Class Notes for CS 819A4 - Spring 2003

6. Trees and Dags

John C. Reynolds
Department of Computer Science
Carnegie Mellon University

Revised May 23, 2003

S-expressions (a la LISP)

$\tau \in$ S-exps iff

$$\tau \in \text{Atoms}$$

$$\text{or } \tau = (\tau_1 \cdot \tau_2) \text{ where } \tau_1, \tau_2 \in \text{S-exps.}$$

Representing S-expressions by Trees

For $\tau \in$ S-exps, we define the assertion

$$\mathsf{tree}\, \tau\, (i)$$

by structural induction:

$$\mathsf{tree}\, a\, (i) \text{ iff } \mathbf{emp} \wedge i = a$$

$$\begin{aligned}&\mathsf{tree}\, (\tau_1 \cdot \tau_2)\, (i) \text{ iff}\\ &\qquad \exists i_1, i_2.\ \ i \mapsto i_1, i_2 \ * \ \mathsf{tree}\, \tau_1\, (i_1) \ * \ \mathsf{tree}\, \tau_2\, (i_2)\end{aligned}$$

Representing S-expressions by Dags

For $\tau \in$ S-exps, we define

$$\mathsf{dag}\, \tau\, (i)$$

by:

$$\mathsf{dag}\, a\, (i) \text{ iff } i = a$$

$$\begin{aligned}&\mathsf{dag}\, (\tau_1 \cdot \tau_2)\, (i) \text{ iff}\\ &\qquad \exists i_1, i_2.\ \ i \mapsto i_1, i_2 \ * \ (\mathsf{dag}\, \tau_1\, (i_1) \wedge \mathsf{dag}\, \tau_2\, (i_2))\end{aligned}$$

Intuitionistic Assertions

An assertion p is *intuitionistic* iff, for all stores s and heaps h and h':

$$h \subseteq h' \text{ and } s, h \vDash p \text{ implies } s, h' \vDash p.$$

An assertion p is intuitionistic iff

$$p \ast \mathbf{true} \Rightarrow p.$$

(The opposite implication always holds.)

The assertion $\mathsf{dag}\,\tau\,(i)$ is intuitionistic. The proof is by induction on the structure of τ. If τ is an atom a,

$$\begin{aligned}
&\mathsf{dag}\,a\,(i) \ast \mathbf{true} \\
&\quad \Rightarrow i = a \ast \mathbf{true} \\
&\quad \Rightarrow i = a \\
&\quad \Rightarrow \mathsf{dag}\,a\,(i).
\end{aligned}$$

since $i = a$ is pure. Otherwise, $\tau = (\tau_1 \cdot \tau_2)$, and

$$\begin{aligned}
&\mathsf{dag}\,(\tau_1 \cdot \tau_2)\,(i) \ast \mathbf{true} \\
&\quad \Rightarrow \exists i_1, i_2.\ i \mapsto i_1, i_2 \ast (\mathsf{dag}\,\tau_1\,(i_1) \wedge \mathsf{dag}\,\tau_2\,(i_2)) \ast \mathbf{true} \\
&\quad \Rightarrow \exists i_1, i_2.\ i \mapsto i_1, i_2 \ast \\
&\qquad\quad ((\mathsf{dag}\,\tau_1\,(i_1) \ast \mathbf{true}) \wedge (\mathsf{dag}\,\tau_2\,(i_2) \ast \mathbf{true})) \\
&\quad \Rightarrow \exists i_1, i_2.\ i \mapsto i_1, i_2 \ast (\mathsf{dag}\,\tau_1\,(i_1) \wedge \mathsf{dag}\,\tau_2\,(i_2)) \\
&\quad \Rightarrow \mathsf{dag}\,(\tau_1 \cdot \tau_2)\,(i),
\end{aligned}$$

by the induction hypothesis for τ_1 and τ_2.

Copying Trees

We assume that the recursive procedure $\mathsf{copytree}$ satisfies

$$\{\mathsf{tree}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{tree}\,\tau(\mathsf{i})\ *\ \mathsf{tree}\,\tau(\mathsf{j})\}.$$

Then

$$
\begin{array}{l}
\{\mathsf{tree}\,\tau(\mathsf{i})\}\\
\textbf{if}\ \mathsf{isatom}(\mathsf{i})\ \textbf{then}\\
\quad \{\mathsf{isatom}(\tau) \wedge \mathbf{emp} \wedge \mathsf{i} = \tau\}\\
\quad \{\mathsf{isatom}(\tau) \wedge ((\mathbf{emp} \wedge \mathsf{i} = \tau)\ *\ (\mathbf{emp} \wedge \mathsf{i} = \tau))\}\\
\quad \mathsf{j} := \mathsf{i}\\
\quad \{\mathsf{isatom}(\tau) \wedge ((\mathbf{emp} \wedge \mathsf{i} = \tau)\ *\ (\mathbf{emp} \wedge \mathsf{j} = \tau))\}\\
\textbf{else}\\
\quad \{\exists \tau_1, \tau_2.\ \ \tau = (\tau_1 \cdot \tau_2) \wedge \mathsf{tree}\,(\tau_1 \cdot \tau_2)(\mathsf{i})\}\\
\quad \textbf{newvar}\ \mathsf{i}_1, \mathsf{i}_2, \mathsf{j}_1, \mathsf{j}_2\ \textbf{in}\\
\quad\quad \big(\mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i}+1]\ ;\\
\quad\quad \{\exists \tau_1, \tau_2.\ \ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ *\\
\quad\quad\quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2))\}\\
\quad\quad \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ;\\
\quad\quad \{\exists \tau_1, \tau_2.\ \ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ *\\
\quad\quad\quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1))\}\\
\quad\quad \vdots
\end{array}
$$

