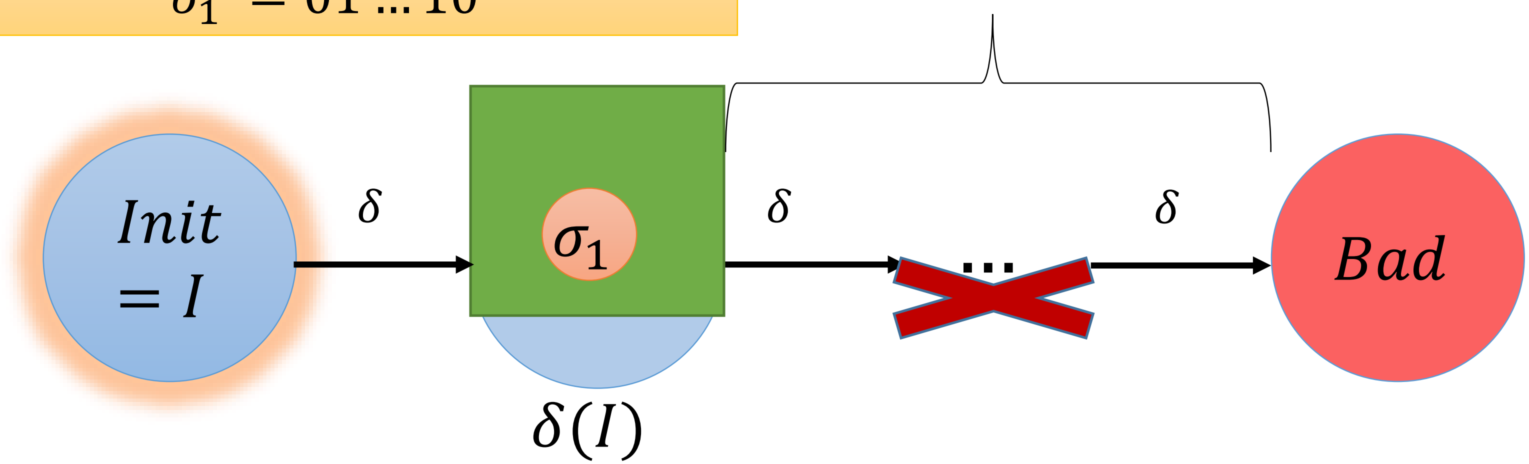
Init:
$(x_1, \dots, x_n) \coloneqq 0 \dots 0$

Bad:
$(x_1, \dots, x_n) = 1 \dots 1$

$\underline{\delta}$:
$y_1, \dots, y_n \coloneqq *$
$x_1, \dots, x_n \coloneqq (x_1, \dots, x_n) + 2 \cdot (y_1, \dots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 = 0 \land x_2 = 1 \land \cdots \land x_n = 1 \land x_n = 0)$

$\sigma_1 = 01 \dots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

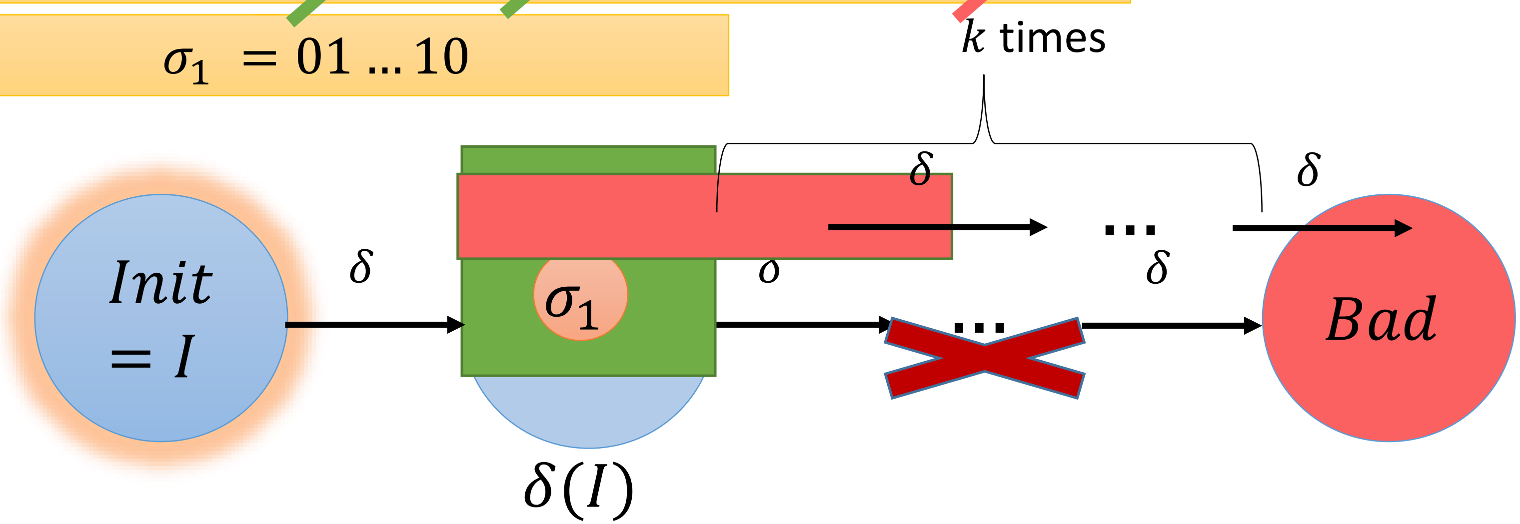
Init:
$(x_1, \ldots, x_n) := 0 \ldots 0$

Bad:
$(x_1, \ldots, x_n) = 1 \ldots 1$

δ:
$y_1, \ldots, y_n := *$
$x_1, \ldots, x_n := (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_n = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \ldots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

