Symbolic Optimization with SMT Solvers

 Aws Albarghouthi / UToronto
 Marsha Chechik / UToronto
 Arie Gurfinkel / CMU
 Zachary Kincaid / UToronto
 Yi Li / UToronto

 POPL 2014 / San Diego, CA
SMT Explosion!

SMT solvers appear everywhere. Why?

• Amazing performance!

• Support a large range of logical theories

• We’ve become really good at casting problems as SMT queries!
SMT Applications

Verification
  • Checking VCs, invariant generation, etc.

Bug finding
  • Symbolic execution, BMC, fuzzing, etc.

Synthesis
  • Circuit synthesis, sketching, superoptimization, etc.

Functional programming
  • Liquid types
SMT Applications

Verification
• Checking VCs, invariant generation, etc.

Bug finding
• Symbolic execution, BMC, fuzzing, etc.

Synthesis
• Circuit synthesis, sketching, superoptimization, etc.

Functional programming
• Liquid types

~22% of POPL’14 papers mention SMT solvers!
How are SMT Solvers Used?

Finding models

• **Bug finding:** erroneous traces

• **Synthesis:** program/circuit

Proving unsatisfiability (validity)

• **Verification:** VC holds

• **Refinement types:** subtyping relation holds
How are SMT Solvers Used?

What about optimization?
Optimal Models

$\varphi \xrightarrow{\text{SMT solver}} m \models \varphi$
Optimal Models

\[ \varphi \rightarrow SMT \text{ solver} \rightarrow m \models \varphi \]

\[ \varphi, f \rightarrow \text{Optimizing SMT solver} \]
Optimal Models

\[ \varphi, f \xrightarrow{\text{Optimizing SMT solver}} m \models \varphi \]

\[ \varphi \xrightarrow{\text{SMT solver}} m \models \varphi \]

\[ \max f(m) \]
Why Should You Care?

Plenty of applications for optimization:

• *Numerical invariant generation*

• *Counterexample generation*

• *Program synthesis*

• *Constraint programming*

• *... and many others*
Problem Statement

\( \varphi \in T \cup LRA \)

signature disjoint
## Problem Statement

<table>
<thead>
<tr>
<th>$\varphi \in T \cup LRA$</th>
<th>E.g.: $3x + 2y \leq 0 \lor z \geq 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>signature disjoint</td>
<td></td>
</tr>
</tbody>
</table>
Problem Statement

\[ \varphi \in \mathcal{T} \cup \mathcal{LRA} \]

signature disjoint

Set of linear objective functions: \( f_1, \ldots, f_n \)

E.g.: \( x + 2y, z \)

E.g.: \( 3x + 2y \leq 0 \lor z \geq 3 \)
Problem Statement

\[ \varphi \in T \cup LRA \quad \text{signature disjoint} \]

E.g.: \[ 3x + 2y \leq 0 \lor z \geq 3 \]

Set of linear \textit{objective functions}: \( f_1, \ldots, f_n \)

E.g.: \( x + 2y, z \)

\textbf{Goal: find assignments} \( m_1, \ldots, m_n \)

\[ m_1 \models \varphi \text{ s.t. } \max f_1(m_1) \]

\[ \ldots \]

\[ m_n \models \varphi \text{ s.t. } \max f_n(m_n) \]
Problem Statement

\[ \varphi \in T \cup LRA \]

Set of linear \textit{objective functions}: \( f_1, \ldots, f_n \)
Problem Statement

\[ \varphi \in T \cup LRA \]
Set of linear *objective functions*: \( f_1, \ldots, f_n \)
Problem Statement

φ ∈ $T \cup LRA$

Set of linear **objective functions**: $f_1, \ldots, f_n$

$f_1 \leq k_1$

$f_2 \leq k_2$
Problem Statement

\( \varphi \in T \cup LRA \)

Set of linear **objective functions**: \( f_1, \ldots, f_n \)
Problem Statement

\[ \varphi \in T \cup LRA \]

Set of linear \textit{objective functions}: \( f_1, \ldots, f_n \)

Find strongest \( \bigwedge_{i \in [1,n]} f_i \leq k_i \) that contains \( \varphi \)
Challenges & Contributions

_Symba_: an SMT-based optimization algorithm

- Non-convex optimization
- Linear arithmetic modulo theories
- Multiple independent objectives
- SMT solver as a black box
Outline

Symba by example

Application and evaluation

What’s next?
Example

\[ \varphi \equiv 1 \leq y \leq 3 \land (1 \leq x \leq 3 \lor x \geq 4) \]

