SNESL formalization Level-0

Dandan Xue

September 21, 2017

0 Level-0

Draft version 0.0.7:

- changed WithCtrl: added import and export list
- adjusted section structure
- small changes of some function notations
- Note: the symbols/functions used in the main correctness theroem have not been updated yet

1 Source Language

1.1 Source language syntax

SNESL Expressions:

$$e ::= x \mid \textbf{let } x = e_1 \textbf{ in } e_2 \mid \phi(x_1, ..., x_k) \mid \{e : x \textbf{ in } y \textbf{ using } \cdot\}$$

$$\phi = \textbf{const}_n \mid \textbf{iota} \mid \textbf{plus}$$

Values:

$$n \in \mathbf{Z}$$

$$v ::= n \mid \{v_1, ..., v_k\}$$

1.2 Type system

$$\tau ::= \textbf{int} | \{\tau_1\}$$

Type environment $\Gamma = [x_1 \mapsto \tau_1, ..., x_i \mapsto \tau_i]$.

- Expression typing rules:

  Judgment $\boxed{\Gamma \vdash e : \tau}$

  $$\frac{}{\Gamma \vdash x : \tau}\ (\Gamma(x) = \tau) \qquad \frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma[x \mapsto \tau_1] \vdash e_2 : \tau}{\Gamma \vdash \textbf{let } x = e_1 \textbf{ in } e_2 : \tau}$$

  $$\frac{\phi : (\tau_1, ..., \tau_k) \to \tau}{\Gamma \vdash \phi(x_1, ..., x_k) : \tau}\ ((\Gamma(x_i) = \tau_i)_{i=1}^k) \qquad \frac{[x \mapsto \tau_1] \vdash e : \tau}{\Gamma \vdash \{e : x \textbf{ in } y \textbf{ using } \cdot\} : \{\tau\}}\ (\Gamma(y) = \{\tau_1\})$$

- Auxiliary Judgment $\boxed{\phi : (\tau_1, ..., \tau_k) \to \tau}$

  $$\frac{}{\textbf{const}_n : () \to \textbf{int}} \qquad \frac{}{\textbf{iota} : (\textbf{int}) \to \{\textbf{int}\}} \qquad \frac{}{\textbf{plus} : (\textbf{int}, \textbf{int}) \to \textbf{int}}$$

- Value typing rules:

  Judgment $\boxed{v : \tau}$

  $$\frac{}{n : \mathbf{int}} \qquad \frac{(v_i : \tau)_{i=1}^{k}}{\{v_1, ..., v_k\} : \{\tau\}}$$

1.3 Source language semantics

$\rho = [x_1 \mapsto v_1, ..., x_i \mapsto v_i]$

- Judgment $\boxed{\rho \ \vdash \ e \downarrow v}$

  $$\frac{}{\rho \ \vdash \ x \downarrow v} \ (\rho(x) = v) \qquad \frac{\rho \ \vdash \ e_1 \downarrow v_1 \qquad \rho[x \mapsto v_1] \ \vdash \ e_2 \downarrow v}{\rho \ \vdash \ \mathbf{let} \ e_1 = x \ \mathbf{in} \ e_2 \downarrow v}$$

  $$\frac{\phi(v_1, ..., v_k) \ \vdash \ v}{\rho \ \vdash \ \phi(x_1, ..., x_k) \downarrow v} \ ((\rho(x_i) = v_i)_{i=1}^{k})$$

  $$\frac{([x \mapsto v_i] \ \vdash \ e \downarrow v_i')_{i=1}^{k}}{\rho \ \vdash \ \{e : x \ \mathbf{in} \ y \ \mathbf{using} \ \cdot\} \downarrow \{v_1', ..., v_k'\}} \ (\rho(y) = \{v_1, ..., v_k\})$$

- Auxiliary Judgment $\boxed{\phi(v_1, ..., v_k) \ \vdash \ v}$

  $$\frac{}{\mathbf{const}_n() \ \vdash \ n} \qquad \frac{}{\mathbf{iota}(n) \ \vdash \ \{0, 1, ..., n-1\}} \ (n \geq 0)$$

  $$\frac{}{\mathbf{plus}(n_1, n_2) \ \vdash \ n_3} \ (n_3 = n_1 + n_2)$$

2 Target language

2.1 SVCODE syntax

(1) Stream id:

$$s \in \mathbf{SId} = \mathbf{N} = \{0, 1, 2...\}$$

A list of **SId**:

$$\mathbf{S} = [s_1, ..., s_i]$$

Note: for simplicity, in some cases where duplicate of some elements does not affect the correctness we will also use $\mathbf{S}$ to stand for a set of **SId**s which keeps only one copy of each element in $\mathbf{S}$.

(2) SVCODE operations:

$$\psi ::= \texttt{Const}_{\texttt{a}} \mid \texttt{ToFlags} \mid \texttt{Usum} \mid \texttt{MapTwo}_{\oplus} \mid \texttt{ScanPlus}_{n_0}$$

where $\oplus$ stands for some binary operation on **int**.

(3) SVCODE program:

$$\begin{aligned} p ::= \ & \epsilon \\ & \mid s := \psi(s_1, ..., s_i); p_1 \\ & \mid \mathbf{S}_{out} := \texttt{WithCtrl}(s, \mathbf{S}_{in}, p_1); p_2 \qquad (\mathbf{S}_{in} = \{s\} \cup \texttt{fv}(p_1), \mathbf{S}_{out} \subseteq \texttt{dv}(p_1)) \end{aligned}$$

Note: here this $\mathbf{S}_{in}$ is actually slightly different from the import list used in our streaming interpreter which does not contain the new control stream id. So it may cause some confusion later when we introduce the streaming language. We may want to remove this $\mathbf{S}_{in}$ from WithCtrl instruction if it turns out it is not necessary to make it an explicit component here.

(4) Differentce of sets:
For two sets A and B,

$$A - B = \{s | s \in A, s \notin B\}$$

It is easy to prove the following properties:

- For any three sets A, B and C:

$$(A - B) \cap C = (A \cap C) - B = A \cap (C - B)$$

- For two sets A and B,

$$A \cap B = \emptyset \Leftrightarrow A - B = A$$

(5) Free variables: a set (or list) of stream ids that are not defined but referred to by a SVCODE program

$$\begin{aligned}
&\texttt{fv}(\epsilon) \ = \{\} \\
&\texttt{fv}(s := \psi(s_1, ..., s_i); p_1) = \{s_1, ..., s_i\} \cup \texttt{fv}(p_1) - \{s\} \\
&\texttt{fv}(\mathbf{S}_{out} := \texttt{WithCtrl}(s_c, \mathbf{S}_{in}, p_1); p_2) = \mathbf{S}_{in} \cup \texttt{fv}(p_2) - \mathbf{S}_{out}
\end{aligned}$$

(6) Defined variables: a set (or list) of stream ids that are defined by a SVCODE progrom and accessible to the outside environment of the program

$$\begin{aligned}
&\texttt{dv}(\epsilon) \ = \{\} \\
&\texttt{dv}(s := \psi(s_1, ..., s_i); p_1) = \{s\} \cup \texttt{dv}(p_1) \\
&\texttt{dv}(\mathbf{S}_{out} := \texttt{WithCtrl}(s_c, \mathbf{S}_{in}, p_1); p_2) = \mathbf{S}_{out} \cup \texttt{dv}(p_2)
\end{aligned}$$

