Separation Logic is incomplete

Hans-Dieter A. Hiep
⟨hdh@drheap.nl⟩

(joint work with Frank S. de Boer)

European Research Network on Formal Proofs
WG3 Workshop on Program Verification
September 17, 2025

Talk overview

1. **The discovery of incompleteness**
2. **The road to completeness**

Part I

The discovery of incompleteness

In the beginning...

- Propositional separation logic
- Semantics: **separation algebras** (monoids)
- Proof system: **bunched logics**
- Soundness and completeness ✓

☞ *The logic of bunched implications*,
O'Hearn, Pym (1999)

☞ *The Semantics of BI and Resource Tableaux*,
Galmiche, Méry, Pym (2005)

☞ *Expressivity properties of Boolean BI through relational models*,
Galmiche and Larchey-Wendling (2006)

Separation logic

- First-order separation logic

$$p, q ::= \ldots \mid p * q \mid p \mathrel{-\!\!*} q \mid (x \hookrightarrow y)$$

- Heaps are **partial functions**
- Scalability argument of separation logic

Separation logic

- First-order separation logic

$$p, q ::= \ldots \mid p * q \mid p \mathrel{-\!\!*} q \mid (x \hookrightarrow y)$$

- Heaps are **partial functions**
- Scalability argument of separation logic

$$(\exists x)(x \hookrightarrow y)$$

Separation logic

- First-order separation logic

$$p, q ::= \ldots \mid p * q \mid p \mathrel{-\!\!*} q \mid (x \hookrightarrow y)$$

- Heaps are **partial functions**
- Scalability argument of separation logic

$$(\exists x)(x \hookrightarrow y) * (\exists x)(x \hookrightarrow y)$$

$$\exists x.\ (x \hookrightarrow y) \wedge \exists z.\ z \neq x \wedge (z \hookrightarrow y)$$

Separation logic

- First-order separation logic

$$p, q ::= \ldots \mid p * q \mid p \mathrel{-\!\!*} q \mid (x \hookrightarrow y)$$

- Heaps are **partial functions**
- Scalability argument of separation logic

$$(\exists x)(x \hookrightarrow y) * (\exists x)(x \hookrightarrow y) * (\exists x)(x \hookrightarrow y)$$

$$\exists x.\ (x \hookrightarrow y) \wedge \exists z.\ z \neq x \wedge (z \hookrightarrow y) \wedge \exists w.\ w \neq x \wedge w \neq z \wedge (w \hookrightarrow y)$$

What about 'points to'?

John C. Reynolds:

"Finally, we give axiom schemata for the predicate $\mapsto$. Regrettably, these are far from complete."

$$(x \mapsto y) \wedge (z \mapsto w) \leftrightarrow (x \mapsto y) \wedge x = z \wedge y = w$$

$$(x \hookrightarrow y) * (z \hookrightarrow w) \rightarrow x \neq z$$

$$\mathbf{emp} \leftrightarrow \forall x.\ \neg(x \hookrightarrow -)$$

- However, 'points to' **interacts** with separating connectives

☞ *Separation logic: A logic for shared mutable data structures*, Reynolds (2002)

What about 'points to'?

John C. Reynolds:

"Finally, we give axiom schemata for the predicate $\mapsto$. Regrettably, these are far from complete."

$$(x \mapsto y) \wedge (z \mapsto w) \leftrightarrow (x \mapsto y) \wedge x = z \wedge y = w$$

$$(x \hookrightarrow y) * (z \hookrightarrow w) \rightarrow x \neq z$$

$$\mathbf{emp} \leftrightarrow \forall x.\ \neg(x \hookrightarrow -)$$

- However, 'points to' **interacts** with separating connectives

☞ *Separation logic: A logic for shared mutable data structures*, Reynolds (2002)

The missing axioms

- Define **emp** as $(\forall x, y.\ \neg(x \hookrightarrow y))$
- Define $(x \mapsto y)$ as $(x \hookrightarrow y) \wedge (\forall z.(z \hookrightarrow -) \rightarrow z = x)$

$$
\begin{array}{ll}
\textbf{A1} & \forall x, y.\ ((x \hookrightarrow y) * \textbf{true}) \rightarrow (x \hookrightarrow y) \\
\textbf{A2} & \forall x, y.\ (x \hookrightarrow y) \rightarrow \neg((x \not\mapsto y) * (x \not\mapsto y)) \\
\textbf{A3} & \forall x, y, z.\ \neg((x \hookrightarrow y) * (x \hookrightarrow z)) \\
\textbf{A4} & \forall x, y, z.\ ((x \hookrightarrow y) \wedge (x \hookrightarrow z)) \rightarrow y = z
\end{array}
$$

