A PROOF

We give complete proof for the correctness that a program, when fused with a direct manipulation, executes to produce precisely the manipulated output. This is essentially the same as verifying that the bidirectional transformation, established by the fusion and standard evaluation, adheres to the PutGet property. The lemmas used in the proof and their respective proof are also provided below.

Theorem A.1 (PutGet). *If $E \vdash e \Rightarrow v$, $dv \triangleright v \rightsquigarrow v'$, and $dv \triangleright E \vdash e \rightsquigarrow E' \vdash e'$, then $E' \vdash e' \Rightarrow v'$.*

Proof. By induction on the fusion derivation. Below, we show the A-Com, P-App, and P-Case three cases as representatives.

(1) The A-Com case.

$$\frac{dv_1 \triangleright E \vdash e \rightsquigarrow E_1 \vdash e_1 \qquad dv_2 \triangleright E_1 \vdash e_1 \rightsquigarrow E_2 \vdash e_2}{dv_2 \circ dv_1 \triangleright E \vdash e \rightsquigarrow E_2 \vdash e_2}$$

(a) According to D-Com, there must be some v_1 such that $dv_1 \triangleright v \rightsquigarrow v_1$ and $dv_2 \triangleright v_1 \rightsquigarrow v'$.

(b) Apply the induction hypothesis for $E \vdash e \Rightarrow v$ and $dv_1 \triangleright v \rightsquigarrow v_1$, we have $E_1 \vdash e_1 \Rightarrow v_1$.

(c) Apply the induction hypothesis for $E_1 \vdash e_1 \Rightarrow v_1$ and $dv_2 \triangleright v_1 \rightsquigarrow v'$, we have $E_2 \vdash e_2 \Rightarrow v'$, which is the final goal.

(2) The P-App case.

$$\frac{\begin{array}{c} E \vdash e_1 \Rightarrow (E_f, \lambda x.e_f) \qquad E \vdash e_2 \Rightarrow v_2 \qquad dv \triangleright E_f, x \mapsto {v_2}^{\mathtt{id}} \vdash e_f \rightsquigarrow E'_f, x \mapsto v_2^{dv_2 \circ \mathtt{id}} \vdash e'_f \\ (E'_f, \lambda x.e'_f) \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e'_1 \qquad dv_2 \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e'_2 \qquad (E', e''_1, e''_2) = E_1 {}^{e'_1}\!\otimes^{e'_2} E_2 \end{array}}{dv \triangleright E \vdash e_1\ e_2 \rightsquigarrow E' \vdash e''_1\ e''_2}$$

For the goal, we must apply E-App because $e''_1\ e''_2$ is an application. Therefore, we must in turn reason about the evaluation of $E' \vdash e''_1$ and $E' \vdash e''_2$. Both are results of a merge, and are respectively related to $E_1 \vdash e'_1$ and $E_2 \vdash e'_2$.

For $E_1 \vdash e'_1$, it is closely related to the evaluation of $E_1 \vdash e_1$:

(a) Introduce v'_2 such that $dv_2 \triangleright v_2 \rightsquigarrow v'_2$.

(b) Apply the induction hypothesis for $E \vdash e_1 \Rightarrow (E_f, \lambda x.e_f)$ and $(E'_f, \lambda x.e'_f) \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e'_1$, we have $E_1 \vdash e'_1 \Rightarrow (E'_f, \lambda x.e'_f)$.

Similarly, for $E_1 \vdash e'_2$, it is closely related to the evaluation of $E_1 \vdash e_2$:

(c) Apply the induction hypothesis for $E \vdash e_2 \Rightarrow v_2$ and $dv_2 \triangleright v_2 \rightsquigarrow v'_2$, we have $E_2 \vdash e'_2 \Rightarrow v'_2$.

Now we derive the evaluation of $E' \vdash e''_1$ and $E' \vdash e''_2$ respectively from (b) and (c):

(d) By Lemma A.4 and the symmetry of $\otimes$, we have $E' \vdash e''_1 \Rightarrow (E'_f, \lambda x.e'_f)$ and $E' \vdash e''_2 \Rightarrow v'_2$.

