Linear Intersection Types for Implicit Computational Complexity

Kazushige Terui

terui@nii.ac.jp

National Institute of Informatics

Sougou Kenkyu Daigakuin Daigaku

Motivation

- Verification of computational complexity:

Motivation

- Verification of computational complexity:
- Given a program M, if one can give it a certificate, (eg. typing derivation, ordering, polynomial interpretation) then M can be evaluated in, say, Ptime.

Motivation

- Verification of computational complexity:
- Given a program M, if one can give it a certificate, (eg. typing derivation, ordering, polynomial interpretation) then M can be evaluated in, say, Ptime.
- Nowadays there are plenty of Ptime systems.

Motivation

- Verification of computational complexity:
- Given a program M, if one can give it a certificate, (eg. typing derivation, ordering, polynomial interpretation) then M can be evaluated in, say, Ptime.
- Nowadays there are plenty of Ptime systems.
- They are all extensionally complete (enough functions), but intensionally too poor (few algorithms).

Motivation

- Verification of computational complexity:
- Given a program M, if one can give it a certificate, (eg. typing derivation, ordering, polynomial interpretation) then M can be evaluated in, say, Ptime.
- Nowadays there are plenty of Ptime systems.
- They are all extensionally complete (enough functions), but intensionally too poor (few algorithms).
- More expressive systems are definitely called for...

Motivation

- Difficulties in complexity verification: we have to deal with

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)
  - Dynamic properties (eg. β-reduction)

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)
  - Dynamic properties (eg. β-reduction)
- Intersection types: Dynamic $\Longrightarrow$ Static

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)
  - Dynamic properties (eg. β-reduction)
- Intersection types: Dynamic $\Longrightarrow$ Static
- Standard intersection types:

  SN = Typability

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)
  - Dynamic properties (eg. β-reduction)
- Intersection types: Dynamic $\Longrightarrow$ Static
- Standard intersection types:
  SN = Typability
- Linear intersection types: in addition,
  Normalization length = Derivation size

Motivation

- Difficulties in complexity verification: we have to deal with
  - Asymptotic properties (infinitely many runs)
  - Dynamic properties (eg. β-reduction)
- Intersection types: Dynamic $\Longrightarrow$ Static
- Standard intersection types:
  SN = Typability
- Linear intersection types: in addition,
  Normalization length = Derivation size
- Hope this will help complexity verification...

Outline

- In this talk, we only consider pure lambda calculus.

1. General picture: how hopeless our situation is
2. Linear intersection types:
   derivation size = normalization length
3. Application: a proof of
   Ptime = [Church $\Rightarrow$ Scott in $\mathbf{AL}_{!\multimap\forall^l\mu^l}$]

Preliminary

- Λ: the set of untyped lambda terms.
- $W = \{0,1\}^*$.
- Fix a Church-coding of binary words in W, eg.

$$\overline{01101} \equiv \lambda f_0 f_1 z.f_0(f_1(f_1(f_0(f_1 z)))).$$

- Any $M \in \Lambda$ represents a partial function $f_M : W \longrightarrow W$:

$$\begin{aligned} f_M(w) &= w' \quad \text{if } M\overline{w} \longrightarrow^* \overline{w}' \\ &= \uparrow \quad \text{otherwise.} \end{aligned}$$

- A type system $\mathbf{L}$ (such as System F) determines a set $\mathcal{T} \subseteq \Lambda$ of typable terms.
- Usually, such $\mathcal{T}$ is r.e.

Lambda-characterizations of FP

- $\mathcal{T} \subseteq \Lambda$ is P-sound if

$$M \in \mathcal{T} \Longrightarrow f_M \in \mathbf{FP}.$$

Examples: $\mathbf{MLL}, \emptyset$

Lambda-characterizations of FP

- $\mathcal{T} \subseteq \Lambda$ is P-sound if

$$M \in \mathcal{T} \Longrightarrow f_M \in \mathbf{FP}.$$

Examples: $\mathbf{MLL}, \emptyset$

- $\mathcal{T}$ is extensionally P-complete if in addition

$$f \in \mathbf{FP} \Longrightarrow \exists M \in \mathcal{T}(f = f_M).$$

Examples: $\Lambda\mathbf{BC}$ (Bellantoni-Cook 92) translated into Λ, $\mathbf{1}\lambda^p(\mathbf{W})$ (Leivant-Marion 94), $\mathbf{LLL}$ (Girard 98), $\mathbf{SLL}$ (Lafont 04), $\mathbf{DLAL}$ (Baillot-Terui 04)