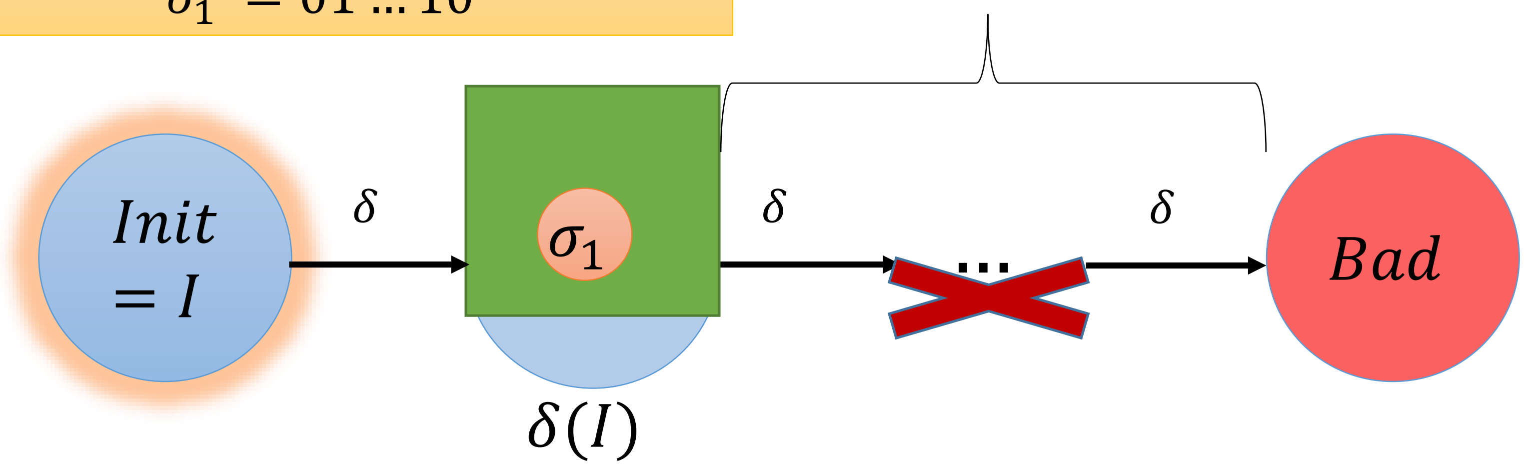
Init:
$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

Bad:
$(x_1, \ldots, x_n) = 1 \ldots 1$

$\underline{\delta}$:
$y_1, \ldots, y_n \coloneqq *$
$x_1, \ldots, x_n \coloneqq (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_n = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \ldots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

<u>Init</u>:

$(x_1, \dots, x_n) \coloneqq \color{blue}{0 \dots 0}$

<u>Bad</u>:

$(x_1, \dots, x_n) = \color{blue}{1 \dots 1}$

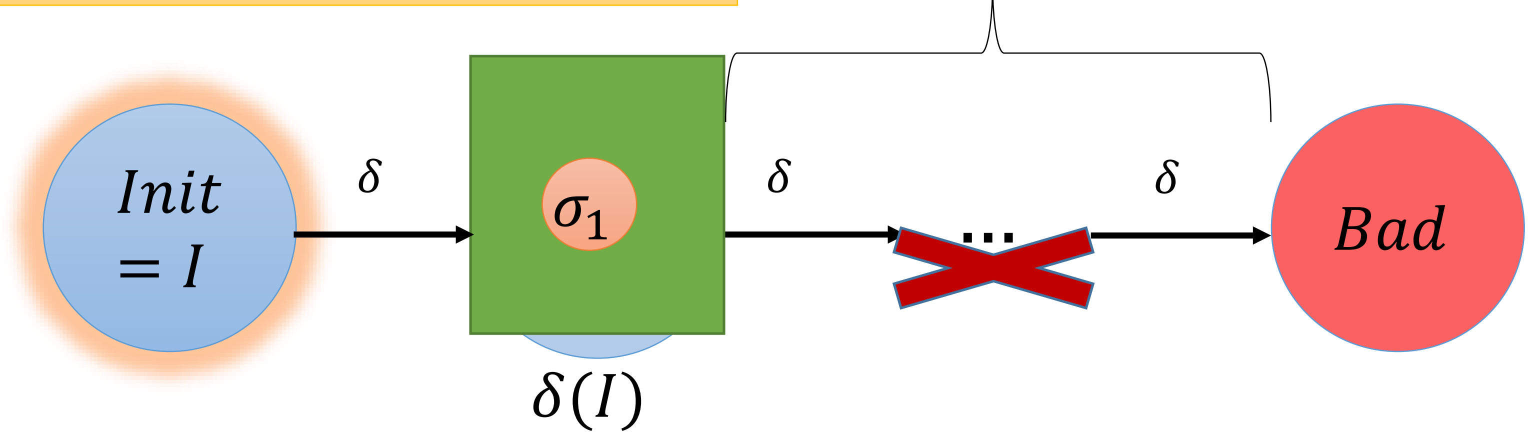
<u>δ</u>:

$y_1, \dots, y_n \coloneqq *$

$x_1, \dots, x_n \coloneqq (x_1, \dots, x_n) + \color{blue}{2} \cdot (y_1, \dots, y_n) \pmod{2^n}$

$Interpolant_1 = (x_1 \neq 0 \wedge x_2 \neq 1 \wedge \cdots \wedge x_n \neq 1 \wedge x_n = 0)$

$\sigma_1 = 01 \dots 10$



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

Init:

$(x_1, \ldots, x_n) \;\coloneqq\; 0 \ldots 0$

Bad:

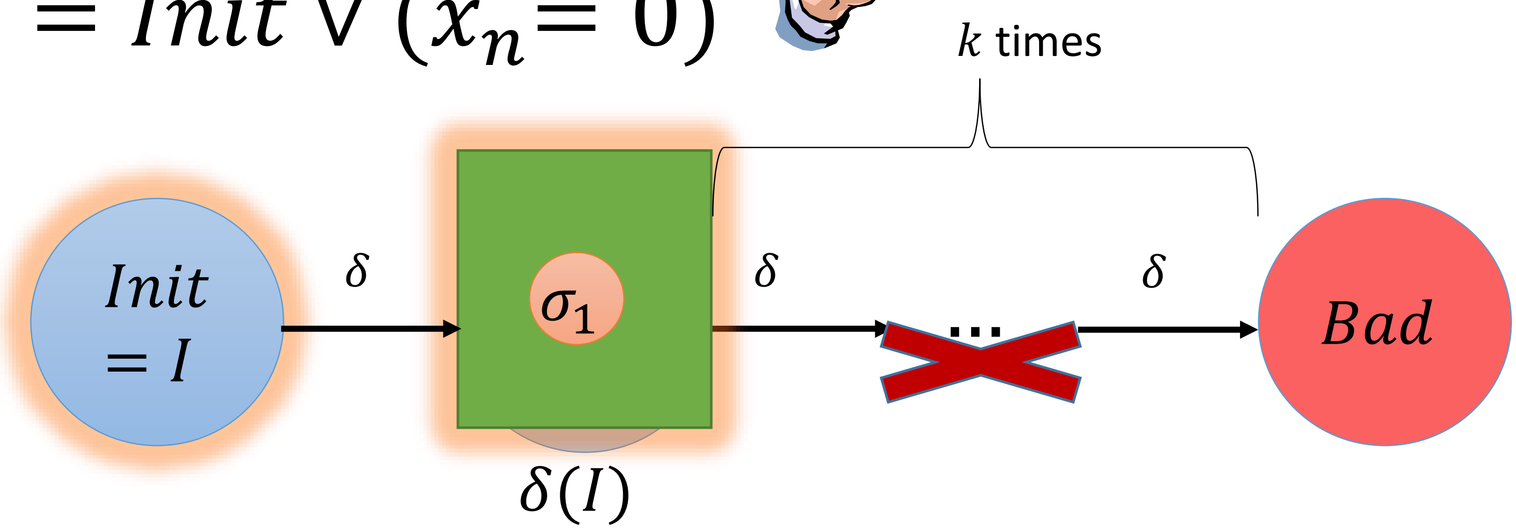
$(x_1, \ldots, x_n) \;=\; 1 \ldots 1$

δ :

