SIG Proceedings Paper in LaTeX Format[*]

Extended Abstract[†]

Lars Thørväld[‡]
The Thørväld Group
Hekla, Iceland
larst@affiliation.org

ABSTRACT

This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings.[1]

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

KEYWORDS

ACM proceedings, LaTeX, text tagging

ACM Reference Format:
Lars Thørväld. 1997. SIG Proceedings Paper in LaTeX Format: Extended Abstract. In *Proceedings of ACM Woodstock conference (WOODSTOCK'97)*, Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 12 pages. https://doi.org/10.475/123_4

1 PATHS AND ALIASING

Model dynamics using judgement of the form:

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

Where V : Var → Val × State, H : Loc → Val, $R \subseteq$ Loc, $F \subseteq$ Loc, and Σ : Var → FTypes. This can be read as: under stack V, heap H, roots R, freelist F, and program P with signature Σ, the expression e evaluates to v, and engenders a new heap H' and freelist F'.

A *program* is then a Σ indexed map P from Var to pairs $(y_f, e_f)_{f \in \Sigma}$, where $\Sigma(y_f) = A \rightarrow B$, and $\Sigma; y_f : A \vdash e_f : B$ (typing rules are discussed in 6). We write $P : \Sigma$ to mean P is a program with signature Σ. Because the signature Σ for the mapping of function names to 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,

BTypes	τ ::=		
	`nat`	`nat`	natura
	`unit`	`unit`	unit
	`bool`	`bool`	boolea
	`prod`$(\tau_1; \tau_2)$	$\tau_1 \times \tau_2$	produ
	`list`(τ)	$L(\tau)$	list
FTypes	ρ ::=		
	`arr`$(\tau_1; \tau_2)$	$\tau_1 \rightarrow \tau_2$	first or
Exp	e ::=		
	x	x	variab
	`nat`$[n]$	$\overline{n}$	numbe
	`unit`	$()$	unit
	`T`	`T`	true
	`F`	`F`	false
	`if`$(x; e_1; e_2)$	`if` x `then` e_1 `else` e_2	if
	`lam`$(x : \tau.e)$	$\lambda\, x : \tau.e$	abstra
	`ap`$(f; x)$	$f(x)$	applica
	`tpl`$(x_1; x_2)$	$\langle x_1, x_2 \rangle$	pair
	`case`$(x_1, x_2.e_1)$	`case` p $\{(x_1; x_2) \hookrightarrow e_1\}$	match
	`nil`	$[]$	nil
	`cons`$(x_1; x_2)$	$x_1 :: x_2$	cons
	`case`$\{l\}(e_1; x, xs.e_2)$	`case` l $\{$`nil` $\hookrightarrow e_1 \mid$ `cons`$(x; xs) \hookrightarrow e_2\}$	match
	`let`$(e_1; x : \tau.e_2)$	`let` $x = e_1$ `in` e_2	let
	`share`$(x; x_1, x_2.e)$	`share` x `as` x_1, x_2 `in` e	share
Val	v ::=		
	`val`(n)	n	numer
	`val(T)`	`T`	true va
	`val(F)`	`F`	false v
	`val(Null)`	`Null`	null va
	`val`$($`cl`$(V; x.e))$	$(V, x.e)$	functio
	`val`(l)	l	loc val
	`val`$($`pair`$(v_1; v_2))$	$\langle v_1, v_2 \rangle$	pair va
State	s ::=		
	`alive`	`alive`	live va
	`dead`	`dead`	dead v
Loc	l ::=		
	`loc`(l)	l	locatio
Var	l ::=		
	`var`(x)	x	variab

[*]Produces the permission block, and copyright information
[†]The full version of the author's guide is available as `acmart.pdf` document
[‡]This author is the one who did all the really hard work.
[1]This is an abstract footnote

WOODSTOCK'97, July 1997, El Paso, Texas USA

ACM ISBN 123-4567-24-567/08/06...$15.00
https://doi.org/10.475/123_4

each of which is a set of "top-level" first-order declarations to be used during evaluation.

For a partial map $f : A \rightarrow B$, 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 $reach : Val \rightarrow \{\{\text{Loc}\}\}$ that maps stack values its the root *multiset*, the multiset of locations that's already on the stack.
Next we define reachability of values:

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

For a multiset S, we write $\mu_S : S \rightarrow \mathbb{N}$ 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 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) set(r_x), 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)**:** $\forall x \in V.\ reach_H(V(x)) \cap F = \emptyset$
disjoint(C)**:** $\forall X, Y \in C.\ X \cap Y = \emptyset$

For a stack V and a heap H, whenever 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 $FV^\star(e)$, the multiset of free variables of e. As the usual FV, 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 $FV^\star(e)$.

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 *literal size* of a value, e.g. the size to store its address.

