Linear Rank Intersection Types

Fábio Reis

Faculty of Sciences, University of Porto, Portugal

Joint work with: Sandra Alves, Mário Florido

2022

Intersection Types

Intersection types $\sigma, \sigma_1, \sigma_2, \ldots$ are defined by the following grammar, where $n \geq 1$ and α is a type variable:

$$\sigma ::= \alpha \mid \sigma_1 \cap \cdots \cap \sigma_n \to \sigma.$$

Intersection Types

Intersection types $\sigma, \sigma_1, \sigma_2, \ldots$ are defined by the following grammar, where $n \geq 1$ and α is a type variable:

$$\sigma ::= \alpha \mid \sigma_1 \cap \cdots \cap \sigma_n \to \sigma.$$

- In the first intersection type systems, $\cap$ is idempotent.

Intersection Types

Intersection types $\sigma, \sigma_1, \sigma_2, \ldots$ are defined by the following grammar, where $n \geq 1$ and α is a type variable:

$$\sigma ::= \alpha \mid \sigma_1 \cap \cdots \cap \sigma_n \rightarrow \sigma.$$

- In the first intersection type systems, $\cap$ is idempotent.
- **Quantitative types** are the non-idempotent intersection types ($\cap$ is non-idempotent): $\alpha \cap \alpha \rightarrow \beta \neq \alpha \rightarrow \beta$.

Finite Rank

- Typability is undecidable for unrestricted intersection type systems.

Finite Rank

- Typability is undecidable for unrestricted intersection type systems.
- Restricting intersection types to a finite rank makes typability decidable.

Definition (Rank of intersection types)

Let $\mathbb{T}_0$ be the set of simple types and
$\mathbb{T}_1 = \{\tau_1 \cap \cdots \cap \tau_m \mid \tau_1, \ldots, \tau_m \in \mathbb{T}_0, m \geq 1\}$.

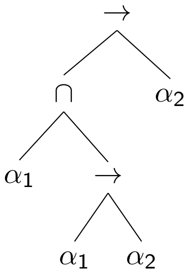
The set $\mathbb{T}_k$, of rank k intersection types (for $k \geq 2$), can be defined recursively in the following way ($n \geq 3$, $m \geq 2$):

$$\mathbb{T}_2 = \mathbb{T}_0 \cup \{\vec{\tau} \to \sigma \mid \vec{\tau} \in \mathbb{T}_1, \sigma \in \mathbb{T}_2\}$$
$$\mathbb{T}_n = \mathbb{T}_{n-1} \cup \{\vec{\tau}_1 \cap \cdots \cap \vec{\tau}_m \to \sigma \mid \vec{\tau}_1, \ldots, \vec{\tau}_m \in \mathbb{T}_{n-1}, \sigma \in \mathbb{T}_n\}$$

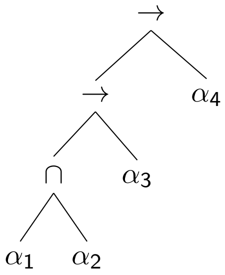
Finite Rank - Example

- The rank of an intersection type is related to the depth of the nested intersections.

$\alpha_1 \cap (\alpha_1 \to \alpha_2) \to \alpha_2$:



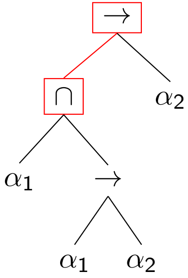
$(\alpha_1 \cap \alpha_2 \to \alpha_3) \to \alpha_4$:



Finite Rank - Example

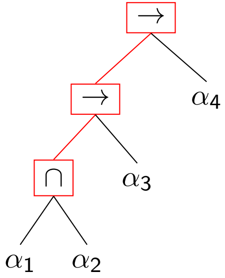
- The rank of an intersection type is related to the depth of the nested intersections.

$\alpha_1 \cap (\alpha_1 \to \alpha_2) \to \alpha_2$:



Rank 2

$(\alpha_1 \cap \alpha_2 \to \alpha_3) \to \alpha_4$:



Rank 3

Motivation

Consider the term $\lambda fx.f(fx)$.

