Efficient Interpolant Generation in Satisfiability Modulo Linear Integer Arithmetic

Alberto Griggio
FBK-IRST, Trento

joint work with Thi Thieu Hoa Le and Roberto Sebastiani, DISI - Univ. Trento
Introduction

- **(Craig) Interpolation** for ground first-order theories successfully applied in formal verification
- **Efficient** SMT-based algorithms for several theories and combinations (e.g. EUF, LA(Q), DL, UTVPI)
- **Interpolation for full LA(Z) is harder**
  - Some promising recent work [Brillout et al IJCAR'10, Kroening et al. LPAR'10], but still some drawbacks

- **This work**: propose a novel, general technique for interpolation in LA(Z)
  - to overcome some drawbacks of current approaches
Outline

♦ Background

♦ Current techniques for interpolation in LA(Z)

♦ A novel interpolation technique for LA(Z)

♦ Experimental evaluation
(Craig) Interpolant for an ordered pair $(A, B)$ of formulas s.t.

\[ A \land B \models_{\mathcal{T}} \perp \] is a formula $I$ s.t.

a) $A \models_{\mathcal{T}} I$

b) $B \land I \models_{\mathcal{T}} \perp$

c) all the uninterpreted (in $\mathcal{T}$) symbols of $I$ occur in both $A$ and $B$
Background - Interpolants

- Interpolants can be generated from proofs of unsatisfiability [McMillan]
Interpolants can be generated from proofs of unsatisfiability [McMillan]

Proof of unsatisfiability in SMT:

- **Boolean part** (ground resolution) + **\( \mathcal{T} \)-specific part (for conjunctions of constraints)**
Interpolants can be generated from proofs of unsatisfiability [McMillan]

Proof of unsatisfiability in SMT:

- Boolean part (ground resolution)
- $\mathcal{T}$-specific part (for conjunctions of constraints)

$\mathcal{T}$-specific interpolation for conjunctions only
Interpolants can be generated from proofs of unsatisfiability [McMillan]

Proof of unsatisfiability in SMT:

- Boolean part (ground resolution)
- $\mathcal{T}$-specific part (for conjunctions of constraints)

Standard Boolean interpolation

Problem reduced to finding an interpolant for sets of $\mathcal{T}$-literals
Outline

- Background
- Current techniques for interpolation in LA(Z)
- A novel interpolation technique for LA(Z)
- Experimental evaluation
Interpolation and LA(Z)

- **Linear Integer Arithmetic**: constraints of the form
  \[ \sum_i c_i x_i + c \triangleright 0, \quad \triangleright \in \{\leq, =\} \]

- In general, no quantifier-free interpolation for LA(Z)!
  [McMillan05]

Example:

\[ A := (y - 2x = 0) \quad B := (y - 2z - 1 = 0) \]

The only interpolant is: \( \exists w. (y = 2w) \)

- **Solution**: extend the signature to include modular equations (divisibility predicates)
  \[ (t + c =_d 0) \equiv \exists w. (t + c = d \cdot w), \quad d \in \mathbb{Z}^>0 \]

The interpolant now becomes: \( (y =_2 0) \)
SMT(LA(Z)) with modular equations

- Modular equations can be eliminated via preprocessing:
  - Replace every atom $a := (t + c =_d 0)$ with a fresh Boolean variable $p_a$

- Add the 4 clauses
  $$ p_a \rightarrow (t + c - dw_1 = 0) $$
  $$ \neg p_a \rightarrow (t + c - dw_1 - w_2 = 0) $$
  $$ (-w_2 + 1 \leq 0) $$
  $$ (w_2 - d + 1 \leq 0) $$

  where $w_1$, $w_2$ are fresh integer variables
Using modular equation, interpolants can be constructed via quantifier elimination:

\[ I(A, B) := \text{ExistElim}(x_i \notin B)(A) \]

However, this is very expensive, both in theory and in practice.
Interpolants from LA(Z)-proofs

- **Cutting-plane proof system**: complete proof system for LA(Z)

\[
\begin{align*}
\text{Hyp} & : \quad t \leq 0 \\
\text{Comb} & : \quad \frac{t_1 \leq 0}{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0}, \quad c_1, c_2 > 0 \\
\text{Div} & : \quad \frac{\sum_i c_i x_i + c \leq 0}{\sum_i \frac{c_i}{d} x_i + \left\lceil \frac{c}{d} \right\rceil \leq 0}, \quad d > 0 \text{ divides the } c_i \text{'s}
\end{align*}
\]
Interpolants from LA(Z)-proofs