$y_1, \ldots, y_n \;\coloneqq\; *$

$x_1, \ldots, x_n \;\coloneqq\; (x_1, \ldots, x_n) + 2 \cdot (y_1, \ldots, y_n) \pmod{2^n}$

$$I = Init \vee (x_n = 0)$$



[HVC’12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah
[LPAR’13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

Inferring invariant in DNF:

$$\underbrace{\left(\ell_1^1 \wedge \cdots \wedge \ell_{k_1}^1\right)}_{\text{gen}(\sigma_1)} \vee \; \ldots \; \vee \underbrace{\left(\ell_1^m \wedge \cdots \wedge \ell_{k_m}^m\right)}_{\text{gen}(\sigma_m)}$$

I := false

while $(\_, \sigma')$ counterexample
      to Inductive(δ, I):

  I := $I \vee$ **generalize**(σ')

generalize(σ'):

  drop literals from σ'

  while BMC$(\delta, \sigma', k) \cap \text{Bad} = \emptyset$

Model-Based Interpolation

```
I  := false
while (_,σ′) counterexample
      to Inductive(δ,I):
   I  := I ∨ generalize(σ′)
```

generalize(σ'):

drop literals from σ'

while BMC$(\delta, \sigma', k) \cap \mathrm{Bad} = \emptyset$

Understanding Invariant Inference

I := false

while $(\_, \sigma')$ counterexample
    to Inductive(δ, I):

I := $I \vee$ **generalize**(σ')

generalize(σ'):

drop literals from σ'

while BMC$(\delta, \sigma', k) \cap \text{Bad} = \emptyset$

Understanding Invariant Inference

I := false

while $(\_, \sigma')$ counterexample
    to Inductive(δ, I):

  I := $I \lor$ **generalize**(σ')

generalize(σ'):

  drop literals from σ'

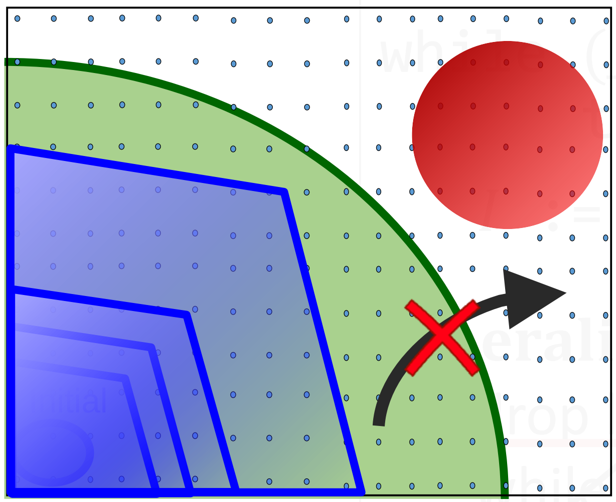
  while BMC$(\delta, \sigma', k) \cap \mathrm{Bad} = \emptyset$

Complexity bounds from exact classification algorithms

Rich SAT queries allow exponentially faster inference

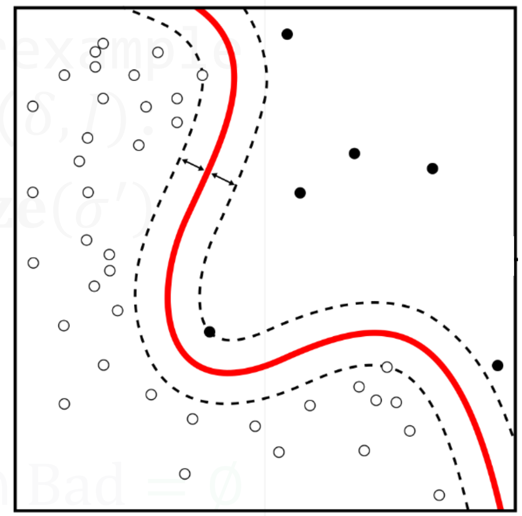
Understanding Invariant Inference

Invariant Inference



vs.