- In a quantitative type system, the term is typable with: $((\alpha \to \alpha) \cap (\alpha \to \alpha)) \to \alpha \to \alpha$.

Motivation

Consider the term $\lambda fx.f(fx)$.

- In a quantitative type system, the term is typable with: $((\alpha \to \alpha) \cap (\alpha \to \alpha)) \to \alpha \to \alpha$.
- Then, in an idempotent system, it can be typed with: $(\alpha \to \alpha) \to \alpha \to \alpha$.

Motivation

Consider the term $\lambda fx.f(fx)$.

- In a quantitative type system, the term is typable with: $((\alpha \to \alpha) \cap (\alpha \to \alpha)) \to \alpha \to \alpha$.
- Then, in an idempotent system, it can be typed with: $(\alpha \to \alpha) \to \alpha \to \alpha$.
- This rank decrease makes it possible to type a term like $(\lambda x.x)(\lambda fx.f(fx))$ in a rank 2 idempotent type system, which would not be typable in a rank 2 quantitative type system ($(\lambda x.x)$ must be typed with a rank 3 type).

Motivation

Consider the term $\lambda fx.f(fx)$.

- In a quantitative type system, the term is typable with: $((\alpha \to \alpha) \cap (\alpha \to \alpha)) \to \alpha \to \alpha$. $\Leftarrow$ f occurs twice
- Then, in an idempotent system, it can be typed with: $(\alpha \to \alpha) \to \alpha \to \alpha$.
- This rank decrease makes it possible to type a term like $(\lambda x.x)(\lambda fx.f(fx))$ in a rank 2 idempotent type system, which would not be typable in a rank 2 quantitative type system ($(\lambda x.x)$ must be typed with a rank 3 type). But **quantitative information is lost**.

Motivation

Consider the term $\lambda fx.f(fx)$.

- In a quantitative type system, the term is typable with: $((\alpha \to \alpha) \cap (\alpha \to \alpha)) \to \alpha \to \alpha$. $\Leftarrow$ f occurs twice
- Then, in an idempotent system, it can be typed with: $(\alpha \to \alpha) \to \alpha \to \alpha$.
- This rank decrease makes it possible to type a term like $(\lambda x.x)(\lambda fx.f(fx))$ in a rank 2 idempotent type system, which would not be typable in a rank 2 quantitative type system ($(\lambda x.x)$ must be typed with a rank 3 type). But **quantitative information is lost**.

Only the linear terms[1] are typed by a simple type in a non-idempotent intersection type system.

[1] Depending on the type system, that can be true for the affine terms.

Linear Rank

- We propose a new definition of rank for intersection types that differs from the previous one in the base case and the introduction of the *linear arrow* $\multimap$.

Definition (Linear rank of intersection types)

Let $\mathbb{T}_{\mathbb{L}0}$ be the set of linear types and
$\mathbb{T}_{\mathbb{L}1} = \{\tau_1 \cap \cdots \cap \tau_m \mid \tau_1, \ldots, \tau_m \in \mathbb{T}_{\mathbb{L}0}, m \geq 1\}$.

The set $\mathbb{T}_{\mathbb{L}k}$, of *linear rank k* intersection types (for $k \geq 2$), can be defined recursively in the following way ($n \geq 3$, $m \geq 2$):

$$
\begin{aligned}
\mathbb{T}_{\mathbb{L}2} = \mathbb{T}_{\mathbb{L}0} &\cup \{\tau \multimap \sigma \mid \tau \in \mathbb{T}_{\mathbb{L}0}, \sigma \in \mathbb{T}_{\mathbb{L}2}\} \\
&\cup \{\tau_1 \cap \cdots \cap \tau_m \to \sigma \mid \tau_1, \ldots, \tau_m \in \mathbb{T}_{\mathbb{L}0}, \sigma \in \mathbb{T}_{\mathbb{L}2}\} \\
\mathbb{T}_{\mathbb{L}n} = \mathbb{T}_{\mathbb{L}n-1} &\cup \{\vec{\tau} \multimap \sigma \mid \vec{\tau} \in \mathbb{T}_{\mathbb{L}n-1}, \sigma \in \mathbb{T}_{\mathbb{L}n}\} \\
&\cup \{\vec{\tau}_1 \cap \cdots \cap \vec{\tau}_m \to \sigma \mid \vec{\tau}_1, \ldots, \vec{\tau}_m \in \mathbb{T}_{\mathbb{L}n-1}, \sigma \in \mathbb{T}_{\mathbb{L}n}\}
\end{aligned}
$$

