Challenges for Fast Synthesis Procedures in SMT

Andrew Reynolds

ARCADE Workshop

August 6, 2017

Synthesis

- SMT solvers act as *subroutines* for automated synthesis
  - For program snippets, planning, digital circuits, programming by examples, …
- More recently, SMT solvers act as *stand-alone tools* for synthesis
  - Leveraging their support for first-order quantification

[Reynolds et al CAV2015]

Synthesis Conjectures

$$\exists f.\forall x.P(f,x)$$

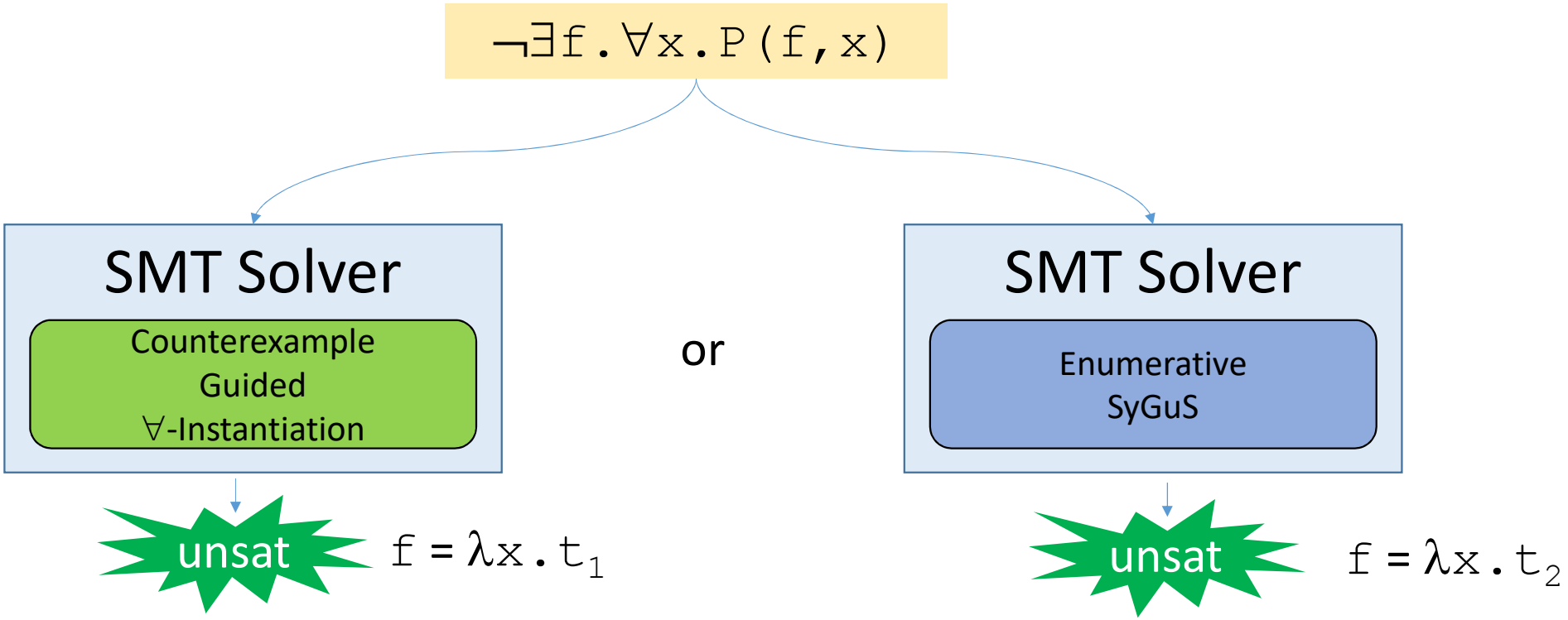
There exists a function `f` for which property `P` holds for all `x`

Refutation-Based Synthesis in SMT

$\neg\exists f.\forall x.P(f,x)$

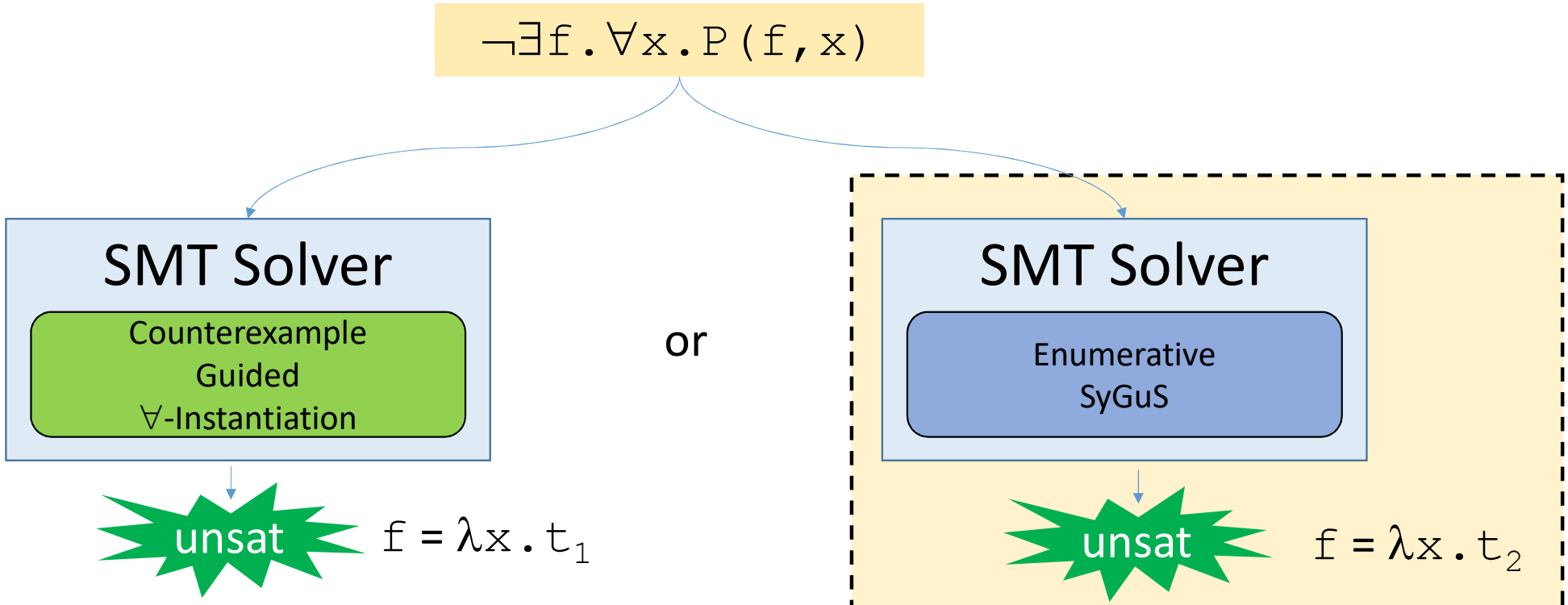
(**negated** synthesis conjecture)

Refutation-Based Synthesis in SMT



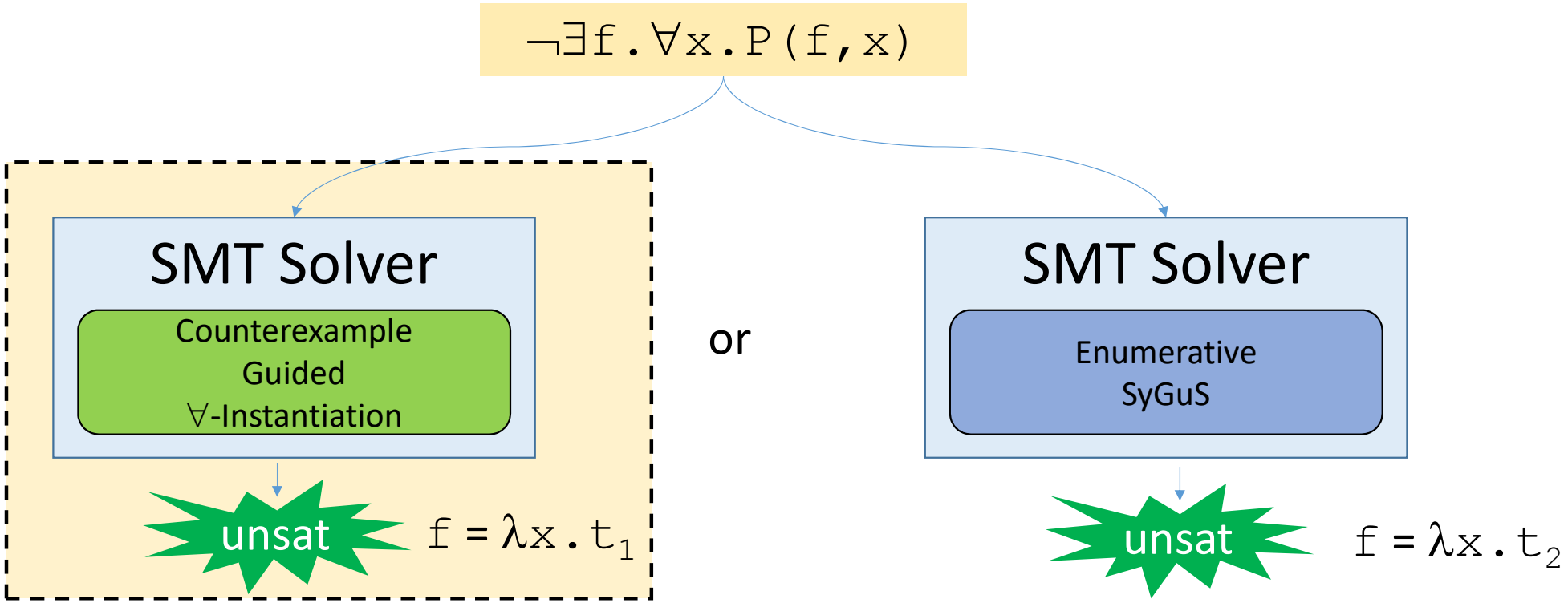
- Two approaches for refutation-based synthesis in SMT solvers **[Reynolds et al CAV2015]**

Refutation-Based Synthesis in SMT



⇒ Based on enumerative search (via syntax-guided synthesis) **[Alur et al 2013]**

Refutation-Based Synthesis in SMT



⇒ Based on first-order quantifier instantiation **(focus of this talk)**

Single Invocation Conjectures

- Some synthesis conjectures are *essentially first-order*:

$$\neg\exists f.\forall xy.\ \mathbf{f(x,y)} \geq x \wedge \mathbf{f(x,y)} \geq y \wedge (\mathbf{f(x,y)} = x \vee \mathbf{f(x,y)} = y)$$

"**f(x,y)** is the maximum of x and y"

Single Invocation Conjectures

$\neg\exists f.\forall xy.\ \mathbf{f(x,y)} \geq x \wedge \mathbf{f(x,y)} \geq y \wedge (\mathbf{f(x,y)} = x \vee \mathbf{f(x,y)} = y)$

$\mathtt{Int} \times \mathtt{Int} \rightarrow \mathtt{Int}$

All occurrence of f are in terms of the form **f(x,y)**

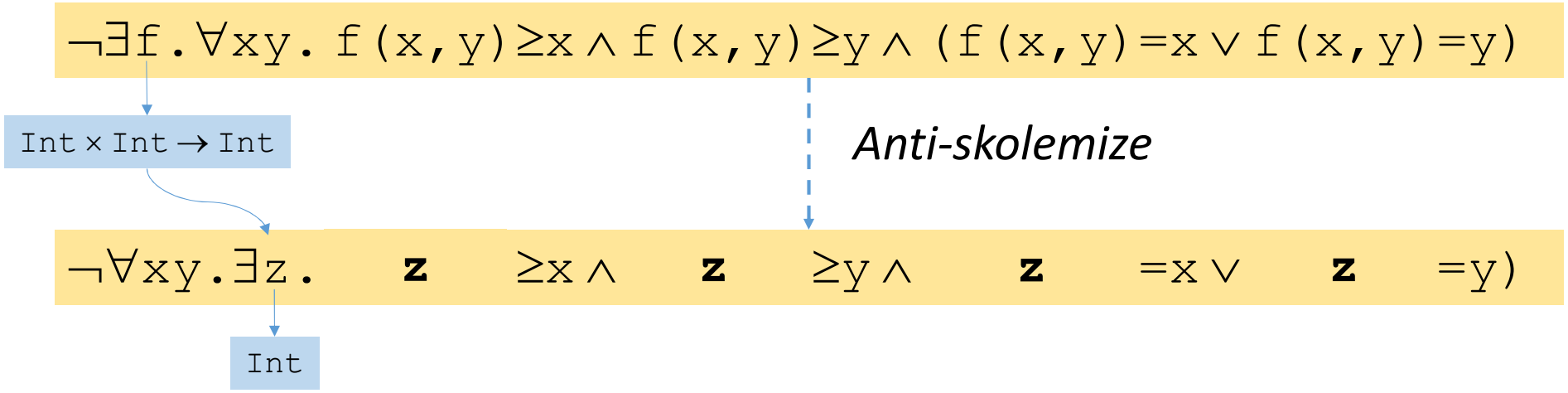
$\Rightarrow$ "single invocation" synthesis conjectures

