Interleaved scope for games and automata

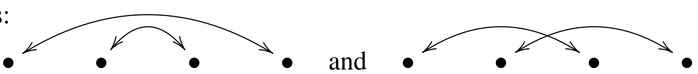
Murdoch J. Gabbay, *http://www.gabbay.org.uk*

Consider a typical example of binding: $\lambda a.(aa)$. Here λa is a binder and determines a *scope* for its binding. Read linearly, the binding has three phases: 1) open the scope; 2) do something in it; 3) close the scope.

In the λ-calculus (for example) these three phases are combined into a monolithic syntactic construct $\lambda a.(\text{-})$. But there are instances where scope is best broken down, and its phases made explicit. Examples include memory allocation and deallocation in C-style languages, opening and closing files and network sockets, automata with resource allocation (and perhaps deallocation), and also modelling games in game semantics. It is then natural that scope can be *interleaved* — for instance, there is no real-life rule that we have to close files in the reverse order that we opened them. In the notation given below, the following are legal *dynamic sequences*:

$$\langle a\, \langle b\; b\rangle\; a\rangle \quad \text{and} \quad \langle a\, \langle b\; a\rangle\; b\rangle$$

We can draw this:



We want scope to be able to dangle, both into the 'future' and into the 'past', so we can sensibly talk about fragments of sequences: so $\langle a\, a$ and $a\, a\rangle$ are both fully legal syntax, as well as $a\langle a$ and $a\rangle a$. Also, matched pairs of brackets should alpha-convert: $\langle a\, \langle b\, a\, a\rangle\, b\rangle$ should be suitably equivalent to $\langle c\, \langle b\, c\, b\rangle\, c\rangle$, though *not* to $\langle a\, \langle a\, a\, a\rangle\, a\rangle$. In previous work we investigated this [GGP15]. We will now sketch:

(1) a new and hopefully simplified semantics for interleaved scope, and also
(2) a *nominal terms* style meta-syntax for interleaved scope (that is, a syntax for describing sequences with interleaved scope that includes variables ranging over sequence-fragments which may include danging $\langle a$ or $a\rangle$).

Thus we give a formal language within which to write axioms about dynamic sequences, and a semantics for it. This is hard, because we need to account for interleaved and danging scopes. This is significantly different from the case of the simple, well-nested, monolithic scope familiar from examples like $\lambda a.(\text{-})$. Nevertheless, our solution below seems pleasingly simple.

Definition 1 (Syntax).(1) Fix a countably infinite set of **atoms** $a, b, c, \dots \in \mathbb{A}$.
(2) A **permutation** $\pi \in \mathit{Perm}$ is a bijection on atoms. We write id for the identity permutation and $\pi \circ \pi'$ for composition of permutations (permutations are traditionally written in prefix notation, so $\pi \circ \pi'$ means 'first π', then π').
(3) Fix a countably infinite set of **unknowns** $X, Y, Z, \dots \in \mathbb{X}$ (disjoint from atoms). We call a pair $\pi{\cdot}X$ of a permutation and an unknown a **moderated unknown**; we may abbreviate $\mathrm{id}{\cdot}X$ to X.
(4) Fix two symbols P called **past** and F called **future**.
P and F will behave very much like a pair of unknowns, but formally we keep them separate so that $P, F \notin \mathbb{X}$.
(5) A **dynamic sequence-fragment** D is a finite sequence of elements from $\pi{\cdot}X$, a, c, $\langle a$, or $a\rangle$, where $a \in \mathbb{A}$ and $X \in \mathbb{X}$ and $\pi \in \mathit{Perm}$. We may write ϵ for the empty dynamic sequence-fragment.
(6) A **dynamic sequence** is a triple $s = \pi'{\cdot}P\, D\, \pi{\cdot}F$ where π' and π are permutations.
(7) A **freshness** is a set of pairs of the form: $a\#X$ for $a \in \mathbb{A}$ and $X \in \mathbb{X}$, or $a\#P$, or $a\#F$.
(8) A **freshness context** is a set of freshnesses.

Notation 2. We may write the dynamic sequence

$$\pi'{\cdot}P\, D\, \pi{\cdot}F \quad \text{as} \quad \pi'«D»\pi.$$

$$\begin{array}{lrl}
(\textbf{denot x}) & [\![x]\!]_\varsigma(p,f) = (p, x{::}f) & x\in\mathbb{A}\\
(\textbf{denot bra}) & [\![\langle a]\!]_\varsigma(p,f) = (p,[a]f) & \\
(\textbf{denot ket}) & [\![a\rangle]\!]_\varsigma(p,f) = ([a]p,f) & \\
(\textbf{denot X}) & [\![\pi{\cdot}X]\!]_\varsigma(p,f) = (p,(\pi{\cdot}\varsigma(X))\circ f) & \\
(\textbf{denot seq}) & [\![DE]\!]_\varsigma(p,f) = [\![D]\!]_\varsigma([\![E]\!]_\varsigma(p,f)) & \\
(\textbf{denot }\epsilon) & [\![\epsilon]\!]_\varsigma(p,f) = (p,f) & \\
(\textbf{denot D}) & [\![\pi'\text{«}\,D\,\text{»}\pi]\!]_\varsigma(p,f) = [\![D]\!]_\varsigma(\pi'{\cdot}p,\pi{\cdot}f) &
\end{array}$$

Fig. 1: Denotation

If π' or π is the identity, then we may omit it, thus writing D for id«$\,D\,$»id.

