cedille

Tooling: Interactive Commands

Interactive Commands

- Backend sends all data in one batch

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands
- Node Commands

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands
- Node Commands
  - “C-i n” – Normalize

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands
- Node Commands
    - “C-i n” – Normalize
    - “C-i h” – Head-normalize

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn't be realistic to include for all nodes
- Hence the need for "Interactive" Commands
- Node Commands
  - "C-i n" – Normalize
  - "C-i h" – Head-normalize
  - "C-i e" – Erase

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands
- Node Commands
  - “C-i n” – Normalize
  - “C-i h” – Head-normalize
  - “C-i e” – Erase
- Beta-Reduction Commands

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn't be realistic to include for all nodes
- Hence the need for "Interactive" Commands
- Node Commands
  - "C-i n" – Normalize
  - "C-i h" – Head-normalize
  - "C-i e" – Erase
- Beta-Reduction Commands
  - "C-i b" – Prompt for expression to open in beta-reduction buffer

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn't be realistic to include for all nodes
- Hence the need for "Interactive" Commands
- Node Commands
  - "C-i n" – Normalize
  - "C-i h" – Head-normalize
  - "C-i e" – Erase
- Beta-Reduction Commands
  - "C-i b" – Prompt for expression to open in beta-reduction buffer
  - "C-i t" – Use selected node's type in beta-reduction buffer

Interactive Commands

- Backend sends all data in one batch
- Sometimes we want more information than was given, but wouldn’t be realistic to include for all nodes
- Hence the need for “Interactive” Commands
- Node Commands
  - “C-i n” – Normalize
  - “C-i h” – Head-normalize
  - “C-i e” – Erase
- Beta-Reduction Commands
  - “C-i b” – Prompt for expression to open in beta-reduction buffer
  - “C-i t” – Use selected node’s type in beta-reduction buffer

Interactive Commands: Beta-Reduction Buffer

- Navigation commands (“f”, “b”, “n”, “p”, etc...)

Interactive Commands: Beta-Reduction Buffer

- Navigation commands (“f”, “b”, “n”, “p”, etc...)
- “C-i n” – Normalize and replace the selected node
- “C-i h” – Head-normalize and replace the selected node

Interactive Commands: Beta-Reduction Buffer

- Navigation commands (“f”, “b”, “n”, “p”, etc...)
- “C-i n” – Normalize and replace the selected node
- “C-i h” – Head-normalize and replace the selected node
- “C-i r” – Prompt for proof of equation, and rewrite the selected node with it

Interactive Commands: Beta-Reduction Buffer

- Navigation commands (“f”, “b”, “n”, “p”, etc...)
- “C-i n” – Normalize and replace the selected node
- “C-i h” – Head-normalize and replace the selected node
- “C-i r” – Prompt for proof of equation, and rewrite the selected node with it
- “C-i p” – Reconstruct proof from steps taken

Interactive Commands: Beta-Reduction Buffer

- Navigation commands (“f”, “b”, “n”, “p”, etc...)
- “C-i n” – Normalize and replace the selected node
- “C-i h” – Head-normalize and replace the selected node
- “C-i r” – Prompt for proof of equation, and rewrite the selected node with it
- “C-i p” – Reconstruct proof from steps taken

Interactive Commands: One More

- “E” – Elaborate to Cedille Core

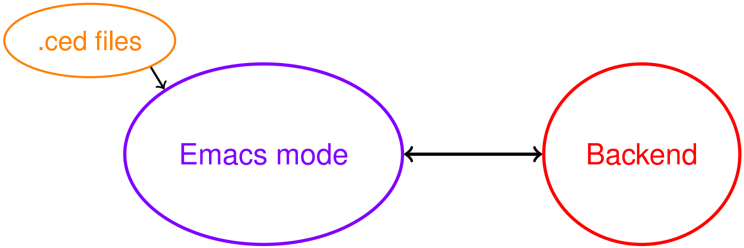
$\rightsquigarrow \texttt{cedille}_{\texttt{core}}$

