Enumerating Choice Terms in Model-Based Quantifier Instantiation

Lydia Kondylidou[1] Andrew Reynolds[2] Jasmin Blanchette[1] Cesare Tinelli[2]

[1]Ludwig-Maximilians-Universität München
[2]University of Iowa

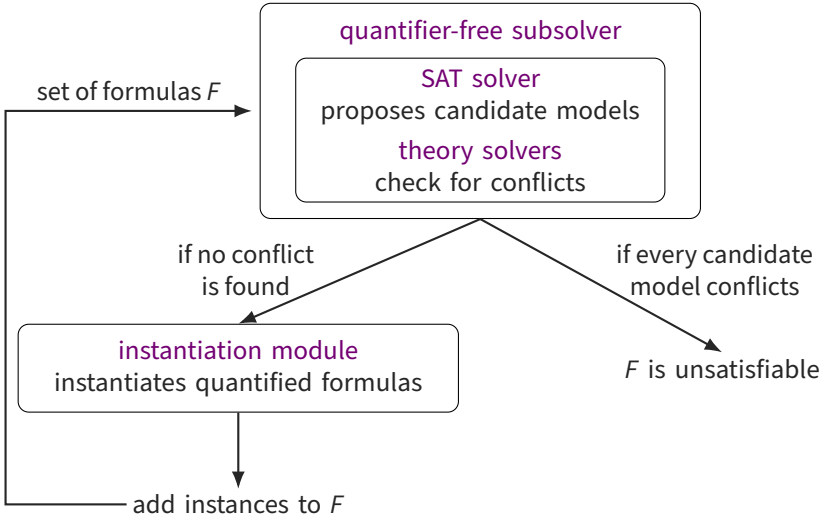
Satisfiability Modulo Theories (SMT) Solvers

SMT solvers combine a SAT solver with decision procedures for theories.

They work by refutation: Assume the negation of the conjecture and derive $\bot$.

They are usually based on first-order logic, but some support higher-order logic.

Satisfiability Modulo Theories (SMT) Solvers



Instantiation Strategies

Instantiation strategies compute substitutions for universally quantified variables, mapping them to ground terms.

Instantiation Strategies

Instantiation strategies compute substitutions for universally quantified variables, mapping them to ground terms.

For a quantified formula

$$\forall x_1, \ldots, x_n.\ \psi,$$

Instantiation Strategies

Instantiation strategies compute substitutions for universally quantified variables, mapping them to ground terms.

For a quantified formula

$$\forall x_1, \ldots, x_n.\ \psi,$$

a strategy computes a substitution

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

Instantiation Strategies

Instantiation strategies compute substitutions for universally quantified variables, mapping them to ground terms.

For a quantified formula

$$\forall x_1, \ldots, x_n.\ \psi,$$

a strategy computes a substitution

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

and the SMT solver produces the lemma

$$q \Longrightarrow \sigma(\psi).$$

Instantiation Strategies

Instantiation strategies compute substitutions for universally quantified variables, mapping them to ground terms.

For a quantified formula

$$\forall x_1, \ldots, x_n.\ \psi,$$

a strategy computes a substitution

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$$

and the SMT solver produces the lemma

$$q \Longrightarrow \sigma(\psi).$$

Existential quantifiers are handled by Skolemization.

MBQI-Enum

Higher-order quantifier instantiation is more involved than in first-order.

Our work in cvc5, MBQI-Enum, extends MBQI with a SyGuS-based enumerator.

MBQI-Enum

Higher-order quantifier instantiation is more involved than in first-order.

Our work in cvc5, MBQI-Enum, extends MBQI with a SyGuS-based enumerator.

Key idea:

Instead of relying only on terms already present in the model,

generate new candidate terms from a grammar built from symbols in the problem,

including λ-abstractions for variables of function type.

