Finitely partitioned forests (work in progress)

Paul Blain Levy, University of Birmingham, UK

Abstract

For a first-order language with full ground references and iteration, we give an explicit denotational semantics. A type denotes a "platform family", consisting of "platforms" that indicate the references included. A state is represented as a forest, showing the result of following chains of references, whose nodes are partitioned into finitely many equivalence classes, showing the aliasing. The model is equivalent to known models that use ends or coends, or strong nominal sets, but is simpler in the sense that a computation simply denotes a function.

1 Introduction

The language feature of "full ground state" allows storage of e.g. pointers to pointers to integers, but not storage of code. For a higher-order language with full ground state, the literature provides two kinds of denotational semantics: a game model [3] and a possible world model [2]. Although these are very different models, it turns out that they are essentially the same at first-order types.

The goal of this work is to reformulate the (common) model of the first-order language in an explicit way, to make clear that a computation simply denotes a *function*, rather than a relation that is functional up to renaming (as in nominal game semantics) or a natural transformation (as in presheaf semantics).

2 The language

Our type syntax is organized as follows, cf. [2]. A set $\mathbf{C}$ of *cell sorts* yields the set of types (or "full ground types") over C via

$$A ::= \quad 0 \mid A + A \mid \sum_{i \in \mathbb{N}} A_i \mid 1 \mid A \times A \mid X \mid \mathtt{rec}\ X.\, A \mid \mathtt{ref}_c\ (c \in \mathbf{C})$$

where $\mathtt{ref}_c$ is intended to be the type of c-sorted cells. A *syntactic full ground signature* consists of a set $\mathbf{C}$ and, for each $c \in C$, a closed type $\mathsf{ctype}(c)$, called the *content type*. It is intended to be the type of values stored in a c-sorted cell.

Using fine-grain call-by-value, the judgements are $\Gamma \vdash^{\mathsf{v}}_w V : A$ for values, and $\Gamma \vdash^{\mathsf{c}}_w M : A$ for computations. Here w is a *world* or sort-list; it indicates that the available cells are $0, \ldots, n-1$, where n is the length of w. The term syntax (omitted) includes reading, writing, cell generation with specified initial values and equality testing. We also include iteration, so evaluation is partial. Some helpful extra judgements:

- An *extended value* $\Gamma \vdash^{\mathsf{xv}}_w (x, V) : A$ consists of a sort-list x, and value V in world $w +\!\!+ x$.
- An *extended state* $\vdash^{\mathsf{xs}}_w (x, s)$ consists of a sort-list x, and state s of the world $w +\!\!+ x$.
- A *stateful value* $\vdash^{\mathsf{xsv}}_w (x, s, V) : A$ consists of a sort-list x, and state s and value V in world $w +\!\!+ x$.
- A *stateful computation* $\vdash^{\mathsf{xsc}}_w (x, s, M) : A$ consists of a sort-list x, and state and computation in world $w +\!\!+ x$.

3 Platforms and platform families

We say that a *platform* over $\mathbf{C}$ is a finite set R equipped with a sort function $p : R \to \mathbf{C}$. We write Inj for the category of platforms and sort-preserving injections, and $\mathbf{PInj}$ for that of platforms and sort-preserving partial injections.

Each world w denotes a platform, and each type a platform family $(R_i)_{i \in I}$. For example, the type $\mathtt{ref}_c \times \mathtt{ref}_c$ has two kinds of value: $\langle l, l \rangle$, for a c-sorted cell l, and $\langle l, l' \rangle$, for distinct c-sorted cells l, l'. So this type denotes

a platform family with two indices, one giving a singleton platform (a c-sorted cell), and the other a doubleton platform (two c-sorted cells).

Given platforms R and S, a *partition* is a sort-respecting equivalence relation E on $R + S$ that restricts to the equality relation on each component; we write $\mathsf{quot}(E)$ for the (finite) set of equivalence classes. The set of all partitions is written $\mathsf{Partit}(R, S)$—it can be identified with $\mathbf{PInj}(R, S)$. The product of platform families $(R_i)_{i\in I}$ and $(S_j)_{j\in J}$ is given by

$$(R_i)_{i\in I} \times (S_j)_{j\in J} \stackrel{\text{def}}{=} (\mathsf{quot}(E))_{i\in I, j\in J, E\in \mathsf{Partit}(R_i, S_j)}$$

Given a platform T and platform family $S = (S_j)_{j\in J}$, we define sets $\mathsf{Val}_T(S)$ and $\mathsf{xVal}_T(S)$ as semantic domains for closed values $\vdash^{\mathsf{v}}_w V : A$ and closed extended values $\vdash^{\mathsf{xv}}_w (x, V) : A$ respectively, where $[\![w]\!] = T$ and $[\![A]\!] = S$. This is done via $\mathsf{Val}_T(S) \stackrel{\text{def}}{=} \sum_{j\in J} \mathsf{Inj}(S_j, T)$ and $\mathsf{xVal}_T(S) \stackrel{\text{def}}{=} \sum_{j\in J} \mathbf{PInj}(S_j, T)$. For a pair $r = \langle j, E\rangle \in \mathsf{xVal}_T(S)$ its quotient set is given by $\mathsf{Quot}(r) \stackrel{\text{def}}{=} \mathsf{Quot}(E)$. If r is the denotation of (x, V), this represents all the cells in $w+\!\!+x$ that appear in V.

