

# From Separation Logic to First-Order Logic

Cristiano Calcagno   Philippa Gardner  
Matthew Hague

April 6, 2005

# Abstract

- A new decision procedure for Propositional Separation Logic.
- Translation into a decidable fragment of First-Order Logic.
- FOL independent of heap structure.
- Symbolic approach.
- May find larger decidable fragments of Separation Logic.
- May extend to Spatial Logics for tree structures and data update.

# Reasoning About Data Structures

## Spatial Logics

- Elegant method for reasoning locally about data structures.
- Provide a basis for Hoare Logics for data update.

## Trees and heaps

- Heaps — Separation Logic. O'Hearn, Reynolds, Yang.
- Trees — Static Ambient Logic. Cardelli, Gordon.
- Both theories of Bunched Logic. O'Hearn, Pym.

# Decidability

## Separation Logic — Calcagno *et al*

- The full Separation Logic is undecidable.
- Propositional Separation Logic is PSPACE-complete. (No quantifiers.)
- Bounded model property.

## Static Ambient Logic

- Calcagno *et al* extended their results.
- Dal Zilio *et al* — Presburger Arithmetic and automata.

## Back to Separation Logic

- Translation to a decidable First-Order Logic independent of heaps.
- Symbolic representation.

# Heaps and Stacks

Heap  $h$ : partial map from locations to values.

1	2	3	4	5	6	7	8	9	10
0		7	3				3		

$$(3 \mapsto 7) * (4 \mapsto 3) * (8 \mapsto 3) * (1 \mapsto \text{nil})$$

Stack  $s$ : assignment of values to program variables.

# Propositional Separation Logic

Assertions on heaps and stacks,

$$s, h \models \phi$$

Where  $\phi ::=$

$$\begin{array}{l|l}
 (E \mapsto E') & \phi_1 \Rightarrow \phi_2 \\
 \text{emp} & \text{false} \\
 \phi_1 * \phi_2 & E = E' \\
 \phi_1 \multimap \phi_2 &
 \end{array}$$

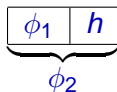
and  $E, E' ::= \text{nil}, x, y, z, \dots$

# Composition and Composition Adjunct

$s, h \models \phi_1 * \phi_2$  iff the heap can be split (disjoint),



$s, h \models \phi_1 \multimap \phi_2$  iff all heaps satisfying  $\phi_1$  that compose with  $h$ ,



# Examples

Location  $s(x)$  maps to  $s(y)$

$$(x \mapsto y) * \text{true}$$

Suppose  $s(x) = 4$  and  $s(y) = 3$ ,

1	2	3	4	5	6	7	8	9	10
0		7	3				3		

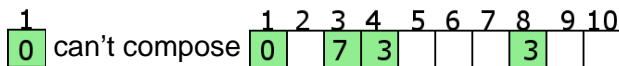
$$\begin{aligned}
 & s, (4 \mapsto 3) \quad \models (x \mapsto y) \\
 s, (3 \mapsto 7) * (8 \mapsto 3) * (1 \mapsto \text{nil}) & \quad \models \text{true}
 \end{aligned}$$

# Examples

Location  $s(x)$  is allocated,

$$(x \mapsto \text{nil}) \rightarrow^* \text{false}$$

Suppose  $s(x) = 1$ ,



**false** is satisfied vacuously.

# Bounded Model Property

Evaluation of  $\text{—*}$  requires universal quantification.

Quantification can be bounded.

- Each formula  $\phi$  has a size  $|\phi|$ .
- Intuitively: the number of cells it can recognise.
- We only consider heaps with at most  $|\phi|$  cells.

If a given heap is too big, we can “trim it”.

# Heaps as Vectors

Given a size  $p$ , we can represent a heap as a vector,

$$h \Rightarrow_p ((b_{1,1}, b_{1,2}), \dots, (b_{p,1}, b_{p,2}))$$

If  $p = 6$ , then

1	2	3	4	5	6	7	8	9	10
0		7	3				3		

$$\Rightarrow_p ((1, 0), (4, 3), (8, 3), (0, -), (0, -), (3, 7))$$

We can quantify over heaps by quantifying over the variables,

$$\mathbf{B}_p = b_{1,1}, b_{1,2}, \dots, b_{p,1}, b_{p,2}$$

# The Translation

$$tr(\phi, \mathbf{B}_p) \triangleq heap(\mathbf{B}_p) \Rightarrow tr'(\phi, \mathbf{B}_p)$$

