Preciseness of Subtyping: from extensional to intensional aspects

Silvia Ghilezan

University of Novi Sad
Mathematical Institute SANU
Serbia

NII Shonan Meeting 115
Intensional and extensional aspects of computation
January 22-25, 2018

Joint work with

- Mariangiola Dezani-Ciacaglini
- Nobuko Yoshida
- Jovanka Pantović
- Svetlana Jakšić
- Alceste Scalas

Subtyping

Subtyping is a binary relation $\leq$ (preorder) on the set of `Types`

$$\sigma \leq \tau$$

Subsumption rule in the type inference system

$$\frac{M : \sigma \quad \sigma \leq \tau}{M : \tau}$$

- λ-calculi, concurrent calculi
- programming languages

1. Soundness and completeness

2. Concurrent λ-calculus

3. Preciseness Results - a roadmap

4. Preciseness - language independent

5. Conclusion

Preciseness of subtyping

Preciseness

- Soundness
- Completeness

Two aspects:

- Denotational preciseness
- Operational preciseness

Denotational Preciseness of Subtyping

$[\![\sigma]\!]$ is a set interpreting type σ

denotational soundness: $\sigma \leq \tau$ implies $[\![\sigma]\!] \subseteq [\![\tau]\!]$

denotational completeness: $[\![\sigma]\!] \subseteq [\![\tau]\!]$ implies $\sigma \leq \tau$

denotational preciseness: $\sigma \leq \tau$ iff $[\![\sigma]\!] \subseteq [\![\tau]\!]$

H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini.
A Filter Lambda Model and the Completeness of Type Assignment.
Journal of Symbolic Logic, 48(4):931–940, 1983.

J. Vouillon.
Subtyping Union Types.
In *CSL*, volume 3210 of *LNCS*, pages 415–429, 2004.

Operational Soundness of Subtyping

If $\sigma \leq \tau$, **then** each context

- that is safe when filled with a term of type τ is also safe when filled with a term of type σ

$$\forall C[\,]\ (\forall M:\tau\ C[M] \not\to^{*} \texttt{error} \implies \forall N:\sigma\ C[N] \not\to^{*} \texttt{error})$$

Example. $\texttt{nat} \leq \texttt{int}$ $C[-5]$ converges, then $C[2]$ converges

Operational soundness of subtyping follows from subject reduction of the type system with the subsumption rule

Operational Completeness of Subtyping

Converse:
If each context that is safe when filled with a term of type τ is also safe when filled with a term of type σ, **then** $\sigma \leq \tau$

Instead:
If $\sigma \not\leq \tau$, **then** there is a context

- that is safe when filled with an arbitrary term of type τ, and
- gives an error when filled with a suitable term of type σ

$$\exists C_0[\,](\forall M : \tau.\ C_0[M] \not\to^* \texttt{error} \bigwedge \exists N_0 : \sigma. C_0[N_0] \to^* \texttt{error})$$

J. Blackburn, I. Hernandez, J. Ligatti, and M. Nachtigal.
On subtyping-relation completeness, with an application to iso-recursive types. ACM Trans. Program. Lang. Syst. 39 (1), 4:1–4:36, 2017 (2014).

Concurrent λ-calculus - Syntax

M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno.
A Filter Model for Concurrent Lambda-Calculus.
SIAM Journal on Computing 27(5):1376–1419, 1998.

$$M ::= x \mid v \mid (\lambda x.M) \mid (\lambda v.M) \mid (MM) \mid (M + M) \mid (M \| M)$$

1. call-by-name and call-by-value variables
2. internal choice
3. parallel operator

$$W ::= v \mid \lambda x.M \mid \lambda v.M \mid W \| W$$
$$V ::= W \mid V \| M \mid M \| V$$

TVal total values:
Val values

Reduction rules

$$(+_L)\ M + N \longrightarrow M \qquad (+_R)\ M + N \longrightarrow N$$

Reduction rules

$$(+_L)\ M + N \longrightarrow M \qquad (+_R)\ M + N \longrightarrow N$$

$$(\|_{app})\ (M\|N)L \longrightarrow ML\|NL \qquad (\|_s)\ \frac{M \longrightarrow M' \quad N \longrightarrow N'}{M\|N \longrightarrow M'\|N'}$$

$$(\|_a)\ \frac{M \longrightarrow M' \quad W \in \text{TVal}}{M\|W \longrightarrow M'\|W,\ W\|M \longrightarrow W\|M'}$$

