Internship subject (Master 1 – POM)

Implementation and experimentation of dataflow explicit futures

Main advisor: Amaury Maillé

Co-advisors: Matthieu Moy and Ludovic Henrio

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 objectives of the internship are the following:

- Extend the current work [6] that is limited to parametric types with a single type parameter. The idea is to replace a piece of code duplication currently done in the Encore compiler by an additional parameter used in generic type instantiation.
- Stabilise the code and implementing new examples or revisiting the existing Encore programs.
- Experiment on the efficiency of the construct and design optimisations for the execution of programs with dataflow explicit futures. Depending on the implementation solution, different optimisations will be possible, possibly inspired by existing results.

- Experiment with the coexistence of dataflow explicit futures and classical explicit futures in the same programming language. This could be seen as the experimental work lacking in the article [4].

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`