On Decidability of Bigraphical Sorting

Giorgio Bacci[1]* Davide Grohmann[2]**

[1] Dept. of Mathematics and Computer Science, University of Udine
giorgio.bacci@uniud.it
[2] Programming, Logic and Semantics Group, IT University of Copenhagen
davg@itu.dk

Abstract. Bigraphs are a general framework for mobile, concurrent, and communicating systems. They have been shown to be suitable for representing several process calculi formalisms, but despite their expressive power, in many cases some disciplines on the structure of bigraphs are needed to faithfully encode the computational model at hand. *Sortings* have been proposed as an abstract technique to discipline bigraphs.
In this paper, we study the decidability problem of bigraphical sorting: to decide whether a bigraph belongs to some sorted bigraph category. Whilst the general problem is undecidable, we propose a decidable subclass of bigraphical sortings, named *match predicate sortings*, which are expressive enough to capture homomorphic sortings and local bigraphs.

1 Introduction

Bigraphical Reactive Systems (BRSs) [15] have been proposed as a promising meta-model for ubiquitous and mobile systems. The states of a BRS are *bigraphs.* Like an ordinary graph, a bigraph has nodes and edges connecting nodes, but unlike an ordinary graph, the nodes can be nested inside one another, hence they allow to represent both locality relationship between entities and (channel) connections. The dynamics of agents are represented by a set of rewrite rules on this semi-structured data.

Notably, Bigraphs and BRSs have been used for representing many domain-specific calculi and models: programming languages, calculi for concurrency and mobility, context-aware systems and web-services [11,12,3,9,5].

Process models (for example CCS, π-calculus, Ambient calculus) define processes syntactically, and in many cases it turns out that the encoding of processes into bigraphs is not exact, because bigraphs have too many degrees of freedom. This problem is usually overcome introducing "specialized versions" of bigraphs. For example, *binding bigraphs* [11] have been introduced to encode scoping for binding inputs in π-calculus: they allow for restricting name scope to a specific portion of a bigraph's locations. Many other variants have been proposed in literature, and almost all of them uses *sorting* techniques in achieving this. Example of sortings are *homomorphic* [15] and *many-one sortings* [12].

* Work funded by MIUR PRIN project "SisteR", prot. 20088HXMYN.
** Work partially funded by CosmoBiz project supported by ITU of Copenhagen and the Danish Research Council for Technology and Production. Grant no. 274-06-0415.

The main drawback in using sorting techniques is that each time a sorting is introduced, the theory of bigraphs must be redefined anew. Recently, Debois and coauthors [4] generalized the ad hoc constructions providing the definition of a class of *sorted categories* for which the behavioral theory of pure bigraphs is preserved, the so called *predicate sortings*. Intuitively predicate sortings rule out bigraphs that do not satisfy the predicate P.

Although using predicate sortings provides a general construction which sustain the behavioural theory of bigraphs, this technique has a main drawback: the systematic construction makes sorted categories very difficult to handle, due to the fact that their objects are defined as pairs of sets of morphisms from the original category, closed by prefix and suffix composition, and most of the difficulties arise when one wants to implement effectively such construction.

In order to overcome these difficulties, but at the same time keeping the technique as general as possible (we do not want to define sortings by hand), we propose to look at predicate sortings from a different point of view. The sorted category will not be constructed at all, but we will use sortings as a way of (automatically) checking if a morphism of the original category has a "counterpart" in the sorted category and, more importantly, if compositions in the non-sorted category are admissible in its sorted variant.

The aim of this paper is to investigate the decidability of the proposed checking procedure for sortings. It turns out that this procedure is undecidable in general, even if we restrict only on bigraphical sortings. For this reason we propose a decidable subclass of bigraphical predicate sortings, named *match predicate sortings*, for which there exists an effective algorithm to check if a bigraph "belongs" to the sorted category.

The key idea relies on the factorization theorem [4], which characterizes decomposable predicates as those that disallow factorisation by a given set of morphisms. Intuitively, given a predicate P such that $P(f)$ holds, there must exists a set Φ of morphism such that f cannot be decomposed in a form $f = g \circ \psi \circ h$ where ψ is in Φ. Intuiticely, P disallows occurrences of ill-formed morphisms, which are exactly those in Φ. In bigraphs it turns out to be much more intuitive to identify occurrences as matches. In this way, (some) decomposable predicates can be expressed as a set of ill-patterns which cannot be matched in well-sorted bigraphical morphisms. This choice permits to apply the bigraphical matching procedure to check whether a (pure) bigraph has a counterpart in the match sorted category, hence this class is decidable. Notably, this sub-class of sortings captures a good variety of sortings proposed in the literature, for example homomorphic sortings [15] and local bigraphs [14,16].

Synopsis The paper is structured as follows. In Sections 2 and 3 we recall the theory of bigraphs and (bigraphical) sortings. In Section 4 we prove the undecidability of the (bigraphical) sorting problem in the general setting. After that, in Section 5 the class of match predicate sortings is introduced and analyzed for decidability. In Section 6 we show how to express two relevant sortings introduced in literature as match predicate sortings. Finally, conclusions and ideas for further developments are in Section 7.

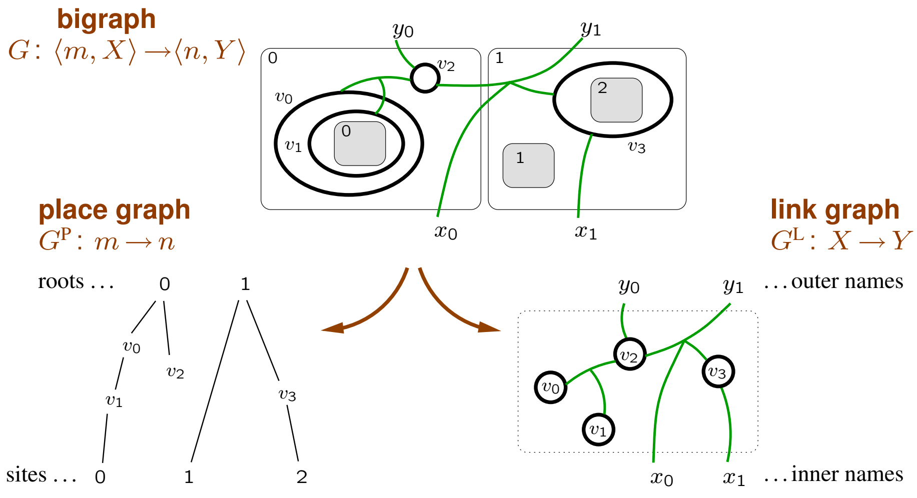


Fig. 1. Bigraph = Place graph + Link graph (picture taken from [15]).

