Control Structures II : Naming monoids

Robin Milner
Edinburgh, July 1994

D R A F T 28/7/94. Comments are welcome.

1 Names in control structures

1.1 Control structure morphisms In control structures the names X are used both in data $\langle x\rangle$ and in the abstractors $\mathbf{ab}_x$. In the initial definition of control structures [1], the names X have no structure; X is just a set. Also a morphism $\Phi : A \to B$ of control structures is constrained to act as the identity upon X. Yet there is practical motivation to let Φ treat names more generally. For example a simple name x in A may be implemented by a composite name $\Phi(x) = y_1 \cdots y_n$ in B; if A is the π-calculus, this would allow a single communication in A to be modelled by several communications in B.

1.2 Naming monoids Even as control structures stand, it is natural to consider their names not to be just X, but to be the free monoid $(X^{\otimes}, \otimes, 1)$ generated by X. For we can extend both data and abstractors to the free monoid simply as follows:

$$\begin{array}{rclcrcl} \langle 1\rangle & \stackrel{\text{def}}{=} & \mathbf{id}_\epsilon & \qquad & \langle x\otimes y\rangle & \stackrel{\text{def}}{=} & \langle x\rangle \otimes \langle y\rangle \\ \mathbf{ab}_1 a & \stackrel{\text{def}}{=} & a & & \mathbf{ab}_{x\otimes y} a & \stackrel{\text{def}}{=} & \mathbf{ab}_x \mathbf{ab}_y a\ . \end{array}$$

This indicates that monoidal structure for names is in complete harmony with control structures, so we immediately ask: Which monoids, other than the free ones, are suitable name structures?

1.3 Names and arities We risk making control structures excessively complex if we allow the arity monoid and the name monoid to vary independently. John Power argues that the name structure should *arise from* the arity monoid in some uniform way, and with Claudio Hermida he is investigating what this way should be.

In full agreement with this view, I here try to answer the question: What conditions should the name monoid satisfy, to preserve important properties enjoyed by control structures at present? It turns out that this question has a good answer without considering arities at all. Thus the answer will help to determine both the class of admissible arity monoids, and the way in which arities uniformly give rise to names.

1.4 Axioms of control structures The first indication that care is needed in defining name structure is the presence of a side condition in two axioms:

$$\begin{array}{lcll} \mathrm{S7} & : & (c\otimes\mathbf{id})\cdot\mathbf{ab}_y\mathbf{ab}_x a = \mathbf{ab}_x\mathbf{ab}_y a\cdot(c\otimes\mathbf{id}) & (x\neq y) \\ \gamma_1 & : & \mathbf{ab}_x\langle y\rangle = \mathbf{id}\otimes\langle y\rangle & (x\neq y)\ . \end{array}$$

If x and y can be complex names – even in the free monoid – then these axioms need adjustment. For example we may have $x \neq y$ and yet x and y share a common factor, say $x = uw$ and $y = vw$. (From now on we shall use concatenation to represent the product of names.) Then we cannot expect γ_1 to hold; for the left side is $\mathbf{ab}_u\mathbf{ab}_w(\langle v\rangle \otimes \langle w\rangle)$ while the right side is $\mathbf{id}\otimes\langle v\rangle\otimes\langle w\rangle$, so w occurs bound on one side and free on the other. The natural adjustment is to expect the side condition to be replaced by a stronger condition $x \# y$ called *separation*, which will coincide with $x \neq y$ in the case of atomic names.

Thus we expect every naming monoid to be equipped with a binary separation relation. It will be convenient to define

$$\#x \stackrel{\text{def}}{=} \{y \mid x \# y\} .$$

Later we shall need to impose further conditions upon $\#$; for now, we just take it to be symmetric.

1.5 Surface properties The first properties of control structures which we aim to preserve in generalising the name structure (though there may be more) are concerned with the *surface* of an action, defined by

$$\mathbf{surf}\, a \stackrel{\text{def}}{=} \{x \mid \mathbf{ab}_x a \neq \mathbf{id} \otimes a\} .$$

The most striking property is to do with the surface of an abstraction:

$$\mathbf{surf}\,\mathbf{ab}_y a = \mathbf{surf}\, a - \{y\} .$$

We shall find it more convenient here to work with the *exterior* of a, i.e. the complement of $\mathbf{surf}\, a$ given by

$$\#a \stackrel{\text{def}}{=} \{x \mid \mathbf{ab}_x a = \mathbf{id} \otimes a\} ,$$

and the property we wish to preserve is

$$\#\mathbf{ab}_y a = \#a \cup \{y\} .$$

Now this is only correct for atomic names; for if $y = y_1 y_2$ say, then we expect $\#\mathbf{ab}_y a$ to contain not only y but also y_1 and y_2. We shall see in Section 4 that the correct property is

$$\#\mathbf{ab}_y a = (\#a \cup \{y\}^\wedge)^\otimes ,$$

where the operations $(\ )^\wedge$ (downward closure) and $(\ )^\otimes$ (product closure) are appropriately defined.

1.6 Outline We summarise our goal as follows: Once the axioms S7 and γ_1 are adjusted to have the side condition $x \# y$, define a *naming* to be a monoid equipped with a separation relation $\#$ satisfying conditions which ensure that suitably generalised surface properties hold.

Section 2 defines namings. Section 3 defines *elementary namings*, those generated by certain elementary names, and isolates an important subclass called *sharp namings*. Several examples of namings appear in Section 4. In Section 5, a theorem is proven about closure operations over sets of names. Finally, Section 6 demonstrates that namings do indeed secure the surface properties.

2 Namings

Throughout, we deal with a monoid $(X, \otimes, 1)$; when there is no confusion we refer to it as X. We let u, v, w, x, y, z range over X, and we shall usually write xy for $x \otimes y$.

2.1 Definition (Separation, s-monoid) Let $\#$ be a symmetric binary relation over X. Then it is called a *separation* for X if it satisfies

S1:	$x \# 1$
S2:	$x \# yz$ iff ($x \# y$ and $x \# z$)
S3:	$x \# x$ implies $x = 1$.

