

# SNESL formalization Level-0

Dandan Xue

September 19, 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 has not updated yet

## 1 Source Language

### 1.1 Source language syntax

SNESL Expressions:

$$e ::= x \mid \mathbf{let} \ x = e_1 \ \mathbf{in} \ e_2 \mid \phi(x_1, \dots, x_k) \mid \{e : x \ \mathbf{in} \ y \ \mathbf{using} \ \cdot\}$$
$$\phi = \mathbf{const}_n \mid \mathbf{iota} \mid \mathbf{plus}$$

Values:

$$n \in \mathbf{Z}$$
$$v ::= n \mid \{v_1, \dots, v_k\}$$

### 1.2 Type system

$$\tau ::= \mathbf{int} \mid \{\tau_1\}$$

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

- Expression typing rules:

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

$$\frac{}{\Gamma \vdash x : \tau} (\Gamma(x) = \tau) \qquad \frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma[x \mapsto \tau_1] \vdash e_2 : \tau}{\Gamma \vdash \mathbf{let} \ x = e_1 \ \mathbf{in} \ e_2 : \tau}$$
$$\frac{\phi : (\tau_1, \dots, \tau_k) \rightarrow \tau}{\Gamma \vdash \phi(x_1, \dots, x_k) : \tau} ((\Gamma(x_i) = \tau_i)_{i=1}^k) \qquad \frac{[x \mapsto \tau_1] \vdash e : \tau}{\Gamma \vdash \{e : x \ \mathbf{in} \ y \ \mathbf{using} \ \cdot\} : \{\tau\}} (\Gamma(y) = \{\tau_1\})$$

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

$$\frac{}{\mathbf{const}_n : () \rightarrow \mathbf{int}} \qquad \frac{}{\mathbf{iota} : (\mathbf{int}) \rightarrow \{\mathbf{int}\}} \qquad \frac{}{\mathbf{plus} : (\mathbf{int}, \mathbf{int}) \rightarrow \mathbf{int}}$$

- Value typing rules:

Judgment  $\boxed{v : \tau}$

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

### 1.3 Source language semantics

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

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

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

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

$$\frac{([x \mapsto v_i] \vdash e \downarrow v'_i)_{i=1}^k}{\rho \vdash \{e : x \mathbf{ in } y \mathbf{ using } \cdot\} \downarrow \{v'_1, \dots, v'_k\}} (\rho(y) = \{v_1, \dots, v_k\})$$

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

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

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

## 2 Target language

### 2.1 SVCODE syntax

- (1) Stream id:

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

A list of **SId**:

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

- (2) SVCODE operations:

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

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

- (3) SVCODE program:

$$\begin{aligned} p ::= & \epsilon \\ & \mid s := \psi(s_1, \dots, s_i) \\ & \mid \mathbf{S}_2 := \mathbf{WithCtrl}(s, \mathbf{S}_1, p_1) \\ & \mid p_1; p_2 \end{aligned}$$

(4) SVCODE streams:

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

(5) Notations and operations about streams:

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

## 2.2 SVCODE semantics

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

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

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

$$\text{P-EMPTY} : \frac{}{\langle \epsilon, \sigma \rangle \downarrow^{\vec{c}} \sigma}$$

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

$$\text{P-WC-EMP} : \frac{}{\langle \mathbf{S}_{out} := \text{WithCtrl}(s_c, \mathbf{S}_{in}, p), \sigma \rangle \downarrow^{\vec{c}} \sigma[s_1 \mapsto \langle \rangle, \dots, s_i \mapsto \langle \rangle]} \quad \left( \begin{array}{l} \sigma(s_c) = \langle \rangle \\ \forall s \in \mathbf{S}_{in}. \sigma(s) = \langle \rangle \\ ??? \forall j \in \{1, \dots, i\}. s_j \in \mathbf{S}_{out} \wedge s_j > s_c \end{array} \right)$$

$$\text{P-WC-NONEMP} : \frac{\langle p, \sigma \rangle \downarrow^{\vec{c}_1} \sigma''}{\langle \mathbf{S}_{out} := \text{WithCtrl}(s_c, \mathbf{S}_{in}, p), \sigma \rangle \downarrow^{\vec{c}} \sigma[s_1 \mapsto \sigma''(s_1), \dots, s_i \mapsto \sigma''(s_i)]} \quad \left( \begin{array}{l} \sigma(s_c) = \vec{c}_1 = \langle () | \vec{c}_2 \rangle \\ ??? \forall j \in \{1, \dots, i\}. s_j \in \mathbf{S}_{out} \wedge s_j > s_c \end{array} \right)$$