Objective functions: \( \{y, x + y\} \)
Example

\[ \varphi \equiv 1 \leq y \leq 3 \land (1 \leq x \leq 3 \lor x \geq 4) \]

Objective functions: \( \{y, x + y\} \)
Example

\[ \varphi \equiv 1 \leq y \leq 3 \land (1 \leq x \leq 3 \lor x \geq 4) \]

Objective functions: \( \{y, x + y\} \)

**Optimal Solution:**

\[ y \leq 3 \land x + y \leq \infty \]
Example

\[ \varphi \equiv 1 \leq y \leq 3 \land (1 \leq x \leq 3 \lor x \geq 4) \]

Objective functions: \( \{y, x + y\} \)

Optimal Solution:
\[ y \leq 3 \land x + y \leq \infty \]
Example

Objective functions: \( \{y, x + y\} \)

Under-approximation of optimal solution: false
Example

Objective functions: \( \{y, x + y\} \)

Under-approximation of optimal solution: \textit{false}

Phase 1: Grow under-approximation

Illustration of \textit{YMBA} on a 2-D example.
Example

Objective functions: \( \{y, x + y\} \)

Under-approximation of optimal solution: \textit{false}

Phase 1: Grow under-approximation
Example

Objective functions: \( \{ y, x + y \} \)

Under-approximation of optimal solution: \(false\)

Phase 1: Grow under-approximation

New under-approx:
\[
\begin{align*}
y & \leq 2 \land \\
x + y & \leq 4
\end{align*}
\]
Example

Objective functions: \{y, x + y\}

Phase 2: Check if \( y \) is unbounded
Example

**Phase 2: Check if \( y \) is unbounded**

Pick point \( p_1 \)

Find point \( p_1' \) s.t.
- increases value of \( y \)
- sits on the same boundaries

![Illustration of S](image)

**Formulas**

3.1 Definitions

Paper and formalize S

then \( U \)

\( \vdash \)

\( \models \)

Geometrically, \( G \)

where

\( y \)

represents a local maximum.

\( \) cannot apply \( U \)

after applying \( U \)

\( p \)

represents the maximum value of \( y \)

\( y \)

is unbounded, \( S \)

represents.

Example

phase

\( p \)

is a valuation of the variables of \( q \)

Combinations of Theories

OPT

For clarity of presentation, we treated

\( T \)

\( \vdash \)

\( \models \)

\( ::= \)

\( \) is an over-approximation of

\( \) is invariant); and

\( \) is an approximation of

\( \) computes
Example

Phase 2: Check if $y$ is unbounded

Pick point $p_1$

Find point $p_1'$ s.t.

- increases value of $y$
- sits on the same boundaries

Find point $p_2$ s.t.

- increases value of $y$
- sits on more boundaries

Figure 1.
Atoms $\vdash y$

Illustration of $\text{YMBA}$

In this case, it might find the point $p$ s.t. $\exists \gamma \models \rho$, where $\gamma$ is a valuation of the variables of $\text{S'}$. $\text{S'}$ becomes $\text{NBOUNDED}$ and terminates with the optimal solution $(p_0, \text{S'} \models \text{NBOUNDED})$. Geometrically, $\text{USH} \cdot p_0 \cdot \text{LOCAL} < y$ and increases value of $y$. As a result, $\text{USH}$ sits on the boundaries $\forall y \in \text{Vars}$. $\text{YMBA}$ computes $\text{opt}$ to denote the set of all satisfying assignments $\gamma$. $\text{USH}$ sits on the boundaries $\forall y \in \text{Vars}$.
Example

Phase 2: Check if $y$ is unbounded

Pick point $p_1$

Find point $p'_1$ s.t.
- increases value of $y$
- sits on the same boundaries

Find point $p_2$ s.t.
- increases value of $y$
- sits on more boundaries

Figure 1. Illustration of $YMBA$

Figure 2. Optimal Solutions
Example

Phase 2: Check if $y$ is unbounded

Pick point $p_1$

Find point $p'_1$ s.t.
- increases value of $y$
- sits on the same boundaries

Find point $p_2$ s.t.
- increases value of $y$
- sits on more boundaries

New under-approx:
\[
\begin{align*}
y &\leq 3 \\
x + y &\leq 6
\end{align*}
\]
Example

Objective functions: \( \{ y, x + y \} \)

Phase 1: Grow under-approximation

\[
\text{Example}
\]

Objective functions: \( \{ y, x + y \} \)

