

# 15-312 Assignment 1

Andrew Carnegie  
(andrew)

October 27, 2017

Type	$\tau ::=$		
	<code>nat</code>	<code>nat</code>	naturals
	<code>unit</code>	<code>unit</code>	unit
	<code>bool</code>	<code>bool</code>	boolean
	<code>prod(<math>\tau_1; \tau_2</math>)</code>	$\tau_1 \times \tau_2$	product
	<code>arr(<math>\tau_1; \tau_2</math>)</code>	$\tau_1 \rightarrow \tau_2$	function
	<code>list(<math>\tau</math>)</code>	$\tau$ list	list
Exp	$e ::=$		
	$x$	$x$	variable
	<code>nat[<math>n</math>]</code>	$\bar{n}$	number
	<code>unit</code>	<code>()</code>	unit
	<code>T</code>	<code>T</code>	true
	<code>F</code>	<code>F</code>	false
	<code>if(<math>x; e_1; e_2</math>)</code>	<code>if <math>x</math> then <math>e_1</math> else <math>e_2</math></code>	if
	<code>lam(<math>x : \tau.e</math>)</code>	$\lambda x : \tau.e$	abstraction
	<code>ap(<math>f; x</math>)</code>	$f(x)$	application
	<code>tpl(<math>x_1; x_2</math>)</code>	$\langle x_1, x_2 \rangle$	pair
	<code>case(<math>x_1, x_2.e_1</math>)</code>	<code>case <math>p \{ (x_1; x_2) \hookrightarrow e_1 \}</math></code>	match pair
	<code>nil</code>	<code>[]</code>	nil
	<code>cons(<math>x_1; x_2</math>)</code>	$x_1 :: x_2$	cons
	<code>case{<math>l</math>}(<math>e_1; x, xs.e_2</math>)</code>	<code>case <math>l \{ \text{nil} \hookrightarrow e_1 \mid \text{cons}(x; xs) \hookrightarrow e_2 \}</math></code>	match list
	<code>let(<math>e_1; x : \tau.e_2</math>)</code>	<code>let <math>x = e_1</math> in <math>e_2</math></code>	let
Val	$v ::=$		
	<code>val(<math>n</math>)</code>	$n$	numeric value
	<code>val(T)</code>	<code>T</code>	true value
	<code>val(F)</code>	<code>F</code>	false value
	<code>val(Null)</code>	<code>Null</code>	null value
	<code>val(cl(<math>V; x.e</math>))</code>	$(V, x.e)$	function value
	<code>val(<math>l</math>)</code>	$l$	loc value
	<code>val(pair(<math>v_1; v_2</math>))</code>	$\langle v_1, v_2 \rangle$	pair value
State	$s ::=$		
	<code>alive</code>	<code>alive</code>	live value
	<code>dead</code>	<code>dead</code>	dead value
Loc	$l ::=$		
	<code>loc(<math>l</math>)</code>	$l$	location
Var	$l ::=$		
	<code>var(<math>x</math>)</code>	$x$	variable

# 1 Paths and aliasing

Model dynamics using judgement of the form:

$$\boxed{V, H, R, F \vdash e \Downarrow v, H', F'}$$

Where  $V : \text{Var} \rightarrow \text{Val} \times \text{State}$ ,  $H : \text{Loc} \rightarrow \text{Val}$ ,  $R \subseteq \text{Loc}$ , and  $F \subseteq \text{Loc}$ . This can be read as: under stack  $V$ , heap  $H$ , roots  $R$ , freelist  $F$ , the expression  $e$  evaluates to  $v$ , and engenders a new heap  $H'$  and freelist  $F'$ .

For a partial map  $f : A \rightarrow B$ , we write  $\text{dom}$  for the defined values of  $f$ . Sometimes we shorten  $x \in \text{dom}(f)$  to  $x \in f$ . We write  $f[x \mapsto y]$  for the extension of  $f$  where  $x$  is mapped to  $y$ , with the constraint that  $x \notin \text{dom}(f)$ . We write  $f\{x \mapsto y\}$  for the update map, which is the same as the extension map, except that  $x$  is remapped to  $y$  when  $x \in \text{dom}(f)$ . Write  $C \triangleleft f : C \rightarrow B$  for the domain restriction of  $f$  to  $C$  where  $C \subseteq A$ . Write  $C \trianglelefteq f : (A \setminus C) \rightarrow B$  for the domain anti-restriction of  $f$  to  $C$ .

Note that the stack maps each variable to a value  $v$  and a state  $s$ . If  $s$  is alive, then  $v$  can still be used, while **dead** indicates that  $v$  is already used and cannot be used again. We write  $\bar{V} = \{x \in V \mid V(x) = (\_, \text{alive})\}$  for the variables in  $V$  that are alive, and  $V^* : \bar{V} \rightarrow \text{Val}$  for the associated restricted map  $x \mapsto \text{fst}(V(x))$  which projects out the value component of live variables.

Roots represents the set of locations required to compute the continuation *excluding* the current expression. We can think of roots as the heap allocations necessary to compute the context with a hole that will be filled by the current expression.

In order prove soundness of the type system, we need some auxiliary judgements to defining properties of a heap. Below we define  $\text{reach} : \text{Val} \rightarrow \{\{ \text{Loc} \}\}$  that maps stack values its the root *multiset*, the multiset of locations that's already on the stack.

