15–819A3 — Class Notes 1
Reasoning about Low-Level Programming Languages

John C. Reynolds
Department of Computer Science
Carnegie Mellon University

February 13, 2001 (revised)

Iterated Independent Conjunction

$$m > n \Rightarrow \left(\bigodot_{i=m}^{n} p(i) \Leftrightarrow \mathbf{emptyheap}\right)$$

$$m = n \Rightarrow \left(\bigodot_{i=m}^{n} p(i) \Leftrightarrow p(m)\right)$$

$$k \leq m \leq n + 1 \Rightarrow \left(\bigodot_{i=k}^{n} p(i) \Leftrightarrow \left(\bigodot_{i=k}^{m-1} p(i) \ * \ \bigodot_{i=m}^{n} p(i)\right)\right)$$

$$\bigodot_{i=m}^{n} p(i) \Leftrightarrow \bigodot_{i=m-k}^{n-k} p(i+k)$$

$$m \leq n \Rightarrow \left(\left(\bigodot_{i=m}^{n} p(i)\right) \ * \ q \Leftrightarrow \bigodot_{i=m}^{n}(p(i) \ * \ q)\right)$$

when q is pure

$$m \leq j \leq n \Rightarrow \left(\left(\bigodot_{i=m}^{n} p(i)\right) \Rightarrow (p(j) \ * \ \mathbf{true})\right)$$

An Inference Rule for Array Allocation

$$\begin{array}{l} \{p\} \\ v := \mathbf{allocate}\ e \\ \{p \ * \ \bigodot_{i=v}^{v+e-1} i \mapsto -\} \end{array}$$

where v does not occur free in p or e.

Subset Lists

Let

$$Q(\sigma,\beta) \stackrel{\mathrm{def}}{=} \#\beta = \#\sigma \wedge \forall_{\mathsf{i}=1}^{\#\beta}(\mathsf{list}\ \sigma_{\mathsf{i}}\,(\beta_{\mathsf{i}},\mathbf{nil})\ *\ \mathbf{true})$$

$$R(\beta) \stackrel{\mathrm{def}}{=} (\beta_{\#\beta} = \mathbf{nil} \wedge \mathbf{emptyheap})\ *$$
$$\bigodot_{\mathsf{i}=1}^{\#\beta-1}(\exists \mathsf{a},\mathsf{k}.\ \ \mathsf{i} < \mathsf{k} \leq \#\beta \wedge \beta_{\mathsf{i}} \mapsto \mathsf{a},\beta_{\mathsf{k}})$$

$$W(\beta,\gamma,\mathsf{a}) \stackrel{\mathrm{def}}{=} \#\gamma = \#\beta \wedge \bigodot_{\mathsf{i}=1}^{\#\gamma}\gamma_{\mathsf{i}} \mapsto \mathsf{a},(\beta^{\dagger})_{\mathsf{i}}$$

$$\#\mathsf{ext}_{\mathsf{a}}\sigma \stackrel{\mathrm{def}}{=} \#\sigma$$
$$(\mathsf{ext}_{\mathsf{a}}\sigma)_i \stackrel{\mathrm{def}}{=} \mathsf{a}{\cdot}\sigma_i$$

$$\mathsf{ss}(\epsilon,\sigma) \stackrel{\mathrm{def}}{=} \sigma = [\epsilon]$$
$$\mathsf{ss}(\mathsf{a}{\cdot}\alpha,\sigma) \stackrel{\mathrm{def}}{=} \exists\sigma'.\ \ \mathsf{ss}(\alpha,\sigma') \wedge \sigma = (\mathsf{ext}_{\mathsf{a}}\sigma')^{\dagger}{\cdot}\sigma'$$

Properties:

$$Q([\epsilon],\mathbf{nil}) \Leftrightarrow \mathbf{true}$$

$$R(\mathbf{nil}) \Leftrightarrow \mathbf{emptyheap}$$

$$W(\beta,\gamma,\mathsf{a})\ *\ \mathsf{g} \mapsto \mathsf{a},\mathsf{b} \Leftrightarrow W(\beta{\cdot}\mathsf{b},\mathsf{g}{\cdot}\gamma,\mathsf{a})$$

$$Q(\sigma,\beta)\ *\ W(\beta,\gamma,\mathsf{a}) \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma,\gamma{\cdot}\beta)$$

$$R(\beta)\ *\ W(\beta,\gamma,\mathsf{a}) \Rightarrow R(\gamma{\cdot}\beta)$$

$$(Q(\sigma,\beta) \wedge R(\beta))\ *\ W(\beta,\gamma,\mathsf{a}) \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma,\gamma{\cdot}\beta) \wedge R(\gamma{\cdot}\beta)$$

Proofs (1)

$$
\begin{aligned}
&W(\beta,\gamma,\mathsf{a}) \ * \ \mathsf{g} \mapsto \mathsf{a},\mathsf{b} \\
&\quad \Leftrightarrow \#\gamma = \#\beta \wedge \Big(\mathsf{g} \mapsto \mathsf{a},\mathsf{b} \ * \ \textstyle\bigodot_{\mathsf{i}=1}^{\#\gamma} \gamma_{\mathsf{i}} \mapsto \mathsf{a},(\beta^{\dagger})_{\mathsf{i}}\Big) \\
&\quad \Leftrightarrow \#\mathsf{g}\cdot\gamma = \#\beta\cdot\mathsf{b} \wedge \Big(\Big(\textstyle\bigodot_{\mathsf{i}=1}^{1}(\mathsf{g}\cdot\gamma)_{\mathsf{i}} \mapsto \mathsf{a},((\beta\cdot\mathsf{b})^{\dagger})_{\mathsf{i}}\Big) \ * \\
&\qquad\quad \Big(\textstyle\bigodot_{\mathsf{i}=1}^{\#\mathsf{g}\cdot\gamma-1}(\mathsf{g}\cdot\gamma)_{\mathsf{i}+1} \mapsto \mathsf{a},((\beta\cdot\mathsf{b})^{\dagger})_{\mathsf{i}+1}\Big)\Big) \\
&\quad \Leftrightarrow \#\mathsf{g}\cdot\gamma = \#\beta\cdot\mathsf{b} \wedge \textstyle\bigodot_{\mathsf{i}=1}^{\#\mathsf{g}\cdot\gamma}(\mathsf{g}\cdot\gamma)_{\mathsf{i}} \mapsto \mathsf{a},((\beta\cdot\mathsf{b})^{\dagger})_{\mathsf{i}} \\
&\quad \Leftrightarrow W(\beta\cdot\mathsf{b},\mathsf{g}\cdot\gamma,\mathsf{a})
\end{aligned}
$$

