Generic Bidirectional Typing for Dependent Type Theories

Thiago Felicissimo

October 3, 2023

Type annotations in dependent type theory

Dependent type theory suffers from verbosity of type annotations

Application: $t @_{A,x.B} u$

Dependent pair: $\langle t, u \rangle_{A,x.B}$

Cons: $t ::_A l$

Type annotations in dependent type theory

Dependent type theory suffers from verbosity of type annotations

$$\text{Application:}\quad t @_{A,x.B} u$$

$$\text{Dependent pair:}\quad \langle t, u \rangle_{A,x.B}$$

$$\text{Cons:}\quad t ::_A l$$

Unusable in practice...

Type annotations in dependent type theory

Dependent type theory suffers from verbosity of type annotations

Application: $t @_{A,x.B} u$

Dependent pair: $\langle t, u \rangle_{A,x.B}$

Cons: $t ::_A l$

Unusable in practice...

Most presentation restore usability by eliding type annotations from syntax

Application: $t\ u$

Dependent pair: $\langle t, u \rangle$

Cons: $t :: l$

Type annotations in dependent type theory

Dependent type theory suffers from verbosity of type annotations

Application: $t @_{A,x.B} u$

Dependent pair: $\langle t, u \rangle_{A,x.B}$

Cons: $t ::_A l$

Unusable in practice...

Most presentation restore usability by eliding type annotations from syntax

Application: $t\ u$

Dependent pair: $\langle t, u \rangle$

Cons: $t :: l$

Syntax so common that many don't realize that an omission is being made

Typechecking without annotations

Omission has a cost Knowing annotations is needed for typing

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, x : A \vdash B \text{ type} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.B}$$

Typechecking without annotations

Omission has a cost Knowing annotations is needed for typing

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, x : A \vdash B \text{ type} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.B}$$

How to find A and B if they're not stored in syntax?

Typechecking without annotations

Omission has a cost Knowing annotations is needed for typing

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, x : A \vdash B \text{ type} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.B}$$

How to find A and B if they're not stored in syntax?

Bidirectional typing Decompose $t : A$ in modes check $t \Leftarrow A$ and infer $t \Rightarrow A$

Typechecking without annotations

Omission has a cost Knowing annotations is needed for typing

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, x : A \vdash B \text{ type} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.B}$$

How to find A and B if they're not stored in syntax?

Bidirectional typing Decompose $t : A$ in modes check $t \Leftarrow A$ and infer $t \Rightarrow A$

Allow specify flow of type information in typing rules, explain how to use them

$$\frac{C \longrightarrow^* \Sigma x : A.B \qquad \Gamma \vdash t \Leftarrow A \qquad \Gamma \vdash u \Leftarrow B[t/x]}{\Gamma \vdash \langle t, u \rangle \Leftarrow C}$$

Typechecking without annotations

Omission has a cost Knowing annotations is needed for typing

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, x : A \vdash B \text{ type} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.B}$$

How to find A and B if they're not stored in syntax?

Bidirectional typing Decompose $t : A$ in modes check $t \Leftarrow A$ and infer $t \Rightarrow A$

Allow specify flow of type information in typing rules, explain how to use them

$$\frac{C \longrightarrow^{*} \Sigma x : A.B \qquad \Gamma \vdash t \Leftarrow A \qquad \Gamma \vdash u \Leftarrow B[t/x]}{\Gamma \vdash \langle t, u \rangle \Leftarrow C}$$

Complements unannotated syntax very well, explains how to recover annotations

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Roadmap

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Roadmap

1. We give a general definition of type theories (or equivalently, a logical framework) supporting non-annotated syntaxes

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Roadmap

1. We give a general definition of type theories (or equivalently, a logical framework) supporting non-annotated syntaxes
2. For each theory, we define declarative and bidirectional type systems

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Roadmap

1. We give a general definition of type theories (or equivalently, a logical framework) supporting non-annotated syntaxes
2. For each theory, we define declarative and bidirectional type systems
3. We show, in a theory-independent fashion, their equivalence

Contribution

Bidirectional type systems have been studied and proposed for many theories

However, general guidelines have remained informal, no unified framework

This talk Generic account of bidirectional typing for class of type theories

Roadmap

1. We give a general definition of type theories (or equivalently, a logical framework) supporting non-annotated syntaxes
2. For each theory, we define declarative and bidirectional type systems
3. We show, in a theory-independent fashion, their equivalence
4. We derive decidability of typing for weak normalizing theories

The theories

The intrinsically-scoped syntax

$$
\begin{array}{rcrcll}
\boxed{\mathrm{Tm}\ \theta\ \gamma} & \ni & t, u, T, U & ::= & \mid x & \text{if } x \in \gamma \\
 & & & & \mid \mathsf{x}\{\vec{t}\ \in \mathrm{Sub}\ \theta\ \gamma\ \delta\} & \text{if } \mathsf{x}\{\delta\} \in \theta \\
 & & & & \mid c(\mathbf{t}\ \in \mathrm{MSub}\ \theta\ \gamma\ \xi) & \text{if } c(\xi) \in \Sigma \\
 & & & & \mid d(t\ \in \mathrm{Tm}\ \theta\ \gamma; \mathbf{t}\ \in \mathrm{MSub}\ \theta\ \gamma\ \xi) & \text{if } d(\xi) \in \Sigma \\
\boxed{\mathrm{Sub}\ \theta\ \gamma\ \delta} & \ni & \vec{t}, \vec{u}, \vec{s}, \vec{v} & ::= & \mid \epsilon & \text{if } \delta = \cdot \\
 & & & & \mid \vec{t}'\ \in \mathrm{Sub}\ \theta\ \gamma\ \delta', t\ \in \mathrm{Tm}\ \theta\ \gamma & \text{if } \delta = \delta', x \\
\boxed{\mathrm{MSub}\ \theta\ \gamma\ \xi} & \ni & \mathbf{t}, \mathbf{u}, \mathbf{s}, \mathbf{v} & ::= & \mid \epsilon & \text{if } \xi = \cdot \\
 & & & & \mid \mathbf{t}'\ \in \mathrm{MSub}\ \theta\ \gamma\ \xi', \vec{x}_\delta.t\ \in \mathrm{Tm}\ \theta\ \gamma.\delta & \text{if } \xi = \xi', \mathsf{x}\{\delta\}
\end{array}
$$

The intrinsically-scoped syntax

$\boxed{\mathrm{Tm}\ \theta\ \gamma} \ni \quad t, u, T, U ::= \mid x \qquad \text{if } x \in \gamma$

$\mid \mathsf{x}\{\vec{t} \in \mathrm{Sub}\ \theta\ \gamma\ \delta\} \qquad \text{if } \mathsf{x}\{\delta\} \in \theta$

$\mid c(\mathbf{t} \in \mathrm{MSub}\ \theta\ \gamma\ \xi) \qquad \text{if } c(\xi) \in \Sigma$

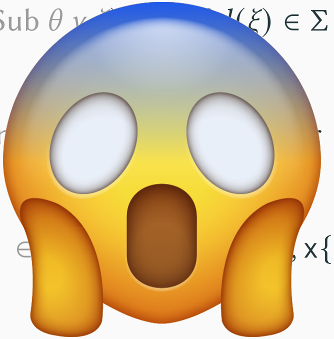
$\mid d(t \in \mathrm{Tm}\ \theta\ \gamma; \mathbf{t} \in \mathrm{MSub}\ \theta\ \gamma$ $(\xi) \in \Sigma$

$\boxed{\mathrm{Sub}\ \theta\ \gamma\ \delta} \ni \quad \vec{t}, \vec{u}, \vec{s}, \vec{v} ::= \mid \epsilon$

$\mid \vec{t}' \in \mathrm{Sub}\ \theta\ \gamma\ \delta', t \in \mathrm{Tr}$

$\boxed{\mathrm{MSub}\ \theta\ \gamma\ \xi} \ni \quad \mathbf{t}, \mathbf{u}, \mathbf{s}, \mathbf{v} ::= \mid \epsilon$

$\mid \mathbf{t}' \in \mathrm{MSub}\ \theta\ \gamma\ \xi', \vec{x}_\delta.t \in$ $, \mathsf{x}\{\delta\}$



