CEGIS(T)

CounterExample Guided Inductive Synthesis modulo Theories

Alessandro Abate[1], Cristina David[2], Pascal Kesseli[3], Daniel Kroening[13],
Elizabeth Polgreen[1]

[1]University of Oxford [2]University of Cambridge [3]Diffblue Ltd

CEGIS(T)

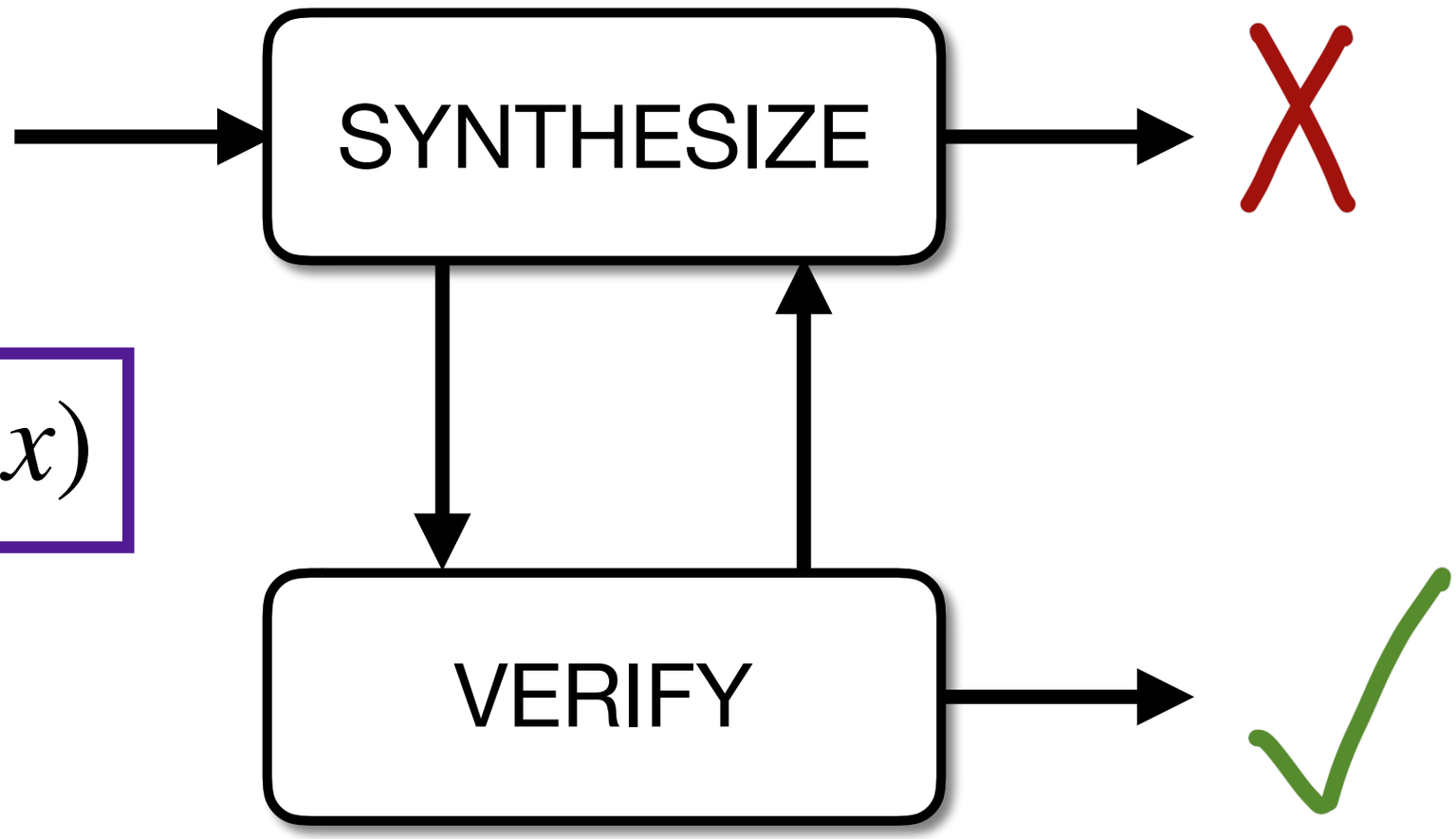
Program synthesis is hard.

CEGIS(T)

- Extension to CEGIS framework
- Uses general counterexamples and candidates
- Avoids enumerating search space
- Can synthesize programs that elude other solvers

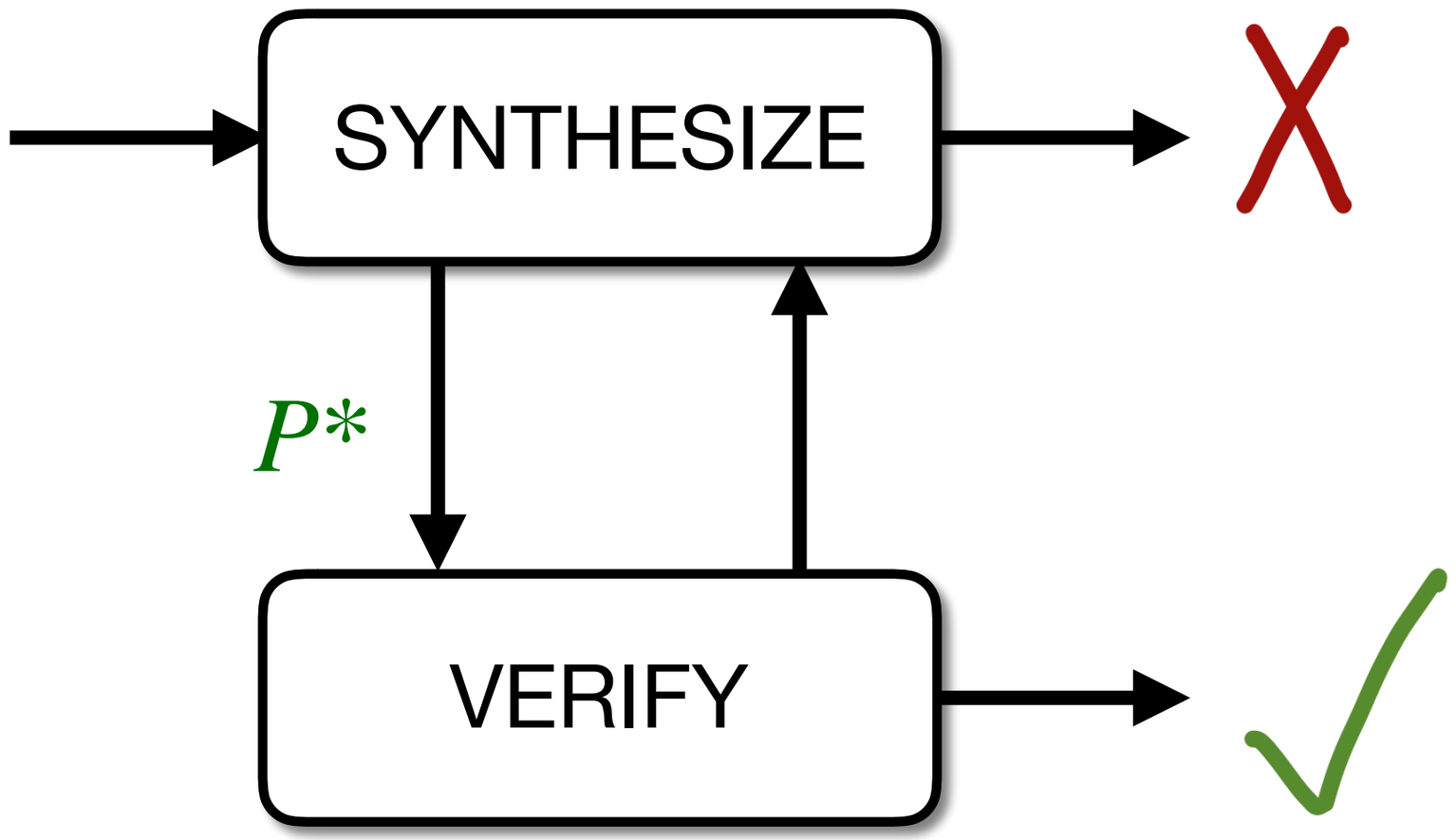
CEGIS

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

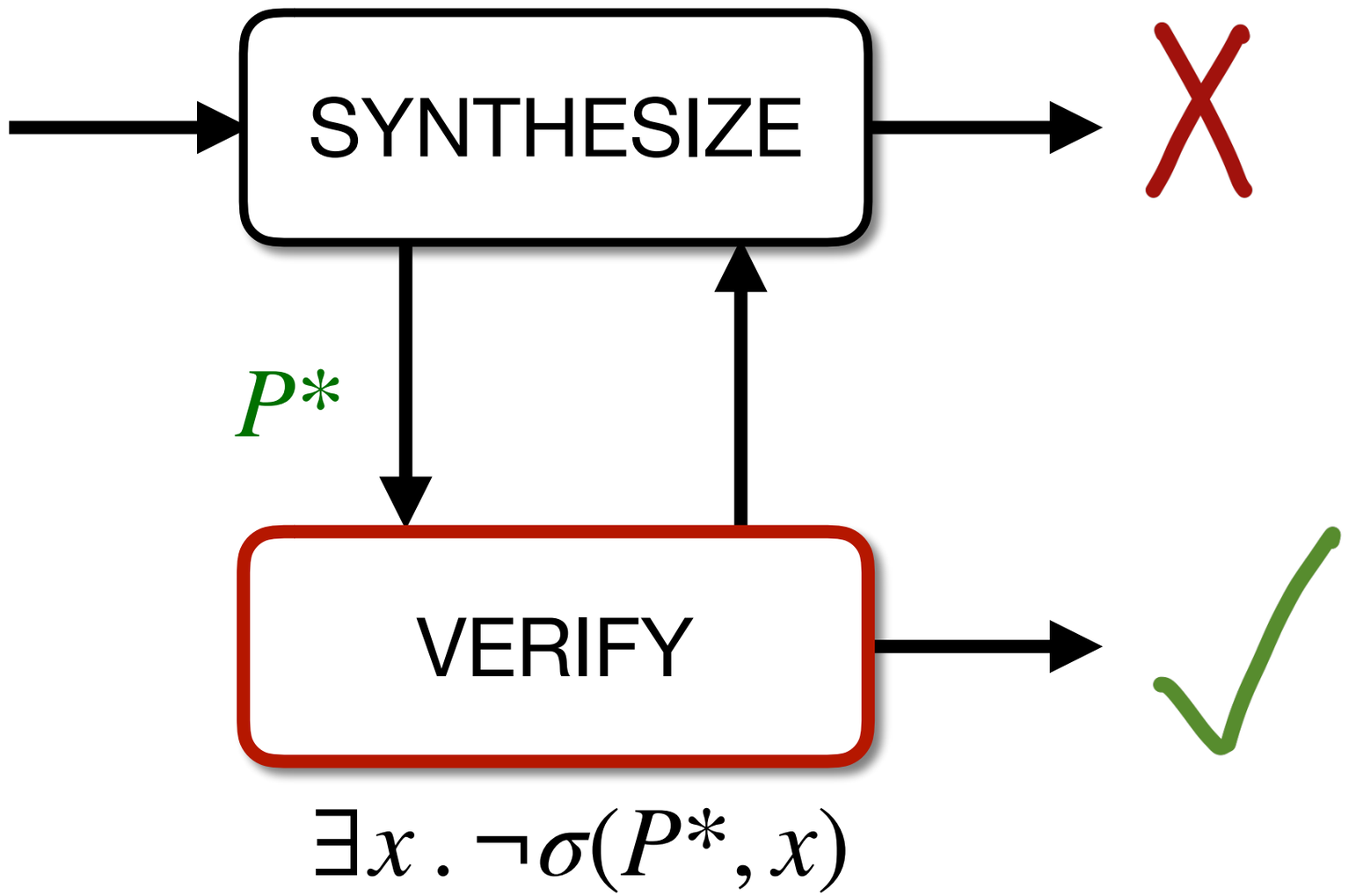


[PLDI 2007]