In this case $(X, \#)$ is called an *s-monoid*.

2.2 Definition (Factorisation) Let $x = x_1 \otimes \cdots \otimes x_r$ (which is taken to be 1 when $r = 0$). Then the sequence $\vec{x} = (x_1, \ldots, x_r)$ is called a *factorisation* of x, and we define $|\vec{x}| \stackrel{\text{def}}{=} x_1 \otimes \cdots \otimes x_r$. If $\vec{y}$ is another factorisation, we write $\vec{x} \succ \vec{y}$ and say that $\vec{y}$ *refines* $\vec{x}$ if $\vec{y}$ is a concatenation of factorisations of $x_1, \ldots, x_r$, i.e. $\vec{y} = (\vec{y}_1, \ldots, \vec{y}_r)$ and $|\vec{y}_i| = x_i$ $(1 \leq i \leq r)$. We write $\vec{x} \# \vec{y}$ if $x \# y$ for every $x \in \vec{x}$ and $y \in \vec{y}$.

2.3 Fact In an s-monoid it follows from S2 that if $u \# v$, $u \succ \vec{x}$ and $v \succ \vec{y}$ then $\vec{x} \# \vec{y}$.

2.4 Definition (Confluence) We say that X is *confluent* if

C: Whenever $|\vec{x}| = |\vec{y}|$ there exists $\vec{z}$ such that $\vec{x} \succ \vec{z}$ and $\vec{y} \succ \vec{z}$.

2.5 Proposition A monoid X is confluent iff the following two conditions hold:

C1: If $x_1 x_2 = 1$ then $x_1 = 1$ and $x_2 = 1$.
C2: Whenever $x_1 x_2 = y_1 y_2$ there exists z such that
either ($x_1 = y_1 z$ and $z x_2 = y_2$) or ($y_1 = x_1 z$ and $z y_2 = x_2$) .

Proof To show C implies C1, assume $x_1 x_2 = 1$. Then $x_1 x_2$ has two factorisations: (x_1, x_2) and $()$ (the empty sequence). But the only refinement of $()$ is itself, therefore $(x_1, x_2) \succ ()$ by C. Hence $x_1 \succ ()$ and $x_2 \succ ()$, so $x_1 = x_2 = |()| = 1$. To show C implies C2, just apply C in the case that the length of both $\vec{x}$ and $\vec{y}$ is 2.

To prove C from C1 and C2, proceed by induction on the sum of the lengths of $\vec{x}$ and $\vec{y}$. If either has length 0 then apply C1. If either has length 1 then take $\vec{z}$ to be the other. The inductive step is routine. ∎

2.6 Definition (Subname) We say y is a *subname* of x, and write $y \leq x$, if there are names x_1 and x_2 for which $x = x_1 y x_2$.

2.7 Definition (Honesty) We say that a factorisation $\vec{x}$ is *honest* if $x = x'$ or $x \# x'$ for all x, x' in $\vec{x}$. We say that the s-monoid $(X, \#)$ is *honest* if

H: For all x_1 and x_2 there is an honest $\vec{y}$ such that $(x_1, x_2) \succ \vec{y}$.

2.8 Definition (Naming) An s-monoid $(X, \#)$ which is honest and confluent is called a *naming*.

2.9 Proposition In a naming, $x \# y$ iff the only subname of both x and y is 1.

Proof ($\Longrightarrow$) Assume $x \# y$, and let z be a common subname, i.e. $x = x_1 z x_2$ and $y = y_1 z y_2$. Then $z \# z$ by S2, and $z = 1$ follows from S3.

($\Longleftarrow$) By H, there exist $\vec{u}$ and $\vec{v}$ such that $x \succ \vec{u}$, $y \succ \vec{v}$ and $(\vec{u}, \vec{v})$ is honest. But by assumption, 1 is the only possible common member of $\vec{u}$ and $\vec{v}$, hence by honesty $u \# v$ for every $u \in \vec{u}$ and $v \in \vec{v}$. It follows by S1 and S2 that $x \# y$. ∎

Thus, in the presence of honesty, the separation relation $\#$ is definable in terms of the monoidal product. However, there seems to be no convenient equivalent axiom to H which does not use $\#$.

2.10 Remark Clearly, (x) is trivially an honest factorisation of x. But suppose x has another factorisation, say $x \succ \vec{x}$; then can we refine $\vec{x}$ further into an honest factorisation? The fact that we can do so is by no means obvious, and is the first non-trivial theorem about namings. We state the theorem now, and devote the remainder of this section to proving it:

2.11 Theorem Let $\vec{x}$ be any factorisation in a naming. Then there is an honest $\vec{y}$ such that $\vec{x} \succ \vec{y}$. ∎

We proceed to the proof via a number of auxiliary results.

2.12 Lemma (Honest refinement) Let $(\vec{x}, \vec{y})$ and $\vec{y'}$ be honest, and let $\vec{y} \succ \vec{y'}$. Then there exists $\vec{x'}$ such that $\vec{x} \succ \vec{x'}$ and $(\vec{x'}, \vec{y'})$ is honest.

$$\begin{array}{ccc} \boxed{\vec{x},\ \vec{y}} & & \vec{x} \\ \downarrow & \Longrightarrow & \downarrow \\ \boxed{\vec{y'}} & & \boxed{\vec{x'},\ \vec{y'}} \end{array}$$

(a box means "honest"; an arrow means "refinement")

Proof Form $\vec{x'}$ from $\vec{x}$ as follows. Let z be any member of $\vec{x}$. If z is not in $\vec{y}$, then $z \# \vec{y}$ and so $(z, \vec{y'})$ is honest; in this case leave all occurrences of z unchanged in forming $\vec{x'}$. If z is in $\vec{y}$ then $z \succ \vec{z}$ for some subsequence $\vec{z}$ of $\vec{y'}$, so $(\vec{z}, \vec{y'})$ is honest; in this case replace all occurrences of z in $\vec{x}$ by $\vec{z}$ in forming $\vec{x'}$.

