COMP3161/COMP9164

Semantics Exercises

Liam O'Connor

September 29, 2019

1. [⋆⋆] **Jankenbon:**

   Below is a language specified to compute the results of rock paper scissors tournaments:

$$L \quad ::= \quad ✊ \mid ✋ \mid ✌ \mid (\texttt{Play}\ L\ L)$$

   We assume the existence of an operation **match** which computes the result of an individual game.

$$\begin{aligned}
\textbf{match}(✊, ✋) &= ✋\\
\textbf{match}(✌, ✋) &= ✌\\
\textbf{match}(✋, ✋) &= ✋\\
\textbf{match}(✊, ✌) &= ✊\\
\textbf{match}(✌, ✌) &= ✌\\
\textbf{match}(✋, ✌) &= ✌\\
\textbf{match}(✊, ✊) &= ✊\\
\textbf{match}(✌, ✊) &= ✊\\
\textbf{match}(✋, ✊) &= ✋
\end{aligned}$$

   Here are some small step semantics to compute the tournament result:

$$\frac{e_1 \mapsto e_1'}{(\texttt{Play}\ e_1\ e_2) \mapsto (\texttt{Play}\ e_1'\ e_2)}$$

$$\frac{v_1 \in \{✌, ✋, ✊\} \qquad e_2 \mapsto e_2'}{(\texttt{Play}\ v_1\ e_2) \mapsto (\texttt{Play}\ v_1\ e_2')}$$

$$\frac{v_1 \in \{✌, ✋, ✊\} \qquad v_2 \in \{✌, ✋, ✊\}}{(\texttt{Play}\ v_1\ v_2) \mapsto \textbf{match}(v_1, v_2)}$$

   Determine a suitable big-step equivalent semantics for this language.

   Solution: The set of evaluable expressions is just L. The set of values is $\{✌, ✋, ✊\}$. We have three axioms:

$$\frac{}{✌ \Downarrow ✌} \qquad \frac{}{✋ \Downarrow ✋} \qquad \frac{}{✊ \Downarrow ✊}$$

   And one other rule:

$$\frac{e_1 \Downarrow v_1 \qquad e_2 \Downarrow v_2}{(\texttt{Play}\ e_1\ e_2) \Downarrow \textbf{match}(v_1, v_2)}$$

2. **Logical Formulae:** Imagine we have a simple propositional expression language[1]:

$$\frac{}{\top\ \textsc{Prop}} \qquad \frac{}{\bot\ \textsc{Prop}} \qquad \frac{x\ \textsc{Prop} \quad y\ \textsc{Prop}}{x \land y\ \textsc{Prop}} \qquad \frac{x\ \textsc{Prop}}{\neg x\ \textsc{Prop}}$$

[1] Yes, the grammar is ambiguous, but assume it's just a symbolic representation of abstract syntax.

(a) Here are the big-step evaluation semantics for this language:

$$\frac{}{\top \Downarrow \texttt{True}}\textsc{True} \qquad \frac{}{\bot \Downarrow \texttt{False}}\textsc{False}$$

$$\frac{x \Downarrow \texttt{True}}{\neg x \Downarrow \texttt{False}}\textsc{Not}_1 \qquad \frac{x \Downarrow \texttt{False}}{\neg x \Downarrow \texttt{True}}\textsc{Not}_2 \qquad \frac{x \Downarrow \texttt{False}}{x \wedge y \Downarrow \texttt{False}}\textsc{And}_1 \qquad \frac{x \Downarrow \texttt{True} \quad y \Downarrow v}{x \wedge y \Downarrow v}\textsc{And}_2$$

Determine a small step (SOS) semantics for the language Prop.

i. [$\star$] Identify the set of states Σ, the set of initial states I, and the set of final states F.

> **Solution:** The set of states Σ is simply the set of all expressions Prop. $F = \{\top, \bot\}$. $I = \Sigma \backslash F$.

ii. [$\star\star$] Provide inference rules for a relation $\mapsto \subseteq \Sigma \times \Sigma$, which performs one step only of the expression evaluation.