The syntax

Terms and substitutions

Write x for variables, x for metavariables, c for constructors and d for destructors

$$\boxed{\mathrm{Tm}} \ni t, u, T, U ::= x \mid \mathsf{x}\{\vec{t}\} \mid c(\mathbf{t}) \mid d(t; \mathbf{t}) \quad (t \text{ is called the } \textit{principal argument})$$

$$\boxed{\mathrm{Sub}} \ni \vec{t}, \vec{u} ::= \epsilon \mid \vec{t}, t$$

$$\boxed{\mathrm{MSub}} \ni \mathbf{t}, \mathbf{u} ::= \epsilon \mid \mathbf{t}, \vec{x}.t$$

The syntax

Terms and substitutions

Write x for variables, x for metavariables, c for constructors and d for destructors

$$\boxed{\mathrm{Tm}} \ni t, u, T, U ::= x \mid \mathsf{x}\{\vec{t}\} \mid c(\mathbf{t}) \mid d(t; \mathbf{t}) \quad (t \text{ is called the } \textit{principal argument})$$

$$\boxed{\mathrm{Sub}} \ni \vec{t}, \vec{u} ::= \epsilon \mid \vec{t}, t$$

$$\boxed{\mathrm{MSub}} \ni \mathbf{t}, \mathbf{u} ::= \epsilon \mid \mathbf{t}, \vec{x}.t$$

Example

$$\begin{aligned} \Sigma_{\lambda\Pi} = \quad & \Pi(\mathsf{A}, \mathsf{B}\{x\}),\ \lambda(\mathsf{t}\{x\}),\ \mathrm{Ty},\ \mathrm{Tm}(\mathsf{A}), && \text{(constructors)} \\ & @(\mathsf{u}) && \text{(destructors)} \end{aligned}$$

$$t, u, A, B ::= x \mid \mathsf{x}\{\vec{t}\} \mid \mathrm{Ty} \mid \mathrm{Tm}(A) \mid @(t; u) \mid \lambda(x.t) \mid \Pi(A, x.B)$$

The syntax

Contexts

$$\boxed{\mathrm{Ctx}} \ni \Gamma, \Delta ::= \cdot \mid \Gamma, x : T$$

$$\boxed{\mathrm{MCtx}} \ni \Theta, \Xi ::= \cdot \mid \Theta, \mathrm{x}\{\Gamma\} : T$$

Example: $\Gamma = A : \mathrm{Ty}, x : \mathrm{Tm}(A), y : \mathrm{Tm}(\Pi(A, z.A))$

The syntax

Contexts

$$\boxed{\mathsf{Ctx}} \ni \Gamma, \Delta ::= \cdot \mid \Gamma, x : T$$

$$\boxed{\mathsf{MCtx}} \ni \Theta, \Xi ::= \cdot \mid \Theta, \mathsf{x}\{\Gamma\} : T$$

Example: $\Gamma = A : \mathrm{Ty}, x : \mathrm{Tm}(A), y : \mathrm{Tm}(\Pi(A, z.A))$

(Linear) Patterns

$$\boxed{\mathsf{Tm}^{\mathsf{P}}} \ni t^{\mathsf{P}}, u^{\mathsf{P}} ::= \mathsf{x}\{\vec{x}\} \mid c(\mathbf{t}^{\mathsf{P}})$$

$$\boxed{\mathsf{MSub}^{\mathsf{P}}} \ni \mathbf{t}^{\mathsf{P}}, \mathbf{u}^{\mathsf{P}} ::= \epsilon \mid \mathbf{t}^{\mathsf{P}}, \vec{x}.t^{\mathsf{P}} \quad (\mathsf{metas}(\mathbf{t}^{\mathsf{P}}) \cap \mathsf{metas}(t^{\mathsf{P}}) = \emptyset)$$

The theories

A *theory* $\mathbb{T}$ is made of *schematic typing rules* and *rewrite rules*.

3 schematic typing rules: *sort rules*, *constructor rules* and *destructor rules*

The theories

A *theory* $\mathbb{T}$ is made of *schematic typing rules* and *rewrite rules*.

3 schematic typing rules: *sort rules*, *constructor rules* and *destructor rules*

Sort rules Used to define judgment forms of the theory.

Example: In MLTT, 2 judgment forms: $\square$ type and $\square : A$ for a type A.

$$\frac{}{\vdash \mathrm{Ty}\ \mathsf{sort}} \qquad\qquad \frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{Tm}(\mathsf{A})\ \mathsf{sort}}$$

The theories

A *theory* $\mathbb{T}$ is made of *schematic typing rules* and *rewrite rules*.

3 schematic typing rules: *sort rules*, *constructor rules* and *destructor rules*

Sort rules Used to define judgment forms of the theory.

Example: In MLTT, 2 judgment forms: $\square$ type and $\square : A$ for a type A.

$$\frac{}{\vdash \mathrm{Ty}\ \mathsf{sort}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{Tm}(\mathsf{A})\ \mathsf{sort}}$$

Formally, of the form $c(\Theta)$ sort, where Θ represent premises.

Example in formal notation: $\mathrm{Ty}(\cdot)$ sort and $\mathrm{Tm}(\mathsf{A} : \mathrm{Ty})$ sort

The theories

Constructor rules Two groups of premises: Θ_1 erased and Θ_2 kept in the syntax.

To recover Θ_1, rule will be used in mode check, so the type should be pattern containing metavariables of Θ_1.

$$\frac{\vdash \mathsf{A} : \mathbf{Ty} \qquad x : \mathbf{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathbf{Ty}}{\vdash \Pi(\mathsf{A}, \mathsf{B}) : \mathbf{Ty}} \qquad \frac{\begin{array}{c}\vdash \mathsf{A} : \mathbf{Ty} \qquad x : \mathbf{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathbf{Ty} \\ x : \mathbf{Tm}(\mathsf{A}) \vdash \mathsf{t} : \mathbf{Tm}(\mathsf{B}\{x\})\end{array}}{\vdash \lambda(\mathsf{t}) : \mathbf{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))}$$

The theories

Constructor rules Two groups of premises: Θ_1 erased and Θ_2 kept in the syntax.

To recover Θ_1, rule will be used in mode check, so the type should be pattern containing metavariables of Θ_1.

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty}}{\vdash \Pi(\mathsf{A}, \mathsf{B}) : \mathrm{Ty}} \qquad \frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \\ x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{t} : \mathrm{Tm}(\mathsf{B}\{x\})\end{array}}{\vdash \lambda(\mathsf{t}) : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))}$$

Formally, of the form $c(\Theta_1; \Theta_2) : U^{\mathsf{P}}$

Example in formal notation: $\Pi(\cdot;\ \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty}) : \mathrm{Ty}$ and $\lambda(\mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty};\ \mathsf{t}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Tm}(\mathsf{B}\{x\})) : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))$.

The theories

Destructor rules Two groups of premises: Θ_1 erased and Θ_2 kept in the syntax.

But also a *principal argument* $\mathsf{x} : T^{\mathsf{P}}$, a pattern allowing to recover Θ_1.

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \qquad \vdash \mathtt{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) \qquad \vdash \mathsf{u} : \mathrm{Tm}(\mathsf{A})}{\vdash @(\mathtt{t}; \mathsf{u}) : \mathrm{Tm}(\mathsf{B}\{\mathtt{t}\})}$$

The theories

Destructor rules Two groups of premises: Θ_1 erased and Θ_2 kept in the syntax.

But also a *principal argument* $\mathsf{x} : T^{\mathsf{P}}$, a pattern allowing to recover Θ_1.

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \qquad \vdash \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) \qquad \vdash \mathsf{u} : \mathrm{Tm}(\mathsf{A})}{\vdash @(\mathsf{t}; \mathsf{u}) : \mathrm{Tm}(\mathsf{B}\{\mathsf{t}\})}$$

Formally, of the form $d(\Theta_1;\ \mathsf{x} : T^{\mathsf{P}};\ \Theta_2) : U$

Example in formal notation:
$@(\mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty};\ \ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}));\ \ \mathsf{u} : \mathrm{Tm}(\mathsf{A})) : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})$.