- Cutting-plane proof system: complete proof system for LA(Z)

**Hyp** \( \frac{t \leq 0}{\text{Comb}} \)

\[
\frac{t_1 \leq 0}{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0}, c_1, c_2 > 0
\]

**Div** \( \frac{\sum_i c_i x_i + c \leq 0}{\sum_i \frac{c_i}{d} x_i + \left\lfloor \frac{c}{d} \right\rfloor \leq 0}, d > 0 \) divides the \( c_i \)'s

LA(Q) rules
Cutting-plane proof system: complete proof system for LA(Z)

\[
\text{Hyp: } t \leq 0
\]

\[
\text{Comb: } \frac{t_1 \leq 0}{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0}, c_1, c_2 > 0
\]

\[
\text{Strengthen: } \frac{\sum_i c_i x_i + c \leq 0}{\sum_i c_i x_i + d \cdot \left\lceil \frac{c}{d} \right\rceil \leq 0}, d > 0 \text{ divides the } c_i \text{'s}
\]
Interpolants from LA(Z)-proofs

♦ Cutting-plane proof system: complete proof system for LA(Z)

\[
\begin{align*}
\text{Hyp: } & \quad t \leq 0 \\
\text{Comb: } & \quad \frac{t_1 \leq 0}{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0}, \quad c_1, c_2 > 0
\end{align*}
\]

Strengthen:
\[
\frac{\sum_i c_i x_i + c \leq 0}{\sum_i c_i x_i + d \cdot \left\lfloor \frac{c}{d} \right\rfloor \leq 0}, \quad d > 0 \text{ divides the } c_i \text{'s}
\]

♦ Interpolation by annotating proof rules [McMillan05, Brillout et al. IJCAR'10]

♦ Annotation (in this talk): a set of pairs \( \{ \langle t_i \leq 0, \bigwedge_j (t_{ij} = 0) \rangle \}_i \)

♦ When \( \bot \) is derived, then

\[
I := \bigvee_i (t_i \leq 0 \land \bigwedge_j \text{ExistElim}(x_i \notin B).(t_{ij} = 0))
\]
is the computed interpolant
Interpolants from cutting-plane proofs

- Annotations for Hyp and Comb from [McMillan05] (same as LA(Q))

\[
\text{Hyp} \quad \frac{-}{\text{if } t \leq 0} \quad \frac{\{\langle t' \leq 0, \top \rangle\}}{t' = \begin{cases} t & \text{if } t \leq 0 \in A \\ 0 & \text{if } t \leq 0 \in B \end{cases}}
\]

\[
\text{Comb} \quad \frac{t_1 \leq 0 \ [I_1]}{t_2 \leq 0 \ [I_2]} \quad \frac{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0 \ [I]}{I := \{\langle c_1 t'_i + c_2 t'_j \leq 0, E_i \land E_j \rangle \mid \langle t'_i, E_i \rangle \in I_1, \langle t'_j, E_j \rangle \in I_2\}}
\]

- k-Strengthen rule of [Brillout et al. IJCAR'10] (special case)

\[
\text{Str.} \quad \frac{\sum_i c_i x_i + c \leq 0 \ [\{\langle t \leq 0, \top \rangle\}] \quad \sum_i c_i x_i + d \cdot \left[\frac{c}{d}\right] \leq 0 \ [I]}{, d > 0 \text{ divides the } c_i \text{'s}}
\]

\[
I := \{\langle (t + n \leq 0), (t + n = 0) \rangle \mid 0 \leq n < d \cdot \left[\frac{c}{d}\right] - c\} \cup \{\langle (t + d \cdot \left[\frac{c}{d}\right] - c \leq 0), \top \rangle\}
\]
Interpolants from cutting-plane proofs

♦ Annotations for Hyp and Comb from [McMillan05] (same as LA(Q))

\[
\text{Hyp} \quad \frac{t \leq 0 \left[ \langle t \leq 0, \top \rangle \right]}{t' = \begin{cases} t & \text{if } t \leq 0 \in A \\ 0 & \text{if } t \leq 0 \in B \end{cases}}
\]

\[
\text{Comb} \quad \frac{t_1 \leq 0 \left[ I_1 \right] \quad t_2 \leq 0 \left[ I_2 \right]}{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0 \left[ I \right]}
\]

\[I := \{ \langle c_1 t'_i + c_2 t'_j \leq 0, E_i \land E_j \rangle \mid \langle t'_i, E_i \rangle \in I_1, \langle t'_j, E_j \rangle \in I_2 \} \]

