Statics and Dynamics: E ($L^{\text{num,str}}$)

Yu Zhang

http://staff.ustc.edu.cn/~yuzhang/tpl/
yuzhang@ustc.edu.cn

References

- PFPL
  - Chapters: E ($L^{\text{num,str}}$)
    - 4 Statics, 5 Dynamics
    - 6 Type Safety
    - 7 Evaluation Dynamics
- TAPL
- The algebra (and calculus!) of algebraic data types

Outline

- Syntax
- Statics
- Dynamics
- Type Safety
- Evaluation Dynamics

Syntax

- Abstract syntax vs. concrete syntax

Typ	τ	::=	num	num	numbers
			str	str	strings
Exp	e	::=	x	x	variable
			num[n]	n	numeral
			str[s]	"s"	literal
			plus(e_1;e_2)	$e_1 + e_2$	addition
			times(e_1;e_2)	$e_1 * e_2$	multiplication
			cat(e_1;e_2)	e_1 ^ e_2	concatenation
			len(e)	$\lvert e \rvert$	length
			let(e_1;x.e_2)	let x be e_1 in e_2	definition

Semantics

- Semantics: judgements, rules
- **Statics** (Type System)
  - Generic hypothetical judgments $\vec{x} \mid \Gamma \vdash e : \tau$,
    - variables
    - typing context: $x : \tau$
  - Rules

$$\frac{}{\Gamma \vdash \texttt{num}[n] : \texttt{num}} \qquad \frac{}{\Gamma, x : \tau \vdash x : \tau}$$

$$\frac{}{\Gamma \vdash \texttt{str}[s] : \texttt{str}}$$

Introduction form of num type (引入num类型的值)

$$\frac{\Gamma \vdash e_1 : \texttt{num} \quad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{plus}(e_1; e_2) : \texttt{num}} \qquad \frac{\Gamma \vdash e_1 : \texttt{num} \quad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{times}(e_1; e_2) : \texttt{num}}$$

Elimination form of num type (消去) (操作num类型的值)

$$\frac{\Gamma \vdash e_1 : \texttt{str} \quad \Gamma \vdash e_2 : \texttt{str}}{\Gamma \vdash \texttt{cat}(e_1; e_2) : \texttt{str}} \qquad \frac{\Gamma \vdash e : \texttt{str}}{\Gamma \vdash \texttt{len}(e) : \texttt{num}}$$

Semantics

- Type system

variable x is not declared in Γ

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma, x : \tau_1 \vdash e_2 : \tau_2}{\Gamma \vdash \texttt{let}(e_1; x.e_2) : \tau_2}$$

  - Unicity(唯一性): $\Gamma \vdash e : \tau$, at most one type for each exp
  - Inversion for typing(定型反转)
    - If $\Gamma \vdash e : \tau$, $e = \text{plus}(e_1; e_2)$, then $\tau = \text{num}$, $\Gamma \vdash e_1 : \text{num}$, and $\Gamma \vdash$ $\dfrac{\Gamma \vdash e_1 : \texttt{num} \quad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{plus}(e_1; e_2) : \texttt{num}}$
  - Weakening(弱化)
    - If $\Gamma \vdash e' : \tau'$, then $\Gamma, x : \tau \vdash e' : \tau'$ for any $x \notin dom(\Gamma)$ and any type τ
  - Substitution(代换)
    - If $\Gamma, x : \tau \vdash e' : \tau'$ and $\Gamma \vdash e : \tau$, then $\Gamma \vdash [e/x]e' : \tau'$
  - Decomposition(分解)
    - If $\Gamma \vdash [e/x]e' : \tau'$, then for each type τ such that $\Gamma \vdash e : \tau$, we have $\Gamma, x : \tau \vdash e' : \tau'$

Dynamics: Transition Systems

- Structural dynamics (step-by-step, small step)
  - Judgements
    - s state，断言s是转换系统的一个状态。
    - s final，对s state，断言s是一个终结状态。
    - s initial，对s state，断言s是一个初始状态。
    - $s \longmapsto s'$，对s state，s' state，断言状态s可以转换到状态 s'。
    - $s \longmapsto^* s'$

$$\frac{}{s \longmapsto^* s} \quad (5.1a)$$

$$\frac{s \longmapsto s' \quad s' \longmapsto^* s''}{s \longmapsto^* s''} \quad (5.1b)$$

