Local Bigraphs and Confluence: Two Conjectures

(Extended Abstract)

Robin Milner

University of Cambridge

Abstract

The notion of confluence is studied on the context of bigraphs. Confluence will be important in modelling real-world systems, both natural (as in biology) and artificial (as in pervasive computing). The paper uses bigraphs in which names have multiple locality; this enables a formulation of the lambda calculus with explicit substitutions. The paper reports work in progress, seeking conditions on a bigraphical reactive system that are sufficient to ensure confluence; the conditions must deal with the way that bigraphical redexes can be intricately intertwined. The conditions should also be satisfied by the lambda calculus. After discussion of these issues, two conjectures are put forward.

Keywords: bigraph, locality, confluence, lambda calculus.

Bigraphs have been used to present a variety of models of concurrency within a single framework, which also provides a theory applicable to all the models. As we seek informatic understanding of extensive real-life systems that reconfigure themselves, we cannot expect that our present repertoire of abstract process calculi (including Petri nets, mobile ambients, CSP and π-calculus) will suffice. So, as we enlarge our repertoire of calculi —perhaps specific to a certain application (e.g. in biology or in pervasive computing)— there is a need for unifying theory.

The bigraphical model is an experiment in this direction. It is not a specific calculus, but rather a framework for defining and combining such calculi. To define a specific bigraphical reactive system (BRS) two ingredients are needed: its *signature* defines its *controls* (the kinds of nodes allowed), and its *reaction rules* define how bigraphs can reconfigure themselves.

Already the model has yielded some elements of a theory, especially of labelled transitions and behavioural congruences [6,4,5,7], which is applicable to a variety of BRSs. The present exercise addresses a different topic. First, in *local bigraphs* [8] we introduce a new treatment of names that allows them to have multiple *locality* (an example follows shortly). Similar work in bigraphs is by Bundgaard and Hilde-

brandt [3]. Second, we study the notion of *confluence* —i.e. independence among actions— in this setting, in the belief that it will arise frequently in applications. One need only think of modelling behaviour within a building: activity at one end of the building is largely independent of activity at the other end.

This summary omits some details, but should be accessible to those unfamiliar with bigraphs. It summarises work whose aims are as follows: to understand how activities in local bigraphs can conflict with one another, leading to non-confluence; to represent the λ-calculus —the classic setting for confluence studies— within local bigraphs; and thereby to learn conditions under which confluence can be assured within this wider setting. The work is in progress; the summary ends with two conjectures.

Mathematical framework: We work in s-categories. They differ from categories in that each arrow f has a *support* $|f|$, a finite set; composition $g \circ f$ is defined only if $|g| \cap |f| = \emptyset$, and then $|g \circ f| = |g| \cup |f|$. Two arrows f and g are *support equivalent*, $f \simeq g$, if they differ only by a bijection between their supports. Support is important for the notion of *occurrence* of one bigraph in another. For example, our Conjecture 1 rests upon analysis of when and how two redex occurrences can overlap each other.

1   Local bigraphs

Local bigraphs are arrows in an s-category whose objects are *interfaces*. An interface $I = \boldsymbol{X} = \langle X_0, \ldots, X_{m-1} \rangle$ has *width* m, a finite ordinal, and assigns to each *location* $i \in m$ a finite set X_i of *names*. The X_i need not be disjoint; thus, for example, any $x \in X_0 \cap X_1$ has dual locality.

If $J = \boldsymbol{Y}$ is another interface with width n, then a local bigraph $G : I \to J$ has m *sites* and n *roots* (or *regions*). Each region contains an unordered tree, whose root is the region and whose other members are either *nodes* or sites; the latter must be leaves. The interfaces dictate an assignment of names to each site and each region; the *inner* and *outer* names of G are those of I and J respectively. The support $|G|$ of G is its set of nodes; we say that F and G *overlap* if their supports are not disjoint.

