First-Order Interpolation in the Grey Area of Proofs

Laura Kovács

joint work with Andrei Voronkov (U. Manchester)
**Given:** a problem (*an interpolation problem*)

**Generate:** a formula (*an interpolant*)
Given: a problem (an interpolation problem)

Generate: a formula (an interpolant)

\[-1 + a + -a = -1 \land \\
\forall x(\neg(x \leq 5) \lor -6 + x \leq -1) \land \\
-(-1 + -1 + a) = -1 \land \\
\forall x((1 \leq x \lor -(1 + a) \lor -(-1 \leq x))) \land \\
(a \leq 6 \lor 1 \leq a - 1) \land \\
\forall x(\neg(-1 \leq x) \lor \neg(x \leq -2)) \land \\
\forall x(-1 \leq x + -a \lor -(1 + a \leq x)) \land \\
\forall x(-1 + x = 1 + -2 + x) \land \\
-a + -1 + a = -1 \land \\
\forall x(\neg(-(1 + a) \leq x) \lor 1 \leq x + -1) \land \\
\forall x((\neg(x \leq 4) \lor -5 + x \leq -1)) \land \\
\forall x(x + -3 \leq -1 \lor \neg(x \leq 2)) \land \\
\forall x(\neg(x \leq 3) \lor -4 + x \leq -1) \land \\
\forall x(x + -a \leq -1 \lor \neg(x \leq -1 + a)) \land \\
\forall x(\neg(-1 + x = -1 + -1 + a + -(1 + a) + x) \land \\
6 \leq b\]
Given: a problem (an interpolation problem)

Generate: a formula (an interpolant)

\[-1 + a + -a = -1 \land \]
\[\forall x (\neg (x \leq 5) \lor -6 + x \leq -1) \land \]
\[-(-1 + -1 + a) = -1 \land \]
\[\forall x ((1 \leq x + --(-1 + a) \lor \neg (-1 \leq x))) \land \]
\[(a \leq 6 \lor 1 \leq a + -1) \land \]
\[\forall x (\neg (-1 \leq x) \lor \neg (x \leq -2)) \land \]
\[\forall x (-1 \leq x + -a \lor \neg (-1 + a \leq x)) \land \]
\[\forall x (-1 + x = 1 + -2 + x) \land \]
\[-a + -1 + a = -1 \land \]
\[\forall x (\neg (-(1 + a) \leq x) \lor 1 \leq x + -1) \land \]
\[\forall x (\neg (x \leq 4) \lor -5 + x \leq -1) \land \]
\[\forall x (x + -3 \leq -1 \lor \neg (x \leq 2)) \land \]
\[\forall x (\neg (x \leq 3) \lor -4 + x \leq -1) \land \]
\[\forall x (x + -a \leq -1 \lor \neg (x \leq -1 + a)) \land \]
\[\forall x (-1 + x = -1 + -1 + a + -(-1 + a) + x) \land \]
\[6 \leq b \]

or

\[-(a \leq 6) \land \]
\[-a \leq -1 \land \]
\[\neg (-1 \leq -a) \land \]
\[a = 3 \land \]
\[1 \leq -1 + a \land \]
\[\neg (2 + a \leq 6) \land \]
\[\neg (-1 + a \leq 1) \land \]
\[(a \neq 6 \lor \neg (b \leq 6)) \]
Given: a problem (an interpolation problem)

Generate: a formula (an interpolant) which is small

\[-1 + a + -a = -1 \land
\forall x((-1 + x + -a) = -1 \land
\forall x((-1 + x) = 1 + -2 + x) \land
-a + -1 + a = -1 \land
\forall x((-1 + -a) \leq x) \lor 1 \leq x + -1) \land
\forall x((-5 + x) 
-a \leq -1 \land
\forall x(1 \leq 5) \lor -6 + x \leq -1) \land
-(-1 + -1 + a) = -1 \land
\forall x((-1 + a) \leq x) \lor 1 \leq x + -1) \land
6 \leq b\]
Interpolation

Small Interpolants

Quantifier Complexity of Interpolants

Conclusions
**Interpolation**

**Craig’s Interpolation Theorem**
Let $R, B$ be closed formulas and let $R \vdash B$.

Then there exists a formula $I$ such that
1. $R \vdash I$ and $I \vdash B$;
2. every symbol of $I$ occurs both in $R$ and $B$;
Interpolation

**Craig’s Interpolation Theorem**
Let \( R, B \) be closed formulas and let \( R \vdash B \).

Then there exists a formula \( I \) such that
1. \( R \vdash I \) and \( I \vdash B \);
2. every symbol of \( I \) occurs both in \( R \) and \( B \);

\( I \) is called an **interpolant** of \( R \) and \( B \).
Craig’s Interpolation Theorem
Let $R, B$ be closed formulas and let $R \vdash B$.

