

FMSoft

Lecture 14 — Wlp- and vc-friendly  
specifications in separation logic

(pre-lecture version)

---

Tadeusz Litak

January 29, 2019

Informatik 8, FAU Erlangen-Nürnberg

## Back to dynamic programs

- We have spent quite some time with weakest (liberal) preconditions
- But all of this has been done for the simple language IMP
- It is time to return to DIMP and separation logic
- We have not discussed semantics, so we will not try to do relative completeness  
and recall from the last lecture such results only go so far anyway
- But even on a purely syntactic level, do we have all the pieces for computing a DIMP analogue of *wlp*?

## Reminder of small axioms

- The small axiom for allocation:  
 $\vdash \{(X==i) \wedge \text{emp}\} \quad X := \text{CONS} \quad \bar{a} \quad \{X: -> \bar{a}[i/X]\}$
- The small axiom for lookup:  
 $\vdash \{(X==i) \wedge (a: \rightarrow i')\} \quad X := [a] \quad \{(X==i') \wedge (a[i/X]: \rightarrow i')\}$
- The small axiom for mutation:  
 $\vdash \{\exists i. a: \rightarrow \_ \} \quad [a] := a' \quad \{a: \rightarrow a'\}$
- The small axiom for deallocation:  
 $\vdash \{\exists i. a: \rightarrow \_ \} \quad \text{DISPOSE} \quad a \quad \{\text{emp}\}$
- Small axioms indeed!
  - Precondition for **CONS** and postcondition for **DISPOSE**: empty heaplet
  - Postcondition for **CONS**: heaplet the size of  $\bar{a}$
  - Everywhere else: heaplet size 1

## Reminder of global specifications (via the frame rule)

- $\vdash \{(a:\rightarrow \_)*B\}$  **DISPOSE**  $a \{B\}$
- $\vdash \{((X = i) \wedge (a:\rightarrow i'))*A[i'/X]\}$   $X:=[a] \{(a[i/X]:\rightarrow i')*A\}$
- $\vdash \{(X = i) \wedge A\}$   $X:=\mathbf{CONS} \bar{a} \{(X:\rightarrow \bar{a}[i/X])*A[i/X]\}$
- $\vdash \{(a:\rightarrow \_)*B\}$   $[a] := a' \{(a:\rightarrow a')*B\}$
- Only the one for deallocation is of the right form for computing **weakest (liberal) preconditions!**
- That is, it has an **arbitrary** assertion as a precondition
- If you recall though, we needed certain features of the boolean language to compute them

- We've had this bit in the definition of  $wlp$  for conditionals:

$$wlp(\text{IF } b \text{ THEN } c_1 \text{ ELSE } c_2, B) := \\ (b \wedge wlp(c_1, B)) \vee (\neg b \wedge wlp(c_2, B))$$

- We've also defined the sequence for while as follows

$$A_0^{b,B,c} := \top$$

$$A_{n+1}^{b,B,c} := (b \wedge wlp(c, A_n^{b,B,c})) \vee (\neg b \wedge B)$$

- Notice a pattern here?
- **Exercise:** Show that

$$(b \wedge A) \vee (\neg b \wedge C) \equiv (b \rightarrow A) \wedge (\neg b \rightarrow C)$$

- It is crucial that  $\rightarrow$  is a (definable) implication well-playing with standard connectives
- But now we have  $*$ , which is a different connective and a different conjunction than those we know from GLoIn

- One way of seeing this is that  $\wedge$  and  $\rightarrow$  are natural partners:

$A \wedge B$  provably implies  $C$  iff  $A$  provably implies  $B \rightarrow C$

Mathematicians speak of *adjoint pairs*

- Note that in particular we have  $\vDash (A \wedge (A \rightarrow B)) \rightarrow B$
- Is there a partner connective like this for  $*$ ?
- Yes, and we did mention it: it's the **magic wand** or **separating implication**  $\rightarrow*$