MBQI-Enum Example

Consider the input problem

$$F = \{ \forall y.\ \exists z.\ y\, z \neq \mathsf{f}\, z \},$$

where $\mathsf{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum Example

Consider the input problem

$$F = \{\forall y.\ \exists z.\ y\,z \neq \mathsf{f}\,z\},$$

where $\mathsf{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum computes $\sigma = \{y \mapsto \lambda x.\ \mathsf{t}\}$ and builds a grammar for y:

$$g_0 ::= \lambda x.\ g_1 \qquad g_1 ::= x \mid \mathsf{f}\,g_1$$

MBQI-Enum Example

Consider the input problem

$$F = \{\forall y.\ \exists z.\ y\,z \neq \mathsf{f}\,z\},$$

where $\mathsf{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum computes $\sigma = \{y \mapsto \lambda x.\ \mathsf{t}\}$ and builds a grammar for y:

$$g_0 ::= \lambda x.\ g_1 \qquad g_1 ::= x \mid \mathsf{f}\,g_1$$

When the enumeration reaches $\lambda x.\ \mathsf{f}\,x$, MBQI-Enum considers $\sigma = \{y \mapsto \lambda x.\ \mathsf{f}\,x\}$.

MBQI-Enum Example

Consider the input problem

$$F = \{\forall y.\ \exists z.\ y\,z \neq \mathsf{f}\,z\},$$

where $\mathsf{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum computes $\sigma = \{y \mapsto \lambda x.\ \mathsf{t}\}$ and builds a grammar for y:

$$g_0 ::= \lambda x.\ g_1 \qquad g_1 ::= x \mid \mathsf{f}\,g_1$$

When the enumeration reaches $\lambda x.\ \mathsf{f}\,x$, MBQI-Enum considers $\sigma = \{y \mapsto \lambda x.\ \mathsf{f}\,x\}$.

The quantified formula becomes $\exists z.\ (\lambda x.\ \mathsf{f}\,x)\,z \neq \mathsf{f}\,z$, i.e. $\exists z.\ \mathsf{f}\,z \neq \mathsf{f}\,z$.

MBQI-Enum Example

Consider the input problem

$$F = \{\forall y.\ \exists z.\ y\,z \neq \mathsf{f}\,z\},$$

where $\mathsf{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum computes $\sigma = \{y \mapsto \lambda x.\ \mathsf{t}\}$ and builds a grammar for y:

$$g_0 ::= \lambda x.\ g_1 \qquad g_1 ::= x \mid \mathsf{f}\,g_1$$

When the enumeration reaches $\lambda x.\ \mathsf{f}\,x$, MBQI-Enum considers $\sigma = \{y \mapsto \lambda x.\ \mathsf{f}\,x\}$.

The quantified formula becomes $\exists z.\ (\lambda x.\ \mathsf{f}\,x)\,z \neq \mathsf{f}\,z$, i.e. $\exists z.\ \mathsf{f}\,z \neq \mathsf{f}\,z$.

The instantiation lemma

$$(\forall y.\ \exists z.\ y\,z \neq \mathsf{f}\,z) \Longrightarrow \exists z.\ \mathsf{f}\,z \neq \mathsf{f}\,z$$

is added to F.

MBQI-Enum Example

Consider the input problem

$$F = \{\forall y.\ \exists z.\ y\,z \neq \mathrm{f}\,z\},$$

where $\mathrm{f} : u \to u$ is a function symbol and $y : u \to u$ is a variable.

MBQI-Enum computes $\sigma = \{y \mapsto \lambda x.\ \mathrm{t}\}$ and builds a grammar for y:

$$g_0 ::= \lambda x.\ g_1 \qquad g_1 ::= x \mid \mathrm{f}\,g_1$$

When the enumeration reaches $\lambda x.\ \mathrm{f}\,x$, MBQI-Enum considers $\sigma = \{y \mapsto \lambda x.\ \mathrm{f}\,x\}$.

The quantified formula becomes $\exists z.\ (\lambda x.\ \mathrm{f}\,x)\,z \neq \mathrm{f}\,z$, i.e. $\exists z.\ \mathrm{f}\,z \neq \mathrm{f}\,z$.

The instantiation lemma

$$(\forall y.\ \exists z.\ y\,z \neq \mathrm{f}\,z) \Longrightarrow \exists z.\ \mathrm{f}\,z \neq \mathrm{f}\,z$$

is added to F.

After Skolemization, the quantifier-free subsolver finds a contradiction.

A Weakness of MBQI-Enum

MBQI-Enum does not try instantiations containing Hilbert's choice operator.

A Weakness of MBQI-Enum

MBQI-Enum does not try instantiations containing Hilbert's choice operator.

A choice term has the form

$$\varepsilon x.\ \varphi,$$

denoting some value of x satisfying φ, if one exists.

A Weakness of MBQI-Enum

MBQI-Enum does not try instantiations containing Hilbert's choice operator.

