Math. Struct. in Comp. Science (2006), *vol.* 16, *pp.* 1133–1163. © 2006 Cambridge University Press
doi:10.1017/S096012950600569X Printed in the United Kingdom

Deriving bisimulation congruences in the DPO approach to graph rewriting with borrowed contexts[†]

HARTMUT EHRIG[‡] and BARBARA KÖNIG[§]

[‡]*Institut für Softwaretechnik und Theoretische Informatik*
Technische Universität Berlin, Germany
Email: `ehrig@cs.tu-berlin.de`

[§]*Institut für Informatik und interaktive Systeme,*
Universität Duisburg-Essen, 47048 Duisburg, Germany
Email: `barbara_koenig@uni-due.de`

Received 12 October 2004; revised 30 May 2006

Motivated by recent work on the derivation of labelled transitions and bisimulation congruences from unlabelled reaction rules, we show how to address this problem in the DPO (double-pushout) approach to graph rewriting. Unlike the case with previous approaches, we consider graphs as objects, rather than arrows, of the category under consideration. This allows us to present a very simple way of deriving labelled transitions (called rewriting steps with borrowed context), which integrates smoothly with the DPO approach, has a very constructive nature and requires only a minimum of category theory. The core part of this paper is the proof that the bisimilarity based on graph rewriting with borrowed contexts is a congruence relation. We will also introduce some proof techniques and compare our approach with the derivation of labelled transitions via relative pushouts.

1. Introduction

In recent years the problem of deriving labelled transitions and bisimulation congruences from unlabelled reaction or rewriting rules has received great attention. This line of research was motivated by the theory of bisimulation congruences for process calculi, such as the π-calculus (Sangiorgi and Walker 2001; Milner 1999). A bisimilarity defined on unlabelled reduction rules is usually not a congruence, that is, it is not closed under the operators of the process calculus. Congruence is a very desirable property since it allows one to replace a subsystem with an equivalent one without changing the behaviour of the overall system and, futhermore, helps make bisimilarity proofs modular.

Previous solutions have been to either require that two processes are related if and only if they are bisimilar under all possible contexts (Milner and Sangiorgi 1992a) or to derive a labelled transition system manually. Since the first solution needs quantification over all possible contexts, proofs of bisimilarity can be very complicated. In the second solution, proofs tend to be much easier, but it is necessary to show that the labelled variant of the transition system is equivalent to the unlabelled variant.

[†] This research was partially supported by the DFG project SANDS, the BMBF project DACHIA, the TMR network SEGRAVIS and EPSRC grant R93346/01.

So the idea formulated in Leifer (2001), Leifer and Milner (2000), Sewell (2002) and Sassone and Sobociński (2003) is to derive automatically a labelled transition system such that the resulting bisimilarity is a congruence. A central concept of this approach is to formalise the notion of the minimal context that enables a process to reduce. Consider, for example, the CCS process $a.P$. It reduces when put into the contexts $\_ \mid \bar{a}.Q$ and $\_ \mid \bar{a}.Q \mid b.R$, but one is interested only in the first context, since it is in some sense smaller than the second. This yields the labelled transition

$$a.P \xrightarrow{\_\mid\bar{a}.Q} P \mid Q,$$

saying that $a.P$ put into this context reacts and reduces to $P \mid Q$. Using all possible contexts as labels would also result in a (coarser) bisimulation congruence, but we do not gain anything compared to quantification over all contexts (for a more detailed study of this congruence, see Bonchi *et al.* (2006)).

The notion of 'minimal context' is formalised in Leifer (2001) and Leifer and Milner (2000) as the categorical concept of a relative pushout (RPO) and an idem pushout (IPO), respectively. This notion has also been applied to bigraphs (Jensen and Milner 2003). However, the theory is complicated by the fact that one cannot work with isomorphism classes of graphs, since in this case the category under consideration would not possess all necessary relative pushouts. Thus, one is forced to give unique names to all edges and nodes in a graph, that is, to add support to a category, and to either work in a precategory or to construct a suitable category starting from such a precategory. A different approach, presented by Sassone and Sobociński (Sassone and Sobociński 2003), which does not require the notion of support, is to construct relative pushouts (so-called GRPOs) in a 2-categorical setting.