Figure 1 shows a local bigraph with three sites (shaded) and two roots; the trees are represented by nesting. Each node may have *ports*, the number depending on the node's *kind* or *control* (not shown). The set of ports and inner names is partitioned into *links*; a link is either *free* (an outer name) or *bound* by a *binding port*. The example has two free links, x' and z', and one link bound by a port on the largest node. Binding ports are shown as circles, free ports as bullets.

There is a scoping discipline: if a link is bound, then its inner names and ports must lie within the node that binds it; if a link is free, with outer name x, then x must be located in every region that contains any inner name or port of the link.

The *composition* of $G : I \to J$ with $F : H \to I$, written $G \circ F$, is easy to define graphically: insert the roots of F in the sites of G, joining links at like names and eliding the names. Observe that, via composition, nodes in different regions can become separated by arbitrarily many node boundaries —while still sharing links.
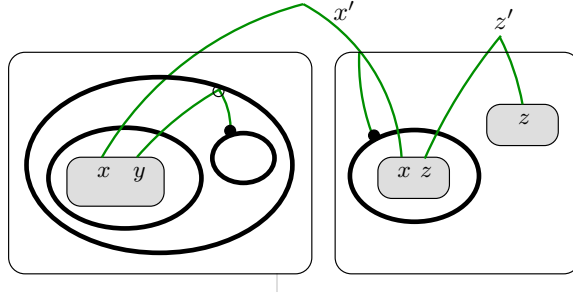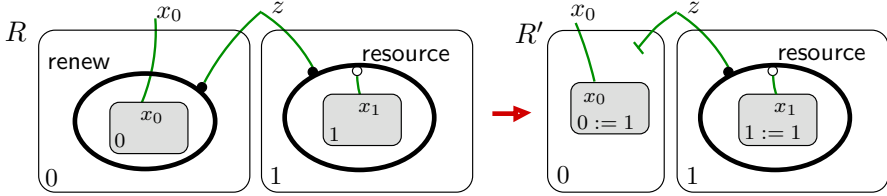
Fig. 1. A local bigraph $G : \langle \{xy\}, \{xz\}, \{z\} \rangle \rightarrow \langle \{x'\}, \{x'z'\} \rangle$

Fig. 2. A parametric reaction rule

An *agent* $a : \epsilon \rightarrow I$ has no sites; ϵ is the trivial interface with width 0. We use lower-case letters for agents.

2   Reaction rules and λ-calculus

We are interested in *parametric (reaction) rules* that reconfigure agents. Such a rule has a redex $R : H \rightarrow K$ and a reactum $R' : H' \rightarrow K$, which may have different numbers m and m' of sites. A *parameter* for the rule is then an agent $a : H \oplus I$, with width m. The interface $H \oplus I$ has width m; it combines two interfaces H and I, each with width m, by taking the union of names at each location. H represents names of a to be bound by R; I represents names of a to be exported by extra free links through R.

Figure 2 shows a parametric rule where R and R' both have two sites. So it takes a parameter $a = a_0 \,\|\, a_1$ of width 2, with factors a_0 and a_1 each of unit width. The the parallel composition $\|$ is derivable from the tensor product in s-categories; if a and b have widths m and n and disjoint supports, then in $a \,\|\, b$ –with width $m+n$– they are placed side-by-side, sharing free links. Sites in R and R' are numbered; an assignment $j := i$ written in the jth site of R' means that the reaction should place here a copy of the ith factor of a. Thus the rule shown will discard a_0 and duplicate a_1, putting one copy at each site of R'. (We omit details of how each copy's names are determined.)

We can think of the rule as the renew node fetching from the resource node (via the shared link z) a new copy of its resource a_1. Since R has two regions, the renew and resource nodes may be arbitrarily far apart in a large bigraph containing an occurrence of the redex R; so the rule offers the possibility of action at a distance.