Once all members z of $\vec{x}$ have been treated in this way, $\vec{x'}$ will have the desired properties. ∎

2.13 Lemma (Honest substitution) Let $\vec{x}$ and $\vec{y}$ be honest, and let $y \succ \vec{y}$. Let $\vec{x'}$ result by replacing every occurrence of y as a member of $\vec{x}$ by $\vec{y}$. Then $\vec{x'}$ is honest.

Proof We need only note that if $z \neq y$ is any other member of $\vec{x}$ then $z \# y$, so $z \# \vec{y}$. ∎

2.14 Proposition (Honest confluence) Let $\vec{x}$ and $\vec{y}$ be honest and $|\vec{x}| = |\vec{y}|$. Then there exists honest $\vec{z}$ such that $\vec{x} \succ \vec{z}$ and $\vec{y} \succ \vec{z}$.

Proof By induction on the length of $\vec{x}$. If the length is 0, then the result is immediate. Otherwise assume the inductive hypothesis for $\vec{x}$, and consider honest $(x, \vec{x})$ and $\vec{y}$. Using confluence, we can express $\vec{y}$ as follows:

$$\vec{y} = (\vec{y_1}, w, \vec{y_2}) \text{ and } w = w_1 w_2, \text{ with } |x| = |(\vec{y_1}, w_1)| \text{ and } |\vec{x}| = |(w_2, \vec{y_2})| \ .$$

By H, there exist $\vec{w_1}$ and $\vec{w_2}$ such that $w_1 \succ \vec{w_1}$, $w_2 \succ \vec{w_2}$ and $(\vec{w_1}, \vec{w_2})$ is honest. By Lemma 2.13, if we replace every occurrence of the member w in $\vec{y}$ by $(\vec{w_1}, \vec{w_2})$, we obtain

$$\vec{y} \succ \vec{y'} = (\vec{y_1'}, \vec{w_1}, \vec{w_2}, \vec{y_2'}) \text{ honest, where } x \succ (\vec{y_1'}, \vec{w_1}) \text{ and } |\vec{x}| = |(\vec{w_2}, \vec{y_2'})| \ .$$

Now by induction, there exists honest $\vec{z_2}$ with $\vec{x} \succ \vec{z_2}$ and $(\vec{w_2}, \vec{y_2'}) \succ \vec{z_2}$. By Lemma 2.12 we also find $\vec{z_1}$ such that

$$(\vec{y_1'}, \vec{w_1}) \succ \vec{z_1} \text{ and } \vec{z} \stackrel{\text{def}}{=} (\vec{z_1}, \vec{z_2}) \text{ is honest.}$$

By construction $(x, \vec{x}) \succ \vec{z}$ and $\vec{y} \succ \vec{z}$, so we are done. ∎

We are now ready to prove Theorem 2.11.

2.15 Proof of Theorem 2.11 We have to find an honest refinement of an arbitrary factorisation $\vec{x}$. The proof is by induction on the length of $\vec{x}$. For length 0 it is trivial. Consider then $(x, \vec{x})$.

First, let $x' = |\vec{x}|$; then by H there exist $\vec{u}$ and $\vec{v}$ with $x \succ \vec{u}$, $x' \succ \vec{v}$ and $(\vec{u}, \vec{v})$ honest. Second, by induction, there is an honest $\vec{z}$ such that $\vec{x} \succ \vec{z}$.

Now $\vec{v}$ and $\vec{z}$ are both honest refinements of x', and hence by Proposition 2.14 they have a common honest refinement $\vec{y_2}$.

Since $(\vec{u}, \vec{v})$ is honest and $\vec{v} \succ \vec{y_2}$, by Lemma 2.12 there exists $\vec{y_1}$ for which $\vec{u} \succ \vec{y_1}$ and $\vec{y} \stackrel{\text{def}}{=} (\vec{y_1}, \vec{y_2})$ is honest. By construction $\vec{y}$ refines $(x, \vec{x})$, and we are done. ∎

3 Elementary namings

One of the simplest namings is the free monoid generated by a set of *atomic* names; each atom has only itself and 1 as factors. There is some value in considering a naming in which there is a set E of *elementary names* in terms of which all others are expressible, but which may themselves have proper factors.

3.1 Definition (Basis) If $(M, \otimes, 1)$ is any monoid, then a subset E of M is a *basis* for M if it does not contain 1 and if, for all $m \in M$, there exist $e_1, \ldots, e_r \in E$ $(r \geq 0)$ such that $m = e_1 \otimes \cdots \otimes e_r$.

We shall use d, e, f, g, h to range over a basis E.

3.2 Definition (Elementary naming) Let E be a basis for the s-monoid $(X, \#)$. Then $(X, \#, E)$ is an *elementary naming*, and we call the members of E its *elements* or *elementary names*, if the following hold:

H_E:	For all $\vec{e}$ there is an honest $\vec{f}$ such that $\vec{e} \succ \vec{f}$.
C_E:	Whenever $\lvert\vec{e}\rvert = \lvert\vec{f}\rvert$ there exists $\vec{g}$ such that $\vec{e} \succ \vec{g}$ and $\vec{f} \succ \vec{g}$.

We now show that every naming yields an elementary naming and vice versa.

3.3 Proposition (1) If $(X, \#)$ is a naming, then $(X, \#, X - \{1\})$ is an elementary naming. (2) If $(X, \#, E)$ is an elementary naming then $(X, \#)$ is a naming.

Proof (1) We have to derive H_E and C_E from H and C, in the case $E = X - \{1\}$. This is straightforward using the basis property and Theorem 2.11, with the fact that 1 has the factorisation (). (2) To derive H and C from H_E and C_E is even easier. ∎

We now show that separation can be characterized in terms of elementary subnames.

3.4 Proposition $x \# y$ iff x and y have no common elementary subname.

Proof Direct from Proposition 2.9, noting that 1 is not an element. ∎

3.5 Alternative axioms By placing constraints upon elements, we can find special classes of namings. One particular class, the *sharp* elementary namings, will be of particular interest.