> **Solution:**
>
> $$\frac{a \mapsto a'}{\neg a \mapsto \neg a'}\text{SOS-}\textsc{Not}_1 \qquad \frac{}{\neg \top \mapsto \bot}\text{SOS-}\textsc{Not}_2 \qquad \frac{}{\neg \bot \mapsto \top}\text{SOS-}\textsc{Not}_3$$
>
> $$\frac{a \mapsto a'}{a \wedge b \mapsto a' \wedge b}\text{SOS-}\textsc{And}_1 \qquad \frac{}{\top \wedge b \mapsto b}\text{SOS-}\textsc{And}_2 \qquad \frac{}{\bot \wedge b \mapsto \bot}\text{SOS-}\textsc{And}_3$$

(b) We shall now prove that the *reflexive, transitive closure* of $\mapsto$, $\overset{\star}{\mapsto}$, is implied by the big-step semantics above. $\overset{\star}{\mapsto}$ is defined by the following rules:

$$\frac{}{x \overset{\star}{\mapsto} x}\textsc{Refl}^* \qquad \frac{x \mapsto y \quad y \overset{\star}{\mapsto} z}{x \overset{\star}{\mapsto} z}\textsc{Trans}^*$$

i. [$\star\star$] First prove the following transitivity lemma:

$$\frac{p \overset{\star}{\mapsto} q \quad q \overset{\star}{\mapsto} r}{p \overset{\star}{\mapsto} r}\textsc{Transitive}$$

> **Solution:** We will proceed by rule induction on the first premise, that $p \overset{\star}{\mapsto} q$. Therefore, for each case, we must show:
>
> $$\frac{q \overset{\star}{\mapsto} r}{p \overset{\star}{\mapsto} r}$$
>
> *Base case (where $p = q$ from rule* Refl**).* For this case, as $p = q$, our goal is just
>
> $$\frac{q \overset{\star}{\mapsto} r}{q \overset{\star}{\mapsto} r}$$
>
> which is trivial.
>
> *Inductive case (from rule* Trans**).* We know that $p \mapsto p'$ $(*)$ and $p' \overset{\star}{\mapsto} q$ $(**)$. We must show that $q \overset{\star}{\mapsto} r$ $(***)$ implies $p \overset{\star}{\mapsto} r$, given the inductive hypothesis from $(**)$ that:
>
> $$\frac{q \overset{\star}{\mapsto} r}{p' \overset{\star}{\mapsto} r}IH$$
>
> We can now show our goal:
>
> $$\frac{\dfrac{}{p \mapsto p'}(*) \qquad \dfrac{\dfrac{}{q \overset{\star}{\mapsto} r}(***)}{p' \mapsto r}IH}{p \overset{\star}{\mapsto} r}\textsc{Trans}^*$$
>
> □

ii. [$\star\star$] Now prove the following two lemmas about $\textsc{Not}$:

$$\frac{x \overset{\star}{\mapsto} \top}{\neg x \overset{\star}{\mapsto} \bot}\textsc{Lemma-Not}_1 \qquad \frac{x \overset{\star}{\mapsto} \bot}{\neg x \overset{\star}{\mapsto} \top}\textsc{Lemma-Not}_2$$

Solution:

Proof of $\textsc{Lemma-Not}_1$*.* We proceed by rule induction on the premise, that $x \overset{\star}{\mapsto} \top$, with the goal of proving that $\neg x \overset{\star}{\mapsto} \bot$.
Base Case (from $\textsc{Refl}^*$*)*, where $\top \overset{\star}{\mapsto} \top$, we must show that $\neg\top \overset{\star}{\mapsto} \bot$:

$$\dfrac{\dfrac{}{\neg\top \mapsto \bot}\text{SOS-}\textsc{Not}_2 \qquad \dfrac{}{\bot \overset{\star}{\mapsto} \bot}\textsc{Refl}^*}{\neg\top \overset{\star}{\mapsto} \bot}\textsc{Trans}^*$$