Proofs (2)

Let

$$p(\mathsf{i}) \stackrel{\text{def}}{=} \mathsf{list}\ \sigma_{\mathsf{i}}\,(\beta_{\mathsf{i}}, \mathbf{nil})$$

$$q(\mathsf{i}) \stackrel{\text{def}}{=} \gamma_{\mathsf{i}} \mapsto \mathsf{a}, (\beta^{\dagger})_{\mathsf{i}}$$

$$n \stackrel{\text{def}}{=} \#\sigma$$

Then

$$\begin{aligned}
&Q(\sigma, \beta) \ * \ W(\beta, \gamma, \mathsf{a}) \\
&\quad \Rightarrow \Big(\#\beta = n \wedge \forall_{\mathsf{i}=1}^{\#\beta}(p(\mathsf{i}) \ * \ \mathbf{true})\Big) \ * \ \Big(\#\gamma = \#\beta \wedge \textstyle\bigodot_{\mathsf{i}=1}^{\#\gamma} q(\mathsf{i})\Big) \\
&\quad \Rightarrow \#\beta = n \wedge \#\gamma = n \wedge \Big(\forall_{\mathsf{i}=1}^{n}(p(\mathsf{i}) \ * \ \mathbf{true}) \ * \ \textstyle\bigodot_{\mathsf{i}=1}^{n} q(\mathsf{i})\Big) \\
&\quad \Rightarrow \#\beta = n \wedge \#\gamma = n \wedge \forall_{\mathsf{i}=1}^{n}\Big(p(\mathsf{i}) \ * \ \mathbf{true} \ * \ \textstyle\bigodot_{\mathsf{j}=1}^{n} q(\mathsf{j})\Big) \\
&\quad \Rightarrow \#\beta = n \wedge \#\gamma = n \wedge \forall_{\mathsf{i}=1}^{n}\Big(p(\mathsf{i}) \ * \ \mathbf{true}\Big) \wedge \\
&\qquad\quad \forall_{\mathsf{i}=1}^{n}\Big(p(n+1-\mathsf{i}) \ * \ \mathbf{true} \ * \ \textstyle\bigodot_{\mathsf{j}=1}^{n} q(\mathsf{j})\Big) \\
&\quad \Rightarrow \#\beta = n \wedge \#\gamma = n \wedge \forall_{\mathsf{i}=1}^{n}\Big(p(\mathsf{i}) \ * \ \mathbf{true}\Big) \wedge \\
&\qquad\quad \forall_{\mathsf{i}=1}^{n}\Big(p(n+1-\mathsf{i}) \ * \ \mathbf{true} \ * \ q(\mathsf{i})\Big) \\
&\quad \Rightarrow \#\beta = n \wedge \#\gamma = n \wedge \forall_{\mathsf{i}=1}^{n}\Big(p(\mathsf{i}) \ * \ \mathbf{true}\Big) \wedge \\
&\qquad\quad \forall_{\mathsf{i}=1}^{n}\Big(\mathsf{list}\ (\sigma^{\dagger})_{\mathsf{i}}\,((\beta^{\dagger})_{\mathsf{i}}, \mathbf{nil}) \ * \ \mathbf{true} \ * \ \gamma_{\mathsf{i}} \mapsto \mathsf{a}, (\beta^{\dagger})_{\mathsf{i}}\Big) \\
&\quad \Rightarrow \#\gamma = n \wedge \#\beta = n \wedge \forall_{\mathsf{i}=1}^{n}\Big(\mathsf{list}\ \sigma_{\mathsf{i}}\,(\beta_{\mathsf{i}}, \mathbf{nil}) \ * \ \mathbf{true}\Big) \wedge \\
&\qquad\quad \forall_{\mathsf{i}=1}^{n}\Big(\mathsf{list}\ ((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger})_{\mathsf{i}}\,(\gamma_{\mathsf{i}}, \mathbf{nil}) \ * \ \mathbf{true}\Big) \\
&\quad \Rightarrow \#\gamma{\cdot}\beta = \#(\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma \wedge \\
&\qquad\quad \forall_{\mathsf{i}=1}^{\#\gamma\cdot\beta}\Big(\mathsf{list}\ ((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma)_{\mathsf{i}}\,((\gamma{\cdot}\beta)_{\mathsf{i}}, \mathbf{nil}) \ * \ \mathbf{true}\Big) \\
&\quad \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma, \gamma{\cdot}\beta)
\end{aligned}$$

