Refinement in a Separation Context

Ivana Mijajlović
Queen Mary,
University of London

Noah Torp-Smith
IT University of Copenhagen

SPACE 2004
Introduction

• Hoare did data refinement for imperative programs
Introduction

• Hoare did data refinement for imperative programs

• Pointers + Data abstraction = Trouble
Introduction

- Hoare did data refinement for imperative programs

- Pointers + Data abstraction = Trouble

- As usual dangling pointers are the problem
Introduction

• Hoare did data refinement for imperative programs

• Pointers + Data abstraction = Trouble

• As usual dangling pointers are the problem

• Linguistic approaches haven't worked
Modeling Clients and Modules

A relation \( M \subseteq S \times H \) is **precise** if for any state \( s, h \) there is at most one subheap \( h_0 \sqsubseteq h \), such that \( (s, h_0) \in M \).
Modeling Clients and Modules

A relation $M \subseteq S \times H$ is **precise** if for any state $s, h$ there is at most one subheap $h_0 \sqsubseteq h$, such that $(s, h_0) \in M$.

The **separating conjunction of unary relations** $M, M' \subseteq S \times H$

$M \ast M' = \{(s, h) | \exists h_0, h_1. h_0 \# h_1 \land h = h_0 \ast h_1 \land (s, h_0) \in M \land (s, h_1) \in M'\}$. 
Modeling Clients and Modules

A relation $M \subseteq S \times H$ is **precise** if for any state $s, h$ there is at most one subheap $h_0 \sqsubseteq h$, such that $(s, h_0) \in M$.

The **separating conjunction of unary relations** $M, M' \subseteq S \times H$

$M \ast M' = \{(s, h) \mid \exists h_0, h_1. h_0 \# h_1 \land h = h_0 \ast h_1 \land (s, h_0) \in M \land (s, h_1) \in M'\}$.

Let $t \subseteq (S \times H) \times (S \times H) \uplus \{\text{wrong}\}$. The relation $M \subseteq S \times H$ is **preserved** by relation $t$ if for all $(s, h), (s', h')$, $(s, h) \in M$ and $(s, h)[t](s', h')$, imply $(s', h') \in M$. 
Separation Context

\[ c_{user} ::= \text{oper}_i, \ i \in I \mid \text{skip} \mid x := e \mid x := [e] \mid [e] := e \mid c_1; c_2 \]
\[ \mid \text{if } e \text{ then } c_1 \text{ else } c_2 \mid \text{while } e \text{ do } c \]
Separation Context

\[
c_{\text{user}} ::= \text{oper}_i, \ i \in I \mid \text{skip} \mid x := e \mid x := [e] \mid [e] := e \mid c_1; c_2 \\
| \text{if} \ e \ \text{then} \ c_1 \ \text{else} \ c_2 \mid \text{while} \ e \ \text{do} \ c
\]

Let \( M \subseteq S \times H \) be a precise unary relation, and for \( i \in I \) let \( \text{oper}_i \) preserve relation \( M \ast T \). A program \( c \) is a \textbf{unary separation context} for \( M \) and \((\text{oper}_i)_{i \in I}\) if for all executions and all \( (s, h) \in M \ast T c, s, h \not\leftrightarrow \text{av} \) and \( c, s, h \not\leftrightarrow \text{wrong} \).
Separation Context

\[ c_{\text{user}} ::= \text{oper}_i, \ i \in I \mid \text{skip} \mid x := e \mid x := [e] \mid [e] := e \mid c_1; c_2 \]
\[ \mid \text{if } e \text{ then } c_1 \text{ else } c_2 \mid \text{while } e \text{ do } c \]

Let \( M \subseteq S \times H \) be a precise unary relation, and for \( i \in I \) let \( \text{oper}_i \) preserve relation \( M \ast T \). A program \( c \) is a \textbf{unary separation context} for \( M \) and \((\text{oper}_i)_{i \in I} \) if for all executions and all \((s, h) \in M \ast T \) \( c, s, h \not\rightarrow \text{av} \) and \( c, s, h \not\rightarrow \text{wrong} \).

Let \( M \subseteq S \times H \) be a precise relation, and for \((i \in I)\) let \( \text{oper}_i \) preserve \( M \ast T \), and let \( c \) be a separation context for \( M \) and \((\text{oper}_i)_{i \in I} \). If \((s, h) \in M \ast T \), and \( c, s, h \leadsto s', h' \), then \((s', h') \in M \ast T \).
Separation context

dispose(x);
Separation context

dispose(x); x=new();
Separation context

dispose(x); x=new(); y=[x]
Non-separation context

ls

x=new()

47

x

47

x=new()
Non-separation context

ls
47
dispose(x);
Binary Relations for Refinement

We say that binary relation $R$ is **precise**, if each of its two projections on the set of states is precise.
We say that binary relation \( R \) is **precise**, if each of its two projections on the set of states is precise.

Separating conjunction of binary relations
Refinement

\[ \begin{align*} 
(s_A, h_A) & \xrightarrow{\text{oper}_1} \exists (s'_A, h'_A) \quad \text{oper}_2 \\
R \ast \text{Id} & \\
(s_C, h_C) & \xrightarrow{\text{oper}_2} (s'_C, h'_C) \\
R \ast \text{Id} & \quad \text{oper}_2 \\
(s_A, h_A) & \xrightarrow{\text{oper}_1} \text{wrong} \\
R \ast \text{Id} & \\
(s_C, h_C) & \xrightarrow{\text{oper}_2} \text{wrong} 
\end{align*} \]
The Result

• A separation context for the abstract data type is a separation context for all its refinements
The Result

- A separation context for the abstract data type is a separation context for all its refinements

- Separation contexts preserve $R \ast \text{Id}$
Example - *new()* and *dispose()*

Abstract − Magic
RI: *emp*

Intermediate − Set
RI: $\forall p \in f. p \mapsto \_ \_ \_ $

Concrete − List
RI: *list*(\(\alpha, ls.nil\))
Example - new() and dispose()

$R_1 = \{ ((s_A, h_A), (s_C, h_C)) \mid s_A, h_A \models emp \land (s_C, h_C \models \forall p \in f. p \mapsto \_, \_)) \}$
Future Work

• This is only a model
Future Work

• This is only a model

• We would like to have a logic