A choice term has the form

$$\varepsilon x.\ \varphi,$$

denoting some value of x satisfying φ, if one exists.

Hilbert's choice is characterized by the higher-order axiom

$$\forall y.\ (\exists x.\ y\,x) \Longrightarrow y\,(\varepsilon x.\ y\,x).$$

A Weakness of MBQI-Enum

MBQI-Enum does not try instantiations containing Hilbert's choice operator.

A choice term has the form

$$\varepsilon x.\ \varphi,$$

denoting some value of x satisfying φ, if one exists.

Hilbert's choice is characterized by the higher-order axiom

$$\forall y.\ (\exists x.\ y\,x) \Longrightarrow y\,(\varepsilon x.\ y\,x).$$

ε terms can be the right instantiations even when they do not occur in the input problem.

MBQI-Enum with Choice

Our strategy extends MBQI-Enum by:

- adding ε-terms to the grammar;
- introducing fresh Skolem symbols to represent these terms;
- filtering redundant or invalid candidates; and
- returning a substitution and also auxiliary lemmas characterizing the new Skolems.

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m. \psi,$$

the strategy proceeds as follows:

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m. \psi,$$

the strategy proceeds as follows:

1. Compute an initial substitution σ from the current model.

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi,$$

the strategy proceeds as follows:

1. Compute an initial substitution σ from the current model.
2. Build a grammar for each quantified variable y_i and augment it with terms of the form $\varepsilon x.\, \varphi$.

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi,$$

the strategy proceeds as follows:

1. Compute an initial substitution σ from the current model.
2. Build a grammar for each quantified variable y_i and augment it with terms of the form $\varepsilon x.\, \varphi$.
3. Enumerate candidates e from the grammar and set $\sigma' = \sigma[y_i \mapsto e]$.

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi,$$

the strategy proceeds as follows:

1. Compute an initial substitution σ from the current model.
2. Build a grammar for each quantified variable y_i and augment it with terms of the form $\varepsilon x.\, \varphi$.
3. Enumerate candidates e from the grammar and set $\sigma' = \sigma[y_i \mapsto e]$.
4. Update σ if $\neg\sigma'(\psi)$ is satisfiable in the current model.

The Strategy

For a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi,$$

the strategy proceeds as follows:

1. Compute an initial substitution σ from the current model.
2. Build a grammar for each quantified variable y_i and augment it with terms of the form $\varepsilon x.\, \varphi$.
3. Enumerate candidates e from the grammar and set $\sigma' = \sigma[y_i \mapsto e]$.
4. Update σ if $\neg\sigma'(\psi)$ is satisfiable in the current model.
5. Return σ together with lemmas that characterize the generated choice terms.

Example of MBQI-Enum with Choice

Let $\mathsf{f} : u \to u$ be an arbitrary function.

Proof goal: if f is injective, then it has a left inverse, that is,

$$(\forall x, y.\ \mathsf{f}\, x = \mathsf{f}\, y \Longrightarrow x = y) \Longrightarrow \exists g.\ \forall z.\ g\,(\mathsf{f}\, z) = z.$$

Example of MBQI-Enum with Choice

We consider the set $F = \{\forall x, y.\, \mathsf{f}\, x = \mathsf{f}\, y \Longrightarrow x = y,\ \forall g.\, \exists z.\, g\,(\mathsf{f}\, z) \neq z\}$.

Example of MBQI-Enum with Choice

We consider the set $F = \{\forall x, y.\ \mathsf{f}\, x = \mathsf{f}\, y \Longrightarrow x = y,\ \forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z\}$.

For $g : u \to u$, the grammar includes

$$
\begin{array}{lcl}
g_0 & ::= & \lambda y.\ g_1 \\
p_0 & ::= & \mathsf{true} \mid \mathsf{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0
\end{array}
\qquad
\begin{array}{lcl}
g_1 & ::= & \varepsilon x.\ p_0 \mid y \mid \mathsf{f}\, g_1 \\
p_1 & ::= & y \mid x \mid \mathsf{f}\, p_1
\end{array}
$$

Example of MBQI-Enum with Choice

We consider the set $F = \{\forall x, y.\ \mathsf{f}\, x = \mathsf{f}\, y \Longrightarrow x = y,\ \forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z\}$.