Lambda-characterizations of FP

- $\mathcal{T} \subseteq \Lambda$ is P-sound if

$$M \in \mathcal{T} \Longrightarrow f_M \in \mathbf{FP}.$$

  Examples: $\mathbf{MLL}, \emptyset$

- $\mathcal{T}$ is extensionally P-complete if in addition

$$f \in \mathbf{FP} \Longrightarrow \exists M \in \mathcal{T}(f = f_M).$$

  Examples: $\Lambda\mathbf{BC}$ (Bellantoni-Cook 92) translated into Λ, $\mathbf{1}\lambda^p(\mathbf{W})$ (Leivant-Marion 94), $\mathbf{LLL}$ (Girard 98), $\mathbf{SLL}$ (Lafont 04), $\mathbf{DLAL}$ (Baillot-Terui 04)

- $\mathcal{T}$ is intensionally P-complete if in addition

$$f_M \in \mathbf{FP} \Longrightarrow M \in \mathcal{T}.$$

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:
  - polynomials

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:
  - polynomials
  - one-step transitions of TM

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:
  - polynomials
  - one-step transitions of TM
  - (restricted) iteration scheme

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:
  - polynomials
  - one-step transitions of TM
  - (restricted) iteration scheme
- But so what?

Extensional vs Intensional P-completeness

- Intensional P-completeness implies extensional one.
- Extensional P-completeness is very easy to achieve: it is basically sufficient if the following are typable:
  - polynomials
  - one-step transitions of TM
  - (restricted) iteration scheme
- But so what?
- Intensional P-completeness is desired...

Intensional P-completeness is not r.e.

- $\Lambda\mathbf{P}$: the set of terms representing a polynomial time function:

$$M \in \Lambda\mathbf{P} \iff f_M \in \mathbf{FP}.$$

Intensional P-completeness is not r.e.

- $\Lambda\mathbf{P}$: the set of terms representing a polynomial time function:

$$M \in \Lambda\mathbf{P} \iff f_M \in \mathbf{FP}.$$

- Fact: $\Lambda\mathbf{P}$ is neither r.e. nor co-r.e.

Intensional P-completeness is not r.e.

- $\Lambda\mathbf{P}$: the set of terms representing a polynomial time function:

$$M \in \Lambda\mathbf{P} \iff f_M \in \mathbf{FP}.$$

- Fact: $\Lambda\mathbf{P}$ is neither r.e. nor co-r.e.
- Proof: Reduction of Hilbert's 10th problem.

Intensional P-completeness is not r.e.

- $\Lambda\mathbf{P}$: the set of terms representing a polynomial time function:

$$M \in \Lambda\mathbf{P} \iff f_M \in \mathbf{FP}.$$

- Fact: $\Lambda\mathbf{P}$ is neither r.e. nor co-r.e.
- Proof: Reduction of Hilbert's 10th problem.
- For any polynomial $P(X_1, \ldots, X_n)$ with integer coefficients, there is M_P that works on unary integers as follows:

$$\begin{array}{rcll} M_P(0) & = & 1 & \\ M_P(x+1) & = & 1 & \text{if } \exists z_1, \ldots, z_n \leq x. P(z_1, \ldots, z_n) = 0 \\ & = & 2 \cdot M_P(x) & \text{otherwise.} \end{array}$$

Intensional P-completeness is not r.e.

$$\begin{aligned} M_P(0) &= 1 \\ M_P(x+1) &= 1 && \text{if } \exists z_1, \ldots, z_n \leq x . P(z_1, \ldots, z_n) = 0 \\ &= 2 \cdot M_P(x) && \text{otherwise.} \end{aligned}$$

- M_P can be considered as a program on binary words by:

$$\text{u2b}(n) = \underbrace{11 \cdots 111}_{n \ times} \qquad \text{b2u}(\underbrace{011 \cdots 100}_{n \ times}) = n.$$

- $M_P \in \Lambda\mathbf{P}$ iff $P(X_1, \ldots, X_n) = 0$ admits an integer solution.
- Hence $\Lambda\mathbf{P}$ is not r.e (nor co-r.e).

Intensional P-completeness is not r.e.

- Intensional P-completeness cannot be achieved via standard type systems.

Intensional P-completeness is not r.e.

- Intensional P-completeness cannot be achieved via standard type systems.
- The same holds even if Λ is replaced by, say, Λ_F (System F typable terms).

Intensional P-completeness is not r.e.

- Intensional P-completeness cannot be achieved via standard type systems.
- The same holds even if Λ is replaced by, say, Λ_F (System F typable terms).
- Cf. Given $M \in \Lambda_F$, it is decidable in Ptime whether M is typable in **DLAL** (Atassi-Baillot-Terui 2006).

Intensional P-completeness is not r.e.

