Class Notes for CS 818A3 - Spring 2005

AN INTRODUCTION TO SEPARATION LOGIC

5. Iterated Separating Conjunction

John C. Reynolds
Department of Computer Science
Carnegie Mellon University

February 18, 2005

©2005 John C. Reynolds

Iterated Separating Conjunction

$$\langle \text{assert} \rangle ::= \cdots \mid \bigodot_{\langle \text{var} \rangle = \langle \text{exp} \rangle}^{\langle \text{exp} \rangle} \langle \text{assert} \rangle$$

$s, h \vDash \bigodot_{v=e}^{e'} p$ iff

$\quad$ **let** $m = [\![e]\!]_{\text{exp}} s,\ n = [\![e']\!]_{\text{exp}} s,\ I = \{\, i \mid m \leq i \leq n \,\}$ **in**

$\quad\quad \exists H \in I \to \text{Heaps}.$

$\quad\quad\quad \forall i, j \in I.\ i \neq j$ implies $Hi \perp Hj$

$\quad\quad\quad$ and $h = \bigcup \{\, Hi \mid i \in I \,\}$

$\quad\quad\quad$ and $\forall i \in I.\ [\, s \mid v{:}\, i \,], Hi \vDash p.$

Axiom Schemata

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

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

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

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

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

when q is pure and $i \notin \text{FV}(q)$

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

Array Allocation

$$\langle \text{comm} \rangle ::= \cdots \mid \langle \text{var} \rangle := \mathbf{allocate}\ \langle \text{exp} \rangle$$

Semantics

$$\frac{}{(s,h)\,[\![v := \mathbf{allocate}\ e]\!]_{\text{comm}}\,([\, s \mid v{:}\,\ell \,], h \cdot h')},$$

where $h \perp h'$ and $\operatorname{dom} h' = \{\, i \mid \ell \leq i < \ell + [\![e]\!]_{\text{exp}} s \,\}$.

Inference Rules

Noninterfering:

$$\frac{}{\{r\}\ v := \mathbf{allocate}\ e\ \{(\bigodot_{i=v}^{v+e-1} i \mapsto -)\ *\ r\}},$$

where v does not occur free in r or e.

The local form:

$$\frac{}{\{v = v' \wedge \mathbf{emp}\}\ v := \mathbf{allocate}\ e\ \{\bigodot_{i=v}^{v+e'-1} i \mapsto -\}},$$

where v' is distinct from v, and e' denotes $e/v \to v'$.

The global form:

$$\frac{}{\{r\}\ v := \mathbf{allocate}\ e\ \{\exists v'.\ (\bigodot_{i=v}^{v+e'-1} i \mapsto -)\ *\ r'\}},$$

where v' is distinct from v, $v' \notin \mathrm{FV}(e, r)$, e' denotes $e/v \to v'$, and r' denotes $r/v \to v'$.

The backward-reasoning form:

$$\frac{}{\{\forall v''.\ (\bigodot_{i=v''}^{v''+e-1} i \mapsto -) \mathrel{-\!\!*} p''\}\ v := \mathbf{allocate}\ e\ \{p\}},$$

where v'' is distinct from v, $v'' \notin \mathrm{FV}(e, p)$, and p'' denotes $p/v \to v''$.

A Cyclic Buffer Using an Array

We assume that an n-element array has been allocated at location l. Let $\phi(x)$ be the unique integer such that

$$\phi(x) \equiv_{\mathrm{mod}\ \mathsf{n}} x \qquad\qquad \mathsf{l} \le \phi(x) < \mathsf{l} + \mathsf{n}.$$

We will use the variables

m	number of active elements
i	pointer to first active element
j	pointer to first inactive element

Let R abbreviate the assertion

$$R \stackrel{\mathrm{def}}{=} 0 \le \mathsf{m} \le \mathsf{n} \wedge \mathsf{l} \le \mathsf{i} < \mathsf{l} + \mathsf{n} \wedge \mathsf{l} \le \mathsf{j} < \mathsf{l} + \mathsf{n} \wedge \mathsf{j} \equiv_{\mathrm{mod}\ \mathsf{n}} \mathsf{i} + \mathsf{m}$$

It is easy to show that

$$\{R \wedge \mathsf{m} < \mathsf{n}\}$$
$$\mathsf{m} := \mathsf{m} + 1\ ;\ \textbf{if}\ \mathsf{j} = \mathsf{l} + \mathsf{n} - 1\ \textbf{then}\ \mathsf{j} := \mathsf{l}\ \textbf{else}\ \mathsf{j} := \mathsf{j} + 1$$
$$\{R\}$$

and

$$\{R \wedge \mathsf{m} > 0\}$$
$$\mathsf{m} := \mathsf{m} - 1\ ;\ \textbf{if}\ \mathsf{i} = \mathsf{l} + \mathsf{n} - 1\ \textbf{then}\ \mathsf{i} := \mathsf{l}\ \textbf{else}\ \mathsf{i} := \mathsf{i} + 1$$
$$\{R\}$$