We can now apply E-App, and it then suffice to show that $E'_f, x \mapsto {v'_2}^{\mathtt{id}} \vdash e'_f \Rightarrow v'$. Again, we must reason about the evaluation of e'_f, which is related to e_f:

(e) According to E-App, since $E \vdash e_1\ e_2 \Rightarrow v$ and $E \vdash e_2 \Rightarrow v_2$, we have $E, x \mapsto {v_2}^{\mathtt{id}} \vdash e_2 \Rightarrow v$.

(f) Apply the induction hypothesis for $E, x \mapsto {v_2}^{\mathtt{id}} \vdash e_2 \Rightarrow v$ and $dv \triangleright E_f, x \mapsto v_2^{\mathtt{id}} \vdash e_f \rightsquigarrow E'_f, x \mapsto v_2^{dv_2 \circ \mathtt{id}} \vdash e'_f$, we have $E'_f, x \mapsto v_2^{dv_2 \circ \mathtt{id}} \vdash e'_f \Rightarrow v'$.

We are almost done; the above conclusion is slightly different in the environment. By Lemma A.3, it suffice to show the following equivalence: $E'_f, x \mapsto {v'_2}^{\mathtt{id}} \equiv E'_f, x \mapsto v_2^{dv_2 \circ \mathtt{id}}$.

(g) By E-Var, $E'_f, x \mapsto v_2^{dv_2 \circ \mathtt{id}} \vdash x \Rightarrow v'_2$. The two environments agree on the value of x.

(h) They also agree on any other variable, which is determined by the shared E'_f part.

(i) By Definition A.2, the two environments are equivalent.

(3) The P-Case case.

$$\frac{E \vdash e_0 \Rightarrow v_0 \qquad E_m = \mathit{match}\, v_0\, p_j \qquad dv \triangleright E \cup E_m \vdash e_j \rightsquigarrow E_j \cup E'_m \vdash e'_j \\ \mathit{subst}\, E'_m\, p_j \triangleright E \vdash e_0 \rightsquigarrow E_0 \vdash e'_0 \qquad (E', e''_0, e''_j) = E_0 {}^{e'_0}{\otimes}^{e_j'} E_j}{dv \triangleright E \vdash \texttt{case}\ e_0\ \texttt{of}\ \{p_i \longrightarrow e_i\}_{i=1}^{n} \rightsquigarrow E' \vdash \texttt{case}\ e''_0\ \texttt{of}\ \{p_i \longrightarrow e_i\}_{i=1 \wedge i \neq j}^{n} \cup \{p_j \longrightarrow e''_j\}}$$

(a) Introduce v'_1 such that $\mathit{subst}\, E'_m\, p_j \triangleright v_0 \rightsquigarrow v'_1$.
(b) Apply the induction hypothesis for $E \vdash e_0 \Rightarrow v_0$ and $\mathit{subst}\, E'_m\, p_j \triangleright E \vdash e_0 \rightsquigarrow E_0 \vdash e'_0$, we have $E_0 \vdash e' \Rightarrow v'_1$.
(c) By Lemma A.3, we conclude that $E' \vdash e'' \Rightarrow v'_1$.
(d) According to E-Case, since $E \vdash \texttt{case}\ e_0\ \texttt{of}\ \{p_i \longrightarrow e_i\}_{i=1}^{n} \Rightarrow v$, we have $E \cup E_m \vdash e_j \Rightarrow v$.
(e) Apply the induction hypothesis for $E_j \cup E_m \vdash e_j \Rightarrow v$ and $dv \triangleright E \cup E_m \vdash e_j \rightsquigarrow E_j \cup E'_m \vdash e'_j$, we have $E_j \cup E'_m \vdash e'_j \Rightarrow v'$.
(f) By Lemma A.3, we have $E' \cup E'_m \vdash e'_j \Rightarrow v'$.
(g) The algorithm preserves the execution path. By definition, $E'_m \equiv \mathit{match}\, v'_1\, p_j$. Therefore, by Lemma A.4, we have $E' \cup \mathit{match}\, v'_1\, p_j \vdash e'_j \Rightarrow v'$.
(h) By E-Case, $E' \vdash \texttt{case}\ e''\ \texttt{of}\ \{p_i \longrightarrow e_i\}_{i=1 \wedge i \neq j}^{n} \cup \{p_j \longrightarrow e''_j\} \Rightarrow v'$, which is the goal. □

