PTFAE – TFAE with Parametric Polymorphism

1 INTRODUCTION

PTFAE is a toy language for the COSE212 course at Korea University. PTFAE stands for an extension of the TFAE language with **parametric polymorphism**, and it supports the following features:

- **number (integer) values**
- **basic arithmetic operators**: addition (+) and multiplication (*)
- **first-class functions**
- **immutable variables** (val)
- **parametric polymorphism** (forall)
- **static type checking**

This document is the specification of PTFAE. First, Section 2 describes the concrete syntax, and Section 3 describes the abstract syntax. Then, Section 4 describes the type system. Finally, Section 5 describes the big-step operational (natural) semantics of PTFAE.

2 CONCRETE SYNTAX

The concrete syntax of PTFAE is written in a variant of the extended Backus–Naur form (EBNF). The notation `<nt>` denotes a nonterminal, and `"t"` denotes a terminal. We use ? to denote an optional element and + (or *) to denote one or more (or zero or more) repetitions of the preceding element. We use butnot to denote a set difference to exclude some strings from a producible set of strings. We omit some obvious terminals using the ellipsis (...) notation.

```
// basic elements
<digit>    ::= "0" | "1" | "2" | ... | "9"
<number>   ::= "-"? <digit>+
<alphabet> ::= "A" | "B" | "C" | ... | "Z" | "a" | "b" | "c" | ... | "z"
<idstart>  ::= <alphabet> | "_"
<idcont>   ::= <alphabet> | "_" | <digit>
<keyword>  ::= "val" | "Number" | "forall"
<id>       ::= <idstart> <idcont>* butnot <keyword>

// expressions
<expr>     ::= <number> | <expr> "+" <expr> | <expr> "*" <expr>
             | "(" <expr> ")" | "{" <expr> "}"
             | "val" <id> "=" <expr> ";"? <expr> | <id>
             | "(" <id> ":" <type> ")" "=>" <expr> | <expr> "(" <expr> ")"
             | "forall" "[" <type> "]" <expr> | <expr> "[" <type> "]"

// types
<type>     ::= "(" <type> ")" | "Number" | <id> | "[" <id> "]" <type>
```

For types, the arrow (=>) operator is right-associative. For expressions, the precedence and associativity of operators are defined as follows:

Operator	Associativity	Precedence
*	left	1
+	left	2

3 ABSTRACT SYNTAX

The abstract syntax of PTFAE is defined as follows:

$$
\begin{array}{llll}
\text{Expressions} \quad \mathbb{E} \ni e ::= & n & \text{(Num)} \\
& \mid e + e & \text{(Add)} \\
& \mid e \times e & \text{(Mul)} \\
& \mid \text{val } x{=}e;\ e & \text{(Val)} \\
& \mid x & \text{(Id)} \\
& \mid \lambda x{:}\tau.e & \text{(Fun)} \\
& \mid e(e) & \text{(App)} \\
& \mid \forall \alpha.e & \text{(TypeAbs)} \\
& \mid e[\tau] & \text{(TypeApp)}
\end{array}
$$

$$
\begin{array}{llll}
\text{Numbers} & n & \in \mathbb{Z} & \text{(BigInt)} \\
\text{Identifiers} & x & \in \mathbb{X} & \text{(String)} \\
\text{Type Variables} & \alpha & \in \mathbb{X}_\alpha & \text{(String)} \\
\\
\text{Types} & \mathbb{T} \ni \tau ::= & \text{num} & \text{(NumT)} \\
& \mid \tau \rightarrow \tau & \text{(ArrowT)} \\
& \mid \alpha & \text{(VarT)} \\
& \mid \forall \alpha.\tau & \text{(PolyT)}
\end{array}
$$

4 TYPE SYSTEM

This section explains type system of PTFAE, and we use the following notations:

$$
\text{Type Environments} \quad \Gamma \in (\mathbb{X} \xrightarrow{\text{fin}} \mathbb{T}) \times \mathcal{P}(\mathbb{X}_\alpha) \quad \text{(TypeEnv)}
$$

In the type system, type checking is defined with the following typing rules:

$$
\boxed{\Gamma \vdash e : \tau}
$$

$$
\tau{-}\text{Num} \; \frac{}{\Gamma \vdash n : \text{num}}
\qquad
\tau{-}\text{Add} \; \frac{\Gamma \vdash e_1 : \text{num} \quad \Gamma \vdash e_2 : \text{num}}{\Gamma \vdash e_1 + e_2 : \text{num}}
\qquad
\tau{-}\text{Mul} \; \frac{\Gamma \vdash e_1 : \text{num} \quad \Gamma \vdash e_2 : \text{num}}{\Gamma \vdash e_1 \times e_2 : \text{num}}
$$