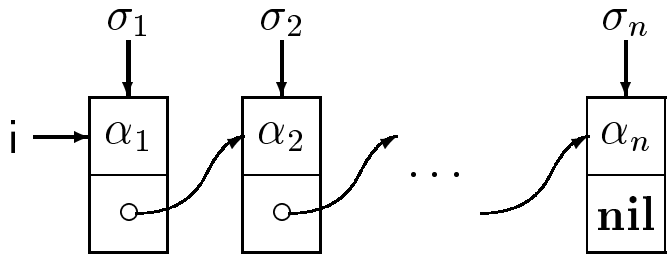
When the buffer contains a sequence α, it should satisfy

$$((\bigodot_{\mathsf{k}=0}^{\mathsf{m}-1} \phi(\mathsf{i}+\mathsf{k}) \mapsto \alpha_{\mathsf{k}+1}) \ *\ (\bigodot_{\mathsf{k}=0}^{\mathsf{n}-\mathsf{m}-1} \phi(\mathsf{j}+\mathsf{k}) \mapsto -)) \wedge \mathsf{m} = \#\alpha \wedge R.$$

Inserting an Element

$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto \alpha_{k+1}) \ * \ (\bigodot_{k=0}^{n-m-1} \phi(j+k) \mapsto -))$$
$$\quad \wedge\, m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto \alpha_{k+1}) \ * \ (\bigodot_{k=0}^{0} \phi(j+k) \mapsto -) \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto \alpha_{k+1}) \ * \ \phi(j) \mapsto - \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$[j] := x\ ;$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto \alpha_{k+1}) \ * \ \phi(j) \mapsto x \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto \alpha_{k+1}) \ * \ \phi(i+m) \mapsto x \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ * \ \phi(i+m) \mapsto (\alpha{\cdot}x)_{m+1} \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ * \ (\bigodot_{k=m}^{m} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ *$$
$$\quad (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -)) \wedge m = \#\alpha \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ * \ (\bigodot_{k=1}^{n-m-1} \phi(j+k) \mapsto -))$$
$$\quad \wedge\, m+1 = \#(\alpha{\cdot}x) \wedge R \wedge m < n\}$$
$$\{((\bigodot_{k=0}^{m} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ * \ (\bigodot_{k=0}^{n-m-2} \phi(j+k+1) \mapsto -))$$
$$\quad \wedge\, m+1 = \#(\alpha{\cdot}x) \wedge R \wedge m < n\}$$
$$m := m+1\ ;\ \textbf{if}\ j = l+n-1\ \textbf{then}\ j := l\ \textbf{else}\ j := j+1$$
$$\{((\bigodot_{k=0}^{m-1} \phi(i+k) \mapsto (\alpha{\cdot}x)_{k+1}) \ * \ (\bigodot_{k=0}^{n-m-1} \phi(j+k) \mapsto -))$$
$$\quad \wedge\, m = \#(\alpha{\cdot}x) \wedge R\}$$

Connecting Two Views of Lists



If

$$\mathsf{list}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathbf{emp} \wedge \mathsf{i} = \mathbf{nil}$$

$$\mathsf{list}\ (\mathsf{a}{\cdot}\alpha)\ \mathsf{i} \stackrel{\text{def}}{=} \exists \mathsf{j}.\ \mathsf{i} \mapsto \mathsf{a}, \mathsf{j}\ *\ \mathsf{list}\ \alpha\ \mathsf{j}$$

and

$$\mathsf{listN}\ \epsilon\ \mathsf{i} \stackrel{\text{def}}{=} \mathbf{emp} \wedge \mathsf{i} = \mathbf{nil}$$

$$\mathsf{listN}\ (\mathsf{b}{\cdot}\sigma)\ \mathsf{i} \stackrel{\text{def}}{=} \mathsf{b} = \mathsf{i} \wedge \exists \mathsf{j}.\ \mathsf{i}+1 \mapsto \mathsf{j}\ *\ \mathsf{listN}\ \sigma\ \mathsf{j},$$

then

$$\mathsf{list}\ \alpha\ \mathsf{i} \Leftrightarrow \exists \sigma.\ \#\sigma = \#\alpha \wedge (\mathsf{listN}\ \sigma\ \mathsf{i}\ *\ \bigodot_{\mathsf{k}=1}^{\#\alpha} \sigma_k \mapsto \alpha_k).$$

The proof is by induction on α.

Specifying Subset Lists

Let $\alpha, \beta \in \mathbf{Z}^*$ and $\sigma \in (\mathbf{Z}^*)^*$ in

$\{\mathsf{list}\,\alpha\,\mathsf{i}\}$

"Set j to list of lists of subsets of i"

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

where

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

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

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

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

Some Properties

The predicates

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

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

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

satisfy

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

$$R([\mathbf{nil}]) \Leftrightarrow \mathbf{emp}$$

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

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

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

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

Proofs (1)

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

Proofs (2)

Let

$$p(\mathsf{i}) \stackrel{\text{def}}{=} \mathsf{list}\ \sigma_{\mathsf{i}}\ \beta_{\mathsf{i}}$$
$$q(\mathsf{i}) \stackrel{\text{def}}{=} \gamma_{\mathsf{i}} \mapsto \mathsf{a}, (\beta^{\dagger})_{\mathsf{i}}$$
$$n \stackrel{\text{def}}{=} \#\sigma.$$

Then

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

Proofs (3)

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

From (2) and (3), we have

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

An Iterative Program for Subset Lists

Final Assertion

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

Outer Invariant

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

Inner Invariant

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

After inner loop:

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

Iterative Subset Lists: The Main Program

$\{\mathsf{list}\,\alpha\,\mathsf{i}\}$

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

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

$\{\mathsf{list}\,\alpha\,\mathsf{i} \;*\; \mathsf{list}\,[\mathbf{nil}]\,\mathsf{j}\}$

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

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

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

while $\mathsf{i} \neq \mathbf{nil}$ **do** *Body of the Outer Loop*

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

Iterative Subset Lists: Body of the Outer Loop

$\{\exists \alpha', \alpha'', \sigma, \beta, \mathsf{a}, \mathsf{k}.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{i} \mapsto \mathsf{a}, \mathsf{k} \ * \ \mathsf{list}\, \alpha''\, \mathsf{k} \ * \ \mathsf{list}\, \beta\, \mathsf{j} \ * \ (Q(\sigma, \beta) \wedge R(\beta)))\}$
$\big(\mathsf{a} := [\mathsf{i}]\ ;\ \mathsf{k} := [\mathsf{i}+1]\ ;\ \mathbf{dispose}\ \mathsf{i}\ ;\ \mathbf{dispose}\ \mathsf{i}+1\ ;\ \mathsf{i} := \mathsf{k}\ ;$
$\{\exists \alpha', \alpha'', \sigma, \beta.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{list}\, \beta\, \mathsf{j} \ * \ (Q(\sigma, \beta) \wedge R(\beta)))\}$
$\{\exists \alpha', \alpha'', \sigma, \beta.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{lseg}\, \epsilon\, (\mathsf{j}, \mathsf{j}) \ * \ \mathsf{lseg}\, \epsilon\, (\mathsf{j}, \mathsf{j}) \ * \ \mathsf{list}\, \beta\, \mathsf{j} \ *$
$\qquad (Q(\sigma, \epsilon \cdot \beta) \wedge R(\epsilon \cdot \beta)) \ * \ W(\epsilon, \epsilon, \mathsf{a}))\}$
$\mathsf{l} := \mathsf{j}\ ;\ \mathsf{m} := \mathsf{j}\ ;$
$\{\exists \alpha', \alpha'', \sigma, \beta.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{lseg}\, \epsilon\, (\mathsf{l}, \mathsf{j}) \ * \ \mathsf{lseg}\, \epsilon\, (\mathsf{j}, \mathsf{m}) \ * \ \mathsf{list}\, \beta\, \mathsf{m} \ *$
$\qquad (Q(\sigma, \epsilon \cdot \beta) \wedge R(\epsilon \cdot \beta)) \ * \ W(\epsilon, \epsilon, \mathsf{a}))\}$
$\{\exists \alpha', \alpha'', \sigma, \beta', \beta'', \gamma.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ * \ \mathsf{lseg}\, \beta'\, (\mathsf{j}, \mathsf{m}) \ * \ \mathsf{list}\, \beta''\, \mathsf{m} \ *$
$\qquad (Q(\sigma, \beta' \cdot \beta'') \wedge R(\beta' \cdot \beta'')) \ * \ W(\beta', \gamma, \mathsf{a}))\}$
$\mathbf{while}\ \mathsf{m} \neq \mathbf{nil}\ \mathbf{do}$ *Body of the Inner Loop*
$\{\exists \alpha', \alpha'', \sigma, \beta', \gamma.\ \ \alpha'^{\dagger} \cdot \mathsf{a} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ * \ \mathsf{list}\, \beta'\, \mathsf{j} \ *$
$\qquad (Q(\sigma, \beta') \wedge R(\beta')) \ * \ W(\beta', \gamma, \mathsf{a}))\}$
$\{\exists \alpha', \alpha'', \sigma, \beta', \gamma.\ \ (\mathsf{a} \cdot \alpha')^{\dagger} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\mathsf{a} \cdot \alpha', (\mathsf{ext}_{\mathsf{a}} \sigma)^{\dagger} \cdot \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{list}\, (\gamma \cdot \beta')\, \mathsf{l} \ * \ (Q((\mathsf{ext}_{\mathsf{a}} \sigma)^{\dagger} \cdot \sigma, \gamma \cdot \beta') \wedge R(\gamma \cdot \beta')))\}$
$\{\exists \alpha', \alpha'', \sigma, \beta.\ \ \alpha'^{\dagger} \cdot \alpha'' = \alpha \wedge \mathsf{ss}(\alpha', \sigma) \wedge$
$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ * \ \mathsf{list}\, \beta\, \mathsf{l} \ * \ (Q(\sigma, \beta) \wedge R(\beta)))\}$
$\mathsf{j} := \mathsf{l}\big)$

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta'\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{m} \mapsto \mathsf{b}, \mathsf{m}'' \ *$

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

$\big(\mathsf{b} := [\mathsf{m}]\ ;$

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta'\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{m} \mapsto \mathsf{b}, \mathsf{m}'' \ *$

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

$\mathsf{m} := [\mathsf{m} + 1]\ ;$

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta'\, (\mathsf{j}, \mathsf{m}') \ *\ \mathsf{m}' \mapsto \mathsf{b}, \mathsf{m} \ *$

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

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta' \cdot \mathsf{b}\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{list}\, \beta''\, \mathsf{m} \ *$