Definition A.2 (Equivalent Environment). E_1 and E_2 are equivalent (denoted $E_1 \equiv E_2$) if they are structural equivalent and $\forall x \in E_1$, $E_1 \vdash x \Rightarrow v$ and $E_2 \vdash x \Rightarrow v$.

Lemma A.3. *If $E_1 \equiv E_2$ and $E_1 \vdash e \Rightarrow v$, then $E_2 \vdash e \Rightarrow v$.*

Proof. By induction on the structure of e. □

Lemma A.4 (Merge Equivalency). *If $(E, e'_1, e'_2) = E_1 {}^{e_1}{\otimes}^{e_2} E_2$, then for every E_0, $E_0 \cup E_1 \vdash e_1 \Rightarrow v$ implies $E_0 \cup E \vdash e'_1 \Rightarrow v$.*

Proof. We use induction on the length n of E_1, so the inductive case involves breaking the environment E_1 into $x \mapsto v^{dv_1}$ (mapping for a single variable) and E'_1 (the rest), apply the induction hypothesis to handle the E'_1 part, and proceed to x for the final conclusion. Therefore, we need to reason about the merge operator $\times$, or more specifically, its behaviour on environments with shape $E_1 = (E'_1, x \mapsto v^{dv_1})$. If we have $(E, E^I_1, E^I_2) = E_1 {}^{e_1}{\times}^{e_2} E_2$ and $(E', E^{I'}_1, E^{I'}_2) = E'_1 {}^{e_1}{\times}^{e_2} E'_2$, then E and E' (or E^I_1 and $E^{I'}_1$) can only differ in the mapping for x. Specifically, the extra deltas[1] for x in E and E^I_1 (compared respectively to E' and $E^{I'}_1$) necessarily compose into the delta for x in E_1 (i.e., dv_1), with the former being the prefix and the latter being the suffix. In case (2), we consider the two different possibilities of this decomposition of dv_1.

By definition of the merge operator ($\otimes$), assume we have $(E, E^I_1, E^I_2) = E_1 {}^{e_1}{\times}^{e_2} E_2$ and $e'_1 = E^I_1 \odot e_1$.

(1) $E_1 = \emptyset$ and $n = 0$. By definition of the merge operator ($\otimes$), we have $E = \emptyset$ and $e_1 = e'_1$. Therefore, the lemma is simple tautology.
(2) Assume for every E'_1 of length n, the lemma is true, i.e., for any E'_2 and E'_0, if $(E', e'_1, e'_2) = E'_1 {}^{e_1}{\otimes}^{e_2} E'_2$ and $E'_0 \cup E'_1 \vdash e_1 \Rightarrow v$, we have $E'_0 \cup E' \vdash e'_1 \Rightarrow v$. Assume we have $(E', E^{I'}_1, E^{I'}_2) = E'_1 {}^{e_1}{\times}^{e_2} E'_2$ and $e'_1 = E^{I'}_1 \odot e_1$. Then consider $E_1 = (E'_1, x \mapsto v_1^{dv_1})$ of length $n+1$. E'_1 is of length n, so the induction hypothesis applies.

[1]In this proof, for our convenience, if a variable x is not mentioned in an environment E, we say that the delta for x in E is id, as if the environment were $(E, x \mapsto v^{\mathrm{id}})$ instead.