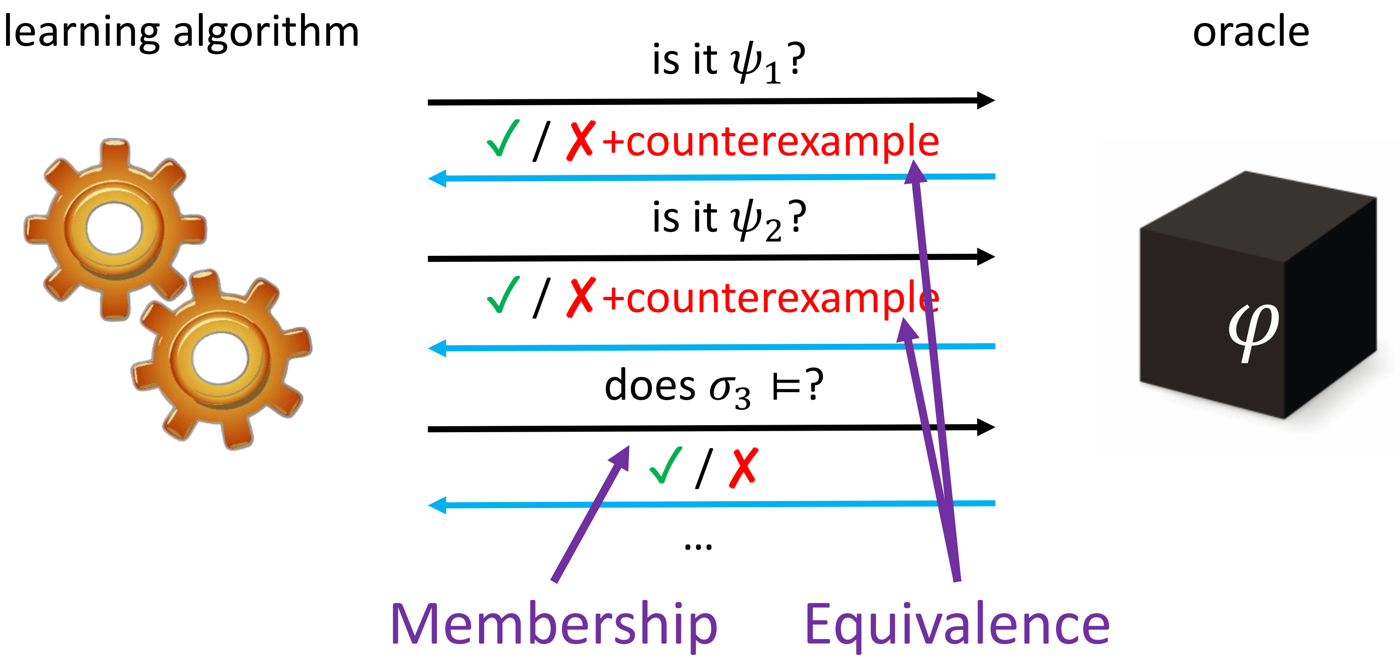
Exact Concept Learning



Complexity bounds from
exact classification algorithms

Rich SAT queries allow
exponentially faster inference

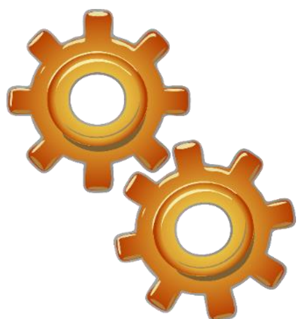
Exact Concept Learning with Equivalence & Membership Queries



[ML'87] Queries and Concept Learning. Angluin

Invariant Inference with Equivalence & Membership Queries

learning algorithm



Need to implement this

is it ψ_1?

✓ / ✗+counterexample

is it ψ_2?

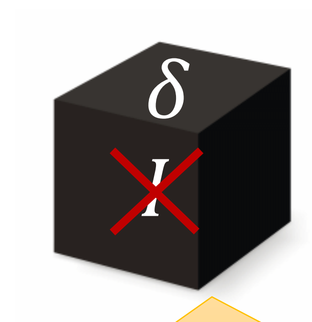
✓ / ✗+counterexample

does $\sigma_3 \vDash$?

✓ / ✗

…

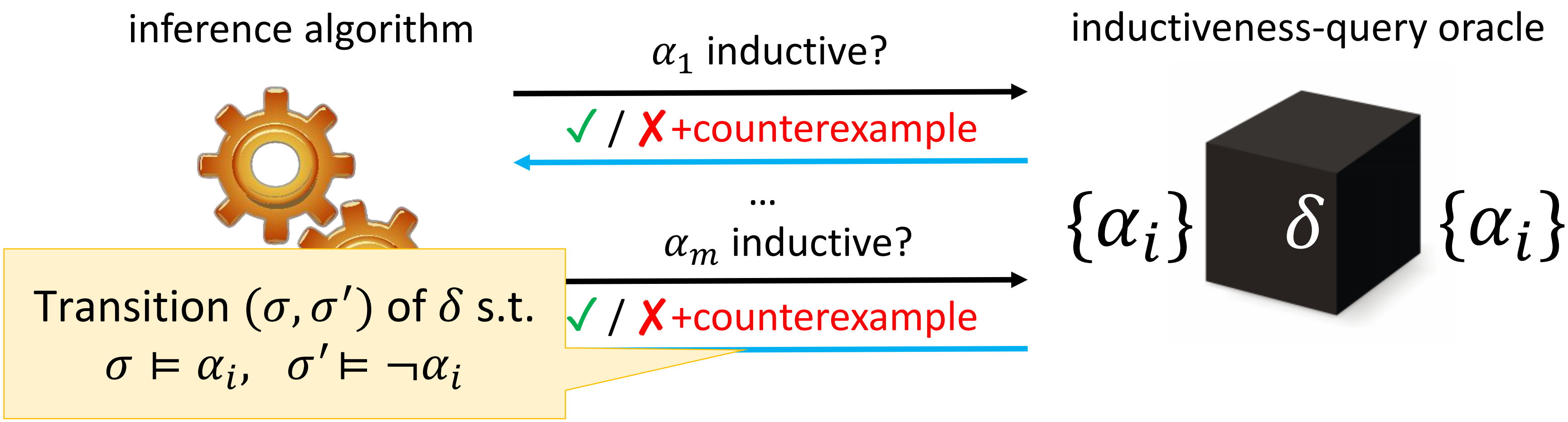
oracle



Need to implement this

[CAV'14] ICE: A Robust Framework for Learning Invariants. Garg, Löding, Madhusudan, Neider

Inductiveness-Query Model



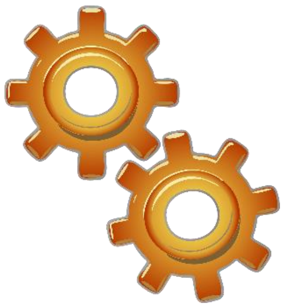
Algorithms cannot access the transition relation directly, only perform inductiveness queries

Complexity: # inductiveness queries worst case amongst possible counterexamples

[CAV'14] ICE: A Robust Framework for Learning Invariants. Garg, Löding, Madhusudan, Neider

Inductiveness-Query Model

inference algorithm



α_1 inductive?

✓ / ✗+counterexample

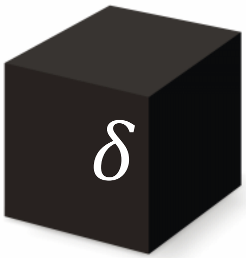
…

α_m inductive?

✓ / ✗+counterexample

inductiveness-query oracle

$\{\alpha_i\}$



δ

$\{\alpha_i\}$

I := false

while $(\_, \sigma')$ counterexample
    to Inductive(δ, I):
  I := $I \vee$ **generalize**(σ')

generalize(σ'):
  drop literals from σ'
  while BMC$(\delta, \sigma', k) \cap \text{Bad} = \emptyset$