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

$\mathsf{g} := \mathbf{cons}(\mathsf{a}, \mathsf{b})\ ;$

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta' \cdot \mathsf{b}\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{list}\, \beta''\, \mathsf{m} \ *$

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

$\mathsf{l} := \mathbf{cons}(\mathsf{g}, \mathsf{l})$

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{l} \mapsto \mathsf{g}, \mathsf{l}' \ *\ \mathsf{lseg}\, \gamma\, (\mathsf{l}', \mathsf{j}) \ *\ \mathsf{lseg}\, \beta' \cdot \mathsf{b}\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{list}\, \beta''\, \mathsf{m} \ *$

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

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

$\quad (\mathsf{list}\, \alpha''\, \mathsf{i} \ *\ \mathsf{lseg}\, \mathsf{g} \cdot \gamma\, (\mathsf{l}, \mathsf{j}) \ *\ \mathsf{lseg}\, \beta' \cdot \mathsf{b}\, (\mathsf{j}, \mathsf{m}) \ *\ \mathsf{list}\, \beta''\, \mathsf{m} \ *$

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

A Recursive Program for Subset Lists

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

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

satisfies

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

since

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

Reasoning about the Call

From the recursion assumption:

$\{\mathsf{list}\,\beta\,\mathsf{i}\}$

$\mathsf{extapp}(\mathsf{a},\mathsf{i},\mathsf{j};\mathsf{k})$

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

by free variable substitution (FVS):

$\{\mathsf{list}\,\beta'\,\mathsf{i}'\}$

$\mathsf{extapp}(\mathsf{a},\mathsf{i}',\mathsf{j};\mathsf{k})$

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

by renaming (RN) in the postcondition:

$\{\mathsf{list}\,\beta'\,\mathsf{i}'\}$

$\mathsf{extapp}(\mathsf{a},\mathsf{i}',\mathsf{j};\mathsf{k})$

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

by the frame rule (FR):

$\{(\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{i}')\ *\ \mathsf{list}\,\beta'\,\mathsf{i}'\}$

$\mathsf{extapp}(\mathsf{a},\mathsf{i}',\mathsf{j};\mathsf{k})$;

$\{(\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{i}')\ *$

$\quad(\exists\gamma'.\ \ \mathsf{list}\,\beta'\,\mathsf{i}'\ *\ \mathsf{lseg}\,\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\beta',\gamma',\mathsf{a}))\}$

by introducing existential quantification (EQ):

$\{\exists\beta'.\ \ (\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{i}')\ *\ \mathsf{list}\,\beta'\,\mathsf{i}'\}$

$\mathsf{extapp}(\mathsf{a},\mathsf{i}',\mathsf{j};\mathsf{k})$;

$\{\exists\beta'.\ \ (\beta = \mathsf{b}{\cdot}\beta' \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{i}')\ *$

$\quad(\exists\gamma'.\ \ \mathsf{list}\,\beta'\,\mathsf{i}'\ *\ \mathsf{lseg}\,\gamma'\,(\mathsf{k},\mathsf{j})\ *\ W'(\beta',\gamma',\mathsf{a}))\}$

Another Recursive 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,

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

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

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

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

which satisfy

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

$$R([\mathbf{nil}]) \Leftrightarrow \mathbf{emp}$$

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

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

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

Then the procedure

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

satisfies

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

since

$\{\mathsf{list}\ \alpha\ \mathsf{i}\}$

$\mathbf{if}\ \mathsf{i} = \mathbf{nil}\ \mathbf{then}\ \mathsf{j} := \mathbf{cons}(\mathbf{nil}, \mathbf{nil})\ \mathbf{else}$

$\{\exists \mathsf{a}, \mathsf{i}', \alpha'.\ \ \alpha = \mathsf{a}{\cdot}\alpha' \wedge (\mathsf{i} \mapsto \mathsf{a}, \mathsf{i}'\ *\ \mathsf{list}\ \alpha'\ \mathsf{i}')\}$

$\mathbf{newvar}\ \mathsf{a}, \mathsf{i}', \mathsf{j}'\ \mathbf{in}$

$\Big(\mathsf{a} := [\mathsf{i}]\ ;\ \mathsf{i}' := [\mathsf{i}+1]\ ;$

$\{\exists \alpha'.\ \ \alpha = \mathsf{a}{\cdot}\alpha' \wedge (\mathsf{i} \mapsto \mathsf{a}, \mathsf{i}'\ *\ \mathsf{list}\ \alpha'\ \mathsf{i}')\}$

$\{\exists \alpha'.\ \ (\alpha = \mathsf{a}{\cdot}\alpha' \wedge \mathsf{i} \mapsto \mathsf{a}, \mathsf{i}')\ *\ \mathsf{list}\ \alpha'\ \mathsf{i}'\}$

$\mathsf{subsets}(\mathsf{i}'; \mathsf{j}')\ ;$

$\{\exists \alpha'.\ \ (\alpha = \mathsf{a}{\cdot}\alpha' \wedge \mathsf{i} \mapsto \mathsf{a}, \mathsf{i}')\ *$

$\Big(\exists \sigma', \beta'.\ \ \mathsf{ss}'(\alpha', \sigma') \wedge \Big(\mathsf{list}\ \alpha'\ \mathsf{i}'\ *$

$\mathsf{list}\ \beta'\ \mathsf{j}'\ *\ (Q(\sigma', \beta') \wedge R(\beta'))\Big)\Big)\}$

