Instantiation for Theory Reasoning in Vampire

Giles Reger   Martin Riener

Vampire Workshop 2018
July 13, Oxford, UK

School of Computer Science, University of Manchester, UK

Instantiation

Instantiation in a Nutshell

Consider the clause:

$$14x \neq x^2 + 49 \vee p(x)$$

Solving it via axioms is hard.

Suppose we guess $x = 7$:

Instantiation in a Nutshell

Consider the clause:

$$14x \neq x^2 + 49 \vee p(x)$$

Solving it via axioms is hard.

Suppose we guess $x = 7$:

$$14 \cdot 7 \neq 7^2 + 49 \vee p(7)$$

Instantiation in a Nutshell

Consider the clause:

$$14x \neq x^2 + 49 \vee p(x)$$

Solving it via axioms is hard.

Suppose we guess $x = 7$:

$$14 \cdot 7 \neq 7^2 + 49 \vee p(7)$$

$$\rightsquigarrow \quad \text{evaluate}$$

$$98 \neq 98 \vee p(7)$$

Instantiation in a Nutshell

Consider the clause:

$$14x \neq x^2 + 49 \vee p(x)$$

Solving it via axioms is hard.

Suppose we guess $x = 7$:

$$14 \cdot 7 \neq 7^2 + 49 \vee p(7)$$

$\rightsquigarrow$ evaluate

$$98 \neq 98 \vee p(7)$$

$\rightsquigarrow$ remove trivial inequality

$$p(7)$$

Instantiation

- Find instance that makes theory part of a clause false
- Substitute and delete theory part
- Rule

$$\frac{P \vee D}{D\theta} \text{ theory instance}$$

  - P pure (all constant symbols have a fixed interpretation)
  - $P\theta$ unsatisfiable in the theory

Instantiation

- Why pure?

Instantiation

- Why pure?
  $\Rightarrow$ We pass $\neg P$ to an SMT solver!

Instantiation

- Why pure?
  $\Rightarrow$ We pass $\neg P$ to an SMT solver!
- $\neg P$ has a model: construct θ from model
  - $14x = x^2 + 49$ has a model for $x = 7$
  - $\theta = \{x \mapsto 7\}$

Instantiation

- Why pure?
  $\Rightarrow$ We pass $\neg P$ to an SMT solver!
- $\neg P$ has a model: construct θ from model
  - $14x = x^2 + 49$ has a model for $x = 7$
  - $\theta = \{x \mapsto 7\}$
- Model construction needs purity (for now)

Abstraction

- Suppose we want to resolve

$$r(14y)$$

$$\neg r(x^2 + 49) \lor p(x)$$

$\Rightarrow$ No pure literals

Abstraction

- Suppose we want to resolve

$$r(14y)$$

$$\neg r(x^2 + 49) \lor p(x)$$

  $\Rightarrow$ No pure literals

- Abstract to

$$z \neq 14y \lor r(z)$$

$$u \neq x^2 + 49 \lor \neg r(u) \lor p(x)$$

Problems with Abstraction

- Eager application too expensive, fold into unification

Problems with Abstraction

- Eager application too expensive, fold into unification
- Instantiation undoes abstraction:

$$
\begin{array}{cl}
p(1,5) & \\
\rightsquigarrow & \text{abstract} \\
x \neq 1 \lor y \neq 5 \lor p(x,y) & \\
\rightsquigarrow & \text{instantiate} \\
p(1,5) &
\end{array}
$$

Trivial Literals

- Form: $x \neq t$ (x not in t)
- Pure
- x only occurs in other trivial literals or other non-pure literals

Updated Rule

$$\frac{P \vee D}{D\theta} \text{ theory instance}$$

- $P\theta$ unsatisfiable in the theory
- P pure
- P does not contain trivial literals

Improvements to Vampire

SMT-LIB

Logic	New solutions	Uniquely solved
ALIA	1	0
LIA	14	0
LRA	4	0
UFDTLIA	5	0
UFLIA	28	14
UFNIA	13	4

Ongoing Work

Theory Instantiation for Arrays

Axioms (universally closed):

- $select(store(A, I, E), I) = E$
- $I \neq J \rightarrow select(store(A, I, E), J) = select(A, J)$
- $A \neq B \rightarrow select(A, sk(A, B)) \neq select(B, sk(A, B)))$

Sorts:

$$\begin{array}{ll} A & : array(\alpha, \beta) \\ I, J & : \alpha \\ E & : \beta \\ select & : array(\alpha, \beta) * \alpha > \beta \\ store & : array(\alpha, \beta) * \alpha * \beta > array(\alpha, \beta) \end{array}$$

Theory Instantiation for Arrays

- Focus on: $array[int, int]$

Theory Instantiation for Arrays

- Focus on: $array[int, int]$
- Example clause:

$$\begin{aligned} & select(A, 0) \leqslant select(A, 1) \vee \\ & select(A, 1) \leqslant select(A, 2) \vee \\ & p(A) \end{aligned}$$

