

# Fibred Computational Effects

(thesis overview)

Danel Ahman

(Prosecco team, INRIA Paris)

June 29, 2017

# Overall aims

## Investigate language design principles for combining

- refinement types  $(\langle \text{op} \rangle(\tau_1, \dots, \tau_n), \dots)$
- dependent types  $(\Pi, \Sigma, V =_A W, \dots)$
- computational effects (state, I/O, probability, recursion, ...)

## The goal of this work was to establish that

- refinement types and comp. effects admit a natural combination
- dependent types and comp. effects admit a **natural combination**
- this combination covers a **wide range** of computational effects

## The work was driven by two guiding questions

- Should one allow effectful programs in types?
- How should one treat type-dependency in sequential composition?

# Thesis structure

- **Chapter 1:** Introduction and related work
- **Chapter 2:** Preliminaries of models of effects and dep. types
- **Chapter 3:** The core language eMLTT
- **Chapter 4:** Categorical models of eMLTT
- **Chapter 5:** Interpretation of eMLTT, soundness, completeness
- **Chapter 6:** An extension of eMLTT with algebraic effects
- **Chapter 7:** An extension of eMLTT with handlers
- **Chapter 8:** Conclusion and future work
- **Appendices:** Details of longer proofs

# Chapter 3: The core language eMLTT

## Values and computations

- clear distinction, at the level of types and terms (CBPV, EEC)
- both kinds of types are allowed to depend only on values
- selective dependency on computations is possible via thunks

## Value and computation types

$$A, B ::= \dots \mid \Sigma x:A. B \mid \Pi x:A. B \mid V =_A W \mid U\underline{C} \mid \underline{C} \multimap D$$
$$\underline{C}, \underline{D} ::= FA \mid \Sigma x:A. \underline{C} \mid \Pi x:A. \underline{C}$$

## Value, computation, and **homomorphism** terms

$$V, W ::= x \mid \dots \mid \text{thunk } M \mid \lambda z:\underline{C}. K$$
$$\begin{aligned} M, N ::= & \text{force}_{\underline{C}} V \mid \text{return } V \mid M \text{ to } x:A \text{ in}_{\underline{C}} N \\ & \mid \langle V, M \rangle \mid M \text{ to } (x:A, z:\underline{C}) \text{ in}_{\underline{D}} K \mid \dots \end{aligned}$$
$$K, L ::= z \mid K \text{ to } x:A \text{ in}_{\underline{C}} M \mid \langle V, K \rangle \mid \dots$$

# Chapter 3: The core language eMLTT

## Values and computations

- clear distinction, at the level of types and terms (CBPV, EEC)
- both kinds of types are allowed to depend only on values
- selective dependency on computations is possible via thunks

## Value and computation types

$$A, B ::= \dots \mid \Sigma x:A. B \mid \Pi x:A. B \mid V =_A W \mid U \underline{C} \mid \underline{C} \multimap D$$
$$\underline{C}, \underline{D} ::= F A \mid \Sigma x:A. \underline{C} \mid \Pi x:A. \underline{C}$$

## Value, computation, and homomorphism terms

$$V, W ::= x \mid \dots \mid \text{thunk } M \mid \lambda z:\underline{C}. K$$
$$M, N ::= \text{force}_{\underline{C}} V \mid \text{return } V \mid M \text{ to } x:A \text{ in}_{\underline{C}} N \\ \mid \langle V, M \rangle \mid M \text{ to } (x:A, z:\underline{C}) \text{ in}_{\underline{D}} K \mid \dots$$
$$K, L ::= z \mid K \text{ to } x:A \text{ in}_{\underline{C}} M \mid \langle V, K \rangle \mid \dots$$

# Chapter 3: The core language eMLTT

## Values and computations

- clear distinction, at the level of types and terms (CBPV, EEC)
- both kinds of types are allowed to depend only on values
- selective dependency on computations is possible via thunks

## Value and computation types

$$A, B ::= \dots \mid \Sigma x:A. B \mid \Pi x:A. B \mid V =_A W \mid U\underline{C} \mid \underline{C} \multimap \underline{D}$$
$$\underline{C}, \underline{D} ::= FA \mid \Sigma x:A. \underline{C} \mid \Pi x:A. \underline{C}$$

## Value, computation, and homomorphism terms

$$V, W ::= x \mid \dots \mid \text{thunk } M \mid \lambda z:\underline{C}. K$$
$$M, N ::= \text{force}_{\underline{C}} V \mid \text{return } V \mid M \text{ to } x:A \text{ in}_{\underline{C}} N \\ \mid \langle V, M \rangle \mid M \text{ to } (x:A, z:\underline{C}) \text{ in}_{\underline{D}} K \mid \dots$$
$$K, L ::= z \mid K \text{ to } x:A \text{ in}_{\underline{C}} M \mid \langle V, K \rangle \mid \dots$$