For this development it is convenient to find another formulation of condition H_E. Define two special cases of the condition, as follows:

H^1_E:	Whenever $e \succ \vec{e}$ there is an honest $\vec{f}$ such that $\vec{e} \succ \vec{f}$.
H^2_E:	For all e_1, e_2 there is an honest $\vec{f}$ such that $(e_1, e_2) \succ \vec{f}$.

In the following theorem, we claim that the conjunction of these two conditions has the same strength as H_E. The proof is non-trivial, and is done via three lemmas at the end of this section.

3.6 Theorem Let $(X, \#, E)$ be an s-monoid satisfying C_E. Then it is a elementary naming iff it satisfies conditions H^1_E and H^2_E. ∎

3.7 Remark Let us consider the relative power of the axioms H_E, H^1_E and H^2_E. Comparing H^2_E with H, and particularly in view of Theorem 2.11, we may expect that H^2_E and C_E together imply H_E, and this would render H^1_E superfluous. This may be so, but I do not see a proof; in particular, if we try to prove H_E from H^2_E along the lines of Theorem 2.11, we run into difficulty in proving the analogue of honest confluence (Proposition 2.14). This question remains open.

However, H^1_E and C_E together do *not* imply H_E. Consider the free s-monoid generated by two distinct elements, e and f, and take the separation $\#$ to be minimal – i.e. $x \# y$ iff $x = 1$ or $y = 1$. It is easy to check that H^1_E and C_E hold; but (e, f) has no honest refinement since $e \# f$ does not hold, so H^2_E fails.

3.8 Definition (Sharpness) The factorisation $(x_1, \ldots, x_k)$ is *sharp* if whenever $i \neq j$ then $x_i \# x_j$. A name x is *sharp* if all its factorisations are sharp. A basis E is *sharp* if all its members are sharp. An elementary naming is *sharp* if its basis is sharp.

This definition shows the benefit of considering elements and elementary namings. It makes no sense to constrain a naming by requiring that *all* its names x are sharp: just consider $x = yy$. But it is very natural to consider namings generated by sharp elements; a sharp name is just one which contains no repeated factor, so in a sharp elementary naming, repetitions are only obtained by taking products of elements. In fact, the elementary namings which are of interest in control structures may turn out to be sharp.

If E is a sharp basis, then H^1_E is automatically secured. Therefore we have the following:

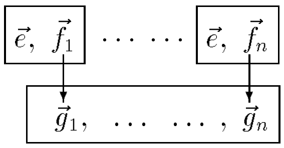
3.9 Proposition Let an s-monoid $(X, \#)$ have a sharp basis E. Then if it also satisfies H^2_E and C_E, it is a (sharp) elementary naming. ■

Thus, if we have sharpness and confluence, we only have to check the binary honesty condition H^2_E to ensure that we have a naming.

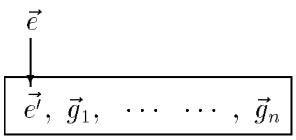
3.10 Remark Sharpness does not *imply* the condition H^2_E; for it can be checked that the previous counter-example 3.7 has a sharp basis.

We now prove Proposition 3.6. For each of the following three lemmas, we assume an s-monoid $(X, \#)$ with basis E, satisfying no other axioms unless explicitly mentioned.

3.11 Lemma Let $\vec{f} = (\vec{f}_1, \ldots, \vec{f}_n)$ and $\vec{g} = (\vec{g}_1, \ldots, \vec{g}_n)$ with $\vec{g}$ honest and $\vec{f}_i \succ \vec{g}_i$ $(1 \leq i \leq n)$. Let $\vec{e}$ be such that $(\vec{e}, \vec{f}_i)$ is honest for each i.



Then there exists $\vec{e'}$ such that $\vec{e} \succ \vec{e'}$ and $(\vec{e'}, \vec{g})$ is honest.

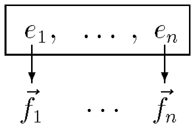


Proof Consider a typical element d occurring (perhaps repeatedly) in $\vec{e}$. To form $\vec{e'}$ from $\vec{e}$, we replace each occurrence of d by the sequence $\vec{d}$ defined as follows:

If $d \# \vec{f}_i$ for all i, then $d \# \vec{f}$ and hence $d \# \vec{g}$; take $\vec{d}$ to be just d. If $d \in \vec{f}_i$ for some i, then take $\vec{d}$ to be the subsequence of $\vec{g}_i$ for which $d \succ \vec{d}$.

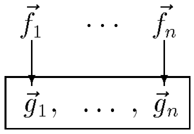
In both cases, clearly $(\vec{d}, \vec{g})$ is honest. Also $\vec{e'}$ is honest, since for distinct d and d' in $\vec{e}$ we have $d \# d'$, so their replacements in $\vec{e'}$ are also separated. Hence finally $(\vec{e'}, \vec{g})$ is honest. ■

3.12 Lemma Assume H^1_E and C_E. Let $\vec{e}$ be honest and $\vec{e} \succ \vec{f}$.



$$\vec{f} = (\vec{f}_1, \ldots, \vec{f}_n)$$

Then there exists honest $\vec{g}$ for which $\vec{f} \succ \vec{g}$.



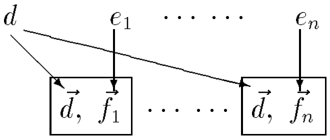
$$\vec{g} = (\vec{g}_1, \ldots, \vec{g}_n)$$

Proof Suppose $\vec{e} = (e_1, \ldots, e_n)$; by assumption, there are factorisations $e_i \succ \vec{f}_i$ $(1 \leq i \leq n)$ such that $\vec{f} = (\vec{f}_1, \ldots, \vec{f}_n)$.