Copying Trees (continued)

$$
\begin{aligned}
&\qquad \vdots\\
&\qquad \{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge (\mathsf{i} \mapsto \mathsf{i}_1,\mathsf{i}_2 \ * \\
&\qquad\quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1))\}\\
&\qquad \mathbf{copytree}(\mathsf{i}_2;\mathsf{j}_2)\ ;\\
&\qquad \{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge\\
&\qquad\quad (\mathsf{i} \mapsto \mathsf{i}_1,\mathsf{i}_2\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\\
&\qquad\quad \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{j}_2))\}\\
&\qquad \mathsf{j} := \mathbf{cons}(\mathsf{j}_1,\mathsf{j}_2)\\
&\qquad \{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge\\
&\qquad\quad (\mathsf{i} \mapsto \mathsf{i}_1,\mathsf{i}_2\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\\
&\qquad\quad \mathsf{j} \mapsto \mathsf{j}_1,\mathsf{j}_2\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{j}_2))\}\\
&\qquad \{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge\\
&\qquad\quad (\mathsf{tree}\,(\tau_1{\cdot}\tau_2)\,(\mathsf{i})\ *\ \mathsf{tree}\,(\tau_1{\cdot}\tau_2)\,(\mathsf{j}))\}\Big)\\
&\ \{\mathsf{tree}\,\tau(\mathsf{i})\ *\ \mathsf{tree}\,\tau(\mathsf{j})\}.
\end{aligned}
$$

so that we may define

$$
\begin{aligned}
&\mathsf{copytree}(\mathsf{i};\mathsf{j}) =\\
&\quad \mathbf{if}\ \mathsf{isatom}(\mathsf{i})\ \mathbf{then}\ \mathsf{j} := \mathsf{i}\ \mathbf{else}\\
&\qquad \mathbf{newvar}\ \mathsf{i}_1,\mathsf{i}_2,\mathsf{j}_1,\mathsf{j}_2\ \mathbf{in}\\
&\qquad\quad \Big(\mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i}+1]\ ;\\
&\qquad\quad \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ;\ \mathbf{copytree}(\mathsf{i}_2;\mathsf{j}_2)\ ;\ \mathsf{j} := \mathbf{cons}(\mathsf{j}_1,\mathsf{j}_2)\Big)
\end{aligned}
$$

Proof of the First Recursive Call

$$\dfrac{\dfrac{\{\mathsf{tree}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{tree}\,\tau(\mathsf{i})\ *\ \mathsf{tree}\,\tau(\mathsf{j})\}}{\{\mathsf{tree}\,\tau_1(\mathsf{i}_1)\}\ \mathsf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ \{\mathsf{tree}\,\tau_1(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_1(\mathsf{j}_1)\}}}{\begin{array}{l}\{(\tau = (\tau_1 \cdot \tau_2) \land \mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2)\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\} \\ \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ; \\ \{(\tau = (\tau_1 \cdot \tau_2) \land \mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2)\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\}\end{array}}$$

$$\dfrac{\begin{array}{l}\{\tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2))\} \\ \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ; \\ \{\tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1))\}\end{array}}{\begin{array}{l}\{\exists \tau_1, \tau_2.\ \ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2))\} \\ \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ; \\ \{\exists \tau_1, \tau_2.\ \ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\ \quad \mathsf{tree}\,\tau_1\,(\mathsf{i}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{i}_2)\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1))\},\end{array}}$$

using the substitution rule, the frame rule, the rule of consequence (with the axiom that $(p \land q)\ *\ r \Leftrightarrow p \land (q\ *\ r)$ when p is pure), and the existential rule.

A Problem

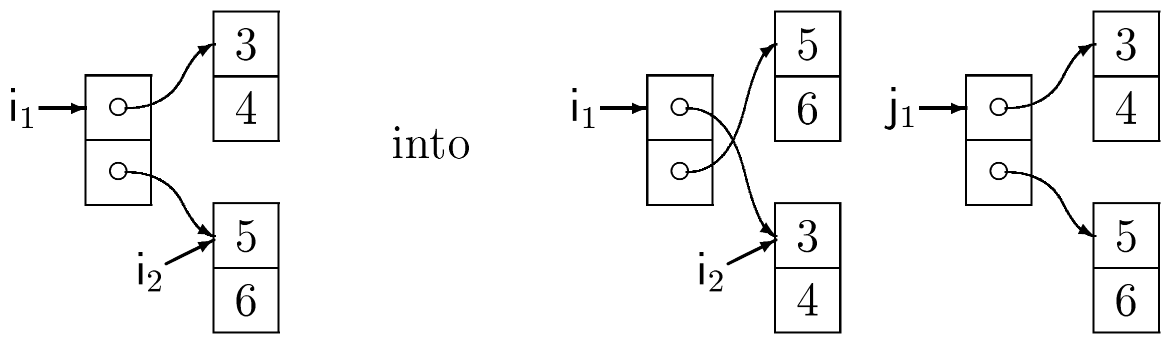
Suppose we wish to prove that

$$\{\mathsf{dag}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{dag}\,\tau(\mathsf{i}) \ * \ \mathsf{tree}\,\tau(\mathsf{j})\}$$

Then, we must use this specification as a hypothesis in proving that the recursive calls in the procedure body satisfy:

$$\begin{array}{l}
\{\mathsf{dag}\,\tau_1(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2(\mathsf{i}_2)\} \\
\mathsf{copytree}(\mathsf{i}_1;\mathsf{j}_1) \\
\{(\mathsf{dag}\,\tau_1(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2(\mathsf{i}_2)) \ * \ \mathsf{tree}\,\tau_1(\mathsf{j}_1)\} \\
\mathsf{copytree}(\mathsf{i}_2;\mathsf{j}_2) \\
\{(\mathsf{dag}\,\tau_1(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2(\mathsf{i}_2)) \ * \ \mathsf{tree}\,\tau_1(\mathsf{j}_1) \ * \ \mathsf{tree}\,\tau_2(\mathsf{j}_2)\}
\end{array}$$

But the hypothesis is not strong enough to imply this. For example, suppose $\tau_1 = ((3 \cdot 4) \cdot (5 \cdot 6))$ and $\tau_2 = (5 \cdot 6)$. Then $\mathsf{copytree}(\mathsf{i}_1;\mathsf{j}_1)$ might change the state from



where $\mathsf{dag}\,\tau_2\,(\mathsf{i}_2)$ is false.

Possible Solutions

1. Introduce ghost variables denoting heaps, e.g.

$$\begin{array}{l} \{\mathbf{this}(\mathsf{h}) \wedge \mathsf{dag}\,\tau(\mathsf{i})\} \\ \mathsf{copytree}(\mathsf{i};\mathsf{j}) \\ \{\mathbf{this}(\mathsf{h}) \ * \ \mathsf{tree}\,\tau(\mathsf{j})\} \end{array}$$

2. Introduce ghost variables denoting assertions, e.g.

$$\begin{array}{l} \{\mathsf{p} \wedge \mathsf{dag}\,\tau(\mathsf{i})\} \\ \mathsf{copytree}(\mathsf{i};\mathsf{j}) \\ \{\mathsf{p} \ * \ \mathsf{tree}\,\tau(\mathsf{j})\} \end{array}$$

3. Introduce passivity, i.e., the ability to assert that portions of the heap remain unchanged throughout the execution of a command.