4 Finitely partitioned forests

A *semantic full ground signature* consists of a set $\mathbf{C}$ and, for each $c\in C$, a platform family $\mathsf{cpf}(c)$ called the *content platform family* of c. Thus a syntactic signature denotes a semantic one. Our goal is to construct a model of computations depending on the semantic signature.

Firstly consider an extended state $\vdash^{\mathsf{xs}}_w (x, s)$ where $[\![w]\!] = T$. To give its denotation, only reachable x-cells should appear (the frugality condition in [3]), and moreover they should have *no name and no order*. We achieve this by giving two pieces of data.

1. To represent all the tags and the local aliasing (i.e. aliasing between cells stored in the same cell, or between cells mentioned in V), we give a *forest* σ, i.e. a strategy for the following game between P and O. Firstly O plays $l_0 \in R$ of sort c_0 with content platform family $(S_i)_{i\in I}$, and P replies with $i_0 \in I$. Then O plays $l_1 \in S_{i_0}$ of sort c_1 with content platform family $(S'_j)_{j\in J}$ and P replies with $i_1 \in J$. And so on.

   A play is represented as a sequence $l_0 : c_0, i_0, l_1 : c_1, i_1, \ldots$, and the forest σ as a set of passive-ending (i.e. even-length) plays that is prefix-closed and contains ε.

2. For each passive-ending play s of σ, we write M_s for the platform of possible next O-moves. To represent the global aliasing, we give a *partition* E on $(M_s)_{s\in\sigma}$, i.e. a sort-respecting equivalence relation on $\sum_{s\in\sigma} M_s$ that is equality on each component. Equivalent O-moves must be followed by the same P-move, and the corresponding next O-moves are again equivalent. E must be *finite* i.e. have only finitely many equivalence relations, as there are only finitely many cells.

We call $t = (\sigma, E)$ a *finitely partitioned forest* on R, and its quotient is written $\mathsf{quot}(t) \stackrel{\text{def}}{=} \mathsf{quot}(E)$, representing all the reachable cells. The set of all finitely partitioned forests is written $\mathsf{FPForest}(R)$. The semantic signature is omitted as it has been fixed.

Given a platform R and platform family S, we define a set $\mathsf{xsVal}_T(S)$ as a semantic domain for stateful values $\vdash^{\mathsf{xsv}}_w (x, s, V) : A$, where $[\![w]\!] = T$ and $[\![A]\!] = S$. This is done via $\mathsf{xsVal}_T(S) \stackrel{\text{def}}{=} \sum_{r\in \mathsf{xVal}_T S} \mathsf{FPForest}(\mathsf{Quot}(r))$. Likewise a stateful computation $\vdash^{\mathsf{xsc}}_w (x, s, M) : A$ denotes an element of $\mathsf{xsVal}_T(S)_\perp$, as $w+\!\!+x, s, M$ either diverges or evaluates to something of the form $w+\!\!+x', s', V$.

5 Semantics of judgements

Given a platform $T = [\![w]\!]$ and platform families $R = [\![\Gamma]\!]$ and $S = [\![A]\!]$,

- a value $\Gamma \vdash^{\mathsf{v}}_w V : A$ should denote an element of $\mathsf{Val}_T(R, S) \stackrel{\text{def}}{=} \prod_{r\in \mathsf{xVal}_T(R)} \mathsf{Val}_{\mathsf{Quot}(r)}(S)$
- an extended value $\Gamma \vdash^{\mathsf{xv}}_w (x, V) : A$ an element of $\mathsf{xVal}_T(R, S) \stackrel{\text{def}}{=} \prod_{r\in \mathsf{xVal}_T(R)} \mathsf{xVal}_{\mathsf{Quot}(r)}(S)$
- a computation $\Gamma \vdash^{\mathsf{c}} M : A$ an element of $\mathcal{K}_T(R, S) \stackrel{\text{def}}{=} \prod_{t\in \mathsf{xsVal}_T(R)} \mathsf{xsVal}_{\mathsf{Quot}(t)}(S)_\perp$.

For each platform T, we hope to obtain a distributive Freyd category $\mathsf{Val}_T \to \mathcal{K}_T$ with uniform Elgot structure. The same categorical structure is used for a different purpose in [1].

References

[1] Jad Elkhaleq Ghalayini and Neel Krishnaswami. The denotational semantics of SSA, 2024.

[2] Ohad Kammar, Paul Blain Levy, Sean K. Moss, and Sam Staton. A monad for full ground reference cells. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017.

[3] A. S. Murawski and N. Tzevelekos. Algorithmic games for full ground references. In *ICALP (2)*, volume 7392 of *LNCS*, pages 312–324, 2012.