(7) SVCODE streams:

$$\begin{aligned}
b &\in \{\texttt{T}, \texttt{F}\} \\
a &::= n \mid b \mid () \\
\vec{b} &= \langle b_1, ..., b_i \rangle \\
\vec{a} &= \langle a_1, ..., a_i \rangle
\end{aligned}$$

(8)

(9) Notations and operations about streams:

- For some a_0 and $\vec{a} = \langle a_1, ..., a_i \rangle$, let $\langle a_0 | \vec{a} \rangle = \langle a_0, a_1, ..., a_i \rangle$.
- $\langle a_1, ..., a_i \rangle ++ \langle a'_1, ..., a'_j \rangle = \langle a_1, ..., a_i, a'_1, ..., a'_j \rangle$

2.2 SVCODE semantics

SVCODE stores $\sigma = [s_1 \mapsto \vec{a}_1, ..., s_i \mapsto \vec{a}_i]$.

- Judgment $\boxed{\langle p, \sigma \rangle \downarrow^{\vec{c}} \sigma'}$

  $\vec{c}$ is the control stream.

  P-Empty : $\dfrac{}{\langle \epsilon, \sigma \rangle \downarrow^{\vec{c}} \sigma}$

$$\text{P-Xducer}: \frac{\psi(\vec{a}_1,...,\vec{a}_k)\downarrow^{\vec{c}}\vec{a} \qquad \langle p_1,\sigma[s\mapsto\vec{a}]\rangle\downarrow^{\vec{c}}\sigma'}{\langle s:=\psi(s_1,...,s_k);p_1,\sigma\rangle\downarrow^{\vec{c}}\sigma'}\ ((\sigma(s_i)=\vec{a}_i)_{i=1}^k)$$

$$\text{P-Wc-Emp}: \frac{\langle p_2,\sigma[s_1\mapsto\langle\rangle,...,s_i\mapsto\langle\rangle]\rangle\downarrow^{\vec{c}}\sigma'}{\langle \mathbf{S}_{out}:=\texttt{WithCtrl}(s_c,\mathbf{S}_{in},p_1);p_2,\sigma\rangle\downarrow^{\vec{c}}\sigma'}\begin{pmatrix}\forall s\in\mathbf{S}_{in}.\sigma(s)=\langle\rangle\\ \mathbf{S}_{out}=[s_1,...,s_i]\end{pmatrix}$$

$$\text{P-Wc-Nonemp}: \frac{\langle p_1,\sigma\rangle\downarrow^{\vec{c}_1}\sigma'' \qquad \langle p_2,\sigma[s_1\mapsto\sigma''(s_1),...,s_i\mapsto\sigma''(s_i)]\rangle\downarrow^{\vec{c}}\sigma'}{\langle \mathbf{S}_{out}:=\texttt{WithCtrl}(s_c,\mathbf{S}_{in},p_1);p_2,\sigma\rangle\downarrow^{\vec{c}}\sigma'}\begin{pmatrix}\sigma(s_c)=\vec{c}_1=\langle()|...\rangle\\ \mathbf{S}_{out}=[s_1,...,s_i]\end{pmatrix}$$

$$\text{P-Seq}: \frac{\langle p_1,\sigma\rangle\downarrow^{\vec{c}}\sigma'' \qquad \langle p_2,\sigma''\rangle\downarrow^{\vec{c}}\sigma'}{\langle p_1;p_2,\sigma\rangle\downarrow^{\vec{c}}\sigma'}$$

- *Transducer* semantics:

  Judgment $\boxed{\psi(\vec{a}_1,...,\vec{a}_k)\downarrow^{\vec{c}}\vec{a}}$

  $$\text{P-X-Loop}: \frac{\psi(\vec{a}_{11},...,\vec{a}_{k1})\Downarrow\vec{a}_1 \qquad \psi(\vec{a}_{12},...,\vec{a}_{k2})\downarrow^{\vec{c}}\vec{a}_2}{\psi(\vec{a}_{11}{+\!+}\vec{a}_{12},...,\vec{a}_{k1}{+\!+}\vec{a}_{k2})\downarrow^{\langle a_0|\vec{c}\rangle}\vec{a}}\ (\vec{a}=\vec{a}_1{+\!+}\vec{a}_2)$$

  $$\text{P-X-Termi}: \frac{}{\psi(\langle\rangle_1,...,\langle\rangle_k)\downarrow^{\langle\rangle}\langle\rangle}\ ^{1}$$

- Transducer *block* semantics:

  Judgment $\boxed{\psi(\vec{a}_1,...,\vec{a}_k)\Downarrow\vec{a}}$

  $$\text{P-Const}: \frac{}{\texttt{Const}_{\texttt{a}}()\Downarrow\langle a\rangle}$$
  $$\text{P-ToFlags}: \frac{}{\texttt{ToFlags}(\langle n\rangle)\Downarrow\langle \texttt{F}_1,...,\texttt{F}_n,\texttt{T}\rangle}$$

  $$\text{P-MapTwo}: \frac{}{\texttt{MapTwo}_{\oplus}(\langle n_1\rangle,\langle n_2\rangle)\Downarrow\langle n_3\rangle}\ (n_3=n_1\oplus n_2)$$

  $$\text{P-UsumF}: \frac{\texttt{Usum}(\vec{b})\Downarrow\vec{a}}{\texttt{Usum}(\langle\texttt{F}|\vec{b}\rangle)\Downarrow\langle()|\vec{a}\rangle}$$
  $$\text{P-UsumT}: \frac{}{\texttt{Usum}(\langle\texttt{T}\rangle)\Downarrow\langle\rangle}$$

  $$\text{P-ScanF}: \frac{\texttt{ScanPlus}_{n_0+n}(\vec{b},\vec{a})\Downarrow\vec{a}'}{\texttt{ScanPlus}_{n_0}(\langle\texttt{F}|\vec{b}\rangle,\langle n|\vec{a}\rangle)\Downarrow\langle n_0|\vec{a}'\rangle}$$
  $$\text{P-ScanT}: \frac{}{\texttt{ScanPlus}_{n_0}(\langle\texttt{T}\rangle,\langle\rangle)\Downarrow\langle\rangle}$$

  Or if we want to use *unary* semantics maybe for later:

  $$\frac{\psi(\langle\texttt{F}\rangle,...,\vec{a}_{k1})\Downarrow\!\Downarrow\vec{a}_1 \qquad \psi(\vec{a}_{12},...,\vec{a}_{k2})\Downarrow\vec{a}_2}{\psi(\langle\texttt{F}\rangle{+\!+}\vec{a}_{12},...,\vec{a}_{k1}{+\!+}\vec{a}_{k2})\Downarrow\vec{a}}\ (\vec{a}=\vec{a}_1{+\!+}\vec{a}_2)$$

  $$\frac{\psi(\langle\texttt{T}\rangle,...,\vec{a}_k)\Downarrow\!\Downarrow\vec{a}}{\psi(\langle\texttt{T}\rangle,...,\vec{a}_k)\Downarrow\vec{a}}$$

[1] For convenience, in this thesis we add subscripts to a sequence of constants, such as $\langle\rangle,\texttt{F},1$, to denote the total number of these constants.