We will explore the second approach.

Assertion Variables

$$
\begin{aligned}
\langle\text{assert}\rangle &::= \langle\text{avar}\rangle \mid \cdots \\
\text{AStores}_A &= A \to (\text{Heaps} \to \mathbf{B}) \\
\text{AStates}_{AV} &= \text{AStores}_A \times \text{Stores}_V \times \text{Heaps} \\
[\![p \in \langle\text{assert}\rangle]\!]_{\text{assert}} &\in \bigcup_{A \overset{\text{fin}}{\supseteq} \text{AV}(p), V \overset{\text{fin}}{\supseteq} \text{FV}(p)} \text{AStates}_{AV} \to \mathbf{B},
\end{aligned}
$$

where $\text{AV}(p)$ is the set of assertion variables occurring in p.

Instead of $[\![p]\!]_{\text{assert}}(as, s, h)$, however, we write $as, s, h \vDash p$.

When a is an assertion variable,

$$as, s, h \vDash a \text{ iff } \ as\, a\, h.$$

The Substitution Law for Assertions, Revisited

Proposition 11 *Let δ abbreviate the substitution*

$$a_1 \to p_1, \ldots, a_m \to p_m, v_1 \to e_1, \ldots, v_n \to e_n,$$

let as and s be such that $\text{AV}(p_1) \cup \cdots \cup \text{AV}(p_m) \subseteq \text{dom}\, as$ *and* $\text{FV}(p_1) \cup \cdots \cup \text{FV}(p_m) \cup \text{FV}(e_1) \cup \cdots \cup \text{FV}(e_n) \subseteq \text{dom}\, s$*, and let*

$$
\begin{aligned}
\widehat{as} &= [\, a_1 \colon [\![p_1]\!]_{\text{exp}} as\; s \mid \ldots \mid a_m \colon [\![p_m]\!]_{\text{exp}} as\; s \,] \\
\widehat{s} &= [\, v_1 \colon [\![e_1]\!]_{\text{exp}} s \mid \ldots \mid v_n \colon [\![e_n]\!]_{\text{exp}} s \,].
\end{aligned}
$$

If p is an assertion such that $\text{AV}(p) \subseteq \{a_1, \ldots, a_m\}$ *and* $\text{FV}(p) \subseteq \{v_1, \ldots, v_n\}$*, then*

$$as, s, h \vDash p/\delta \quad \textit{iff} \quad \widehat{as}, \widehat{s}, h \vDash p.$$

Specifications Revisited

Let $A = \mathrm{AV}(p) \cup \mathrm{AV}(q)$ and $V = \mathrm{FV}(p) \cup \mathrm{FV}(c) \cup \mathrm{FV}(q)$. Then

Partial correctness:

$\{p\}\ c\ \{q\}$ *holds* iff $\forall (as, s, h) \in \mathrm{AStates}_{AV}.\ as, s, h \vDash p$ implies

$(s, h), \textbf{abort} \notin [\![c]\!]_{\mathrm{comm}}$

$\{$i.e., $c, (s, h)$ is safe$\}$

and

$$\Big(\forall (s', h') \in \mathrm{States}_V.\ (s, h), (s', h') \in [\![c]\!]_{\mathrm{comm}} \text{ implies } as, s', h' \vDash q\Big).$$

Total correctness:

$[\ p\ ]\ c\ [\ q\ ]$ *holds* iff $\forall (as, s, h) \in \mathrm{AStates}_{AV}.\ as, s, h \vDash p$ implies

$(s, h), \textbf{abort} \notin [\![c]\!]_{\mathrm{comm}}$ and $(s, h), \bot \notin [\![c]\!]_{\mathrm{comm}}$

$\{$i.e., $c, (s, h)$ must terminate normally$\}$

and

$$\Big(\forall (s', h') \in \mathrm{States}_V.\ (s, h), (s', h') \in [\![c]\!]_{\mathrm{comm}} \text{ implies } as, s', h' \vDash q\Big).$$

The Substitution Rule (SUB) Revisited

$$\frac{\{p\}\ c\ \{q\}}{(\{p\}\ c\ \{q\})/a_1 \to p_1, \ldots, a_m \to p_m, v_1 \to e_1, \ldots, v_n \to e_n}$$

where $a_1, \ldots, a_m$ are the assertion variables occurring in p or q, and $v_1, \ldots, v_n$ 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 or in any p_j.

In $\{\mathsf{a}\}\ \mathsf{x} := \mathsf{y}\ \{\mathsf{a}\}$, for example, we can substitute $\mathsf{a} \to \mathsf{y} = \mathsf{z}, \mathsf{x} \to \mathsf{x}, \mathsf{y} \to \mathsf{y}$ to obtain

$$\{\mathsf{y} = \mathsf{z}\}\ \mathsf{x} := \mathsf{y}\ \{\mathsf{y} = \mathsf{z}\},$$

but we cannot substitute $\mathsf{a} \to \mathsf{x} = \mathsf{z}, \mathsf{x} \to \mathsf{x}, \mathsf{y} \to \mathsf{y}$ to obtain

$$\{\mathsf{x} = \mathsf{z}\}\ \mathsf{x} := \mathsf{y}\ \{\mathsf{x} = \mathsf{z}\}.$$

Proof: Assume $\{p\}\ c\ \{q\}$, and suppose $as, s, h \vDash p/\delta$, where δ is the substitution in the above rule. From the substitution law for assertions, $\widehat{as}, \widehat{s}, h \vDash p$.

Then since $\{p\}\ c\ \{q\}$, the definition of specification implies that $(\widehat{s}, h), \textbf{abort}$ does not belong to $[\![c]\!]_{\mathrm{comm}}$ (and in the total-correctness case, $(\widehat{s}, h), \perp$ does not belong to $[\![c]\!]_{\mathrm{comm}}$). Then the substitution law for commands (Proposition 5) implies that $(s, h), \textbf{abort}$ does not belong to $[\![c/\delta]\!]_{\mathrm{comm}}$ (and in the total-correctness case, $(s, h), \perp$ does not belong to $[\![c/\delta]\!]_{\mathrm{comm}}$).

Now suppose $(s, h), (s', h') \in [\![c/\delta]\!]_{\mathrm{comm}}$. By the substitution law for commands, $(\widehat{s}, h), (\widehat{s'}, h') \in [\![c]\!]_{\mathrm{comm}}$. Then $\{p\}\ c\ \{q\}$ and $\widehat{as}, \widehat{s}, h \vDash p$ imply $\widehat{as}, \widehat{s'}, h' \vDash q$, and the substitution law for assertions gives $as, s', h' \vDash q/\delta$.