Let e be a typical member of $\vec{e}$ occurring at positions $i_1, \ldots, i_r$, i.e. $e = e_{i_1} = \cdots = e_{i_r}$. The subsequences $\vec{f}_{i_1}, \ldots, \vec{f}_{i_r}$ of $\vec{f}$ may differ; using C_E there is $\vec{h}$ such that $\vec{f}_{i_1} \succ \vec{h}, \ldots, \vec{f}_{i_r} \succ \vec{h}$. Since $e \succ \vec{h}$, using H^1_E we may assume that $\vec{h}$ is honest. Now form $\vec{g}$ from $\vec{f}$ by replacing each of the subsequences $\vec{f}_{i_1}, \ldots, \vec{f}_{i_r}$ of $\vec{f}$ by $\vec{h}$; moreover, do this for each distinct member of $\vec{e}$. Clearly $\vec{f} \succ \vec{g}$.

Let e' be a member of $\vec{e}$ distinct from e, contributing in this way several occurrences of a subsequence $\vec{h'}$ to $\vec{g}$; then $e \# e'$ since $\vec{e}$ is honest, hence $\vec{h} \# \vec{h'}$. It follows that $\vec{g}$ is honest. ■

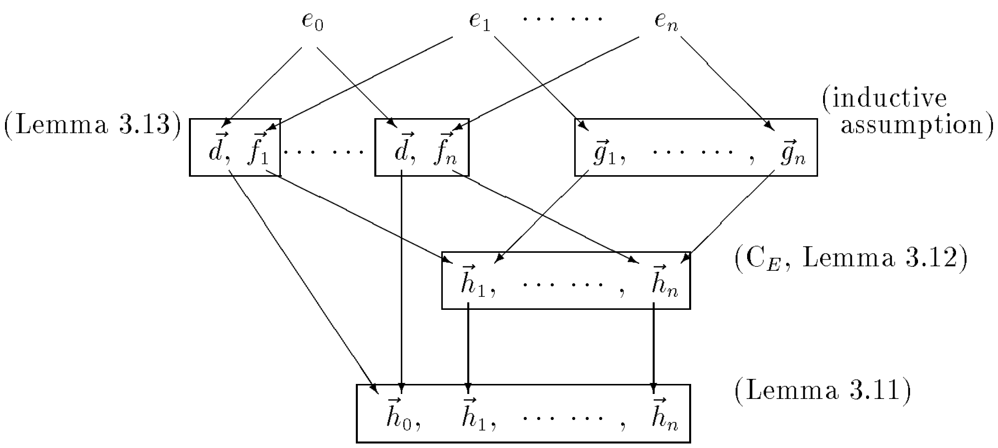
3.13 Lemma Assume H^1_E, H^2_E and C_E. Let $d, e_1, \ldots, e_n$ be elements. Then there exist $\vec{d}$ and $\vec{f}_i$ for each i $(1 \leq i \leq n)$ such that $d \succ \vec{d}$, $e_i \succ \vec{f}_i$ and $(\vec{d}, \vec{f}_i)$ is honest for each i.



Proof By H^2_E, for each i there exist $\vec{d}_i$ and $\vec{e}_i$ for which $d \succ \vec{d}_i$, $e_i \succ \vec{e}_i$ and $(\vec{d}_i, \vec{e}_i)$ is honest. By C_E, there exists $\vec{d}$ such that $\vec{d}_i \succ \vec{d}$ for each i; also by H^1_E we may assume $\vec{d}$ to be honest. Then by honest substitution (Lemma 2.13), for each i there exists $\vec{f}_i$ such that $\vec{e}_i \succ \vec{f}_i$ and $(\vec{d}, \vec{f}_i)$ is honest. ■

We are now ready for the main result of this section.

3.14 Proof of Theorem 3.6 Let $(X, \#, E)$ be an s-monoid satisfying C_E. It is immediate that H_E implies H^1_E and H^2_E. For the converse, we prove H_E by induction on the length of $\vec{e}$. It is trivial for length 0; for length 1 it is a simple consequence of H^1_E. We now prove the induction step: If $(e_1, \ldots, e_n)$ has an honest refinement then so does $(e_0, e_1, \ldots, e_n)$. The proof can be presented completely as a diagram:



■

4 Examples of namings

We give several examples of namings, to indicate their variety. In fact we present each as an elementary naming $(X, \#, E)$; we give its elements E, its equational laws and its separation laws. We take equality to be the smallest congruence satisfying the equational laws, and separation to be the smallest symmetric relation satisfying S1, S2, S3 and the separation laws.

4.1 The free monoid

$$\begin{aligned} \text{Elements} &: \text{ Some set } E\ . \\ \text{Equations} &: \text{ None }. \\ \text{Separation laws} &: \ e \# f \text{ for all distinct } e, f \in E\ . \end{aligned}$$

This is a sharp naming. All distinct elements are separate. The subname ordering $>$ is well-founded.

4.2 Tree-names In this example we write the monoidal product explicitly, to avoid confusing it with the concatenation of sequences of 0's and 1's. Each element e is a finite binary sequence such as 0010; it can be thought of as addressing a node in the infinite binary tree, so a product like $x = 100 \otimes 0010 \otimes 100 \otimes 101$ names a sequence (allowing repetitions) of such nodes. But we also assume that the consecutive occurrence of two sibling nodes $e0$, $e1$ in this sequence is the same as an occurrence of their parent node e. Thus if x is as above we have $x = 100 \otimes 0010 \otimes 10$.

$$\begin{aligned} \text{Elements} &: \text{ Finite sequences over } \{0, 1\}\ . \\ \text{Equations} &: \ e0 \otimes e1 = e, \text{ for all } e \in E\ . \\ \text{Separation laws} &: \ e0 \# e1, \text{ for all } e \in E\ . \end{aligned}$$

This is also a sharp naming. Distinct elements need not be separate, e.g. $e > e1$. But for all $e, f \in E$ either $e \# f$ or $e \leq f$ or $e \geq f$, i.e. if e and f have a proper subname in common then $e \leq f$ or $e \geq f$. The subname ordering $>$ is not well-founded.