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

Let $card(S)$ denote the number of unique elements, e.g. the cardinality of a multiset S. We write $\|v\|_H$ for $card(reach_H(v))$
As usual, we extend it to stacks V: $\|V\|_H = \sum_{V(x)=v} \|v\|_H$

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

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

2 GARBAGE COLLECTION SEMANTICS

$$\frac{V(x) = v}{V, H, R, F \ \vdash x \Downarrow v, H, F}\,(\mathrm{S}_1)$$

$$\frac{}{V, H, R, F \ \vdash \overline{n} \Downarrow \mathsf{val}(n), H, F}\,(\mathrm{S}_2)$$

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

$$\frac{}{V, H, R, F \ \vdash \mathsf{F} \Downarrow \mathsf{val}(\mathsf{F}), H, F}\,(\mathrm{S}_4)$$

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

$$\frac{\begin{array}{c} V = V'[x \mapsto \mathsf{T}] \qquad g = \{l \in H \mid l \notin F \cup R \cup locs_{V,H}(e_1)\} \\ V', H, R, F \cup g \ \vdash e_1 \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash \mathsf{if}(x; e_1; e_2) \Downarrow v, H', F'}\,(\mathrm{S}_6)$$

$$\frac{\begin{array}{c} V = V'[x \mapsto \mathsf{F}] \qquad g = \{l \in H \mid l \notin F \cup R \cup locs_{V,H}(e_2)\} \\ V', H, R, F \cup g \ \vdash e_2 \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash \mathsf{if}(x; e_1; e_2) \Downarrow v, H', F'}\,(\mathrm{S}_7)$$

$$\frac{\begin{array}{c} V(x) = v' \\ P(f) = (y_f, e_f) \qquad g = \{l \in H \mid l \notin F \cup R \cup locs_{V,H}(e_f)\} \\ [y_f \mapsto v'], H, R, F \cup g \ \vdash e_f \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash f(x) \Downarrow v, H', F'}\,(\mathrm{S}_8)$$

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

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

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

$$\frac{v = \langle V(x_1), V(x_2) \rangle \qquad H' = H\{l \mapsto v\}}{V, H, R, F \ \vdash \mathsf{cons}(x_1; x_2) \Downarrow l, H', F'}\,(\mathrm{S}_{12})$$

$$\frac{\begin{array}{c} V = V'[x \mapsto v'] \\ g = reach_H(v') \qquad V', H, R, F \cup g \ \vdash e \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash \mathsf{drop}(x; e) \Downarrow v, H', F'}\,(\mathrm{S}_{13})$$

$$\frac{\begin{array}{c} V = V'[x \mapsto v'] \\ L \subseteq F \qquad |L| = \left\| v' \right\|_H \qquad H', v'' = copy(H, L, v') \\ V'[x_1 \mapsto v', x_2 \mapsto v''], H', R, F \setminus L \ \vdash e \Downarrow v, H'', F' \end{array}}{V, H, R, F \ \vdash \mathsf{shareCopy}\ x\ \mathsf{as}\ x_1, x_2\ \mathsf{in}\ e \Downarrow v, H'', F'}\,(\mathrm{S}_{14})$$

$$\frac{\begin{array}{c} V(x) = \mathsf{Null} \qquad V' \subseteq V \qquad dom(V') = FV(e_1) \\ g = \{l \in H \mid l \notin F \cup R \cup locs_{V,H}(e_1)\} \\ V', H, R, F \cup g \ \vdash e_1 \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash \mathsf{case}\, x\, \{\mathsf{nil} \hookrightarrow e_1 \mid \mathsf{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H', F'}\,(\mathrm{S}_{15})$$

$$\frac{\begin{array}{c} V(x) = l \qquad H(l) = \langle v_h, v_t \rangle \\ V' \subseteq V \qquad dom(V') = FV(e_2) \setminus \{x_h, x_t\} \\ V'' = V'[x_h \mapsto v_h, x_t \mapsto v_t] \\ g = \{l \in H \mid l \notin F \cup R \cup locs_{V'',H}(e_2)\} \\ V'', H, R, F \cup g \ \vdash e_2 \Downarrow v, H', F' \end{array}}{V, H, R, F \ \vdash \mathsf{case}\, x\, \{\mathsf{nil} \hookrightarrow e_1 \mid \mathsf{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H', F'}\,(\mathrm{S}_{16})$$

3 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.

$$\frac{\begin{array}{c} v = \langle V(x_1), V(x_2)\rangle \\ (L \sqcup \{l\}) \cap dom(H) = \emptyset \qquad H', l = copy(H, L, v) \end{array}}{V, H \ \vdash \mathsf{cons}(x_1; x_2) \Downarrow l, H'} (\mathrm{S}_{18})$$

$$\frac{\begin{array}{c} V(x) = l \qquad H(l) = \langle v_h, v_t\rangle \\ V' \subseteq V \qquad dom(V') = FV(e_2) \setminus \{x_h, x_t\} \\ V'' = V'[x_h \mapsto v_h, x_t \mapsto v_t] \qquad V'', H \ \vdash e_2 \Downarrow v, H' \end{array}}{V, H \ \vdash \mathsf{case}\, x\, \{\mathsf{nil} \hookrightarrow e_1 \mid \mathsf{cons}(x_h; x_t) \hookrightarrow e_2\} \Downarrow v, H'} (\mathrm{S}_{19})$$

