t	$::=$		terms
		b	boolean value
		n	numeric value
		op	operator
		$\lambda x.\ t$	abstraction
		x	variable
		$t\ t$	application
		$\mathsf{mlet}\ x = t\ \mathsf{in}\ t$	overloading let
		$t :: T$	ascription
b	$::=$		boolean value
		true	true value
		false	false value
op	$::=$		operators
		add1	sum
		not	negation

T	$::=$		types
		Int	type of integers
		Bool	type of booleans
		$T \to T$	type of functions
v	$::=$		configuration − values
		b	boolean value
		n	numeric value
		op	operator
		$(\lambda x.\ t)[s]$	closure
c	$::=$		configurations
		v	
		$t[s]$	
		$c\ c$	
		$\mathsf{mlet}\ x = c\ \mathsf{in}\ c$	
		$c :: T$	
		error	
s	$::=$		explicit substitutions
		$\bullet$	empty substitution
		$x \mapsto \{\overline{v}\}, s$	variable substitution

Figure 1: Syntax of the simply typed lambda-calculus vith overloading.

- Non deterministic.
- Type error detection.
- Without type annotation in lambda functions or mlet, only in ascription without use.

$$\boxed{c \longrightarrow c}$$

$$b[s] \longrightarrow b \qquad \text{(False)}$$

$$n[s] \longrightarrow n \qquad \text{(Num)}$$

$$op[s] \longrightarrow op \qquad \text{(Op)}$$

$$x[\,] \longrightarrow \mathsf{error} \qquad \text{(ErrVarFail)}$$

$$x[x \mapsto \{\overline{v}\}, s] \longrightarrow v_i \qquad \text{(VarOk)}$$

$$\frac{x \neq y}{x[y \mapsto \{\overline{v}\}, s] \longrightarrow x[s]} \qquad \text{(VarNext)}$$

$$(t :: T)[s] \longrightarrow t[s] :: T \qquad \text{(AscSub)}$$

$$(\mathsf{mlet}\ x = t_1\ \mathsf{in}\ t_2)[s] \longrightarrow \mathsf{mlet}\ x = t_1[s]\ \mathsf{in}\ t_2[s] \qquad \text{(LetSub)}$$

$$(t_1\ t_2)[s] \longrightarrow t_1[s]\ t_2[s] \qquad \text{(AppSub)}$$

$$v :: T \longrightarrow v \qquad \text{(Asc)}$$

$$\mathsf{mlet}\ x = v\ \mathsf{in}\ t_2[s] \longrightarrow t_2[x \mapsto v \oplus s] \qquad \text{(Let)}$$

$$(\lambda x.\ t_2)[s]\ v \longrightarrow ([x \mapsto v]t_2)[s] \qquad \text{(App)}$$

$$\mathsf{add1}\ n \longrightarrow n + 1 \qquad \text{(Sum)}$$

$$\mathsf{not}\ b \longrightarrow \neg\ b \qquad \text{(Negation)}$$

$$\frac{c \longrightarrow c'}{c :: T \longrightarrow c' :: T} \qquad \text{(Asc1)}$$

$$\frac{c_1 \longrightarrow c_1'}{\mathsf{mlet}\ x = c_1\ \mathsf{in}\ c_2 \longrightarrow \mathsf{mlet}\ x = c_1'\ \mathsf{in}\ c_2} \qquad \text{(Let1)}$$

$$\frac{c_1 \longrightarrow c_1'}{c_1\ c_2 \longrightarrow c_1'\ c_2} \qquad \text{(App1)}$$

$$\frac{c \longrightarrow c'}{v\ c \longrightarrow v\ c'} \qquad \text{(App2)}$$

Figure 2: Configuration reduction rules.