sujet de stage de Master 2 recherche (2023)

Types for complexity analysis in a process calculus

Encadrant : Patrick Baillot (patrick.baillot@univ-lille.fr)
Page web : `https://pro.univ-lille.fr/patrick-baillot/`

Laboratoire d'accueil : CRIStAL, Université de Lille, Équipe *Sycomores*
Site web du laboratoire : `https://www.cristal.univ-lille.fr/?lang=fr`

Localisation : Lille

Context :
Several type systems have been introduced to analyse the time complexity of functional programs, see for instance [HAH12]. With such a system, given a type derivation for a program M, one can extract from it an upper bound on the (sequential) execution time of M on any input. A more recent challenge is that of complexity analysis for parallel or concurrent computation.

Objective :
The π-calculus [SW03] is a formal calculus to study parallel and concurrent computation, just as λ-calculus allows to study functional computation. It represents processes communicating by messages sent through channels. Some systems have recently been proposed [BG21, BGK21, Ghy21, BG22] to analyse the parallel time complexity of π-calculus programs. The first system [BG21] deals with parallel behaviour, while the second one [BGK21] also accounts for concurrent behaviour, at the price of a more involved syntax. In both cases the execution time analysis bounds the time needed to reach an inactive state and is expressed parametrically with respect to the sizes of input messages.

However for concurrent systems there is not yet any standard notion of time complexity similar to that of input-output functional computation. For this reason we propose to extend the kinds of complexity that can be analysed by the types. For instance for some systems we might be interested in the time needed to reach certain particular states (e.g. expressing that a certain task has been completed, even if some computation is still going on in other threads), or we might be interested in expressing the complexity not only with respect to the *sizes* of input messages, but rather with respect to their *number*. For motivating this second extension think of a process that needs to handle a large number of messages, each of which is of small bounded size.

The work in this internship will consist first in exploring which notions of complexity can be relevant for π-calculus concurrent systems, and then in introducing a type system to analyse these complexities and proving its properties.

Expected background : It is expected to have some knowledge about λ-calculus or functional programming languages as well as about type systems. Some notions of concurrent process calculi such as π-calculus or CCS, as well as more generally about concurrency theory, would also be welcome but are not compulsory.

Références

[BG21] Patrick Baillot and Alexis Ghyselen. Types for complexity of parallel computation in pi-calculus. In *Proceedings of ESOP 2021*, volume 12648 of *LNCS*, pages 59–86. Springer, 2021. https ://hal.archives-ouvertes.fr/hal-03126973.

[BG22] Patrick Baillot and Alexis Ghyselen. Types for complexity of parallel computation in pi-calculus. *ACM Trans. Program. Lang. Syst.*, 44(3) :15 :1–15 :50, 2022.

[BGK21] Patrick Baillot, Alexis Ghyselen, and Naoki Kobayashi. Sized types with usages for parallel complexity of pi-calculus processes. In *Proceedings of CONCUR 2021, International Conference on Concurrency Theory, CONCUR 2021*, volume 203 of *LIPIcs*, pages 34 :1–34 :22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[Ghy21] Alexis Ghyselen. *Sized Types Methods and their Applications to Complexity Analysis in Pi-Calculus*. PhD thesis, ENS de Lyon, September 2021.

[HAH12] Jan Hoffmann, Klaus Aehlig, and Martin Hofmann. Multivariate amortized resource analysis. *ACM Trans. Program. Lang. Syst.*, 34(3) :14 :1–14 :62, 2012.

[SW03] Davide Sangiorgi and David Walker. *The pi-calculus : a Theory of Mobile Processes*. Cambridge university press, 2003.