Next we define reachability of values:

$$\begin{aligned} \text{reach}_H(\langle v_1, v_2 \rangle) &= \text{reach}_H(v_1) \uplus \text{reach}_H(v_2) \\ \text{reach}_H(l) &= \{l\} \uplus \text{reach}_H(H(l)) \\ \text{reach}_H(-) &= \emptyset \end{aligned}$$

For a multiset  $S$ , we write  $\mu_S : S \rightarrow \mathbb{N}$  for the multiplicity function of  $S$ , which maps each element to the count of its occurrence. If  $\forall s \in S. \mu(s) = 1$ , then  $S$  is a property set, and we denote it by  $\text{set}(S)$ . Additionally,  $A \uplus B$  denotes counting union of sets where  $\mu_{A \uplus B}(s) = \mu_A(s) + \mu_B(s)$ , and  $A \cup B$  denotes the usual union where  $\mu_{A \cup B}(s) = \max(\mu_A(s), \mu_B(s))$ . For the disjoint union of sets  $A$  and  $B$ , we write  $A \sqcup B$ .

Next, we define the predicates **no\_alias**, **no\_ref**, and **disjoint**:

**no\_alias**( $V, H$ ):  $\forall x, y \in \bar{V}, x \neq y. \text{ Let } r_x = \text{reach}_H(\bar{V}(x)), r_y = \text{reach}_H(\bar{V}(y)). \text{ Then:}$

1.  $\text{set}(r_x), \text{set}(r_y)$

$$2. \ r_x \cap r_y = \emptyset$$

$$\text{no\_ref}(V, H, v): \quad \forall x \in \overline{V}. \text{reach}_H(V^\star(x)) \cap \text{reach}_H(v) = \emptyset.$$

$$\text{safe}(V, H, F): \quad \forall x \in \overline{V}. \text{reach}_H(V^\star(x)) \cap F = \emptyset$$

$$\text{disjoint}(\mathcal{C}): \quad \forall X, Y \in \mathcal{C}. X \cap Y = \emptyset$$

For a stack  $V$  and a heap  $H$ , whenever  $\text{no\_alias}(V, H)$  holds, visually, one can think of the situation as the following: the induced graph of heap  $H$  with variables on the stack as additional leaf nodes is a forest: a disjoint union of arborescences (directed trees); consequently, there is at most one path from a live variable on the stack  $V$  to a location in  $H$  by following the pointers.

Next, we define  $\text{locs}_{V,H}$  using the previous notion of reachability.  $\text{size}$  calculates the number of cells a value occupies.  $\text{copy}(H, L, v)$  takes a heap  $H$ , a set of locations  $L$ , and a value  $v$ , and returns a new heap  $H'$  and a location  $l$  such that  $l$  maps to  $v$  in  $H'$ .