Then there exists a formula $I$ such that
1. $R \vdash I$ and $I \vdash B$;
2. every symbol of $I$ occurs both in $R$ and $B$;

$I$ is called an interpolant of $R$ and $B$.

Given an unsatisfiable set $\{R, B\}$.
A reverse interpolant $I$ of $R$ and $B$ is a formula such that:
1. $R \vdash I$ and $\{I, B\}$ is unsatisfiable;
2. every symbol of $I$ occurs both in $R$ and $B$. 


Interpolation Through Colors

- There are three colors: blue, red and grey.
Interpolation Through Colors

- There are three colors: blue, red and grey.
- Each symbol (function or predicate) is colored in exactly one of these colors.
Interpolation Through Colors

- There are three colors: blue, red and grey.
- Each symbol (function or predicate) is colored in exactly one of these colors.
- We have two formulas: $R$ and $B$.
- Each symbol in $R$ is either red or grey.
- Each symbol in $B$ is either blue or grey.
There are three colors: blue, red and grey.

Each symbol (function or predicate) is colored in exactly one of these colors.

We have two formulas: $R$ and $B$.

Each symbol in $R$ is either red or grey.

Each symbol in $B$ is either blue or grey.

We know that $\vdash R \rightarrow B$.

Task of interpolation: find a grey formula $I$ such that

1. $\vdash R \rightarrow I$;
2. $\vdash I \rightarrow B$. 

Interpolation in Applications

- bounded model checking;
- generating invariants or other program properties;
Interpolation in Applications

- bounded model checking;
- generating invariants or other program properties;

Steps of interpolation-based verification techniques:

- compute interpolants;
- prove program properties using interpolants;
Interpolation in Applications

- bounded model checking;
- generating invariants or other program properties;

Steps of interpolation-based verification techniques:

- **compute** interpolants;
- **prove** program properties using interpolants;

What is a good interpolant?

- logical strength [Jhala07, D’Silva09, McMillan08];
- small size [Kroening10, Brillout11, Griggio11];
Our Interest

Small Interpolants

- in size;
- in weight;
- in the number of quantifiers;
- ...

Given $\Gamma \vdash R \rightarrow B$, find a small grey formula $I$:

- $\Gamma \vdash R \rightarrow I$;
- $\vdash I \rightarrow B$;
- $I$ is small.
Our Interest

Small Interpolants

- in size;
- in weight;
- in the number of quantifiers;
- ... 

Given $\vdash R \rightarrow B$, find a grey formula $I$:

1. $\vdash R \rightarrow I$;
2. $\vdash I \rightarrow B$;
3. $I$ is small.
Local proofs: No inference mixes blue and red symbols
Local proofs: No inference mixes blue and red symbols

- $R := \forall x (x = a)$
- $B := c = b$
Local proofs: No inference mixes blue and red symbols

- \( R := \forall x (x = a) \)
- \( B := c = b \)
Extracting Interpolants from Local Proofs

Local proofs: No inference mixes blue and red symbols

- $R := \forall x (x = a)$
- $B := c = b$

Non-local proof

\[
\begin{align*}
\frac{x = a}{c = a} & \quad \frac{x = a}{b = a} \\
\hline
& \quad c = b & \quad c \neq b \\
\hline
\hline
\bot
\end{align*}
\]
Extracting Interpolants from Local Proofs

Local proofs: No inference mixes blue and red symbols

- R := ∀x(x = a)
- B := c = b

Non-local proof

Local Proof

\[
\begin{align*}
\frac{x = a}{c = a} & \quad \frac{x = a}{b = a} \\
\frac{c = b}{c \neq b} & \quad \frac{x = y}{c \neq b} \\
\frac{y \neq b}{\bot} & \quad \frac{y \neq b}{\bot}
\end{align*}
\]
Extracting Interpolants from Local Proofs

\[ \text{Interpolant: boolean combination of } \{ G_1, \ldots, G_4 \} \]

[McMillan05, KV09]
Extracting Interpolants from Local Proofs

Interpolant: boolean combination of \( \{ G_1, \ldots, G_4 \} \)

[McMillan05, KV09]
Extracting Interpolants from Local Proofs

Interpolant: boolean combination of \( \{G_1, \ldots, G_4\} \)