$\{\exists \alpha', \sigma', \beta'.\ \ \Big(\alpha = \mathsf{a}{\cdot}\alpha' \wedge \mathsf{ss}'(\alpha', \sigma') \wedge$

$(\mathsf{list}\ (\mathsf{a}{\cdot}\alpha')\ \mathsf{i}\ *\ (Q(\sigma', \beta') \wedge R(\beta')))\Big)\ *$

$\mathsf{list}\ \beta'\ \mathsf{j}'\}$

$\mathsf{extapp}(\mathsf{a}, \mathsf{j}', \mathsf{j}'; \mathsf{j})$

$\{\exists \alpha', \sigma', \beta'.\ \ \Big(\alpha = \mathsf{a}{\cdot}\alpha' \wedge \mathsf{ss}'(\alpha', \sigma') \wedge$

$(\mathsf{list}\ (\mathsf{a}{\cdot}\alpha')\ \mathsf{i}\ *\ (Q(\sigma', \beta') \wedge R(\beta')))\Big)\ *$

$(\exists \gamma.\ \ \mathsf{list}\ \beta'\ \mathsf{j}'\ *\ \mathsf{lseg}\ \gamma\ (\mathsf{j}, \mathsf{j}')\ *\ W'(\beta', \gamma, \mathsf{a}))\}$

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

$\Big(\mathsf{list}\ (\mathsf{a}{\cdot}\alpha')\ \mathsf{i}\ *\ \mathsf{list}\ (\gamma{\cdot}\beta')\ \mathsf{j}\ *$

$(Q((\mathsf{ext}_{\mathsf{a}}\sigma'){\cdot}\sigma', \gamma{\cdot}\beta') \wedge R(\gamma{\cdot}\beta'))\Big)\}\Big)$

$\{\exists \sigma, \beta.\ \ \mathsf{ss}'(\alpha, \sigma) \wedge \Big(\mathsf{list}\ \alpha\ \mathsf{i}\ *\ \mathsf{list}\ \beta\ \mathsf{j}\ *\ (Q(\sigma, \beta) \wedge R(\beta))\Big)\}$

Arrays that Denote Sequences

$$\mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{b}) \stackrel{\text{def}}{=} \#\alpha = \mathsf{b} - \mathsf{a} + 1 \wedge \bigodot_{\mathsf{i}=\mathsf{a}}^{\mathsf{b}} \mathsf{i} \mapsto \alpha_{\mathsf{i}-\mathsf{a}+1}.$$

Properties

$$\mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{b}) \Rightarrow \#\alpha = \mathsf{b} - \mathsf{a} + 1$$

$$\mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{b}) \Rightarrow \mathsf{i} \hookrightarrow \alpha_{\mathsf{i}-\mathsf{a}+1} \quad \text{when } \mathsf{a} \leq \mathsf{i} \leq \mathsf{b}$$

$$\mathsf{array}\ \epsilon\ (\mathsf{a}, \mathsf{b}) \Leftrightarrow \mathsf{b} = \mathsf{a} - 1 \wedge \mathbf{emp}$$

$$\mathsf{array}\ \mathsf{x}\ (\mathsf{a}, \mathsf{b}) \Leftrightarrow \mathsf{b} = \mathsf{a} \wedge \mathsf{a} \mapsto \mathsf{x}$$

$$\mathsf{array}\ \mathsf{x}{\cdot}\alpha\ (\mathsf{a}, \mathsf{b}) \Leftrightarrow \mathsf{a} \mapsto \mathsf{x} \ * \ \mathsf{array}\ \alpha\ (\mathsf{a}+1, \mathsf{b})$$

$$\mathsf{array}\ \alpha{\cdot}\mathsf{x}\ (\mathsf{a}, \mathsf{b}) \Leftrightarrow \mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{b}-1) \ * \ \mathsf{b} \mapsto \mathsf{x}$$

$$\begin{aligned}\mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{c}) \ * \ \mathsf{array}\ \beta\ (\mathsf{c}+1, \mathsf{b}) \\ \Leftrightarrow \mathsf{array}\ \alpha{\cdot}\beta\ (\mathsf{a}, \mathsf{b}) \wedge \mathsf{c} = \mathsf{a} + \#\alpha - 1 \\ \Leftrightarrow \mathsf{array}\ \alpha{\cdot}\beta\ (\mathsf{a}, \mathsf{b}) \wedge \mathsf{c} = \mathsf{b} - \#\beta\end{aligned}$$

A Subtlety

In the following annotated specification, note that the assertion following the second **else** depends upon

$$\mathsf{c}+1 \hookrightarrow \mathsf{x} \wedge \mathsf{d}-1 \hookrightarrow \mathsf{y} \wedge \mathsf{x} > \mathsf{r} \wedge \mathsf{y} \leq r \Rightarrow \mathsf{c}+1 \neq \mathsf{d}-1.$$

Partition

$\{\mathsf{array}\ \alpha(\mathsf{a},\mathsf{b})\}$

$\mathbf{newvar}\ \mathsf{d},\mathsf{x},\mathsf{y}\ \mathbf{in}\ \big(\mathsf{c} := \mathsf{a}-1\ ;\ \mathsf{d} := \mathsf{b}+1\ ;$