It is our aim to achieve similar results in the context of graph rewriting (Rozenberg 1997), which is a framework that allows one to model dynamic and concurrent systems consisting of interconnected components in a natural and intuitive way. Many process calculi, such as the π-calculus (Gadducci and Montanari 2002; Montanari and Pistore 1995; König 2000) and the ambient calculus (Gadducci and Montanari 2001), can be translated into this framework. We are specifically interested in the double-pushout (DPO) approach (Corradini *et al.* 1997; Ehrig *et al.* 1973), which is one of the standard approaches to graph rewriting. However, there is not yet a uniform theory of bisimulation for graph transformation systems. Previous work on this topic appeared in Baldan *et al.* (2002) and König and Montanari (2001). Adding support, as explained earlier, would be possible in theory, but contradicts the philosophy behind graph rewriting in which graphs are considered only up to isomorphism. Furthermore, we are aiming for a more straighforward approach for deriving labels of a transition system. Compared to earlier approaches, in which the derivation of labels is a somewhat complex task, our approach is rather straightforward and simple.

The approach presented in this paper is motivated by the work of Leifer and Milner, together with other contributions to this area, but does not directly rely on their theory. Instead, we present an uncomplicated way of deriving minimal contexts, which we call borrowed contexts, that smoothly extends the DPO approach and has a very constructive nature. The only categorical concepts that are needed are pushouts and pullbacks. The main difference to previous approaches is that in our case graphs are objects and not arrows of the category under consideration. Our arrows are, instead, graph morphisms, which provide the necessary tracking information for nodes and edges, and which, it turns out, can be provided in the case of graphs as arrows by either adding support to a category or by working in a 2-categorical framework. This work is based on ideas presented in Ehrig (2002), which also points out similarities and differences between Milner's bigraphs (Jensen and Milner 2003; Milner 2001) and the DPO approach to graph rewriting. Note, however, that in order to obtain the congruence property, our notion of rewriting with borrowed contexts is slightly different to the one proposed in Ehrig (2002).

Our main result states that bisimilarity defined on graph rewriting with borrowed contexts is indeed a congruence relation (see Theorem 4.3).

The paper is structured as follows. In Section 2 we will give a short introduction to the DPO approach. This is followed in Section 3 by the definition of rewriting with borrowed contexts. Section 4 provides the proof that the resulting bisimilarity is a congruence. After introducing two proof techniques in Section 5, Section 6 continues with an example showing borrowed contexts at work. We conclude in Section 7 with a comparison between our approach to the relative pushouts (RPOs) of Leifer and Milner and the GRPOs of Sassone and Sobociński. In fact, it turns out that our method of deriving labels corresponds exactly to the construction of GRPOs in a cospan category. This fact is presented in more detail in Sassone and Sobociński (2004).

This paper requires no more than a basic knowledge of category theory (Pierce 1991; Mac Lane 1971). In fact, we only need pushouts and pullbacks, including some general as well as specific preservation, composition and decomposition properties. The general properties hold in any category and the specific ones hold for graph structures, but also, more generally, for the adhesive categories introduced recently in Lack and Sobociński (2004) and Lack and Sobociński (2005) (see Appendix A). Moreover, they have been generalised and shown for adhesive high-level replacement categories in Ehrig *et al.* (2004), where monomorphisms are generalised to a suitable subclass M of all morphisms.

2. The DPO approach to graph rewriting

We will first define a family of categories of graphs and graph morphisms in a very general way by defining graph structures (Ehrig *et al.* 1997), which include different forms of graphs such as directed graphs and hypergraphs.

Definition 2.1 (Graph structures). A *graph structure signature* $GS = (S, OP, \Sigma)$ consists of a set of sorts S, a family $(OP_{s,s'})_{s,s' \in S}$ of *unary* operator symbols and a family $(\Sigma_s)_{s \in S}$ of labelling alphabets.