TVal *total values*: $W ::= v \mid \lambda x.M \mid \lambda v.M \mid W\|W$

Reduction rules

$$(+_L)\ M + N \longrightarrow M \qquad (+_R)\ M + N \longrightarrow N$$

$$(\|_{app})\ (M\|N)L \longrightarrow ML\|NL \qquad (\|_s)\ \frac{M \longrightarrow M' \quad N \longrightarrow N'}{M\|N \longrightarrow M'\|N'}$$

$$(\|_a)\ \frac{M \longrightarrow M' \quad W \in \mathrm{TVal}}{M\|W \longrightarrow M'\|W,\ W\|M \longrightarrow W\|M'}$$

$$(\beta)\ (\lambda x.M)N \longrightarrow M[N/x] \qquad (\beta_v)\ \frac{W \in \mathrm{TVal}}{(\lambda v.M)W \longrightarrow M[W/v]}$$

$$(\beta_v\|)\ \frac{V \longrightarrow V' \quad V \in \mathrm{Val}}{(\lambda v.M)V \longrightarrow M[V/v]\|(\lambda v.M)V'}$$

TVal *total values*: $W ::= v \mid \lambda x.M \mid \lambda v.M \mid W\|W$
Val *values* $V ::= W \mid V\|M \mid M\|V$

Reduction rules

$$(+_L)\ M+N \longrightarrow M \qquad (+_R)\ M+N \longrightarrow N$$

$$(\|_{app})\ (M\|N)L \longrightarrow ML\|NL \qquad (\|_s)\ \frac{M \longrightarrow M' \quad N \longrightarrow N'}{M\|N \longrightarrow M'\|N'}$$

$$(\|_a)\ \frac{M \longrightarrow M' \quad W \in \mathrm{TVal}}{M\|W \longrightarrow M'\|W,\ W\|M \longrightarrow W\|M'}$$

$$(\beta)\ (\lambda x.M)N \longrightarrow M[N/x] \qquad (\beta_v)\ \frac{W \in \mathrm{TVal}}{(\lambda v.M)W \longrightarrow M[W/v]}$$

$$(\beta_v\|)\ \frac{V \longrightarrow V' \quad V \in \mathrm{Val}}{(\lambda v.M)V \longrightarrow M[V/v]\|(\lambda v.M)V'}$$

$$(\mu_v)\ \frac{N \longrightarrow N' \quad N \notin \mathrm{Val}}{(\lambda v.M)N \longrightarrow (\lambda v.M)N'} \qquad (\nu)\ \frac{M \longrightarrow M' \quad M \notin \mathrm{Val} \bigcup \mathrm{Par}}{MN \longrightarrow M'N}$$

TVal *total values*: $W ::= v \mid \lambda x.M \mid \lambda v.M \mid W\|W$
Val *values* $V ::= W \mid V\|M \mid M\|V$
$\mathrm{Par} = \{M\|N\}$

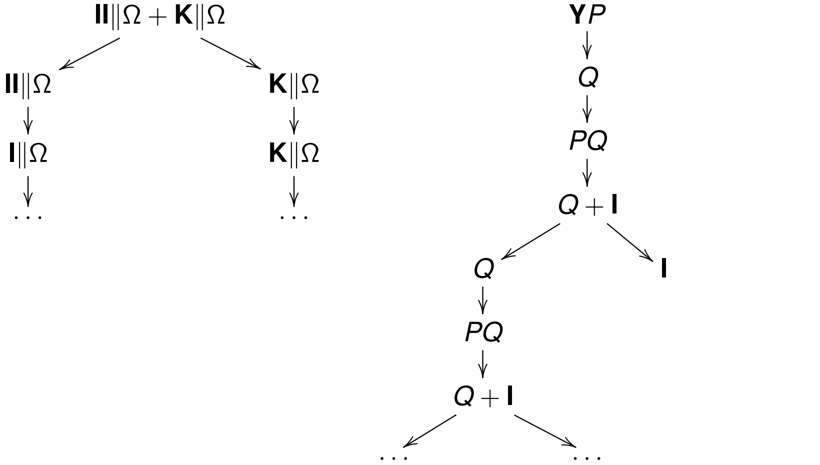
Convergence

reduction tree

$$
\begin{array}{ccc}
 & \mathbf{II}\|\Omega + \mathbf{K}\|\Omega & \\
\swarrow & & \searrow \\
\mathbf{II}\|\Omega & & \mathbf{K}\|\Omega \\
\downarrow & & \downarrow \\
\mathbf{I}\|\Omega & & \mathbf{K}\|\Omega \\
\downarrow & & \downarrow \\
\cdots & & \cdots
\end{array}
$$