– Transducer *unary* semantics:

Judgment $\boxed{\psi(\langle b\rangle, ..., \vec{a}_k) \downdownarrows \vec{a}}$

$$\frac{}{\texttt{Usum}(\langle \texttt{F}\rangle) \downdownarrows \langle()\rangle} \qquad \frac{}{\texttt{Usum}(\langle \texttt{T}\rangle) \downdownarrows \langle\rangle}$$

– Transducer block with *accumulator*:

Judgment $\boxed{\psi_n(\vec{a}_1, ..., \vec{a}_k) \Downarrow \vec{a}}$

$$\frac{\psi_{n_0}(\langle \texttt{F}\rangle, ..., \vec{a}_{k1}) \downdownarrows^{n'_0} \langle n_1\rangle \qquad \psi_{n'_0}(\vec{a}_{12}, ..., \vec{a}_{k2}) \Downarrow \vec{a}_2}{\psi_{n_0}(\langle \texttt{F}\rangle{+\!+}\vec{a}_{12}, ..., \vec{a}_{k1}{+\!+}\vec{a}_{k2}) \Downarrow \langle n_1\rangle{+\!+}\vec{a}_2}$$

$$\frac{\psi_{n_0}(\langle \texttt{T}\rangle, ..., \vec{a}_k) \downdownarrows \vec{a}}{\psi_{n_0}(\langle \texttt{T}\rangle, ..., \vec{a}_k) \Downarrow \vec{a}}$$

– Transducer unary with *accumulator*:

Judgment $\boxed{\psi_n(\langle \texttt{F}\rangle, ..., \vec{a}_k) \downdownarrows^{n'} \vec{a}}$

$$\frac{}{\texttt{ScanPlus}_{n_0}(\langle \texttt{F}\rangle, \langle n\rangle) \downdownarrows^{n_0+n} \langle n_0\rangle}$$

Judgment $\boxed{\psi_n(\langle \texttt{T}\rangle, ..., \vec{a}_k) \downdownarrows \vec{a}}$

$$\frac{}{\texttt{ScanPlus}_{n_0}(\langle \texttt{T}\rangle, \langle\rangle) \downdownarrows \langle\rangle}$$

2.3 Definitions

We first define a binary relation $\overset{\mathbf{S}}{\sim}$ on stores to denote that two stores are *similar*: they have identical domains, and their bound values by $\mathbf{S}$ are the same. We call this $\mathbf{S}$ an *overlap* of these two stores.

Definition 2.1 (**Stores similarity**). $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_2$ *iff*
(1) $dom(\sigma_1) = dom(\sigma_2)$
(2) $\forall s \in \mathbf{S}.\sigma_1(s) = \sigma_2(s)$

According to this definition, it is only meaningful to have $\mathbf{S} \subseteq dom(\sigma_1)$ $(= dom(\sigma_2))$. When $\mathbf{S} = dom(\sigma_1) = dom(\sigma_2)$, σ_1 and σ_2 are identical. It is easy to show that this relation $\overset{\mathbf{S}}{\sim}$ is transitive.

- If $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_2$ and $\sigma_2 \overset{\mathbf{S}}{\sim} \sigma_3$, then $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_3$.

We define another binary operation $\overset{\mathbf{S}}{\bowtie}$ on stores to denote a kind of specical concatenation of two similar stores: the *concatenation* of two similar stores is a new store, in which the bound values by $\mathbf{S}$ are from any of the parameter stores, and the others are the concatenation of the values from the two stores. In other words, a *concatenation* of two similar stores is only a concatenation of the bound values that *maybe* different in these stores.

Definition 2.2. $\sigma_1 \overset{\mathbf{S}}{\bowtie} \sigma_2 = \sigma$ *iff*
(1) $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_2$
(2) $\sigma(s) = \begin{cases} \sigma_i(s), & s \in \mathbf{S}, i \in \{1, 2\} \\ \sigma_1(s){+\!+}\sigma_2(s), & otherwise \end{cases}$

Lemma 2.1. *If* $\sigma_1 \overset{\mathbf{S}}{\bowtie} \sigma_2 = \sigma$, *then* $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma$ *and* $\sigma_2 \overset{\mathbf{S}}{\sim} \sigma$.

This lemma says that the concatenation result of two similar stores is still similar to each of them.

Lemma 2.2. *If* $\psi(\vec{a}_{11},...,\vec{a}_{1k}) \downarrow^{\vec{c}_1} \vec{a}_1$, *and* $\psi(\vec{a}_{21},...,\vec{a}_{2k}) \downarrow^{\vec{c}_2} \vec{a}_2$, *then* $\psi(\vec{a}_{11}{+\!\!+}\vec{a}_{21},...,\vec{a}_{1k}{+\!\!+}\vec{a}_{2k}) \downarrow^{\vec{c}_1{+\!\!+}\vec{c}_2} \vec{a}_1{+\!\!+}\vec{a}_2$.

Lemma 2.3. *If*

(i) $\sigma_1 \overset{S}{\sim} \sigma_2$

(ii) $\langle p, \sigma_1\rangle \downarrow^{\vec{c}} \sigma$

(iii) $\mathtt{fv}(p) \cap \mathbf{S} = \emptyset$

(iv) $\forall s \in \mathtt{fv}(p).\sigma_2(s) = \langle\rangle$

then

(v) $\langle p, \sigma_1 \overset{\mathbf{S}}{\bowtie} \sigma_2\rangle \downarrow^{\vec{c}} \sigma'$

(vi) $\forall s' \in \mathtt{dv}(p).\sigma(s') = \sigma'(s')$

Lemma 2.4 (**Stores concatenation lemma**). *If*

(i) $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_2$

(ii) $\langle p, \sigma_1\rangle \downarrow^{\vec{c}_1} \sigma_1'$ *(by some derivation* $\mathcal{P}_1$*)*

(iii) $\langle p, \sigma_2\rangle \downarrow^{\vec{c}_2} \sigma_2'$ *(by some derivation* $\mathcal{P}_2$*)*

(iv) $\mathtt{fv}(p) \cap \mathbf{S} = \emptyset$

then $\langle p, \sigma_1 \overset{\mathbf{S}}{\bowtie} \sigma_2\rangle \downarrow^{\vec{c}_1{+\!\!+}\vec{c}_2} \sigma_1' \overset{\mathbf{S}}{\bowtie} \sigma_2'$ *(by* $\mathcal{P}$*).*

We need this lemma to prove that the results of single computations inside a comprehension body (i.e. p in the lemma) can be concatenated to express a parallel computation. From the other direction, we can consider this process as distributing or splitting the compuation p on even smaller degree of parallel computations, in which all the supplier streams, i.e., $\mathtt{fv}(p)$, are splitted to feed the transducers. The splitted parallel degrees are specified by the control streams, i.e., $\vec{c}_1$ and $\vec{c}_2$ in the lemma. Other untouched **SId**s in all σs (i.e., $\mathbf{S}$) have no change throughout the process.