The theories

Rewrite rules Define the definitional equality (aka conversion) $\equiv$ of the theory.

$$@(\lambda(x.\mathtt{t}\{x\}); \mathtt{u}) \longmapsto \mathtt{t}\{\mathtt{u}\}$$

In general, of the form $d(t^{\mathsf{P}}; \mathbf{u}^{\mathsf{P}}) \longmapsto r$ with $(\mathrm{metas}(t^{\mathsf{P}}) \cap \mathrm{metas}(\mathbf{u}^{\mathsf{P}}) = \emptyset)$.

The theories

Rewrite rules Define the definitional equality (aka conversion) $\equiv$ of the theory.

$$@(\lambda(x.\mathtt{t}\{x\});\mathtt{u}) \longmapsto \mathtt{t}\{\mathtt{u}\}$$

In general, of the form $d(t^{\mathsf{P}};\mathbf{u}^{\mathsf{P}}) \longmapsto r$ with $(\mathrm{metas}(t^{\mathsf{P}}) \cap \mathrm{metas}(\mathbf{u}^{\mathsf{P}}) = \emptyset)$.

Condition: no two left-hand sides unify.

Therefore, rewrite systems are orthogonal, hence confluent by construction!

The theories

Rewrite rules Define the definitional equality (aka conversion) ≡ of the theory.

$$@(\lambda(x.\mathsf{t}\{x\});\mathsf{u}) \longmapsto \mathsf{t}\{\mathsf{u}\}$$

In general, of the form $d(t^{\mathsf{P}};\mathbf{u}^{\mathsf{P}}) \longmapsto r$ with $(\mathrm{metas}(t^{\mathsf{P}}) \cap \mathrm{metas}(\mathbf{u}^{\mathsf{P}}) = \emptyset)$.

Condition: no two left-hand sides unify.

Therefore, rewrite systems are orthogonal, hence confluent by construction!

Full example Theory $\mathbb{T}_{\lambda\Pi}$.

$$\mathrm{Ty}(\cdot)\ \mathsf{sort} \qquad \mathrm{Tm}(\mathsf{A} : \mathrm{Ty})\ \mathsf{sort} \qquad \Pi(\cdot;\ \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty}) : \mathrm{Ty}$$

$$\lambda(\mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty};\ \ \mathsf{t}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Tm}(\mathsf{B}\{x\})) : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))$$

$$@(\mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty};\ \ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}));\ \ \mathsf{u} : \mathrm{Tm}(\mathsf{A})) : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})$$

$$@(\lambda(x.\mathsf{t}\{x\});\mathsf{u}) \longmapsto \mathsf{t}\{\mathsf{u}\}$$

Declarative typing

Declarative typing rules

Each theory $\mathbb{T}$ defines a declarative type system.

Judgments $\Theta; \Gamma \vdash T$ sort and $\Theta; \Gamma \vdash t : T$ and $\Theta; \Gamma \vdash \vec{t} : \Delta$ and $\Theta; \Gamma \vdash \mathbf{t} : \Xi$.

Declarative typing rules

Each theory $\mathbb{T}$ defines a declarative type system.

Judgments $\Theta; \Gamma \vdash T$ sort and $\Theta; \Gamma \vdash t : T$ and $\Theta; \Gamma \vdash \vec{t} : \Delta$ and $\Theta; \Gamma \vdash \mathbf{t} : \Xi$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

Declarative typing rules

Each theory $\mathbb{T}$ defines a declarative type system.

Judgments $\Theta; \Gamma \vdash T$ sort and $\Theta; \Gamma \vdash t : T$ and $\Theta; \Gamma \vdash \vec{t} : \Delta$ and $\Theta; \Gamma \vdash \mathbf{t} : \Xi$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$c(\Xi) \text{ sort} \in \mathbb{T} \; \overset{\textsc{Sort}}{\dfrac{\Theta; \Gamma \vdash \mathbf{t} : \Xi}{\Theta; \Gamma \vdash c(\mathbf{t}) \text{ sort}}}$$

Declarative typing rules

Each theory $\mathbb{T}$ defines a declarative type system.

Judgments $\Theta;\Gamma \vdash T$ sort and $\Theta;\Gamma \vdash t : T$ and $\Theta;\Gamma \vdash \vec{t} : \Delta$ and $\Theta;\Gamma \vdash \mathbf{t} : \Xi$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$\textsc{Sort}\quad \frac{c(\Xi)\ \text{sort} \in \mathbb{T} \qquad \Theta;\Gamma \vdash \mathbf{t} : \Xi}{\Theta;\Gamma \vdash c(\mathbf{t})\ \text{sort}}$$

$$\textsc{Cons}\quad \frac{c(\Xi_1;\Xi_2) : T \in \mathbb{T} \qquad \Theta;\Gamma \vdash \mathbf{t}, \mathbf{u} : \Xi_1.\Xi_2}{\Theta;\Gamma \vdash c(\mathbf{u}) : T[\mathbf{t}]}$$

Declarative typing rules

Each theory $\mathbb{T}$ defines a declarative type system.

Judgments $\Theta; \Gamma \vdash T$ sort and $\Theta; \Gamma \vdash t : T$ and $\Theta; \Gamma \vdash \vec{t} : \Delta$ and $\Theta; \Gamma \vdash \mathbf{t} : \Xi$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$\textsc{Sort}\quad \frac{c(\Xi) \text{ sort} \in \mathbb{T} \qquad \Theta; \Gamma \vdash \mathbf{t} : \Xi}{\Theta; \Gamma \vdash c(\mathbf{t}) \text{ sort}}$$

$$\textsc{Cons}\quad \frac{c(\Xi_1; \Xi_2) : T \in \mathbb{T} \qquad \Theta; \Gamma \vdash \mathbf{t}, \mathbf{u} : \Xi_1.\Xi_2}{\Theta; \Gamma \vdash c(\mathbf{u}) : T[\mathbf{t}]}$$

$$\textsc{Dest}\quad \frac{d(\Xi_1; \mathsf{x} : T; \Xi_2) : U \in \mathbb{T} \qquad \Theta; \Gamma \vdash \mathbf{t}, t, \mathbf{u} : \Xi_1.(\mathsf{x} : T).\Xi_2}{\Theta; \Gamma \vdash d(t; \mathbf{u}) : U[\mathbf{t}, t, \mathbf{u}]}$$

Example

$$
\begin{array}{llllllll}
d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\
@(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})
\end{array}
$$

$$
\frac{\Theta; \Gamma \vdash \mathbf{t}, t, \mathbf{u} : \Xi_1.(\mathsf{x} : T).\Xi_2}{\Theta; \Gamma \vdash d(t; \mathbf{u}) : U[\mathbf{t}, t, \mathbf{u}]}
$$

Example

$$
\begin{array}{llllllll}
d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\
@(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})
\end{array}
$$

$$
\frac{\Theta; \Gamma \vdash (A,\ x.B,\ t,\ u) : (\mathsf{A} : \mathrm{Ty},\ \mathsf{B}(x : \mathrm{Tm}(\mathsf{A})) : \mathrm{Ty},\ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})),\ \mathsf{u} : \mathrm{Tm}(\mathsf{A}))}{\Theta; \Gamma \vdash @(t; u) : \mathrm{Tm}(B[u/x])}
$$

Example

$$\begin{array}{lllll} d(& \Xi_1 & ;\ \mathsf{x} : T^{\mathsf{P}} & ;\ \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ;\ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ;\ \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{c}\Theta; \Gamma \vdash (A,\ x.B,\ t) : (\mathsf{A} : \mathrm{Ty},\ \mathsf{B}(x : \mathrm{Tm}(\mathsf{A})) : \mathrm{Ty},\ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))) \\ \Theta; \Gamma \vdash u : \mathrm{Tm}(A)\end{array}}{\Theta; \Gamma \vdash @(t; u) : \mathrm{Tm}(B[u/x])}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{c}\Theta; \Gamma \vdash (A,\ x.B) : (\mathsf{A} : \mathrm{Ty},\ \mathsf{B}(x : \mathrm{Tm}(\mathsf{A})) : \mathrm{Ty}) \\ \Theta; \Gamma \vdash t : \mathrm{Tm}(\Pi(A, x.B)) \qquad \Theta; \Gamma \vdash u : \mathrm{Tm}(A)\end{array}}{\Theta; \Gamma \vdash @(t; u) : \mathrm{Tm}(B[u/x])}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{cc} \Theta; \Gamma \vdash (A) : (\mathsf{A} : \mathrm{Ty}) & \Theta; \Gamma, x : \mathrm{Tm}(A) \vdash B : \mathrm{Ty} \\ \Theta; \Gamma \vdash t : \mathrm{Tm}(\Pi(A, x.B)) & \Theta; \Gamma \vdash u : \mathrm{Tm}(A) \end{array}}{\Theta; \Gamma \vdash @(t; u) : \mathrm{Tm}(B[u/x])}$$

