First Year Report: Symmetric Action Calculi

Lucian Wischik
Supervised by Philippa Gardner
31st July 1998

1 INTRODUCTION

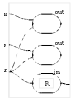
The Action Calculi provide a uniform foundation into which can be embedded the syntax and semantics of diverse interactive calculi, such as the π-calculus, CCS and petri-nets. In this report we introduce a modified version, called the Symmetric Action Calculus, which is more general and which leads to a simpler system.

The action calculi have three presentations. First, they are defined as the quotient of a term algebra by certain equations. Terms are generated from a set of operators common to all embeddings, and from a set of *controls* particular to each embedding. For instance the control *signature* PIC $= \{\boldsymbol{\nu}, \mathbf{in}, \mathbf{out}\}$ characterises a fragment of the π-calculus. Second, the action calculi can be presented as an assembly of *particles* bound together by names. These *molecular forms* subsume in their structural congruence many of those equalities that have to be derived in the term algebra. Third, *action graphs* are a convenient graphical presentation of molecular forms. The basic action calculi has also been extended with a reflexion operator to give the Reflexive Action Calculi, and also with higher order features.

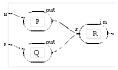
A central idea in all action calculi is the use of names to model interaction. Gardner's Closed Action Calculi are a name-free account of the standard action calculi: this actually makes explicit the role normally played by names.

The symmetric action calculi presented in this report are described at all levels—as term algebras with a structural congruence, as molecular forms, as action graphs, and in closed form. The graphs are a useful and intuitive aid to understanding: we present the other levels formally and prove them to be isomorphic.

The symmetric calculi are motivated by the simplification they bring to expressing the π-calculus. Consider the term $\overline{x}\langle u\rangle \mid \overline{x}\langle v\rangle \mid x(z).P$ in which an x can be sent from either of two places. In the conventional action calculus it is possible to copy the name x but not to merge it: this gives rise to the rather awkward idiom illustrated below. The symmetric calculus allows two names to be merged and so permits a more natural translation from the π-calculus:



Communication in conventional action calculus



Communication in reflexive action calculus

By making the action calculi symmetric we also remove irregularities. For instance in the action calculus that corresponds to the π-calculus, we can discard a name using only the common operators; but to introduce a new name requires a separate control. When the π-calculus is instead expressed as a symmetric action calculus both tasks can be accomplished using only the common operators.

A third motivation is to separate two aspects of the binding of names. In the λ-expression $(\lambda x.fx)\ y$ the name x becomes restricted in *scope* to the body of the λ-term $\lambda x.fx$; and it is also *assigned* the value of y. In the symmetric calculus the scoping and assigning of names are indicated with two separate syntactic constructs: it is possible to assign a name without restricting its scope, and to restrict its scope without necessarily assigning it. [1]

Actions in the conventional action calculi are directed: they have input ports and output ports; and it is only the output ports that bind names. But in the symmetric action calculi any input port can be turned into an output port—or vice versa—by placing the action in a simple context. The difference between input and output ports becomes therefore merely a pragmatic one; and the symmetric action calculi are seen to be similar to Honda's *undirected actions*.

The symmetric action calculi appear to be a conservative extension of the reflexive. Proving this, and relating the symmetric calculi to the fusion calculus of Parrow and Victor, are the immediate areas of ongoing research.

Overview of the rest of the report In this report we introduce the symmetric action calculi. Section 2 describes the existing action calculi, and section 3 reports our work on the symmetric version. Similarly,

[1] The term *bound variable* is commonly used but would be confusing for the purposes of this report. The word 'bound' is the past participle of 'bind', meaning *to assign*; but it is also an infinitive meaning *to restrict in scope*. When we say that x is bound in the λ-expression $\lambda x.fx$ we are making a pun: for it is both bound and bounded!

section 4 describes first Gardner's conventional closed action calculi, and then our work on the symmetric closed action calculi. And in section 5 we describe our work on symmetric molecular forms. The main result is that all three presentations are isomorphic. Applications of the symmetric calculi are considered in the proposal.

2 ACTION CALCULI

We give a brief account of the action calculi as the quotient of a term algebra, and of action graphs. This is followed by a discussion of possible ways in which the basic term algebra might be made symmetric. For a full description of the action calculi see [Milner 1996].

Conventional action calculi The conventional action calculi consist of a set of terms, an equational theory on terms and a preorder on the equivalence classes called a *reaction relation*. It presupposes a set of *arities* generated from a set of prime arities with the identity ϵ under tensor composition $\otimes$, and a set of names each associated with a prime arity. We write $x : p$ to indicate that x has arity p. Terms are generated from the basic operators $\mathbf{id}$,$\cdot$, $\otimes$, $\mathbf{ab}_x$,ω, x^+ and from a set of controls peculiar to the embedding in question. (The datum operator has conventionally been written $< x >$; we use the form x^+ for internal consistency in this report.) Terms are associated with an input and output arity m and n, written $t : m \to n$. Let s, t range over terms, p, q over primes and x, y over names.

$$t ::= \mathbf{id} \mid s \cdot t \mid s \otimes t \mid K(\vec{t}) \mid \mathbf{ab}_x t \mid \omega \mid x^+$$

$$\mathbf{id}_m : m \to m \qquad \frac{s : k \to \ell \qquad t : \ell \to m}{s \cdot t : k \to m} \qquad \frac{s : k \to m \qquad t : \ell \to n}{s \otimes t : \ k \otimes \ell \to m \otimes n}$$

$$\frac{t : m \to n \qquad x : p}{\mathbf{ab}_x t : p \otimes m \to p \otimes n} \qquad x^+ : \epsilon \to p, \qquad x : p \qquad \omega_p : p \to \epsilon$$

$$\frac{t_1 : m_1 \to n_1 \ \cdots \ t_r : m_r \to n_r}{K(t_1, \cdots, t_r) : m \to n}$$

The abstraction operator $\mathbf{ab}_x t$ binds and bounds x; datum x^+ represents a free occurrence of x.

Action graphs There is a natural graphical presentation of terms in the action calculi, as indicated in the following illustrations.

id $s \cdot t$ $s \otimes t$ $\mathbf{ab}_x t$ ω x^+ Control

(In the diagram for $\mathbf{ab}_x t$, all free occurrences of datum x^+ are crossed out—bound—and a line is connected to each of them.) Observe that the graphs subsume basic structural congruences, such as the associativity of tensor and composition.

Derived operators The following derived operators are commonly used for abstraction and permutation. They may be given their own graphical shorthands, as shown.

Derived abstraction operator $(x)t = \mathbf{ab}_x t \cdot (\omega \otimes \mathbf{id})$

Derived permutation operator $\mathbf{perm}_{p,q} = (xy)y^+x^+$

Axioms The axioms of the conventional action calculi are as follows.