A *graph structure* A over GS is a sort-indexed family $(A_s)_{s \in S}$ of carrier sets together with a sort-indexed family of labelling functions $(l_s^A)_{s \in S}$ such that $l_s^A : A_s \to \Sigma_s$ and an OP-indexed family of mappings $(op^A)_{op \in OP}$ such that $op^A : A_s \to A_{s'}$ if $op \in OP_{s,s'}$.

A *graph structure morphism* $\varphi : A \to B$ is a sort-indexed family of mappings $\varphi = (\varphi_s : A_s \to B_s)_{s \in S}$ such that $l_s^A(x) = l_s^B(\varphi(x))$ and $op^B(\varphi(x)) = \varphi(op^A(x))$ for all $x \in A_s$.

For a fixed signature, graph structures and graph structure morphisms define a category **Graph**.

A graph structure morphism φ is called *injective* if all its mappings are injective. It is an *isomorphism* if all mappings are bijective. An isomorphism of the form $\varphi : A \to A$ is called an *automorphism.*

The simplest graph structure signature has two sorts, *node* and *edge*, and two operator symbols $s, t \in OP_{edge,node}$ standing for 'source' and 'target'. Graph structures over this signature are ordinary labelled directed graphs, and graph structure morphisms are standard graph morphisms. The sets Σ_{node} and Σ_{edge} contain node and edge labels, respectively. In the following we will say 'graph' instead of 'graph structure' and 'graph morphism' or just 'morphism' instead of 'graph structure morphism'. For this reason, the category of graph structures and graph structure morphisms based on some graph signature GS is simply denoted by **Graph**.

The category **Graph** has all pushouts and pullbacks, which can be constructed componentwise in the category **Set**. Furthermore, constructing the pushout or pullback of two injective morphisms always gives us two injective morphisms. Working exclusively in the category of injective morphisms is not possible since this category does not have all pushouts and pullbacks, which is due to missing non-injective mediating morphisms. See Appendix A for additional properties of injective morphisms. Working with injective morphisms is a natural requirement, but it is possible to relax the requirement of injectivity to some extent (see conclusion).

More generally, our results hold also for adhesive categories, where the term 'injective morphism' has to be replaced by 'monomorphism'. Note that the category of graph structures defined above is an adhesive category. A more detailed discussion of this issue can also be found in the conclusion and in Appendix A.

Definition 2.2 (Graph transformation system). A *rule* or *production* is a pair $p = (\varphi_L : I \to L, \varphi_R : I \to R)$ of injective graph morphisms. Given the production p and a graph G with injective morphism $g : L \to G$, we obtain a graph transformation $G \Rightarrow H$ if we can find a graph C and morphisms c, h, c_1, c_2 such that in the following diagram (1) becomes a pushout and (2) can be constructed as a pushout of c and φ_R in the category **Graph**, where all morphisms in (1) and (2) are injective.

$$\begin{tikzcd}
L \arrow[d, "g"'] \arrow[dr, phantom, "(1)"] & I \arrow[l, "\varphi_L"'] \arrow[r, "\varphi_R"] \arrow[d, "c"'] \arrow[dr, phantom, "(2)"] & R \arrow[d, "h"] \\
G & C \arrow[l, "c_1"] \arrow[r, "c_2"'] & H
\end{tikzcd}$$

A *graph transformation system* is given by a set $\mathscr{P}$ of productions. Unlike the case with a graph grammar, we do not require a start graph.

The diagram above, consisting of two pushouts, is the reason for the name double-pushout or DPO approach, which was introduced in Ehrig *et al.* (1973). The intuition behind this approach is first to find a left-hand side L in a graph G, then to remove L,

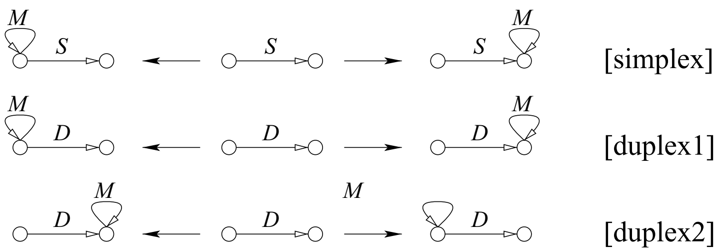

Fig. 1. Rules of a graph transformation system.

apart from the interface I, and, finally, to attach R to the interface in the remaining graph C, resulting in H.