Theory Instantiation for Arrays

- Focus on: $array[int, int]$
- Example clause:

$$\begin{aligned}
&select(A, 0) \leqslant select(A, 1) \vee \\
&select(A, 1) \leqslant select(A, 2) \vee \\
&p(A)
\end{aligned}$$

- SMT Problem:
  $select(a, 0) > select(a, 1) \wedge select(a, 1) > select(a, 2)$

Theory Instantiation for Arrays

CVC4 model (term):

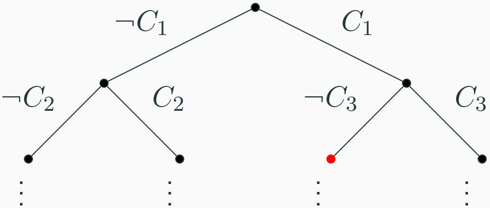
```
(define-fun a () (Array Int Int)
(store (store ((as const (Array Int Int)) 0) 0 1) 2 (- 1)))
```

Theory Instantiation for Arrays

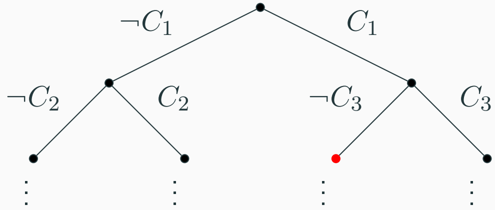
Z3 model (decision tree):

```
(define-fun a () (Array Int Int)
      (_ as-array k!0))
(define-fun k!0 ((x!0 Int)) Int
      (ite (= x!0 2) 7718
      (ite (= x!0 1) 7719
      (ite (= x!0 0) 7720
                    7718))))
```

Decision Tree



Decision Tree



Path to red node: $C_1 \land \neg C_3$

Translations for Decision Trees

- Conditions as guards:
  Infer multiple instances together:

$$X \neq 2 \lor select(A, X) \neq 7718 \lor p(A)$$
$$X = 2 \lor X \neq 1 \lor select(A, X) \neq 7719 \lor p(A)$$
$$X = 2 \lor X = 1 \lor X \neq 0 \lor select(A, X) \neq 7719 \lor p(A)$$

  (can be simplified here, not clear if possible in general)

Translations for Decision Trees

- Conditional + FOOL:

$$
\begin{aligned}
select(A, X) = \$ite(&X = 2, 7718, \\
&\$ite(X = 1, 7719, \\
&\quad \$ite(X = 0, 7720, 7718)))
\end{aligned}
$$

Translations for Decision Trees

- Conversion to term:
  Same as for CVC4, but trees like

$$\$ite(X < 0, 0,$$
$$\$ite(X < 100, 1, 0))$$

  become large.

Guarded Instantiation

- Add guard to rule:

$$\frac{P \vee D}{\neg G \vee D\theta} \text{ theory instance}$$

  - $G \wedge P\theta$ unsatisfiable in the theory
  - P pure
  - P does not contain trivial literals

Conclusion

Conclusion

Summary:

- Instantiation helps for arithmetic reasoning
- Arrays require refinement of the rule
- Guarded instantiation can be used to describe models

Future work:

- Evaluation of array model construction methods
- What about multiple / infinite solutions?
  e.g. extract solved linear equation system from Z3
- What about uninterpreted symbols?
  SMT problem now has universal quantifiers

Conclusion

Summary:

- Instantiation helps for arithmetic reasoning
- Arrays require refinement of the rule
- Guarded instantiation can be used to describe models

Future work:

- What about datatypes?
- Other ways to generalize the model?
  Unsat core, partial models etc.

Thanks!

Bonus Slides

Uninterpreted constants

- Consider the clause

$$c + X = 0 \vee p(X)$$

- Can be seen as skolemized form of

$$\exists C \forall Y. C + X = 0 \vee p(X)$$

- Pick $C + X = 0$ for theory instantation and negate
- We obtain $\forall C \exists Y. C + X \neq Y$
- After Skolemization, we look for a (finite) model of: $C + sk(C) = 0$

The issue with constarr(I)

- A series of store terms describes a finite number of mutations of an array
- $store(\cdots constarr(0)) = constarr(1)$ not solvable in pure theory of arrays
- Might generate lots of unsolvable problems

Partial Function

- Partial functions extended to total functions
- Consider the clause $(1-x) \cdot \frac{1}{(1-x)} \neq 0 \vee p(x)$:
  $(1-x) \cdot \frac{1}{(1-x)} = 0$ has a z3 model $x = 1$. We would infer $p(1)$.
- Can be seen as instantation guarded by $x \neq 1$.
  Instance removed by tautology elimination.