2 Bigraphs

In this section we recall Milner's *bigraphs* [15]. Intuitively, a bigraph represents an open system: it has an inner and an outer interface to "interact" with subsystems and the surrounding environment (see Figure 1). The *width* of the outer interface describes the *roots*, that is, the various locations containing the system components; the width of the inner interface describes the *sites*, that is, the holes where other bigraphs can be inserted. On the other hand, the *names* in the interfaces describe the free links, that is, end points where links from the inner parameters or/and the external environment can be pasted, creating new links among nodes. We refer the reader to [15] for a longer description of bigraphs.

In this paper, we use the following terminology and notation. Natural numbers are frequently treated as finite ordinals, that is, $m = \{0, 1, \ldots, m-1\}$. We write $S \uplus T$ for the union of sets assumed to be disjoint. For two functions f and g with disjoint domains S and T we write $f \uplus g$ for the function with domain $S \uplus T$ such that $(f \uplus g) \restriction S = f$ and $(f \uplus g) \restriction T = g$. We write id_S for the identity function on the set S. In defining bigraphs we assume that names, node-identifiers and edge-identifiers are drawn from three infinite sets, respectively $\mathcal{X}$, $\mathcal{V}$ and $\mathcal{E}$, disjoint from each other.

The bigraphical category is defined over a *signature* $\mathcal{K}$ of controls with arity function $ar: \mathcal{K} \to \mathbb{N}$ which identifies the *ports* $ar(k)$ of a control $k \in \mathcal{K}$.

Definition 1 (Interface). *An* interface *is a pair* $\langle m, X\rangle$ *where* m *is a finite ordinal (named* width*) and* X *is a finite set of names.*

Definition 2 (Bigraphs). *A* bigraph $G: \langle m, X\rangle \to \langle n, Y\rangle$ *compounds a* place graph G^P *and a* link graph G^L *which describe the nesting of nodes and the* *(hyper-)links among nodes, respectively.*

$$\begin{aligned} G &= (V, E, ctrl, G^P, G^L)\colon \langle m, X\rangle \to \langle n, Y\rangle && \textit{(bigraph)}\\ G^P &= (V, ctrl, prnt)\colon m \to n && \textit{(place graph)}\\ G^L &= (V, E, ctrl, link)\colon X \to Y && \textit{(link graph)} \end{aligned}$$

where V, E are (finite) sets of nodes and edges respectively; $ctrl\colon V \to \mathcal{K}$ is the control map*, assigning a control to each node; $prnt\colon m \uplus V \to V \uplus n$ is the (acyclic)* parent map*; $link\colon X \uplus P \to E \uplus Y$ is the* link map, *where $P = \sum_{v\in V} ar(ctrl(v))$ is the set of ports.*

Definition 3 (Bigraph category). *The category of bigraphs over a signature $\mathcal{K}$, denoted as $\mathbf{Big}(\mathcal{K})$, has interfaces as objects and bigraphs as morphisms.*

Given two bigraphs $G\colon \langle m, X\rangle \to \langle n, Y\rangle$, $H\colon \langle n, Y\rangle \to \langle k, Z\rangle$, the composition $H \circ G\colon \langle m, X\rangle \to \langle k, Z\rangle$ is defined by composing their place and link graphs:

$$\begin{aligned} H^P \circ G^P &= (V, ctrl, (id_{V_G} \uplus prnt_H) \circ (prnt_G \uplus id_{V_H}))\colon m \to k\\ H^L \circ G^L &= (V, E, ctrl, (id_{E_G} \uplus link_H) \circ (link_G \uplus id_{P_H}))\colon X \to Z, \end{aligned}$$

where $V = V_G \uplus V_H$, $ctrl = ctrl_G \uplus ctrl_H$, and $E = E_G \uplus E_H$.

An important operation on bigraphs, is the *tensor product*. Intuitively, this operator puts "side by side" two bigraphs, i.e., given $G\colon \langle m, X\rangle \to \langle n, Y\rangle$ and $H\colon \langle m', X'\rangle \to \langle n', Y'\rangle$, their tensor product is $G \otimes H\colon \langle m + m', X \uplus X'\rangle \to \langle n + n', Y \uplus Y'\rangle$ defined when name sets X, X' and Y, Y' are pairwise disjoint.

As shown in [15], all bigraphs can be constructed by composition and tensor product from a set of *elementary bigraphs:*

- $1\colon \langle 0, \emptyset\rangle \to \langle 1, \emptyset\rangle$ is the barren (i.e., empty) root.
- $merge_n\colon \langle n, \emptyset\rangle \to \langle 1, \emptyset\rangle$ merges n roots into a single one.
- $\gamma_{m,n}\colon \langle m + n, \emptyset\rangle \to \langle n + m, \emptyset\rangle$ is a *symmetry*, that swaps the first m roots with the following n roots.
- $/x\colon \langle 0, \{x\}\rangle \to \langle 0, \emptyset\rangle$ is a *closure*, that is it maps x to an edge.
- $y/X\colon \langle 0, X\rangle \to \langle 0, \{y\}\rangle$ substitutes the names in X with y, i.e., it maps the whole set X to y. Notice that X can be the empty set, i this way y is linked to nothing and it is said to be *idle*.
- $K_{\vec{x}}\colon \langle 1, \emptyset\rangle \to \langle 1, \{x_1, \ldots, x_n\}\rangle$ is a control which may contain other bigraphs, and it has ports linked to the name in $\vec{x} = x_1, \ldots, x_n$.

A bigraph is said a *renaming* if it is of the form $x_1/\{y_1\} \otimes \cdots \otimes x_n/\{y_n\}$ (abbreviated to $\vec{x}/\vec{y}$, where $\vec{x} = x_1, \ldots, x_n$ and $\vec{y} = y_1, \ldots, y_n$); a *permutation* if it is formed by composition and tensor product of symmetries; a *prime* when it has no inner names and its outer width is 1, a *discrete* when its link map is a bijection. Two useful variants of tensor product can be defined using tensor and composition: the *parallel product*, denoted as $G \parallel H$, merges shared outer names of G and H, the *merge product* written $G \mid H$, which moreover merges all roots in a single one is defined as $merge_n \circ (G \parallel H)$ (with n outer width of $G \parallel H$).

