This Talk

About refinement intersection type systems that refute judgements of other type systems.

\[ \not \vdash M : \tau \]

\[ \iff \vdash M : \neg \tau \]
Background

Refinement intersection type systems are the basis for

- model checkers of higher-order model checking (cf. [Kobayashi 09] [Broadbent&Kobayashi 11] [Ramsay+ 14]),
- software model-checker for higher-order programs (cf. MoCHi [Kobayashi+ 11]).

In those type systems,

- a derivation gives a witness of derivability,
- but nothing witnesses that a given derivation is not derivable.
Motivation

A witness of underivability would be useful for:

- a compact representation of an error trace
- an efficient model-checker in collaboration with the affirmative system
  - Cf. [Ramsay+ 14] [Godefroid+ 10]
- development of a type system proving safety
  - In some cases (e.g. [T&Kobayashi 14]), a type system proving failure is easier to be developed.
Contribution

Development of type systems refuting derivability in some type systems such as

- a basic type system for the $\lambda$-calculus
- a type system for call-by-value reachability

Theoretical study of the development
Outline

• Reviewing a refinement intersection type system for higher-order model checking

• Negative type system

• Extensions

• Discussions
A simply typed calculus equipped with $\beta\eta$-equivalence.

Kinds (i.e. simple types):

$$A, B ::= o \mid A \to A$$

Terms:

$$M, N ::= x \mid \lambda x^A.M \mid M\,M$$

Typing rules:

$$\begin{align*}
\frac{(x :: A) \in \Delta}{\Delta \vdash x :: A} & \quad \frac{\Delta, x :: A \vdash M :: B}{\Delta \vdash \lambda x^A.M :: A \to B} \\
\frac{\Delta \vdash M :: A \to B \quad \Delta \vdash N :: A}{\Delta \vdash MN :: B}
\end{align*}$$
Refinement types

Types are parameterised by kinds and ground type sets:

\[ Ty_Q(o) := Q \]
\[ Ty_Q(A \rightarrow B) := \mathcal{P}(Ty_Q(A)) \times Ty_Q(B) \]

We use the following syntax for types:

\[ \tau, \sigma ::= q \mid X \rightarrow \tau \]
\[ X, Y \in \mathcal{P}(Ty_Q(A)) \]
Alternative definition

Let $A$ be a kind.

The set $\text{Ty}_Q(A)$ of types that refines $A$ is given by

$$\text{Ty}_Q(A) = \{ \tau \mid \tau :: A \}$$

where is the refinement relation:

$$q \in Q \quad \frac{\forall \sigma \in X. \sigma :: A \quad \tau :: B}{q :: o \quad (X \to \tau) :: A \to B}$$
The subtyping relation is defined by induction on kinds.

\[ q \leq_o q \]

\[ X \geq_{!A} Y \quad \tau \leq_{B} \sigma \]
\[ \frac{\quad (X \rightarrow \tau) \leq_{A\rightarrow B} (Y \rightarrow \sigma) \quad}{\forall \sigma \in Y. \exists \tau \in X. \tau \leq_{A} \sigma} \]

\[ X \leq_{!A} Y \]
Type Environments

A (finite) map from variables to sets of types (or intersection types).

\[ \Gamma ::= x_1 : X_1, \ldots, x_n : X_n \quad (n \geq 0) \]
Typing rules

\[(x : X) \in \Gamma \quad \tau \in X \quad \tau \preceq \sigma\]
\[
\Gamma \vdash x : \sigma
\]

\[
\Gamma, x : X \vdash M : \tau
\]
\[
\Gamma \vdash \lambda x.M : X \rightarrow \tau
\]

\[
\Gamma \vdash M : X \rightarrow \tau \quad \Gamma \vdash N : X
\]
\[
\Gamma \vdash MN : \tau
\]

\[
\forall \tau \in X. \Gamma \vdash M : \tau
\]
\[
\Gamma \vdash M : X
\]
Fact: Invariance under $\beta\eta$-equivalence