Inductive Case (from $\textsc{Trans}^*$*)*, where $x \mapsto x'$ $(*)$ and $x' \overset{\star}{\mapsto} \top (**)$, we must show that $\neg x \overset{\star}{\mapsto} \bot$, with the inductive hypothesis from $(**)$ that $\neg x' \overset{\star}{\mapsto} \bot$.

$$\dfrac{\dfrac{\dfrac{}{x \mapsto x'}(*)}{\neg x \mapsto \neg x'}\text{SOS-}\textsc{Not}_1 \qquad \dfrac{}{\neg x' \overset{\star}{\mapsto} \bot}I.H}{\neg x \overset{\star}{\mapsto} \bot}\textsc{Trans}^*$$

Proof of $\textsc{Lemma-Not}_2$ is very similar and is omitted. □

iii. [$\star\star$] Now prove the following lemmas about $\textsc{And}$:

$$\frac{x \overset{\star}{\mapsto} \bot}{x \land y \overset{\star}{\mapsto} \bot}\textsc{Lemma-And}_1 \qquad \frac{x \overset{\star}{\mapsto} \top}{x \land y \overset{\star}{\mapsto} y}\textsc{Lemma-And}_2$$

Solution:

Proof of $\textsc{Lemma-And}_1$*.* We proceed by rule induction on the premises that $x \overset{\star}{\mapsto} \bot$, with the goal of proving that $x \land y \overset{\star}{\mapsto} \bot$.
Base Case (from $\textsc{Refl}^*$*)*, where $\bot \overset{\star}{\mapsto} \bot$, we must show that $\bot \land y \overset{\star}{\mapsto} \bot$:

$$\dfrac{\dfrac{}{\bot \land y \mapsto \bot}\text{SOS-}\textsc{And}_3 \qquad \dfrac{}{\bot \overset{\star}{\mapsto} \bot}\textsc{Refl}^*}{\bot \land y \overset{\star}{\mapsto} \bot}\textsc{Trans}^*$$

Inductive Case (from $\textsc{Trans}^*$*)*, where $x \mapsto x'$ $(*)$ and $x' \overset{\star}{\mapsto} \bot$ $(**)$, we must show that $x \land y \overset{\star}{\mapsto} \bot$, with the inductive hypothesis from $(**)$ that $x' \land y \overset{\star}{\mapsto} \bot$.

$$\dfrac{\dfrac{\dfrac{}{x \mapsto x'}(*)}{x \land y \mapsto x' \land y}\text{SOS-}\textsc{And}_1 \qquad \dfrac{}{x' \land y \overset{\star}{\mapsto} \bot}I.H}{x \land y \overset{\star}{\mapsto} \bot}\textsc{Trans}^*$$

Proof of $\textsc{Lemma-And}_2$ is very similar, and is therefore omitted. □

iv. [$\star\star\star$] Using these lemmas or otherwise, show that $E \Downarrow V$ implies $\sigma_E \overset{\star}{\mapsto} \sigma_V$, where σ_E is the state corresponding to the expression E and σ_V is the final state corresponding to the value V.

Solution: We define $\sigma_{\texttt{True}} = \top$, $\sigma_{\texttt{False}} = \bot$, and $\sigma_E = E$ for all expressions E.
We proceed by rule induction on the rules of $\Downarrow$.

Base case (from rule $\textsc{True}$*).* We must show that $\top \overset{\star}{\mapsto} \sigma_{\texttt{True}}$, i.e $\top \overset{\star}{\mapsto} \top$ which is true by rule $\textsc{Refl}^*$.
Base case (from rule $\textsc{False}$*)* We must show that $\bot \overset{\star}{\mapsto} \sigma_{\texttt{False}}$, i.e $\bot \overset{\star}{\mapsto} \bot$ which is true by rule $\textsc{Refl}^*$.
Not cases:
Inductive case (from rule $\textsc{Not}_1$*).* We must show, assuming $x \Downarrow \texttt{True}$ $(*)$, that $\neg x \overset{\star}{\mapsto} \bot$.