♦ k-Strengthen rule of [Brillout et al. IJCAR'10] (special case)

\[
\text{Str.} \quad \frac{\sum_i c_i x_i + c \leq 0 \left[ \langle t \leq 0, \top \rangle \rangle \right]}{\sum_i c_i x_i + d \cdot \left[ \frac{c}{d} \right] \leq 0 \left[ I \right]}, d > 0 \text{ divides the } c_i \text{'s}
\]

\[I := \{ \langle (t + n \leq 0), (t + n = 0) \rangle \mid 0 \leq n < d \cdot \left[ \frac{c}{d} \right] - c \} \cup \{ \langle (t + d \cdot \left[ \frac{c}{d} \right] - c \leq 0), \top \rangle \} \]
Interpolants from cutting-plane proofs

- Annotations for **Hyp** and **Comb** from [McMillan05] (same as LA(Q))

\[
\text{Hyp} \quad t \leq 0 \quad \Rightarrow \quad t' = \begin{cases} 
  t & \text{if } t \leq 0 \in A \\
  0 & \text{if } t \leq 0 \in B
\end{cases}
\]

\[
\text{Comb} \quad \frac{t_1 \leq 0 \ [I_1]}{t_2 \leq 0 \ [I_2]} \quad \frac{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0 \ [I]}
\]

\[
I := \{ \langle c_1 t'_i + c_2 t'_j \leq 0, E_i \land E_j \rangle \mid \langle t'_i, E_i \rangle \in I_1, \langle t'_j, E_j \rangle \in I_2 \}
\]

- **k-Strengthen** rule of [Brillout et al. IJCAR'10] (special case)

\[
\text{Str.} \quad \frac{\sum_i c_i x_i + c \leq 0 \ [\{ \langle t \leq 0, \top \rangle \}] \quad \sum_i c_i x_i + d \cdot \left\lfloor \frac{c}{d} \right\rfloor \leq 0 \ [I]}{d > 0 \text{ divides the } c_i \text{'s}}
\]

\[
I := \{ \langle (t + n \leq 0), (t + n = 0) \rangle \mid 0 \leq n < d \cdot \left\lfloor \frac{c}{d} \right\rfloor - c \} \cup \{ \langle (t + d \cdot \left\lfloor \frac{c}{d} \right\rfloor - c \leq 0), \top \rangle \}
\]
Example [Kroening et al. LPAR'10]

\[
A := \begin{cases} 
  -y - 4x - 1 \leq 0 \\
  y + 4x \leq 0
\end{cases}
\]

\[
B := \begin{cases} 
  -y - 4z + 1 \leq 0 \\
  y + 4z - 2 \leq 0
\end{cases}
\]

\[
y + 4x \leq 0 \quad -y - 4z + 1 \leq 0
\]

\[
4x - 4z + 1 \leq 0
\]

\[
-4x + 4z - 3 \leq 0
\]

\[
(1 \leq 0) \equiv \bot
\]
Example – with annotations

\[ A := \begin{cases} 
  -y - 4x - 1 \leq 0 \\
  y + 4x \leq 0 
\end{cases} \quad B := \begin{cases} 
  -y - 4z + 1 \leq 0 \\
  y + 4z - 2 \leq 0 
\end{cases} \]

\[ y + 4x \leq 0 \quad -y - 4z + 1 \leq 0 \]

\[ \{ \langle y + 4x \leq 0, \top \rangle \} \quad \{ \langle 0 \leq 0, \top \rangle \} \]

\[ 4x - 4z + 1 \leq 0 \]

\[ \{ \langle y + 4x \leq 0, \top \rangle \} \]

\[ -y - 4x - 1 \leq 0 \quad y + 4z - 2 \leq 0 \]

\[ \{ \langle -y - 4x - 1 \leq 0, \top \rangle \} \quad \{ \langle 0 \leq 0, \top \rangle \} \]

\[ 4x - 4z + 1 + 3 \leq 0 \]

\[ \{ \langle y + 4x + n \leq 0, y + 4x + n = 0 \rangle \mid 0 \leq n < 3 \} \cup \{ \langle y + 4x + 2 \leq 0, \top \rangle \} \]

\[ -4x + 4z - 3 \leq 0 \]

\[ \{ \langle -y - 4x - 1 \leq 0, \top \rangle \} \]

\[ (1 \leq 0) \equiv \bot \]