- Intensional P-completeness cannot be achieved via standard type systems.
- The same holds even if Λ is replaced by, say, Λ_F (System F typable terms).
- Cf. Given $M \in \Lambda_F$, it is decidable in Ptime whether M is typable in **DLAL** (Atassi-Baillot-Terui 2006).
- We are looking for a better approximation of $\Lambda\mathbf{P}$.

Subclasses of $\Lambda\mathbf{P}$

- $\Lambda\mathbf{P}_{SN}$: the class of Ptime strongly normalizable lambda terms:

  $M \in \Lambda\mathbf{P}_{SN} \Longleftrightarrow$

  $\forall w \in W$. for any reduction sequence

  $$M\overline{w} \longrightarrow M_1 \longrightarrow \cdots \longrightarrow M_k$$

  k and $|M_i|$ are polynomial in $|w|$.

Subclasses of ΛP

- R: feasible reduction strategy (such as leftmost, innermost): given M, R picks up a redex of M, if any, in Ptime.
- $\Lambda\mathbf{P}_R$: the class of Ptime R-normalizable lambda terms M: $\forall w \in W$. for the R-reduction sequence

$$M\overline{w} \longrightarrow_R M_1 \longrightarrow_R \cdots \longrightarrow M_k$$

with M_k in normal form
k and $|M_i|$ are polynomial in $|w|$.

- Finally,

$$\Lambda\mathbf{P}_{WN} := \bigcup_{R\text{:feasible strategy}} \Lambda\mathbf{P}_R.$$

Subclasses of ΛP

- $\mathbf{\Lambda P}_{SN} = \mathbf{\Lambda P}_{perp} \subseteq \mathbf{\Lambda P}_{R} \subseteq \mathbf{\Lambda P}_{WN} \subseteq \mathbf{\Lambda P}$

Subclasses of $\Lambda\mathbf{P}$

- $\Lambda\mathbf{P}_{SN} = \Lambda\mathbf{P}_{perp} \subseteq \Lambda\mathbf{P}_{R} \subseteq \Lambda\mathbf{P}_{WN} \subseteq \Lambda\mathbf{P}$
- $\Lambda\mathbf{P}_{SN}, \Lambda\mathbf{P}_{WN}$ are neither r.e. nor co-r.e.

Subclasses of $\Lambda\mathbf{P}$

- $\Lambda\mathbf{P}_{SN} = \Lambda\mathbf{P}_{perp} \subseteq \Lambda\mathbf{P}_{R} \subseteq \Lambda\mathbf{P}_{WN} \subseteq \Lambda\mathbf{P}$
- $\Lambda\mathbf{P}_{SN}, \Lambda\mathbf{P}_{WN}$ are neither r.e. nor co-r.e.
- $\mathbf{DLAL} \subset \Lambda\mathbf{P}_{SN}$

Subclasses of $\Lambda\mathbf{P}$

- $\Lambda\mathbf{P}_{SN} = \Lambda\mathbf{P}_{perp} \subseteq \Lambda\mathbf{P}_{R} \subseteq \Lambda\mathbf{P}_{WN} \subseteq \Lambda\mathbf{P}$
- $\Lambda\mathbf{P}_{SN}, \Lambda\mathbf{P}_{WN}$ are neither r.e. nor co-r.e.
- $\mathbf{DLAL} \subset \Lambda\mathbf{P}_{SN}$
- $\Lambda\mathbf{BC} \not\subset \Lambda\mathbf{P}_{SN}$, $\Lambda\mathbf{BC} \subset \Lambda\mathbf{P}_{innermost}$.

Intersection types

- First introduced by (Coppo-Dezani 80)

Intersection types

- First introduced by (Coppo-Dezani 80)
- Later on, various systems have been considered:

Intersection types

- First introduced by (Coppo-Dezani 80)
- Later on, various systems have been considered:
  - With ω (CDV, Coppo-Dezani-Venneri 81),

Intersection types

- First introduced by (Coppo-Dezani 80)
- Later on, various systems have been considered:
  - With ω (CDV, Coppo-Dezani-Venneri 81),
  - With subtyping (BCD, Barendregt-Coppo-Dezani 83),

Intersection types

- First introduced by (Coppo-Dezani 80)
- Later on, various systems have been considered:
  - With ω (CDV, Coppo-Dezani-Venneri 81),
  - With subtyping (BCD, Barendregt-Coppo-Dezani 83),
  - With expansion variables (System I, Kfouly-Wells 03), etc.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.
- Contraction: $\sigma \leq \sigma \cap \sigma$.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.
- Contraction: $\sigma \leq \sigma \cap \sigma$.
- Weakening: $\sigma \cap \tau \leq \sigma$.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.
- Contraction: $\sigma \leq \sigma \cap \sigma$.
- Weakening: $\sigma \cap \tau \leq \sigma$.
- Linear intersection types satisfy neither of them.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.
- Contraction: $\sigma \leq \sigma \cap \sigma$.
- Weakening: $\sigma \cap \tau \leq \sigma$.
- Linear intersection types satisfy neither of them.
- Affine intersection types satisfy Weakening but not Idempotency/Contraction.