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

- *Transducer* semantics:

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

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

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

- Transducer *block* semantics:

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

$$\text{P-CONST} : \frac{}{\text{Const}_a() \Downarrow \langle a \rangle}$$

$$\text{P-TOFLAGS} : \frac{}{\text{ToFlags}(\langle n \rangle) \Downarrow \langle \mathbf{F}_1, \dots, \mathbf{F}_n, \mathbf{T} \rangle}$$

<sup>1</sup>For convenience, in this thesis we add subscripts to a sequence of constants, such as  $\langle \rangle, \mathbf{F}, 1$ , to denote the total number of these constants.

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

$$\text{P-USUMF} : \frac{\text{Usum}(\vec{b}) \Downarrow \vec{a}}{\text{Usum}(\langle \mathbf{F} | \vec{b} \rangle) \Downarrow \langle () | \vec{a} \rangle} \quad \text{P-USUMT} : \frac{}{\text{Usum}(\langle \mathbf{T} \rangle) \Downarrow \langle \rangle}$$

$$\text{P-SCANF} : \frac{\text{ScanPlus}_{n_0+n}(\vec{b}, \vec{a}) \Downarrow \vec{a'}}{\text{ScanPlus}_{n_0}(\langle \mathbf{F} | \vec{b} \rangle, \langle n | \vec{a} \rangle) \Downarrow \langle n_0 | \vec{a'} \rangle} \quad \text{P-SCANT} : \frac{}{\text{ScanPlus}_{n_0}(\langle \mathbf{T} \rangle, \langle \rangle) \Downarrow \langle \rangle}$$

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

$$\frac{\psi(\langle \mathbf{F} \rangle, \dots, \vec{a}_{k1}) \Downarrow \vec{a}_1 \quad \psi(\vec{a}_{12}, \dots, \vec{a}_{k2}) \Downarrow \vec{a}_2}{\psi(\langle \mathbf{F} \rangle ++ \vec{a}_{12}, \dots, \vec{a}_{k1} ++ \vec{a}_{k2}) \Downarrow \vec{a}} \quad (\vec{a} = \vec{a}_1 ++ \vec{a}_2)$$
  

$$\frac{\psi(\langle \mathbf{T} \rangle, \dots, \vec{a}_k) \Downarrow \vec{a}}{\psi(\langle \mathbf{T} \rangle, \dots, \vec{a}_k) \Downarrow \vec{a}}$$
  

– Transducer *unary* semantics:

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

$$\frac{}{\text{Usum}(\langle \mathbf{F} \rangle) \Downarrow \langle () \rangle} \quad \frac{}{\text{Usum}(\langle \mathbf{T} \rangle) \Downarrow \langle \rangle}$$
  

– Transducer block with *accumulator*:

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

$$\frac{\psi_{n_0}(\langle \mathbf{F} \rangle, \dots, \vec{a}_{k1}) \Downarrow^{n'_0} \langle n_1 \rangle \quad \psi_{n'_0}(\vec{a}_{12}, \dots, \vec{a}_{k2}) \Downarrow \vec{a}_2}{\psi_{n_0}(\langle \mathbf{F} \rangle ++ \vec{a}_{12}, \dots, \vec{a}_{k1} ++ \vec{a}_{k2}) \Downarrow \langle n_1 \rangle ++ \vec{a}_2}$$
  

$$\frac{\psi_{n_0}(\langle \mathbf{T} \rangle, \dots, \vec{a}_k) \Downarrow \vec{a}}{\psi_{n_0}(\langle \mathbf{T} \rangle, \dots, \vec{a}_k) \Downarrow \vec{a}}$$

– Transducer unary with *accumulator*:

Judgment  $\boxed{\psi_n(\langle \mathbf{F} \rangle, \dots, \vec{a}_k) \Downarrow^{n'} \vec{a}}$

$$\frac{}{\text{ScanPlus}_{n_0}(\langle \mathbf{F} \rangle, \langle n \rangle) \Downarrow^{n_0+n} \langle n_0 \rangle}$$

