Towards Smarter MACE-style Model Finders

Mikoláš Janota[1,2] Martin Suda[2]

[1] IST/INESC-ID, University of Lisbon, Portugal
[2] Czech Technical University in Prague

LPAR, 2018, Ethiopia

Does a FOL formula have a finite model?

Does a FOL formula have a finite model?

- Finite models useful:
  - “debugging” of wrong theorems
  - “debugging” of wrong programs
  - information for lemma selection learning

Background and Motivation

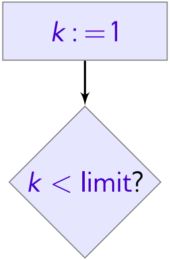
TÉCNICO LISBOA

Does a FOL formula have a finite model?

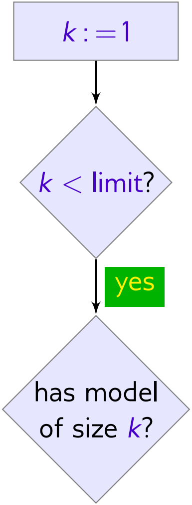
- Finite models useful:
  - "debugging" of wrong theorems
  - "debugging" of wrong programs
  - information for lemma selection learning

- **Advatage:**
  - If a finite model exists, it is fount in finite time.
  - Complete for some theories (Bernays-Schönfinkel, a.k.a. EPR)

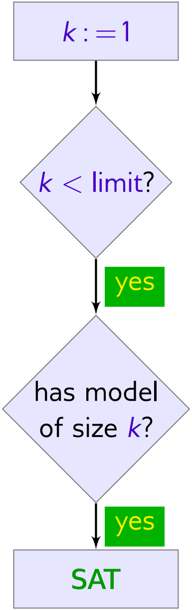
MACE-style Framework



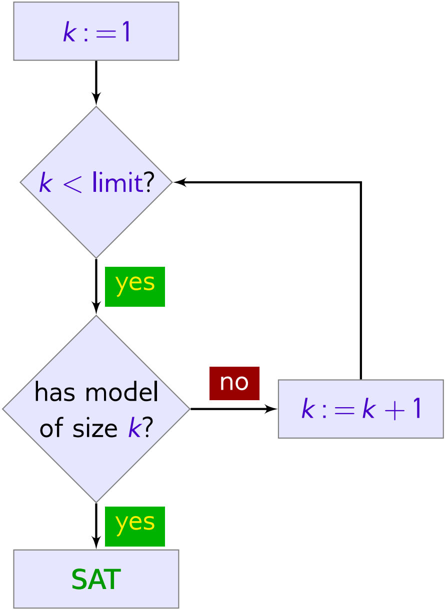
MACE-style Framework



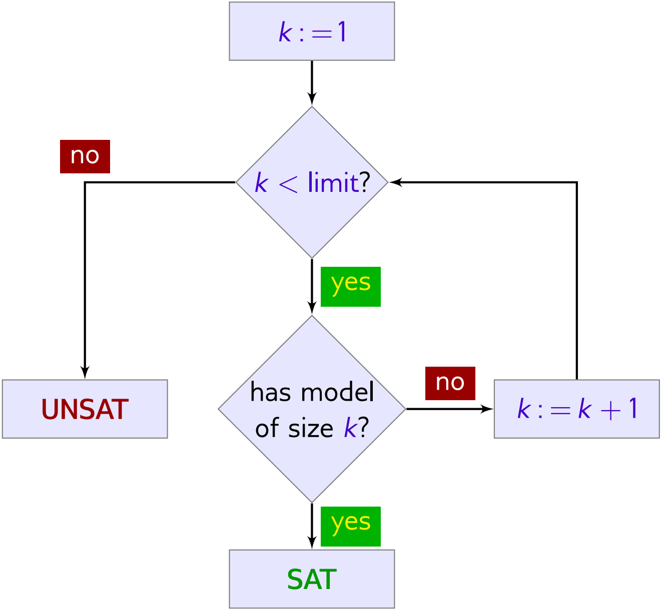
MACE-style Framework



MACE-style Framework



MACE-style Framework



Avoiding Space Explosion

Issue:
Encoding directly to SAT is exponential, eventually blows up

Avoiding Space Explosion

Issue:
Encoding directly to SAT is exponential, eventually blows up

Remedy:
Encode into SAT **lazily** by
Count-**E**example **A**bstraction **G**uided **R**efinement (**CEGAR**)