$$tr'(E_1 = E_2, \mathbf{B}_p) \triangleq E_1 = E_2$$

$$tr'(\text{false}, \mathbf{B}_p) \triangleq \text{false}$$

$$tr'(\phi_1 \Rightarrow \phi_2, \mathbf{B}_p) \triangleq tr'(\phi_1, \mathbf{B}_p) \Rightarrow tr'(\phi_2, \mathbf{B}_p)$$

$$tr'(\text{emp}, \mathbf{B}_p) \triangleq \bigwedge_i b_{i,1} = 0$$

$$tr'(E \mapsto E', \mathbf{B}_p) \triangleq \bigvee_i \left( \begin{array}{l} b_{i,1} \neq 0 \wedge \bigwedge_{j \neq i} b_{j,1} = 0 \\ \wedge b_{i,1} = E \\ \wedge b_{i,2} = E' \end{array} \right)$$

# The Translation

$$tr'(\phi_1 * \phi_2, \mathbf{B}_p) \triangleq \exists \mathbf{B}'_p, \mathbf{B}''_p. \left( \begin{array}{l} \mathbf{B}_p = \mathbf{B}'_p \circledast \mathbf{B}''_p \\ \wedge tr'(\phi_1, \mathbf{B}'_p) \\ \wedge tr'(\phi_2, \mathbf{B}''_p) \end{array} \right)$$

Where  $\circledast$  “divides” a vector into two disjoint heaps: 

$\phi_1$	$\phi_2$
----------	----------

$$tr'(\phi_1 \multimap \phi_2, \mathbf{B}_p) \triangleq \forall \mathbf{B}'_q. \left( \begin{array}{l} tr'(\phi_1, \mathbf{B}'_q) \\ \wedge heap(\mathbf{B}_p \bullet \mathbf{B}'_q) \\ \Rightarrow tr'(\phi_2, \mathbf{B}_p \bullet \mathbf{B}'_q) \end{array} \right)$$

Where  $\bullet$  is the appending of vectors: 

$\phi_1$	$h$
----------	-----

  
 $\phi_2$

# Example

The translation of  $(x \mapsto y) * \text{true}$  is,

$$\exists \mathbf{B}'_p, \mathbf{B}''_p. \left( \begin{array}{l} \mathbf{B}_p = \mathbf{B}'_p \circledast \mathbf{B}''_p \\ \wedge \left( \bigvee_i \left( \begin{array}{l} b'_{i,1} \neq 0 \wedge \bigwedge_{j \neq i} [b'_{j,1} = 0] \\ \wedge b'_{i,1} = x \\ \wedge b'_{i,2} = y \end{array} \right) \right) \\ \wedge \text{true} \end{array} \right)$$

# Translation Properties

Satisfaction: let  $h \Rightarrow \mathbf{b}_p$  and  $\mathbf{X} = FV(\phi)$

$$(s, h \models \phi) \iff [\mathbf{B}_p \leftarrow \mathbf{b}_p, \mathbf{X} \leftarrow vs(s)] \models tr'(\phi, \mathbf{B}_p)$$

Validity: let  $\mathbf{X} = FV(\phi)$ ,

$$(\forall (s, h). (s, h) \models \phi) \iff \models \forall \mathbf{B}_p, \mathbf{X}. tr(\phi, \mathbf{B}_p)$$

# Extensions and Related Work

Implementation and optimisation.

Decision procedures for larger fragments,

- We do not exploit the logic fully, or may use richer logics.
- Maintain bounded model property.
- Guarded quantification? Stack arithmetic?

Related fragments

- Separation Logic for lists — Berdine, Calcagno, O'Hearn.
  - No magic wand.
  - $Is(E, E')$  —  $E'$  reachable from  $E$ .
  - Has a bounded model property.

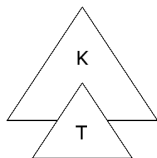
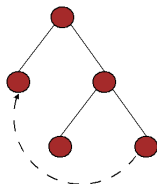
## Further Work – Trees

Trees with pointers,

- Unique node locations, cross-references.
- Combine with Dal Zilio *et al.*

Context Logic

- Contexts:  $K[T]$
- “somewhere” modalities:  $(\text{true})[T]$
- Tree update.
- Bounded model property?



# From Separation Logic to First-Order Logic

Cristiano Calcagno   Philippa Gardner  
Matthew Hague

April 6, 2005