Definition 3 (Permutation action). Define a **permutation action** $\pi{\cdot}s$ such that π acts on atoms in s and $\pi{\cdot}(\pi'{\cdot}X) = (\pi\circ\pi'){\cdot}X$ and similarly for $\pi'{\cdot}P$ and $\pi'{\cdot}F$. Thus:

$$\pi{\cdot}\big(\mathrm{id}\text{«}\,a\ \langle a\ X\ b\rangle\,\text{»}\pi^{-1}\big) = \pi\text{«}\,\pi(a)\ \langle\pi(a)\ \pi{\cdot}X\ \pi(b)\rangle\,\text{»id}.$$

Definition 4 (Composition). Suppose $s_1 = \pi_1'\text{«}\,D_1\,\text{»}\pi_1$ and $s_2 = \pi_2'\text{«}\,D_2\,\text{»}\pi_2$ are two dynamic sequences. Define their **composition** $s_1\circ s_2$ by

$$s_1\circ s_2 = (\pi_2'\circ\pi_1')\text{«}\,(\pi_2'{\cdot}D_1)\circ(\pi_1{\cdot}D_2)\,\text{»}(\pi_1\circ\pi_2).$$

Definition 5.(1) A **sequence (of atoms)** is a sequence of atoms $l = (l_0, l_1, l_2, \dots)$. Write $\mathbb{L}$ for the set of sequences of atoms.

(2) A **sequence with binding** is an α-equivalence class $[l']l = [(l', l)]_\alpha$ where the atoms in l' are distinct and are considered to be bound in l.[1]
Write $\mathbb{S}$ for the set of sequences with binding.

(3) Given $S = [l']l \in \mathbb{S}$ and $a \notin l'$ write $[a]S$ for $[a :: l']l$. Intuitively this is the sequence obtained by binding a in $[l']l$.

Definition 6. A **valuation** ς is a map from unknowns X to elements $\varsigma(X)\in\mathbb{S}$.
Define a **denotation**

$$[\![D]\!]_\varsigma : \mathbb{S}^2 \to \mathbb{S}^2 \quad\text{and}\quad [\![s]\!]_\varsigma : \mathbb{S}^2\to\mathbb{S}^2$$

of dynamic sequence-fragments and dynamic sequences by the rules in Figure 1.

Remark 7. Rule (**denot seq**) can be rewritten as $[\![D\ E]\!]_\varsigma = [\![D]\!]_\varsigma\circ[\![E]\!]_\varsigma$, where $\circ$ denotes function composition.

Lemma 8. *The denotation of Figure 1 is compositional in the sense of Definition 4. In symbols:*

$$[\![DD']\!]_\varsigma = [\![D]\!]_\varsigma\circ[\![D']\!]_\varsigma,$$

or equivalently: $[\![DD']\!]_\varsigma(p,f) = [\![D]\!]_\varsigma([\![D']\!]_\varsigma(p,f))$.

Proof. By a routine induction on D' using (**denot seq**) and Remark 7. □

Definition 9. Call a relation $\mathcal{R}$ on dynamic sequences **compositional** when $s_1\ \mathcal{R}\ s_1'$ implies $s_1\circ s_2\ \mathcal{R}\ s_1'\circ s_2$ and $s_2\circ s_1\ \mathcal{R}\ s_2\circ s_1'$.

Definition 10. Define **alpha-equivalence** on dynamic sequences to be the least compositional relation satisfying the rules in Figure 2.[2]

[1] We can define $=_\alpha$ on $\mathbb{L}\times\mathbb{L}$ formally as follows: $(l', l) =_\alpha (m', m)$ when there exists some n' with $n'\#l', l, m', m$ such that $(n'\ l'){\cdot}l = (m'\ l'){\cdot}m$, where $n'\#l', l, m', m$ means that the atoms in n' are disjoint from those of l', l, m', and m, and $(n'\ l')$ is the permutation swapping the ith element of n' with the ith element of l'.

