Concurrent Separation Logics: Logical Abstraction, Logical Atomicity and Environment Liveness Conditions

Philippa Gardner ✉
Imperial College London, UK

Abstract

Scalable verification for concurrent programs with shared memory is a long-standing, difficult problem. In 2004, O'Hearn and Brookes introduced concurrent separation logic to provide compositional reasoning about coarse-grained concurrent programs with synchronisation primitives (Gödel prize, 2016).

In 2010, I introduced logical abstraction (the fiction of separation) to CSL, developing the CAP logic for reasoning about fine-grained concurrent programs in general and fine-grained lock algorithms in particular. In one logic, it was possible to provide two-sided specifications of concurrent operations, with formally verified implementations and clients.

In 2014, I introduced logical atomicity (the fiction of atomicity) to concurrent separation logics, developing the TaDA logic to capture when individual operations behave atomically. Unlike CAP, where synchronisation primitives leak into the specifications, with TaDA the specifications are "just right" in that they provide more general atomic functions specifications to capture, for example, the full behaviour of lock operations.

In 2021, I introduced environment liveness conditions to concurrent separation logics, developing the TaDA Live logic for reasoning compositionally about the termination of blocking fine-grained concurrent programs. The crucial challenge is how to deal with abstract atomic blocking: that is, abstract atomic operations that have blocking behaviour arising from busy-waiting patterns as found in, for example, fine-grained spin locks. The fundamental innovation is with the design of abstract specifications that capture this blocking behaviour as liveness assumptions on the environment.

In this talk, I will explain this on-going journey in the wonderful world of concurrent separation logics. I will also explain why I have a bright green office chair in the corner of my office, patterned in gold lamé.

Many thanks to my fabulous coauthors on concurrent separation logics: Thomas Dinsdale-Young, Emanuele D'Osualdo, Mike Dodds, Azadeh Farzan, Matthew Parkinson, Pedro da Rocha Pinto, Julian Sutherland, Viktor Vafeiadis and more.

Suggested Reading:

- Peter O'Hearn: Resources, Concurrency and Local Reasoning, *Journal of Theoretical Computer Science*, Festschrift for John C Reynolds 70th birthday, 2007.
- Thomas Dinsdale-Young, Pedro da Rocha Pinto and Philippa Gardner: A Perspective on Specifying and Verifying Concurrent Modules, *Journal of Logical and Algebraic Methods in Programming*, 2018.
- Emanuele D'Osualdo, Azadeh Farzan, Philippa Gardner and Julian Sutherland: TaDA Live: Compositional Reasoning for Termination of Fine-grained Concurrent Programs, *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 2021.

2012 ACM Subject Classification Theory of computation → Concurrency

Keywords and phrases Concurrent separation logic

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2022.2

Category Invited Talk