Dynamics: Transition Systems

- Values: e val $\quad \overline{\mathtt{num}[n]\ \mathtt{val}} \quad \overline{\mathtt{str}[s]\ \mathtt{val}}$
- Transition judgment e $\longmapsto e'$

$$\frac{n_1 + n_2 = n}{\mathtt{plus}(\mathtt{num}[n_1];\mathtt{num}[n_2]) \longmapsto \mathtt{num}[n]} \qquad \frac{s_1 \hat{} s_2 = s\ \mathtt{str}}{\mathtt{cat}(\mathtt{str}[s_1];\mathtt{str}[s_2]) \longmapsto \mathtt{str}[s]}$$

$$\frac{e_1 \longmapsto e_1'}{\mathtt{plus}(e_1;e_2) \longmapsto \mathtt{plus}(e_1';e_2)} \qquad \frac{e_1 \longmapsto e_1'}{\mathtt{cat}(e_1;e_2) \longmapsto \mathtt{cat}(e_1';e_2)}$$

$$\frac{e_1\ \mathtt{val} \quad e_2 \longmapsto e_2'}{\mathtt{plus}(e_1;e_2) \longmapsto \mathtt{plus}(e_1;e_2')} \qquad \frac{e_1\ \mathtt{val} \quad e_2 \longmapsto e_2'}{\mathtt{cat}(e_1;e_2) \longmapsto \mathtt{cat}(e_1;e_2')}$$

$$\frac{e_1 \longmapsto e_1'}{\mathtt{let}(e_1;x.e_2) \longmapsto \mathtt{let}(e_1';x.e_2)} \qquad \frac{[e_1\ \mathtt{val}]}{\mathtt{let}(e_1;x.e_2) \longmapsto [e_1/x]e_2}$$

Search transitions
决定指令执行次序
Instruction transitions
基础的求值步

Included for call-by-value, Omitted for call-by-name

Dynamics: Transition Systems

- Examples

$$\begin{aligned}&\mathtt{let}(\mathtt{plus}(\mathtt{num}[1];\mathtt{num}[2]);x.\mathtt{plus}(\mathtt{plus}(x;\mathtt{num}[3]);\mathtt{num}[4]))\\ &\longmapsto \mathtt{let}(\mathtt{num}[3];x.\mathtt{plus}(\mathtt{plus}(x;\mathtt{num}[3]);\mathtt{num}[4]))\\ &\longmapsto \mathtt{plus}(\mathtt{plus}(\mathtt{num}[3];\mathtt{num}[3]);\mathtt{num}[4])\\ &\longmapsto \mathtt{plus}(\mathtt{num}[6];\mathtt{num}[4])\\ &\longmapsto \mathtt{num}[10]\end{aligned}$$

- Finality of values
  - For no e, we have both e val and $e \longmapsto e'$
- Determinacy
  - If $e \longmapsto e'$ and $e \longmapsto e''$, then e' and e'' are α-equivalent

Contextual Dynamics

- Contextual Dynamics
  - Judgement $\mathcal{E}$ ectxt
    - The position of the next instruction step is specified by a "*hole*"
  - Judgement $e' = \mathcal{E}\{e\}$
    - filling the hole in the evaluation context $\mathcal{E}$ with the exp. e

$$\frac{}{\circ\ \mathtt{ectxt}} \qquad \frac{\mathcal{E}_1\ \mathtt{ectxt}}{\mathtt{plus}(\mathcal{E}_1;e_2)\ \mathtt{ectxt}} \qquad \frac{e_1\ \mathtt{val} \quad \mathcal{E}_2\ \mathtt{ectxt}}{\mathtt{plus}(e_1;\mathcal{E}_2)\ \mathtt{ectxt}}$$

$$\frac{}{e = \circ\{e\}} \qquad \frac{e_1 = \mathcal{E}_1\{e\}}{\mathtt{plus}(e_1;e_2) = \mathtt{plus}(\mathcal{E}_1;e_2)\{e\}} \qquad \frac{e_1\ \mathtt{val} \quad e_2 = \mathcal{E}_2\{e\}}{\mathtt{plus}(e_1;e_2) = \mathtt{plus}(e_1;\mathcal{E}_2)\{e\}}$$

Contextual Dynamics

$$\frac{e = \mathcal{E}\{e_0\} \quad e_0 \to e_0' \quad e' = \mathcal{E}\{e_0'\}}{e \longmapsto e'}$$

Type Safety

- A language is *safe*/*sound* if satisfying two theorems
  - Preservation
    - If $e:\tau$ and $e \longmapsto e'$, then $e':\tau$
  - Progress
    - If $e:\tau$, then either e val, or there exists e' such that $e \longmapsto e'$

Prove the theorem by *induction* on the *typing rules*

interpreter.ml

Run-time Errors

- Zero divisor

$$\frac{e_1 : \mathtt{num} \quad e_2 : \mathtt{num}}{\mathtt{div}(e_1;e_2) : \mathtt{num}}$$