## The magic wand $\multimap$

- $\sigma, \mathfrak{h} \models^I B_1 \multimap B_2$  if for all  $\mathfrak{h}'$  s.t.
  - $\text{dom} \mathfrak{h} \cap \text{dom} \mathfrak{h}' = \emptyset$
  - $\sigma, \mathfrak{h}' \models^I B_1$

we have that  $\sigma, \mathfrak{h} \cup \mathfrak{h}' \models^I B_2$

- Note that  $\models (A \multimap (A \multimap B)) \rightarrow B$
- The class of **assertions** of (*infinitary*) *separation logic with  $\multimap$*  (dynamic Hoare logic) **SepAss** is defined as

$B_1, B_2 ::= \top \mid a_1 == a_2 \mid a_1 < a_2 \mid \neg B_1 \mid \bigwedge_{z \in \mathbb{Z}} B_z \mid$   
**emp**  $\mid a_1 :-> a_2 \mid B_1 * B_2 \mid B_1 \multimap B_2 .$

## Benefits of not waving the magic wand around too often?

- Not present in the formalism of (Implicit) Dynamic Frames and frameworks based on IDF, in particular VeriFast developed (well!) at KU Leuven, often used in our tutorials
- **Problematic for model checking**: quantification over **all possible extensions** of heap(let)
- A line of work on **adjunction elimination**  
Calcagno et al., *Adjunct elimination in Context Logic for trees*, Information and Computation 2010; Dawar et al., *Adjunct Elimination Through Games in Static Ambient Logic*, FSTTCS 2004; Lozes, *Adjuncts elimination in the static ambient logic*, EXPRESS 2003
- But used in framework based on **proof assistants!**
  - Iris and its derivatives (Iron, RustBelt)  
Aarhus U., MPI-SWS, TU Delft, KU Leuven ...
  - Verified Software Toolchain and its derivatives (CompCert)  
mostly Princeton, also INRIA, Galois.com ...



# Mutation

$$\vdash \{(a \rightarrow \_) * B\} [a] := a' \{(a \rightarrow a') * B\}$$

$$\vdash \{(a \rightarrow \_) * ((a \rightarrow a') * A)\} [a] := a' \{(a \rightarrow a') * ((a \rightarrow a') * A)\}$$
$$\vDash (\alpha * (\alpha * \beta)) \rightarrow \beta$$

---

$$\vdash \{(a \rightarrow \_) * ((a \rightarrow a') * A)\} [a] := a' \{A\}.$$

# Allocation: basic facts about quantifiers and $\rightarrow$

$$\vdash \{(X = i) \wedge A\} X := \text{CONS } \bar{a} \{(X := \bar{a}[i/X]) * A[i/X]\}$$

$$\begin{aligned} \vdash \{(X = i) \wedge \alpha\} X := \text{CONS } \bar{a} \{(X := \bar{a}[i/X]) * \alpha[i/X]\} \\ \models \beta \rightarrow \exists i'. (X = i') \wedge \beta[i'/X] \end{aligned}$$

$$\begin{aligned} \vdash \{(X = i) \wedge \alpha\} X := \text{CONS } \bar{a} \{\exists i'. (X = i') \wedge ((i' := \bar{a}[i/X]) * \alpha[i/X])\} \\ \models (\forall i'. \beta) \rightarrow \beta \end{aligned}$$

$$\vdash \{\forall i'. (X = i) \wedge \alpha\} X := \text{CONS } \bar{a} \{\exists i'. (X = i') \wedge ((i' := \bar{a}[i/X]) * \alpha[i/X])\}$$

SUBST

$$\begin{aligned} \vdash \{\forall i'. (X = i) \wedge ((i' := \bar{a}[i/X]) \rightarrow * A[i'/X])\} X := \text{CONS } \bar{a} \\ \{\exists i'. (X = i') \wedge ((i' := \bar{a}[i/X]) * ((i' := \bar{a}[i/X]) \rightarrow * A[i'/X]))\} \\ \models \forall i'. \alpha \rightarrow \forall i'. ((X = i) \wedge \alpha[i/X]) \quad \models (\gamma \wedge \alpha * (\alpha \rightarrow * \beta)) \rightarrow \gamma \wedge \beta \end{aligned}$$

$$\begin{aligned} \vdash \{\forall i'. (i' := \bar{a}) \rightarrow * A[i'/X]\} X := \text{CONS } \bar{a} \{\exists i'. (X = i') \wedge A[i'/X]\} \\ \models (\exists i'. (X = i') \wedge \alpha[i'/X]) \rightarrow \alpha \end{aligned}$$

$$\vdash \{\forall i'. (i' := \bar{a}) \rightarrow * A[i'/X]\} X := \text{CONS } \bar{a} \{A\}$$