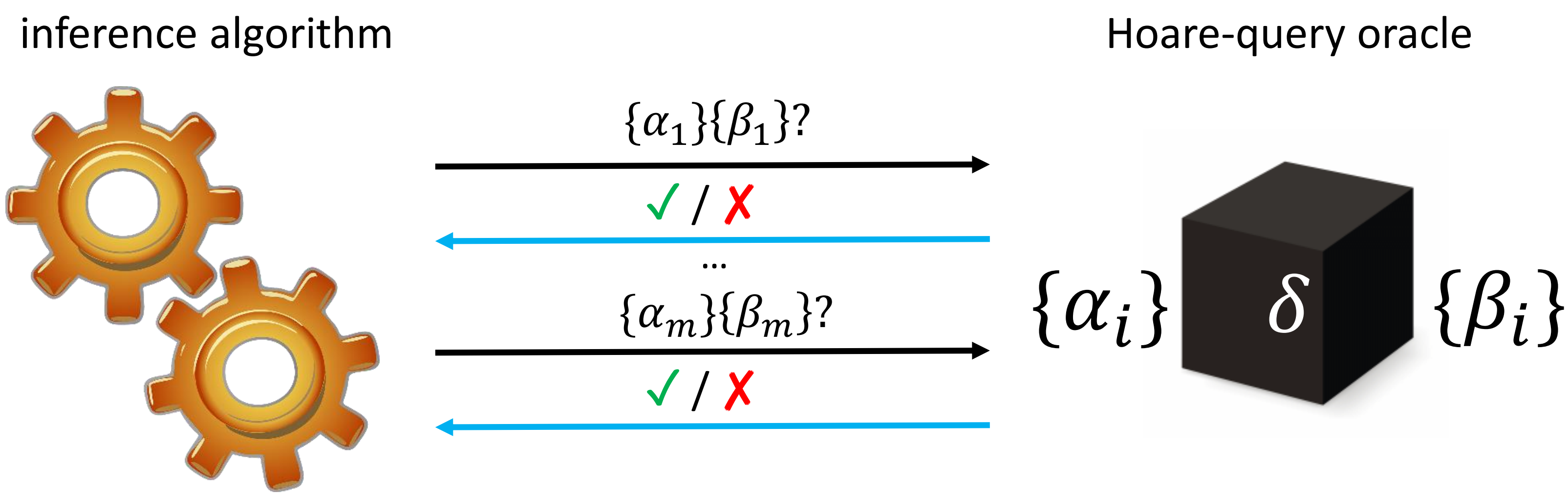
I inductive?

✓ / ✗

?

Hoare-Query Model

Capable of modeling several interesting algorithms



Algorithms cannot access the transition relation directly, only perform Hoare queries

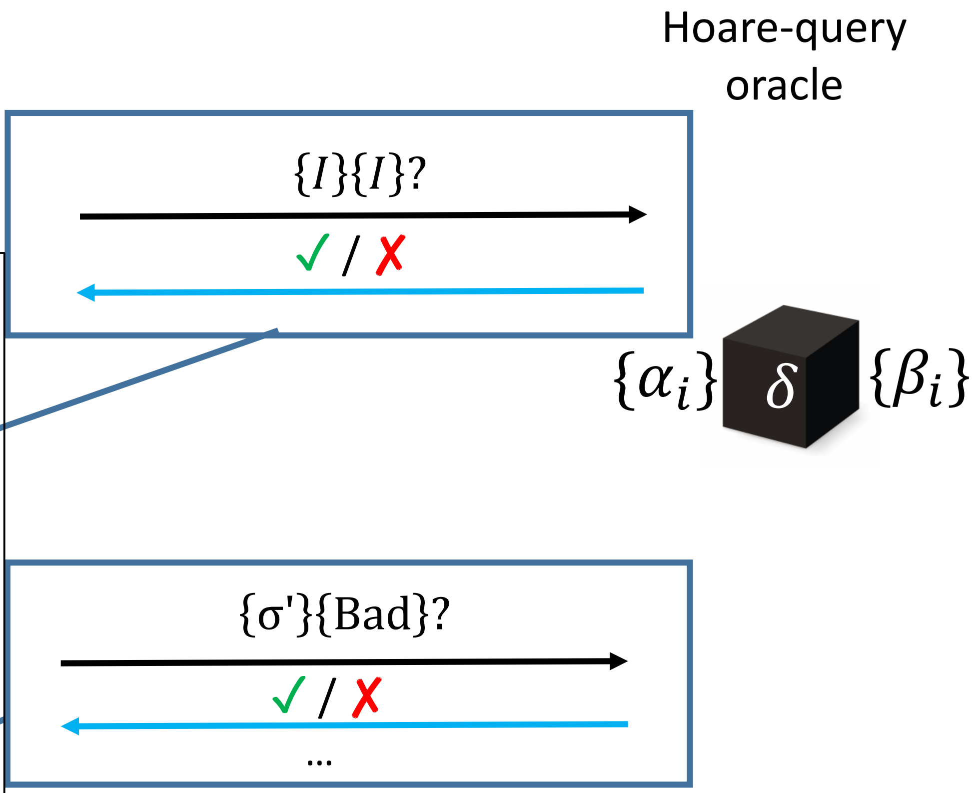
Hoare-Query Model

Capable of modeling several interesting algorithms

model of $\neg\{I\}\delta\{I\}$, $O(n)$ queries

I := false
while $(\_, \sigma')$ counterexample
    to Inductive(δ, I):
  I := $I \vee$ **generalize**(σ')

generalize(σ'):
  drop literals from σ'
  while BMC$(\delta, \sigma', 1) \cap \text{Bad} = \emptyset$



Hoare > Inductiveness

<u>Thm</u>: There exists a class of transition systems $\mathcal{P}$, so that for solving polynomial-length inference:

1. **∃** Hoare-query algorithm with $\text{poly}(n)$ queries
2. **∀** inductiveness-query algorithm requires $2^{\Omega(n)}$ queries

a simple case of IC3/PDR

$\Rightarrow$ ICE cannot model PDR,

and the extension of [VMCAI'17] is necessary

[POPL'20] Complexity and Information in Invariant Inference. Feldman, Immerman, Shoham, Sagiv
[VMCAI'17] IC3 - Flipping the E in ICE. Vizel, Gurfinkel, Shoham, Malik.

Hoare > Inductiveness

Thm: There exists a class of transition systems $\mathcal{P}$, so that for solving polynomial-length inference:

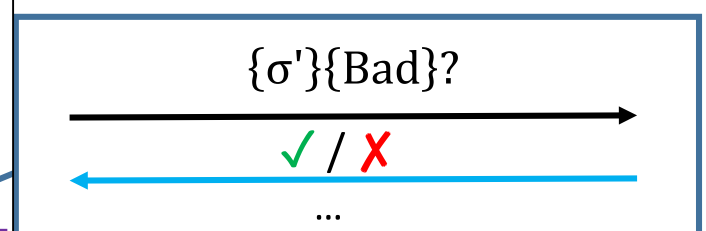
1. ∃ Hoare-query algorithm with $\text{poly}(n)$ queries
2. ∀ inductiveness-query algorithm requires $2^{\Omega(n)}$ queries

I := false

while $(\_, \sigma')$ counterexample
  to Inductive(δ, I):

  I := $I \lor$ **generalize**(σ')

generalize(σ'):
  drop literals from σ'
  while $\text{BMC}(\delta, \sigma', 1) \cap \text{Bad} = \emptyset$