Digest
Extracting Interpolants from Local Proofs

G is in the digest:
- comes from a red block
- followed by a blue or grey block

Interpolant: boolean combination of \{G_1, \ldots, G_4\}

[McMillan05, KV09]
Extracting Interpolants from Local Proofs

G is in the digest:
- comes from a red block
- followed by a blue or grey block
or
- comes from a blue block
- followed by a red

Interpolant: boolean combination of \( \{G_1, \ldots, G_4\} \)

\[ \text{Digest} \]
Summary of Our Contributions

Contribution 1: Localizing proofs

Contribution 2: Small interpolants

Contribution 3: Quantifier complexity of interpolants
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

\[ \text{Given } R(a) \vdash B \text{ where } a \text{ is an uninterpreted constant not occurring in } B. \]

Then, \[ R(a) \vdash (\exists x) R(x) \] and \[ (\exists x) R(x) \vdash B. \]
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

Idea: quantify away colored symbols

\[
\downarrow
\]

colored symbols replaced by logical variables.
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

Idea: quantify away colored symbols

↓

colored symbols replaced by logical variables.

Given $R(a) \vdash B$ where $a$ is an uninterpreted constant not occurring in $B$.

Then, $R(a) \vdash (\exists x)R(x)$ and $(\exists x)R(x) \vdash B$. 
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

Idea: quantify away colored symbols

\[ \downarrow \]

colored symbols replaced by logical variables.

Given \( R(a) \vdash B \) where \( a \) is an uninterpreted constant not occurring in \( B \).

Then, \( R(a) \vdash (\exists x)R(x) \) and \( (\exists x)R(x) \vdash B \).
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

Idea: quantify away colored symbols

↓

colored symbols replaced by logical variables.

Cons: Comes at the cost of using extra quantifiers.

Given $R(a) \vdash B$ where $a$ is an uninterpreted constant not occurring in $B$.

Then, $R(a) \vdash (\exists x)R(x)$ and $(\exists x)R(x) \vdash B$. 

\[
\begin{array}{c}
\frac{R_1(a)}{R_2(a)} \frac{B}{A}
\end{array}
\quad \quad \quad 
\begin{array}{c}
\frac{R_1(a)}{(\exists x)R_2(x)} \frac{B}{A}
\end{array}
\]
Contrib. 1: Localizing Proofs

Task: eliminate non-local inferences

Idea: quantify away colored symbols

↓

colored symbols replaced by logical variables.

Cons: Comes at the cost of using extra quantifiers.
But we can minimise the number of quantifiers in the interpolant.

Given \( R(a) \vdash B \) where \( a \) is an uninterpreted constant not occurring in \( B \).

Then, \( R(a) \vdash (\exists x)R(x) \) and \( (\exists x)R(x) \vdash B \).
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest

Hercule Poirot:

*It is the little GREY CELLS, mon ami, on which one must rely.*

*Mon Dieu, mon ami, but use your little GREY CELLS!*
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest

Idea: Change the grey areas of the local proof
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest

Idea: Change the grey areas of the local proof

Slicing off formulas

\[
\begin{array}{c}
A_1 \ldots A_n \\
\overbrace{A_{n+1} \ldots A_m}^A \\
\hline
A_0
\end{array}
\quad \rightarrow
\quad
\begin{array}{c}
A_1 \ldots A_n \\
\overbrace{A_{n+1} \ldots A_m}^A \\
\hline
A_0
\end{array}
\]

If \( A \) is grey: Grey slicing
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest

Idea: Change the grey areas of the local proof

Slicing off formulas

\[ A_1 \cdots A_n \frac{A_{n+1} \cdots A_m}{A} \frac{A_0}{\text{slicing off } A} \]

If \( A \) is grey: Grey slicing
Contrib. 2: Small Interpolants

Task: minimise interpolants = minimise digest

Idea: Change the grey areas of the local proof

Slicing off formulas

\[
\begin{array}{c}
B_0 & \overline{G_0} & R_0 & \overline{G_1} \\
\hline
\end{array}
\]

If \( A \) is grey: Grey slicing
**Contrib. 2: Small Interpolants**

Task: minimise interpolants = minimise digest

Idea: Change the grey areas of the local proof, but preserve locality!

Slicing off formulas

\[
\begin{align*}
B_0 & \quad \frac{R_0}{G_1} \\
G_0 & \\
\end{align*}
\]

slicing off \( G_1 \)

\[
\begin{align*}
B_0 & \quad R_0 \\
G_0 & \\
\end{align*}
\]
Contrib. 2: Small Interpolants

\[ \frac{R_1}{G_3} \mid \frac{G_1}{G_4} \quad \frac{B_1}{G_7} \rightarrow \frac{G_5}{G_6} \]

\[ \frac{R_3}{G_7} \quad \frac{R_4}{G_6} \]

Reverse interpolant:

Note that the interpolant has changed from \( G_4 \rightarrow G_7 \) to \( \neg G_6 \).