4.3 Array-names This is a finite example; the maximal elements are the composite name a of a 3×2 array and its transpose a'; also each row, each column and each entry in the array is named (r_i for a row, c_j for a column, e_{ij} for an entry).

$$\begin{array}{rcl} \text{Elements} & : & a, a';\ r_1, r_2, r_3;\ c_1, c_2;\ e_{11}, e_{12}, e_{21}, e_{22}, e_{31}, e_{32}\ . \\ \text{Equations} & : & a = r_1 \otimes r_2 \otimes r_3,\ a' = c_1 \otimes c_2, \\ & & r_i = e_{i1} \otimes e_{i2}\ (i = 1,2,3), \\ & & c_j = e_{1j} \otimes e_{2j} \otimes e_{3j}\ (j = 1,2)\ . \\ \text{Separation laws} & : & e_{ij} \# e_{i'j'}\ (i \neq i' \text{ or } j \neq j')\ . \end{array}$$

This is a sharp naming, but two incomparable elements may have a common proper subname (e.g. a and a', or r_1 and c_2).

4.4 A non-sharp naming There are only two names, $X = \{1, e\}$.

$$\begin{array}{rcl} \text{Elements} & : & e\ . \\ \text{Equations} & : & e = e \otimes e\ . \\ \text{Separation laws} & : & \text{None}\ . \end{array}$$

This is not a sharp naming, but every factorisation of e is honest, containing arbitrarily many copies of 1 and of e. Cancellation fails; $x \otimes y = x \otimes z \not\Rightarrow y = z$.

4.5 Remark I do not claim that all the above namings will be useful in control structures; the examples merely serve to indicate that the axioms S1, S2, S3, C and H are not too restrictive.

Finally:

4.6 A non-naming It was remarked in 3.7 and 3.10 that the s-monoid freely generated by the basis $E = \{e, f\}$ is sharp but not a naming, if its separation relation is minimal (i.e. $x \# y$ iff $x = 1$ or $y = 1$). The same applies to any free monoid with more than one generator. However, a free monoid can always be made into a naming simply by making $e \# f$ for every two distinct generators e and f.

In contrast, consider the following:

$$\begin{array}{rcl} \text{Elements} & : & e, a\ . \\ \text{Equations} & : & a = e \otimes a\ . \\ \text{Separation laws} & : & \text{None}\ . \end{array}$$

The separation axioms hold (by fiat), and confluence holds; but the element a cannot be honestly factorised. Moreover, this example is *essentially* a non-naming. Its underlying monoid cannot be made into an honest s-monoid by *any* separation relation; the reason is that $e < a$, so we can never have $e \# a$.

5 Closure operations on names

This section proves one theorem which relates three closure operations on sets of names. The theorem will be used later to show that in control structures both the surface function $\mathbf{surf}\, a$ and its complement $\#a$, the exterior of a, respect the abstraction operations in a natural sense.

Throughout this section we assume that X is a naming. We use $\overline{S}$ for the complement of any set $S \subseteq X$ of names, and we define $\#S \stackrel{\text{def}}{=} \{x \mid \forall y \in S.\ x \# y\}$.

5.1 Definition Let $S \subseteq X$. The *downward closure* $S^{\wedge}$, *upward closure* $S^{\vee}$ and *monoidal closure* $S^{\otimes}$ of S are defined as follows:

$$\begin{aligned} S^{\wedge} &\stackrel{\text{def}}{=} \{x \mid \exists y \in S.\ x \leq y\} \\ S^{\vee} &\stackrel{\text{def}}{=} \{x \mid \exists y \in S.\ x \geq y\} \\ S^{\otimes} &\stackrel{\text{def}}{=} \{x_1 \otimes \cdots \otimes x_n \mid x_1, \ldots, x_n \in S,\ n \geq 0\}\ . \end{aligned}$$

By convention, $1 \in S^{\otimes}$. We say that S is *downward closed* if $S = S^{\wedge}$; similarly for the other two closures.

The following simple properties depend upon neither confluence nor honesty; proofs are trivial.

5.2 Fact

1. S is downward closed iff $\overline{S}$ is upward closed.
2. S is upward and monoidal closed iff $S = X$.
3. $\#S$ is downward and monoidal closed. ■

It will also turn out that in control structures $\#a$ is downward and monoidal closed; thus we are interested in sets with this property. The following property is slightly less trivial:

5.3 Proposition If S is downward closed then so is $S^{\otimes}$.

Proof A straightforward use of confluence. ■

Note that if S is monoidal closed, it does *not* follow that $S^{\wedge}$ is monoidal closed. A counterexample is $S = (x \otimes y)^{\otimes}$ in the free monoid, where x and y are distinct generators; for we have $xx \notin S^{\wedge}$ but $xx \in S^{\wedge\otimes}$.

The following is a direct consequence of Proposition 2.9.

5.4 Proposition $\overline{\#S} = (S^{\wedge} - \{1\})^{\vee}$. ■

We now turn to the most important result concerning closure.

5.5 Theorem Let S be downward and monoidal closed. Then

$$\overline{(S \cup x^{\wedge})^{\otimes}} \;=\; (\overline{S} \cap \#x)^{\vee}\ .$$

Proof ($\subseteq$) Let y be a member of the left-hand side. By honesty, there exist $\vec{x}$ and $\vec{y}$ with $x \succ \vec{x}$, $y \succ \vec{y}$ and $(\vec{x}, \vec{y})$ honest. So for some z in $\vec{y}$, we have $z \notin S \cup x^{\wedge}$; that is, $z \in \overline{S}$ and $z \not\leq x$. It follows from the latter and the honesty of $(\vec{x}, \vec{y})$ that $z \# \vec{x}$, hence $z \in \#x$. But $z \leq y$, so $y \in (\overline{S} \cap \#x)^{\vee}$.

