The Type Base and Undecidability in Type Theory
PRL Seminar

Abhishek Anand

September 21, 2012
The Type Base

- members: all terms of the underlying computation system
- equality : squiggle equality (\(\sim\))\(^1\)
- Informally: 2 squiggle equal terms can be substituted for each other in any context. Some of it’s key properties are:
  - \(t \mapsto t' \Rightarrow t \sim t'\)
  - \((t \sim t' \land t \in T) \Rightarrow t = t' \text{ in } T\)
  - Any 2 diverging terms are squiggle equal

\(^1\)Howe89.
Possibly Simplest Undecidability proof

- Can there be \( h : Base \rightarrow bool \) such that \( h(t) \) is true iff \( t \) converges?
  - \( h \) is total, computable.
- \( \bot \overset{\text{def}}{=} fix(\lambda x.x) \)
- \( d \overset{\text{def}}{=} \lambda y. \text{if } h(y) \text{ then } \bot \text{ else } \lambda x.x \)
- What is \( h(fix(d)) \)?
- \( h(fix(d)) \mapsto h(f(fix(d))) \mapsto h(\text{if } h(fix(d)) \text{ then } \bot \text{ else } \lambda x.x) \)
- So, we have \( h(fix(d)) \sim h(\text{if } h(fix(d)) \text{ then } \bot \text{ else } \lambda x.x) \)
- \( h(fix(d)) \) is true iff it is false.
Domain Theoretic Argument

- \( d \overset{\text{def}}{=} \lambda y. \text{if } h(y) \text{ then } \bot \text{ else } \lambda x.x \)
- \( d \) is not monotonic
  - \( \bot \sqsubseteq \lambda x.x \)
  - \( d(\bot) \nsubseteq d(\lambda x.x) \)
- \( \lambda x.x \nsubseteq \bot \)
- Monotonocity is implied by congruence of squiggle ordering
Some (Unconventional) Properties of Base

- \( A \subseteq B \overset{\text{def}}{=} (x = y \text{ in } A) \implies (x = y \text{ in } B) \)
- \( Base \rightarrow Base \subseteq Base \)
- \( Base \rightarrow \text{bool} \subseteq Base \)
Bar Types

- given a type $T$ which does not equate a converging and diverging term, we can form a type $\bar{T}$
  - members: all terms which are members of $T$ if they converge
  - equality: $t = t'$ in $\bar{T}$
    * $\text{has-value}(t) \Leftrightarrow \text{has-value}(t') \land$
    * $\text{has-value}(t) \Rightarrow t = t'$ in $T$

- A type $T$ is admissible\(^2\) if:
  - $f \in \bar{T} \rightarrow \bar{T} \Rightarrow \text{fix}(f) \in \bar{T}$

- the type $Z$ is admissible\(^3\)
- $\bar{Z} \subseteq \text{Base}$

---

\(^2\)Crary98.

\(^3\)Crary98.
Can there be $h : \bar{\mathbb{Z}} \to bool$ such that $h(t)$ is true iff $t$ converges?

$\bot \overset{\text{def}}{=} \text{fix}(\lambda x. x)$

$d \overset{\text{def}}{=} \lambda y. \text{if } h(y) \text{ then } \bot \text{ else } 0$

$d \in \bar{\mathbb{Z}} \to \bar{\mathbb{Z}} \Rightarrow \text{fix}(f) \in \bar{\mathbb{Z}}$

What is $h(\text{fix}(f))$?

$h(\text{fix}(f)) \mapsto h(f(\text{fix}(f))) \mapsto h(\text{if } h(\text{fix}(f)) \text{ then } \text{fix}(\lambda x. x) \text{ else } 0)$

So, we have $h(\text{fix}(f)) \sim h(\text{if } h(\text{fix}(f)) \text{ then } \text{fix}(\lambda x. x) \text{ else } 0)$

$h(\text{fix}(f))$ is true iff it is false.
Corollary: $\neg \forall P : \text{prop.} (P \lor \neg P)$

- has-value(t) is a well-formed proposition for $t: \mathbb{Z}$
- by assuming $\forall P : \text{prop.} (P \lor \neg P)$, we get a decider for halting (has-value) on terms of $\mathbb{Z}$

```
* top 1 1 1
  [-]
  1. magic : $\forall P : \text{P.} \ (P \lor (\neg P))@i'`
  [-]
  |- n : \text{bar(\mathbb{Z})} \rightarrow \{ b : \text{B} | \uparrow b \iff \text{has-value}(n) \}

, * BY UseWitness \{\lambda n. \text{isl(magic has-value(n))}\}'.
```

- Contradiction!

Abhishek Anand
The Type Base and Undecidability in Type Theory
September 21, 2012 8 / 9
How to restore harmony?

- Cannot have both
  - Types which have diverging elements and do not equate a diverging and a converging element. e.g. Base, Bar Types
  - $\forall P : \text{prop.} (P \lor \lnot P)$

- What to give up?
- We could assume
  $\forall P : \text{prop.} \{ (P \lor \lnot P) \}$
  - $\{\circ\}$ erases the constructive content (decider)