3 Sortings

When one is adopting the bigraphical framework for defining algebraic models or programming languages, it turns out that such framework is too general and one has to discipline it with some constraints to fit precisely the problem at hand. To this end, general and powerful techniques, named *sortings*, have been developed by Debois and coauthors in [4].

Definition 4 (Sortings). *A* sorting *of a category* $\mathbf{C}$ *is a functor* $F : \mathbf{X} \to \mathbf{C}$*, that is faithful and surjective on objects. We call* $\mathbf{X}$ sorted category*.*

Intuitively, a sorting functor F defines $\mathbf{X}$ by refining the category $\mathbf{C}$. The objects (i.e., interfaces) of $\mathbf{X}$ carry more information than the original ones, thus morphism (i.e., system) composition turns out to be finer-grained. This yields back a category $\mathbf{X}$ where morphisms are more informative than those in $\mathbf{C}$ in the sense that, some compositions in $\mathbf{C}$ no longer hold in $\mathbf{X}$.

Due to the very general nature of sorting refinements needed by each particular application, it could be tricky to construct a sorting directly using the definition above. Moreover, each time a sorting is adopted the behavioral theory of bigraphs must be redeveloped. Debois observed that most sortings in the literature on bigraphs are actually means of banning particular morphisms from the original category. In each of these cases, it is possible to identify a predicate on the morphisms of the category in question, which holds precisely at the morphisms in the image of the sorting functor. Unfortunately a predicate on morphisms might not give rise to a subcategory, indeed we might have composable morphisms which individually satisfy the predicate, but whose composite does not. Debois proved that for the construction of such a sorted category it is sufficient that the predicate is decomposable.

Definition 5 (Decomposable predicate). *A* predicate P *on morphisms of a category is* decomposable *iff* P *holds on identities and* $P(f \circ g) \Rightarrow P(f) \wedge P(g)$*.*

Notably, the class of decomposable predicates can be characterized as those morphisms that disallow factorization by a given set of morphisms.

Theorem 1 (Factorization, [4, Proposition 14]). *A predicate* P *on morphisms of a category* $\mathbf{C}$ *is* decomposable *iff there exists a set of morphisms* Φ *such that* $P(f)$ *holds iff for any* g, ψ, h *we have* $f = g \circ \psi \circ h$ *implies* $\psi \notin \Phi$*.*

In [4] it is also given a method to systematically construct a well-behaved sorting for any decomposable predicate.

Definition 6 (Predicate Sorting). *Let* $\mathbf{C}$ *be a category and let* P *be a decomposable predicate on the morphisms of* $\mathbf{C}$*. The* predicate sorting $\mathcal{S}_P : \mathbf{X} \to \mathbf{C}$ *is defined as follows. The category* $\mathbf{X}$ *has pairs* (X, Y) *as objects, where, for some object* C *of* $\mathbf{C}$*,* X *is a set of* $\mathbf{C}$*-morphisms with codomain* C *and* Y *is a set of* $\mathbf{C}$*-morphisms with domain* C*, subject to the following conditions.*

$$\begin{array}{lcr} id_C \in X & f \in X \cup Y \Rightarrow P(f) & g \circ f \in X \Rightarrow g \in X \\ id_C \in Y & f \in X,\ g \in Y \Rightarrow P(g \circ f) & g \circ f \in Y \Rightarrow f \in Y\,. \end{array}$$

There is a morphism $f : (X,Y) \to (U,V)$ whenever the following holds.

$$f \in Y \cap U \qquad x \in X \Rightarrow f \circ x \in U \qquad v \in V \Rightarrow v \circ f \in Y\,.$$

Proposition 1. *Let P be a decomposable predicate on a category $\mathbf{C}$. The image of the predicate sorting $\mathcal{S}_P$ is precisely the set of morphisms satisfying P.*

All the above definitions and results apply naturally to the bigraph category.

4 Undecidability of bigraphical sortings

In this section, we focus our attention on the undecidability issues of predicate sortings and in particular for the case of bigraphical sortings.

Looking at the Definition 6, it is obvious that an exhaustive construction of a predicate sorted category is unfeasible (one must quantify on all morphisms to construct an object of the sorted category). Instead, what can be done is not to establish the decidability of the construction of the category, but looking at the problem of checking if a given morphism f from the base category has a pre-image in the sorted one. The key idea behind this approach is to “simulate” the existence of the sorted category, and checking that each morphism that comes into play is actually well-sorted.

When a predicate sorting $\mathcal{S}_P \colon \mathbf{X} \to \mathbf{C}$ is used, the existence of the pre-image $x = \mathcal{S}_P(f)$ in the sorted category $\mathbf{X}$ is garanteed whenever $P(f)$ holds by Proposition 1. Unfortunately, decidability cannot be assumed neither for general predicate P nor for decomposable predicates, even if we restrict to consider only decomposable predicates over bigraphical morphisms.

Let us define a decomposable predicate over bigraphs which will be proved to be undecidable by a reduction from the Post Correspondence Problem (PCP) [17]. We use α, β, γ for words in $\Sigma = \{a, b\}^*$, and ϵ for the empty word. An instance of PCP is a set of pairs of words $\{(\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n)\}$ over the two-letter alphabet $\{a, b\}$ (that is, $\alpha_i, \beta_i \in \Sigma$). The question is whether there exists a sequence $i_0, i_1, \dots, i_k$ ($1 \le i_j \le n$ for all $0 \le j \le k$) such that $\alpha_{i_0} \cdot \dots \cdot \alpha_{i_k} = \beta_{i_0} \cdot \dots \cdot \beta_{i_k}$, where $\cdot$ denotes word concatenation.

