中国科学技术大学
University of Science and Technology of China

Control Flow

《程序语言设计和程序分析》

张昱

yuzhang@ustc.edu.cn

中国科学技术大学

计算机科学与技术学院

References

PFPL

- **Chapters**
  - Chapter 28 Control Stacks
  - Chapter 29 Exception
  - Chapter 30 Continuations

Why Introducing Abstract Machine

Structural dynamics

- **Convenient for proving properties of languages, e.g. safety**
- **Less convenient as a guide for implementation**
  - **Search transition**：does not spell out how to find,
    - Traverse to fine e_1 and then reconstruct the expression plus($e_1'; e_2$)

$$\frac{n_1 + n_2 = n \text{ nat}}{\texttt{plus}(\texttt{num}[n_1]; \texttt{num}[n_2]) \mapsto \texttt{num}[n]}$$

$$\frac{e_1 \mapsto e_1'}{\texttt{plus}(e_1; e_2) \mapsto \texttt{plus}(e_1'; e_2)} \qquad \frac{e_1 \text{ val} \quad e_2 \mapsto e_2'}{\texttt{plus}(e_1; e_2) \mapsto \texttt{plus}(e_1; e_2')}$$

➔ **Make the process explicit by**

introducing control stack and define abstract machine

Control Stack

在实现时，会更愿意用某种机制来记录“当前位于表达式哪儿”，
这样在执行简化后可以从所记录的位置“恢复”。
——可以通过“控制栈”(跟踪记录指令步的上下文)来达到。

利用控制栈，转换规则无须任何前提，即每一规则都是公理！

在实现时，就无须对表达式进行遍历或重构。

真实机器和抽象机

- 动态语义说明：
  如果对实现所需的机制(如控制栈等)揭示得越多，则越接近真实机器。
- 抽象机的定义：从语言的结构语义开始，逐步引入实现所需的多种机制，向汇编级描述发展，所得到的描述即是特定的抽象机。
- 动态语义描述越接近真实机器，则它越“具体”，而越不“抽象”。
- 抽象和具体之间没有明显的分界线，而与实现细节被揭示的程度有关。

Abstract Machine K for PCF

- State s: control stack k and closed expression e
  - ***evaluation*** state $k \triangleright e$, evaluate e on stack k
  - ***return*** state $k \triangleleft e$, e val, evaluate k on closed value e
- Control stack
  - Record the current location of evaluation, and context
  - Explicitly record pending computations

$$\frac{}{\varepsilon \text{ stack}} \qquad \frac{f \text{ frame} \qquad k \text{ stack}}{k; f \text{ stack}} \tag{28.1}$$

  - Frames of the K, "–"represents current location of evaluation

$$\frac{}{\text{s}(-) \text{ frame}} \qquad \frac{}{\text{ifz}\{e_0; x.\, e_1\}(-) \text{ frame}} \qquad \frac{}{\text{ap}\,(-; e_2) \text{ frame}} \tag{28.2}$$

Dynamics of K

- **Eager semantics for successor**

$$\frac{}{k \rhd z \mapsto k \lhd z} \qquad \frac{}{k \rhd \mathrm{s}(e) \mapsto k; \mathrm{s}(-) \rhd e} \qquad \frac{}{k; \mathrm{s}(-) \lhd e \mapsto k \lhd \mathrm{s}(e)} \tag{28.3}$$

- **Rules for case analysis**

$$\frac{}{k \rhd \mathrm{ifz}\{e_0; x.\, e_1\}(e) \mapsto k; \mathrm{ifz}\{e_0; x.\, e_1\}(-) \rhd e}$$

$$\frac{}{k; \mathrm{ifz}\{e_0; x.\, e_1\}(-) \lhd z \mapsto k \rhd e_0} \tag{28.4}$$

$$\frac{}{k; \mathrm{ifz}\{e_0; x.\, e_1\}(-) \lhd \mathrm{s}(e) \mapsto k \rhd [e/x]e_1}$$