Let us now define a certain λ-calculus, Λ_{sub}, in the usual way. It is a version
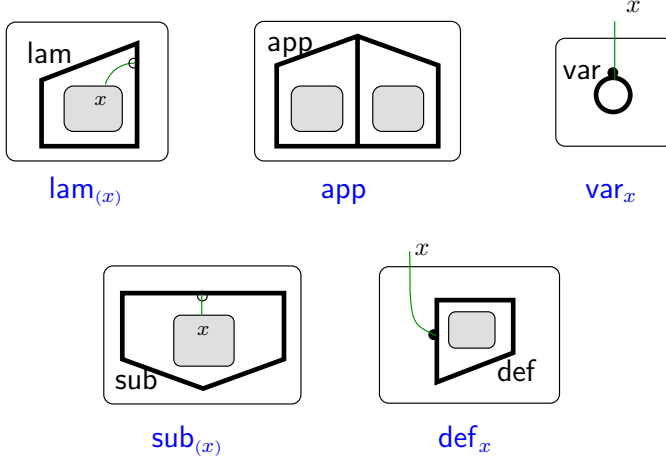
Fig. 3. Ions for the ^ΛBIG, with their algebraic representation

with explicit substitutions, but with coarser steps than that of Abadi et al [1]. The terms are

$$M \quad ::= \quad x \;\mid\; \lambda x M \;\mid\; MN \;\mid\; M\,[x{:=}N]$$

The final term construction should be read 'M where x means N'; it should not to be confused with $\{N/x\}M$, the result of *replacing* all free occurrences of x in M by N.

Definition 2.1 (reduction)   The reduction rules in Λ_{sub} are as follows:

$$(\lambda x M)N \;\longrightarrow\; M\,[x{:=}N]$$

$$(\{x/y\}M)\,[x{:=}N] \;\longrightarrow\; (\{N/y\}M)\,[x{:=}N] \quad \text{where } M \text{ has a unique}$$
$$\text{free occurrence of } y$$

$$M\,[x{:=}N] \;\longrightarrow\; M \quad \text{where } M \text{ has no free occurrence of } x \;.$$

Reductions may be applied to any subterm of a term.

Thus reductions are allowed even inside an explicit substitution. In the second rule, $\{x/y\}M$ distinguishes a particular free occurrence of x to be replaced by N. The three rules together achieve β-reduction. The explicit substitution $[x{:=}N]$ acts 'at a distance' on each free occurrence of x in turn, rather than migrating a copy of itself towards each such occurrence as in [1].

   We now turn to ^ΛBIG, the BRS corresponding to Λ_{sub}. Figure 3 shows its signature both graphically and algebraically. There are five controls (kinds of node), shown as *ions* (elementary bigraphs); a var-node has no sites, an app-node has two, and the rest have one. lam- and sub-nodes bind a link; var- and def-nodes have one port. The shapes of a node is unimportant, except that the shape of the app-

node signifies that its sites are in left-to-right order.[1] Note that binding names are parenthesized.

To export free names from their occupants, the ions with sites are generalised to

$$\mathsf{lam}_{(x)} \oplus \mathsf{id}_Z \quad \mathsf{app} \oplus (\mathsf{id}_Y \,|\, \mathsf{id}_Z)$$

$$\mathsf{sub}_{(x)} \oplus \mathsf{id}_Z \quad \mathsf{def}_x \oplus \mathsf{id}_Z \ .$$

The app-ion exports names Y from its first site, Z from its second.

Two new operators appear here. In this abstract we do not define operators formally, but illustrate their meaning by examples. A *prime* composition $F \,|\, G$ is like $F \,\|\, G$ (and derivable from it), but it merges the outer regions of F and G into one. The operator $\oplus$ is called *extension*. The extension $I \oplus I'$ of interfaces (with same width) was defined earlier. Given $G : I \to J$ and $\omega : I' \to J'$ (a wiring, i.e. a node-free bigraph) one can form $G \oplus \omega : I \oplus I' \to J \oplus J'$ provided the interface extensions are defined; it has the same tree structure as G, but the linkage of G is extended by adding the linkage of ω. Thus $\mathsf{id}_Y \,|\, \mathsf{id}_Z$, with inner width 2 and outer width 1, is a suitable extension for app; it exports the union of the inner name-sets Y and Z as outer names. The operators $\circ$, $\|$, $|$ and $\oplus$, though partial, have a rich algebraic theory.