Proofs (3)

$$
\begin{aligned}
& R(\beta) \ * \ W(\beta, \gamma, \mathsf{a}) \\
& \quad \Rightarrow (\beta_{\#\beta} = \mathbf{nil} \wedge \mathbf{emptyheap}) \ * \\
& \qquad \textstyle\bigodot_{\mathsf{i}=1}^{\#\gamma} \gamma_{\mathsf{i}} \mapsto \mathsf{a}, (\beta^{\dagger})_{\mathsf{i}} \ * \\
& \qquad \textstyle\bigodot_{\mathsf{i}=1}^{\#\beta-1} (\exists \mathsf{a}, \mathsf{k}.\ \ \mathsf{i} < \mathsf{k} \leq \#\beta \wedge \beta_{\mathsf{i}} \mapsto \mathsf{a}, \beta_{\mathsf{k}}) \\
& \quad \Rightarrow ((\gamma{\cdot}\beta)_{\#\gamma\cdot\beta} = \mathbf{nil} \wedge \mathbf{emptyheap}) \ * \\
& \qquad \textstyle\bigodot_{\mathsf{i}=1}^{\#\gamma} (\exists \mathsf{a}, \mathsf{k}.\ \ \#\gamma < \mathsf{k} \leq \#\gamma{\cdot}\beta \wedge (\gamma{\cdot}\beta)_{\mathsf{i}} \mapsto \mathsf{a}, (\gamma{\cdot}\beta)_{\mathsf{k}}) \ * \\
& \qquad \textstyle\bigodot_{\mathsf{i}=\#\gamma+1}^{\#\gamma\cdot\beta-1} (\exists \mathsf{a}, \mathsf{k}.\ \ \mathsf{i} < \mathsf{k} \leq \#\gamma{\cdot}\beta \wedge (\gamma{\cdot}\beta)_{\mathsf{i}} \mapsto \mathsf{a}, (\gamma{\cdot}\beta)_{\mathsf{k}}) \\
& \quad \Rightarrow ((\gamma{\cdot}\beta)_{\#\gamma\cdot\beta} = \mathbf{nil} \wedge \mathbf{emptyheap}) \ * \\
& \qquad \textstyle\bigodot_{\mathsf{i}=1}^{\#\gamma\cdot\beta-1} (\exists \mathsf{a}, \mathsf{k}.\ \ \mathsf{i} < \mathsf{k} \leq \#\gamma{\cdot}\beta \wedge (\gamma{\cdot}\beta)_{\mathsf{i}} \mapsto \mathsf{a}, (\gamma{\cdot}\beta)_{\mathsf{k}}) \\
& \quad \Rightarrow R(\gamma{\cdot}\beta)
\end{aligned}
$$

$$
\begin{aligned}
& (Q(\sigma, \beta) \wedge R(\beta)) \ * \ W(\beta, \gamma, \mathsf{a}) \\
& \quad \Rightarrow (Q(\sigma, \beta) \ * \ W(\beta, \gamma, \mathsf{a})) \wedge (R(\beta) \ * \ W(\beta, \gamma, \mathsf{a})) \\
& \quad \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma, \gamma{\cdot}\beta) \wedge R(\gamma{\cdot}\beta)
\end{aligned}
$$

$\{\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil})\}$

$\mathsf{j} := \mathbf{cons}(\mathbf{nil},\mathbf{nil})\;;$

$\{\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{j} \mapsto \mathbf{nil},\mathbf{nil}\}$

$\{\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\mathbf{nil}\,(\mathsf{j},\mathbf{nil})\}$

$\{\mathsf{ss}(\epsilon,[\epsilon])\wedge(\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\mathbf{nil}\,(\mathsf{j},\mathbf{nil}) \;*\; (Q([\epsilon],\mathbf{nil})\wedge R(\mathbf{nil})))\}$

$\{\exists\alpha',\alpha'',\sigma,\beta.\;\; \alpha'^{\dagger}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil}) \;*\; (Q(\sigma,\beta) \wedge R(\beta)))\}$

$\mathbf{while}\ \mathsf{i} \neq \mathbf{nil}\ \mathbf{do}$

$\quad\{\exists\alpha',\alpha'',\sigma,\beta,\mathsf{a},\mathsf{k}.\;\; \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad\quad(\mathsf{i} \mapsto \mathsf{a},\mathsf{k} \;*\; \mathsf{list}\,\alpha''\,(\mathsf{k},\mathbf{nil}) \;*\; \mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil}) \;*\; (Q(\sigma,\beta)\wedge R(\beta)))\}$

$\quad\big(\mathsf{a} := [\mathsf{i}]\;;\; \mathsf{k} := [\mathsf{i}+1]\;;\; \mathbf{dispose}\ \mathsf{i}\;;\; \mathbf{dispose}\ \mathsf{i}+1\;;\; \mathsf{i} := \mathsf{k}\;;$

$\quad\{\exists\alpha',\alpha'',\sigma,\beta.\;\; \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil}) \;*\; (Q(\sigma,\beta) \wedge R(\beta)))\}$

$\quad\{\exists\alpha',\alpha'',\sigma,\beta.\;\; \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\epsilon\,(\mathsf{j},\mathsf{j}) \;*\; \mathsf{list}\,\epsilon\,(\mathsf{j},\mathsf{j}) \;*\; \mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil}) \;*$

$\quad\quad\quad(Q(\sigma,\epsilon{\cdot}\beta) \wedge R(\epsilon{\cdot}\beta)) \;*\; W(\epsilon,\epsilon,\mathsf{a}))\}$

$\quad\mathsf{l} := \mathsf{j}\;;\; \mathsf{m} := \mathsf{j}\;;$

$\quad\{\exists\alpha',\alpha'',\sigma,\beta.\;\; \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\epsilon\,(\mathsf{l},\mathsf{j}) \;*\; \mathsf{list}\,\epsilon\,(\mathsf{j},\mathsf{m}) \;*\; \mathsf{list}\,\beta\,(\mathsf{m},\mathbf{nil}) \;*$

$\quad\quad\quad(Q(\sigma,\epsilon{\cdot}\beta) \wedge R(\epsilon{\cdot}\beta)) \;*\; W(\epsilon,\epsilon,\mathsf{a}))\}$

$\quad\{\exists\alpha',\alpha'',\sigma,\beta',\beta'',\gamma.\;\; \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha',\sigma) \wedge$

$\quad\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil}) \;*\; \mathsf{list}\,\gamma\,(\mathsf{l},\mathsf{j}) \;*\; \mathsf{list}\,\beta'\,(\mathsf{j},\mathsf{m}) \;*\; \mathsf{list}\,\beta''\,(\mathsf{m},\mathbf{nil}) \;*$