For $g : u \to u$, the grammar includes

$$
\begin{array}{llll}
g_0 & ::= & \lambda y.\ g_1 & \\
p_0 & ::= & \mathsf{true} \mid \mathsf{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 &
\end{array}
\qquad
\begin{array}{llll}
g_1 & ::= & \varepsilon x.\ p_0 \mid y \mid \mathsf{f}\, g_1 & \\
p_1 & ::= & y \mid x \mid \mathsf{f}\, p_1 &
\end{array}
$$

The strategy generates

$$\lambda y.\ \varepsilon x.\ y = \mathsf{f}\, x.$$

Example of MBQI-Enum with Choice

We consider the set $F = \{\forall x, y.\ \mathsf{f}\, x = \mathsf{f}\, y \Longrightarrow x = y,\ \forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z\}$.

For $g : u \to u$, the grammar includes

$$\begin{array}{llll} g_0 & ::= & \lambda y.\ g_1 & \\ p_0 & ::= & \mathsf{true} \mid \mathsf{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 & \end{array} \qquad \begin{array}{llll} g_1 & ::= & \varepsilon x.\ p_0 \mid y \mid \mathsf{f}\, g_1 & \\ p_1 & ::= & y \mid x \mid \mathsf{f}\, p_1 & \end{array}$$

The strategy generates

$$\lambda y.\ \varepsilon x.\ y = \mathsf{f}\, x.$$

The ε subterm is abstracted by a fresh Skolem $\mathsf{h} : u \to u$, yielding

$$\lambda y.\ \mathsf{h}\, y.$$

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\, x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\, y)$ lemma characterizing h

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\, x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\, y)$ lemma characterizing h

$(\forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z) \Longrightarrow \exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z$ from instantiating g in F with $\lambda y.\ \mathsf{h}\, y$

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\,x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\,y)$	lemma characterizing h
$(\forall g.\ \exists z.\ g\,(\mathsf{f}\,z) \neq z) \Longrightarrow \exists z.\ \mathsf{h}\,(\mathsf{f}\,z) \neq z$	from instantiating g in F with $\lambda y.\ \mathsf{h}\,y$
$(\exists z.\ \mathsf{h}\,(\mathsf{f}\,z) \neq z) \Longrightarrow \mathsf{h}\,(\mathsf{f}\,\mathsf{sk}) \neq \mathsf{sk}$	from Skolemizing the existential conclusion

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\, x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\, y)$	lemma characterizing h
$(\forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z) \Longrightarrow \exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z$	from instantiating g in F with $\lambda y.\ \mathsf{h}\, y$
$(\exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z) \Longrightarrow \mathsf{h}\,(\mathsf{f}\,\mathsf{sk}) \neq \mathsf{sk}$	from Skolemizing the existential conclusion
$\mathsf{f}\,\mathsf{sk} \neq \mathsf{f}\,\mathsf{sk} \vee \mathsf{f}\,\mathsf{sk} = \mathsf{f}\,(\mathsf{h}\,(\mathsf{f}\,\mathsf{sk}))$	from instantiating y with f sk

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\, x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\, y)$	lemma characterizing h
$(\forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z) \Longrightarrow \exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z$	from instantiating g in F with $\lambda y.\ \mathsf{h}\, y$
$(\exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z) \Longrightarrow \mathsf{h}\,(\mathsf{f}\,\mathsf{sk}) \neq \mathsf{sk}$	from Skolemizing the existential conclusion
$\mathsf{f}\,\mathsf{sk} \neq \mathsf{f}\,\mathsf{sk} \vee \mathsf{f}\,\mathsf{sk} = \mathsf{f}\,(\mathsf{h}\,(\mathsf{f}\,\mathsf{sk}))$	from instantiating y with f sk
$\mathsf{f}\,\mathsf{sk} \neq \mathsf{f}\,(\mathsf{h}\,(\mathsf{f}\,\mathsf{sk})) \vee \mathsf{sk} = \mathsf{h}\,(\mathsf{f}\,\mathsf{sk})$	from instantiating x and y in F with sk and h (f sk)