There is no obvious logical relation between \( G_4 \rightarrow G_7 \) and \( \neg G_6 \), for example none of these formulas implies the other one; these formulas may even have no common atoms or no common symbols.
Contrib. 2: Small Interpolants

\[
\frac{R_1}{G_3} \quad \frac{B_1}{G_4} \\
\frac{G_1}{G_3} \quad \frac{G_2}{G_4} \\
\frac{G_5}{G_6} \\
\frac{R_3}{G_5} \\
\frac{G_7}{G_6} \\
\frac{R_4}{G_7} \\
\frac{G_7}{G_7}
\]

Digest: \{G_4, G_7\}

Reverse interpolant: \(G_4 \rightarrow G_7\)
Contrib. 2: Small Interpolants

\[ \frac{R_1 \; G_1 \; B_1 \; G_2}{G_3} \quad \frac{G_5}{G_6} \]

\[ \frac{R_3}{G_5} \quad \frac{G_6}{G_7} \]

\[ \frac{R_4}{\perp} \]

Digest:

Reverse interpolant:

Note that the interpolant has changed from \( G_4 \rightarrow G_7 \) to \( \neg G_6 \).

\[ \rightarrow \]

There is no obvious logical relation between \( G_4 \rightarrow G_7 \) and \( \neg G_6 \), for example none of these formulas implies the other one;

These formulas may even have no common atoms or no common symbols.
Contrib. 2: Small Interpolants

\begin{align*}
\frac{R_1}{G_1} & \quad \frac{B_1}{G_2} \\
\frac{}{G_3} & \quad \frac{}{G_5} \\
\frac{R_3}{G_5} & \quad \frac{G_6}{G_7} \\
\frac{R_4}{G_7} & \quad \frac{}{G_7}
\end{align*}

Digest: \{G_5, G_7\}

Reverse interpolant: \(G_5 \rightarrow G_7\)
Contrib. 2: Small Interpolants

\[
\frac{R_1}{G_3} \quad \frac{G_1}{B_1} \quad \frac{G_2}{G_3}
\]

\[
R_3 \quad \overline{G_6}
\]

\[
R_4 \quad \frac{G_6}{G_7} \quad \perp
\]

Digest:
Reverse interpolant:

Note that the interpolant has changed from $G_4 \rightarrow G_7$ to $\neg G_6$.

There is no obvious logical relation between $G_4 \rightarrow G_7$ and $\neg G_6$, for example none of these formulas implies the other one; these formulas may even have no common atoms or no common symbols.
Contrib. 2: Small Interpolants

\[ \frac{R_1}{G_3} \quad \frac{G_1}{B_1} \quad \frac{G_2}{G_3} \]

\[ \frac{R_3}{G_6} \]

\[ \frac{R_4}{G_7} \]

\[ \neg G_6 \]

Digest: \{G_6, G_7\}

Reverse interpolant: \( G_6 \rightarrow G_7 \)
Contrib. 2: Small Interpolants

\[
\frac{R_1 \ G_1 \quad B_1 \ G_2}{G_3}
\]

\[
\frac{R_3}{G_6}
\]

\[
\frac{R_4}{\bot}
\]

Digest:
Reverse interpolant:
Note that the interpolant has changed from \(G_4 \rightarrow G_7\) to \(\neg G_6\).

▶ There is no obvious logical relation between \(G_4 \rightarrow G_7\) and \(\neg G_6\), for example none of these formulas implies the other one;
▶ These formulas may even have no common atoms or no common symbols.
Contrib. 2: Small Interpolants

\[ \frac{R_1}{G_1} \quad \frac{B_1}{G_2} \]
\[ \frac{R_3}{G_6} \]
\[ \frac{R_4}{\perp} \]

Digest: \( \{G_6\} \)

Reverse interpolant: \( \neg G_6 \)
Contrib. 2: Small Interpolants

Note that the interpolant has changed from $G_4 \rightarrow G_7$ to $\neg G_6$. 
Contrib. 2: Small Interpolants

Note that the interpolant has changed from $G_4 \rightarrow G_7$ to $\neg G_6$.

- There is no obvious logical relation between $G_4 \rightarrow G_7$ and $\neg G_6$, for example none of these formulas implies the other one;
- These formulas may even have no common atoms or no common symbols.
Contrib. 2: Small Interpolants

If grey slicing gives us very different interpolants, we can use it for finding small interpolants.
If grey slicing gives us very different interpolants, we can use it for finding small interpolants.