Let $\mathcal{K} = \{\mathsf{list}\colon 0, \mathsf{pair}\colon 0, \mathsf{a}\colon 0, \mathsf{b}\colon 0\}$ be a bigraphical signature of 0-arity controls. Any word $\gamma \in \Sigma$ can be represented in $\mathbf{Big}(\mathcal{K})$ by means of the encoding $\mathit{wrd}\colon \Sigma \to \mathbf{Big}(\mathcal{K})$, pairs of words by $\mathit{pair}\colon \Sigma \times \Sigma \to \mathbf{Big}(\mathcal{K})$ and n-length lists of word pairs by $\mathit{lst}_n\colon (\Sigma \times \Sigma)^n \to \mathbf{Big}(\mathcal{K})$ defined as follows:

$$\mathit{wrd}(\epsilon) = 1, \qquad \mathit{wrd}(a \cdot \gamma) = \mathsf{a} \circ \mathit{wrd}(\gamma), \qquad \mathit{wrd}(b \cdot \gamma) = \mathsf{b} \circ \mathit{wrd}(\gamma),$$
$$\mathit{pair}(\alpha, \beta) = \mathsf{pair} \circ (\mathit{wrd}(\alpha) \mid \mathit{wrd}(\beta)),$$
$$\mathit{lst}_n((\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n)) = \mathsf{list} \circ (\mathit{pair}(\alpha_1, \beta_1) \mid \cdots \mid \mathit{pair}(\alpha_n, \beta_n)).$$

Proposition 2. *Let Φ_{PCP} be the following a set of morphisms in $\mathbf{Big}(\mathcal{K})$:*

$$\Phi_{PCP} = \{\mathit{lst}_n((\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n)) \mid (\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n) \in PCP\}$$

then the set U below is a decomposable and undecidable predicate over $\mathbf{Big}(\mathcal{K})$.

$$U = \{f \text{ morphism of } \mathbf{Big}(\mathcal{K}) \mid \forall g, \phi, h.\ f = g \circ \phi \circ h \Rightarrow \phi \notin \Phi_{PCP}\}$$

Proof. By Theorem 1, U is a decomposable predicate. Let us prove its undecidability. By contradiction, assume U to be decidable, hence the characteristic function P_U for U, defined as $P(u) = 1$ if $u \in U$, $P(u) = 0$ otherwise, must be computable. This obviously contradicts the fact that PCP is undecidable, since an algorithm for P_U will decides also PCP because for $u = lst_n((\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n))$,

$$P_U(u) = 0 \iff u \notin U \iff ((\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)) \in \text{PCP}$$

Notice that u has an obvious occurrence of a morphism $\psi \in \Phi_{PCP}$, that is, $\psi = u$ since $u = id \circ u \circ id$. $\square$

Theorem 2. *Let $\mathcal{S}_P \colon \mathbf{X} \to \mathbf{C}$ be a predicate sorting over a decomposable predicate P. The problem of checking if a given morphism f in $\mathbf{C}$ has a pre-image $x = \mathcal{S}_P(f)$ in the sorted category $\mathbf{X}$ is undecidable.*

Proof. It follows immediately from Propositions 1 and 2. $\square$

As a corollary, checking the existence of a sorted pre-image is undecidable.

Corollary 1. *Let $\mathcal{S} \colon \mathbf{X} \to \mathbf{C}$ be sorting. The problem of checking if a given morphism f in $\mathbf{C}$ has a pre-image $x = \mathcal{S}(f)$ in $\mathbf{X}$ is undecidable.*

5 Match predicate sortings

In this section, we introduce a characterization of a decidable class of bigraphical sortings, which turns out to be a proper subclass of the predicate sortings.

Following the observations that drove the definition of predicate sortings of Debois and coauthors, that is, that most sortings in literature are actually defined for banning particular ill-formed patterns, we propose to identify the notion of pattern occurrence with that of pattern match occurrence. A bigraph G has a match within a bigraph H if and only if $H = F \circ (G \otimes id_X) \circ E$ for some name set X and bigraphs F, E. The problem of finding a match of a bigraph into another was investigated in [2] where and inductive characterization of this problem was proposed and a resolutive algorithm is given.

This scenario suggests the definition of a family of decomposable predicates based on the bigraphical matching problem.

Definition 7 (Match predicate). *Let $\mathcal{R}$ be a recursive set of bigraphs. We say that $P_{\mathcal{R}}$ is a* match predicate *with respect to the set $\mathcal{R}$, if for every bigraph G, $P_{\mathcal{R}}(G)$ holds iff every $R \in \mathcal{R}$ does not have a match in G.*

Proposition 3. *Any match predicate is a decomposable predicate.*

Proof. Let $\mathcal{R}$ be a set of redexes and P its match predicate. Now, suppose by absurdity that P is not decomposable. So there exist two bigraphs such that $P(G \circ H)$ holds and one between $P(G)$ or $P(H)$ does not. Suppose $P(H)$ does not hold (the other case is analogous), this means that there exist C, D such that $H = (id \otimes C) \circ (id \otimes R) \circ D$, for some $R \in \mathcal{R}$. Therefore $G \circ H = (G \circ (id \otimes C)) \circ (id \otimes R) \circ D$ is a match of R in $G \circ H$, an absurd by hypothesis. $\square$

Hence, the class of match predicates is a *proper subclass* of decomposable predicates, and it is decidable by means of the matching algorithm.

Proposition 4 (Decidability). *Any match predicate is decidable.*

Proof (Sketch). Let $G \colon \langle m, X\rangle \to \langle n, Y\rangle$ be a bigraph, we give a decidable decision procedure for $P_{\mathcal{R}}(G)$. Since $\mathcal{R}$ is a generic recursive set, it could be infinite in general, hence it is not be possible to check for all elements of $\mathcal{R}$ whether they have a match within the given bigraph G. Note, however, that G is finite, that is, it has finite node and edge sets, and finite interfaces as well. Let $\mathcal{K}_G$ be the set of controls used by nodes in G, and S the set of bigraphs using only controls taken from $\mathcal{K}_G$ and such that they have at most $|V_G|$ nodes, $|E_G|$ edges, $|X|$ inner names, $|X| + |P|$ outer names (where P is the set of ports, hence depends on the chosen set of nodes and controls), and m, n inner and outer width.