Let $\sigma_1 \overset{<s}{=\!=} \sigma_2$ denote $\forall s' < s.\sigma_1(s') = \sigma_2(s')$.

Lemma 2.5. *If* $\sigma_1 \overset{\mathbf{S}_1}{\sim} \sigma_1'$, $\sigma_2 \overset{\mathbf{S}_2}{\sim} \sigma_2'$, $\sigma_1 \overset{<s}{=\!=} \sigma_2$, *and* $\sigma_1' \overset{<s}{=\!=} \sigma_2'$ *then* $\sigma_1 \overset{\mathbf{S}_1}{\bowtie} \sigma_1' \overset{<s}{=\!=} \sigma_2 \overset{\mathbf{S}_2}{\bowtie} \sigma_2'$.

2.4 SVCODE determinism theroem

Definition 2.3. $\vec{a}$ *is a prefix of* $\vec{a}'$ *if* $\vec{a} \sqsubseteq \vec{a}'$*:*
Judgment $\boxed{\vec{a} \sqsubseteq \vec{a}'}$

$$\frac{}{\langle\rangle \sqsubseteq \vec{a}} \qquad \frac{\vec{a} \sqsubseteq \vec{a}'}{\langle a_0|\vec{a}\rangle \sqsubseteq \langle a_0|\vec{a}'\rangle}$$

Lemma 2.6. *If*

(i) $(\vec{a}_i' \sqsubseteq \vec{a}_i)_{i=1}^k$ *and* $\psi(\vec{a}_1',...,\vec{a}_k') \Downarrow \vec{a}'$,

(ii) $(\vec{a}_i'' \sqsubseteq \vec{a}_i)_{i=1}^k$ *and* $\psi(\vec{a}_1'',...,\vec{a}_k'') \Downarrow \vec{a}''$

then

(i) $(\vec{a}_i' = \vec{a}_i'')_{i=1}^k$

(ii) $\vec{a}' = \vec{a}''$.

Lemma 2.7. *If* $\psi(\vec{a}_1,...,\vec{a}_k) \downarrow^{\vec{c}} \vec{a}$, *and* $\psi(\vec{a}_1,...,\vec{a}_k) \downarrow^{\vec{c}} \vec{a}'$, *then* $\vec{a} = \vec{a}'$.

Theorem 2.1 (**SVCODE determinism**). *If* $\langle p, \sigma\rangle \downarrow^{\vec{c}} \sigma'$ *and* $\langle p, \sigma\rangle \downarrow^{\vec{c}} \sigma''$, *then* $\sigma' = \sigma''$.

3 Translation

3.1 Translation rules

(1) Stream tree:

$$\mathbf{STree} \ni st ::= s \mid (st_1, s)$$

(2) Convert a stream tree to a list of stream ids:

$$\begin{aligned} &\bar{}: \mathbf{STree} \to \mathbf{S} \\ &\overline{s} = [s] \\ &\overline{(st, s)} = \overline{st} +\!\!+ [s] \end{aligned}$$

(3) Translation environment:

$$\delta = [x_1 \mapsto st_1, ..., x_i \mapsto st_i]$$

- Judgment $\boxed{\delta \ \vdash \ e \Rightarrow_{s_1}^{s_0} (p, st)}$

$$\frac{}{\delta \ \vdash \ x \Rightarrow_{s_0}^{s_0} (\epsilon, st)} \ (\delta(x) = st) \qquad \frac{\delta \ \vdash \ e_1 \Rightarrow_{s_0'}^{s_0} (p_1, st_1) \qquad \delta[x \mapsto st_1] \ \vdash \ e_2 \Rightarrow_{s_1}^{s_0'} (p_2, st)}{\delta \ \vdash \ \mathbf{let} \ x = e_1 \ \mathbf{in} \ e_2 \Rightarrow_{s_1}^{s_0} (p_1; p_2, st)}$$

$$\frac{\phi(st_1, ..., st_k) \Rightarrow_{s_1}^{s_0} (p, st)}{\delta \ \vdash \ \phi(x_1, ..., x_k) \Rightarrow_{s_1}^{s_0} (p, st)} \ ((\delta(x_i) = st_i)_{i=1}^k)$$

$$\frac{[x \mapsto st_1] \ \vdash \ e \Rightarrow_{s_1}^{s_0+1} (p_1, st)}{\delta \ \vdash \ \{e : x \ \mathbf{in} \ y \ \mathbf{using} \ \cdot\} \Rightarrow_{s_1}^{s_0} (s_0 := \texttt{Usum}(s_2); \mathbf{S}_{out} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), (st, s_2))} \begin{pmatrix} \delta(y) = (st_1, s_2) \\ \mathbf{S}_{in} = \{s_0\} \cup \texttt{fv}(p_1) \\ \mathbf{S}_{out} = \overline{st} \cap \texttt{dv}(p_1) \end{pmatrix}$$

- Auxiliary Judgment $\boxed{\phi(st_1, ..., st_k) \Rightarrow_{s_1}^{s_0} (p, st)}$

$$\frac{}{\mathbf{const}_n() \Rightarrow_{s_0+1}^{s_0} (s_0 := \texttt{Const}_\texttt{n}(), s_0)}$$

$$\frac{}{\mathbf{iota}(s) \Rightarrow_{s_4}^{s_0} (p, (s_3, s_0))} \begin{pmatrix} s_{i+1} = s_i + 1, \forall i \in \{0, ..., 3\} \\ p = s_0 := \texttt{ToFlags}(s); \\ \quad s_1 := \texttt{Usum}(s_0); \\ \quad \overline{s_2} := \texttt{WithCtrl}(s_1, [s_1], s_2 := \texttt{Const}_1()); \\ \quad s_3 := \texttt{ScanPlus}_0(s_0, s_2) \end{pmatrix}$$

$$\frac{}{\mathbf{plus}(s_1, s_2) \Rightarrow_{s_0+1}^{s_0} (s_0 := \texttt{MapTwo}_+(s_1, s_2), s_0)}$$

3.2 Value representation

1. SVCODE values:

$$\mathbf{SvVal} \ni w ::= \vec{a} \mid (w, \vec{b})$$

2. SVCODE values concatenation:

$$\begin{aligned} &+\!\!+ : \mathbf{SvVal} \to \mathbf{SvVal} \to \mathbf{SvVal} \\ &\langle \vec{a}_1, ..., \vec{a}_i \rangle +\!\!+ \langle \vec{a}'_1, ..., \vec{a}'_j \rangle = \langle \vec{a}_1, ..., \vec{a}_i, \vec{a}'_1, ..., \vec{a}'_j \rangle \\ &(w_1, \vec{b}_1) +\!\!+ (w_2, \vec{b}_2) = (w_1 +\!\!+ w_2, \vec{b}_1 +\!\!+ \vec{b}_2) \end{aligned}$$