(a) If $E(x) = E_1(x) = v_1^{dv_1}$, i.e., E and E_1 agrees on x, let $E = E', x \mapsto v_1^{dv_1}$. Then we have

$$E_0 \cup E_1 = E_0 \cup (E_1', x \mapsto v_1^{dv_1}) = (E_0, x \mapsto v_1^{dv_1}) \cup E_1' = E_0' \cup E_1'$$
$$E_0 \cup E = E_0 \cup (E', x \mapsto v_1^{dv_1}) = (E_0, x \mapsto v_1^{dv_1}) \cup E' = E_0' \cup E'$$

where we define $E_0' = (E_0, x \mapsto v_1^{dv_1})$. Also, since E and E_1 agrees on x, by definition of the merge operator, we have either $E_1^{I'} = E_1^I$ or $(E_1^{I'}, x \mapsto v^{\mathrm{id}}) = E_1^I$, and therefore $E_1^{I'} \odot e_1 = E_1^I \odot e_1$. Now we may apply the induction hypothesis and the lemma is proved.

(b) Otherwise $E = (E', x \mapsto v_1^{dv})$, $E_1^I = (E_1^{I'}, x \mapsto v_1^{dv_1^D})$, and we have the factorisation $dv_1 = dv_1^D \circ dv$. Then we can reason as follows:

$$\begin{aligned}
& E_0 \cup E_1 \vdash e_1 \Rightarrow v \\
\Leftrightarrow\quad & \{\ E_1 = (E_1', x \mapsto v_1^{dv_1}), dv_1 = dv_1^D \circ dv\ \} \\
& E_0 \cup E_1', x \mapsto v_1^{dv_1^D \circ dv} \vdash e_1 \Rightarrow v \\
\Rightarrow\quad & \{\ \text{Lemma A.5}\ \} \\
& E_0 \cup E_1', x \mapsto v_1^{dv} \vdash e_1[x \mapsto \texttt{exp}(dv_1^D)\ x] \Rightarrow v_1 \\
\Leftrightarrow\quad & \{\ \text{Let } E_0' = (E_0, x \mapsto v_1^{dv})\ \} \\
& E_0' \cup E_1' \vdash e_1[x \mapsto \texttt{exp}(dv_1^D)\ x] \Rightarrow v_1 \\
\Rightarrow\quad & \{\ \text{induction hypothesis}\ \} \\
& E_0' \cup E' \vdash (E_1^{I'} \odot e_1[x \mapsto \texttt{exp}(dv_1^D)\ x]) \Rightarrow v_1 \\
\Leftrightarrow\quad & \{\ \text{definition of } \odot\ \} \\
& E_0' \cup E' \vdash ((E_1^{I'}, x \mapsto v_1^{dv_1^D}) \odot e_1) \Rightarrow v_1 \\
\Leftrightarrow\quad & \{\ E_1^I = (E_1^{I'}, x \mapsto v_1^{dv_1^D})\ \} \\
& E_0' \cup E' \vdash (E_1^I \odot e_1) \Rightarrow v_1 \\
\Leftrightarrow\quad & \{\ E = (E', x \mapsto v_1^{dv}), E_0' = (E_0, x \mapsto v_1^{dv})\ \} \\
& E_0 \cup E \vdash (E_1^I \odot e_1) \Rightarrow v_1
\end{aligned}$$

Therefore, by induction on n, the lemma holds for E_1 of any length. □

Lemma A.5. *If* $E, x \mapsto v^{dv_1 \circ dv_2} \vdash e \Rightarrow v_1$, *then* $E, x \mapsto v^{dv_2} \vdash e[x \mapsto \texttt{exp}(dv_1)\ x] \Rightarrow v_1$.

Proof. By induction on the structure of e. □

B SEMANTICS OF DELTA LANGUAGE

$$[\text{D-Id}]\ \frac{}{\texttt{id} \triangleright v \rightsquigarrow v} \qquad [\text{D-Repl}]\ \frac{\emptyset \vdash aexp \Rightarrow v'}{\texttt{repl}\ aexp \triangleright v \rightsquigarrow v'} \qquad [\text{D-Com}]\ \frac{dv_1 \triangleright v \rightsquigarrow v' \qquad dv_2 \triangleright v' \rightsquigarrow v''}{dv_2 \circ dv_1 \triangleright v \rightsquigarrow v''}$$