**Problem:** if the proof contains $n$ grey formulas, the number of possible different slicing off transformations is $2^n$. 
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT
Contrib. 2: Small Interpolants

Solution:
- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
\hline
G_1 & G_2 \\
\hline
G_3 \\
\end{array}
\]
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{c c c c}
R & B \\
G_1 & G_2 \\
G_3
\end{array}
\]

\(G_3\), and at most one of \(G_1, G_2\) can be sliced off.
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{c|c|c}
R & B \\
G_1 & G_2 \\
G_3 & \\
\end{array}
\]

Some predicates on grey formulas:

- \text{sliced}(G): G was sliced off;
- \text{red}(G): the trace of G contains a red formula;
- \text{blue}(G): the trace of G contains a blue formula;
- \text{grey}(G): the trace of G contains only grey formulas;
- \text{digest}(G): G belongs to the digest.
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
G_1 & G_2 \\
\hline
G_3
\end{array}
\]

\(\neg\text{sliced}(G_1) \rightarrow \text{grey}(G_1)\)

\(\text{sliced}(G_1) \rightarrow \text{red}(G_1)\)

Some predicates on grey formulas:

- \text{sliced}(G): G was sliced off;
- \text{red}(G): the trace of G contains a red formula;
- \text{blue}(G): the trace of G contains a blue formula;
- \text{grey}(G): the trace of G contains only grey formulas;
- \text{digest}(G): G belongs to the digest.
Contrib. 2: Small Interpolants

Solution:
- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{c|c|c}
R & B \\
\hline
G_1 & G_2 \\
\hline
G_3 \\
\end{array}
\]

Some predicates on grey formulas:
- \text{sliced}(G): G was sliced off;
- \text{red}(G): the trace of G contains a red formula;
- \text{blue}(G): the trace of G contains a blue formula;
- \text{grey}(G): the trace of G contains only grey formulas;
- \text{digest}(G): G belongs to the digest.

\neg \text{sliced}(G_3) \rightarrow \text{grey}(G_3)
\text{sliced}(G_3) \rightarrow (\text{grey}(G_3) \leftrightarrow \text{grey}(G_1) \land \text{grey}(G_2))
\text{sliced}(G_3) \rightarrow (\text{red}(G_3) \leftrightarrow \text{red}(G_1) \lor \text{red}(G_2))
\text{sliced}(G_3) \rightarrow (\text{blue}(G_3) \leftrightarrow \text{blue}(G_1) \lor \text{blue}(G_2))
Contrib. 2: Small Interpolants

Solution:
- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{c|c}
R & B \\
G_1 & G_2 \\
G_3 & \\
\end{array}
\]

Some predicates on grey formulas:
- \( \text{sliced}(G) \): \( G \) was sliced off;
- \( \text{red}(G) \): the trace of \( G \) contains a red formula;
- \( \text{blue}(G) \): the trace of \( G \) contains a blue formula;
- \( \text{grey}(G) \): the trace of \( G \) contains only grey formulas;
- \( \text{digest}(G) \): \( G \) belongs to the digest.

\( \text{digest}(G_1) \rightarrow \neg \text{sliced}(G_1) \)
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
G_1 & G_2 \\
\hline
G_3
\end{array}
\]

Some predicates on grey formulas:

- \(\text{sliced}(G)\): \(G\) was sliced off;
- \(\text{red}(G)\): the trace of \(G\) contains a red formula;
- \(\text{blue}(G)\): the trace of \(G\) contains a blue formula;
- \(\text{grey}(G)\): the trace of \(G\) contains only grey formulas;
- \(\text{digest}(G)\): \(G\) belongs to the digest.

\(-\text{sliced}(G_1) \rightarrow \text{grey}(G_1)\)
\(\text{sliced}(G_1) \rightarrow \text{red}(G_1)\)
\(-\text{sliced}(G_3) \rightarrow \text{grey}(G_3)\)
\(\text{sliced}(G_3) \rightarrow (\text{grey}(G_3) \leftrightarrow \text{grey}(G_1) \land \text{grey}(G_2))\)
\(\text{sliced}(G_3) \rightarrow (\text{red}(G_3) \leftrightarrow \text{red}(G_1) \lor \text{red}(G_2))\)
\(\text{sliced}(G_3) \rightarrow (\text{blue}(G_3) \leftrightarrow \text{blue}(G_1) \lor \text{blue}(G_2))\)
\(\text{digest}(G_1) \rightarrow \neg\text{sliced}(G_1)\)
\[
\cdots
\]
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{ccc}
R & B \\
\hline
G_1 & G_2 \\
\hline
G_3
\end{array}
\]

