SAT-Based Invariant Inference and Its Relation to Concept Learning

Sharon Shoham

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

erc Supervised Verification of Infinite-State Systems

Yotam Feldman

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

Neil Immerman

UNIVERSITY OF MASSACHUSETTS AMHERST 1863

Mooly Sagiv

TEL AVIV UNIVERSITY אוניברסיטת תל אביב

CERTORA

James R. Wilcox

UNIVERSITY of WASHINGTON

SAT-Based Invariant Inference

- predicate abstraction
  [CAV’97, POPL’02]
- symbolic abstraction
  [VMCAI’04,’16]
- interpolation
  [CAV’03, TACAS’06]
- IC3/PDR
  [VMCAI’11, FMCAD’11]
- abduction
  [OOPSLA’13]
- SyGuS
  [FMCAD’13,…]
- ICE learning
  [CAV’14, POPL’15]
- …

Why do they succeed?

Why do they fail?

(How can we make them better?)

Goal

Understand SAT-based invariant inference
from the perspective of exact learning with queries

[POPL’20] Complexity and information in invariant inference. Feldman, Immerman, Sagiv, Shoham
[POPL’21] Learning the boundary of inductive invariants. Feldman, Sagiv, Shoham, Wilcox
[POPL’22] Property-directed reachability as abstract interpretation in the monotone theory. Feldman, Sagiv, Shoham, Wilcox
[SAS’22] Invariant Inference With Provable Complexity From the Monotone Theory. Feldman, Shoham

Safety of Transition Systems

Safety: no bad state is reachable from the initial states

Init:

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

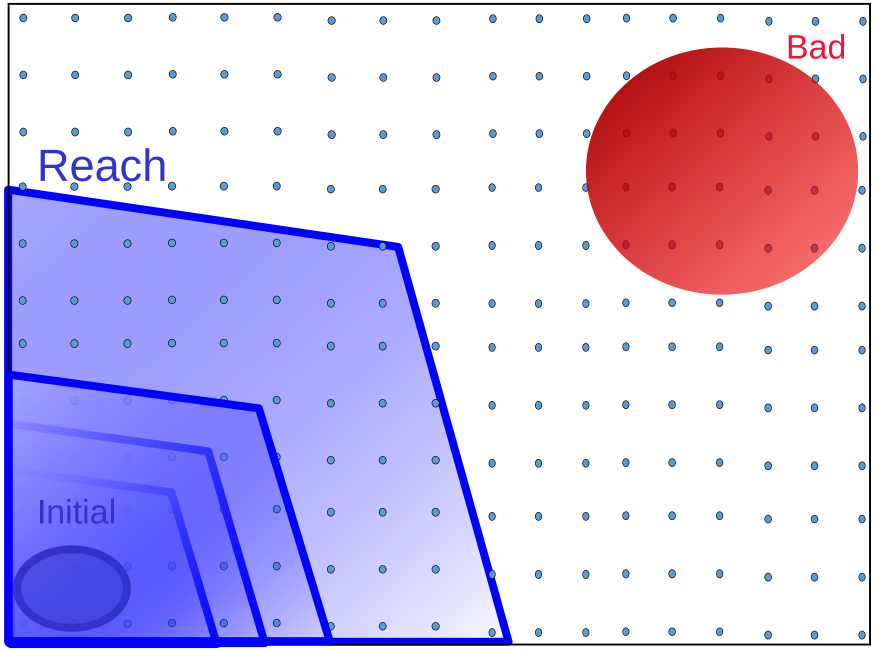
Bad:

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

δ:

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

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



Inductive Invariants

Safety: no bad state is reachable from the initial states

Init:

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

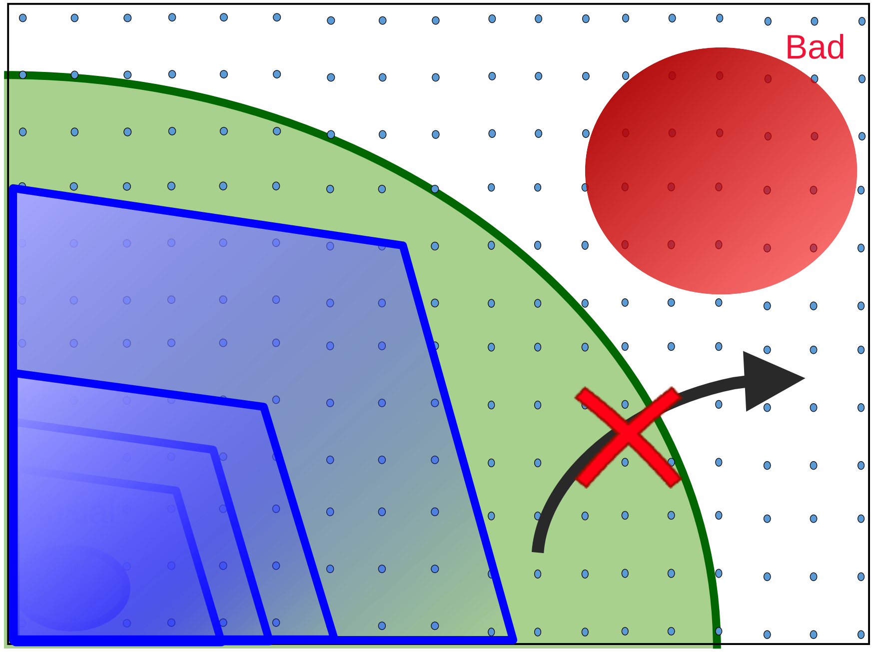
Bad:

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

δ:

$y_1, \ldots, y_n \coloneqq *$

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



Initiation: $\text{Init} \subseteq I$

Safety: $I \cap \text{Bad} = \emptyset$

Consecution: $\{I\}\ \delta\ \{I\}$

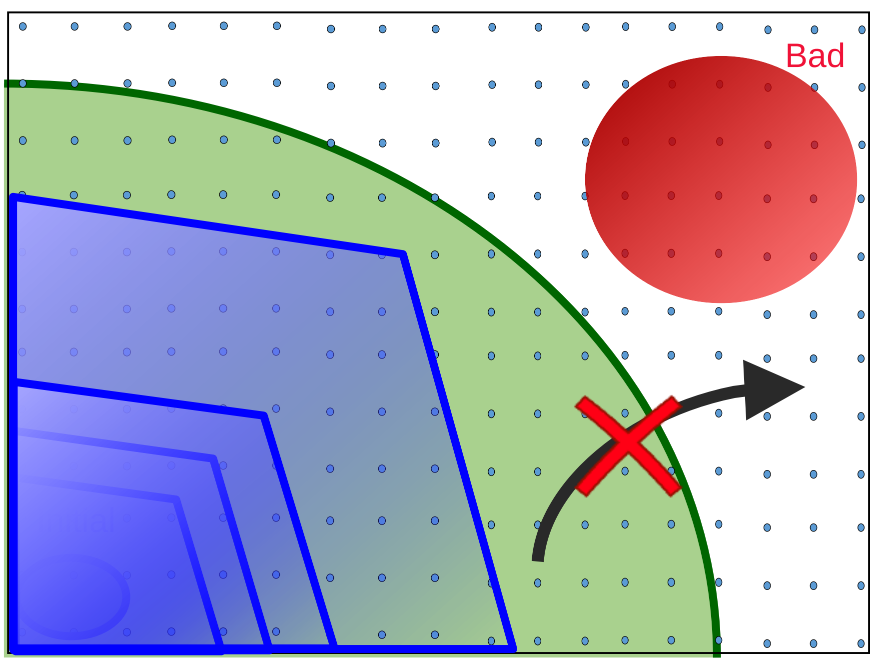
Inductive Invariants

Safety: no bad state is reachable from the initial states

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

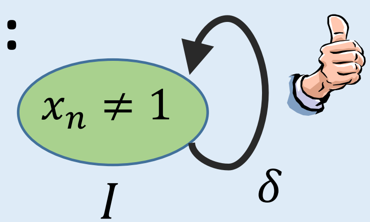
Bad:
$(x_1, \ldots, x_n) = 1 \ldots 1$

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



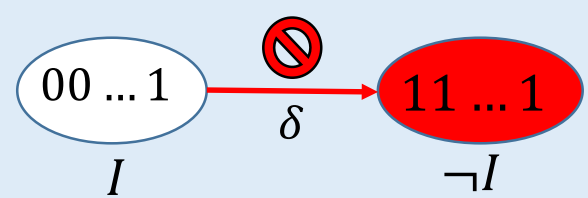
I: $x_n \neq 1$

Inductive:



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

Not inductive:



Invariant Inference

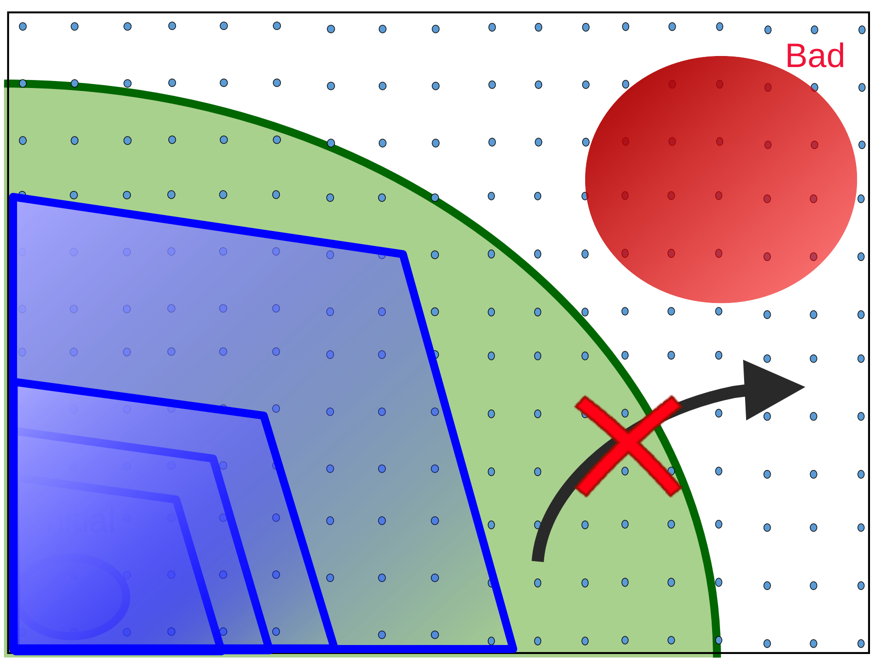
Safety: no bad state is reachable from the initial states

Goal: Find inductive invariants automatically

Bad:

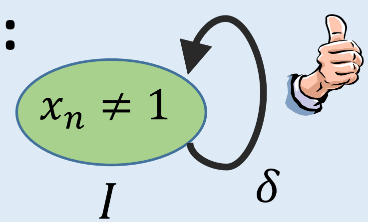
$(x_1, \dots, x_n) = 1 \dots 1$

$2 \cdot (y_1, \dots, y_n) \pmod{2^n}$



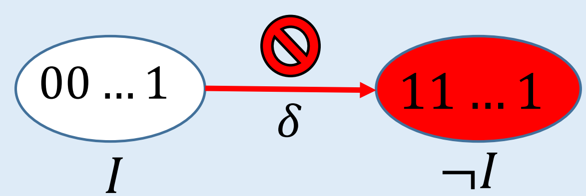
I: $x_n \neq 1$

Inductive:



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

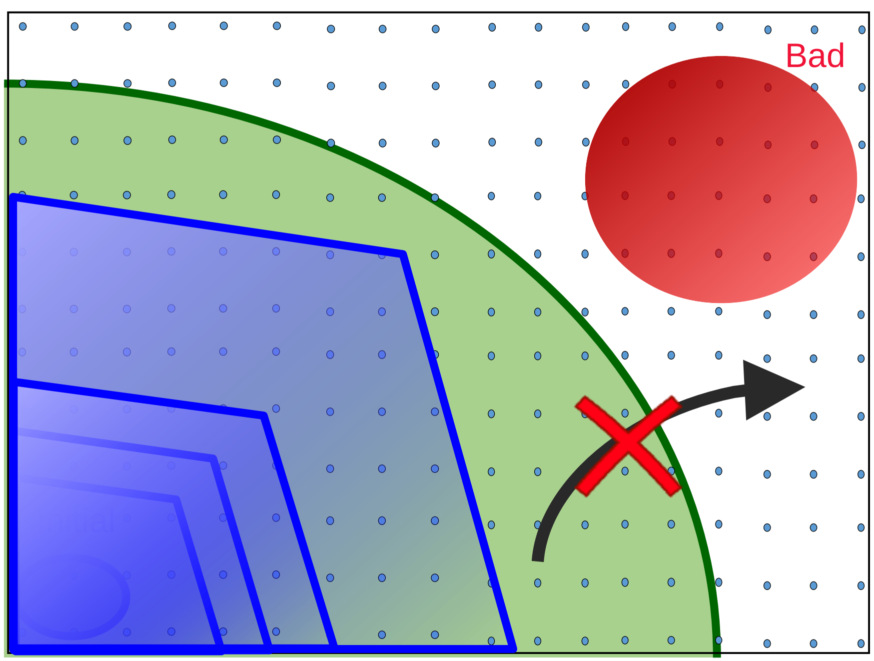
Not inductive:



SAT-based Invariant Inference

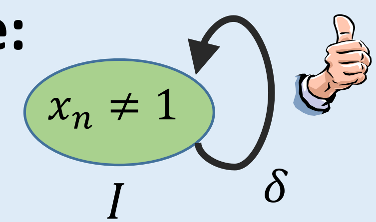
Goal: Find inductive invariants automatically

Means: Employ a SAT solver



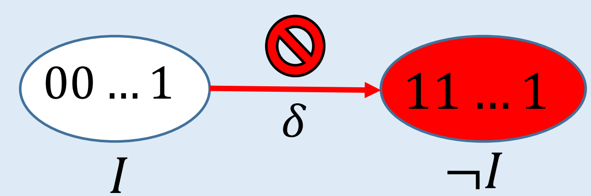
I: $x_n \neq 1$

Inductive:



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

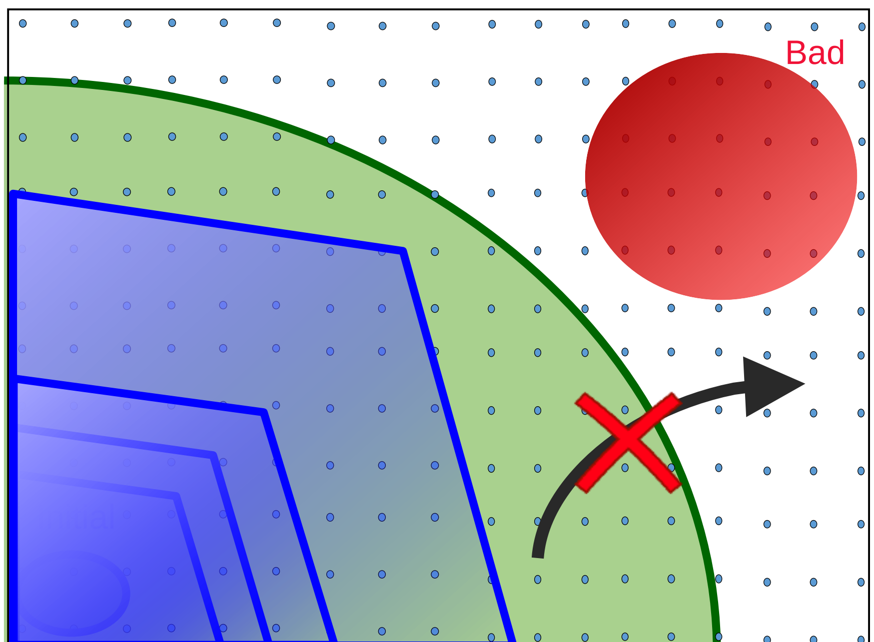
Not inductive:



SAT-based Invariant Inference

Goal: Find inductive invariants automatically

Means: Employ a SAT solver



Init, Bad: formulas over V

$\boldsymbol{\delta}$: formula over V, V'

SAT query Examples:

Initiation: **Init** $\wedge \neg \boldsymbol{I}$ unsat?

Safety: $\boldsymbol{I} \wedge$ **Bad** unsat?