$$\frac{\begin{array}{c} V = V_1 \sqcup V_2 \qquad dom(V_1) = FV(e_1) \\ dom(V_2) = FV(\mathsf{lam}(x : \tau.e_2)) \qquad V_1, H \vdash e_1 \Downarrow v_1, H_1 \\ V_2' = V_2[x \mapsto v_1] \qquad V_2', H_1 \vdash e_2 \Downarrow v_2, H_2 \end{array}}{V, H \ \vdash \mathsf{let}(e_1; x : \tau.e_2) \Downarrow v_2, H_2} (\mathrm{S}_{20})$$

$$\frac{\begin{array}{c} V = V'[x \mapsto v'] \\ L\ fresh \qquad |L| = \|v'\|_H \qquad H', v'' = copy(H, L, v') \\ V'[x_1 \mapsto v', x_2 \mapsto v''], H' \ \vdash e \Downarrow v, H'' \end{array}}{V, H \ \vdash \mathsf{shareCopy}\ x\ \mathsf{as}\ x_1, x_2\ \mathsf{in}\ e \Downarrow v, H''} (\mathrm{S}_{21})$$

4 WELL DEFINED ENVIRONMENTS

In order to define the potential for first-order types, we need a notion of well-define environments, one that relates heap values to semantic values of a type. We first give a denotational semantics for the first-order types:

$$\begin{array}{c} () \in [\![\mathsf{unit}]\!] \\ \bot \in [\![\mathsf{bool}]\!] \\ \top \in [\![\mathsf{bool}]\!] \\ 0 \in [\![\mathsf{nat}]\!] \\ n + 1 \in [\![\mathsf{nat}]\!] \text{ if } n \in [\![\mathsf{nat}]\!] \\ [] \in [\![L(A)]\!] \\ \pi(a, l) \in [\![L(A)]\!] \text{ if } a \in [\![A]\!] \text{ and } l \in [\![L(A)]\!] \end{array}$$

Where semantic set for each type is the least set such that the above holds. Note $\pi(x, y)$ is the usual set-theoretic pairing function, and write $[a_1, ..., a_n]$ for $\pi(a_1, ..., \pi(a_n, []))$.

Now we give the judgements relating heap values to semantic values, in the form $\boxed{H \vDash v \mapsto a : A}$, which can be read as: under heap H, heap value v defines the semantic value $a \in [\![A]\!]$.

$$\frac{n \in \mathbb{Z}}{H \vDash n \mapsto n : \mathsf{nat}} (\text{V:ConstI})$$

$$\frac{}{H \vDash \mathsf{Null} \mapsto n : \mathsf{unit}} (\text{V:ConstI})$$

$$\frac{A \in \mathsf{BType}}{H \vDash \mathsf{Null} \mapsto n : L(A)} (\text{V:Nil})$$

$$\frac{}{H \vDash \mathsf{T} \mapsto \top : \mathsf{bool}} (\text{V:True}) \qquad \frac{}{H \vDash \mathsf{F} \mapsto \bot : \mathsf{bool}} (\text{V:False})$$

$$\frac{\begin{array}{c} l \in \mathsf{Loc} \qquad H(l) = \langle v_h, v_t\rangle \\ H \vDash v_h \mapsto a_1 : A \qquad H \vDash v_t \mapsto [a_2, \ldots, a_n] : L(A) \end{array}}{H \vDash l \mapsto [a_1, \ldots, a_n] : L(A)} (\text{V:Cons})$$

5 STACK VS HEAP ALLOCATED TYPES

In order to share variables, we need to distinguish between types that are allocated on the stack and the heap. We write $\boxed{\mathsf{stack}(A)}$ to denote that values of type A will be allocataed *entirely* on the stack at run time (no references into the heap).

$$\frac{A \in \{\mathsf{unit}, \mathsf{bool}, \mathsf{nat}\}}{\mathsf{stack}(A)} (\text{S:Const})$$

$$\frac{\mathsf{stack}(A_1) \qquad \mathsf{stack}(A_2)}{\mathsf{stack}(A_1 \times A_2)} (\text{S:Product})$$

6 LINEAR GARBAGE COLLECTION TYPE RULES

The linear version of 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. The second let rule expresses the fact that since stack types don't reference heap cells, any heap cells used in the evaluation of e_1 can be deallocated, as there are no longer references to them in v_1.

$$\frac{n \in \mathbb{Z}}{\Sigma; \emptyset \big|\frac{q}{q}\, n : \mathtt{nat}}\,(\text{L:ConstI}) \qquad \frac{}{\Sigma; \emptyset \big|\frac{q}{q}\, () : \mathtt{unit}}\,(\text{L:ConstU})$$

$$\frac{}{\Sigma; \emptyset \big|\frac{q}{q}\, \mathtt{T} : \mathtt{bool}}\,(\text{L:ConstT}) \qquad \frac{}{\Sigma; \emptyset \big|\frac{q}{q}\, \mathtt{F} : \mathtt{bool}}\,(\text{L:ConstF})$$

$$\frac{}{\Sigma; x : B \big|\frac{q}{q}\, x : B}\,(\text{L:Var}) \qquad \frac{\Sigma(f) = A \xmapsto{q/q'} B}{\Sigma; x : A \big|\frac{q}{q'}\, f(x) : B}$$

$$\frac{\Sigma; \Gamma \big|\frac{q}{q'}\, e_t : B \qquad \Sigma; \Gamma \big|\frac{q}{q'}\, e_f : B}{\Sigma; \Gamma, x : \mathtt{bool} \big|\frac{q}{q'}\, \mathtt{if}\ x\ \mathtt{then}\ e_t\ \mathtt{else}\ e_f : B}\,(\text{L:Cond})$$