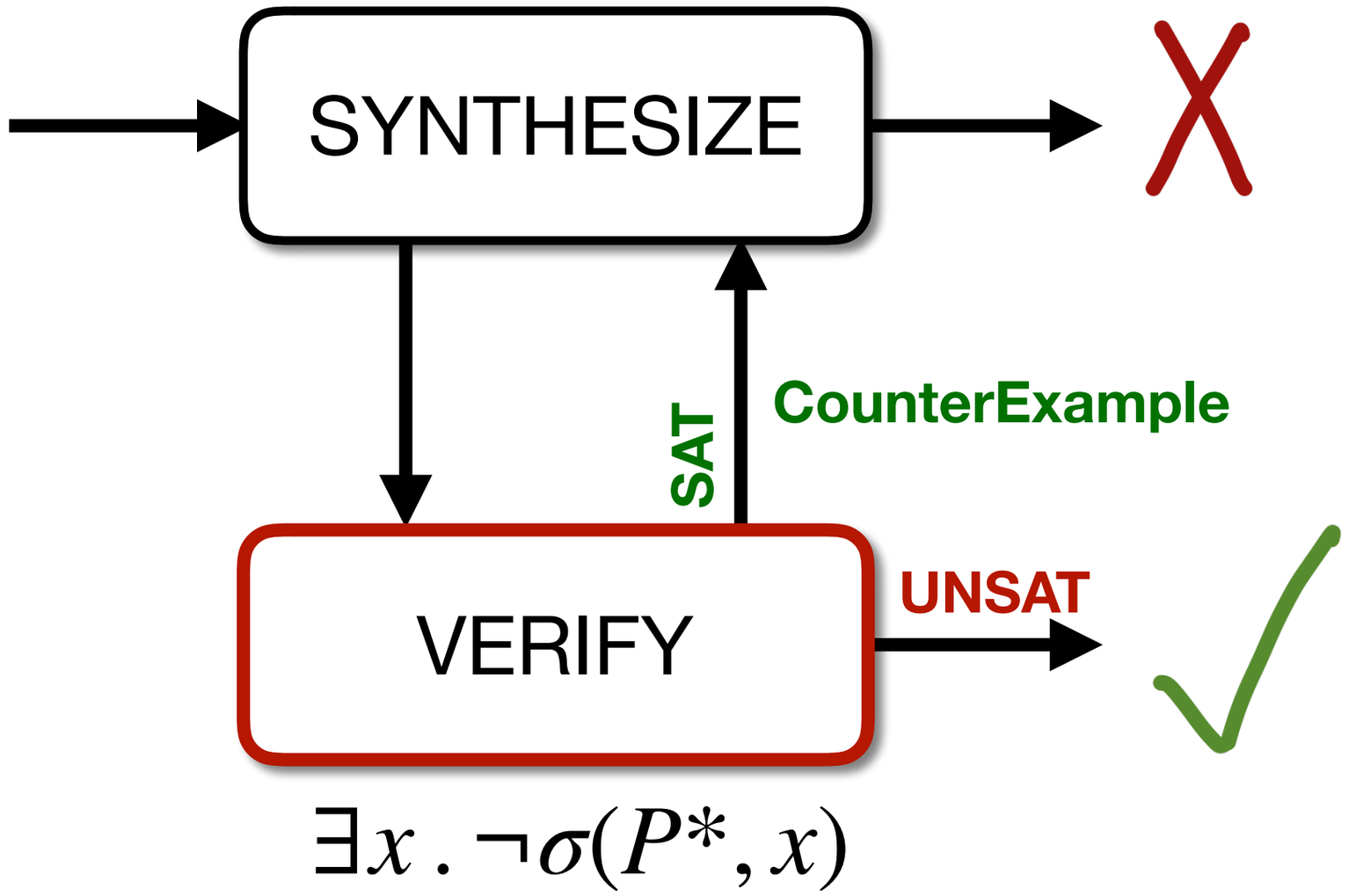
CEGIS



CEGIS

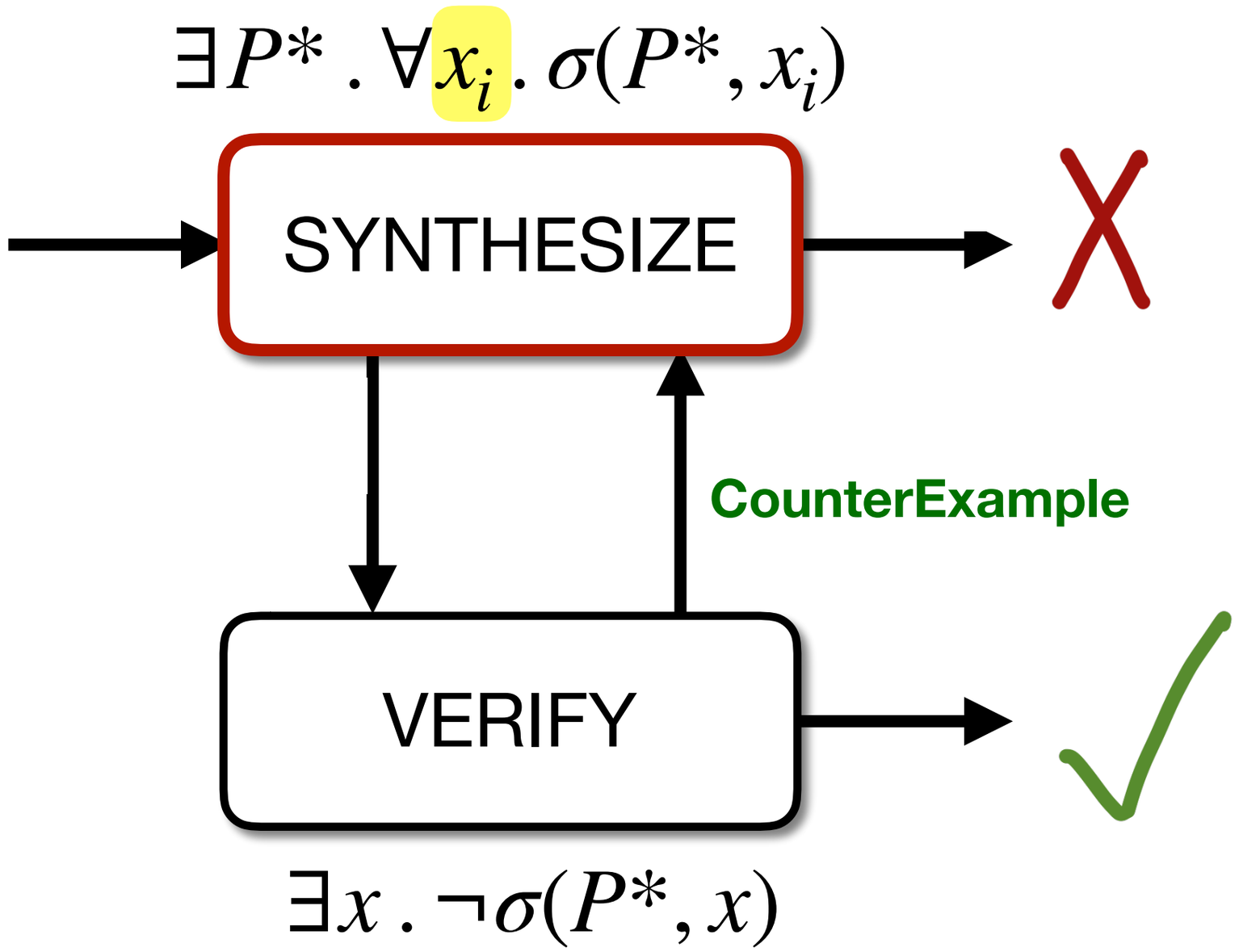


CEGIS

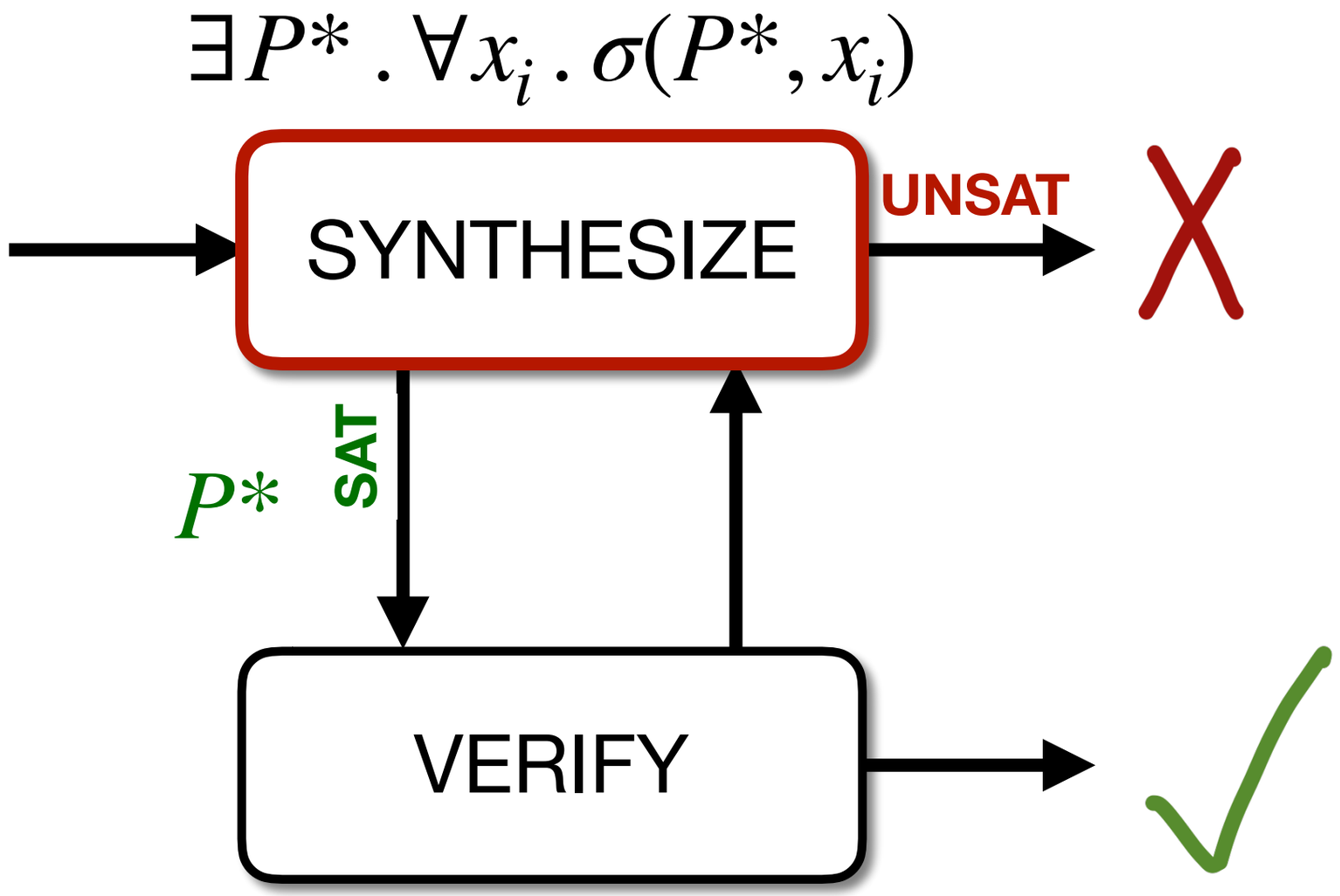


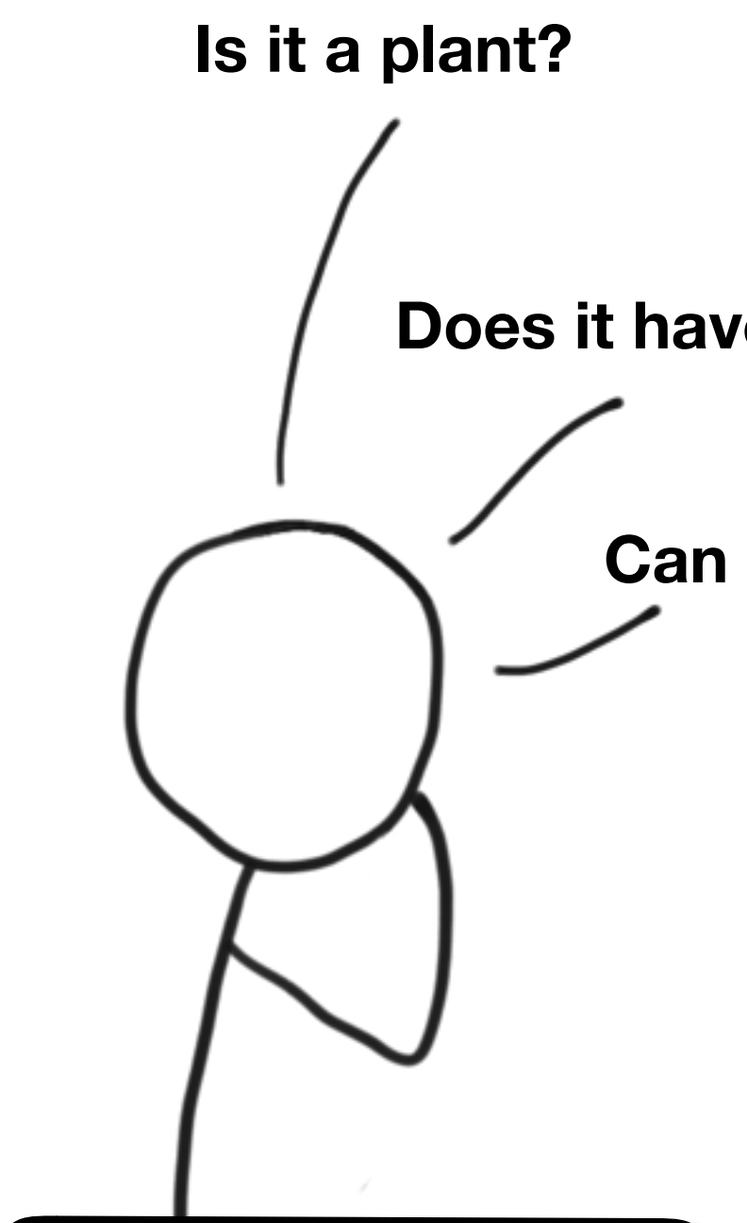
$$\exists x \,.\, \neg\sigma(P^*, x)$$