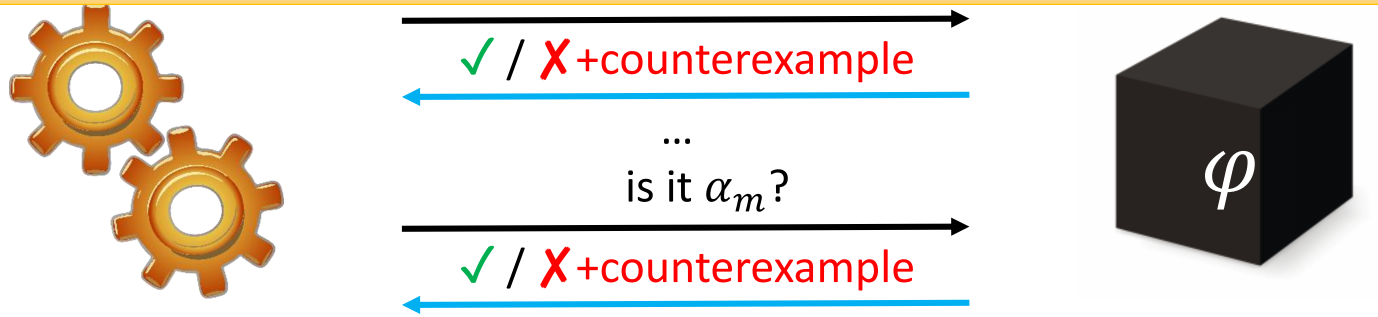
Hoare > Inductiveness

Thm: There exists a class of transition systems $\mathcal{P}$, so that for solving polynomial-length inference:

1. ∃ Hoare-query algorithm with $\mathrm{poly}(n)$ queries
2. ∀ inductiveness-query algorithm requires $2^{\Omega(n)}$ queries

Learning from Counterexamples to Equivalence Queries

Thm: Learning from counterexamples to induction is **harder** than learning from labeled examples.



Positive/negative examples:

$\sigma^+ \vDash \varphi$, $\sigma^- \vDash \neg\varphi$

Counterexamples to induction:

$\sigma \vDash \neg\varphi$ **or** $\sigma' \vDash \varphi$

Learning monotone DNF:

subexponential

this work: $\mathbf{2^{\Omega(n)}}$

[ML'87] Queries and Concept Learning, Angluin
[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.
[POPL'20] Complexity and Information in Invariant Inference. Feldman, Immerman, Shoham, Sagiv

Understanding Invariant Inference

I := false

while $(\_, \sigma')$ counterexample
    to Inductive(δ, I):

  I := $I \vee$ **generalize**(σ')

generalize(σ'):

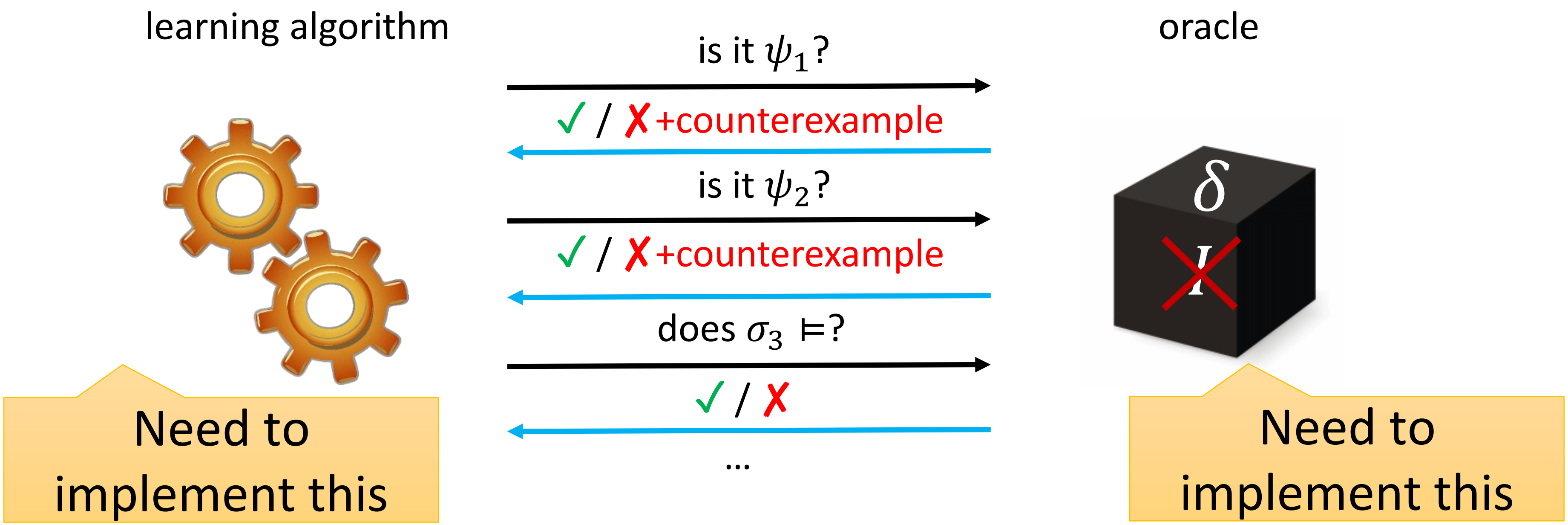
  drop literals from σ'

  while BMC$(\delta, \sigma', k) \cap \text{Bad} = \emptyset$

Complexity bounds from exact classification algorithms

Rich SAT queries allow exponentially faster inference

Invariant Inference with Equivalence & Membership Queries



Thm. In general, in the Hoare-query model, **no efficient way** to implement a teacher for equivalence and membership queries

[CAV'14] ICE: A Robust Framework for Learning Invariants. Garg, Löding, Madhusudan, Neider

[POPL'20] Complexity and Information in Invariant Inference. Feldman, Immerman, Shoham, Sagiv

Invariant Inference with Equivalence & Membership Queries

learning algorithm

teacher

is it ψ_1?

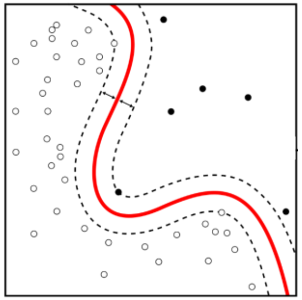
...