[2] Actually this is not quite true: composition is also subject to freshness constraints given by the freshness context. Developing this is beyond the scope of this brief abstract; but see [UPG04].

$$\begin{array}{lll}
(\mathbf{alphaR}) & b\#F \vdash \langle a = \langle b \text{»}(b\,a) & \\
(\mathbf{alphaL}) & b\#P \vdash a\rangle =_\alpha (b\,a)\text{«}\, b\rangle & \\
(\mathbf{permX}) & \Delta \vdash \pi\cdot X =_\alpha \pi'\cdot X & \forall a{\in}\mathbb{A}.(\pi(a) \neq \pi'(a) \Rightarrow a\#X \in \Delta) \\
(\mathbf{permP}) & \Delta \vdash \pi\text{«}\epsilon\text{»}\mathrm{id} =_\alpha \pi'\text{«}\epsilon\text{»}\mathrm{id} & \forall a{\in}\mathbb{A}.(\pi(a) \neq \pi'(a) \Rightarrow a\#P \in \Delta) \\
(\mathbf{permF}) & \Delta \vdash \mathrm{id}\text{«}\epsilon\text{»}\pi =_\alpha \mathrm{id}\text{«}\epsilon\text{»}\pi' & \forall a{\in}\mathbb{A}.(\pi(a) \neq \pi'(a) \Rightarrow a\#F \in \Delta)
\end{array}$$

Fig. 2: alpha-equivalence

Remark 11. Definition 10 and the syntax underlying it is clearly based on *nominal terms* [UPG04]. The conditions on $(\mathbf{permX})$, $(\mathbf{permP})$, and $(\mathbf{permF})$ are a little long to write out but are actually very simple: if the differences between π and π' are fresh for (not free in) X, P, or F respectively, then the action of π will be equal to the action of π'. We have co-opted the structure of nominal unknowns to represent past and future, via P and F.

Definition 12.(1) If $S = [l']l \in \mathbb{S}$ then write $a\#S$ when $a \notin l \backslash l'$ (abusing sets notation for lists); in words, a is **fresh for** S when a does not occur free in S.
(2) Write $\varsigma \vDash \Delta$ when $a\#\varsigma(X)$ for every $a\#X \in \Delta$.
(3) If $p, f \in \mathbb{S}$ then write $\varsigma, p, f \vDash \Delta$ when $\varsigma \vDash \Delta$, and futhermore $a\#p$ for every $a\#P \in \Delta$ and $a\#f$ for every $a\#F \in \Delta$.

Theorem 13 (Soundness). *If* $\varsigma, p, f \vDash \Delta$ *and* $\Delta \vdash s =_\alpha t$ *then* $[\![s]\!]_\varsigma = [\![t]\!]_\varsigma$.

Proof. We consider just the case of $(\mathbf{alphaR})$. Suppose $b\#f$. Then

$$[\![\langle a]\!]_\varsigma(p, f) = (p, [a]f) \quad \text{and} \quad [\![\langle b \text{»}(b\;a)]\!]_\varsigma(p, f) = (p, [b](b\;a)\cdot f).$$

Because $b\#f$, it is a fact of alpha-equivalence in $\mathbb{S}$ that $[b](b\;a)\cdot f = [a]f$. □

Example 14. In [GGP15, Definition 6] alpha-equivalence was expressed using a non-local rule that involved judgements having to do with balancing brackets. Essentially, alpha-equivalence in that paper contained a PDA to calculate balanced brackets.

Rules $(\mathbf{alphaR})$ and $(\mathbf{alphaL})$ in Figure 2 work together to model the same alpha-equivalence, but these rules are local and no judgement is required in the meta-language aside from nominal terms style freshness $a\#X$.

Also, Definition 1 is an explicit nominal terms style syntax with unknowns; so for instance we can now think about unification, rewriting, and even enriching with explicit quantifiers over X to obtain a first-order logic for reasoning on dynamic sequences.

Example 15. We illustrate our syntax by formally specifying a pair of **scope-extrusion** rules as a nominal algebra specification. Below, ν is some constant symbol, which we can easily add to syntax:

$$b\#X \vdash b\rangle\, X = X\; b\rangle \quad \text{and} \quad b\#X \vdash X\; \nu\langle b = \nu\langle b\; X$$

There is plenty more to do: examples; a fuller treatment of alpha-equivalence; a sound theory of substitution of terms for unknowns; full theories of unification, rewriting and algebra; completeness; applications, including to linked linear structures (of which there are many); generalisations to non-linear stuctures, logics based on the term syntax and semantics; and so forth.

REFERENCES

[GGP15] Murdoch J. Gabbay, Dan R. Ghica, and Daniela Petrisan, *Leaving the Nest: Nominal Techniques for Variables with Interleaving Scopes*, 24th EACSL Annual Conference on Computer Science Logic (CSL 2015) (Dagstuhl, Germany) (Stephan Kreutzer, ed.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 41, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015, pp. 374–389.

[UPG04] Christian Urban, Andrew M. Pitts, and Murdoch J. Gabbay, *Nominal Unification*, Theoretical Computer Science **323** (2004), no. 1–3, 473–497.