We have the I.H. from $(*)$ that $x \overset{\star}{\mapsto} \sigma_{\texttt{True}}$ i.e. $x \overset{\star}{\mapsto} \top$. By Lemma-Not$_1$, we can conclude that $\neg x \overset{\star}{\mapsto} \bot$ as required.

Inductive case (from rule Not$_2$*).* We must show, assuming $x \Downarrow \texttt{False}$ $(*)$, that $\neg x \overset{\star}{\mapsto} \top$. We have the I.H. from $(*)$ that $x \overset{\star}{\mapsto} \sigma_{\texttt{False}}$ i.e. $x \overset{\star}{\mapsto} \bot$. By Lemma-Not$_2$, we can conclude that $\neg x \overset{\star}{\mapsto} \top$ as required.

Inductive case (from rule And$_1$*).* We must show, assuming $x \Downarrow \texttt{False}$ $(*)$, that $x \wedge y \overset{\star}{\mapsto} \bot$. We have the I.H. from $(*)$ that $x \overset{\star}{\mapsto} \bot$. Thus, we have our goal from Lemma-And$_1$ with the I.H.

Inductive case (from rule And$_3$*).* We must show, assuming $x \Downarrow \texttt{True}$ $(*)$ and $y \Downarrow v$ $(**)$, that $x \wedge y \overset{\star}{\mapsto} \sigma_v$. We have two inductive hypotheses:

1. $x \overset{\star}{\mapsto} \top$ from $(*)$
2. $y \overset{\star}{\mapsto} \sigma_v$ from $(**)$

We can show our goal as follows:

$$\dfrac{\dfrac{\dfrac{}{x \overset{\star}{\mapsto} \top}\,IH_1}{x \wedge y \overset{\star}{\mapsto} y}\,\text{Lemma-And}_2 \qquad \dfrac{}{y \overset{\star}{\mapsto} \sigma_v}\,IH_2}{x \wedge y \overset{\star}{\mapsto} \sigma_v}\,\text{Transitive}$$

Thus, by induction, we have shown that $E \Downarrow V \implies E \mapsto \sigma_V$ □

(c) Now we will prove the other direction of the equivalence.

i. [$\star\star\star\star$] Prove the following lemma by rule induction on the small-step premise:

$$\frac{e \mapsto e' \quad e' \Downarrow v}{e \Downarrow v}$$

Hint: It might help to keep v arbitrary in the induction, and thus prove for each case:

$$\forall v.\ \frac{e' \Downarrow v}{e \Downarrow v}$$

Solution: We begin rule induction on the cases for $e \mapsto e'$.

Base case (from rule SOS-Not$_2$*).* Where $e = \neg\top$ and $e' = \bot$. We must show that $\bot \Downarrow v$ implies $\neg\top \Downarrow v$. The only way for $\bot \Downarrow v$ to hold is if $v = \texttt{False}$. Thus we can show $\neg\top \Downarrow v$ using rules Not$_1$ and True.

Base case (from rule SOS-Not$_3$*)* Where $e = \neg\bot$ and $e' = \top$. We must show that $\top \Downarrow v$ implies $\neg\bot \Downarrow v$. The only way for $\top \Downarrow v$ to hold is if $v = \texttt{True}$. Therefore we must show $\neg\bot \Downarrow \texttt{True}$, trivial from rules Not$_2$ and False.

Base case (from rule SOS-And$_2$*)* Where $e = \top \wedge e'$. We must show that $e' \Downarrow v$ implies $\top \wedge e' \Downarrow v$. This is trivially shown by rule And$_2$ and True.

Base case (from rule SOS-And$_3$*)* Where $e = \bot \wedge y$ and $e' = \bot$. We must show that, assuming $\bot \wedge y \Downarrow v$ $(*)$, that $\bot \Downarrow v$. If v were $\texttt{True}$, the assumption $(*)$ would indicate that $\bot \Downarrow \texttt{True}$, which is impossible. Therefore, v must be $\texttt{False}$, and our goal can be trivially shown by the rule False.