$$(\exists \vec{p}\,\vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$

CEGAR for Fixed Size k

$$(\exists \vec{p}\, \vec{f}\,)(\forall \vec{x}\,)\, \phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α

$$(\exists \vec{p}\,\vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α
3. If α UNSAT, **RETURN** false

$$(\exists \vec{p}\,\vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α
3. If α UNSAT, **RETURN** false
4. Find counterexample μ to $\mathcal{I}$ in original formula

$$(\exists \vec{p}\,\vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α
3. If α UNSAT, **RETURN** false
4. Find counterexample μ to $\mathcal{I}$ in original formula
5. If no counterexample, **RETURN** true

CEGAR for Fixed Size k

$$(\exists \vec{p}\, \vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α
3. If α UNSAT, **RETURN** false
4. Find counterexample μ to $\mathcal{I}$ in original formula
5. If no counterexample, **RETURN** true
6. Strengthen abstraction: $\alpha := \alpha \wedge \phi[\mu/\vec{x}]$

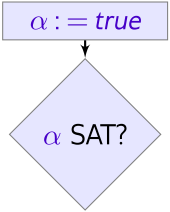
$$(\exists \vec{p}\,\vec{f}\,)(\forall \vec{x}\,)\,\phi$$

$\vec{p}$ predicates, $\vec{f}$ functions, $\vec{x}$ FOL variables

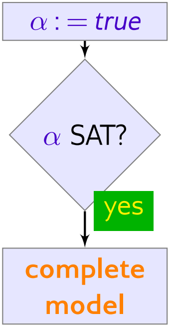
Algorithm sketch:

1. $\alpha := \mathit{true}$
2. Find model $\mathcal{I}$ for α
3. If α UNSAT, **RETURN** false
4. Find counterexample μ to $\mathcal{I}$ in original formula
5. If no counterexample, **RETURN** true
6. Strengthen abstraction: $\alpha := \alpha \wedge \phi[\mu/\vec{x}]$
7. **GOTO** 2