Since the set of nodes V_G is finite, $\mathcal{K}_G$ must be finite ($|\mathcal{K}_G| \leq |V_G|$); by a similar argument also S is finite. The set S contains all the possible bigraphs such that they have a match in G (the proof is by contradiction and omitted due to lack of space), by the finiteness of S and by the hypothesis that $\mathcal{R}$ is recursive, the set $\mathcal{R} \cap S$ is computable and finite. It is not difficult to prove that $P_{\mathcal{R}}(G)$ holds iff R does not have a match in G, for all $R \in \mathcal{R} \cap S$, hence, since the bigraphical matching problem is decidable [8], $P_{\mathcal{R}}$ is decidable as well. $\square$

Now we specialize the Factorization Theorem 1 in the following sense: given a recursive set of bigraphs M, the set Φ of unwanted bigraphs is defined from M as $\Phi = \{m \otimes id_X \mid m \in M \wedge X \text{ is a set of names}\}$.

Theorem 3 (Factorization). *A predicate P is a* match predicate *iff there exists a recursive set of morphisms M such that $P(f)$ holds iff for any g, ψ, h and any set of names X we have $f = g \circ (\psi \otimes id_X) \circ h$ implies $\psi \notin M$.*

Proof. Direct consequence of Proposition 3 and Theorem 1. $\square$

In this way, deciding if a bigraph G is well-sorted is reduced to decide if no $m \in M$ has a match into G. Moreover, we can use the Proposition 3 in combination with the Definition 6 to define the bigraphical sorted category. We call this class of sortings *match predicate sortings*. Consequence of the decidability of any match predicate is that the set M is *recursive*, indeed it is essential to not contradict Proposition 4. Although not forbidden, M is supposed to contain no identities, otherwise (almost) all bigraphs are sorted out, resulting in a useless sorting strategy. Finally, our match predicate sortings work up-to tensor product with identities, i.e., the unwanted bigraphical structures are "homset independent", indeed a match can be found in any context, so we cannot force the decomposition to work only with some particular interfaces.

6 Sortings in literature and their decidability

In order to investigate the expressive power of our decidable class of sortings, we analyze some sortings introduced in literature. We focus our attention on the sortings shown in [7, Table 6.1], which are replaceable by a predicate sorting. In particular, we consider *homomorphic sortings* [15] and the bigraph's variant known as *local bigraphs* [14,16]. In this section we show that each decomposable predicate used in [7] can be characterized as a match predicate of Definition 7. Notice that the construction of the sorted category is left unchanged because match predicates are decomposable by Proposition 3.

6.1 Homomorphic sortings and CCS

Firstly, we recall homomorphic sortings as given in [15] with the variants of [7][3]. In order to exemplify the use of the match predicate sortings, we also recall the encoding of CCS [13] into bigraphs proposed by Milner in [15] which adopts a particular homorphic sorting.

We start giving the definition of place-sorted bigraph.

Definition 8 (Place-sorted interface). *Let Θ be a set of sorts. An interface $I = \langle m, X\rangle$ is Θ-*place-sorted *if it is enriched by ascribing a sort to each place $i \in m$. If I is place-sorted, we denote its underlying unsorted interface by $U(I)$.*

We denote by $\mathbf{Big}(\mathcal{K}, \Theta)$ the category in which the objects are place-sorted interfaces, and each morphism $G : I \to J$ is a bigraph $G : U(I) \to U(J)$.

Such definition refines only the objects of the bigraph category, the next one is cutting down some morphisms (i.e., bigraphs).

Definition 9 (Place-sorting). *A* place-sorting *is a triple $\Sigma = \{\mathcal{K}, \Theta, \Phi\}$, where Φ is a condition on the place graph of Θ-sorted bigraphs over $\mathcal{K}$. The condition Φ must be satisfied by identities and preserved by composition and tensor.*

A bigraph in $\mathbf{Big}(\mathcal{K}, \Theta)$ is Σ-place-sorted if it satisfies Φ. The Σ-sorted bigraphs form a sub-category of $\mathbf{Big}(\mathcal{K}, \Theta)$ denoted by $\mathbf{Big}(\Sigma)$.

Notably, Milner in [15, Proposition 10.3] shows that U can be extended to a functor $U : \mathbf{Big}(\Sigma) \to \mathbf{Big}(\mathcal{K}, \Theta)$ which is surjective on objects and faithful, and hence a sorting by Definition 4.

Due to the very general nature of place-sorting, Milner defines a particular class of such sortings, named *homomorphic sortings.*

Definition 10 (Homomorphic sorting). *A place-sorting $\Sigma = \{\mathcal{K}, \Theta, \Phi\}$ is an* homomorphic sorting *if the condition Φ assigns a sort $\theta \in \Theta$ to each control in $\mathcal{K}$ by means of a surjective function $sort : \mathcal{K} \to \Theta$ and it also defines a parent map $prnt_\Theta : \Theta \to \Theta$ over sorts. (We impose that Θ has a least two elements[4].)*

In a bigraph G, via its control map, the sort assignment to $\mathcal{K}$ determines a sort for each node. The Φ requires that, for each site or node w in G with sort θ:

[3] The variants are quite technical and do not change the resulting sorted categories.
[4] Otherwise the homomorphic sorting sorts out no bigraph, hence it is useless.

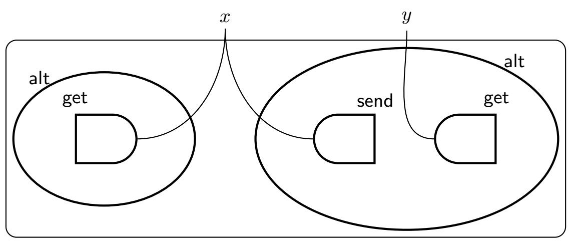


Fig. 2. A bigraph encoding the CSS process $x.\mathbf{0} \mid (\bar{x}.\mathbf{0} + y.\mathbf{0})$.