Example of MBQI-Enum with Choice

$\forall y.\ (\exists x.\ y = \mathsf{f}\, x) \Longrightarrow y = \mathsf{f}\,(\mathsf{h}\, y)$	lemma characterizing h
$(\forall g.\ \exists z.\ g\,(\mathsf{f}\, z) \neq z) \Longrightarrow \exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z$	from instantiating g in F with $\lambda y.\ \mathsf{h}\, y$
$(\exists z.\ \mathsf{h}\,(\mathsf{f}\, z) \neq z) \Longrightarrow \mathsf{h}\,(\mathsf{f}\,\mathsf{sk}) \neq \mathsf{sk}$	from Skolemizing the existential conclusion
$\mathsf{f}\,\mathsf{sk} \neq \mathsf{f}\,\mathsf{sk} \vee \mathsf{f}\,\mathsf{sk} = \mathsf{f}\,(\mathsf{h}\,(\mathsf{f}\,\mathsf{sk}))$	from instantiating y with f sk
$\mathsf{f}\,\mathsf{sk} \neq \mathsf{f}\,(\mathsf{h}\,(\mathsf{f}\,\mathsf{sk})) \vee \mathsf{sk} = \mathsf{h}\,(\mathsf{f}\,\mathsf{sk})$	from instantiating x and y in F with sk and h (f sk)

Augmented Grammar

For a variable of function type y_i, the grammar contains rules such as

$$g_0 ::= \lambda x_1, \ldots, x_n.\, g_1 \qquad g_1 ::= x_1 \mid \cdots \mid x_n$$

Augmented Grammar

For a variable of function type y_i, the grammar contains rules such as

$$g_0 ::= \lambda x_1, \ldots, x_n.\, g_1 \qquad g_1 ::= x_1 \mid \cdots \mid x_n$$

We introduce a fresh variable x of the return type of y_i and build a predicate grammar

$$p_0 ::= \text{true} \mid \text{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 \qquad p_1 ::= x_1 \mid \cdots \mid x_n \mid x$$

Augmented Grammar

For a variable of function type y_i, the grammar contains rules such as

$$g_0 ::= \lambda x_1, \ldots, x_n.\, g_1 \qquad g_1 ::= x_1 \mid \cdots \mid x_n$$

We introduce a fresh variable x of the return type of y_i and build a predicate grammar

$$p_0 ::= \text{true} \mid \text{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 \qquad p_1 ::= x_1 \mid \cdots \mid x_n \mid x$$

We add the production

$$g_1 ::= \varepsilon x.\, p_0.$$

Augmented Grammar

For a variable of function type y_i, the grammar contains rules such as

$$g_0 ::= \lambda x_1, \ldots, x_n.\, g_1 \qquad g_1 ::= x_1 \mid \cdots \mid x_n$$

We introduce a fresh variable x of the return type of y_i and build a predicate grammar

$$p_0 ::= \text{true} \mid \text{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 \qquad p_1 ::= x_1 \mid \cdots \mid x_n \mid x$$

We add the production

$$g_1 ::= \varepsilon x.\, p_0.$$

The final extended grammar is

$$\begin{array}{llll} g_0 & ::= & \lambda x_1, \ldots, x_n.\, g_1 & \qquad g_1 ::= \varepsilon x.\, p_0 \mid x_1 \mid \cdots \mid x_n \\ p_0 & ::= & \text{true} \mid \text{false} \mid p_1 = p_1 \mid \neg p_0 \mid p_0 \wedge p_0 & \qquad p_1 ::= x_1 \mid \cdots \mid x_n \mid x \end{array}$$

Skolem Symbols

Consider a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi$$

and a substitution

$$\{y_i \mapsto \lambda x_1, \ldots, x_n.\, t\}.$$

Skolem Symbols

Consider a quantified formula

$$\forall y_1, \ldots, y_m.\, \psi$$

and a substitution

$$\{y_i \mapsto \lambda x_1, \ldots, x_n.\, t\}.$$

Suppose t contains an ε subterm

$$\varepsilon x.\, \varphi(x_1, \ldots, x_n, x),$$

Skolem Symbols

Consider a quantified formula

$$\forall y_1, \ldots, y_m.\ \psi$$

and a substitution

$$\{y_i \mapsto \lambda x_1, \ldots, x_n.\ t\}.$$

Suppose t contains an ε subterm

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

where the free variables are

$$x_1, \ldots, x_n.$$

Skolem Symbols

Consider a quantified formula

$$\forall y_1, \ldots, y_m.\ \psi$$

and a substitution

$$\{y_i \mapsto \lambda x_1, \ldots, x_n.\ t\}.$$

Suppose t contains an ε subterm

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

where the free variables are

$$x_1, \ldots, x_n.$$

The goal is to abstract this ε term into a form that the SMT solver can reason about.