For lookup we need something more:

- An observation regarding **strictly exact** assertions
- A missing rule regarding fresh variables in preconditions

## Warm-up for strict exactness

- Recall:  $\sigma, \mathfrak{h} \models^I B_1 * B_2$  if there exist  $\mathfrak{h}_1, \mathfrak{h}_2$  s.t.
  - $\text{dom} \mathfrak{h}_1 \cap \text{dom} \mathfrak{h}_2 = \emptyset$
  - $\mathfrak{h}_1 \cup \mathfrak{h}_2 = \mathfrak{h}$
  - $\sigma, \mathfrak{h}_1 \models^I B_1$  and  $\sigma, \mathfrak{h}_2 \models^I B_2$
- Recall:  $\sigma, \mathfrak{h} \models^I B_1 \rightarrow B_2$  if for all  $\mathfrak{h}'$  s.t.  $\text{dom} \mathfrak{h} \cap \text{dom} \mathfrak{h}' = \emptyset$  and  $\sigma, \mathfrak{h}' \models^I B_1$  we have that  $\sigma, \mathfrak{h} \cup \mathfrak{h}' \models^I B_2$

### Exercise:

- Show:  $\models (A * (A \rightarrow B)) \rightarrow B$
- Deduce:  $\models (A * (A \rightarrow B)) \rightarrow (A * \text{true}) \wedge B$
- Disprove:  $\models (A * \text{true}) \wedge B \rightarrow (A * (A \rightarrow B))$
- (Hint:  $A := \neg \text{emp}, B := 0 \rightarrow 0$ )
- While we're at it, disprove:  $\models (A \wedge (A \rightarrow B)) \rightarrow B$
- (Hint:  $A := 0 \rightarrow 0, B := \text{false}$ )

- For some subclasses of SL expressions, we have more laws governing  $\rightarrow^*$
- In particular, this applies to **strictly exact assertions**, isolated by Hongseok Yang
- $A$  is *strictly exact* iff for any  $\sigma$ ,  $\mathfrak{h}$  and  $\mathfrak{h}'$  we have that  $\mathfrak{h} = \mathfrak{h}'$  whenever  $\sigma, \mathfrak{h} \models^I A$  and  $\sigma, \mathfrak{h}' \models^I A$

### Fact

*Assertions built from  $\rightarrow$ -atoms using  $*$  (i.e., descriptions of concrete finite heap(let)s!) are strictly exact.*

- The importance of this concept for us:

### Fact

*Whenever  $A$  is strictly exact and  $B$  is any assertion,*

$$\models (A * \mathbf{true}) \wedge B \rightarrow (A * (A \rightarrow^* B))$$

- **Exercise:** prove it!
- **Exercise:**  $\neg \mathbf{emp}$  obviously fails to be strictly exact.

## Fresh variables in preconditions

- Logical and arithmetical reasoning is entirely delegated to the consequence rule
- There is however a rule corresponding to a logical rule for quantifiers which needs to be stated separately:

$$\frac{\vdash \{A\} c \{B\} \quad i \in \text{fresh}(B)}{\vdash \{A[a/i]\} c \{B\}} \text{VPREL}$$

- **Exercise:** derive from this

$$\frac{\vdash \{X=i \wedge A\} c \{B\} \quad i \in \text{fresh}(B) \cap \text{fresh}(A)}{\vdash \{A\} c \{B\}} \text{VPREL}'$$

- If  $\exists$  primitive, take still stronger (why?)

$$\frac{\vdash \{A\} c \{B\} \quad i \in \text{fresh}(B)}{\vdash \{\exists i.A\} c \{B\}} \text{VPREL}\exists$$