$$\text{locs}_{V,H}(e) = \bigcup_{x \in FV(e)} \text{reach}_H(V(x))$$

$$\text{size}(\langle v_1, v_2 \rangle) = \text{size}(v_1) + \text{size}(v_2)$$

$$\text{size}(\_) = 1$$

$$\text{copy}(H, L, \langle v_1, v_2 \rangle) =$$

$$\text{let } L_1 \sqcup L_2 \subseteq L$$

$$\text{where } |L_1| = \text{size}(v_1), |L_2| = \text{size}(v_2)$$

$$\text{let } H_1 = \text{copy}(H, L_1, v_1)$$

$$\text{let } H_2 = \text{copy}(H_1, L_2, v_2) \text{ in}$$

$$H_2\{l \mapsto v\}$$

$$\text{copy}(H, L, v) =$$

$$\text{let } l \in H \text{ in}$$

$$H\{l \mapsto v\}$$

## 2 Garbage collection semantics

$$\begin{array}{c}
\frac{V(x) = (v, \mathbf{alive})}{V, H, R, F \vdash x \Downarrow v, H, F} (S_1) \qquad \frac{}{V, H, R, F \vdash \bar{n} \Downarrow \mathbf{val}(n), H, F} (S_2) \\
\\
\frac{}{V, H, R, F \vdash \mathbf{T} \Downarrow \mathbf{val}(\mathbf{T}), H, F} (S_3) \qquad \frac{}{V, H, R, F \vdash \mathbf{F} \Downarrow \mathbf{val}(\mathbf{F}), H, F} (S_4) \\
\\
\frac{}{V, H, R, F \vdash () \Downarrow \mathbf{val}(\mathbf{Null}), H, F} (S_5) \\
\\
\frac{V(x) = \mathbf{T} \quad g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V,H}(e_1)\} \quad V, H, R, F \cup g \vdash e_1 \Downarrow v, H', F'}{V, H, R, F \vdash \mathbf{if}(x; e_1; e_2) \Downarrow v, H', F'} (S_6) \\
\\
\frac{V(x) = \mathbf{F} \quad g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V,H}(e_2)\} \quad V, H, R, F \cup g \vdash e_2 \Downarrow v, H', F'}{V, H, R, F \vdash \mathbf{if}(x; e_1; e_2) \Downarrow v, H', F'} (S_7) \\
\\
\frac{l \in F \quad F' = F \setminus \{l\} \quad H' = H[l \mapsto (V, x.e)]}{V, H, R, F \vdash \mathbf{lam}(x : \tau.e) \Downarrow l, H', F'} (S_8) \\
\\
\frac{V(f) = (V_1, x.e) \quad V(x) = v_1 \quad V_1[x \mapsto v_1], H, R \vdash e \Downarrow v, H', F'}{V, H, R, F \vdash f(x) \Downarrow v, H', F'} (S_9) \\
\\
\frac{V(x_1) = v_1 \quad V(x_2) = v_2}{V, H, R, F \vdash \langle x_1, x_2 \rangle \Downarrow \langle v_1, v_2 \rangle, H, F} (S_{10}) \\
\\
\frac{V(x) = (\langle v_1, v_2 \rangle, \mathbf{alive}) \quad g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V,H}(e)\} \quad V' = V\{x \mapsto (\langle v_1, v_2 \rangle, \mathbf{dead})\} \quad V'' = V'[x_1 \mapsto (v_1, \mathbf{alive}), x_2 \mapsto (v_2, \mathbf{alive})] \quad V'', H, R, F \cup g \vdash e \Downarrow v, H', F'}{V, H, R, F \vdash \mathbf{case } x \{ (x_1; x_2) \hookrightarrow e \} \Downarrow v, H', F'} (S_{11}) \\
\\
\frac{}{V, H, R, F \vdash \mathbf{nil} \Downarrow \mathbf{val}(\mathbf{Null}), H, F} (S_{12}) \\
\\
\frac{|L| = \text{size}_H(v) \quad v = \langle V^*(x_1), V^*(x_2) \rangle \quad L \sqcup \{l\} \subseteq F \quad F' = F \setminus (L \sqcup \{l\}) \quad H' = \text{copy}(H, L, v) \quad H'' = H'\{l \mapsto v\}}{V, H, R, F \vdash \mathbf{cons}(x_1; x_2) \Downarrow l, H'', F'} (S_{13}) \\
\\
\frac{V(x) = \mathbf{Null} \quad g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V,H}(e_1)\} \quad V, H, R, F \cup g \vdash e_1 \Downarrow v, H', F'}{V, H, R, F \vdash \mathbf{case } x \{ \mathbf{nil} \hookrightarrow e_1 \mid \mathbf{cons}(x_h; x_t) \hookrightarrow e_2 \} \Downarrow v, H', F'} (S_{14}) \\
\\
\frac{V(x) = (l, \mathbf{alive}) \quad H(l) = \langle v_h, v_t \rangle \quad V' = V\{x \mapsto (l, \mathbf{dead})\} \quad V'' = V'[x_h \mapsto (v_h, \mathbf{alive}), x_t \mapsto (v_t, \mathbf{alive})] \quad g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V'',H}(e_2)\} \quad V'', H, R, F \cup g \vdash e_2 \Downarrow v, H', F'}{V, H, R, F \vdash \mathbf{case } x \{ \mathbf{nil} \hookrightarrow e_1 \mid \mathbf{cons}(x_h; x_t) \hookrightarrow e_2 \} \Downarrow v, H', F'} (S_{15}) \\
\\
\frac{R' = R \cup \text{locs}_{V,H}(\mathbf{lam}(x : \tau.e_2)) \quad V, H, R', F \vdash e_1 \Downarrow v_1, H_1, F_1 \quad V' = V\{z \mapsto (l, \mathbf{dead}) \mid z \in FV(e_1) \wedge V(z) \not\models (l, \_)\} \quad V'' = V'[x \mapsto (v_1, \mathbf{alive})] \quad R'' = R \cup \text{locs}_{V'',H_1}(e_2) \quad g = \{l \in H_1 \mid l \notin R'' \cup F_1\} \quad V'', H_1, R, F_1 \cup g \vdash e_2 \Downarrow v_2, H_2, F_2}{V, H, R, F \vdash \mathbf{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2, F_2} (S_{16})
\end{array}$$

### 3 Operational semantics

In order to prove the soundness of the type system, we also define a simplified operational semantics that does not account for garbage collection.

$$\boxed{V, H \vdash e \Downarrow v, H'}$$

This can be read as: under stack  $V$ , heap  $H$  the expression  $e$  evaluates to  $v$ , and engenders a new heap  $H'$ . We write the representative rules.

$$\frac{v = \langle V(x_1), V(x_2) \rangle \quad H', l = \text{copy}(H, L, v)}{V, H \vdash \text{cons}(x_1; x_2) \Downarrow l, H'} \text{(S}_{17}\text{)}$$

$$\frac{\begin{array}{l} V(x) = (l, \text{alive}) \quad H(l) = \langle v_h, v_t \rangle \quad V' = V\{x \mapsto (l, \text{dead})\} \\ V'' = V'[x_h \mapsto (v_h, \text{alive}), x_t \mapsto (v_t, \text{alive})] \quad V'', H \vdash e_2 \Downarrow v, H' \end{array}}{V, H \vdash \text{case } x \{ \text{nil} \hookrightarrow e_1 \mid \text{cons}(x_h; x_t) \hookrightarrow e_2 \} \Downarrow v, H'} \text{(S}_{18}\text{)}$$

$$\frac{V, H \vdash e_1 \Downarrow v_1, H_1 \quad V' = V[x \mapsto v_1] \quad V', H_1 \vdash e_2 \Downarrow v_2, H_2}{V, H \vdash \text{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2} \text{(S}_{19}\text{)}$$

### 4 Type rules

The type system takes into account of garbage collected cells by returning potential locally in a match construct. Since we are interested in the number of heap cells, all constants are assumed to be nonnegative.