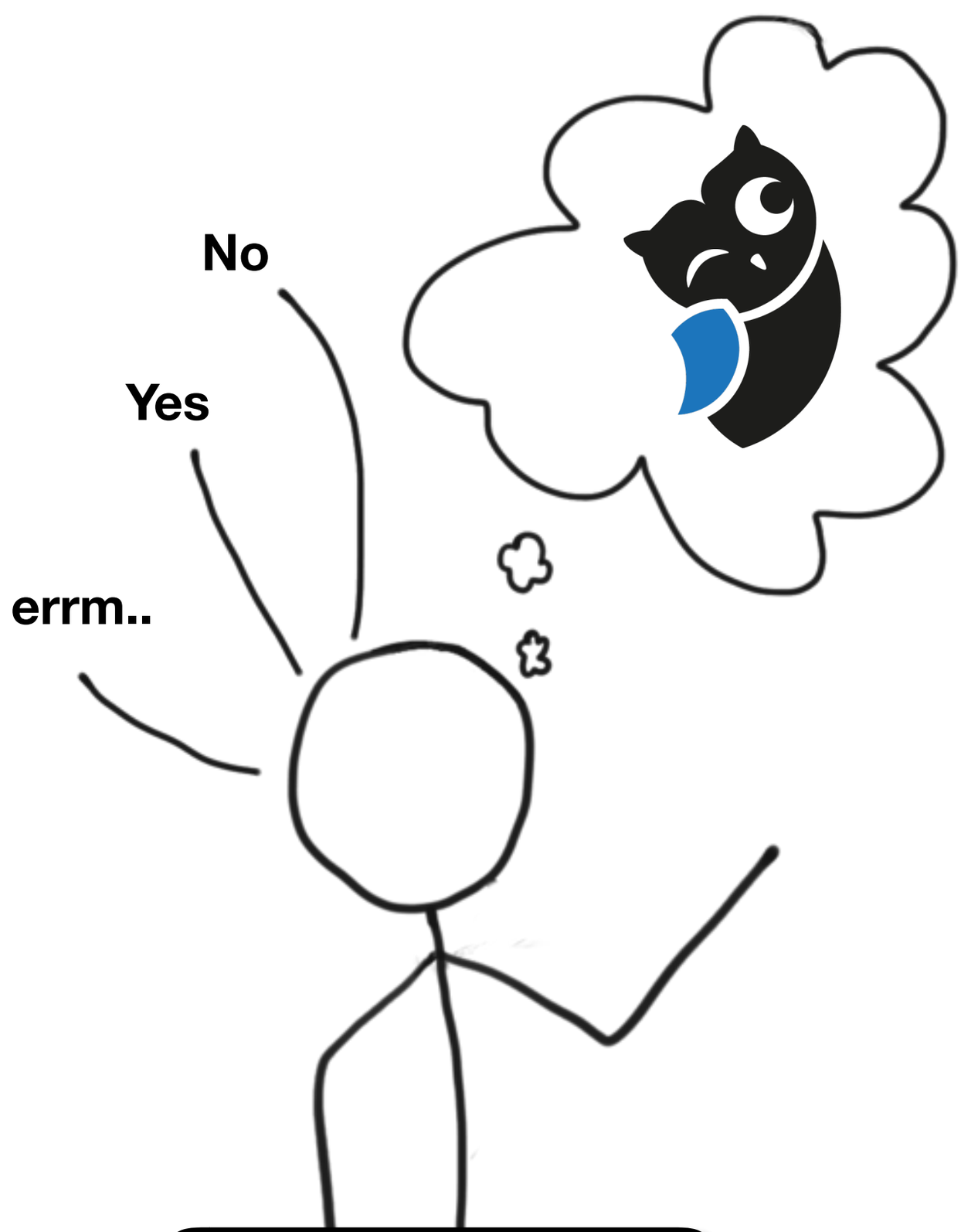
CEGIS

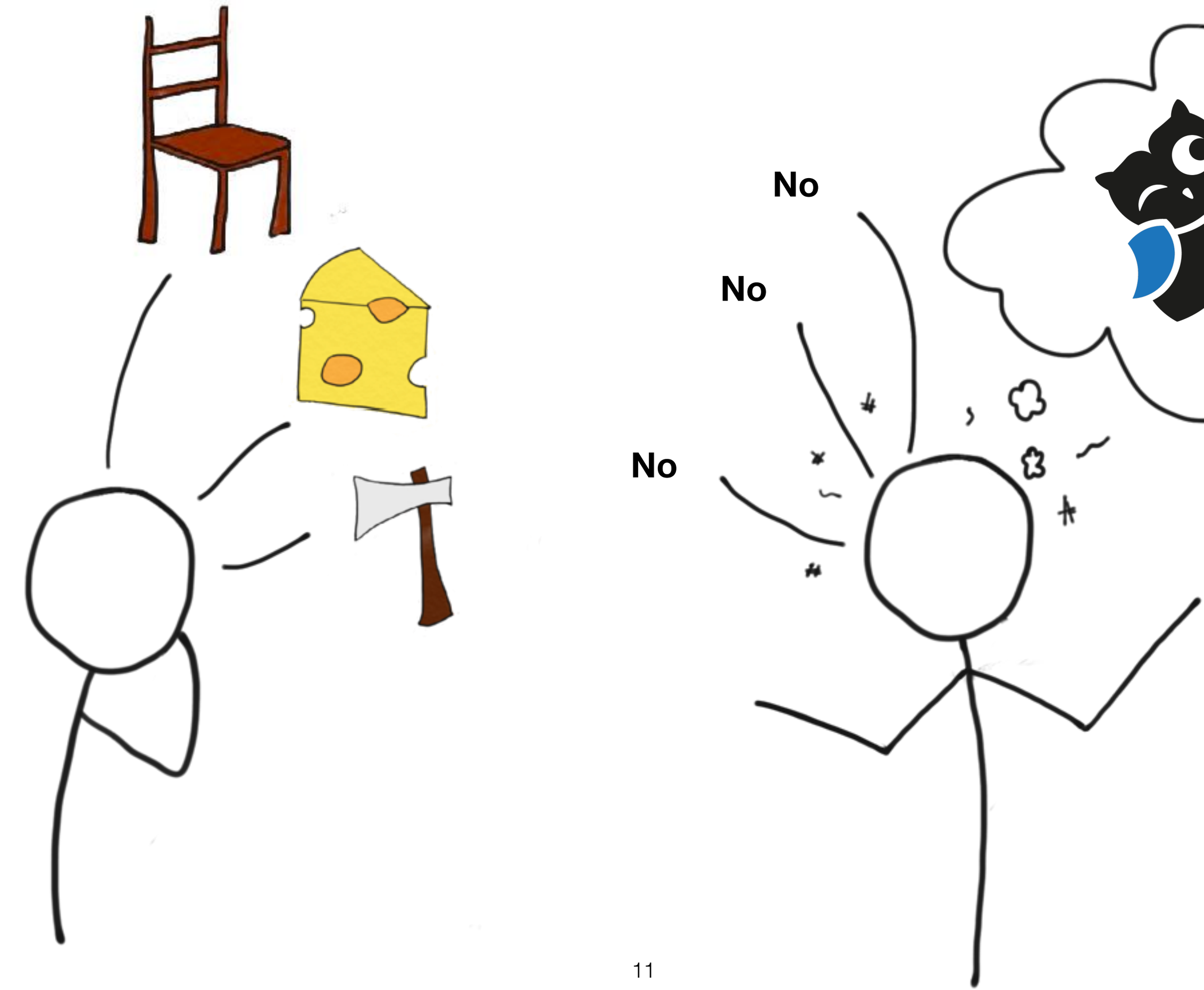

$\exists P^* . \forall x_i . \sigma(P^*, x_i)$
SYNTHESIZE
X
CounterExample
VERIFY
✓
$\exists x . \neg\sigma(P^*, x)$