$\quad\quad\quad(Q(\sigma,\beta'{\cdot}\beta'') \wedge R(\beta'{\cdot}\beta'')) \;*\; W(\beta',\gamma,\mathsf{a}))\}$

$$
\begin{array}{l}
\textbf{while } \mathsf{m} \neq \textbf{nil do}\\
\quad \{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma, \mathsf{b}, \mathsf{m}'.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge\\
\qquad (\mathsf{list}\ \alpha''\,(\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\,(\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\ \beta'\,(\mathsf{j}, \mathsf{m}) \ *\\
\qquad\quad \mathsf{m} \mapsto \mathsf{b}, \mathsf{m}' \ * \ \mathsf{list}\ \beta''\,(\mathsf{m}', \mathbf{nil}) \ *\\
\qquad\quad (Q(\sigma, \beta'{\cdot}\mathsf{b}{\cdot}\beta'') \wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\}\\
\quad \big(\mathsf{b} := [\mathsf{m}]\ ;\\
\quad \{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma, \mathsf{m}'.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge\\
\qquad (\mathsf{list}\ \alpha''\,(\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\,(\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\ \beta'\,(\mathsf{j}, \mathsf{m}) \ *\\
\qquad\quad \mathsf{m} \mapsto \mathsf{b}, \mathsf{m}' \ * \ \mathsf{list}\ \beta''\,(\mathsf{m}', \mathbf{nil}) \ *\\
\qquad\quad (Q(\sigma, \beta'{\cdot}\mathsf{b}{\cdot}\beta'') \wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\}\\
\quad \{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma, \mathsf{m}', \mathsf{m}''.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge\\
\qquad (\mathsf{list}\ \alpha''\,(\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\,(\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\ \beta'\,(\mathsf{j}, \mathsf{m}'') \ *\\
\qquad\quad \mathsf{m}'' \mapsto \mathsf{b}, \mathsf{m}' \ * \ \mathsf{list}\ \beta''\,(\mathsf{m}', \mathbf{nil}) \ *\\
\qquad\quad (Q(\sigma, \beta'{\cdot}\mathsf{b}{\cdot}\beta'') \wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\\
\qquad \wedge\, \mathsf{m} + 1 \hookrightarrow \mathsf{m}'\}\\
\quad \mathsf{m} := [\mathsf{m} + 1]\ ;\\
\quad \{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma, \mathsf{m}''.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge\\
\qquad (\mathsf{list}\ \alpha''\,(\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\,(\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\ \beta'\,(\mathsf{j}, \mathsf{m}'') \ *\\
\qquad\quad \mathsf{m}'' \mapsto \mathsf{b}, \mathsf{m} \ * \ \mathsf{list}\ \beta''\,(\mathsf{m}, \mathbf{nil}) \ *\\
\qquad\quad (Q(\sigma, \beta'{\cdot}\mathsf{b}{\cdot}\beta'') \wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\}\\
\quad \{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge\\
\qquad (\mathsf{list}\ \alpha''\,(\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\,(\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\ \beta'{\cdot}\mathsf{b}\,(\mathsf{j}, \mathsf{m}) \ *\\
\qquad\quad \mathsf{list}\ \beta''\,(\mathsf{m}, \mathbf{nil}) \ *\\
\qquad\quad (Q(\sigma, \beta'{\cdot}\mathsf{b}{\cdot}\beta'') \wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\}\\
\quad \mathsf{g} := \mathbf{cons}(\mathsf{a}, \mathsf{b})\ ;
\end{array}
$$

$$
\begin{array}{l}
\quad\{\exists\alpha',\alpha'',\sigma,\beta',\beta'',\gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\alpha',\sigma)\wedge\\
\qquad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma\,(\mathsf{l},\mathsf{j})\ *\ \mathsf{list}\,\beta'{\cdot}\mathsf{b}\,(\mathsf{j},\mathsf{m})\ *\\
\qquad\quad\mathsf{list}\,\beta''\,(\mathsf{m},\mathbf{nil})\ *\\
\qquad\quad(Q(\sigma,\beta'{\cdot}\mathsf{b}{\cdot}\beta'')\wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta''))\ *\ W(\beta',\gamma,\mathsf{a})\ *\\
\qquad\quad\mathsf{g}\mapsto\mathsf{a},\mathsf{b})\}\\
\quad\mathsf{l}:=\mathbf{cons}(\mathsf{g},\mathsf{l})\\
\quad\{\exists\alpha',\alpha'',\sigma,\beta',\beta'',\gamma,\mathsf{l}'.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\alpha',\sigma)\wedge\\
\qquad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{l}\mapsto\mathsf{g},\mathsf{l}'\ *\ \mathsf{list}\,\gamma\,(\mathsf{l}',\mathsf{j})\ *\ \mathsf{list}\,\beta'{\cdot}\mathsf{b}\,(\mathsf{j},\mathsf{m})\ *\\
\qquad\quad\mathsf{list}\,\beta''\,(\mathsf{m},\mathbf{nil})\ *\\
\qquad\quad(Q(\sigma,\beta'{\cdot}\mathsf{b}{\cdot}\beta'')\wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta''))\ *\ W(\beta',\gamma,\mathsf{a})\ *\\
\qquad\quad\mathsf{g}\mapsto\mathsf{a},\mathsf{b})\}\\
\quad\{\exists\alpha',\alpha'',\sigma,\beta',\beta'',\gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\alpha',\sigma)\wedge\\
\qquad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\mathsf{g}{\cdot}\gamma\,(\mathsf{l},\mathsf{j})\ *\ \mathsf{list}\,\beta'{\cdot}\mathsf{b}\,(\mathsf{j},\mathsf{m})\ *\\
\qquad\quad\mathsf{list}\,\beta''\,(\mathsf{m},\mathbf{nil})\ *\\
\qquad\quad(Q(\sigma,\beta'{\cdot}\mathsf{b}{\cdot}\beta'')\wedge R(\beta'{\cdot}\mathsf{b}{\cdot}\beta''))\ *\ W(\beta'{\cdot}\mathsf{b},\mathsf{g}{\cdot}\gamma,\mathsf{a}))\}\big)\,;\\
\{\exists\alpha',\alpha'',\sigma,\beta',\gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\alpha',\sigma)\wedge\\
\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma\,(\mathsf{l},\mathsf{j})\ *\ \mathsf{list}\,\beta'\,(\mathsf{j},\mathbf{nil})\ *\\
\qquad(Q(\sigma,\beta')\wedge R(\beta'))\ *\ W(\beta',\gamma,\mathsf{a}))\}\\
\{\exists\alpha',\alpha'',\sigma,\beta',\gamma.\ \ (\mathsf{a}{\cdot}\alpha')^{\dagger}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\mathsf{a}{\cdot}\alpha',(\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma)\wedge\\
\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma{\cdot}\beta'\,(\mathsf{l},\mathbf{nil})\ *\\
\qquad(Q((\mathsf{ext}_{\mathsf{a}}\sigma)^{\dagger}{\cdot}\sigma,\gamma{\cdot}\beta')\wedge R(\gamma{\cdot}\beta')))\}\\
\{\exists\alpha',\alpha'',\sigma,\beta.\ \ \alpha'^{\dagger}{\cdot}\alpha''=\alpha\wedge\mathsf{ss}(\alpha',\sigma)\wedge\\
\quad(\mathsf{list}\,\alpha''\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\beta\,(\mathsf{l},\mathbf{nil})\ *\ (Q(\sigma,\beta)\wedge R(\beta)))\}\\
\mathsf{j}:=\mathsf{l}\big)\\
\{\exists\sigma,\beta.\ \ \mathsf{ss}(\alpha^{\dagger},\sigma)\wedge(\mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil})\ *\ (Q(\sigma,\beta)\wedge R(\beta)))\}
\end{array}
$$