	(action structure axioms)	
A1:	$t{\cdot}\mathbf{id} = t = \mathbf{id}{\cdot}t$	
A2:	$t \otimes \mathbf{id}_\epsilon = t = \mathbf{id}_\epsilon \otimes t$	
A3:	$id \otimes \mathbf{id} = \mathbf{id}$	
A4:	$s{\cdot}(t{\cdot}u) = (s{\cdot}t){\cdot}u$	
A5:	$s \otimes (t \otimes u) = (s \otimes t) \otimes u$	
A6:	$(s{\cdot}t) \otimes (u{\cdot}v) = (s \otimes u){\cdot}(t \otimes v)$	
A7:	$\mathbf{ab}_x\mathbf{id} = \mathbf{id}$	
A8:	$\mathbf{ab}_x(s{\cdot}t) = \mathbf{ab}_x s{\cdot}\mathbf{ab}_x t$	
	(concrete axioms)	
$\boldsymbol{\gamma}$:	$(x)t \ = \omega \otimes t$	$(x \notin \mathbf{fn}\ t)$
$\boldsymbol{\delta}$:	$(x)(x^+ \otimes \mathbf{id}_m) = \mathbf{id}_{p \otimes m}$	$(x : p)$
$\boldsymbol{\zeta}$:	$\mathbf{perm}_{k,m}{\cdot}(t \otimes s) = (s \otimes t){\cdot}\mathbf{perm}\ell, n$	$(s : k \to \ell, t : m \to n)$
$\boldsymbol{\sigma}$:	$(y^+ \otimes \mathbf{id}_m){\cdot}(x)t = t\{y/x\}$	$(t : m \to n)$

Symmetry The conventional action calculi are not symmetric: for instance, we can write a derived operator $\omega = (x)\mathbf{id}_\epsilon$ which discards names, but not a derived operator ν which introduces them. And there is a datum $x^+ : \epsilon \to p$ but no inverse datum $x^- : p \to \epsilon$. Below we make a few suggestions for ways in which they can be made symmetric.

Symmetric action calculi with inverse datum: $\{x^+,\ x^-,\ (\cdot)_x\ \}$ The following pictures illustrate the trend from $\mathbf{ab}_x t$ (with an input and output port) to $(x)t$ (with only an inport port). A logical extension of this trend would be a *scoping* operator with no extra ports, which we will write $(t)_x$.

$\mathbf{ab}_x t$: 2 ports $\quad (x)t$: 1 port $\quad (t)_x$: 0 ports

Just as $\mathbf{ab}_x t$ can be written in terms of $(x)t$ and a datum, so too $(x)t$ would be written in terms of $(t)_x$ and an inverse datum.

$\mathbf{ab}_x t$ derived from $(x)t$ $\quad (x)t$ derived from $(t)_x$

Observe that both ν and ω can be constructed using the inverse datum and the new scoping operator. And a graph can be drawn that illustrates an effect similar to reflexion.

Derived ω $\quad$ Derived ν $\quad$ Derived reflexion

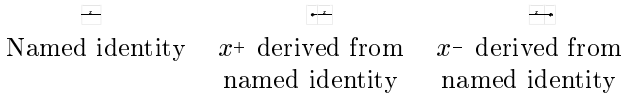
This particular symmetric form is the one that has been adopted, primarily because it can express such things as ω, ν and reflexion with only a small number of operators. However, a slightly different form of action graph seems appropriate to the symmetric action calculi: this is detailed in the following section.

Symmetric action calculi with ν: $\{\ x^+,\ x^-,\ \mathbf{ab}_x(\cdot),\ \omega,\ \nu\ \}$ Alternatively, one might simply list all the operators in the conventional action calculi, and introduce for each its mirror image. Abstraction $\mathbf{ab}_x$ is already symmetric and so needs no mirror. The datum operator would lead to an inverse-datum, presumably with the same behaviour as above. And ω would be mirrored with a ν operator.

From the abstraction operator $\mathbf{ab}_x$ we would be able to derive two further abstraction operators: left-handed binding $(x)t = \mathbf{ab}_x t{\cdot}(\omega \otimes \mathbf{id})$ and right-handed binding $[x]t = (\nu \otimes \mathbf{id}){\cdot}\mathbf{ab}_x t$.

This approach appears inelegant. It has two more operators that the previous suggestion but is no more expressive. (Proof: $\mathbf{ab}_x(\cdot)$, ω, ν can all be derived from x^+, x^- and $(\cdot)_x$).

Symmetric action calculi with named identities: { x, ω, ν, $(\cdot)_x$ } A third possibility is to remove the datum operator and have named identities $x : p \to p$ as illustrated below. The expression $\nu \cdot x$ would be equivalent to a datum, and $x \cdot \omega$ to an inverse datum. The scoping operator would presumably be the same as in $(t)_x$ above.



Named identity | x^+ derived from named identity | x^- derived from named identity

It is not clear what the expression $x \otimes x$ would mean, nor $x \cdot y$. Perhaps the former might merge all inputs and outputs together, and the latter might identify x and y. If this does happen, then this named-identity calculi would be exactly as expressive as the first symmetric calculi (through the equivalence $x = x^- \otimes x^+$).

The choice of symmetric calculi While there are several possible choices for the symmetric term algebra, there is less choice for the closed symmetric action calculi (presented in a later section). Since it is the case that the first suggested symmetric action calculi is indeed isomorphic to the symmetric closed calculi, we are not unjustified for the present in choosing it over the others. The alternatives should be checked in the future. The chosen symmetric calculi are detailed more fully in the next section.

3 SYMMETRIC ACTION CALCULI

In this section we introduce the symmetric action calculi. Terms in the symmetric calculi are generated by the following rules. Composition binds more tightly than tensor product.

$$t ::= \mathbf{id} \mid s \cdot t \mid s \otimes t \mid K(t) \mid (t)_x \mid x^+ \mid x^-$$

$$\mathbf{id}_m : m \to m \qquad \frac{a : k \to \ell \qquad b : \ell \to m}{a \cdot b : k \to m} \qquad \frac{a : k \to m \qquad b : \ell \to n}{a \otimes b : \; k \otimes \ell \to m \otimes n}$$

$$x^+ : \epsilon \to p \qquad x^- : p \to \epsilon \qquad \frac{a : m \to n}{(a)_x : m \to n}$$

The operator $(t)_x$ restricts the *scope* of x; and x^+ and x^- represent *free* occurrences of x. Results for free and local names, for substitution and for alpha-conversion are as standard. We use the terms *local* and *scoped names* interchangeably.

Free and local names, and their graphs Where a free name is repeated, as in $x^+ \otimes x^+$, we choose to draw both data coming from a single labelled node x: this indicates that all occurrences of the name x implicitly refer to the *same* name. For local names we leave the node unlabelled, as in $(x^+ \otimes x^+)_x$, indicating that the name is now private. This is a convenient notation, because the graphical equivalent of the restriction operator $(\cdot)_x$ is simply expressed as the erasing of a label from a node. (Compare this with the confusing notation for abstraction in conventional action graphs.)

Free names, in the expression $x^+ \otimes x^+$

Local names are represented with unlabelled nodes, as in the expression $(x^+ \otimes x^+)_x$

The graphical equivalent of restriction $(\cdot)_x$ is simply to cross the label off the node

Lines are undirected In the conventional action calculi and reflexive action calculi, binding happens only in one direction: ports to the right always bind, and ports to the left never bind, and the two are quite different. In the symmetric calculi, however, a simple context can switch a port from left to right or vice versa. This indicates that there is no essential difference between input and output ports: so we leave lines undirected in the action graphs.

$$\begin{array}{lll} \text{From left to right :} & t : p \otimes m \to n & \Rightarrow ((x^+ \otimes \mathbf{id}) \cdot (x^+ \otimes t))_x : m \to p \otimes n \\ \text{From right to left :} & t : m \to p \otimes n & \Rightarrow ((x^- \otimes t) \cdot (x^- \otimes \mathbf{id}))_x : p \otimes m \to n \end{array}$$

The same phenomenon is observed in Honda's undirected actions [1997], where it is called *switching*.