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

$$\frac{\Sigma; \Gamma, x_1 : A_1, x_2 : A_2 \big|\frac{q}{q'}\, e : B}{\Sigma; \Gamma, x : (A_1, A_2) \big|\frac{q}{q'}\, \mathtt{case}\ x\ \{(x_1; x_2) \hookrightarrow e\} : B}\,(\text{L:MatP})$$

$$\frac{}{\Sigma; \emptyset \big|\frac{q}{q}\, \mathtt{nil} : L^p(A)}\,(\text{L:Nil})$$

$$\frac{}{\Sigma; x_h : A, x_t : L^p(A) \big|\frac{q+p+1}{q}\, \mathtt{cons}(x_h; x_t) : L^p(A)}\,(\text{L:Cons})$$

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

$$\frac{\Sigma; \Gamma_1 \big|\frac{q}{p}\, e_1 : A \qquad \Sigma; \Gamma_2, x : A \big|\frac{p}{q'}\, e_2 : B}{\Sigma; \Gamma_1, \Gamma_2 \big|\frac{q}{q'}\, \mathtt{let}(e_1; x : \tau.e_2) : B}\,(\text{L:Let})$$

$$\frac{\Sigma; \Gamma_1 \big|\frac{q}{p}\, e_1 : A \qquad \mathtt{stack}(A) \qquad \Sigma; \Gamma_2, x : A \big|\frac{\max(p,q)}{q'}\, e_2 : B}{\Sigma; \Gamma_1, \Gamma_2 \big|\frac{q}{q'}\, \mathtt{let}(e_1; x : \tau.e_2) : B}\,(\text{L:LetS})$$

$$\frac{\Sigma; \Gamma \big|\frac{q}{q'}\, e : B}{\Sigma; \Gamma, x : A \big|\frac{q}{q'}\, \mathtt{drop}(x; e) : B}\,(\text{L:Drop})$$

$$\frac{\mathtt{stack}(A) \qquad \Sigma; \Gamma, x_1 : A, x_2 : A \big|\frac{q}{q'}\, e : B}{\Sigma; \Gamma, x : A \big|\frac{q}{q'}\, \mathtt{share}\ x\ \mathtt{as}\ x_1, x_2\ \mathtt{in}\ e : B}\,(\text{L:Share})$$

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

7 TYPE RULES FOR SHARING

$$\begin{aligned} L^p(A) - n &= L^{\max(p-n,0)}(A - n) \\ A_1 \times A_2 - n &= A_1 - n \times A_2 - n \\ A - n &= A \end{aligned}$$

$$\frac{A \curlyvee A_1, A_2, 1 \qquad \Sigma; \Gamma, x_1 : A_1, x_2 : A_2 \big|\frac{q}{q'}\, e : B}{\Sigma; \Gamma, x : A \big|\frac{q}{q'}\, \mathtt{share}\ x\ \mathtt{as}\ x_1, x_2\ \mathtt{in}\ e : B}\,(\text{M:Share})$$

$$\frac{\begin{array}{c}\Sigma; \Gamma_1 \big|\frac{q}{p}\, e_1 : A \\ \Sigma; \Gamma_1 \big|\frac{\mathrm{cf}}{}\, e_1 : A' \qquad \Sigma; \Gamma_2, x : (A' - 1) \big|\frac{p}{q'}\, e_2 : B\end{array}}{\Sigma; \Gamma_1, \Gamma_2 \big|\frac{q}{q'}\, \mathtt{let}(e_1; x : \tau.e_2) : B}\,(\text{M:Let})$$

Where $A \curlyvee A_1, A_2, n$ is the sharing relation defined as:

$$\begin{array}{ll} L^p(A) \curlyvee L^q(A_1), L^r(A_2), n & \text{if } p = q + r + n \text{ and } A \curlyvee A_1, A_2, n \\ A \curlyvee A_1, A_2 & \text{if } \mathtt{stack}(A), \mathtt{stack}(A_1), \mathtt{stack}(A_2) \text{ and } A \equiv A_1 \equiv A_2 \end{array}$$

8 SOUNDNESS FOR LINEAR GC

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

Definition 8.1 (Well-formed computation)**.** When considering the input mode arguments of a evaluation judgment $V, H, R, F \vdash e \Downarrow v, H', F'$, we say the 5-tuple (V, H, R, F, e) is a *well-formed computation* given the following:

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

And we write $\text{wfc}(V, H, R, F, e)$ to denote this fact.

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* $\text{set}(FV^\star(e))$ *and* $dom(\Gamma) = FV(e)$.

Proof. Induction on the typing judgement. □

Lemma 1.3. *Let* $H \vDash v \mapsto a : A$. *For all sets of locations* R, *if* $reach_H(v) \subseteq R$ *and* $\text{stable}(R, H, H')$, *then* $H' \vDash v \mapsto a : A$ *and* $reach_H(v) = reach_{H'}(v)$.