$\{\exists\alpha_1,\alpha_2,\alpha_3.\ (\mathsf{array}\ \alpha_1\ (\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+1,\mathsf{d}-1)\ *\ \mathsf{array}\ \alpha_3\ (\mathsf{d},\mathsf{b}))$
$\quad \wedge\ \alpha_1{\cdot}\alpha_2{\cdot}\alpha_3 \sim \alpha \wedge \{\alpha_1\} \leq^* \mathsf{r} \wedge \{\alpha_3\} >^* \mathsf{r}\}$

$\mathbf{while}\ \mathsf{d} > \mathsf{c}+1\ \mathbf{do}\ \big(\mathsf{x} := [\mathsf{c}+1];$

$\quad \mathbf{if}\ \mathsf{x} \leq \mathsf{r}\ \mathbf{then}$

$\quad\quad \{\exists\alpha_1,\alpha_2,\alpha_3.\ (\mathsf{array}\ \alpha_1\ (\mathsf{a},\mathsf{c})\ *\ \mathsf{c}+1 \mapsto \mathsf{x}\ *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+2,\mathsf{d}-1)$
$\quad\quad\quad *\ \mathsf{array}\ \alpha_3\ (\mathsf{d},\mathsf{b})) \wedge \alpha_1{\cdot}\mathsf{x}{\cdot}\alpha_2{\cdot}\alpha_3 \sim \alpha \wedge \{\alpha_1{\cdot}\mathsf{x}\} \leq^* \mathsf{r} \wedge \{\alpha_3\} >^* \mathsf{r}\}$

$\quad\quad \mathsf{c} := \mathsf{c}+1$

$\quad \mathbf{else}\ \big(\mathsf{y} := [\mathsf{d}-1];$

$\quad\quad \mathbf{if}\ \mathsf{y} > \mathsf{r}\ \mathbf{then}$

$\quad\quad\quad \{\exists\alpha_1,\alpha_2,\alpha_3.\ (\mathsf{array}\ \alpha_1\ (\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+1,\mathsf{d}-2)\ *\ \mathsf{d}-1 \mapsto \mathsf{y}$
$\quad\quad\quad\quad *\ \mathsf{array}\ \alpha_3\ (\mathsf{d},\mathsf{b})) \wedge \alpha_1{\cdot}\alpha_2{\cdot}\mathsf{y}{\cdot}\alpha_3 \sim \alpha \wedge \{\alpha_1\} \leq^* \mathsf{r} \wedge \{\mathsf{y}{\cdot}\alpha_3\} >^* \mathsf{r}\}$

$\quad\quad\quad \mathsf{d} := \mathsf{d}-1$

$\quad\quad\quad \mathbf{else}$

$\quad\quad\quad\quad \{\exists\alpha_1,\alpha_2,\alpha_3.\ (\mathsf{array}\ \alpha_1\ (\mathsf{a},\mathsf{c})\ *\ \mathsf{c}+1 \mapsto \mathsf{x}$
$\quad\quad\quad\quad\quad *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+2,\mathsf{d}-2)\ *\ \mathsf{d}-1 \mapsto \mathsf{y}\ *\ \mathsf{array}\ \alpha_3\ (\mathsf{d},\mathsf{b}))$
$\quad\quad\quad\quad\quad \wedge\ \alpha_1{\cdot}\mathsf{x}{\cdot}\alpha_2{\cdot}\mathsf{y}{\cdot}\alpha_3 \sim \alpha \wedge \{\alpha_1\} \leq^* \mathsf{r} \wedge \{\alpha_3\} >^* \mathsf{r} \wedge \mathsf{x} > \mathsf{r} \wedge \mathsf{y} \leq \mathsf{r}\}$

$\quad\quad\quad ([\mathsf{c}+1] := \mathsf{y}\ ;\ [\mathsf{d}-1] := \mathsf{x}\ ;\ \mathsf{c} := \mathsf{c}+1\ ;\ \mathsf{d} := \mathsf{d}-1)\big)\big)\big)$

$\{\exists\alpha_1,\alpha_2.\ (\mathsf{array}\ \alpha_1(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2(\mathsf{c}+1,\mathsf{b}))$
$\quad \wedge\ \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \mathsf{r} \wedge \mathsf{r} <^* \{\alpha_2\}\}$

From Partition to Quicksort

Thus, if we define

$$
\begin{array}{l}
\mathsf{partition}(\mathsf{a},\mathsf{b},\mathsf{r};\mathsf{c}) = \\
\quad \textbf{newvar}\ \mathsf{d},\mathsf{x},\mathsf{y}\ \textbf{in}\ \big(\mathsf{c} := \mathsf{a}-1\ ;\ \mathsf{d} := \mathsf{b}+1\ ; \\
\quad \textbf{while}\ \mathsf{d} > \mathsf{c}+1\ \textbf{do} \\
\qquad \big(\mathsf{x} := [\mathsf{c}+1]\ ;\ \textbf{if}\ \mathsf{x} \le \mathsf{r}\ \textbf{then}\ \mathsf{c} := \mathsf{c}+1\ \textbf{else} \\
\qquad\quad \big(\mathsf{y} := [\mathsf{d}-1]\ ;\ \textbf{if}\ \mathsf{y} > \mathsf{r}\ \textbf{then}\ \mathsf{d} := \mathsf{d}-1\ \textbf{else} \\
\qquad\qquad ([\mathsf{c}+1] := \mathsf{y}\ ;\ [\mathsf{d}-1] := \mathsf{x}\ ;\ \mathsf{c} := \mathsf{c}+1\ ;\ \mathsf{d} := \mathsf{d}-1)\big)\big)\big),
\end{array}
$$