CEGIS




Is it a plant?
Does it have legs?
Can I eat it?
SYNTHESIZE


No
Yes
errm..
VERIFY


No
No
No

Safety invariant

```
int x = 5;

while ( x < 1000)
   x++;

assert( 5 < x && x < 1005)
```

$$init(x) \iff x = 0$$

$$trans(x, x') \iff x' = x + 1$$

find $inv(x)$ **such that:**

$$init(x) \implies inv(x)$$

$$inv(x) \wedge (x < 1000) \wedge trans(x, x') \implies inv(x')$$

$$inv(x) \wedge \neg(x < 1000) \implies (x < 1005) \wedge (x > 5)$$

Safety invariant

```
int x = 5;

while ( x < 1000)
   x++;

assert( 5 < x && x < 1005)
```

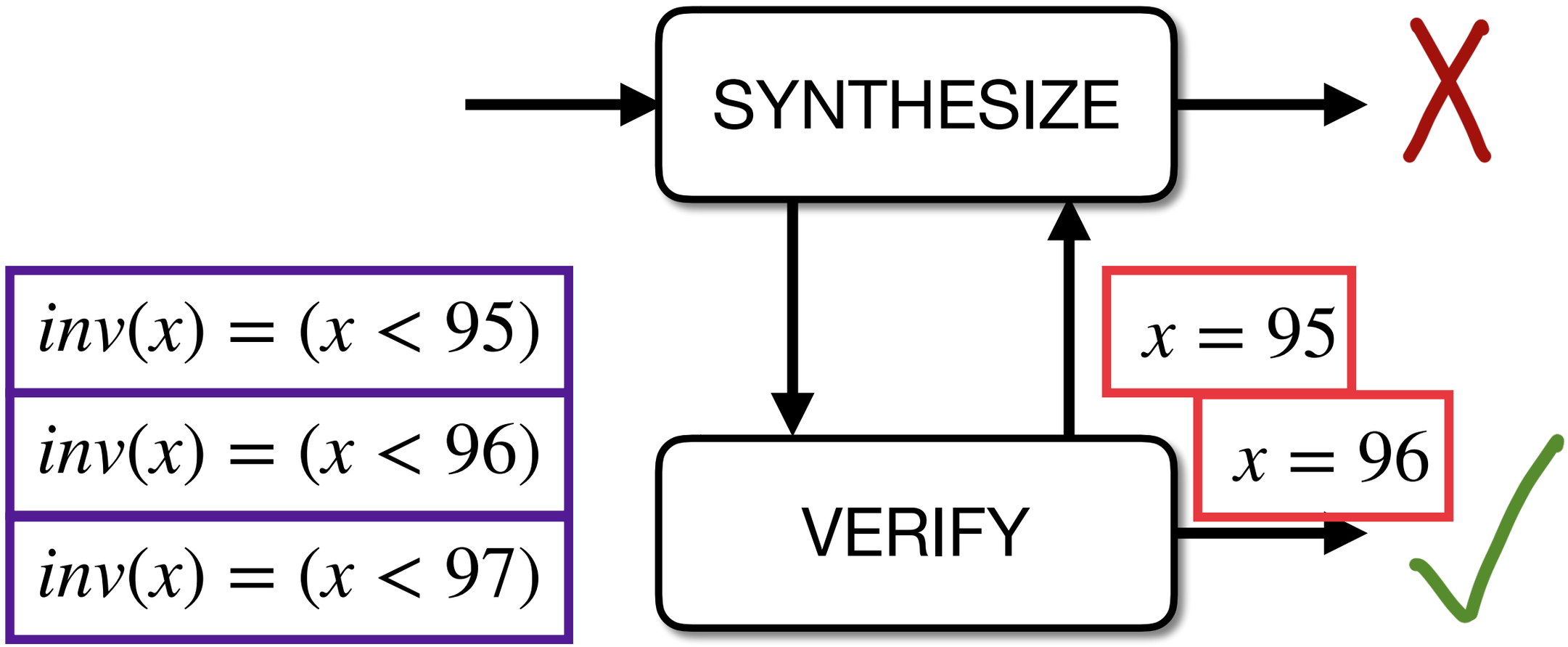
$$init(x) \iff x = 0$$

$$trans(x, x') \iff x' = x + 1$$

$$inv(x) = (4 < x) \wedge (x < 1003)$$

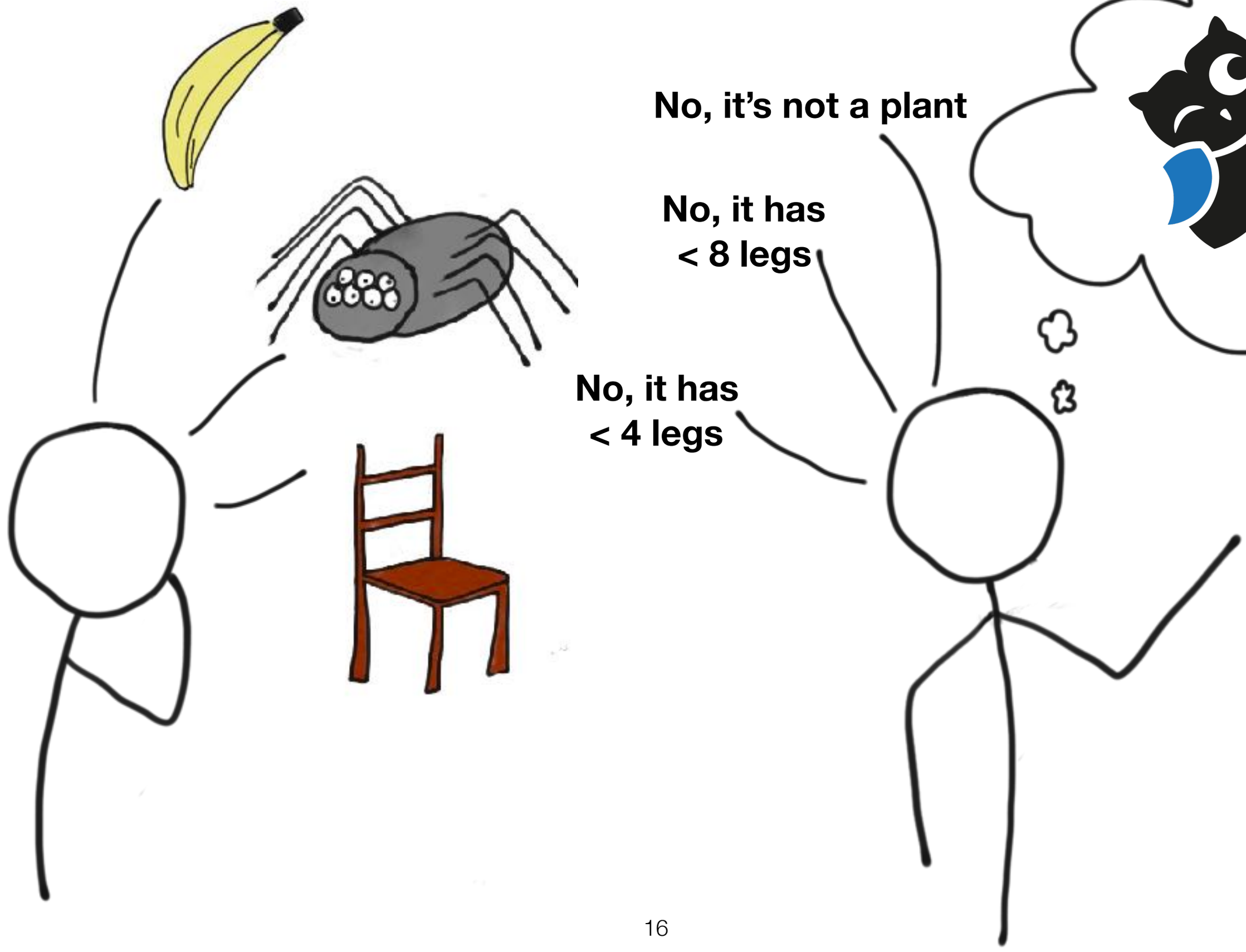
Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$



And so on ..

Can we ask more general questions?


No, it's not a plant
No, it has < 8 legs
No, it has < 4 legs

Can we give more general answers?