3. SVCODE value construction from a stream tree:

$$\begin{aligned} &\sigma : \mathbf{STree} \to \mathbf{SvVal} \\ &\sigma(s) = \vec{a} \\ &\sigma((st, s)) = (\sigma(st), \sigma(s)) \end{aligned}$$

4. Value representation rules

   - Judgment $\boxed{v \rhd_\tau w}$

$$\frac{}{n \rhd_{\mathbf{int}} \langle n \rangle} \qquad \frac{(v_i \rhd_\tau w_i)_{i=1}^k}{\{v_1, ..., v_k\} \rhd_{\{\tau\}} (w, \langle \mathtt{F}_1, ..., \mathtt{F}_k, \mathtt{T} \rangle)} \, (w = w_1 {+\!+} ... {+\!+} w_k)$$

Lemma 3.1 (**Value translation backwards determinism**). *If $v \rhd_\tau w$, $v' \rhd_\tau w$, then $v = v'$.*

3.3 Correctness proof

Lemma 3.2 (???). *If $\Gamma \vdash e : \{\tau\}$, $\rho \vdash e \downarrow \{v_1, ..., v_k\}$, and $\delta \vdash e \Rightarrow_{s_1}^{s_0} (p, (st, s))$, then $s \notin \mathtt{sids}(st)$.*

Lemma 3.3. *If*

(i) $\phi : (\tau_1, ..., \tau_k) \to \tau$ *(by some derivation $\mathcal{T}$)*

(ii) $\phi(v_1, ..., v_k) \vdash v$ *(by $\mathcal{E}$)*

(iii) $\phi(st_1, ..., st_k) \Rightarrow_{s_1}^{s_0} (p, st)$ *(by $\mathcal{C}$)*

(iv) $(v_i \rhd_{\tau_i} \sigma(st_i))_{i=1}^k$

(v) $\bigcup_{i=1}^k \mathtt{sids}(st_i) \lessdot s_0$

then

(vi) $\langle p, \sigma \rangle \downarrow^{\langle () \rangle} \sigma'$ *(by $\mathcal{P}$)*

(vii) $v \rhd_\tau \sigma'(st)$ *(by $\mathcal{R}$)*

(viii) $\sigma' \overset{< s_0}{=\!=} \sigma$

(ix) $s_0 \leq s_1$

(x) $\mathtt{sids}(st) \lessdot s_1$

Proof. By inducntion on the syntax of ϕ.

- Case $\phi = \mathbf{const}_n$
  There is only one possibility for each of $\mathcal{T}$, $\mathcal{E}$ and $\mathcal{C}$:

$$\mathcal{T} = \frac{}{\mathbf{const}_n : () \to \mathbf{int}}$$

$$\mathcal{E} = \frac{}{\vdash \mathbf{const}_n() \downarrow n}$$

$$\mathcal{C} = \frac{}{\mathbf{const}_n() \Rightarrow_{s_0+1}^{s_0} (s_0 := \mathtt{Const_n}(), s_0)}$$

  So $k = 0, \tau = \mathbf{int}, v = n, p = s_0 := \mathtt{Const_n}()$, $s_1 = s_0 + 1$, and $st = s_0$
  By P-Xducer, P-X-Loop, P-X-Termi and P-Const, we can construct $\mathcal{P}$ as follows:

$$\mathcal{P} = \frac{\dfrac{\dfrac{}{\mathtt{Const_n}() \Downarrow \langle n \rangle} \quad \dfrac{}{\mathtt{Const_n}() \downarrow^{\langle \rangle} \langle \rangle}}{\mathtt{Const_n}() \downarrow^{\langle () \rangle} \langle n \rangle}}{\langle s_0 := \mathtt{Const_n}(), \sigma \rangle \downarrow^{\langle () \rangle} \sigma[s_0 \mapsto \langle n \rangle]}$$

  So $\sigma' = \sigma[s_0 \mapsto \langle n \rangle]$.
  Then we take $\mathcal{R} = \dfrac{}{n \rhd_{\mathbf{int}} \sigma'(s_0)}$.
  Also clearly, $\sigma' \overset{< s_0}{=\!=} \sigma$, $s_0 \leq s_0 + 1$, $\mathtt{sids}(s_0) \lessdot s_0 + 1$, and we are done.

- Case $\phi = \mathbf{plus}$
  We must have
  $$\mathcal{T} = \frac{}{\mathbf{plus} : (\mathbf{int}, \mathbf{int}) \to \mathbf{int}}$$
  $$\mathcal{E} = \frac{}{\vdash\ \mathbf{plus}(n_1, n_2) \downarrow n_3}$$
  where $n_3 = n_2 + n_1$, and
  $$\mathcal{C} = \frac{}{\mathbf{plus}(s_1, s_2) \Rightarrow^{s_0}_{s_0+1} (s_0 := \mathtt{MapTwo}_+(s_1, s_2), s_0)}$$
  So $k = 2, \tau_1 = \tau_2 = \tau = \mathbf{int}, v_1 = n_1, v_2 = n_2, v = n_3, st_1 = s_1, st_2 = s_2, st = s_0, s_1 = s_0 + 1$ and $p = s_0 := \mathtt{MapTwo}_+(s_1, s_2)$.

  Assumption (iv) gives us $\overline{n_1 \triangleright_{\mathbf{int}} \sigma(s_1)}$ and $\overline{n_2 \triangleright_{\mathbf{int}} \sigma(s_2)}$, which implies $\sigma(s_1) = \langle n_1 \rangle$ and $\sigma(s_2) = \langle n_2 \rangle$ respectively.

  For (v) we have $s_1 < s_0$ and $s_2 < s_0$.

  Then using P-XDUCER with $\sigma(s_1) = \langle n_1 \rangle$ and $\sigma(s_2) = \langle n_2 \rangle$, and using P-X-LOOP and P-X-TERMI, we can build $\mathcal{P}$ as follows:
  $$\frac{\dfrac{\overline{\mathtt{MapTwo}_+(\langle n_1 \rangle, \langle n_2 \rangle) \Downarrow \langle n_3 \rangle} \quad \overline{\mathtt{MapTwo}_+(\langle\rangle, \langle\rangle) \downarrow^{\langle\rangle}\ \langle\rangle}}{\mathtt{MapTwo}_+(\langle n_1 \rangle, \langle n_2 \rangle) \downarrow^{\langle()\rangle}\ \langle n_3 \rangle}}{\langle s_0 := \mathtt{MapTwo}_+(s_1, s_2), \sigma \rangle \downarrow^{\langle()\rangle}\ \sigma[s_0 \mapsto \langle n_3 \rangle]}$$
  Therefore, $\sigma' = \sigma[s_0 \mapsto \langle n_3 \rangle]$.
  Now we can take $\mathcal{R} = \overline{n_3 \triangleright_{\mathbf{int}} \sigma'(s_0)}$, and it is clear that $\sigma' \overset{<s_0}{=\!=} \sigma$, $s_0 \leq s_0 + 1$ and $\mathtt{sids}(s_0) \lessdot s_0 + 1$ as required.
- Case $\phi = \mathbf{iota}$

□

Theorem 3.1. ***If***

(i) $\Gamma\ \vdash\ e : \tau$ *(by some derivation* $\mathcal{T}$*)*

(ii) $\rho\ \vdash\ e \downarrow v$ *(by some* $\mathcal{E}$*)*

(iii) $\delta\ \vdash\ e \Rightarrow^{s_0}_{s_1} (p, st)$ *(by some* $\mathcal{C}$*)*

(iv) $\forall x \in dom(\Gamma).\ \vdash\ \rho(x) : \Gamma(x) \wedge \mathtt{sids}(\delta(x)) \lessdot s_0 \wedge \rho(x) \triangleright_{\Gamma(x)} \sigma(\delta(x))$
then

(v) $\langle p, \sigma \rangle \downarrow^{\langle()\rangle} \sigma'$ *(by some derivation* $\mathcal{P}$*)*

(vi) $v \triangleright_\tau \sigma'(st)$ *(by some* $\mathcal{R}$*)*

(vii) $\sigma' \overset{<s_0}{=\!=} \sigma$

(viii) $s_0 \leq s_1$

(ix) $\mathtt{sids}(st) \lessdot s_1$

Proof. By induction on the syntax of e.

