TFAE – FAE with Type System

1 INTRODUCTION

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

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

This document is the specification of TFAE. 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 TFAE.

2 CONCRETE SYNTAX

The concrete syntax of TFAE 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"
<id>       ::= <idstart> <idcont>* butnot <keyword>

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

// types
<type>     ::= "Num" | <type> "=>" <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 TFAE is defined as follows:

$$
\begin{array}{llll}
\text{Expressions} & \mathbb{E} \ni e ::= n & (\texttt{Num}) \\
& \quad \mid e + e & (\texttt{Add}) \\
& \quad \mid e \times e & (\texttt{Mul}) \\
& \quad \mid \texttt{val}\ x\texttt{=}e\texttt{;}\ e & (\texttt{Val}) \\
& \quad \mid x & (\texttt{Id}) \\
& \quad \mid \lambda x\!:\!\tau.e & (\texttt{Fun}) \\
& \quad \mid e(e) & (\texttt{App})
\end{array}
\qquad
\begin{array}{llll}
\text{Numbers} & n \in \mathbb{Z} & (\texttt{BigInt}) \\
\text{Identifiers} & x \in \mathbb{X} & (\texttt{String}) \\
\text{Types} & \mathbb{T} \ni \tau ::= \texttt{Num} & (\texttt{NumT}) \\
& \quad \mid \tau \to \tau & (\texttt{ArrowT})
\end{array}
$$

4 TYPE SYSTEM

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

$$\text{Type Environments} \quad \Gamma \in \mathbb{X} \xrightarrow{\text{fin}} \mathbb{T} \quad (\texttt{TypeEnv})$$

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

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

$$
\tau\texttt{-Num}\ \frac{}{\Gamma \vdash n : \texttt{Num}}
\qquad
\tau\texttt{-Add}\ \frac{\Gamma \vdash e_1 : \texttt{Num} \qquad \Gamma \vdash e_2 : \texttt{Num}}{\Gamma \vdash e_1 + e_2 : \texttt{Num}}
\qquad
\tau\texttt{-Mul}\ \frac{\Gamma \vdash e_1 : \texttt{Num} \qquad \Gamma \vdash e_2 : \texttt{Num}}{\Gamma \vdash e_1 \times e_2 : \texttt{Num}}
$$

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

$$
\tau\texttt{-Fun}\ \frac{\Gamma[x \mapsto \tau] \vdash e : \tau'}{\Gamma \vdash \lambda x\!:\!\tau.e : \tau \to \tau'}
\qquad
\tau\texttt{-App}\ \frac{\Gamma \vdash e_0 : \tau_1 \to \tau_2 \qquad \Gamma \vdash e_1 : \tau_1}{\Gamma \vdash e_0(e_1) : \tau_2}
$$

5 SEMANTICS

We use the following notations in the semantics:

$$
\begin{array}{lll}
\text{Values} & \mathbb{V} \ni v ::= n & (\texttt{NumV}) \\
& \quad \mid \langle \lambda x.e, \sigma \rangle & (\texttt{CloV}) \\
\text{Environments} & \sigma \in \mathbb{X} \xrightarrow{\text{fin}} \mathbb{V} & (\texttt{Env})
\end{array}
$$

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

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

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

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

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