Final Assertion

$$\exists \sigma, \beta.\ \ \mathsf{ss}(\alpha^{\dagger}, \sigma) \wedge (\mathsf{list}\, \beta\, (\mathsf{j}, \mathbf{nil})\ *\ (Q(\sigma, \beta) \wedge R(\beta)))$$

Outer Invariant

$$\exists \alpha', \alpha'', \sigma, \beta.\ \ \alpha'^{\dagger}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge \\ (\mathsf{list}\, \alpha''\, (\mathsf{i}, \mathbf{nil})\ *\ \mathsf{list}\, \beta\, (\mathsf{j}, \mathbf{nil})\ *\ (Q(\sigma, \beta) \wedge R(\beta)))$$

Inner Invariant

$$\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge \\ (\mathsf{list}\, \alpha''\, (\mathsf{i}, \mathbf{nil})\ *\ \mathsf{list}\, \gamma\, (\mathsf{l}, \mathsf{j})\ *\ \mathsf{list}\, \beta'\, (\mathsf{j}, \mathsf{m})\ *\ \mathsf{list}\, \beta''\, (\mathsf{m}, \mathbf{nil})\ * \\ (Q(\sigma, \beta'{\cdot}\beta'') \wedge R(\beta'{\cdot}\beta''))\ *\ W(\beta', \gamma, \mathsf{a}))$$

After inner loop:

$$\exists \alpha', \alpha'', \sigma, \beta', \gamma.\ \ \alpha'^{\dagger}{\cdot}\mathsf{a}{\cdot}\alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge \\ (\mathsf{list}\, \alpha''\, (\mathsf{i}, \mathbf{nil})\ *\ \mathsf{list}\, \gamma\, (\mathsf{l}, \mathsf{j})\ *\ \mathsf{list}\, \beta'\, (\mathsf{j}, \mathbf{nil})\ * \\ (Q(\sigma, \beta') \wedge R(\beta'))\ *\ W(\beta', \gamma, \mathsf{a}))$$

Generalizing "Changing Free Variables"

$$\frac{\{p\}\ c\ \{q\}}{(\{p\}\ c\ \{q\})/v_1 \to e_1,\ \ldots\ ,\ v_n \to e_n,}$$

where $v_1, \ldots, v_n$ are the variables occurring free in p, c, or q, and, if v_i is assigned by c, then e_i is a variable that does not occur free in any other e_j.

Examples

In

$$\{\mathsf{x} = \mathsf{y}\}\ \mathsf{x} := \mathsf{x} + \mathsf{y}\ \{\mathsf{x} = 2 \times \mathsf{y}\},$$

one can substitute $\mathsf{x}, \mathsf{y}\ \to\ \mathsf{w}, 2 \times \mathsf{z} - 1$ to infer

$$\{\mathsf{w} = 2 \times \mathsf{z} - 1\}\ \mathsf{w} := \mathsf{w} + 2 \times \mathsf{z} - 1\ \{\mathsf{w} = 2 \times (2 \times z - 1)\}.$$

But one cannot substitute $\mathsf{x}, \mathsf{y}\ \to\ \mathsf{z}, 2 \times \mathsf{z} - 1$ to infer

$$\{\mathsf{z} = 2 \times \mathsf{z} - 1\}\ \mathsf{z} := \mathsf{z} + 2 \times \mathsf{z} - 1\ \{\mathsf{z} = 2 \times (2 \times z - 1)\}.$$

Simple Procedures

A procedure definition has the form

$$h(x_1, \cdots, x_m; y_1, \cdots, y_n) = c,$$

where $y_1, \cdots, y_n$ are the free variables assigned by c. and $x_1, \cdots, x_m$ are the other free variables of c.

If one can prove the specification

$$\{p\}\ c\ \{q\},$$

then one can assume the specification

$$\{p\}\ h(x_1, \cdots, x_m; y_1, \cdots, y_n)\ \{q\}$$

in proving a specification of a command that contains calls of h. (The free variables assigned by the call are $y_1, \cdots, y_n$.)

