Certified Interpolant Generation for EUF

Andrew Reynolds
Cesare Tinelli
University of Iowa

Liana Hadarean
New York University

July 14, 2011
• Interpolation
  – Importance of certification
• Certified Interpolants via Proof Checking
  – Proof checker LFSC
  – Interpolant Generating Calculi
  – Generation of Interpolants via Type Inference
• Interpolation Calculus for EUF
• Results
• Conclusions and Future Work
• For theory T, a T-interpolant I for (A,B)
  
  (1) $A \models_T I$
  
  (2) $B, I \models_T \bot$

  (3) $L(I) \subseteq L(A) \cap L(B)$

• In some cases, may be efficiently generated from pf

• Applications
  – Model Checking
  – Predicate Abstraction
  – ...

• In some, correctness of interpolant is critical
• Interpolant over-approximates reachable states
• Alleged Interpolants that violate $B, I \models_T \bot$
  lead to spurious error states
• Alleged Interpolants that violate $A \models_T I$ may lead to unsoundness

Safe instead of Unsafe
• Clearly, correctness of interpolant is important
• SMT solvers produce interpolants
  – None do so in a verified way

• Goal: Certify interpolants via proof checker
  – Certification via Interpolating Calculi
  – Alternatively, may generate interpolants
    • Certified Correct by Construction
• LFSC: Meta-logical Framework (Stump ’08)
  – Proof + user-defined Signature
• Based on Edinburgh Logical Framework (LF)
• Extends LF with
  – Computational Side Conditions
  – Support for Integer, Rational arithmetic
• Proofs as Terms
  – Proof checking amounts to type checking
Proof Checking

Formula \( \varphi \) \n
Solver

\begin{align*}
\text{sat} & \rightarrow \text{Model} \\
\text{unsat} & \rightarrow \text{Proof} \ + \ \text{Signature}
\end{align*}

\begin{align*}
\text{Model} & \rightarrow \ldots \\
\text{Proof} & \rightarrow \text{LFSC Pf Checker} \\
\text{Signature} & \rightarrow \ldots
\end{align*}

pf valid \quad pf invalid
(check
  (\%  \ldots
  (\%  \ldots
  (\%  v  (proof  \varphi)
  (:  (proof  false)
    \(P\)
  ))  \ldots)

- LFSC proofs reside in check commands
- (:\textcolor{red}{T}s) - Check whether term s has type T
- Use of (proof \varphi) type for formula \varphi
- If success, we have certified \(\varphi \models \bot\)
• Since LFSC is meta-framework, we can extend signature to type-check proofs about interpolants
Proof Checking: Interpolants

Formula $A \land B$

Solver

Model

sat

unsat, interpolant $I$

Proof + Extended Signature

LFSC Pf Checker

pf valid

pf invalid
(check
  (% ... 
  (% u (proof A)
  (% v (proof B)
  (: (interpolant I) \textcolor{green}{(P)}
  )))...)

• Use of \texttt{(interpolant I)} type for formula \textit{I}

• If \textit{P} has type \texttt{(interpolant I)},
  – \textit{I} is a certified interpolant for \textit{(A, B)}
• SMT solver produces interpolant + proof

• LFSC verifies that proof:
  (1) Successfully type checks, and
  (2) Shows claimed interpolant is an interpolant.

• If success, we have a certified interpolant

• Solver + Checker must agree on the interpolant
• Alternatively:

*Use proof checker as the interpolant generator*

• Solver writes proof in same signature
  – Constructs term of type \((\text{interpolant } I)\),
    • for some value of \(I\), unknown a priori
  – Value of \(I\) computed by type inference
• LFSC terms may contain hole symbols “_”
• For example:

```
(trans _ _ _ (= t_1 t_2 ) (= t_2 t_3 ))
```

• Allow proof checker to fill in value of interpolant
  – Certified correct by construction
(check
  (% ...%
  (% u (proof A))
  (% v (proof B))
  (: (interpolant _) (P))))...

• The interpolant field is left unspecified “_”
• If P has type (interpolant I) for some I,
  – Value of I is given to user
  – I is a certified interpolant for (A, B)
Interpolant Generation via Proof Checking

Formula: $A \land B$

Solver

- sat (Model)
- unsat (Proof + Extended Signature)

LFSC Pf Checker

- pf valid,
- pf invalid

Interpolant $I$
• Interpolant generating calculi encoded in LFSC
• Augment rules with extra information
  – Encoding of partial interpolants $\varphi [\varphi_1 \varphi_2 c]$
    • where $[\varphi_1 \varphi_2 c]$ is annotation for $\varphi$
    • (p_interpolant $\varphi \varphi_1 \varphi_2 c$) type
• Tested LFSC framework for interpolants
• Examined theory of equality (EUF)
  – Simple calculus for interpolation in EUF
  – 203 lines of type declarations
  – 21 lines of side condition code
• Use CVC3 for proof generation

• Preliminary experiments on other theories
  – Boolean, QF_LRA, QFPA, ...
Interpolating Calculus for EUF

- Interpolating Calculus for EUF
  - Proposed by McMillan ’03
- Modified version of this calculus
  - Based on method given by Fuchs et. al. ’09
  - Simpler, flexibility in interpolants produced
- Extension of standard EUF proof calculus
  - Reflexivity, Symmetry, Transitivity, Congruence
  - Deduces only colorable equalities
• A term, literal, or formula is:
  • $A$-colorable ($B$-colorable) if:
    – Its free non-logical symbols contained in $L(A)$ ($L(B)$)
  • colorable if:
    – It is either $A$-colorable or $B$-colorable
  • $AB$-colorable if:
    – It is both $A$-colorable and $B$-colorable
• To produce certified interpolant:
  • Obtain standard proof of UNSAT from CVC3