The free names in a bigraph built from the above ions correspond exactly to the free variables in a λ-term. Thus $\lambda x\, x(xy)$ will translate into the bigraph

$$(\mathsf{lam}_{(x)} \oplus \mathsf{id}_y) \circ (\mathsf{app} \oplus (\mathsf{id}_x \,|\, \mathsf{id}_{xy})) \circ (\mathsf{var}_x \,\|\, ((\mathsf{app} \oplus (\mathsf{id}_x \,|\, \mathsf{id}_y)) \circ (\mathsf{var}_x \,\|\, \mathsf{var}_y))) \ .$$

(Here a set such as $\{xy\}$ has been written without curly brackets.) Of course, this notation is not recommended for developing λ-calculus theory! – but it has the advantage that the free names exported with each term constructor are made explicit.
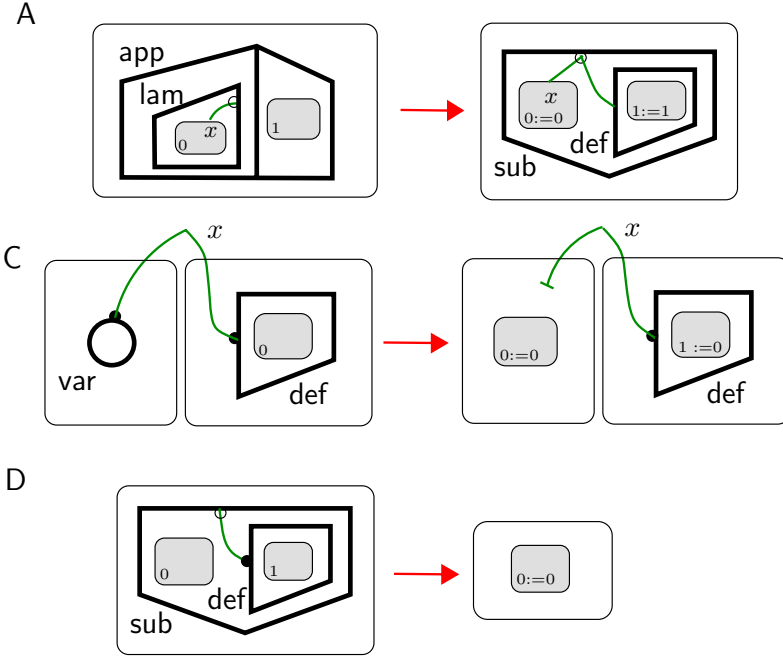
We now translate Λ_{sub} into $\hat{\Lambda}\mathrm{BIG}$. The translation function $[\![M]\!]_X$ is indexed by the set X, which must include all the free variables of M. Thus each term M has many bigraph images. This technique was used to model the asynchronous π-calculus [4].

Definition 2.2 (λ-terms into bigraphs)

$$[\![x]\!]_{X \uplus x} \stackrel{\mathrm{def}}{=} \mathsf{var}_x \oplus X$$

$$[\![\lambda x M]\!]_X \stackrel{\mathrm{def}}{=} (\mathsf{lam}_{(x)} \oplus \mathsf{id}_X) \circ [\![M]\!]_{X \uplus x}$$

$$[\![MN]\!]_X \stackrel{\mathrm{def}}{=} (\mathsf{app} \oplus (\mathsf{id}_X \,|\, \mathsf{id}_X)) \circ ([\![M]\!]_X \,\|\, [\![N]\!]_X)$$

$$[\![M[x{:=}N]]\!]_X \stackrel{\mathrm{def}}{=} (\mathsf{sub}_{(x)} \oplus \mathsf{id}_X) \circ ([\![M]\!]_{X \uplus x} \,|\, ((\mathsf{def}_x \oplus \mathsf{id}_X) \circ [\![N]\!]_X)) \ .$$

We shall not discuss this translation fully. But it is worth noting that alpha-convertible λ-terms have equal images; this is because bound names are elided by

---

[1] Multiple-site Controls are definable from single-site ones, site, with the help of a sorting discipline.