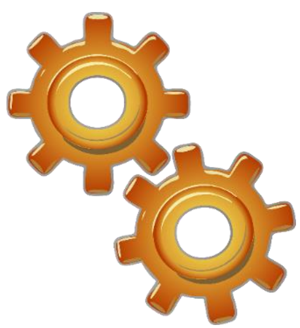
Cons.: $\boldsymbol{I} \wedge \boldsymbol{\delta} \wedge \neg \boldsymbol{I'}$ unsat?

* $I' = I[V \mapsto V']$

Exact Concept Learning with Equivalence & Membership Queries

Goal: learn an unknown concept φ

learning algorithm



is it ψ_1?

✓ / ✗+counterexample

is it ψ_2?

✓ / ✗+counterexample

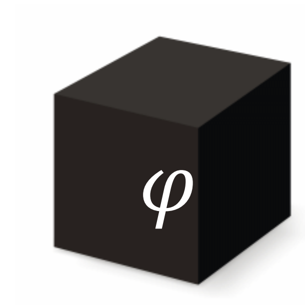
does $\sigma_3 \vDash$?

✓ / ✗

...

Membership Equivalence

oracle

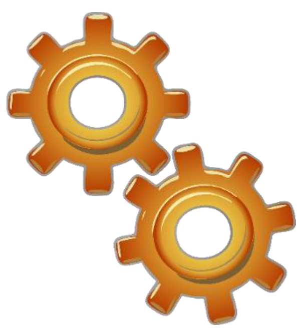


[ML'87] Queries and Concept Learning. Angluin

SAT-Based Invariant Inference as Inference with Queries

Goal: infer an unknown inductive invariant I

learning algorithm ↦ inference algorithm



Q →

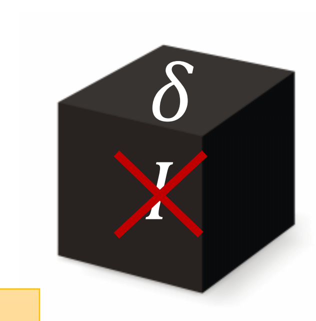
A ←

Q →

A ←

Q

oracle ↦ SAT-solver

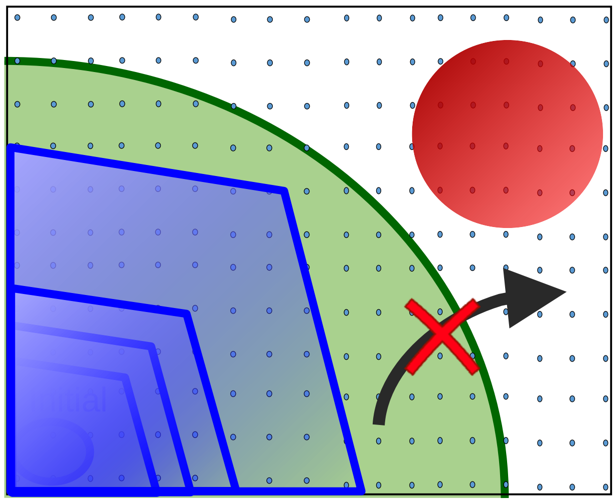


Which SAT queries?

Algorithms cannot access the transition relation directly, only through SAT queries

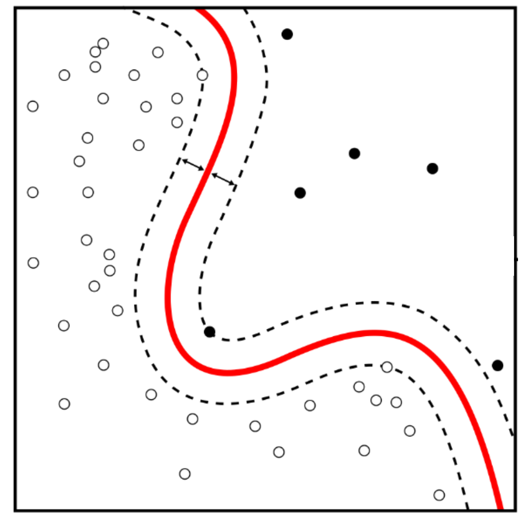
This Talk

Invariant Inference



vs.

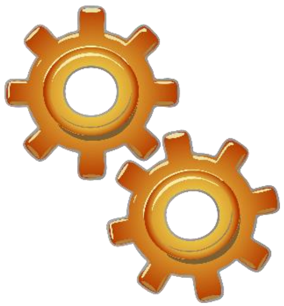
Exact Concept Learning



- Query-based learning models for invariant inference
- Complexity lower and upper bounds for each model
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from concept learning algorithms

Inductiveness-Query Model

inference algorithm



α_1 inductive? →

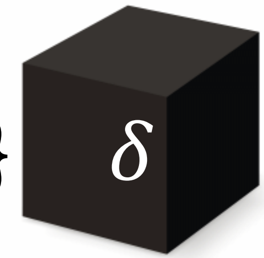
← ✓ / ✗+counterexample

…

α_m inductive? →

← ✓ / ✗+counterexample

inductiveness-query oracle

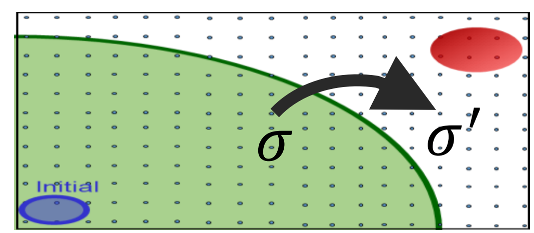
$\{\alpha_i\}$  $\{\alpha_i\}$

$\boldsymbol{\alpha_i \wedge \delta \wedge \neg\alpha_i'}$ unsat?

Cex to Induction (CTI):
Transition (σ, σ') of δ s.t.
$\sigma \vDash \alpha_i, \quad \sigma' \vDash \neg\alpha_i$

ICE framework - Learn from examples:

Positive : $\sigma \vDash I$ (e.g., initial)
Negative: $\sigma \nvDash I$ (e.g., bad)
Implication: $\sigma \vDash I$ implies $\sigma' \vDash I$ (CTI)

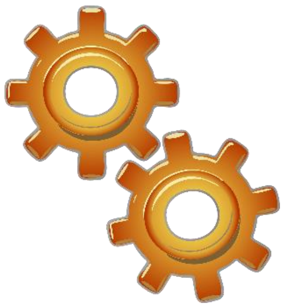


* $\boldsymbol{\alpha_i'}=\boldsymbol{\alpha_i}[V \mapsto V']$

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

Inductiveness-Query Model

inference algorithm



α_1 inductive?

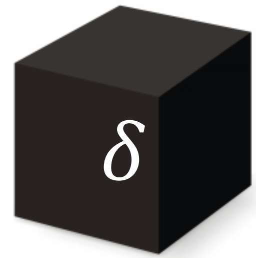
✓ / ✗+counterexample

…

α_m inductive?

✓ / ✗+counterexample

inductiveness-query oracle

$\{\alpha_i\}$ 

$\{\alpha_i\}$

$\boldsymbol{\alpha_i \wedge \delta \wedge \neg\alpha_i'}$ unsat?

Cex to Induction (CTI):
Transition (σ, σ') of δ s.t.
$\sigma \vDash \alpha_i, \quad \sigma' \vDash \neg\alpha_i$

ICE framework - Learn from examples:

Positive : $\sigma \vDash I$ (e.g., initial)

Negative: $\sigma \nvDash I$ (e.g., bad)

Implication: $\sigma \vDash I$ implies $\sigma' \vDash I$ (CTI)

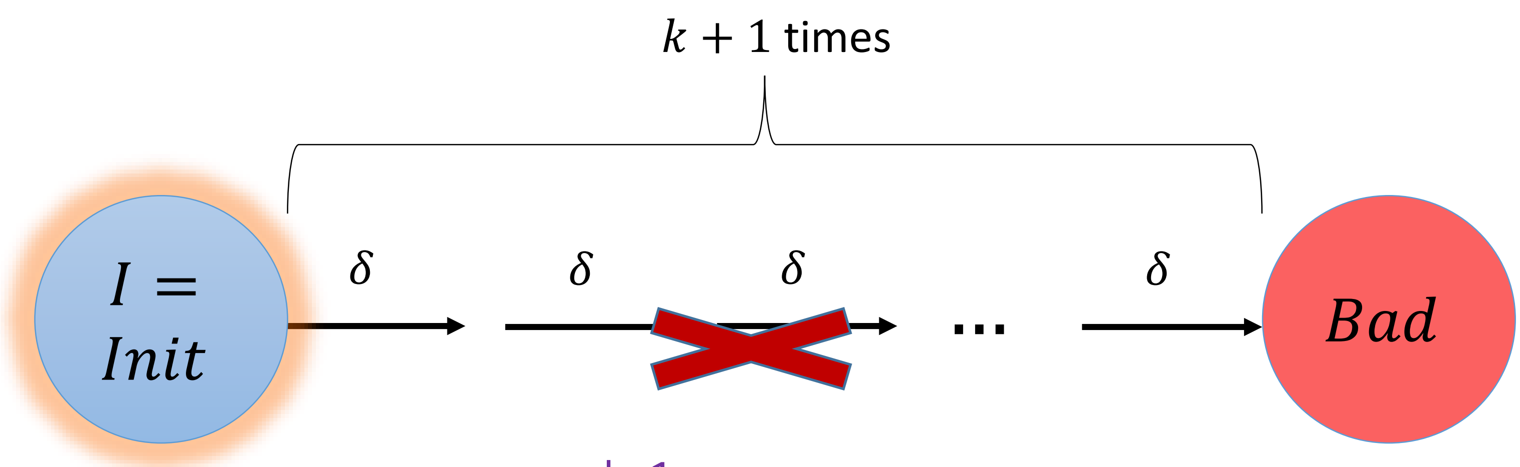
Is it sufficient to capture existing SAT-based algorithms?

* $\alpha_i' = \alpha_i[V \mapsto V']$

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

Interpolation-Based Inference

$I = Init$ **Inductive ?**

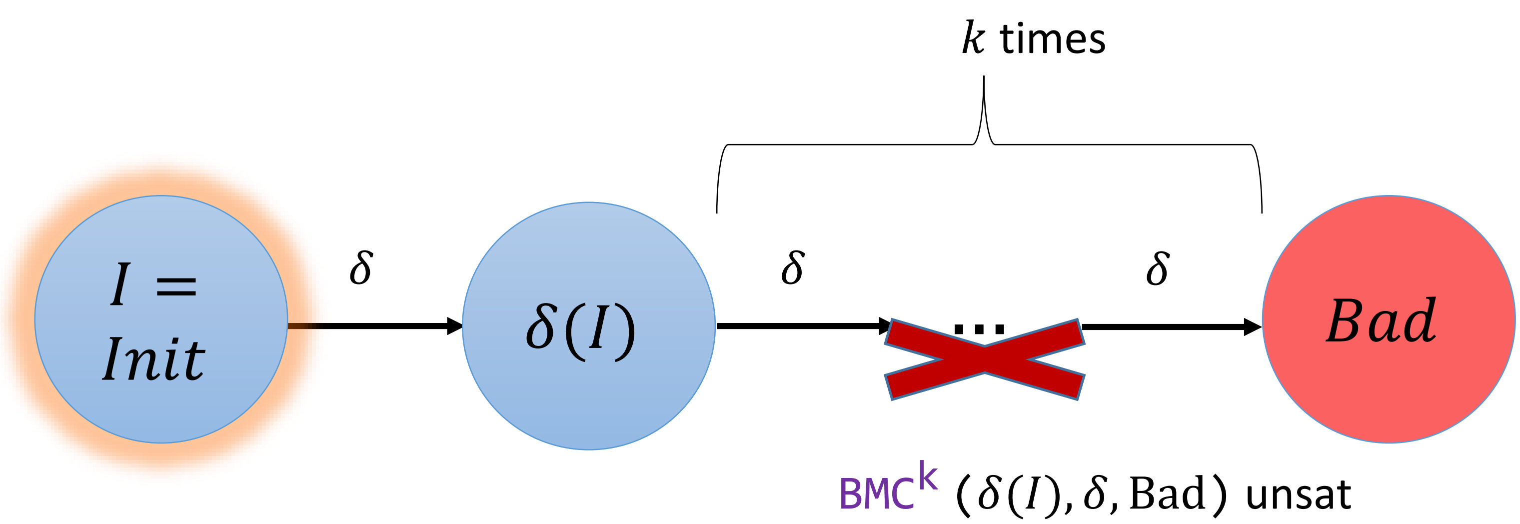


BMC^{k+1} $(I, \delta, \mathrm{Bad})$ unsat

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

Interpolation-Based Inference

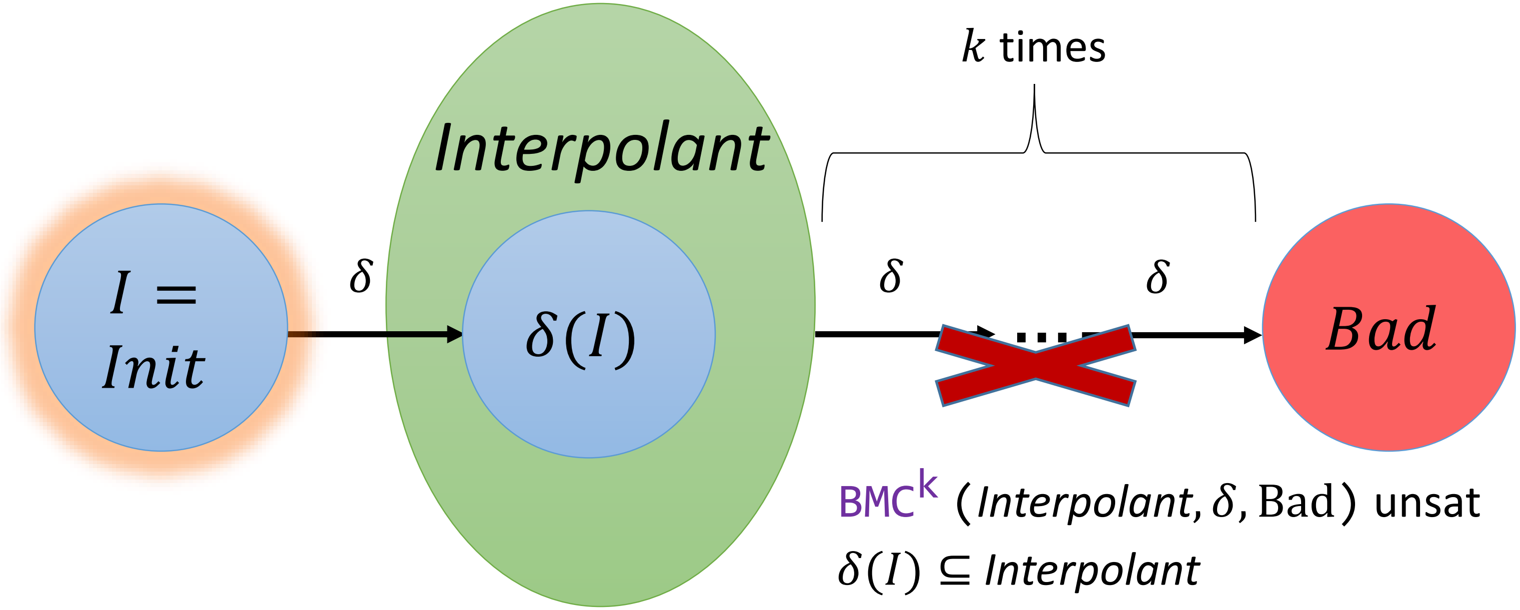
$I = Init$



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

Interpolation-Based Inference

$$I = Init$$



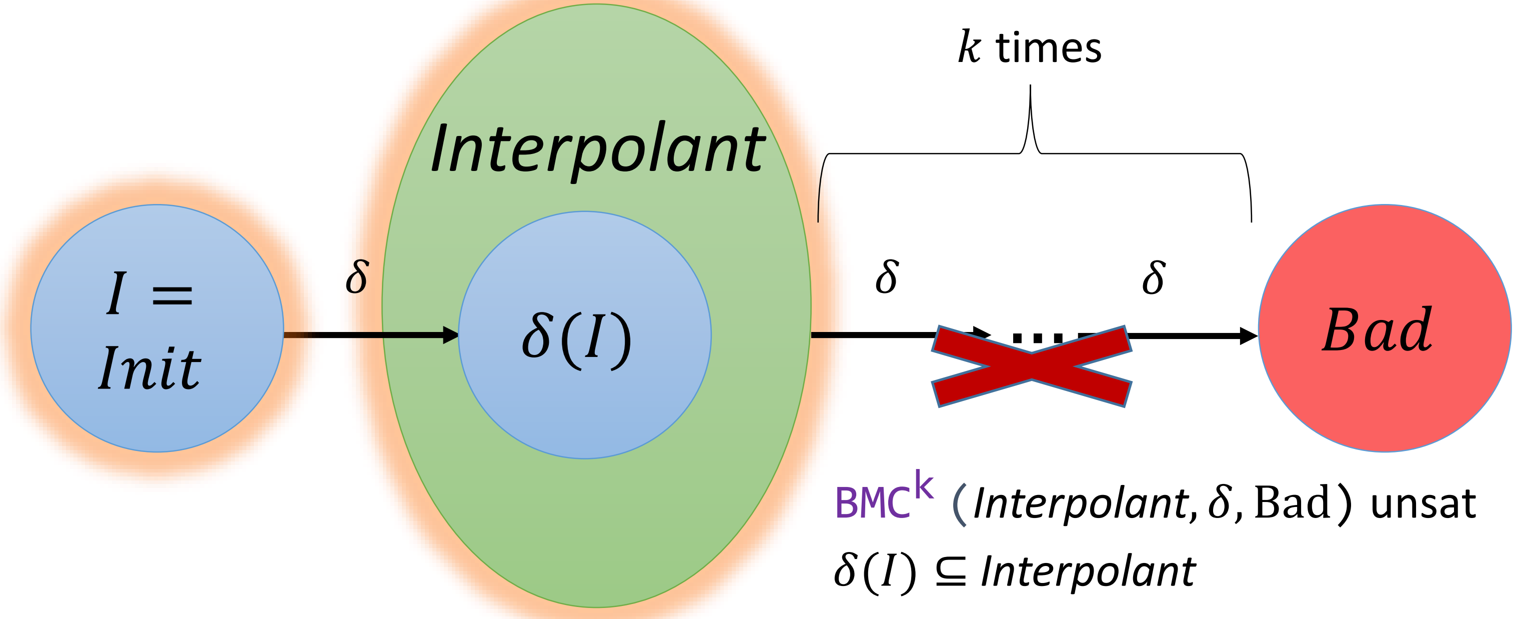
BMC^k $(Interpolant, \delta, \mathrm{Bad})$ unsat

