15-312 Assignment 1

Andrew Carnegie
(andrew)

November 7, 2017

1 Introduction

In this paper, we propose a model for deriving asymptotically tight bounds for first order functional programs. We choose a fragment of OCaml as the target language. The abstract and concrete syntax of the language is show below. Note that we only allow first order functions of type $\tau_1 to \tau_2$, where τ_1 and τ_2 are base types: unit, bool, product, or lists.

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

2 Paths and aliasing

Model dynamics using judgement of the form:

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

Where $V : \mathsf{Var} \to \mathsf{Val} \times \mathsf{State}$, $H : \mathsf{Loc} \to \mathsf{Val}$, $R \subseteq \mathsf{Loc}$, and $F \subseteq \mathsf{Loc}$. This can be read as: under stack V, heap H, roots R, freelist F, and signature Σ, the expression e evaluates to v, and engenders a new heap H' and freelist F'. Because the signature Σ for the set of the first order functions does not change during evaluation, we drop the subscript Σ from $\vdash_\Sigma$ when the context of evaluation is clear. It is convenient to think of the evaluation judgement $\vdash$ as being indexed by a family of signatures Σ's, each of which is a set of "top-level" first-order declarations to be used during evaluation.

For a partial map $f : A \to B$, we write *dom* for the defined values of f. Sometimes we shorten $x \in dom(f)$ 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 $x \notin dom(f)$.

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 $reach : Val \to \{\{\mathsf{Loc}\}\}$ that maps stack values its the root *multiset*, the multiset of locations that's already on the stack.
Next we define reachability of values:

$$\begin{aligned}
&reach_H(\langle v_1, v_2\rangle) = reach_H(v_1) \uplus reach_H(v_2)\\
&reach_H(l) = \{l\} \uplus reach_H(H(l))\\
&reach_H(_) = \emptyset
\end{aligned}$$

For a multiset S, we write $\mu_S : S \to \mathbb{N}$ for the multiplicity function of S, which maps each element to the count of its occurence. If $\mu_S(x) \geq 1$ for a multiset S, then we write $x \in S$ as in the usual set membership relation. If for all $s \in S$, $\mu(s) = 1$, then S is a property set, and we denote it by $\mathsf{set}(S)$. Addtionally, $A \uplus B$ 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, stable, and disjoint:

no_alias(V, H): $\forall x, y \in V,\ x \neq y$. Let $r_x = reach_H(V(x))$, $r_y = reach_H(V(y))$. Then:

1. $\mathsf{set}(r_x), \mathsf{set}(r_y)$
2. $r_x \cap r_y = \emptyset$

stable(R, H, H'): $\forall l \in R.\ H(l) = H'(l)$.

safe(V, H, F): $\forall x \in V.\ reach_H(V(x)) \cap F = \emptyset$

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

For a stack V and a heap H, whenever $\mathsf{no\_alias}(V, H)$ 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.

First, we define $FV(e)$, the multiset of free variables of e. It is defined inductively over the structure of e; the only unusual thing is that multiple occurences of a free variable x in e will be reflected in the multiplicity of $FV(e)$.

Next, we define $locs_{V,H}$ using the previous notion of reachability.

$$locs_{V,H}(e) = \bigcup_{x \in FV(e)} reach_H(V(x))$$

$size$ calculates the number of cells a value occupies.

$$\begin{aligned} &size(\langle v_1, v_2 \rangle) = size(v_1) + size(v_2) \\ &size(\_) = 1 \end{aligned}$$

$copy(H, L, v)$ 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'.

$$\begin{aligned} &copy(H, L, \langle v_1, v_2 \rangle) = \\ &\quad \mathsf{let}\ L_1 \sqcup L_2 \subseteq L \\ &\qquad \mathsf{where}\ |L_1| = size(v_1)\ , |L_2| = size(v_2) \\ &\quad \mathsf{let}\ H_1 = copy(H, L_1, v_1) \\ &\quad \mathsf{let}\ H_2 = copy(H_1, L_2, v_2)\ \mathsf{in} \\ &\quad H_2\{l \mapsto v\} \\ &copy(H, L, v) = \\ &\quad \mathsf{let}\ l \in H\ \mathsf{in} \\ &\quad H\{l \mapsto v\} \end{aligned}$$