Suppose that $M =_{\beta\eta} N$. Then

$$\Gamma \vdash M : \tau \iff \Gamma \vdash N : \tau$$

• This fact will not be used in the sequel.
Convention: Subtyping closure

In what follows, sets of types are assumed to be closed under the subtyping relation.

\[ \tau \geq \sigma \in X \implies \tau \in X \]

The rule for variables becomes simpler.

\[
\frac{(x : X) \in \Gamma \quad \tau \in X}{\Gamma \vdash x : \tau}
\]
Outline

• Reviewing a refinement intersection type system
  • Negative type system
• Extensions
• Discussions
Negative Types

Negative types are those constructed from the negative ground types \( \overline{Q} := \{ \bar{q} \mid q \in Q \} : \)

\[
\overline{\mathrm{Ty}}_Q(A) := \mathrm{Ty}_{\overline{Q}}(A)
\]

\( \bar{\tau}, \bar{\sigma} ::= \bar{q} \mid \bar{X} \rightarrow \bar{\tau} \)

\( \bar{X}, \bar{Y} \in u(\overline{\mathrm{Ty}}_\overline{Q}(A)) \)

Typing rules are the same as the affirmative system.
Goal and approach

Giving an anti-monotone bijections on prime types

\[ \neg_A : \text{Ty}_Q(A) \rightarrow \overline{\text{Ty}_Q(A)} \]

such that, for every term \( M :: A \),

\[ \not\exists M : \tau \iff \vdash M : \neg_A \tau \]

This implies that

\[ \not\exists M. (\vdash M : \tau) \& (\not\vdash M : \neg \tau) \]

We shall first study this relation.
(In)consistency \ cf. \ [Salvati \ & \ Walukiewicz \ 2011] 

(Intuitively) \ \tau \in \text{Ty}_Q(A) \ \text{and} \ \bar{\sigma} \in \overline{\text{Ty}_Q(A)} \ \text{are} \ \textbf{consistent} \ \text{if} \\
\exists d \in A. \ (d \models \tau) \ \& \ (d \models \bar{\sigma}) 

\text{and} \ \textbf{inconsistent} \ \text{otherwise.} 

\textbf{Inference rules} \\
\tau \parallel \bar{\sigma} \ \iff \ \tau \ \text{and} \ \bar{\sigma} \ \text{are consistent} \\
\tau \preceq \bar{\sigma} \ \iff \ \tau \ \text{and} \ \bar{\sigma} \ \text{are inconsistent} \\

\frac{q \in Q \ \exists \tau \in X. \exists \bar{\sigma} \in \bar{Y} \cdot \tau \preceq \bar{\sigma}}{q \preceq \bar{q}} \\
\frac{\exists \tau \in X. \exists \bar{\sigma} \in \bar{Y} \cdot \tau \preceq \bar{\sigma}}{X \preceq \bar{Y}} \\
\frac{X \parallel \bar{Y}}{(X \rightarrow \tau) \preceq (\bar{Y} \rightarrow \bar{\sigma})} \\
\neg (\tau \preceq \bar{\sigma}) \\
\tau \parallel \bar{\sigma}
(In)consistency  cf. [Salvati & Walukiewicz 2011]

(Intuitively) \( \tau \in Ty_Q(A) \) and \( \bar{\sigma} \in \overline{Ty_Q(A)} \) are **consistent** if

\[
\exists d \in A. \ (d \models \tau) \ \& \ (d \models \bar{\sigma})
\]

and **inconsistent** otherwise.