Single Invocation Conjectures

$$\neg\exists f.\forall xy.\ f(x,y)\geq x \wedge f(x,y)\geq y \wedge (f(x,y)=x \vee f(x,y)=y)$$

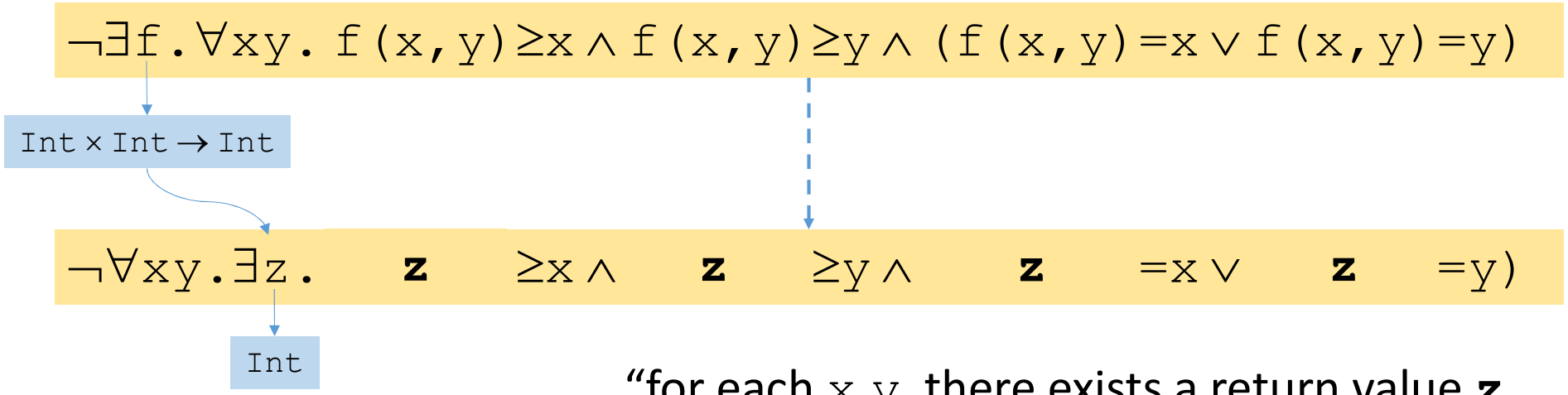
$$\texttt{Int} \times \texttt{Int} \rightarrow \texttt{Int}$$

Single Invocation Conjectures



[Reynolds et al CAV2015]

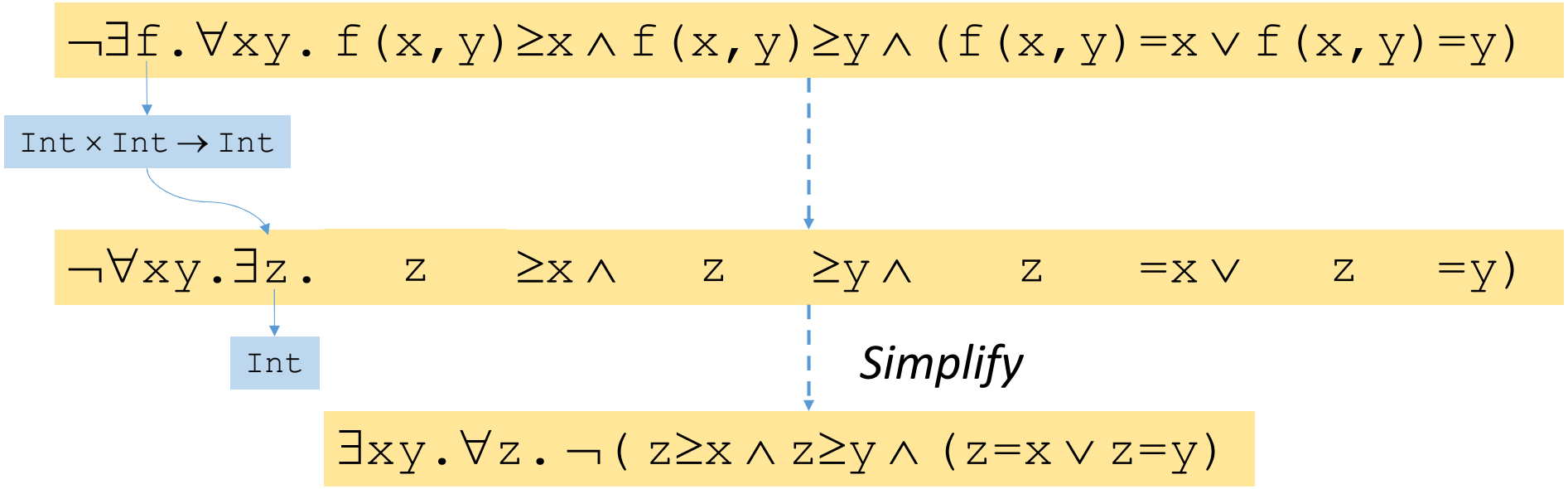
Single Invocation Conjectures



"for each x,y, there exists a return value **z** that is the maximum of x and y"

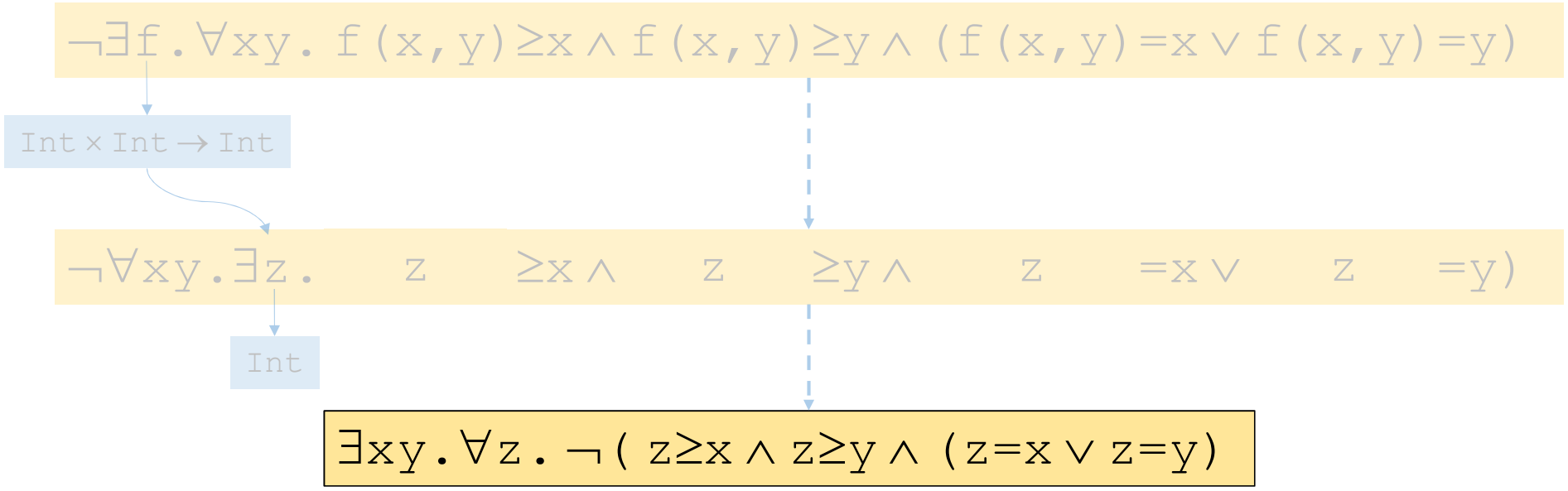
[Reynolds et al CAV2015]

Single Invocation Conjectures



[Reynolds et al CAV2015]

Single Invocation Conjectures



First-order linear arithmetic $\Rightarrow$ *Solvable by first-order* $\forall$*-instantiation*

[Reynolds et al CAV2015]

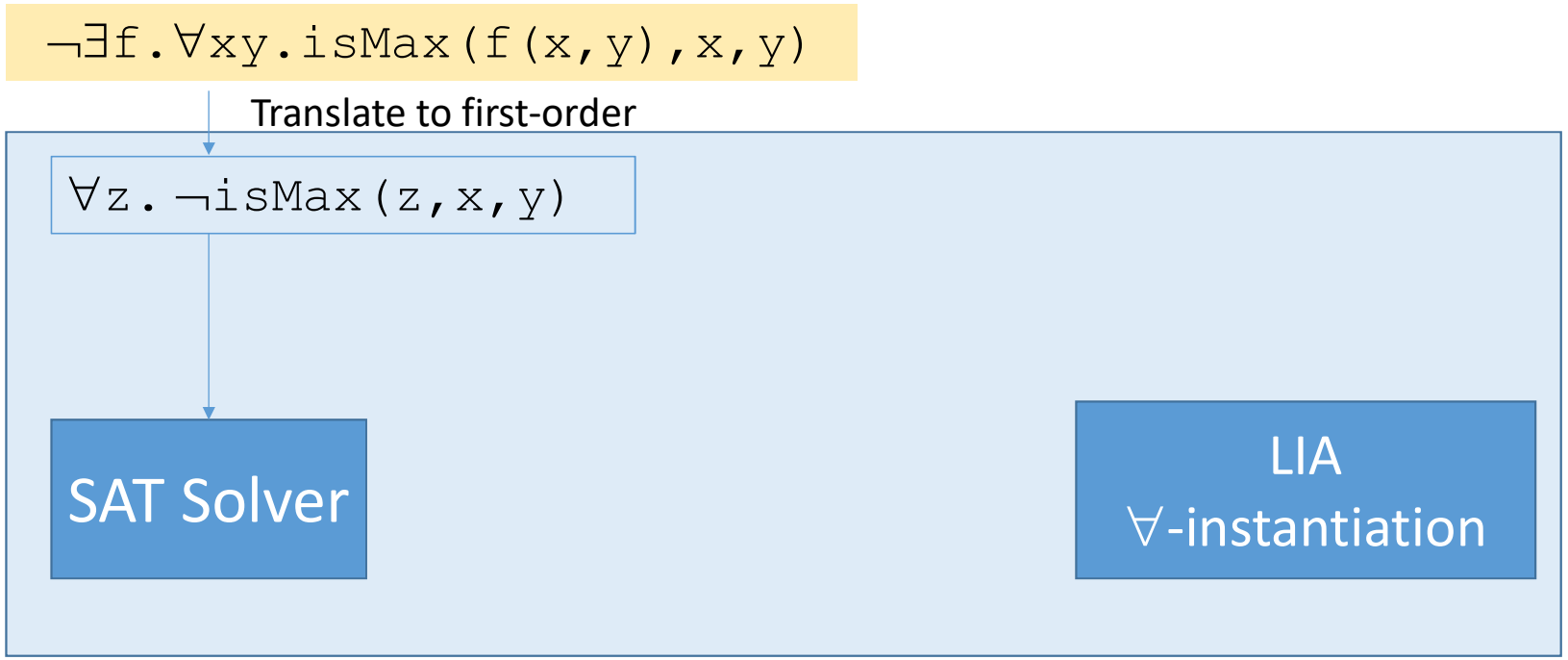
Single Invocation Synthesis in SMT

$\neg\exists f.\forall xy.\ f(x,y)\geq x \wedge f(x,y)\geq y \wedge (f(x,y)=x \vee f(x,y)=y)$