# Chapter 3: The core language eMLTT

## Values and computations

- clear distinction, at the level of types and terms (CBPV, EEC)
- both kinds of types are allowed to depend only on values
- selective dependency on computations is possible via thunks

## Value and computation types

$$A, B ::= \dots \mid \Sigma x:A. B \mid \Pi x:A. B \mid V =_A W \mid U \underline{C} \mid \underline{C} \multimap \underline{D}$$
$$\underline{C}, \underline{D} ::= FA \mid \Sigma x:A. \underline{C} \mid \Pi x:A. \underline{C}$$

## Value, computation, and homomorphism terms

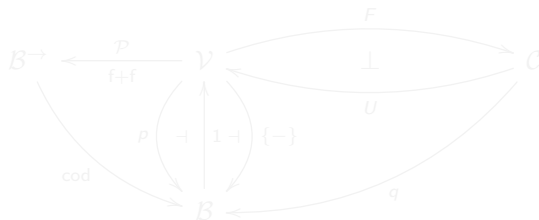
$$V, W ::= x \mid \dots \mid \text{thunk } M \mid \lambda z:\underline{C}. K$$
$$\begin{aligned} M, N ::= & \text{force}_{\underline{C}} V \mid \text{return } V \mid M \text{ to } x:A \text{ in}_{\underline{C}} N \\ & \mid \langle V, M \rangle \mid M \text{ to } (x:A, z:\underline{C}) \text{ in}_{\underline{D}} K \mid \dots \end{aligned}$$
$$K, L ::= z \mid K \text{ to } x:A \text{ in}_{\underline{C}} M \mid \langle V, K \rangle \mid \dots$$

# Chapter 4: Fibred adjunction models

## A natural combination of

- fibrational models of dependent types
- adjunction-based models of computational effects

as depicted in



## Also in Chapter 4

- structures for modelling eMLTT's value and comp. types
- examples of such models, based on models of EEC, families of sets fibration, fibred EM-resolution, and continuous families fibration

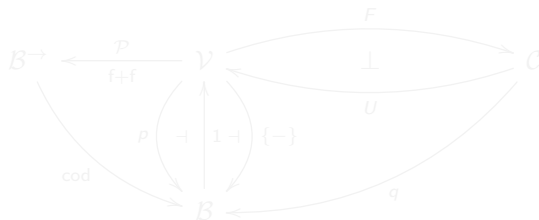


# Chapter 4: Fibred adjunction models

## A natural combination of

- fibrational models of dependent types
- adjunction-based models of computational effects

as depicted in



## Also in Chapter 4

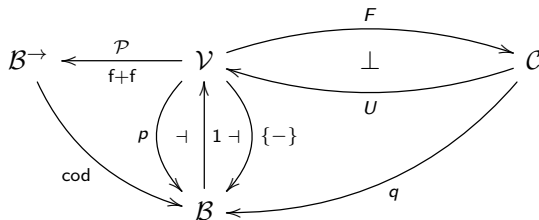
- structures for modelling eMLTT's value and comp. types
- examples of such models, based on models of EEC, families of sets fibration, fibred EM-resolution, and continuous families fibration

# Chapter 4: Fibred adjunction models

## A natural combination of

- fibrational models of dependent types
- adjunction-based models of computational effects

as depicted in



Also in Chapter 4

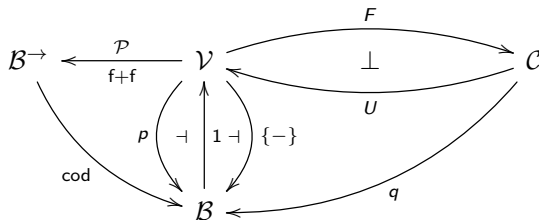
- structures for modelling eMLTT's value and comp. types
- examples of such models, based on models of EEC, families of sets fibration, fibred EM-resolution, and continuous families fibration

# Chapter 4: Fibred adjunction models

## A natural combination of

- fibrational models of dependent types
- adjunction-based models of computational effects

as depicted in



## Also in Chapter 4

- [structures](#) for modelling eMLTT's value and comp. types
- [examples](#) of such models, based on models of EEC, families of sets fibration, fibred EM-resolution, and continuous families fibration