Derived operators Reflexion, abstraction, ω, ν, permutation, copy and merge can all be expressed in the symmetric calculi as follows.

$$\begin{array}{rcl} \omega & = & (x^-)_x \\ \nu & = & (x^+)_x \\ \mathbf{ab}_x t & = & (x^- \otimes x^+ \otimes t)_x \\ (x)t & = & (x^- \otimes t)_x \\ \uparrow t & = & ((x^+ \otimes \mathbf{id}) \cdot t \cdot (x^- \otimes \mathbf{id}))_x \\ \text{copy} & = & (x^- \otimes x^+ \otimes x^+)_x \\ \text{merge} & = & (x^- \otimes x^- \otimes x^+)_x \\ \text{perm} & = & (x^- \otimes y^- \otimes y^+ \otimes x^+)_{xy} \end{array}$$

$$\omega = (x^-)_x \qquad \nu = (x^+)_x \qquad \mathbf{ab}_x t = (x^- \otimes x^+ \otimes t)_x$$

$$(x)t = (x^- \otimes t)_x \qquad \uparrow t = ((x^+ \otimes \mathbf{id}) \cdot t \cdot (x^- \otimes \mathbf{id}))_x \qquad \mathrm{copy} = (x^- \otimes x^+ \otimes x^+)_x$$

$$\mathrm{merge} = (x^- \otimes x^- \otimes x^+)_x \qquad \mathrm{perm} = (x^- \otimes y^- \otimes y^+ \otimes x^+)_{xy}$$

(We draw a curved line that does not quite reach the name, to indicate that the term may contain some occurrences of that name.)

Axioms Axioms are as in the conventional action calculi but for the following changes and additions. Motivation for all but the last axiom is apparent from the graphical notation, as illustrated. Two additional control axioms will be discussed later. (The work had initially been done with an extra axiom $\boldsymbol{\epsilon} : x^+ \cdot x^- = \mathbf{id}_\epsilon$, but we have since found that this can be deduced from $\boldsymbol{\sigma}$ and A7.)

$$\begin{array}{ll} \text{A7:} & (\mathbf{id})_x = \mathbf{id} \\ \text{A8:} & (s \cdot t)_x = (x^+ \otimes s)_x \cdot (x^- \otimes t)_x \\ \boldsymbol{\gamma}: & (t \otimes \mathbf{id})_x = (t)_x \otimes \mathbf{id}, \quad (\mathbf{id} \otimes t)_x = \mathbf{id} \otimes (t)_x \\ \boldsymbol{\delta}: & (x^+ \otimes x^-)_x = \mathbf{id} \\ \boldsymbol{\sigma}: & (y^+ \cdot x^- \otimes t)_x = (x^+ \cdot y^- \otimes t)_x = (t[y/x])_x \end{array}$$

$$(\mathbf{id})_x = \mathbf{id} \qquad (s \cdot t)_x = (x^+ \otimes s)_x \cdot (x^- \otimes t)_x \qquad (t \otimes \mathbf{id})_x = (t)_x \otimes \mathbf{id} \qquad (x^+ \otimes x^-)_x = \mathbf{id}$$

Free substitution axiom An immediate consequence of the substitution axiom is that, where two local nodes in an action graph are joined together, they can be merged into one. This is illustrated in the picture below. Algebraically a 'join' between two nodes is written $x^+ \cdot y^-$, and the graphs below illustrate the general rule $(y^+ \cdot x^- \otimes t)_{xy} = (t[x/y])_x$.

When two local nodes are joined together
they can be merged, as in $\uparrow \mathrm{copy} = \nu$

A second consequence of the substitution axiom, expressed algebraically, is that the above result holds even for free names: $y^+ \cdot x^- \otimes t = y^+ \cdot x^- \otimes t[x/y]$. In the presence of a join between two nodes, any reference to one is equivalent to a reference to the other. Because it allows substitution even of free names, we call this the *free substitution axiom*.

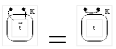
In the presence of a join between two nodes, any reference
to one is equivalent to a reference to the other.

This fact leads to a simple graphical convention: when two names are joined, we merge their nodes and merge the labels on the nodes. Effectively a node represents an equivalence class of names, and the labels on that node represent the names in that equivalence class.

Control axioms When the symmetric action calculus was being designed, it was not clear to the author what role was played by controls or how they should interact with names. An arbitrary general principle was formulated that controls should be transparent to names. This led directly to the two control axioms.

$$\begin{array}{ll} \kappa\text{-scope}: & K((t)_x) = (K(t))_x \\ \kappa\text{-name}: & K(x^+ \cdot y^- \otimes t) = x^+ \cdot y^- \otimes K(t) \end{array}$$

$K((t)_x) = (K(a))_x$



$K(x^+ \cdot y^- \otimes t) = x^+ \cdot y^- \otimes K(t)$

There appears to be a problem with these control axioms. In the proposal we explain why they are not useful, and suggest how they might be removed.

4 SYMMETRIC CLOSED ACTION CALCULI

In the previous sections we have examined the conventional action calculi and the symmetric action calculi. Both were presented first as term algebras and then as graphs. A feature of the term algebras is their use of names to indicate links. In the expression $(x^+ \otimes x^+)_x$, for instance, the fact that the two datums have the same name implies that they are linked. This connection— implicit in the term algebra—is made explicit in the graphs, where a line is actually drawn from the node to the datums. We might say that in a graph the names are 'wired by hand'.

In this section we first present the conventional *closed* action calculi, and then give the symmetric version. The closed calculi are name-free term algebras that, like action graphs, make explicit all wiring. (We use the term *open* to refer to the action calculi with names.) The closed calculi bring two benefits. First, they correspond more closely to category theory. This external connection will hopefully reassure us that we are on the right track. Second, while there were several choices for how to make the conventional action calculi symmetric, there is less leeway for the closed.

The main result regarding the closed symmetric action calculi is that they are isomorphic to the open symmetric action calculi.

Conventional closed reflexive action calculus Just as for the open action calculus, the closed calculi are defined as the quotient of a term algebra. See [Gardner 1995] for full details. (The notation in this report has been adapted for internal consistency: Gardner writes Δ instead of $\lhd$).

$$t ::= \mathbf{id}_m \mid s \cdot t \mid s \otimes t \mid K_m(\vec{s}) \mid \mathbf{perm}_{m,n} \mid \uparrow_m t \mid \lhd_m \mid \omega_m$$

$$\mathbf{id}_m : m \to m \qquad \mathbf{perm}_{m,n} : m \otimes n \to n \otimes m \qquad \lhd_m : m \to m \otimes m \qquad \omega_m : m \to \epsilon$$

$$\frac{s : k \to \ell \qquad t : \ell \to m}{s \cdot t : k \to m} \qquad \frac{s : k \to m \qquad t : \ell \to n}{s \otimes t : \ k \otimes \ell \to m \otimes n} \qquad \frac{t : m \otimes \ell \to m \otimes n}{\uparrow_m t : \ell \to n}$$

$$\frac{t_1 : \ell_1 \to n_1 \ \cdots \ t_r : \ell_r \to n_r}{K_m(t_1, \cdots, t_r) : m \otimes \ell \to m \otimes n}$$

