Linear Rank Intersection Types

Fábio Reis[1,2], Sandra Alves[1,2,3], and Mário Florido[1,2]

[1] DCC-FCUP, University of Porto, Porto, Portugal
[2] LIACC - Artificial Intelligence and Computer Science Laboratory
[3] CRACS, INESC-TEC - Centre for Research in Advanced Computing Systems

1 Intersection Types

Intersection type systems [2] characterize termination, in the sense that a λ-term is strongly-normalizable if and only if it is typable in an intersection type system. Thus, typability is undecidable for these systems.

To get around this, some current intersection type systems are restricted to types of finite rank [4, 8, 9, 12], using a notion of rank first defined by Daniel Leivant in [11]. This restriction makes type inference decidable [9]. Despite using finite-rank intersection types, these systems are still very powerful and useful. For instance, rank 2 intersection type systems [4, 8, 12] are more powerful, in the sense that they can type strictly more terms, than popular systems like the ML type system. Intersections arise in different systems in different scopes. Here we follow several previous presentations where intersections are only allowed in the left-hand side of arrow types [2, 3, 8, 13].

Definition 1 (Rank of intersection types)**.** Let $\mathbb{T}_0$ be the set of simple types and $\mathbb{T}_1 = \{\rho_1 \cap \cdots \cap \rho_m \mid \rho_1, \ldots, \rho_m \in \mathbb{T}_0, m \geq 1\}$. The set $\mathbb{T}_k$, of rank k intersection types (for $k \geq 2$), can be defined recursively in the following way ($n \geq 3$):

$$\mathbb{T}_2 = \mathbb{T}_0 \cup \{\sigma \to \tau \mid \sigma \in \mathbb{T}_1, \tau \in \mathbb{T}_2\}$$
$$\mathbb{T}_n = \mathbb{T}_{n-1} \cup \{\sigma_1 \cap \cdots \cap \sigma_m \to \tau \mid \sigma_1, \ldots, \sigma_m \in \mathbb{T}_{n-1}, \tau \in \mathbb{T}_n\}$$

Note that the rank of an intersection type is related to the depth of the nested intersections and it can be easily determined by examining it in tree form: a type is of rank k if no path from the root of the type to an intersection type constructor $\cap$ passes to the left of k arrows.

Quantitative types [1, 5, 6, 10] provide more than just qualitative information about programs and are particularly useful in contexts where we are interested in measuring the use of resources, as they are related to the consumption of time and space in programs. These systems are based on non-idempotent intersection types, using non-idempotence to count the number of evaluation steps and the size of the result.

In this work, we explore a type inference algorithm for non-idempotent rank 2 intersection types. Note that the set of terms typed using idempotent rank 2 intersection types and non-idempotent rank 2 intersection types is not the same. For instance, the term $(\lambda x.xx)(\lambda fx.f(fx))$ is typable with a simple type when using idempotent intersection types, but not when using non-idempotent intersection types. In fact, the only terms typable with a simple type in a non-idempotent intersection type system are the linear terms. This motivated us to come up with a new notion of rank for non-idempotent intersection types, based on linear types, that we here call *linear rank*.

2 Linear Rank

The relation between non-idempotent intersection types and linearity was introduced by Kfoury [10] and further explored by de Carvalho [5] who established its relation with linear logic.

Here we propose a new definition of rank for intersection types, which we call *linear rank* and differs from the previous one in the base case – instead of simple types, *linear rank* 0 intersection types are the linear types (the ones typed in a linear type system – a substructural type system in which each assumption must be used exactly once, corresponding to the implicational fragments of linear logic [7]).

Definition 2 (Linear rank of intersection types)**.** Let $\mathbb{T}_{\mathbb{L}0}$ be the set of linear types and $\mathbb{T}_{\mathbb{L}1} = \{\rho_1 \cap \cdots \cap \rho_m \mid \rho_1, \ldots, \rho_m \in \mathbb{T}_{\mathbb{L}0}, m \geq 1\}$. 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 ($n \geq 3$, $m \geq 2$):

$$\begin{aligned}
\mathbb{T}_{\mathbb{L}2} &= \mathbb{T}_{\mathbb{L}0} \cup \{\rho \multimap \tau \mid \rho \in \mathbb{T}_{\mathbb{L}0}, \tau \in \mathbb{T}_{\mathbb{L}2}\} \\
&\quad \cup \{\sigma_1 \cap \cdots \cap \sigma_m \to \tau \mid \sigma_1, \ldots, \sigma_m \in \mathbb{T}_{\mathbb{L}0}, \tau \in \mathbb{T}_{\mathbb{L}2}\} \\
\mathbb{T}_{\mathbb{L}n} &= \mathbb{T}_{\mathbb{L}n-1} \cup \{\sigma \multimap \tau \mid \sigma \in \mathbb{T}_{\mathbb{L}n-1}, \tau \in \mathbb{T}_{\mathbb{L}n}\} \\
&\quad \cup \{\sigma_1 \cap \cdots \cap \sigma_m \to \tau \mid \sigma_1, \ldots, \sigma_m \in \mathbb{T}_{\mathbb{L}n-1}, \tau \in \mathbb{T}_{\mathbb{L}n}\}
\end{aligned}$$

