Partiality and Dependent Types

Implementing a specification logic in a DTT

Kasper Svendsen, Lars Birkedal and Aleksandar Nanevski

June 2, 2011

Introduction

Dependent Type Theory

- higher-order functional programming language
- with integrated reasoning & verification

Introduction

Dependent Type Theory

- higher-order functional programming language
- with integrated reasoning & verification
- lacks general recursion
- lacks effects such as state, IO, etc.

Introduction

Dependent Type Theory

- higher-order functional programming language
- with integrated reasoning & verification
- lacks general recursion
- lacks effects such as state, IO, etc.

This talk: cheap way of adding general recursion

Partiality

Partiality

- unrestricted recursion breaks propositions-as-types

Partiality

Partiality

- unrestricted recursion breaks propositions-as-types

Our setting

- treat partiality as an effect
- use monads to add encapsulate effects in a pure language

$$\begin{aligned} O &: \mathbf{Set} \to \mathbf{Set} \\ \mathbf{fix}_\tau &: (O(\tau) \to O(\tau)) \to O(\tau) \qquad + \mathit{ret}, \mathit{bind} \end{aligned}$$

Admissibility: The problem

Problem: $\mathbf{fix}_\tau$ unsound in sufficient expressive TTs

- the type of $\mathbf{fix}_\tau$

$$\mathbf{fix}_\tau : (O(\tau) \to O(\tau)) \to O(\tau)$$

corresponds to fixpoint induction

$$\forall f : X \to X.\ \forall P \subseteq_{adm} X.\ (\forall x \in P.\ f(x) \in P) \Rightarrow \mathbf{fix}(f) \in P$$

Admissibility: The problem

Problem: $\mathbf{fix}_\tau$ unsound in sufficient expressive TTs

- the type of $\mathbf{fix}_\tau$

$$\mathbf{fix}_\tau : (O(\tau) \to O(\tau)) \to O(\tau)$$

corresponds to fixpoint induction

$$\forall f : X \to X.\ \forall P \subseteq_{adm} X.\ (\forall x \in P.\ f(x) \in P) \Rightarrow \mathbf{fix}(f) \in P$$

- in STT all partial types are admissible
- but in DTT there exists inadmissible types, e.g.,

$$O(\{c : \mathbb{N} \to O(\mathbb{N}) \mid \exists n \in \mathbb{N}.\ c(n) = \Omega_{\mathbb{N}}\})$$

where $\Omega_{\mathbb{N}} = \mathbf{fix}_{\mathbb{N}}(id_{O(\mathbb{N})})$

Admissibility: Previous work

Crary: introduce explicit admissibility proofs on **fix**

- very expressive & allows for easy implementation
- significant proof obligation for every use of **fix**

$$\mathbf{fix}_\tau : adm(O(\tau)) \to (O(\tau) \to O(\tau)) \to O(\tau)$$

- complicated admissibility theory for subset- & Σ-types

Admissibility: Previous work

Crary: introduce explicit admissibility proofs on **fix**

- very expressive & allows for easy implementation
- significant proof obligation for every use of **fix**

$$\mathbf{fix}_\tau : adm(O(\tau)) \to (O(\tau) \to O(\tau)) \to O(\tau)$$

- complicated admissibility theory for subset- & Σ-types

HTT: restrict to admissible types

- omits subset-types, strong Σ-types, inductive families

Admissibility: This talk

Idea

Only allow reasoning about effectful computations through specs (as in a program logic for an imperative language.)

Admissibility: This talk

Idea

Only allow reasoning about effectful computations through specs (as in a program logic for an imperative language.)

How?

- collapse equality on effectful computations

$$\text{if } M, N : O(\tau) \text{ then } M =_{O(\tau)} N$$

- types as only specification

Admissibility: This talk

Collapsed equality

- usual type constructors closed under admissible types

$$\Sigma, \Pi, \{x : \tau \mid P\}, W, O$$

- $\{x : O(\tau) \mid P(x)\}$ trivially admissible, as P is constant

Admissibility: This talk

Collapsed equality

- usual type constructors closed under admissible types

$$\Sigma, \Pi, \{x : \tau \mid P\}, W, O$$

- $\{x : O(\tau) \mid P(x)\}$ trivially admissible, as P is constant

- in particular,

$$\{c : \mathbb{N} \to O(\mathbb{N}) \mid \exists n \in \mathbb{N}.\ c(n) = \Omega_{\mathbb{N}}\} \cong \mathbb{N} \to O(\mathbb{N})$$

Admissibility: This talk

Collapsed equality

- subsets of partial types useless

$$\{c : \mathbb{N} \to O(\mathbb{N}) \mid \exists n \in \mathbb{N}.\ c(n) = \Omega_{\mathbb{N}}\} \cong \mathbb{N} \to O(\mathbb{N})$$

- but partial subset types are not

$$\Pi n : \mathbb{N}.\ \Pi G : \mathbb{G}.\ O(1 + \{f : V_G \to \mathbb{N} \mid \mathit{coloring}(G, f, n)\})$$

- **they express partial correctness specs**

Admissibility: This talk

Benefits

- avoid all admissibility conditions
- full power of underlying dependent type theory
- easily implementable as extension of existing DTT

Drawbacks

- no equational reasoning about effectful computations

Cheap implemention of a spec logic in a DTT

Hoare Type Theory

Hoare Type Theory

- extends DTT with partial stateful computations

Hoare Type Theory

Hoare Type Theory

- extends DTT with partial stateful computations
- new version: extends CIC
- **implementable as axiomatic extension of Coq**