Proof. Induction on the structure of v. □

Corollary 1.3.1. *Let* $H \vDash V : \Gamma$. *For all sets of locations* R, *if* $\bigcup_{x \in V} reach_H(V(x)) \subseteq R$ *and* $\text{stable}(R, H, H')$, *then* $H' \vDash V : \Gamma$.

Proof. Follows from Lemma 1.3. □

Lemma 1.4. *Let* $H \vDash v \mapsto a : A$. *If* $\mathtt{stack}(A)$, *then* $\Phi_H(v : A) = 0$.

Proof. Induction on $H \vDash v \mapsto a : A$. □

Lemma 1.5 (heap conservation). *Let* $\text{wfc}(V, H, R, F, e), V, H, R, F \vdash e \Downarrow v, H', F'$, *and* $g = \text{gc}(H', R, F')$. *Then* $\|V\|_H + |F| \leq \|v\|_{H'} + |F' \cup g|$.

Proof. Induction on evaluation.

Case 1: E:Var

$$V = [x \mapsto v] \qquad \text{(since } dom(V) = FV(e) = \{x\}\text{)}$$
$$\|V\|_H = \|v\|_H \qquad \text{(def of } \|\cdot\|_H\text{)}$$
$$\|V\|_H + |F| \leq \|v\|_{H'} + |F \cup g|$$

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

$$V = \emptyset \qquad \text{(since } dom(V) = FV(e) = \emptyset\text{)}$$
$$\|V\|_H = \|v\|_H \qquad \text{(def of } \|\cdot\|_H\text{)}$$
$$\|V\|_H + |F| \leq \|v\|_{H'} + |F \cup g|$$

Case 4: E:App
Case 5: E:CondT Similar to E:MatNil
Case 6: E:CondF Similar to E:CondT
Case 7: E:Let

$$\|V_1\|_H + |F| \leq \|v_1\|_{H_1} + |F_1 \cup g| \qquad \text{(IH on first premise)}$$
Let $g' = \text{gc}(H_2, R, F_2)$
$$\|V_2'\|_{H_1} + |F_1 \cup g| \leq \|v_2\|_{H_2} + |F \cup g'| \qquad \text{(IH on second premise)}$$
$$\|V_2'\|_{H_1} = \|V_2\|_{H_1} + \|v_1\|_{H_1} \qquad \text{(definition of semantic size)}$$
$$= \|V_2\|_H + \|v_1\|_{H_1} \qquad \text{(main lemma)}$$
$$\|V_2\|_H + \|v_1\|_{H_1} + |F_1 \cup g| \leq \|v_2\|_{H_2} + |F \cup g'|$$
$$\|V_1\|_H + \|V_2\|_H + \|v_1\|_{H_1} + |F| + |F_1 \cup g| \leq \|v_1\|_{H_1} + \|v_2\|_{H_2} + |F_1 \cup g| + |F \cup$$
$$\|V\|_H + |F| \leq \|v_2\|_{H_2} + |F \cup g'|$$

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

$$V = [x_1 \mapsto v_1, x_2 \mapsto v_2] \quad \text{(since } dom(V) = FV(e) = \{x_1, x_2\}\text{)}$$
$$\|V\|_H = \|v_1\|_H + \|v_2\|_H \qquad \text{(def of } \|\cdot\|_H\text{)}$$
$$\|l\|_{H'} = 1 + \|H'(l)\|_{H'} = 1 + \|v\|_{H''} = 1 + \|v_1\|_{H''} + \|v_2\|_{H''} \qquad \text{(def of semantic size)}$$
$$= 1 + \|v_1\|_H + \|v_1\|_H$$
$$= 1 + \|V\|_H$$
$$L \sqcup \{l\} \subseteq g \qquad (R \cap F = \emptyset \text{ and } L \sqcup \{l\} \subseteq H'')$$
$$|g| \geq |L \sqcup \{l\}| = size(v) + 1$$
$$|F' \cup g| \geq |F|$$
$$\|V\|_H + |F| \leq \|v\|_{H'} + |F \cup g|$$

Case 12: E:MatNil

Case 13: E:MatCons

Let $g' = \text{gc}(H', R, F')$
$$\|V''\|_H + |F \cup g| \leq |F' \cup g'| \quad \text{(IH (wfc from main lemma))}$$
$$\|V''\|_H = \|V'[x_h \mapsto v_h, x_t \mapsto v_t]\|_H$$
$$= \|V'\|_H + \|v_h\|_H + \|v_t\|_H$$
$$= \|V'\|_H + \|l\|_H - 1$$
$$= \|V\|_H - 1$$
$$\|V\|_H - 1 + |F \cup g| \leq |F' \cup g'|$$
$$\|v\|_H - 1 + |F| + |g| \leq |F' \cup g'| \qquad (F \cap g = \emptyset)$$
$$\|v\|_H + |F| \leq |F' \cup g'| \qquad (|g| \geq 1 \text{ from main lemma})$$