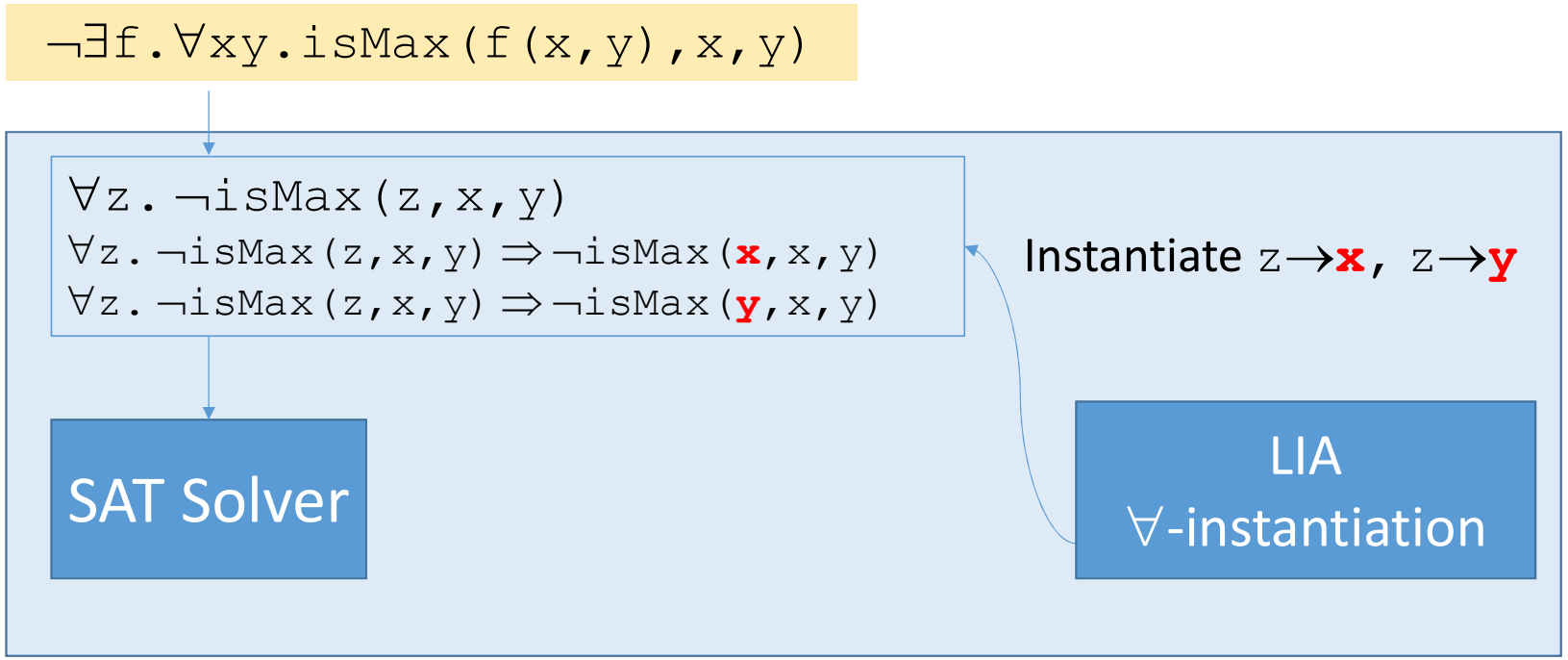
Single Invocation Synthesis in SMT

$\neg\exists f.\forall xy.\mathtt{isMax}(f(x,y),x,y)$

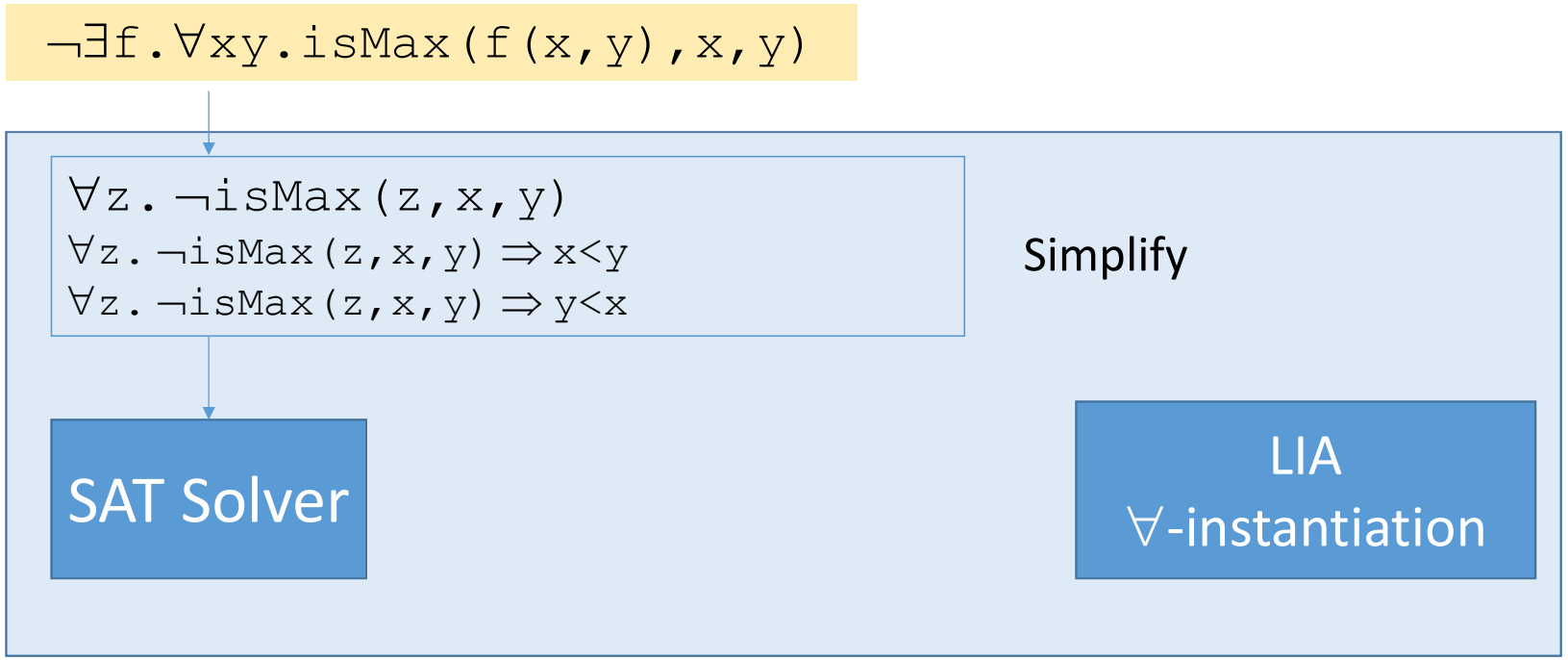
Single Invocation Synthesis in SMT



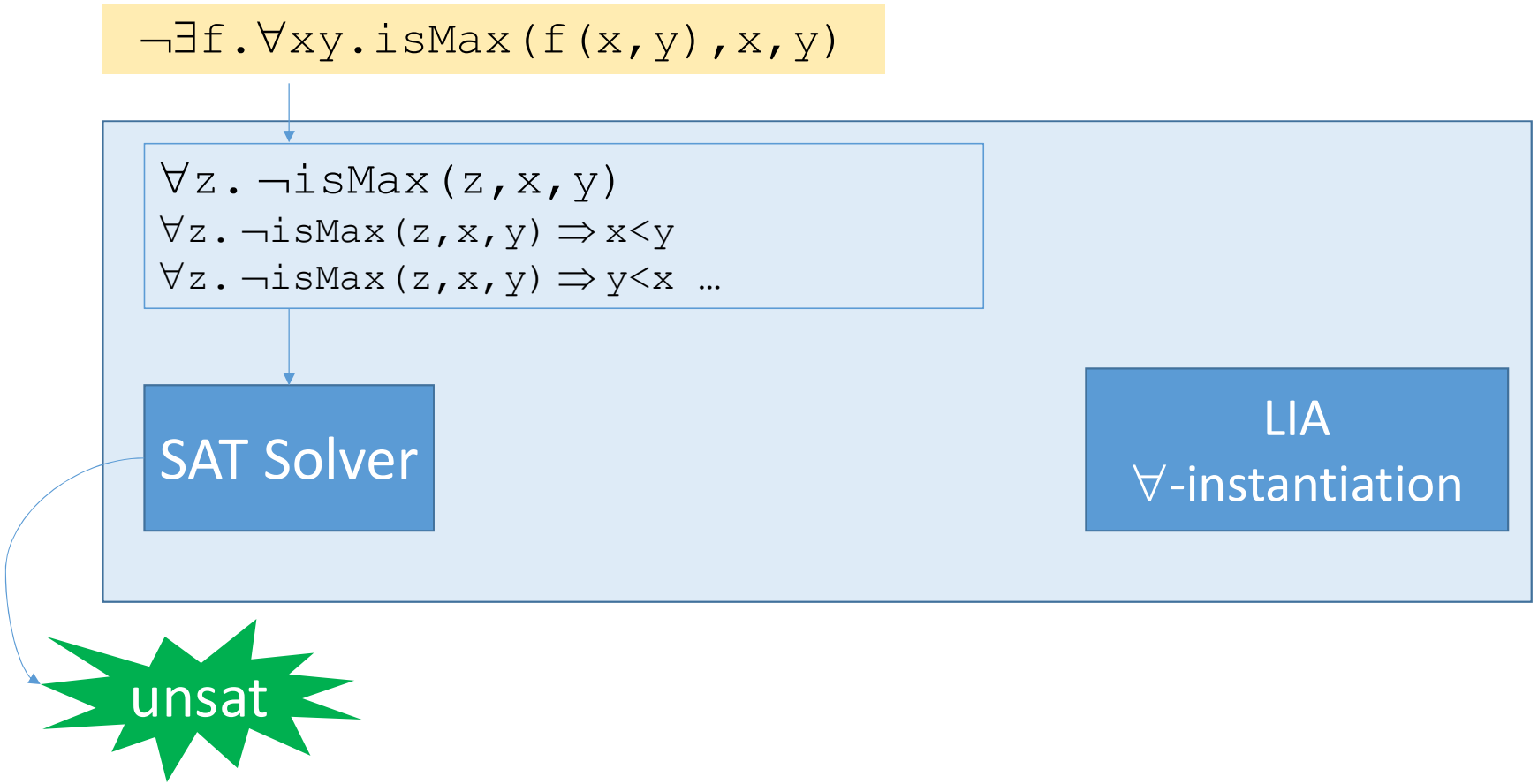
Single Invocation Synthesis in SMT



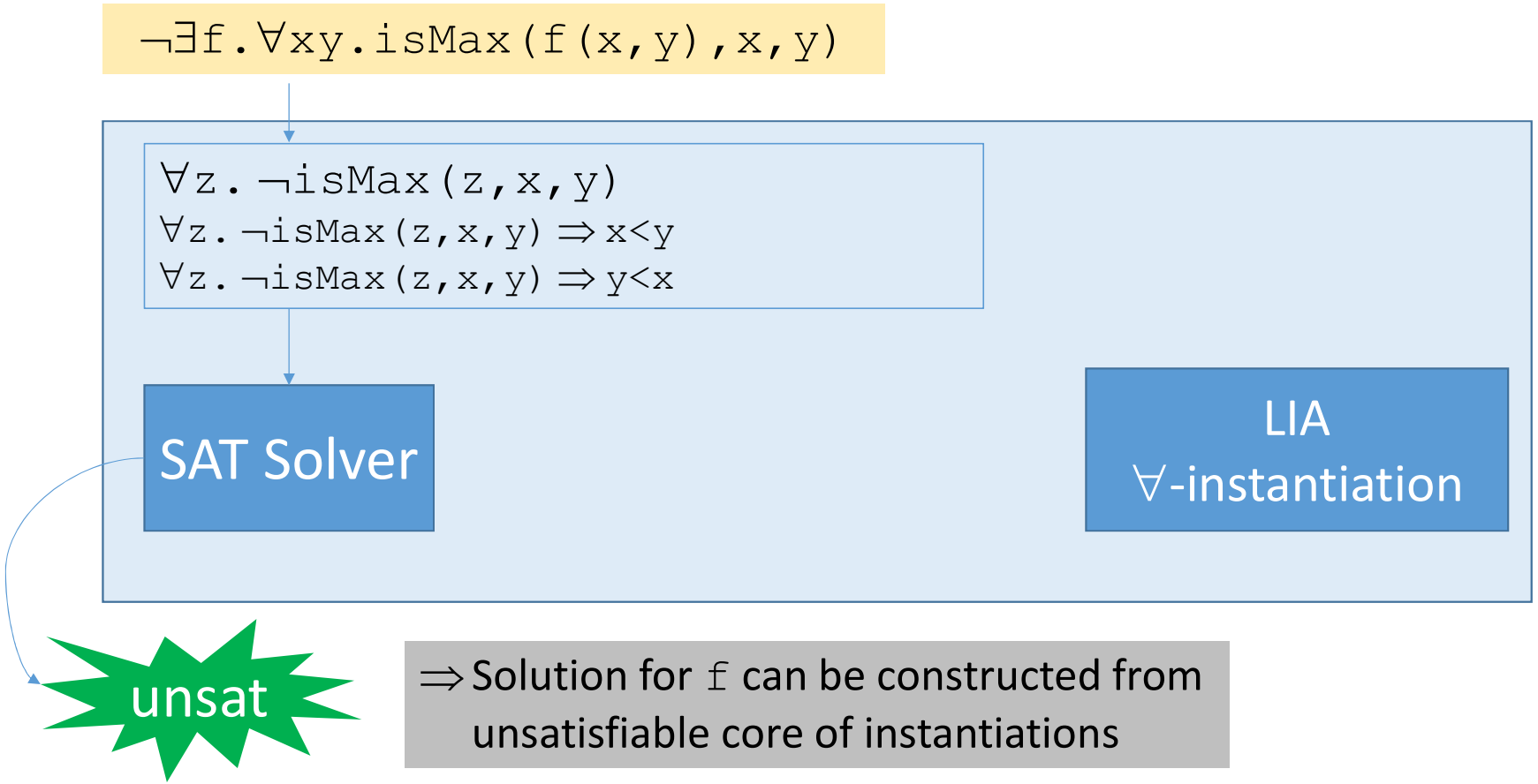
Single Invocation Synthesis in SMT



Single Invocation Synthesis in SMT

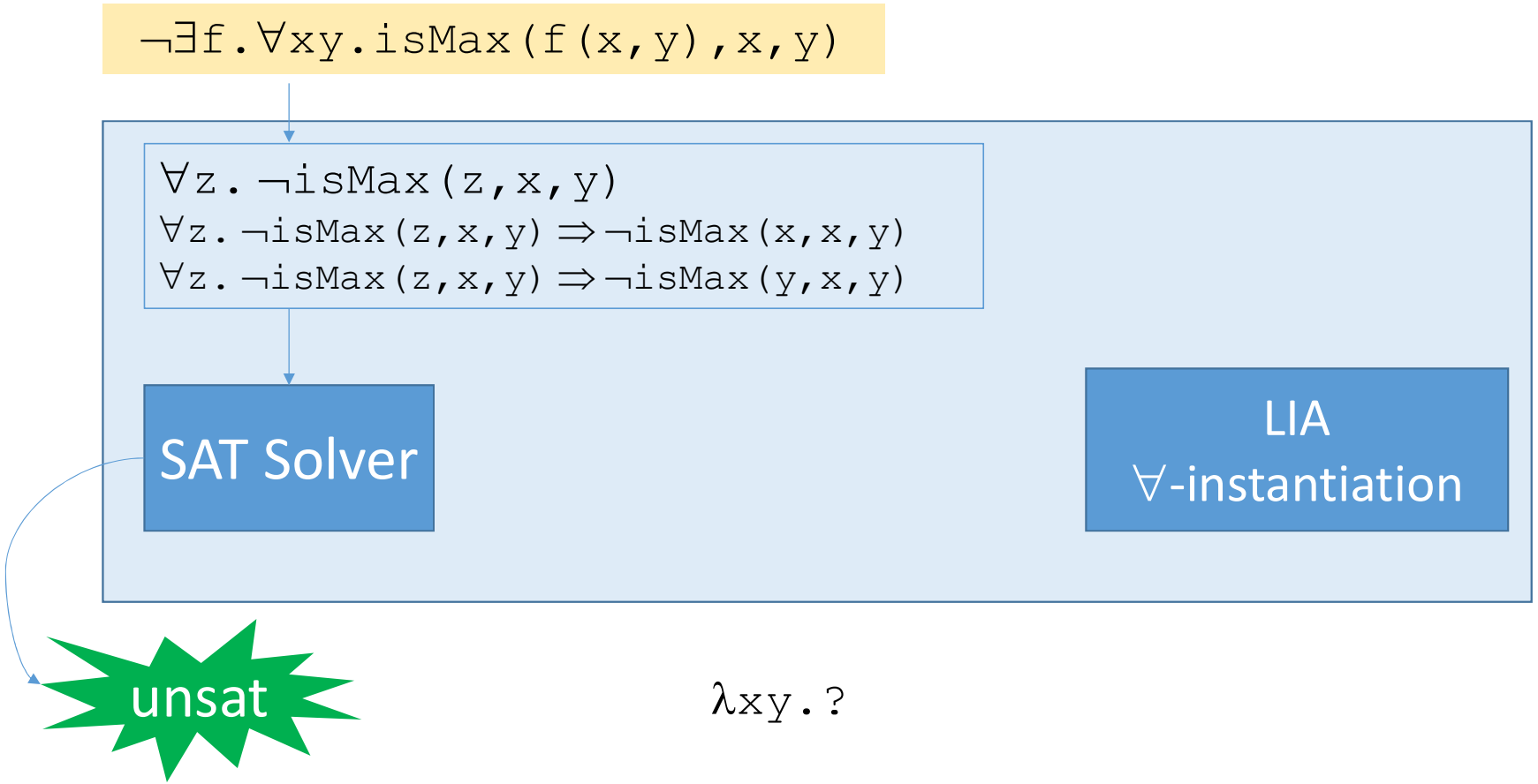


Single Invocation Synthesis in SMT

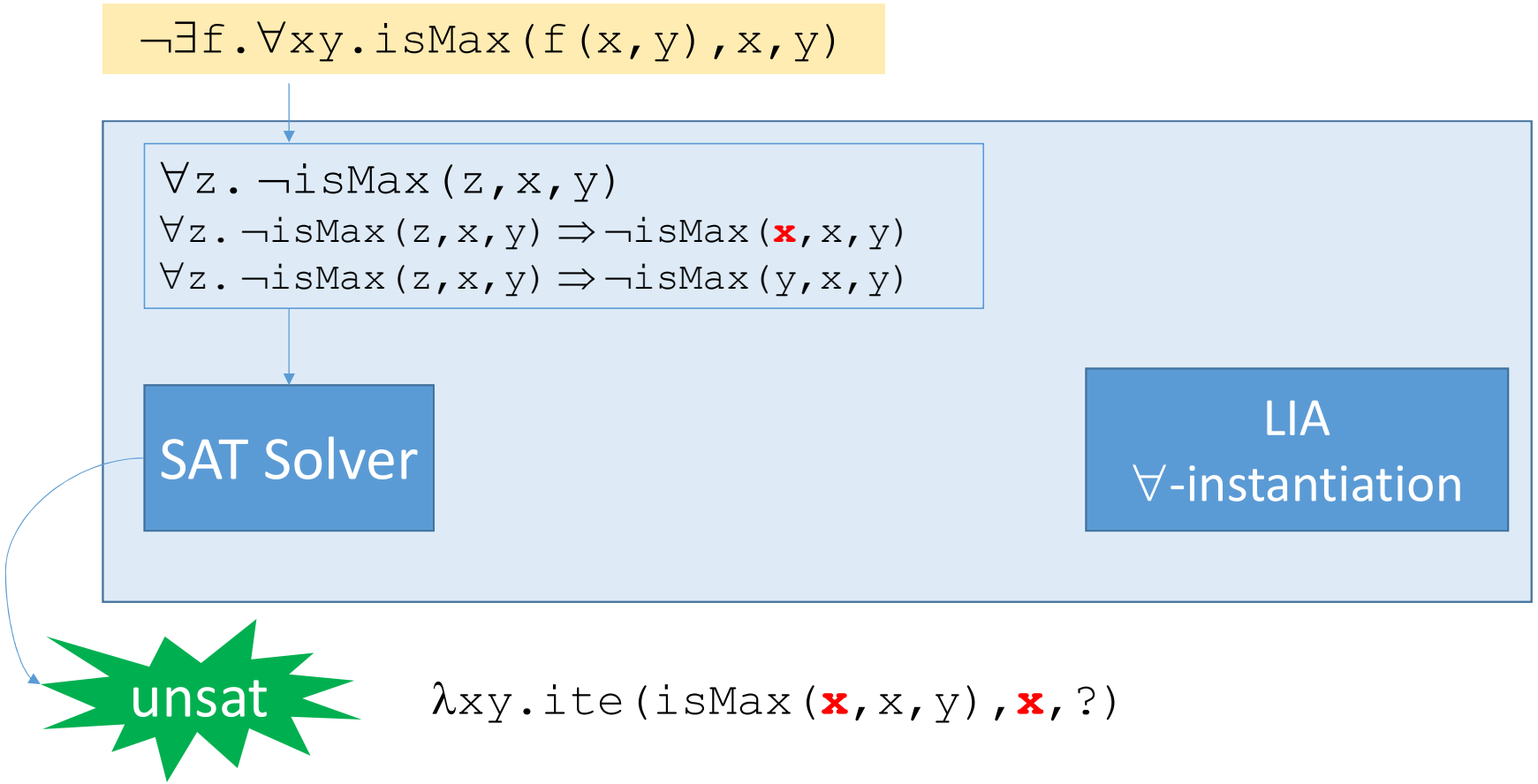


⇒ Solution for f can be constructed from unsatisfiable core of instantiations

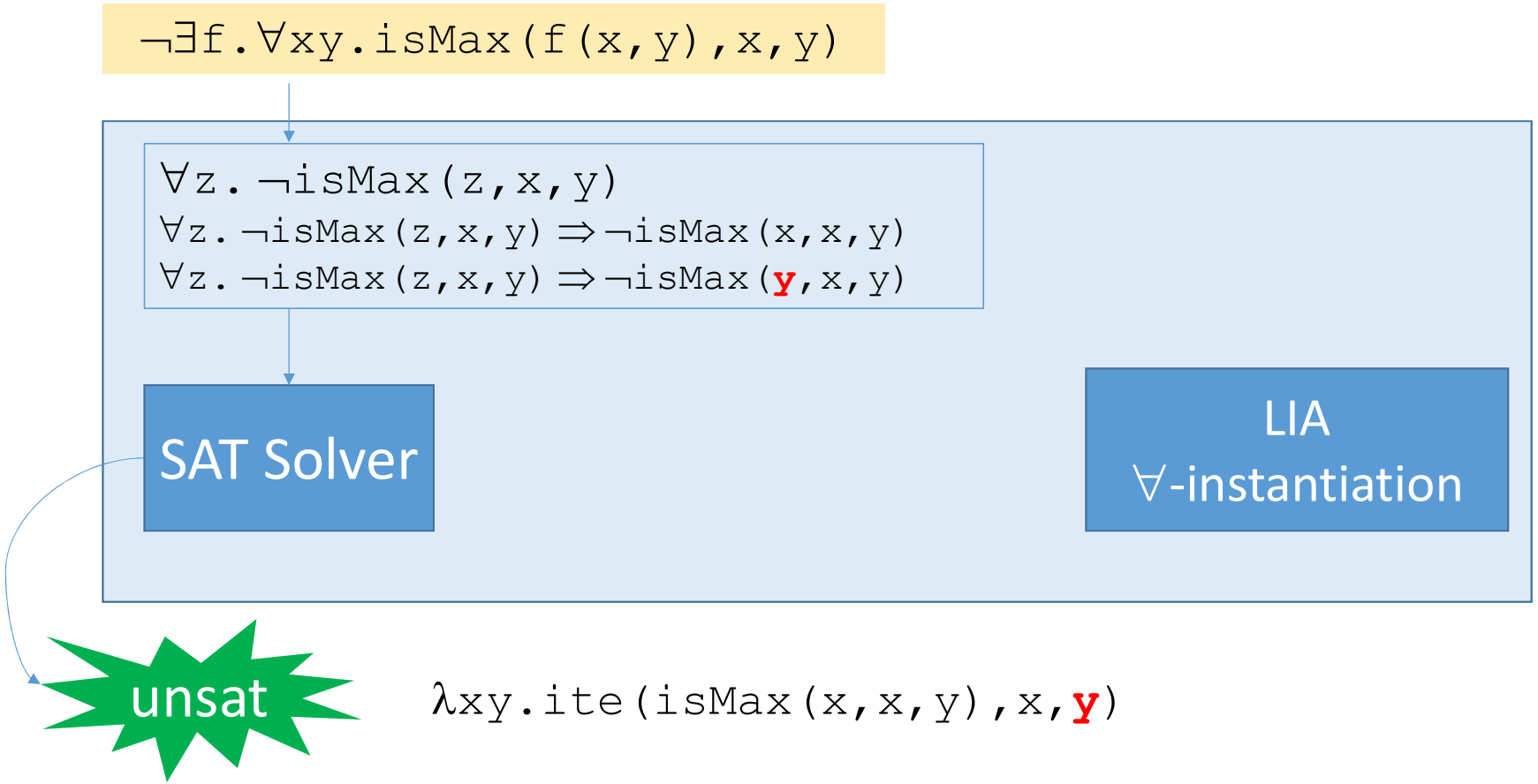
Single Invocation Synthesis in SMT



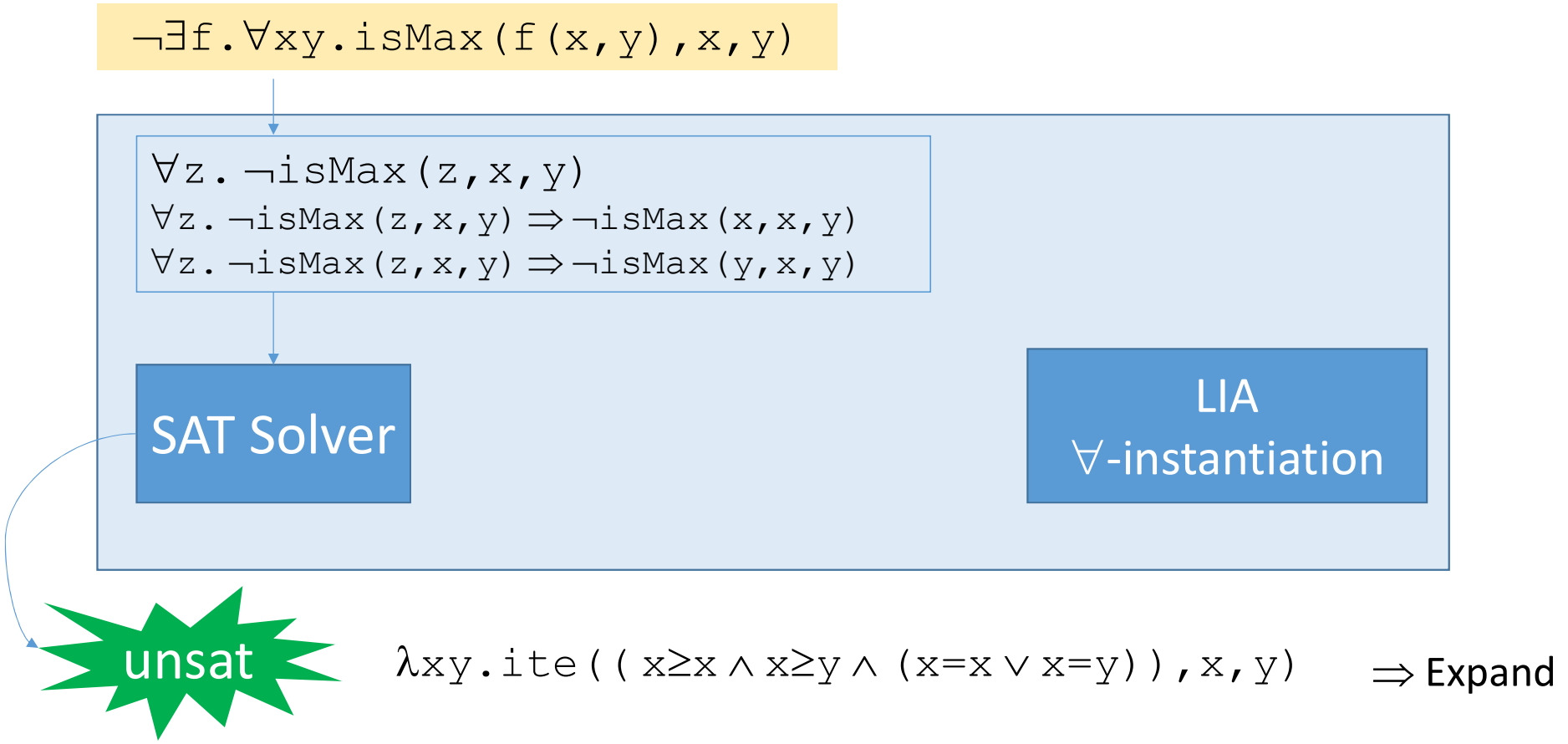
Single Invocation Synthesis in SMT



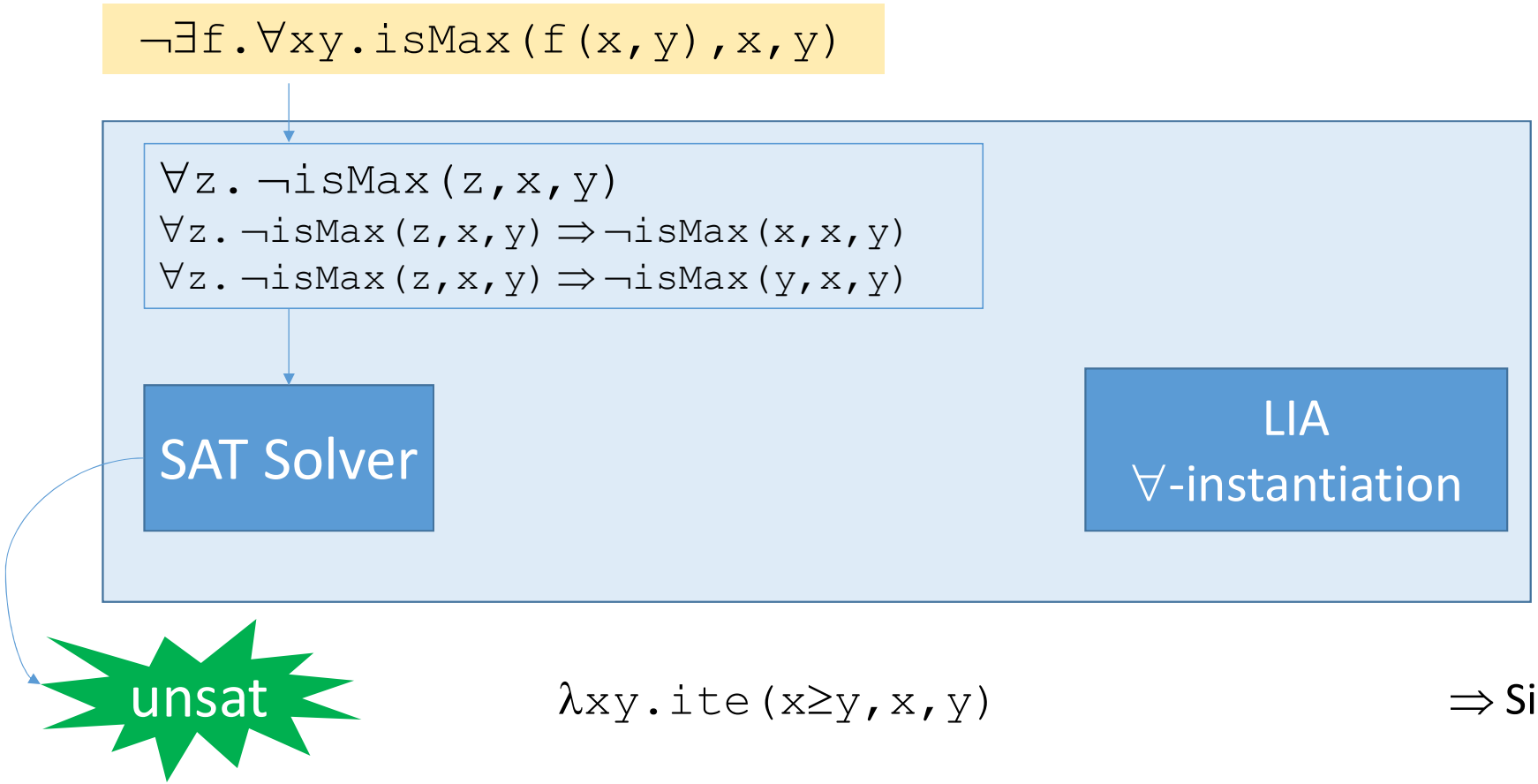
Single Invocation Synthesis in SMT



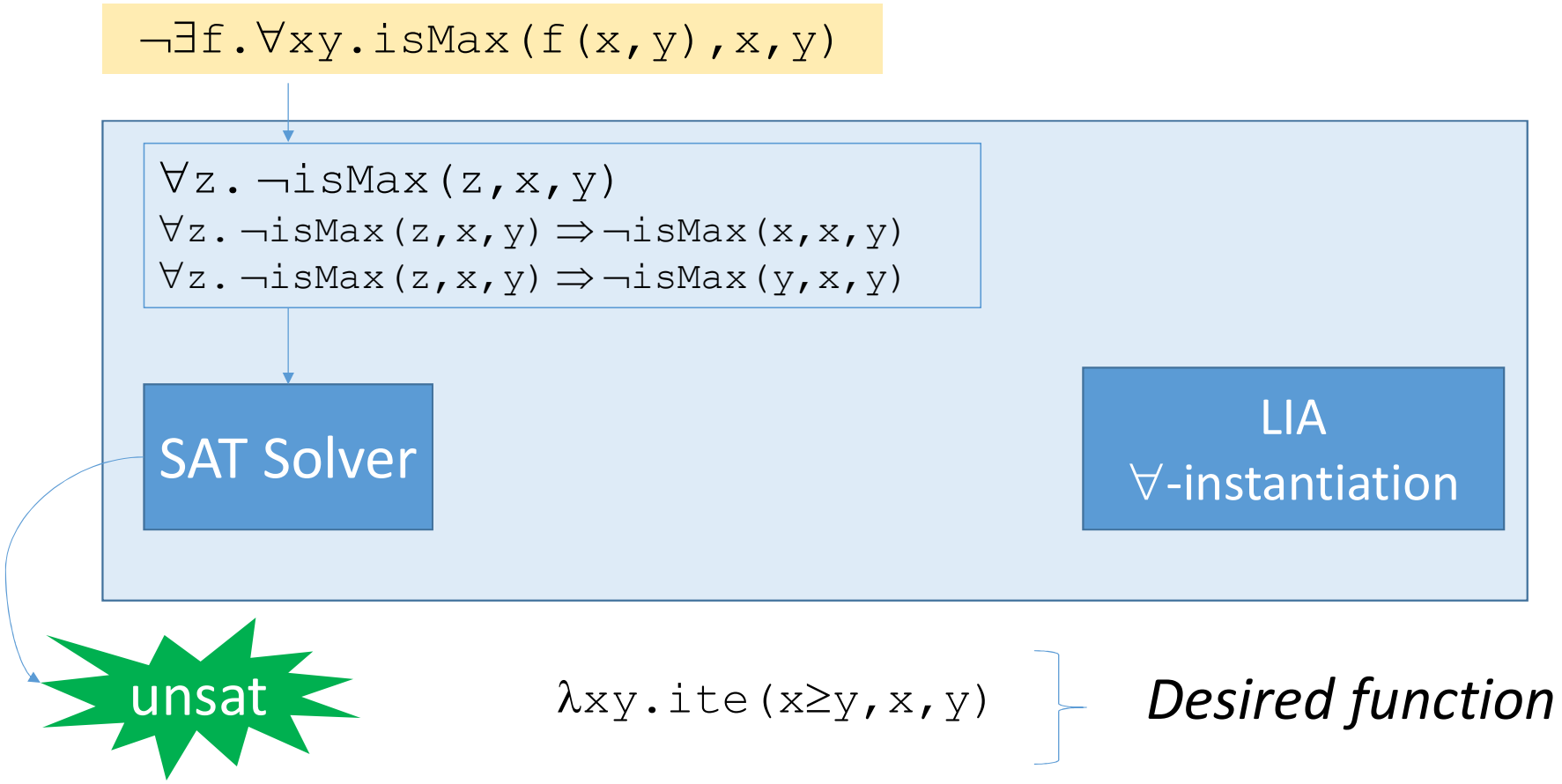
Single Invocation Synthesis in SMT



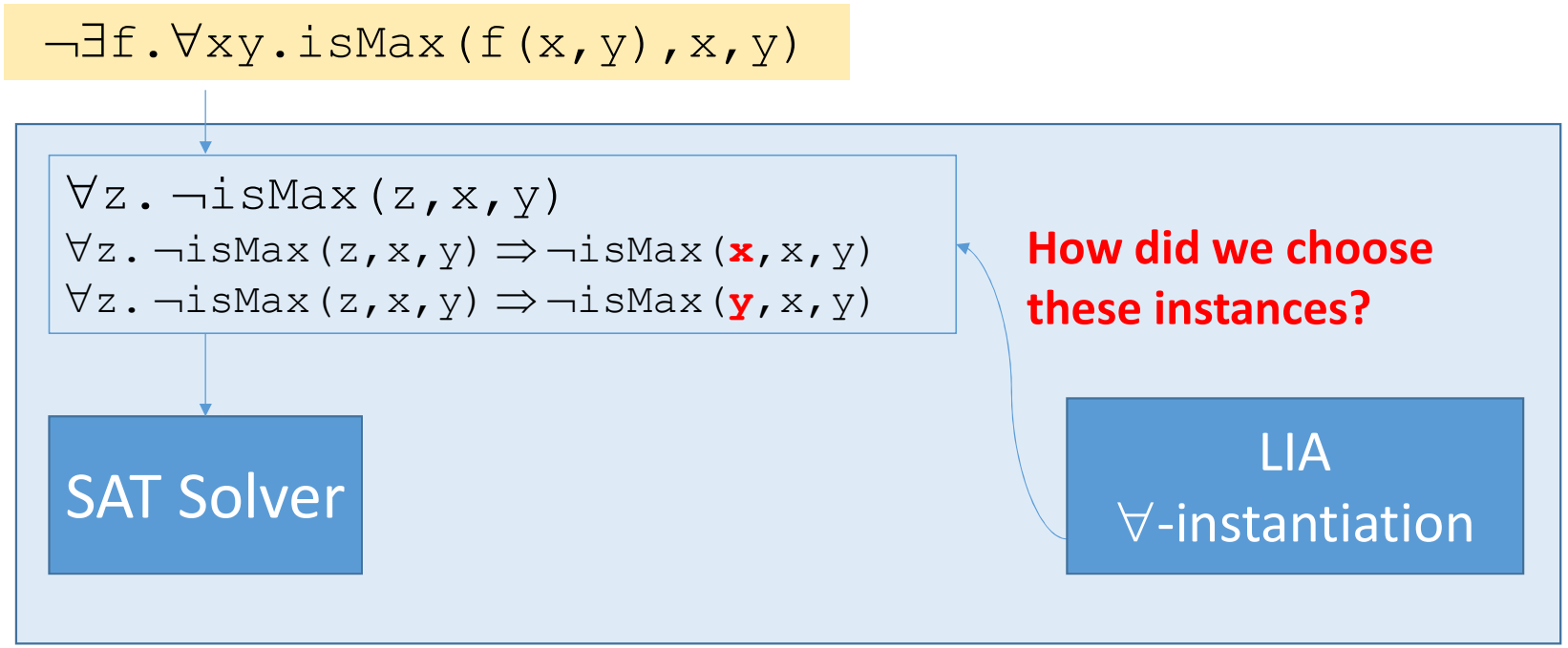