Judgment  $\boxed{\psi_n(\langle \mathbf{T} \rangle, \dots, \vec{a}_k) \Downarrow \vec{a}}$

$$\frac{}{\text{ScanPlus}_{n_0}(\langle \mathbf{T} \rangle, \langle \rangle) \Downarrow \langle \rangle}$$

## 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)  $\text{dom}(\sigma_1) = \text{dom}(\sigma_2)$
- (2)  $\forall s \in \mathbf{S}. \sigma_1(s) = \sigma_2(s)$

According to this definition, it is only meaningful to have  $\mathbf{S} \subseteq \text{dom}(\sigma_1)$  ( $= \text{dom}(\sigma_2)$ ). When  $\mathbf{S} = \text{dom}(\sigma_1) = \text{dom}(\sigma_2)$ ,  $\sigma_1$  and  $\sigma_2$  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 special 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)  $\sigma(s) = \begin{cases} \sigma_i(s), & s \in \mathbf{S}, i \in \{1, 2\} \\ \sigma_1(s) ++ \sigma_2(s), & \text{otherwise} \end{cases}$

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

**Lemma 2.3 (Stores concatenation lemma).** If  $\sigma_1 \overset{\mathbf{S}}{\sim} \sigma_2$ ,  $\langle p, \sigma_1 \rangle \downarrow^{\vec{c}_1} \sigma'_1$  (by some derivation  $\mathcal{P}_1$ ),  $\langle p, \sigma_2 \rangle \downarrow^{\vec{c}_2} \sigma'_2$  (by  $\mathcal{P}_2$ ), and  $\text{fv}(p, \vec{c}_1) \cap \mathbf{S} = \emptyset$ ,  $\text{fv}(p, \vec{c}_2) \cap \mathbf{S} = \emptyset$  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 computation  $p$  on even smaller degree of parallel computations, in which all the supplier streams, i.e.,  $\text{fv}(p, \cdot)$ , 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 **S**Ids in all  $\sigma$ s (i.e.,  $\mathbf{S}$ ) have no change throughout the process.

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

**Lemma 2.4.** If  $\sigma_1 \overset{\mathbf{S}_1}{\sim} \sigma'_1$ ,  $\sigma_2 \overset{\mathbf{S}_2}{\sim} \sigma'_2$ ,  $\sigma_1 \overset{\leq s}{=} \sigma_2$ , and  $\sigma'_1 \overset{\leq s}{=} \sigma'_2$  then  $\sigma_1 \overset{\mathbf{S}_1}{\bowtie} \sigma'_1 \overset{\leq s}{=} \sigma_2 \overset{\mathbf{S}_2}{\bowtie} \sigma'_2$ .

## 2.4 SVCODE determinism theroem

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

Judgment  $\boxed{\vec{a} \sqsubseteq \vec{a}'}$

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

**Lemma 2.5.** If

- (i)  $(\vec{a}'_i \sqsubseteq \vec{a}_i)_{i=1}^k$  and  $\psi(\vec{a}'_1, \dots, \vec{a}'_k) \downarrow \vec{a}'$ ,
- (ii)  $(\vec{a}''_i \sqsubseteq \vec{a}_i)_{i=1}^k$  and  $\psi(\vec{a}''_1, \dots, \vec{a}''_k) \downarrow \vec{a}''$

then

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

**Lemma 2.6.** If  $\psi(\vec{a}_1, \dots, \vec{a}_k) \downarrow^{\vec{c}} \vec{a}$ , and  $\psi(\vec{a}_1, \dots, \vec{a}_k) \downarrow^{\vec{c}} \vec{a}'$ , 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:

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

(3) Function  $\mathbf{fv}(p, s_c)$  takes an SVCODE program and the control stream id as parameters, returns a list of the free variables (i.e., stream ids) of  $p$ :

$$\begin{aligned} \mathbf{fv}(\epsilon, \cdot) &= [] \\ \mathbf{fv}(s := \psi(s_1, \dots, s_i), s_c) &= [s_c, s_1, \dots, s_i] \\ \mathbf{fv}(\mathbf{S}_{out} := \mathbf{WithCtrl}(s'_c, \mathbf{S}_{in}, p_1), s_c) &= [s \mid s \in \mathbf{S}_{in}, s < s_c] \\ \mathbf{fv}(p_1; p_2, s_c) &= \mathbf{fv}(p_1, s_c) ++ \mathbf{fv}(p_2, s_c) \end{aligned}$$