Intersection types and linearity

- Idempotency: $\sigma = \sigma \cap \sigma$.
- Contraction: $\sigma \leq \sigma \cap \sigma$.
- Weakening: $\sigma \cap \tau \leq \sigma$.
- Linear intersection types satisfy neither of them.
- Affine intersection types satisfy Weakening but not Idempotency/Contraction.
- Relationship with linear logic is suggested by (Regnier 92, Mairson-Møller 04, Carlier-Wells 04).

Type system iMAL

- Variables: α, β
- Intersections: $D, E \quad ::= A_1 \otimes \ldots \otimes A_n \quad (n \geq 1)$
- Types: $A, B, C \quad ::= \alpha \mid D \multimap A$
- Environments: $\Gamma, \Delta, \Sigma \quad ::= \{x_1 : A_1, \ldots, x_n : A_n\}$

(Multiset. $x_1, \ldots, x_n$ not necessarily distinct.)

- Type inference rules:

$$\frac{}{\Gamma, x : A \vdash x : A}\ (var) \qquad \frac{\Gamma, x : A_1, \ldots, x : A_n \vdash M : B \quad x \notin Var(\Gamma)}{\Gamma \vdash \lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B}\ (\multimap I)$$

$$\frac{\Gamma_0 \vdash M : A_1 \otimes \cdots \otimes A_n \multimap B \quad \Gamma_1 \vdash N : A_1 \quad \cdots \quad \Gamma_n \vdash N : A_n}{\Gamma_0, \Gamma_1, \ldots, \Gamma_n \vdash MN : B}\ (\multimap E)$$

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?
- Normalization length = Derivation size in $i\mathbf{MAL}$.

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?
- Normalization length = Derivation size in $i\mathbf{MAL}$.
- What is the strength of a logic?

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?
- Normalization length = Derivation size in $i\mathbf{MAL}$.
- What is the strength of a logic?
- Explicit typing: How many terms it types.

$$\mathbf{MAL}_{-\circ} \subsetneq \textbf{SimTyp} \subsetneq \textbf{SysF}$$

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?
- Normalization length = Derivation size in $i\mathbf{MAL}$.
- What is the strength of a logic?
- Explicit typing: How many terms it types.

$$\mathbf{MAL}_{-\circ} \subsetneq \textbf{SimTyp} \subsetneq \textbf{SysF}$$

- Implicit typing: How short typing derivations are.

Strength of logic

- SN = Typability in $i\mathbf{MAL}$. Then what's the difference from the standard one?
- Normalization length = Derivation size in $i\mathbf{MAL}$.
- What is the strength of a logic?
- Explicit typing: How many terms it types.

$$\mathbf{MAL}_{\multimap} \subsetneq \textbf{SimTyp} \subsetneq \textbf{SysF}$$

- Implicit typing: How short typing derivations are.
- Connection with propositional proof systems?

Type system iMAL

- $\mathcal{D} \rhd \Gamma \vdash M : A \Longleftrightarrow \mathcal{D}$ is a derivation for $\Gamma \vdash M : A$.
- $\mathcal{D} \rhd M \Longleftrightarrow$ there are Γ, A such that $\mathcal{D} \rhd \Gamma \vdash M : A$.
- $|M|$: the number of λ and applications in the term M.
- $|\mathcal{D}|$: the number of $(\multimap I)$ and $(\multimap E)$ in the derivation $\mathcal{D}$.
- Lemma: $\mathcal{D} \rhd M \Longrightarrow |\mathcal{D}| \geq |M|$.
- Lemma: M is in nf $\Longrightarrow$ there is $\mathcal{D}$ such that $\mathcal{D} \rhd M$ and $|\mathcal{D}| = |M|$.

