MLF type inference and semi-unification

Didier Rémy

Master internship, 2020

Subject: MLF type inference and semi-unification

Supervisor: Didier Rémy

Location: Centre Inria de Paris

Context

MLF [2, 3] (see also [7, 8]), which merges ML and System F, is neither implicitly nor explicitly typed or, rather, both simultaneously: some expressions can be typed without any type annotation at all, while others require some type annotations. More precisely, only parameters of a function that are used polymorphically in its body need to be annotated.

MLF introduces instantiation-bounded quantification of the form $\forall\alpha{\geq}\sigma.\,\tau$ where σ is itself an arbitrary (polymorphic) type, which makes it slightly more expressive than System F, but in an unessential way. Type inference uses first-order unification and type generalization as in ML, but in the presence of second-order polymorphic types, and exploits polymorphism from type generalization and type annotations, but never guesses polymorphism.

Full type inference for System F amounts to semi-unification [9], which generalizes unification: given a multiset $(\sigma_i, \tau_i)^{i \in I}$ of pairs interpreted as inequations $\sigma_i \leq^? \tau_i$ (instead of equations) to be solved, it searches for substitutions μ that satisfy all the inequations simultaneously, *i.e.* such that for all i in I, the inequation $\mu\sigma_i \leq \mu\tau_i$ holds, which in turn means that there exists a substitution ν_i such that the equation $\nu_i(\mu\sigma_i) = \mu\tau_i$ holds. Semi-unification is unfortunately undecidable and has thus not been much exploited for type inference in System F; only a few decidable subclasses of unification problems [6, 5] have been proposed.

Internship description

The goal of the internship is to relate MLF type inference to semi-unification problems. More precisely, it means finding restrictions of semi-unification problems so that only those corresponding to solvable MLF type inference problems are themselves solvable. In fact, we expect to solve not exactly those problems but a slightly larger category of similar problems where second-order types would still not be guessed but would be better propagated.

While the first objective is a mere transposition of the problem, an other expected output is to gain more insight into propagation of type annotations in MLF and hopefully find a more principled approach.

In fact, several restrictions of MLF [1, 4], which infer less but have a simpler meta-theoretical formalization, have been proposed. We also hope to find new, better compromises for partial type inference for system F.

References

[1] Jacques Garrigue and Didier Rémy. Extending ML with semi-explicit higher-order polymorphism. *Information and Computation*, 155(1/2):134–169, 1999. URL `http://www.springerlink.com/content/m303472288241339/`. A preliminary version appeared in TACS'97.

[2] Didier Le Botlan and Didier Rémy. MLF: Raising ML to the power of System F. In *Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming*, pages 27–38, August 2003.

[3] Didier Le Botlan and Didier Rémy. Recasting MLF. *Information and Computation*, 207 (6):726–785, 2009. ISSN 0890-5401. doi: 10.1016/j.ic.2008.12.006. URL `http://dx.doi.org/10.1016/j.ic.2008.12.006`.

[4] Daan Leijen. Flexible types: robust type inference for first-class polymorphism. In Zhong Shao and Benjamin C. Pierce, editors, *Proceedings of the 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, Savannah, GA, USA, January 21-23, 2009*, pages 66–77. ACM, 2009. doi: 10.1145/1480881.1480891. URL `https://doi.org/10.1145/1480881.1480891`.

[5] Brad Lushman and Gordon V. Cormack. A Larger Decidable Semiunification Problem. In *Proceedings of the 9th ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming*, PPDP '07, pages 143–152, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-769-8. doi: 10.1145/1273920.1273939. URL `http://doi.acm.org/10.1145/1273920.1273939`. event-place: Wroclaw, Poland.

[6] Alberto Oliart and Wayne Snyder. Fast algorithms for uniform semi-unification. *Journal of Symbolic Computation*, 37(4):455 – 484, 2004. ISSN 0747-7171. doi: https://doi.org/10.1016/j.jsc.2003.03.002. URL `http://www.sciencedirect.com/science/article/pii/S0747717103001202`.

[7] Didier Rémy and Boris Yakobowski. A graphical presentation of MLF types with a linear-time unification algorithm. In *Proceedings of the 2007 ACM SIGPLAN International Workshop on Types in Languages Design and Implementation (TLDI'07)*, pages 27–38, Nice, France, January 2007. ACM Press. ISBN 1-59593-393-X. URL `http://gallium.inria.fr/~remy/work/mlf/`.

[8] Didier Rémy and Boris Yakobowski. Efficient Type Inference for the MLF language: a graphical and constraints-based approach. In *The 13th ACM SIGPLAN International Conference on Functional Programming (ICFP'08)*, pages 63–74, Victoria, BC, Canada, September 2008. doi: http://doi.acm.org/10.1145/1411203.1411216.

[9] J. B. Wells. Typability and type checking in System F are equivalent and undecidable. *Annals of Pure and Applied Logic*, 98(1):111 – 156, 1999. ISSN 0168-0072. doi: https://doi.org/10.1016/S0168-0072(98)00047-5. URL `http://www.sciencedirect.com/science/article/pii/S0168007298000475`.