Case 13: E:Drop

Let $g' = \text{gc}(H', R, F')$
$$\|V'\|_H + |F \cup g| \leq \|v\|_{H'} + |F' \cup g'| \qquad \text{(IH)}$$
$$HV = \|+\|_{V'}\|$$
$$v' \; \|V\|_H - \|+\|_{v'} |F \cup reach_H(v')| \leq \|v\|_{H'} + |F' \cup g'|$$
$$\|V\|_H - \|+\|_{v'} |F| + |reach_H(v')| \leq \|v\|_{H'} + |F' \cup g'|$$

$\|V\|_H + |F| \le \|v\|_{H'} + |F' \cup g'|$

Case 13: E:ShareCopy

$e = \texttt{shareCopy}\ x\ \texttt{as}\ x_1, x_2\ \texttt{in}\ e$ (case)

Let $g' = \mathrm{gc}(H', R, F')$

$\left\|V'[x_1 \mapsto v', x_2 \mapsto v'']\right\|_{H'} + |F \setminus L| \le \|v\|_{H''} + |F' \cup g'|$
(IH, well-formedness from main lemma)

$\left\|V'[x_1 \mapsto v', x_2 \mapsto v'']\right\|_{H'} + |F \setminus L| = \|V\|_H + \left\|v''\right\|_{H'} + |F| - |L|$
(stability lemma for copy)

$= \|V\|_H + |L| + |F| - |L|$ (lemma about copy)

$= \|V\|_H + |F|$

□

Lemma 1.6. *Let* $\Sigma; \Gamma \left|\frac{q}{q'}\right. e : B$ *and* $V, H, R, F \vdash e \Downarrow v, H', F'$. *Then* $\|V\|_H - \|v\|_{H'} + q \ge q'$.

Lemma 1.7 (main lemma). *For all stacks* V *and heaps* H*, let* $V, H, R, F \vdash e \Downarrow v, H', F'$ *and* $\Sigma; \Gamma \vdash e : B$. *Then given the following:*

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

We have the follwoing:

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

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

Case 1: E:Var

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

$\mathrm{set}(reach_H(v))$ ($\mathrm{no\_alias}(V, H)$)

$\mathrm{disjoint}(\{R, F, reach_H(v)\})$ ($\mathrm{disjoint}(\{R, F, locs_{V,H}(e)\})$)

$\mathrm{no\_alias}(V, H)$ (Sp.)

$\mathrm{stable}(R, H, H')$ ($H = H'$)

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

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

$\mathrm{set}(reaach_H(v))$ ($reach_H(v) = \emptyset$)

$\mathrm{disjoint}(\{R, F, \emptyset\})$ ($\mathrm{disjoint}(R, F)$)

$\mathrm{no\_alias}(V, H)$ (Sp.)

$\mathrm{stable}(R, H, H')$ ($H = H'$)

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

$H \vDash V_1 : \Gamma_1$ (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) $\mathrm{no\_alias}(V_1, H)$ ($\mathrm{no\_alias}(V, H)$ and $V_1 \subseteq V$)

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

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

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

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

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

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

By IH,

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

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

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

$V_2', H_1, R, F_1 \cup g \vdash e_2 \Downarrow v_2, H_2, F_2$ (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) $\mathrm{no\_alias}(V_2', H_1)$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Thus we have no_alias(V_2', H_1)

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

$R \cap F_1 = \emptyset$ (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')

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

$reach_{H_1}(V_2'(x')) \cap F_1 = \emptyset$
(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 $locs_{V,H}$)

$l \notin R$ (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')

$reach_{H_1}(V_2'(x')) \cap R = \emptyset$
(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)* disjoint$(R, F_1 \cup g, locs_{V_2', H_1}(e_2))$

By instantiating the invariant on J_2, we have

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

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

(3) stable(R, H_1, H_2)

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

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

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

(3) stable(R, H_1, H_2)

Let $l \in R$ be arb.

$H(l) = H_1(l)$ (stable(R', H, H_1) from 1.3)

$H_1(l) = H_2(l)$ (stable(R, H_1, H_2) from 2.3)

$H(l) = H_2(l)$

Hence 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 $H \vDash V : \Gamma, dom(V) = FV(e)$, no_alias$(V, H)$, disjoint$(\{R, F, locs_{V,H}(e)\})$

NTS (1) - (3) holds after evaluation

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

stable$(\{locs_{V,H}(e)\}, H, H'')$
(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 $reach_H$)

set$(reach_{H''}(l))$ ($l \notin locs_{V,H}(e)$ and no_alias(V, H))

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

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

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

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

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

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

Case 12: E:MatNil

Suppose $H \vDash V : \Gamma, dom(V) = FV(e)$, no_alias$(V, H)$, 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) no_alias(V', H) (no_alias(V, H) and $V' \subseteq V$)

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

Instantiating invariant on J_1,

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

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

(3) 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 $H \vDash V : \Gamma, dom(V) = FV(e)$, no_alias$(V, H)$, 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) 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 no_alias(V, H)

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

set(r_{x_1})

(since set$(reach_H(V(x)))$ from no_alias(V, H))

set(r_{x_2}) (since 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 no_alias(V, H))

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

case: $x_1 = x_h, x_2 = x_t$

set(r_{x_1}) since set$(reach_H(V(x)))$ from no_alias(V, H)

set(r_{x_2}) since set$(reach_H(V(x)))$ from no_alias(V, H)

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

case: otherwise

similar to the above

Thus we have no_alias(V'', H)

(3) 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 $locs_{V,H}$)

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

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

tom $l' \in reach_H(v_h)$

$l' \in reach_H(V(x))$ (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$(\{R, F \cup g, locs_{V'',H}(e_2)\})$

Instantiating invariant on J_1,

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

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

(3) stable(R, H, H')

Case 13: E:Drop

$e = \texttt{drop}(x; e')$ (case)

$V', H, R, F \cup g \vdash e' \Downarrow v, H', F'(\mathcal{J}_1)$ (ad.)