Derivation size bounds normalization cost

- Subject Reduction Theorem: $\mathcal{D} \rhd \Gamma \vdash M : B$ and $M \longrightarrow N \Longrightarrow$ $\mathcal{D}' \rhd \Gamma \vdash N : B$ and $|\mathcal{D}| > |\mathcal{D}'|$.

$$\dfrac{\dfrac{\begin{array}{c}[x : A_1] \cdots [x : A_n] \\ \vdots \\ M : B\end{array}}{\lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B} \quad \begin{array}{c}\vdots \\ N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots \\ N : A_n\end{array}}{(\lambda x.M)N : B} \Longrightarrow \dfrac{\begin{array}{c}\vdots \\ N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots \\ N : A_n\end{array}}{M[N/x] : B}$$

Derivation size bounds normalization cost

- Subject Reduction Theorem: $\mathcal{D} \rhd \Gamma \vdash M : B$ and $M \longrightarrow N \Longrightarrow$ $\mathcal{D}' \rhd \Gamma \vdash N : B$ and $|\mathcal{D}| > |\mathcal{D}'|$.

$$\dfrac{\dfrac{[x : A_1] \cdots [x : A_n]}{\vdots}\ \lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B \quad N : A_1 \quad \cdots \quad N : A_n}{(\lambda x.M)N : B} \Longrightarrow \dfrac{N : A_1 \quad \cdots \quad N : A_n}{M[N/x] : B}$$

- Corollary: If $\mathcal{D} \rhd M$, then

Derivation size bounds normalization cost

- Subject Reduction Theorem: $\mathcal{D} \rhd \Gamma \vdash M : B$ and $M \longrightarrow N \Longrightarrow$ $\mathcal{D}' \rhd \Gamma \vdash N : B$ and $|\mathcal{D}| > |\mathcal{D}'|$.

$$\dfrac{\dfrac{\begin{array}{c}[x : A_1] \cdots [x : A_n] \\ \vdots \\ M : B\end{array}}{\lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B} \quad \begin{array}{c}\vdots \\ N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots \\ N : A_n\end{array}}{(\lambda x.M)N : B} \Longrightarrow \begin{array}{c}\begin{array}{ccc}\vdots & & \vdots \\ N : A_1 & \cdots & N : A_n\end{array} \\ \vdots \\ M[N/x] : B\end{array}$$

- Corollary: If $\mathcal{D} \rhd M$, then
  1. M strongly normalizes in $|\mathcal{D}|$ steps.

Derivation size bounds normalization cost

- Subject Reduction Theorem: $\mathcal{D} \rhd \Gamma \vdash M : B$ and $M \longrightarrow N \Longrightarrow$ $\mathcal{D}' \rhd \Gamma \vdash N : B$ and $|\mathcal{D}| > |\mathcal{D}'|$.

$$\dfrac{\dfrac{\begin{array}{c}[x:A_1]\ \cdots\ [x:A_n] \\ \vdots \\ M : B\end{array}}{\lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B} \quad \begin{array}{c}\vdots \\ N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots \\ N : A_n\end{array}}{(\lambda x.M)N : B} \Longrightarrow \dfrac{\begin{array}{c}\vdots \\ N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots \\ N : A_n\end{array}}{M[N/x] : B}$$

- Corollary: If $\mathcal{D} \rhd M$, then
  1. M strongly normalizes in $|\mathcal{D}|$ steps.
  2. $M \longrightarrow^* N \Longrightarrow |N| \leq |\mathcal{D}|$.

Normalization cost bounds derivation size

- We refine Møller-Neergaard's simple proof of SN $\Rightarrow$ Typability.
- Difficulty 1: Subject expansion does not hold in general.

$$(\lambda x.M)N \longrightarrow M \quad (x \notin FV(M))$$

- N might not have derivation (eg. $N = \Omega$)
- If N is in nf, then N has a derivation.
- Difficulty 2: Redex might be located above intersections. Then the derivation size increase too much.

$$\dfrac{\begin{array}{c} R : B_1 \\ \vdots \\ L : A_1 \end{array} \quad \cdots \quad \begin{array}{c} R : B_1 \\ \vdots \\ L : A_n \end{array}}{\begin{array}{c} L : A_1 \otimes \cdots \otimes A_n \\ \vdots \end{array}} \not\Longrightarrow \dfrac{\begin{array}{c} R' : B_1 \\ \vdots \\ L : A_1 \end{array} \quad \cdots \quad \begin{array}{c} R' : B_n \\ \vdots \\ L : A_n \end{array}}{\begin{array}{c} L : A_1 \otimes \cdots \otimes A_n \\ \vdots \end{array}}$$