A Tree-Copying Procedure

We will prove that the procedure

$$
\begin{array}{l}
\mathsf{copytree}(\mathsf{i};\mathsf{j}) = \\
\quad \textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then } \mathsf{j} := \mathsf{i} \textbf{ else} \\
\qquad \textbf{newvar } \mathsf{i}_1, \mathsf{i}_2, \mathsf{j}_1, \mathsf{j}_2 \textbf{ in} \\
\qquad\quad \big(\mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i}+1]\ ; \\
\qquad\quad \textbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ;\ \textbf{copytree}(\mathsf{i}_2;\mathsf{j}_2)\ ;\ \mathsf{j} := \textbf{cons}(\mathsf{j}_1,\mathsf{j}_2)\big)
\end{array}
$$

satisfies

$$\{\mathsf{p} \wedge \mathsf{dag}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{p} \ * \ \mathsf{tree}\,\tau(\mathsf{j})\}.$$

We can take p to be $\mathsf{dag}\,\tau(\mathsf{i})$, to obtain

$$\{\mathsf{dag}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{dag}\,\tau(\mathsf{i}) \ * \ \mathsf{tree}\,\tau(\mathsf{j})\},$$

but this specification is too weak to serve as a recursion hypothesis.

The Proof

$$
\begin{array}{l}
\{\mathsf{p} \wedge \mathsf{dag}\,\tau(\mathsf{i})\} \\
\textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then} \\
\quad \{\mathsf{p} \wedge \mathsf{isatom}(\tau) \wedge \tau = \mathsf{i}\} \\
\quad \{\mathsf{p} \ * \ (\mathsf{isatom}(\tau) \wedge \tau = \mathsf{i} \wedge \mathbf{emp})\} \\
\quad \mathsf{j} := \mathsf{i} \\
\quad \{\mathsf{p} \ * \ (\mathsf{isatom}(\tau) \wedge \tau = \mathsf{j} \wedge \mathbf{emp})\} \\
\textbf{else} \\
\quad \vdots
\end{array}
$$

Proof of copytree (continued)

$$\begin{array}{l}
\quad\vdots \\
\quad\{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,(\tau_1\cdot\tau_2)(\mathsf{i})\} \\
\quad\mathbf{newvar}\ \mathsf{i}_1,\mathsf{i}_2,\mathsf{j}_1,\mathsf{j}_2\ \mathbf{in} \\
\qquad\big(\mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i}+1]\ ; \\
\qquad\{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge \\
\qquad\quad \mathsf{p} \wedge (\mathsf{i} \mapsto \mathsf{i}_1,\mathsf{i}_2\ *\ (\mathsf{dag}\,\tau_1\,(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\} \\
\qquad\{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge \\
\qquad\quad \mathsf{p} \wedge (\mathbf{true}\ *\ (\mathsf{dag}\,\tau_1\,(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\} \\
\qquad\{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge \\
\qquad\quad \mathsf{p} \wedge ((\mathbf{true}\ *\ \mathsf{dag}\,\tau_1\,(\mathsf{i}_1)) \wedge (\mathbf{true}\ *\ \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\} \\
\qquad\{\exists\tau_1,\tau_2.\ \ \tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2) \wedge \mathsf{dag}\,\tau_1\,(\mathsf{i}_1)\} \\
\qquad\mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ; \\
\qquad\{\exists\tau_1,\tau_2.\ \ (\tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2))\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\} \\
\qquad\mathbf{copytree}(\mathsf{i}_2;\mathsf{j}_2)\ ; \\
\qquad\{\exists\tau_1,\tau_2.\ \ (\tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p})\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{j}_2)\} \\
\qquad\mathsf{j} := \mathbf{cons}(\mathsf{j}_1,\mathsf{j}_2) \\
\qquad\{\exists\tau_1,\tau_2.\ \ (\tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p})\ * \\
\qquad\quad \mathsf{j} \mapsto \mathsf{j}_1,\mathsf{j}_2\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\ *\ \mathsf{tree}\,\tau_2\,(\mathsf{j}_2)\} \\
\qquad\{\exists\tau_1,\tau_2.\ \ (\tau = (\tau_1\cdot\tau_2) \wedge \mathsf{p})\ *\ \mathsf{tree}\,(\tau_1\cdot\tau_2)\,(\mathsf{j})\}\big) \\
\{\mathsf{p}\ *\ \mathsf{tree}\,\tau(\mathsf{j})\}
\end{array}$$

Proof of the First Recursive Call

Using the substitution rule and the existential rule:

$$\dfrac{\dfrac{\{\mathsf{p} \wedge \mathsf{dag}\,\tau(\mathsf{i})\}\ \mathsf{copytree}(\mathsf{i};\mathsf{j})\ \{\mathsf{p}\ *\ \mathsf{tree}\,\tau(\mathsf{j})\}}{\begin{array}{l}\{(\tau = (\tau_1 \cdot \tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)) \wedge \mathsf{dag}\,\tau_1\,(\mathsf{i}_1)\}\\ \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ;\\ \{(\tau = (\tau_1 \cdot \tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2))\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\}\end{array}}}{\begin{array}{l}\{\exists \tau_1, \tau_2.\ \ (\tau = (\tau_1 \cdot \tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)) \wedge \mathsf{dag}\,\tau_1\,(\mathsf{i}_1)\}\\ \mathbf{copytree}(\mathsf{i}_1;\mathsf{j}_1)\ ;\\ \{\exists \tau_1, \tau_2.\ \ (\tau = (\tau_1 \cdot \tau_2) \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2))\ *\ \mathsf{tree}\,\tau_1\,(\mathsf{j}_1)\}.\end{array}}$$

Substitution of an S-expression for an Atom

$$
\begin{aligned}
a/a \to \tau' &= \tau' \\
b/a \to \tau' &= b \qquad \text{when } b \in \text{Atoms} - \{a\} \\
(\tau_1 \cdot \tau_2)/a \to \tau' &= ((\tau_1/a \to \tau') \cdot (\tau_2/a \to \tau'))
\end{aligned}
$$