(4) Translation environment:

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

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

$$\begin{aligned} &\frac{}{\delta \vdash x \Rightarrow_{s_0}^{s_0} (\epsilon, st)} (\delta(x) = st) && \frac{\delta \vdash e_1 \Rightarrow_{s_0'}^{s_0} (p_1, st_1) \quad \delta[x \mapsto st_1] \vdash e_2 \Rightarrow_{s_1}^{s_0'} (p_2, st)}{\delta \vdash \mathbf{let } x = e_1 \mathbf{ in } e_2 \Rightarrow_{s_1}^{s_0} (p_1; p_2, st)} \\ &\frac{\phi(st_1, \dots, st_k) \Rightarrow_{s_1}^{s_0} (p, st)}{\delta \vdash \phi(x_1, \dots, x_k) \Rightarrow_{s_1}^{s_0} (p, st)} ((\delta(x_i) = st_i)_{i=1}^k) \\ &\frac{[x \mapsto st_1] \vdash e \Rightarrow_{s_1}^{s_0+1} (p, st)}{\delta \vdash \{e : x \mathbf{ in } y \mathbf{ using } \cdot\} \Rightarrow_{s_1}^{s_0} (s_0 := \mathbf{Usum}(s_2); \overline{st} := \mathbf{WithCtrl}(s_0, \mathbf{S}_{in}, p), (st, s_2))} \left( \begin{array}{l} \delta(y) = (st_1, s_2) \\ \mathbf{S}_{in} = \mathbf{fv}(p, s_0) \end{array} \right) \end{aligned}$$

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

$$\begin{aligned} &\frac{}{\mathbf{const}_n() \Rightarrow_{s_0+1}^{s_0} (s_0 := \mathbf{Const}_n(), s_0)} \\ &\frac{}{\mathbf{iota}(s) \Rightarrow_{s_4}^{s_0} (p, (s_3, s_0))} \left( \begin{array}{l} s_{i+1} = s_i + 1, \forall i \in \{0, \dots, 3\} \\ p = s_0 := \mathbf{ToFlags}(s); \\ s_1 := \mathbf{Usum}(s_0); \\ \overline{s_2} := \mathbf{WithCtrl}(s_1, [s_1], s_2 := \mathbf{Const}_1()); \\ s_3 := \mathbf{ScanPlus}_0(s_0, s_2) \end{array} \right) \\ &\frac{}{\mathbf{plus}(s_1, s_2) \Rightarrow_{s_0+1}^{s_0} (s_0 := \mathbf{MapTwo}_+(s_1, s_2), s_0)} \end{aligned}$$

#### 3.2 Value representation

1. SVCODE values:

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

2. SVCODE values concatenation:

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

3. SVCODE value construction from a stream tree:

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

4. Value representation rules

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

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

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

### 3.3 Correctness proof

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

**Lemma 3.3.** *If*

- (i)  $\phi : (\tau_1, \dots, \tau_k) \rightarrow \tau$  (by some derivation  $\mathcal{T}$ )
- (ii)  $\phi(v_1, \dots, v_k) \vdash v$  (by  $\mathcal{E}$ )
- (iii)  $\phi(st_1, \dots, st_k) \Rightarrow_{s_1}^{s_0} (p, st)$  (by  $\mathcal{C}$ )
- (iv)  $(v_i \triangleright_{\tau_i} \sigma(st_i))_{i=1}^k$
- (v)  $\bigcup_{i=1}^k \mathbf{sids}(st_i) \leq s_0$

then

- (vi)  $\langle p, \sigma \rangle \downarrow^{\langle () \rangle} \sigma'$  (by  $\mathcal{P}$ )
- (vii)  $v \triangleright_\tau \sigma'(st)$  (by  $\mathcal{R}$ )
- (viii)  $\sigma' \stackrel{\leq s_0}{=} \sigma$
- (ix)  $s_0 \leq s_1$
- (x)  $\mathbf{sids}(st) \leq s_1$

*Proof.* By induction on the syntax of  $\phi$ .