Note: Instead of writing $(\varphi_L : I \to L, \varphi_R : I \to R)$, we will in the following abbreviate a rule by $(L \xleftarrow{\varphi_L} I \xrightarrow{\varphi_R} R)$, or even $(L \leftarrow I \to R)$ if there is no danger of misunderstanding. This short form will be used for other morphisms as well.

Throughout the paper we will use a running example, which is deliberately kept very simple. Figure 1 shows three spans $L \leftarrow I \to R$, which form the rule set $\mathscr{P}$ of our example graph transformation system. The graphs are directed graphs with edge labels where nodes are unlabelled (or are labelled with a dummy label). We give rules for a simplex connection S and a duplex connection D, over both of which messages M, which are represented by a loop, are sent. A duplex connection can be used in both directions, whereas a simplex connection has a fixed direction. The connections themselves are preserved, and are therefore in the interfaces of the rules. An alternative choice, which is also covered by the concept of graph structures, would have been to model this situation by hypergraphs with unary edges (for messages) and binary edges (for connections).

In order to state congruence results, we first need a notion of contexts and contexualisation.

Definition 2.3 (Graphs with interfaces and graph contexts). A graph G with *interface* J is an injective morphism $J \to G$. Furthermore, a *context* or *cospan* consists of two injective morphisms $J \to E \leftarrow \overline{J}$.

The composition of a graph with interface $J \to G$ and a context $J \to E \leftarrow \overline{J}$ is a graph with interface $\overline{J} \to \overline{G}$, which is obtained by constructing $\overline{G}$ as the pushout of $J \to G$ and $J \to E$:

$$\begin{tikzcd}
J \arrow[r] \arrow[d] & E \arrow[d] & \overline{J} \arrow[l] \arrow[dl, dashed] \\
G \arrow[r] & \overline{G} &
\end{tikzcd}$$

Note that composition is defined only up to isomorphism, since the pushout object is unique only up to isomorphism.

This notion of interfaces, contexts and composition is within the spirit of the DPO approach, where the pushouts for G and H in Definition 2.2 can be interpreted as the composition of L with C and R with C, respectively, along interface I. In the context of this paper, however, it is important also to consider the graph G with interface J leading to $\overline{G}$ with interface $\overline{J}$, which requires a context E with two interfaces J and $\overline{J}$. Discrete interfaces, which are a special case, have already been used – see, for instance, Gadducci and Heckel (1997).

3. Rewriting with borrowed contexts

We are now ready for the central definition of this paper: graph rewriting with borrowed contexts on graphs with interfaces. The underlying idea is to allow not only total, but also partial matches of a left-hand side. The missing part of the left-hand side is then displayed as the label of the resulting transition.

Definition 3.1 (Rewriting with borrowed contexts). Let $\mathscr{P}$ be a set of graph productions of the form $(L \leftarrow I \rightarrow R)$, and let $J \rightarrow G$ be a graph with interface. We say that $J \rightarrow G$ reduces to $K \rightarrow H$ with transition label $(J \rightarrow F \leftarrow K)$ whenever there is a production $(L \leftarrow I \rightarrow R) \in \mathscr{P}$ and there are graphs D, G^+, C and additional morphisms such that the following diagram commutes and the squares are, as indicated, either pushouts (PO) or pullbacks (PB) with injective morphisms.

