```
configurations
                                                                             c
                                                                                    ::=
                                                               values
v
       ::=
                                                                                           w^*
                                                           true value
              true
                                                                                           nv[s]
              false
                                                          false value
                                                                                           c :: T
                                                      numeric value
                                                                                           olet x:T=c in c
              \lambda x : T. t
                                                         abstraction
                                                                                           c c
                                                                                           error
                                                       non-values
nv
      ::=
                                                                             T
                                                                                                                                        types
                                                                                    ::=
                                                             variable
              \boldsymbol{x}
                                                                                           Int
                                                                                                                            type of integers
                                                         application
              t_1 t_2
                                                                                           Bool
                                                                                                                           type of booleans
              olet x:T=t in t
                                                     overloading let
                                                                                           T \to T
                                                                                                                           type of functions
              t :: T
                                                           ascription
                                                                                                                               multi - types
                                                               terms
t
      ::=
                                                                                                                                \mathsf{multi}-\mathsf{type}
                                                                                           \{\overline{T}\}
                                                               values
              v
                                                       non - values
              nv
                                                                             Γ
                                                                                                                             typing contexts
                                                                                                                              empty context
      ::=
                                                   labeled - values
w
                                                                                           \Gamma, x : T
                                                                                                                      term variable binding
              < true, Bool >
                                               labeled - true value
              < false, Bool >
                                              labeled - false value
                                                                                                                    multi - typing contexts
                                                                             \phi
                                                                                    ::=
                                          labeled - numeric value
              < n, \mathsf{Int} >
                                                                                                                              empty context
              <\lambda x:T.\ t, \mathsf{Fun}>[s]
                                              {\sf labeled-abstraction}
                                                                                           \phi, x: T^*
                                                                                                                      term variable binding
                                          labeled - multi - values
w^*
                                                                             s
                                                                                                                       explicit substitutions
              \{\overline{w}\}
                                                                                                                         empty substitution
                                                                                           x \mapsto \{(\overline{w:T})\}, s
                                                                                                                       variable substitution
```

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

$$v^*[s] \longrightarrow v^* \qquad (\text{Multi - Values})$$

$$x[x \mapsto \{(\overline{w:T_1})\}, s \longrightarrow \{\overline{w}\} \qquad (\text{VarOk})$$

$$\frac{x \neq y}{x[y \mapsto \{(\overline{w:T_1})\}, s] \longrightarrow x[s]} \qquad (\text{VarNext})$$

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

$$\frac{\text{selvt}(w^*, T) = w}{w^*::T \longrightarrow w} \qquad (\text{Asc})$$

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

$$(\text{olet } x:T_1 = t_1 \text{ in } t_2)[s] \longrightarrow \text{olet } x:T_1 = t_1[s] \text{ in } t_2[s] \qquad (\text{LetSub})$$

$$\frac{\text{selvt}(w^*, T_1) = w}{\text{olet } x:T_1 = w^* \text{ in } t_2[s] \longrightarrow t_2[x \mapsto (w:T_1) \oplus s]} \qquad (\text{Let})$$

$$\frac{c_1 \longrightarrow c'_1}{\text{olet } x:T_1 = c_1 \text{ in } t_2[s] \longrightarrow \text{olet } x:T_1 = c'_1 \text{ in } t_2[s]} \qquad (\text{AppSub})$$

$$\frac{c_1 \longrightarrow c'_1}{\text{olet } x:T_1 = t_2[s], \text{Fun} > w \longrightarrow ([x \mapsto w]t_2)[s]} \qquad (\text{App})$$

$$\frac{\langle (\lambda x:T_1, t_2)[s], \text{Fun} > w \longrightarrow ([x \mapsto w]t_2)[s]}{\langle (\lambda x:T_1, t_2)[s], \text{Fun} > w_2j} \qquad (\text{App1})$$

$$\frac{c_1 \longrightarrow c'_1}{c_1 c_2 \longrightarrow c'_1 c_2} \qquad (\text{App2})$$

$$\frac{c_1 \longrightarrow c'_1}{c_1 c_2 \longrightarrow c'_1 c_2} \qquad (\text{App3})$$

Figure 2: Configuration reduction rules.

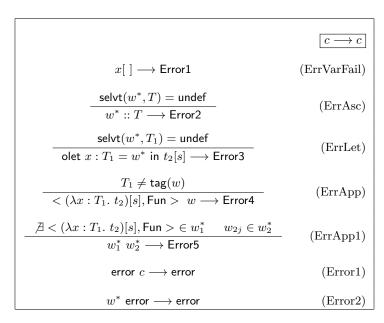


Figure 3: Configuration reduction rules.