$\delta(I) \subseteq Interpolant$

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

Interpolation-Based Inference

$$I = Init \vee Interpolant$$



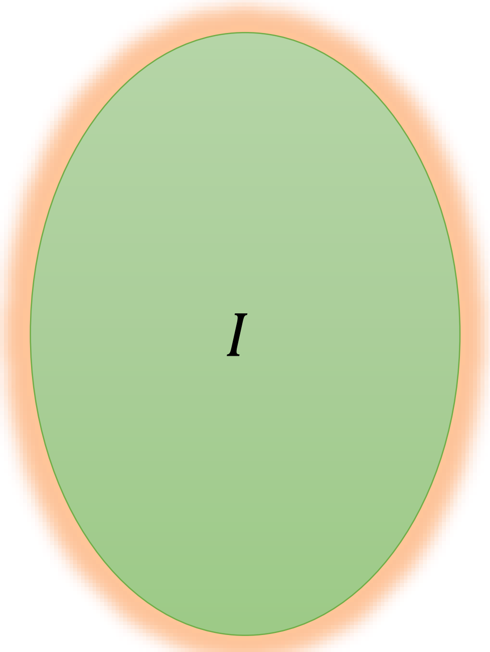
BMC^k $(Interpolant, \delta, \mathrm{Bad})$ unsat

$\delta(I) \subseteq Interpolant$

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

Interpolation-Based Inference

$$I = Init \vee Interpolant$$

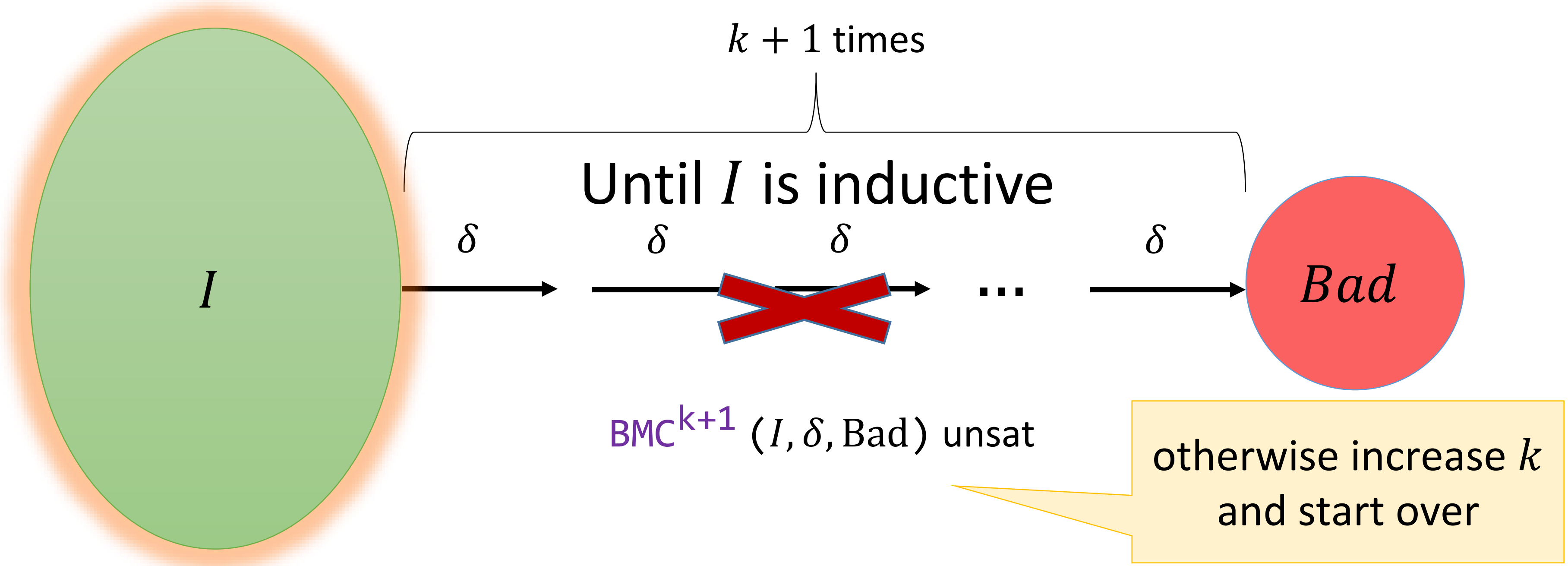


Inductive ?

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

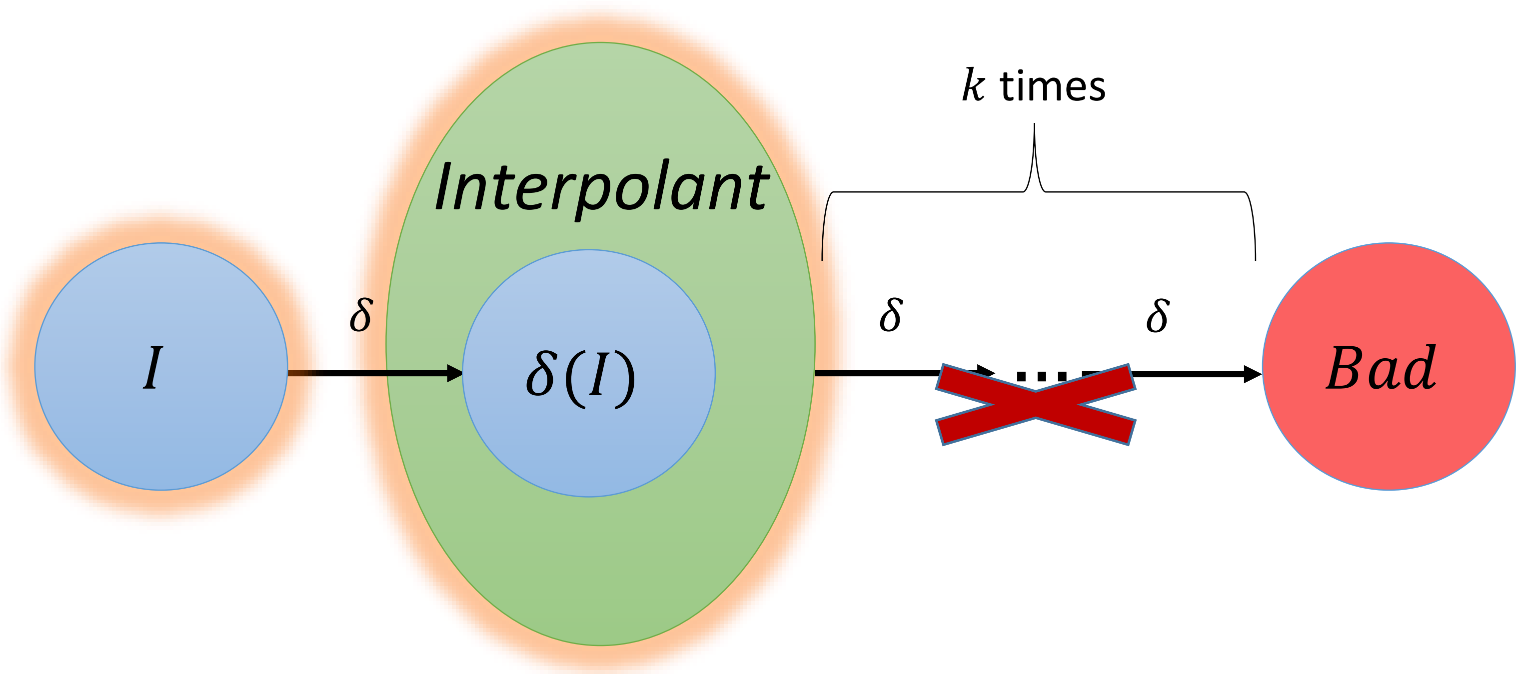
Interpolation-Based Inference

$$I = Init \lor Interpolant \lor Interpolant_2 \lor \ \ldots$$



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

Computing an Interpolant



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

Model-Based Interpolation

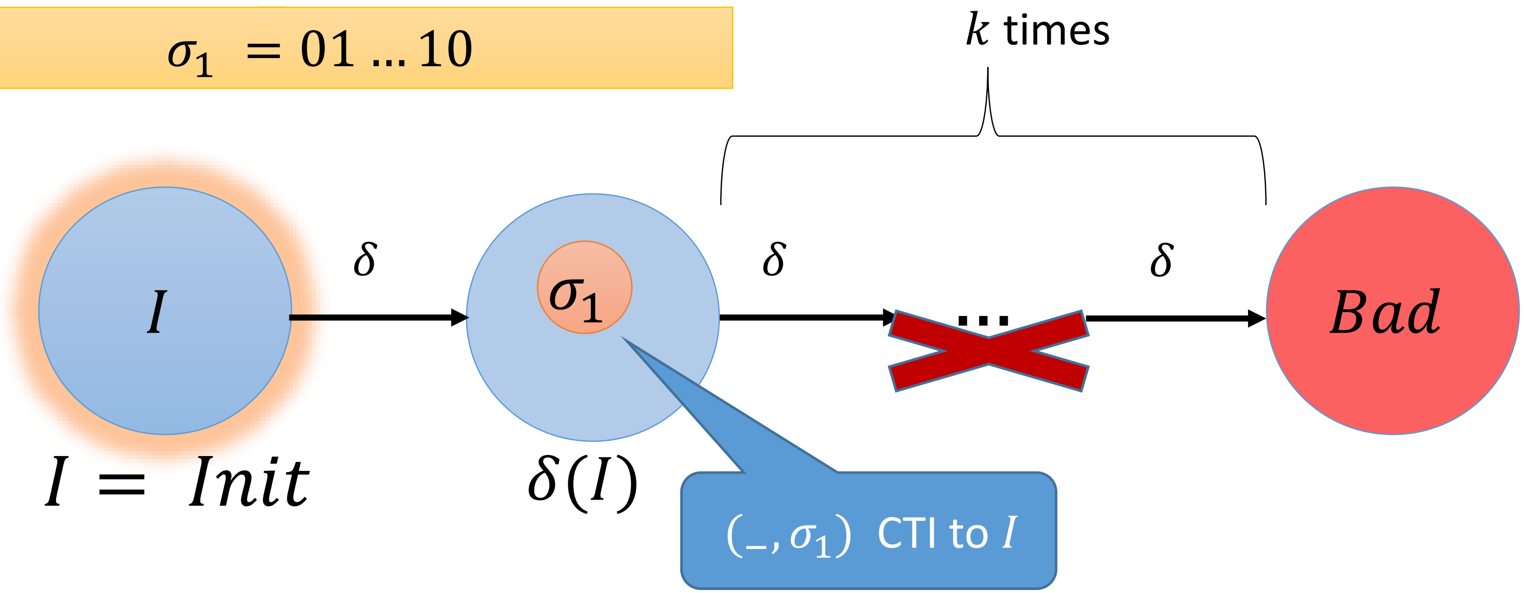
Init:
$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

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

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

$$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_{n-1} = 1 \wedge x_n = 0)$$

$$\sigma_1 = 01 \ldots 10$$



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

Model-Based Interpolation

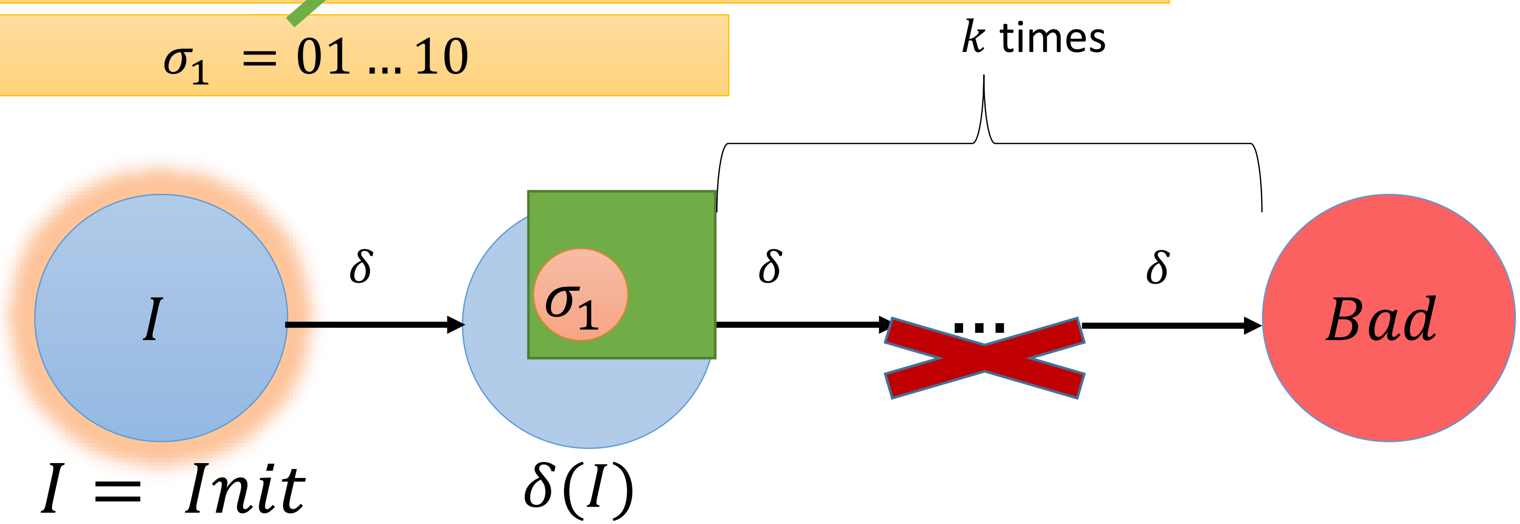
Init:
$(x_1, \dots, x_n) \coloneqq 0 \dots 0$

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

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

$Interpolant_1 = (\cancel{x_1 = 0} \wedge x_2 = 1 \wedge \cdots \wedge x_{n-1} = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \dots 10$