Axioms The action structure axioms A1–A6 from the open action calculus section hold also for the closed. In addition we have the following:

$$\begin{aligned}
&\text{B1}: && \lhd_m \cdot (\omega \otimes \mathbf{id}) = \mathbf{id} \\
&\text{B2}: && \lhd_m \cdot \mathbf{perm}_{m,m} = \lhd_m \\
&\text{B3}: && \mathbf{perm}_{k,m} \cdot (s \otimes t) = (t \otimes s) \cdot \mathbf{perm} \ell, n \\
&\text{B4}: && \mathbf{perm}_{m,n} \cdot \mathbf{perm} n, m = \mathbf{id} \\
&\text{B5}: && \mathbf{perm}_{m \otimes n,k} = (\mathbf{id} \otimes \mathbf{perm}_{n,k}) \cdot (\mathbf{perm}_{m,n} \otimes \mathbf{id}) \\
&\text{B6}: && \omega_{m \otimes n} = \omega_m \otimes \omega_n \\
&\text{B7}: && \lhd_{m \otimes n} = (\lhd_m \otimes \lhd_n) \cdot (\mathbf{id} \otimes \mathbf{perm}_{m,n} \otimes \mathbf{id}) \\
&\text{B8}: && \lhd_m \cdot (\lhd_m \otimes \mathbf{id}) = \lhd_m \cdot (\mathbf{id} \otimes \lhd_m)
\end{aligned}$$

$$\begin{aligned}
&\boldsymbol{\rho}_1: && \mathbf{id} = \uparrow_m \mathbf{perm}_{m,m} \\
&\boldsymbol{\rho}_2: && \uparrow_m t \otimes id = \uparrow_m (t \otimes \mathbf{id}) \\
&\boldsymbol{\rho}_3: && (\uparrow_m s) \cdot t = \uparrow_m (s \cdot (\mathbf{id}_m \otimes t)) \\
&\boldsymbol{\rho}_4: && s \cdot (\uparrow_m t) = \uparrow_m ((\mathbf{id}_m \otimes s) \cdot t) \\
&\boldsymbol{\rho}_5: && \uparrow_n \uparrow_m t = \uparrow_m \uparrow_n ((\mathbf{perm}_{n,m} \otimes \mathbf{id}) \cdot t \cdot (\mathbf{perm}_{m,n} \otimes \mathbf{id})) \\
&\boldsymbol{\rho}_6: && \uparrow_{m \otimes n} t = \uparrow_n \uparrow_m t
\end{aligned}$$

$$\begin{aligned}
&\boldsymbol{\kappa}_1: && K_{q \otimes p}(\omega_q \otimes t) = \omega_q \otimes K_p(t) \\
&\boldsymbol{\kappa}_2: && K_{n \otimes p \otimes q}((\mathbf{id}_n \otimes \mathbf{perm}_{p,q} \otimes \mathbf{id}) \cdot t) = (\mathbf{id}_n \otimes \mathbf{perm}_{p,q} \otimes \mathbf{id}) \cdot K_{n \otimes q \otimes p}(t) \\
&\boldsymbol{\kappa}_3: && K_{n \otimes p}((\mathbf{id}_n \otimes \lhd_p \otimes \mathbf{id}) \cdot t) = (\mathbf{id}_n \otimes \lhd_p \otimes \mathbf{id}) \cdot K_{n \otimes p \otimes p}(t)
\end{aligned}$$

Each control in the normal action calculi corresponds to an indexed family of controls in the closed action calculi. This provides an interface between free names inside the control, and the names outside. This interface is described more fully below in the context of controls in the symmetric action calculi.

Details of the translation between the open and closed convention reflexive action calculus and of the isomorphism between the two can be found in [Gardner 95].

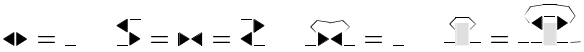
Closed symmetric action calculi The closed symmetric action calculi builds upon the closed reflexive calculi. We remove the ω operator and add a new operator, $\rhd_m$, with the following arity rule:

$$\rhd_m : m \to m \otimes m$$

All the action-structure structure axioms A1–A6 carry over from the closed reflexive calculus. So too do the axioms B2, B3, B4, B5, B7 and B8 which do not contain ω, along with their mirror-images, and the reflexion axioms $\boldsymbol{\rho}_1$ to $\boldsymbol{\rho}_6$. In addition there are four new axioms describing the interaction between the operators *merge* $\rhd$ and *copy* $\lhd$, and several new control axioms. The translation from open calculus to closed will motivate the control axioms: we therefore present the control axioms only after giving the translation.

All of our axioms can be generated from the following intuition. Consider the merge operator with its two input ports and one output. We can say that it asserts that the wires attached to all three ports are equal. The copy operator does essentially the same thing. The reflexion operator is drawn as a loop, and asserts that the wire at the start of the loop is equal to the wire at the end. Given this intuition all the structure axioms are obvious. Here are the four axioms relating merge and copy:

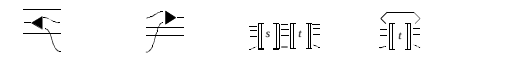
$$\begin{aligned}
\text{line}: &\quad \lhd_m \cdot \rhd_m \;=\; \mathbf{id}_m \\
\text{railroad}: &\quad (\lhd_m \otimes \mathbf{id}_m)\cdot(\mathbf{id}_m \otimes \rhd_m) \;=\; \rhd_m \cdot \lhd_m \;=\; (\mathbf{id}_m \otimes \lhd_m)\cdot(\rhd_m \otimes \mathbf{id}_m) \\
\text{bow-tie}: &\quad \uparrow_n \rhd_n \cdot \lhd_n \;=\; \mathbf{id}_n \\
\text{loop-same}: &\quad \uparrow_n t \;=\; \uparrow\Big((\lhd_n \otimes \mathbf{id})\cdot(\mathbf{id}_n \otimes t)\cdot(\rhd_n \otimes \mathbf{id})\Big)
\end{aligned}$$



(line) (railorad) (bow-tie) (loop-same)

Translation from open symmetric calculi to closed, using a name-line Translating from the open to the closed symmetric calculus involves a *name-line* containing all used names. The expression $[\![t]\!]_{xyz}$ means 'translate t from the open to the closed, with respect to a name-line carrying x, y and z.' The following equations give the translation from open action calculus to closed. (The rules governing the translation of controls are explained below.) Observe that a name in the open calculus is translated by a reflexive loop in the closed.

$$\begin{aligned}
[\![\mathbf{id}_n]\!]_{\vec{x}} &= \mathbf{id}_{|\vec{x}|\otimes n} \\
[\![s\cdot t]\!]_{\vec{x}} &= [\![s]\!]_{\vec{x}} \cdot [\![t]\!]_{\vec{x}} \\
[\![s\otimes t]\!]_{\vec{x}} &= [\![s]\!]_{\vec{x}} \cdot (\mathbf{id}_{|\vec{x}|} \otimes \mathbf{perm}) \cdot [\![t]\!]_{\vec{x}} \cdot (\mathbf{id}_{|\vec{x}|} \otimes \mathbf{perm}) \\
[\![(t)_y]\!]_{\vec{x}} &= \uparrow_{|y|} [\![t[z/y]]\!]_{z,\vec{x}}, \quad z \notin \vec{x} \\
[\![y^+]\!]_{\vec{x},y,\vec{z}} &= (\mathbf{id}_{|\vec{x}|} \otimes \lhd_p \otimes \mathbf{id}_{|\vec{z}|}) \cdot (\mathbf{id}_{|\vec{x}|} \otimes \mathbf{id}_p \otimes \mathbf{perm}_{p,|\vec{z}|}) \\
[\![y^-]\!]_{\vec{x},y,\vec{z}} &= (\mathbf{id}_{|\vec{x}|} \otimes \mathbf{id}_p \otimes \mathbf{perm}_{|\vec{z}|,p}) \cdot (\mathbf{id}_{|\vec{x}|} \otimes \rhd_p \otimes \mathbf{id}_{|\vec{z}|}) \\
[\![K(\vec{s})]\!]_{\vec{x}} &= K_{|\vec{x}|}([\![\vec{s}]\!]_{\vec{x}})
\end{aligned}$$