$$\begin{array}{c}
\frac{n \in \mathbb{Z}}{\Sigma; \emptyset \mid \frac{q}{q} n : \text{nat}} (\text{L:ConstI}) \quad \frac{}{\Sigma; \emptyset \mid \frac{q}{q} () : \text{unit}} (\text{L:ConstU}) \quad \frac{}{\Sigma; \emptyset \mid \frac{q}{q} \text{T} : \text{bool}} (\text{L:ConstT}) \\
\\
\frac{}{\Sigma; \emptyset \mid \frac{q}{q} \text{F} : \text{bool}} (\text{L:ConstF}) \quad \frac{}{\Sigma; x : B \mid \frac{q}{q} x : B} (\text{L:Var}) \\
\\
\frac{\Sigma; \Gamma \mid \frac{q}{q'} e_t : B \quad \Sigma; \Gamma \mid \frac{q}{q'} e_f : B}{\Sigma; \Gamma, x : \text{bool} \mid \frac{q}{q'} \text{if } x \text{ then } e_t \text{ else } e_f : B} (\text{L:Cond}) \\
\\
\frac{}{\Sigma; x_1 : A_1, x_2 : A_2 \mid \frac{q}{q} \langle x_1, x_2 \rangle : A_1 \times A_2} (\text{L:Pair}) \\
\\
\frac{\Sigma; \Gamma, x_1 : A_1, x_2 : A_2 \mid \frac{q}{q'} e : B}{\Sigma; \Gamma, x : (A_1, A_2) \mid \frac{q}{q'} \text{case } x \{ (x_1; x_2) \hookrightarrow e \} : B} (\text{L:MatP}) \quad \frac{}{\Sigma; \emptyset \mid \frac{q}{q} \text{nil} : L^p(A)} (\text{L:Nil}) \\
\\
\frac{}{\Sigma; \Gamma, x_h : A, x_t : L^p(A) \mid \frac{q+p+1}{q} \text{cons}(x_h; x_t) : L^p(A)} (\text{L:Cons}) \\
\\
\frac{\Sigma; \Gamma \mid \frac{q}{q'} e_1 : B \quad \Sigma; \Gamma, x_h : A, x_t : L^p(A) \mid \frac{q+p+1}{q'} e_2 : B}{\Sigma; \Gamma, x : L^p(A) \mid \frac{q}{q'} \text{case } x \{ \text{nil} \hookrightarrow e_1 \mid \text{cons}(x_h; x_t) \hookrightarrow e_2 \} : B} (\text{L:MatL}) \\
\\
\frac{\Sigma; \Gamma_1 \mid \frac{q}{p} e_1 : A \quad \Sigma; \Gamma_2, x : A \mid \frac{p}{q'} e_2 : B}{\Sigma; \Gamma_1, \Gamma_2 \mid \frac{q}{q'} \text{let}(e_1; x : \tau.e_2) : B} (\text{L:Let})
\end{array}$$

Now if we take  $\dagger : L^p(A) \mapsto L(A)$  as the map that erases resource annotations, we obtain a simpler typing judgement  $\boxed{\Sigma^\dagger; \Gamma^\dagger \vdash e : B^\dagger}$ .

## 5 Soundness for garbage collection semantics

We simplify the soundness proof of the type system for the general metric to one with monotonic resource. (No function types for now)

**Lemma 1.1.** *If  $\Sigma; \Gamma \mid \frac{q}{q'} e : B$ , then  $\Sigma^\dagger; \Gamma^\dagger \vdash e : B^\dagger$ .*

**Lemma 1.2.** *If  $V, H, R, F \vdash e \Downarrow v, H', F'$ , then  $\forall x \in V, \text{reach}_H(V(x)) = \text{reach}_{H'}(V(x))$ .*

*Proof.* Induction on the evaluation judgement. □

**Lemma 1.3.** *For all stacks  $V$  and heaps  $H$ , if  $V, H, R, F \vdash e \Downarrow v, H', F'$ ,  $\Sigma; \Gamma \vdash e : B^\dagger$ ,  $H \models V : \Gamma$ ,  $\text{no\_alias}(FV(e) \triangleleft V, H)$ ,  $\text{safe}(FV(e) \triangleleft V, H, F)$ , and  $\text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$ , then  $\text{set}(\text{reach}_{H'}(v))$ ,  $\text{disjoint}(\{R, F', \text{reach}_{H'}(v)\})$ , and  $\text{no\_alias}(FV(e) \triangleleft V, H')$ .*

*Proof.* Nested induction on the evaluation judgement and the typing judgement.

**Case 1: E:Var**

Suppose  $H \models V : \Gamma, \text{no\_alias}(V, H), \text{safe}(FV(e) \triangleleft V, H, F), \text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$   
 $\text{set}(\text{reach}_H(v))$  (no\_alias( $V, H$ ))  
 $\text{disjoint}(\{R, F, \text{reach}_H(v)\})$  (disjoint( $\{R, F, \text{locs}_{V,H}(e)\}$ ))  
 $\text{no\_alias}(V, H)$  (Sp.)  
 NTS  $\text{no\_ref}(FV(e) \trianglelefteq V, H, v)$   
 Let  $x' \in \text{dom}(\overline{FV(e)} \trianglelefteq \overline{V})$  be arb.  
 $x' \in \text{dom}(\overline{\{x\}} \trianglelefteq \overline{V})$   
 $x' \neq x$   
 $x' \in \overline{V}$   
 $x \in \overline{V}$  (Ad.)  
 $\text{reach}_H(V(x')) \cap \text{reach}_H(V(x)) = \emptyset$  (no\_alias( $V, H$ ))  
 Thus  $\text{no\_ref}(FV(e) \trianglelefteq V, H, v)$

**Case 2: E:Const\*** Due to similarity, we show only for E:ConstI

Suppose  $H \models V : \Gamma, \text{no\_alias}(V, H), \text{safe}(FV(e) \triangleleft V, H, F), \text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$   
 $\text{set}(\text{reach}_H(v))$  ( $\text{reach}_H(v) = \emptyset$ )  
 $\text{disjoint}(\{R, F, \emptyset\})$  (disjoint( $R, F$ ))  
 $\text{no\_ref}(FV(e) \trianglelefteq V, H, v)$  ( $\text{reach}_H(v) = \emptyset$ )  
 $\text{no\_alias}(V, H)$

