Dequeue Side Semantics

Disclaimer: we assume familiarity with citeOG, adopt its notational conventions, and steal its definitions! Further, we work with a specific subset of *Rio*, denoted **Rio**, namely

$$\frac{c \in \mathbf{Class}}{\mathbf{edf}[c], \mathbf{fifo}[c] \in \mathbf{Rio}}\ \text{set2stream} \qquad \frac{n \in \mathbb{N} \qquad rs \in \mathbf{Rio}^n}{\mathbf{strict}[rs], \mathbf{rr}[rs] \in \mathbf{Rio}}\ \text{stream2stream}$$

where **Class** is an opaque collection of *classes*.

1 Structure and Semantics of Rio Trees

1.1 Structure

Definition 1.1. For topology $t \in \mathbf{Topo}$, the set $\mathbf{RioTree}(t)$ of *Rio trees* over t is defined by

$$\frac{p \in \mathbf{PIFO}(\mathbf{Pkt}) \qquad c \in \mathbf{Class}}{\mathrm{Leaf}(p, c) \in \mathbf{RioTree}(*)} \qquad \frac{ts \in \mathbf{Topo}^n \qquad \forall 1 \leq i \leq n.\ qs[i] \in \mathbf{RioTree}(ts[i])}{\mathrm{Internal}(qs) \in \mathbf{RioTree}(\mathrm{Node}(ts))}$$

These are trees with leaves decorated by both classes and PIFOs.

Definition 1.2. For topology $t \in \mathbf{Topo}$, the set $\mathbf{OrdTree}(t)$ of *ordered trees* over t is defined by

$$\frac{}{\mathrm{Leaf} \in \mathbf{OrdTree}(*)} \qquad \frac{ts \in \mathbf{Topo}^n \qquad rs \in \mathbf{Rk}^n \qquad \forall 1 \leq i \leq n.\ os[i] \in \mathbf{OrdTree}(ts[i])}{\mathrm{Internal}(rs, os) \in \mathbf{OrdTree}(\mathrm{Node}(ts))}$$

These are trees with each internal node's child given a rank, thereby inducing a total ordering of children.

1.2 Semantics

Let $\mathbf{flow} : \mathbf{Pkt} \to \mathbf{Class}$ be an opaque mapping from packets to the class they belong to (flow inference).

Definition 1.3. For $t \in \mathbf{Topo}$, define $\mathrm{push} : \mathbf{RioTree}(t) \times \mathbf{Pkt} \times \mathbf{Rk} \rightharpoonup \mathbf{RioTree}(t)$ such that

$$\frac{\mathbf{flow}(\mathrm{pkt}) = c \qquad \mathrm{push}(p, \mathrm{pkt}, r) = p'}{\mathrm{push}(\mathrm{Leaf}(p, c), \mathrm{pkt}, r) = \mathrm{Leaf}(p', c)} \qquad \frac{\forall 1 \leq i \leq |qs|.\ \mathrm{push}(qs[i], \mathrm{pkt}, r) = qs'[i]}{\mathrm{push}(\mathrm{Internal}(qs), \mathrm{pkt}, r) = \mathrm{Internal}(qs')} \qquad \frac{\mathbf{flow}(\mathrm{pkt}) \neq c}{\mathrm{push}(\mathrm{Leaf}(p, c), \mathrm{pkt}, r) = \mathrm{Leaf}(p, c)}$$

Informally, we recursively push to all subtrees but only the PIFOs on leaves with the packet's flow are updated.

Definition 1.4. For $t \in \mathbf{Topo}$, define $\mathrm{pop} : \mathbf{RioTree}(t) \times \mathbf{OrdTree}(t) \rightharpoonup \mathbf{Pkt} \times \mathbf{RioTree}(t)$ such that

$$\frac{\mathrm{pop}(p) = (\mathrm{pkt}, p')}{\mathrm{pop}(\mathrm{Leaf}(p, c), \mathrm{Leaf}) = (\mathrm{pkt}, \mathrm{Leaf}(p', c))} \qquad \frac{\begin{array}{c}\exists 1 \leq i \leq |qs|.\ \mathrm{pop}(qs[i], os[i]) = (\mathrm{pkt}, q) \\ \forall 1 \leq j \leq |qs|.\ j \neq i \wedge \mathrm{pop}(qs[j], os[j]) = (\mathrm{pkt}', q') \implies rs[i] < rs[j]\end{array}}{\mathrm{pop}(\mathrm{Internal}(qs), \mathrm{Internal}(rs, os)) = (\mathrm{pkt}, \mathrm{Internal}(qs[q/i]))}$$

Informally, we recursively pop the smallest ranked, poppable subtree.