Linear Rank 2 Intersection Type System

In the Linear Rank 2 Intersection Type System, we say that M has type σ given the environment Γ, and write $\Gamma \vdash_2 M : \sigma$ if it can be obtained from the following *derivation rules*:

$$[x : \tau] \vdash_2 x : \tau \qquad \text{(Axiom)}$$

$$\frac{\Gamma_1, x : \vec{\tau}_1, y : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, y : \vec{\tau}_2, x : \vec{\tau}_1, \Gamma_2 \vdash_2 M : \sigma} \qquad \text{(Exchange)}$$

$$\frac{\Gamma_1, x_1 : \vec{\tau}_1, x_2 : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, x : \vec{\tau}_1 \cap \vec{\tau}_2, \Gamma_2 \vdash_2 M[x/x_1, x/x_2] : \sigma} \qquad \text{(Contraction)}$$

$$\frac{\Gamma, x : \tau_1 \cap \cdots \cap \tau_n \vdash_2 M : \sigma \qquad n \geq 2}{\Gamma \vdash_2 \lambda x.M : \tau_1 \cap \cdots \cap \tau_n \to \sigma} \qquad (\to \text{Intro})$$

$$\frac{\Gamma \vdash_2 M_1 : \tau_1 \cap \cdots \cap \tau_n \to \sigma \qquad \Gamma_1 \vdash_2 M_2 : \tau_1 \ \cdots \ \Gamma_n \vdash_2 M_2 : \tau_n \qquad n \geq 2}{\Gamma, \sum_{i=1}^{n} \Gamma_i \vdash_2 M_1 M_2 : \sigma} \qquad (\to \text{Elim})$$

$$\frac{\Gamma, x : \tau \vdash_2 M : \sigma}{\Gamma \vdash_2 \lambda x.M : \tau \multimap \sigma} \qquad (\multimap \text{Intro})$$

$$\frac{\Gamma_1 \vdash_2 M_1 : \tau \multimap \sigma \qquad \Gamma_2 \vdash_2 M_2 : \tau}{\Gamma_1, \Gamma_2 \vdash_2 M_1 M_2 : \sigma} \qquad (\multimap \text{Elim})$$

Linear Rank 2 Intersection Type System

In the Linear Rank 2 Intersection Type System, we say that M has type σ given the environment Γ, and write $\Gamma \vdash_2 M : \sigma$ if it can be obtained from the following *derivation rules*:

$$[x : \tau] \vdash_2 x : \tau \qquad \text{(Axiom)}$$

$$\frac{\Gamma_1, x : \vec{\tau}_1, y : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, y : \vec{\tau}_2, x : \vec{\tau}_1, \Gamma_2 \vdash_2 M : \sigma} \qquad \text{(Exchange)}$$

$$\frac{\Gamma_1, x_1 : \vec{\tau}_1, x_2 : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, x : \vec{\tau}_1 \cap \vec{\tau}_2, \Gamma_2 \vdash_2 M[x/x_1, x/x_2] : \sigma} \qquad \text{(Contraction)}$$

$$\frac{\Gamma, x : \tau_1 \cap \cdots \cap \tau_n \vdash_2 M : \sigma \qquad n \geq 2}{\Gamma \vdash_2 \lambda x.M : \tau_1 \cap \cdots \cap \tau_n \to \sigma} \qquad (\to \text{Intro})$$

$$\frac{\Gamma \vdash_2 M_1 : \tau_1 \cap \cdots \cap \tau_n \to \sigma \qquad \Gamma_1 \vdash_2 M_2 : \tau_1 \ \cdots \ \Gamma_n \vdash_2 M_2 : \tau_n \qquad n \geq 2}{\Gamma, \sum_{i=1}^{n} \Gamma_i \vdash_2 M_1 M_2 : \sigma} \qquad (\to \text{Elim})$$

