Semantics of mutable state

Koen Wermer Robert Hensing

June 25, 2015

What

- A model language with mutable state
- Hoare type system
- Small step semantics
- Type safety proof
- Normalization proof
- Some neat Hoare logic theorems

Basic semantics for mutable state (1)

```
data Type : Set where
  ...
  <>      : Type
  POINTER : Type -> Type
```

Basic semantics for mutable state (2)

New terms

- create cell with value
- update cell with value
- read cell value
- redirect pointer to other cell
- ;
- the actual cell

Basic semantics for mutable state (3)

```
data State : TypeEnv -> Set where
  state : (f : TypeEnv) -> ((p : Pointer)
            -> (Term (f p))) -> State f

data Step  : {ty : Type} {f : TypeEnv}
               -> State f -> Term ty
               -> State f -> Term ty -> Set where
     ...
```

Type system

Hoare logic

$$\{P\}\ c\ \{Q\}$$

Type system

Hoare logic

$$\{P\}\ c\ \{Q\}$$

$$\frac{\{P\}\ c_1\ \{Q\},\quad \{Q\}\ c_2\ \{R\}}{\{P\}\ c_1\ ;\ c_2\ \{R\}}$$

Type system

'Hoare Type Theory'

$$\{P\}\ c : t\ \{Q\}$$

$$\frac{\{P\}\ c_1 : t_1\ \{Q\},\ \ \{Q\}\ c_2 : t_2\ \{R\}}{\{P\}\ (c_1; c_2)\ :\ t_2\ \{R\}}$$

Type system

Separation
"Modular reasoning"

$$\frac{\{P\}\ c\ \{Q\}, \quad \mathit{modifiedBy}(c) \cap \mathit{freeVars}(R) = \emptyset}{\{P * R\}\ c\ \{Q * R\}}$$

Limitations

- Our work: No λ
- 'Fundamental': No programs in heap (or non-termination)
- Practical: Proving stuff is hard :)

Questions?

Literature

- Swierstra, Wouter. A functional specification of effects. Diss. University of Nottingham, 2009.
- Pierce, Benjamin C. Types and programming languages. MIT press, 2002.
- Nanevski, Aleksandar, Greg Morrisett, and Lars Birkedal. Hoare type theory, polymorphism and separation. Journal of Functional Programming 18.5-6 (2008): 865-911.
- John C. Reynolds. Separation Logic: A Logic for Shared Mutable Data Structures. LICS 2002.