Fig. 4. Parametric reaction rules for $\hat{\Lambda}\textsc{big}$

composition. We are now ready to present the reaction rules for the BRS $\hat{\Lambda}\textsc{big}$.

Definition 2.3 (dynamics) $\hat{\Lambda}\textsc{big}$ has three reaction rules: A (apply), C (copy) and D (discard). They are shown both graphically and algebraically in Figure 4.

Note that rule C has width 2. Thus, in C, an occurrence of the 'variable' x may be distant from the defining equation that will replace it with a 'term'. This rule exploits the multiple locality of names in local bigraphs; it is similar to the rule of Figure 2.

We now assert that reaction in $\hat{\Lambda}\textsc{big}$ exactly matches reduction in Λ_{sub}:

Proposition 2.4 (reaction matches reduction) $[\![M]\!]_X \longrightarrow g$ *if and only if* $M \longrightarrow M'$ *for some M' such that $[\![M']\!]_X \simeq g$.*

In fact, for each reduction by a rule for Λ_{sub} there is a matching reaction by the corresponding rule for $\hat{\Lambda}\textsc{big}$, and conversely. In a recent draft O'Conchuir [9] has proved (strong) confluence directly for Λ_{sub}, so this translates immediately into a confluence proof for $\hat{\Lambda}\textsc{big}$. Our purpose here is different; we use the bigraphical
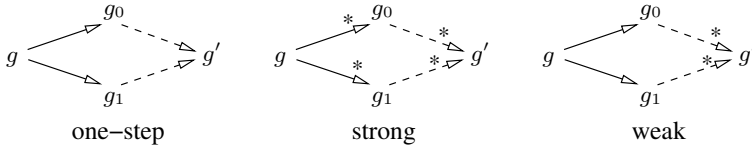
Fig. 5. Three notions of confluence

representation to illustrate the confluence properties that we seek for bigraphs in general.

3 Confluence in bigraphs

Recall that for any given reduction or reaction relation ' $\longrightarrow$ ' there are three familiar notions of *confluence*, shown in Figure 5. They all say that if g can react to become either g_0 or g_1, then these two reacta can in turn react to reach a common result. Clearly *one-step* $\Rightarrow$ *strong* $\Rightarrow$ *weak*, and it is well-known that these implications are strict in general. The most positive result for a BRS would be that strong confluence holds outright. This is indeed true (the Church-Rosser theorem) for the classical λ-calculus, and (by O'Conchuir) for Λ_{sub} also. However, in bigraphs we cannot expect this in general. Instead, we shall look for conditions that ensure non-interference between two competing reactions $g \longrightarrow g_0$ and $g \longrightarrow g_1$; such conditions may depend on the reaction rules that underlie the two translations, and on the extent to which the two redices overlap (if at all) in g. Moreover, it is in general easier to establish weak confluence in such cases.

   If we succeed in showing that weak confluence always holds for a certain class of agents under certain reaction rules, and if this class is itself preserved by reaction, then we may look to well-known methods from the theory of the λ-calculus that allow us to deduce strong from weak confluence. One such method is based upon *developments* [2]. A development is a reduction sequence $M \longrightarrow M_1 \longrightarrow M_2 \longrightarrow \cdots$ in which the only redices reduced are the residuals of an arbitrary set of redices present initially in M. The method is based upon the theorem that if all developments are of finite length, then weak confluence implies strong confluence.