Example

$$\begin{array}{llllllll}
d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\
@(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})
\end{array}$$

$$\frac{\begin{array}{c}\Theta; \Gamma \vdash A : \mathrm{Ty} \qquad \Theta; \Gamma, x : \mathrm{Tm}(A) \vdash B : \mathrm{Ty} \\ \Theta; \Gamma \vdash t : \mathrm{Tm}(\Pi(A, x.B)) \qquad \Theta; \Gamma \vdash u : \mathrm{Tm}(A)\end{array}}{\Theta; \Gamma \vdash @(t; u) : \mathrm{Tm}(B[u/x])}$$

Properties of declarative system

In the following, we assume theory the $\mathbb{T}$ to be *valid* (definition not given here).

Properties of declarative system

In the following, we assume theory the $\mathbb{T}$ to be *valid* (definition not given here).

Weakening, substitution, sorts are well-typed Easy proofs

Properties of declarative system

In the following, we assume theory the $\mathbb{T}$ to be *valid* (definition not given here).

Weakening, substitution, sorts are well-typed Easy proofs

Moreover, our valid theories all satisfy **subject reduction**!
(Important to show soundness of bidirectional system)

$$\Gamma \vdash t : T \text{ and } t \longrightarrow t' \text{ implies } \Gamma \vdash t' : T$$

Properties of declarative system

In the following, we assume theory the $\mathbb{T}$ to be *valid* (definition not given here).

Weakening, substitution, sorts are well-typed Easy proofs

Moreover, our valid theories all satisfy **subject reduction**!
(Important to show soundness of bidirectional system)

$$\Gamma \vdash t : T \text{ and } t \longrightarrow t' \text{ implies } \Gamma \vdash t' : T$$

Proof uses the following key lemma:

Key property of patterns If $\Theta \vdash t^{\mathsf{P}} : T$ and $\Delta \vdash t^{\mathsf{P}}[\mathbf{v}] : T[\mathbf{v}]$ then $\Delta \vdash \mathbf{v} : \Theta$

Bidirectional typing

Matching modulo rewriting

In bidirectional typing, we need matching modulo rewriting to recover missing arguments.

$$\frac{\Gamma \vdash t \Rightarrow U \qquad \dots}{\Gamma \vdash @(t; u) \Rightarrow}$$

Matching modulo rewriting

In bidirectional typing, we need matching modulo rewriting to recover missing arguments.

$$\frac{\Gamma \vdash t \Rightarrow U \qquad \dots}{\Gamma \vdash @(t;u) \Rightarrow}$$

We know

$$U \equiv \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\}))[A/\mathsf{A},\ x.B/\mathsf{B}]$$

but how to recover A and B from U?

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \rightsquigarrow \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \rightsquigarrow \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

$$\frac{}{\mathsf{x}\{\vec{x}\} \prec u \rightsquigarrow \vec{x}.u} \qquad\qquad u \longrightarrow^{*} c(\mathbf{u}) \frac{\mathbf{t}^{\mathsf{P}} \prec \mathbf{u} \rightsquigarrow \mathbf{v}}{c(\mathbf{t}^{\mathsf{P}}) \prec u \rightsquigarrow \mathbf{v}}$$

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \rightsquigarrow \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

$$\frac{}{\mathsf{x}\{\vec{x}\} \prec u \rightsquigarrow \vec{x}.u} \qquad\qquad u \longrightarrow^{*} c(\mathbf{u})\,\frac{\mathbf{t}^{\mathsf{P}} \prec \mathbf{u} \rightsquigarrow \mathbf{v}}{c(\mathbf{t}^{\mathsf{P}}) \prec u \rightsquigarrow \mathbf{v}}$$

Soundness If $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}$ then $T^{\mathsf{P}}[\mathbf{v}] \equiv U$

Proof: induction on matching rules

Completeness If $T^{\mathsf{P}}[\mathbf{v}] \equiv U$ then $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}'$ with $\mathbf{v} \equiv \mathbf{v}'$

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \leadsto \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

$$\frac{}{\mathsf{x}\{\vec{x}\} \prec u \leadsto \vec{x}.u} \qquad u \longrightarrow^{*} c(\mathbf{u}) \frac{\mathbf{t}^{\mathsf{P}} \prec \mathbf{u} \leadsto \mathbf{v}}{c(\mathbf{t}^{\mathsf{P}}) \prec u \leadsto \mathbf{v}}$$

Soundness If $T^{\mathsf{P}} \prec U \leadsto \mathbf{v}$ then $T^{\mathsf{P}}[\mathbf{v}] \equiv U$

Proof: induction on matching rules

Completeness If $T^{\mathsf{P}}[\mathbf{v}] \equiv U$ then $T^{\mathsf{P}} \prec U \leadsto \mathbf{v}'$ with $\mathbf{v} \equiv \mathbf{v}'$

Proof ?

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \rightsquigarrow \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

$$\frac{}{\mathsf{x}\{\vec{x}\} \prec u \rightsquigarrow \vec{x}.u} \qquad\qquad u \longrightarrow^{*}_{\mathrm{m/o}} c(\mathbf{u}) \frac{\mathbf{t}^{\mathsf{P}} \prec \mathbf{u} \rightsquigarrow \mathbf{v}}{c(\mathbf{t}^{\mathsf{P}}) \prec u \rightsquigarrow \mathbf{v}}$$

Soundness If $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}$ then $T^{\mathsf{P}}[\mathbf{v}] \equiv U$

Proof: induction on matching rules

Maximal outermost strategy m/o contracts all outermost redexes in one step

Completeness If $T^{\mathsf{P}}[\mathbf{v}] \equiv U$ then $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}'$ with $\mathbf{v} \equiv \mathbf{v}'$

Proof ?

Matching modulo rewriting

Given t^{P} and u, judgment $t^{\mathsf{P}} \prec u \rightsquigarrow \mathbf{v}$ tries to compute $\mathbf{v}$ with $t^{\mathsf{P}}[\mathbf{v}] \equiv u$.

$$\frac{}{\mathsf{x}\{\vec{x}\} \prec u \rightsquigarrow \vec{x}.u} \qquad\qquad u \longrightarrow^{*}_{\mathrm{m/o}} c(\mathbf{u})\,\frac{\mathbf{t}^{\mathsf{P}} \prec \mathbf{u} \rightsquigarrow \mathbf{v}}{c(\mathbf{t}^{\mathsf{P}}) \prec u \rightsquigarrow \mathbf{v}}$$

Soundness If $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}$ then $T^{\mathsf{P}}[\mathbf{v}] \equiv U$

Proof: induction on matching rules

Maximal outermost strategy m/o contracts all outermost redexes in one step

Completeness If $T^{\mathsf{P}}[\mathbf{v}] \equiv U$ then $T^{\mathsf{P}} \prec U \rightsquigarrow \mathbf{v}'$ with $\mathbf{v} \equiv \mathbf{v}'$

Proof: Uses the fact that $\longrightarrow_{\mathrm{m/o}}$ is head-normalizing for orthogonal systems.