Inference rules

\[
\begin{align*}
\tau \parallel \bar{\sigma} & \iff \tau \text{ and } \bar{\sigma} \text{ are consistent} \\
\tau \preceq \bar{\sigma} & \iff \tau \text{ and } \bar{\sigma} \text{ are inconsistent}
\end{align*}
\]

\[
\begin{align*}
q \in Q \quad & \exists \tau \in X. \exists \bar{\sigma} \in \overline{Y}. \tau \preceq \bar{\sigma} \\
q \preceq \bar{q} \quad & X \parallel \overline{Y} \\
\tau \preceq \bar{\sigma} \quad & (X \rightarrow \tau) \preceq (\overline{Y} \rightarrow \bar{\sigma})
\end{align*}
\]

Assume \( \exists f. \ (f \models X \rightarrow \tau) \ \& \ (f \models \overline{Y} \rightarrow \bar{\sigma}) \)

Then \((f(d) \models \tau) \ \& \ (f(d) \models \bar{\sigma})\), contradiction
Negation is weakest inconsistent type

Recall that

\[ \not\exists \ M : \tau \iff \vdash M : \neg \neg \tau \]

- **[Inconsistent]**
  We have \( \tau \preceq \neg \tau \)

- **[Weakest]**
  Assume that \( \tau \preceq \bar{\sigma} \). Then

    \[ \vdash M : \bar{\sigma} \quad \Rightarrow \quad \not\exists \ M : \tau \]

    \[ \quad \Rightarrow \quad \vdash M : \neg \tau \]
Negation is weakest inconsistent type

Recall that

\[ \forall M : \tau \iff \vdash M : \neg_A \tau \]

- **[Inconsistent]**
  We have \( \tau \preceq \neg \tau \)

- **[Weakest]**
  Assume that \( \tau \preceq \overline{\sigma} \). Then

  \[ \vdash M : \overline{\sigma} \Rightarrow \forall M : \tau \]

  \[ \Rightarrow \vdash M : \neg \tau \]
Definition of the negation

Define the two anti-monotone bijections on types as follows:

\[ \neg_A : Ty_Q(A) \longrightarrow Ty_Q(A) \]

\[ \models_A : u(Ty_Q(A)) \longrightarrow u(Ty_Q(A)) \]

as follows:

\[ \neg_o q := \bar{q} \]

\[ \neg_{A \rightarrow B}(X \rightarrow \tau) := (\models_A X) \rightarrow (\neg_B \tau) \]

\[ \models_A X := \{ \neg_A \tau \mid \tau \notin X \} \]
Definition of the negation

Define the two anti-monotone bijections on types as follows:

$\neg_A : \overline{\text{Ty}_Q(A)} \rightarrow \overline{\text{Ty}_Q(A)}$

$\exists_A : \overline{u(\text{Ty}_Q(A))} \rightarrow \overline{u(\text{Ty}_Q(A))}$

as follows:

$\neg_{oq} := \overline{q}$

$\neg_{A \rightarrow B}(X \rightarrow \tau) := (\exists_A X) \rightarrow (\neg_B \tau)$

$\exists_A X := \{ \neg_A \tau \mid \tau \not\in X \}$
Negation $\neg_A : \text{Ty}_Q(A) \rightarrow \overline{\text{Ty}_Q(A)}$

$\text{Ty}_Q(o)$

$q_1$ \quad $\neg_o$ \quad $\overline{\neg_o}$ \quad $\overline{q}_1$
$q_2$ \quad $\neg_o$ \quad $\overline{\neg_o}$ \quad $\overline{q}_2$
$q_3$ \quad $\neg_o$ \quad $\overline{\neg_o}$ \quad $\overline{q}_3$

$\text{Ty}_Q(o) = \text{Ty}_{\overline{Q}}(o)$
Natural \[ 
\ll_A : u(T_y^Q(A)) \rightarrow u(T_{y_Q}(A)) 
\]

\[
\begin{array}{c}
\text{T}_y^Q(o) \\
q_1 \\
q_2 \\
q_3 \\
\end{array} \\
\begin{array}{c}
\text{T}_{y_Q}(o) = \text{T}_{y_{\overline{Q}}}(o) \\
\overline{q}_1 \\
\overline{q}_2 \\
\overline{q}_3 \\
\end{array}
\]
Natural

\[ \llbracket_A : u(Ty_Q(A)) \rightarrow u(\overline{Ty_Q(A)}) \]

\[ Ty_Q(o) = Ty_{\overline{Q}}(o) \]