Inductive case (from rule SOS-Not$_1$*)* Where $e = \neg a$ and $e' = \neg a'$ and $a \mapsto a'$. We get the inductive hypothesis that:

$$\forall v.\ \frac{a' \Downarrow v}{a \Downarrow v}$$

And we must show that, assuming $\neg a' \Downarrow v$, that $\neg a \Downarrow v$. If $\neg a' \Downarrow v$, this must indicate that either $a' \Downarrow \texttt{True}$, in which case $v = \texttt{False}$ or $a' \Downarrow \texttt{False}$, in which case $v = \texttt{True}$. Either way,

we can use our I.H. to conclude that $a \Downarrow \texttt{True}$ or $a \Downarrow \texttt{False}$ respectively. Then we may show our goal using either $\textsc{Not}_1$ or $\textsc{Not}_2$.
Inductive case (from rule SOS-$\textsc{And}_1$*)* Where $e = a \wedge b$ and $e' = a' \wedge b$ and $a \mapsto a'$. We get the inductive hypothesis that:

$$\forall v.\ \frac{a' \Downarrow v}{a \Downarrow v}$$

And we must show that, assuming $a' \wedge b \Downarrow v$, that $a \wedge b \Downarrow v$. If $a' \wedge b \Downarrow v$, this must indicate that either $a' \Downarrow \texttt{True}$, in which case $b \Downarrow v$, or $a' \Downarrow \texttt{False}$, in which case $v = \texttt{False}$. Either way, we can use our I.H. to conclude that $a \Downarrow \texttt{True}$ or $a \Downarrow \texttt{False}$ respectively. Then we may show our goal using either $\textsc{And}_1$ or $\textsc{And}_2$.

□

ii. [$\star$] Now prove that every completed small step trace has a correspoding big step evaluation:

$$\frac{e \overset{\star}{\mapsto} \sigma_v}{e \Downarrow v}$$

Solution: We begin rule induction on the cases for $e \overset{\star}{\mapsto} \sigma_v$.

Base case (from rule $\textsc{Refl}^*$*).* Where $e = \sigma_v$. The term e can either be $\top$ or $\bot$, and in either case we have that $e \Downarrow v$.

Inductive case (from rule $\textsc{Trans}^*$) Where $e \mapsto e'$ and $e' \overset{\star}{\mapsto} \sigma_v$. We get the inductive hypothesis that $e' \Downarrow v$. We must show that $e \Downarrow v$, which can be done easily using the lemma proven in the previous part. □

(d) [$\star\star$] Suppose we wanted to add quantifiers and variables to our logic language:

$$\frac{x\ \textsc{Prop}}{\exists v.\ x\ \textsc{Prop}} \qquad \frac{x\ \textsc{Prop}}{\forall v.\ x\ \textsc{Prop}} \qquad \frac{v \text{ is a variable name}}{v\ \textsc{Prop}}$$

Write a static semantics judgement for this language, written $\vdash e\ \mathit{Ok}$ (with whatever context you like before the $\vdash$), that ensures that there are *no free variables* in a given logical formula. Remember that a *free variable* is a variable that is not bound by a quantifier or lambda.

Solution: The context shall be a set of variable names, denoted Γ.

$$\frac{v \in \Gamma}{\Gamma \vdash v\ \mathit{Ok}}\textsc{LookupEnv} \qquad \frac{\Gamma \cup \{v\} \vdash x\ \mathit{Ok}}{\Gamma \vdash \exists v.\ x\ \mathit{Ok}}\textsc{Exists} \qquad \frac{\Gamma \cup \{v\} \vdash x\ \mathit{Ok}}{\Gamma \vdash \forall v.\ x\ \mathit{Ok}}\textsc{Forall}$$

$$\frac{\Gamma \vdash x\ \mathit{Ok}}{\Gamma \vdash \neg x\ \mathit{Ok}}\textsc{Not} \qquad \frac{\Gamma \vdash x\ \mathit{Ok} \quad \Gamma \vdash y\ \mathit{Ok}}{\Gamma \vdash x \wedge y\ \mathit{Ok}}\textsc{And}$$

This is more or less analogous to the scope-checking rules presented in lectures.