- **Functions evaluated by-name, general recursion**

$$\frac{}{k \rhd \mathrm{lam}\{\tau\}(x.\, e) \mapsto k \lhd \mathrm{lam}\{\tau\}(x.\, e)} \qquad \frac{}{k \rhd \mathrm{ap}(e_1; e_2) \mapsto k; \mathrm{ap}(-; e_2) \rhd e_1}$$

$$\frac{}{k; \mathrm{ap}(-; e_2) \lhd \mathrm{lam}\{\tau\}(x.\, e) \mapsto k \rhd [e_2/x]\, e}$$

$$\frac{}{k \rhd \mathrm{fix}\{\tau\}(x.\, e) \mapsto k \rhd [\mathrm{fix}\{\tau\}(x.\, e)/x]\, e}$$

Dynamics of K

Initial and final state of K

$$\frac{}{\varepsilon \rhd e \text{ initial}} \qquad \frac{e \text{ val}}{\varepsilon \lhd e \text{ final}} \qquad (28.6)$$

Safety

- **New typing judgement $k \lhd: \tau$, k expects a value of type τ**

$$\frac{}{\varepsilon \lhd: \tau} \qquad \frac{k \lhd: \tau' \quad f : \tau \rightsquigarrow \tau'}{k; f \lhd: \tau} \qquad (28.6)$$

- Auxiliary judgement $f : \tau \rightsquigarrow \tau'$: frame f transforms a value of type τ to a value of type τ'

Safety of K

- **Rules for auxiliary judgement** $f : \tau \rightsquigarrow \tau'$

$$\frac{}{\mathrm{s}(-) : \mathrm{nat} \rightsquigarrow \mathrm{nat}} \qquad \frac{e_0 : \tau \quad x : \mathrm{nat} \vdash e_1 : \tau}{\mathrm{ifz}\{e_0; x.\, e_1\}(-) : \mathrm{nat} \rightsquigarrow \tau} \qquad \frac{e_2 : \tau}{\mathrm{ap}(-; e_2) : \mathrm{parr}(\tau_2; \tau) \rightsquigarrow \tau} \tag{28.8}$$

- **States of PCF are well-formed if** k **and** e **match**

$$\frac{k \lhd : \tau \quad e : \tau}{k \rhd e \ \mathrm{ok}} \qquad \frac{k \lhd : \tau \quad e : \tau \quad e \ \mathrm{val}}{k \lhd e \ \mathrm{ok}} \tag{28.9}$$

- **Safety**
  - **If** $s \ \mathrm{ok}$ **and** $s \mapsto s'$**, then** $s' \ \mathrm{ok}$
  - **If** $s \ \mathrm{ok}$**, then either** $s \ \mathrm{final}$ **or there exists** s'**such that** $s \mapsto s'$

Correctness of K

Does K implement PCF correctly?

The answer can be derived from the following facts

- **Completeness**

  If $e \mapsto^* e'$, where e' val then $\varepsilon \rhd e \mapsto^* \varepsilon \lhd e'$

  - Proof by induction on the definition of multi-step transition
    - **If e val, then $\varepsilon \rhd e \mapsto^* \varepsilon \lhd e$ (induction on the structure of e)**
    - **If $e \mapsto^* e'$, then, for every v val, if $\varepsilon \rhd e' \mapsto^* \varepsilon \lhd v$, then $\varepsilon \rhd e \mapsto^* \varepsilon \lhd v$**

- **Soundness**

  If $\varepsilon \rhd e \mapsto^* \varepsilon \lhd e'$, then $e \mapsto^* e'$ with e' val

中国科学技术大学
University of Science and Technology of China

Exceptions

Failures and Exceptions

- ***Failures*: with no associated data**
- ***Exceptions:* with associated data**
- **FPCF: enrich PCF with a failure**
  - **Syntax**