1. *if $prnt_G(w)$ is a node then its sort is $prnt_\Theta(\theta)$;*
2. *if $prnt_G(w)$ is a root then its sort is θ.*

As already mentioned, the translation of (finite) CCS in bigraphs provides a interesting non-trivial example of homomorphic sorting. Suppose a universal set of name $\mathcal{N}$ and let $x, y, z, \ldots$ range over $\mathcal{N}$, whilst P, Q range over processes and A over summations. The CCS syntax is

$$\begin{aligned} P &::= (\nu x)P \mid P \mid P \mid \;\; A \\ A &::= \quad \mathbf{0} \quad \mid \; x.P \;\mid \bar{x}.P \mid A + A\,. \end{aligned}$$

Intuitively $\mathbf{0}$ denotes termination. $(\nu x)P$ means that the name x is restricted in P. $x.P$ and $\bar{x}.P$ are the input and output actions respectively. Finally, $\mid$ is the parallel composition and $+$ the non-deterministic choice. The set of free names is composed by all names of the process not under the scope of a ν. In [15] processes are taken up-to the following structural equivalence ($\equiv$): it contains the α-equivalence on processes, $\mid$ and $+$ are commutative and associative under $\equiv$, and the following rules hold

$$A + 0 \equiv 0 \qquad (\nu x)(A + \alpha.P) \equiv A + \alpha.(\nu x)P \quad \text{if } \alpha \in \{y, \bar{y}\} \text{ and } x \neq y$$
$$(\nu x)(\nu y)P \equiv (\nu y)(\nu x)P \qquad\qquad (\nu x)P \equiv P \quad \text{if } x \notin fn(P)$$
$$(\nu x)(P \mid Q) \equiv P \mid (\nu x)Q \quad \text{if } x \notin fn(P)\,.$$

Notice that $P \mid \mathbf{0} \not\equiv P$, but we can prove that they are bisimilar.

Now we introduce the homorphic sorting for encoding CCS into bigraphs.

$$\Sigma_{CCS} = (\{\mathsf{a}, \mathsf{p}\}, \{\mathsf{alt} : 0, \mathsf{get} : 1, \mathsf{send} : 1\}, \Phi)$$

where a, p are types representing "summations" and "processes". The control alt encodes a summation and the controls get and send denote input and output actions. The condition Φ assigns the type a to alt and p to both get and send. Φ also imposes an alternation of controls of type a and p in the place graphs, i.e., $prnt_\Theta(\mathsf{a}) = \mathsf{p}$ and $prnt_\Theta(\mathsf{p}) = \mathsf{a}$. Both composition and tensor preserves Φ.

The translation of a CCS process into a bigraphs is defined as follows. We map processes into ground homset having a single root typed with p, i.e., $\epsilon \to \langle \mathsf{p}, X \rangle$, and analogously summations into $\epsilon \to \langle \mathsf{a}, X \rangle$. In order to do this we define two

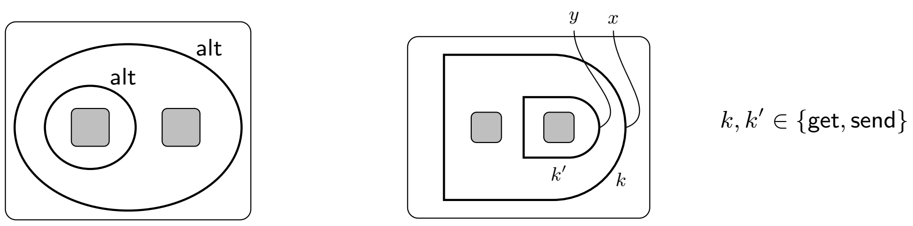


Fig. 3. The ill-bigraphs used to define a match predicate sorting for the CCS.

translation operators $\mathcal{P}_X[\cdot]$ and $\mathcal{A}_X[\cdot]$ each indexed on a (finite) set of names X. They are defined by mutual recursion:

$$\begin{array}{ll}
\mathcal{P}_X[(\nu x)P] = /y \circ \mathcal{P}_{X \uplus \{y\}}[P\{y/x\}] & \mathcal{A}_X[\mathbf{0}] = X \mid 1 \\
\mathcal{P}_X[P \mid Q] = \mathcal{P}_X[P] \mid \mathcal{P}_X[Q] & \mathcal{A}_X[a.P] = (\mathsf{get}_x \mid \mathsf{id}_X) \circ \mathcal{P}_X[P]\ (x \in X) \\
\mathcal{P}_X[A] = (\mathsf{alt} \mid \mathsf{id}_X) \circ \mathcal{A}_X[A] & \mathcal{A}_X[\bar{a}.P] = (\mathsf{send}_x \mid \mathsf{id}_X) \circ \mathcal{P}_X[P]\ (x \in X) \\
 & \mathcal{A}_X[A + B] = \mathcal{A}_X[A] \mid \mathcal{A}_X[B]\,.
\end{array}$$

As an example consider the CSS process $x.\mathbf{0} \mid (\bar{x}.\mathbf{0} + y.\mathbf{0})$, its translation in a bigraph is depicted in Figure 2.

In order to construct a predicate from a homomorphic sorting, it is sufficient to restrict the condition of Φ to consider just 1. and dropping 2., indeed we can focus only on constraining the internal components (i.e., nodes) of the bigraph. The roots belong to the interfaces, and those are refined automatically by the predicate sortings (see Definition 6).

The predicate constructed by Debois in [7] is the following.

Definition 11. *Let $\Sigma = \{\mathcal{K}, \Theta, \Phi\}$ be a homomorphic sorting, and let $prnt_\Theta$ be the parent maps on sorts defined by Φ. The predicate P_Σ holds on a bigraph G iff whenever the control of a node v in G has sort $\theta \in \Theta$ and $w = prnt_G(v)$ is a node, then the control of w has sort $prnt_\Theta(\theta)$.*

On such definition, it is easy to yield a "negative" version of the predicate by means of the Factorization Theorem 1: the set of unwanted bigraphs Φ can be constructed by complementing the condition 1.. This observation also suggests a way of deriving a match predicate by means of our Factorization Theorem 3.

Definition 12. *Let $\Sigma = \{\mathcal{K}, \Theta, \Phi\}$ be a homomorphic sorting, and let $prnt_\Theta$ be the parent maps on sorts defined by Φ. The match predicate $P(M_\Sigma)$ for Σ can be defined on the set of bigraphs M_Σ below and by using the Theorem 3.*

$$M_\Sigma \triangleq \{(K_{\vec{x}} \otimes \mathsf{id}_{\vec{y}}) \circ (H_{\vec{y}} \otimes \mathsf{id}_1) \mid K, H \in \mathcal{K} \wedge prnt_\Theta(sort(H)) \neq sort(K)\} \quad (1)$$

It is trivial to prove that the meaning of the two predicates coincides, indeed if a match exists condition 1. is violated, otherwise it does not hold. It is important for decidability to notice that the set M_Σ defined in equation (1) is finite.

