Backwards and Forwards in Separation Logic

Peter Höfner
(joint work with C. Bannister and G. Klein)

May 2018

Floyd-Hoare Logic (Forwards and Backwards)

- Hoare triple $\{P\}\ C_1\ \{Q\}$
- partial and total correctness
- if $\{P\}\ C_1\ \{Q\}$ and $\{Q\}\ C_2\ \{R\}$ then $\{P\}\ C_1\,;C_2\ \{R\}$
- strengthening and weakening

$$\frac{P_2 \Rightarrow P_1 \quad \{P_1\}\ C\ \{Q_1\} \quad Q_1 \Rightarrow Q_2}{\{P_2\}\ C\ \{Q_2\}}$$

- weakest precondition $\mathtt{wp}(C, Q)$

$$\mathtt{wp}(C_1\,; C_2, Q) = \mathtt{wp}(C_1, \mathtt{wp}(C_2, Q))$$

- similar for strongest postcondition $\mathtt{sp}(C, P)$

Separation Logic (Reynolds, O'Hearn et al.)

- extension to Hoare logic
- based on separation a*lgebras* of abstract heaps
- captures the notion of *disjointness* in the world

Frame Rule

$$\frac{\{P\}\ C\ \{Q\}}{\{P * R\}\ C\ \{Q * R\}}\ (mod(C) \cap fv(R) = \emptyset)$$

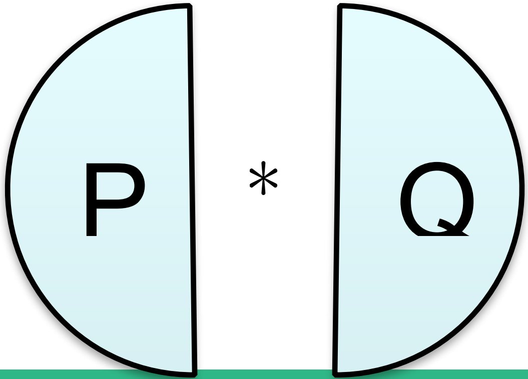
- R is the 'Frame'
  - extending an environment with a disjoint portion changes nothing
  - local reasoning
  - compositional

Separation Logic (Reynolds, O'Hearn et al.)

Motivation

- $s, h \models P$
  where *s* is a store, *h* is a heap, and *P* is an *assertion* over the given store and heap

$$\begin{array}{ll} & s, h \models P * Q \\ \Leftrightarrow & \exists h_1, h_2.\ h_1 \perp h_2 \text{ and} \\ & h = h_1 \cup h_2 \text{ and } s, h_1 \models P \text{ and } s, h_2 \models Q \end{array}$$



Separation Logic II (Forwards and Backwards)

- problem with frame calculation
- specification: $\{p \mapsto a * q \mapsto b\}\ \mathrm{swap}\ p\ q\ \{p \mapsto b * q \mapsto a\}$
- assume the following precondition for forward reasoning

$$r \mapsto a * q \mapsto b * s \mapsto c * t \mapsto d * p \mapsto a$$

- how to find the frame
- situation gets worse in case other operators of separation logic are used

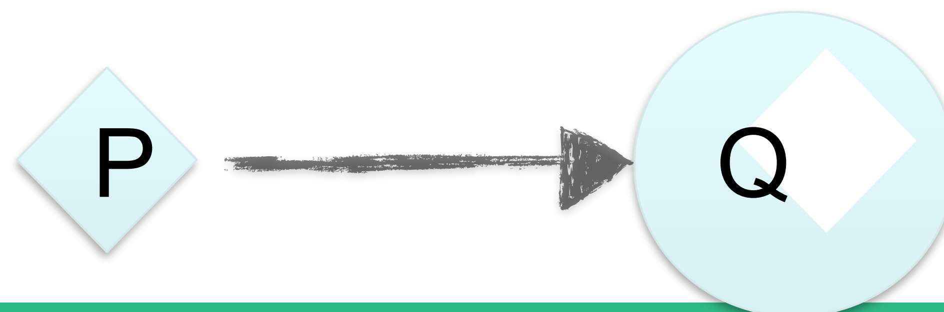
Separating Implication

Magic Wand

- separating Implication $P \mathrel{-\!\!*} Q$
  - extending by P produces Q over the combination