Inferable and checkable terms

Not all unannotated terms can be algorithmically typed

$$\frac{\dfrac{?}{\Gamma \vdash \lambda(x.t) \Rightarrow ?} \qquad \dots}{\Gamma \vdash @(\lambda(x.t); u) \Rightarrow ?}$$

Inferable and checkable terms

Not all unannotated terms can be algorithmically typed

$$\dfrac{\dfrac{?}{\Gamma \vdash \lambda(x.t) \Rightarrow ?} \qquad \cdots}{\Gamma \vdash @(\lambda(x.t); u) \Rightarrow ?}$$

Avoided by defining bidirectional typing only for *inferrable* and *checkable* terms.

$$\boxed{\mathsf{Tm}^{\mathsf{i}}} \ni t^{\mathsf{i}}, u^{\mathsf{i}} ::= x \mid d(t^{\mathsf{i}}; \mathbf{t}^{\mathsf{c}})$$

$$\boxed{\mathsf{Tm}^{\mathsf{c}}} \ni t^{\mathsf{c}}, u^{\mathsf{c}} ::= c(\mathbf{t}^{\mathsf{c}}) \mid \underline{t}^{\mathsf{i}}$$

$$\boxed{\mathsf{MSub}^{\mathsf{c}}} \ni \mathbf{t}^{\mathsf{c}}, \mathbf{u}^{\mathsf{c}} ::= \epsilon \mid \mathbf{t}^{\mathsf{c}}, \vec{x}.t^{\mathsf{c}}$$

Inferable and checkable terms

Not all unannotated terms can be algorithmically typed

$$\dfrac{\dfrac{?}{\Gamma \vdash \lambda(x.t) \Rightarrow ?} \qquad \dots}{\Gamma \vdash @(\lambda(x.t); u) \Rightarrow ?}$$

Avoided by defining bidirectional typing only for *inferrable* and *checkable* terms.

$$\boxed{\mathsf{Tm}^{\mathsf{i}}} \ni t^{\mathsf{i}}, u^{\mathsf{i}} ::= x \mid d(t^{\mathsf{i}}; \mathbf{t}^{\mathsf{c}})$$

$$\boxed{\mathsf{Tm}^{\mathsf{c}}} \ni t^{\mathsf{c}}, u^{\mathsf{c}} ::= c(\mathbf{t}^{\mathsf{c}}) \mid \underline{t}^{\mathsf{i}}$$

$$\boxed{\mathsf{MSub}^{\mathsf{c}}} \ni \mathbf{t}^{\mathsf{c}}, \mathbf{u}^{\mathsf{c}} ::= \epsilon \mid \mathbf{t}^{\mathsf{c}}, \vec{x}.t^{\mathsf{c}}$$

Principal argument of a destructor can only be variable or another destructor.

For most theories: Tm^{c} = normal forms, and Tm^{i} = neutrals

Bidirectional typing rules

Judgments $\Gamma \vdash T^{\mathsf{c}} \Leftarrow \mathsf{sort}$ and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$ and $\Gamma \mid \mathbf{v} : \Xi \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Theta$.

Bidirectional typing rules

Judgments $\Gamma \vdash T^{\mathrm{c}} \Leftarrow \mathrm{sort}$ and $\Gamma \vdash t^{\mathrm{c}} \Leftarrow T$ and $\Gamma \vdash t^{\mathrm{i}} \Rightarrow T$ and $\Gamma \mid \mathbf{v} : \Xi \vdash \mathbf{t}^{\mathrm{c}} \Leftarrow \Theta$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

Bidirectional typing rules

Judgments $\Gamma \vdash T^{\mathsf{c}} \Leftarrow \mathsf{sort}$ and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$ and $\Gamma \mid \mathbf{v} : \Xi \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Theta$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$c(\Xi)\ \mathsf{sort} \in \mathbb{T}\ \dfrac{\Gamma \mid \varepsilon : (\cdot) \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Xi}{\Gamma \vdash c(\mathbf{t}^{\mathsf{c}}) \Leftarrow \mathsf{sort}}\ \textsc{Sort}$$

Bidirectional typing rules

Judgments $\Gamma \vdash T^{\mathsf{c}} \Leftarrow \mathsf{sort}$ and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$ and $\Gamma \mid \mathbf{v} : \Xi \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Theta$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$\textsc{Sort}\quad \frac{c(\Xi)\ \mathsf{sort} \in \mathbb{T} \qquad \Gamma \mid \varepsilon : (\cdot) \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Xi}{\Gamma \vdash c(\mathbf{t}^{\mathsf{c}}) \Leftarrow \mathsf{sort}}$$

$$\textsc{Cons}\quad \frac{c(\Xi_1;\Xi_2) : T \in \mathbb{T} \qquad T \prec U \rightsquigarrow \mathbf{v} \qquad \Gamma \mid \mathbf{v} : \Xi_1 \vdash \mathbf{u}^{\mathsf{c}} \Leftarrow \Xi_2}{\Gamma \vdash c(\mathbf{u}^{\mathsf{c}}) \Leftarrow U}$$

Bidirectional typing rules

Judgments $\Gamma \vdash T^{\mathsf{c}} \Leftarrow \text{sort}$ and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$ and $\Gamma \mid \mathbf{v} : \Xi \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Theta$.

Main rules, instantiate the schematic rules of $\mathbb{T}$:

$$\textsc{Sort}\quad c(\Xi)\ \text{sort} \in \mathbb{T}\ \frac{\Gamma \mid \varepsilon : (\cdot) \vdash \mathbf{t}^{\mathsf{c}} \Leftarrow \Xi}{\Gamma \vdash c(\mathbf{t}^{\mathsf{c}}) \Leftarrow \text{sort}}$$

$$\textsc{Cons}\quad c(\Xi_1;\Xi_2) : T \in \mathbb{T}\ \frac{T < U \rightsquigarrow \mathbf{v} \qquad \Gamma \mid \mathbf{v} : \Xi_1 \vdash \mathbf{u}^{\mathsf{c}} \Leftarrow \Xi_2}{\Gamma \vdash c(\mathbf{u}^{\mathsf{c}}) \Leftarrow U}$$

$$\textsc{Dest}\quad d(\Xi_1; \mathtt{t} : T; \Xi_2) : U \in \mathbb{T}\ \frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow V \qquad T < V \rightsquigarrow \mathbf{v} \qquad \Gamma \mid (\mathbf{v}, t) : (\Xi_1, \mathtt{t} : T) \vdash \mathbf{u}^{\mathsf{c}} \Leftarrow \Xi_2}{\Gamma \vdash d(t^{\mathsf{i}}; \mathbf{u}^{\mathsf{c}}) \Rightarrow U[\mathbf{v}, t, \mathbf{u}]}$$

Example

$$\begin{array}{llclclcl} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

Dest

$$\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow V \qquad T \prec V \rightsquigarrow \mathbf{v} \qquad \Gamma \mid (\mathbf{v}, t) : (\Xi_1, \mathsf{t} : T) \vdash \mathbf{u}^{\mathsf{c}} \Leftarrow \Xi_2}{\Gamma \vdash d(t^{\mathsf{i}}; \mathbf{u}^{\mathsf{c}}) \Rightarrow U[\mathbf{v}, t, \mathbf{u}]}$$

Example

$$\begin{array}{lllll} d(& \Xi_1 & ;\ \mathsf{x} : T^{\mathsf{P}} & ;\ \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ;\ \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ;\ \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow ?}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow T'}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{c}\Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \\ \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow ?\end{array}}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$\begin{array}{llllll}
d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\
@(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})
\end{array}$$

$$\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \qquad \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow A/\mathsf{A},\ x.B/\mathsf{B}}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{c} \Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \\ \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow A/\mathsf{A},\ x.B/\mathsf{B} \\ \Gamma \mid\ (A,\ x.B, t) : (\mathsf{A},\ \mathsf{B},\ \mathsf{t}) \vdash (u^{\mathsf{c}}) \Leftarrow (\mathsf{u} : \mathrm{Tm}(\mathsf{A}))) \end{array}}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\begin{array}{c} \Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \\ \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow A/\mathsf{A},\ x.B/\mathsf{B} \\ \Gamma \vdash u^{\mathsf{c}} \Leftarrow \mathrm{Tm}(A) \end{array}}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow ?}$$

Example

$$
\begin{array}{llllll}
d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & : U \\
@(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})
\end{array}
$$

$$
\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \qquad \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow A/\mathsf{A},\ x.B/\mathsf{B} \qquad \Gamma \vdash u^{\mathsf{c}} \Leftarrow \mathrm{Tm}(A)}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\})[A/\mathsf{A},\ x.B/\mathsf{B},\ t/\mathsf{t},\ u/\mathsf{u}]}
$$