$$[\text{D-Add}]\ \frac{}{+_\Delta atom \triangleright n \rightsquigarrow n + \texttt{eval}(atom)} \qquad [\text{D-Mul}]\ \frac{}{*_\Delta atom \triangleright n \rightsquigarrow n * \texttt{eval}(atom)}$$

$$[\text{D-Tuple}]\ \frac{dv_1 \triangleright v_1 \rightsquigarrow v_1' \qquad dv_2 \triangleright v_2 \rightsquigarrow v_2'}{(dv_1, dv_2)_\Delta \triangleright (v_1, v_2) \rightsquigarrow (v_1', v_2')} \qquad [\text{D-Cons}]\ \frac{dv_1 \triangleright v_1 \rightsquigarrow v_1' \qquad dv_2 \triangleright v_2 \rightsquigarrow v_2'}{dv_1 ::_\Delta dv_2 \triangleright v_1 :: v_2 \rightsquigarrow v_1' :: v_2'}$$

$$[\text{D-Mod-1}]\ \frac{n > 0 \qquad \texttt{modify}\ (n-1)\ dv \triangleright v_2 \rightsquigarrow v_2'}{\texttt{modify}\ n\ dv \triangleright v_1 :: v_2 \rightsquigarrow v_1 :: v_2'} \qquad [\text{D-Mod-2}]\ \frac{dv \triangleright v_1 \rightsquigarrow v_1'}{\texttt{modify}\ 0\ dv \triangleright v_1 :: v_2 \rightsquigarrow v_1' :: v_2}$$

$$[\text{D-Ins-1}]\ \frac{n > 0 \qquad \texttt{insert}\ (n-1)\ atom \triangleright v_2 \rightsquigarrow v_2'}{\texttt{insert}\ n\ atom \triangleright v_1 :: v_2 \rightsquigarrow v_1 :: v_2'} \qquad [\text{D-Ins-2}]\ \frac{}{\texttt{insert}\ 0\ atom \triangleright v_1 :: v_2 \rightsquigarrow atom :: v_1 :: v_2}$$

$$[\text{D-Del-1}]\ \frac{n > 0 \qquad \texttt{delete}\ (n-1) \triangleright v_2 \rightsquigarrow v_2'}{\texttt{delete}\ n \triangleright v_1 :: v_2 \rightsquigarrow v_1 :: v_2'} \qquad [\text{D-Del-2}]\ \frac{}{\texttt{delete}\ 0 \triangleright v_1 :: v_2 \rightsquigarrow v_2}$$

$$[\text{D-Fold}]\ \frac{\begin{array}{c}\emptyset \vdash \mathit{derive}\ \mathit{acc} \Rightarrow (\mathit{arg}, \mathit{acc}') \qquad \mathit{todelta} = \lambda x.dv \qquad dv_1 = dv[x \mapsto \mathit{arg}] \\ dv_1 \triangleright v_1 \rightsquigarrow v_1' \qquad \texttt{dfold}\ \mathit{derive}\ \mathit{todelta}\ \mathit{acc}' \triangleright v_2 \rightsquigarrow v_2'\end{array}}{\texttt{dfold}\ \mathit{derive}\ \mathit{todelta}\ \mathit{acc} \triangleright v_1 :: v_2 \rightsquigarrow v_1' :: v_2'}$$

$$[\text{D-Constraint}]\ \frac{v_1 = v\ |_v\ \mathit{select} \qquad dv[x \mapsto v_1] \triangleright v \rightsquigarrow v'}{\texttt{intro}\ x\ \texttt{by}\ \mathit{select}\ \texttt{into}\ dv \triangleright v \rightsquigarrow v'}$$

C SEMANTICS OF SELECTORS FOR VALUES

$(v_1 :: v_2)$	$\|_v\ \texttt{head} \circ \mathit{select}$	$= v_1\ \|_v\ \mathit{select}$	[S-Head]	$(v_1 :: v_2)$	$\|_v\ \texttt{tail} \circ \mathit{select}$	$= v_2\ \|_v\ \mathit{select}$	[S-Tail]
(v_1, v_2)	$\|_v\ \texttt{fst} \circ \mathit{select}$	$= v_1\ \|_v\ \mathit{select}$	[S-Fst]	(v_1, v_2)	$\|_v\ \texttt{snd} \circ \mathit{select}$	$= v_2\ \|_v\ \mathit{select}$	[S-Snd]
v	$\|_v\ \texttt{id}$	$= v$	[S-Id]				