Express \text{digest}(G)
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
G_1 & G_2 \\
G_3 &
\end{array}
\]

Express \text{digest}(G)

by considering the possibilities:

- \(G\) comes from a
  - red/ blue/ grey formula

- \(G\) is followed by a
  - red/ blue/ grey formula
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
G_1 & G_2 \\
G_3 & \\
\end{array}
\]

Express \text{digest}(G)

by considering the possibilities:

- \(G\) comes from a
  - \text{red}/ \text{blue}/ \text{grey} formula

\[rc(G)/bc(G)\]

- \(G\) is followed by a
  - \text{red}/ \text{blue}/ \text{grey} formula

\[bf(G)/rf(G)\]
Contrib. 2: Small Interpolants

Solution:
▶ encode all sequences of transformations as an instance of SAT

\[
\begin{array}{cc}
R & B \\
G_1 & G_2 \\
\hline \\
G_3
\end{array}
\]

Express \text{digest}(G)

by considering the possibilities:

▶ \(G\) comes from a
red/ blue/ grey formula
\(rc(G)/bc(G)\)

▶ \(G\) is followed by a
red/ blue/ grey formula
\(bf(G)/rf(G)\)

\[
digest(G_3) \leftrightarrow (rc(G_3) \land rf(G_3)) \lor (bc(G_3) \land bf(G_3))
\]

\[
rc(G_3) \leftrightarrow (\neg \text{sliced}(G_3) \land (\text{red}(G_1) \lor \text{red}(G_2)))
\]
## Contrib. 2: Small Interpolants

**Solution:**

- encode all sequences of transformations as an instance of SAT

<table>
<thead>
<tr>
<th>R</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>G₁</td>
<td>G₂</td>
</tr>
<tr>
<td>G₃</td>
<td></td>
</tr>
</tbody>
</table>

Express \( \text{digest}(G) \) by considering the possibilities:

- \( G \) comes from a red/ blue/ grey formula
  
  \( \text{rc}(G)/\text{bc}(G) \)

- \( G \) is followed by a red/ blue/ grey formula
  
  \( \text{bf}(G)/\text{rf}(G) \)

\[ \neg \text{sliced}(G₁) \rightarrow \text{grey}(G₁) \]
\[ \text{sliced}(G₁) \rightarrow \text{red}(G₁) \]
\[ \neg \text{sliced}(G₃) \rightarrow \text{grey}(G₃) \]
\[ \text{sliced}(G₃) \rightarrow (\text{grey}(G₃) \leftrightarrow \text{grey}(G₁) \land \text{grey}(G₂)) \]
\[ \text{sliced}(G₃) \rightarrow (\text{red}(G₃) \leftrightarrow \text{red}(G₁) \lor \text{red}(G₂)) \]
\[ \text{sliced}(G₃) \rightarrow (\text{blue}(G₃) \leftrightarrow \text{blue}(G₁) \lor \text{blue}(G₂)) \]
\[ \text{digest}(G₁) \rightarrow \neg \text{sliced}(G₁) \]

\[ \text{digest}(G₃) \leftarrow (\text{rc}(G₃) \land \text{rf}(G₃)) \lor (\text{bc}(G₃) \land \text{bf}(G₃)) \]
\[ \text{rc}(G₃) \leftarrow (\neg \text{sliced}(G₃) \land (\text{red}(G₁) \lor \text{red}(G₂))) \]
\[ \ldots \]
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT
- solutions encode all slicing off transformations

\[
\begin{array}{ccc}
R & B \\
G_1 & G_2 & G_3
\end{array}
\]

Express \( \text{digest}(G) \) by considering the possibilities:

- \( G \) comes from a red/ blue/ grey formula
  \( \text{rc}(G)/\text{bc}(G) \)

- \( G \) is followed by a red/ blue/ grey formula
  \( \text{bf}(G)/\text{rf}(G) \)