- \( q_1 \)
- \( q_2 \)
- \( q_3 \)

- \( \overline{q}_1 \)
- \( \overline{q}_2 \)
- \( \overline{q}_3 \)
Natural

\[ \exists_A : u(Ty_Q(A)) \rightarrow u(\overline{Ty_Q(A)}) \]

\[ Ty_Q(A) = Ty_{\overline{Q}}(A) \]

\[ \begin{align*}
\tau_1 \\
\tau_2 \\
\vdots \\
\tau_n
\end{align*} \]
Definition of the negation

Define the two anti-monotone bijections on types as follows:

\[ \neg_A : \text{Ty}_Q(A) \rightarrow \overline{\text{Ty}_Q(A)} \]

\[ \forall A : u(\text{Ty}_Q(A)) \rightarrow u(\overline{\text{Ty}_Q(A)}) \]

as follows:

\[ \neg_o q := \overline{q} \]

\[ \neg_{A \rightarrow B}(X \rightarrow \tau) := (\forall A X) \rightarrow (\neg_B \tau) \]

\[ \forall A X := \{ \neg_A \tau \mid \tau \notin X \} \]
\( \neg(X \rightarrow \tau) \) is weakest inconsistent type

\[ \neg_{A \rightarrow B}(X \rightarrow \tau) := (\ulcorner A X \urcorner) \rightarrow (\neg_B \tau) \]

### a) inconsistent

#### Strongest consistent

\[
\begin{align*}
    X &\parallel \ulcorner X \\
    \tau &\simeq \neg \tau \\
    (X \rightarrow \tau) &\simeq (\ulcorner X \rightarrow \neg \tau)
\end{align*}
\]

#### Weakest inconsistent

### b) weakest

Assume \((X \rightarrow \tau) \simeq (\bar{Y} \rightarrow \bar{\sigma})\)

Then \(X \parallel \bar{Y}\) and \(\tau \simeq \bar{\sigma}\). So

\[
\begin{align*}
    \bar{Y} &\geq \ulcorner X \\
    \bar{\sigma} &\leq \neg \tau \\
    (\bar{Y} \rightarrow \bar{\sigma}) &\leq (\ulcorner X \rightarrow \neg \tau)
\end{align*}
\]

#### Strongest consistent

#### Weakest inconsistent
Main Theorem

Theorem

• $\Gamma \not\models M : \tau$ if and only if $\models \Gamma \vdash M : \neg \tau$, where $\models (x_1 : X_1, \ldots, x_n : X_n) := x_1 : (\models X_1), \ldots, x_n : (\models X_n)$

• Let $X = \{ \tau \mid \Gamma \vdash M : \tau \}$. Then

$$\models \Gamma \vdash M : \models X$$

Proof) By mutual induction on the structure of the term.
Main Theorem

For a closed term $M$, 

$$\not\vdash M : \tau \iff \vdash M : \neg \tau$$

Theorem

- $\Gamma \not\vdash M : \tau$ if and only if $\models \Gamma \vdash M : \neg \tau$, where $\models (x_1 : X_1, \ldots, x_n : X_n) := x_1 : (\models X_1), \ldots, x_n : (\models X_n)$

- Let $X = \{ \tau | \Gamma \vdash M : \tau \}$. Then 

$$\models \Gamma \vdash M : \models X$$

Proof) By mutual induction on the structure of the term.
Main Theorem

Theorem

• \( \Gamma \not\models M : \tau \) if and only if \( \not\models \Gamma \vdash M : \neg \tau \),
  where \( \langle x_1 : X_1, \ldots, x_n : X_n \rangle := x_1 : (\langle X_1 \rangle), \ldots, x_n : (\langle X_n \rangle) \)