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

$$\mathcal{T} = \overline{\mathbf{const}_n : () \rightarrow \mathbf{int}}$$

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

$$\mathcal{C} = \overline{\text{const}_n() \Rightarrow_{s_0+1}^{s_0} (s_0 := \text{Const}_n(), s_0)}$$

So  $k = 0, \tau = \mathbf{int}, v = n, p = s_0 := \text{Const}_n(), s_1 = s_0 + 1$ , and  $st = s_0$

By P-XDUCER, P-X-LOOP, P-X-TERMI and P-CONST, we can construct  $\mathcal{P}$  as follows:

$$\mathcal{P} = \frac{\frac{\overline{\text{Const}_n() \Downarrow \langle n \rangle} \quad \overline{\text{Const}_n() \Downarrow^{\langle \rangle} \langle \rangle}}{\text{Const}_n() \Downarrow^{\langle \rangle} \langle n \rangle}}{\langle s_0 := \text{Const}_n(), \sigma \rangle \Downarrow^{\langle \rangle} \sigma[s_0 \mapsto \langle n \rangle]}$$

So  $\sigma' = \sigma[s_0 \mapsto \langle n \rangle]$ .

Then we take  $\mathcal{R} = \overline{n \triangleright_{\mathbf{int}} \sigma'(s_0)}$ .

Also clearly,  $\sigma' \leq_{s_0} \sigma$ ,  $s_0 \leq s_0 + 1$ ,  $\mathbf{sids}(s_0) < s_0 + 1$ , and we are done.

- Case  $\phi = \mathbf{plus}$

We must have

$$\begin{aligned} \mathcal{T} &= \overline{\mathbf{plus} : (\mathbf{int}, \mathbf{int}) \rightarrow \mathbf{int}} \\ \mathcal{E} &= \overline{\vdash \mathbf{plus}(n_1, n_2) \Downarrow n_3} \end{aligned}$$

where  $n_3 = n_2 + n_1$ , and

$$\mathcal{C} = \overline{\mathbf{plus}(s_1, s_2) \Rightarrow_{s_0+1}^{s_0} (s_0 := \text{MapTwo}_+(s_1, s_2), s_0)}$$

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  $p = s_0 := \text{MapTwo}_+(s_1, s_2)$ .

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:

$$\frac{\frac{\overline{\text{MapTwo}_+(\langle n_1 \rangle, \langle n_2 \rangle) \Downarrow \langle n_3 \rangle} \quad \overline{\text{MapTwo}_+(\langle \rangle, \langle \rangle) \Downarrow^{\langle \rangle} \langle \rangle}}{\text{MapTwo}_+(\langle n_1 \rangle, \langle n_2 \rangle) \Downarrow^{\langle \rangle} \langle n_3 \rangle}}{\langle s_0 := \text{MapTwo}_+(s_1, s_2), \sigma \rangle \Downarrow^{\langle \rangle} \sigma[s_0 \mapsto \langle n_3 \rangle]}$$

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  $\sigma' \leq_{s_0} \sigma$ ,  $s_0 \leq s_0 + 1$  and  $\mathbf{sids}(s_0) < s_0 + 1$  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)  $\delta \vdash e \Rightarrow_{s_1}^{s_0} (p, st)$  (by some  $\mathcal{C}$ )



(iv)  $\forall x \in \text{dom}(\Gamma). \vdash \rho(x) : \Gamma(x) \wedge \mathbf{sids}(\delta(x)) \leq s_0 \wedge \rho(x) \triangleright_{\Gamma(x)} \sigma(\delta(x))$   
**then**

(v)  $\langle p, \sigma \rangle \downarrow^{(0)} \sigma'$  (by some derivation  $\mathcal{P}$ )

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

(vii)  $\sigma' \stackrel{\leq s_0}{=} \sigma$

(viii)  $s_0 \leq s_1$

(ix)  $\mathbf{sids}(st) \leq s_1$

*Proof.* By induction on the syntax of  $e$ .

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

We must have:

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

$$(ii) \quad \mathcal{E} = \frac{\mathcal{E}_i \quad ([x \mapsto v_i] \vdash e_1 \downarrow v'_i)_{i=1}^k}{\rho \vdash \{e_1 : x \text{ in } y \text{ using } \cdot\} \downarrow \{v'_1, \dots, v'_k\}} \quad (\rho(y) = \{v_1, \dots, v_k\})$$

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

So  $p = (s_0 := \mathbf{Usum}(s_2); \overline{st_2} := \mathbf{WithCtrl}(s_0, \mathbf{S}_{in}, p_1))$ ,  $\tau = \{\tau_2\}$ ,  $v = \{v'_1, \dots, v'_k\}$ ,  $st = (st_2, s_2)$ .