Exp	e	::=	$\texttt{fail}$	$\texttt{fail}$ signal a failure
			$\texttt{catch}(e_1; e_2)$	$\texttt{catch}\, e_1\, \texttt{ow}\, e_2$ catch a failure

  - **Statics**

$$\frac{}{\Gamma \vdash \texttt{fail} : \tau} \tag{29.1a}$$

$$\frac{\Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash \texttt{catch}(e_1; e_2) : \tau} \tag{29.1b}$$

Failures: FPCF

FPCF

- **Dynamics: using a techniqure called stack unwinding**
  - Failure state: $k\blacktriangleleft$

$$\frac{}{k \triangleright \texttt{fail} \longmapsto k \blacktriangleleft} \quad (29.2a)$$

$$\frac{}{k \triangleright \texttt{catch}(e_1; e_2) \longmapsto k;\texttt{catch}(-; e_2) \triangleright e_1} \quad (29.2b)$$

当对catch求值时，会在控制栈中安置一个失败处理器

$$\frac{}{k;\texttt{catch}(-; e_2) \triangleleft v \longmapsto k \triangleleft v} \quad (29.2c)$$

$$\frac{}{k;\texttt{catch}(-; e_2) \blacktriangleleft \longmapsto k \triangleright e_2} \quad (29.2d)$$

在当前控制栈上下文下对处理器求值，故处理器中的失败可以进一步向上传播

$$\frac{(f \neq \texttt{catch}(-; e))}{k; f \blacktriangleleft \longmapsto k \blacktriangleleft} \quad (29.2e)$$

当对fail求值时，会出栈直至遇到最近的外层失败处理器来展开控制栈，将控制传给该处理器

$$\frac{}{\epsilon \text{ initial}}$$

$$\frac{e \text{ val}}{\epsilon \triangleleft e \text{ final}}$$

$$\frac{}{\epsilon \blacktriangleleft \text{ final}}$$

Exceptions: XPCF

XPCF

- **Syntax**

Exp	e	$::=$	$\texttt{raise}(e)$	raise an exception
			$\texttt{try}(e_1; x.e_2)$	handle an exception

- **Statics:** τ_{exn} **type of exception values**

$$\frac{\Gamma \vdash e : \tau_{\text{exn}}}{\Gamma \vdash \texttt{raise}(e) : \tau} \tag{29.4a}$$

$$\frac{\Gamma \vdash e_1 : \tau \quad \Gamma, x : \tau_{\text{exn}} \vdash e_2 : \tau}{\Gamma \vdash \texttt{try}(e_1; x.e_2) : \tau} \tag{29.4b}$$

Exceptions: XPCF

XPCF

Dynamics

$$\frac{}{k \triangleright \mathtt{raise}(e) \longmapsto k;\mathtt{raise}(-) \triangleright e} \quad (29.5a)$$

$$\frac{}{k;\mathtt{raise}(-) \triangleleft e \longmapsto k \blacktriangleleft e} \quad (29.5b)$$

$$\frac{}{k \triangleright \mathtt{try}(e_1; x.e_2) \longmapsto k;\mathtt{try}(-; x.e_2) \triangleright e_1} \quad (29.5c)$$

$$\frac{}{k;\mathtt{try}(-; x.e_2) \triangleleft e \longmapsto k \triangleleft e} \quad (29.5d)$$

$$\frac{}{k;\mathtt{try}(-; x.e_2) \blacktriangleleft e \longmapsto k \triangleright [e/x]e_2} \quad (29.5e)$$

$$\frac{(f \neq \mathtt{try}(-; x.e_2))}{k; f \blacktriangleleft e \longmapsto k \blacktriangleleft e} \quad (29.5f)$$

$$\frac{}{\epsilon \triangleright e \text{ initial}}$$

$$\frac{e \text{ val}}{\epsilon \triangleleft e \text{ final}}$$

$$\frac{}{\epsilon \blacktriangleleft e \text{ final}}$$

THANKS