3 Garbage collection semantics

$$\frac{V(x) = v}{V, H, R, F \vdash x \Downarrow v, H, F}(\mathrm{S}_1) \qquad \frac{}{V, H, R, F \vdash \overline{n} \Downarrow \mathtt{val}(n), H, F}(\mathrm{S}_2)$$

$$\frac{}{V, H, R, F \vdash \mathtt{T} \Downarrow \mathtt{val}(\mathtt{T}), H, F}(\mathrm{S}_3) \qquad \frac{}{V, H, R, F \vdash \mathtt{F} \Downarrow \mathtt{val}(\mathtt{F}), H, F}(\mathrm{S}_4)$$

$$\frac{}{V, H, R, F \vdash () \Downarrow \mathtt{val}(\mathtt{Null}), H, F}(\mathrm{S}_5)$$

$$\frac{V = V'[x \mapsto \mathtt{T}] \qquad g = \{l \in H \mid l \notin F \cup R \cup \mathit{locs}_{V,H}(e_1)\} \qquad V', H, R, F \cup g \vdash e_1 \Downarrow v, H', F'}{V, H, R, F \vdash \mathtt{if}(x; e_1; e_2) \Downarrow v, H', F'}(\mathrm{S}_6)$$

$$\frac{V = V'[x \mapsto \mathtt{F}] \qquad g = \{l \in H \mid l \notin F \cup R \cup \mathit{locs}_{V,H}(e_2)\} \qquad V', H, R, F \cup g \vdash e_2 \Downarrow v, H', F'}{V, H, R, F \vdash \mathtt{if}(x; e_1; e_2) \Downarrow v, H', F'}(\mathrm{S}_7)$$

$$\frac{V = V'[x \mapsto v'] \qquad V'[y_f \mapsto v'], H, R, F \vdash e_f \Downarrow v, H', F'}{V, H, R, F \vdash f(x) \Downarrow v, H', F'}(\mathrm{S}_8)$$

$$\frac{V(x_1) = v_1 \qquad V(x_2) = v_2}{V, H, R, F \vdash \langle x_1, x_2 \rangle \Downarrow \langle v_1, v_2 \rangle, H, F}(\mathrm{S}_9)$$

$$\frac{\begin{array}{c} V = V'[x \mapsto \langle v_1, v_2 \rangle] \qquad g = \{l \in H \mid l \notin F \cup R \cup \mathit{locs}_{V,H}(e)\} \\ V'' = V'[x_1 \mapsto v_1, x_2 \mapsto v_2] \qquad V'', H, R, F \cup g \vdash e \Downarrow v, H', F' \end{array}}{V, H, R, F \vdash \mathtt{case}\ x\ \{(x_1; x_2) \hookrightarrow e\} \Downarrow v, H', F'}(\mathrm{S}_{10})$$

$$\frac{}{V, H, R, F \vdash \mathtt{nil} \Downarrow \mathtt{val}(\mathtt{Null}), H, F}(\mathrm{S}_{11})$$

$$\frac{g = \mathit{reach}_H(v) \qquad V, H, R, F \cup g \vdash e \Downarrow v, H', F'}{V[x \mapsto v], H, R, F \vdash \mathtt{drop}(x; e) \Downarrow v, H', F'}(\mathrm{S}_{12})$$

$$\frac{\begin{array}{c} v = \langle V(x_1), V(x_2) \rangle \qquad L \sqcup \{l\} \subseteq F \\ |L| = \mathit{size}_H(v) \qquad F' = F \setminus (L \sqcup \{l\}) \qquad H' = \mathit{copy}(H, L, v) \qquad H'' = H'\{l \mapsto v\} \end{array}}{V, H, R, F \vdash \mathtt{cons}(x_1; x_2) \Downarrow l, H'', F'}(\mathrm{S}_{13})$$

$$\frac{\begin{array}{c} V(x) = \mathtt{Null} \qquad V' \subseteq V \qquad \mathit{dom}(V') = \mathit{FV}(e_1) \\ g = \{l \in H \mid l \notin F \cup R \cup \mathit{locs}_{V,H}(e_1)\} \qquad V', H, R, F \cup g \vdash e_1 \Downarrow v, H', F' \end{array}}{V, H, R, F \vdash \mathtt{case}\, x\, \{\mathtt{nil} \hookrightarrow e_1 \mid \mathtt{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H', F'}(\mathrm{S}_{14})$$

$$\frac{\begin{array}{c} V(x) = l \\ H(l) = \langle v_h, v_t \rangle \qquad V' \subseteq V \qquad \mathit{dom}(V') = \mathit{FV}(e_2) \setminus \{x_h, x_t\} \qquad V'' = V'[x_h \mapsto v_h, x_t \mapsto v_t] \\ g = \{l \in H \mid l \notin F \cup R \cup \mathit{locs}_{V'',H}(e_2)\} \qquad V'', H, R, F \cup g \vdash e_2 \Downarrow v, H', F' \end{array}}{V, H, R, F \vdash \mathtt{case}\, x\, \{\mathtt{nil} \hookrightarrow e_1 \mid \mathtt{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H', F'}(\mathrm{S}_{15})$$