\[ \{ \langle n - 1 \leq 0, y + 4x + n = 0 \rangle \mid 0 \leq n < 3 \} \cup \{ \langle 2 - 1 \leq 0, \top \rangle \} \]}
Example – with annotations

\[
A := \begin{cases} 
- y - 4x - 1 \leq 0 \\
 y + 4x \leq 0 
\end{cases}
\]

\[
B := \begin{cases} 
- y - 4z + 1 \leq 0 \\
 y + 4z - 2 \leq 0 
\end{cases}
\]

\[
y + 4x \leq 0 \quad - y - 4z + 1 \leq 0 \\
\{\langle y + 4x \leq 0, \top \rangle\} \quad \{\langle 0 \leq 0, \top \rangle\}
\]

\[
4x - 4z + 1 \leq 0 \\
\{\langle y + 4x \leq 0, \top \rangle\}
\]

\[
\overline{4x - 4z + 1 + 3 \leq 0} \\
\{\langle y + 4x + n \leq 0, y + 4x + n = 0 \rangle \mid 0 \leq n < 3\} \cup \{\langle y + 4x + 2 \leq 0, \top \rangle\}
\]

\[
(1 \leq 0) \equiv \bot \\
\{\langle n - 1 \leq 0, y + 4x + n = 0 \rangle \mid 0 \leq n < 3\} \cup \{\langle 2 - 1 \leq 0, \top \rangle\}
\]

Interpolant:
\[
(y =_4 0) \lor (y + 1 =_4 0)
\]
Drawback of Strengthen

♦ Interpolation of Strengthen creates potentially very big disjunctions

♦ Linear in the strengthening factor  \( k := d\left[\frac{c}{d}\right] - c \)

♦ Can be exponential in the size of the proof

Example:

\[
A := \begin{cases}
    -y - 4x - 1 \leq 0 \\
y + 4x \leq 0
\end{cases}
\]

\[
B := \begin{cases}
    -y - 4z + 1 \leq 0 \\
y + 4z - 2 \leq 0
\end{cases}
\]

Interpolant: \((y = 4 \ 0) \lor (y + 1 = 4 \ 0)\)
Drawback of Strengthen

- Interpolation of Strengthen creates potentially very big disjunctions
- Linear in the strengthening factor $k := d \left\lfloor \frac{c}{d} \right\rfloor - c$
- Can be exponential in the size of the proof

Example:

$A := \left\{ \begin{array}{l}
-y - 2nx - n + 1 \leq 0 \\
y + 2nx \leq 0
\end{array} \right.$

$B := \left\{ \begin{array}{l}
-y - 2nz + 1 \leq 0 \\
y + 2nz - n \leq 0
\end{array} \right.$

Interpolant: $(y =_{2n} 0) \lor (y + 1 =_{2n} 0) \lor \ldots \lor (y =_{2n} n - 1)$
Drawback of Strengthen

♦ Interpolation of Strengthen creates potentially very big disjunctions

♦ Linear in the strengthening factor \( k := d \left[ \frac{c}{d} \right] - c \)

♦ Can be exponential in the size of the proof

Example:

\[
A := \begin{cases} 
-y - 2nx - n + 1 & \leq 0 \\
y + 2nx & \leq 0
\end{cases}
\]

\[
B := \begin{cases} 
-y - 2nz + 1 & \leq 0 \\
y + 2nz - n & \leq 0
\end{cases}
\]

Interpolant: \((y =_{2n} 0) \lor (y + 1 =_{2n} 0) \lor \ldots \lor (y =_{2n} n - 1)\)

♦ The problem are AB-mixed cuts:

Strengthen

\[
\frac{\sum_{x_i \notin B} c_i x_i + \sum_{y_j \notin A} c_j y_j + c}{\sum_{x_i \notin B} c_i x_i + \sum_{y_j \notin A} c_j y_j + d \cdot \left[ \frac{c}{d} \right]} \leq 0
\]
Avoid the problem by avoiding mixed cuts

- Algorithm based on reduction to LA(Q) + Diophantine equations
- Generate interpolants linear in the size of proofs

However, this is a strong restriction:

- Forbids use of popular LA(Z) techniques like Gomory cuts, cuts from proofs [Dillig et al CAV'09], the Omega test [Pugh91]
- Might generate much larger proofs
Solution of [Kroening et al. LPAR'10]