- describes a *mapping* between heaps and 'holes'

$$s, h \models P \mathrel{-\!\!*} Q \quad \Leftrightarrow \quad \forall h'.\ (h' \perp h \text{ and } s, h' \models P) \\ \text{implies } s, h' \cup h \models Q$$



Separation Algebras

- separation logic can be lifted to algebra

- allows abstract reasoning
- transfers knowledge
- ideal for interactive and automated theorem proving

Conjunction version Implication

- modus ponens

$$Q * (Q \mathrel{-\!\!*} P) \;\Rightarrow\; P$$

- currying/decurrying

$$(P * Q \;\Rightarrow\; R) \;\Leftrightarrow\; (P \;\Rightarrow\; Q \mathrel{-\!\!*} R)$$

Galois connection
(gives plenty of properties for free)

Relationships between Operators

$$P * Q \xleftrightarrow{\text{Galois}} P \mathrel{-\!\!*} Q$$

Backwards Reasoning (Recap)

- backward reasoning / reasoning in weakest precondition style

- for given postcondition Q and given program C, determine weakest precondition $\texttt{wp}(C, Q)$
- but what about separation logic where frames occur?

$$\{P * R\}\ C\ \{Q * R\}$$

(problem with frame calculation)

Backwards Reasoning II

- from Galois connection we get *

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \quad \Leftrightarrow \quad (\forall R.\ \{P * (Q \mathrel{-\!\!*} R)\}\ C\ \{R\})$$

- used to transform specifications for example

$$\{p \mapsto \_ * R\}\ \mathrm{set\_ptr}\ p\ v\ \{p \mapsto v * R\}$$

* only in a setting where there are no free variable exist (as in our Isabelle/HOL implementation)

Backwards Reasoning II

- from Galois connection we get *

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \quad \Leftrightarrow \quad (\forall R.\ \{P * (Q \mathrel{-\!\!*} R)\}\ C\ \{R\})$$

- used to transform specifications
  for example

$$\{p \mapsto \_ * (p \mapsto v \mathrel{-\!\!*} R)\}\ \mathrm{set\_ptr}\ p\ v\ \{R\}$$

* only in a setting where there are no free variable exist (as in our Isabelle/HOL implementation)

Backwards Reasoning III

- allows full backwards reasoning without calculating the frame in every step

- supported by an Isabelle/HOL-framework

- easy patterns (alternation between implication and conjunction) allow automated simplifications

Forward Reasoning II

- ideal world

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \quad \Leftrightarrow \quad (\forall R.\ \{R\}\ C\ \{Q * (P \multimap\!\circledast\ R)\})$$

where $\multimap\!\circledast$ is some subtraction operator

More Separation Logic

- there is another operator in the literature: septraction

$$s, h \models P \mathbin{-\!\circledast} Q$$

$$\Leftrightarrow \; \exists h_2.\, h \text{ subheap of } h_2 \text{ and } s, h_2 - h \models P \text{ and } s, h_2 \models Q$$

- algebraically:

$$P \mathbin{-\!\circledast} Q \;\Leftrightarrow\; \neg(P \mathbin{-\!\!*} (\neg Q))$$

Forward Reasoning II

- ideal world seemingly impossible

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \not\Leftrightarrow (\forall R.\ \{R\}\ C\ \{Q * (P \multimap\!\circledast\ R)\})$$

- can't describe what happens in case where precondition doesn't hold

$$\{emp\}\ \text{delete}\ p\ \{??\}$$

Relationships between Operators

$$\begin{tikzcd} P * Q \arrow[r, leftrightarrow, bend left, "\text{Galois}"] & P \mathrel{-\!\!*} Q \arrow[d, leftrightarrow, "\text{dual}"'] \\ & P \mathrel{-\!\!\circledast} Q \end{tikzcd}$$

Separating ‘Coimplication’

Magic Snake

$$P \rightsquigarrow\!\!\ast\, Q \;\Leftrightarrow\; \neg(P \ast (\neg Q))$$

- removing P produces Q over the reduction
- every time we can find a P in our heap, the rest of the heap is a Q

$$s, h \models P \rightsquigarrow\!\!\ast\, Q \;\Leftrightarrow\; \forall h_1\; h_2. (h_1 \perp h_2 \;\text{ and }\; h = h_1 \cup h_2 \text{ and } s, h_1 \models P_1) \\ \text{implies } s, h_2 \models Q$$

Separating 'Coimplication'

