Modern Separation Logic — Derek Dreyer

Lecture 1 - Introduction
June 29, 2026

1 Introduction

Iris is a framework for building and reasoning about separation logic, in particular, concurrent separation logic.

1.1 A Brief History of Separation Logic

Separation logic [5, 2] (SL) was proposed to extend Hoare logic to address the latter's weakness in reasoning about correctness of pointer-manipulating programs, or programs with mutable, manipulable heaps. In SL, we extend Hoare logic with state assertions P, which states that P holds over some state (or equivalently, P has *ownership* over some state.) The most basic assertion is the cell assertion $x \mapsto v$, which means that x points to some value v. We also have *separating conjunctions* $P \star Q$, which can be read as "the heap splits into two disjoint parts, one satisfying P and the other satisfying Q", or from an ownership point of view, "P and Q own separate and disjoint parts of a heap."

1.2 Concurrent SL

A few years after SL was proposed, O'Hearn [3] and Brookes [1] noted that that separation logic lends itself well to concurrent verification.

$$\frac{\{P_1\}C_1\{Q_1\} \quad \{P_2\}C_2\{Q_2\}}{\{P_1 \star P_2\}\ C_1 \mid\mid C_2\ \{Q_1 \star Q_2\}}\ \textsc{Parallel Composition}$$

Basically, if C_1 and C_2 operate on disjoint heaps, then they can be verified in parallel. However, this assumes that the programs operate *disjointly*. What if the concurrent threads share heap resources?

Compiled By:
Ricky Cheng, Kalab Assefa, Ziyang Wang

To resolve that, we can add the invariant rule, which looks something like the following:

$$\frac{\{P \star R_{\text{Inv}}\} C_{\text{Atomic}} \{Q \star R_{\text{Inv}}\}}{\{P\} C_{\text{Atomic}} \{Q\}} \text{ Invariant Rule}$$

In the above rule, R is an *invariant* resource where threads can get temporary ownership to. However, C must be assumed to be *atomic* due to arbitrary scheduling.

1.3 Origin of Iris

Separation logic evolved over the years, with different systems adding different rules to describe how state changes and control who can make changes to what. This led to increasingly complex systems, each with different foundations and requiring separate soundness proofs [4].

Iris was conceived to address this lack of standardization in separation logic by distilling the complex machinery of previous systems into a small *base logic*, on top of which other systems and reasoning principles can be built. This imitates the spirit of how Rocq's core type theory serves as a common foundation for a wide ecosystem of derived tactics and libraries.

2 Introduction to Separation Logic

The Hoare logic is quite limited in that it can only be used to reason about pure, functional programs. To support more expressive reasoning about programs with a heap, we now turn to a successor of Hoare logic: *separation logic*.

We extend our propositions, terms, and values. As noted before, an assertion P denotes that P has *ownership* over some resource.

$$\begin{array}{llrll} \text{Propositions} & P, Q, R & ::= & \cdots \mid \ell \mapsto v \mid P * Q & \\ \text{Terms} & e & ::= & \cdots \mid \text{new}(e) \mid e_1 \leftarrow e_2 \mid !e & (1) \\ \text{Values} & v & ::= & \cdots \mid \ell & \end{array}$$

The proposition $\ell \mapsto v$ (read "ℓ points to v") asserts that the location ℓ currently stores the value v. The proposition $P * Q$ is called the *separating conjunction*. Like conjunction, it asserts that both P and Q are true. What makes it special—and the distinguishing feature of separation logic—is that it ensures P and Q are satisfied by *disjoint* parts of the heap. That is, the proposition $\ell \mapsto v * \ell \mapsto w$ is false (even if $v = w$), because both conjuncts do not refer to separate parts of the heap. Correspondingly, the proposition $\ell \mapsto v * \ell' \mapsto w$ ensures that $\ell \neq \ell'$ and $\ell \mapsto v$ and $\ell' \mapsto w$.