Substitution by Copying

$$
\begin{array}{l}
\mathsf{subst1}(\mathsf{a}, \mathsf{j}\,;\mathsf{i}) = \\
\quad \mathbf{if}\ \mathsf{isatom}(\mathsf{i})\ \mathbf{then}\ \mathbf{if}\ \mathsf{i} = \mathsf{a}\ \mathbf{then}\ \mathsf{copytree}(\mathsf{j}\,;\mathsf{i})\ \mathbf{else\ skip} \\
\quad \mathbf{else\ newvar}\ \mathsf{i}_1, \mathsf{i}_2\ \mathbf{in}\ (\mathsf{i}_1 := [\mathsf{i}]\,;\mathsf{i}_2 := [\mathsf{i}+1]\,; \\
\qquad \mathsf{subst1}(\mathsf{a}, \mathsf{j}\,;\mathsf{i}_1)\,;\mathsf{subst1}(\mathsf{a}, \mathsf{j}\,;\mathsf{i}_2)\,;[\mathsf{i}] := \mathsf{i}_1\,;[\mathsf{i}+1] := \mathsf{i}_2)
\end{array}
$$

satisfies

$$
\begin{array}{l}
\{\mathsf{tree}\,\tau\,(\mathsf{i}) \ * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\mathsf{subst1}(\mathsf{a}, \mathsf{j}\,;\mathsf{i}) \\
\{\mathsf{tree}\,(\tau/\mathsf{a} \to \tau')\,(\mathsf{i}) \ * \ \mathsf{p}\}
\end{array}
$$

Substitution by Copying (continued)

since

$$
\begin{array}{l}
\{\mathsf{tree}\,\tau\,(\mathsf{i}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then} \\
\quad \{(\mathsf{isatom}(\tau) \wedge \tau = \mathsf{i} \wedge \mathbf{emp}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\quad \textbf{if } \mathsf{i} = \mathsf{a} \textbf{ then} \\
\quad\quad \{(\mathsf{isatom}(\tau) \wedge \tau = \mathsf{a} \wedge \mathbf{emp}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\quad\quad \{((\tau/\mathsf{a} \to \tau') = \tau' \wedge \mathbf{emp}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\quad\quad \mathsf{copytree}(\mathsf{j}\,;\mathsf{i}) \\
\quad\quad \{((\tau/\mathsf{a} \to \tau') = \tau' \wedge \mathbf{emp}) \;*\; \mathsf{tree}\,\tau'\,(\mathsf{i}) \;*\; \mathsf{p}\} \\
\quad \textbf{else} \\
\quad\quad \{(\mathsf{isatom}(\tau) \wedge \tau \neq \mathsf{a} \wedge \tau = \mathsf{i} \wedge \mathbf{emp}) \;*\; \mathsf{p}\} \\
\quad\quad \{((\tau/\mathsf{a} \to \tau') = \tau \wedge \mathsf{isatom}(\tau) \wedge \tau = \mathsf{i} \wedge \mathbf{emp}) \;*\; \mathsf{p}\} \\
\quad\quad \textbf{skip} \\
\quad\quad \vdots
\end{array}
$$

Substitution by Copying (continued)

$$
\begin{array}{l}
\qquad \vdots \\
\textbf{else newvar}\ \mathsf{i}_1, \mathsf{i}_2\ \textbf{in}\ \big(\\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_1, \mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\
\qquad \mathsf{tree}\, \tau_1\, (\mathsf{i}_1)\ *\ \mathsf{tree}\, \tau_2\, (\mathsf{i}_2)\ *\ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j})))\} \\
\quad \mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i} + 1]\ ; \\
\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto -, -\ * \\
\qquad \mathsf{tree}\, \tau_2\, (\mathsf{i}_2)\ *\ \mathsf{tree}\, \tau_1\, (\mathsf{i}_1)\ *\ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j}) \wedge \mathsf{dag}\, \tau'\, (\mathsf{j})))\} \\
\quad \mathsf{subst1}(\mathsf{a}, \mathsf{j}\ ;\ \mathsf{i}_1)\ ; \\
\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto -, -\ * \\
\qquad \mathsf{tree}\, \tau_2\, (\mathsf{i}_2)\ *\ \mathsf{tree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1)\ *\ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j})))\} \\
\quad \mathsf{subst1}(\mathsf{a}, \mathsf{j}\ ;\ \mathsf{i}_2)\ ; \\
\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto -, -\ * \\
\qquad \mathsf{tree}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{i}_2)\ *\ \mathsf{tree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1)\ *\ \mathsf{p})\} \\
\quad [\mathsf{i}] := \mathsf{i}_1\ ;\ [\mathsf{i} + 1] := \mathsf{i}_2 \\
\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\ * \\
\qquad \mathsf{tree}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{i}_2)\ *\ \mathsf{tree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1)\ *\ \mathsf{p})\} \\
\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge (\mathsf{tree}\, ((\tau_1/\mathsf{a} \to \tau') \cdot (\tau_2/\mathsf{a} \to \tau'))\, (\mathsf{i})\ *\ \mathsf{p})\}\big) \\
\{\mathsf{tree}\, (\tau/\mathsf{a} \to \tau')\, (\mathsf{i})\ *\ \mathsf{p}\}
\end{array}
$$

Substitution without Copying

$$\begin{array}{l}
\mathsf{subst2}(\mathsf{a},\mathsf{j}\,;\mathsf{i}) = \\
\textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then if } \mathsf{i} = \mathsf{a} \textbf{ then } \mathsf{i} := \mathsf{j} \textbf{ else skip} \\
\textbf{else newvar } \mathsf{i}_1, \mathsf{i}_2 \textbf{ in } (\mathsf{i}_1 := [\mathsf{i}]\,;\mathsf{i}_2 := [\mathsf{i}+1]\,; \\
\quad \mathsf{subst2}(\mathsf{a},\mathsf{j}\,;\mathsf{i}_1)\,;\mathsf{subst2}(\mathsf{a},\mathsf{j}\,;\mathsf{i}_2)\,;[\mathsf{i}] := \mathsf{i}_1\,;[\mathsf{i}+1] := \mathsf{i}_2)
\end{array}$$

satisfies

$$\begin{array}{l}
\{\mathsf{tree}\,\tau\,(\mathsf{i}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\mathsf{subst2}(\mathsf{a},\mathsf{j}\,;\mathsf{i}) \\
\{\mathsf{dag}\,(\tau/\mathsf{a} \to \tau')\,(\mathsf{i}) \wedge (\mathsf{p} \;*\; \textbf{true})\}
\end{array}$$