D PROPAGATION RULES OF FUSION ALGORITHM

$$[\text{E-Var}]\ \frac{E(x) = v^{dv} \qquad dv \triangleright v \rightsquigarrow v'}{E \vdash x \Rightarrow v'} \qquad [\text{P-Var}]\ \frac{E(x) = v^{dv}}{dv_1 \triangleright E \vdash x \rightsquigarrow E[x \mapsto v^{dv_1 \circ dv}] \vdash x}$$

$$[\text{E-Lam}]\ \frac{}{E \vdash \lambda p.e \Rightarrow (E, \lambda p.e)} \qquad [\text{P-Lam}]\ \frac{}{(E', \lambda p.e') \triangleright E \vdash \lambda p.e \rightsquigarrow E' \vdash \lambda p.e'}$$

$$[\text{E-App}]\ \frac{E \vdash e_1 \Rightarrow (E_f, \lambda x.e_f) \qquad E \vdash e_2 \Rightarrow v_2 \qquad E_f, x \mapsto v_2^{\texttt{id}} \vdash e_f \Rightarrow v}{E \vdash e_1\ e_2 \Rightarrow v}$$

$$[\text{P-App}]\ \frac{\begin{array}{c}E \vdash e_1 \Rightarrow (E_f, \lambda x.e_f) \qquad E \vdash e_2 \Rightarrow v_2 \qquad dv \triangleright E_f, x \mapsto v_2^{\texttt{id}} \vdash e_f \rightsquigarrow E_f', x \mapsto v_2^{dv_2 \circ \texttt{id}} \vdash e_f' \\ (E_f', \lambda x.e_f') \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e_1' \qquad dv_2 \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1'', e_2'') = E_1{}^{e_1'}\otimes^{e_2'} E_2\end{array}}{dv \triangleright E \vdash e_1\ e_2 \rightsquigarrow E' \vdash e_1''\ e_2''}$$

$$[\text{E-Case}]\ \frac{E \vdash e_0 \Rightarrow v_0 \qquad E_m = \mathit{match}\ v_0\ p_j \qquad E \cup E_m \vdash e_j \Rightarrow v_j}{E \vdash \texttt{case}\ e_0\ \texttt{of}\ \{p_i \rightarrow e_i\}_{i=1}^{n} \Rightarrow v_j}$$

$$[\text{P-Case}]\ \frac{\begin{array}{c}E \vdash e_0 \Rightarrow v_0 \qquad E_m = \mathit{match}\ v_0\ p_j \qquad dv \triangleright E \cup E_m \vdash e_j \rightsquigarrow E_j \cup E_m' \vdash e_j' \\ \mathit{subst}\ E_m'\ p_j \triangleright E \vdash e_0 \rightsquigarrow E_0 \vdash e_0' \qquad (E', e_0'', e_j'') = E_0{}^{e_0'}\otimes^{e_j'} E_j\end{array}}{dv \triangleright E \vdash \texttt{case}\ e_0\ \texttt{of}\ \{p_i \rightarrow e_i\}_{i=1}^{n} \rightsquigarrow E' \vdash \texttt{case}\ e_0''\ \texttt{of}\ \{p_i \rightarrow e_i\}_{i=1 \wedge i \neq j}^{n} \cup \{p_j \rightarrow e_j''\}}$$

$$[\text{P-Add}]\ \frac{\begin{array}{c}+_\Delta n \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e_1' \\ (E', e_1'', e_2') = E_1{}^{e_1'}\otimes^{e_2} E\end{array}}{+_\Delta n \triangleright E \vdash e_1 + e_2 \rightsquigarrow E' \vdash e_1'' + e_2'} \qquad [\text{P-Mul}]\ \frac{\begin{array}{c}E \vdash e_2 \Rightarrow n_2 \\ +_\Delta (n/n_2) \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e_1' \qquad (E', e_1'', e_2') = E_1{}^{e_1'}\otimes^{e_2} E\end{array}}{+_\Delta n \triangleright E \vdash e_1 * e_2 \rightsquigarrow E' \vdash e_1'' * e_2'}$$