2.1 Common Rules

We begin by reviewing a few structural rules of SL assertions. As noted before an assertion P denotes that P has *ownership* over some resource.

$$\textsc{SepWeaken}\ \frac{}{P * Q \vdash P} \qquad \textsc{SepTrue}\ \frac{}{P \vdash P * \mathsf{True}} \qquad \textsc{SepComm}\ \frac{}{P * Q \dashv\vdash Q * P} \qquad \textsc{SepSplit}\ \frac{P \vdash P' \qquad Q \vdash Q'}{P * Q \vdash P' * Q'}$$

$$\textsc{SepAssoc}\ \frac{}{P * (Q * R) \dashv\vdash (P * Q) * R} \qquad \textsc{ExistsSep}\ \frac{}{(\exists x : X.\, P(x)) * Q \vdash (\exists x : X.\, P(x) * Q)}$$

$$\textsc{PointsToSep}\ \frac{}{\ell \mapsto v * \ell \mapsto w \vdash \mathsf{False}} \qquad \textsc{PointsToAnd}\ \frac{}{\ell \mapsto v \wedge \ell \mapsto w \vdash v = w}$$

2.2 Magic Wand

In this section, we introduce the *separating implicator* (also known as the *magic wand*).

Intuitively, the magic wand $P \mathrel{-\!\!*} Q$ expresses that Q holds under the assumption of P. That is, Q holds under the assumption of *ownership* of the heap fragment described by P. More formally, $P \mathrel{-\!\!*} Q$ holds of a heap h iff $\forall h' \# h$ (i.e., h' is disjoint from h, defined as $h \# h' \triangleq \mathrm{dom}(h) \cap \mathrm{dom}(h') = \emptyset$): if P holds of h', then Q holds of $h \uplus h'$.

There are only two proof rules for $P \mathrel{-\!\!*} Q$ and they are quite simple. As shown below, they mirror the introduction and elimination rules for standard propositional implication:

$$\frac{P * Q \vdash R}{P \vdash Q \mathrel{-\!\!*} R}\ \textsc{WandIntro} \qquad \frac{P \vdash Q \mathrel{-\!\!*} R}{P * Q \vdash R}\ \textsc{WandElim}$$

With the addition of the magic wand, we can prove one of the rules of separating conjunction that introduced as an axiom in previous definition 2.1:

Lemma 1. ExistsSep
$(\exists x : X.\, P(x)) * Q \vdash \exists x : X.\, P(x) * Q$

Proof. Using WandElim, we can revert the assumption Q from our context, leaving us to prove $(\exists x : X.\, P(x)) \vdash Q \mathrel{-\!\!*} \exists x : X.\, P(x) * Q$. We use ExistElim to eliminate the existential quantifier in our assumptions. We have to show $P(x) \vdash Q \mathrel{-\!\!*} \exists x : X.\, P(x) * Q$ for arbitrary $x : X$. By WandIntro, we have to show $P(x) * Q \vdash \exists x : X.\, P(x) * Q$, which follows by ExistIntro. □

References

[1] Stephen Brookes. A semantics for concurrent separation logic. *Theor. Comput. Sci.*, 375(1–3):227–270, April 2007.

[2] Samin S. Ishtiaq and Peter W. O'Hearn. Bi as an assertion language for mutable data structures. *SIGPLAN Not.*, 36(3):14–26, January 2001.

[3] Peter W. OHearn. Resources, concurrency, and local reasoning. *Theor. Comput. Sci.*, 375(1–3):271–307, April 2007.

[4] Matthew Parkinson. The next 700 separation logics. In *Proceedings of the Third International Conference on Verified Software: Theories, Tools, Experiments*, VSTTE'10, page 169–182, Berlin, Heidelberg, 2010. Springer-Verlag.

[5] John C. Reynolds. Separation logic: A logic for shared mutable data structures. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science*, LICS '02, page 55–74, USA, 2002. IEEE Computer Society.