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

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

$\mathbf{sids}(\delta(y)) \leq s_0$  gives us

$$\mathbf{sids}(\delta(y)) = \mathbf{sids}((st_1, s_2)) = \mathbf{sids}(st_1) \cup \{s_2\} \leq s_0 \quad (2)$$

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

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

therefore

$$\sigma(st_1) = w \quad (4)$$

and

$$\sigma(s_2) = \langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle. \quad (5)$$

First we shall show:

(v)  $\langle s_0 := \mathbf{Usum}(s_2); \overline{st_2} := \mathbf{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma \rangle \downarrow^{(0)} \sigma'$  by some  $\mathcal{P}$

(vi)  $\{v'_1, \dots, v'_k\} \triangleright_{\{\tau_2\}} \sigma'((st_2, s_2))$  by some  $\mathcal{R}$

(vii)  $\sigma' \xrightarrow{\leq s_0} \sigma$

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

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

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

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

So  $\sigma_0 = \sigma[s_0 \mapsto \vec{a}]$ .

We split  $\langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle$  into two parts:  $\vec{b}_1 = \langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle$  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:

$$\mathcal{P}'_0 = \frac{\mathcal{P}''_0 \quad \text{Usum}(\vec{b}_1) \downarrow \vec{a} \quad \text{Usum}(\vec{b}_2) \downarrow^{(\langle \rangle)} \langle \rangle}{\text{Usum}(\langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle) \downarrow^{(\langle \rangle)} \vec{a}}$$

Then using P-USUMF  $k$  times and P-USUMT once, we obtain

$$\mathcal{P}''_0 = \frac{\overline{\text{Usum}(\langle \mathbf{T} \rangle) \downarrow \langle \rangle} \quad \vdots \quad \text{Usum}(\langle \mathbf{F}_2, \dots, \mathbf{F}_k, \mathbf{T} \rangle) \downarrow \langle ()_2, \dots, ()_k \rangle}{\text{Usum}(\langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle) \downarrow \langle ()_1, \dots, ()_k \rangle}$$

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

Since we have

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

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

$$\forall z \in \text{dom}(\Gamma_1). \vdash \rho_i(z) : \Gamma_1(z) \wedge \mathbf{sids}(\delta_1(z)) \leq s_0.$$

Let  $i$  range from 1 to  $k$ : we take  $\sigma_i \stackrel{st_1}{\sim} \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle]$  such that  $\sigma_i(st_1) = w_i$ .

From  $\mathcal{R}_i$  in (3) we know that

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

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

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

$$(v'_i \triangleright_{\tau_2} \sigma'_i(st_2))_{i=1}^k \tag{7}$$

$$(\sigma'_i \xrightarrow{\leq s_0+1} \sigma_i)_{i=1}^k \tag{8}$$

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

$$\mathbf{sids}(st_2) \leq s_1 \tag{10}$$

Assume  $\mathbf{sids}(st_2) = \{s'_1, \dots, s'_j\}$ .

There are two possibilities:

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

$$\mathcal{P}_1 = \frac{}{\langle \overline{st_2} := \text{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma \rangle \downarrow^{(\langle \rangle)} \sigma[s_0 \mapsto \langle \rangle, s'_1 \mapsto \langle \rangle, \dots, s'_j \mapsto \langle \rangle]}$$

So in this subcase,

$$\sigma' = \sigma[s_0 \mapsto \langle \rangle, s'_1 \mapsto \langle \rangle, \dots, s'_j \mapsto \langle \rangle].$$

Since  $k = 0$ , then  $v = \{\}$ ,  $\sigma(s_2) = \langle \mathbf{T} \rangle$  (from (5)), we have

$\sigma'(s_2) = \sigma(s_2) = \langle \mathbf{T} \rangle$  (?? not correct if  $s_2 \in \mathbf{sids}(st_2)/\mathbf{sids}(st_1)$ ),  
and  $\sigma'(st_2) = (\dots((\langle \rangle, \langle \rangle)_1, \langle \rangle)_2, \dots)_{j-1}$ .