since

$$\begin{array}{l}
\{\mathsf{tree}\,\tau\,(\mathsf{i}) \;*\; (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then} \\
\quad \{\mathsf{isatom}(\tau) \wedge \tau = \mathsf{i} \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j})\} \\
\quad \textbf{if } \mathsf{i} = \mathsf{a} \textbf{ then} \\
\qquad \{\mathsf{isatom}(\tau) \wedge \tau = \mathsf{a} \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j})\} \\
\qquad \{(\tau/\mathsf{a} \to \tau') = \tau' \wedge \mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j})\} \\
\qquad \mathsf{i} := \mathsf{j} \\
\quad \textbf{else} \\
\qquad \{\mathsf{isatom}(\tau) \wedge \tau \neq \mathsf{a} \wedge \tau = \mathsf{i} \wedge \mathsf{p}\} \\
\qquad \{(\tau/\mathsf{a} \to \tau') = \tau \wedge \mathsf{isatom}(\tau) \wedge \tau = \mathsf{i} \wedge \mathsf{p}\} \\
\qquad \textbf{skip} \\
\quad \{\mathsf{dag}\,(\tau/\mathsf{a} \to \tau')\,(\mathsf{i}) \wedge \mathsf{p}\} \\
\qquad \vdots
\end{array}$$

$$\vdots$$

$$
\begin{aligned}
&\textbf{else newvar } \mathsf{i}_1, \mathsf{i}_2 \textbf{ in } \Big(\\
&\quad \{\exists \tau_1, \tau_2, \mathsf{i}_1, \mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2 \ * \\
&\qquad \mathsf{tree}\, \tau_1\, (\mathsf{i}_1) \ * \ \mathsf{tree}\, \tau_2\, (\mathsf{i}_2) \ * \ (\mathsf{p} \land \mathsf{dag}\, \tau'\, (\mathsf{j}))))\}\\
&\quad \mathsf{i}_1 := [\mathsf{i}]\ ;\ \mathsf{i}_2 := [\mathsf{i}+1]\ ;\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad \mathsf{tree}\, \tau_2\, (\mathsf{i}_2) \ * \ \mathsf{tree}\, \tau_1\, (\mathsf{i}_1) \ * \ (\mathsf{p} \land \mathsf{dag}\, \tau'\, (\mathsf{j}) \land \mathsf{dag}\, \tau'\, (\mathsf{j}))))\}\\
&\quad \mathsf{subst2}(\mathsf{a}, \mathsf{j}\ ;\ \mathsf{i}_1)\ ;\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad \mathsf{tree}\, \tau_2\, (\mathsf{i}_2) \ * \ (\mathsf{dag}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1) \land ((\mathsf{p} \land \mathsf{dag}\, \tau'\, (\mathsf{j})) \ * \ \mathbf{true})))\}\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad \mathsf{tree}\, \tau_2\, (\mathsf{i}_2) \ * \ (\mathsf{dag}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1) \land (\mathsf{p} \ * \ \mathbf{true}) \land \mathsf{dag}\, \tau'\, (\mathsf{j})))\}\\
&\quad \mathsf{subst2}(\mathsf{a}, \mathsf{j}\ ;\ \mathsf{i}_2)\ ;\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad (\mathsf{dag}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{i}_2) \land \\
&\qquad\quad ((\mathsf{dag}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1) \land (\mathsf{p} \ * \ \mathbf{true})) \ * \ \mathbf{true})))\}\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad (\mathsf{dag}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{i}_2) \land \mathsf{dag}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1) \land (\mathsf{p} \ * \ \mathbf{true})))\}\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land (\mathsf{i} \mapsto -, - \ * \\
&\qquad (\mathsf{dag}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{i}_2) \land \mathsf{dag}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1))) \land (\mathsf{p} \ * \ \mathbf{true})\}\\
&\quad [\mathsf{i}] := \mathsf{i}_1\ ;\ [\mathsf{i}+1] := \mathsf{i}_2\\
&\quad \{\exists \tau_1, \tau_2.\ \tau = (\tau_1 \cdot \tau_2) \land \mathsf{dag}\, ((\tau_1/\mathsf{a} \to \tau') \cdot (\tau_2/\mathsf{a} \to \tau'))\, (\mathsf{i}) \land \\
&\qquad\quad (\mathsf{p} \ * \ \mathbf{true})\}\Big)\\
&\{\mathsf{dag}\, (\tau/\mathsf{a} \to \tau')\, (\mathsf{i}) \land (\mathsf{p} \ * \ \mathbf{true})\}
\end{aligned}
$$

Representing S-expressions by Back-Linked Trees

For $\tau \in$ S-exps, we define

$$\mathsf{bktree}\,\tau\,(i, b, t)$$

by structural induction:

$$\mathsf{bktree}\,a\,(i, b, t) \text{ iff } i \mapsto b, t, a, -$$

$$\begin{aligned}&\mathsf{bktree}\,(\tau_1 \cdot \tau_2)\,(i, b, t) \text{ iff}\\ &\qquad \exists i_1, i_2.\;\; i \mapsto b, t, i_1, i_2 \;*\; \mathsf{bktree}\,\tau_1\,(i_1, i, 0) \;*\; \mathsf{bktree}\,\tau_2\,(i_2, i, 1).\end{aligned}$$

Then

$$\begin{aligned}&\mathsf{copybktree(i, j)} =\\ &\quad \textbf{if } \mathsf{isatom(i)} \textbf{ then } [\mathsf{j} + 2] := \mathsf{i}\\ &\quad \textbf{else newvar } \mathsf{i'}, \mathsf{j'} \textbf{ in}\\ &\qquad \big(\mathsf{i'} := [\mathsf{i}]\,;\, \mathsf{j'} := \textbf{cons}(\mathsf{j}, 0, 0, 0)\,;\, \mathsf{copybktree(i', j')}\,;\, [\mathsf{j} + 2] := \mathsf{j'}\,;\\ &\qquad \mathsf{i'} := [\mathsf{i} + 1]\,;\, \mathsf{j'} := \textbf{cons}(\mathsf{j}, 1, 0, 0)\,;\, \mathsf{copybktree(i', j')}\,;\, [\mathsf{j} + 3] := \mathsf{j'}\big)\end{aligned}$$

satisfies

$$\begin{aligned}&\{(\mathsf{p} \wedge \mathsf{dag}\,\tau\,(\mathsf{i})) \;*\; \mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\}\\ &\mathsf{copybktree(i, j)}\\ &\{\mathsf{p} \;*\; \mathsf{bktree}\,\tau\,(\mathsf{j}, \mathsf{b}, \mathsf{t})\}\end{aligned}$$