$[\![y^+]\!]_{xyz}$ $[\![x^-]\!]_{xyz}$ $[\![s\cdot t]\!]$ $[\![(t)_y]\!]_x$

Translation from closed symmetric calculi to open The translation from closed action calculus to open is as follows.

$$\begin{array}{rcl}
|\mathbf{id}| & = & \mathbf{id} \\
|s \cdot t| & = & |s| \cdot |t| \\
|s \otimes t| & = & |s| \otimes |t| \\
|\mathbf{perm}| & = & \mathbf{perm} \\
|\uparrow t| & = & \uparrow |t| \;=\; ((x^+ \otimes \mathbf{id}) \cdot t \cdot (x^- \otimes \mathbf{id}))_x \\
|\lhd| & = & \lhd \;=\; (x^- \otimes x^+ \otimes x^+)_x \\
|\rhd| & = & \rhd \;=\; (x^- \otimes x^- \otimes x^+)_x \\
|K_m(\vec{s})| & = & \Big(\vec{x}^- \otimes \vec{x}^+ \otimes K\Big((x^+ \otimes \mathbf{id}) \cdot |\vec{s}| \cdot (x^- \otimes \mathbf{id})\Big)\Big)_{\vec{x}}
\end{array}$$

Controls fit into the state-line Controls in the closed calculus fit directly into the state-line and are indexed according to the arity of the state-line. Thus the control K_{pqp} fits into a state-line with three names of arity p, q and p in that order. What in the open calculus is a single control, in the closed calculus is a family of related controls. This is the same as in Gardner's closed conventional calculus.

$$[\![K(s)]\!]_{\vec{x}} \quad = \quad K_{|\vec{x}|}([\![s]\!]_{\vec{x}})$$

The intuition is that those wires going into its state-line ports are equal to the wires going out. This intuition is made explicit by the axiom K-same. The other control axioms all indicate how changes to the state-line outside a control affect its contents: essentially they allow that $K(t)[y/x] = K(t[y/x])$. These axioms fulfil the principle stated earlier: that controls should be transparent to names. (In the proposal it is explained why removing axiom $\boldsymbol{\rho}_7$ might be desirable.)

$$\begin{array}{rl}
\text{K-same}: & K_n(s) \;=\; (\lhd_n \otimes \mathbf{id}) \cdot (\mathbf{id}_n \otimes K(s)) \cdot (\rhd_n \otimes \mathbf{id}) \\
\text{K-surround}: & K_{mpn}(\vec{a}) \;=\; (\mathbf{id} \otimes \lhd_p \otimes \mathbf{id}) \cdot K_{mppn}\Big((\mathbf{perm}_{m,p} \otimes \mathbf{id}) \cdot (\mathbf{id}_p \otimes \vec{a}) \cdot (\mathbf{perm}_{p,m} \otimes \mathbf{id})\Big) \cdot (\mathbf{id} \otimes \rhd_p \otimes \mathbf{id}) \\
\text{K-perm}: & (\mathbf{id} \otimes \mathbf{perm} \otimes \mathbf{id}) \cdot K_{mpqn}(\vec{a}) \cdot (\mathbf{id} \otimes \mathbf{perm} \otimes \mathbf{id}) \;=\; K_{mqpn}\Big((\mathbf{id} \otimes \mathbf{perm} \otimes \mathbf{id}) \cdot \vec{a} \cdot (\mathbf{id} \otimes \mathbf{perm} \otimes \mathbf{id})\Big) \\
\text{K-line}: & K_{pm}(\mathbf{id}_p \otimes \vec{a}) \;=\; \mathbf{id}_p \otimes K_m(\vec{a}) \\
\text{K-inject}: & (\mathbf{id} \otimes (\rhd \cdot \lhd) \otimes \mathbf{id}) \cdot K_{mppn}(a) \;=\; K_{mppn}\Big((\mathbf{id} \otimes (\rhd \cdot \lhd) \otimes \mathbf{id}) \cdot \vec{a}\Big) \\
\boldsymbol{\rho}_7: & K_m(\uparrow_n t) \;=\; \uparrow_n K_{nm}(t)
\end{array}$$

Results The important result is that the closed symmetric calculi are isomorphic to the open:

$$\begin{array}{l}
s =_{open} t \Rightarrow [\![s]\!] =_{closed} [\![t]\!] \\
s =_{closed} t \Rightarrow |s| =_{open} |t| \\
|[\![t]\!]| =_{open} t \\
[\![|t|]\!] =_{closed} t
\end{array}$$

There is an interesting result regarding the translation from the open to the closed. In the open term $s \otimes t$ we could have s first followed by t, or t followed by s, or both in parallel. This happens because the name-line is preserved across translated terms.

The significance of this result is not yet understood. It is explored in the proposal, in connection with the premonoidal graphs of Jeffrey.

5 SYMMETRIC MOLECULECULAR FORM

Molecular forms can be viewed as weak normal forms for the term algebra introduced previously. Molecular forms subsume many of the equivalences that must be derived in the term algebra: if two molecular forms are equal, then they will be written down the same (up to an equivalence relation on names). Molecular forms are also closely related to action graphs.

In previous sections we had first introduced the conventional reflexive calculus and then given the symmetric form. In this section however we only present symmetric molecular forms, for reasons of simplicity. An account of conventional reflexive molecular forms may be found in [Milner 1994].

Name-equivalence relations As mentioned in a previous section, nodes in action graphs represent equivalence-classes of names. These equivalence classes are key to the symmetric molecular forms; they are represented with a *name-equivalence relation*. We say that two terms are *name-equivalent* if they are equal up to their name equivalence classes.

This equivalence relation is an abstraction of the properties of 'joins' x^+y^- in the algebra: they are reflexive ($\mathbf{id} = \mathbf{id} \otimes x^+ \cdot x^-$), symmetric ($x^+ \cdot y^- = x^+ \cdot y^- \otimes y^+ \cdot x^-$) and transitive ($x^+ \cdot y^- \otimes y^+ \cdot z^- = x^+ \cdot y^- \otimes y^+ \cdot z^- \otimes x^+ \cdot z^-$).

Consider α-conversion of some term involving the names x, x, y and z in that order. Any α-conversion upon the term would end with the first two names still related, and the last two names still unrelated. We see that α-conversion does not change the equivalence classes of names in a term: it is a special case of name-equivalence.

Definition A term in molecular form is called an *action*. Actions $a, b \ldots$ and *molecules* $\mu, \eta \ldots$ are defined by the following abstract grammar:

$$\begin{array}{rcll} a & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R} & a : |\vec{x}| \to |\vec{y}| \\ \mu & ::= & \vec{x} \cdot K a^{\rightarrow}_{int} \cdot \vec{y} & K a^{\rightarrow}_{int} : |\vec{x}| \to |\vec{y}| \\ a_{int} & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y}) & a_{int} : |\vec{x}| \to |\vec{y}| \end{array}$$

In a and a_{int}, the ordered lists $\vec{x}$ and $\vec{y}$ are the input and output ports, $\vec{\mu}$ is a multiset of molecules, $\vec{z}$ is a set of local names and R is an arity-respecting equivalence relation between names. The subscript $()_{\vec{z}}$ restricts the *scope* of names; and any occurrence of a name *assigns* it. (In the proposal we suggest a modification to this definition.)