Elaboration to Cedille Core

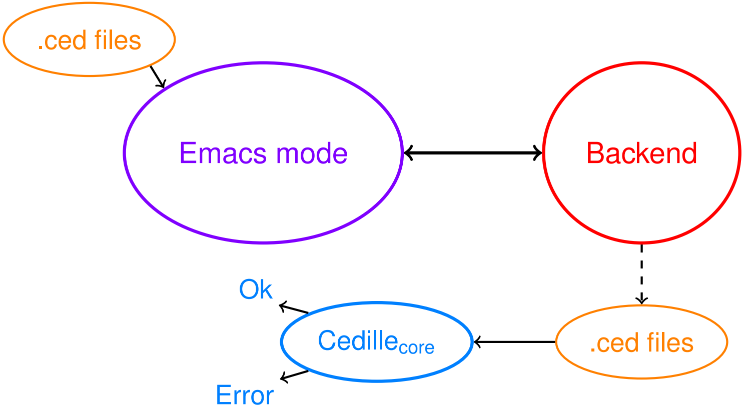
What is Cedille Core?

- Independent implementation of CDLE
- Full annotations required
    - No type inference
    - No bidirectionality
- More verbose, much easier to check
- Fewer than 1000 lines of Haskell code

Architecture of Cedille



Architecture of Cedille



Cedille vs. Cedille Core

$(\lambda\ x.\ t)$

$(\lambda\ x : T.\ t)$

$[\ t_1,\ t_2\ ]$

δ

Type Checking

$(\rho\ e\ @\ x.\ T_x\ -\ t)$

φ β

$(\beta\text{<}u_T\text{>}\{u_t\})$

Scope Shadowing

ε

$[\ t_1,\ t_2\ @\ x.\ T_x\ ]$

$(\beta\{u_t\})$

Type Synthesis

ς

χ

$(\rho\text{<}3\ 4\ 6\text{>}\ e\ -\ t)$

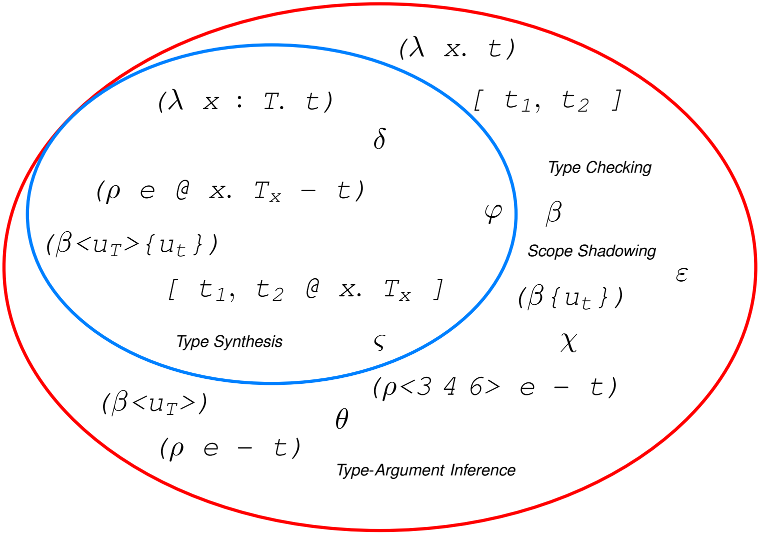
$(\beta\text{<}u_T\text{>})$

θ

$(\rho\ e\ -\ t)$

Type-Argument Inference

Cedille vs. Cedille Core



Cedille $\rightsquigarrow$ Cedille Core: λ-terms

$\Gamma = X : \star$

$$\begin{array}{c} X \to X \\ \Downarrow \\ \lambda\ x.\ x \\ \rightsquigarrow \\ \lambda\ x : X.\ x \end{array}$$

Cedille $\rightsquigarrow$ Cedille Core: ι-pairs

$$\Gamma = A : \star,\ B : (A \to \star),\ a : A,\ b : B\ a$$

$$\iota\ x : A.\ B\ x$$

$$\Downarrow$$

$$[\ a,\ b\ ]$$

$$\rightsquigarrow$$

$$[\ a,\ b\ @\ x.\ B\ x\ ]$$

Cedille $\rightsquigarrow$ Cedille Core: β-terms

Γ = T : $\star$, t : T

$$\{\ t \simeq t\ \}$$

$$\Downarrow$$

$$\beta$$

$$\rightsquigarrow$$

$$\beta\texttt{<}t\texttt{>}\{\lambda\ x.\ x\}$$

Cedille $\leadsto$ Cedille Core: δ-contradictions

Γ = *n : Nat*

$$\{\ suc\ n \simeq zero\ \}$$
$$\leadsto$$
$$\{\ suc\ n\ ff\ (\lambda\ \_.\ tt) \simeq zero\ ff\ (\lambda\ \_.\ tt)\ \}$$