3. **Bizarro-Poland:** Imagine we have a reverse Polish notation calculator language. Reverse Polish notation is an old calculator format that does not require the use of parenthetical expressions. To achieve this, it moves all operators to post-fix, rather than in-fix order. E.g $1 + 2$ becomes $1\ 2\ +$, or $1 - (3 + 2)$ becomes $1\ 3\ 2\ +\ -$. These calculators evaluated these expressions by pushing symbols onto a stack until an operator was encountered, when two symbols would be popped off and the result of the operation pushed on. The grammar is easily defined:

$$\frac{x \in \mathbb{N}}{x \text{ Symbol}} \qquad \frac{x \in \{+, -, /, *\}}{x \text{ Symbol}}$$

$$\frac{}{\epsilon \text{ RPN}} \qquad \frac{x \text{ Symbol} \quad xs \text{ RPN}}{x\ xs \text{ RPN}}$$

(a) The issue is that this grammar allows for invalid programs (such as $1 + 2$ or $-\ +\ *$).

i. [$\star\star$] Write some static semantics inference rules for a judgement $\vdash e\ \mathit{Ok}$ to ensure that programs are well formed.

Solution: We will equip our rules with context that includes the *number of values* that are available on the stack.
The empty program is only valid if there is exactly one value left on the stack - this means the program will evaluate to one result:

$$\dfrac{}{1 \vdash \epsilon\ Ok}\textsc{Empty}$$

Prepending a number to a program means that one less value is required on the stack:

$$\dfrac{x \in \mathbb{N} \quad (n+1) \vdash xs\ Ok}{n \vdash x\ xs\ Ok}\textsc{Num}$$

Prepending a symbol will require two values on the stack, and produce one value. The structure of the rule we write reflects this:

$$\dfrac{x \in \{+, -, *, /\} \quad (n+1) \vdash xs\ Ok}{(n+2) \vdash x\ xs\ Ok}\textsc{Op}$$

ii. [$\star$] Show that $\vdash 1\ 3\ 2 + -\ Ok$.

Solution:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{1 \vdash \epsilon\ Ok}\textsc{Empty}}{2 \vdash -\ Ok}\textsc{Op}}{3 \vdash +-\ Ok}\textsc{Op}}{2 \vdash 2 + -\ Ok}\textsc{Num}}{1 \vdash 3\ 2 + -\ Ok}\textsc{Num}}{0 \vdash 1\ 3\ 2 + -\ Ok}\textsc{Num}.$$

(b) We will now define some big-step evaluation semantics for this calculator. It may be helpful to read the program from right-to-left rather than left-to-right.

i. [$\star$] Identify the set of evaluable expressions E, and the set of result values V. It may be helpful to use a *stack*, defined as follows:

$$\dfrac{}{\circ\ \text{Stack}} \qquad \dfrac{x \in \mathbb{N} \quad xs\ \text{Stack}}{x \rhd xs\ \text{Stack}}$$

Solution: The set E is simply the set of all e such that e *RPN*. V is the set of all stacks.

ii. [$\star\star$] Define a relation $\Downarrow \subseteq E \times V$ which evaluates RPN programs.