- Case $e = \{e_1 : x\ \mathbf{in}\ y\ \mathbf{using}\ \cdot\}$.

  We must have:

(i) $$\mathcal{T} = \frac{\overset{\mathcal{T}_1}{[x \mapsto \tau_1] \ \vdash \ e_1 : \tau_2}}{\Gamma \ \vdash \ \{e_1 : x \ \textbf{in} \ y \ \textbf{using} \ \cdot\} : \{\tau_2\}} \ (\Gamma(y) = \{\tau_1\})$$

(ii) $$\mathcal{E} = \frac{\overset{\mathcal{E}_i}{([x \mapsto v_i] \ \vdash \ e_1 \downarrow v_i')_{i=1}^k}}{\rho \ \vdash \ \{e_1 : x \ \textbf{in} \ y \ \textbf{using} \ \cdot\} \downarrow \{v_1', ..., v_k'\}} \ (\rho(y) = \{v_1, ..., v_k\})$$

(iii) $$\mathcal{C} = \frac{\overset{\mathcal{C}_1}{[x \mapsto st_1] \ \vdash \ e_1 \Rightarrow_{s_1}^{s_0+1} (p_1, st_2)}}{\delta \ \vdash \ \{e_1 : x \ \textbf{in} \ y \ \textbf{using} \ \cdot\} \Rightarrow_{s_1}^{s_0} (s_0 := \texttt{Usum}(s_2); \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), (st_2, s_2))} \ (\delta(y) = (st_1, s_2))$$

So $p = (s_0 := \texttt{Usum}(s_2); \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1)), \tau = \{\tau_2\}, v = \{v_1', ..., v_k'\}, st = (st_2, s_2)$.

(iv) $\vdash \ \rho(y) : \Gamma(y)$ gives us $\vdash \ \{v_1, ..., v_k\} : \{\tau_1\}$, which must have the derivation:

$$\frac{(\vdash \ v_i : \tau_1)_{i=1}^k}{\vdash \ \{v_1, ..., v_k\} : \{\tau_1\}} \tag{1}$$

$\texttt{sids}(\delta(y)) \lessdot s_0$ gives us

$$\texttt{sids}(\delta(y)) = \texttt{sids}((st_1, s_2)) = \texttt{sids}(st_1) \cup \{s_2\} \lessdot s_0 \tag{2}$$

$\rho(y) \triangleright_{\Gamma(y)} \sigma(\delta(y)) = \{v_1, ..., v_k\} \triangleright_{\{\tau_1\}} \sigma((st_1, s_2))$ must have the derivation:

$$\frac{\overset{\mathcal{R}_i}{(v_i \triangleright_{\tau_1} w_i)_{i=1}^k}}{\{v_1, ..., v_k\} \triangleright_{\{\tau_1\}} (w, \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle)} \ (w = w_1 \ +\!\!+ ... +\!\!+ \ w_k) \tag{3}$$

therefore

$$\sigma(st_1) = w \tag{4}$$

and

$$\sigma(s_2) = \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle. \tag{5}$$

First we shall show:

(v) $\langle s_0 := \texttt{Usum}(s_2); \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma \rangle \downarrow^{\langle () \rangle} \sigma'$ by some $\mathcal{P}$

(vi) $\{v_1', ..., v_k'\} \triangleright_{\{\tau_2\}} \sigma'((st_2, s_2))$ by some $\mathcal{R}$

(vii) $\sigma' \overset{< s_0}{=\!=} \sigma$

By P-SEQ, we can build $\mathcal{P}$ as follow:

$$\frac{\overset{\mathcal{P}_0}{\langle s_0 := \texttt{Usum}(s_2), \sigma \rangle \downarrow^{\langle () \rangle} \sigma_0} \qquad \overset{\mathcal{P}_1}{\langle \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma_0 \rangle \downarrow^{\langle () \rangle} \sigma'}}{\langle s_0 := \texttt{Usum}(s_2); \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma \rangle \downarrow^{\langle () \rangle} \sigma'}$$

By P-XDUCER with $\sigma(s_2) = \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle$, we can continue to build $\mathcal{P}_0$ as follow:

$$\mathcal{P}_0 = \frac{\overset{\mathcal{P}_0'}{\texttt{Usum}(\langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle) \downarrow^{\langle () \rangle} \vec{a}}}{\langle s_0 := \texttt{Usum}(s_2), \sigma \rangle \downarrow^{\langle () \rangle} \sigma[s_0 \mapsto \vec{a}]}$$

So $\sigma_0 = \sigma[s_0 \mapsto \vec{a}]$.
We split $\langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle$ into two parts: $\vec{b}_1 = \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T} \rangle$ and $\vec{b}_2 = \langle \rangle$.
By P-X-LOOP and P-X-TERMI with $\vec{b}_1$ and $\vec{b}_2$, we continue building $\mathcal{P}_0'$ as follow:

$$\mathcal{P}_0' = \dfrac{\dfrac{\mathcal{P}_0''}{\texttt{Usum}(\vec{b}_1) \Downarrow \vec{a}} \qquad \dfrac{}{\texttt{Usum}(\vec{b}_2) \downarrow^{\langle\rangle} \langle\rangle}}{\texttt{Usum}(\langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T}\rangle) \downarrow^{\langle()\rangle} \vec{a}}$$

Then using P-UsumF k times and P-UsumT once, we obtain

$$\mathcal{P}_0'' = \dfrac{\dfrac{\dfrac{}{\texttt{Usum}(\langle \texttt{T}\rangle) \Downarrow \langle\rangle}}{\vdots}}{\texttt{Usum}(\langle \texttt{F}_2, ..., \texttt{F}_k, \texttt{T}\rangle) \Downarrow \langle()_2, ..., ()_k\rangle}}{\texttt{Usum}(\langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T}\rangle) \Downarrow \langle()_1, ..., ()_k\rangle}$$