$$\frac{\Gamma, x : \tau \vdash_2 M : \sigma}{\Gamma \vdash_2 \lambda x.M : \tau \multimap \sigma} \qquad (\multimap \text{Intro})$$

$$\frac{\Gamma_1 \vdash_2 M_1 : \tau \multimap \sigma \qquad \Gamma_2 \vdash_2 M_2 : \tau}{\Gamma_1, \Gamma_2 \vdash_2 M_1 M_2 : \sigma} \qquad (\multimap \text{Elim})$$

Linear Rank 2 Intersection Type System

Structural rules

$$\frac{\Gamma_1, x : \vec{\tau}_1, y : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, y : \vec{\tau}_2, x : \vec{\tau}_1, \Gamma_2 \vdash_2 M : \sigma} \quad \text{(Exchange)}$$

$$\frac{\Gamma_1, x_1 : \vec{\tau}_1, x_2 : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, x : \vec{\tau}_1 \cap \vec{\tau}_2, \Gamma_2 \vdash_2 M[x/x_1, x/x_2] : \sigma} \quad \text{(Contraction)}$$

Linear Rank 2 Intersection Type System

In the Linear Rank 2 Intersection Type System, we say that M has type σ given the environment Γ, and write $\Gamma \vdash_2 M : \sigma$ if it can be obtained from the following *derivation rules*:

$$[x : \tau] \vdash_2 x : \tau \qquad \text{(Axiom)}$$

$$\frac{\Gamma_1, x : \vec{\tau}_1, y : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, y : \vec{\tau}_2, x : \vec{\tau}_1, \Gamma_2 \vdash_2 M : \sigma} \qquad \text{(Exchange)}$$

$$\frac{\Gamma_1, x_1 : \vec{\tau}_1, x_2 : \vec{\tau}_2, \Gamma_2 \vdash_2 M : \sigma}{\Gamma_1, x : \vec{\tau}_1 \cap \vec{\tau}_2, \Gamma_2 \vdash_2 M[x/x_1, x/x_2] : \sigma} \qquad \text{(Contraction)}$$

$$\frac{\Gamma, x : \tau_1 \cap \cdots \cap \tau_n \vdash_2 M : \sigma \qquad n \geq 2}{\Gamma \vdash_2 \lambda x.M : \tau_1 \cap \cdots \cap \tau_n \to \sigma} \qquad (\to \text{Intro})$$

$$\frac{\Gamma \vdash_2 M_1 : \tau_1 \cap \cdots \cap \tau_n \to \sigma \qquad \Gamma_1 \vdash_2 M_2 : \tau_1 \ \cdots \ \Gamma_n \vdash_2 M_2 : \tau_n \qquad n \geq 2}{\Gamma, \sum_{i=1}^{n} \Gamma_i \vdash_2 M_1 M_2 : \sigma} \qquad (\to \text{Elim})$$

$$\frac{\Gamma, x : \tau \vdash_2 M : \sigma}{\Gamma \vdash_2 \lambda x.M : \tau \multimap \sigma} \qquad (\multimap \text{Intro})$$

$$\frac{\Gamma_1 \vdash_2 M_1 : \tau \multimap \sigma \qquad \Gamma_2 \vdash_2 M_2 : \tau}{\Gamma_1, \Gamma_2 \vdash_2 M_1 M_2 : \sigma} \qquad (\multimap \text{Elim})$$

Linear Rank 2 Intersection Type System

$\rightarrow$ rules

$$\frac{\Gamma, x : \tau_1 \cap \cdots \cap \tau_n \vdash_2 M : \sigma \qquad n \geq 2}{\Gamma \vdash_2 \lambda x.M : \tau_1 \cap \cdots \cap \tau_n \rightarrow \sigma} \quad (\rightarrow \text{Intro})$$

$$\frac{\begin{array}{c}\Gamma \vdash_2 M_1 : \tau_1 \cap \cdots \cap \tau_n \rightarrow \sigma \\ \Gamma_1 \vdash_2 M_2 : \tau_1 \quad \cdots \quad \Gamma_n \vdash_2 M_2 : \tau_n \qquad n \geq 2\end{array}}{\Gamma, \sum_{i=1}^{n} \Gamma_i \vdash_2 M_1 M_2 : \sigma} \quad (\rightarrow \text{Elim})$$

Type Inference Algorithm

- We defined a new, simpler type inference algorithm based on Trevor Jim's.

