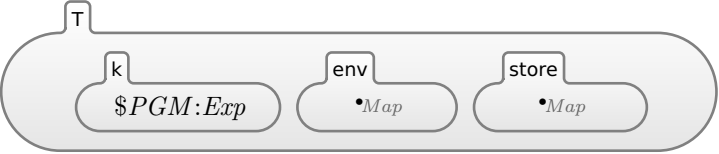


LAMBDA

MODULE LAMBDA

SYNTAX $Exp ::= Id$
 $\mid \lambda Id. Exp$
 $\mid Exp \ Exp \text{ [strict]}$
 $\mid (Exp) \text{ [bracket]}$

CONFIGURATION:



SYNTAX $Val ::= \text{closure } (Map, Id, Exp) \text{ [klabel('closure)]}$

SYNTAX $Exp ::= Val$

SYNTAX $KResult ::= Val$

RULE [structural]

RULE

RULE

RULE [structural]

SYNTAX $Val ::= Int$
 $\mid Bool$

SYNTAX $Exp ::= Exp * Exp \text{ [strict]}$
 $\mid Exp / Exp \text{ [strict]}$
 $\mid Exp + Exp \text{ [strict]}$
 $\mid Exp <= Exp \text{ [strict]}$

RULE $\frac{I1 * I2}{I1 *_{Int} I2}$

RULE $\frac{I1 / I2}{I1 \div_{Int} I2}$

RULE $\frac{I1 + I2}{I1 +_{Int} I2}$

RULE $\frac{I1 <= I2}{I1 \leq_{Int} I2}$

SYNTAX $Exp ::= \text{if } Exp \text{ then } Exp \text{ else } Exp \text{ [strict(1)]}$

RULE $\frac{\text{if true then } E \text{ else } —}{E}$

RULE $\frac{\text{if false then } — \text{ else } E}{E}$

SYNTAX $Exp ::= \text{let } Id = Exp \text{ in } Exp$

RULE $\frac{\text{let } X = E \text{ in } E':Exp}{(\lambda X. \bar{E}') \ E}$ [macro]

SYNTAX $Exp ::= \text{letrec } Id \ Id = Exp \text{ in } Exp$

SYNTAX $Id ::= \$x$
 $\mid \$y$

RULE $\frac{\text{letrec } F:Id \ X:Id = E \text{ in } E'}{\text{let } F = (\lambda \$x. ((\lambda F. \lambda X. E) \ (\lambda \$y. (\$x \ \$x \ \$y)))) \ (\lambda \$x. ((\lambda F. \lambda X. E) \ (\lambda \$y. (\$x \ \$x \ \$y)))) \text{ in } E'}$ [macro]

END MODULE