Internship subject (Master 2 or Master 1)

Dataflow explicit futures: Formalisation and experimentation

Main advisor: Ludovic Henrio

Co-advisors: Matthieu Moy and Amaury Maillé

Place: Laboratoire de l'Informatique du Parallélisme (LIP)
École Normale Supérieure de Lyon

Context

A future is a place-holder for a value being computed, and we generally say that a future is resolved when the associated value is computed. In existing languages futures are either implicit, if there is no syntactic or typing distinction between futures and non-future values, or explicit when futures are typed by a parametric type and dedicated functions exist for manipulating futures. We defined in [1] a new form of future, named data-flow explicit futures, with specific typing rules that do not use classical parametric types. The new futures allow at the same time code reuse and the possibility for recursive functions to return futures like with implicit futures, and let the programmer declare which values are futures and where synchronisation occurs, like with explicit futures. We prove that the obtained programming model is as expressive as implicit futures but exhibits a different behaviour compared to explicit futures.

The research report [1] describes a type system and a semantics for dataflow explicit futures. These have been implemented as a modification of the Encore [2, 3] language and its type system.

Futures are used in many languages, but they are central in actor languages like Encore. Actors and active object languages [2, 5] are based on asynchronous communications between mono-threaded entities and massively use futures to represent replies to asynchronous messages. We illustrate our proposal on an active object languages but the approach is generalisable to other languages using futures.

Objectives

The internship should start with an experimentation phase to show the benefits of the approach and experiment with dataflow and control flow futures. Then the internship should proceed with the implementation of control-flow futures, based on the data-flow futures implemented last year during the internship of Amaury Maillé, following the methofdology of [4].

Then the next steps of the internship consist either in a more formal study or further experiments, depending on the preference of the intern.

A more experimental internship consists in the following objectives the objective is

- to explore the new design opportunities offered by dataflow futures,
- to evaluate the performance of programs with different synchronisation patterns offered by different kind of futures.

A theoretical internship should address at least one of the two following objectives:

- Prove the correctness of the optimisations based on (extensions of) the existing semantics
- Formalise the language and prove existing results or the correctness of optimisations in a theorem prover.

Of course, the internship can be partly theoretical and partly experimental. A Master 1 internship would probably consist on a reduced experimental part of the internship though some simple theoretical properties could as well be studied.

References

[1] "Data-flow Explicit Futures" – Ludovic Henrio. `https://hal.archives-ouvertes.fr/hal-01758734`

[2] Stephan Brandauer, Elias Castegren, Dave Clarke, Kiko Fernandez-Reyes, Einar Broch Johnsen, Ka I Pun, Silvia Lizeth Tapia Tarifa, Tobias Wrigstad, and Albert Mingkun Yang. Parallel objects for multicores: A glimpse at the parallel language Encore.

[3] Kiko Fernandez-Reyes, Dave Clarke, Elias Castegren, and Huu-Phuc Vo. Forward to a promising future. In Giovanna Di Marzo Serugendo and Michele Loreti, editors, *Proc. 20th IFIP WG 6.1 Intl. Conf. on Coordination Models and Languages (COORDINATION 2018)*, volume 10852 of *Lecture Notes in Computer Science*, pages 162–180. Springer, 2018.

[4] Kiko Fernandez-Reyes, Dave Clarke, Ludovic Henrio, Einar Broch Johnsen, and Tobias Wrigstad Godot: All the Benefits of Implicit and Explicit Futures In proceedings of 33rd European Conference on Object-Oriented Programming (ECOOP 2019).

[5] `https://team.inria.fr/scale/software/proactive/`

[6] `http://amaille.fr/rapports/memoire_sriv.pdf`