Separation Logic

Niccolò Piazzesi

Università degli studi di Pisa
Anno Accademico 2021-22

January 3, 2022

Outline

Outline

Brief recap: reasoning about code

- Program semantics described by logical conditions satisfied by language constructs
- Classical model, first put forward by Robert W. Floyd and Tony Hoare

Floyd-Hoare Logic in 1 slide

$$\{P\}S\{Q\}$$

- P : pre-conditions
- S : statement
- Q : post conditions

Partial correctness: **If the inital state fullfils pre-conditions and the statement terminates**, the final state satisfies the post conditions.

Total correctness: **If the initial state fullfils the pre-conditions** then the statement terminates and the final state satisfies the post-conditions.

Limitations

Does not work for non terminating programs

Limitations

Does not work for non terminating programs

Becomes complex with modular constructs such as objects and unconditional jumps

Limitations

Does not work for non terminating programs

Becomes complex with modular constructs such as objects and unconditional jumps

Global view of state becomes a burden when introducing pointers(think of pointer aliasing..)

Motivating example

```c
void deletetree(struct node *root){
    if(root != NULL){
    struct node *left = root->l;
    struct node *right = root->r;
    deletetree(left);
    deletetree(right);
    free(root);
    }
}
```

How can we prove memory safety?

First attempt

Outline

Earliest work on separation logic in the late 90s.

The model

$$\text{Ints} \triangleq \{\cdots, -1, 0, 1, \cdots\}$$

$$\text{Atoms, Locations} \subseteq \mathit{Ints}$$

$$\text{Stores} \triangleq \text{Variables} \rightharpoonup_{\mathit{fin}} \text{Ints}$$

$$\text{Variables} \triangleq \{x, y, \cdots\}$$

$\text{Locations} \cap \text{Atoms} = \{\}$, $\text{nil} \in \text{Atoms}$

$$\text{Heaps} \triangleq \text{Locations} \rightharpoonup_{\mathit{fin}} \text{Ints}$$

$$\text{States} \triangleq \text{Stores} \times \text{Heaps}$$

$$[\![E]\!]_s \in \mathit{Ints},\ [\![B]\!]_s \in \{\text{true, false}\}$$

$$h \in \mathit{Heaps},\ h[E] \in \mathit{Ints}$$

The language

Expressions:

$$\begin{aligned} E, F, G &:= x, y, \cdots \mid 0 \mid 1 \mid E + F \mid E \times F \mid E - F \\ B &:= \mathit{false} \mid B \Rightarrow B \mid E = F \mid E < F \mid \mathit{isatom?}(E) \mid \mathit{isloc?}(E) \end{aligned}$$

The language

Expressions:

$$E, F, G := x, y, \cdots \mid 0 \mid 1 \mid E + F \mid E \times F \mid E - F$$

$$B := \mathit{false} \mid B \Rightarrow B \mid E = F \mid E < F \mid \mathit{isatom?}(E) \mid \mathit{isloc?}(E)$$

Assertions:

$$\begin{array}{rll} P, Q, R ::= & B \mid E \mapsto F & \mathit{Atomic\ Formulae} \\ & \mid \mathit{false} \mid P \Rightarrow Q \mid \forall x.P & \mathit{Classical\ Logic} \\ & \mid \mathit{emp} \mid P * Q \mid P \rightarrow * Q & \mathit{Spatial\ Connectives} \end{array}$$

The language

Expressions:

$$E, F, G := x, y, \cdots \mid 0 \mid 1 \mid E + F \mid E \times F \mid E - F$$
$$B := \mathit{false} \mid B \Rightarrow B \mid E = F \mid E < F \mid \mathit{isatom?}(E) \mid \mathit{isloc?}(E)$$

Assertions:

$$\begin{array}{rll} P, Q, R ::= & B \mid E \mapsto F & \mathit{Atomic\ Formulae} \\ & \mid \mathit{false} \mid P \Rightarrow Q \mid \forall x.P & \mathit{Classical\ Logic} \\ & \mid \mathit{emp} \mid P * Q \mid P \rightarrow * Q & \mathit{Spatial\ Connectives} \end{array}$$

$\neg P = P \Rightarrow \mathit{False}$
$\mathit{true} = \neg(\mathit{false})$
$P \vee Q = \neg(P) \Rightarrow Q$

The language

Expressions:

$$\begin{aligned} E, F, G &:= x, y, \cdots \mid 0 \mid 1 \mid E + F \mid E \times F \mid E - F \\ B &:= \mathit{false} \mid B \Rightarrow B \mid E = F \mid E < F \mid \mathit{isatom?}(E) \mid \mathit{isloc?}(E) \end{aligned}$$

Assertions:

$$\begin{aligned} P, Q, R ::= B \mid E \mapsto F & \quad \mathit{Atomic\ Formulae} \\ \mid \mathit{false} \mid P \Rightarrow Q \mid \forall x.P & \quad \mathit{Classical\ Logic} \\ \mid \mathit{emp} \mid P * Q \mid P \rightarrow * Q & \quad \mathit{Spatial\ Connectives} \end{aligned}$$

$$\begin{aligned} \neg P &= P \Rightarrow \mathit{False} \\ \mathit{true} &= \neg(\mathit{false}) \\ P \vee Q &= \neg(P) \Rightarrow Q \end{aligned}$$

$$\begin{aligned} P \wedge Q &= \neg(\neg P \vee \neg Q) \\ \exists x.P &= \neg \forall x. \neg P \end{aligned}$$

Semantics

Outline

Outline

References I

Peter O'Hearn, John Reynolds, and Hongseok Yang.
Local reasoning about programs that alter data structures.
In *International Workshop on Computer Science Logic*, pages 1–19. Springer, 2001.

Cristiano Calcagno, Dino Distefano, Jérémy Dubreil, Dominik Gabi, Pieter Hooimeijer, Martino Luca, Peter O'Hearn, Irene Papakonstantinou, Jim Purbrick, and Dulma Rodriguez.
Moving fast with software verification.
In *NASA Formal Methods Symposium*, pages 3–11. Springer, 2015.

Dino Distefano, Peter W. O'Hearn, and Hongseok Yang.
A local shape analysis based on separation logic.
In Holger Hermanns and Jens Palsberg, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 287–302, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

References II

Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn.
Smallfoot: Modular automatic assertion checking with separation logic.
In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, and Willem-Paul de Roever, editors, *Formal Methods for Components and Objects*, pages 115–137, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

James Brotherston, Nikos Gorogiannis, Max Kanovich, and Reuben Rowe.
Model checking for symbolic-heap separation logic with inductive predicates.
ACM SIGPLAN Notices, 51(1):84–96, 2016.

References III

Josh Berdine, Byron Cook, and Samin Ishtiaq.
Slayer: Memory safety for systems-level code.
In *International Conference on Computer Aided Verification*, pages 178–183. Springer, 2011.