# Chapter 5: Interpretation of eMLTT

## Interpretation $\llbracket - \rrbracket$

- an a priori partial interpretation function that, if defined, maps
  - contexts to objects of  $\mathcal{B}$
  - value types to objects of  $\mathcal{V}$  and computation types to objects of  $\mathcal{C}$
  - value and computation terms to global elements in the fibres of  $\mathcal{V}$
  - homomorphism terms to vertical morphisms in  $\mathcal{C}$

## Soundness

- weakening is interpreted as reindexing along  $\pi_{\llbracket \Gamma; A \rrbracket}$
- substitution is interpreted as reindexing along  $s(\llbracket \Gamma; V \rrbracket)$
- $\llbracket - \rrbracket$  is defined on well-formed syntax
- $\llbracket - \rrbracket$  identifies definitionally equal contexts, types, and terms

## Completeness

- constructing a classifying model from the syntax of eMLTT

# Chapter 5: Interpretation of eMLTT

## Interpretation $\llbracket - \rrbracket$

- an a priori partial interpretation function that, if defined, maps
  - **contexts** to objects of  $\mathcal{B}$
  - **value types** to objects of  $\mathcal{V}$  and **computation types** to objects of  $\mathcal{C}$
  - **value** and **computation terms** to global elements in the fibres of  $\mathcal{V}$
  - **homomorphism terms** to vertical morphisms in  $\mathcal{C}$

## Soundness

- weakening is interpreted as reindexing along  $\pi_{\llbracket \Gamma; A \rrbracket}$
- substitution is interpreted as reindexing along  $s(\llbracket \Gamma; V \rrbracket)$
- $\llbracket - \rrbracket$  is defined on well-formed syntax
- $\llbracket - \rrbracket$  identifies definitionally equal contexts, types, and terms

## Completeness

- constructing a classifying model from the syntax of eMLTT

# Chapter 5: Interpretation of eMLTT

## Interpretation $\llbracket - \rrbracket$

- an a priori partial interpretation function that, if defined, maps
  - **contexts** to objects of  $\mathcal{B}$
  - **value types** to objects of  $\mathcal{V}$  and **computation types** to objects of  $\mathcal{C}$
  - **value** and **computation terms** to global elements in the fibres of  $\mathcal{V}$
  - **homomorphism terms** to vertical morphisms in  $\mathcal{C}$

## Soundness

- **weakening** is interpreted as reindexing along  $\pi_{\llbracket \Gamma; A \rrbracket}$
- **substitution** is interpreted as reindexing along  $s(\llbracket \Gamma; V \rrbracket)$
- $\llbracket - \rrbracket$  is defined on well-formed syntax
- $\llbracket - \rrbracket$  identifies definitionally equal contexts, types, and terms

## Completeness

- constructing a classifying model from the syntax of eMLTT

# Chapter 5: Interpretation of eMLTT

## Interpretation $\llbracket - \rrbracket$

- an a priori partial interpretation function that, if defined, maps
  - **contexts** to objects of  $\mathcal{B}$
  - **value types** to objects of  $\mathcal{V}$  and **computation types** to objects of  $\mathcal{C}$
  - **value** and **computation terms** to global elements in the fibres of  $\mathcal{V}$
  - **homomorphism terms** to vertical morphisms in  $\mathcal{C}$

## Soundness

- **weakening** is interpreted as reindexing along  $\pi_{\llbracket \Gamma; A \rrbracket}$
- **substitution** is interpreted as reindexing along  $s(\llbracket \Gamma; V \rrbracket)$
- $\llbracket - \rrbracket$  is defined on well-formed syntax
- $\llbracket - \rrbracket$  identifies definitionally equal contexts, types, and terms

## Completeness

- constructing a classifying model from the syntax of eMLTT

# Chapter 6: eMLTT with algebraic effects

## Fibred effect theories $\mathcal{T}_{\text{eff}} = (\mathcal{S}_{\text{eff}}, \mathcal{E}_{\text{eff}})$

- dependently typed operation symbols  $\text{op} : (x : I) \longrightarrow O$
- effect terms  $T ::= w(V) \mid \text{op}_V(y.T) \mid \dots$
- equations  $\Gamma \mid \Delta \vdash T_1 = T_2$
- examples
  - global state with location-dependent store types
  - dependently typed update monads

## Algebraic effects in eMLTT $_{\mathcal{T}_{\text{eff}}}$

- algebraic operations as comp. terms  $M ::= \dots \mid \text{op}_{\underline{C}}^{\underline{C}}(y.M)$
- equations are included via translation  $(\Gamma \mid \Delta \vdash T)_{A, \vec{V}_i, \vec{V}_j', \vec{W}_{\text{op}}}$
- general algebraicity equation (for  $\Gamma \mid z : \underline{C} \Vdash K : \underline{D}$ )

$$\Gamma \Vdash K[\text{op}_{\underline{C}}^{\underline{C}}(y.M)/z] = \text{op}_{\underline{V}}^{\underline{D}}(y.K[M/z]) : \underline{D}$$