More general questions

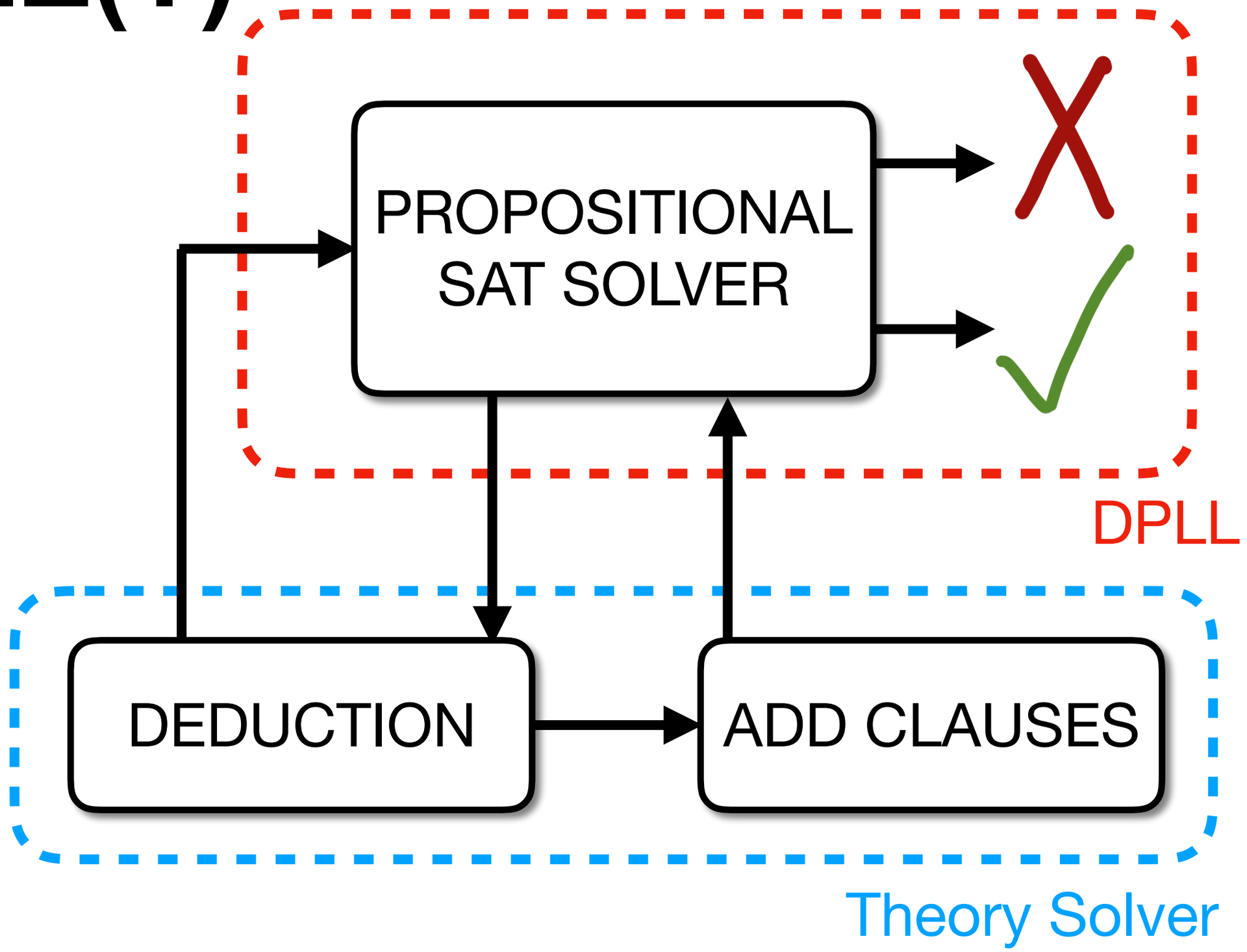
+

More general answers

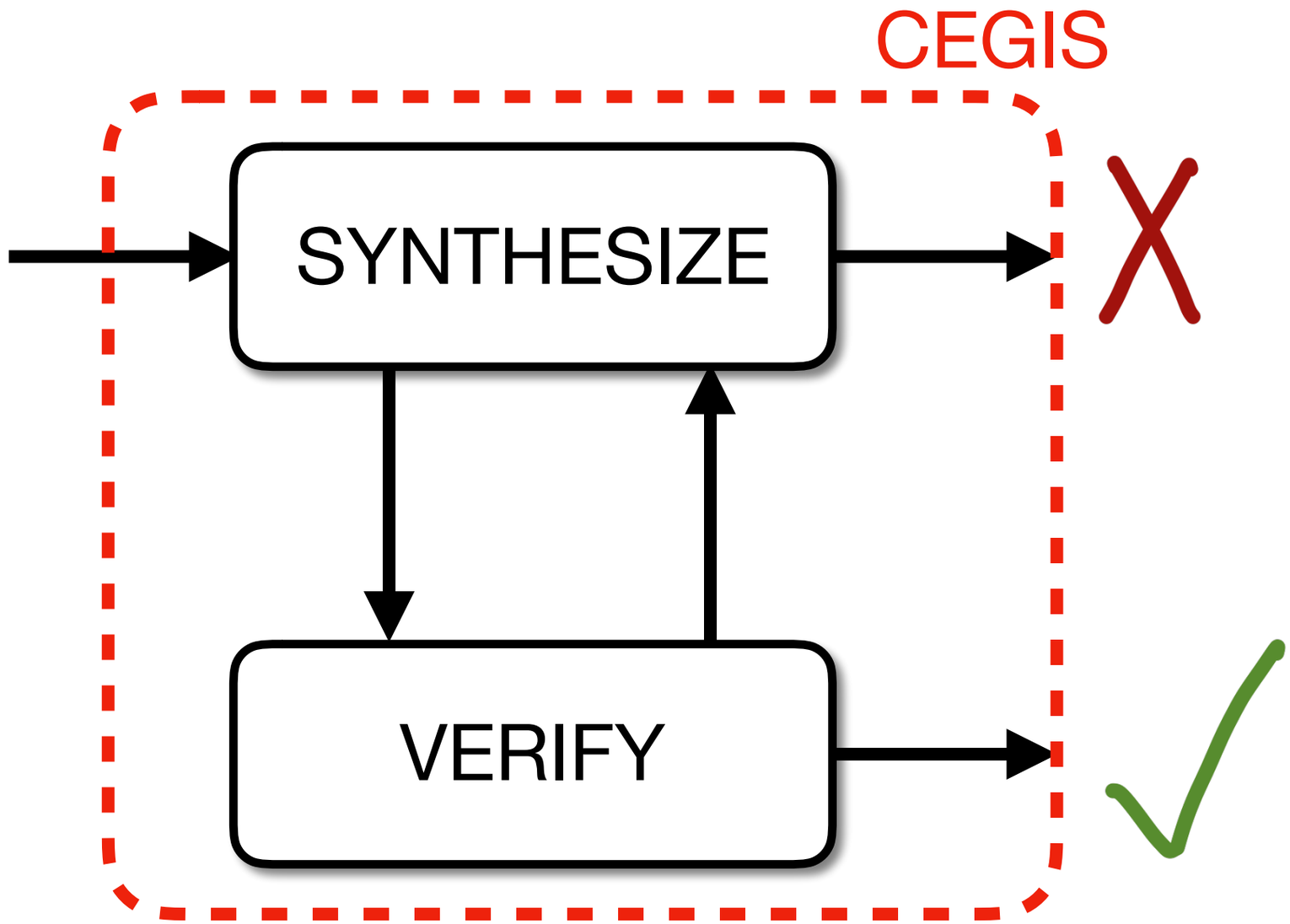
=

CEGIS(T)

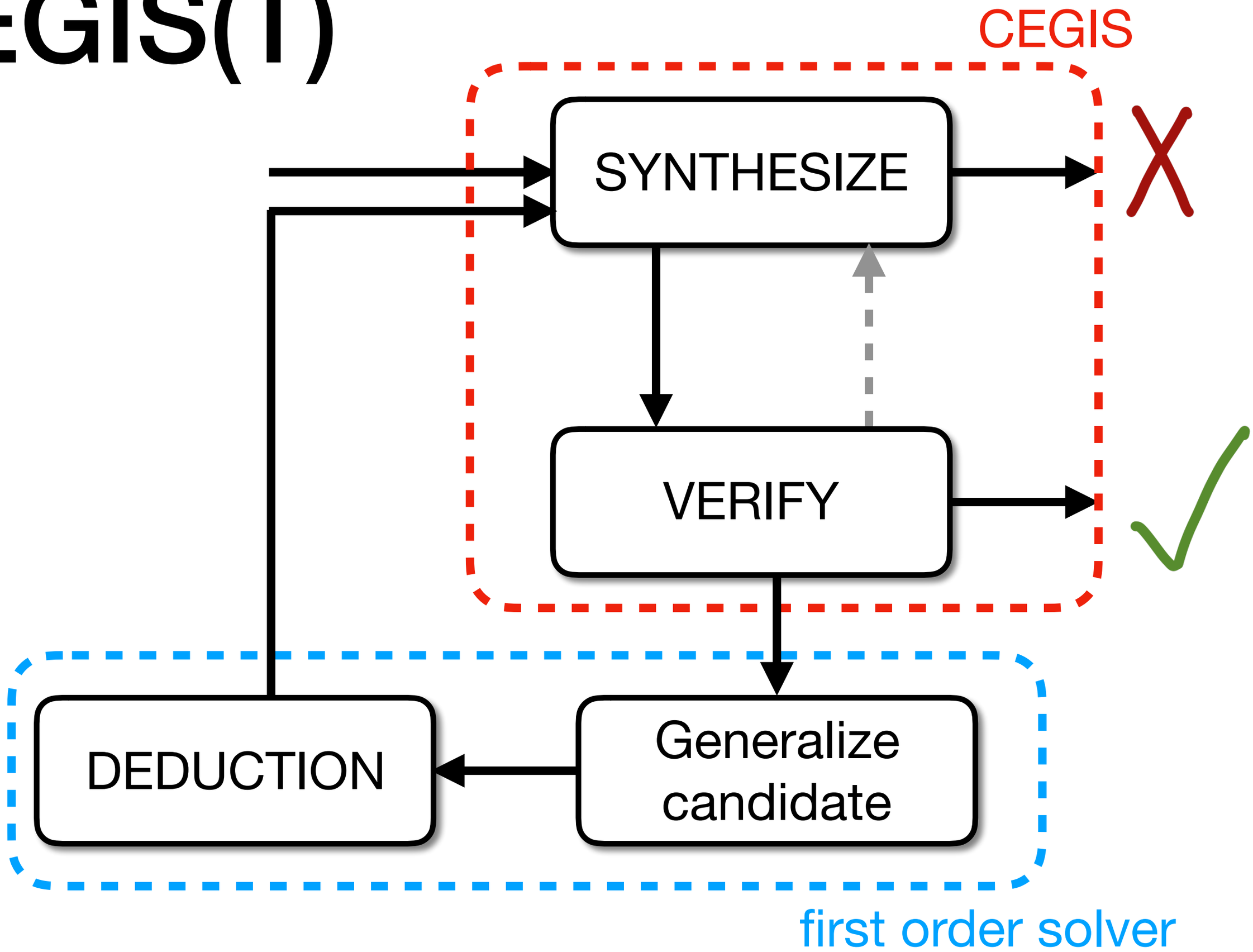
DPLL(T)


