CS 839 Systems Verification

Lecture 7: Separation Logic (part 1)

this lecture will be mostly on the board

Learning outcomes

1. Appreciate why reasoning about pointers is hard
2. Reason about separation logic predicates

Recap

Hoare logic: pre and post conditions, soundness

Pointers are hard

```
type list struct {
  X    int
  Next *list
}

x := &list{X: 2, Next: nil}
y := &list{X: 3, Next: x}
z := &list{X: 10, Next: &list{X: 2, Next:
nil}}
// x = [2], y = [3, 2], z = [10, 2]
```

```
func f(l *list) { ... }

x := ... // setup from earlier
f(x)
// what is true here?
```

What might be true now?

3 MIN think about it

Example of framing: calling function could modify any reachable pointer

Separation logic: an overview

semantics: add a heap

predicates: language to describe the heap

logic: new ways of reasoning

Syntax/semantics: add pointers and heap allocation

Predicates: need to describe heap as well as variables

Logic: new ways of reasoning, slightly new soundness theorem

Syntax and semantics

syntax: $\mathrm{alloc}\, e$, $!\ell$ (load), $\ell \leftarrow v$ (store)

semantics needs heap: `loc -> option val`

$(e, h)\ \ > (e', h')$

Propositional logic

syntax:

$P ::= P \wedge Q \mid P \vee Q \mid \neg P \mid P \to Q \mid \exists x.\, P(x) \mid$
$\forall x.\, P(x) \mid x = y$

entailment: $P \vdash Q$ (*not* a proposition)

Proofs in propositional logic

$$P \land Q \vdash P \quad P \land Q \vdash Q \quad \frac{P \vdash Q \quad P \vdash R}{P \vdash Q \land R}$$

$$\frac{\forall x.\, (P \vdash Q(x))}{P \vdash \forall x.\, Q(x)}\ \text{all-intro}$$

$$\frac{\forall x.\, (P(x) \vdash Q)}{\exists x.\, P(x) \vdash Q}\ \text{exists-elim}$$

Let's be precise about how to prove properties - easy enough right now, but more complicated when we get to separation logic

Example: prove AND commutes, $P \land Q \vdash Q \land P$

5-min break

Heap predicates

Need a language of *heap predicates*

hProp := heap → Prop

Predict: soundness theorem

Recall semantics now looks like $(e, h) \; > (e', h')$

$P : \mathrm{heap} \to \mathrm{Prop}$

Write down soundness definition for $\{P\}\, e\, \{\lambda v.\, Q(v)\}$

Answer: soundness of separation logic

$\{P\}\, e\, \{\lambda v.\, Q(v)\}$

If $P(h)$ holds and $(e, h) \;\; > (e', h')$ then

- (1) (e', h') is not stuck
- (2) $e' = v'$ and $Q(v')(h')$ holds

Separation logic propositions

$$P ::= \ell \mapsto v \mid P * Q \mid \mathrm{emp}$$

$$\mid \varphi \mid P \vee Q \mid \forall x.\, P(x) \mid \exists x.\, P(x)$$

(where φ is a normal proposition)

$$\ell \mapsto v$$

"ℓ **points to** v"

True for exactly one heap: the one where l maps to v and nothing else is allocated.

(definition: use dom for heap domain)

$$P * Q$$

"P and separately Q"

The key to separation logic is the separating conjunction.

(definition: use $\perp$ for disjoint)

Derived rules

where $P \vdash Q$ means $\forall h.\, P(h) \to Q(h)$

$$
\begin{aligned}
P \star Q &\vdash Q \star P && \text{sep-comm} \\
P \star (Q \star R) &\vdash (P \star Q) \star R && \text{sep-assoc} \\
(\exists x.\, P(x)) \star Q &\vdash (\exists x.\, P(x) \star Q) && \text{sep-exists} \\
\ell \mapsto v \star \ell \mapsto w &\vdash \text{False} && \text{pointsto-sep} \\
P &\vdash P \star \text{emp} && \text{sep-id}
\end{aligned}
$$

Exercise: prove sep-monotone-left

We have a "right" version, what about the "left"?

$$\frac{Q \vdash Q'}{P \star Q \vdash P \star Q'}\ \text{sep-monotone}$$

Exercise: draw some heaps

(assume ℓ_1, ℓ_2 are distinct)

1. $\ell_1 \mapsto \ell_2 * \ell_2 \mapsto \ell_1$
2. $\ell_2 \mapsto \mathbf{3} * \ell_1 \mapsto \ell_2 * \ell_3 \mapsto \ell_2$
3. $\ell_3 \mapsto \ell_4 * \ell_2 \mapsto \ell_2 * \ell_1 \mapsto \mathbf{3}$