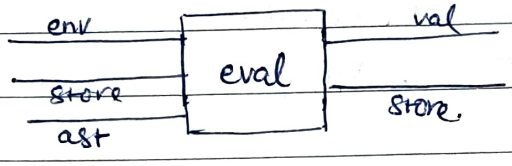
STORES



modelling stores:

dependent type

var n : ℕ A(n) = Num + Bool + Proc + $n_<$ [0 . . . n-1]

store (n, A(n)) = $n_<$ ⟶ A(n)

F* - dependently typed language.

STORE OPERATIONS

store extend: store (n, A(n)), A(n+1)

↓

store (n+1, A(n+1))

store-get: store (n, A(n)), $n_<$ → A(n)

Store-set : store (n, A(n)), $n_<$, A(n) → Store (n, A(n))

IMPLEMENTATION

store-extend (s, v) = {n → v} s

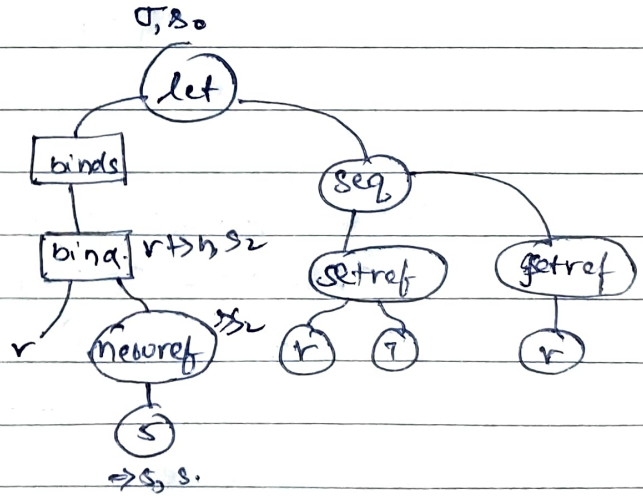
store-get (s, i) = s(i)

store-set (s, i, v) = {i → v} s

ExpVal = A(n)

Language e ::= n | b | x | (if e e e) | (fun x̄ e) | (let x̄ e e) | (letrec f x̄ e e) | newref e | getref e | setref e e | (seq ē)

(let ([r (newref 5)])
  (seq
    (setref r 7)
    (getref r))))



$s_1 = s_0$

$s_2 : (n+1, A(n+1))$

$s = \{n \mapsto 5\}\, s_1$

$r \to n$