($\supseteq$) We shall derive a contradiction from $y \in (S \cup x^{\wedge})^{\otimes}$ and $y \in (\overline{S} \cap \#x)^{\vee}$. From the latter we have $y \succ (y_1, v, y_2)$ for some $v \in \overline{S} \cap \#x$; from the former we have $y \succ (u_1, \ldots, u_k)$ with $u_i \in S \cup x^{\wedge}$ for each i. By confluence, these two factorisations of y have a common refinement $\vec{w}$, and we may assume w.l.o.g. that 1 is not in $\vec{w}$.

Now for each w in $\vec{w}$, since $w \leq u_i$ for some i and $S \cup x^\wedge$ is downward closed, we have $w \in S$ or $w \leq x$. But also $v \succ \vec{v}$ for some subsequence $\vec{v}$ of $\vec{w}$. So we distinguish two cases:

Case 1: $w \in S$ for all w in $\vec{v}$. Then $v \in S$ since S is monoidal closed.

Case 2: $w \leq x$ for some w in $\vec{v}$. Then $w \notin \#x$ (since $w \neq 1$), so $v \notin \#x$ since $\#x$ is downward closed.

But each case contradicts $v \in \overline{S} \cap \#x$ which we deduced from our assumption, so we are done. ∎

5.6 Discussion In the proof of the above theorem, it is remarkable that the first part ($\subseteq$) depends upon honesty but not upon confluence, while the second part ($\supseteq$) depends upon confluence but not upon honesty. The theorem also has a distinct topological flavour, which needs to be analysed. Finally, I have resisted trying to state the theorem in a more general form than needed here, but generalisation should be investigated.

6 Surface properties in control structures

6.1 Extended axioms As indicated in Section 1, we generalise axioms S7 and γ_1 of control structures to allow for arbitrary namings as follows:

$$\boxed{\begin{array}{lll} \mathrm{S7} & : & (c \otimes \mathbf{id}) \cdot \mathbf{ab}_y \mathbf{ab}_x a = \mathbf{ab}_x \mathbf{ab}_y a \cdot (c \otimes \mathbf{id}) \quad (x \# y) \\ \gamma_1 & : & \mathbf{ab}_x \langle y \rangle = \mathbf{id} \otimes \langle y \rangle \qquad\qquad\qquad\qquad\quad (x \# y)\ . \end{array}}$$

We also introduce axioms which state that data-construction and abstraction respect the monoidal structure of names:

$$\boxed{\begin{array}{lrclrcl} \mathrm{M1}: & \langle 1 \rangle & = & \mathbf{id}_\epsilon\ , & \langle x \otimes y \rangle & = & \langle x \rangle \otimes \langle y \rangle \\ \mathrm{M2}: & \mathbf{ab}_1 a & = & a\ , & \mathbf{ab}_{x \otimes y} a & = & \mathbf{ab}_x \mathbf{ab}_y a\ . \end{array}}$$

Finally we assume that we can always find a new name at any arity; to be precise, we make the following *fresh name assumption*:

$$\boxed{\text{F : For each name } x \text{ and arity } m\text{, there exists } y : m \text{ such that } x \# y.}$$

6.2 Exterior properties Recall that we are currently working with the exterior $\#a \stackrel{\text{def}}{=} \{x \mid \mathbf{ab}_x a = \mathbf{id} \otimes a\}$ in place of surface. (At the end of this section we deduce surface properties.) The exterior properites we wish to establish are

$$\begin{array}{ll} \#\mathbf{id} = \#\mathbf{c} = X & \#\langle x \rangle \supseteq \#x \\ \#(a \cdot b) \supseteq \#a \cap \#b & \#\boldsymbol{\omega} = X \\ \#(a \otimes b) \supseteq \#a \cap \#b & \#K(a_1, \ldots, a_r) \supseteq \#a_1 \cap \cdots \cap \#a_r \end{array}$$
$$\#\mathbf{ab}_x a = (\#a \cup x^\wedge)^\otimes\ .$$

The proofs of all but the last are immediate, as in [1]; note in particular that $\#\langle x \rangle \supseteq \#x$ is a direct consequence of the generalised γ_1. The bulk of this section is devoted to proving the last and central property, which we called the *abstraction* property of the exterior.

We begin with a key lemma; it explains why we can hope for the abstraction property to be an equation, not merely an inequation.

6.3 Lemma Let $x \succ \vec{x}$, and let $(y, \vec{x})$ be honest. Then

$$y \in \#\mathbf{ab}_x a \text{ iff } (y \in \#a \text{ or } y \in \vec{x}) \ .$$

Proof There is no need to give full details of the proof, since it is simply a matter of manipulating the control structure axioms and requires no results about namings except the definition of honesty.

Let $\vec{x} = (x_1, \ldots, x_k)$. Consider the implication from left to right; assume $\mathbf{ab}_y \mathbf{ab}_x a = \mathbf{id} \otimes \mathbf{ab}_x a$. We take the left side, i.e. $\mathbf{ab}_y \mathbf{ab}_{x_1} \cdots \mathbf{ab}_{x_k} a$, and using S7 push $\mathbf{ab}_y$ through each $\mathbf{ab}_{x_i}$ such that $y \# x_i$ until and if we find $y = x_i$ for some i. If this occurs then $y \in \vec{x}$; if not, then we can push $\mathbf{ab}_y$ right through and obtain

$$\mathbf{ab}_x \mathbf{ab}_y a = \mathbf{ab}_x(\mathbf{id} \otimes a) \ ;$$

it then only remains to apply $(\langle x \rangle \otimes \mathbf{id}) \cdot (-) \cdot (\boldsymbol{\omega} \otimes \mathbf{id})$ to both sides to get $\mathbf{ab}_y a = \mathbf{id} \otimes a$, i.e. $y \in \#a$.

The other implication is no harder. ∎

We now show that the abstraction $\mathbf{ab}_x$, as its name implies, makes x (and all its sub-names) exterior.