The idea for this change arose from our interest in using rank-restricted intersection types to estimate the number of evaluation steps of a λ-term while inferring its type. While defining the intersection type system to obtain quantitative information, we realized that the ranks could be potentially more useful for that matter if the base case was changed to types that give more quantitative information in comparison to simple types, which is the case for linear types – for instance, if a term is typed with a *linear rank* 2 intersection type, one knows that its arguments are linear, meaning that they will be used exactly once.

It is not clear, and most likely non-trivial, the relation between the standard definition of rank and our definition of *linear rank*, but it is an interesting question that we would like to explore in the future.

A glimpse of how differently we type linear and non-linear functions is given by the two arrow elimination rules of our Linear Rank 2 Intersection Type System:

$$\frac{\Gamma \vdash_2 M_1 : \rho_1 \cap \cdots \cap \rho_n \to \tau \qquad \Gamma_1 \vdash_2 M_2 : \rho_1 \quad \cdots \quad \Gamma_n \vdash_2 M_2 : \rho_n \qquad n \geq 2}{\Gamma, \sum_{i=1}^{n} \Gamma_i \vdash_2 M_1 M_2 : \tau} \quad (\to \text{ Elim})$$

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

Note that intersection is non-idempotent, thus when a term is typed by the ($\to$ Elim) rule, it is necessarily non-linear.

3 Contributions

The main contributions of this work are: a new notion of *linear rank*, a *linear rank* 2 non-idempotent intersection type system and a type inference algorithm which is sound and complete with respect to the type system. Our algorithm is inspired by previous type inference algorithms for rank 2 idempotent intersection types [8, 12]. Non-idempotent intersection types are quantitative types, thus we argue that our type inference algorithm is a first step towards the automatic inference of quantitative types.

Acknowledgments This work was financially supported within projects UIDB/00027/2020 and UIDB/50014/2020, funded by national funds through the FCT/MCTES (PIDDAC).

References

[1] Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. Non-idempotent intersection types for the lambda-calculus. *Log. J. IGPL*, 25(4):431–464, 2017.

[2] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 10 1980.

[3] Mario Coppo. An extended polymorphic type system for applicative languages. In Piotr Dembinski, editor, *Mathematical Foundations of Computer Science 1980 (MFCS'80), Proceedings of the 9th Symposium, Rydzyna, Poland, September 1-5, 1980*, volume 88 of *Lecture Notes in Computer Science*, pages 194–204. Springer, 1980.

[4] Ferruccio Damiani. Rank 2 intersection for recursive definitions. *Fundamenta Informaticae*, 77(4):451–488, 2007.

[5] Daniel de Carvalho. *Sémantiques de la logique linéaire et temps de calcul*. Phd thesis, Université Aix-Marseille II, 2007.

[6] Philippa Gardner. Discovering needed reductions using type theory. In *TACS*, volume 789 of *LNCS*, pages 555–574. Springer, 1994.

[7] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987.

[8] Trevor Jim. Rank 2 type systems and recursive definitions. *Massachusetts Institute of Technology, Cambridge, MA*, 1995.

[9] A. J. Kfoury and J. B. Wells. Principality and decidable type inference for finite-rank intersection types. In *POPL '99, Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 161–174. ACM, 1999.

[10] Assaf Kfoury. A linearization of the lambda-calculus and consequences. *Journal of Logic and Computation*, 10(3):411–436, 2000.

[11] Daniel Leivant. Polymorphic type inference. In *Proceedings of the 10th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL)*, pages 88–98, 1983.

[12] Steffen Van Bakel. *Intersection type disciplines in lambda calculus and applicative term rewriting systems*. Phd thesis, Mathematisch Centrum, Katholieke Universiteit Nijmegen, 1993.

[13] Steffen van Bakel. Rank 2 intersection type assignment in term rewriting systems. *Fundam. Informaticae*, 26(2):141–166, 1996.