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

Model-Based Interpolation

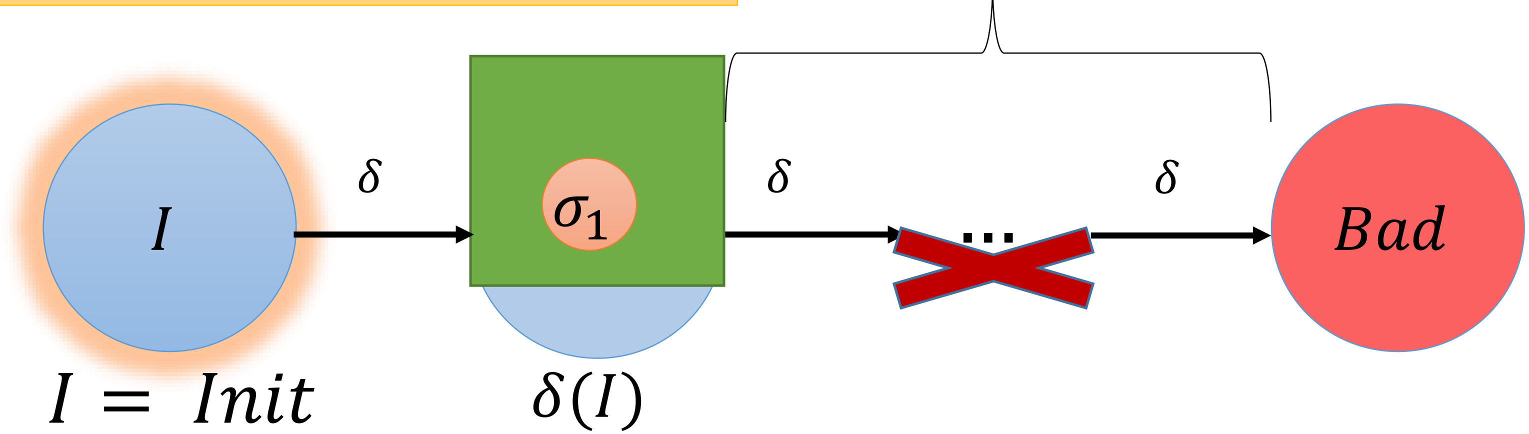
Init:
$(x_1, \ldots, x_n) \;\coloneqq\; 0 \ldots 0$

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

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

$Interpolant_1 = (\cancel{x_1 = 0} \wedge \cancel{x_2 = 1} \wedge \cdots \wedge x_{n-1} = 1 \wedge x_n = 0)$

$\sigma_1 \;= 01 \ldots 10$



[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:

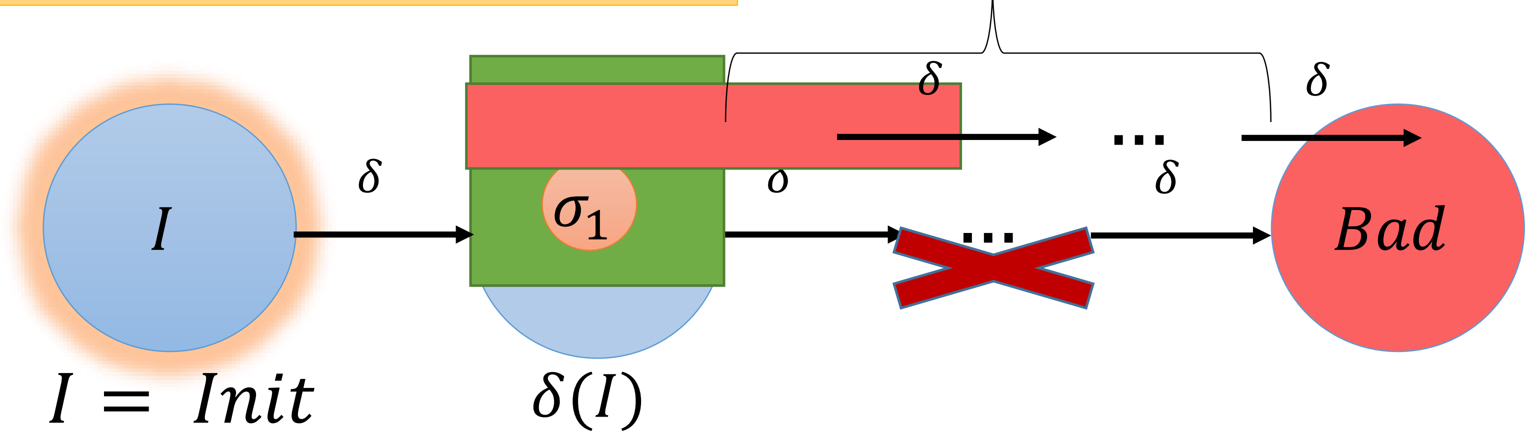
$y_1, \ldots, y_n \coloneqq *$

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

$Interpolant_1 = (\cancel{x_1 = 0} \wedge \cancel{x_2 = 1} \wedge \cdots \wedge x_{n-1} = 1 \wedge \cancel{x_n = 0})$

$\sigma_1 = 01 \ldots 10$

k times



[HVC'12] Computing Interpolants without Proofs. Chockler, Ivrii, Matsliah

[LPAR'13] Instantiations, Zippers and EPR Interpolation. Bjørner, Gurfinkel, Korovin, Lahav

Model-Based Interpolation

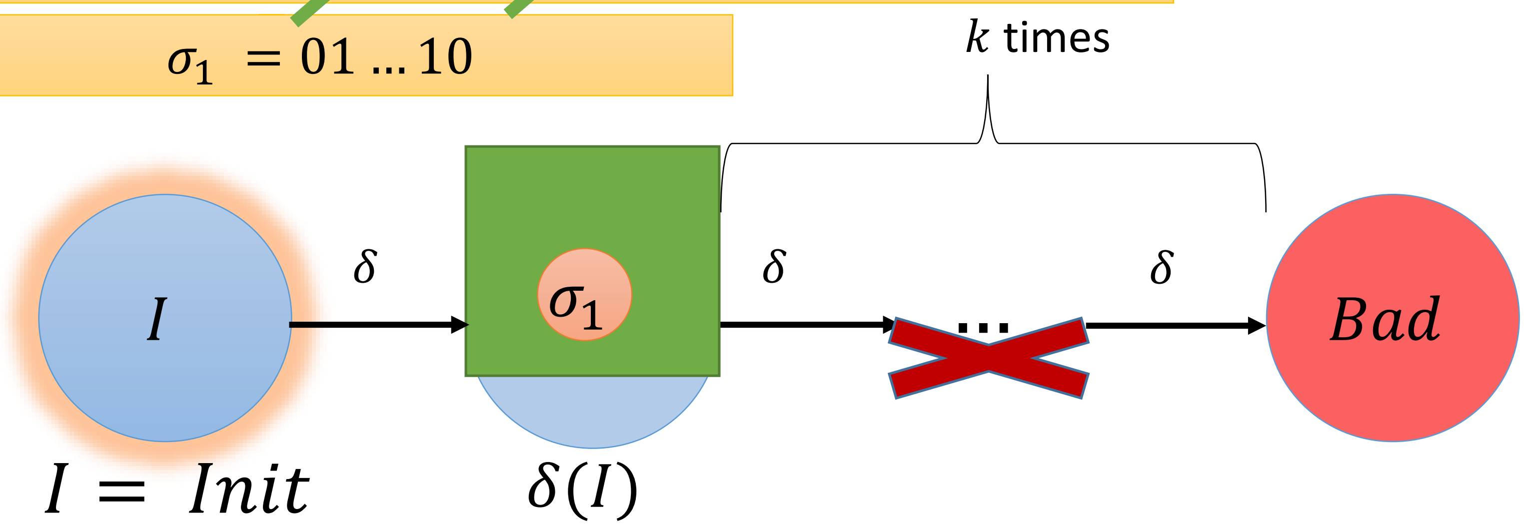
<u>Init</u>:
$(x_1, \ldots, x_n) \coloneqq 0 \ldots 0$

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

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

$Interpolant_1 = (\cancel{x_1 = 0} \wedge \cancel{x_2 = 1} \wedge \cdots \wedge x_{n-1} = 1 \wedge x_n = 0)$

$\sigma_1 = 01 \ldots 10$



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

Model-Based Interpolation

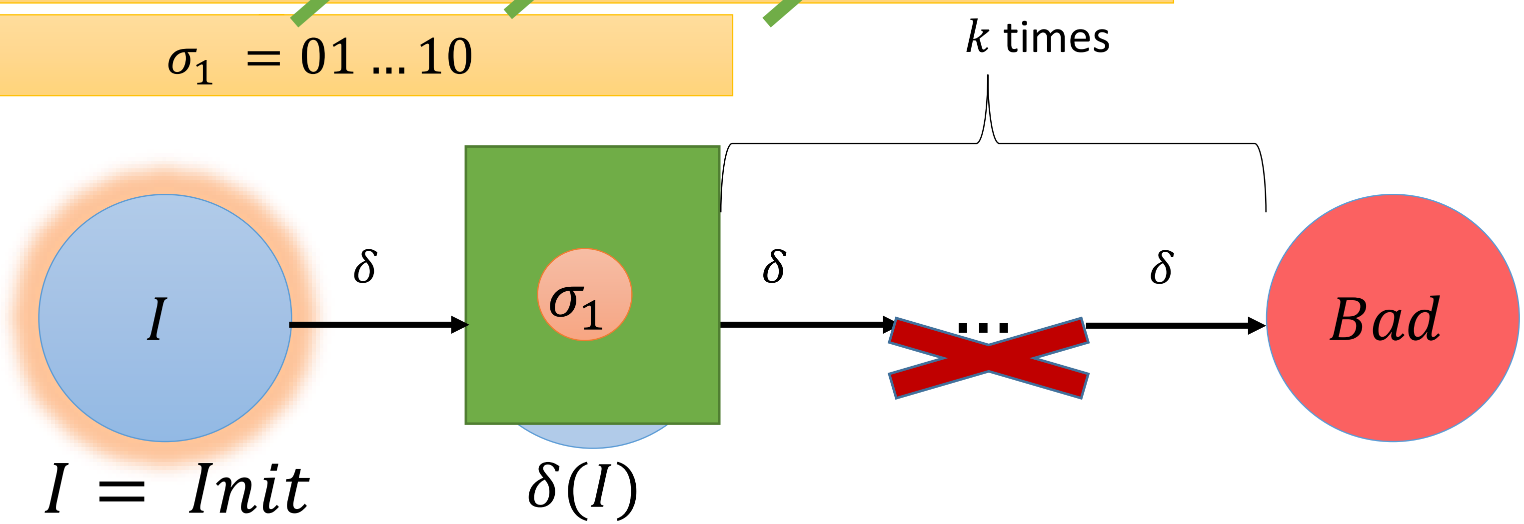
Init:
$(x_1, \dots, x_n) \;:=\; 0 \dots 0$

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

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

$$Interpolant_1 = (x_1 = 0 \wedge x_2 = 1 \wedge \cdots \wedge x_{n-1} = 1 \wedge x_n = 0)$$

$$\sigma_1 \;= 01 \dots 10$$



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

Model-Based Interpolation

Init:

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

Bad:

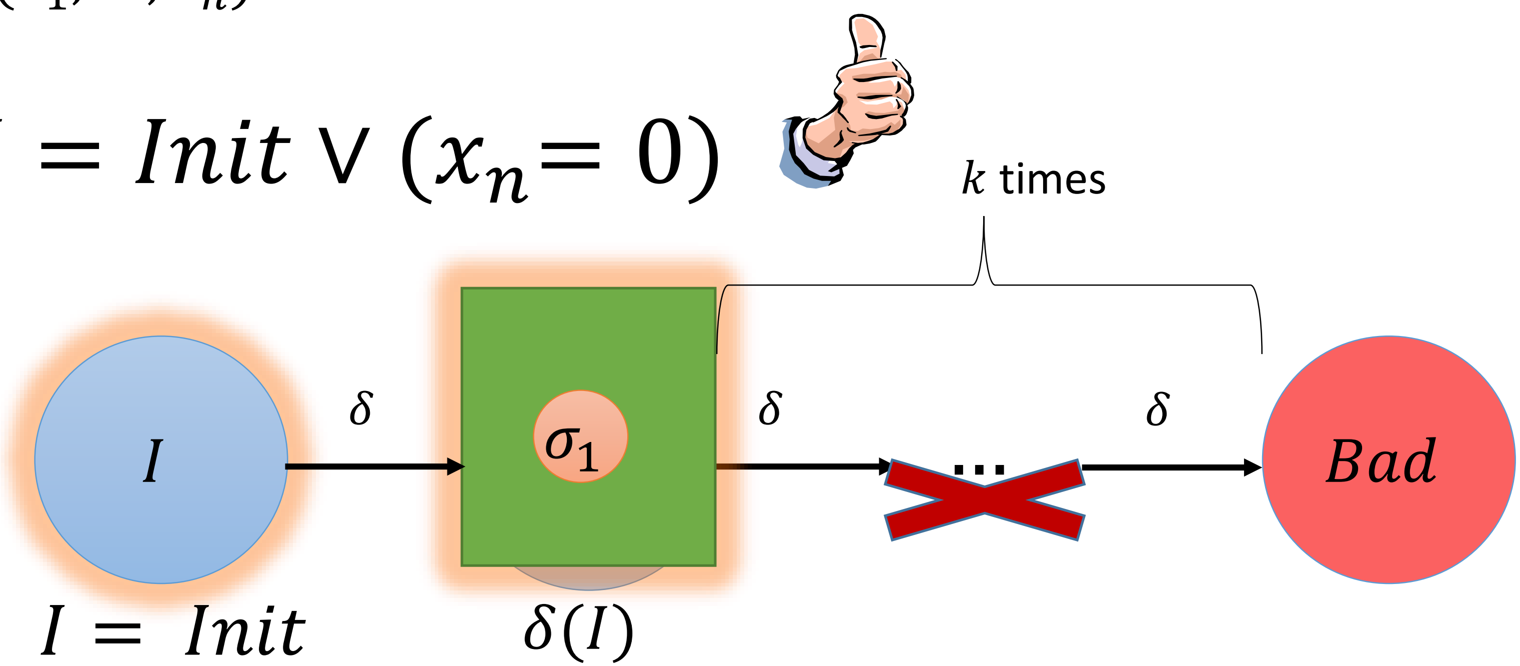
$(x_1, \dots, x_n) \;=\; 1 \dots 1$

δ:

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

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

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



[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:

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

ITP-k:

I := false

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

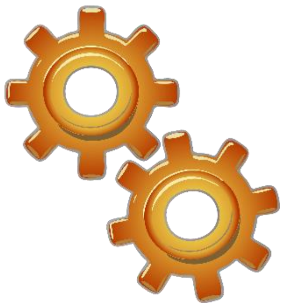
I := I ∨ **generalize**(σ')

generalize(σ'):

drop literals from σ'
while BMC$^k(\sigma', \delta, \text{Bad})$ unsat

Inductiveness-Query Model

inference algorithm



α_1 inductive?

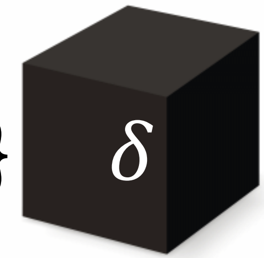
✓ / ✗+counterexample

…

α_m inductive?

✓ / ✗+counterexample

inductiveness-query oracle

$\{\alpha_i\}$  $\{\alpha_i\}$

I := false

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

I := $I \vee$ **generalize**(σ')

generalize(σ'):

drop literals from σ'

while BMC$^k(\sigma', \delta, \mathrm{Bad})$ unsat

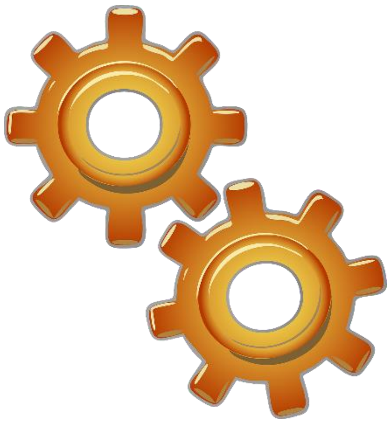
I inductive?

✓ / ✗

?