Normalization cost bounds derivation size

- Consider the following perpetual reduction strategy:

$$\frac{x \in FV(M)}{(\lambda x.M)N\vec{L} \longrightarrow_P M[N/x]\vec{L}} \qquad \frac{x \notin FV(M) \quad N_1 \longrightarrow_P N_2}{(\lambda x.M)N_1\vec{L} \longrightarrow_P (\lambda x.M)N_2\vec{L}}$$

$$\frac{x \notin FV(M) \quad N \text{ in nf}}{(\lambda x.M)N\vec{L} \longrightarrow_P M\vec{L}} \qquad \frac{M_1 \longrightarrow_P M_2}{\lambda x.M_1 \longrightarrow_P \lambda x.M_2} \qquad \frac{N_1 \longrightarrow_P N_2}{x\vec{M}N_1\vec{K} \longrightarrow_P x\vec{M}N_2\vec{K}}$$

- Weak Subject Expansion Theorem: If $\mathcal{D} \rhd N$ (with $\mathcal{D}$ "canonical") and $M \longrightarrow_P N$, then there is a "canonical" $\mathcal{D}'$ such that $\mathcal{D}' \rhd M$ and $|\mathcal{D}'| \leq |\mathcal{D}| + |M| + 1$.

Characterization of $\Lambda \mathbf{P}_{SN}$

- Theorem 1: $\mathcal{D} \rhd M \Longrightarrow M$ strongly normalizes in length $|\mathcal{D}|$ and size $|\mathcal{D}|$.

Characterization of $\Lambda \mathbf{P}_{SN}$

- Theorem 1: $\mathcal{D} \rhd M \Longrightarrow M$ strongly normalizes in length $|\mathcal{D}|$ and size $|\mathcal{D}|$.
- Theorem 2: M strongly normalizes in length m and size $n \Longrightarrow$ there is $\mathcal{D}$ such that $\mathcal{D} \rhd M$ and $|\mathcal{D}| \leq n(m+1)$.

Characterization of $\Lambda \mathbf{P}_{SN}$

- Theorem 1: $\mathcal{D} \triangleright M \Longrightarrow M$ strongly normalizes in length $|\mathcal{D}|$ and size $|\mathcal{D}|$.
- Theorem 2: M strongly normalizes in length m and size $n \Longrightarrow$ there is $\mathcal{D}$ such that $\mathcal{D} \triangleright M$ and $|\mathcal{D}| \leq n(m+1)$.
- Static Characterization of Λ_{PSN}: $M \in \Lambda \mathbf{P}_{SN} \Longleftrightarrow$ $\{M\overline{w}\}_{w \in W}$ have polynomial size derivations.

Characterization of $\Lambda \mathbf{P}_{SN}$

- Theorem 1: $\mathcal{D} \triangleright M \Longrightarrow M$ strongly normalizes in length $|\mathcal{D}|$ and size $|\mathcal{D}|$.
- Theorem 2: M strongly normalizes in length m and size $n \Longrightarrow$ there is $\mathcal{D}$ such that $\mathcal{D} \triangleright M$ and $|\mathcal{D}| \leq n(m+1)$.
- Static Characterization of Λ_{PSN}: $M \in \Lambda \mathbf{P}_{SN} \Longleftrightarrow$ $\{M\overline{w}\}_{w \in W}$ have polynomial size derivations.

Beyond $\Lambda\mathbf{P}_{SN}$

- $\Lambda\mathbf{P}_{SN}$ is too small to be of practical interest.

Beyond $\Lambda\mathbf{P}_{SN}$

- $\Lambda\mathbf{P}_{SN}$ is too small to be of practical interest.
- Iteration of if-then-else not allowed in $\Lambda\mathbf{P}_{SN}$.

Beyond $\Lambda \mathbf{P}_{SN}$

- $\Lambda \mathbf{P}_{SN}$ is too small to be of practical interest.
- Iteration of if-then-else not allowed in $\Lambda \mathbf{P}_{SN}$.
- Let $M \equiv \lambda x.(\text{if } x = 0 \text{ then } x \text{ else } x)$. Then

$$M^n\overline{0} \longrightarrow^*_{CBV} M^{n-1}\overline{0} \longrightarrow^*_{CBV} \cdots \longrightarrow^*_{CBV} \overline{0}$$

takes $O(n)$ steps.

$$M\overline{0} \longrightarrow^* \text{if } M^{n-1}\overline{0} = \overline{0} \text{ then } M^{n-1}\overline{0} \text{ else } M^{n-1}\overline{0}$$
$$\longrightarrow^* \cdots \longrightarrow^* 0$$

takes $O(2^n)$ steps.