Cedille $\rightsquigarrow$ Cedille Core: ρ-terms

$$\frac{\Gamma = m : Nat,\ n : Nat,\ e : \{\ suc\ m \simeq n\ \}}{}$$

$$\{\ suc\ (suc\ m) \simeq suc\ n\ \}$$
$$\Downarrow$$
$$\rho\ e\ -\ \beta$$
$$\rightsquigarrow$$
$$\rho\ e\ @\ x.\ \{\ suc\ x \simeq suc\ n\ \}\ -\ \beta{<}suc\ n{>}\{\lambda\ x.\ x\}$$

Cedille $\rightsquigarrow$ Cedille Core: ρ-terms

$$\frac{\Gamma = A : \star,\ B : (A \to \star),\ a : A,\ e : \{\ a \simeq \lambda\ x.\ x\ x\}}{}$$

$$B\ a \to B\ a$$

$$\Downarrow$$

$$\rho\ e\ -\ \lambda\ b.\ b$$

$$\rightsquigarrow$$

$$\rho\ e\ @\ x.\ (B\ x \to B\ x)\ -\ \lambda\ b : B\ \underbrace{(\lambda\ x.\ x\ x)}_{\phi\ e\ -\ a\ \{\lambda\ x.\ x\ x\}}.\ b$$

Cedille $\rightsquigarrow$ Cedille Core: ρ-terms

$$\Gamma = A : \star,\ B : (\Pi\ a : A.\ \Pi\ a' : A.\ \{\ a \simeq a'\ \} \to \star),\ a : A,\ a' : A,\ e : \{ a \simeq a' \},\ p : \{ a \simeq \lambda\ x.\ x\ x \}$$

$$B\ a\ a'\ e \to B\ a\ a'\ e$$
$$\Downarrow$$
$$\rho\ p\ -\ \lambda\ b.\ b$$
$$\rightsquigarrow$$
$$\rho\ e\ @\ x.\ (B\ x\ a'\ e \to B\ x\ a'\ e)\ -$$
$$\lambda\ b : B\ (\phi\ p\ -\ a\ \{\lambda\ x.\ x\ x\})\ a'\ \underbrace{e\ .}_{\rho\ \varsigma\ p\ @\ x.\ \{\ x \simeq a'\ \}\ -\ e}\ b$$

Cedille $\rightsquigarrow$ Cedille Core: Type Inference

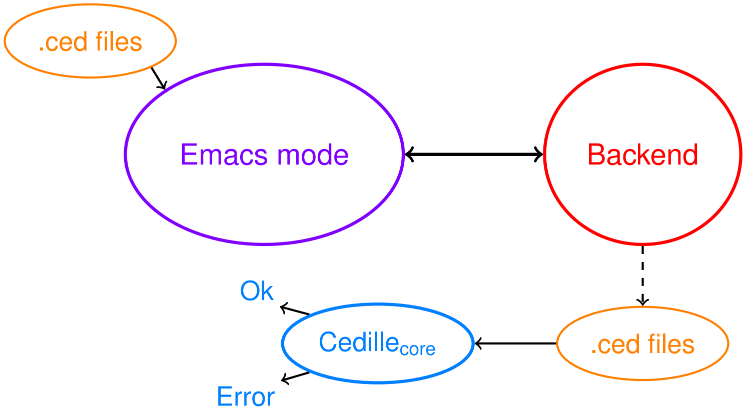
$$\Gamma = \mathit{id} : (\forall\ X : \star.\ X \to X)$$