Magic Snake

$$P \rightsquigarrow\!\!\ast\, Q \;\Leftrightarrow\; \neg(P \ast (\neg Q))$$

$$(P \multimap\!\circledast\; Q \;\Rightarrow\; R) \;\Leftrightarrow\; (Q \;\Rightarrow\; (P \rightsquigarrow\!\!\ast\, Q))$$

(Galois connection)

- many properties come for free from the Galois connection

Relationships between Operators

(complete)

$$\begin{tikzcd} P * Q \arrow[rr, leftrightarrow, bend left, "\text{Galois}"] \arrow[d, leftrightarrow, "\text{dual}"] && P \mathrel{-\!\!*} Q \arrow[d, leftrightarrow, "\text{dual}"'] \\ P \mathrel{\rightsquigarrow\!\!\!*} Q \arrow[rr, leftrightarrow, bend right, "\text{Galois}"'] && P \mathrel{-\!\!\circledast} Q \end{tikzcd}$$

Specifications with Separating Coimplication

- P not satisfied by any subheap

$$P \rightsquigarrow\!\!\ast\, false$$

- specification of delete

$$\{p \mapsto \_ \rightsquigarrow\!\!\ast\, R\} \text{ delete } p \ \{R\}$$

Back to Forward Reasoning

- Ideal world seemingly impossible

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \not\Leftrightarrow (\forall R.\ \{R\}\ C\ \{Q * (P \multimap\!\circledast\ R)\})$$

- Relax specifications/requirements

$$\{P \rightsquigarrow\!\!\ast R\}\ C\ \{Q * R\}$$

- another example

$$\{p \mapsto \_ \rightsquigarrow\!\!\ast R\}\ \mathrm{set\_ptr}\ p\ v\ \{p \mapsto v * R\}$$

Forward Reasoning III

- Ideal world seemingly impossible

$$(\forall R.\ \{P * R\}\ C\ \{Q * R\}) \ \not\Leftrightarrow\ (\forall R.\ \{R\}\ C\ \{Q * (P \mathbin{-\circledast} R)\})$$

- By Galois connections and dualities we get a rule for forward reasoning

$$(\forall R.\ \{P \mathbin{\rightsquigarrow\!\!*} R\}\ C\ \{Q * R\}) \ \Leftrightarrow\ (\forall R.\ \{R\}\ C\ \{Q * (P \mathbin{-\circledast} R)\})$$

Forwards Reasoning IV

- allows forwards reasoning without calculating the frame in every step
- supported in Isabelle/HOL
- easy patterns (alternation between implication and conjunction) allow automated simplifications
- *partial correctness* only
  if failure occurs anything is possible

Problems, Excitements & Open Questions

Generalised Correctness

- introduce explicit one or many failure states
- always describe what actually occurs

$$\{P\}\ C\ \{Q\} \ \Leftrightarrow\ \forall s.\ P(s) \rightarrow Q(C(s))$$

- requirements/wishes:
  - failed program execution cannot be recovered
  - failure is separate from false
  - used in combination with Hoare logic
  - keep the algebraic connections
  - we can determine whether or not we succeeded

A Single Failure Element

- assumptions:

$$\text{false} * \mathit{fail} = \text{false} \quad \text{and} \quad P * \mathit{fail} = \mathit{fail} \ \ (P \neq \text{false})$$

- associativity is lost (or no ‘inverses’ or $\text{false} = \mathit{fail}$)

$$(P * P) * \mathit{fail} = \text{false} * \mathit{fail} = \text{false} \quad \text{and}$$

$$P * (P * \mathit{fail}) = P * \mathit{fail} = \mathit{fail}$$

- predicates define a partial order; where does fail sit

if $\mathit{fail} \Rightarrow P$
then the weakening rule of Hoare logic is lost

A Single Failure Element

- assumptions:

$$P * \mathit{fail} = \mathit{fail} \quad (\text{for all } P)$$

- the Galois connections are lost
  - still gives a decent model
  - not useful for our approach for forward and backwards reasoning

- heaps define a partial order; where does fail sit

$$\text{if } \mathit{fail} \Rightarrow P$$
$$\text{then the weakening rule of Hoare logic is lost}$$

Failure ‘Flag’ for Every Set

- predicate is set of heaps (satisfying predicate)
- add a single element to this set