6.4 Lemma $x^\wedge \subseteq \#\mathbf{ab}_x a$.

Proof Let $y \le x$. By honesty, there are honest factorisations $x \succ \vec{x}$ and $y \succ \vec{y}$, where $\vec{x} = (\vec{x_1}, \vec{y}, \vec{x_2})$. Now let $\vec{y} = (y_1, \ldots, y_k)$. Then $y_i \in \vec{x}$ for each i, so by Lemma 6.3 we have $y_i \in \#\mathbf{ab}_x a$, i.e. $\mathbf{ab}_{y_i} \mathbf{ab}_x a = \mathbf{id} \otimes \mathbf{ab}_x a$. By using control structure axioms we deduce more generally

$$\mathbf{ab}_{y_i}(\mathbf{id} \otimes \mathbf{ab}_x a) = \mathbf{id} \otimes \mathbf{ab}_x a$$

where the identities $\mathbf{id}$ may have any arities which make sense. Putting these together for $i = 1, \ldots, k$ we deduce $\mathbf{ab}_y \mathbf{ab}_x a = \mathbf{id} \otimes \mathbf{ab}_x a$. ∎

We now prove closure properties of the exterior function.

6.5 Proposition $\#a$ is downward and monoidal closed.

Proof For downward closure, assume $x \in \#a$ and $y \le x$. By Lemma 6.4 we deduce $y \in \#\mathbf{ab}_x a$, hence $y \in \#(\mathbf{id} \otimes a)$; that is, $\mathbf{ab}_y(\mathbf{id} \otimes a) = \mathbf{id} \otimes \mathbf{id} \otimes a$. Now, by the fresh name assumption F, we find some $z \# y$ of suitable arity, and apply $(\mathbf{id} \otimes \langle z \rangle \otimes \mathbf{id}) \cdot (-) \cdot (\mathbf{id} \otimes \boldsymbol{\omega} \otimes \mathbf{id})$ to both sides; using $\mathbf{id} \otimes \langle z \rangle = \mathbf{ab}_y \langle z \rangle$ from $\boldsymbol{\gamma}_1$, we then get $\mathbf{ab}_y a = \mathbf{id} \otimes a$ as required.

For monoidal closure, note first that $1 \in \#a$ by axiom M1. Now let $x, y \in \#a$; using axiom M2 we find

$$\begin{aligned}
\mathbf{ab}_{xy} a &= \mathbf{ab}_x \mathbf{ab}_y a \\
&= \mathbf{ab}_x(\mathbf{id} \otimes a) \\
&= (\mathbf{c} \otimes \mathbf{id}) \cdot (\mathbf{id} \otimes \mathbf{ab}_x a) \cdot (\mathbf{c} \otimes \mathbf{id}) \\
&= (\mathbf{c} \otimes \mathbf{id}) \cdot (\mathbf{id} \otimes \mathbf{id} \otimes a) \cdot (\mathbf{c} \otimes \mathbf{id}) \\
&= \mathbf{id} \otimes a \ .
\end{aligned}$$

Thus $xy \in \#a$, and we have shown $\#a$ to be monoidal closed. ∎

We now reach our first main result:

6.6 Theorem (Exterior of abstraction) $\#\mathbf{ab}_x a = (\#a \cup x^\wedge)^\otimes$.

Proof For any name y, by honesty we can find factorisations $x \succ \vec{x}$ and $y \succ \vec{y} = \{y_1, \ldots, y_k\}$ with $(\vec{x}, \vec{y})$ honest. We shall use this to prove the inequality in both directions.

($\subseteq$) Assume $y \in \#\mathbf{ab}_x a$. Then each $y_i \in \#\mathbf{ab}_x a$ by downward closure (Proposition 6.5), hence by Lemma 6.3 we deduce $y_i \in \#a$ or $y_i \in \vec{x}$, which implies $y_i \in \#a \cup x^\wedge$. Hence $y \in (\#a \cup x^\wedge)^\otimes$.

($\supseteq$) Since $\#\mathbf{ab}_x a$ is monoidal closed (Proposition 6.5) it is enough to show that $\#a \cup x^\wedge \subseteq \#\mathbf{ab}_x a$; using Lemma 6.4 it only remains to show $\#a \subseteq \#\mathbf{ab}_x a$. Now let $y \in \#a$. Using the above-stated factorisations, for each i we have $(y_i, \vec{x})$ honest and $y_i \in \#a$ since $\#a$ is downward closed; from Lemma 6.3 it follows that $y_i \in \#\mathbf{ab}_x a$ for each i. By monoidal closure $y \in \#\mathbf{ab}_x a$ follows, and we are done. ■

It remains only to rephrase our exterior properties as surface properties. We have everything we need to do so.

6.7 Theorem (Surface properties) In any control structure, the following properties hold of the surface function:

$$\begin{array}{ll}
\mathbf{surf\,id} = \mathbf{surf\,c} = \emptyset & \mathbf{surf}\langle x \rangle \subseteq (x^\wedge - \{1\})^\vee \\
\mathbf{surf}(a \cdot b) \subseteq \mathbf{surf}\,a \cup \mathbf{surf}\,b & \mathbf{surf}\,\boldsymbol{\omega} = \emptyset \\
\mathbf{surf}(a \otimes b) \subseteq \mathbf{surf}\,a \cup \mathbf{surf}\,b & \mathbf{surf}\,K(a_1, \ldots, a_r) \subseteq \mathbf{surf}\,a_1 \cup \cdots \cup \mathbf{surf}\,a_r
\end{array}$$
$$\mathbf{surf\,ab}_x a = (\mathbf{surf}\,a \cap \#x)^\vee \ .$$

Proof In almost all cases, the proof is just as in Propositions 3.5 and 4.4 of [1]. Recalling that $\mathbf{surf}\,a = \overline{\#a}$, the property for $\mathbf{surf}\langle x \rangle$ follows immediately from Proposition 5.4 and the corresponding exterior property in 6.2. Finally, the surface property of abstraction follows immediately from Theorem 6.6 and Theorem 5.5, recalling that the exterior is downward and monoidal closed (Proposition 6.5). ■

References

[1] Alex Mifsud, Robin Milner and John Power, *Control Structures*, draft paper, LFCS, June 1994.