Beyond $\Lambda\mathbf{P}_{SN}$

- $\Lambda\mathbf{P}_{SN}$ is too small to be of practical interest.
- Iteration of if-then-else not allowed in $\Lambda\mathbf{P}_{SN}$.
- Let $M \equiv \lambda x.(\text{if } x = 0 \text{ then } x \text{ else } x)$. Then

  $$M^n\overline{0} \longrightarrow^*_{CBV} M^{n-1}\overline{0} \longrightarrow^*_{CBV} \cdots \longrightarrow^*_{CBV} \overline{0}$$

  takes $O(n)$ steps.

  $$M\overline{0} \longrightarrow^* \text{if } M^{n-1}\overline{0} = \overline{0} \text{ then } M^{n-1}\overline{0} \text{ else } M^{n-1}\overline{0}$$
  $$\longrightarrow^* \cdots \longrightarrow^* 0$$

  takes $O(2^n)$ steps.
- In particular, $\Lambda\mathbf{BC} \not\subset \Lambda\mathbf{P}_{SN}$.

Beyond $\Lambda\mathbf{P}_{SN}$

- Let R be a feasible reduction strategy.
- An (abstract) size function s : Derivations $\longrightarrow \mathbb{N}$ is admissible for (M, R) if for any M_1 such that $M\overline{w} \longrightarrow_R^* M_1$,
  1. $\mathcal{D} \rhd M_1 \implies s(\mathcal{D}) \geq |M_1|$
  2. $(\mathcal{D}_1 \rhd M_1 \longrightarrow_R \mathcal{D}_2 \rhd M_2) \implies s(\mathcal{D}_1) > s(\mathcal{D}_2)$
- Theorem: $M \in \Lambda\mathbf{P}_R \iff$
  1. there is a size function s admissible for (M, R)
  2. $\{M\overline{w}\}_{w \in \mathbf{W}}$ has polynomial size derivations w.r.t. s.
- In practice, we have to find suitable size function s for each term M to be analyzed.

Case Study: a ramified Ptime system

- Recall (Leivant-Marion 94) characterizes $\mathbf{FP}$ based on lambda-calculus with pairing and some constants, where
- Lower tier: word algebra terms ϵ, $0(\epsilon)$, $1(0(\epsilon))$, ... of base type o
- Higher tier: Church words: $\lambda f x.f0(f1(f1(x)))$, ...
- $f \in \mathbf{FP} \Longleftrightarrow f$ represented by

$$M_f : ((A \to A) \to A \to A) \to o$$

where $A \equiv o \times \cdots \times o$.

Case Study: a ramified Ptime system

- We consider a logical variant based on pure lambda-calculus with linear polymorphism and linear recursive types, where
- Lower tier: Scott words of type $\forall\alpha.\mu\beta.(\mathbf{B} \multimap \beta \multimap \alpha) \multimap \alpha \multimap \alpha$
- Higher tier: Church words of type $\forall\alpha.(\mathbf{B} \Rightarrow \alpha \Rightarrow \alpha) \Rightarrow \alpha \Rightarrow \alpha$ (with $A \Rightarrow B \equiv !A \multimap B$)

The system $\mathbf{AL}_{!\multimap\forall^l\mu^l}$

- $\mathbf{AL}_{!\multimap\forall^l\mu^l}$: Intuitionistic affine linear logic with

$$\frac{\Gamma \vdash M : A \quad \alpha \notin FV(\Gamma)}{\Gamma \vdash M : \forall\alpha.A} \qquad \frac{\Gamma \vdash M : \forall\alpha.A}{\Gamma \vdash M : A[L/\alpha]}$$

$$\frac{\Gamma \vdash M : L[\mu\beta.L/\beta]}{\Gamma \vdash M : \mu\beta.L} \qquad \frac{\Gamma \vdash M : \mu\beta.L}{\Gamma \vdash M : L[\mu\beta.L/\beta]}$$

- where L is purely linear, i.e., without !, and β occurs at most once in L.

Scott numerals

- For simplicity, we consider unary numerals rather than words.
- Church numerals of type $\mathbf{N}_C \equiv \forall\alpha.(\alpha \Rightarrow \alpha) \Rightarrow \alpha \Rightarrow \alpha$
  nonlinear, support iteration
- Scott numerals of type $\mathbf{N}_S \equiv \forall\alpha.\mu\beta.(\beta \multimap \alpha) \multimap \alpha \multimap \alpha$
  linear, support basic functions

$$
\begin{array}{rcll}
\overline{0} & \equiv & \lambda xy.y & : \mathbf{N}_S \\
\overline{n+1} & \equiv & \lambda xy.x\overline{n} & : \mathbf{N}_S \\
suc & \equiv & \lambda z.\lambda xy.xz & : \mathbf{N}_S \multimap \mathbf{N}_S \\
prd & \equiv & \lambda z.z(\lambda x.x)\overline{0} & : \mathbf{N}_S \multimap \mathbf{N}_S \\
cond & \equiv & \lambda z_1 z_2 z_3.z_1(\lambda w.z_2)z_3 & : \mathbf{N}_S^3 \multimap \mathbf{N}_S
\end{array}
$$