since

$$
\begin{array}{l}
\{(\mathsf{p} \wedge \mathsf{dag}\,\tau\,(\mathsf{i}))\;*\;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\} \\
\textbf{if } \mathsf{isatom}(\mathsf{i}) \textbf{ then} \\
\quad \{(\mathsf{p} \wedge \mathsf{isatom}(\tau) \wedge \tau = i)\;*\;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\} \\
\quad [\mathsf{j} + 2] := \mathsf{i} \\
\quad \{(\mathsf{p} \wedge \mathsf{isatom}(\tau) \wedge \tau = i)\;*\;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, \mathsf{i}, -\} \\
\textbf{else newvar } \mathsf{i}', \mathsf{j}' \textbf{ in} \\
\quad \Big(\{\exists \tau_1, \tau_2, \mathsf{i}_1, \mathsf{i}_2.\; \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad \Big(\Big(p \wedge (\mathsf{i} \mapsto \mathsf{i}_1, \mathsf{i}_2\;*\;(\mathsf{dag}\,\tau_1\,(\mathsf{i}_1) \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\Big) * \;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\Big)\} \\
\quad \mathsf{i}' := [\mathsf{i}]\;; \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_2.\; \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad \Big(\Big(p \wedge (\mathsf{i} \mapsto \mathsf{i}', \mathsf{i}_2\;*\;(\mathsf{dag}\,\tau_1\,(\mathsf{i}') \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\Big) * \;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\Big)\} \\
\quad \mathsf{j}' := \mathbf{cons}(\mathsf{j}, 0, 0, 0)\;; \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_2.\; \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad \Big(\Big(p \wedge (\mathsf{i} \mapsto \mathsf{i}', \mathsf{i}_2\;*\;(\mathsf{dag}\,\tau_1\,(\mathsf{i}') \wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}_2)))\Big) * \;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, - \\
\qquad\qquad * \;\mathsf{j}' \mapsto \mathsf{j}, 0, -, -\Big)\} \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_2.\; \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad \Big(\Big(p \wedge (\mathsf{i} \mapsto -, \mathsf{i}_2\;*\;\mathsf{dag}\,\tau_2\,(\mathsf{i}_2)) \wedge \mathsf{dag}\,\tau_1\,(\mathsf{i}')\Big) * \;\mathsf{j}' \mapsto \mathsf{j}, 0, -, - \\
\qquad\qquad * \;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\Big)\} \\
\quad \mathsf{copybktree}(\mathsf{i}', \mathsf{j}')\;; \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_2.\; \tau = (\tau_1 \cdot \tau_2) \wedge \\
\quad \Big(\Big(p \wedge (\mathsf{i} \mapsto -, \mathsf{i}_2\;*\;\mathsf{dag}\,\tau_2\,(\mathsf{i}_2))\Big) * \;\mathsf{bktree}\,\tau_1\,(\mathsf{j}', \mathsf{j}, 0)\;*\;\mathsf{j} \mapsto \mathsf{b}, \mathsf{t}, -, -\Big)\} \\
\quad [\mathsf{j} + 2] := \mathsf{j}'\;; \\
\qquad \vdots
\end{array}
$$

$$\vdots$$

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1,\mathsf{i}_2.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$\Big(\big(p\wedge(\mathsf{i}\mapsto -,\mathsf{i}_2\ *\ \mathsf{dag}\,\tau_2\,(\mathsf{i}_2))\big) *\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,-\Big)\}$$

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

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$\Big(\big(p\wedge(\mathsf{i}\mapsto -,\mathsf{i}'\ *\ \mathsf{dag}\,\tau_2\,(\mathsf{i}'))\big) *\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,-\Big)\}$$

$$\mathsf{j}' := \mathbf{cons}(\mathsf{j},1,0,0)\ ;$$

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$\Big(\big(p\wedge(\mathsf{i}\mapsto -,\mathsf{i}'\ *\ \mathsf{dag}\,\tau_2\,(\mathsf{i}'))\big) *\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,-$$
$$*\ \mathsf{j}'\mapsto \mathsf{j},1,-,-\Big)\}$$

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$\Big(\big(p\wedge \mathsf{dag}\,\tau_2\,(\mathsf{i}')\big) *\ \mathsf{j}'\mapsto \mathsf{j},1,-,-$$
$$*\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,-\Big)\}$$

$$\mathsf{copybktree}(\mathsf{i}',\mathsf{j}')\ ;$$

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$(p\ *\ \mathsf{bktree}\,\tau_2\,(\mathsf{j}',\mathsf{j},1)\ *\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,-)\}$$

$$[\mathsf{j}+3] := \mathsf{j}'$$

$$\{\exists\tau_1,\tau_2,\mathsf{j}_1,\mathsf{j}_2.\ \tau = (\tau_1\cdot\tau_2)\ \wedge$$
$$(p\ *\ \mathsf{bktree}\,\tau_2\,(\mathsf{j}_2,\mathsf{j},1)\ *\ \mathsf{bktree}\,\tau_1\,(\mathsf{j}_1,\mathsf{j},0)\ *\ \mathsf{j}\mapsto \mathsf{b},\mathsf{t},\mathsf{j}_1,\mathsf{j}_2)\}\Big)$$

$$\{\mathsf{p}\ *\ \mathsf{bktree}\,\tau\,(\mathsf{j},\mathsf{b},\mathsf{t})\}.$$

Substitution in Back-Linked Trees

We first note that

$$\begin{aligned}
&\mathsf{bktree}\,\tau\,(\mathsf{i},\mathsf{b},\mathsf{t}) \Rightarrow\\
&\quad \exists\mathsf{k}.\ \mathsf{i} \hookrightarrow \mathsf{b},\mathsf{t},\mathsf{k},-\\
&\qquad \wedge (\mathsf{isatom}(\mathsf{k}) \Rightarrow \tau = \mathsf{k} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-)\\
&\qquad \wedge (\neg\,\mathsf{isatom}(\mathsf{k}) \Rightarrow \exists\tau_1,\tau_2,\mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge\\
&\qquad\quad (\mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},\mathsf{i}_2 \ *\ \mathsf{bktree}\,\tau_1\,(\mathsf{k},\mathsf{i},0)\ *\ \mathsf{bktree}\,\tau_2\,(\mathsf{i}_2,\mathsf{i},1)).
\end{aligned}$$