Recursion

For partial correctness, if c contains recursive calls, one can also use the assumption

$$\{p\}\ h(x_1, \cdots, x_m; y_1, \cdots, y_n)\ \{q\}$$

in proving

$$\{p\}\ c\ \{q\}.$$

An Example

Let

$$W'(\beta, \gamma, \mathsf{a}) \stackrel{\text{def}}{=} \#\gamma = \#\beta \wedge \bigodot_{\mathsf{i}=1}^{\#\gamma} \gamma_{\mathsf{i}} \mapsto \mathsf{a}, \beta_{\mathsf{i}},$$

which satisfies

$$W'(\beta, \gamma, \mathsf{a}) \ * \ \mathsf{g} \mapsto \mathsf{a}, \mathsf{b} \Leftrightarrow W'(\mathsf{b}{\cdot}\beta, \mathsf{g}{\cdot}\gamma, \mathsf{a}).$$

Then the procedure

$$
\begin{array}{l}
\mathsf{extapp}(\mathsf{a}, \mathsf{i}, \mathsf{j}; \mathsf{k}) \ = \\
\quad \mathbf{if}\ \mathsf{i} = \mathbf{nil}\ \mathbf{then}\ \mathsf{k} := \mathsf{j}\ \mathbf{else} \\
\qquad \mathbf{newvar}\ \mathsf{b}, \mathsf{i}', \mathsf{g}\ \mathbf{in} \\
\qquad\quad \big(\mathsf{b} := [\mathsf{i}]\ ;\ \mathsf{i}' := [\mathsf{i} + 1]\ ; \\
\qquad\quad \mathsf{extapp}(\mathsf{a}, \mathsf{i}', \mathsf{j}; \mathsf{k})\ ; \\
\qquad\quad \mathsf{g} := \mathbf{cons}(\mathsf{a}, \mathsf{b})\ ;\ \mathsf{k} := \mathbf{cons}(\mathsf{g}, \mathsf{k})\big)
\end{array}
$$

satisfies

$$
\begin{array}{l}
\{\mathsf{list}\ \beta\ (\mathsf{i}, \mathbf{nil})\} \\
\mathsf{extapp}(\mathsf{a}, \mathsf{i}, \mathsf{j}; \mathsf{k}) \\
\{\exists \gamma.\ \ \mathsf{list}\ \beta\ (\mathsf{i}, \mathbf{nil}) \ * \ \mathsf{list}\ \gamma\ (\mathsf{k}, \mathsf{j}) \ * \ W'(\beta, \gamma, \mathsf{a})\}
\end{array}
$$

since

$$
\begin{array}{l}
\{\mathsf{list}\,\beta\,(\mathsf{i},\mathbf{nil})\}\\
\mathbf{if}\ \mathsf{i}=\mathbf{nil}\ \mathbf{then}\ \mathsf{k}:=\mathsf{j}\ \mathbf{else}\\
\quad\{\exists \mathsf{b},\mathsf{i}',\beta'.\ \ \beta=\mathsf{b}{\cdot}\beta'\wedge(\mathsf{i}\mapsto\mathsf{b},\mathsf{i}'\ *\ \mathsf{list}\,\beta'\,(\mathsf{i}',\mathbf{nil}))\}\\
\quad\mathbf{newvar}\ \mathsf{b},\mathsf{i}',\mathsf{g}\ \mathbf{in}\\
\qquad\big(\mathsf{b}:=[\mathsf{i}]\ ;\ \mathsf{i}':=[\mathsf{i}+1]\ ;\\
\qquad\{\exists\beta'.\ \ \beta=\mathsf{b}{\cdot}\beta'\wedge(\mathsf{i}\mapsto\mathsf{b},\mathsf{i}'\ *\ \mathsf{list}\,\beta'\,(\mathsf{i}',\mathbf{nil}))\}\\
\qquad\{\exists\beta'.\ \ (\beta=\mathsf{b}{\cdot}\beta'\wedge\mathsf{i}\mapsto\mathsf{b},\mathsf{i}')\ *\ \mathsf{list}\,\beta'\,(\mathsf{i}',\mathbf{nil})\}\\
\qquad\mathsf{extapp}(\mathsf{a},\mathsf{i}',\mathsf{j};\mathsf{k})\ ;\\
\qquad\{\exists\beta'.\ \ (\beta=\mathsf{b}{\cdot}\beta'\wedge\mathsf{i}\mapsto\mathsf{b},\mathsf{i}')\ *\\
\qquad\quad(\exists\gamma'.\ \ \mathsf{list}\,\beta'\,(\mathsf{i}',\mathbf{nil})\ *\ \mathsf{list}\,\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\beta',\gamma',\mathsf{a}))\}\\
\qquad\{\exists\beta',\gamma'.\ \ \beta=\mathsf{b}{\cdot}\beta'\wedge\\
\qquad\quad(\mathsf{list}\,\mathsf{b}{\cdot}\beta'\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\beta',\gamma',\mathsf{a}))\}\\
\qquad\mathsf{g}:=\mathbf{cons}(\mathsf{a},\mathsf{b});\\
\qquad\{\exists\beta',\gamma'.\ \ \beta=\mathsf{b}{\cdot}\beta'\wedge\\
\qquad\quad(\mathsf{list}\,\mathsf{b}{\cdot}\beta'\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\mathsf{b}{\cdot}\beta',\mathsf{g}{\cdot}\gamma',\mathsf{a}))\}\\
\qquad\mathsf{k}:=\mathbf{cons}(\mathsf{g},\mathsf{k})\\
\qquad\{\exists\beta',\gamma'.\ \ \beta=\mathsf{b}{\cdot}\beta'\wedge\\
\qquad\quad(\mathsf{list}\,\mathsf{b}{\cdot}\beta'\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\mathsf{g}{\cdot}\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\mathsf{b}{\cdot}\beta',\mathsf{g}{\cdot}\gamma',\mathsf{a}))\}\\
\qquad\big)\\
\{\exists\gamma.\ \ \mathsf{list}\,\beta\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma\,(\mathsf{k},\mathsf{j})\ *\ W'(\beta,\gamma,\mathsf{a})\}
\end{array}
$$