Solution: The empty program evaluates to the empty stack:

$$\dfrac{}{\epsilon \Downarrow \circ}\text{BS-}\textsc{Empty}$$

The non-empty program ending in an operator performs the operation on the stack resulting from the previous symbols:

$$\dfrac{xs \Downarrow a \rhd b \rhd s \quad x \in \{+, -, /, *\}}{xs\ x \Downarrow (a\ x\ b) \rhd s}\text{BS-}\textsc{Op}$$

The non-empty program ending in a number simply pushes a number onto the stack from the previous symbols:

$$\dfrac{xs \Downarrow s \quad x \in \mathbb{N}}{xs\ x \Downarrow x \rhd s}\text{BS-}\textsc{Num}$$

(c) [$\star\star$] Now we will try small-step semantics.
Our states Σ will be of the form $s \vdash p$ where s is a stack of natural numbers and p is an RPN program.
Initial states are all states of the form $\circ \vdash p$.
Final states are all states of the form $n \rhd \circ \vdash \epsilon$.

Define a relation $\mapsto \subseteq \Sigma \times \Sigma$, which evaluates one step of the calculation.

Solution: Numbers simply push to the stack:

$$\frac{x \in \mathbb{N}}{s \vdash x\ xs \mapsto x \rhd s \vdash xs}\text{SS-N\textsc{um}}$$

Operations simply pop two elements from the stack and operate on them.

$$\frac{x \in \{+, -, *, /\}}{a \rhd b \rhd s \vdash x\ xs \mapsto a\ x\ b \rhd s \vdash xs}\text{SS-O\textsc{p}}$$

(d) [★★★★] Now we will prove the easier direction of the equivalence proof. Show using rule induction on $\Downarrow$ that, for all expressions e, if $e \Downarrow v$ then $\sigma_e \overset{\star}{\mapsto} \sigma_v$ (where σ_e and σ_v are initial and final states respectively corresponding to e and v).
Hint: You may find it helpful to assume the following lemma (A proof of it is provided in the solutions):

$$\frac{\circ \vdash xs \overset{\star}{\mapsto} v \vdash \epsilon}{\circ \vdash xs\ x \overset{\star}{\mapsto} v \vdash x}\text{A\textsc{ppend}}$$

It is worth noting that the lemma Transitive from Question 1 applies here also.

Solution: We shall define σ_e as $\circ \vdash e$, and σ_v as $v \vdash \epsilon$.

Base case (from rule BS-Empty*).* Where $e = \epsilon$ and $v = \circ$. We must show $\circ \vdash \epsilon \overset{\star}{\mapsto} \circ \vdash \epsilon$, trivially true by rule Refl*.
Inductive case (from rule BS-Num*).* Here $e = xs\ x$ for some $x \in \mathbb{N}$, and $v = x \rhd v'$ for some stack v'. We know that $xs \Downarrow v'$ which gives rise to the I.H. that $\circ \vdash xs \overset{\star}{\mapsto} v' \vdash \epsilon$. Using Append with that I.H., we can conclude that $\circ \vdash xs\ x \overset{\star}{\mapsto} v' \vdash x$ $(*)$. As we know that $\overset{\star}{\mapsto}$ is Transitive, we can write a chain of reasoning here rather than a derivation tree.

$$\begin{array}{rcll} \circ \vdash xs\ x & \overset{\star}{\mapsto} & v' \vdash x & (*) \\ & \mapsto & x \rhd v' \vdash \epsilon & (\text{SS-N\textsc{um}}) \end{array}$$

Inductive case (from rule BS-Op*).* Here $e = xs\ x$ for some operator x and $v = (a\ x\ b) \rhd v'$ for some stack v'. We know that $xs \Downarrow a \rhd b \rhd v'$, which gives rise to the inductive hypothesis that $\circ \vdash xs \overset{\star}{\mapsto} a \rhd b \rhd v' \vdash \epsilon$, we must show that $\circ \vdash xs\ x \overset{\star}{\mapsto} (a\ x\ b) \rhd v' \vdash \epsilon$. From Append, we can therefore conclude that $\circ \vdash xs\ x \overset{\star}{\mapsto} a \rhd b \rhd v' \vdash x$ $(*)$.

$$\begin{array}{rcll} \circ \vdash xs\ x & \overset{\star}{\mapsto} & a \rhd b \rhd v' \vdash x & (*) \\ & \mapsto & (a\ x\ b) \rhd v' \vdash \epsilon & (\text{SS-O\textsc{p}}) \end{array}$$

Thus we have shown by induction that the big step semantics map to the small step semantics. □

We still need to prove Append. We proceed rule induction on the premise, however we will strengthen our proof goal to the more general rule below:

$$\frac{s \vdash xs \overset{\star}{\mapsto} v \vdash \epsilon}{s \vdash xs\ x \overset{\star}{\mapsto} v \vdash x}\text{A\textsc{ppend}'}$$

This trivially implies Append by setting $s = \circ$.