we have

$$
\begin{array}{l}
\{\mathsf{array}\ \alpha(\mathsf{a},\mathsf{b})\} \\
\mathsf{partition}(\mathsf{a},\mathsf{b},\mathsf{r};\mathsf{c}) \\
\{\exists \alpha_1,\alpha_2.\ (\mathsf{array}\ \alpha_1(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2(\mathsf{c}+1,\mathsf{b})) \\
\qquad \wedge\ \alpha_1 \cdot \alpha_2 \sim \alpha \wedge \{\alpha_1\} \le^* \mathsf{r} \wedge \mathsf{r} <^* \{\alpha_2\}\}.
\end{array}
$$

Now assume

$$
\begin{array}{l}
\{\mathsf{array}\ \alpha\ (\mathsf{a},\mathsf{b})\} \\
\mathsf{quicksort}(\mathsf{a},\mathsf{b}) \\
\{\exists \beta.\ \mathsf{array}\ \beta\ (\mathsf{a},\mathsf{b}) \wedge \beta \sim \alpha \wedge \mathbf{ord}\ \beta\}.
\end{array}
$$

Then

$\{\mathsf{array}\ \alpha\ (\mathsf{a}, \mathsf{b})\}$

if $\mathsf{a} < \mathsf{b}$ **then newvar** c **in**

$\big(\{\exists \mathsf{x}_1, \alpha_0, \mathsf{x}_2.\ (\mathsf{a} \mapsto \mathsf{x}_1\ *\ \mathsf{array}\ \alpha_0(\mathsf{a}+1, \mathsf{b}-1)\ *\ \mathsf{b} \mapsto \mathsf{x}_2)$

$\quad \wedge\ \mathsf{x}_1 \cdot \alpha_0 \cdot \mathsf{x}_2 \sim \alpha\}$

newvar $\mathsf{x1}, \mathsf{x2}, \mathsf{r}$ **in**

$\big(\mathsf{x1} := [\mathsf{a}]\ ;\ \mathsf{x2} := [\mathsf{b}]\ ;$

if $\mathsf{x1} > \mathsf{x2}$ **then** $([\mathsf{a}] := \mathsf{x2}\ ;\ [\mathsf{b}] := \mathsf{x1})$ **else skip** ;

$\mathsf{r} := (\mathsf{x1} + \mathsf{x2}) \div 2\ ;$

$\{\exists \mathsf{x}_1, \alpha_0, \mathsf{x}_2.\ (\mathsf{a} \mapsto \mathsf{x}_1\ *\ \mathsf{array}\ \alpha_0(\mathsf{a}+1, \mathsf{b}-1)\ *\ \mathsf{b} \mapsto \mathsf{x}_2)$

$\quad \wedge\ \mathsf{x}_1 \cdot \alpha_0 \cdot \mathsf{x}_2 \sim \alpha \wedge \mathsf{x}_1 \leq \mathsf{r} \leq \mathsf{x}_2\}$

$\mathsf{partition}(\mathsf{a}+1, \mathsf{b}-1, \mathsf{r}; \mathsf{c})$

$\{\exists \mathsf{x}_1, \alpha_1, \alpha_2, \mathsf{x}_2.$

$\quad (\mathsf{a} \mapsto \mathsf{x}_1 * \mathsf{array}\ \alpha_1(\mathsf{a}+1, \mathsf{c}) * \mathsf{array}\ \alpha_2(\mathsf{c}+1, \mathsf{b}-1) * \mathsf{b} \mapsto \mathsf{x}_2)$

$\quad \wedge\ \mathsf{x}_1 \cdot \alpha_1 \cdot \alpha_2 \cdot \mathsf{x}_2 \sim \alpha \wedge \mathsf{x}_1 \leq \mathsf{r} \leq \mathsf{x}_2 \wedge \{\alpha_1\} \leq^* \mathsf{r} \wedge \mathsf{r} <^* \{\alpha_2\}\}\big)\ ;$

$\{\exists \alpha_1, \alpha_2.\ (\mathsf{array}\ \alpha_1\ (\mathsf{a}, \mathsf{c})\ *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+1, \mathsf{b}))$

$\quad \wedge\ \alpha_1 \cdot \alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\}\}$

$\mathsf{quicksort}(\mathsf{a}, \mathsf{c})\ ;$

$\{\exists \beta_1, \alpha_2.\ (\mathsf{array}\ \beta_1\ (\mathsf{a}, \mathsf{c})\ *\ \mathsf{array}\ \alpha_2\ (\mathsf{c}+1, \mathsf{b}))$

$\quad \wedge\ \beta_1 \cdot \alpha_2 \sim \alpha \wedge \{\beta_1\} \leq^* \{\alpha_2\} \wedge \mathbf{ord}\ \beta_1\}$