Sufficient conditions for

exact learning algorithms $\Longrightarrow$ invariant inference algorithms

Thm. In general, in the Hoare-query model, **no efficient way** to implement a teacher for equivalence and membership queries

From Learning to Inference



Exact **learning**
DNF formulas

ψ := false

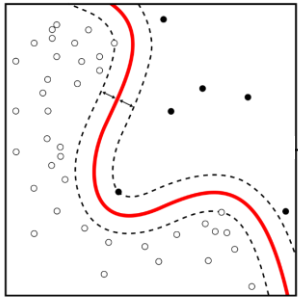
while σ' counterexample
to Equivalence(ψ):

ψ := $\psi \vee$ **generalize**(σ')

generalize(σ'):
drop literals from σ'
while Membership(σ')= ✓

[CACM'84] A Theory of the Learnable. Valiant
[ML'87] Queries and Concept Learning. Angluin
[ML'95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

From Learning to Inference



Exact **learning** DNF formulas $\Longrightarrow$

ψ := false

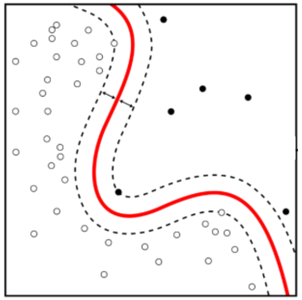
while σ' counterexample
to **Equivalence**(ψ): → **Inductive**(I)

ψ := $\psi \vee$ **generalize**(σ')

generalize(σ'):
drop literals from σ'
while **Membership**(σ')= ✓ → $\text{BMC}(\sigma', k) \cap \textbf{Bad} = \emptyset$

[CACM'84] A Theory of the Learnable. Valiant
[ML'87] Queries and Concept Learning. Angluin
[ML'95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

From Learning to Inference



Exact **learning** DNF formulas $\Longrightarrow$

Inferring DNF invariants

ψ := false

while σ' counterexample
    to **Equivalence**($\boldsymbol{\psi}$):

    ψ := $\psi \vee$ **generalize**(σ')

generalize(σ'):
    drop literals from σ'
    while **Membership**($\boldsymbol{\sigma'}$) = ✓

I := false

while $(\_, \sigma')$ counterexample
    to **Inductive**($\boldsymbol{I}$):

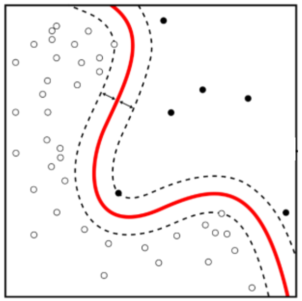
    I := $I \vee$ **generalize**(σ')

generalize(σ'):
    drop literals from σ'
    while **BMC**$(\boldsymbol{\sigma'}, \boldsymbol{k}) \cap$ **Bad** $= \emptyset$

[CACM'84] A Theory of the Learnable. Valiant
[ML'87] Queries and Concept Learning. Angluin
[ML'95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

[CAV'03] Interpolation and SAT-Based Model Checking, McMillan
[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah

From Learning to Inference



Efficiently

Exact **learning**
DNF formulas

$\Longrightarrow$

Efficiently

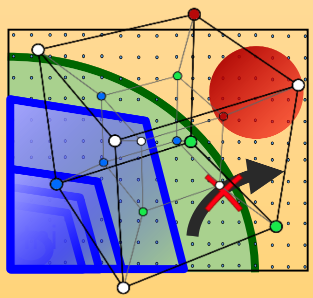
Inferring
DNF invariants

ψ := false

while σ' counte… to **Equiva**…

ψ := ψ ∨ **gen**…

generalize(σ'):
drop literals from σ'
while **Membership**(σ')= ✓



The invariant is ***k*-fenced**

…′) counterexample
Inductive(I):

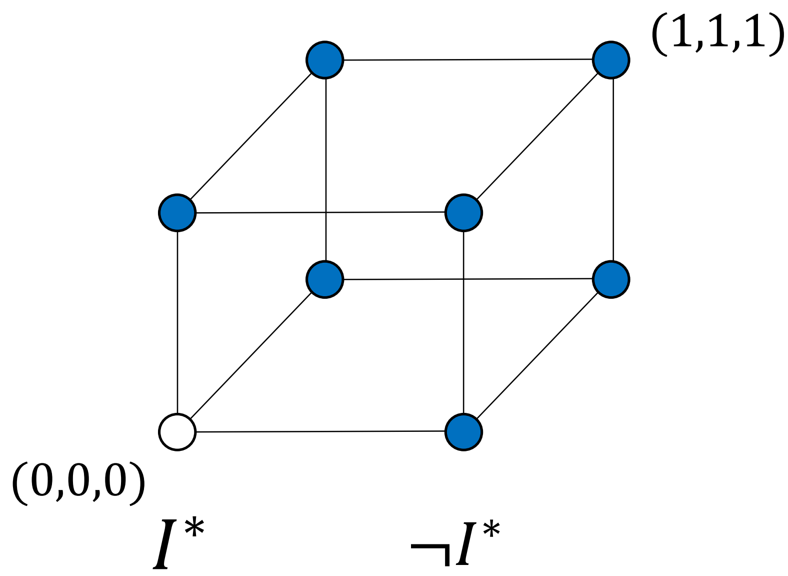
∨ **generalize**(σ')

generalize(σ'):
drop literals from σ'
while **BMC**(σ', k) ∩ **Bad** = ∅

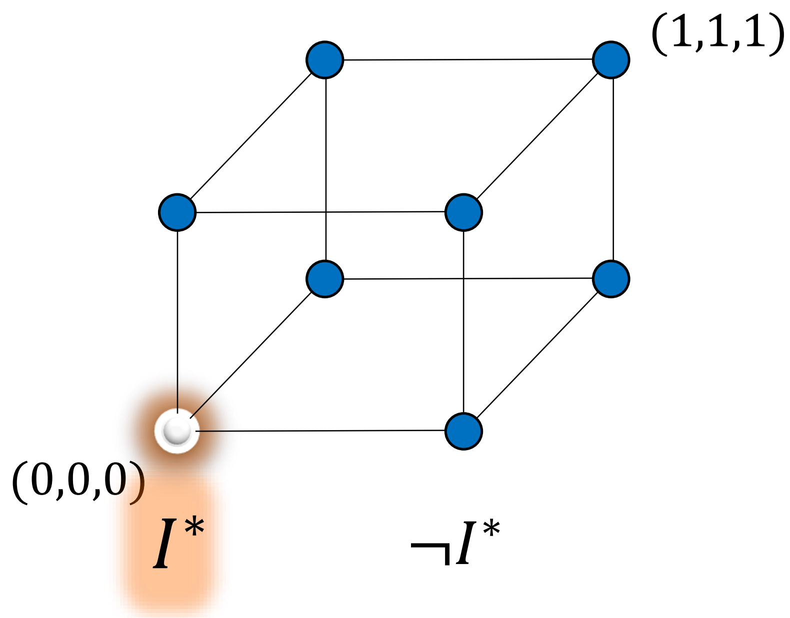
[CACM’84] A Theory of the Learnable. Valiant
[ML’87] Queries and Concept Learning. Angluin
[ML’95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