$$
\tau{-}\text{Val} \; \frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma[x : \tau_1] \vdash e_2 : \tau_2}{\Gamma \vdash \text{val } x{=}e_1;\ e_2 : \tau_2}
\qquad
\tau{-}\text{Id} \; \frac{x \in \text{Domain}(\Gamma)}{\Gamma \vdash x : \Gamma(x)}
$$

$$
\tau{-}\text{Fun} \; \frac{\Gamma[x : \tau] \vdash e : \tau'}{\Gamma \vdash \lambda x{:}\tau.e : \tau \rightarrow \tau'}
\qquad
\tau{-}\text{App} \; \frac{\Gamma \vdash e_0 : \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash e_1 : \tau_3 \quad \tau_1 \equiv \tau_3}{\Gamma \vdash e_0(e_1) : \tau_2}
$$

$$
\tau{-}\text{TypeAbs} \; \frac{\alpha \notin \text{Domain}(\Gamma) \quad \Gamma[\alpha] \vdash e : \tau}{\Gamma \vdash \forall \alpha.e : \forall \alpha.\tau}
\qquad
\tau{-}\text{TypeApp} \; \frac{\Gamma \vdash \tau \quad \Gamma \vdash e : \forall \alpha.\tau'}{\Gamma \vdash e[\tau] : \tau'[\alpha \leftarrow \tau]}
$$

the following rules for well-formedness of types:

$$
\boxed{\Gamma \vdash \tau}
$$

$$
\frac{}{\Gamma \vdash \text{num}}
\qquad
\frac{\Gamma \vdash \tau \quad \Gamma \vdash \tau'}{\Gamma \vdash \tau \rightarrow \tau'}
\qquad
\frac{\alpha \in \text{Domain}(\Gamma)}{\Gamma \vdash \alpha}
\qquad
\frac{\Gamma[\alpha] \vdash \tau}{\Gamma \vdash \forall \alpha.\tau}
$$

and the following rules for type equivalence:

$$
\boxed{\tau \equiv \tau}
$$

$$
\frac{}{\text{num} \equiv \text{num}}
\qquad
\frac{\tau_1 \equiv \tau_1' \quad \tau_2 \equiv \tau_2'}{(\tau_1 \rightarrow \tau_2) \equiv (\tau_1' \rightarrow \tau_2')}
\qquad
\frac{}{\alpha \equiv \alpha}
\qquad
\frac{\tau \equiv \tau'[\alpha' \leftarrow \alpha]}{\forall \alpha.\tau \equiv \forall \alpha'.\tau'}
$$

5 SEMANTICS

We use the following notations in the semantics:

Values $\quad \mathbb{V} \ni v ::= n \qquad \qquad$ (NumV) $\qquad \qquad$ Environments $\quad \sigma \in \mathbb{X} \xrightarrow{\text{fin}} \mathbb{V}$ (Env)

$\qquad \qquad \qquad \mid \langle \lambda x.e, \sigma \rangle \quad$ (CloV)

$\qquad \qquad \qquad \mid \langle \forall \alpha.e, \sigma \rangle \quad$ (TypeAbsV)

The big-step operational (natural) semantics of PTFAE is defined as follows:

$$\boxed{\sigma \vdash e \Rightarrow v}$$

$$\text{Num} \; \frac{}{\sigma \vdash n \Rightarrow n} \qquad \text{Add} \; \frac{\sigma \vdash e_1 \Rightarrow n_1 \qquad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 + e_2 \Rightarrow n_1 + n_2} \qquad \text{Mul} \; \frac{\sigma \vdash e_1 \Rightarrow n_1 \qquad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 \times e_2 \Rightarrow n_1 \times n_2}$$

$$\text{Val} \; \frac{\sigma \vdash e_1 \Rightarrow v_1 \qquad \sigma[x \mapsto v_1] \vdash e_2 \Rightarrow v_2}{\sigma \vdash \text{val } x = e_1; \; e_2 \Rightarrow v_2} \qquad \text{Fun} \; \frac{}{\sigma \vdash \lambda x : \tau.e \Rightarrow \langle \lambda x.e, \sigma \rangle}$$

$$\text{Id} \; \frac{x \in \text{Domain}(\sigma)}{\sigma \vdash x \Rightarrow \sigma(x)} \qquad \text{App} \; \frac{\sigma \vdash e_0 \Rightarrow \langle \lambda x.e_2, \sigma' \rangle \qquad \sigma \vdash e_1 \Rightarrow v_1 \qquad \sigma'[x \mapsto v_1] \vdash e_2 \Rightarrow v_2}{\sigma \vdash e_0(e_1) \Rightarrow v_2}$$

$$\text{TypeAbs} \; \frac{}{\sigma \vdash \forall \alpha.e \Rightarrow \langle \forall \alpha.e, \sigma \rangle} \qquad \text{TypeApp} \; \frac{\sigma \vdash e \Rightarrow \langle \forall \alpha.e', \sigma' \rangle \qquad \sigma' \vdash e' \Rightarrow v}{\Gamma \vdash e[\tau] : v}$$