Hoare-Query Model

inference algorithm



Hoare-query oracle

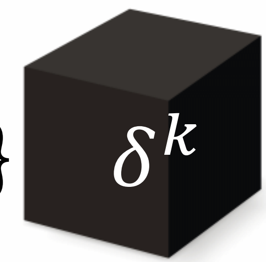
$\{\alpha_1\}\delta^k\{\beta_1\}?$

✓ / ✗ +counterexample

…

$\{\alpha_m\}\delta^k\{\beta_m\}?$

✓ / ✗ +counterexample

$\{\alpha_i\}$ 

$\{\beta_i\}$

$\mathrm{BMC}^k\ (\alpha_i, \delta, \neg\beta_i)$ unsat?

Trace $(\sigma_0, \ldots, \sigma_k)$ of δ s.t.
$\sigma_0 \vDash \alpha_i, \quad \sigma_k \vDash \neg\beta_i$

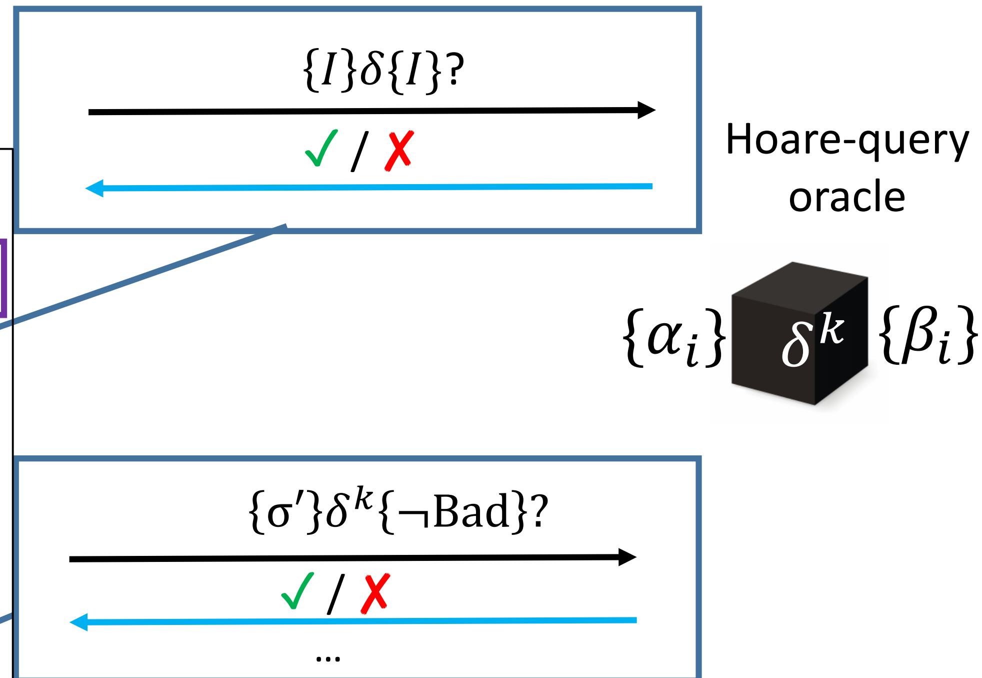
Capable of modeling several interesting algorithms

Hoare-Query Model

ITP-k:

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

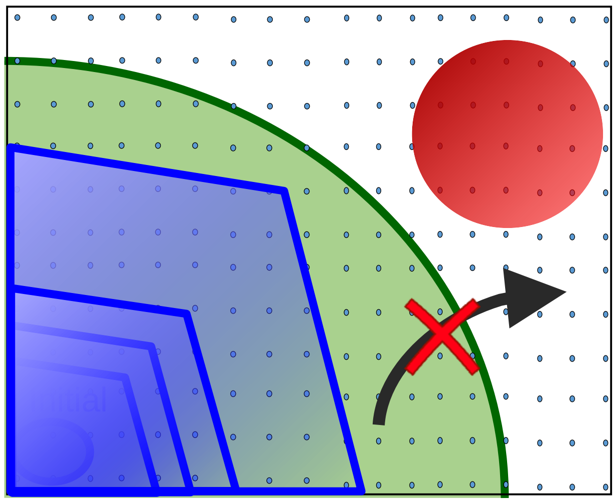
generalize(σ'):
  drop literals from σ'
  while BMC$^k(\sigma', \delta, \text{Bad})$ unsat



Also captures IC3/PDR

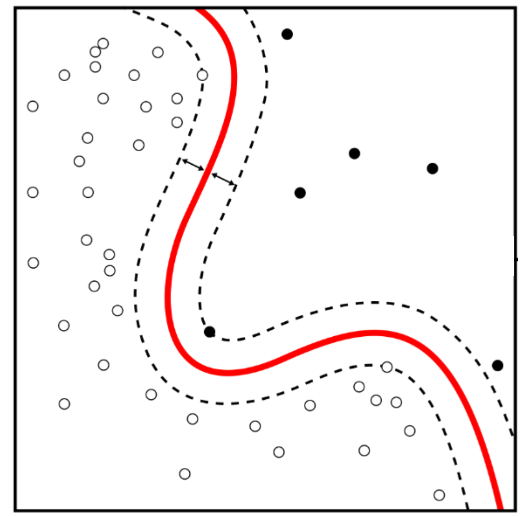
Outline

Invariant Inference



vs.

Exact Concept Learning



- Query-based learning models for invariant inference
- Complexity lower and upper bounds for each model
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from concept learning algorithms

Hoare-Query Complexity

Thm: Every Hoare-query algorithm requires $\mathbf{2^{\Omega(n)}}$ queries in the worst case for inferring $I \in$ DNF s.t. $|I| \leq \text{poly}(n)$

Throughout the talk

n is the vocabulary size, $k = \text{poly}(n)$

- even with unlimited computational power
- unconditional lower bound

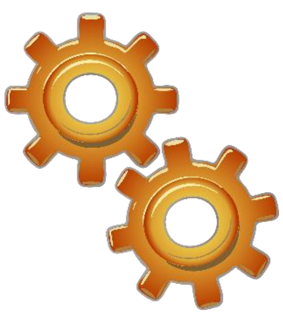
[POPL'20] Complexity and Information in Invariant Inference. Feldman, Immerman, Shoham, Sagiv

Hoare-Query Complexity

<u>Thm</u>: Every Hoare-query algorithm requires $\mathbf{2^{\Omega(n)}}$ queries in the worst case for inferring $I \in$ DNF s.t. $|I| \leq \mathrm{poly}(n)$

<u>Proof sketch:</u>

given
inference algorithm
with $<2^{c \cdot n}$ queries



$\{\alpha_1\}\{\beta_1\}$?

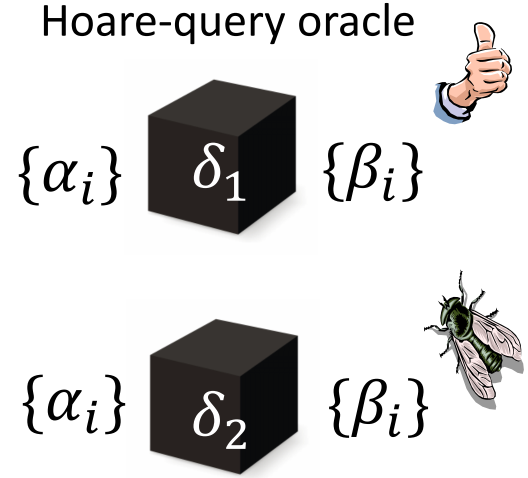
✓ / ✗

...

$\{\alpha_m\}\{\beta_m\}$?

✓ / ✗

construct δ_1, δ_2 for
Hoare-query oracle



1. δ_1 has an inductive invariants with at most n cubes
2. δ_2 does not (in fact, unsafe)
3. all queries return the same answer for δ_1, δ_2

Hoare-Query Complexity

Thm: Every Hoare-query algorithm requires $\mathbf{2^{\Omega(n)}}$ queries in the worst case for inferring $I \in$ DNF s.t. $|I| \leq \mathrm{poly}(n)$

Proof sketch:

given

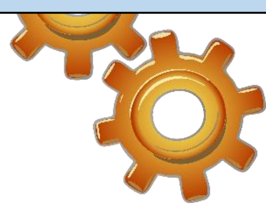
infe

wit

construct δ_1, δ_2 for

take δs checking validity of Boolean quantified formulas
$\exists x_1, \ldots, x_n.\ \forall y_1, \ldots, y_n.\ \phi(x_1, \ldots, x_n, y_1, \ldots, y_n)$

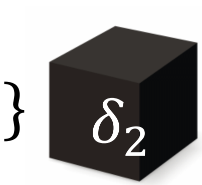
a sub-exponential number of valuations do not determine validity!



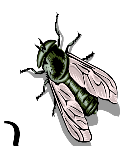
…

$\{\alpha_m\}\{\beta_m\}$?

✓ / ✗

$\{\alpha_i\}$ 

$\{\beta_i\}$



1. δ_1 has an inductive invariants with at most n cubes
2. δ_2 does not (in fact, unsafe)
3. all queries return the same answer for δ_1, δ_2

Hoare-Query Complexity

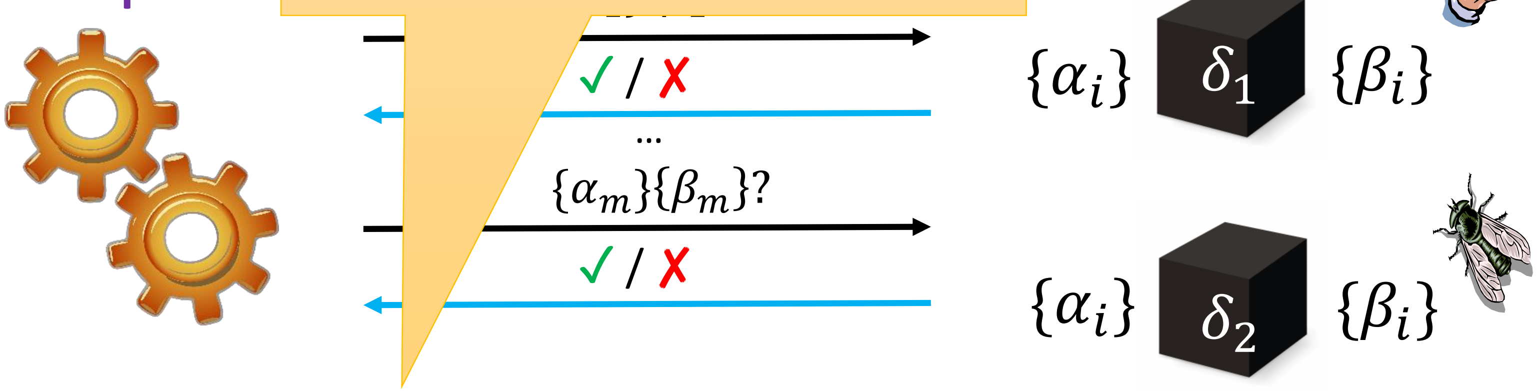
Thm: Every Hoare-query algorithm requires $\mathbf{2^{\Omega(n)}}$ queries in the worst case for inferring $I \in$ DNF s.t. $|I| \leq \mathrm{poly}(n)$

Proof sketch:

given inference algorithm **with <$2^{c \cdot n}$queries**

I is monotone: propositions appear only positively

construct δ_1, δ_2 for Hoare-query oracle



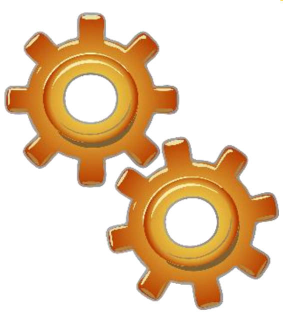
1. δ_1 has an inductive invariants with at most n cubes
2. δ_2 does not (in fact, unsafe)
3. all queries return the same answer for δ_1, δ_2

Hoare-Query Complexity

Thm: Every Hoare-query algorithm requires $2^{\Omega(n)}$ queries in the worst case for inferring $I \in$ DNF s.t. $|I| \leq \mathrm{poly}(n)$

Proof sketch:

given inference algorithm **with <$2^{c \cdot n}$ queries**



I is monotone: propositions appear only positively

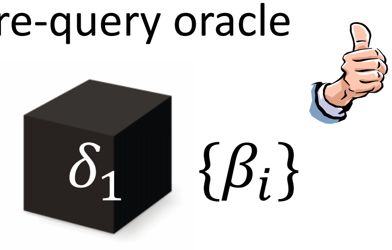
✓ / ✗

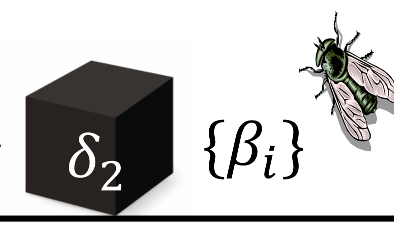
...

$\{\alpha_m\}\{\beta_m\}$?

✓ / ✗

construct δ_1, δ_2 for Hoare-query oracle

$\{\alpha_i\}$  $\{\beta_i\}$

$\{\alpha_i\}$  $\{\beta_i\}$

Cor: Every Hoare-query algorithm requires $2^{\Omega(n)}$ queries in the worst case for inferring short monotone DNF invariants

Hoare > Inductiveness

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

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

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

Hoare > Inductiveness

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

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

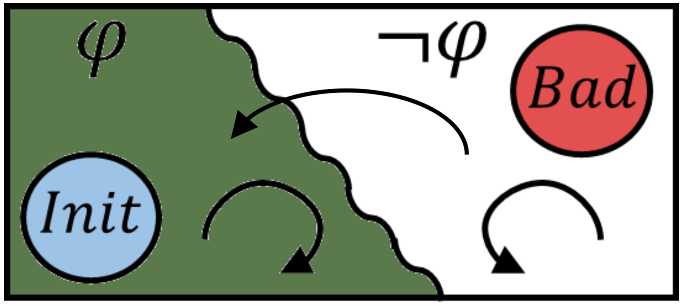
Proof:

$\mathcal{P}$ = maximal transition systems for monotone DNF with n cubes

propositions appear only positively

$$\varphi = x_1 \vee (x_2 \wedge x_3)$$

Maximal system for φ:



Hoare > Inductiveness

Upper bound:

∃ Hoare-query algorithm (with k=1) with $\mathrm{poly}(n)$ queries

Proof: **ITP-1** takes $\boldsymbol{O}(\boldsymbol{n}^{\mathbf{2}})$ queries

I := false

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

  I := I ∨ **generalize**(σ')

generalize(σ'):

  drop literals from σ'

  while BMC[1]$(\sigma', \delta, \mathrm{Bad})$ unsat

minimal

$\sigma' \Rightarrow \varphi$

φ is monotone

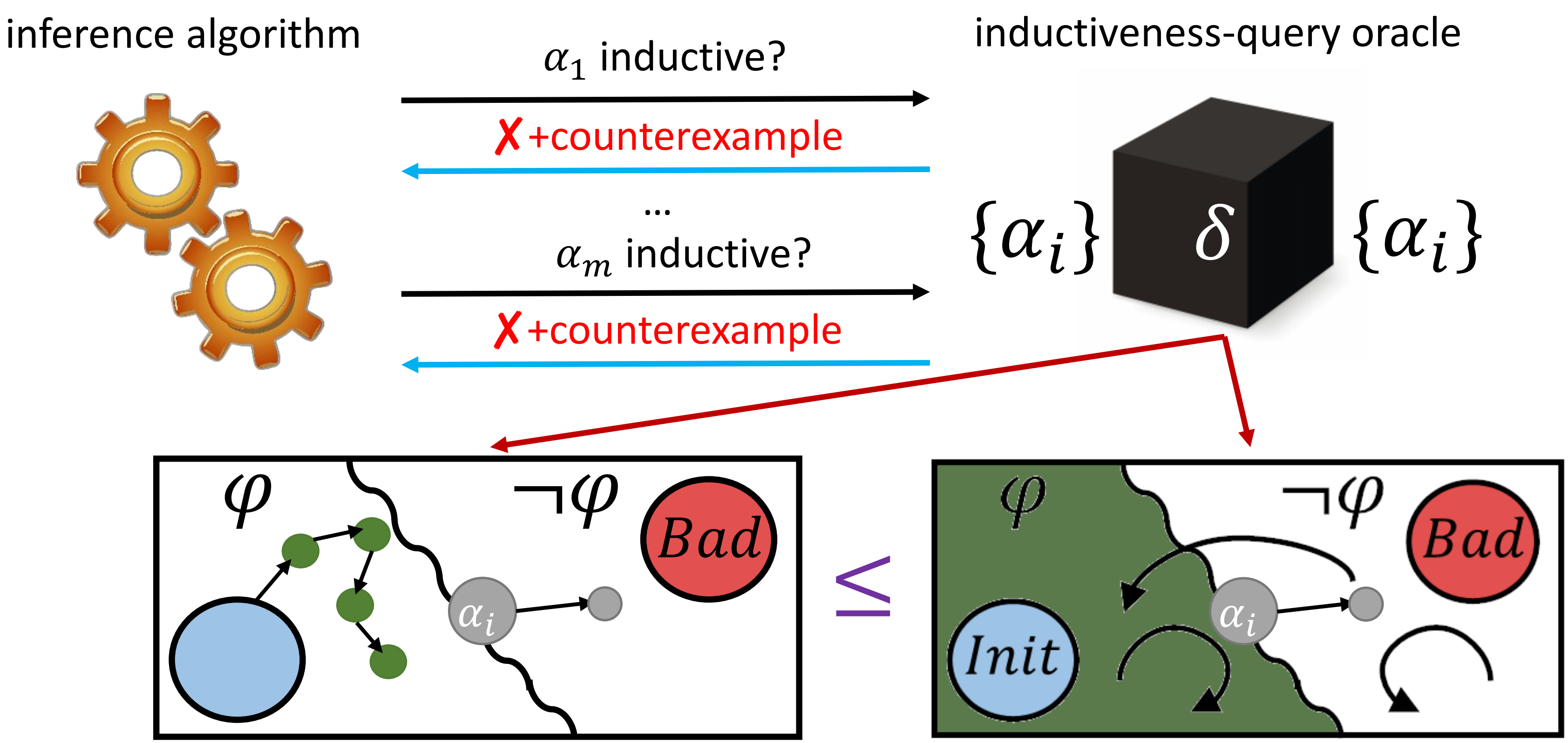
$$\varphi = \underbrace{x_1}_{\text{1 iteration}} \vee \underbrace{(x_2 \wedge x_3)}_{\text{1 iteration}}$$