Phase 1: Grow under-approximation
**Example**

**Objective functions:** $\{y, x + y\}$

**Phase 1: Grow under-approximation**
Example

**Objective functions:** \( \{ y, x + y \} \)

**Phase 1: Grow under-approximation**

**New under-approx:**
\[
\begin{align*}
y & \leq 3 \\
x + y & \leq 8
\end{align*}
\]
Example

Objective functions: \( \{y, x + y\} \)

Phase 2: Check if \( x + y \) is unbounded
Example

Objective functions: \{y, x + y\}

Phase 2: Check if \(x + y\) is unbounded

Can keep increasing without hitting a boundary
Example

Objective functions: \( \{ y, x + y \} \)

Phase 2: Check if \( x + y \) is unbounded

Can keep increasing without hitting a boundary

Optimal solution:
\[
\begin{align*}
y & \leq 3 \\
x + y & \leq \infty
\end{align*}
\]
Symba in a Nutshell

Alternate between two phases

- *Sampling*: grow under-approximation
- *Check if objective function is unbounded*
Symba in a Nutshell

Alternate between two phases

- *Sampling*: grow under-approximation
- *Check if objective function is unbounded*

*Fair* alternation ensures completeness
Symba in a Nutshell

Alternate between two phases

- *Sampling*: grow under-approximation
- *Check if objective function is unbounded*

*Fair* alternation ensures completeness

Algorithm also maintains an over-approx

- See *paper*
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes

Find $p_1, p_2$ in same boundary class s.t.

$f(p_1) < f(p_2)$

no $p_3$ exists in stronger boundary class where $f(p_3) \geq f(p_2)$
Symba Abstractly

Arrange infinitely many models into finitely many **boundary classes**

Find $p_1, p_2$ in same boundary class s.t.

$f(p_1) < f(p_2)$

no $p_3$ exists in stronger boundary class where $f(p_3) \geq f(p_2)$

**Necessary and sufficient condition to prove unboundedness of $f$**
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes

Find \( p_1, p_2 \) in same boundary class s.t.
\[
f(p_1) < f(p_2)
\]
no \( p_3 \) exists in stronger boundary class where \( f(p_3) \geq f(p_2) \)

Necessary and sufficient condition to prove unboundedness of \( f \)

Symba searches through lattice of classes!
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes

Find \( p_1, p_2 \) in same boundary class s.t.
\[
f(p_1) < f(p_2)
\]
no \( p_3 \) exists in stronger boundary class where \( f(p_3) \geq f(p_2) \)

Necessary and sufficient condition to prove unboundedness of \( f \)

Symba searches through lattice of classes!
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes

Find \( p_1, p_2 \) in same boundary class s.t.
\[
f(p_1) < f(p_2)
\]
no \( p_3 \) exists in stronger boundary class where \( f(p_3) \geq f(p_2) \)

Necessary and sufficient condition to prove unboundedness of \( f \)

Symba searches through lattice of classes!
Symba Abstractly

Arrange infinitely many models into finitely many boundary classes

Find $p_1, p_2$ in same boundary class s.t.

$\begin{align*}
    f(p_1) &< f(p_2) \\
    \text{no } p_3 \text{ exists in stronger boundary class where } & f(p_3) \geq f(p_2)
\end{align*}$

Necessary and sufficient condition to prove unboundedness of $f$

Symba searches through lattice of classes!
Application

Implemented Symba using Z3

Application: Computing *precise abstract transformers*:

- *TCM domains (intervals, octagons, etc.)*
  [Sankaranarayanan et al., VMCAI‘05]

- *Complex transition relations (multiple paths)*
Application

Implemented Symba using Z3

Application: Computing *precise abstract transformers*:

- *TCM domains (intervals, octagons, etc.*)*  
  [Sankaranarayanan et al., VMCAI‘05]

- *Complex transition relations (multiple paths)*

\[ S \]

*Initial states*
Application

Implemented Symba using Z3

Application: Computing *precise abstract transformers*:

- *TCM domains (intervals, octagons, etc.)*
  [Sankaranarayanan et al., VMCAI‘05]

- *Complex transition relations (multiple paths)*

\[ S \rightarrow T \]

*Initial states*  
*Transition relation (multiple paths)*
Application

Implemented Symba using Z3

Application: Computing precise abstract transformers:

- TCM domains (intervals, octagons, etc.)
  [Sankaranarayanan et al., VMCAI‘05]

- Complex transition relations (multiple paths)

\[ S \land T \]

Initial states

Transition relation (multiple paths)
Application

Implemented Symba using Z3

Application: Computing *precise abstract transformers*:

- *TCM domains* (intervals, octagons, etc.)
  [Sankaranarayanan et al., VMCAI‘05]
- *Complex transition relations* (multiple paths)

\[ S \land T \]