$\Gamma = \Gamma', x : A$ (case)

$\Sigma; \Gamma' \,\Big|\frac{q}{q'}\, e' : B$

Suppose $dom(V) = FV(e)$, no_alias(V, H), disjoint$(\{R, F, locs_{V,H}(e)\})$

By IH, we have invariant on $\mathcal{J}_1$

NTS (1) - (3) for $\mathcal{J}_1$

(1) $dom(V') = FV(e')$ ($dom(V) = FV(e)$ and def of FV)

(2) no_alias(V', H) (no_alias(V, H) and $V' \subseteq V$)

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

$g = reach_H(v')$ (case)

$g \subseteq locs_{V,H}(e)$ (def of $locs_{V,H}$)

$R \cap (F \cup g) = \emptyset$

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

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

(disjoint({R,locs$_{V,H}(e)$}) and $locs_{V',H}(e) \subseteq locs_{V,H}(e)$)

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

(disjoint({F,locs$_{V,H}(e)$}) and $locs_{V',H}(e) \subseteq locs_{V,H}(e)$)

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

Instantiating invariant on $\mathcal{J}_1$,

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

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

(3) stable(R, H, H')

Case 13: E:Share

$e = \texttt{shareCopy } x \texttt{ as } x_1, x_2 \texttt{ in } e'$ (case)

Suppose $H \vDash V : \Gamma, dom(V) = FV(e)$, no_alias$(V, H)$, disjoint$(\{R, F, locs_{V,H}(e)\})$

(def. of wtf)

Let $V_2 = V'[x_1 mapstov', x_2 mapstov'']$

We show the subsequent computation is also well-formed to invocate the IH:

(1) $dom(V_2) = FV(e')$ ($dom(V) = FV(e)$ and def of FV)

(2) no_alias$(V'[x_1 \mapsto v', x_2 \mapsto v''], H)$

(no_alias(V, H))

Let $x' \mapsto v''' \in V'[x_1 \mapsto v']$.STS $reach_{H'}(v''') \cap reach_{H'}(v'') = \emptyset$

$reach_{H'}(v'') \subseteq L \subseteq F$ (lemma about copy)

$reach_{H'}(v''') \subseteq locs_{V'[x_1 \mapsto v'],H'}(e') \subseteq locs_{V,H}(e)$

(stability lemma for copy)

(3) $\mathsf{disjoint}(\{R, F \setminus L, locs_{V_2,H'}(e')\})$

By IH:

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

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

STS $\mathsf{stable}(R, H, H')$, which follows from $L \cap R = \emptyset$ and stability for copy

□

Task 1.8 (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

$V, H, R, F \vdash x \Downarrow V(x), H, F$ (admissibility)

$\Sigma; x : B \left|\frac{q}{q}\right. x : B$ (admissibility)

$|F| - |F'|$ (1)

$= |F| - |F|$ (ad.)

$= 0$ (2)

$\Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q')$ (3)

$= \Phi_{V,H}(x : B) + q - (\Phi_H(V(x) : B) + q)$ (ad.)

$= \Phi_H(V(x) : B) + q - (\Phi_H(V(x) : B) + q)$ (def. of $\Phi_{V,H}$)

$= 0$ (4)

$|F| - |F'| \leq \Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q')$ ((3),(5))

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

$|F| - |F'| = |F| - |F|$ (ad.)

$= 0$

$\Phi_{V,H}(\Gamma) + q - (\Phi_{H'}(v : B) + q') = \Phi_{V,H}(\emptyset) + q - (\Phi_H(v : int) + q)$ (ad.)

$= 0$ (def of $\Phi_{V,H}$)

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

Case 4: E:App
Case 5: E:CondT

$\Gamma = \Gamma', x : \mathtt{bool}$ (ad.)

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

$\Sigma; \Gamma' \left|\frac{q}{q'}\right. e_t : B$ (ad.)

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

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

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

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

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 $|F| \geq$

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

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

(Similar to case in Lemma 1.7)

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

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 1.7)

(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 1.7)

(4) $|F'' \cup g| \geq \Phi_{V_2',H_1}(\Gamma_2, x : (A - 1)) + p + C$

STS $|F'' \cup g| \geq \Phi_{V_2,H_1}(\Gamma_2) + \Phi_{H_1}(v_1 : (A - 1)) + p + C$

$|F'' \cup g| \geq \|V_1\|_H + |F| - \|v_1\|_{H_1}$ (conservation lemma)

$\geq \Phi_{V,H}(\Gamma) + q + C + \|V_1\|_H - \|v_1\|_{H_1}$
$(|F| \geq \Phi_H(V) + q + C)$