In the case of CCS, the set of undesired graph $M_{\Sigma_{CCS}}$ must contain the bigraphs which has ill-nested controls. In particular, we should forbid the nesting of a a-typed control into another a-typed one (analogously for the type p). In other words this means that we do not allow the nesting of two consecutive alt nodes or two send and/or get nodes. Formally, the set of ill-formed bigraphs for the match predicate sorting are defined below and depicted in Figure 3.

$$M_{\Sigma_{CCS}} = \left\{ \begin{array}{l} \mathsf{alt} \circ (\mathsf{alt} \otimes \mathsf{id}_1), \\ (\mathsf{get}_x \otimes \mathsf{id}_{\{y\}}) \circ (\mathsf{get}_y \otimes \mathsf{id}_1), (\mathsf{get}_x \otimes \mathsf{id}_{\{y\}}) \circ (\mathsf{send}_y \otimes \mathsf{id}_1), \\ (\mathsf{send}_x \otimes \mathsf{id}_{\{y\}}) \circ (\mathsf{get}_y \otimes \mathsf{id}_1), (\mathsf{send}_x \otimes \mathsf{id}_{\{y\}}) \circ (\mathsf{send}_y \otimes \mathsf{id}_1) \end{array} \right\} .$$

The following result follows directly from the above considerations.

Theorem 4. *Homomorphic sortings correspond exactly to match predicate sortings over the predicate M_Σ.*

Proof. It follows from the characterization given in [7, Section 6.3] and by the fact that $P_\Sigma = M_\Sigma$. □

Remarkably, homomorphic sortings are of particular interest in the setting of bigraphical encoding of process algebras, in fact, they have been employed in the encoding of π-calculus variants [18] (cfr. Jensen [10]), but also in the definition of variants pure bigraphs, e.g. kind bigraphs [6].

Corollary 2. *Homomorphic sortings are decidable.*

Proof. Direct consequence of Theorem 4 and Proposition 4. □

6.2 Local bigraphs

In this section first we recall Milner's *local bigraphs* [14,16] and then we discuss how to use a match predicate sorting on (pure) bigraphs to catch local bigraphs.

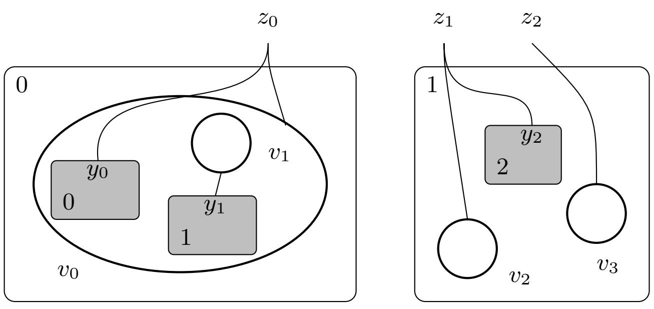
Intuitively, a local bigraph is like a standard (pure) bigraph but it has names which are deeply connected with placing, i.e., there is a precise scoping rule: linking must respect the nesting of nodes. An example of a local bigraph is shown in Figure 4, Note that inner names are "localized" on the bigraph's sites and outer names on the roots.

Let $\mathcal{K}$ be a *binding signature* of controls and $ar : \mathcal{K} \to \mathbb{N} \times \mathbb{N}$ be the arity function. The arity pair (h, k) consists of the *binding arity* h and the *free arity* k, indexing respectively the binding ports and the free ports of a control.

Definition 13. *A* local interface *is a list $(X_0, \ldots, X_{n-1})$, where n is the* width *and X_is are disjoint sets of names. X_i represents the names located at i.*

Definition 14. *A* local bigraph *$G : (\vec{X}) \to (\vec{Y})$ is defined as a (pure) bigraph $G^u : \langle |\vec{X}|, \bigcup \vec{X} \rangle \to \langle |\vec{Y}|, \bigcup \vec{Y} \rangle$ satisfying certain locality conditions. Let π_1 and π_2 be the canonical projections of pair components, let $P = \sum_{v \in V} \pi_1(ar(ctrl(v)))$ be the set of ports, and let $B = \sum_{v \in V} \pi_2(ar(ctrl(v)))$ be the set of bindings (associated to all nodes), the* link map *is $link : X \uplus P \to E \uplus B \uplus Y$.*

The locality conditions are the following:



$G : (\{y_0\}, \{y_1\}, \{y_2\}) \to (\{z_0\}, \{z_1, z_2\})$

Fig. 4. An example of a local bigraph.

1. *if a link is bound, then its inner names and ports must lie within the node that binds it;*
2. *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.*

Definition 15. *The category $\mathbf{Lbg}(\mathcal{K})$ of local bigraphs over a binding signature $\mathcal{K}$ has local interfaces as objects, and local bigraphs as morphisms. Composition and tensor product are defined analogously as for (pure) bigraphs.*

Local bigraphs have been introduced in literature by Milner for encoding the λ-calculus [1] into bigraphs and investigating confluence properties for bigraphical reactive systems [16].

In [7] it is given the predicate that follows, and it is proven that predicate sortings can be replaced with the category of local bigraphs.

Definition 16 ([7, Definition 6.22]). *Let Σ be a binding signature. Define P_Σ to be the predicate on the morphisms of $\mathbf{Big}(U(\Sigma))$ given by $P_\Sigma(f)$ iff in f*

"all ports linked to a binding port of a node v lie under v".

It is straightforward to prove that such predicate is decomposable, but we want to characterize it as a matching predicate, that is, provide a recursive set of unwanted redex patterns. Fortunately the predicate P_Σ is very simple to falsify, indeed it is enough to find a match of a redex with the form in (2) below (see also Figure 5). We denote a node as $K_{(\vec{x})\vec{y}}$, which means that the node has control K, its free ports are linked to the outer names in $\vec{y}$, and the inner names $\vec{x}$ are linked to its binding ports.

$$(K_{(\vec{x}w\vec{y})\vec{z}} \parallel \mathsf{id}_{\langle 1,\vec{b}\vec{c}\rangle}) \circ (\mathsf{id}_{\langle 1,\vec{x}w\vec{y}\rangle} \parallel N_{(\vec{a})\vec{b}w\vec{c}}) \qquad (2)$$

for all controls $K, N \in \Sigma$. Intuitively, if a bigraph has a match for a redex with the form in (2), that match is a counterexample for P_Σ (the binding port targeted by w of the K-node has as peer the port linked to w of the another node N which is not beneath the K-node).