# Chapter 6: eMLTT with algebraic effects

## Fibred effect theories $\mathcal{T}_{\text{eff}} = (\mathcal{S}_{\text{eff}}, \mathcal{E}_{\text{eff}})$

- dependently typed **operation symbols**  $\text{op} : (x:I) \longrightarrow O$
- **effect terms**  $T ::= w(V) \mid \text{op}_V(y.T) \mid \dots$
- **equations**  $\Gamma \mid \Delta \vdash T_1 = T_2$
- examples
  - global state with location-dependent store types
  - dependently typed update monads

## Algebraic effects in eMLTT $_{\mathcal{T}_{\text{eff}}}$

- algebraic operations as comp. terms  $M ::= \dots \mid \text{op}_{\underline{V}}^{\underline{C}}(y.M)$
- equations are included via translation  $(\Gamma \mid \Delta \vdash T)_{A, \vec{V}_i, \vec{V}_j', \vec{W}_{\text{op}}}$
- general algebraicity equation (for  $\Gamma \mid z : \underline{C} \Vdash K : \underline{D}$ )

$$\Gamma \Vdash K[\text{op}_{\underline{V}}^{\underline{C}}(y.M)/z] = \text{op}_{\underline{V}}^{\underline{D}}(y.K[M/z]) : \underline{D}$$

# Chapter 6: eMLTT with algebraic effects

## Fibred effect theories $\mathcal{T}_{\text{eff}} = (\mathcal{S}_{\text{eff}}, \mathcal{E}_{\text{eff}})$

- dependently typed **operation symbols**  $\text{op} : (x : I) \longrightarrow O$
- **effect terms**  $T ::= w(V) \mid \text{op}_V(y.T) \mid \dots$
- **equations**  $\Gamma \mid \Delta \vdash T_1 = T_2$
- examples
  - global state with location-dependent store types
  - dependently typed update monads

## Algebraic effects in eMLTT $_{\mathcal{T}_{\text{eff}}}$

- algebraic operations as comp. terms  $M ::= \dots \mid \text{op}_{\underline{V}}^{\underline{C}}(y.M)$
- equations are included via translation  $(\Gamma \mid \Delta \vdash T)_{A, \vec{V}_i, \vec{V}_j', \vec{W}_{\text{op}}}$
- general algebraicity equation (for  $\Gamma \mid z : \underline{C} \Vdash K : \underline{D}$ )

$$\Gamma \Vdash K[\text{op}_{\underline{V}}^{\underline{C}}(y.M)/z] = \text{op}_{\underline{V}}^{\underline{D}}(y.K[M/z]) : \underline{D}$$

# Chapter 6: eMLTT with algebraic effects

**Fibred effect theories**  $\mathcal{T}_{\text{eff}} = (\mathcal{S}_{\text{eff}}, \mathcal{E}_{\text{eff}})$

- dependently typed **operation symbols**  $\text{op} : (x : I) \longrightarrow O$
- **effect terms**  $T ::= w(V) \mid \text{op}_V(y.T) \mid \dots$
- **equations**  $\Gamma \mid \Delta \vdash T_1 = T_2$
- examples
  - global state with location-dependent store types
  - dependently typed update monads

**Algebraic effects in eMLTT** $_{\mathcal{T}_{\text{eff}}}$

- **algebraic operations** as comp. terms  $M ::= \dots \mid \text{op}_{\vec{V}}^{\vec{C}}(y.M)$
- equations are included via **translation**  $\langle \Gamma \mid \Delta \vdash T \rangle_{A, \vec{V}_i, \vec{V}_j', \vec{W}_{\text{op}}}$
- general **algebraicity equation** (for  $\Gamma \mid z : \underline{C} \Vdash K : \underline{D}$ )

$$\Gamma \Vdash K[\text{op}_{\vec{V}}^{\vec{C}}(y.M)/z] = \text{op}_{\vec{V}}^{\vec{D}}(y.K[M/z]) : \underline{D}$$

# Chapter 7: eMLTT $_{\mathcal{T}_{\text{eff}}}$ with handlers

## Problem with the conventional term-level def. of handlers

- as handling denotes a homomorphism, natural to include

$K$  handled with  $\{\text{op}_x(x') \mapsto N_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}}$  to  $y:A$  in  $N_{\text{ret}}$