$$\frac{\begin{array}{c} V = V_1 \sqcup V_2 \qquad \mathit{dom}(V_1) = \mathit{FV}(e_1) \qquad \mathit{dom}(V_2) = \mathit{FV}(\mathtt{lam}(x : \tau.e_2)) \\ R' = R \cup \mathit{locs}_{V_2,H}(\mathtt{lam}(x : \tau.e_2)) \qquad V_1, H, R', F \vdash e_1 \Downarrow v_1, H_1, F_1 \qquad V_2' = V_2[x \mapsto v_1] \\ g = \{l \in H_1 \mid l \notin F_1 \cup R \cup \mathit{locs}_{V_2',H_1}(e_2)\} \qquad V_2', H_1, R, F_1 \cup g \vdash e_2 \Downarrow v_2, H_2, F_2 \end{array}}{V, H, R, F \vdash \mathtt{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2, F_2}(\mathrm{S}_{16})$$

4 Operational semantics

In order to prove the soundess 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.

$$\frac{v = \langle V(x_1), V(x_2)\rangle \qquad (L \sqcup \{l\}) \cap \mathit{dom}(H) = \emptyset \qquad H', l = \mathit{copy}(H, L, v)}{V, H \ \vdash \texttt{cons}(x_1; x_2) \Downarrow l, H'} (\mathrm{S}_{17})$$

$$\frac{\begin{array}{c} V(x) = l \qquad H(l) = \langle v_h, v_t\rangle \qquad V' \subseteq V \\ \mathit{dom}(V') = FV(e_2) \setminus \{x_h, x_t\} \qquad V'' = V'[x_h \mapsto v_h, x_t \mapsto v_t] \qquad V'', H \ \vdash e_2 \Downarrow v, H' \end{array}}{V, H \ \vdash \texttt{case}\, x\, \{\texttt{nil} \hookrightarrow e_1 \mid \texttt{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H'} (\mathrm{S}_{18})$$

$$\frac{\begin{array}{c} V = V_1 \sqcup V_2 \qquad \mathit{dom}(V_1) = FV(e_1) \qquad \mathit{dom}(V_2) = FV(\texttt{lam}(x : \tau.e_2)) \\ V_1, H \vdash e_1 \Downarrow v_1, H_1 \qquad V_2' = V_2[x \mapsto v_1] \qquad V_2', H_1 \vdash e_2 \Downarrow v_2, H_2 \end{array}}{V, H \ \vdash \texttt{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2} (\mathrm{S}_{19})$$

5 Type rules

The type system takes into account of garbaged 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.

$$\frac{n \in \mathbb{Z}}{\Sigma; \emptyset \mid^{q}_{q} n : \texttt{nat}}\text{(L:ConstI)} \qquad \frac{}{\Sigma; \emptyset \mid^{q}_{q} () : \texttt{unit}}\text{(L:ConstU)} \qquad \frac{}{\Sigma; \emptyset \mid^{q}_{q} \texttt{T} : \texttt{bool}}\text{(L:ConstT)}$$

$$\frac{}{\Sigma; \emptyset \mid^{q}_{q} \texttt{F} : \texttt{bool}}\text{(L:ConstF)} \qquad \frac{}{\Sigma; x : B \mid^{q}_{q} x : B}\text{(L:Var)}$$

$$\frac{\Sigma; \Gamma \mid^{q}_{q'} e_t : B \qquad \Sigma; \Gamma \mid^{q}_{q'} e_f : B}{\Sigma; \Gamma, x : \texttt{bool} \mid^{q}_{q'} \texttt{if}\, x \,\texttt{then}\, e_t \,\texttt{else}\, e_f : B}\text{(L:Cond)}$$

$$\frac{}{\Sigma; x_1 : A_1, x_2 : A_2 \mid^{q}_{q} \langle x_1, x_2 \rangle : A_1 \times A_2}\text{(L:Pair)}$$

$$\frac{\Sigma; \Gamma, x_1 : A_1, x_2 : A_2 \mid^{q}_{q'} e : B}{\Sigma; \Gamma, x : (A_1, A_2) \mid^{q}_{q'} \texttt{case}\, x\, \{(x_1; x_2) \hookrightarrow e\} : B}\text{(L:MatP)} \qquad \frac{}{\Sigma; \emptyset \mid^{q}_{q} \texttt{nil} : L^p(A)}\text{(L:Nil)}$$

$$\frac{}{\Sigma; \Gamma, x_h : A, x_t : L^p(A) \mid^{q+p+1}_{q} \texttt{cons}(x_h; x_t) : L^p(A)}\text{(L:Cons)}$$

$$\frac{\Sigma; \Gamma \mid^{q}_{q'} e_1 : B \qquad \Sigma; \Gamma, x_h : A, x_t : L^p(A) \mid^{q+p+1}_{q'} e_2 : B}{\Sigma; \Gamma, x : L^p(A) \mid^{q}_{q'} \texttt{case}\, x\, \{\texttt{nil} \hookrightarrow e_1 \mid \texttt{cons}(x_h; x_t) \hookrightarrow e_2\} : B}\text{(L:MatL)}$$

$$\frac{\Sigma; \Gamma_1 \mid^{q}_{p} e_1 : A \qquad \Sigma; \Gamma_2, x : A \mid^{p}_{q'} e_2 : B}{\Sigma; \Gamma_1, \Gamma_2 \mid^{q}_{q'} \texttt{let}(e_1; x : \tau.e_2) : B}\text{(L:Let)} \qquad \frac{\Sigma; \Gamma \mid^{q}_{q'} e : B}{\Sigma; \Gamma, x : A \mid^{q}_{q'} \texttt{drop}(x; e) : B}\text{(L:Drop)}$$