Then

$$\begin{aligned}
&\mathsf{subst3}(\mathsf{a},\mathsf{j},\mathsf{i}) =\\
&\quad \textbf{newvar}\ \mathsf{k}\ \textbf{in}\ \Big(\mathsf{k} := [\mathsf{i}+2]\,;\\
&\qquad \textbf{if}\ \mathsf{isatom}(\mathsf{k})\ \textbf{then}\\
&\qquad\quad \textbf{if}\ \mathsf{k} = \mathsf{a}\ \textbf{then}\ \mathsf{copybktree}(\mathsf{j},\mathsf{i})\ \textbf{else skip}\\
&\qquad \textbf{else}\ (\mathsf{subst3}(\mathsf{a},\mathsf{j},\mathsf{k})\,;\mathsf{k} := [\mathsf{i}+3]\,;\mathsf{subst3}(\mathsf{a},\mathsf{j},\mathsf{k}))\Big)
\end{aligned}$$

satisfies

$$\begin{aligned}
&\{\mathsf{bktree}\,\tau\,(\mathsf{i},\mathsf{b},\mathsf{t})\ *\ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\}\\
&\mathsf{subst3}(\mathsf{a},\mathsf{j},\mathsf{i})\\
&\{\mathsf{bktree}\,(\tau/\mathsf{a} \to \tau')\,(\mathsf{i},\mathsf{b},\mathsf{t})\ *\ \mathsf{p}\}
\end{aligned}$$

since

$$
\begin{array}{l}
\{\mathsf{bktree}\,\tau\,(\mathsf{i},\mathsf{b},\mathsf{t}) \ * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\textbf{newvar}\ \mathsf{k}\ \textbf{in}\ \Big(\mathsf{k} := [\mathsf{i}+2]\,; \\
\quad \{(\mathsf{i} \hookrightarrow \mathsf{b},\mathsf{t},\mathsf{k},- \wedge (\mathsf{isatom}(\mathsf{k}) \Rightarrow \tau = \mathsf{k} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-) \\
\qquad\quad \wedge\ (\neg\,\mathsf{isatom}(\mathsf{k}) \Rightarrow \exists \tau_1,\tau_2,\mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad\qquad (\mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},\mathsf{i}_2 \ * \ \mathsf{bktree}\,\tau_1\,(\mathsf{k},\mathsf{i},0) \ * \ \mathsf{bktree}\,\tau_2\,(\mathsf{i}_2,\mathsf{i},1)))) \\
\qquad * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\quad \textbf{if}\ \mathsf{isatom}(\mathsf{k})\ \textbf{then} \\
\qquad \{(\mathsf{isatom}(\tau) \wedge \tau = \mathsf{k} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-) \ * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\qquad \textbf{if}\ \mathsf{k} = \mathsf{a}\ \textbf{then} \\
\qquad\quad \{(\mathsf{isatom}(\tau) \wedge \tau = \mathsf{a} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-) \ * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j}))\} \\
\qquad\quad \{(\tau/\mathsf{a} \to \tau') = \tau' \wedge (\mathsf{i} \mapsto \mathsf{b},\mathsf{t},-,- \ * \ (\mathsf{p} \wedge \mathsf{dag}\,\tau'\,(\mathsf{j})))\} \\
\qquad\quad \mathsf{copybktree}(\mathsf{j},\mathsf{i}) \\
\qquad\quad \{(\tau/\mathsf{a} \to \tau') = \tau' \wedge (\mathsf{bktree}\,\tau'\,(\mathsf{i},\mathsf{b},\mathsf{t}) \ * \ \mathsf{p})\} \\
\qquad \textbf{else} \\
\qquad\quad \{(\mathsf{isatom}(\tau) \wedge \tau \neq \mathsf{a} \wedge \tau = \mathsf{k} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-) \ * \ \mathsf{p}\} \\
\qquad\quad \{((\tau/\mathsf{a} \to \tau') = \tau \wedge \mathsf{isatom}(\tau) \wedge \tau = \mathsf{k} \wedge \mathsf{i} \mapsto \mathsf{b},\mathsf{t},\mathsf{k},-) \ * \ \mathsf{p}\} \\
\qquad\quad \textbf{skip} \\
\quad \vdots
\end{array}
$$

$$
\begin{array}{l}
\quad \vdots \\
\textbf{else} \\
\quad \Big(\{\exists \tau_1, \tau_2, \mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad (\mathsf{i} \mapsto \mathsf{b}, \mathsf{t}, \mathsf{k}, \mathsf{i}_2 \ * \ \mathsf{bktree}\, \tau_1\, (\mathsf{k}, \mathsf{i}, 0) \ * \ \mathsf{bktree}\, \tau_2\, (\mathsf{i}_2, \mathsf{i}, 1) \\
\qquad * \ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j}) \wedge \mathsf{dag}\, \tau'\, (\mathsf{j}))))\} \\
\quad \mathsf{subst3}(\mathsf{a}, \mathsf{j}, \mathsf{k})\ ; \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_2.\ \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad (\mathsf{i} \mapsto \mathsf{b}, \mathsf{t}, \mathsf{k}, \mathsf{i}_2 \ * \ \mathsf{bktree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{k}, \mathsf{i}, 0) \ * \ \mathsf{bktree}\, \tau_2\, (\mathsf{i}_2, \mathsf{i}, 1) \\
\qquad * \ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j}))))\} \\
\quad \mathsf{k} := [\mathsf{i} + 3]\ ; \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_1.\ \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad (\mathsf{i} \mapsto \mathsf{b}, \mathsf{t}, \mathsf{i}_1, \mathsf{k} \ * \ \mathsf{bktree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1, \mathsf{i}, 0) \ * \ \mathsf{bktree}\, \tau_2\, (\mathsf{k}, \mathsf{i}, 1) \\
\qquad * \ (\mathsf{p} \wedge \mathsf{dag}\, \tau'\, (\mathsf{j}))))\} \\
\quad \mathsf{subst3}(\mathsf{a}, \mathsf{j}, \mathsf{k}) \\
\quad \{\exists \tau_1, \tau_2, \mathsf{i}_1.\ \tau = (\tau_1 \cdot \tau_2) \wedge \\
\qquad (\mathsf{i} \mapsto \mathsf{b}, \mathsf{t}, \mathsf{i}_1, \mathsf{k} \ * \ \mathsf{bktree}\, (\tau_1/\mathsf{a} \to \tau')\, (\mathsf{i}_1, \mathsf{i}, 0) \\
\qquad * \ \mathsf{bktree}\, (\tau_2/\mathsf{a} \to \tau')\, (\mathsf{k}, \mathsf{i}, 1) \ * \ \mathsf{p})\}\Big)\Big) \\
\{\mathsf{bktree}\, (\tau/\mathsf{a} \to \tau')\, (\mathsf{i}, \mathsf{b}, \mathsf{t}) \ * \ \mathsf{p}\}
\end{array}
$$