Example

$$\begin{array}{llllllll} d(& \Xi_1 & ; & \mathsf{x} : T^{\mathsf{P}} & ; & \Xi_2) & & : U \\ @(& \mathsf{A} : \mathrm{Ty},\ \mathsf{B}\{x : \mathrm{Tm}(\mathsf{A})\} : \mathrm{Ty} & ; & \mathsf{t} : \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) & ; & \mathsf{u} : \mathrm{Tm}(\mathsf{A})) & & : \mathrm{Tm}(\mathsf{B}\{\mathsf{u}\}) \end{array}$$

$$\frac{\Gamma \vdash t^{\mathsf{i}} \Rightarrow T' \qquad \mathrm{Tm}(\Pi(\mathsf{A}, x.\mathsf{B}\{x\})) < T' \rightsquigarrow A/\mathsf{A},\ x.B/\mathsf{B} \qquad \Gamma \vdash u^{\mathsf{c}} \Leftarrow \mathrm{Tm}(A)}{\Gamma \vdash @(t^{\mathsf{i}}; u^{\mathsf{c}}) \Rightarrow \mathrm{Tm}(B[u/x])}$$

Equivalence with declarative typing

Suppose underlying theory $\mathbb{T}$ is valid.

Equivalence with declarative typing

Suppose underlying theory $\mathbb{T}$ is valid.

Soundness If $\Gamma \vdash$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$, or $\Gamma \vdash T$ sort and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$, then $\Gamma \vdash t : T$.

Proof: Uses soundness of matching, subject reduction and key property of patterns

Equivalence with declarative typing

Suppose underlying theory $\mathbb{T}$ is valid.

Soundness If $\Gamma \vdash$ and $\Gamma \vdash t^{\mathsf{i}} \Rightarrow T$, or $\Gamma \vdash T$ sort and $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$, then $\Gamma \vdash t : T$.

Proof: Uses soundness of matching, subject reduction and key property of patterns

Completeness For t^{i} inferable, if $\Gamma \vdash t : T$ then $\Gamma \vdash t^{\mathsf{i}} \Rightarrow U$ with $T \equiv U$.
For t^{c} checkable, if $\Gamma \vdash t : T$ then $\Gamma \vdash t^{\mathsf{c}} \Leftarrow T$.

Proof: Uses completeness of matching

Equivalence with declarative typing

Suppose underlying theory $\mathbb{T}$ is valid.

Soundness If $\Gamma \vdash$ and $\Gamma \vdash t^{\mathrm{i}} \Rightarrow T$, or $\Gamma \vdash T$ sort and $\Gamma \vdash t^{\mathrm{c}} \Leftarrow T$, then $\Gamma \vdash t : T$.

Proof: Uses soundness of matching, subject reduction and key property of patterns

Completeness For t^{i} inferable, if $\Gamma \vdash t : T$ then $\Gamma \vdash t^{\mathrm{i}} \Rightarrow U$ with $T \equiv U$.
For t^{c} checkable, if $\Gamma \vdash t : T$ then $\Gamma \vdash t^{\mathrm{c}} \Leftarrow T$.

Proof: Uses completeness of matching

Decidability If $\mathbb{T}$ weak normalizing, then inference is decidable for inferable terms, and checking is decidable for checkable terms.

More examples

Dependent sums

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty}}{\vdash \Sigma(\mathsf{A}, \mathsf{B}) : \mathrm{Ty}}$$

$$\frac{\begin{array}{cc} \vdash \mathsf{A} : \mathrm{Ty} & x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \\ \vdash \mathsf{t} : \mathrm{Tm}(\mathsf{A}) & \vdash \mathsf{u} : \mathrm{Tm}(\mathsf{B}\{\mathsf{t}\}) \end{array}}{\vdash \mathrm{pair}(\mathsf{t}, \mathsf{u}) : \mathrm{Tm}(\Sigma(\mathsf{A}, x.\mathsf{B}\{x\}))}$$

$$\frac{\begin{array}{cc} \vdash \mathsf{A} : \mathrm{Ty} & x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \\ \multicolumn{2}{c}{\vdash \mathsf{t} : \mathrm{Tm}(\Sigma(\mathsf{A}, x.\mathsf{B}\{x\}))} \end{array}}{\vdash \mathrm{proj}_1(\mathsf{t}; \cdot) : \mathrm{Tm}(\mathsf{A})}$$

$$\frac{\begin{array}{cc} \vdash \mathsf{A} : \mathrm{Ty} & x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \\ \multicolumn{2}{c}{\vdash \mathsf{t} : \mathrm{Tm}(\Sigma(\mathsf{A}, x.\mathsf{B}\{x\}))} \end{array}}{\vdash \mathrm{proj}_2(\mathsf{t}; \cdot) : \mathrm{Tm}(\mathsf{B}\{\mathrm{proj}_1(\mathsf{t})\})}$$

$$\mathrm{proj}_1(\mathrm{pair}(\mathsf{t}, \mathsf{u}); \varepsilon) \longmapsto \mathsf{t} \qquad \mathrm{proj}_2(\mathrm{pair}(\mathsf{t}, \mathsf{u}); \varepsilon) \longmapsto \mathsf{u}$$

Lists

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{List}(\mathsf{A}) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{nil} : \mathrm{Tm}(\mathrm{List}(\mathsf{A}))} \qquad \frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{x} : \mathrm{Tm}(\mathsf{A}) \\ \vdash \mathsf{l} : \mathrm{Tm}(\mathrm{List}(\mathsf{A}))\end{array}}{\vdash \mathrm{cons}(\mathsf{x}, \mathsf{l}) : \mathrm{Tm}(\mathrm{List}(\mathsf{A}))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{l} : \mathrm{Tm}(\mathrm{List}(\mathsf{A})) \qquad x : \mathrm{Tm}(\mathrm{List}(\mathsf{A})) \vdash \mathsf{P} : \mathrm{Ty} \qquad \vdash \mathsf{pnil} : \mathrm{Tm}(\mathsf{P}\{\mathrm{nil}\}) \\ x : \mathrm{Tm}(\mathsf{A}), y : \mathrm{Tm}(\mathrm{List}(\mathsf{A})), z : \mathrm{Tm}(\mathsf{P}\{y\}) \vdash \mathsf{pcons} : \mathrm{Tm}(\mathsf{P}\{\mathrm{cons}(x, y)\})\end{array}}{\vdash \mathrm{ListRec}(\mathsf{l}; \mathsf{P}, \mathsf{pnil}, \mathsf{pcons}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{l}\})}$$

$$\mathrm{ListRec}(\mathrm{nil}; x.\mathsf{P}\{x\}, \mathsf{pnil}, xyz.\mathsf{pcons}\{x, y, z\}) \longmapsto \mathsf{pnil}$$

$$\mathrm{ListRec}(\mathrm{cons}(\mathsf{x}, \mathsf{l}); x.\mathsf{P}\{x\}, \mathsf{pnil}, xyz.\mathsf{pcons}\{x, y, z\}) \longmapsto$$
$$\mathsf{pcons}\{\mathsf{x}, \mathsf{l}, \mathrm{ListRec}(\mathsf{l}; x.\mathsf{P}\{x\}, \mathsf{pnil}, xyz.\mathsf{pcons}\{x, y, z\})\}$$