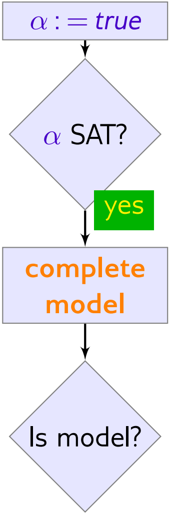
CEGAR for Fixed Size k



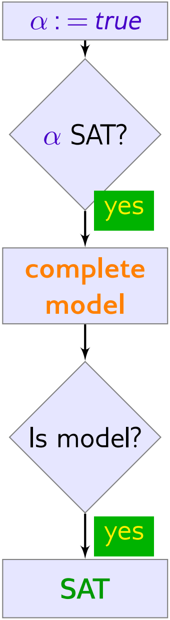
CEGAR for Fixed Size k



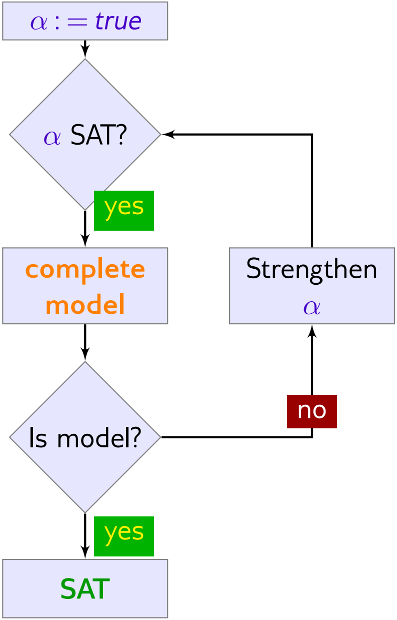
CEGAR for Fixed Size k

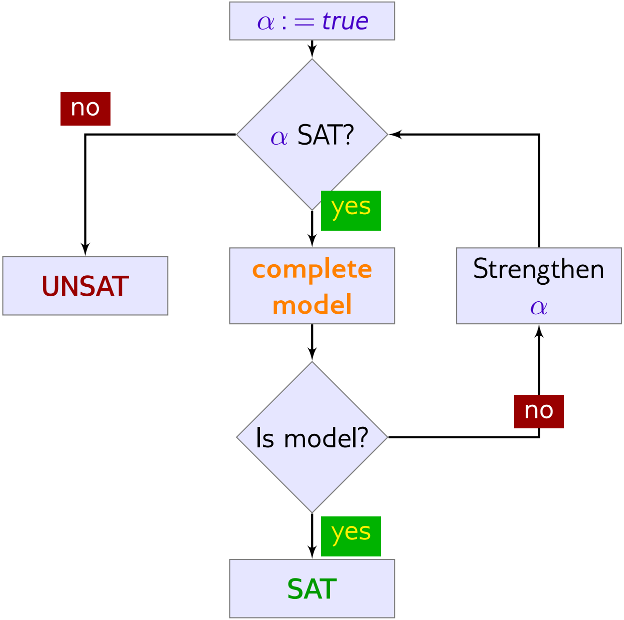


CEGAR for Fixed Size k



CEGAR for Fixed Size k




$\alpha := true$
α SAT?
no
UNSAT
yes
complete model
Strengthen α
Is model?
no
yes
SAT

We have model of ground α,

Example:

$$p(0), \neg q(0) \models p(0) \vee q(0)$$

Completing Models

We have model of ground α,
Example:

$$p(0), \neg q(0) \models p(0) \vee q(0)$$

We need to complete into **interpretation** of original
Examples for $(\forall x)(p(x) \vee \neg q(y))$

$$p \triangleq \{0\}, q \triangleq \{\}$$

$$p \triangleq \{0\}, q \triangleq \{1\}$$

$$p \triangleq 2^{1..k}, q \triangleq \{\}$$

Completing Models Contd.

Natural approach: set undefined to false/true

Examples

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq 2^{1..k} \setminus \{1, 2\}$$

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq \{p(0)\}$$

Completing Models Contd.

Natural approach: set undefined to false/true

Examples

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq 2^{1..k} \setminus \{1, 2\}$$

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq \{p(0)\}$$

Issue: completion uninformed

Completing Models Contd.

Natural approach: set undefined to false/true

Examples

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq 2^{1..k} \setminus \{1, 2\}$$

$$\{p(0), \neg p(1), \neg p(2)\} \;\ldots\; p \triangleq \{p(0)\}$$

Issue: completion uninformed

Remedy:
Learn the completion with Machine Learning techniques.

Learning in Finite Models' CEGAR, Example

1. $(\forall\vec{x})\ p(x_1, \ldots, x_n) \leftrightarrow (x_1 = t)$

1. $(\forall\vec{x})\ p(x_1, \ldots, x_n) \leftrightarrow (x_1 = t)$
2. Ground by $\{x_1 \mapsto 0, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(0, \ldots, 0) \leftrightarrow 0 = t$

Learning in Finite Models' CEGAR, Example

1. $(\forall \vec{x})\ p(x_1, \ldots, x_n) \leftrightarrow (x_1 = t)$
2. Ground by $\{x_1 \mapsto 0, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(0, \ldots, 0) \leftrightarrow 0 = t$
3. Ground by $\{x_1 \mapsto 1, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(1, \ldots, 0) \leftrightarrow 1 = t$

1. $(\forall\vec{x})\ p(x_1,\ldots,x_n) \leftrightarrow (x_1 = t)$
2. Ground by $\{x_1 \mapsto 0, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(0,\ldots,0) \leftrightarrow 0 = t$
3. Ground by $\{x_1 \mapsto 1, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(1,\ldots,0) \leftrightarrow 1 = t$
4. $\alpha = (p(0,\ldots,0) \leftrightarrow 0 = t) \wedge (p(1,\ldots,0) \leftrightarrow 1 = t)$

1. $(\forall \vec{x})\ p(x_1, \ldots, x_n) \leftrightarrow (x_1 = t)$
2. Ground by $\{x_1 \mapsto 0, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(0, \ldots, 0) \leftrightarrow 0 = t$
3. Ground by $\{x_1 \mapsto 1, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(1, \ldots, 0) \leftrightarrow 1 = t$
4. $\alpha = (p(0, \ldots, 0) \leftrightarrow 0 = t) \wedge (p(1, \ldots, 0) \leftrightarrow 1 = t)$
5. Model of α:
   $t \triangleq 1$
   $p(0, \ldots, 0) \triangleq \text{False}$
   $p(1, \ldots, 0) \triangleq \text{True}$