PROPOSITIONAL
SAT SOLVER
DPLL
DEDUCTION
ADD CLAUSES
Theory Solver

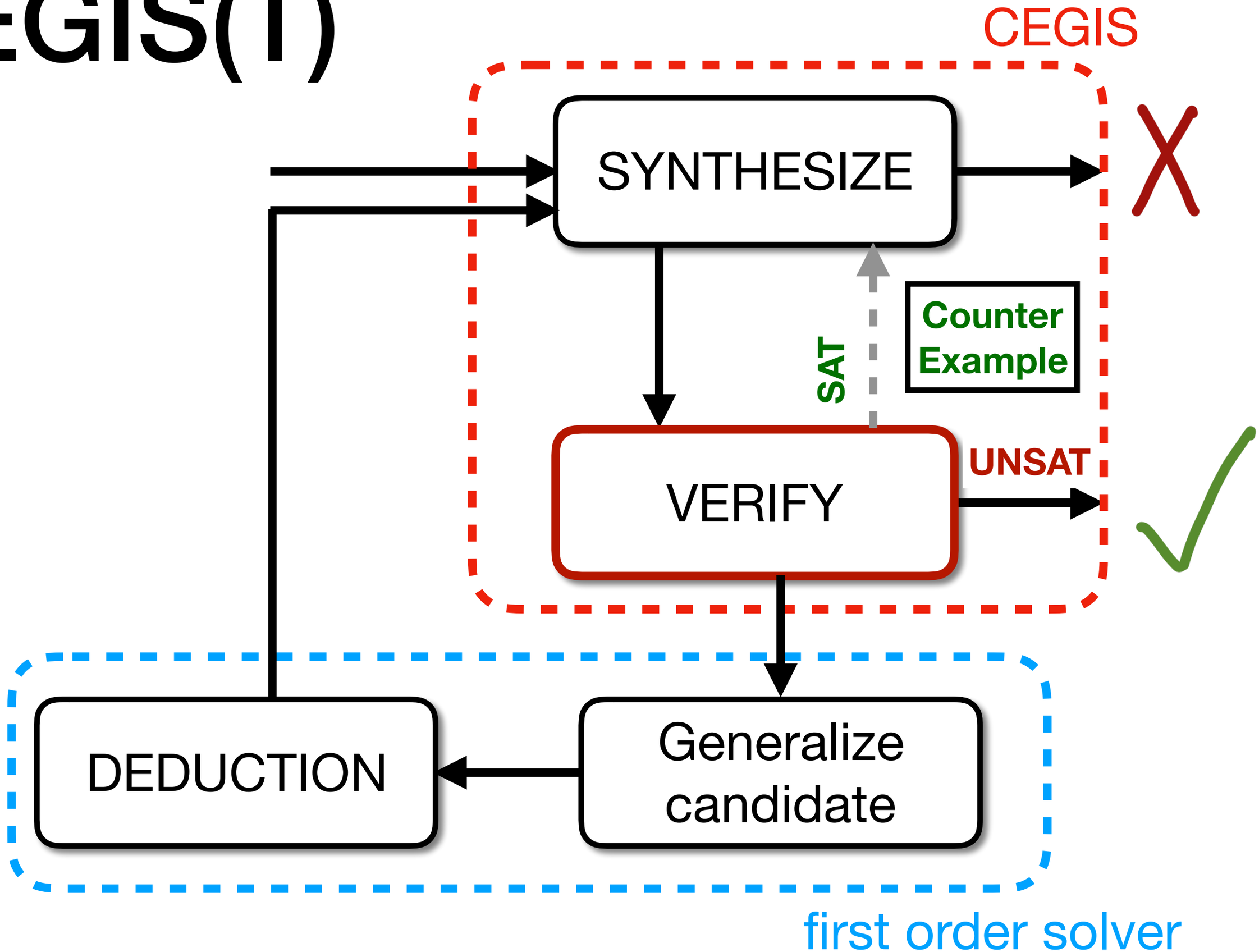
CEGIS(T)



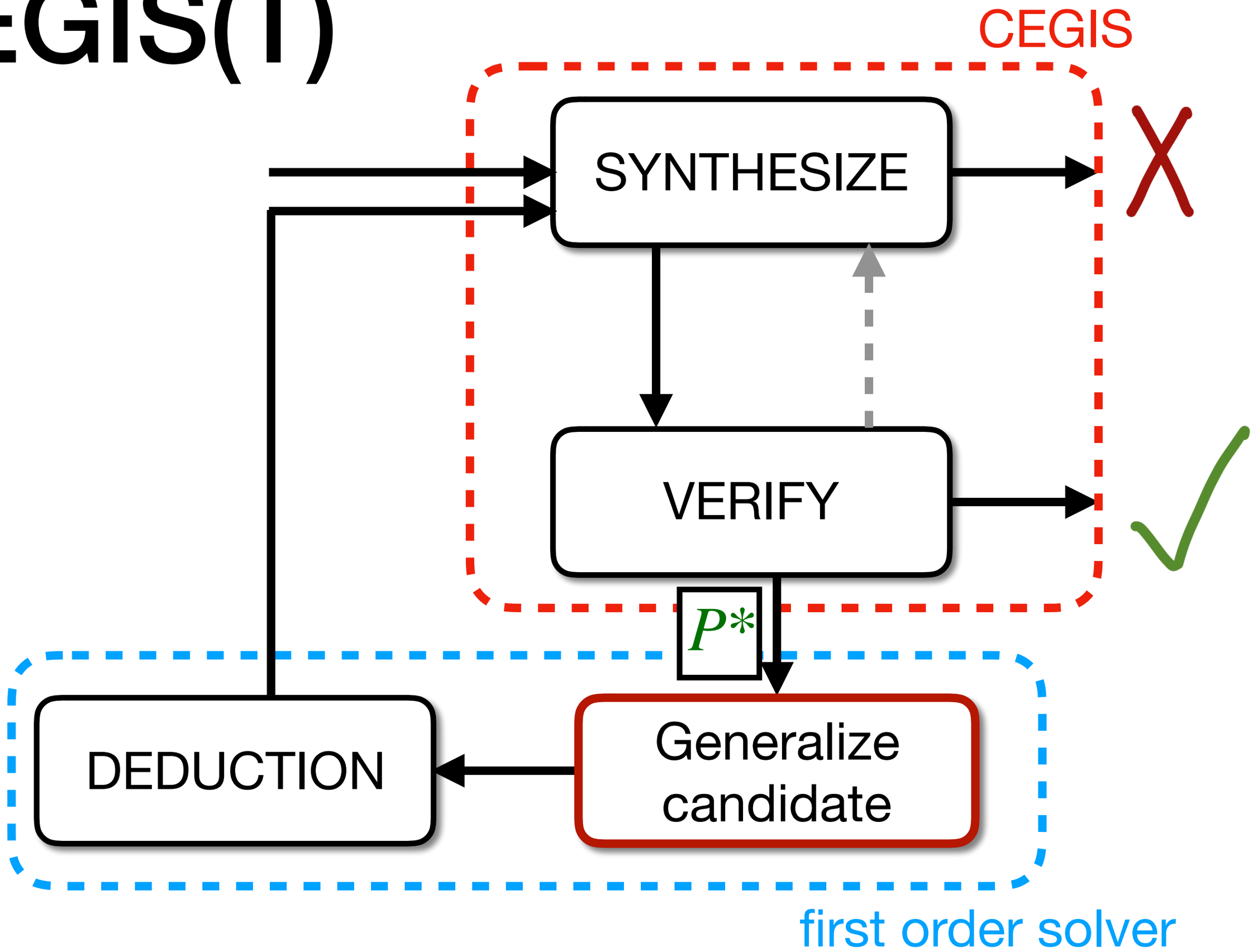
CEGIS(T)


CEGIS
SYNTHESIZE
VERIFY
DEDUCTION
Generalize candidate
first order solver

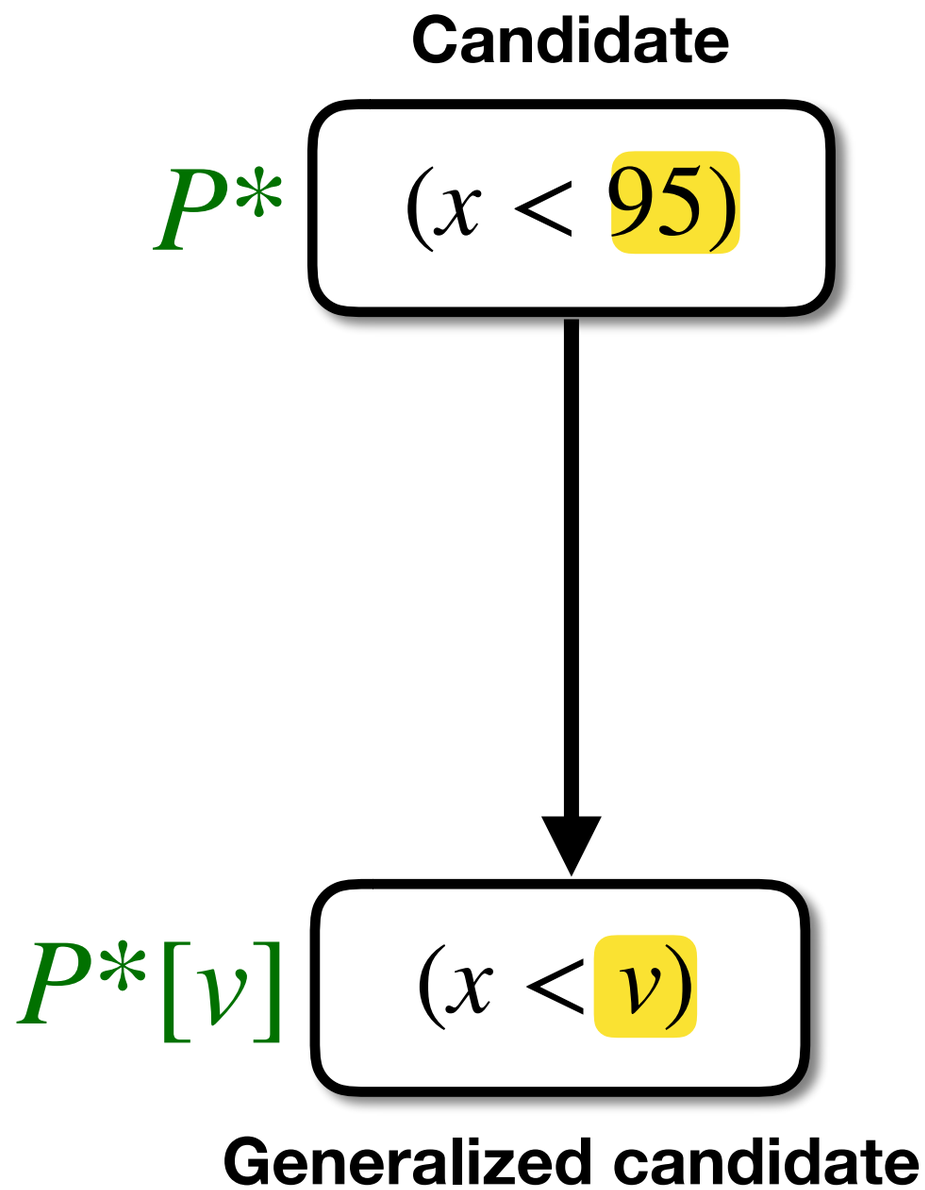
CEGIS(T)


CEGIS
SYNTHESIZE
X
Counter Example
SAT
UNSAT
VERIFY
DEDUCTION
Generalize candidate
first order solver