Single Invocation Synthesis in SMT



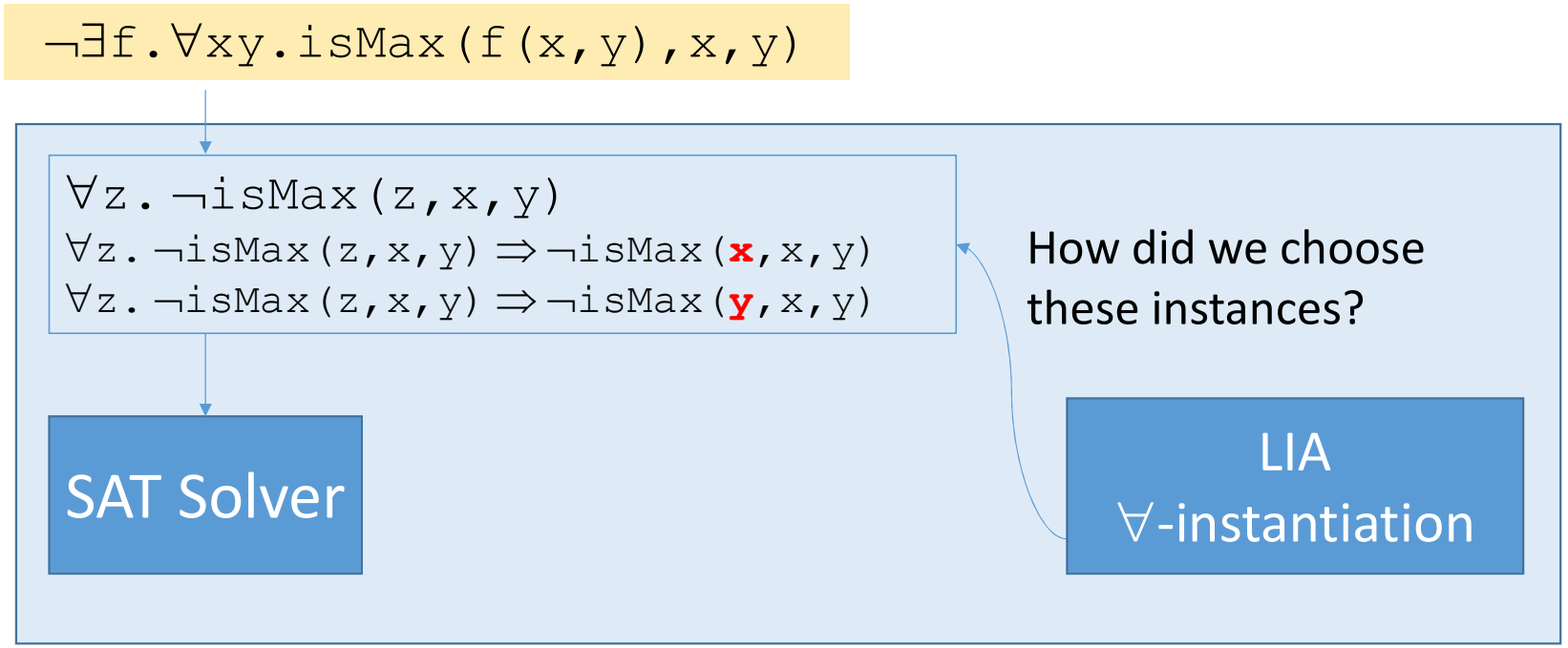
Single Invocation Synthesis in SMT



Single Invocation Synthesis in SMT



Single Invocation Synthesis in SMT



⇒ Use *counterexample-guided quantifier instantiation* (CEGQI)

Variants used in **[Monniaux 2010, Komuravelli et al 2014, Reynolds et al 2015, Dutertre 2015, Bjorner/Janota 2016, Fedyukovich et al 2016, Preiner et al 2017]**

Counterexample-Guided ∀-Instantiation

Quantifier Elimination Procedures

$\Leftarrow(\Rightarrow)^{?}$

Instantiation-Based procedures for $\exists\forall$ formulas

$\Leftarrow\Rightarrow$

Synthesis procedures for single-invocation properties

Counterexample-Guided ∀-Instantiation

- SMT+∀ linear arithmetic **[Monniaux 2010, Reynolds et al 2015, Dutertre 2015, Bjorner/Janota 2016]**
  - Based on maximal lower (minimal upper) bounds
    Analogous to **[Loos+Wiespfenning 93]**
  - Based on interior point method:
    Analogous to **[Ferrante+Rackoff 79]**
  - For integers: based on maximal lower (minimal upper) bounds (+⊂)
    Analogous to **[Cooper 72]**
- SMT + ∀ BV, QBF, finite domains **[Wintersteiger et al 2013, Rabe et al 2016, Preiner et al 2017]**
  - Based on model value, SyGuS, others?
- SMT + ∀ Strings, sets, floating points, datatypes
  - ???

***Finite** instantiation strategy ⇔ **sound** and **complete** synthesis procedure for s.i.*

Counterexample-Guided ∀-Instantiation

- SMT+∀ linear arithmetic **[Monniaux 2010, Reynolds et al 2015, Dutertre 2015, Bjorner/Janota 2016]**
  - Based on maximal lower (minimal upper) bounds
    Analogous to **[Loos+Wiespfenning 93]**
  - Based on interior point method:
    Analogous to **[Ferrante+Rackoff 79]**
  - For integers: based on maximal lower (minimal upper) bounds (+⊂)
    Analogous to **[Cooper 72]**
- SMT + ∀ BV, QBF, finite domains **[Wintersteiger et al 2013, Rabe et al 2016, Preiner et al 2017]**
  - Based on model value, SyGuS, others?
- SMT + ∀ Strings, sets, floating points, datatypes
  - ???

CHALLENGE #1:
How do we develop instantiation procedures for new SMT *theories*?

Finite *instantiation strategy* ⇔ ***sound*** *and* ***complete*** *synthesis procedure for s.i.*

Comparison of Synthesis Approaches

- *SMT + ∀-instantiation*
  - **Pro:** Very fast
  - **Pro:** Complete for (in)feasibility
  - **Con:** Non-optimal solutions
  - **Con:** Only for single-invocation conjectures
- Enumerative Search
  - **Con:** Typically very slow
  - **Con:** Cannot show infeasibility
  - **Pro:** Optimal (shortest) solutions
  - **Pro:** Applies to all second-order conjectures

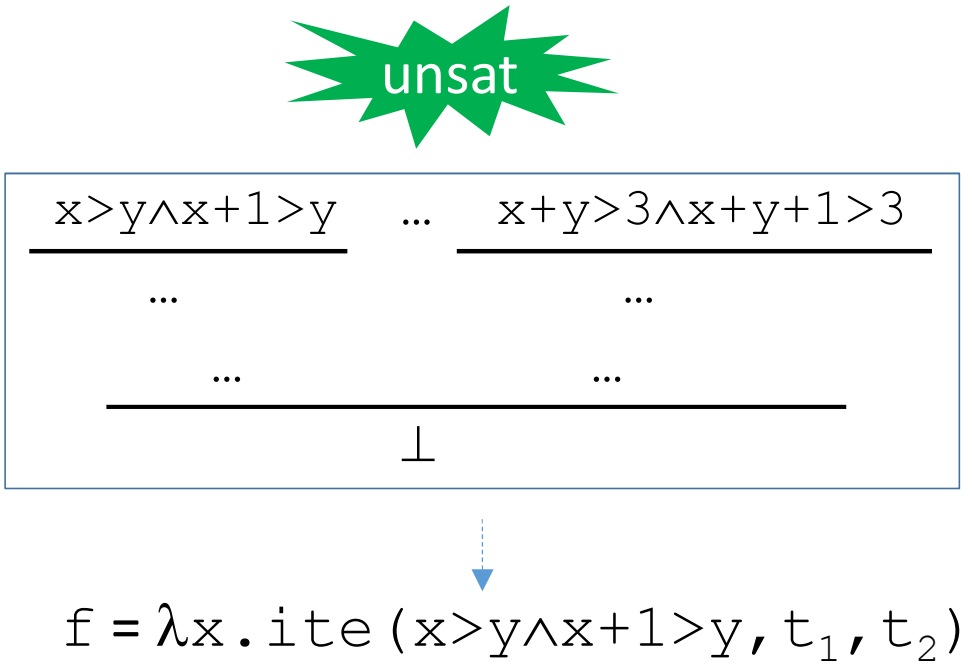
Comparison of Synthesis Approaches

- *SMT + ∀-instantiation*
  - **Pro:** Very fast
  - **Pro:** Complete for (in)feasibility
  - **Con:** Non-optimal solutions
  - **Con:** Only for single-invocation conjectures