- basically same problems as before
(even when considering more sophisticated orderings, such as Egli-Milner, Hoare, Plotkin, Smyth, …)

Failure ‘Flag’ for Every Heap

- each heap carries a flag
- isomorphic to pairs of sets

- similar problems as before

Failure 'Flag' for Every Heap

- each heap carries a flag
- isomorphic to pairs of sets

- similar problems as before
- however some models 'work'

Extending the Model

- New Septraction operator for grabbing resources

Old

$$
\begin{aligned}
& s, h \models P \multimap\!\!\circledast\ Q \\
\Leftrightarrow\ & \exists h_2 .\, h \text{ subheap of } h_2 \text{ and } s, h_2 - h \models P \text{ and } s, h_2 \models Q \\
\Leftrightarrow\ & \exists h_1, h_2 .\, h_1 \models P \text{ and } s, h_1 \models Q \text{ and} \\
& \text{and } h \perp h_1,\ h_2 = h + h_1
\end{aligned}
$$

New

$$
\begin{aligned}
& s, h \models P \multimap\!\!\circledast\ Q \\
\Leftrightarrow\ & \exists h_1, h_2 .\, h_1 \models P \text{ and } s, h_1 \models Q \text{ and} \\
& \qquad \textbf{if } \text{flag}(h) \textbf{ then } h \perp h_1,\ h_2 = h + h_1 \\
& \qquad \textbf{else } \text{flag}(h_2) \rightarrow (\text{flag}(h_1) \rightarrow h_1 \perp h_2) \wedge (\text{flag}(h) \rightarrow h \perp h_2)
\end{aligned}
$$

Extending the Model

- new operators satisfy Galois connections and dualities

$$
\begin{tikzcd}
P * Q \arrow[rr, leftrightarrow, bend left, "\text{Galois}"] \arrow[d, leftrightarrow, "\text{dual}"] & & P \mathrel{-\!\!*} Q \arrow[d, leftrightarrow, "\text{dual}"'] \\
P \mathrel{\leadsto\!\!*} Q \arrow[rr, leftrightarrow, bend right, "\text{Galois}"'] & & P \mathrel{-\!\!\circledast} Q
\end{tikzcd}
$$

- separation algebra is identical to the ‘old’ in case of **no failure**

- in case of failure, associativity of separating conjunction is lost
- non-intuitive when failure occurs
- does not carry enough information

Negative Heaps

- sets of pairs of heaps (before we had pair of sets of heaps)
- inspired by the construction of integers out of natural numbers

$$(2,5) \;\equiv\; -3 \;\equiv\; (0,3)$$

- operations

$$s, \mathtt{h} \models p \mathrel{-\!\!\circledast} q \;\Leftrightarrow\; \exists \mathtt{h}_1 : \mathtt{h}_1^\star \perp \mathtt{h}^\star \text{ and } s, \mathtt{h}_1 \models p \text{ and } s, \mathtt{h} \cup \mathtt{h}_1 \models q$$

where $\mathtt{h}$ is a pair of heaps and * heap reduction

$$(p \mapsto v, \mathtt{emp}) \mathrel{-\!\!\circledast} (p \mapsto v, \mathtt{emp}) \;=\; (\mathtt{emp}, \mathtt{emp})$$
$$(p \mapsto v, \mathtt{emp}) \mathrel{-\!\!\circledast} (\mathtt{emp}, \mathtt{emp}) \Rightarrow (\mathtt{emp}, p \mapsto v)$$

Negative Heaps II

- but you cannot subtract a heap twice

$$(p \mapsto v, \mathtt{emp}) \multimap\!\!\circledast\ (\mathtt{emp}, p \mapsto v) \ = \mathtt{false}$$

- can we have something like

$$(p \mapsto v, \mathtt{emp}) \multimap\!\!\circledast\ (\mathtt{emp}, p \mapsto v) \ = (\mathtt{emp}, [p \mapsto v, p \mapsto v])$$

Conclusion
The Good, the Bad, the Ugly

- framework for
  backwards reasoning using weakest preconditions and
  forward reasoning using strongest postconditions for partial
  (and generalised correctness)
- automation
- basic examples demonstrated

- adding failure to achieve generalised correctness seems to loose at least one crucial property

- generalised correctness is not nice; can we do better?

Thank you

Data61
Peter Höfner

t +61 2 9490 5861
e peter.hoefner@data61.csiro.au
w www.data61.csiro.au