*Initial states*  
*Transition relation (multiple paths)*

**Objective functions:**

**Intervals domain:**

\[ \{ x, -x, \ldots \} \]
Application

 Implemented Symba using Z3

 Application: Computing *precise abstract transformers*:

  - *TCM domains* (intervals, octagons, etc.)
    [Sankaranarayanan et al., VMCAI’05]
  - *Complex transition relations* (multiple paths)

\[
S \land T
\]

Initial states

Transition relation (multiple paths)

Objective functions:

**Intervals domain:**
\[
\{ x, -x, \ldots \}
\]

**Octagons domain:**
\[
\{ x + y, \ldots \}
\]
Evaluation

Instrumented UFO [CAV’12] to generate abstract post queries from SV-COMP programs

• Took the ~1000 hardest benchmarks
• Average # of variables: ~900 (max: ~19,000)
• Average # of objective functions: 56 (max: 386)
Evaluation

Compared w/ OptMathSAT [Sebastiani & Tomasi IJCAR’12]
- Modifies SIMPLEX within SMT solver to find a local optimum
- Handles a single objective function at a time
Evaluation

Compared w/ OptMathSAT [Sebastiani & Tomasi IJCAR’12]

- Modifies SIMPLEX within SMT solver to find a local optimum
- Handles a single objective function at a time

**Figure 5.**

**Table 2.**

<table>
<thead>
<tr>
<th></th>
<th>OptMathSAT</th>
<th>SymbaIR (40)</th>
<th>SymbaIR (60)</th>
<th>SymbaIR (100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#SMT calls</td>
<td>132,051</td>
<td>84,814</td>
<td>20,854</td>
<td>60</td>
</tr>
<tr>
<td>Time (s)</td>
<td>6,867</td>
<td>3,841</td>
<td>164,156</td>
<td>577,068</td>
</tr>
<tr>
<td>Opt-Z3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATH</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SymbaIR's Configurations

- (40) handles a single objective function at a time.
- (60) and (100) capture the results of running SymbaIR with different timeouts.

**Diagram:**

The diagram shows the performance comparison between OptMathSAT and Symba (vanilla). The x-axis represents Symba (vanilla) time in seconds, and the y-axis represents OptMathSAT time in seconds. The scatter plot indicates a linear relationship between the two, with a notable increase in time for OptMathSAT as the complexity of the benchmarks increases.
OptMathSAT

Compared w/ OptMathSAT [Sebastiani & Tomasi IJCAR’12]

- Modifies SIMPLEX within SMT solver to find a local optimum
- Handles a single objective function at a time

**Evaluation**

**Time in s / benchmark**

- **OptMathSAT**
- **Symba (vanilla)**

**Symba outperforms OptMathSAT**

**Average speedup: 2.2x**
Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark

Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

Our OptMathSAT (in Z3)

Time in s / benchmark
Evaluation

Implemented [Sebastiani & Tomasi, IJCAR’12] within Z3

- Extended it to optimize multiple objectives simultaneously

<table>
<thead>
<tr>
<th></th>
<th>Time in s / benchmark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our OptMathSAT (in Z3)</td>
<td></td>
</tr>
<tr>
<td><strong>Symba (vanilla)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Symba consistently slower</strong></td>
<td>than OptMathSAT(Z3)</td>
</tr>
</tbody>
</table>

Our OptMathSAT (in Z3)

Symba consistently slower than OptMathSAT(Z3)
Evaluation

Optimized Symba

- Spends 40% of the time (at most) performing unbounded checks
- Uses a modified, “locally optimal” Z3 for growing under-approx
Evaluation

Optimized Symba
- Spends 40% of the time (at most) performing unbounded checks
- Uses a modified, “locally optimal” Z3 for growing under-approx

Our OptMathSAT (in Z3)

Time in s / benchmark

Symba (optimized)
Evaluation

Optimized Symba

- Spends 40% of the time (at most) performing unbounded checks
- Uses a modified, “locally optimal” Z3 for growing under-approx

![Scatter plot showing Time in s / benchmark vs Symba (optimized) with 1.5x speedup over OptMathSAT(Z3) No timeouts with Best Symba config. (see paper for more)]
Conclusion

_Symba_: non-convex optimization

- Efficient SMT-based implementation
- Many applications in program analysis and beyond

Future work

- Integer arithmetic
- Non-linear arithmetic
- Parallelization
Conclusion

**Symba**: non-convex optimization

- Efficient SMT-based implementation
- Many applications in program analysis and beyond

Future work

- Integer arithmetic
- Non-linear arithmetic
- Parallelization

bitbucket.org/arieg/ufo