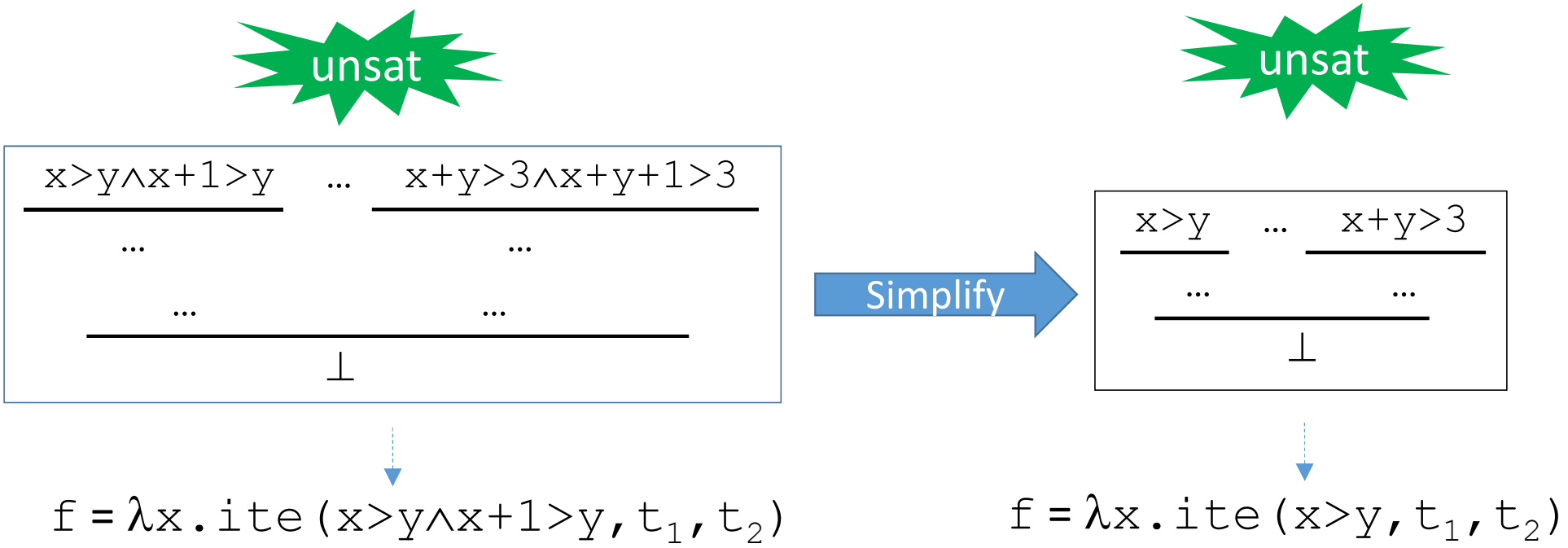
CHALLENGES

- Enumerative Search
  - **Con:** Typically very slow
  - **Con:** Cannot show infeasibility
  - **Pro:** Optimal (shortest) solutions
  - **Pro:** Applies to all second-order conjectures

Shorter Solutions via Proof Analysis



Shorter Solutions via Proof Analysis



CHALLENGE #2:

What *proofs/analysis* techniques are relevant for solution minimization?

What if conjecture is *Partially Single Invocation?*

$$\exists I.\forall xx'.(pre(x)\Rightarrow I(x))\wedge((I(x)\wedge T(x,x'))\Rightarrow I(x'))\wedge(I(x)\Rightarrow post(x))$$

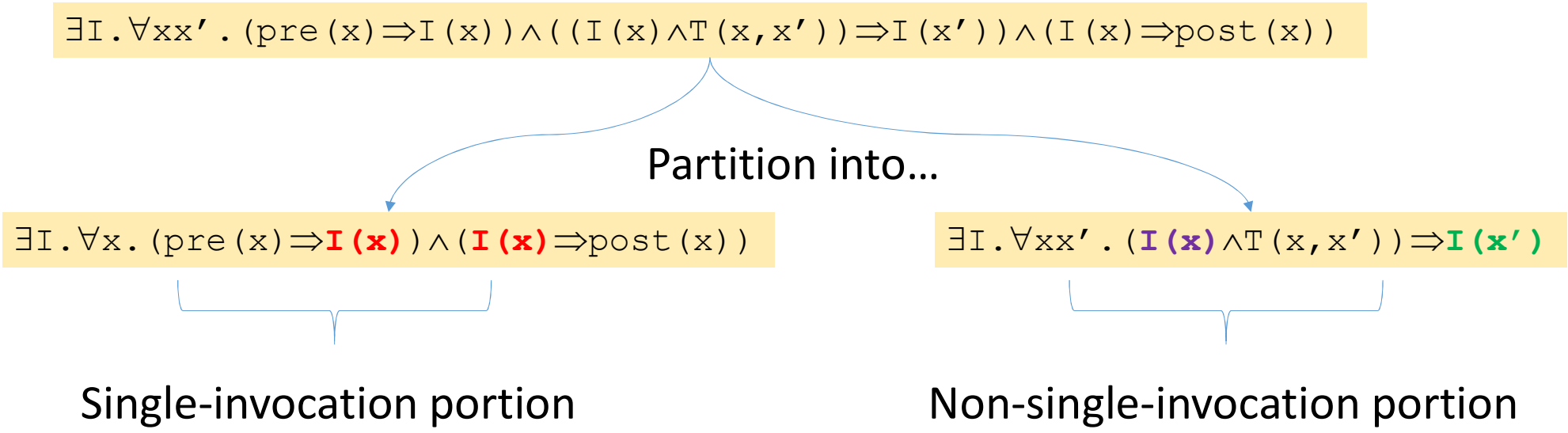
- Can single invocation techniques be leveraged beyond *single-invocation* conjectures?

What if conjecture is *Partially Single Invocation?*

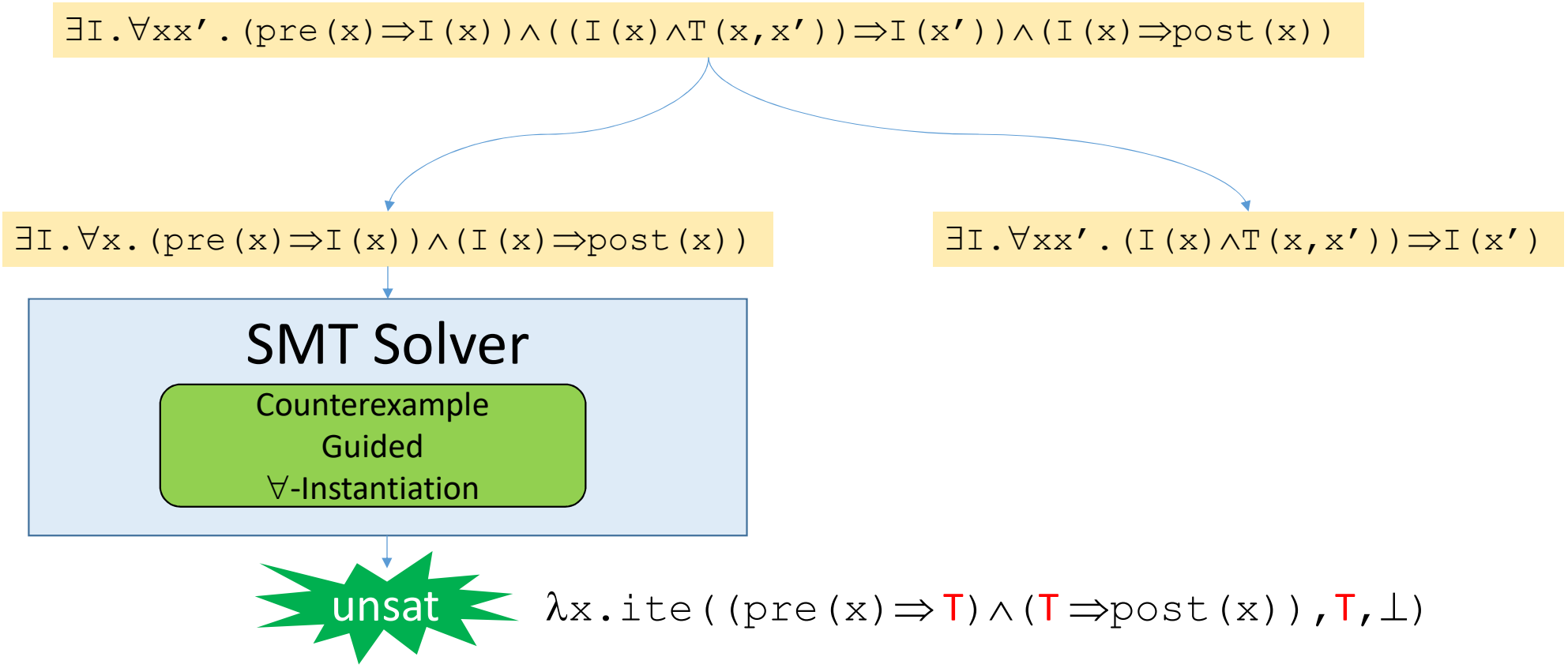
$$\exists I.\forall xx'.(pre(x)\Rightarrow I(x))\wedge((I(x)\wedge T(x,x'))\Rightarrow I(x'))\wedge(I(x)\Rightarrow post(x))$$

E.g. invariant synthesis problem for `I` w.r.t `pre`, `T`, `post`

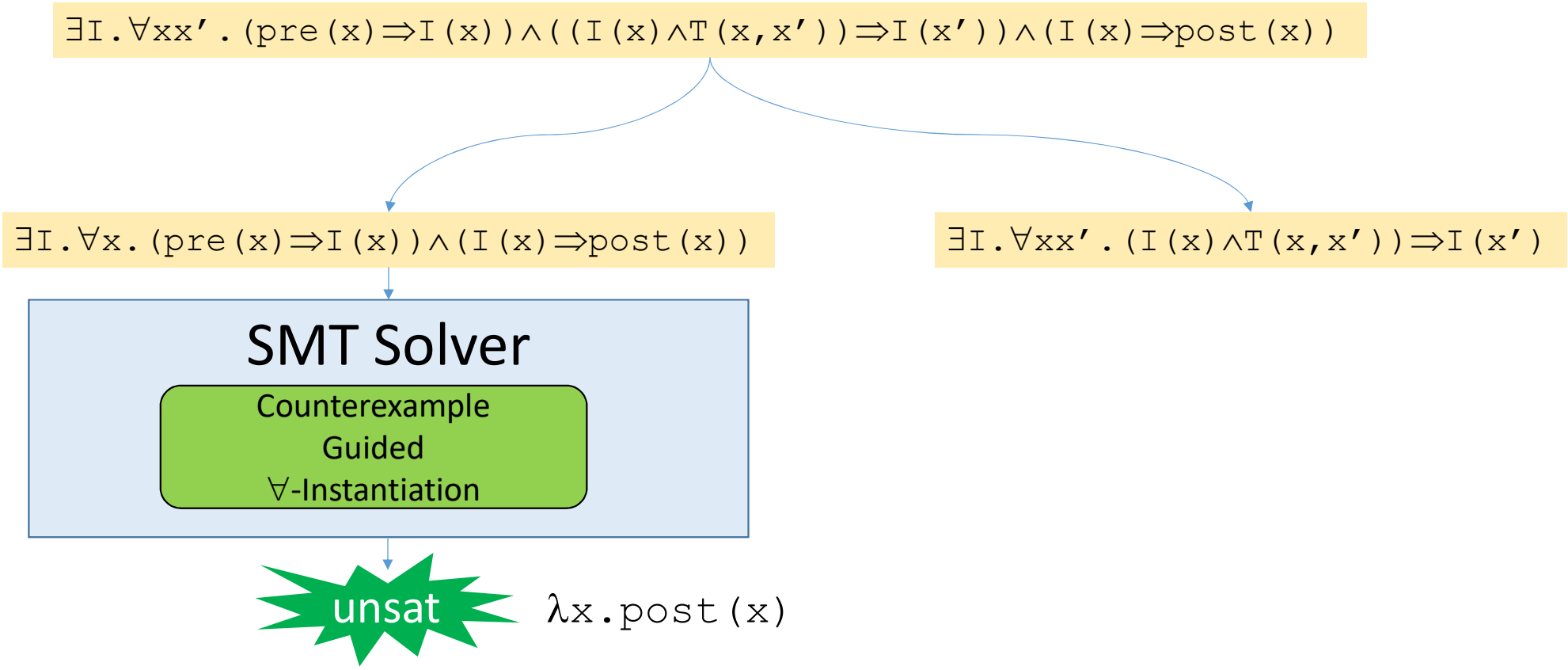
What if conjecture is *Partially Single Invocation?*