**Case 4: E:App**

**Case 5: E:CondT** Similar to E:MatNil

**Case 6: E:CondF** Similar to E:CondT

**Case 7: E:Let**

**Case 8: E:Pair** Similar to E:Var

**Case 9: E:MatP** Similar to E:MatCons

**Case 10: E:Nil** Similar to E:Const\*

**Case 11: E:Cons**

$V, H, R, F \vdash e \Downarrow l, H'', F'$  (case)  
 Suppose  $H \models V : \Gamma, \text{no\_alias}(V, H), \text{safe}(FV(e) \triangleleft V, H, F), \text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$



$$\begin{array}{ll}
\text{set}(V^*(x_1)), \text{set}(V^*(x_2)) & (\text{no\_alias}(V, H)) \\
\text{set}(\text{reach}_H(v)) & (\text{def of set}) \\
\text{set}(\text{reach}_H''(v)) & (\text{Lemma 1.2}) \\
l \notin \text{reach}_H''(V) & (l \in F \text{ and } \text{safe}(V, H, F)) \\
\text{set}(\text{reach}_H''(l)) & (\text{def of set and } \text{reach}) \\
\text{NTS disjoint}(\{R, F', \text{reach}_{H''}(l)\}) & \\
R \cap F' = \emptyset & (F' \subseteq F) \\
l \notin R & (l \in F) \\
R \cap \text{reach}_{H''}(v) = \emptyset & (R \cap \text{locs}_{V,H}(e) = \emptyset) \\
R \cap \text{reach}_{H''}(l) = \emptyset & (\text{def of } \text{reach}) \\
l \notin F' & (\text{similar}) \\
\text{Thus disjoint}(\{R, F', \text{reach}_{H''}(l)\}) & \\
\text{NTS no\_ref}(FV(e) \trianglelefteq V, H', l) & \\
\text{Let } x \in \overline{FV(e) \trianglelefteq V} \text{ be arb.} & \\
l \notin \text{reach}_{H''}(V^*(x)) & (\text{safe}(V, H, F)) \\
\text{reach}_{H''}(V^*(x)) \cap \text{reach}_{H''}(v) = \emptyset & (x \notin \{x_1, x_2\} \text{ and } \text{no\_alias}(V, H)) \\
\text{Thus no\_ref}(FV(e) \trianglelefteq V, H', l) & \\
\text{no\_alias}(V, H'') & (\text{Lemma 1.2})
\end{array}$$

### Case 12: E:MatNil

$$\begin{array}{ll}
\text{Suppose } H \models V : \Gamma, \text{no\_alias}(V, H), \text{safe}(FV(e) \triangleleft V, H, F) \text{ disjoint}(\{R, F, \text{locs}_{V,H}(e)\}) & \\
\Sigma; \Gamma' \vdash e_1 : B & (\text{ad.}) \\
V, H, R, F \cup g \vdash e_1 \Downarrow v, H', F' & (\text{ad.}) \\
H \models V : \Gamma' & (\text{def of W.D.E}) \\
\text{WTS } \text{safe}(FV(e_1) \triangleleft V, H, F \cup g) & \\
\text{Let } x' \in \overline{FV(e_1) \triangleleft V} \text{ be arb.} & \\
\text{reach}_H(V^*(x)) \cap F = \emptyset & (FV(e_1) \subseteq FV(e) \text{ and } \text{safe}(FV(e) \triangleleft V, H, F)) \\
\text{reach}_H(V^*(x)) \cap g = \emptyset & (\text{def of } g) \\
\text{Thus } \text{safe}(FV(e_1) \triangleleft V, H, F \cup g) & \\
\text{disjoint}(\{R, F \cup g, \text{locs}_{V,H}(e_1)\}) & (\text{def of } g \text{ and } \text{locs}_{V,H}(e_1) \subseteq \text{locs}_{V,H}(e)) \\
\text{set}(\text{reach}_{H'}(v)), \text{disjoint}(\{R, F', \text{reach}_{H'}(v)\}), \text{no\_ref}(FV(e_1) \trianglelefteq V, H', v) \text{no\_alias}(V, H') & (\text{IH}) \\
\text{no\_ref}(FV(e) \trianglelefteq V, H', v) & (FV(e_1) \subseteq FV(e))
\end{array}$$