• Let \( \mathcal{X} = \{ \tau \mid \Gamma \vdash M : \tau \} \). Then

\[ \not\models \Gamma \vdash M : \not\models X \]

\( \Gamma \vdash M : X \) iff \( \not\models \Gamma \vdash M : \not\models X \)

under a certain condition
Remark

Only **prime type judgements** have negations

\[ \Gamma \nvdash M : \tau \iff \not\exists \Gamma \vdash M : \neg \tau \]

**Prime type**

Negation of an intersection type judgement needs **meta-level union**

\[ \Gamma \nvdash M : \bigwedge X \iff \exists \tau \in X. \not\exists \Gamma \vdash M : \neg \tau \]

**Intersection type**
Outline

• Reviewing a refinement intersection type system
• Negation of the type system
• Extensions
  • Additional constants (e.g. recursion)
  • Categorical formalisation
• Discussions
\( \lambda \rightarrow + \text{ constants} \)

\[
M, N ::= x | \lambda x^A.M | MM | c
\]

It is easy to handle additional constants provided that we have an affirmative type system.

**Affirmative side**

\[
\frac{\tau \in T_c}{\Gamma \vdash c : \tau}
\]

**Negative side**

\[
\frac{\bar{\sigma} \in \text{\#}T_c}{\bar{\Delta} \vdash c : \bar{\sigma}}
\]

The set of prime types for \( c \)
Example: recursion

Target language: \[ M, N ::= x \mid \lambda x^A.M \mid M \, M \mid Y_A \]

Affirmative side

\[
\exists (Y \to \tau) \in X. \forall \sigma \in Y. \Gamma \vdash Y : X \to \sigma \\
\Gamma \vdash Y : X \to \tau
\]

coinductive

Negative side

\[
\bar{\sigma} \in \exists \{ \theta \mid \Gamma \vdash Y : \theta \} \\
\Gamma \vdash Y : \bar{\sigma}
\]
Example: recursion

Target language: \[ M, N ::= x \mid \lambda x^A.M \mid M M \mid Y_A \]

Affirmative side

\[ \exists (Y \to \tau) \in X. \forall \sigma \in Y. \Gamma \vdash Y : X \to \sigma \]

\[ \Gamma \vdash Y : X \to \tau \]

coinductive

Negative side

Equivalent to the inductive version of the above rule

\[ \bar{\sigma} \in \{ \theta \mid \Gamma \vdash Y : \theta \} \]

\[ \bar{\Delta} \vdash Y : \bar{\sigma} \]
Outline

• Reviewing a refinement intersection type system
• Negation of the type system

• Extensions
  • Additional constants (e.g. recursion)
  • Categorical formalisation

• Discussions
## Semantics of terms via type system

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Kind</strong></td>
<td><strong>Poset</strong></td>
</tr>
<tr>
<td>$A$</td>
<td>$(\text{Ty}_Q(A)/\simeq, \leq)$</td>
</tr>
<tr>
<td><strong>Term</strong></td>
<td><strong>Relation</strong></td>
</tr>
<tr>
<td>$x :: A \vdash M :: B$</td>
<td>${(X, \tau) \mid x : X \vdash M : \tau}$</td>
</tr>
</tbody>
</table>
Category $\text{ScottL}_u$

**Definition**  
The category $\text{ScottL}_u$ is given by:

**Object**  
Poset $(A, \leq_A)$.

**Morphism**  
An upward-closed relation

\[ R \subseteq u(A)^{op} \times B \]

**Composition**  
Let

\[ R \subseteq u(A)^{op} \times B \]

\[ S \subseteq u(B)^{op} \times C \]

Then

\[ \exists Y \in u(B). \left( \forall b \in Y. (X, b) \in R \text{ and } (Y, c) \in S \right) \]

\[ (X, c) \in (S \circ R) \]
Interpretation of CbN $\lambda \to$ in $\text{ScottL}_u$

**Fact** $\text{ScottL}_u$ is a cartesian closed category.

Interpretation of kinds is given by:

\[
\begin{align*}
[\circ]_Q & := (Q, =) \\
[A \to B]_Q & := u([A]_Q)^{op} \times [B]_Q
\end{align*}
\]

Hence $[A]_Q \cong \text{Ty}_Q(A)$.