Hoare > Inductiveness

<u>Lower bound</u>:

∀ inductiveness-query algorithm requires $2^{\Omega(n)}$ queries

<u>Proof:</u>



Hoare > Inductiveness

Lower bound:

∀ inductiveness-query algorithm requires $2^{\Omega(n)}$ queries

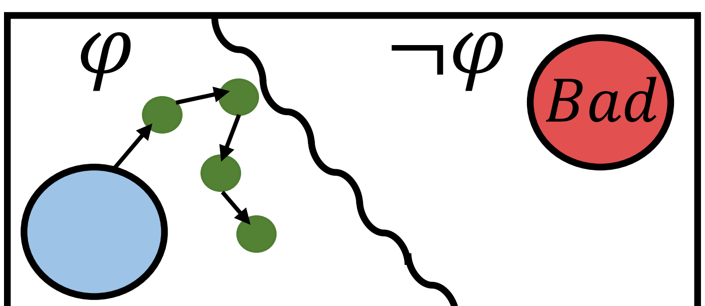
Proof:

previous Corollary

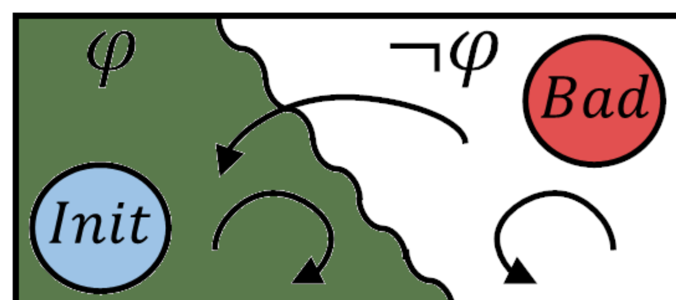
general systems
monotone DNF invariants

maximal systems
monotone DNF invariants

$2^{\Omega(n)} \leq$



$\leq$



Hoare > Inductiveness

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

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

Similar proof works with a simple case of IC3/PDR

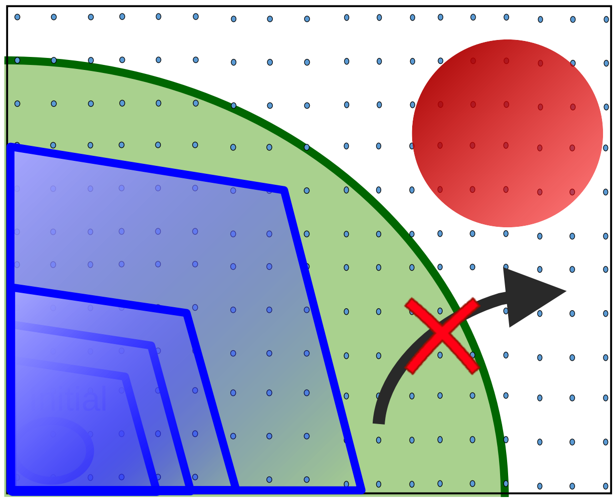
$\Rightarrow$ ICE cannot model PDR,

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.

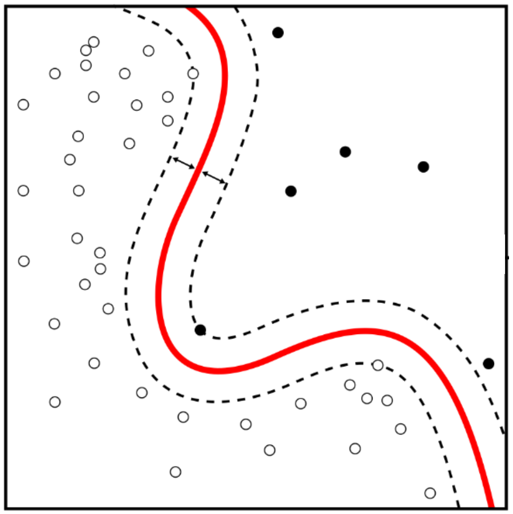
Outline

Invariant Inference



vs.

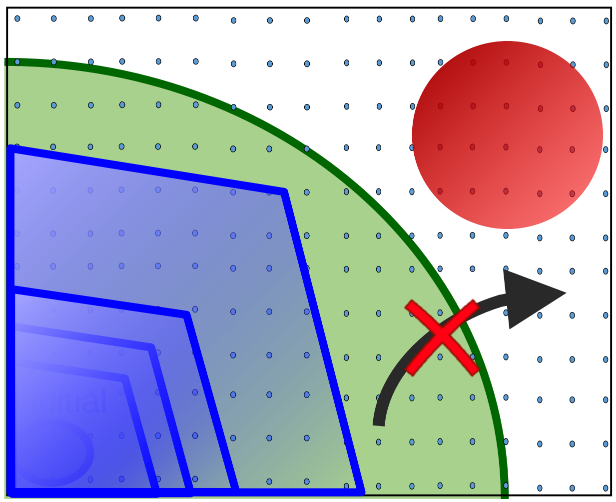
Exact Concept Learning



- Query-based learning models for invariant inference
- Complexity lower and upper bounds for each model
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from concept learning algorithms

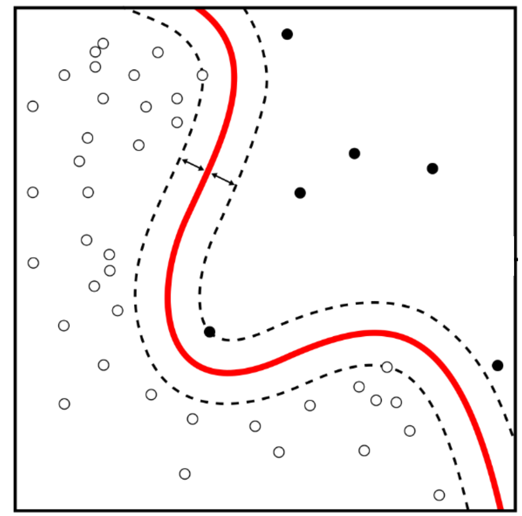
Inferring Monotone DNF

Invariant Inference



vs.

Exact Concept Learning



	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$2^{\Omega(n)}$

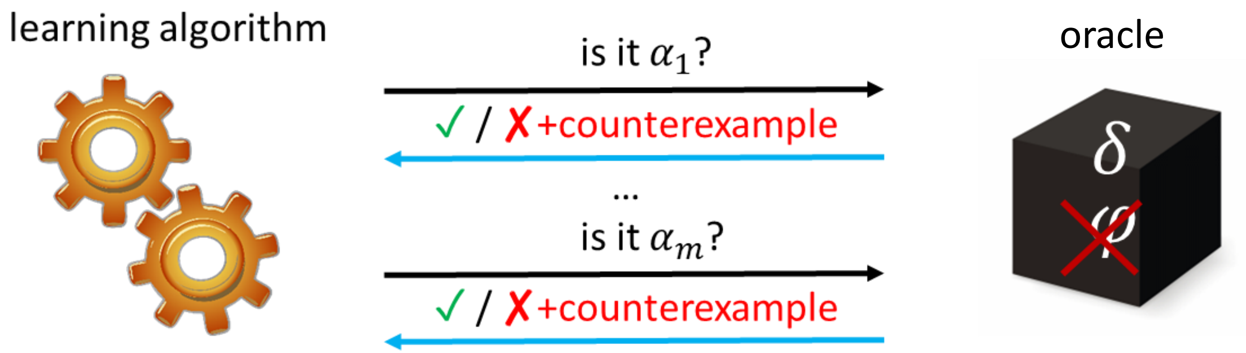
Equiv	sub-exponential
Equiv + mem	poly

[ML'87] Queries and Concept Learning, Angluin
[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.

Inductiveness vs. Equivalence Queries

Invariant Inference

Exact Concept Learning



<u>Counterexamples to induction:</u>

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

	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$2^{\Omega(n)}$

<u>Positive/negative examples:</u>

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

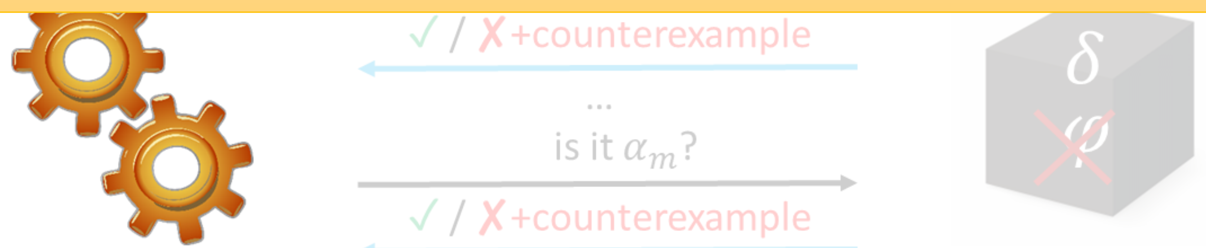
Equiv	sub-exponential
Equiv + mem	poly

[ML'87] Queries and Concept Learning, Angluin

[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.

Inductiveness vs. Equivalence Queries

<u>Thm</u>: Learning from counterexamples to induction is **harder** than learning from positive/negative examples.



<u>Counterexamples to induction:</u>

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

	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$2^{\Omega(n)}$

<u>Positive/negative examples:</u>

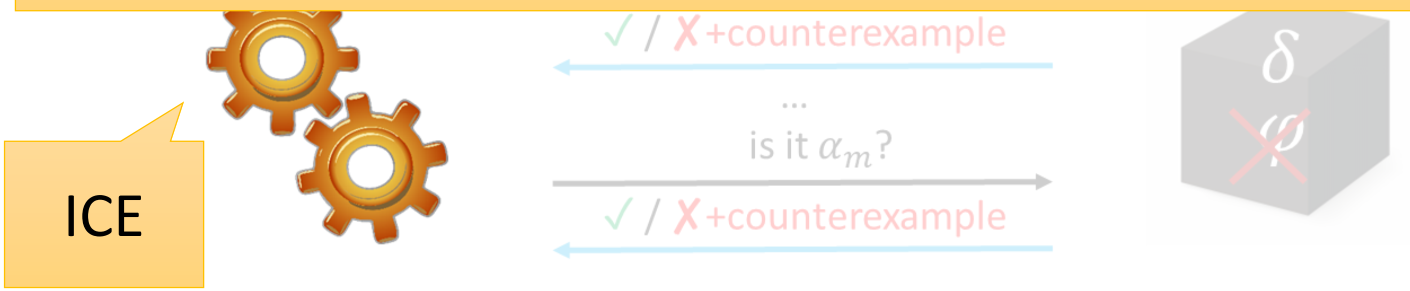
$\sigma^+ \models \varphi\ ,\ \sigma^- \models \neg\varphi$

Equiv	sub-exponential
Equiv + mem	poly

[ML'87] Queries and Concept Learning, Angluin
[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.

Inductiveness vs. Equivalence Queries

<u>Thm</u>: Learning from counterexamples to induction is **harder** than learning from positive/negative examples.



<u>Counterexamples to induction:</u>

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

	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$2^{\Omega(n)}$

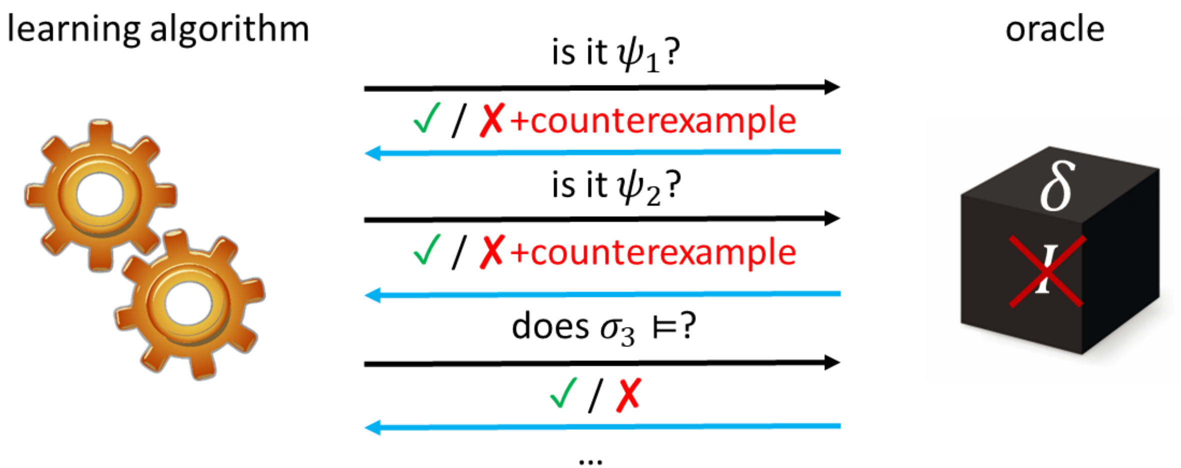
<u>Positive/negative examples:</u>

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

Equiv	sub-exponential
Equiv + mem	poly

[ML'87] Queries and Concept Learning, Angluin
[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.
[CAV'14] ICE: A Robust Framework for Learning Invariants. Garg, Löding, Madhusudan, Neider

Invariant Inference with Equivalence & Membership Queries



Invariant Inference

	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$\boxed{2^{\Omega(n)}}$

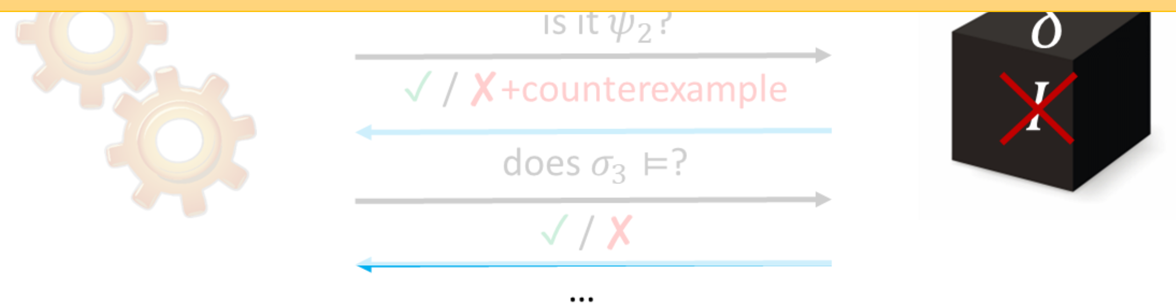
Exact Concept Learning

Equiv	sub-exponential
Equiv + mem	$\boxed{\text{poly}}$

[ML'87] Queries and Concept Learning, Angluin
[COLT'12] Tight Bounds on Proper Equivalence Query Learning of DNF, Hellerstein et al.