Skolem Symbols

For each generated subterm of the form

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

Skolem Symbols

For each generated subterm of the form

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

we introduce a fresh Skolem symbol h and replace the subterm by $\mathsf{h}\,(x_1, \ldots, x_n)$.

Skolem Symbols

For each generated subterm of the form

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

we introduce a fresh Skolem symbol h and replace the subterm by $\mathsf{h}\,(x_1, \ldots, x_n)$.

We construct the lemma

$$\forall x_1, \ldots, x_n.\ (\exists x.\ \varphi(x_1, \ldots, x_n, x)) \Longrightarrow \varphi(x_1, \ldots, x_n, \mathsf{h}\,(x_1, \ldots, x_n)).$$

Skolem Symbols

For each generated subterm of the form

$$\varepsilon x.\ \varphi(x_1, \ldots, x_n, x),$$

we introduce a fresh Skolem symbol h and replace the subterm by $\mathsf{h}\,(x_1, \ldots, x_n)$.

We construct the lemma

$$\forall x_1, \ldots, x_n.\ (\exists x.\ \varphi(x_1, \ldots, x_n, x)) \Longrightarrow \varphi(x_1, \ldots, x_n, \mathsf{h}\,(x_1, \ldots, x_n)).$$

The rewritten term is used in the substitution.

Term Filtering

Redundant candidates are discarded, so equivalent terms are considered only once.

For $\varepsilon x.\ \varphi$, the bound variable x must occur in φ; otherwise, the term is excluded.

cvc5 already performs extensive theory-specific filtering.

Evaluation

System	Sat	Unsat	Total	Unknown	Timeouts
Satallax					
Vampire					
Zipperposition					
cvc5 + MBQI-Enum					
cvc5 + MBQI-Enum + choice					

- We evaluated on 3757 higher-order problems from TPTP with a 60s timeout.

Evaluation

System	Sat	Unsat	Total	Unknown	Timeouts
Satallax	196	2162	2358	15	1384
Vampire	14	**2303**	2317	16	1424
Zipperposition	0	2083	2083	0	1674
cvc5 + MBQI-Enum	**204**	2155	2359	158	1240
cvc5 + MBQI-Enum + choice	**204**	2178	**2382**	154	1221

- We evaluated on 3757 higher-order problems from TPTP with a 60s timeout.
- Our approach solves the largest number of problems (2382), 24 more than the next-best prover.

Evaluation

System	Sat	Unsat	Total	Unknown	Timeouts
Satallax	196	2162	2358	15	1384
Vampire	14	**2303**	2317	16	1424
Zipperposition	0	2083	2083	0	1674
cvc5 + MBQI-Enum	**204**	2155	2359	158	1240
cvc5 + MBQI-Enum + choice	**204**	2178	**2382**	154	1221

- We evaluated on 3757 higher-order problems from TPTP with a 60s timeout.
- Our approach solves the largest number of problems (2382), 24 more than the next-best prover.
- Compared with the baseline, our approach solves 23 additional unsatisfiable benchmarks.

Evaluation

System	Sat	Unsat	Total	Unknown	Timeouts
Satallax	196	2162	2358	15	1384
Vampire	14	**2303**	2317	16	1424
Zipperposition	0	2083	2083	0	1674
cvc5 + MBQI-Enum	**204**	2155	2359	158	1240
cvc5 + MBQI-Enum + choice	**204**	2178	**2382**	154	1221

- We evaluated on 3757 higher-order problems from TPTP with a 60s timeout.
- Our approach solves the largest number of problems (2382), 24 more than the next-best prover.
- Compared with the baseline, our approach solves 23 additional unsatisfiable benchmarks.
- TPTP contains very few problems needing choice; most are choice-free, so the effect of choice support is hard to assess.

Conclusion

We extended MBQI-Enum with support for Hilbert's choice.

This makes it possible to solve problems containing choice,
but also problems where the right instantiation is an ε term.

We implemented the approach in the SMT solver cvc5.

Future Work

A main limitation is that useful ε-terms often appear too late in enumeration.

Since enumeration proceeds by increasing term size, these terms may be too complex.

Better enumeration heuristics could help generate the right terms earlier.

Enumerating Choice Terms in Model-Based Quantifier Instantiation

Lydia Kondylidou[1] Andrew Reynolds[2] Jasmin Blanchette[1] Cesare Tinelli[2]

[1]Ludwig-Maximilians-Universität München
[2]University of Iowa