  - Well-typed, yet stuck!
  - How to correct
    - Enhance the type system ☹
    - Add dynamic checks ☺

$$\frac{e_1\ \mathtt{val}}{\mathtt{div}(e_1;\mathtt{num}[0])\ \mathtt{err}} \qquad \frac{e_1\ \mathtt{err}}{\mathtt{div}(e_1;e_2)\ \mathtt{err}} \qquad \frac{e_1\ \mathtt{val} \quad e_2\ \mathtt{err}}{\mathtt{div}(e_1;e_2)\ \mathtt{err}}$$

Statics $\quad \Gamma \vdash \mathtt{error} : \tau$

Dynamics $\quad \overline{\mathtt{error}\ \mathtt{err}}$

Progress with error

If $e:\tau$, then either e val, or e err, or there exists e' such that $e \longmapsto e'$

Evaluation Dynamics

- Evaluation Dynamics (big step)
  - Evaluation judgment: $e \Downarrow v$
  - Rules

$$\frac{}{\text{num}[n] \Downarrow \text{num}[n]} \qquad \frac{}{\text{str}[s] \Downarrow \text{str}[s]}$$

$$\frac{e_1 \Downarrow \text{num}[n_1] \quad e_2 \Downarrow \text{num}[n_2] \quad n_1 + n_2 \; is \; n \text{ nat}}{\text{plus}(e_1; e_2) \Downarrow \text{num}[n]} \qquad \frac{e_1 \Downarrow \text{str}[s_1] \quad e_2 \Downarrow \text{str}[s_2] \quad s_1 \wedge s_2 = s \text{ str}}{\text{cat}(e_1; e_2) \Downarrow \text{str}[s]}$$

$$\frac{e \Downarrow \text{str}[s] \quad |s| = n \text{ nat}}{\text{len}(e) \Downarrow \text{num}[n]} \qquad \frac{[e_1/x]e_2 \Downarrow v_2}{\text{let}(e_1; x.e_2) \Downarrow v_2} \text{ by-name interpretation}$$

$$\frac{e_1 \Downarrow v_1 \quad [v_1/x]e_2 \Downarrow v_2}{\text{let}(e_1; x.e_2) \Downarrow v_2} \text{ by-value interpretation}$$

  - If $e \Downarrow v$, then v val

Structural vs. Evaluation Dynamics

- For all closed expressions (闭式) e and values v, $e \mapsto^* v$ iff $e \Downarrow v$.
  - If $e \Downarrow v$, then $e \mapsto^* v$
  - If $e \mapsto e'$ and $e' \Downarrow v$, then $e \Downarrow v$
- Type safety
  - Cannot be proved if using evaluation dynamics
    - Have an analog of the preservation property, but no clear analog of the progress property
    - **Progress**: If $e : \tau$, then $e \Downarrow v$ for some v requires that every expression evaluate to a value! If E were extended to admit non-terminating expressions or operations that may cause an error, the progress would fail.

Cost Dynamics

- *Time complexity* for programs
  - Structural dynamics: the number of steps
  - Evaluation dynamics: does not provide
- Cost dynamics
  - Evaluation judgment $e \Downarrow^k v$
  - Rules

$$\frac{}{\text{num}[n] \Downarrow^0 \text{num}[n]} \qquad \frac{e_1 \Downarrow^{k_1} \text{num}[n_1] \quad e_2 \Downarrow^{k_2} \text{num}[n_2]}{\text{plus}(e_1; e_2) \Downarrow^{k_1+k_2+1} \text{num}[n_1 + n_2]}$$

$$\frac{}{\text{str}[s] \Downarrow^0 \text{str}[s]} \qquad \frac{e_1 \Downarrow^{k_1} s_1 \quad e_2 \Downarrow^{k_2} s_1}{\text{cat}(e_1; e_2) \Downarrow^{k_1+k_2+1} \text{str}[s_1 \wedge s_2]}$$

by-name interpretation

$$\frac{[e_1/x]e_2 \Downarrow^{k_2} v_2}{\text{let}(e_1; x.e_2) \Downarrow^{k_2+1} v_2}$$

by-value interpretation

$$\frac{e_1 \Downarrow^{k_1} v_1 \quad [v_1/x]e_2 \Downarrow^{k_2} v_2}{\text{let}(e_1; x.e_2) \Downarrow^{k_1+k_2+1} v_2}$$

  - For all closed expressions e and values v, $e \Downarrow^k v$ iff $e \mapsto^k v$.