Thus so far we have constructed $\mathcal{P}_0$ of $\langle s_0 := \texttt{Usum}(s_2), \sigma\rangle \downarrow^{\langle()\rangle} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]$.
Since we have

$$\begin{aligned} \mathcal{T}_1 = \ & [x \mapsto \tau_1] \ \vdash \ e_1 : \tau_2 \\ (\mathcal{E}_i = \ & [x \mapsto v_i] \ \vdash \ e_1 \downarrow v_i' \qquad)_{i=1}^k \\ \mathcal{C}_1 = \ & [x \mapsto st_1] \ \vdash \ e_1 \Rightarrow_{s_1}^{s_0+1} (p_1, st_2) \end{aligned}$$

Let $\Gamma_1 = [x \mapsto \tau_1], \rho_i = [x \mapsto v_i]$ and $\delta_1 = [x \mapsto st_1]$.
From (1) and (2) it is clear that

$$\forall z \in dom(\Gamma_1). \ \vdash \ \rho_i(z) : \Gamma_1(z) \wedge \texttt{sids}(\delta_1(z)) \lessdot s_0.$$

Let i range from 1 to k: we take $\sigma_i \overset{st_1}{\sim} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]$ such that $\sigma_i(st_1) = w_i$.
From $\mathcal{R}_i$ in (3) we know that

$$\forall z \in dom(\Gamma_1). \rho_i(z) \triangleright_{\Gamma_1(z)} \sigma_i(\delta_1(z)).$$

Then by IH (k times) on $\mathcal{T}_1$ with $\mathcal{E}_i, \mathcal{C}_1$ we obtain the following result:

$$(\langle p_1, \sigma_i\rangle \downarrow^{\langle()\rangle} \sigma_i')_{i=1}^k \tag{6}$$

$$(v_i' \triangleright_{\tau_2} \sigma_i'(st_2))_{i=1}^k \tag{7}$$

$$(\sigma_i' \overset{\leq s_0+1}{=\!=\!=} \sigma_i)_{i=1}^k \tag{8}$$

$$s_0 + 1 \leq s_1 \tag{9}$$

$$\texttt{sids}(st_2) \lessdot s_1 \tag{10}$$

Assume $\texttt{sids}(st_2) = \{s_1', ..., s_j'\}$.

There are two possibilities:

- Subcase $k = 0$, that is $\sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle](s_0) = \langle\rangle$.
  By P-Wc-Emp We build

  $$\mathcal{P}_1 = \dfrac{}{\langle \overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma\rangle \downarrow^{\langle()\rangle} \sigma[s_0 \mapsto \langle\rangle, s_1' \mapsto \langle\rangle, ..., s_j' \mapsto \langle\rangle]}$$

  So in this subcase,

  $$\sigma' = \sigma[s_0 \mapsto \langle\rangle, s_1' \mapsto \langle\rangle, ..., s_j' \mapsto \langle\rangle].$$

  Since $k = 0$, then $v = \{\}$, $\sigma(s_2) = \langle\texttt{T}\rangle$ (from (5)), we have
  $\sigma'(s_2) = \sigma(s_2) = \langle\texttt{T}\rangle$ (?? not correct if $s_2 \in \texttt{sids}(st_2)/\texttt{sids}(st_1)$),
  and $\sigma'(st_2) = (...((\langle\rangle, \langle\rangle)_1, \langle\rangle)_2, ...)_{j-1}$.

  Therefore $\sigma'((st_2, s_2)) = (\sigma'(st_2), \sigma'(s_2))$, with which we construct

  $$\mathcal{R} = \dfrac{}{\{\} \triangleright_{\{\tau_2\}} ((...(\langle\rangle, \langle\rangle)_1, ...)_{j-1}, \langle\texttt{T}\rangle)}$$

as required.

Since $k = 0$, from (4) we know $\forall s' \in \texttt{sids}(st_1).\sigma(s') = \langle\rangle$. For any $s' \in \texttt{sids}(st_2)$ and $s' < s_0$, it must have $s' \in \texttt{sids}(st_1)$ (because $codom(\delta_1) = \{st_1\}$), hence $\sigma(s') = \langle\rangle = \sigma'(s')$. Therefore,

$$\sigma' \overset{<s_0}{=\!=} \sigma.$$

– Subcase $k > 0$, that is $\sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle] = \langle()|\vec{a}\rangle$ for some $\vec{a}$.
By P-Wc-Nonemp, we take $\mathcal{P}_1 =$

$$\frac{\mathcal{P}_1' \qquad \langle p_1, \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]\rangle \downarrow^{\langle()_1,...,()_k\rangle} \sigma''}{\overline{st_2} := \texttt{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]\rangle \downarrow^{\langle()\rangle} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle, s_1' \mapsto \sigma''(s_1'), ..., s_j' \mapsto \sigma''(s_j')]}$$

So in this subcase

$$\sigma' = \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle, s_1' \mapsto \sigma''(s_1'), ..., s_j' \mapsto \sigma''(s_j')].$$

Using Lemma 2.4 (k-1) times on (6) gives us

$$\langle p_1, (\overset{st_1}{\bowtie} \sigma_i)_{i=1}^k\rangle \downarrow^{\langle()_1,...,()_k\rangle} (\overset{st_2'}{\bowtie} \sigma_i')_{i=1}^k \tag{11}$$

where $st_2' = \texttt{sids}(st_1) \cup \texttt{sids}(st_2)$ (???) .
By Definition 2.2 we have

$$(\overset{st_1}{\bowtie} \sigma_i)_{i=1}^k = \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]. \tag{12}$$

Then by Theorem 2.1 on $\mathcal{P}_1'$ with (11), we get

$$\sigma'' = (\overset{st_2'}{\bowtie} \sigma_i')_{i=1}^k \tag{13}$$

Therefore, $\sigma''(st_2) = \sigma_1'(st_2)++...++\sigma_k'(st_2)$ by Definition 2.2.
Let $\sigma_i'(st_2) = w_i'$ and $\sigma''(st_2) = w'$, then $w' = w_1'++...++w_k'$.

Since $\sigma'(st_2) = \sigma''(st_2) = w'$, and $\sigma'(s_2) = \sigma(s_2) = \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T}\rangle$, we now have $\sigma'((st_2, s_2)) = (\sigma'(st_2), \sigma'(s_2)) = (w', \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T}\rangle)$. With (7), we can construct

$$\mathcal{R} = \frac{(v_i' \triangleright_{\tau_2} w_i')_{i=1}^k}{\{v_1', ..., v_k'\} \triangleright_{\{\tau_2\}} (w', \langle \texttt{F}_1, ..., \texttt{F}_k, \texttt{T}\rangle)}$$

as required.

By Lemma 2.1 on (12) we get $\sigma_i \overset{st_1}{\sim} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle]$, and similarly $\sigma_i' \overset{st_2'}{\sim} \sigma''$ from (13).

Since (8) implies

$$(\sigma_i' \overset{<s_0}{=\!=} \sigma_i)_{i=1}^k$$

using Lemma 2.5 (k-1) times, we obtain

$$\sigma'' \overset{<s_0}{=\!=} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle].$$

Therefore, $\sigma' \overset{<s_0}{=\!=} \sigma[s_0 \mapsto \langle()_1, ..., ()_k\rangle] \overset{<s_0}{=\!=} \sigma$.