Invariant Inference with Equivalence & Membership Queries

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



Invariant Inference

	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$\boxed{2^{\Omega(n)}}$

Exact Concept Learning

Equiv	sub-exponential
Equiv + mem	$\boxed{\text{poly}}$

[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

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

Sufficient conditions for

exact learning algorithms $\Longrightarrow$ invariant inference algorithms

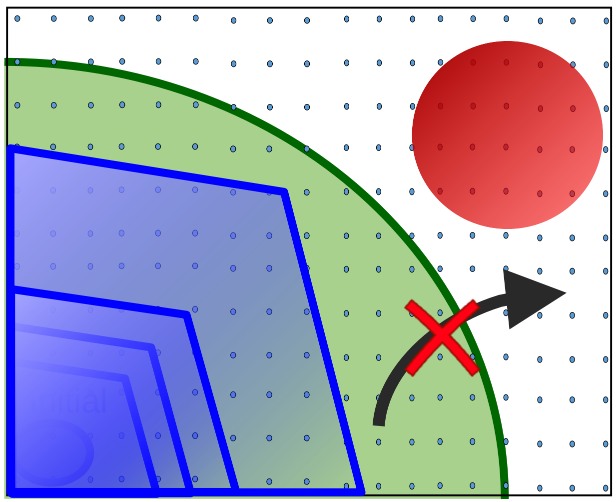
	Maximal	General
Inductive	$2^{\Omega(n)}$	$2^{\Omega(n)}$
Hoare	poly	$\boxed{2^{\Omega(n)}}$

Equiv	sub-exponential
Equiv + mem	$\boxed{\text{poly}}$

[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

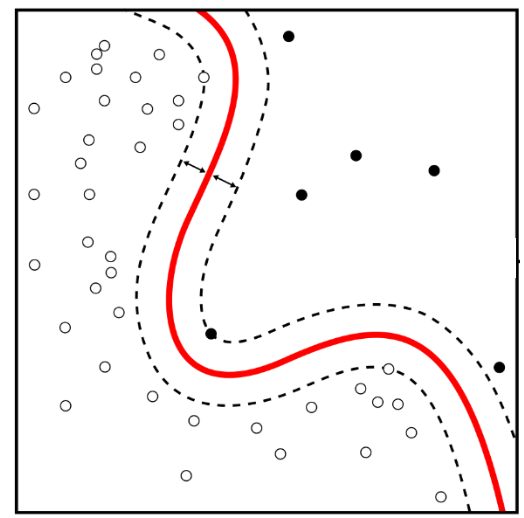
Outline

Invariant Inference



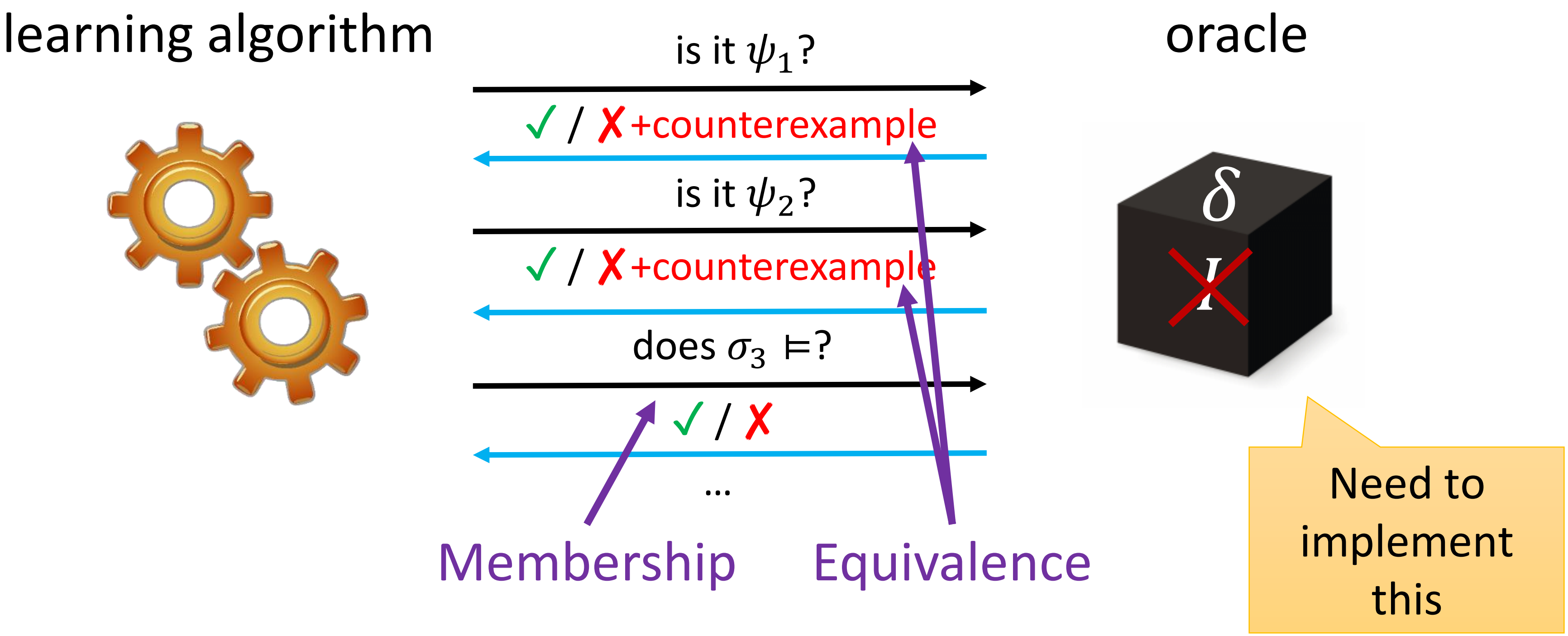
vs.

Exact Concept Learning

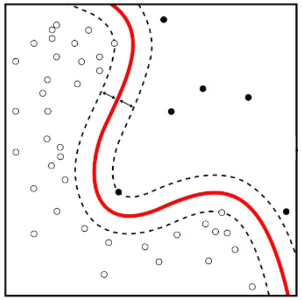


- Query-based learning models for invariant inference
- Complexity lower and upper bounds for each model
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from concept learning algorithms

From Learning to Inference



From Learning to Inference



Exact **learning**
DNF formulas

ψ := false

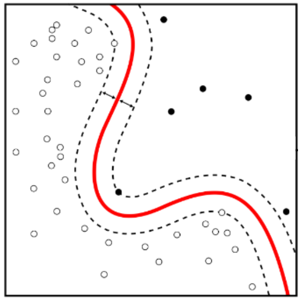
while σ' counterexample
    to Equivalence(ψ):

  ψ := $\psi \lor \mathbf{generalize}(\sigma')$

$\mathbf{generalize}(\sigma')$:
  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

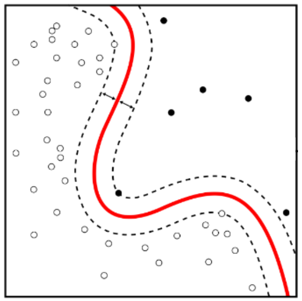
while σ' counterexample
to **Equivalence**(ψ): → **Inductive**(I)

ψ := $\psi \lor$ **generalize**(σ')

generalize(σ'):
drop literals from σ'
while **Membership**(σ')= ✓ → $\mathrm{BMC}^k(\sigma', \delta, \mathbf{Bad})$ **unsat**

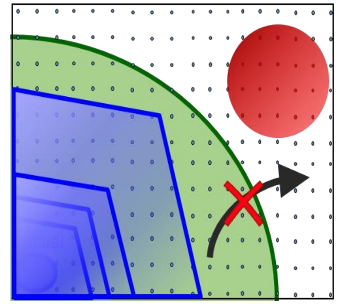
[CACM’84] A Theory of the Learnable. Valiant
[ML’87] Queries and Concept Learning. Angluin
[ML’95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

From Learning to Inference



Exact **learning**
DNF formulas

$\Longrightarrow$



Inferring
DNF invariants

ψ := false

while σ' counterexample
 to **Equivalence**(ψ):
  ψ := $\psi \vee$ **generalize**(σ')

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

I := false

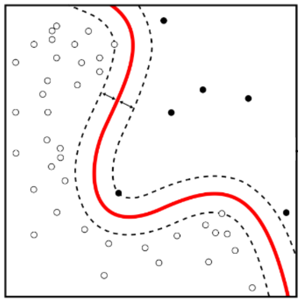
while $(\_,\sigma')$ counterexample
 to **Inductive**(I):
  I := $I \vee$ **generalize**(σ')

generalize(σ'):
  drop literals from σ'
  while $\mathbf{BMC}^k(\sigma', \delta, \mathbf{Bad})$ **unsat**

[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

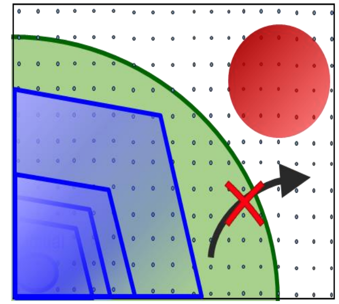


Efficiently

Exact **learning**

DNF formulas

$\Longrightarrow$



Efficiently

Inferring

DNF invariants

When is the transformation correct?

ψ := false

while σ' counte... to **Equiva**...

ψ := $\psi \vee$ **generalize**(σ')

generalize(σ'):

drop literals from σ'

while **Membership**(σ')= ✓

I := false

...') counterexample to **Inductive**(I):

I := $I \vee$ **generalize**(σ')

generalize(σ'):

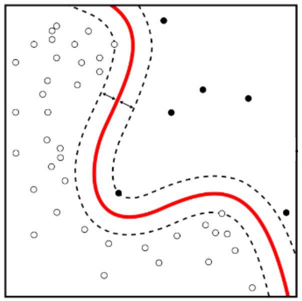
drop literals from σ'

while **BMC**^{k}(σ', δ, **Bad**) **unsat**

[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

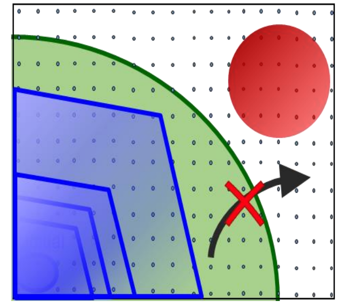


Efficiently

Exact **learning**

DNF formulas

$\Longrightarrow$



Efficiently

Inferring

DNF invariants

Thm: can implement queries when
the invariant is k-fenced
and the algorithm's queries are one-sided

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

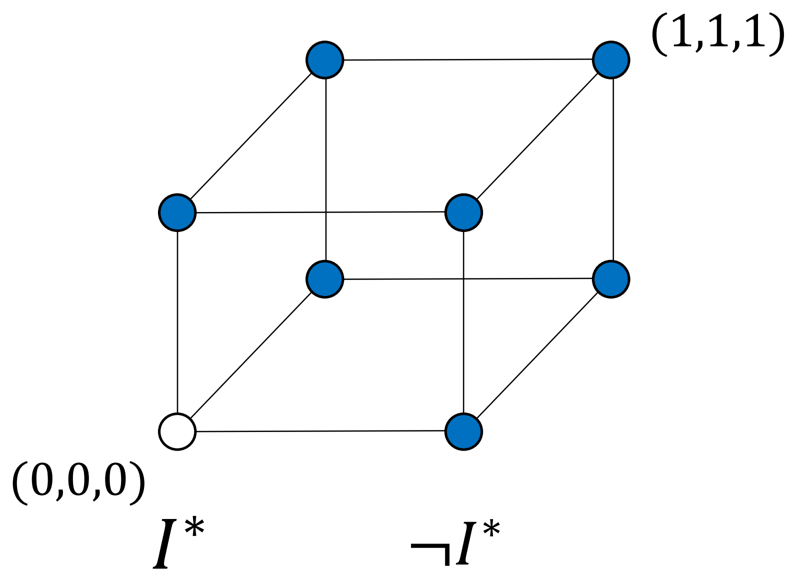
$\longrightarrow$

generalize(σ'):
  drop literals from σ'
  while $\mathbf{BMC}^{\mathbf{k}}(\boldsymbol{\sigma'}, \boldsymbol{\delta}, \mathbf{Bad})$ **unsat**

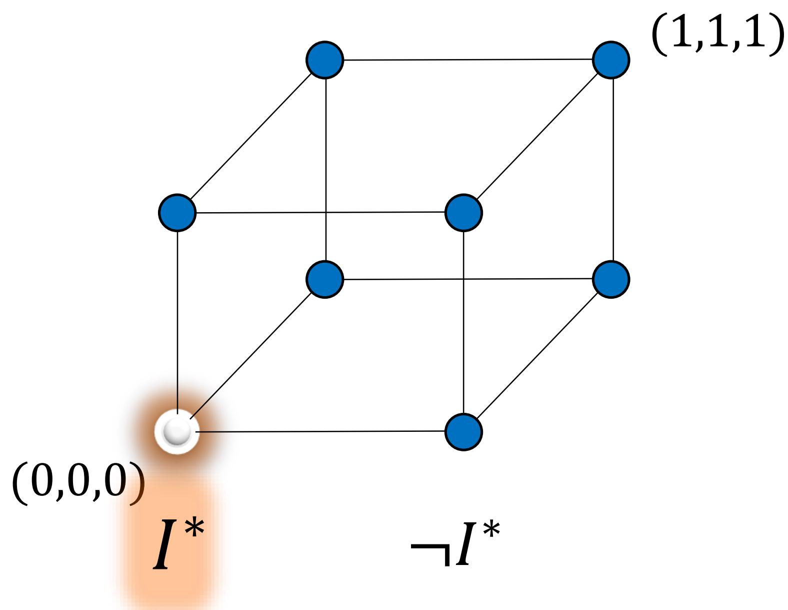
[CACM'84] A Theory of the Learnable. Valiant
[ML'87] Queries and Concept Learning. Angluin
[ML'95] On the Learnability of Disjunctive Normal Form Formulas. Aizenstein and Pitt

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

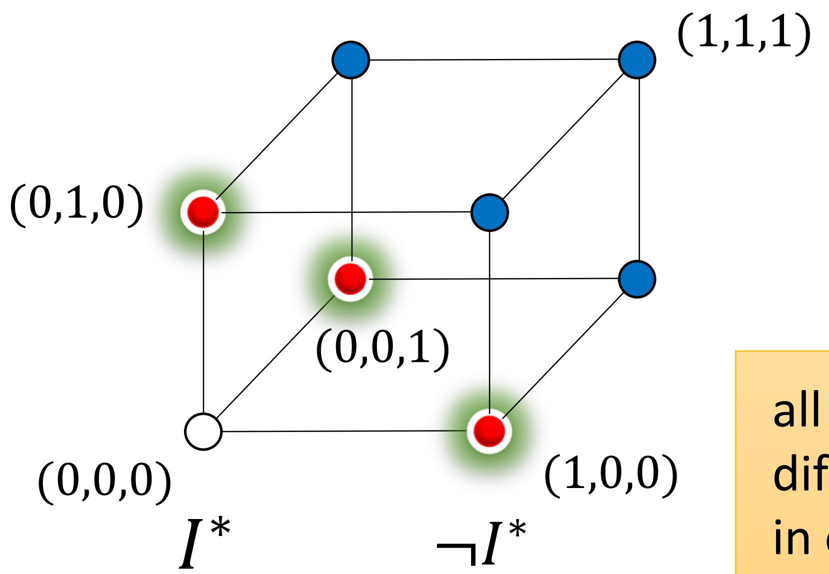
k-Fenced Invariants



k-Fenced Invariants



k-Fenced Invariants

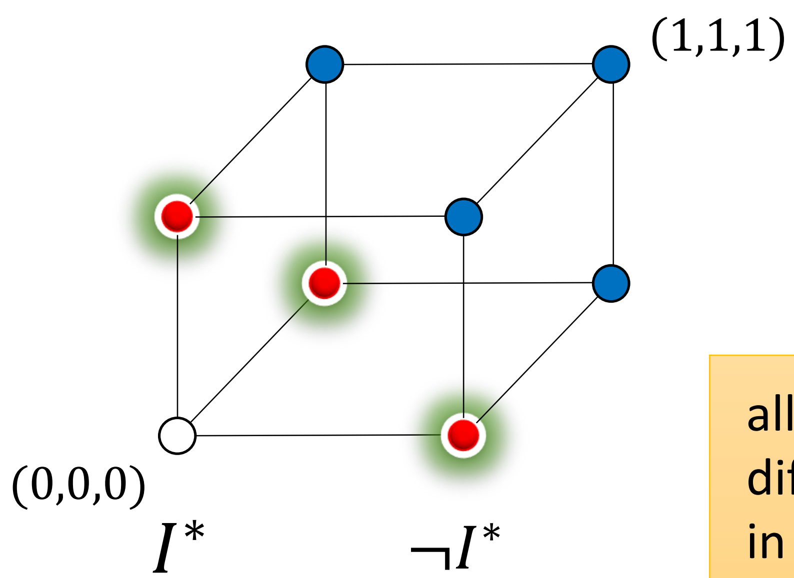


all states $\sigma \notin I^*$ that differ from some $\sigma' \in I^*$ in one bit