- Every separation algebra that satisfies these axioms is isomorphic (**categorical axiomatization**)
- These axioms hold in the standard model

Tool support

$$(x \hookrightarrow -) \land ((x = y \land z = w) \lor (x \neq y \land (y \hookrightarrow z)))$$

$$\equiv$$

$$(x \hookrightarrow -) * ((x \hookrightarrow w) \mathrel{-\!\!*} (y \hookrightarrow z))$$

CVC4/CVC5: bug producing incorrect counter-example

Iris: not provable without adding extra axioms

Tool support

$$(x \hookrightarrow -) \wedge ((x = y \wedge z = w) \vee (x \neq y \wedge (y \hookrightarrow z)))$$

$$\equiv$$

$$(x \hookrightarrow -) * ((x \hookrightarrow w) \mathrel{-\!\!*} (y \hookrightarrow z))$$

CVC4/CVC5: bug producing incorrect counter-example

Iris: not provable without adding extra axioms

Part II

The road to completeness

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

General separation logic

- Desiderata: finitary proof system, soundness, completeness
- Gödel's incompleteness of arithmetic
- Generalize to arbitrary models: weak, full, general
- All finite heaps: lacks compactness, so not complete
- All (infinite) heaps: expressivity of finiteness (not compact)
- Henkin's general models with first-order purely definable heaps

☞ *Model theory of second order logic*
Väänänen, Jouko (2023)

First-order hybrid separation logic

- Generalization of **satisfaction operator** @ of hybrid logic
- Logical counterpart of '**virtual memory**'

$$p@q(x, y)$$

where variables x, y in q are bound by @, and q is functional.

Example (Proof rules)

- $((t \hookrightarrow t')@q) \leftrightarrow q[x, y := t, t']$
- $$\frac{((p * q)@r) \quad (r \equiv R_1 \uplus R_2) \to (p@R_1) \to (q@R_2) \to r'}{r'}$$

- Semantics: heap **extensionality** and **comprehension**
- Prototype in logical framework Coq/Rocq

First-order hybrid separation logic

- Generalization of **satisfaction operator** @ of hybrid logic
- Logical counterpart of '**virtual memory**'

$$p@q(x, y)$$

where variables x, y in q are bound by @, and q is functional.

Example (Proof rules)

- $((t \hookrightarrow t')@q) \leftrightarrow q[x, y := t, t']$
- $$\frac{((p * q)@r) \quad (r \equiv R_1 \uplus R_2) \to (p@R_1) \to (q@R_2) \to r'}{r'}$$

- Semantics: heap **extensionality** and **comprehension**
- Prototype in logical framework Coq/Rocq

First-order hybrid separation logic

- Generalization of **satisfaction operator** @ of hybrid logic
- Logical counterpart of '**virtual memory**'

$$p@q(x, y)$$

where variables x, y in q are bound by @, and q is functional.

Example (Proof rules)

- $((t \hookrightarrow t')@q) \leftrightarrow q[x, y := t, t']$
- $$\frac{((p * q)@r) \quad (r \equiv R_1 \uplus R_2) \to (p@R_1) \to (q@R_2) \to r'}{r'}$$

- Semantics: heap **extensionality** and **comprehension**
- Prototype in logical framework Coq/Rocq

First-order hybrid separation logic

- Generalization of **satisfaction operator** @ of hybrid logic
- Logical counterpart of '**virtual memory**'

$$p@q(x,y)$$

where variables x, y in q are bound by @, and q is functional.

Example (Proof rules)

- $((t \hookrightarrow t')@q) \leftrightarrow q[x,y := t,t']$
- $$\frac{((p * q)@r) \quad (r \equiv R_1 \uplus R_2) \to (p@R_1) \to (q@R_2) \to r'}{r'}$$

- Semantics: heap **extensionality** and **comprehension**
- Prototype in logical framework Coq/Rocq

Full separation logic

- Expressivity of (Dedekind-)finite universe:

$$\blacksquare(\mathit{tot}(\hookrightarrow) \wedge \mathit{inj}(\hookrightarrow) \to \mathit{surj}(\hookrightarrow))$$

- Open problem: can you express that heap has finite domain?
- First-order logic $\subset$ full separation logic $\stackrel{?}{=}$ second-order logic
- Breaking the 'local' spell of separation logic
  i.e. having more than one heap in scope