Convergence

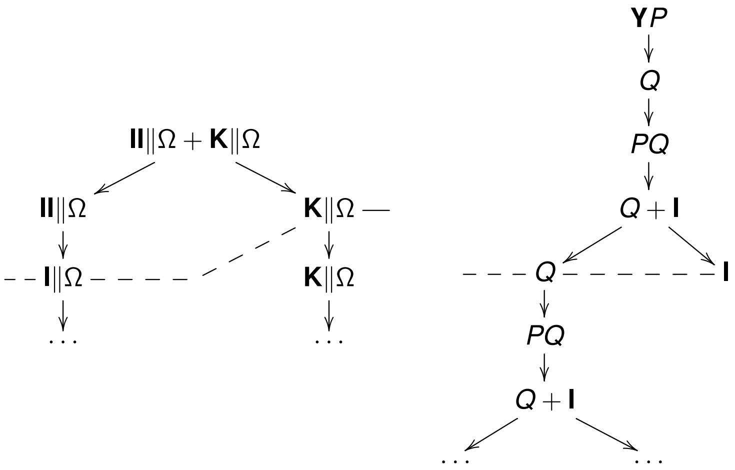
reduction tree $P = \lambda x.(x + \mathbf{I}) \quad Q = (\lambda x.P(xx))(\lambda x.P(xx))$



Convergence

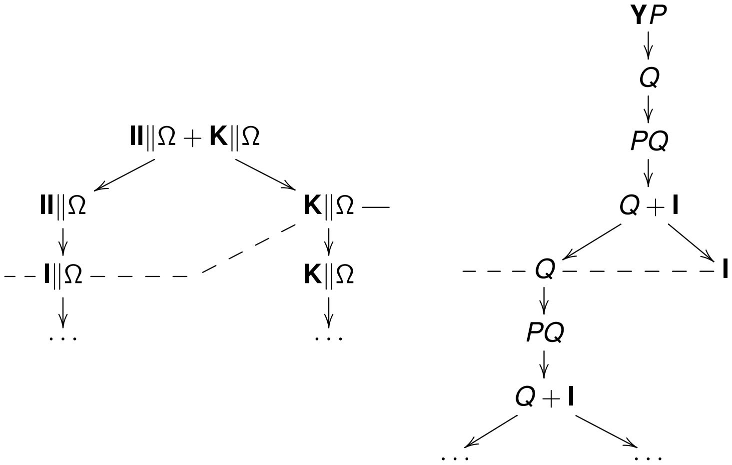
reduction tree $\qquad P = \lambda x.(x + \mathbf{I}) \quad Q = (\lambda x.P(xx))(\lambda x.P(xx))$

Bar is a subset of nodes of the reduction tree such that each maximal path intersects the bar at exactly one node



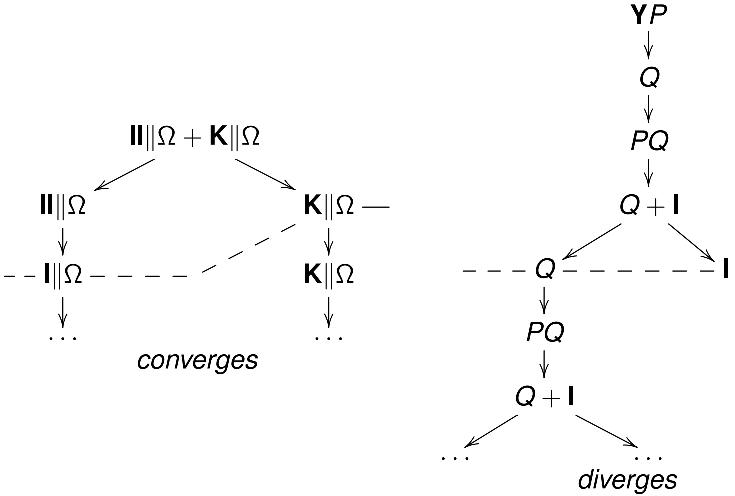
Convergence

a term converges if there is a bar of values in its reduction tree



Convergence

a term converges if there is a bar of values in its reduction tree



Types and Subtyping

$\texttt{Type}\colon \sigma ::= \omega \mid \sigma \to \sigma \mid \sigma \wedge \sigma \mid \sigma \vee \sigma$

$\sigma \leq \tau$ is the smallest pre-order on types such that

1. $\langle \texttt{Type}, \leq \rangle$ is a distributive lattice, in which $\wedge$ is the meet, $\vee$ is the join and ω is the top;
2. the arrow satisfies
   1. $\sigma \to \omega \leq \omega \to \omega$;
   2. $(\sigma \to \rho) \wedge (\sigma \to \tau) \leq \sigma \to \rho \wedge \tau$;
   3. $\sigma \geq \sigma', \tau \leq \tau' \Rightarrow \sigma \to \tau \leq \sigma' \to \tau'$.