The following example illustrates the link between molecular forms and graphs:

$$(xy\ ;\ x \cdot K \cdot zy\ ;\ y)_{x\ ;\ x=z}$$

Operations on molecules The four operations on actions are as below. Let $a = (\vec{x_1}\ ;\ \vec{\mu}\ ;\ \vec{x_2})_{\vec{x}\ ;\ R}$, $b = (\vec{y_1}\ ;\ \vec{\eta}\ ;\ \vec{y_2})_{\vec{y}\ ;\ S}$, $c = (\vec{x_1}\vec{y_1}\ ;\ \vec{\mu}\ ;\ \vec{x_2}\vec{y_2})_{\vec{z}\ ;\ R}$ with $\vec{x}$ and $\vec{y}$ disjoint, with y present in R only as a singleton and similarly x in S. These conditions are present to ensure that there is no clash between names in one term and local names in the other. (We write $R \uplus S$ for the closure of $R \cup S$ under equivalence.)

$$\begin{array}{rcll} a \otimes b & = & (\vec{x_1}\vec{y_1}\ ;\ \vec{\mu}\vec{\eta}\ ;\ \vec{x_2}\vec{y_2})_{\vec{x}\vec{y}\ ;\ R \uplus S} & \\ a \cdot b & = & (\vec{x_1}\ ;\ \vec{\mu}\vec{\eta}\ ;\ \vec{y_2})_{\vec{x}\vec{y}\ ;\ R \uplus S \uplus \vec{x_2} = \vec{y_1}} & |x_2| = |y_1| \\ \uparrow_m c & = & (\vec{y_1}\ ;\ \vec{\mu}\ ;\ \vec{x_2})_{\vec{z}\ ;\ R \uplus \vec{x_1} = \vec{x_2}} & |\vec{x_1}| = |\vec{x_2}| = m \\ (a)_{\vec{y}} & = & (\vec{x_1}\ ;\ \vec{\mu}\ ;\ \vec{x_2})_{\vec{y}\vec{x}\ ;\ R} & \vec{x}, \vec{y} \text{ disjoint} \end{array}$$

Axioms for molecular form There are two straightforward axioms. The ν axiom allows us to introduce a fresh local name that may be equal to any other name, or in reverse to dispense with a local name if it is unused. The σ axiom allows any occurrence of a name in the action to be replaced by any other name from the same equivalence class. (We write $R - v$ for an equivalence relation which contains v only as a singleton equivalence class.)

$$\begin{array}{rrcll} \nu. & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R - v} & = & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{v\vec{z}\ ;\ R} & (v \text{ not in } \vec{x}, \vec{y}, \vec{\mu}, \vec{z}) \\ \sigma. & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R} & = & (\sigma\vec{x}\ ;\ \sigma\vec{\mu}\ ;\ \sigma\vec{y})_{\vec{z}\ ;\ R} & (\sigma = \{u/v\}, uRv) \end{array}$$

Translation The following table gives the translation from symmetric molecular forms into the symmetric action calculi:

$$\begin{aligned} \widehat{R} &= \bigotimes x^+ \cdot y^- \quad (xRy, x \neq y) \\ \widehat{\vec{u} \cdot K\vec{b} \cdot \vec{v}} &= \vec{u}^+ \cdot K\widehat{\vec{b}} \cdot \vec{v}^- \\ (\vec{x}\ ;\ \widehat{\vec{\mu}\ ;\ \vec{y}})_{\vec{z}\ ;\ R} &= (\vec{x}^- \otimes \vec{y}^+ \otimes \widehat{\vec{\mu}} \otimes \widehat{R})_{\vec{z}} \end{aligned}$$

And the translation from the symmetric calculi into molecular form is as follows:

$$[\![\mathbf{id}_n]\!] = id_n \qquad [\![s \cdot t]\!] = [\![s]\!] \cdot [\![t]\!] \qquad [\![s \otimes t]\!] = [\![s]\!] \otimes [\![t]\!]$$

$$[\![(t)_y]\!] = ([\![t]\!])_y \qquad [\![y^+]\!] = y^+ \qquad [\![y^-]\!] = y^-$$

$$[\![K(s)]\!] = (\vec{x}\ ;\ \vec{x} \cdot K(\vec{u}; \vec{\mu}; \vec{v}) \cdot \vec{y}\ ;\ \vec{y})_{\vec{x}\vec{y}\vec{z}\ ;\ R} \qquad \text{where } [\![s]\!] = (\vec{u}\ ;\ \vec{\mu}\ ;\ \vec{v})_{\vec{z}\ ;\ R}$$

Results The first result about symmetric molecular forms is that they are isomorphic to the open symmetric calculus. This completes the main result of this paper: that all three formal presentations of the symmetric action calculus are isomorphic.

A second result (not yet formally proved) is that with a particular modification, symmetric molecular forms become a conservative extension of the reflexive molecular forms. This modification is the same one as was required in the term algebra. The modification and the conservativity result are discussed in the proposal.

Research Proposal:
The Role of Names in Graphical Models

Lucian Wischik
Supervised by Philippa Gardner
31st July 1998

1 REVIEW

In the report we indicated how the conventional action calculi might naturally be extended to make them symmetric. In this proposal we review the symmetric action calculi, outline future developments of the calculi, and indicate where they might fit into the broader context. A specific plan of study is detailed in the conclusion.

Action calculi All action calculi start from a symmetric monoidal category. To this we add names, in the form of operators and their associated equivalences. The different sorts of action calculi—conventional, reflexive, higher-order, symmetric—all add different naming operators. Finally we add control operators, and rules to govern the dynamical interaction of the controls; through the choice of controls and rules we can tailor the calculi to their particular applications.

Symmetric action calculi The symmetric action calculi have three operators for names: datum x^+, inverse datum x^- and scope-restriction $(\cdot)_x$. We give below the abstract grammar for terms in the symmetric calculi, and the additional axioms for the three operators. (As will be explained in the next section, it seems desirable simply to remove the two control axioms.)

$$t ::= \mathbf{id} \mid s \cdot t \mid s \otimes t \mid K(\vec{t}) \mid (t)_x \mid x^+ \mid x^-$$

$$\begin{array}{rl}
\text{A7:} & (\mathbf{id})_x = \mathbf{id} \\
\text{A8:} & (s \cdot t)_x = (x^+ \otimes s)_x \cdot (x^- \otimes t)_x \\
\boldsymbol{\gamma}: & (t \otimes \mathbf{id})_x = (t)_x \otimes \mathbf{id}, \quad (\mathbf{id} \otimes t)_x = \mathbf{id} \otimes (t)_x \\
\boldsymbol{\delta}: & (x^+ \otimes x^-)_x = \mathbf{id} \\
\boldsymbol{\sigma}: & (y^+ \cdot x^- \otimes t)_x = (x^+ \cdot y^- \otimes t)_x = (t[y/x])_x \\
\kappa\text{-scope}: & K((t)_x) = (K(a))_x \\
\kappa\text{-name}: & K(x^+ \cdot y^- \otimes t) = x^+ \cdot y^- \otimes K(t)
\end{array}$$

While the above axioms are adequate, we hope to find an alternative set of axioms closer in style to those of Hasegawa. The expectation is that such an alternative set would be more concise and expressive.

Other possible symmetric operators were considered in the report, instead of x^+, x^- and $(\cdot)_x$. These three were adopted for reason of their simplicity; but the other possibilities merit further investigation.

Examples The original motivation for the symmetric action calculi was to remove irregularities from the conventional action calculi (where, for instance, ω is taken as a basic primitive but ν is not). Justification, if any, will come from the ways in which the symmetric action calculi relate to other calculi.