\[
\neg \text{sliced}(G_1) \rightarrow \text{grey}(G_1) \\
\text{sliced}(G_1) \rightarrow \text{red}(G_1) \\
\neg \text{sliced}(G_3) \rightarrow \text{grey}(G_3) \\
\text{sliced}(G_3) \rightarrow (\text{grey}(G_3) \leftrightarrow \text{grey}(G_1) \land \text{grey}(G_2)) \\
\text{sliced}(G_3) \rightarrow (\text{red}(G_3) \leftrightarrow \text{red}(G_1) \lor \text{red}(G_2)) \\
\text{sliced}(G_3) \rightarrow (\text{blue}(G_3) \leftrightarrow \text{blue}(G_1) \lor \text{blue}(G_2)) \\
\text{digest}(G_1) \rightarrow \neg \text{sliced}(G_1) \\
\text{digest}(G_3) \leftrightarrow (\text{rc}(G_3) \land \text{rf}(G_3)) \lor (\text{bc}(G_3) \land \text{bf}(G_3)) \\
\text{rc}(G_3) \leftrightarrow (\neg \text{sliced}(G_3) \land (\text{red}(G_1) \lor \text{red}(G_2))) \\
\ldots
\]
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT;
- solutions encode all slicing off transformations;
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT;
- solutions encode all slicing off transformations;
- compute small interpolants: smallest digest of grey formulas;

\[
\min \{G_{i_1}, \ldots, G_{i_n}\} \left( \sum_{G_i} \text{digest}(G_i) \right)
\]
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT;
- solutions encode all slicing off transformations;
- compute small interpolants: smallest digest of grey formulas;

\[
\min\{G_{i_1},...,G_{i_n}\} \left( \sum_{G_i} \text{digest}(G_i) \right)
\]

\[
\min\{G_{i_1},...,G_{i_n}\} \left( \sum_{G_i} \text{quantifier\_number}(G_i) \cdot \text{digest}(G_i) \right)
\]

- use a pseudo-boolean optimisation tool or an SMT solver to minimise interpolants;
- minimising interpolants is an NP-complete problem.
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT;
- solutions encode all slicing off transformations;
- compute small interpolants: smallest digest of grey formulas;

\[
\min \{G_{i_1}, ..., G_{i_n}\} \left( \sum_{G_i} \text{digest}(G_i) \right)
\]

\[
\min \{G_{i_1}, ..., G_{i_n}\} \left( \sum_{G_i} \text{quantifier number}(G_i) \text{ digest}(G_i) \right)
\]

- use a pseudo-boolean optimisation tool or an SMT solver to minimise interpolants;
Contrib. 2: Small Interpolants

Solution:

- encode all sequences of transformations as an instance of SAT;
- solutions encode all slicing off transformations;
- compute small interpolants: smallest digest of grey formulas;

\[
\min\{G_{i_1}, \ldots, G_{i_n}\} \left( \sum_{G_i} \text{digest}(G_i) \right)
\]

\[
\min\{G_{i_1}, \ldots, G_{i_n}\} \left( \sum_{G_i} \text{quantifier\_number}(G_i) \text{digest}(G_i) \right)
\]

- use a pseudo-boolean optimisation tool or an SMT solver to minimise interpolants;
- minimising interpolants is an NP-complete problem.
Contrib. 1 & 2: Experiments with Small Interpolants

- Implemented in **Vampire** theorem prover;
- We used Yices for solving pseudo-boolean constraints;

Experimental results:
- 9632 first-order examples from the TPTP library: for example, for 2000 problems the size of the interpolants became 20-49 times smaller;
- 4347 SMT examples:
  - We used Z3 for proving SMT examples;
  - Z3 proofs were localised in Vampire;
  - small interpolants were generated for 2123 SMT examples.
Contrib. 1 & 2: Experiments with Small Interpolants

- Implemented in Vampire theorem prover;
- We used Yices for solving pseudo-boolean constraints;

**Experimental results:**

- 9632 first-order examples from the TPTP library:
  for example, for 2000 problems the size of the interpolants became 20-49 times smaller.
Contrib. 1 & 2: Experiments with Small Interpolants

- Implemented in *Vampire* theorem prover;

- We used Yices for solving pseudo-boolean constraints;

- **Experimental results:**
  
  - 9632 first-order examples from the TPTP library: for example, for 2000 problems the size of the interpolants became 20-49 times smaller;

- 4347 SMT examples:
  
  - we used Z3 for proving SMT examples;
  - Z3 proofs were localised in Vampire;
  - small interpolants were generated for 2123 SMT examples.
Contrib. 3: Quantifier Complexity of Interpolants

Local Proofs Do Not Always Exist

- \( R: (\forall x)p(r, x) \)
- \( B: (\forall y)\neg p(y, b) \)
- Reverse interpolant \( I \) of \( R \) and \( B \): \( (\exists y)(\forall x)p(y, x) \).
Local Proofs Do Not Always Exist

- **R**: $(\forall x) p(r, x)$
- **B**: $(\forall y) \neg p(y, b)$
- Reverse interpolant $I$ of $R$ and $B$: $(\exists y)(\forall x) p(y, x)$.
- **Note**: $R$ and $B$ contain no quantifier alternations, yet $I$ contains quantifier alternations. One can prove that every interpolant of this formula must have at least one quantifier alternation.
Contrib. 3: Quantifier Complexity of Interpolants

Local Proofs Do Not Always Exist

- $R$: $(\forall x)p(r, x)$
- $B$: $(\forall y)\neg p(y, b)$
- Reverse interpolant $I$ of $R$ and $B$: $(\exists y)(\forall x)p(y, x)$.