### Case 13: E:MatCons

$$V(x) = (l, \text{alive}) \quad (\text{ad.})$$

$H(l) = \langle v_h, v_t \rangle$  (ad.)  
 $\Gamma = \Gamma', x : L(A)$  (ad.)  
 $\Sigma; \Gamma', x_h : A, x_t : L(A) \vdash e_2 : B$  (ad.)  
 $V'', H, R, F \cup g \vdash e_2 \Downarrow v_2, H_2, F'$  (ad.)  
 Suppose  $H \models V : \Gamma, \text{no\_alias}(V, H)$ , and  $\text{disjoint}(\{F, R, \text{locs}_{V,H}(e)\})$   
 $H \models V(x) : L(A)$  (def of W.D.E)  
 $H'' \models v_h : A, H'' \models v_t : L(A)$  (ad.)  
 $H \models v_h : A, H \models v_t : L(A)$  (???)  
 $H \models V'' : \Gamma', x_h : A, x_t : L(A)$  (def of W.D.E)  
 NTS  $\text{no\_alias}(V'', H)$   
 Let  $x_1, x_2 \in \bar{V}'', x_1 \neq x_2, r_{x_1} = \text{reach}_H(\bar{V}''(x_1)), r_{x_2} = \text{reach}_H(\bar{V}''(x_2))$   
**case:**  $x_1 \notin \{x_h, x_t\}, x_2 \notin \{x_h, x_t\}$   
 (1), (2) from  $\text{no\_alias}(V, H)$   
**case:**  $x_1 = x_h, x_2 \notin \{x_h, x_t\}$   
 $\text{set}(r_{x_1})$  (since  $\text{set}(H(l))$  from  $\text{no\_alias}(V, H)$ )  
 $\text{set}(r_{x_2})$  (since  $\text{no\_alias}(V, H)$ )  
 AFSOC, suppose  $l' \in r_{x_1} \cap r_{x_2}$   
 but  $\text{reach}_H(\bar{V}(x)) \cap r_{x_2} = \emptyset$ , contradiction (def of  $\text{reach}$ )  
 hence  $r_{x_1} \cap r_{x_2} = \emptyset$   
**case:**  $x_1 = x_h, x_2 = x_t$   
 $\text{set}(r_{x_1})$  since  $\text{set}(H(l))$  from  $\text{no\_alias}(V, H)$   
 $\text{set}(r_{x_2})$  since  $\text{set}(H(l))$  from  $\text{no\_alias}(V, H)$   
 AFSOC, suppose  $l' \in r_{x_1} \cap r_{x_2}$   
 but then  $\mu_{\text{reach}_H(l)}(l') \geq 2$ , and  $\text{set}(H(l))$  does not hold.  
 hence  $r_{x_1} \cap r_{x_2} = \emptyset$   
**case: otherwise**  
 similar to the above  
 Thus we have  $\text{no\_alias}(V'', H)$   
 $(F \cup g) \cap R = \emptyset$  (since  $F \cap R = \emptyset$  and by def of  $g$ )  
 NTS  $R \cap \text{locs}_{V'',H}(e_2) = \emptyset$   
 Let  $l' \in \text{locs}_{V'',H}(e_2)$  be arb.  
**case:**  $l' \in \text{reach}_H(V''(x'))$  for some  $x' \in FV(e_2)$  where  $x' \notin \{x_h, x_t\}$   
 $x' \in V$  (def of  $V''$ )  
 $l' \in \text{reach}_H(V(x'))$   
 $x' \in FV(e)$  (def of  $FV$ )  
 $l' \in \text{locs}_{V,H}(e)$  (def of  $\text{locs}_{V,H}$ )  
 $l' \notin R$  ( $\text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$ )

**case:**  $l' \in reach_H(V''(x_h))$

$l' \in reach_H(v_h)$

$l' \in reach_H(V^*(x))$

(def of *reach*)

$l' \in locs_{V,H}(e)$

(def of  $locs_{V,H}$ )

$l' \notin R$

(since  $disjoint(\{F, R, locs_{V,H}(e)\})$ )

**case:**  $l' \in reach_H(V''(x_t))$

similar to above

Hence  $R \cap locs_{V'',H}(e_2) = \emptyset$

$F \cap locs_{V'',H}(e_2) = \emptyset$

(Similar to above)

$g \cap locs_{V'',H}(e_2) = \emptyset$

(def. of  $g$ )

$(F \cup g) \cap locs_{V'',H}(e_2) = \emptyset$

Thus  $disjoint(\{R, F \cup g, locs_{V'',H}(e_2)\})$

By IH,  $set(reach_{H'}(v))$ ,  $disjoint(\{R, F', reach_{H'}(v)\})$ ,  $no\_ref(V'', H', v)$ , and  $no\_alias(V'', H')$

NTS  $no\_ref(V, H', v)$

Let  $l' \in reach_{H'}(\overline{V}(x))$  be arb

$l' \in reach_H(l)$

(Lemma 1.2, ad.)

Then  $l' \in reach_{H'}(v_h)$  or  $l' \in reach_{H'}(v_t)$

(def of *reach*)

Wlog  $l' \in reach_{H'}(v_h)$

$l' \in reach_{H'}(V''(x_h))$

(def of  $V''$ )

$l' \notin reach_{H'}(v)$

( $no\_ref(V'', H', v)$ )

$(reach_{H'}(v)) \cap (\bigcup_{x' \in \overline{V} \setminus x} reach_{H'}(V(x'))) = \emptyset$

( $no\_ref(V'', H', v)$ )

$(reach_{H'}(v)) \cap (\bigcup_{x' \in \overline{V}} reach_{H'}(V(x'))) = \emptyset$

$no\_ref(V, H', v)$

NTS  $no\_alias(V, H')$

Let  $x_1, x_2 \in \overline{V}, x_1 \neq x_2, r_{x_1} = reach_H(\overline{V}(x_1)), r_{x_2} = reach_H(\overline{V}(x_2))$  be arb.

**case:**  $x_1 \neq x, x_2 \neq x$

(1), (2)

( $no\_alias(V'', H')$ )

**case:**  $x_1 = x, x_2 \neq x$

$set(r_{x_1})$

( $no\_alias(V'', H')$ )

$set(r_{x_2})$

( $no\_alias(V'', H')$ )

**case: otherwise**

similar to above

Thus  $no\_alias(V, H')$

Thus  $no\_ref(V, H', v)$  and  $no\_alias(V, H')$

□

**Task 1.4** (Soundness). *let  $H \models V : \Gamma, \Sigma; \Gamma \mid_{q'}^q e : B$ , and  $V, H \vdash e \Downarrow v, H'$ . Then  $\forall C \in \mathbb{Q}^+$  and  $\forall F, R \subseteq \text{Loc}$ , if  $\text{no\_alias}(V, H)$ ,  $\text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$ , and  $|F| \geq \Phi_{V,H}(\Gamma) + q + C$ , then there exists  $F' \subseteq \text{Loc}$  s.t.*

1.  $V, H, R, F \vdash e \Downarrow v, H', F'$
2.  $|F'| \geq \Phi_{H'}(v : B) + q' + C$

*Proof.* Induction on the evaluation judgement.