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

6 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* $\Sigma; \Gamma \left|\frac{q}{q'}\right. e : B$, *then* $\Sigma^\dagger; \Gamma^\dagger \vdash e : B^\dagger$.

Lemma 1.2. *If* $\Sigma; \Gamma \left|\frac{q}{q'}\right. e : B$, *then* $\mathsf{set}(FV(e))$ *and* $dom(\Gamma) = FV(e)$.

Proof. Induction on the typing judgement. □

Lemma 1.3. *For all values* v*, heaps* H*,* H'*, set of locations* R*, if* $reach_H(v) \subseteq R$ *and* $\mathsf{stable}(R, H, H')$*, then* $reach_H(v) = reach_{H'}(v)$*.*

Proof. Induction on the structure of v. □

Lemma 1.4. *For all stacks* V *and heaps* H*, let* $V, H, R, F \ \vdash e \Downarrow v, H', F'$*,* $\Sigma; \Gamma \vdash e : B$*, and* $H \vDash V : \Gamma$*. Then given the following:*

1. $dom(V) = FV(e)$
2. $\mathsf{no\_alias}(V, H)$*, and*
3. $\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

We have the follwoing:

1. $\mathsf{set}(reach_{H'}(v))$
2. $\mathsf{disjoint}(\{R, F', reach_{H'}(v)\})$*, and*
3. $\mathsf{stable}(R, H, H')$

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

Case 1: E:Var

Suppose $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

$\mathsf{set}(reach_H(v))$	$(\mathsf{no\_alias}(V, H))$
$\mathsf{disjoint}(\{R, F, reach_H(v)\})$	$(\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\}))$
$\mathsf{no\_alias}(V, H)$	(Sp.)
$\mathsf{stable}(R, H, H')$	$(H = H')$

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

Suppose $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

$\mathsf{set}(reaach_H(v))$	$(reach_H(v) = \emptyset)$
$\mathsf{disjoint}(\{R, F, \emptyset\})$	$(\mathsf{disjoint}(R, F))$
$\mathsf{no\_alias}(V, H)$	(Sp.)
$\mathsf{stable}(R, H, H')$	$(H = H')$

Case 4: E:App

Case 5: E:CondT Similar to E:MatNil

Case 6: E:CondF Similar to E:CondT

Case 7: E:Let

$V, H, R, F \vdash \texttt{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2, F_2$ (case)

$V, H, R', F \vdash e_1 \Downarrow v_1, H_1, F_1$ (ad.)

$\Sigma; \Gamma_1, \Gamma_2 \vdash \texttt{let}(e_1; x : \tau.e_2) : B$ (case)

$\Sigma; \Gamma_1 \vdash e_1 : A$ (ad.)

Suppose $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

$H \vDash V_1 : \Gamma_1$ (def of W.D.E and Lemma 1.2)

By IH, we have invariant on J_1

NTS (1) - (3) to instantiate invariant on J_1

(1) $dom(V_1) = FV(e_1)$ (def of V_1)

(2) $\mathsf{no\_alias}(V_1, H)$ ($\mathsf{no\_alias}(V, H)$ and $V_1 \subseteq V$)

(3) $\mathsf{disjoint}(R', F, locs_{V,H}(e_1))$

$F \cap R' = \emptyset$ ($F \cap locs_{V,H}(e) = \emptyset$ and $locs_{V_2,H}(\texttt{lam}(x : \tau.e_2)) \subseteq locs_{V,H}(e)$)

$FV(e_1) \cap FV(\texttt{lam}(x : \tau.e_2)) = \emptyset$ (Lemma 1.2)

$locs_{V,H}(e_1) \cap locs_{V_2,H}(\texttt{lam}(x : \tau.e_2)) = \emptyset$ ($\mathsf{no\_alias}(V, H)$)

$R' \cap locs_{V,H}(e_1) = \emptyset$ ($\mathsf{disjoint}(\{R, locs_{V,H}(e)\})$)

$F \cap locs_{V,H}(e_1) = \emptyset$ (Sp.)