W types

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty}}{\vdash \mathrm{W}(\mathsf{A}, \mathsf{B}) : \mathrm{Ty}} \qquad \frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \\ \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{f} : \mathrm{Tm}(\Pi(\mathsf{B}\{\mathsf{a}\}, x'.\mathrm{W}(\mathsf{A}, x.\mathsf{B}\{x\})))\end{array}}{\vdash \mathrm{sup}(\mathsf{a}, \mathsf{f}) : \mathrm{Tm}(\mathrm{W}(\mathsf{A}, x.\mathsf{B}\{x\}))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad x : \mathrm{Tm}(\mathsf{A}) \vdash \mathsf{B} : \mathrm{Ty} \qquad \vdash \mathsf{t} : \mathrm{Tm}(\mathrm{W}(\mathsf{A}, x.\mathsf{B}\{x\})) \qquad x : \mathrm{Tm}(\mathrm{W}(\mathsf{A}, x.\mathsf{B}\{x\})) \vdash \mathsf{P} : \mathrm{Ty} \\ x : \mathrm{Tm}(\mathsf{A}), y : \mathrm{Tm}(\Pi(\mathsf{B}\{x\}, x'.\mathrm{W}(\mathsf{A}, x.\mathsf{B}\{x\}))), z : \mathrm{Tm}(\Pi(\mathsf{B}\{x\}, x'.\mathsf{P}\{@(y, x')\})) \vdash \mathsf{p} : \mathrm{Tm}(\mathsf{P}\{\mathrm{sup}(x, y)\})\end{array}}{\vdash \mathrm{WRec}(\mathsf{t}; \mathsf{P}, \mathsf{p}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{t}\})}$$

$$\mathrm{WRec}(\mathrm{sup}(\mathsf{a}, \mathsf{f}); x.\mathsf{P}\{x\}, xyz.\mathsf{p}\{x, y, z\}) \longmapsto \mathsf{p}\{\mathsf{a}, \mathsf{f}, \lambda(x.\mathrm{WRec}(@(\mathsf{f}, x); x.\mathsf{P}\{x\}, xyz.\mathsf{p}\{x, y, z\}))\}$$

Equality

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{refl} : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{a}))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{t} : \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) \\ x : \mathrm{Tm}(\mathsf{A}), y : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, x)) \vdash \mathsf{P} : \mathrm{Ty} \qquad \vdash \mathsf{p} : \mathrm{Tm}(\mathsf{P}\{\mathsf{a}, \mathrm{refl}\})\end{array}}{\vdash \mathrm{J}(\mathsf{t}; \mathsf{P}, \mathsf{p}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{b}, \mathsf{t}\})}$$

$$\mathrm{J}(\mathrm{refl}, xy.\mathsf{P}\{x, y\}, \mathsf{p}) \longmapsto \mathsf{p}$$

Equality

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{refl} : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{a}))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{t} : \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) \\ x : \mathrm{Tm}(\mathsf{A}), y : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, x)) \vdash \mathsf{P} : \mathrm{Ty} \qquad \vdash \mathsf{p} : \mathrm{Tm}(\mathsf{P}\{\mathsf{a}, \mathrm{refl}\})\end{array}}{\vdash \mathrm{J}(\mathsf{t}; \mathsf{P}, \mathsf{p}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{b}, \mathsf{t}\})}$$

$$\mathrm{J}(\mathrm{refl}, xy.\mathsf{P}\{x, y\}, \mathsf{p}) \longmapsto \mathsf{p}$$

Equality

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A})}{\vdash \mathrm{refl} : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{a}))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{a} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathsf{t} : \mathrm{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) \\ x : \mathrm{Tm}(\mathsf{A}), y : \mathrm{Tm}(\mathrm{Eq}(\mathsf{A}, \mathsf{a}, x)) \vdash \mathsf{P} : \mathrm{Ty} \qquad \vdash \mathsf{p} : \mathrm{Tm}(\mathsf{P}\{\mathsf{a}, \mathrm{refl}\})\end{array}}{\vdash \mathrm{J}(\mathsf{t}; \mathsf{P}, \mathsf{p}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{b}, \mathsf{t}\})}$$

$$\mathrm{J}(\mathrm{refl}, xy.\mathsf{P}\{x, y\}, \mathsf{p}) \longmapsto \mathsf{p}$$

We add conversion premises to constructor rules (not shown in this talk)

Equality

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathbf{Ty} \qquad \vdash \mathsf{a} : \mathbf{Tm}(\mathsf{A}) \qquad \vdash \mathsf{b} : \mathbf{Tm}(\mathsf{A})}{\vdash \mathbf{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) : \mathbf{Ty}} \qquad \frac{\begin{array}{cc}\vdash \mathsf{A} : \mathbf{Ty} & \vdash \mathsf{a} : \mathbf{Tm}(\mathsf{A}) \\ \vdash \mathsf{b} : \mathbf{Tm}(\mathsf{A}) & \mathsf{a} = \mathsf{b}\end{array}}{\vdash \mathbf{refl} : \mathbf{Tm}(\mathbf{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}))}$$

$$\frac{\begin{array}{cccc}\vdash \mathsf{A} : \mathbf{Ty} & \vdash \mathsf{a} : \mathbf{Tm}(\mathsf{A}) & \vdash \mathsf{b} : \mathbf{Tm}(\mathsf{A}) & \vdash \mathsf{t} : \mathbf{Eq}(\mathsf{A}, \mathsf{a}, \mathsf{b}) \\ \multicolumn{3}{c}{x : \mathbf{Tm}(\mathsf{A}), y : \mathbf{Tm}(\mathbf{Eq}(\mathsf{A}, \mathsf{a}, x)) \vdash \mathsf{P} : \mathbf{Ty}} & \vdash \mathsf{p} : \mathbf{Tm}(\mathsf{P}\{\mathsf{a}, \mathbf{refl}\})\end{array}}{\vdash \mathbf{J}(\mathsf{t}; \mathsf{P}, \mathsf{p}) : \mathbf{Tm}(\mathsf{P}\{\mathsf{b}, \mathsf{t}\})}$$

$$\mathbf{J}(\mathbf{refl}, xy.\mathsf{P}\{x, y\}, \mathsf{p}) \longmapsto \mathsf{p}$$

We add conversion premises to constructor rules (not shown in this talk)

Vectors

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat})}{\vdash \mathrm{Vec}(\mathtt{A}, \mathtt{n}) : \mathrm{Ty}} \qquad \frac{\vdash \mathtt{A} : \mathrm{Ty}}{\vdash \mathrm{nil} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, 0))} \qquad \frac{\vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{x} : \mathrm{Tm}(\mathtt{A}) \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}))}{\vdash \mathrm{cons}(\mathtt{n}, \mathtt{x}, \mathtt{l}) : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, \mathrm{S}(\mathtt{n})))}$$

$$\frac{\begin{array}{c} \vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, \mathtt{n})) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, x)) \vdash \mathtt{P} : \mathrm{Ty} \qquad \vdash \mathtt{pnil} : \mathrm{Tm}(\mathtt{P}\{0, \mathrm{nil}\}) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathtt{A}), z : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, x)), w : \mathrm{Tm}(\mathtt{P}\{x, z\}) \vdash \mathtt{pcons} : \mathrm{Tm}(\mathtt{P}\{\mathrm{S}(x), \mathrm{cons}(x, y, z)\}) \end{array}}{\vdash \mathrm{VecRec}(\mathtt{l}; \mathtt{P}, \mathtt{pnil}, \mathtt{pcons}) : \mathrm{Tm}(\mathtt{P}\{\mathtt{n}, \mathtt{l}\})}$$

$$\mathrm{VecRec}(\mathrm{nil}; x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\}) \longmapsto \mathtt{pnil}$$

$$\mathrm{VecRec}(\mathrm{cons}(\mathtt{n}, \mathtt{x}, \mathtt{l}); x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\}) \longmapsto$$

$$\mathtt{pcons}\{\mathtt{n}, \mathtt{x}, \mathtt{l}, \mathrm{VecRec}(\mathtt{l}; x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\})\}$$

