Foundations and Applications of

Separation Logic

CIS 6700, Spring 2024

Benjamin C. Pierce

University of Pennsylvania

What is Separation Logic?

A little program:

$$\mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

(pause to take in notation…)

H.T. John Reynolds

What does it do??

$$\mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

H.T. John Reynolds

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

H.T. John Reynolds

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

Invariant for the while loop:

$$\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

where…

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

(pause again for notation…)

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

Does this work?

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

Does this work?

No!

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

Does this work? No!

Can we fix it?

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta)$$
$$\wedge\ (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{i}, \mathsf{k}) \wedge \mathbf{reachable}(\mathsf{j}, \mathsf{k}) \Rightarrow \mathsf{k} = \mathbf{nil}),$$

where

$$\mathbf{reachable}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists n \geq 0.\ \mathbf{reachable}_n(\mathsf{i}, \mathsf{j})$$

$$\mathbf{reachable}_0(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \mathsf{i} = \mathsf{j}$$

$$\mathbf{reachable}_{n+1}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists \mathsf{a}, \mathsf{k}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{k} \wedge \mathbf{reachable}_n(\mathsf{k}, \mathsf{j})$$

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta)$$
$$\wedge\ (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{i}, \mathsf{k}) \wedge \mathbf{reachable}(\mathsf{j}, \mathsf{k}) \Rightarrow \mathsf{k} = \mathbf{nil}),$$

where

$$\mathbf{reachable}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists n \geq 0.\ \mathbf{reachable}_n(\mathsf{i}, \mathsf{j})$$

$$\mathbf{reachable}_0(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \mathsf{i} = \mathsf{j}$$

$$\mathbf{reachable}_{n+1}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists \mathsf{a}, \mathsf{k}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{k} \wedge \mathbf{reachable}_n(\mathsf{k}, \mathsf{j})$$

Now does it work?

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta)$$
$$\wedge\ (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{i}, \mathsf{k}) \wedge \mathbf{reachable}(\mathsf{j}, \mathsf{k}) \Rightarrow \mathsf{k} = \mathbf{nil}),$$

where

$$\mathbf{reachable}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists n \geq 0.\ \mathbf{reachable}_n(\mathsf{i}, \mathsf{j})$$

$$\mathbf{reachable}_0(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \mathsf{i} = \mathsf{j}$$

$$\mathbf{reachable}_{n+1}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists \mathsf{a}, \mathsf{k}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{k} \wedge \mathbf{reachable}_n(\mathsf{k}, \mathsf{j})$$

Now does it work?

Still no!

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

$$LREV \;=\; \mathsf{j} := \mathbf{nil}\,;\, \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\,;\, [\mathsf{i}+1] := \mathsf{j}\,;\, \mathsf{j} := \mathsf{i}\,;\, \mathsf{i} := \mathsf{k})$$

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta)$$
$$\wedge\ (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{i}, \mathsf{k}) \wedge \mathbf{reachable}(\mathsf{j}, \mathsf{k}) \Rightarrow \mathsf{k} = \mathbf{nil}),$$

where

$$\mathbf{reachable}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists n \geq 0.\ \mathbf{reachable}_n(\mathsf{i}, \mathsf{j})$$

$$\mathbf{reachable}_0(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \mathsf{i} = \mathsf{j}$$

$$\mathbf{reachable}_{n+1}(\mathsf{i}, \mathsf{j}) \stackrel{\text{def}}{=} \exists \mathsf{a}, \mathsf{k}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{k} \wedge \mathbf{reachable}_n(\mathsf{k}, \mathsf{j})$$

Now does it work? Still no!

Can we fix it?