STS $\Phi_{V_1,H}(\Gamma_1) + q + C \, \|V_1\|_H - \|v_1\|_{H_1} \geq \Phi_{H_1}(v_1 : (A-1)) + p + C$

$\Phi_{V_1,H}(\Gamma_1) \geq \Phi_{H_1}(v_1 : (A-1))$ (lemma about cf typing)

STS $\|V_1\|_H - \|v_1\|_{H_1} + q \geq p$ (done by aux lemma)

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 14: E:Let1

$V, H \vdash e \Downarrow v_2, H_2$ (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 $dom(V) = FV(e)$, no_alias(V, H), 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 $R' = R \cup locs_{V,H}(\mathtt{lam}(x : \tau.e_2))$

disjoint$(\{R', F, locs_{V,H}(e_1)\})$
(Similar to case in Lemma 1.7)

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) no_alias(V_1, H)

(3) disjoint$(\{R, F, locs_{V,H}(e)\})$
((1) - (3) all verbatim as in Lemma 1.7)

(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:

$\Sigma; \Gamma_2, x : A \left|\frac{\max(p,q)}{q'}\right. e_2 : B$ (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 $g = \{l \in H_1 \mid l \notin F'' \cup R \cup locs_{V',H_1}(e_2)\}$

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) no_alias(V_2', H_1)

(3) disjoint$(\{R, F'' \cup g, locs_{V_2',H_1}(e_2)\})$
((1) - (3) all verbatim as in Lemma 1.7)

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

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

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

$V, H \vdash \mathsf{cons}(x_1; x_2) \Downarrow l, H'$ (case)

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

Suppose $dom(V) = FV(e)$, no_alias(V, H), disjoint$(\{R, F, locs_{V,H}(e)\})$, $|F| \geq \Phi_{V,H}$

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 $F' = F$

Case 12: E:MatNil Similar to E:Cond*
Case 13: E:MatCons

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

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

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

$\Sigma; \Gamma', x_h : A, x_t : L^p(A) \left|\frac{q+p+1}{q'}\right. e_2 : B$ (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 no_alias(V, H), 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 $l \in g$

$l \notin F \cup R$ (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 $reach_H(\overline{V}(x')) \cap reach_H((\bar{V}x)) = \emptyset$ from no_alias(V, H))

WLOG let $x' = x_h$

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

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

Hence $l \in g$

Next, we have no_alias(V'', H) and 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

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

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

$= |F| + |g|$ (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)

Case 13: E:Share

$V, H \vdash e \Downarrow v, H''$ (case)

$V'[x_1 \mapsto v', x_2 \mapsto v''], H' \vdash e' \Downarrow v, H''$ (ad)

$\Sigma; \Gamma, x : A \left|\frac{q}{q'}\right. e : B$ (case)

$A \curlyvee A_1, A_2, 1$ (ad.)

$\Sigma; \Gamma, x_1 : A_1, x_2 : A_2 \left|\frac{q}{q'}\right. e : B$ (ad.)

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

Suppose no_alias(V, H), disjoint($\{R, F, locs_{V,H}(e)\}$), and $|F| \geq \Phi_{V,H}(\Gamma, x : A) + q + C$

NTF F'' s.t.

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

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

We need to show the freelist is sufficient for the subsequent computation to invoke the IH:

Instantiate with $C, F \setminus L$, and$R$

STS $|F \setminus L| \geq \Phi_{V_2,H'}(\Gamma, x_1 : A_1, x_2 : A_2) + q + C$

$\iff |F| - |L| \geq \Phi_{V_2,H'}(\Gamma) + \Phi_{V_2,H'}(x_1 : A_1) + \Phi_{V_2,H'}(x_2 : A_2) + q + C$

$\iff |F| \geq \Phi_{V_2,H'}(\Gamma) + \Phi_{V_2,H'}(x_1 : A_1) + \Phi_{V_2,H'}(x_2 : A_2) + \|v'\|_H + q + C$

$\iff |F| \geq \Phi_{V_2,H'}(\Gamma) + \Phi_{V,H}(x : A) + q + C$ (definition of sharing relation)

$\iff |F| \geq \Phi_{V,H}(\Gamma, x : A) + q + C$ (stability of copying)

done from assumption

By IH, we get F'' fulfilling the previous two points for the case.

□

9 COPY-FREE GARBAGE COLLECTION SEMANTICS

Consider the GC semantics (from now on copy semantics) above, with the share rule replaced with the following:

$$\frac{V = V'[x \mapsto v'] \qquad V'[x_1 \mapsto v', x_2 \mapsto v'], H', R, F \ \vdash e \Downarrow v, H'', F'}{V, H, R, F \ \vdash \texttt{share}\ x\ \texttt{as}\ x_1, x_2\ \texttt{in}\ e \Downarrow v, H'', F'} (\mathrm{S}_{22})$$

Call this new semantics free semantics for copy-free. It is easy to see that any terminating compuation in copy has a corresponding one in free that can be instantiated with an equal or smaller freelist. This is expressed as the following lemma:

Lemma 1.9. *Let $V, H, R, F \vdash^{\text{copy}} v, H', F'$. Then there exists an F'' s.t. $V, H, R, F \vdash^{\text{free}} v, H', F''$ and $|F''| \geq |F'|$.*