$\mathsf{quicksort}(\mathsf{c}+1, \mathsf{b})$

$\{\exists \beta_1, \beta_2.\ (\mathsf{array}\ \beta_1\ (\mathsf{a}, \mathsf{c})\ *\ \mathsf{array}\ \beta_2\ (\mathsf{c}+1, \mathsf{b}))$

$\quad \wedge\ \beta_1 \cdot \beta_2 \sim \alpha \wedge \{\beta_1\} \leq^* \{\beta_2\} \wedge \mathbf{ord}\ \beta_1 \wedge \mathbf{ord}\ \beta_2\}\big)$

else skip

$\{\exists \beta.\ \mathsf{array}\ \beta\ (\mathsf{a}, \mathsf{b}) \wedge \beta \sim \alpha \wedge \mathbf{ord}\ \beta\}$

Quicksort (continued)

so that we may define

$$
\begin{array}{l}
\mathsf{quicksort}(\mathsf{a},\mathsf{b}) = \\
\quad \textbf{if } \mathsf{a} < \mathsf{b} \textbf{ then newvar } \mathsf{c} \textbf{ in} \\
\quad\quad \Big(\textbf{newvar } \mathsf{x1},\mathsf{x2},\mathsf{r} \textbf{ in} \\
\quad\quad\quad \Big(\mathsf{x1} := [\mathsf{a}] \,;\, \mathsf{x2} := [\mathsf{b}] \,; \\
\quad\quad\quad \textbf{if } \mathsf{x1} > \mathsf{x2} \textbf{ then } ([\mathsf{a}] := \mathsf{x2} \,;\, [\mathsf{b}] := \mathsf{x1}) \textbf{ else skip} \,; \\
\quad\quad\quad \mathsf{r} := (\mathsf{x1} + \mathsf{x2}) \div 2 \,;\, \mathsf{partition}(\mathsf{a}+1, \mathsf{b}-1, \mathsf{r}; \mathsf{c})\Big) \,; \\
\quad\quad \mathsf{quicksort}(\mathsf{a},\mathsf{c}) \,;\, \mathsf{quicksort}(\mathsf{c}+1,\mathsf{b})\Big) \\
\quad \textbf{else skip}.
\end{array}
$$

Reasoning about the First Recursive Call

From the assumption

$$
\begin{array}{l}
\{\mathsf{array}\ \alpha\ (\mathsf{a},\mathsf{b})\} \\
\mathsf{quicksort}(\mathsf{a},\mathsf{b}) \\
\{\exists\beta.\ \mathsf{array}\ \beta\ (\mathsf{a},\mathsf{b}) \wedge \beta \sim \alpha \wedge \mathbf{ord}\ \beta\},
\end{array}
$$

substitution (SUB) gives

$$
\begin{array}{l}
\{\mathsf{array}\ \alpha_1\ (\mathsf{a},\mathsf{c})\} \\
\mathsf{quicksort}(\mathsf{a},\mathsf{c}) \\
\{\exists\beta_1.\ \mathsf{array}\ \beta_1\ (\mathsf{a},\mathsf{c}) \wedge \beta_1 \sim \alpha_1 \wedge \mathbf{ord}\ \beta_1\}.
\end{array}
$$

Reasoning about the First Recursive Call (continued)

Then the frame rule (FR) gives the main step in

$$
\begin{array}{l}
\{(\mathsf{array}\ \alpha_1\,(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}))\\
\quad \wedge\ \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\}\}\\
\{\mathsf{array}\ \alpha_1\,(\mathsf{a},\mathsf{c})\\
\quad *\ (\mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}) \wedge \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\})\}\\
\mathsf{quicksort}(\mathsf{a},\mathsf{c})\\
\{(\exists\beta_1.\ \mathsf{array}\ \beta_1\,(\mathsf{a},\mathsf{c}) \wedge \beta_1 \sim \alpha_1 \wedge \mathbf{ord}\ \beta_1)\\
\quad *\ (\mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}) \wedge \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\})\}\\
\{\exists\beta_1.\ (\mathsf{array}\ \beta_1\,(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}))\\
\quad \wedge\ \beta_1 \sim \alpha_1 \wedge \mathbf{ord}\ \beta_1 \wedge \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\}\},
\end{array}
$$

and existential quantification (EQ) gives the main step in

$$
\begin{array}{l}
\{\exists\alpha_1,\alpha_2.\ (\mathsf{array}\ \alpha_1\,(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}))\\
\quad \wedge\ \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\}\}\\
\mathsf{quicksort}(\mathsf{a},\mathsf{c})\ ;\\
\{\exists\alpha_1,\alpha_2,\beta_1.\ (\mathsf{array}\ \beta_1\,(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}))\\
\quad \wedge\ \beta_1 \sim \alpha_1 \wedge \mathbf{ord}\ \beta_1 \wedge \alpha_1{\cdot}\alpha_2 \sim \alpha \wedge \{\alpha_1\} \leq^* \{\alpha_2\}\}\\
\{\exists\beta_1,\alpha_2.\ (\mathsf{array}\ \beta_1\,(\mathsf{a},\mathsf{c})\ *\ \mathsf{array}\ \alpha_2\,(\mathsf{c}+1,\mathsf{b}))\\
\quad \wedge\ \beta_1{\cdot}\alpha_2 \sim \alpha \wedge \{\beta_1\} \leq^* \{\alpha_2\} \wedge \mathbf{ord}\ \beta_1\}.
\end{array}
$$