  ![Diagram]

  - Proof is “lifted” to a proof with:
    - Only colorable equalities
    - Color annotations
  - Lifting process can be described by colored congruence graphs
• Proof lifting via Colored Congruence Graphs
  • Edges are assumptions or applications of congruence
  • Edges annotated with a color
• Edges between AB-colorable terms can be colored either A or B

\[
\begin{align*}
c_3 & \rightarrow f(c_1) \rightarrow a_1 \rightarrow c_2 \\
c_1 & \rightarrow b_2 \rightarrow g(c_3) \rightarrow g(c_2) \\
a_1 & \rightarrow f(c_1) \rightarrow f(g(c_2)) \rightarrow b_1
\end{align*}
\]
• Build equality chains of A-, B- colorable terms

\[ t_1 \underbrace{\cdots} \underbrace{t_2, \ t_1 \underbrace{\cdots} \underbrace{t_2, \ etc.}} \]

• Partial Interpolant of form \( t_1 \approx t_2 [\varphi, \psi, c] \)

where

1. \( A \models \varphi; \)
2. \( B, \varphi \models \psi; \)
3. \( A, \psi \models t_1 \approx t_2; \) and
4. \( t_i \) is \( c \)-colorable for \( i = 1, 2. \)
• Rules for A- and B- colored chains

\[ \frac{t_1 \approx t_2 \quad t_2 \approx t_3}{t_1 \approx t_3} \quad \text{trans} \]

\[ t_1 \approx t_2 \left[ \varphi_1, \psi_1, c \right] \]
\[ t_2 \approx t_3 \left[ \varphi_2, \psi_2, c' \right] \]
\[ \{ \text{t}_1, \text{t}_3 \text{ are A-colorable} \} \]
\[ t_1 \approx t_3 \left[ \varphi_1 \land \varphi_2, \psi_1 \land \psi_2, A \right] \]

\[ t_1 \approx t_2 \left[ \varphi_1, \psi_1, c \right] \]
\[ t_2 \approx t_3 \left[ \varphi_2, \psi_2, c' \right] \]
\[ \{ \text{t}_1, \text{t}_3 \text{ are B-colorable} \} \]
\[ t_1 \approx t_3 \left[ \varphi_1 \land \varphi_2 \land (\psi_1 \land \psi_2) \rightarrow (t_1 \approx t_3), t_1 \approx t_3, B \right] \]

\[ \text{t}_1 \quad \text{---} \quad \text{t}_3 \]
\[ \text{t}_1 \quad \text{---} \quad \text{t}_3 \]
\[
t_1 \approx t_2 [\varphi_1, \psi_1, c] \\
t_2 \approx t_3 [\varphi_2, \psi_2, c'] \\
\{ \text{t}_1, \text{t}_3 \text{ are A-colorable} \} \\
\frac{t_1 \approx t_3 [\varphi_1 \land \varphi_2, \psi_1 \land \psi_2, A]}{(trans-A)}
\]
• Advantages of Calculus
  – Flexibility (Fuchs et. al. ‘09)
    • Coloring between AB-colorable equalities
      – Logical strength
      – Interpolant size
  – Fewer Side Conditions
    • Only two side conditions (term colorability)
      – 21 lines of sc code
    • Can be implemented naturally in LF
• CVC3 for proof generation
• Tested on EUF theory lemmas
  – Extracted from SMT LIB
  – Unique, ≥ 5 edges in congruence graph
• Tested various partitions of (A,B)
  – k/6 in set A for k = 1...5
• Tested configurations
  – **euf**: proof checking
  – **eufi**: proof checking with interpolant generation
• Proof checking fast w.r.t to solving
  – **euf** 11x faster than solving
  – **eufi** 5x faster than solving
• Interpolants come at small overhead
  – **eufi** 22% overhead with respect to solving + pf generation
• Alternative: Verify interpolants directly

• For alleged interpolant $I$, prove:

\begin{align*}
(1) & \quad A \models_T I \\
(2) & \quad B, I \models_T \bot
\end{align*}

Note: AB-colorability can be easily verified
Offline vs Online Certification

\[ A \land B \]

\[ (A \land \neg I) \lor (B \land I) \]

CVC3

\[ \text{Proof, } I \]

\[ \text{Int Proof} \]

\[ \text{Unsat?} \]

\[ \text{Proof} \]

LFSC

\[ \text{Int } I \]

\[ \text{Proof valid?} \]

Online Approach

Offline Approach
• Certification faster via LFSC
  – $b1 + b2$ 56% of the time of $a + c$
• Proof Checking is faster than Solving
• Idea: generate multiple interpolants from same proof
  – Need only call solver once

\[ A \land B \]

Solver

Proof

Signature 1

Proof Checker

Signature n

Interpolant 1

Interpolant n
Conclusions

• Efficient method for certified interpolants
• Simple calculus for EUF interpolation
  – Coloring options
  – Few side conditions
• Flexibility of signature
  – Multiple interpolants from same proof
  – Certification of other properties
• Integration with CVC4
• Extension to other theories
  – Boolean + theory lemmas
• Use of new release of LFSC
  – Efficient generation of certified interpolants
• Applications of Interpolants
  – Use of LFSC framework for generation
Questions?