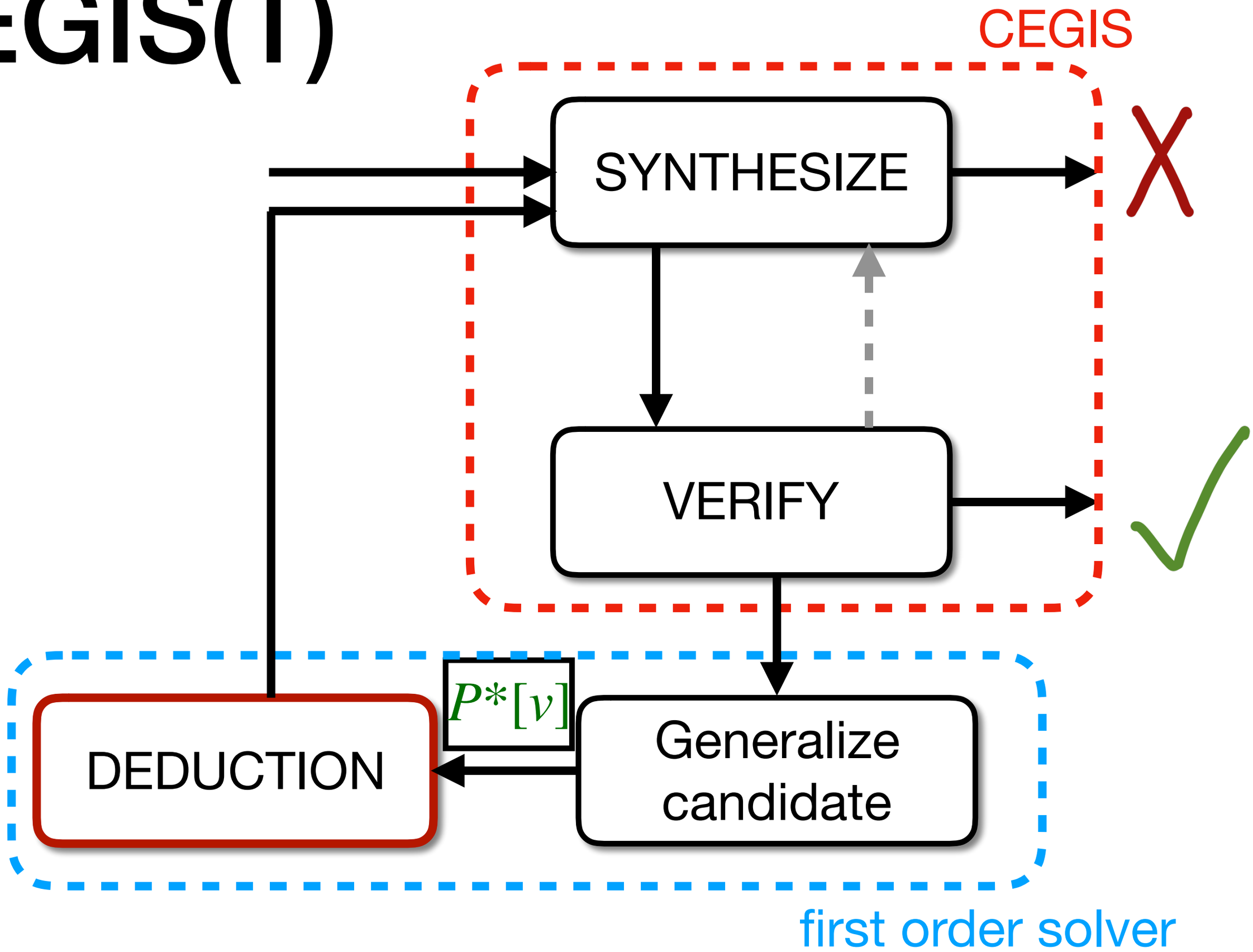
CEGIS(T)


CEGIS
SYNTHESIZE
X
VERIFY
✓
P*
Generalize candidate
DEDUCTION
first order solver

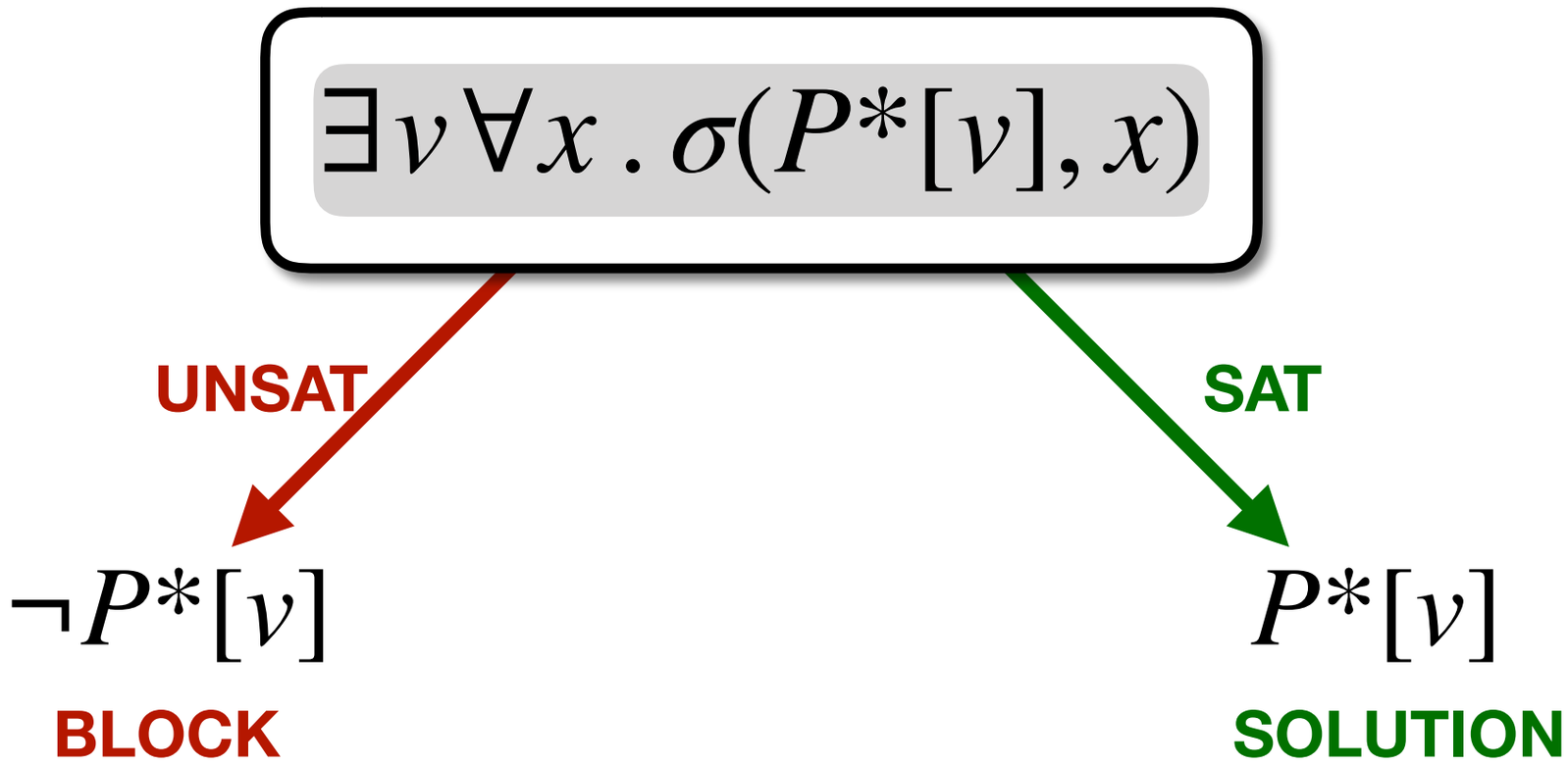
Generalize



CEGIS(T)

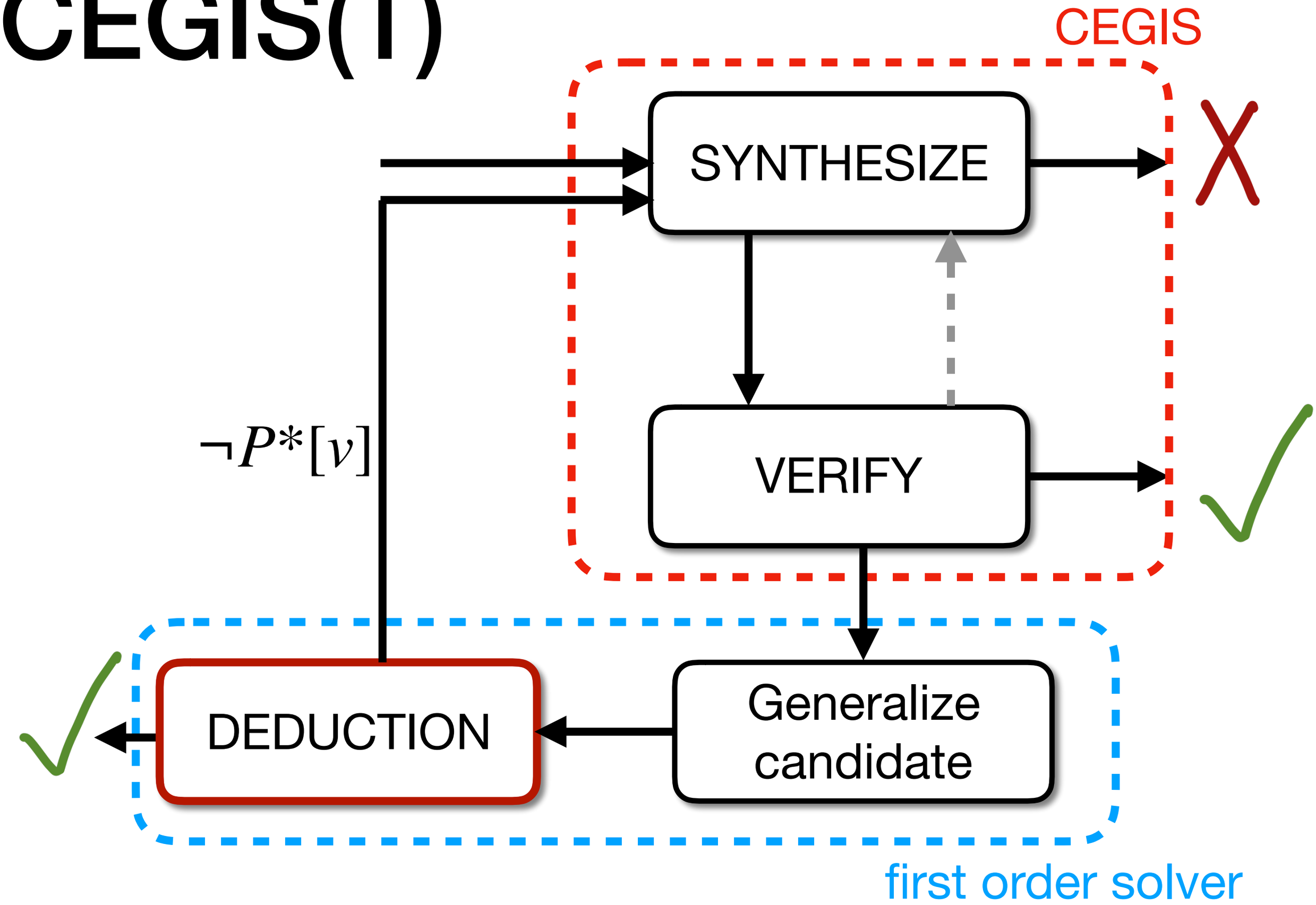

CEGIS
SYNTHESIZE
X
VERIFY
✓
DEDUCTION
P*[v]
Generalize candidate
first order solver