E APPLICATION RULES OF FUSION ALGORITHM

$$[\text{A-Id}]\ \frac{}{\texttt{id} \triangleright E \vdash e \rightsquigarrow E \vdash e} \qquad [\text{A-Add}]\ \frac{n = n_1 + n_2}{+_\Delta n_1 \triangleright E \vdash n_2 \rightsquigarrow E \vdash n} \qquad [\text{A-Mul}]\ \frac{n = n_1 * n_2}{*_\Delta n_1 \triangleright E \vdash n_2 \rightsquigarrow E \vdash n}$$

$$[\text{A-Repl}]\ \frac{}{\texttt{repl}\ \mathit{aexp} \triangleright E \vdash e \rightsquigarrow E \vdash \mathit{aexp}} \qquad [\text{A-Com}]\ \frac{dv_1 \triangleright E \vdash e \rightsquigarrow E_1 \vdash e_1 \qquad dv_2 \triangleright E_1 \vdash e_1 \rightsquigarrow E_2 \vdash e_2}{dv_2 \circ dv_1 \triangleright E \vdash e \rightsquigarrow E_2 \vdash e_2}$$

$$[\text{A-Tuple}]\ \frac{dv_1 \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e_1' \qquad dv_2 \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1'', e_2'') = E_1{}^{e_1'}\otimes^{e_2'} E_2}{(dv_1, dv_2) \triangleright E \vdash (e_1, e_2) \rightsquigarrow E' \vdash (e_1'', e_2'')}$$

$$[\text{A-Cons}]\ \frac{dv_1 \triangleright E \vdash e_1 \rightsquigarrow E_1 \vdash e_1' \qquad dv_2 \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1'', e_2'') = E_1{}^{e_1'}\otimes^{e_2'} E_2}{dv_1 :: dv_2 \triangleright E \vdash e_1 :: e_2 \rightsquigarrow E' \vdash e_1'' :: e_2''}$$

$$[\text{A-Mod}]\ \frac{n > 0 \qquad \texttt{modify}\ (n-1)\ dv \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1', e_2'') = E^{e_1}\otimes^{e_2'} E_2}{\texttt{modify}\ n\ dv \triangleright E \vdash e_1 :: e_2 \rightsquigarrow E' \vdash e_1' :: e_2''}$$

$$[\text{A-Ins}]\ \frac{n > 0 \qquad \texttt{insert}\ (n-1)\ \mathit{atom} \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1', e_2'') = E^{e_1} \otimes^{e_2'} E_2}{\texttt{insert}\ n\ \mathit{atom} \triangleright E \vdash e_1 :: e_2 \rightsquigarrow E' \vdash e_1' :: e_2''}$$

$$[\text{A-Del}]\ \frac{n > 0 \qquad \texttt{delete}\ (n-1) \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad (E', e_1', e_2'') = E^{e_1} \otimes^{e_2'} E_2}{\texttt{delete}\ n \triangleright E \vdash e_1 :: e_2 \rightsquigarrow E' \vdash e_1' :: e_2''}$$

$$[\text{A-Constraint}]\ \frac{\begin{array}{c} e\ |_e\ \mathit{select}_x = e_2; e_f \qquad E \vdash e_2 \Rightarrow v_2 \qquad dv \triangleright E, x \mapsto v_2^{\texttt{id}} \vdash e_f \rightsquigarrow E_1, x \mapsto v_2^{dv_2 \circ \texttt{id}} \vdash e_f' \\ dv_2 \triangleright E \vdash e_2 \rightsquigarrow E_2 \vdash e_2' \qquad\qquad (E', e_f'', e_2'') = E_1{}^{e_f'} \otimes^{e_2'} E_2 \end{array}}{\texttt{intro}\ x\ \texttt{by}\ \mathit{select}\ \texttt{into}\ dv \triangleright E \vdash e \rightsquigarrow E' \vdash (\lambda x. e_f'')\ e_2''}$$