[CAV’03] Interpolation and SAT-Based Model Checking, McMillan
[HVC’12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah

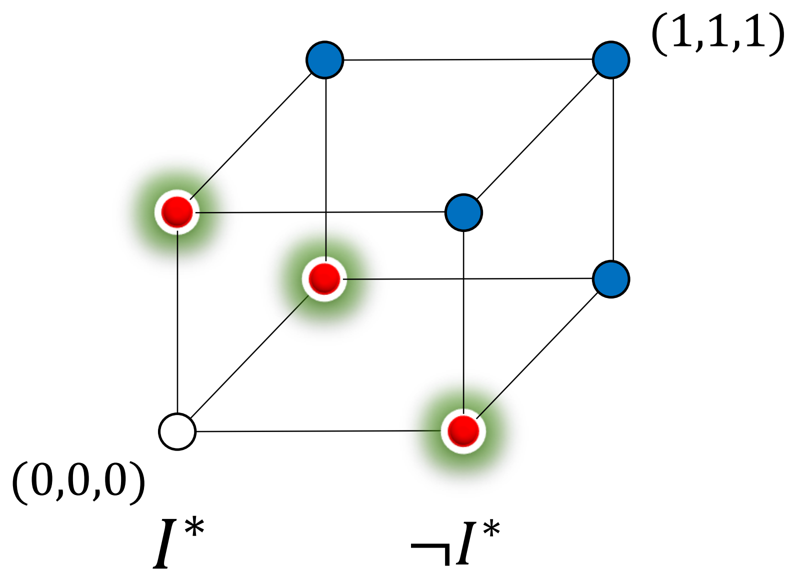
k-Fenced Invariants



k-Fenced Invariants

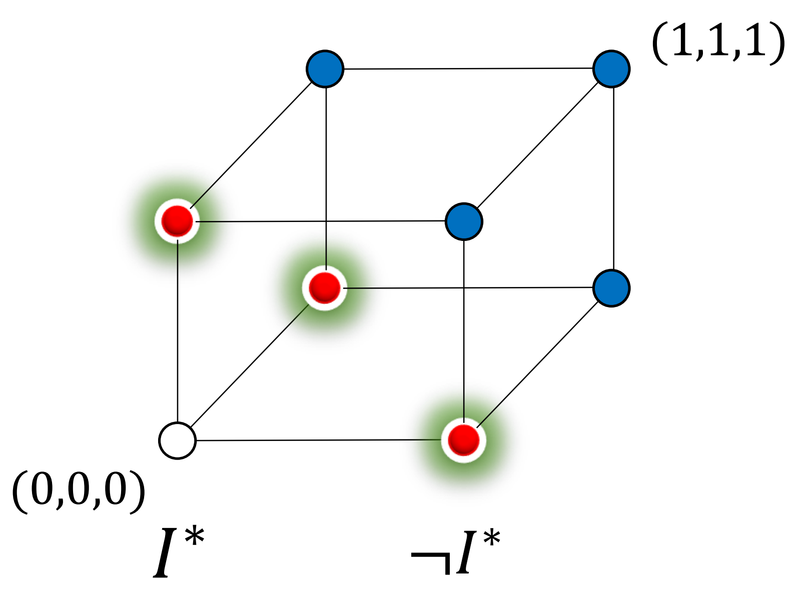


k-Fenced Invariants



$$\partial^-(I^*)$$

k-Fenced Invariants

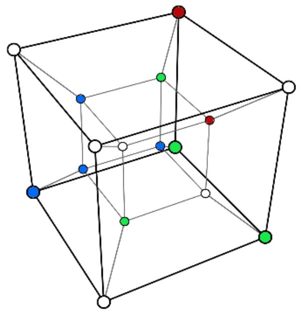


All the states in $\partial^-(I^*)$
can get to a bad state in at most k steps

Complexity Upper Bounds

Thm. Interpolation-based inference finds an invariant in a polynomial number of SAT queries when

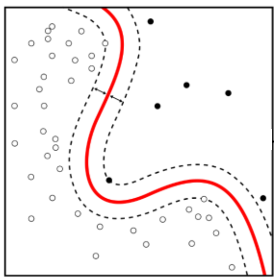
$\exists I^*.$



Fence condition: the Hamming boundary of I^* reaches bad states in k steps

No negated variables

+



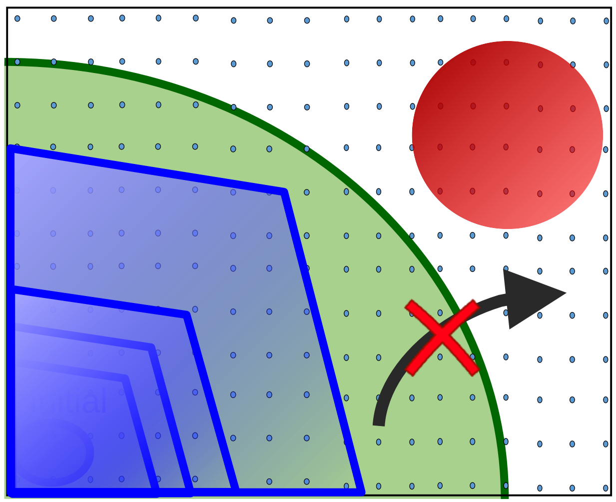
I^* is a short monotone DNF (via Angluin)
or
I^* is a short almost-monotone DNF (via Bshouty)

$O(1)$ terms with negated variables

[POPL'21] Learning the Boundary of Inductive Invariants. Feldman, Sagiv, Shoham, Wilcox

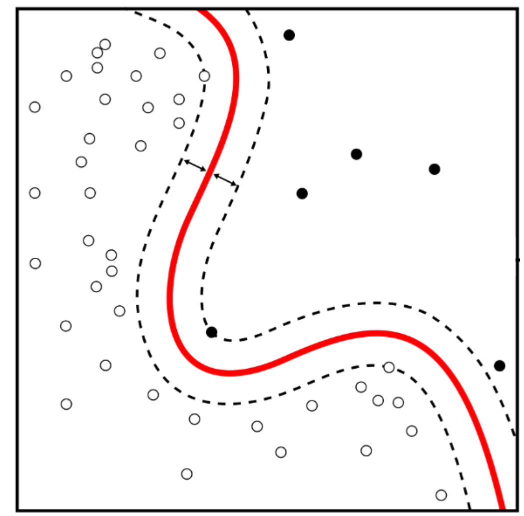
Conclusion

Invariant Inference



vs.

Exact Concept Learning



- Query-based learning models for invariant inference
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from classification algorithms