1. $(\forall\vec{x})\ p(x_1,\ldots,x_n) \leftrightarrow (x_1 = t)$
2. Ground by $\{x_1 \mapsto 0, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(0,\ldots,0) \leftrightarrow 0 = t$
3. Ground by $\{x_1 \mapsto 1, x_2 \mapsto 0, \ldots, x_n \mapsto 0\}$: $p(1,\ldots,0) \leftrightarrow 1 = t$
4. $\alpha = (p(0,\ldots,0) \leftrightarrow 0 = t) \wedge (p(1,\ldots,0) \leftrightarrow 1 = t)$
5. Model of α:
   $t \triangleq 1$
   $p(0,\ldots,0) \triangleq \text{False}$
   $p(1,\ldots,0) \triangleq \text{True}$
6. **Learn:**
   $t \triangleq 1$
   $p(x_1,\ldots,x_n) \triangleq (x_1 = 1)$

Technical Remarks (QFM)

- Learning by **decision trees**

Technical Remarks (QFM)

- Learning by **decision trees**
- Function and predicates eliminated by **Ackermann reduction**

Technical Remarks (QFM)

- Learning by **decision trees**
- Function and predicates eliminated by **Ackermann reduction**
- Finite domains encoded to SAT by **unary encoding**

Technical Remarks (QFM)

- Learning by **decision trees**
- Function and predicates eliminated by **Ackermann reduction**
- Finite domains encoded to SAT by **unary encoding**
- Incremental SAT (`minisat`)

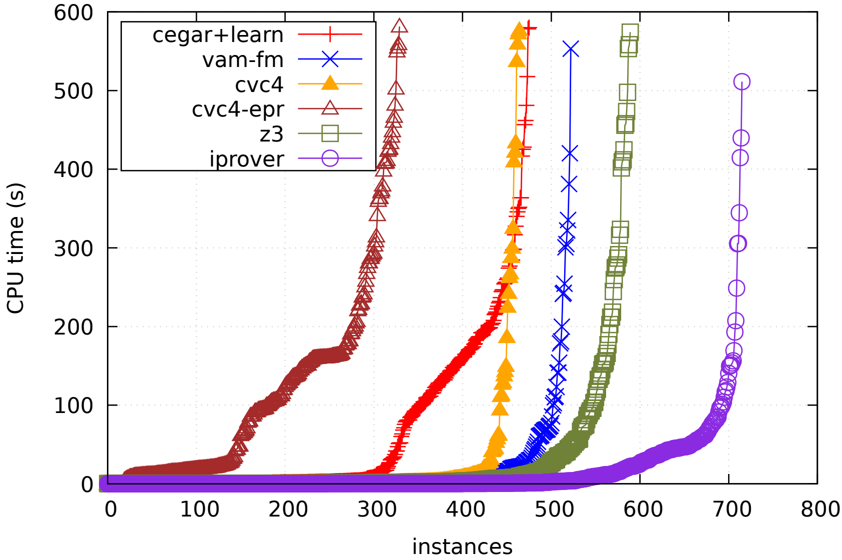
Technical Remarks (QFM)

- Learning by **decision trees**
- Function and predicates eliminated by **Ackermann reduction**
- Finite domains encoded to SAT by **unary encoding**
- Incremental SAT (`minisat`)
- Support for non-prenex

Technical Remarks (QFM)