Reasoning about the Call

From the recursion hypothesis:

$$\{\mathsf{list}\ \beta\ (\mathsf{i}, \mathbf{nil})\}$$
$$\mathsf{extapp}(\mathsf{a}, \mathsf{i}, \mathsf{j}; \mathsf{k})$$
$$\{\exists\gamma.\ \ \mathsf{list}\ \beta\ (\mathsf{i}, \mathbf{nil})\ *\ \mathsf{list}\ \gamma\ (\mathsf{k}, \mathsf{j})\ *\ W'(\beta, \gamma, \mathsf{a})\}$$

by changing free variables:

$$\{\mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\}$$
$$\mathsf{extapp}(\mathsf{a}, \mathsf{i}', \mathsf{j}; \mathsf{k})$$
$$\{\exists\gamma.\ \ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\ *\ \mathsf{list}\ \gamma\ (\mathsf{k}, \mathsf{j})\ *\ W'(\beta', \gamma, \mathsf{a})\}$$

by renaming in the postcondition:

$$\{\mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\}$$
$$\mathsf{extapp}(\mathsf{a}, \mathsf{i}', \mathsf{j}; \mathsf{k})$$
$$\{\exists\gamma'.\ \ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\ *\ \mathsf{list}\ \gamma'\ (\mathsf{k}, \mathsf{j})\ *\ W'(\beta', \gamma', \mathsf{a})\}$$

by frame introduction:

$$\{(\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b}, \mathsf{i}')\ *\ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\}$$
$$\mathsf{extapp}(\mathsf{a}, \mathsf{i}', \mathsf{j}; \mathsf{k})\ ;$$
$$\{(\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b}, \mathsf{i}')\ *$$
$$\quad (\exists\gamma'.\ \ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\ *\ \mathsf{list}\ \gamma'\ (\mathsf{k}, \mathsf{j})\ *\ W'(\beta', \gamma', \mathsf{a}))\}$$

by exponential introduction:

$$\{\exists\beta'.\ \ (\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b}, \mathsf{i}')\ *\ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\}$$
$$\mathsf{extapp}(\mathsf{a}, \mathsf{i}', \mathsf{j}; \mathsf{k})\ ;$$
$$\{\exists\beta'.\ \ (\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b}, \mathsf{i}')\ *$$
$$\quad (\exists\gamma'.\ \ \mathsf{list}\ \beta'\ (\mathsf{i}', \mathbf{nil})\ *\ \mathsf{list}\ \gamma'\ (\mathsf{k}, \mathsf{j})\ *\ W'(\beta', \gamma', \mathsf{a}))\}$$

Another Procedure

Let

$$\mathsf{ss}'(\epsilon,\sigma) \stackrel{\text{def}}{=} \sigma = [\epsilon]$$

$$\mathsf{ss}'(\mathsf{a}{\cdot}\alpha,\sigma) \stackrel{\text{def}}{=} \exists\sigma'.\ \ \mathsf{ss}'(\alpha,\sigma') \wedge \sigma = (\mathsf{ext}_{\mathsf{a}}\sigma'){\cdot}\sigma'$$

and, as before,

$$Q(\sigma,\beta) \stackrel{\text{def}}{=} \#\beta = \#\sigma \wedge \forall_{\mathsf{i}=1}^{\#\beta}(\mathsf{list}\ \sigma_{\mathsf{i}}\,(\beta_{\mathsf{i}},\mathbf{nil})\ *\ \mathbf{true})$$

$$R(\beta) \stackrel{\text{def}}{=} (\beta_{\#\beta} = \mathbf{nil} \wedge \mathbf{emptyheap})\ *\ \bigodot_{\mathsf{i}=1}^{\#\beta-1}(\exists\mathsf{a},\mathsf{k}.\ \ \mathsf{i} < \mathsf{k} \leq \#\beta \wedge \beta_{\mathsf{i}} \mapsto \mathsf{a},\beta_{\mathsf{k}})$$

$$\#\mathsf{ext}_{\mathsf{a}}\sigma \stackrel{\text{def}}{=} \#\sigma$$

$$(\mathsf{ext}_{\mathsf{a}}\sigma)_i \stackrel{\text{def}}{=} \mathsf{a}{\cdot}\sigma_i,$$

which satisfy

$$Q([\epsilon],\mathbf{nil}) \Leftrightarrow \mathbf{true}$$

$$R(\mathbf{nil}) \Leftrightarrow \mathbf{emptyheap}$$

$$Q(\sigma,\beta)\ *\ W'(\beta,\gamma,\mathsf{a}) \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma){\cdot}\sigma,\gamma{\cdot}\beta)$$

$$R(\beta)\ *\ W'(\beta,\gamma,\mathsf{a}) \Rightarrow R(\gamma{\cdot}\beta)$$

$$(Q(\sigma,\beta) \wedge R(\beta))\ *\ W'(\beta,\gamma,\mathsf{a}) \Rightarrow Q((\mathsf{ext}_{\mathsf{a}}\sigma){\cdot}\sigma,\gamma{\cdot}\beta) \wedge R(\gamma{\cdot}\beta).$$

