Imperial College London

MEng Individual Project

Department of Computing

Imperial College of Science, Technology and Medicine

Ochre: A Dependently Typed Systems Programming Language

Author:
Charlie Lidbury

Supervisor(s):
Steffen van Bakel
Nicolas Wu

June 21, 2024

Submitted in partial fulfillment of the requirements for the MEng Computing of Imperial College London

```
Bool = 'true | 'false;
Pair = (Bool, Bool);

overwrite = (p: &mut Pair) -> 'unit {
    *p = ('true, 'true);
    'unit
};

pair = ('false, 'false);
overwrite(&mut pair);
```

```
type Pair = (bool, bool);

fn overwrite(p: &mut Pair) {
    *p = ('true, 'true);
}

fn main() {
    let mut pair = (false, false);
    overwrite(&mut pair);
}
```

```
Bool = 'true | 'false;
Pair = (Bool, L -> L);

overwrite = (p: &mut Pair) -> 'unit {
    *p = ('true, 'true);
    'unit
};

pair = ('false, 'false);
overwrite(&mut pair);
```

```
type Same = | MkSame: b:bool -> b2:bool{b == b2} -> Same

val overwrite: p:ptr Same -> Stack unit
let overwrite p =
  p := Same false false

let main () : Stack unit =
  let pair = alloc (MkSame true true) in
  overwrite pair;
  ()
```

$$\frac{\langle def. \Rightarrow for\ x \rangle \qquad \Omega' = \Omega \left[\frac{x \mapsto \top}{x \mapsto v} \right]}{\Omega \vdash x \Rightarrow m \dashv \Omega'}$$

Figure 1: Reading removes a value from the environment, whereas writing adds a value.

```
                                  // Ω₀ = ∅
Same = ('a | 'b, L -> L);         // Ω₁ = ∅, Same ↦ ({'a, 'b}, L → L)

overwrite = (p: &mut Same) -> 'unit {
                                  // Ω₁₀ = Ω₁, p ↦ borrow^m l ({'a, 'b}, L → L), l ↦ ({'a, 'b}, L → L)
    (*p).0 = 'a;                  // Ω₁₁ = Ω₁, p ↦ borrow^m l ('a, _ → {'a, 'b}), l ↦ ({'a, 'b}, L → L)
    (*p).1 = 'a;                  // Ω₁₂ = Ω₁, p ↦ borrow^m l ('a, _ → 'a), l ↦ ({'a, 'b}, L → L)
    'unit                         // Ω₁₂ ⊢ drop
}                                 // Ω₂ = Ω₁, overwrite ↦ (p: &mut Same) → 'unit

pair = ('b, 'b);                  // Ω₃ = Ω₂, pair ↦ ('b, _ → 'b)
overwrite(&mut pair);             // Ω₄ = Ω₂, pair ↦ ({'a, 'b}, L → L)

pair // ({'a, 'b}, L → L)
```

```
Bool = 'true | 'false;
Same = (Bool, L -> L);

overwrite = (p: &mut Pair) -> 'unit {


    *p.0 = 'true;
    *p.1 = 'true;

    'unit
};
```

$\{x \mapsto \{\text{'five}\}\} \vdash x \Rightarrow \{\text{'five}\} \dashv \{x \mapsto \top\}$

$\Omega \vdash M \diamond t \dashv \Omega'$

$$\left\{ \begin{array}{ccc} \Rightarrow, & \rightarrow, & \dashrightarrow, \\ \Leftarrow, & \leftarrow, & \dashleftarrow \end{array} \right\}$$