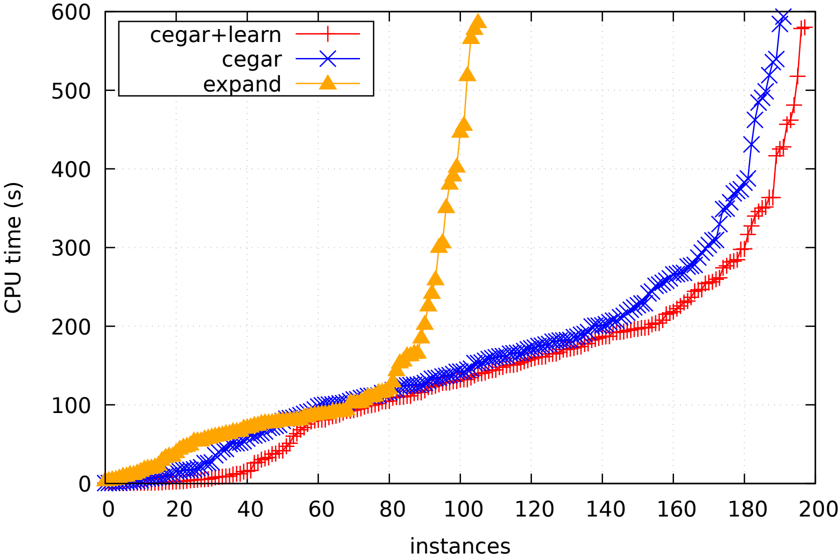
- Learning by **decision trees**
- Function and predicates eliminated by **Ackermann reduction**
- Finite domains encoded to SAT by **unary encoding**
- Incremental SAT (`minisat`)
- Support for non-prenex
- Symmetry breaking, e.g. $c_1 \triangleq 0$

Results EPR

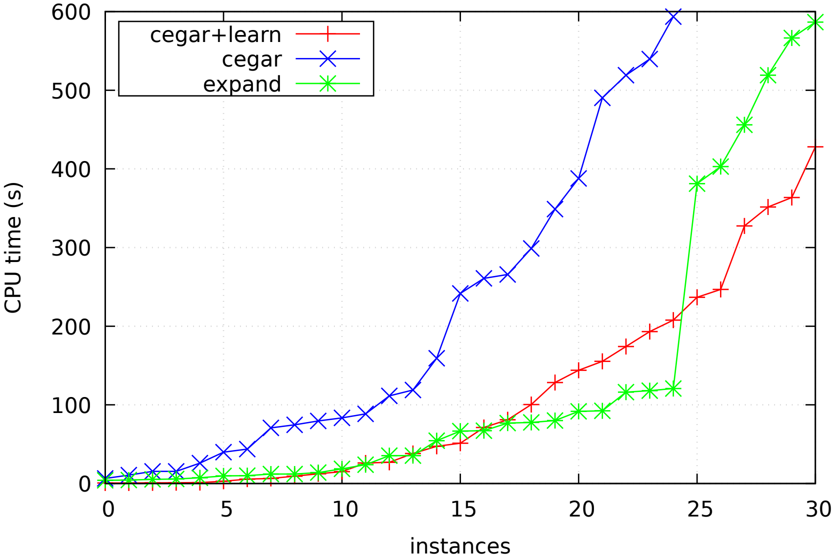


600
500
400
300
200
100
0
CPU time (s)
0
100
200
300
400
500
600
700
800
instances
cegar+learn
vam-fm
cvc4
cvc4-epr
z3
iprover