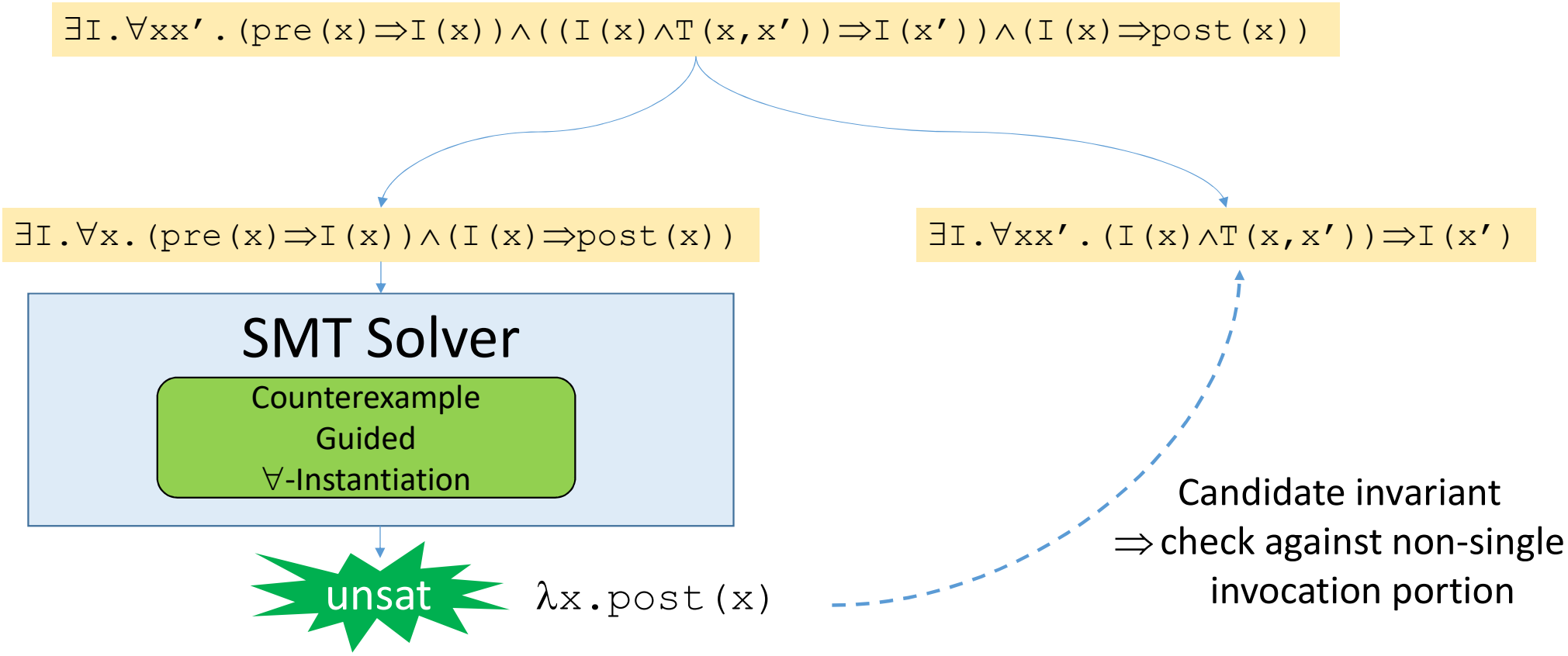
What if conjecture is *Partially Single Invocation?*



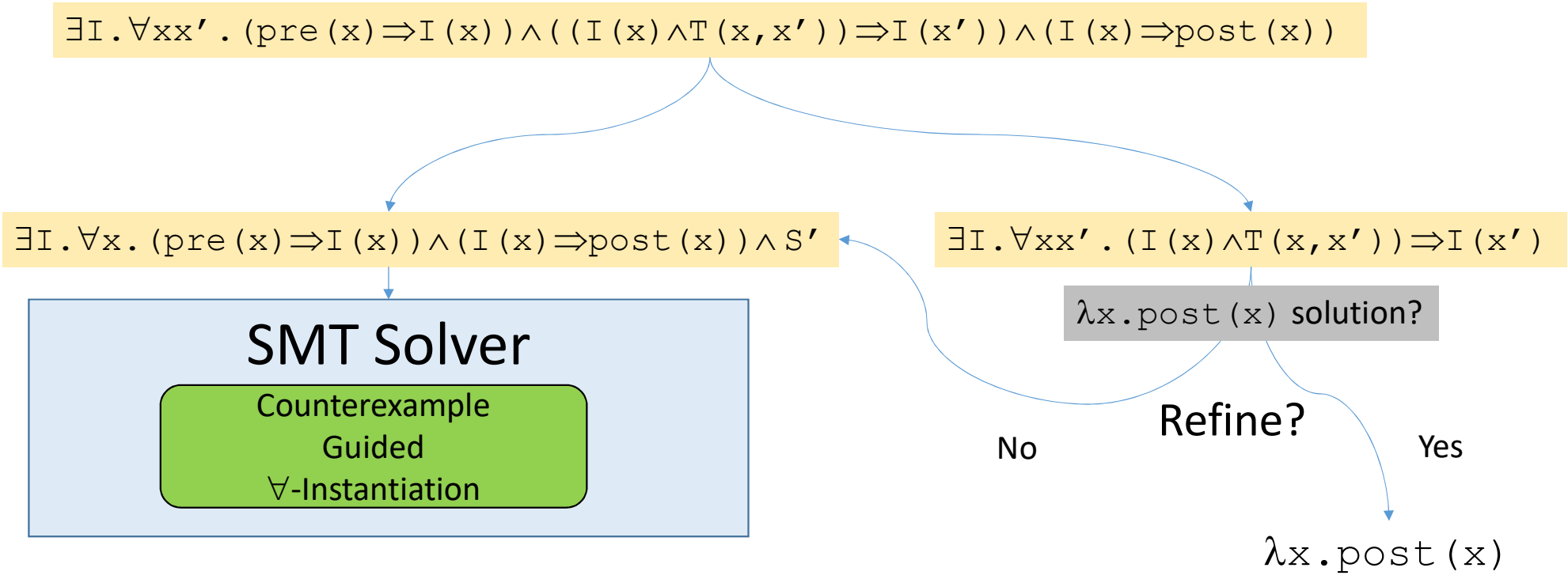
What if conjecture is *Partially Single Invocation?*



What if conjecture is *Partially Single Invocation?*



What if conjecture is *Partially Single Invocation?*



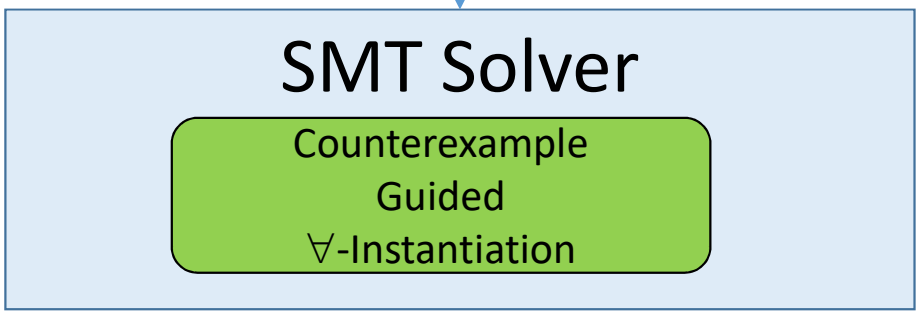
⇒Related to property-directed reachability (PDR) **[Bradley 2011]**

What if conjecture is *Partially Single Invocation?*

$\exists I.\forall xx'.(pre(x)\Rightarrow I(x))\wedge((I(x)\wedge T(x,x'))\Rightarrow I(x'))\wedge(I(x)\Rightarrow post(x))$

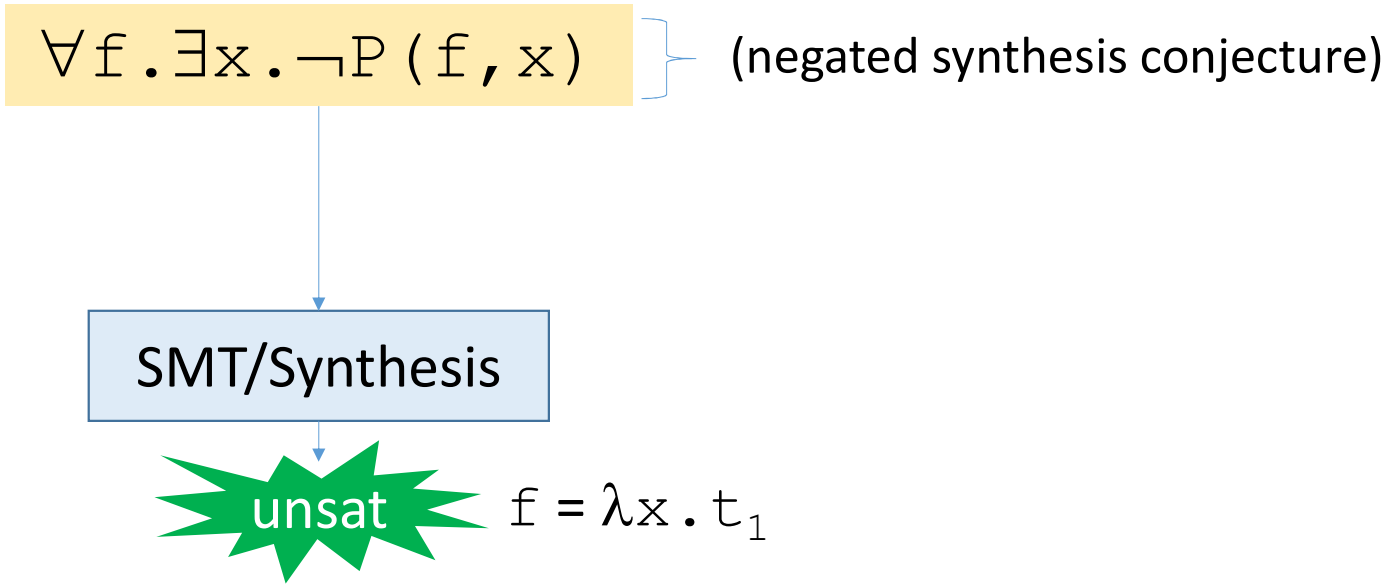
$\exists I.\forall x.(pre(x)\Rightarrow I(x))\wedge(I(x)\Rightarrow post(x))\wedge S'$

$\exists I.\forall xx'.(I(x)\wedge T(x,x'))\Rightarrow I(x')$

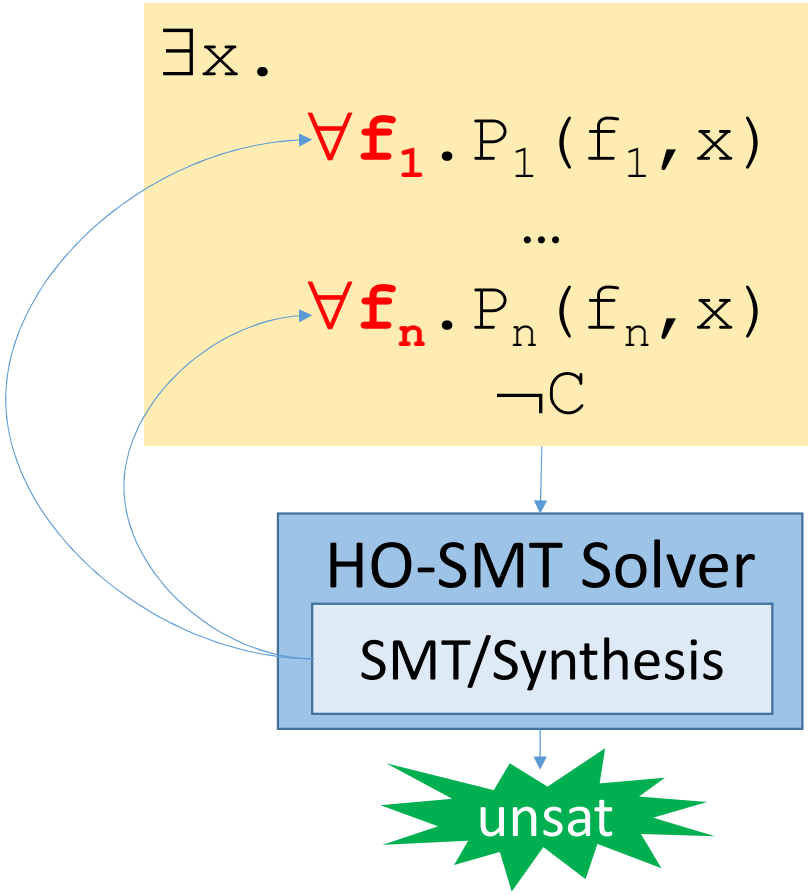


CHALLENGE #3:
How can we combine first-order and second-order techniques for *function* synthesis?

Synthesis for Higher-Order Theorem Proving?



Synthesis for Higher-Order Theorem Proving?



(free function symbols)

(multiple higher-order axioms)

(negated conjecture)

CHALLENGE #4:
Can we use synthesis techniques for *higher order* instantiation?