Thus we have $\mathsf{disjoint}(R', F, locs_{V,H}(e_1))$

By IH,

(1) $\mathsf{set}(reach_{H_1}(v_1))$

(2) $\mathsf{disjoint}(\{R', F_1, reach_{H_1}(v_1)\})$

(3) $\mathsf{stable}(R', H, H_1)$

$V_2', H_1, R, F_1 \cup g \vdash e_2 \Downarrow v_2, H_2, F_2$ (ad.)

$\Sigma; \Gamma_2, x : A \vdash e_2 : B$ (ad.)

$H_1 \vDash V_2' : (\Gamma_2, x : A)$ (???)

By IH, we have invariant on J_2

NTS (1) - (3) to instantiate invariant on J_2

(1) $dom(V_2') = FV(e_2)$ (def of V_2')

(2) $\mathsf{no\_alias}(V_2', H_1)$

Let $x_1, x_2 \in V2', x_1 \neq x_2$ be arb.

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

$reach_H(V_2'(x_1)) \subseteq R'$ ($reach_H(V_2'(x_1)) \subseteq locs_{V_2',H}(\texttt{lam}(x : \tau.e_2))$)

$reach_H(V_2'(x_2)) \subseteq R'$ ($reach_H(V_2'(x_2)) \subseteq locs_{V_2',H}(\texttt{lam}(x : \tau.e_2))$)

$reach_H(V_2'(x_1)) = reach_{H_1}(V_2'(x_1)), reach_H(V_2'(x_2)) = reach_{H_1}(V_2'(x_2))$
($\mathsf{stable}(R', H, H_1)$ and Lemma 1.3)

$reach_{H_1}(V_2'(x_1)) = reach_H(V(x_1)), reach_{H_1}(V_2'(x_2)) = reach_H(V(x_2))$
($\mathsf{stable}(R', H, H_1)$ and Lemma 1.3)

$\mathsf{no\_alias}(V_2', H_1)$ ($\mathsf{no\_alias}(V, H)$)

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

$reach_{H_1}(V_2'(x_1)) = reach_{H_1}(v_1)$ (def of V_2')

$reach_{H_1}(V_2'(x_2)) \subseteq R'$ (same as above)

$\mathsf{set}(reach_{H_1}(v_1))$ (IH 1.1)

$reach_{H_1}(V_2'(x_2)) = reach_H(V(x_2))$ (same as above)

$\mathsf{set}(reach_{H_1}(V_2'(x_2)))$ ($\mathsf{no\_alias}(V, H)$)

$reach_{H_1}(V_2'(x_1)) \cap reach_{H_1}(V_2'(x_2)) = \emptyset$ ($\mathsf{disjoint}(\{R', reach_{H_1}(v_1)\})$)

Thus we have $\mathsf{no\_alias}(V_2', H_1)$

(3) $\mathsf{disjoint}(\{R, F_1 \cup g, locs_{V_2', H_1}(e_2)\})$

$R \cap F_1 = \emptyset$ ($\mathsf{disjoint}(\{R', F_1\})$ from 1.2 and $R \subseteq R'$)

$R \cap (F_1 \cup g) = \emptyset$ (def of g)

NTS $(F_1 \cup g) \cap locs_{V_2', H_1}(e_2) = \emptyset$

Let $l \in locs_{V_2', H_1}(e_2)$ be arb.

$l \in reach_{H_1}(V_2'(x'))$ for some $x' \in V_2'$

case: $x' \neq x$

$reach_H(V_2(x')) = reach_{H_1}(V_2'(x'))$ (same as above)

$reach_{H_1}(V_2'(x')) \subseteq R'$ (def of R')

$reach_{H_1}(V_2'(x')) \cap F_1 = \emptyset$ ($\mathsf{disjoint}(\{R', F_1\})$ from 1.2)

case: $x' = x$

$reach_{H_1}(V_2'(x')) = reach_{H_1}(v_1)$ (def of V_2')

$reach_{H_1}(V_2'(x')) \cap F_1 = \emptyset$ ($\mathsf{disjoint}(\{F_1, reach_{H_1}(v_1)\})$ from 1.2)

$reach_{H_1}(V_2'(x')) \subseteq locs_{V_2', H_1}(e_2)$ (def of $locs_{V,H}$)

$reach_{H_1}(V_2'(x')) \cap g = \emptyset$ (def of g)

Thus $reach_{H_1}(V_2'(x')) \cap (F_1 \cup g) = \emptyset$

NTS $R \cap locs_{V_2', H_1}(e_2) = \emptyset$

Let $l \in locs_{V_2', H_1}(e_2)$ be arb.