$$\mathit{id}\ \mathit{zero}$$

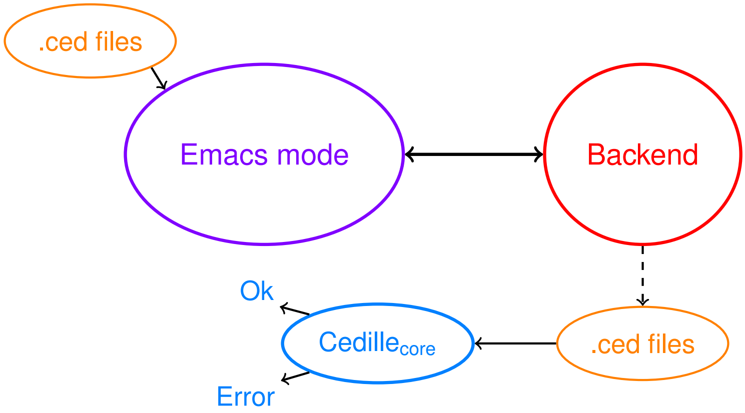
$$\rightsquigarrow$$

$$\mathit{id} \cdot \mathit{Nat}\ \mathit{zero}$$

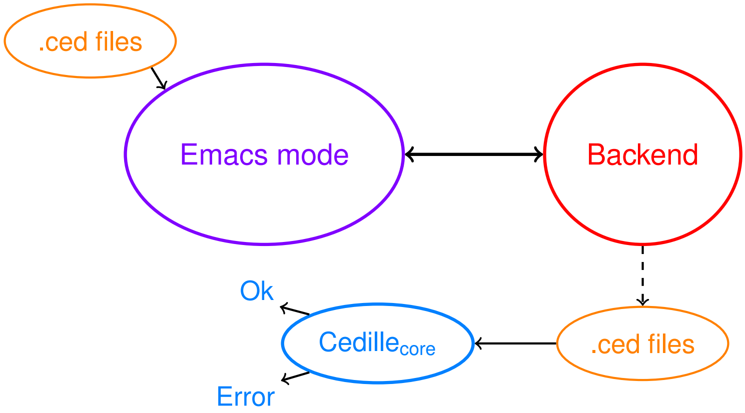
Architecture of Cedille



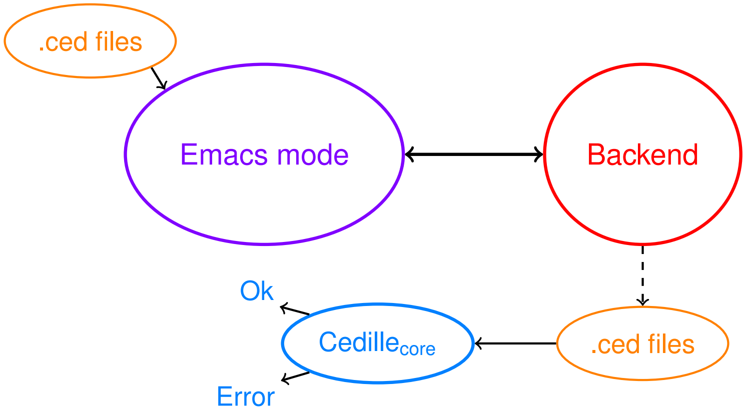
Architecture of Cedille



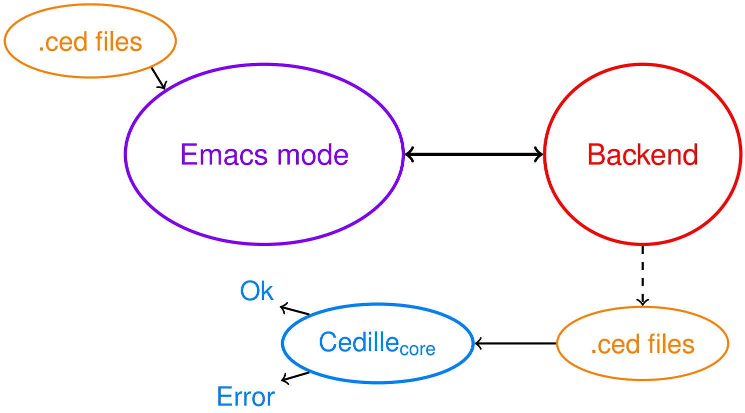
Architecture of Cedille



Architecture of Cedille



Architecture of Cedille


.ced files
Emacs mode
Backend
.ced files
Cedille$_{core}$
Ok
Error