(viii) TS: $s_0 \leq s_1$
From (9) we immediately get $s_0 \leq s_1 - 1 < s_1$.

(ix) TS: $\texttt{sids}((st_2, s_2)) \lessdot s_1$
From (2) we know $s_2 < s_0$, thus $s_2 < s_0 \leq s_1$. And we already have (10). Therefore,

$$\texttt{sids}((st_2, s_2)) = \texttt{sids}(st_2) \cup \{s_2\} \lessdot s_1.$$

- Case $e = x$.
We must have

$$\mathcal{T} = \frac{}{\Gamma \ \vdash \ x : \tau}\,(\Gamma(x) = \tau)$$
$$\mathcal{E} = \frac{}{\rho \ \vdash \ x \downarrow v}\,(\rho(x) = v)$$
$$\mathcal{C} = \frac{}{\delta \ \vdash \ x \Rightarrow_{s_0}^{s_0} (\epsilon, st)}\,(\delta(x) = st)$$

So $p = \epsilon$.
Immediately we have $\mathcal{P} = \dfrac{}{\langle \epsilon, \sigma \rangle \downarrow^{\langle () \rangle} \sigma}$
So $\sigma' = \sigma$, which implies $\sigma' \overset{< s_0}{=\!=} \sigma$.
From the assumption we already have $v \triangleright_\tau \sigma(st)$, and $\texttt{sids}(st) \lessdot s_0$.
Finally it's clear that $s_0 \leq s_0$, and we are done.

- Case $e = \textbf{let}\ x = e_1\ \textbf{in}\ e_2$.
We must have:

$$\mathcal{T} = \frac{\mathcal{T}_1 \qquad \mathcal{T}_2}{\Gamma \ \vdash \ e_1 : \tau_1 \qquad \Gamma[x \mapsto \tau_1] \ \vdash \ e_2 : \tau}{\Gamma \ \vdash \ \textbf{let}\ x = e_1\ \textbf{in}\ e_2 : \tau}$$

$$\mathcal{E} = \frac{\mathcal{E}_1 \qquad \mathcal{E}_2}{\rho \ \vdash \ e_1 \downarrow v_1 \qquad \rho[x \mapsto v_1] \ \vdash \ e_2 \downarrow v}{\rho \ \vdash \ \textbf{let}\ x = e_1\ \textbf{in}\ e_2 \downarrow v}$$

$$\mathcal{C} = \frac{\mathcal{C}_1 \qquad \mathcal{C}_2}{\delta \ \vdash \ e_1 \Rightarrow_{s_0'}^{s_0} (p_1, st_1) \qquad \delta[x \mapsto st_1] \ \vdash \ e_2 \Rightarrow_{s_1}^{s_0'} (p_2, st)}{\delta \ \vdash \ \textbf{let}\ x = e_1\ \textbf{in}\ e_2 \Rightarrow_{s_1}^{s_0} (p_1; p_2, st)}$$

So $p = p_1; p_2$.

By IH on $\mathcal{T}_1$ with $\mathcal{E}_1, \mathcal{C}_1$, we get

(a) $\mathcal{P}_1$ of $\langle p_1, \sigma \rangle \downarrow^{\langle () \rangle} \sigma_1$
(b) $\mathcal{R}_1$ of $v_1 \triangleright_{\tau_1} \sigma_1(st_1)$
(c) $\sigma_1 \overset{< s_0}{=\!=} \sigma$
(d) $s_0 \leq s_0'$
(e) $\texttt{sids}(st_1) \lessdot s_0'$

From (b), we know $\rho[x \mapsto v_1](x) : \Gamma[x \mapsto \tau_1](x)$ and $\rho[x \mapsto v_1](x) \triangleright_{\Gamma[x \mapsto \tau_1](x)} \sigma_1(\delta[x \mapsto st_1](x))$ must hold. From (e), we have $\texttt{sids}(\delta[x \mapsto st_1](x)) \lessdot s_0'$.

Then by IH on $\mathcal{T}_2$ with $\mathcal{E}_2, \mathcal{C}_2$, we get

(f) $\mathcal{P}_2$ of $\langle p_2, \sigma_1 \rangle \downarrow^{\langle () \rangle} \sigma_2$
(g) $\mathcal{R}_2$ of $\sigma_2 \triangleright_\tau \sigma_2(st)$
(h) $\sigma_2 \overset{< s_0'}{=\!=} \sigma_1$
(i) $s_0' \leq s_1$
(j) $\texttt{sids}(st) \lessdot s_1$

So we can construct:

$$\mathcal{P} = \frac{\begin{array}{cc}\mathcal{P}_1 & \mathcal{P}_2\\ \langle p_1, \sigma\rangle \downarrow^{\langle()\rangle} \sigma_1 & \langle p_2, \sigma_1\rangle \downarrow^{\langle()\rangle} \sigma_2\end{array}}{\langle p_1; p_2, \sigma\rangle \downarrow^{\langle()\rangle} \sigma_2}$$

From (c), (d) and (h), it is clear that $\sigma_2 \overset{<s_0}{=\!=} \sigma_1 \overset{<s_0}{=\!=} \sigma$. From (d) and (i), $s_0 \leq s_1$. Take $\sigma' = \sigma_2$ (thus $\mathcal{R} = \mathcal{R}_2$) and we are done.

- Case $e = \phi(x_1, ..., x_k)$
  We must have

$$\mathcal{T} = \frac{\begin{array}{c}\mathcal{T}_1\\ \phi : (\tau_1, ..., \tau_k) \to \tau\end{array}}{\Gamma \ \vdash \ \phi(x_1, ..., x_k) : \tau} \ ((\Gamma(x_i) = \tau_i)_{i=1}^k)$$

$$\mathcal{E} = \frac{\begin{array}{c}\mathcal{E}_1\\ \vdash \ (v_1, ..., v_k)(\downarrow) \ \vdash \ v\end{array}}{\rho \ \vdash \ \phi(x_1, ..., x_k) \downarrow v} \ ((\rho(x_i) = v_i)_{i=1}^k)$$

$$\mathcal{C} = \frac{\begin{array}{c}\mathcal{C}_1\\ \phi(st_1, ..., st_k) \Rightarrow_{s_1}^{s_0} (p, st)\end{array}}{\delta \ \vdash \ \phi(x_1, ..., x_k) \Rightarrow_{s_1}^{s_0} (p, st)} \ ((\delta(x_i) = st_i)_{i=1}^k)$$

From our assumption (iv), for all $i \in \{1, ..., k\}$:

(a) $\vdash \ \rho(x_i) : \Gamma(x_i)$, that is, $\vdash \ v_i : \tau_i$

(b) $\texttt{sids}(\delta(x_i)) \lessdot s_0$, that is, $\texttt{sids}(st_i) \lessdot s_0$

(c) $\rho(x_i) \rhd_{\Gamma(x_i)} \sigma(st_i)$, that is, $v_i \rhd_{\tau_i} \sigma(st_i)$

So using Lemma 3.3 on $\mathcal{T}_1, \mathcal{E}_1, \mathcal{C}_1, (a), (b)$ and (c) gives us exactly what we shall show.

□