Type Inference Algorithm

- We defined a new, simpler type inference algorithm based on Trevor Jim's.
- Solely based on first order unification.

Type Inference Algorithm

- We defined a new, simpler type inference algorithm based on Trevor Jim's.
- Solely based on first order unification.
- Sound and complete with respect to the Linear Rank 2 Intersection Type System:

Theorem (Soundness)

If $T(M) = (\Gamma, \sigma)$*, then* $\Gamma \vdash_2 M : \sigma$*.*

Theorem (Completeness)

If $\Gamma \vdash_2 M : \sigma$*, then* $T(M) = (\Gamma', \sigma')$ *(for some environment* Γ' *and type* σ'*) and there is a substitution* $\mathbb{S}$ *such that* $\mathbb{S}(\sigma') = \sigma$ *and* $\mathbb{S}(\Gamma') \equiv \Gamma$*.*

Type Inference Algorithm - Example

3. If $M = M_1 M_2$, then:
   - b. if $T(M_1) = (\Gamma'_1, \tau'_1 \cap \cdots \cap \tau'_n \to \sigma'_1)$ (with $n \geq 2$) and, for each $1 \leq i \leq n$, $T(M_2) = (\Gamma_i, \tau_i)$,

     then $T(M) = (\mathbb{S}(\Gamma'_1 + \sum_{i=1}^{n} \Gamma_i), \mathbb{S}(\sigma'_1))$,

     where $\mathbb{S} = UNIFY(\{\tau_i = \tau'_i \mid 1 \leq i \leq n\})$;

Type Inference Algorithm - Example

3. If $M = M_1 M_2$, then:
   b. if $T(M_1) = (\Gamma'_1, \tau'_1 \cap \cdots \cap \tau'_n \to \sigma'_1)$ (with $n \geq 2$) and, for each $1 \leq i \leq n$, $T(M_2) = (\Gamma_i, \tau_i)$,

      then $T(M) = (\mathbb{S}(\Gamma'_1 + \sum_{i=1}^{n} \Gamma_i), \mathbb{S}(\sigma'_1))$,

      where $\mathbb{S} = UNIFY(\{\tau_i = \tau'_i \mid 1 \leq i \leq n\})$;

Example

For $M = (\lambda fx.f(fx))(\lambda y.y)$, we have:

- $T(\lambda fx.f(fx)) = ([], ((\alpha_1 \multimap \alpha_2) \cap (\alpha_2 \multimap \alpha_3)) \to \alpha_1 \multimap \alpha_3)$
- $T(\lambda y.y) = ([], \beta_1 \multimap \beta_1) = ([], \beta_2 \multimap \beta_2)$
- $\mathbb{S} = [\alpha_3/\beta_1, \alpha_3/\beta_2, \alpha_3/\alpha_1, \alpha_3/\alpha_2]$

and so $T(M) = (\mathbb{S}([]), \mathbb{S}(\alpha_1 \multimap \alpha_3)) = ([], \alpha_3 \multimap \alpha_3)$.

Work In Progress

- Type system that extracts quantitative measures (number of reduction steps to normal form).

Work In Progress

- Type system that extracts quantitative measures (number of reduction steps to normal form).
- Type inference algorithm that infers the number of reduction steps to normal form.

Work In Progress

- Type system that extracts quantitative measures (number of reduction steps to normal form).
- Type inference algorithm that infers the number of reduction steps to normal form.
- In the future, we would like to further explore the relation between our definition of linear rank and the traditional definition of rank, adapt the system and algorithm for other evaluation strategies, and extend them for a programming language.

Thank You!