- Avoid the problem by **avoiding mixed cuts**
  - Algorithm based on reduction to LA(Q) + Diophantine equations
- Generate interpolants **linear** in the size of proofs
- However, this is a **strong restriction**:
  - Forbids use of popular LA(Z) techniques like **Gomory cuts**, **cuts from proofs** [Dillig et al CAV'09], the **Omega test** [Pugh91]
  - Might generate **much larger proofs**

Example:

\[
A := \begin{cases} 
-y - 2nx - n + 1 \leq 0 \\
y + 2nx \leq 0
\end{cases} \quad B := \begin{cases} 
-y - 2nz + 1 \leq 0 \\
y + 2nz - n \leq 0
\end{cases}
\]

Without \(AB\)-mixed cuts, proof of **exponential size**

Same interpolant as with **Strengthen**

In fact, [Kroening et al. LPAR'10] shows this is the **only** interpolant for \((A, B)\)
Avoid the problem by avoiding mixed cuts

- Algorithm based on reduction to LA(Q) + Diophantine equations
- Generate interpolants linear in the size of proofs

However, this is a strong restriction:

- Forbids use of popular LA(Z) techniques like Gomory cuts, cuts from proofs [Dillig et al CAV'09], the Omega test [Pugh91]
- Might generate much larger proofs

Example:

\[
A := \begin{cases} 
  -y - 2nx \\ 
y + 2nx \leq 0
\end{cases}
\]

Implicit assumption: we are in the signature \( \mathbb{Z} \cup \{+, \cdot, =, \leq\} \cup \{=g\} \) for \( g \in \mathbb{Z}^> 0 \)

Without AB-mixed cuts, proof of exponential size

Same interpolant as with Strengthen

In fact, [Kroening et al. LPAR'10] shows this is the only interpolant for (A, B)
Outline

♦ Background

♦ Current techniques for interpolation in LA(Z)

♦ A novel interpolation technique for LA(Z)

♦ Experimental evaluation
Interpolation with ceilings

- Idea: use a different extension of the signature of LA(Z), and extend also its domain
  - Introduce the ceiling function $\lceil \cdot \rceil$ [Pudlák '97]
  - Allow non-variable terms to be non-integers (e.g. $\frac{x}{2}$)
- Much simpler interpolation procedure
  - Proof annotations are single inequalities ($t \leq 0$)
Interpolation with ceilings

- Idea: use a different extension of the signature of LA(Z), and extend also its domain
- Introduce the ceiling function \( \lceil \cdot \rceil \) [Pudlák '97]
- Allow non-variable terms to be non-integers (e.g. \( \frac{x}{2} \))
- Much simpler interpolation procedure
- Proof annotations are single inequalities \( t \leq 0 \)

\[
\begin{array}{c}
\text{Hyp} \quad t \leq 0 \quad [t' \leq 0] \\
\sum y_j \not\in B \quad \frac{a_j y_j}{\text{Div}} \quad \sum z_k \not\in A \quad b_k z_k + \sum x_i \in A \land B \quad c_i x_i + c
\end{array}
\]

\[
\begin{array}{c}
\text{Comb} \quad \frac{t_1 \leq 0 \quad [t'_1 \leq 0]}{}
\quad \frac{t_2 \leq 0 \quad [t'_2 \leq 0]}{}
\quad \frac{c_1 \cdot t_1 + c_2 \cdot t_2 \leq 0}{c_1 \cdot t'_1 + c_2 \cdot t'_2 \leq 0}
\end{array}
\]

\[
\begin{array}{c}
\sum y_j \not\in B \quad \frac{a_j y_j}{\text{Div}} \quad \sum z_k \not\in A \quad \frac{b_k z_k}{\text{Div}} + \sum x_i \in A \land B \quad \frac{c_i x_i + c'}{d}
\end{array}
\]
Interpolation with ceilings - example

♦ No blowup of interpolants wrt. the size of the proofs

\[
A := \begin{cases} 
- y - 2nx - n + 1 \leq 0 \\
y + 2nx \leq 0 
\end{cases} \\
B := \begin{cases} 
- y - 2nzw + 1 \leq 0 \\
y + 2nzw - n \leq 0 
\end{cases}
\]

\[
y + 2nx \leq 0 \quad -y - 2nzw + 1 \leq 0
\]

\[
2nx - 2nzw + 1 \leq 0
\]

\[
y - 2nx - n + 1 \leq 0 \quad y + 2nzw - n \leq 0
\]

\[
2n \cdot (x - z + 1 \leq 0)
\]

\[
-2nx + 2nzw - 2n + 1 \leq 0
\]