- Note: $R$ and $B$ contain no quantifier alternations, yet $I$ contains quantifier alternations. One can prove that every interpolant of this formula must have at least one quantifier alternation.

- There is no local refutation of $R, B$ in the resolution/superposition calculus.
- There is a non-local one:

$$
\begin{array}{c}
p(r, x) \\
\neg p(y, b) \\
\hline \\
\bot
\end{array}
$$
Theorem There is no lower bound on the number of quantifier alternations in interpolants of universal sentences.

That is, for every positive integer $n$ there exist universal sentences $R, B$ such that $\{R, B\}$ is unsatisfiable and every reverse interpolant of $R$ and $B$ has at least $n$ quantifier alternations.
Contrib. 3: Quantifier Complexity of Interpolants

Example

Take the formula $A: \forall x_1 \exists y_1 \forall x_1 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$ and let $R$ be obtained by skolemizing $A$ and $B$ be obtained by skolemizing $\neg A$:

$$R = \forall x_1 \forall x_2 \ldots p(x_1, r_1(x_1), x_2, r_2(x_1, x_2), \ldots)$$

$$B = \forall y_1 \forall y_2 \ldots \neg p(b_1, y_1, b_2(y_1), y_2, \ldots)$$

$$I = \forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$$

There is no reverse interpolant with a smaller number of quantifier alternations. The resolution refutation consists of a single step deriving the empty clause from $R$ and $B$. 
Example

Take the formula $A: \forall x_1 \exists y_1 \forall x_1 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$ and let $R$ be obtained by skolemizing $A$ and $B$ be obtained by skolemizing $\neg A$:

$$R = \forall x_1 \forall x_2 \ldots p(x_1, r_1(x_1), x_2, r_2(x_1, x_2), \ldots)$$

$$B = \forall y_1 \forall y_2 \ldots \neg p(b_1, y_1, b_2(y_1), y_2, \ldots)$$

$$I = \forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$$

There is no reverse interpolant with a smaller number of quantifier alternations.
Example

Take the formula $A: \forall x_1 \exists y_1 \forall x_1 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$ and let $R$ be obtained by skolemizing $A$ and $B$ be obtained by skolemizing $\neg A$:

$$R = \forall x_1 \forall x_2 \ldots p(x_1, r_1(x_1), x_2, r_2(x_1, x_2), \ldots)$$

$$B = \forall y_1 \forall y_2 \ldots \neg p(b_1, y_1, b_2(y_1), y_2, \ldots)$$

$$I = \forall x_1 \exists y_1 \forall x_2 \exists y_2 \ldots p(x_1, y_1, x_2, y_2, \ldots)$$

There is no reverse interpolant with a smaller number of quantifier alternations.

The resolution refutation consists of a single step deriving the empty clause from $R$ and $B$. 
Bad News for Local Proof Systems

Let $S$ be an inference system with the following property: for every red formula $R$ and blue formula $B$, if \{ $R$, $B$ \} is unsatisfiable, then there is a local refutation of $R$, $B$ in $S$.

Then the number of quantifier alternations in refutations of universal formulas of $S$ is not bound by any positive integer.
Contrib. 3: Quantifier Complexity of Interpolants

- There is **no bound on the number of quantifier alternations** in reverse interpolants of universal formulas.
There is no bound on the number of quantifier alternations in reverse interpolants of universal formulas.

Any complete local proof system for first-order predicate logic must have inferences dealing with formulas of an arbitrary quantifier complexity, even if the input formulas have no quantifier alternations.
Contrib. 3: Quantifier Complexity of Interpolants

- There is no bound on the number of quantifier alternations in reverse interpolants of universal formulas.

- Any complete local proof system for first-order predicate logic must have inferences dealing with formulas of an arbitrary quantifier complexity, even if the input formulas have no quantifier alternations.

- There is no simple modification of the superposition calculus for logic with/without equality in which every unsatisfiable formula has a local refutation.
Conclusions

- We localise proofs by quantifying away colored constants;

- We build small interpolants by:
  - expressing constraints on grey formulas;
  - finding a minimal interpolants as a solution to the constraint system;

- There is no lower bound on the number of quantifier alternations in interpolants of universal sentences.

No simple modification of the superposition calculus that is complete for local proofs.