Short-Term Scientific Mission Grant
- APPLICATION FORM[1] -

Action number: CA20111

Applicant name: Andreia Mordido

Details of the STSM

Title: Deadlock-free context-free session types

Start and end date: 19/01/2025 to 25/01/2025

Detail of the cost in EUROS:

- Transport (upload screen capture): 806.07

- Hotel/day (upload screen capture): 177.25 €

- Food+transports/day: 65 €

TOTAL: 2325 €

Goals of the STSM

This STSM concerns program verification (WG3). We target programs that leverage *message-passing* to seamlessly coordinate the action of distributed software components. Building upon traditional data types, *session types* are a typed-based approach to the specification and static verification of message-passing programs.

We are specifically interested in *context-free session types*, designed to handle non-regular, tree-structured data. A distinguishing feature of context-free session types is that they can specify protocols in which processes form *circular topologies*---which are common in practical applications. Although the robust theory underlying context-free session types ensures the verification of protocol conformance, there is a limitation: well-typed programs may still lead to *deadlocks*, a scenario where programs are blocked waiting for messages that never arrive.

We aim to *strengthen the formal guarantees* of *context-free session types* by ensuring the *absence of deadlocks*. This STSM will start a new collaboration between an expert in context-free session types (Mordido, PT) and an expert in session types and deadlock freedom in concurrency (Pérez, NL). It will also kickstart a larger project aimed at establishing a robust formal basis for deadlock-free programs in *FreeST*, an existing typed concurrent programming language that incorporates context-free session types but does not yet exclude deadlocks.

Working Plan

We aim to establish deadlock-freedom guarantees in context-free session types [1, 7] by leveraging nested session types. *Nested session types*, developed by Mordido and collaborators [2], are a metatheory of session types endowed with parametric type constructors and nested recursion. As a logic-based formulation of session types, nested session types ensure deadlock freedom through typing without additional mechanisms [3, 4]. Furthermore,

1 This form is part of the application for a grant to visit a host organisation located in a different country than the country of affiliation. It is submitted to the COST Action MC via e-COST. The Grant Awarding Coordinator coordinates the evaluation on behalf of the Action MC and informs the Grant Holder of the result of the evaluation for issuing the Grant Letter.

nested session types were proved to strictly generalize context-free session types [2]. Therefore, we aim to infer the absence of deadlocks in context-free session types from the guarantees offered by nested session types.

However, the work of Dardha and Pérez has shown that formulations of session types derived from Curry-Howard correspondence between session types and linear logic are intrinsically limited: they cannot ensure deadlock-freedom for circular networks; they can only analyze tree-like networks, i.e., networks in which each pair of processes shares at most one channel [5].

Therefore, our plans for this STSM are to first establish deadlock freedom for context-free session types by assuming that processes form tree-like topologies and then extend the result to circular topologies. The methodology will involve specializing nested session types for context-free session types and then use priorities [6] to allow circular topologies.

This STSM will serve as the starting point for an ambitious research project in which the foundations of deadlock-free context-free session types are brought to the practice of program verification in the form of a new version of the FreeST typed programming language. As such, this STSM will serve as a kick-off for a collaboration that will go beyond this specific mission.

References:

[1] Peter Thiemann and Vasco T. Vasconcelos. Context-free session types. In Jacques Garrigue, Gabriele Keller, and Eijiro Sumii, editors, Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming, ICFP 2016, Nara, Japan, September 18-22, 2016, pages 462–475. ACM, 2016.

[2] Ankush Das, Henry DeYoung, Andreia Mordido, and Frank Pfenning. Nested session types. ACM Trans. Program. Lang. Syst., 44(3):19:1–19:45, 2022.

[3] Luís Caires and Frank Pfenning. Session types as intuitionistic linear propositions. In Paul Gastin and François Laroussinie, editors, CONCUR 2010 - Concurrency Theory, 21th International Conference, CONCUR 2010, Paris, France, August 31-September 3, 2010. Proceedings, volume 6269 of Lecture Notes in Computer Science, pages 222–236. Springer, 2010.

[4] Philip Wadler. Propositions as sessions. In Peter Thiemann and Robby Bruce Findler, editors, ACM SIGPLAN International Conference on Functional Programming, ICFP'12, Copenhagen, Denmark, September 9-15, 2012, pages 273–286. ACM, 2012.

[5] Ornela Dardha and Jorge A. Pérez. Comparing type systems for deadlock freedom. J. Log. Algebraic Methods Program., 124:100717, 2022.

[6] Naoki Kobayashi. A new type system for deadlock-free processes. In Christel Baier and Holger Hermanns, editors, CONCUR 2006 - Concurrency Theory, 17th International Conference, CONCUR 2006, Bonn, Germany, August 27-30, 2006, Proceedings, volume 4137 of Lecture Notes in Computer Science, pages 233–247. Springer, 2006.

[7] Bernardo Almeida, Andreia Mordido, Peter Thiemann and Vasco T. Vasconcelos. Polymorphic lambda calculus with context-free session types. Inf. Comput., 289, 2022.

Expected outputs and contribution to the Action MoU objectives and deliverables.

Working groups to which this mission contributes: WG3.

The main output of this project is a type system for context-free session types where the absence of deadlocks is ensured by typing.

The development of type systems for programming languages fosters the correct implementation of programs *by design*, where the compiler ensures the static verification of programs. The more expressive the types, the richer

the programs can be. Context-free session types are expressive types that allow the specification of communication protocols whose patterns go beyond regular languages, enabling, for instance, the serialization of tree-structured data (e.g., JSON or XML). By providing them with deadlock-freedom guarantees, we ensure that well-typed programs can no longer lead to deadlocks, the impact of which could be significant in terms of both computational resources and money. In short:

- with more expressive types, we enable a more significant number of properties to be expressed as types;
- by ensuring deadlock-freedom at compile-time, we provide further guarantees of the programs' correct behavior.

For these reasons, this work aims to contribute to one of the goals of the Cost Action: *make techniques for program verification more effective and more accessible to all stakeholders*.

We also aim to provide a valuable contribution to WG3, namely contributing to the goal of *increasing the expressiveness of techniques for the verification of program correctness*. The main contribution of this work consists of providing mechanisms for static verification of programs that simultaneously offer expressiveness and guarantees of correct behavior, namely by guaranteeing that well-typed programs do not run into deadlocks.

This mission will also allow us to address at least one capacity building objective: “Create an excellent and inclusive network of researchers in Europe with lasting collaboration beyond the lifetime of the Action”.