Deduction



is there a value for v that makes $(x < v)$ a valid invariant

CEGIS(T)

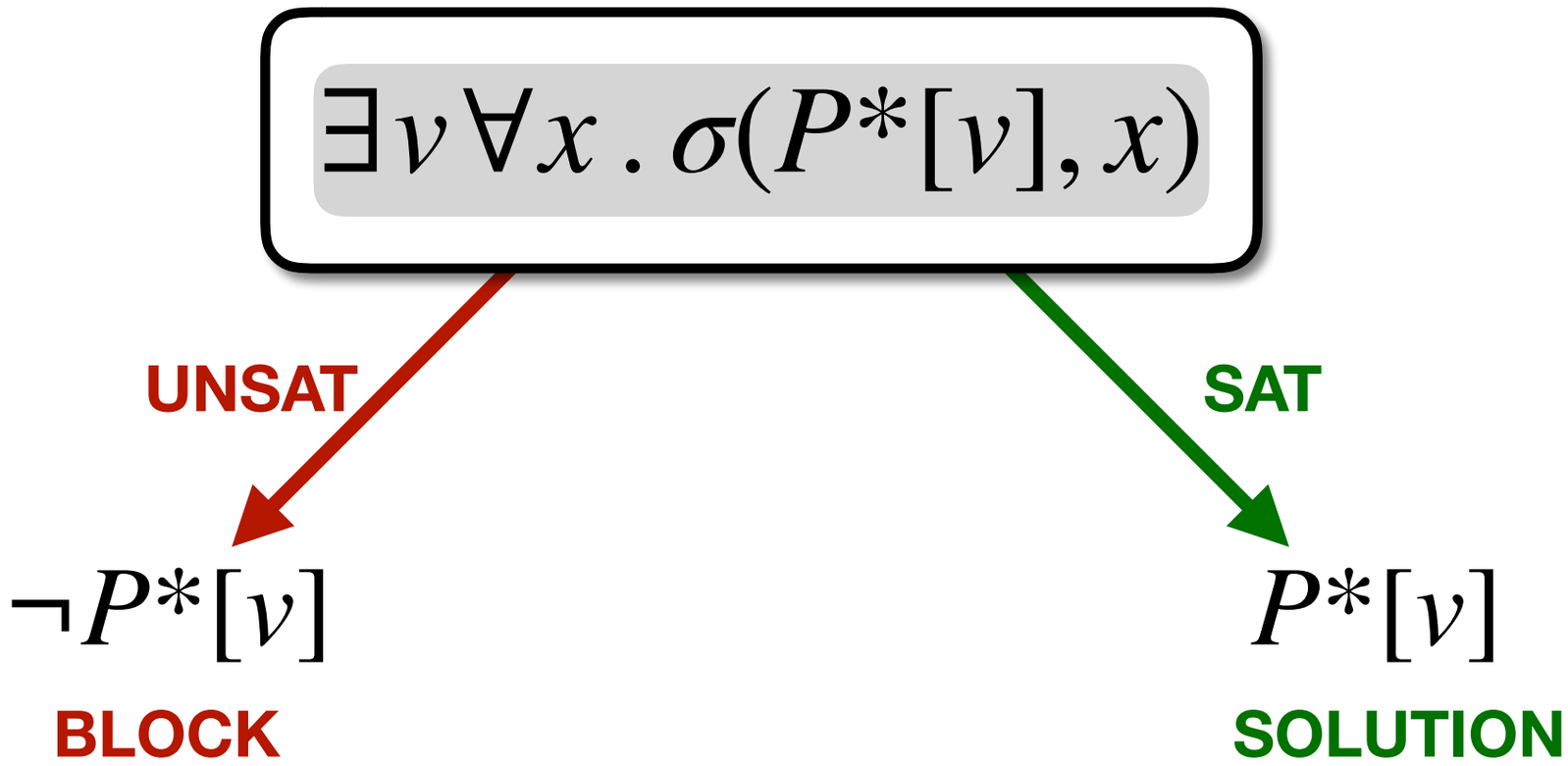

CEGIS
SYNTHESIZE
VERIFY
$\neg P^*[v]$
DEDUCTION
Generalize candidate
first order solver

First order solver

Solves 1st order formula with:

- Arbitrary propositional structure
- 1 quantifier alternation

CEGIS(T) - SMT


$\exists v \forall x \,.\, \sigma(P^*[v], x)$
UNSAT
SAT
$\neg P^*[v]$
BLOCK
$P^*[v]$
SOLUTION

CEGIS(T) - SMT

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < c)$$

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > c)$$

$\neg P^*[v]$

BLOCK

$v > c$

CONSTRAINT

$v < c$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < c)$$

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > c)$$

$$P^* = (x < 95)$$

$$P^*[v] = (x < v)$$

$\neg P^*[v]$
BLOCK

$v > c$
CONSTRAINT

$v < c$
CONSTRAINT

$P^*[v]$
SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < 95)$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > 95)$

$$P^* = (x < 95)$$

$$P^*[v] = (x < v)$$

$\neg P^*[v]$
BLOCK

$v > 95$
CONSTRAINT

$v < 95$
CONSTRAINT

$P^*[v]$
SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

UNSAT

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < 95)$

UNSAT

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > 95)$

$\neg P^*[v]$

BLOCK

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

UNSAT

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < 95)$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > 95)$

$\neg P^*[v]$

BLOCK

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v < 95)$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v > 95)$

UNSAT

$\neg P^*[v]$

BLOCK

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 < 95)$$

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 > 95)$$

$$P^* = (10 < x) \wedge (x < 95)$$

$$P^*[v] = (v_0 < x) \wedge (x < v_1)$$

$\neg P^*[v]$
BLOCK

$v > 95$
CONSTRAINT

$v < 95$
CONSTRAINT

$P^*[v]$
SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

UNSAT

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 < 95)$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 > 95)$

$\neg P^*[v]$

BLOCK

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

$$inv(x) = (4 < x) \wedge (x < 1003)$$

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 < 95)$

$\neg P^*[v]$

BLOCK

SAT

$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 > 95)$

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Target:

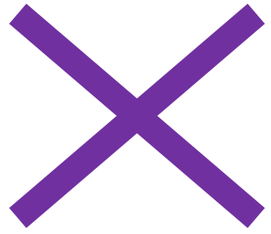
$$inv(x) = (4 < x) \wedge (x < 1003)$$

TIMEOUT

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 < 95)$$

TIMEOUT

$$\exists v \forall x \,.\, \sigma(P^*[v], x) \wedge (v_1 > 95)$$



$\neg P^*[v]$

BLOCK

$v > 95$

CONSTRAINT

$v < 95$

CONSTRAINT

$P^*[v]$

SOLUTION

Experiments

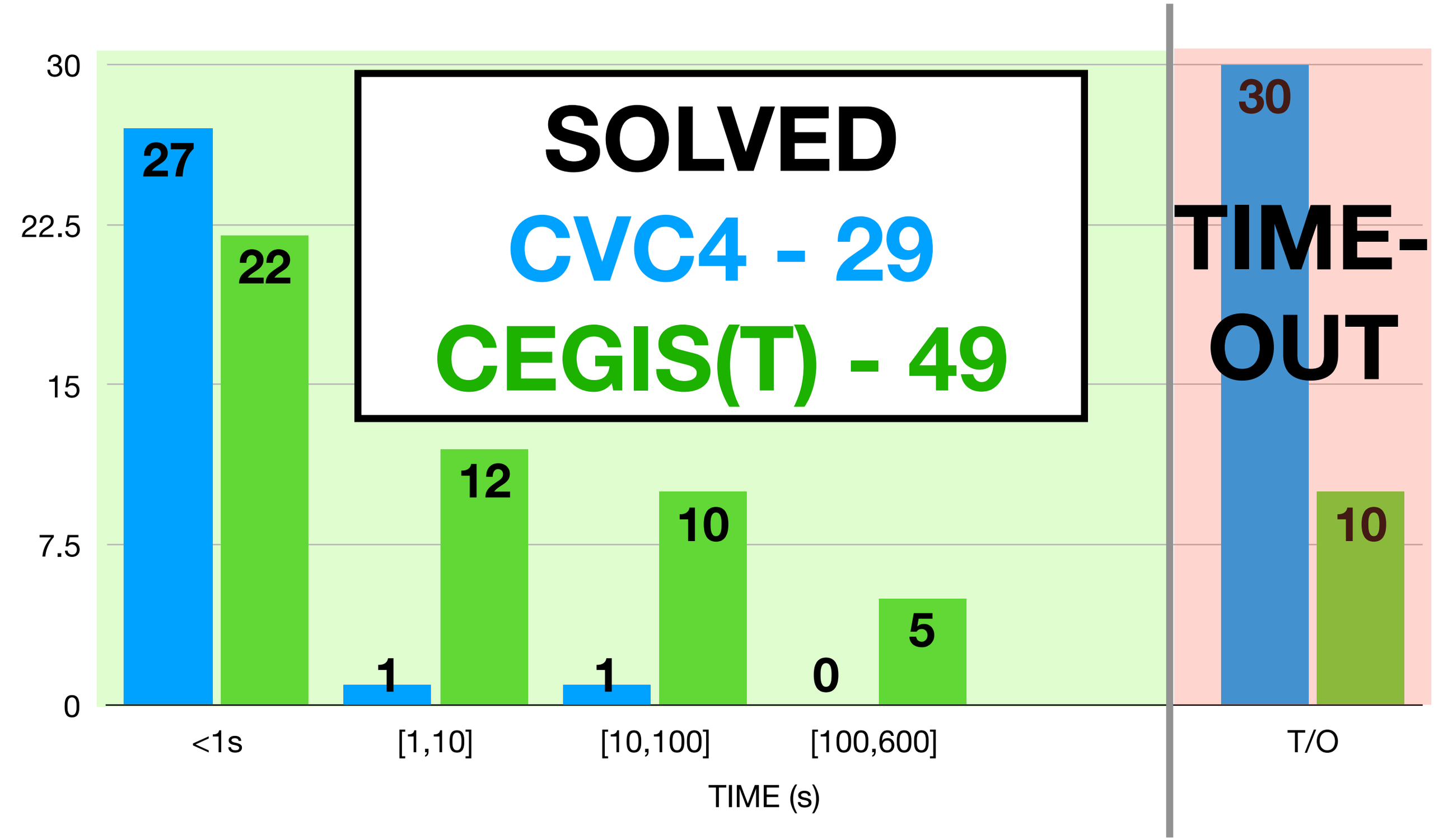
Benchmarks:

- Bitvectors
- Syntax-guided Synthesis competition (without the syntax)
- Loop invariants
- Danger invariants

Solvers:

- CVC4
- EUSolver, E3Solver, LoopInvGen – bitvectors with no grammar unsupported

Experiments



CEGIS(T)

CEGIS(T) solves program synthesis via 1st order solvers that support quantifiers:

- Enables use of existing solvers

Algorithmic insights:

- verify generalized candidate solutions
- return generalized counterexamples

www.cprover.org/synthesis

CEGIS(T) wants YOUR solvers

elizabeth.polgreen@cs.ox.ac.uk