### Case 1: E:Var

$$\begin{aligned}
V, H, R, F &\vdash x \Downarrow V(x), H, F && \text{(admissibility)} \\
\Sigma; x : B &\mid_{q'}^q x : B && \text{(admissibility)} \\
|F| - |F'| &&& (1) \\
&= |F| - |F| && \text{(ad.)} \\
&= 0 && (2) \\
\Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') &&& (3) \\
&= \Phi_{V,H}(x : B) + q - (\Phi_H(V(x) : B) + q) && \text{(ad.)} \\
&= \Phi_H(V(x) : B) + q - (\Phi_H(V(x) : B) + q) && \text{(def. of } \Phi_{V,H}) \\
&= 0 && (4) \\
|F| - |F'| &\leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') && ((3),(5))
\end{aligned}$$

**Case 2: E:Const\*** Due to similarity, we show only for E:ConstI

$$\begin{aligned}
|F| - |F'| &= |F| - |F| && \text{(ad.)} \\
&= 0 \\
\Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') &= \Phi_{V,H}(\emptyset) + q - (\Phi_H(v : \text{int}) + q) && \text{(ad.)} \\
&= 0 && \text{(def of } \Phi_{V,H}) \\
|F| - |F'| &\leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q')
\end{aligned}$$

### Case 4: E:App

### Case 5: E:CondT

$$\begin{aligned}
\Gamma &= \Gamma', x : \text{bool} && \text{(ad.)} \\
H &\models V : \Gamma' && \text{(def of W.F.E)} \\
\Sigma; \Gamma' &\mid_{q'}^q e_t : B && \text{(ad.)} \\
V, H, R, F \cup g &\vdash e_t \Downarrow v, H', F' && \text{(ad.)} \\
|F \cup g| - |F'| &\leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') && \text{(IH)} \\
|F| - |F'| &\leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q')
\end{aligned}$$

**Case 6: E:CondF** Similar to E:CondT

**Case 7: E:Let**

$$V, H \vdash e \Downarrow v_2, H_2 \quad (\text{case})$$

$$V, H \vdash e_1 \Downarrow v_1, H_1 \quad (\text{ad.})$$

$$\Sigma; \Gamma_1 \mid_p^q e_1 : A \quad (\text{ad.})$$

$$H \models V : \Gamma_1 \quad (\Gamma_1 \subseteq \Gamma)$$

Let  $C \in \mathbb{Q}^+$ ,  $F, R \subseteq \text{Loc}$  be arb.

Suppose  $\text{no\_alias}(V, H)$ ,  $\text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$ , and  $|F| \geq \Phi_{V,H}(\Gamma) + q + C$

NTF  $F'$  s.t.

$$1. V, H, R, F \vdash e \Downarrow v_2, H_2, F' \text{ and}$$

$$2. |F'| \geq \Phi_{H_2}(v_2 : B) + q' + C$$

Let  $R' = R \cup \text{locs}_{V,H}(\text{lam}(x : \tau.e_2))$

$\text{disjoint}(\{R', F, \text{locs}_{V,H}(e_1)\})$  (Similar to case in Lemma 1.2)

Instantiate IH with  $C = C + \Phi_{V,H}(\Gamma_2)$ ,  $F = F$ ,  $R = R'$ , we get  $F''$  s.t.

$$1. V, H, R', F \vdash e_1 \Downarrow v_1, H_1, F'' \text{ and}$$

$$2. |F''| \geq \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V',H_1}(\Gamma_2)$$

Where  $|F| \geq \Phi_{V,H}(\Gamma_1) + q + C + \Phi_{V,H}(\Gamma_2)$  since  $|F| \geq \Phi_{V,H}(\Gamma) + q + C$

For the second premise:

$$\Sigma; \Gamma_2, x : A \mid_{q'}^p e_2 : B \quad (\text{ad.})$$

$$H_1 \models v_1 : A \text{ and} \quad (\text{Theorem 3.3.4})$$

$$H_1 \models V : \Gamma_2 \quad (???)$$

$$H_1 \models V' : \Gamma_2, x : A \quad (\text{def of } \models)$$

$$V', H_1 \vdash e_2 \Downarrow v_2, H_2 \quad (\text{ad.})$$

Let  $g = \{l \in H_1 \mid l \notin F_1 \cup R \cup \text{locs}_{V',H_1}(e_2)\}$

Then we have  $\text{no\_alias}(V', H_1)$  and  $\text{disjoint}(\{R, F'' \cup g, \text{locs}_{V',H_1}(e_2)\})$   
(similar to case in Lemma 1.2)

Instantiate IH with  $C = C$ ,  $F = F'' \cup g$ ,  $R = R$ , we get  $F^{(3)}$  s.t.

$$1. V', H_1, R, F'' \cup g \vdash e_2 \Downarrow v_2, H_2, F^{(3)}$$

$$2. |F^{(3)}| \geq \Phi_{H_2}(v_2 : B) + q' + C$$

Where we verify the precondition  $|F'' \cup g| \geq \Phi_{V',H_1}(\Gamma_2, x : A) + p + C$

$$\begin{aligned} |F'' \cup g| &\geq |F''| \\ &\geq \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V,H}(\Gamma_2) \end{aligned} \quad (\text{IH})$$

$$= \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V',H_1}(\Gamma_2) \quad (\text{Lemma 4.3.3})$$