Vectors

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat})}{\vdash \mathrm{Vec}(\mathtt{A}, \mathtt{n}) : \mathrm{Ty}} \qquad \frac{\vdash \mathtt{A} : \mathrm{Ty}}{\vdash \mathrm{nil} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, 0))} \qquad \frac{\vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathtt{x} : \mathrm{Tm}(\mathtt{A}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}))}{\vdash \mathrm{cons}(\mathtt{n}, \mathtt{x}, \mathtt{l}) : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, \mathrm{S}(\mathtt{n})))}$$

$$\frac{\begin{array}{c}\vdash \mathtt{A} : \mathrm{Ty} \qquad \vdash \mathtt{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, \mathtt{n})) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, x)) \vdash \mathtt{P} : \mathrm{Ty} \qquad \vdash \mathtt{pnil} : \mathrm{Tm}(\mathtt{P}\{0, \mathrm{nil}\}) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathtt{A}), z : \mathrm{Tm}(\mathrm{Vec}(\mathtt{A}, x)), w : \mathrm{Tm}(\mathtt{P}\{x, z\}) \vdash \mathtt{pcons} : \mathrm{Tm}(\mathtt{P}\{\mathrm{S}(x), \mathrm{cons}(x, y, z)\})\end{array}}{\vdash \mathrm{VecRec}(\mathtt{l}; \mathtt{P}, \mathtt{pnil}, \mathtt{pcons}) : \mathrm{Tm}(\mathtt{P}\{\mathtt{n}, \mathtt{l}\})}$$

$$\mathrm{VecRec}(\mathrm{nil}; x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\}) \longmapsto \mathtt{pnil}$$

$$\mathrm{VecRec}(\mathrm{cons}(\mathtt{n}, \mathtt{x}, \mathtt{l}); x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\}) \longmapsto$$

$$\mathtt{pcons}\{\mathtt{n}, \mathtt{x}, \mathtt{l}, \mathrm{VecRec}(\mathtt{l}; x.\mathtt{P}\{x\}, \mathtt{pnil}, xyzw.\mathtt{pcons}\{x, y, z, w\})\}$$

Vectors

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{n} : \mathrm{Tm}(\mathrm{Nat})}{\vdash \mathrm{Vec}(\mathsf{A}, \mathsf{n}) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{nil} : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A}, 0))}$$

$$\frac{\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{n}' : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathsf{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathsf{x} : \mathrm{Tm}(\mathsf{A}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A})) \qquad \mathsf{n}' = \mathrm{S}(\mathsf{n})}{\vdash \mathrm{cons}(\mathsf{n}, \mathsf{x}, \mathtt{l}) : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A}, \mathsf{n}'))}$$

$$\frac{\begin{array}{c}\vdash \mathsf{A} : \mathrm{Ty} \qquad \vdash \mathsf{n} : \mathrm{Tm}(\mathrm{Nat}) \qquad \vdash \mathtt{l} : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A}, \mathsf{n})) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A}, x)) \vdash \mathsf{P} : \mathrm{Ty} \qquad \vdash \mathsf{pnil} : \mathrm{Tm}(\mathsf{P}\{0, \mathrm{nil}\}) \\ x : \mathrm{Tm}(\mathrm{Nat}), y : \mathrm{Tm}(\mathsf{A}), z : \mathrm{Tm}(\mathrm{Vec}(\mathsf{A}, x)), w : \mathrm{Tm}(\mathsf{P}\{x, z\}) \vdash \mathsf{pcons} : \mathrm{Tm}(\mathsf{P}\{\mathrm{S}(x), \mathrm{cons}(x, y, z)\})\end{array}}{\vdash \mathrm{VecRec}(\mathtt{l}; \mathsf{P}, \mathsf{pnil}, \mathsf{pcons}) : \mathrm{Tm}(\mathsf{P}\{\mathsf{n}, \mathtt{l}\})}$$

$$\mathrm{VecRec}(\mathrm{nil}; x.\mathsf{P}\{x\}, \mathsf{pnil}, xyzw.\mathsf{pcons}\{x, y, z, w\}) \longmapsto \mathsf{pnil}$$

$$\mathrm{VecRec}(\mathrm{cons}(\mathsf{n}, \mathsf{x}, \mathtt{l}); x.\mathsf{P}\{x\}, \mathsf{pnil}, xyzw.\mathsf{pcons}\{x, y, z, w\}) \longmapsto$$

$$\mathsf{pcons}\{\mathsf{n}, \mathsf{x}, \mathtt{l}, \mathrm{VecRec}(\mathtt{l}; x.\mathsf{P}\{x\}, \mathsf{pnil}, xyzw.\mathsf{pcons}\{x, y, z, w\})\}$$

Universes

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{}{\vdash \mathrm{U}(\cdot) : \mathrm{Ty}} \qquad \frac{\vdash a : \mathrm{Tm}(\mathrm{U})}{\vdash \mathrm{El}(a; \cdot) : \mathrm{Ty}}$$

Then, two options:

Universes

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{}{\vdash \mathrm{U}(\cdot) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{a} : \mathrm{Tm}(\mathrm{U})}{\vdash \mathrm{El}(\mathsf{a}; \cdot) : \mathrm{Ty}}$$

Then, two options:

Tarski-style Adds codes for all types

$$\frac{}{\vdash \mathrm{u}(\cdot) : \mathrm{Tm}(\mathrm{U})} \qquad \mathrm{El}(\mathrm{u}; \varepsilon) \longmapsto \mathrm{U}$$

$$\frac{\vdash \mathsf{a} : \mathrm{Tm}(\mathrm{U}) \qquad x : \mathrm{Tm}(\mathrm{El}(\mathsf{a})) \vdash \mathsf{b} : \mathrm{Tm}(\mathrm{U})}{\vdash \pi(\mathsf{a}, \mathsf{b}) : \mathrm{Tm}(\mathrm{U})}$$

$$\mathrm{El}(\pi(\mathsf{a}, x.\mathsf{b}\{x\}); \varepsilon) \longmapsto \Pi(\mathrm{El}(\mathsf{a}; \varepsilon), x.\mathrm{El}(\mathsf{b}\{x\}; \varepsilon))$$

Universes

Extends $\mathbb{T}_{\lambda\Pi}$ with

$$\frac{}{\vdash \mathrm{U}(\cdot) : \mathrm{Ty}} \qquad \frac{\vdash \mathsf{a} : \mathrm{Tm}(\mathrm{U})}{\vdash \mathrm{El}(\mathsf{a}; \cdot) : \mathrm{Ty}}$$

Then, two options:

Tarski-style Adds codes for all types

$$\frac{}{\vdash \mathrm{u}(\cdot) : \mathrm{Tm}(\mathrm{U})} \qquad \mathrm{El}(\mathrm{u}; \varepsilon) \longmapsto \mathrm{U}$$

$$\frac{\vdash \mathsf{a} : \mathrm{Tm}(\mathrm{U}) \qquad x : \mathrm{Tm}(\mathrm{El}(\mathsf{a})) \vdash \mathsf{b} : \mathrm{Tm}(\mathrm{U})}{\vdash \pi(\mathsf{a}, \mathsf{b}) : \mathrm{Tm}(\mathrm{U})}$$

$$\mathrm{El}(\pi(\mathsf{a}, x.\mathsf{b}\{x\}); \varepsilon) \longmapsto \Pi(\mathrm{El}(\mathsf{a}; \varepsilon), x.\mathrm{El}(\mathsf{b}\{x\}; \varepsilon))$$

(Weak) Coquand-style

Adds a code constructor c

$$\frac{\vdash \mathsf{A} : \mathrm{Ty}}{\vdash \mathrm{c}(\mathsf{A}) : \mathrm{Tm}(\mathrm{U})}$$

$$\mathrm{El}(\mathrm{c}(\mathsf{A}); \varepsilon) \longmapsto \mathsf{A}$$

Conclusion

Conclusion

We have given a generic account of bidirectional type for a class of type theories

Conclusion

We have given a generic account of bidirectional type for a class of type theories

Bidirectional system implemented in a prototype, available at

`https://github.com/thiagofelicissimo/BiTTs`

Conclusion

We have given a generic account of bidirectional type for a class of type theories

Bidirectional system implemented in a prototype, available at

`https://github.com/thiagofelicissimo/BiTTs`

Future work Extending class of type theories we can handle by

Conclusion

We have given a generic account of bidirectional type for a class of type theories

Bidirectional system implemented in a prototype, available at

`https://github.com/thiagofelicissimo/BiTTs`

Future work Extending class of type theories we can handle by

1. Going beyond the constructor/destructor separation (do we want this?)

$$\mathrm{Tm}(\mathrm{U}) \longrightarrow \mathrm{Ty}$$

Conclusion

We have given a generic account of bidirectional type for a class of type theories

Bidirectional system implemented in a prototype, available at

`https://github.com/thiagofelicissimo/BiTTs`

Future work Extending class of type theories we can handle by

1. Going beyond the constructor/destructor separation (do we want this?)

$$\mathrm{Tm}(\mathrm{U}) \longrightarrow \mathrm{Ty}$$

2. Going beyond rewriting rules, needed for eta/proof-irrelevance

Thank you for your attention!