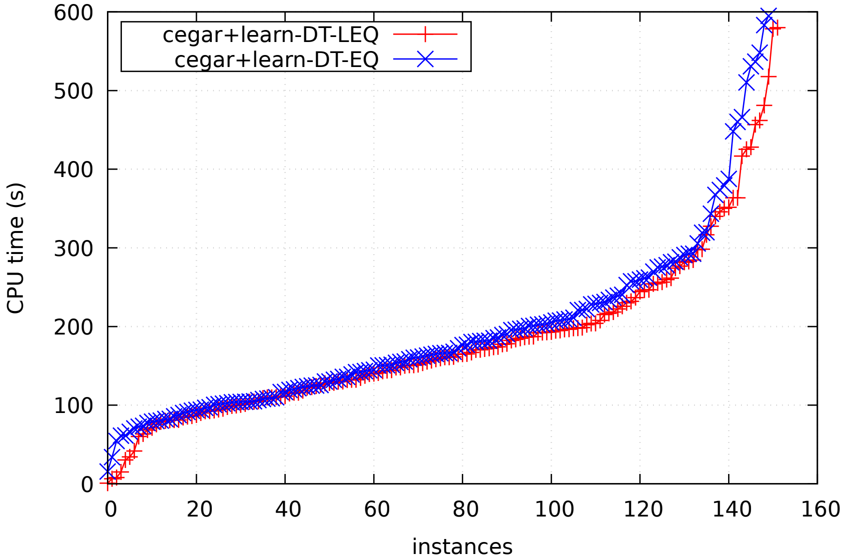
Results EPR: QFM



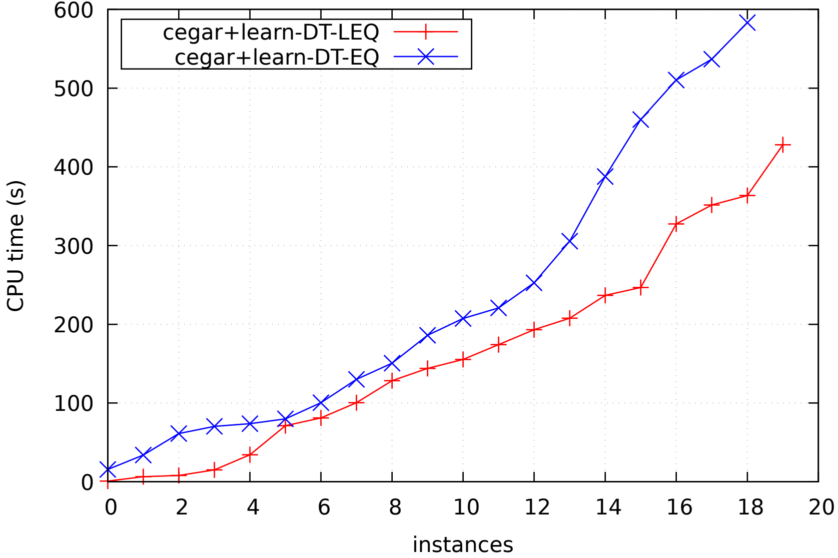
Results EPR QFM (SAT)



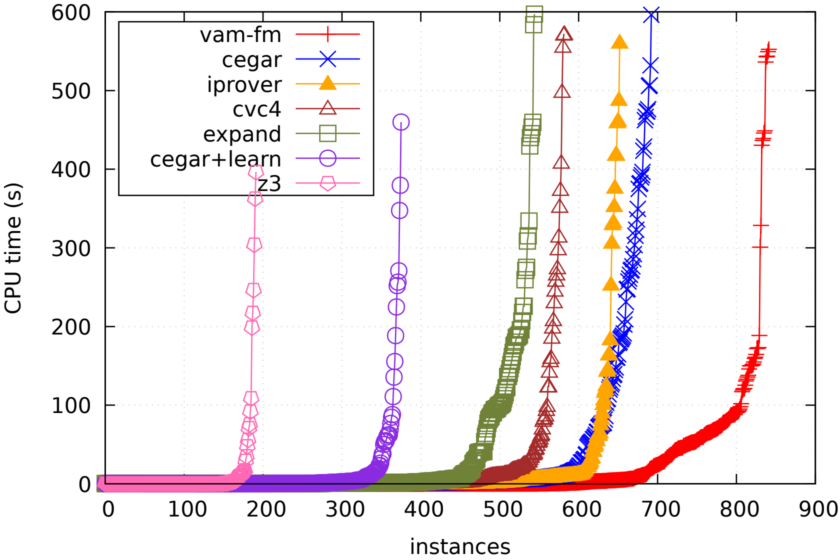
Results EPR: Learning Method



Results EPR: Learning Method (SAT)



Results SAT NON-EPR



Summary and Future

- CEGAR for lazy SAT-based model finite model finding

Summary and Future

- CEGAR for lazy SAT-based model finite model finding
- Observing a formula while solving, learn from that

Summary and Future

- CEGAR for lazy SAT-based model finite model finding
- Observing a formula while solving, learn from that
- Better learning methods?

Summary and Future

- CEGAR for lazy SAT-based model finite model finding
- Observing a formula while solving, learn from that
- Better learning methods?
- Learning in the presence of theories?

Summary and Future

- CEGAR for lazy SAT-based model finite model finding
- Observing a formula while solving, learn from that
- Better learning methods?
- Learning in the presence of theories?
- Infinite domains?

http://sat2019.tecnico.ulisboa.pt