Now we can define the binding match predicate.

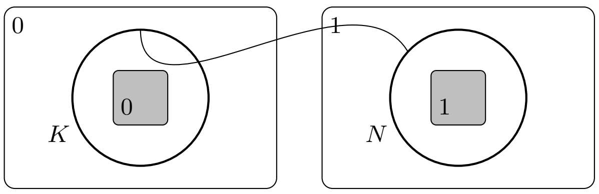


Fig. 5. A (simplified) example of ill-formed bigraph.

Definition 17. *Let Σ be a binding signature and let $\mathcal{R}_{bind}$ be the following set of bigraphs:*

$$\mathcal{R}_{bind} = \{(K_{(\vec{x}w\vec{y})\vec{z}} \parallel \mathsf{id}_{\langle 1,\vec{b}\vec{c}\rangle}) \circ (\mathsf{id}_{\langle 1,\vec{x}w\vec{y}\rangle} \parallel N_{(\vec{a})\vec{b}w\vec{c}}) \mid K, M \in \Sigma\}$$

Then define M_Σ as the match predicate defined on the (recursive) set $\mathcal{R}_{bind}$.

Theorem 5. *The category of local bigraphs corresponds to the one obtained by applying the match predicate sorting on M_Σ over the morphisms of* **Big**.

Proof. It follows immediately from the characterization given in [7, Section 6.4] and by the fact that $P_\Sigma = M_\Sigma$. $P_\Sigma \subseteq M_\Sigma$ can be proved noticing that any match of a redex from R_Σ in a bigraph f of $\mathbf{Big}(U(\Sigma))$ is a counterexample for $P_\Sigma(f)$; whereas $M_\Sigma \subseteq P_\Sigma$ follows immediately from the fact that R_Σ has a redex for any pair of controls in Σ and for any binding port. $\square$

7 Conclusions

In this paper, we have investigated the decidability problem of (bigraphical) sortings. In particular, we have shown the undecidability of Debois and coauthors' predicate sortings and then we have identified a proper sub-class of them, named *match predicate sortings*, which turned out to be decidable. For the match predicate sortings, we have proposed a characterization that induces the definition of a decision procedure to check if a given morphism in the unsorted category has a pre-image into the sorted one, which holds independently from the chosen predicate. The procedure is based on the bigraphical matching problem, for which a decision algorithm was proposed in [2].

Notably, our match predicate sortings preserve many interesting properties of predicate sortings, such as the possibility of describing unwanted bigraphs by means of BiLog formulae. Moreover, we have shown that the match predicate sortings are powerful enough to capture two important bigraphical sortings proposed in literature: *homomorphic sorting* and *local bigraphs.*

As possible future developments, we plan to investigate if other decidable classes of sortings exist and if there are other (possibly) more efficient algorithms to decide if a bigraph belongs to a sortings (remark that the matching problem for bigraphs is NP-complete). Finally, another interesting future work is the analysis of the problem in a more general setting, not focusing only on bigraphs.

References

1. H. Barendregt. *The lambda calculus: its syntax and its semantics.* Studies in Logic and the Foundations of Mathematics. North-Holland, 1984.
2. L. Birkedal, T. C. Damgaard, A. J. Glenstrup, and R. Milner. Matching of bigraphs. *Electr. Notes Theor. Comput. Sci.*, 175(4):3–19, 2007.
3. L. Birkedal, S. Debois, E. Elsborg, T. Hildebrandt, and H. Niss. Bigraphical models of context-aware systems. In L. Aceto and A. Ingólfsdóttir, editors, *Proc. FoSSaCS*, volume 3921 of *Lecture Notes in Computer Science*, pages 187–201. Springer, 2006.
4. L. Birkedal, S. Debois, and T. T. Hildebrandt. Sortings for reactive systems. In C. Baier and H. Hermanns, editors, *CONCUR*, volume 4137 of *Lecture Notes in Computer Science*, pages 248–262. Springer, 2006.
5. M. Bundgaard, A. J. Glenstrup, T. T. Hildebrandt, E. Højsgaard, and H. Niss. Formalizing higher-order mobile embedded business processes with binding bigraphs. In D. Lea and G. Zavattaro, editors, *COORDINATION*, volume 5052 of *Lecture Notes in Computer Science*, pages 83–99. Springer, 2008.
6. S. Ó. Conchúir. Kind bigraphs. *Electr. Notes Theor. Comput. Sci.*, 225:361–377, 2009.
7. S. Debois. *Sortings and Bigraphs.* PhD thesis, IT University of Copenhagen, 2008. `http://www.itu.dk/people/debois/pubs/thesis.pdf`.
8. A. Glenstrup, T. Damgaard, L. Birkedal, and E. Højsgaard. An implementation of bigraph matching. *IT University of Copenhagen*, 2007. `http://www.itu.dk/~tcd/docs/implBigraphMatching.pdf`.
9. D. Grohmann and M. Miculan. Reactive systems over directed bigraphs. In L. Caires and V. T. Vasconcelos, editors, *Proc. CONCUR 2007*, volume 4703 of *Lecture Notes in Computer Science*, pages 380–394. Springer, 2007.
10. O. H. Jensen. *Mobile Processes in Bigraphs.* PhD thesis, University of Aalborg, 2008. To appear.
11. O. H. Jensen and R. Milner. Bigraphs and transitions. In *Proc. POPL*, pages 38–49, 2003.
12. J. J. Leifer and R. Milner. Transition systems, link graphs and petri nets. *Mathematical Structures in Computer Science*, 16(6):989–1047, 2006.
13. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer, 1980.
14. R. Milner. Bigraphs whose names have multiple locality. Technical Report 603, University of Cambridge, CL, Sept. 2004.
15. R. Milner. Pure bigraphs: Structure and dynamics. *Information and Computation*, 204(1):60–122, 2006.
16. R. Milner. Local bigraphs and confluence: Two conjectures. In *Proc. EXPRESS 2006*, volume 175(3) of *Electronic Notes in Theoretical Computer Science*, pages 65–73. Elsevier, 2007.
17. E. L. Post. Recursively enumerable sets of positive integers and their decision problems. *Bulletin of the American Mathematical Society*, 50:284–316, 1944.
18. D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes.* Cambridge University Press, 2001.