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

$$LREV \quad = \quad \mathsf{j} := \mathbf{nil}\ ;\ \mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}\ (\mathsf{k} := [\mathsf{i}+1]\ ;\ [\mathsf{i}+1] := \mathsf{j}\ ;\ \mathsf{j} := \mathsf{i}\ ;\ \mathsf{i} := \mathsf{k})$$

$$\begin{aligned}
&(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} \wedge \mathsf{list}\ \beta\ \mathsf{j} \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta) \\
&\quad \wedge (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{i}, \mathsf{k}) \wedge \mathbf{reachable}(\mathsf{j}, \mathsf{k}) \Rightarrow \mathsf{k} = \mathbf{nil}) \\
&\quad \wedge\ \mathsf{list}\ \gamma\ \mathsf{x} \\
&\quad \wedge (\forall \mathsf{k}.\ \mathbf{reachable}(\mathsf{x}, \mathsf{k}) \\
&\qquad \wedge (\mathbf{reachable}(\mathsf{i}, \mathsf{k}) \vee \mathbf{reachable}(\mathsf{j}, \mathsf{k})) \Rightarrow \mathsf{k} = \mathbf{nil})
\end{aligned}$$

Oof

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{i} = \mathbf{nil} \qquad \mathsf{list}(\mathsf{a} \cdot \alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \hookrightarrow \mathsf{a}, \mathsf{j} \wedge \mathsf{list}\ \alpha\ \mathsf{j}$$

What's the problem?

In programs, locality (lack of sharing) is the norm

But in specifications, the possibility of sharing is the default

Could we change this default?

Separating conjunction to the rescue!

$P * Q$ = P and Q hold for <u>disjoint</u> portions of the heap

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i}\ *\ \mathsf{list}\ \beta\ \mathsf{j}) \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

$$\{\mathsf{list}\ \alpha\ \mathsf{i}\}\ LREV\ \{\mathsf{list}\ \alpha^\dagger\ \mathsf{j}\}$$

Separating conjunction to the rescue!

$P * Q$ = P and Q hold for <u>disjoint</u> portions of the heap

$$(\exists \alpha, \beta.\ \mathsf{list}\ \alpha\ \mathsf{i} * \mathsf{list}\ \beta\ \mathsf{j}) \wedge \alpha_0^\dagger = \alpha^\dagger \cdot \beta$$

$$\{\mathsf{list}\ \alpha\ \mathsf{i}\}\ LREV\ \{\mathsf{list}\ \alpha^\dagger\ \mathsf{j}\}$$

And by the *frame rule*…

$$\{\mathsf{list}\ \alpha\ \mathsf{i} * \mathsf{list}\ \gamma\ \mathsf{k}\}\ LREV\ \{\mathsf{list}\ \alpha^\dagger\ \mathsf{j} * \mathsf{list}\ \gamma\ \mathsf{k}\}$$

History

- **Burstall ‘72**: correctness proofs for imperative data structures using “distinct nonrepeating tree system” assertions
- **Reynolds ‘99**: Separating conjunction in a flawed extension of Hoare Logic
- **Ishtiaq and O’Hearn ‘00 / Reynolds ’01**: Sound intuitionistic version of the logic
  - Ishtiaq and O’Hearn also introduced *separating implication,* $P \mathrel{-\!\!*} Q$
  - and a more expressive classical version of the logic that supports explicit storage deallocation
- **O’Hearn:** Frame rule
- **Reynolds:** Generalization to unrestricted address arithmetic
- **Brookes ‘04**: Concurrent separation logic (ownership transfer)
- **Since then:** A flood of coolness!

This Course

*… and open to your preferences and suggestions!

Rough* plan for the semester

- *Separation Logic Foundations* textbook, by Charguéraud (4-6 weeks)
- Seminal papers by Reynolds and others (1-2 weeks)
- Iris (2 weeks)
- CN (2-3 weeks)
- Potential further topics (1-2 weeks)
  - VST
  - Concurrent Separation Logic
  - Infer
  - …
- Project presentations (1-2 weeks)

Introductions

- Who are you?
- What is your interest in separation logic?
- How much do you already know about it?

Logistics

- Prerequisites:
  - CIS 5000 or equivalent
- Communication
  - Via EdStem
- Auditors:
  - Drop me an email to make sure you're added to EdStem, etc.
- Required software
  - Coq 8.17
  - SLF files (from a private clone of the SF site, a la CIS 5000)

Grading

- Class participation
  - maybe including giving part of a lecture?
- Homework assignments
  - in Coq
  - in pairs if you want
  - with solutions “incrementally available”
- Course project (designed by you)

Homework

For next Monday:

- Install: Coq 8.17 and SLF
- Read:
  - O'Hearn, *Separation Logic.* CACM, Feb 2019
- Read and do recommended exercises:
  - SLF chapters Preface, Basic, and Repr

And now…

On to SLF…