   Before going further, we note that BRSs can be wilder than the λ-calculus! One property, used again and again in case analyses for the λ-calculus, is that when a term contains two redices then they are either disjoint or else nested (one inside the other). This fails for ground redices in BRSs; worse, it even fails for the parametric redices underlying them. Indeed, Figure 6 shows two possible parametric redices which are intimately entwined, each partly inside the other.
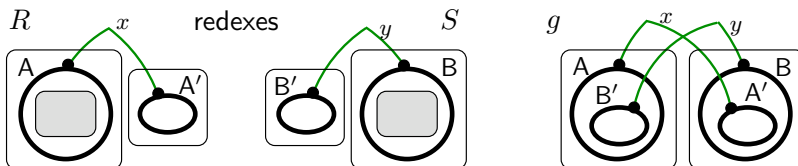

Fig. 6. An agent g containing two intertwined redices R and S

We do not know whether this property —redices nested or disjoint— is essential for weak confluence, or for finiteness of developments, in a BRS. However, recent investigation has explored a classification of ways in which two competing redices can overlap. If a parametric redex R supports a reaction $g \longrightarrow g_0$, then $r = (R \oplus \omega) \circ a$ occurs in g, for some parameter a and wiring ω. Similarly, if redex S supports a reaction $g \longrightarrow g_1$, then $s = (S \oplus \zeta) \circ b$ occurs in g. The *ground* redices r and s can overlap in different ways; for example s may not overlap with R, but may partly overlap with a. The investigation identifies four principal cases for such overlap, and claims that under certain further conditions the weak confluence diagram can be completed by $g_0 \longrightarrow^* g'$ and $g_0 \longrightarrow^* g'$. As this work is not complete we confine ourselves at present to two indeterminate conjectures about reactions in BRSs:

Conjecture 1 (weak confluence in $\hat{}\Lambda\mathrm{BIG}$) *Weak confluence holds for certain sets of agents in certain BRSs, including the set of all images of Λ_{sub}-terms in $\hat{}\Lambda\mathrm{BIG}$.*

Conjecture 2 (finite developments in $\hat{}\Lambda\mathrm{BIG}$) *Developments are finite for certain sets of agents in certain BRSs.*

Together, these two results will lead to strong confluence for the agents mentioned.

Conjecture 1 is reasonably firm, since (as indicated) much of the analysis has been done. Conjecture 2 is left vague at present. It is possible that, in $\hat{}\Lambda\mathrm{BIG}$, developments are finite only under some constraint. O'Conchuir's detailed study [9] may help to identify such a constraint.

Conclusion

The aim of this work is not to find yet another proof of the Church–Rosser theorem for a variant of the λ-calculus, but rather to learn from such proof techniques in order to analyse confluence for a wider class of agents and reaction rules than that for which it has hitherto been studied. This will lead to a better understanding not only of practically useful BRSs, but also of confluence itself.

References

[1] Abadi, M., Cardelli, L., Curien, P-L. and Levy, J-J. (1991), Explicit substitutions. Journal of Functional Programming 1, pp375–416.

[2] Barendregt, H. (1984), The Lambda Calculus: its Syntax and Semantics. North Holland.

[3] Bundgaard, M. and Hildebrandt, T. (2006), Bigraphical semantics of higher-order mobile embedded resources with local names. Proc. GT-VC 2005, Electronic Notes in Theoretical Computer Science 154(2), pp7–29.

[4] Jensen, O-H. and Milner, R. (2003), Bigraphs and transitions. Proc 30th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL), 2003, 16pp.

[5] Jensen, O.H. and Milner, R. (2004), Bigraphs and mobile processes (revised). Technical Report 580, University of Cambridge Computer Laboratory. Available from http://www.cl.cam.ac.uk/users/rm135.

[6] Leifer, J. and Milner, R. (2000), Bisimulation congruences for reactive systems. Proc. CONCUR2000, LNCS, Vol 1877, pp243–258.

[7] Leifer, J. and Milner, R. (2006), Link graphs, transitions and Petri nets. To appear in Mathematical Structures in Computer Science.

[8] Milner, R. (2004), Bigraphs whose names have multiple locality. Technical Report UCAM-CL-TR-603, University of Cambridge, Computer Laboratory. Available from http://www.cl.cam.ac.uk/TechReports/UCAM-CL-TR-603.pdf .

[9] O'Conchuir, S. (2006), Λ_{sub} as an explicit substitution calculus (draft).