Then the procedure

$$
\begin{array}{l}
\mathsf{subsets}(\mathsf{i};\mathsf{j}) = \\
\quad \textbf{if } \mathsf{i} = \textbf{nil} \textbf{ then } \mathsf{j} := \textbf{cons}(\textbf{nil}, \textbf{nil}) \textbf{ else} \\
\quad\quad \textbf{newvar } \mathsf{a}, \mathsf{i}', \mathsf{j}' \textbf{ in} \\
\quad\quad\quad \big(\mathsf{a} := [\mathsf{i}] \,;\, \mathsf{i}' := [\mathsf{i} + 1] \,; \\
\quad\quad\quad \mathsf{subsets}(\mathsf{i}';\mathsf{j}') \,; \\
\quad\quad\quad \mathsf{extapp}(\mathsf{a}, \mathsf{j}', \mathsf{j}'; \mathsf{j})\big)
\end{array}
$$

satisfies

$$
\begin{array}{l}
\{\mathsf{list}\ \alpha\,(\mathsf{i}, \textbf{nil})\} \\
\mathsf{subsets}(\mathsf{i};\mathsf{j}) \\
\{\exists \sigma, \beta.\ \ \mathsf{ss}'(\alpha, \sigma) \wedge \\
\quad \big(\mathsf{list}\ \alpha\,(\mathsf{i}, \textbf{nil}) \ *\ \mathsf{list}\ \beta\,(\mathsf{j}, \textbf{nil}) \ *\ (Q(\sigma, \beta) \wedge R(\beta))\big)\}
\end{array}
$$

since

$$
\begin{array}{l}
\{\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil})\}\\
\mathbf{if}\ \mathsf{i}=\mathbf{nil}\ \mathbf{then}\ \mathsf{j}:=\mathbf{cons}(\mathbf{nil},\mathbf{nil})\ \mathbf{else}\\
\quad\{\exists \mathsf{a},\mathsf{i}',\alpha'.\ \ \alpha=\mathsf{a}{\cdot}\alpha' \wedge (\mathsf{i}\mapsto \mathsf{a},\mathsf{i}' \ *\ \mathsf{list}\,\alpha'\,(\mathsf{i}',\mathbf{nil}))\}\\
\quad\mathbf{newvar}\ \mathsf{a},\mathsf{i}',\mathsf{j}'\ \mathbf{in}\\
\qquad\Big(\mathsf{a}:=[\mathsf{i}]\ ;\ \mathsf{i}':=[\mathsf{i}+1]\ ;\\
\qquad\{\exists\alpha'.\ \ \alpha=\mathsf{a}{\cdot}\alpha' \wedge (\mathsf{i}\mapsto \mathsf{a},\mathsf{i}' \ *\ \mathsf{list}\,\alpha'\,(\mathsf{i}',\mathbf{nil}))\}\\
\qquad\{\exists\alpha'.\ \ (\alpha=\mathsf{a}{\cdot}\alpha' \wedge \mathsf{i}\mapsto \mathsf{a},\mathsf{i}') \ *\ \mathsf{list}\,\alpha'\,(\mathsf{i}',\mathbf{nil})\}\\
\qquad\mathsf{subsets}(\mathsf{i}';\mathsf{j}')\ ;\\
\qquad\{\exists\alpha'.\ \ (\alpha=\mathsf{a}{\cdot}\alpha' \wedge \mathsf{i}\mapsto \mathsf{a},\mathsf{i}') \ *\\
\qquad\quad\Big(\exists\sigma',\beta'.\ \ \mathsf{ss}'(\alpha',\sigma')\wedge\Big(\mathsf{list}\,\alpha'\,(\mathsf{i}',\mathbf{nil})\ *\\
\qquad\qquad\mathsf{list}\,\beta'\,(\mathsf{j}',\mathbf{nil})\ *\ (Q(\sigma',\beta')\wedge R(\beta'))\Big)\Big)\}\\
\qquad\{\exists\alpha',\sigma',\beta'.\ \ \Big(\alpha=\mathsf{a}{\cdot}\alpha'\wedge\mathsf{ss}'(\alpha',\sigma')\wedge\\
\qquad\qquad(\mathsf{list}\,\mathsf{a}{\cdot}\alpha'\,(\mathsf{i},\mathbf{nil})\ *\ (Q(\sigma',\beta')\wedge R(\beta')))\Big)\ *\\
\qquad\quad\mathsf{list}\,\beta'\,(\mathsf{j}',\mathbf{nil})\}\\
\qquad\mathsf{extapp}(\mathsf{a},\mathsf{j}',\mathsf{j}';\mathsf{j})\\
\qquad\{\exists\alpha',\sigma',\beta'.\ \ \Big(\alpha=\mathsf{a}{\cdot}\alpha'\wedge\mathsf{ss}'(\alpha',\sigma')\wedge\\
\qquad\qquad(\mathsf{list}\,\mathsf{a}{\cdot}\alpha'\,(\mathsf{i},\mathbf{nil})\ *\ (Q(\sigma',\beta')\wedge R(\beta')))\Big)\ *\\
\qquad\quad(\exists\gamma.\ \ \mathsf{list}\,\beta'\,(\mathsf{j}',\mathbf{nil})\ *\ \mathsf{list}\,\gamma\,(\mathsf{j},\mathsf{j}')\ *\ W'(\beta',\gamma,\mathsf{a}))\}\\
\qquad\{\exists\alpha',\sigma',\beta',\gamma.\ \ \alpha=\mathsf{a}{\cdot}\alpha'\wedge\mathsf{ss}'(\mathsf{a}{\cdot}\alpha',(\mathsf{ext}_{\mathsf{a}}\sigma'){\cdot}\sigma')\wedge\\
\qquad\quad\Big(\mathsf{list}\,\mathsf{a}{\cdot}\alpha'\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\gamma{\cdot}\beta'\,(\mathsf{j},\mathbf{nil})\ *\\
\qquad\qquad(Q((\mathsf{ext}_{\mathsf{a}}\sigma'){\cdot}\sigma',\gamma{\cdot}\beta')\wedge R(\gamma{\cdot}\beta'))\Big)\}\Big)\\
\{\exists\sigma,\beta.\ \ \mathsf{ss}'(\alpha,\sigma)\wedge\\
\quad\Big(\mathsf{list}\,\alpha\,(\mathsf{i},\mathbf{nil})\ *\ \mathsf{list}\,\beta\,(\mathsf{j},\mathbf{nil})\ *\ (Q(\sigma,\beta)\wedge R(\beta))\Big)\}
\end{array}
$$