- Aside: is  $\text{VPREL}\exists$  derivable from  $\text{VPREL}$ ?
- Recall we encoded  $\exists i.A$  as  $\bigvee_{z \in \mathbb{Z}} A[\mathbf{z}/i]$
- For this, we need to directly postulate

$$\frac{\vdash \{A_i\} c\{B\} \quad \text{for all } i \in \mathbb{Z}}{\vdash \{\bigvee_{i \in \mathbb{Z}} A_i\} c\{B\}}$$

- O'Hearn, Yang and Reynolds: if the disjunction rule not postulated explicitly, even its finitary variant may happen to be admissible without being derivable

## Dually, in postconditions ...

$$\frac{\vdash \{A\} c \{B\} \quad i \in \text{fresh}(A)}{\vdash \{A\} c \{B[a/i]\}} \text{VPOEL}$$

$$\frac{\vdash \{A\} c \{X=i \rightarrow B\} \quad i \in \text{fresh}(A) \cap \text{fresh}(B)}{\vdash \{A\} c \{B\}} \text{VPOEL}'$$

$$\frac{\vdash \{A\} c \{B\} \quad i \in \text{fresh}(A)}{\vdash \{A\} c \{\forall i. B\}} \text{VPOEL}\forall$$

$$\frac{\vdash \{A\} c \{B_i\} \quad \text{for all } i \in \mathbb{Z}}{\vdash \{A\} c \{\bigwedge_{i \in \mathbb{Z}} B_i\}}$$



# Lookup

$$\vdash \{((X=i) \wedge (a:\rightarrow i')) * A'[i'/X]\} X := [a] \{(a[i/X]:\rightarrow i') * A'\}$$

Pick  $i, i'$  fresh for  $A$ :

$$\begin{array}{l} \vdash \{((X=i) \wedge (a:\rightarrow i')) * ((a[i/X]:\rightarrow i') * A[i'/X])\} \\ X := [a] \{(a[i/X]:\rightarrow i') * ((a[i/X]:\rightarrow i') * A)\} \\ \models (X=i) \wedge (\beta * \alpha) \rightarrow ((X=i) \wedge \beta) * \alpha[i/X] \quad \models (\alpha' * (\alpha' * \beta')) \rightarrow \beta' \end{array}$$

$$\text{VPREL}' \frac{\vdash \{(X=i) \wedge ((a:\rightarrow i') * ((a:\rightarrow i') * A[i'/X]))\} X := [a] \{A\}}{\vdash \{(a:\rightarrow i') * ((a:\rightarrow i') * A[i'/X])\} X := [a] \{A\}}$$

$$(*) \frac{\vdash \{(a:\rightarrow i') * ((a:\rightarrow i') * A[i'/X])\} X := [a] \{A\}}{\models (((a:\rightarrow i') * \text{true}) \wedge A[i'/X]) \rightarrow (a:\rightarrow i') * ((a:\rightarrow i') * A[i'/X])}$$

$$\text{VPREL}\exists \frac{\vdash \{((a:\rightarrow i') * \text{true}) \wedge A[i'/X]\} X := [a] \{A\}}{\vdash \{\exists i'. ((a:\rightarrow i') * \text{true}) \wedge A[i'/X]\} X := [a] \{A\}}$$

The (\*) step because of strict exactness!

- Let us take stock ...
- Allocation:

$$\vdash \{ \forall i'. (i' \rightarrow \bar{a}) \ast A[i'/X] \} X := \text{CONS } \bar{a} \{ A \}$$

- Lookup:

$$\vdash \{ \exists i'. ((a \rightarrow i') \ast \text{true}) \wedge A[i'/X] \} X := [a] \{ A \}$$

( $i'$  fresh for  $A$ )

- Mutation:

$$\vdash \{ (a \rightarrow \_) \ast ((a \rightarrow a') \ast A) \} [a] := a' \{ A \}$$

- Deallocation:

$$\vdash \{ (a \rightarrow \_) \ast A \} \text{DISPOSE } a \{ A \}$$

- Just like in the case of ordinary IMP, we can compute weakest (liberal) preconditions