$l \in reach_{H_1}(V_2'(x'))$ for some $x' \in V_2'$

case: $x' \neq x$

$reach_H(V_2(x')) = reach_{H_1}(V_2'(x'))$ (same as above)

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

$l \notin R$ ($\mathsf{disjoint}(\{R, locs_{V,H}(e)\})$ from 0.3)

case: $x' = x$

$reach_{H_1}(V_2'(x')) = reach_{H_1}(v_1)$ (def of V_2')

$reach_{H_1}(V_2'(x')) \cap R = \emptyset$ ($\mathsf{disjoint}(\{R', reach_{H_1}(v_1)\})$ from 1.2 and $R \subseteq R'$)

Thus $reach_{H_1}(V_2'(x')) \cap R = \emptyset$

Hence we have *(3)* $\mathsf{disjoint}(R, F_1 \cup g, locs_{V_2',H_1}(e_2))$

By instantiating the invariant on J_2, we have

(1) $\mathsf{set}(reach_{H_2}(v_2))$

(2) $\mathsf{disjoint}(\{R, F_2, reach_{H_2}(v_2)\})$

(3) $\mathsf{stable}(R, H_1, H_2)$

Lastly, showing (1) - (3) holds for the original case J_0 :

(1) $\mathsf{set}(reach_{H_2}(v_2))$ (By 2.1)

(2) $\mathsf{disjoint}(\{R, F_2, reach_{H_2}(v_2)\})$ (By 2.2)

(3) $\mathsf{stable}(R, H_1, H_2)$

Let $l \in R$ be arb.

$H(l) = H_1(l)$ ($\mathsf{stable}(R', H, H_1)$ from 1.3)

$H_1(l) = H_2(l)$ ($\mathsf{stable}(R, H_1, H_2)$ from 2.3)

$H(l) = H_2(l)$

Hence $\mathsf{stable}(R, H, H_2)$

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 $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

NTS (1) - (3) holds after evaluation

(1) $\mathsf{set}(reach_{H''}(l))$

$\mathsf{stable}(\{locs_{V,H}(e)\}, H, H'')$ ($\mathsf{disjoint}(\{F, locs_{V,H}(e)\})$ and *copy* only updates $l \in L \subseteq F$)

$reach_H(V(x_i)) = reach_{H''}(V(x_i))$ ($reach_H(V(x_i)) \subseteq locs_{V,H}(e)$ and 1.3 for $i = 1, 2$)

$reach_{H''}(l) = \{l\} \cup reach_{H''}(V(x_1)) \cup reach_{H''}(V(x_2))$ (def of $reach_H$)

$\mathsf{set}(reach_{H''}(l))$ ($l \notin locs_{V,H}(e)$ and $\mathsf{no\_alias}(V, H)$)

(2) $\mathsf{disjoint}(\{R, F', reach_{H''}(l)\})$

$R \cap F' = \emptyset$ ($F' \subseteq F$ and $\mathsf{disjoint}(\{R, F\})$)

$R \cap reach_{H''}(l) = \emptyset$ ($l \in F$ and $\mathsf{disjoint}(\{R, locs_{V,H}(e)\})$)

$F' \cap reach_{H''}(l) = \emptyset$ ($F' \subseteq F$ and $\mathsf{disjoint}(\{F, locs_{V,H}(e)\})$)

Thus we have *(2)* $\mathsf{disjoint}(\{R, F', reach_{H''}(l)\})$

(3) $\mathsf{stable}(R, H, H'')$ (since copy only updates $l \in L \subseteq F$ and $F \cap R = \emptyset$)

Case 12: E:MatNil

Suppose $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$

$\Sigma; \Gamma' \vdash e_1 : B$ (ad.)

$V, H, R, F \cup g \vdash e_1 \Downarrow v, H', F'$ (ad.)

$H \vDash V' : \Gamma'$ (def of W.D.E)

By IH, we have invariant on J_1

NTS (1) - (3) to instantiate invariant on J_1

(1) $dom(V') = FV(e_1)$ (def of V')

(2) $\mathsf{no\_alias}(V', H)$ ($\mathsf{no\_alias}(V, H)$ and $V' \subseteq V$)

(3) $\mathsf{disjoint}(\{R, F, locs_{V',H}(e_1)\})$

($\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$ and $locs_{V',H}(e_1) \subseteq locs_{V,H}(e)$)

Instantiating invariant on J_1,

(1) $\mathsf{set}(reach_{H'}(v))$

(2) $\mathsf{disjoint}(\{R, F_1, reach_{H'}(v)\})$

(3) $\mathsf{stable}(R, H, H')$

Case 13: E:MatCons

$V(x) = l$ (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 $H \vDash V : \Gamma, dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{F, R, locs_{V,H}(e)\})$

$H \vDash V(x) : L(A)$ (def of W.D.E)

$H'' \vDash v_h : A,\ H'' \vDash v_t : L(A)$ (ad.)

$H \vDash v_h : A,\ H \vDash v_t : L(A)$ (???)