Hoare Type Theory

Hoare Type Theory

- extends DTT with partial stateful computations
- new version: extends CIC
- **implementable as axiomatic extension of Coq**
- demonstrate approach scales to realistic DTTs
- illustrate expressiveness despite collapsed equality

HTT: Underlying DTT

Universes

- **Prop** and **Set** (impredicative) and **Type** (predicative)

$$\textbf{Prop} : \textbf{Type} \qquad\qquad \textbf{Set} : \textbf{Type}$$

and $\textbf{Prop} \overset{prf}{\subseteq} \textbf{Set} \overset{el}{\subseteq} \textbf{Type}$

Type constructors

- **Set**, **Type**: $1, \Sigma, \Pi, W$
- **Prop**: 1, *weak* Σ, Π

HTT: Effectful computations

- partial stateful computations
- index partial types by pre- and post-condition

$$\Gamma \vdash \{P\}\tau\{Q\} : \mathbf{Set}$$

HTT: Effectful computations

- partial stateful computations
- index partial types by pre- and post-condition

$$\Gamma \vdash \{P\}\tau\{Q\} : \mathbf{Set}$$

- heap type to reason about computation states

$$\mathbf{Heap} : \mathbf{Type}$$

$$\mathbf{empty}, h[l \mapsto_{\tau} v], \ldots$$

HTT: Effectful computations

- partial stateful computations
- index partial types by pre- and post-condition

$$\Gamma \vdash \{P\}\tau\{Q\} : \mathbf{Set}$$

- heap type to reason about computation states

$$\mathbf{Heap} : \mathbf{Type}$$

$$\mathbf{empty}, h[l \mapsto_\tau v], \ldots$$

- pre- and post-condition expressed as **Heap** predicates

$$P : \mathbf{Heap} \to \mathbf{Prop}$$
$$Q : \tau \to \mathbf{Heap} \to \mathbf{Heap} \to \mathbf{Prop}$$

HTT: Example

Stack ADT

$$\Pi\alpha : \textbf{Set}.\ \Sigma\beta : \textbf{Set}.\ \Sigma \mathit{inv} : \beta \times \alpha\ \mathit{seq} \times \textbf{Heap} \to \textbf{Prop}.$$

$$\{\lambda i.\ i = \textbf{empty}\}\beta\{\lambda r, i, t.\ \mathit{inv}(r, [], t)\} \times$$

$$\Pi r : \beta.\ \Pi v : \alpha.\ \{\lambda i.\ \exists l, \mathit{inv}(r, l, i)\}$$

$$1$$

$$\{\lambda r, i, t.\ \forall l, \mathit{inv}(r, l, i) \Rightarrow \mathit{inv}(r, v :: l, t)\} \times$$

$$\ldots$$

- β : abstract representation type
- *inv* : abstract representation predicate

Admissibility in PER models

PER models

- partial equivalence relations over universal pre-domain $\mathbb{V}$

$$\mathbb{V} \cong 1 + \mathbb{N} + (\mathbb{V} \times \mathbb{V}) + (\mathbb{V} \to \mathbb{V}_\perp) + \mathbb{V}_\perp$$

- models a dependent type universe with $1, \Sigma, \Pi, W$-types

Admissibility in PER models

PER models

- partial equivalence relations over universal pre-domain $\mathbb{V}$

$$\mathbb{V} \cong 1 + \mathbb{N} + (\mathbb{V} \times \mathbb{V}) + (\mathbb{V} \to \mathbb{V}_\perp) + \mathbb{V}_\perp$$

- models a dependent type universe with $1, \Sigma, \Pi, W$-types

Partiality

- $\mathbf{fix} : (in_T(R) \to in_T(R)) \to in_T(R)$ for admissible R
- PERs model DTT $+$ $\mathbf{fix}_\tau$ with explicit adm. proofs

Admissibility in PER models

Complete PERs

- closed under limits of ω-chains
- all partial types admissible
- complete PERs **do not** model strong Σ-types

Admissibility in PER models

Monotone PERs

- a PER $R \subseteq \mathbb{V} \times \mathbb{V}$ is monotone iff

$$\forall x, y \in |R|.\ x \leq y \Rightarrow (x, y) \in R$$

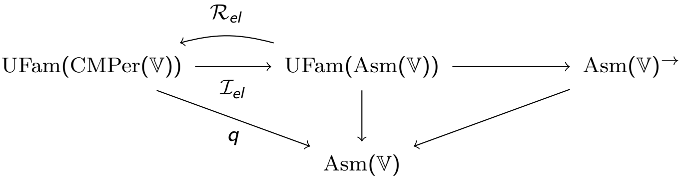
- collapses equality on PERs with a least element
- standard DTT types $(0, 1, \mathbb{N}, +, \Sigma, \Pi, W)$ monotone
- complete monotone PERs **do** model strong Σ-types
- CMPERs model DTT $+$ $\mathbf{fix}_\tau$ with collapsed O-equality

HTT model

Scales to HTT

- contexts and types modelled with assemblies
- small types modelled with complete monotone PERs
- propositions modelled as regular subobjects of assemblies

HTT model



- split fibred reflection $(\mathcal{R}_{el} \dashv \mathcal{I}_{el})$
- the coproducts induced by $(\mathcal{R}_{el} \dashv \mathcal{I}_{el})$ are strong
- split generic object for q in $\mathrm{Asm}(\mathbb{V})$
- $\mathcal{I}_{el} \circ \mathcal{R}_{el}$ preserves W-types from types in the image of $\mathcal{I}_{el}$

Theorem: Underlying DTT is sound.

Summary

We have

- presented a new approach to general recursion in DTT
- presented a semantic account of this approach
- shown that it scales to a model of Coq
- implemented it as an axiomatic extension of Coq