**Fact** (see e.g. [Terui 2012])

\[\Gamma \vdash M : \tau \iff (\Gamma, \tau) \in [M]\]
Negation Functor on $\text{ScottL}_u$

The functor $\varphi : \text{ScottL}_u \to \text{ScottL}_u$ is defined by:

\[
\varphi(A) := A^{op}
\]

\[
\varphi(R) := \{(A\setminus X, b) \in u(A)^{op} \times B \mid (X, b) \notin R\}
\]

**Lemma** $\varphi$ is an isomorphism on $\text{ScottL}_u$.

If $R \in u(A)^{op} \times B$ and $A = \emptyset$, then

\[
\varphi(R) = \{(\emptyset, b) \mid (\emptyset, b) \notin R\}
\]

which is essentially the complement of $R$. 


Applications

Negation $\varphi : \text{ScottL}_u \xrightarrow{\sim} \text{ScottL}_u$

- A type system witnessing call-by-value reachability [T&Kobayashi 14] is the Kleisli category of a monad $T : \text{ScottL}_u \rightarrow \text{ScottL}_u$

  Then

  $\varphi T \varphi^{-1} : \text{ScottL}_u \rightarrow \text{ScottL}_u$

  is also a monad. We can lift the negation to

  $\varphi : (\text{ScottL}_u)_T \rightarrow (\text{ScottL}_u)_{\varphi T \varphi^{-1}}$

A type system proving unreachability
Applications

Negation \( \varphi : \text{ScottL}_u \xrightarrow{\cong} \text{ScottL}_u \)

• A type system for higher-order model checking [Kobayashi\&Ong 09] is coKleisli category of a comonad

\( \Box : \text{ScottL}_u \rightarrow \text{ScottL}_u \) \hspace{1cm} [Grellois\&Melliès 14]

Then

\( \varphi \Box \varphi^{-1} : \text{ScottL}_u \rightarrow \text{ScottL}_u \)

is also a monad. We can lift the negation to

\( \varphi : (\text{ScottL}_u)\Box \rightarrow (\text{ScottL}_u)\varphi \Box \varphi^{-1} \)

Essentially the same as [Kobayashi\&Ong 09]
Outline

• Reviewing a refinement intersection type system
• Negative type system
• Extensions
• Discussions
Automata complementation

Corresponds to negation of a 2nd-order judgement.
Let $A$ be a kind and $B_A$ be the set of all Böhm trees of type $A$. A language is a subset of $B_A$.

**Definition** A language $L \subseteq B_A$ is type-definable if there exists a type $\tau$ such that

$$L = \{ M \in B_A \mid \vdash M : \tau \}$$

in the type system for higher-order model checking [Kobayashi&Ong 09] [T&Ong 14].

**Corollary** The class of type-definable languages are closed under Boolean operations on sets.
Related Work

"Krivine machines and higher-order schemes"
[Salvati&Walkiewicz 12]

• The notion of consistency and inconsistency can be found in their work (called *complementarity* for the former and the latter has no name).
• This talk is partially inspired by their work.
Conclusion

Negation is a definable operation in the refinement intersection type system for the call-by-name $\lambda \rightarrow$.

This observation leads to the construction of negative type systems for other refinement type systems, e.g.,

- call-by-name $\lambda \rightarrow$ + recursion
- the type system for HOMC
- a type system for a call-by-value language

Application to verification needs some work.