The next section starts with a symmetric action calculus that corresponds to that fragment of the π-calculus that has guards but no summation or replication. While the conventional action calculi do not allow names to merge, the symmetric calculi do: this allows for the more natural idiom pictured below. (It is this application of the symmetric calculi that motivates the removal of the two control axioms.)

The next example is Parrow and Victor's fusion calculus. The fusion calculus is a simplification and extension of the π-calculus. It has a number of similarities with the symmetric calculi: it uses, for instance, a scope operator with exactly the same properties. (This application of the symmetric action calculi appears to require a different substitution axiom $\boldsymbol{\sigma}$. We discuss this point in the next section.) The fusion calculus has some useful results about bisimulation; perhaps they can be carried over to the symmetric action calculus.

The third example is the relation between the symmetric action calculi and the reflexive. If we remove the two control axioms, as suggested above, then the symmetric action calculi appear to become a conservative extension of the reflexive action calculi. (The formal proof of this result has not yet been completed).

Closed action calculi A central feature of the term algebra above is its use of names. But action calculi may also be presented graphically with action graphs (detailed in section 2 of the report): these action graphs are largely name-free, using names only where they are free. And closed action calculi (report, section 4) which are isomorphic to the term algebra are wholly name-free. The action calculi thereby provide a metric with which graphical and non-graphical models of computation may be compared.

Graphs The symmetric action calculi in particular have some similarities to a number of name-free graphical models. While conventional action graphs are directed, symmetric action graphs are not: in this way they are similar to Honda's *undirected actions*. The symmetry of the symmetric calculus is also reminiscent of Lafont's interaction nets. These similarities merit further investigation and are discussed in the final section of this proposal.

Additionally, the translation from the open symmetric calculi to the closed makes use of a 'name-line'. This name-line appears similar to the state-line in Jeffrey's premonoidal categories. (It is interesting that Parrow and Victor's fusion calculus also has updates to a shared state. Perhaps the three are related.)

2 EXAMPLES

π-calculus In this section we present a symmetric action calculus which corresponds to a particular fragment of the π-calculus. This fragment has input guarding but no choice or replication. Its terms are generated as follows; they obey the standard reaction rules.

$$P ::= \mathbf{0} \quad | \quad \overline{x}\langle y\rangle \quad | \quad x(y)\cdot P \quad | \quad P|Q \quad | \quad (\nu x)P$$

A first attempt at a corresponding symmetric action calculus uses two controls, **in** and **out**, as follows.

$$\text{Arity rules} \quad \frac{}{\mathbf{out}_{pq} : p \to q} \qquad \frac{t : \epsilon \to \epsilon}{\mathbf{in}_{pq}(t) : p \to q}$$

$$\text{Control rule} \quad x^+ \cdot \mathbf{out}_{pq} \cdot y^- \otimes y^+ \cdot \mathbf{in}_{qp}(t) \cdot z^- \searrow x^+ \cdot z^- \otimes t$$

$$\text{Translation} \quad \begin{array}{rl} \widehat{\mathbf{0}} & = \mathbf{id}_\epsilon \\ \widehat{\overline{y}\langle x\rangle} & = y^+ \cdot \mathbf{out}_{|x|,|y|} \cdot x^- \\ \widehat{y(z)\cdot P} & = x^+ \cdot \mathbf{in}_{|y|,|z|}(\widehat{P}) \cdot z^- \\ \widehat{P|Q} & = \widehat{P} \otimes \widehat{Q} \\ \widehat{(\nu x)P} & = (\widehat{P})_x \end{array}$$

The above symmetric action calculus lets us test the consequences of the two control axioms, κ-scope and κ-name. Consider the following equalities:

$$\begin{array}{rl} & \widehat{(\nu y)z(x)\cdot P} \\ = & (z^+ \cdot \mathbf{in}(\widehat{P}) \cdot x^-)_y \\ = & z^+ \cdot \mathbf{in}((P)_y) \cdot x^- \\ = & \widehat{z(x)\cdot(\nu y)P} \end{array}$$

The $\boldsymbol{\kappa}$-scope axiom has allowed a local name to 'float through' a control. We term this phenomenon the *floating* ν. It is clearly undesirable in this context. To prevent νs from floating it is enough simply to remove the two control axioms κ-scope and κ-name from the symmetric action calculus. For the closed symmetric calculus it appears sufficient to remove the axiom $\boldsymbol{\rho}_7$:

$$\boldsymbol{\rho}_7 : \quad K_m(\uparrow_n t) = \uparrow_n K_{nm}(t)$$

The removal of floating νs from the molecular forms appears to actually make them simpler. Originally, actions in molecular form were as defined by the following abstract grammars:

$$\begin{array}{rcll} a & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R} & a : |\vec{x}| \to |\vec{y}| \\ \mu & ::= & \vec{x}\cdot K\vec{a_{int}}\cdot\vec{y} & K\vec{a_{int}} : |\vec{x}| \to |\vec{y}| \\ a_{int} & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y}) & a_{int} : |\vec{x}| \to |\vec{y}| \end{array}$$

It seems that floating νs can be prevented by simplifying the definition:

$$\begin{array}{rcll} a & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R} & a : |\vec{x}| \to |\vec{y}| \\ \mu & ::= & \vec{x}\cdot K\vec{a}\cdot\vec{y} & K\vec{a} : |\vec{x}| \to |\vec{y}| \end{array}$$

In the π-calculus, νs may float between other νs but not between actions. In the presence of replication a ν should certainly not float: consider the difference between $\uparrow((\nu x)\overline{z}\langle x\rangle)$ and $(\nu x)\uparrow(\overline{z}\langle x\rangle)$. Similarly ν may not float in the ν-calculus of Pitts and Stark [1997]. The fusion calculus appears to allow νs to float, and so too does the ambient calculus. The question of floating νs remains open.

Fusion calculus The fusion calculus of Parrow and Victor [1998] is a variant of the π calculus which has a shared state and a scoping construct. Preliminary attempts at a corresponding symmetric action calculus have suggested that the existing substitution axiom is too strong. It is currently as follows:

$$\boldsymbol{\sigma} : (y^+ \cdot x^- \otimes t)_x = (x^+ \cdot y^- \otimes t)_x = (t[y/x])_x$$

As explained in the report, this axiom allows substitution of free names: $y^+ \cdot x^- \otimes t = y^+ \cdot x^- \otimes t[y/x]$. In the presence of a *join* $y^+ \cdot x^-$ between two names, any reference to one is equivalent to a reference to the other. For this reason we call it the *free substitution axiom.*

A different substitution axiom might be considered:

$$\boldsymbol{\sigma'} : (y^+ \cdot x^- \otimes t)_{xy} = (x^+ \cdot y^- \otimes t)_{xy} = (t[y/x])_y$$

This axiom allows the merging of bound names but not that of free. For this reason we call it the *bound substitution axiom.* Bound substitution appears to have some advantages. It seems that for any expression in a free-substitution symmetric calculus we can write a functional equivalent in a bound-substitution calculus, but not vice versa. And bound substitution appears to be a natural way of expressing the fusion calculus.

Graphically, the bound substitution axiom only allows two nodes to merge when they are both local. Names no longer form equivalence classes, and a very different molecular form would be needed. At the level of the closed symmetric calculi, it appears that bound substition would replace the railroad axiom with an awkward alternative. This axiom is currently under investigation.