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

$$\mathcal{R} = \frac{\{\} \triangleright_{\{\tau_2\}} ((\dots((\langle \rangle, \langle \rangle)_1, \dots)_{j-1}, \langle \mathbf{T} \rangle))$$

as required.

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

$$\sigma' \stackrel{\leq s_0}{=} \sigma.$$

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

$$\frac{\mathcal{P}'_1 \quad \langle p_1, \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle] \rangle \downarrow^{(\langle ()_1, \dots, ()_k \rangle)} \sigma''}{\langle \overline{st_2} := \text{WithCtrl}(s_0, \mathbf{S}_{in}, p_1), \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle] \rangle \downarrow^{(\langle \rangle)} \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle, s'_1 \mapsto \sigma''(s'_1), \dots, s'_j \mapsto \sigma''(s'_j)]}$$

So in this subcase

$$\sigma' = \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle, s'_1 \mapsto \sigma''(s'_1), \dots, s'_j \mapsto \sigma''(s'_j)].$$

Using Lemma 2.3 (k-1) times on (6) gives us

$$\langle p_1, (\boxtimes_{i=1}^{st_1} \sigma_i)^k \rangle \downarrow^{(\langle ()_1, \dots, ()_k \rangle)} (\boxtimes_{i=1}^{st'_2} \sigma'_i)^k \quad (11)$$

where  $st'_2 = \mathbf{sids}(st_1) \cup \mathbf{sids}(st_2)$  (???) .

By Definition 2.2 we have

$$(\boxtimes_{i=1}^{st_1} \sigma_i)^k = \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle]. \quad (12)$$

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

$$\sigma'' = (\boxtimes_{i=1}^{st'_2} \sigma'_i)^k \quad (13)$$

Therefore,  $\sigma''(st_2) = \sigma'_1(st_2) ++ \dots ++ \sigma'_k(st_2)$  by Definition 2.2.

Let  $\sigma'_i(st_2) = w'_i$  and  $\sigma''(st_2) = w'$ , then  $w' = w'_1 ++ \dots ++ w'_k$ .

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

$$\mathcal{R} = \frac{(v'_i \triangleright_{\tau_2} w'_i)_{i=1}^k}{\{v'_1, \dots, v'_k\} \triangleright_{\{\tau_2\}} (w', \langle \mathbf{F}_1, \dots, \mathbf{F}_k, \mathbf{T} \rangle)}$$

as required.

By Lemma 2.1 on (12) we get  $\sigma_i \stackrel{st_1}{\sim} \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle]$ , and similarly  $\sigma'_i \stackrel{st'_2}{\sim} \sigma''$  from (13).

Since (8) implies

$$(\sigma'_i \stackrel{\leq s_0}{=} \sigma_i)_{i=1}^k$$

using Lemma 2.4 (k-1) times, we obtain

$$\sigma'' \stackrel{\leq s_0}{=} \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle].$$

Therefore,  $\sigma' \stackrel{\leq s_0}{=} \sigma[s_0 \mapsto \langle ()_1, \dots, ()_k \rangle] \stackrel{\leq s_0}{=} \sigma$ .

(viii) TS:  $s_0 \leq s_1$

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

(ix) TS:  $\mathbf{sids}((st_2, s_2)) < s_1$

From (2) we know  $s_2 < s_0$ , thus  $s_2 < s_0 \leq s_1$ . And we already have (10). Therefore,

$$\mathbf{sids}((st_2, s_2)) = \mathbf{sids}(st_2) \cup \{s_2\} < s_1.$$

• Case  $e = x$ .

We must have

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

So  $p = \epsilon$ .

Immediately we have  $\mathcal{P} = \frac{}{\langle \epsilon, \sigma \rangle \downarrow^{()}\sigma}$

So  $\sigma' = \sigma$ , which implies  $\sigma' \stackrel{\leq s_0}{=} \sigma$ .

From the assumption we already have  $v \triangleright_{\tau} \sigma(st)$ , and  $\mathbf{sids}(st) < s_0$ .

Finally it's clear that  $s_0 \leq s_0$ , and we are done.

• Case  $e = \mathbf{let } x = e_1 \mathbf{ in } e_2$ .