NB

- $(\sigma \to \rho) \wedge (\tau \to \rho) \leq \sigma \vee \tau \to \rho$ is unsound in concurrent λ-calculus (SR would fail)!

Typing Rules

A basis Γ maps

1. call-by-name variables to types (ω by default) and
2. call-by-value variables to coprime types ($\omega \to \omega$ by default)

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

$$(\to \mathrm{I}_n)\ \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

$$(\to \mathrm{I}_v)\ \frac{\Gamma, v : \sigma_i \vdash M : \tau \quad \sigma = \bigvee_{i \in I} \sigma_i \quad \sigma_i \in \mathtt{CType}\ i \in I}{\Gamma \vdash \lambda v.M : \sigma \to \tau}$$

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

$$(\rightarrow \mathrm{I}_n)\ \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \rightarrow \tau}$$

$$(\rightarrow \mathrm{I}_v)\ \frac{\Gamma, v : \sigma_i \vdash M : \tau \quad \sigma = \bigvee_{i \in I} \sigma_i \quad \sigma_i \in \mathtt{CType}\ i \in I}{\Gamma \vdash \lambda v.M : \sigma \rightarrow \tau}$$

$$(\rightarrow \mathrm{E})\ \frac{\Gamma \vdash M : \sigma \rightarrow \tau\ \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

$$(\to \mathrm{I}_n)\ \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

$$(\to \mathrm{I}_v)\ \frac{\Gamma, v : \sigma_i \vdash M : \tau \quad \sigma = \bigvee_{i \in I} \sigma_i \quad \sigma_i \in \mathtt{CType}\ i \in I}{\Gamma \vdash \lambda v.M : \sigma \to \tau}$$

$$(\to \mathrm{E})\ \frac{\Gamma \vdash M : \sigma \to \tau\ \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

$$(\wedge \mathrm{I})\ \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash M : \tau}{\Gamma \vdash M : \sigma \wedge \tau}$$

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

$$(\to \mathrm{I}_n)\ \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}$$

$$(\to \mathrm{I}_v)\ \frac{\Gamma, v : \sigma_i \vdash M : \tau \quad \sigma = \bigvee_{i \in I} \sigma_i \quad \sigma_i \in \texttt{CType}\ i \in I}{\Gamma \vdash \lambda v.M : \sigma \to \tau}$$

$$(\to \mathrm{E})\ \frac{\Gamma \vdash M : \sigma \to \tau\ \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

$$(\wedge \mathrm{I})\ \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash M : \tau}{\Gamma \vdash M : \sigma \wedge \tau} \qquad (\leq)\ \frac{\Gamma \vdash M : \sigma \quad \sigma \leq \tau}{\Gamma \vdash M : \tau}$$

Typing Rules

$$(\mathrm{Ax})\ \Gamma \vdash \alpha : \Gamma(\alpha) \qquad (\omega)\ \Gamma \vdash M : \omega$$

$$(\rightarrow \mathrm{I}_n)\ \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \rightarrow \tau}$$

$$(\rightarrow \mathrm{I}_v)\ \frac{\Gamma, v : \sigma_i \vdash M : \tau \quad \sigma = \bigvee_{i \in I} \sigma_i \quad \sigma_i \in \mathtt{CType}\ i \in I}{\Gamma \vdash \lambda v.M : \sigma \rightarrow \tau}$$

$$(\rightarrow \mathrm{E})\ \frac{\Gamma \vdash M : \sigma \rightarrow \tau\ \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$$

$$(\wedge \mathrm{I})\ \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash M : \tau}{\Gamma \vdash M : \sigma \wedge \tau} \qquad (\leq)\ \frac{\Gamma \vdash M : \sigma \quad \sigma \leq \tau}{\Gamma \vdash M : \tau}$$

$$(+\mathrm{I})\ \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash N : \tau}{\Gamma \vdash M + N : \sigma \vee \tau} \qquad (\parallel \mathrm{I})\ \frac{\Gamma \vdash M : \sigma \qquad \Gamma \vdash N : \tau}{\Gamma \vdash M \parallel N : \sigma \wedge \tau}$$

Characterisation of Convergence

Each type is either a subtype of $\omega \to \omega$ or it is equivalent to ω.