$H \vDash V'' : \Gamma', x_h : A, x_t : L(A)$ (def of W.D.E)

By IH, we have invariant on J_1

NTS (1) - (3) to instantiate invariant on J_1

(1) $dom(V'') = FV(e_2)$ (def of V'')

(2) $\mathsf{no\_alias}(V'', H)$

Let $x_1, x_2 \in V'', x_1 \neq x_2, r_{x_1} = reach_H(V''(x_1)), r_{x_2} = reach_H(V''(x_2))$

case: $x_1 \notin \{x_h, x_t\}, x_2 \notin \{x_h, x_t\}$

$(1), (2)$ from $\mathsf{no\_alias}(V, H)$

case: $x_1 = x_h, x_2 \notin \{x_h, x_t\}$

$\mathsf{set}(r_{x_1})$ (since $\mathsf{set}(reach_H(V(x)))$ from $\mathsf{no\_alias}(V, H)$)

$\mathsf{set}(r_{x_2})$ (since $\mathsf{no\_alias}(V, H)$)

$x_2 \in FV(e)$ (def of FV)

$reach_H(V(x)) \cap r_{x_2} = \emptyset$ (def of $reach$ and $\mathsf{no\_alias}(V, H)$)

hence $r_{x_1} \cap r_{x_2} = \emptyset$

case: $x_1 = x_h, x_2 = x_t$

$\mathsf{set}(r_{x_1})$ since $\mathsf{set}(reach_H(V(x)))$ from $\mathsf{no\_alias}(V, H)$

$\mathsf{set}(r_{x_2})$ since $\mathsf{set}(reach_H(V(x)))$ from $\mathsf{no\_alias}(V, H)$

$r_{x_1} \cap r_{x_2} = \emptyset$ ($\mathsf{set}(reach_H(V(x)))$)

case: otherwise

similar to the above

Thus we have $\mathsf{no\_alias}(V'', H)$

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

$(F \cup g) \cap R = \emptyset$ (since $F \cap R = \emptyset$ and by def of g)

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

Let $l' \in locs_{V'',H}(e_2)$ be arb.

case: $l' \in reach_H(V''(x'))$ for some $x' \in FV(e_2)$ where $x' \notin \{x_h, x_t\}$

$x' \in V$ (def of V'')

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

$x' \in FV(e)$ (def of FV)

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

$l' \notin R$ ($\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$)

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

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

Instantiating invariant on J_1,

(1) $\mathsf{set}(reach_{H'}(v))$

(2) $\mathsf{disjoint}(\{R, F', reach_{H'}(v)\})$

(3) $\mathsf{stable}(R, H, H')$

$\square$

Task 1.5 (Soundness). *let* $H \vDash V : \Gamma$, $\Sigma; \Gamma \left|\frac{q}{q'}\right. e : B$, *and* $V, H \ \vdash e \Downarrow v, H'$. *Then* $\forall C \in \mathbb{Q}^+$ *and* $\forall F, R \subseteq \mathsf{Loc}$, *if we have the following (existence lemma):*

1. $dom(V) = FV(e)$
2. $\mathsf{no\_alias}(V, H)$
3. $\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$, *and*
4. $|F| \geq \Phi_{V,H}(\Gamma) + q + C$

then there exists $F' \subseteq \mathsf{Loc}$ *s.t.*

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

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

Case 1: E:Var

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

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

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

Case 4: E:App

Case 5: E:CondT

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

Case 6: E:CondF Similar to E:CondT

Case 7: E:Let

$V, H \vdash e \Downarrow v_2, H_2$ (case)

$V, H \vdash e_1 \Downarrow v_1, H_1$ (ad.)

$\Sigma; \Gamma_1 \left|\frac{q}{p}\right. e_1 : A$ (ad.)

$H \vDash V_1 : \Gamma_1$ (def of W.D.E)

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

Suppose $dom(V) = FV(e), \mathsf{no\_alias}(V, H), \mathsf{disjoint}(\{R, F, locs_{V,H}(e)\}),$ and $|F| \geq \Phi_{V,H}(\Gamma) + q + C$

NTF F' s.t.

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

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

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

$\mathsf{disjoint}(\{R', F, locs_{V,H}(e_1)\})$ (Similar to case in Lemma 1.4)

Instantiate IH with $C = C + \Phi_{V_2,H}(\Gamma_2), F = F, R = R',$ we get existence lemma on J_1 :

NTS (1) - (4) to instantiate existence lemma on J_1

(1) $dom(V_1) = FV(e_1)$

(2) $\mathsf{no\_alias}(V_1, H)$

(3) $\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$ ((1) - (3) all verbatim as in Lemma 1.4)

(4) $|F| \geq \Phi_{V_1,H}(\Gamma_1) + q + C + \Phi_{V,H}(\Gamma_2)$

($|F| \geq \Phi_{V,H}(\Gamma) + q + C$ and $\Phi_{V,H}(\Gamma) \geq \Phi_{V_1,H}(\Gamma_1) + \Phi_{V,H}(\Gamma_2)$)

Instantiating existence lemma on J_1, we get F'' s.t.