Base case (when $xs = \epsilon$ and $s = v$, from rule Refl**).* We must show that $v \vdash x \overset{\star}{\mapsto} v \vdash x$, trivial by rule Refl*.
Inductive case (when $s \vdash xs \mapsto s' \vdash xs'$ (†) for some s' and xs', and $s' \vdash xs' \overset{\star}{\mapsto} v \vdash \epsilon$ $(*)$). We get the inductive hypothesis from $(*)$ that $s' \vdash xs'\ x \mapsto v \vdash x$.
Looking at the possible cases for (†), we can always prove that $s \vdash xs\ x \mapsto s' \vdash xs'\ x$ $(**)$, as the rules for the small-step semantics only ever examine the left hand side of a non-empty sequence. Therefore, we may now show our goal:

$$\begin{array}{rcll} s \vdash xs\ x & \mapsto & s' \vdash xs'\ x & (**) \\ & \mapsto & v \vdash x & (\text{I.H.}) \end{array}$$

□

(e) [★★★★] Show that your static semantics defined in (a) ensure that the program will not reach a stuck state. That is, show that $\vdash e\ \mathit{Ok} \implies \circ \vdash e \overset{\star}{\mapsto} s \vdash \epsilon$, for some s. You may find it helpful to generalise your proof goal before beginning induction.

Solution: We shall start by generalising our goal to the more flexible $n \vdash e\ \mathit{Ok} \implies v_1 \triangleright v_2 \triangleright \cdots \triangleright v_n \triangleright \circ \vdash e \overset{\star}{\mapsto} s \vdash \epsilon$ for some s and all v_1 through v_n. This trivially implies our first goal by setting $n = 0$.

Base case (where $e = \epsilon$ and $n = 1$, from rule Empty*).* We must show that $v_1 \triangleright \circ \vdash \epsilon \overset{\star}{\mapsto} s \vdash \epsilon$, for some s. This is trivially true by rule Refl*, where $s = v_1 \triangleright \circ$.

Inductive case (where $e = x\ xs$ for some $x \in \mathbb{N}$, and $n + 1 \vdash xs\ \mathit{Ok}$ $(*)$) Our inductive hypothesis from $(*)$ is that $v'_1 \triangleright \cdots \triangleright v'_n \triangleright v'_{n+1} \triangleright \circ \vdash xs \overset{\star}{\mapsto} s' \vdash \epsilon$ for some s' and all v'_0 through v'_{n+1}.
We can show our goal by setting our goal's $s = s'$, the IH's $v'_1 = x$ and $v'_i = v_{i-1}$ for all subsequent i.

$$\cfrac{\cfrac{}{v_1 \triangleright \cdots \triangleright v_n \triangleright \circ \vdash x\ xs \mapsto x \triangleright v_1 \triangleright \cdots \triangleright v_n \triangleright \circ \vdash xs}\textsc{SS-Num} \qquad \cfrac{}{x \triangleright v_1 \triangleright \cdots \triangleright v_n \triangleright \circ \vdash xs \overset{\star}{\mapsto} s' \vdash \epsilon}\mathit{IH}}{v_1 \triangleright v_2 \triangleright \cdots \triangleright v_n \triangleright \circ \vdash x\ xs \overset{\star}{\mapsto} s' \vdash \epsilon}\textsc{Trans}^*_2$$

The case where x is an operator is very similar, just different stacks. □