④ Formal properties of the operational semantics:

(i) $\gamma \to \gamma_1$ and $\gamma \to \gamma_2 \Rightarrow \gamma_1 = \gamma_2$.

The semantics is deterministic.

(ii) $\forall \gamma \in \Gamma_N.\ \exists \gamma'$ s.t. $\gamma \to \gamma'$.

In this semantics, executions never get stuck.

(iii) From (i) and (ii) it follows that for every $\gamma \in \Gamma$, there exists a unique maximal sequence.

$$\gamma_0, \gamma_1, \gamma_2, \cdots, \gamma_n$$

(γ_n may be infinite.)

Such that

$$\gamma = \gamma_0 \wedge \gamma_0 \to \gamma_1 \to \cdots \to \gamma_n.$$

$\wedge$ γ_n is a terminal configuration or n is infinite.

This maximal finite or infinite sequence represents the full computation starting from γ.

(iv) We write $\gamma\!\downarrow$ if the maximal execution sequence from γ is infinite. ~~We write~~ $\not\equiv$ Then, for all commands c and states σ,

$$[\![c]\!]\sigma = \bot \quad \text{iff} \quad \langle c, \sigma\rangle\!\downarrow$$

$$[\![c]\!]\sigma = \sigma' \quad \text{iff} \quad \langle c, \sigma\rangle \xrightarrow{*} \sigma'.$$

($\xrightarrow{*}$: reflexive and transitive closure of $\to$, i.e. $\xrightarrow{*} \stackrel{\text{def}}{=} \bigcup_{n=0}^{\infty} (\to)^n$)

exercises. Prove (i), (ii) and (iv).

exercise. Explain why (iii) is true. (the reasoning in ^)

4. Extension with newvar.

① Extend the ~~while~~ language with variable declaration:

$\langle comm\rangle ::= \ldots \mid$ newvar $\langle var\rangle := \langle intexp\rangle$ in $\langle comm\rangle$.

② ~~What is~~ How should we modify the $\to$ relation? Add a rule for newvar.

(i) Option 1.

$$\frac{}{\langle \text{newvar } v := e \text{ in } c, \sigma\rangle \to \langle c ; v := n, [\sigma \mid v : [\![e]\!]\sigma]\rangle}$$

where $n = \sigma(v)$.

(ii) Option 2.

$$\frac{\langle c, [\sigma \mid v : [\![e]\!]\sigma]\rangle \to \langle c', \sigma'\rangle}{\langle \text{newvar } v := e \text{ in } c, \sigma\rangle \to \langle \text{newvar } v := \sigma'(v) \text{ in } c', [\sigma' \mid v : \sigma(v)]\rangle}$$

$$\frac{\langle c, [\sigma \mid v : [\![e]\!]\sigma]\rangle \to \sigma'}{\langle \text{newvar } v := e \text{ in } c, \sigma\rangle \to [\sigma' \mid v : \sigma(v)]}$$

(iii) Both options are acceptable. But (ii) is better. Only (ii) works when we extend the language with primitives for concurrent executions.

③ Note that we did not change Γ_N, Γ_T. Thus, adding newvar doesn't change the operational semantics much. In a sense, this small change means that newvar doesn't change the language much, either.

5. Adding fail.

$\langle comm\rangle ::= \ldots \mid$ fail.

① When we add fail, we have to change the set Γ_T of terminal configuration, because we now have two types of terminations, normal one and abnormal one.

$$\Gamma_T \stackrel{\text{def}}{=} \widehat{\Sigma} = \Sigma \cup \{\text{abort}\} \times \Sigma. \quad (\text{or } = \Sigma + \Sigma)$$