\[
(1 \leq 0) \equiv \bot
\]
Interpolation with ceilings - example

- No blowup of interpolants wrt. the size of the proofs

\[ A := \begin{cases} 
- y - 2nx - n + 1 & \leq 0 \\
y + 2nx & \leq 0 
\end{cases} \quad B := \begin{cases} 
- y - 2nz + 1 & \leq 0 \\
y + 2nz - n & \leq 0 
\end{cases} \]

\[
y + 2nx \leq 0 \quad -y - 2nz + 1 \leq 0
\]
\[
\begin{array}{c}
[ y + 2nx \leq 0 ] \\
[ 0 \leq 0 ]
\end{array}
\]

\[
2nx - 2nz + 1 \leq 0
\]
\[
\begin{array}{c}
[ y + 2nx \leq 0 ]
\end{array}
\]

\[
2n \cdot (x - z + 1 \leq 0)
\]
\[
\begin{array}{c}
x + \left[ \frac{y}{2n} \right] \leq 0
\end{array}
\]

\[
(1 \leq 0) \equiv \bot
\]

\[
[2n\left\lceil \frac{y}{2n} \right\rceil - y - n + 1 \leq 0]
\]

\[
-2nx + 2nz - 2n + 1 \leq 0
\]
\[
\begin{array}{c}
[ -y - 2nx - n + 1 \leq 0 ]
\end{array}
\]

\[
y + 2nz - n \leq 0
\]
\[
\begin{array}{c}
[ 0 \leq 0 ]
\end{array}
\]
Interpolation with ceilings - example

♦ No blowup of interpolants wrt. the size of the proofs

\[ A := \begin{cases} 
- y - 2nx - n + 1 \leq 0 \\
y + 2nx \leq 0 
\end{cases} \quad \text{and} \quad B := \begin{cases} 
- y - 2nz + 1 \leq 0 \\
y + 2nz - n \leq 0
\end{cases} \]

\[
\begin{align*}
y + 2nx &\leq 0 \quad -y - 2nz + 1 \leq 0 \\
[y + 2nx \leq 0] &\quad [0 \leq 0] \\
2nx - 2nz + 1 &\leq 0 \\
[y + 2nx \leq 0] &\quad [y + 2nx \leq 0] \\
2n \cdot (x - z + 1 &\leq 0) \\
[x + \left\lfloor \frac{y}{2n} \right\rfloor &\leq 0] \\
\end{align*}
\]

Interpolant:
\[ (2n \left\lfloor \frac{y}{2n} \right\rfloor - y - n + 1 \leq 0) \]

\[
\begin{align*}
- y - 2nx - n + 1 &\leq 0 \quad y + 2nz - n \leq 0 \\
[y + 2nx - n + 1 &\leq 0] &\quad [0 \leq 0] \\
-2nx + 2nz - 2n + 1 &\leq 0 \\
[y + 2nx - n + 1 &\leq 0] &\quad [y + 2nx - n + 1 \leq 0] \\
\end{align*}
\]

\[ (1 \leq 0) \equiv \bot \]

\[ [2n \left\lfloor \frac{y}{2n} \right\rfloor - y - n + 1 \leq 0] \]
SMT(LA(Z)) with ceilings

- Like modular equations, also ceilings can be eliminated via preprocessing:
  - Replace every term \( \lceil t \rceil \) with a fresh integer variable \( x_{\lceil t \rceil} \)
  - Add the 2 unit clauses
    (encoding the meaning of ceiling: \( \lceil t \rceil - 1 < t \leq \lceil t \rceil \))
    \[
    (l \cdot x_{\lceil t \rceil} - l \cdot t + l \leq 0) \\
    (l \cdot t - l \cdot x_{\lceil t \rceil} \leq 0)
    \]
    where \( l \) is the least common multiple of the denominators of the coefficients in \( t \)
Outline

♦ Background

♦ Current techniques for interpolation in LA(Z)

♦ A novel interpolation technique for LA(Z)

♦ Experimental evaluation
Experiments

♦ Implementation on top of MathSAT 5

♦ Use also algorithm for Diophantine equations and Boolean interpolation algorithm for dealing with Branch and Bound

♦ Implemented both algorithm based on Strengthen (MathSAT-ModEq) and on ceilings (MathSAT-Ceil)

♦ Use benchmarks that require non-trivial integer reasoning
Results – Strengthen vs. ceilings

- **Execution Time**
  - MathSAT-Ceil vs. MathSAT-ModEq

- **Size of Interpolants**
  - MathSAT-Ceil vs. MathSAT-ModEq
Thank You