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

Outer boundary

k-Fenced Invariants



all states $\sigma \notin I^*$ that differ from some $\sigma' \in I^*$ in one bit

I^* is k-fenced if

all the states in $\partial^-(I^*)$

can reach a bad state in at most k steps

Example: k-Fenced Invariant

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

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

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

$$I^*: \; x_n \neq 1$$

all the states in $\partial^-(I^*) = \{x_n = 1\}$
can reach a bad state in at most k steps $= 1$

Example: k-Fenced Invariant

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

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

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

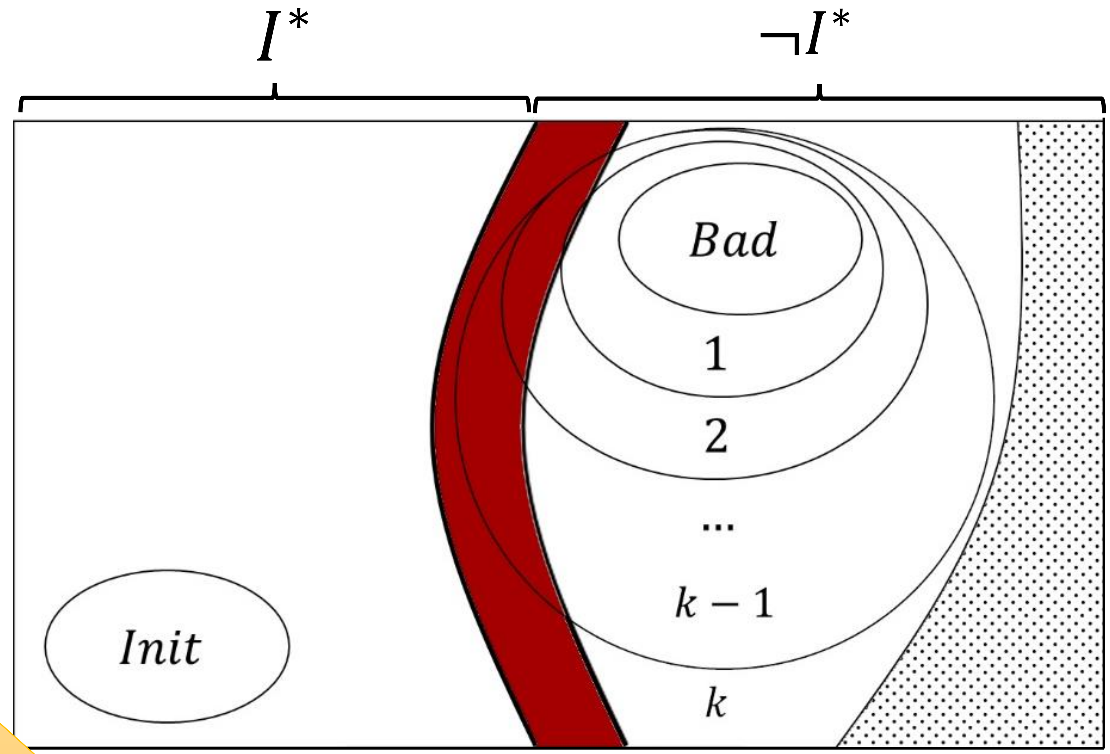
In general **not** all states in $\neg I^*$ need to reach bad

$I^*\!: \;\; x_n \neq 1$

In this example $\neg I^*$

all the states in $\partial^-(I^*) \;=\; \{x_n = 1\}$
can reach a bad state in at most k steps $= 1$

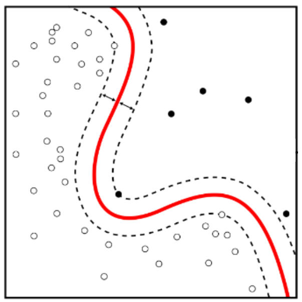
k-Fenced Invariants



In general **not** all states in $\neg I^*$ need to reach bad

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

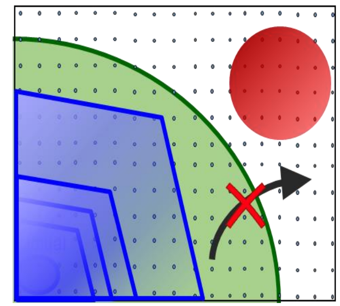
From Learning to Inference



Efficiently

Exact **learning**
DNF formulas

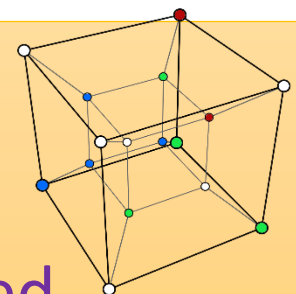
$\Longrightarrow$



Efficiently

Inferring
DNF invariants

Thm: can implement queries when
the invariant is k-fenced ✓
and the algorithm's queries are <u>one-sided</u>



`One-Sided Equivalence`(ψ): $\psi \Longrightarrow \varphi$

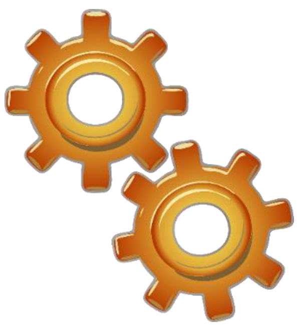
`One-Sided Membership`(σ): $\sigma \in \varphi \cup \partial^{-}(\varphi)$

One-Sided Equivalence Queries to Invariants

inference algorithm

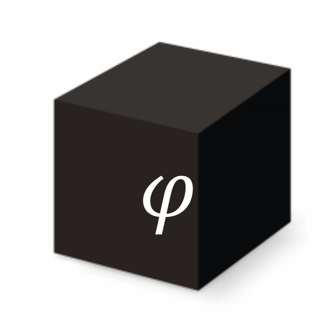
$\psi \Longrightarrow \varphi$

teacher



is it ψ?

✓ / ✗ +counterexample



is ψ an inductive invariant?

✓ yes hooray!

✗ +counterexample transition:

(σ, σ') s.t. $\sigma \vDash \psi, \sigma' \vDash \neg\psi$

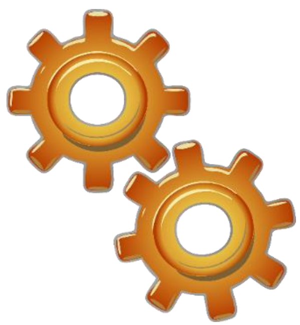
Always return σ' as positive example

One-Sided Membership Queries to k-Fenced Invariants

inference algorithm

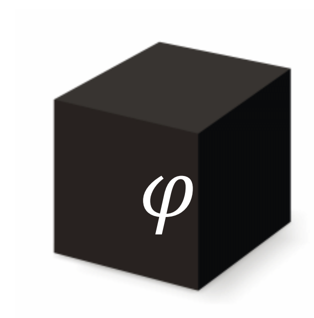
$\boldsymbol{\sigma \in \varphi \cup \partial^{-}(\varphi)}$

teacher



is $\sigma_3 \vDash$?

✓ / ✗



can't σ_3 reach bad states **in k steps?**

BMC^{k}$(\sigma_3, \delta, \mathrm{Bad})$ **unsat?**

✓ then yes

✗ then no

Doesn't always imply that $\sigma_3 \vDash I^*$

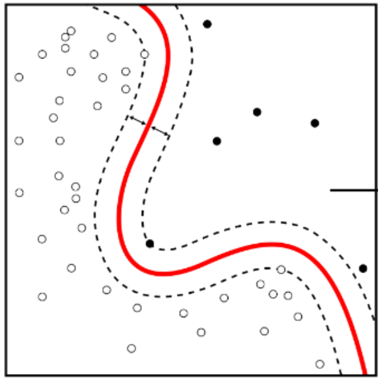
From Learning to Inference

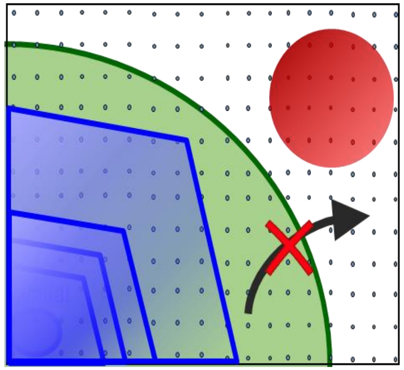
<u>Thm</u>: Let $\mathcal{C}$ be a class of formulas.

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many one-sided queries

$\Longrightarrow$

$\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced





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

Efficient Inference

<u>Thm 1</u>: $\mathcal{C}$ = monotone DNF

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many one-sided queries $\implies$ $\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced

[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

Efficient Inference

Thm 1: $\mathcal{C}$ = monotone DNF

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many one-sided queries $\Longrightarrow$ $\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced

$\psi \Longrightarrow \varphi$

```
ψ := false
while σ′ counterexample
      to Equivalence(ψ):
   ψ := ψ ∨ generalize(σ′)
```

$\sigma' \in \varphi \cup \partial^-(\varphi)$

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

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

generalize(σ'):
drop literals from σ'
while **BMC^k**(σ', δ, Bad) unsat

Efficient Inference

Thm 1: $\mathcal{C}$ = monotone DNF

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many one-sided queries $\implies$ $\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced

Thm 1: The **interpolation-based algorithm** converges in a polynomial number of SAT queries if I^* is

- k-fenced, and
- has a short monotone DNF representation

[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

Efficient Inference

Thm 2: $\mathcal{C}$ = almost-monotone DNF

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many one-sided queries $\Longrightarrow$ $\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced

Thm 2: A **different algorithm** converges in a polynomial number of SAT queries if If I^* is

- k-fenced, and
- has a short almost-monotone DNF representation

at most $O(1)$ terms include negated variables

[Inf. Comput. '95] Exact Learning Boolean Function via the Monotone Theory. Bshouty

Inference from Unrestricted Queries

Thm': Let $\mathcal{C}$ be a class of formulas.

two-sided

$\exists\mathcal{A}$ **identifying** $\varphi \in \mathcal{C}$ with polynomially-many ~~one-sided~~ queries $\Longrightarrow$ $\exists\mathcal{A}$ **inferring** $I^* \in \mathcal{C}$ with polynomially-many SAT queries whenever I^* is k-fenced

Thm 3: A **different algorithm** converges in a polynomial number of SAT queries if I^* is

- two-sided k-fenced, and
- has a short DNF *and* a short CNF representation

e.g., I^* is expressible as a short decision tree

[Inf. Comput. '95] Exact Learning Boolean Function via the Monotone Theory. Bshouty

Inference from Unrestricted Queries

Thm’: Let $\mathcal{C}$ be a class of formulas.

two-sided

$\exists \mathcal{A}$ **inferring** $^* \in \mathcal{C}$ with
ia y-many
que ies
* is k-fenced

Thm: also when I^* is one-sided k-fenced
but **not by transformation from learning**

Thm 3: A **different algorithm** converges in a polynomial number of SAT queries if I^* is

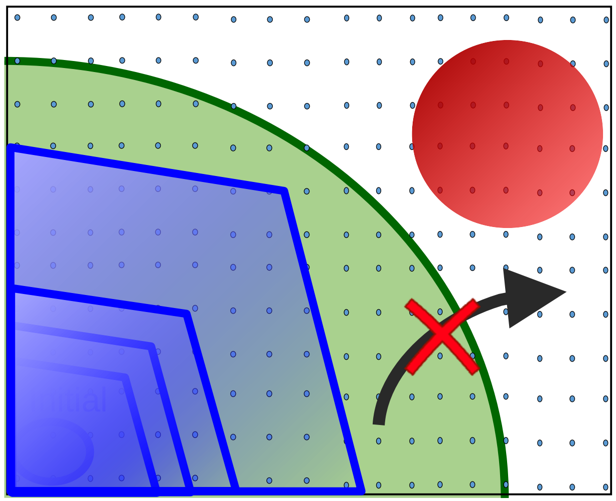
- two-sided k-fenced, and
- has a short DNF *and* a short CNF representation

e.g., I^* is expressible as a short decision tree

[Inf. Comput. ‘95] Exact Learning Boolean Function via the Monotone Theory. Bshouty
[SAS ‘22] Invariant Inference With Provable Complexity From the Monotone Theory. Feldman, Shoham

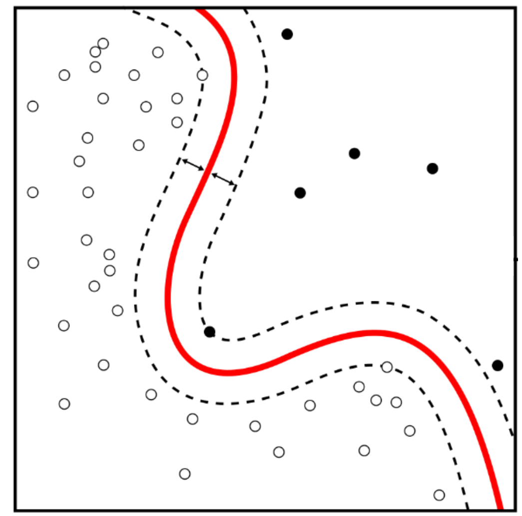
Conclusion (1)

Invariant Inference



vs.

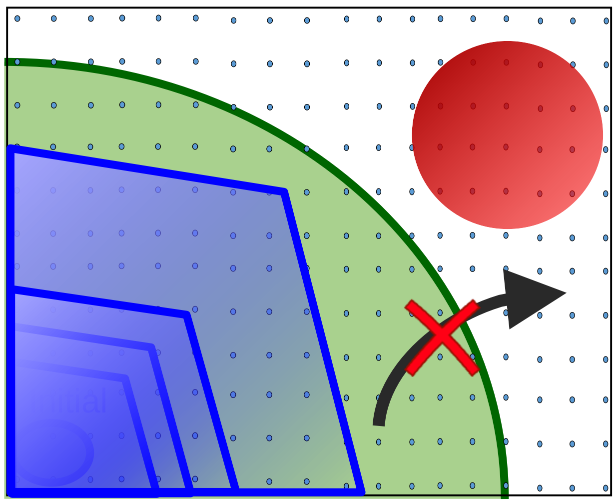
Exact Concept Learning



- Query-based learning models for invariant inference
- Complexity lower and upper bounds for each model
- Invariant inference is harder than concept learning
- Complexity results for invariant inference algorithms from concept learning algorithms

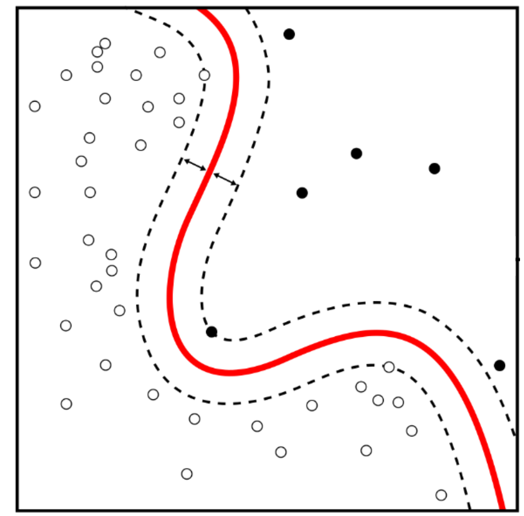
Conclusion (2)

Invariant Inference



vs.

Exact Concept Learning



- What about IC3/PDR?
- Impact of k in the Hoare query model?
- Is the fence condition necessary?
- Other conditions?
- Beyond Boolean programs