We must have:

$$\begin{aligned} \mathcal{T} &= \frac{\mathcal{T}_1 \quad \mathcal{T}_2}{\Gamma \vdash e_1 : \tau_1 \quad \Gamma[x \mapsto \tau_1] \vdash e_2 : \tau} \Gamma \vdash \mathbf{let } x = e_1 \mathbf{ in } e_2 : \tau \\ \mathcal{E} &= \frac{\mathcal{E}_1 \quad \mathcal{E}_2}{\rho \vdash e_1 \downarrow v_1 \quad \rho[x \mapsto v_1] \vdash e_2 \downarrow v} \rho \vdash \mathbf{let } x = e_1 \mathbf{ in } e_2 \downarrow v \\ \mathcal{C} &= \frac{\mathcal{C}_1 \quad \mathcal{C}_2}{\delta \vdash e_1 \Rightarrow_{s_0}^{s_0} (p_1, st_1) \quad \delta[x \mapsto st_1] \vdash e_2 \Rightarrow_{s_1}^{s'_0} (p_2, st)} \delta \vdash \mathbf{let } x = e_1 \mathbf{ in } e_2 \Rightarrow_{s_1}^{s_0} (p_1; p_2, st) \end{aligned}$$

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  $\langle p_1, \sigma \rangle \downarrow^{()}\sigma_1$

(b)  $\mathcal{R}_1$  of  $v_1 \triangleright_{\tau_1} \sigma_1(st_1)$

(c)  $\sigma_1 \stackrel{\leq s_0}{=} \sigma$

- (d)  $s_0 \leq s'_0$
- (e)  $\mathbf{sids}(st_1) \leq s'_0$

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  $\mathbf{sids}(\delta[x \mapsto st_1](x)) \leq s'_0$ .

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

- (f)  $\mathcal{P}_2$  of  $\langle p_2, \sigma_1 \rangle \downarrow^{(\cdot)}$   $\sigma_2$
- (g)  $\mathcal{R}_2$  of  $\sigma_2 \triangleright_{\tau} \sigma_2(st)$
- (h)  $\sigma_2 \stackrel{\leq s'_0}{=} \sigma_1$
- (i)  $s'_0 \leq s_1$
- (j)  $\mathbf{sids}(st) \leq s_1$

So we can construct:

$$\mathcal{P} = \frac{\frac{\mathcal{P}_1}{\langle p_1, \sigma \rangle \downarrow^{(\cdot)} \sigma_1} \quad \frac{\mathcal{P}_2}{\langle p_2, \sigma_1 \rangle \downarrow^{(\cdot)} \sigma_2}}{\langle p_1; p_2, \sigma \rangle \downarrow^{(\cdot)} \sigma_2}$$

From (c), (d) and (h), it is clear that  $\sigma_2 \stackrel{\leq s_0}{=} \sigma_1 \stackrel{\leq s_0}{=} \sigma$ . From (d) and (i),  $s_0 \leq s_1$ .

Take  $\sigma' = \sigma_2$  (thus  $\mathcal{R} = \mathcal{R}_2$ ) and we are done.

- Case  $e = \phi(x_1, \dots, x_k)$

We must have

$$\begin{aligned} \mathcal{T} &= \frac{\mathcal{T}_1}{\Gamma \vdash \phi(x_1, \dots, x_k) : \tau} \quad ((\Gamma(x_i) = \tau_i)_{i=1}^k) \\ \mathcal{E} &= \frac{\mathcal{E}_1}{\rho \vdash \phi(x_1, \dots, x_k) \downarrow v} \quad ((\rho(x_i) = v_i)_{i=1}^k) \\ \mathcal{C} &= \frac{\mathcal{C}_1}{\delta \vdash \phi(x_1, \dots, x_k) \Rightarrow_{s_1}^{s_0} (p, st)} \quad ((\delta(x_i) = st_i)_{i=1}^k) \end{aligned}$$

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

- (a)  $\vdash \rho(x_i) : \Gamma(x_i)$ , that is,  $\vdash v_i : \tau_i$
- (b)  $\mathbf{sids}(\delta(x_i)) \leq s_0$ , that is,  $\mathbf{sids}(st_i) \leq s_0$
- (c)  $\rho(x_i) \triangleright_{\Gamma(x_i)} \sigma(st_i)$ , that is,  $v_i \triangleright_{\tau_i} \sigma(st_i)$

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.

□