$$= \Phi_{V',H_1}(\Gamma_2, x : A) + p + C \quad (\text{def of } \Phi)$$

Take  $F' = F^{(3)}$

$$V, H, R, F \vdash e \Downarrow v_2, H_2, F' \text{ and} \quad (\text{E:Let})$$

$$|F'| \geq \Phi_{H_2}(v_2 : B) + q' + C \quad (\text{from IH})$$

**Case 8: E:Pair** Similar to E:Const\*

**Case 9: E:MatP** Similar to E:MatCons

**Case 10: E:Nil** Similar to E:Const\*

**Case 11: E:Cons**

$$\begin{aligned}
& |F| - |F'| \\
&= |F| - |F \setminus \{l\}| \quad (\text{ad.}) \\
&= 1 \\
&\Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') \\
&= \Phi_{V,H}(x_h : A, x_t : L^p(A)) + q + p + 1 - (\Phi_{H'}(v : L^p(A)) + q) \quad (\text{ad.}) \\
&= \Phi_{V,H}(x_h : A, x_t : L^p(A)) + p + 1 - \Phi_{H'}(v : L^p(A)) \\
&= \Phi_H(V(x_h) : A) + \Phi_H(V(x_t) : L^p(A)) + p + 1 - \Phi_{H'}(v : L^p(A)) \quad (\text{def of } \Phi_{V,H}) \\
&= \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A)) + p + 1 - \Phi_{H'}(v : L^p(A)) \quad (\text{ad.}) \\
&= \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A)) + p + 1 - (p + \Phi_{H'}(v_h : A) + \Phi_{H'}(v_t : L^p(A))) \\
&\quad (\text{Lemma 4.1.1}) \\
&= \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A)) + p + 1 - (p + \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A))) \\
&\quad (\text{Lemma 4.3.3}) \\
&= 1
\end{aligned}$$

Hence,

$$|F| - |F'| \leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q')$$

**Case 12: E:MatNil** Similar to E:Cond\*

**Case 13: E:MatCons**

$$\begin{aligned}
V(x) &= (l, \text{alive}) \quad (\text{ad.}) \\
H(l) &= \langle v_h, v_t \rangle \quad (\text{ad.}) \\
\Gamma &= \Gamma', x : L^p(A) \quad (\text{ad.}) \\
\Sigma; \Gamma', x_h : A, x_t : L^p(A) &\Big|_{\frac{q+p+1}{q'}} e_2 : B \quad (\text{ad.}) \\
V'', H &\vdash e_2 \Downarrow v, H' \quad (\text{ad.}) \\
\text{Let } C \in \mathbb{Q}^+, F, R &\subseteq \text{Loc be arb.} \\
H &\models V(x) : L^p(A) \quad (\text{def of W.D.E}) \\
H'' &\models v_h : A, H'' \models v_t : L^p(A) \quad (\text{ad.}) \\
H &\models v_h : A, H \models v_t : L^p(A) \quad (???) \\
H &\models V'' : \Gamma', x_h : A, x_t : L^p(A) \quad (\text{def of W.D.E}) \\
\text{Suppose } \text{no.alias}(V, H), \text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\}), &\text{ and } |F| \geq \Phi_{V,H}(\Gamma) + q + C
\end{aligned}$$

NTF  $F'$  s.t.

1.  $V, H, R, F \vdash e \Downarrow v, H', F'$  and

2.  $|F'| \geq \Phi_{H'}(v : B) + q' + C$

Let  $g = \{l \in H \mid l \notin F \cup R \cup \text{locs}_{V'',H}(e_2)\}$

We want to  $g$  nonempty, in particular, that  $l \in g$

$l \notin F \cup R$

( $\text{disjoint}(\{R, F, \text{locs}_{V,H}(e)\})$ )

AFSOC  $l \in \text{locs}_{V'',H}(e_2)$

Then  $l \in \text{reach}_H(\bar{V}''(x'))$  for some  $x' \neq x$

$x' \in \{x_h, x_t\}$  (since  $\text{reach}_H(\bar{V}(x')) \cap \text{reach}_H(\bar{V}(x)) = \emptyset$  from  $\text{no\_alias}(V, H)$ )

WLOG let  $x' = x_h$

But then  $\mu_{\text{reach}_H(\bar{V}(x))}(l) \geq 2$  and  $\text{set}(\text{reach}_H(\bar{V}(x)))$  doesn't hold

$l \notin \text{locs}_{V'',H}(e_2)$

Hence  $l \in g$

Next, we have  $\text{no\_alias}(V'', H)$  and  $\text{disjoint}(\{R, F \cup g, \text{locs}_{V'',H}(e_2)\})$

(similar to case in Lemma 1.2)

By IH with  $C' = C, F'' = F \cup g$  and the above conditions, we have:  $F^{(3)}$  s.t.

1.  $V'', H, R, F \cup g \vdash e_2 \Downarrow v, H', F^{(3)}$

2.  $|F^{(3)}| \geq \Phi_{H'}(v : B) + q' + C$

Where we also verify the precondition that  $|F''| \geq \Phi_{V'',H}(\Gamma', x_h : A, x_t : L^p(A)) + q + p + 1 + C' :$

$|F''| = |F \cup g|$

$= |F| + |g|$

( $F$  and  $g$  disjoint)

$\geq \Phi_{V,H}(\Gamma) + q + C + |g|$

(Sp.)

$= \Phi_{V,H}(\Gamma', x_h : A, x_t : L^p(A)) + p + q + C + |g|$

(Lemma 4.1.1)

$= \Phi_{V,H}(\Gamma', x_h : A, x_t : L^p(A)) + p + q + C + 1$

( $g$  nonempty)

Now take  $F' = F^{(3)}$

$V, H, R, F \vdash e \Downarrow v, H', F'$

(E:MatCons)

$|F'| \geq \Phi_{H'}(v : B) + q' + C$

(From the IH)

□