Theorem (Type preservation)

The type system enjoys subject reduction.

Theorem

A closed term is convergent iff it has type $\omega \to \omega$.

Corollary

A closed term is divergent iff it has only types equivalent to ω.

Denotational preciseness for the Concurrent λ-calculus

Theorem

The subtyping $\leq$ is denotationally precise for the concurrent λ-calculus.

$$[\![\sigma]\!] = \{M \mid \vdash M : \sigma\}$$

$$\sigma \leq \tau \ \ \textbf{iff} \ \ [\![\sigma]\!] \subseteq [\![\tau]\!]$$

Operational preciseness for the Concurrent λ-calculus

Theorem

The subtyping $\leq$ is operationally precise for the concurrent λ-calculus.

$\sigma \leq \tau$ **iff** there is no closed terms M_0 such that

- $M_0 P$ converges for all closed terms $P : \tau$ and
- for some $N_0 : \sigma$ the term $M_0 N_0$ diverges

$$\neg \exists M_0 (\forall P : \tau.\ M_0 P \text{ conveges } \bigwedge \exists N_0 : \sigma.\ M_0 N_0 \text{ diverges })$$

M. Dezani-Ciancaglini and SG
Preciseness of subtyping on intersection and union types.
In *RTA-TLCA 2014*, volume 8560 of *LNCS*, pages 194–207 (2014).

Session types, Multiparty session types

Preciseness results a roadmap:

Session (synchronous, asynchronous) types

T. Chen, M. Dezani-Ciancaglini, and N. Yoshida.
On the Preciseness of Subtyping in Session Types.
In *PPDP 2014*, 135–146, 2014.

Multiparty session (synchronous) types

M. Dezani-Ciancaglini, SG, S. Jaksic, J. Pantovic and N. Yoshida.
Precise subtyping for synchronous multiparty sessions.
In *PLACES 2015*, EPTCS 203:29–43, 2016.

M. Dezani-Ciancaglini, SG, S. Jaksic, J. Pantovic and N. Yoshida.
Denotational and Operational Preciseness of Subtyping: A Roadmap.
In *Theory and Practice of Formal Methods 2016*, LNCS 9660: 155–172, 2016.

Operational preciseness - general methodology

Operational soundness follows immediately from

- the subject reduction theorem,
- the subsumption rule, where the subtyping is used

Operational preciseness - general methodology

A general methodology to prove operational completeness is the following one:

- **[Step 1]** Characterise the negation of the subtyping relation by inductive rules
- **[Step 2]** For each type σ define a characteristic term M_σ, which has only the types greater than or equal to σ
- **[Step 3]** For each type σ, define a characteristic context C_σ, which behaves well when filled with terms of type σ
- **[Step 4]** Show that if $\sigma \not\leq \tau$, then bad($C_\tau[M_\sigma]$)

Denotational - general methodology

Theorem

Operational preciseness implies denotational preciseness.

E.g. in λ-calculus with intersection (and union) types, subtyping is denotationally precise (filter models) but it is not operationally precise.

M. Dezani-Ciancaglini, SG, S. Jaksić, J. Pantović, A. Scalas, and N. Yoshida.
Precise subtyping for synchronous multiparty sessions.
(manuscript).

Denotational - general methodology

Theorem

Operational preciseness implies denotational preciseness.

E.g. in λ-calculus with intersection (and union) types, subtyping is denotationally precise (filter models) but it is not operationally precise.

M. Dezani-Ciancaglini, SG, S. Jaksić, J. Pantović, A. Scalas, and N. Yoshida.
Precise subtyping for synchronous multiparty sessions.
(manuscript).

Preciseness for (pure) λ-calculus

Operational completeness requires that all empty (i.e. not inhabited) types are less than all inhabited types

Inhabitation is undecidable for intersection types and for polymorphic types

A complete subtyping on intersection types or on polymorphic types for the pure λ-calculus must be undecidable

This makes unfeasible an operationally complete subtyping for the pure λ-calculus, both in case of intersection and union types and polymorphic types

Open problem: to study the extensions of λ-calculus enjoying operational preciseness for the decidable subtyping between polymorphic types

Conclusion

Preciseness

- denotational
- operational

Languages

- iso-recursive types
- concurrent lambda calculus with intersection and union types
- session types (synchronouse)
- session types (asynchronouse)
- multiparty session types (synchronouse)
- multiparty session types (asynchronouse)?

General language-independent method for preciseness