Representation of FP

- Theorem: Any $f \in \mathbf{FP}$ can be represented by a closed term $M_f : \mathbf{W}_C \Rightarrow \mathbf{W}_S$.
- Proof: we have
  - Polynomials: $\mathbf{N}_C \Rightarrow \mathbf{N}_C$
  - Transitions of TMs based on: $\mathbf{W}_S \Rightarrow \mathbf{W}_S$
  - Iteration: $\mathbf{N}_C \Rightarrow (\mathbf{W}_S \Rightarrow \mathbf{W}_S) \Rightarrow (\mathbf{W}_S \Rightarrow \mathbf{W}_S)$
- $\not\subseteq \Lambda\mathbf{P}_{SN}$. How to show $\subseteq \Lambda\mathbf{P}_{WN}$?

Pruned size of derivations

- Pruned size $s(\mathcal{D})$ defined by:
- When $\mathcal{D}$ is (var), $s(\mathcal{D}) = 0$.
- $s(\mathcal{D}) = s(\mathcal{D}_0) + 1$ when $\mathcal{D}$ is

$$\dfrac{\begin{array}{c}\vdots\ \mathcal{D}_0 \\ \Gamma, x : A_1, \ldots, x : A_n \vdash M : B\end{array}}{\Gamma \vdash \lambda x.M : A_1 \otimes \cdots \otimes A_n \multimap B}\ (\multimap I)$$

- $s(\mathcal{D}) = s(\mathcal{D}_0) + min(s(\mathcal{D}_1), \ldots, s(\mathcal{D}_n)) + |\Gamma_1, A_1, \ldots, \Gamma_n, A_n|$, when $\mathcal{D}$ is

$$\dfrac{\begin{array}{c}\vdots\ \mathcal{D}_0 \\ \Delta_0, \Gamma_0 \vdash M : A_1 \otimes \cdots \otimes A_n \multimap B\end{array} \quad \begin{array}{c}\vdots\ \mathcal{D}_1 \\ \Delta_1, \Gamma_1 \vdash N : A_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots\ \mathcal{D}_n \\ \Delta_n, \Gamma_n \vdash N : A_n\end{array}}{\Delta_0, \Gamma_0, \Delta_1, \Gamma_1, \ldots, \Delta_n, \Gamma_n \vdash MN : B}\ (\multimap E)$$

where Δ_i consists of redundant types (to be explained later).

Normalization bound

- Given $M : \mathbf{N}_C \Rightarrow \mathbf{N}_S$ and $\overline{n} : \mathbf{N}_C$, there is an intersection type derivation $\mathcal{D} \rhd M\overline{n}$.
- Strategy R: First reduce all redices with !-type:

$$M\overline{n} \longrightarrow\!\!\!\!\!\rightarrow^{*} M_1$$

  then reduce the rightmost $(\lambda x.M)N$ with N in nf.
- After M_1, the pruned size s is admissible for R:
  - $\mathcal{D}_i \rhd M_i \implies s(\mathcal{D}_i) \geq |M_i|$
  - $(\mathcal{D}_i \rhd M_i \longrightarrow_R \mathcal{D}_{i+1} \rhd M_{i+1}) \implies s(\mathcal{D}_i) > s(\mathcal{D}_{i+1})$
- the latter because of
  - If $(\lambda x.M)N$ is the redex to be reduced, N is in nf and does not contain a free variable of redundant type.

Normalization bound

- If $\mathcal{D}_1 \rhd M_1$, $s(\mathcal{D}_1)$ is polynomial in $|M_1|$.
- By finite dispatching, the size function s' admissible for (M, R) can be obtained.
- Theorem:

$$M : \mathbf{W}_C \Rightarrow \mathbf{W}_S \text{ in } \mathbf{AL}_{!-\circ\forall^l\mu^l} \implies M \in \Lambda\mathbf{P}_{WN}$$

Summary

- $\mathbf{\Lambda P}$, $\mathbf{\Lambda P}_{SN}$, $\mathbf{\Lambda P}_{WN}$ not r.e.
- Linear intersection types: suitable for verifying dynamic properties statically
- $\mathbf{\Lambda P}_{SN}$ = terms having polysize derivations in $i\mathbf{MAL}$.
- Showed [Church $\Rightarrow$ Scott] belongs to $\mathbf{\Lambda P}_{WN}$ by using linear intersection types and an abstract size function.
- Hope it will lead to automatic verification of practical Ptime functions (like Quick-sort, Mutual-division) using intersection types and abstract size functions.