2 PIFO & Rio Controls

$$\frac{\mathsf{z}_{\mathsf{in}}(s, \mathsf{pkt}) = s' \qquad \mathsf{push}(q, \mathsf{pkt}) = q'}{\mathsf{push}((s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}}), \mathsf{pkt}) = (s', q', \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})}\ \text{RioCtrl-Push}$$

$$\frac{\mathsf{z}_{\mathsf{out}}(s) = (o, s') \qquad \mathsf{pop}(q, o) = (\mathsf{pkt}, q')}{\mathsf{pop}((s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})) = (\mathsf{pkt}, (s', q', \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}}))}\ \text{RioCtrl-Pop}$$

$$\frac{\mathsf{z}_{\mathsf{in}}(s, \mathsf{pkt}) = (pt, s') \qquad \mathsf{push}(q, \mathsf{pkt}, pt) = q'}{\mathsf{push}((s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}}), \mathsf{pkt}) = (s', q', \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})}\ \text{PIFOCtrl-Push}$$

$$\frac{\mathsf{pop}(q) = (\mathsf{pkt}, q') \qquad \mathsf{z}_{\mathsf{out}}(s, \mathsf{pkt}) = s'}{\mathsf{pop}((s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})) = (\mathsf{pkt}, (s', q', \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}}))}\ \text{PIFOCtrl-Pop}$$

Figure 1. Pushing and Popping Controls

For these notes, we'll refer to *controls* from citeOG as *PIFO controls*.

Definition 2.1. For $t \in \mathbf{Topo}$, $\mathbf{PIFOControl}(t)$ is the set of quadruples of $(s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})$ where

$$\mathsf{z}_{\mathsf{in}} : \mathbf{St} \times \mathbf{Pkt} \to \mathbf{Path}(t) \times \mathbf{St}$$
$$\mathsf{z}_{\mathsf{out}} : \mathbf{St} \times \mathbf{Pkt} \to \mathbf{St}$$

$s \in \mathbf{St}$ and $q \in \mathbf{PIFOTree}(t)$.

Here's the dequeue side version.

Definition 2.2. For $t \in \mathbf{Topo}$, $\mathbf{RioControl}(t)$ is the set of quadruples of $(s, q, \mathsf{z}_{\mathsf{in}}, \mathsf{z}_{\mathsf{out}})$ where

$$\mathsf{z}_{\mathsf{in}} : \mathbf{St} \times \mathbf{Pkt} \to \mathbf{St}$$
$$\mathsf{z}_{\mathsf{out}} : \mathbf{St} \to \mathbf{OrdTree}(t) \times \mathbf{St}$$

$s \in \mathbf{St}$ and $q \in \mathbf{RioTree}(t)$.

Both controls admit push and pop operations:

$$\mathsf{push} : \mathbf{RioControl}(t) \times \mathbf{Pkt} \to \mathbf{RioControl}(t) \qquad \mathsf{push} : \mathbf{PIFOControl}(t) \times \mathbf{Pkt} \to \mathbf{PIFOControl}(t)$$
$$\mathsf{pop} : \mathbf{RioControl}(t) \to \mathbf{Pkt} \times \mathbf{RioControl}(t) \qquad \mathsf{pop} : \mathbf{PIFOControl}(t) \to \mathbf{Pkt} \times \mathbf{PIFOControl}(t)$$

Their semantics are written out in full in Figure 1.

3 Round-Robin

Let's put our theory to use by constructing both a **PIFOControl** and **RioControl** for

$$\mathbf{rr}[(\mathbf{FIFO}[F_1], \mathbf{FIFO}[F_2], \ldots, \mathbf{FIFO}[F_n])] \qquad\qquad \text{distinct } F_1, F_2, \ldots, F_n \in \mathbf{Class}$$

and showing they're in simulation. This would show the equivalence of enqueue and dequeue-side semantics for a specific type of program!