F SEMANTICS OF SELECTORS FOR EXPRESSIONS

$$[\text{S-Head}]\ \frac{e_1\ |_e\ \mathit{select}_x = e; e_f}{e_1 :: e_2\ |_e\ (\texttt{head} \circ \mathit{select})_x = e; e_f :: e_2} \qquad [\text{S-Tail}]\ \frac{e_2\ |_e\ \mathit{select}_x = e; e_f}{e_1 :: e_2 (\texttt{tail} \circ \mathit{select})_x = e; e_1 :: e_f}$$

$$[\text{S-Fst}]\ \frac{e_1\ |_e\ \mathit{select}_x = e; e_f}{(e_1, e_2)\ |_e\ (\texttt{fst} \circ \mathit{select})_x = e; (e_f, e_2)} \quad [\text{S-Snd}]\ \frac{e_2\ |_e\ \mathit{select}_x = e; e_f}{(e_1, e_2)\ |_e\ (\texttt{snd} \circ \mathit{select})_x = e; (e_1, e_f)} \quad [\text{S-Id}]\ \frac{}{e\ |_e\ \texttt{id}_x = e; x}$$

G TRANSFORMATIONS OF DELTAS TO EXPRESSION FUNCTIONS

$$(1)\quad \texttt{exp}(\texttt{id}) = \lambda x.x$$

$$(2)\quad \texttt{exp}(\texttt{repl}\ e) = \lambda x.e$$

$$(3)\quad \texttt{exp}(+_\Delta n) = \lambda x.x + n$$

$$(4)\quad \texttt{exp}(*_\Delta n) = \lambda x.x * n$$

$$(5)\quad \texttt{exp}((E, \lambda p.e)) = \lambda f.\lambda p.E \odot e$$

$$(6)\quad \texttt{exp}(dv_1 \circ dv_2) = \lambda x.\texttt{exp}(dv_1)\ (\texttt{exp}(dv_2)\ x)$$

$$(7)\quad \texttt{exp}((dv_1, dv_2)_\Delta) = \lambda(x, y).((\texttt{exp}(dv_1)\ x), (\texttt{exp}(dv_2)\ y))$$

$$(8)\quad \texttt{exp}(dv_1 :: dv_2) = \lambda x :: xs.(\texttt{exp}(dv_1)\ x) :: (\texttt{exp}(dv_2)\ xs)$$

$$(9)\quad \texttt{exp}(\texttt{modify}\ n\ dv) = \mathit{modify}\ n\ \texttt{exp}(dv)$$

$$(10)\quad \texttt{exp}(\texttt{insert}\ n\ \mathit{atom}) = \mathit{insert}\ n\ \mathit{atom}$$

$$(11)\quad \texttt{exp}(\texttt{delete}\ n) = \mathit{delete}\ n$$

$$\begin{aligned}(12)\quad & \texttt{exp}(\texttt{dfold}\ \mathit{derive}\ (\lambda a.dv)\ \mathit{acc}) = \\ & \qquad \lambda ls.\texttt{let}\ f = \lambda x.\lambda(\mathit{res}, \mathit{acc}). \\ & \qquad\qquad \texttt{let}\ (\mathit{arg}, \mathit{acc}') = \mathit{derive}\ \mathit{acc}\ \texttt{in} \\ & \qquad\qquad \texttt{let}\ f_1 = \texttt{exp}(dv[a \mapsto \mathit{arg}])\ \texttt{in} \\ & \qquad\qquad\quad (\mathit{concat}\ \mathit{res}\ [f_1\ x], \mathit{acc}') \\ & \qquad\quad \texttt{in}\ \mathit{foldl}\ f\ ([], \mathit{acc})\ ls \end{aligned}$$

$$(13)\quad \texttt{exp}(\texttt{intro}\ x\ \texttt{by}\ \mathit{select}\ \texttt{into}\ dv) = \lambda y.(\lambda x.\texttt{exp}(dv)\ y)\ (\mathit{select}\ y)$$