- but then can prove unsound equations such as

$$\Gamma \models \text{write}_{\text{true}}^{F1}(\text{return} \star) = \text{write}_{\text{false}}^{F1}(\text{return} \star) : F1$$

## Handlers in eMLTT $_{\mathcal{T}_{\text{eff}}}^{\mathcal{H}}$

- user-defined algebra type  $\underline{C} ::= \dots \mid \langle A, \{\text{V}_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}} \rangle$
- composition operations  $M ::= \dots \mid M \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$   
 $K ::= \dots \mid K \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$

## This extension enables to

- derive the conventional presentation of handlers and handling
- define predicates  $\Gamma \vdash V : UFA \rightarrow VU$  on effectful computations

# Chapter 7: eMLTT $_{\mathcal{T}_{\text{eff}}}$ with handlers

## Problem with the conventional term-level def. of handlers

- as **handling** denotes a homomorphism, natural to include

$K$  handled with  $\{\text{op}_x(x') \mapsto N_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}}$  to  $y:A$  in  $N_{\text{ret}}$

- but then can prove **unsound** equations such as

$$\Gamma \models \text{write}_{\text{true}}^{F1}(\text{return } \star) = \text{write}_{\text{false}}^{F1}(\text{return } \star) : F1$$

## Handlers in eMLTT $_{\mathcal{T}_{\text{eff}}}^{\mathcal{H}}$

- user-defined algebra type  $\underline{C} ::= \dots \mid \langle A, \{\text{V}_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}} \rangle$
- composition operations  $M ::= \dots \mid M \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$   
 $K ::= \dots \mid K \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$

## This extension enables to

- derive the conventional presentation of handlers and handling
- define predicates  $\Gamma \Vdash V : UFA \rightarrow \text{VU}$  on effectful computations

# Chapter 7: eMLTT $_{\mathcal{T}_{\text{eff}}}$ with handlers

## Problem with the conventional term-level def. of handlers

- as **handling** denotes a homomorphism, natural to include

$K$  handled with  $\{\text{op}_x(x') \mapsto N_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}}$  to  $y:A$  in  $N_{\text{ret}}$

- but then can prove **unsound** equations such as

$$\Gamma \models \text{write}_{\text{true}}^{F1}(\text{return } \star) = \text{write}_{\text{false}}^{F1}(\text{return } \star) : F1$$

## Handlers in eMLTT $_{\mathcal{T}_{\text{eff}}}^{\mathcal{H}}$

- user-defined algebra type  $\underline{C} ::= \dots \mid \langle A, \{V_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}} \rangle$
- composition operations  $M ::= \dots \mid M \text{ as } x:U\underline{C} \text{ in}_{\underline{D}} N$   
 $K ::= \dots \mid K \text{ as } x:U\underline{C} \text{ in}_{\underline{D}} N$

This extension enables to

- derive the conventional presentation of handlers and handling
- define predicates  $\Gamma \Vdash V : UFA \rightarrow \text{VU}$  on effectful computations

# Chapter 7: eMLTT $_{\mathcal{T}_{\text{eff}}}$ with handlers

## Problem with the conventional term-level def. of handlers

- as **handling** denotes a homomorphism, natural to include

$K$  handled with  $\{\text{op}_x(x') \mapsto N_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}}$  to  $y:A$  in  $N_{\text{ret}}$

- but then can prove **unsound** equations such as

$$\Gamma \models \text{write}_{\text{true}}^{F1}(\text{return } \star) = \text{write}_{\text{false}}^{F1}(\text{return } \star) : F1$$

## Handlers in eMLTT $_{\mathcal{T}_{\text{eff}}}^{\mathcal{H}}$

- user-defined algebra type  $\underline{C} ::= \dots \mid \langle A, \{V_{\text{op}}\}_{\text{op} \in \mathcal{S}_{\text{eff}}} \rangle$
- composition operations  $M ::= \dots \mid M \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$   
 $K ::= \dots \mid K \text{ as } x:U\underline{C} \text{ in } \underline{D} \ N$

## This extension enables to

- derive the **conventional presentation** of handlers and handling
- define **predicates**  $\Gamma \models V : UFA \rightarrow \text{VU}$  on effectful computations

# Thesis structure

- **Chapter 1:** Introduction and related work
- **Chapter 2:** Preliminaries of models of effects and dep. types
- **Chapter 3:** The core language eMLTT
- **Chapter 4:** Categorical models of eMLTT
- **Chapter 5:** Interpretation of eMLTT, soundness, completeness
- **Chapter 6:** An extension of eMLTT with algebraic effects
- **Chapter 7:** An extension of eMLTT with handlers
- **Chapter 8:** Conclusion and future work
- **Appendices:** Details of longer proofs