Symmetric calculi are a conservative extension of reflexive Recall that symmetric molecular forms are defined as follows. (As discussed above, we disallow floating νs.)

$$\begin{array}{lcll} a & ::= & (\vec{x}\ ;\ \vec{\mu}\ ;\ \vec{y})_{\vec{z}\ ;\ R} & a : |\vec{x}| \to |\vec{y}| \\ \mu & ::= & \vec{x} \cdot K\vec{a} \cdot \vec{y} & K\vec{a} : |\vec{x}| \to |\vec{y}| \end{array}$$

The reflexive action calculi require each name to be assigned exactly once, either on the right-hand side of a control or in the list $\vec{x}$ of input ports in the action a above. By contrast the symmetric action calculi allow a name to be assigned any number of times. So, for example, the symmetric term $(x\ ;\ u.K.v\ ;\ y)_{\ ;\ id}$ corresponds to a term in the reflexive calculi; but $(x\ ;\ u.K.x\ ;\ y)_{\ ;\ id}$ does not—because x is assigned both in the input list and to the right of K.

Briefly, a symmetric term $(\vec{x}\ ;\ \vec{u} \cdot K\vec{a} \cdot \vec{v}, \ldots\ ;\ \vec{y})_{\vec{z}\ ;\ R}$ corresponds to a reflexive term if $\vec{x}$, $\vec{v_i}$ and $\vec{z}$ are disjoint, if $\vec{z}$ contains all of $\vec{x}$ and $\vec{v_i}$, and if R is the identity on names.

The important result is that the symmetric action calculi appear to be a conservative extension of the reflexive. This proof has not yet been verified. [2] [3]

Further developments There are some further possible developments of the symmetric action calculi. The fusion calculus has convenient results concerning bisimulation; perhaps, if we find a symmetric calculus that corresponds to the fusion calculus, these results will carry over. It seems too that a link with category theory would provide independent validation. In particular, Hyland has remarked that the closed symmetric calculi appear to be similar to a Frobenius algebra. This area requires further research.

[2] The key step in the proof is to show that if a and b are equal symmetric terms which both correspond to reflexive terms, then their corresponding reflexive terms are equal. Proof: note that neither of the two symmetric axioms ν nor σ change the equivalence classes of names. Since R is the identity in both a and b, they must both have singleton equivalence classes and so must be related through alpha-conversion.

[3] What in the reflexive calculi is called *reflexive substitution*, is in the symmetric calculi an alpha-conversion to avoid clashes, then a merging of two equivalence classes, then a culling of superfluous names to restore the equivalence relation to the identity. (Perhaps this is why the rules for reflexive substitution look so complicated!)

3 GRAPHICAL MODELS

Role of names The symmetric action calculi separate two aspects of the role of names in binding: *scoping* and *assignment*. Consider the π-calculus expression $\overline{x}\langle y\rangle \mid x(z)\cdot P$, where a single construct indicates both that z is restricted in scope to P, and that it will be assigned the value y. When this term is expressed as a symmetric calculus, the scoping is indicated with $(\cdot)_z$ and the assignment with z^-.

Because all wiring by done 'by hand', each closed calculus makes explicit the role of names in that calculus. This makes the closed calculi a useful tool for comparing the role of names in different calculi.

Undirected graphs The conventional action calculi are directed: names may be bound only at the right-hand side of a control. In the symmetric action calculi, while controls still do have directions, they are not essential. As shown below, a simple context can flip a wire from one side of a term to the other:

So although terms have an input and an output, the difference between the two is merely a convenience. We could imagine an alternative description of the symmetric calculi in which all ports are numbered and we connect them with a list of wiring instructions: 'connect wire s_1 to t_2, s_4 to t_1'. Such a description would be convenient in a graphical presentation but is awkward in text.

The fact that these terms are undirected suggests a connection with Honda's *undirected actions* [1997]. Indeed, the above operation of flipping a wire from one side to another is known as *switching* in undirected actions. And because the symmetric calculi allow us to connect any node with any other node they may be related to Lafont's *interaction combinators* [1997]. Both areas are open for future study.

Jeffrey's premonoidal categories Jeffrey [1997] describes a formalisation of data- and control-flow graphs. The following diagram illustrates how he represents the program `x:=1; return x;`

The line at the top is both a *state line* storing the contents of the variable, and a *control line* specifying the causal order of the program. These program graphs are strongly reminiscent of the translation from the open symmetric calculi to the closed, which makes use of a similar line containing all local variables.

Jeffrey's graphs also have higher-order features. Wrapping up a graph inside a box represents a function; there is a special node to apply a function. In the presence of reflexion these higher-order features allow for recursive functions. Perhaps these graphs relate to a higher-order symmetric calculus.

There is an important difference between Jeffrey's graphs and the translation from open to closed symmetric calculi. Controls in his graphs are generally imperative, and the graphs are sequential: the order of *set* and *get* above cannot be exchanged. By contrast, the symmetric action calculi always allow for such controls to be swapped.

We might be making a category error. The control-line in Jeffrey's graphs describes the dynamics of a program; the name-line in the symmetric action calculi describes the static distribution of names throughout the program, with dynamics being a separate reaction relation. Perhaps this difference arises because dataflow is undirected in the symmetric action calculi, but control flow should always directed.

Jeffrey shows that his graphs correspond to a symmetric premonoidal category. It would be interesting to compare the graphs with the closed symmetric calculi at the level of category theory. It would also be interesting to relate his higher-order features to a higher-order symmetric calculus.

Plan of study The most urgent results are those establishing the validity of the axioms for the symmetric calculi. First, this involves the removal of the control axioms; then we will be able to demonstrate a symmetric action calculus that corresponds to a fragment of the π-calculus. Second, it involves finding a symmetric calculus that corresponds to the fusion calculus. Perhaps we will be able to show that the extension from π-calculus to fusion calculus is equivalent to the extension from conventional action calculi to symmetric action calculi. Honda's undirected actions might also be studied at this stage.

In the medium term, work will focus on a categorical understanding of the closed symmetric calculi. This will involve studying the relation between the closed symmetric calculi and the Frobenius algebras, and the connection with Jeffrey's premonoidal graphs.

The ultimate goal of this research is the expression in mathematical formalisms of some aspects of the roles of names.

BIBLIOGRAPHY

[**Gardner 1995**] *A name-free account of action calculi.* Gardner P.A., in Mathematical Foundations of Programming Semantics, ENTCS 1 ed. Brookes S., Main M., Melton A., Mislove M., Elsevier 1995.

[**Honda 1997**] *A short note on undirected actions.* Honda K., in preparation.

[**Jeffrey 1997**] *Premonoidal categories and a graphical view of programs.* Jeffrey A., http://www.cogs.susx.ac.uk/users/alanje/premon/

[**Lafont 1997**] *Interaction Combinators.* Lafont Y., Information and Computation vol. 137 no. 1.

[**Milner 1994**] *Action Calculi V: Reflexive Molecular Forms.* Milner R., ftp://ftp.cl.cam.ac.uk/users/rm135/ac5.ps

[**Milner 1996**] *Calculi for interaction.* Milner R., Acta Informatica vol. 33 no. 8, Nov. 1996.

[**Parrow 1998**] *The fusion calculus: expresiveness and symmetry in mobile processes.* Parrow J., Victor B., Proceedings of LICS June 1998; also Ph.D. thesis, Dept. of Computer Systems, Uppsala University, Sweden.

[**Stark 1997**] *Names, equations, relations: practical ways to reason about new.* Stark I., BRICS RS-97-39, Dept. of Computer Science, Aarhus.