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

2.$|F''| \geq \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V_2,H_1}(\Gamma_2)$

For the second premise:

$\Sigma; \Gamma_2, x : A \left|\frac{p}{q'}\right. e_2 : B$ (ad.)

$H_1 \vDash v_1 : A$ and (Theorem 3.3.4)

$H_1 \vDash V : \Gamma_2$ (???)

$H_1 \vDash V' : \Gamma_2, x : A$ (def of $\vDash$)

$V', H_1 \vdash e_2 \Downarrow v_2, H_2$ (ad.)

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

Instantiate IH with $C = C, F = F'' \cup g, R = R,$ we get existence lemma on J_2 :

NTS (1) - (4) to instantiate existence lemma on J_1

(1) $dom(V_2') = FV(e_2)$

(2) $\mathsf{no\_alias}(V_2', H_1)$

(3) $\mathsf{disjoint}(\{R, F'' \cup g, locs_{V_2',H_1}(e_2)\})$ ((1) - (3) all verbatim as in Lemma 1.4)

(4) $|F'' \cup g| \geq \Phi_{V_2',H_1}(\Gamma_2, x : A) + p + C$

$$
\begin{aligned}
|F'' \cup g| &\geq |F''| \\
&\geq \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V_2,H}(\Gamma_2) && \text{(IH)} \\
&= \Phi_{H_1}(v_1 : A) + p + C + \Phi_{V_2',H_1}(\Gamma_2) && \text{(Lemma 4.3.3)} \\
&= \Phi_{V_2',H_1}(\Gamma_2, x : A) + p + C && \text{(def of } \Phi)
\end{aligned}
$$

Instantiating existence lemma on $J_2,$ we get $F^{(3)}$ s.t.

1.$V_2', H_1, R, F'' \cup g \vdash e_2 \Downarrow v_2, H_2, F^{(3)}$

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

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

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

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

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

$= \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A)) + p + 1 - (p + \Phi_H(v_h : A) + \Phi_H(v_t : L^p(A)))$ (Lemma 4.3.3)

$= 1$

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

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

$H(l) = \langle v_h, v_t \rangle$ (ad.)

$\Gamma = \Gamma', x : L^p(A)$ (ad.)

$\Sigma; \Gamma', x_h : A, x_t : L^p(A) \,\Big|\!\frac{q+p+1}{q'}\, e_2 : B$ (ad.)

$V'', H \vdash e_2 \Downarrow v, H'$ (ad.)

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

$H \vDash V(x) : L^p(A)$ (def of W.D.E)

$H'' \vDash v_h : A,\ H'' \vDash v_t : L^p(A)$ (ad.)

$H \vDash v_h : A,\ H \vDash v_t : L^p(A)$ (???)

$H \vDash V'' : \Gamma', x_h : A, x_t : L^p(A)$ (def of W.D.E)

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

NTF F' s.t.

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

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

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

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

$l \notin F \cup R$ ($\mathsf{disjoint}(\{R, F, locs_{V,H}(e)\})$)

AFSOC $l \in locs_{V'',H}(e_2)$

Then $l \in reach_H(\overline{V}''(x'))$ for some $x' \neq x$

$x' \in \{x_h, x_t\}$ (since $reach_H(\overline{V}(x')) \cap reach_H(\bar{(}Vx)) = \emptyset$ from $\mathsf{no\_alias}(V, H)$)

WLOG let $x' = x_h$

But then $\mu_{reach_H(\overline{V}(x))}(l) \geq 2$ and $\mathsf{set}(reach_(\overline{V}(x)))$ doesn't hold

$l \notin locs_{V'',H}(e_2)$

Hence $l \in g$

Next, we have $\mathsf{no\_alias}(V'', H)$ and $\mathsf{disjoint}(\{R, F \cup g, locs_{V'',H}(e_2)\})$ (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 $|F''| \geq \Phi_{V'',H}(\Gamma', x_h : A, x_t : L^p(A)) + q + p + 1 + C'$:

$$
\begin{aligned}
|F''| &= |F \cup g| \\
&= |F| + |g| && (F \text{ and } g \text{ disjoint}) \\
&\geq \Phi_{V,H}(\Gamma) + q + C + |g| && (\text{Sp.}) \\
&= \Phi_{V,H}(\Gamma', x_h : A, x_t : L^p(A)) + p + q + C + |g| && (\text{Lemma 4.1.1}) \\
&= \Phi_{V,H}(\Gamma', x_h : A, x_t : L^p(A)) + p + q + C + 1 && (g \text{ nonempty})
\end{aligned}
$$

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)

$\square$