

Terms

$\lambda\text{LF}\bigcirc$ adds staged computation of Davies's $\lambda\bigcirc[1]$ to $\lambda\text{LF}[2]$

$t ::=$

x
 $\lambda x : T. t$
 $t \ t$
next (t)
prev (t)
id(t)
idpeel($t, (x) t$)

(terms)

$T ::=$

X
Eq $_T$
 $\Pi x : T. T$
 $T \ t$
 $\bigcirc T$

(types)

2019-11-23

Staged Dependent Lambda Calculus with Equality Types

└ Introduction to the system

└ Terms

hoge hoge

Terms

$\lambda\text{LF}\bigcirc$ adds staged computation of Davies's $\lambda\bigcirc[1]$ to $\lambda\text{LF}[2]$

$t ::=$

x
 $\lambda x : T. t$
 $t \ t$
next (t)
prev (t)
id(t)
idpeel($t, (x) t$)

(terms)

$T ::=$

X
Eq $_T$
 $\Pi x : T. T$
 $T \ t$
 $\bigcirc T$

(types)

2019-11-23

Staged Dependent Lambda Calculus with Equality Types

└ Equality of terms, types and kinds

Equality of terms, types and kinds

Equality of terms, types and kinds

Equality of terms concerning stages

Do not allow equivalence of terms via code embedding inside next.

$$\frac{\Gamma \vdash^{(n)} t : T \quad \Gamma \vdash^{(n)} T :: * \quad m \leq 1}{\Gamma \vdash^{(n)} \text{prev (next (t))} \equiv_m t : T}$$

(Q-○BETA)

Disallow

$\Gamma \vdash^{(n)} \text{next (next (3 + prev (next (5))))} \equiv_0 \text{next (next (3 + 5))}$

Allow

$\Gamma \vdash^{(n)} \text{next (3 + prev (next (5)))} \equiv_0 \text{next (3 + 5)}$

2019-11-23

- Staged Dependent Lambda Calculus with Equality Types
- └ Equality of terms, types and kinds
- └ Equality of terms concerning stages

Equality of terms concerning stages

Do not allow equivalence of terms via code embedding inside next.

$$\frac{\Gamma \vdash^{(n)} t : T \quad \Gamma \vdash^{(n)} T :: * \quad m \leq 1}{\Gamma \vdash^{(n)} \text{prev (next (t))} \equiv_m t : T} \quad (\text{Q-}\bigcirc\text{BETA})$$

Disallow

$\Gamma \vdash^{(n)} \text{next (next (3 + prev (next (5))))} \equiv_0 \text{next (next (3 + 5))}$

Allow

$\Gamma \vdash^{(n)} \text{next (3 + prev (next (5)))} \equiv_0 \text{next (3 + 5)}$

Compare terms inside types at the current equivalence depth.

Evaluation of terms

Defining values

$v ::=$
 $\lambda x:T. t$
 $\text{next } (v)$ (values)
 $\text{id}(v)$
 $\text{idpeel}(v, (x) v)$

$t' ::=$
 $\llbracket t \rrbracket_d \ (d \in \mathbb{N})$
 (indexed terms)

 $v' ::=$
 $\llbracket v \rrbracket_d \ (d \in \mathbb{N})$
 (indexed values)

Defining values	
$v ::=$ $\lambda x:T. t$ $\text{next } (v)$ $\text{id}(v)$ $\text{idpeel}(v, (x) v)$	$t' ::=$ $\llbracket t \rrbracket_d \ (d \in \mathbb{N})$ (indexed terms) $v' ::=$ $\llbracket v \rrbracket_d \ (d \in \mathbb{N})$ (indexed values)

Meaning of the index

$$\frac{\llbracket t \rrbracket_{d+1} \rightarrow \llbracket t' \rrbracket_{d+1} \quad d \leq 1}{\llbracket \text{next } (t) \rrbracket_d \rightarrow \llbracket \text{next } (t') \rrbracket_d} \quad (\text{E-NEXT})$$

$$\frac{\llbracket t \rrbracket_{d+1} \rightarrow \llbracket t' \rrbracket_{d+1} \quad d \leq 1}{\llbracket \text{prev } (t) \rrbracket_d \rightarrow \llbracket \text{prev } (t') \rrbracket_d} \quad (\text{E-PREV})$$

Rules where index is important

$$\frac{\llbracket t \rrbracket_d \rightarrow \llbracket t' \rrbracket_d \quad d \leq 1}{\llbracket \text{prev (next (t))} \rrbracket_d \rightarrow \llbracket t' \rrbracket_d} \quad (\text{E-STAGEBETA})$$

$$\frac{\llbracket t \rrbracket_d \rightarrow \llbracket t' \rrbracket_d \quad d \leq 1}{\llbracket \text{next (prev (t))} \rrbracket_d \rightarrow \llbracket t' \rrbracket_d} \quad (\text{E-STAGEETA})$$

2019-11-23

Staged Dependent Lambda Calculus with Equality Types

└ Evaluation of terms

└ Rules where index is important

Rules where index is important

$$\frac{\llbracket t \rrbracket_d \rightarrow \llbracket t' \rrbracket_d \quad d \leq 1}{\llbracket \text{prev (next (t))} \rrbracket_d \rightarrow \llbracket t' \rrbracket_d} \quad (\text{E-STAGEBETA})$$
$$\frac{\llbracket t \rrbracket_d \rightarrow \llbracket t' \rrbracket_d \quad d \leq 1}{\llbracket \text{next (prev (t))} \rrbracket_d \rightarrow \llbracket t' \rrbracket_d} \quad (\text{E-STAGEETA})$$

Auxiliary rules

2019-11-23

Staged Dependent Lambda Calculus with Equality Types

└ Evaluation of terms

└ Auxiliary rules

Future work

- TODO

2019-11-23

Staged Dependent Lambda Calculus with Equality Types

- └ Evaluation of terms
 - └ Future work

Future work

- TODO

References

References



Rowan Davies and Frank Pfenning. "A modal analysis of staged computation". In: *Conference Record of the Annual ACM Symposium on Principles of Programming Languages*. Vol. Conference. 3. May 1996, pp. 258–270. DOI: 10.1145/382780.382785. URL: <http://portal.acm.org/citation.cfm?doid=382780.382785>.



Benjamin C Pierce. *Advanced Topics in Types and Programming Languages*. Cambridge, Mass.: The Mit Press, 2019. DOI: [10.7551/mitpress/1104.001.0001](https://doi.org/10.7551/mitpress/1104.001.0001).

Staged Dependent Lambda Calculus with Equality Types

Evaluation of terms

References

Conclusions

-  **Rowan Davies** and **Frank Flennig**, "A modal analysis of staged computation." In: *Conference Record of the Annual ACM Symposium on Principles of Programming Languages*. Vol. Conference. 3. May 1996, pp. 258–270. doi: [10.1145/382780.382785](https://doi.org/10.1145/382780.382785). URL: <http://portal.acm.org/citation.cfm?id=382780.382785>.
-  **Benjamin C Pierce**, *Advanced Topics in Types and Programming Languages*. Cambridge, Mass: The MIT Press, 2019. pp. 0.0. doi: [10.7551/9780262089104_0001](https://doi.org/10.7551/9780262089104_0001).