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[rd, phantom, "\text{PO}"] & L \arrow[d] \arrow[rd, phantom, "\text{PO}"] & I \arrow[l] \arrow[r] \arrow[d] \arrow[rd, phantom, "\text{PO}"] & R \arrow[d] \\
G \arrow[r] & G^+ & C \arrow[l] \arrow[r] & H \\
J \arrow[u] \arrow[r] \arrow[ru, phantom, "\text{PO}"] & F \arrow[u] \arrow[ru, phantom, "\text{PB}"] & K \arrow[l] \arrow[u] \arrow[ru, dotted] &
\end{tikzcd}$$

Symbolically, this is denoted by the transition $(J \rightarrow G) \overset{J \rightarrow F \leftarrow K}{\longrightarrow} (K \rightarrow H)$, which is also called a *rewriting step with borrowed context.*

The squares in the diagram above have the following meaning: the upper left-hand square merges the left-hand side L and the graph G to be rewritten according to a partial match $G \leftarrow D \rightarrow L$ of the left-hand side in G. The resulting graph G^+ contains a total match of L and can be rewritten as in the standard DPO approach, which produces the two remaining squares in the upper row. The pushout in the lower row gives us the borrowed (or minimal) context F, which is missing, in order to obtain a total match of L, along with a morphism $J \rightarrow F$ indicating how F should be attached to G. Finally, we need an interface for the resulting graph H, which can be obtained by 'intersecting' the borrowed context F and the graph C via a pullback.

The graph C in the figure above is called a *pushout complement* of the arrows $I \rightarrow L$ and $L \rightarrow G^+$. Similarly, F is a pushout complement of $J \rightarrow G$ and $G \rightarrow G^+$. These two pushout complements may not exist: the middle square in the upper row can only be completed if the dangling edge condition is satisfied, that is, if the left-hand side L is connected to the rest of the graph G^+ exclusively via its interface I and no edges would be left 'dangling' by removing it; and the left square in the lower row can only be completed if there is a way to extend the partial match to a left-hand side L by attaching some

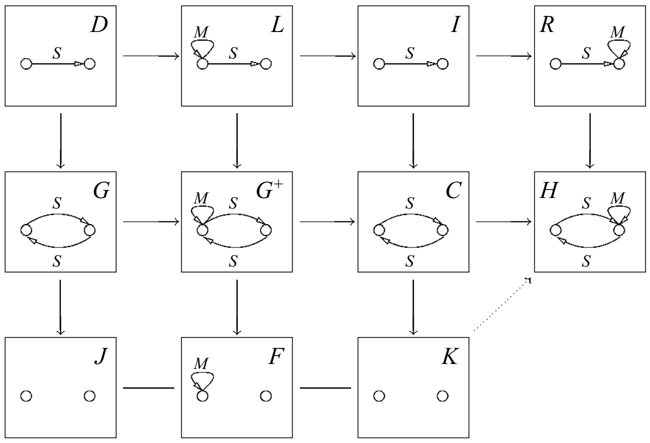

Fig. 2. Rewriting with borrowed contexts in the example graph transformation system.

context $J \to F$ to $J \to G$. In other words, the dangling edge condition is also required for the morphism $G \to G^+$ with respect to the interface morphism $J \to G$.

In this case, the borrowed context F is minimal in the following sense. Given the partial match $G \leftarrow D \to L$, the pushout G^+ is the minimal graph containing both G and L that is attached according to the partial match. The borrowed context F is a pushout complement of the injective morphisms $J \to G \to G^+$, leading to the injective morphisms $J \to F \to G^+$. Note that pushout complements along injective morphisms are unique up to isomorphism (see Property A.8), which implies that F is the unique graph (up to isomorphism) that is needed to extend G to the minimal graph G^+.

From the properties stated in Appendix A, one can infer that all morphisms in the diagram above are injective. It is thus possible to depict this situation by drawing graphs as Venn-like diagrams as shown in Figures 7 and 8 in Appendix B. This figure also illustrates that the new interface K is the 'union' of the interfaces I and J, minus the graph components that are internal to either G or L.

In order to illustrate Definition 3.1, we will consider the [simplex] rule of Figure 1 and an example graph G consisting of two S-edges for which we find a partial match of the left-hand side. This results in the derivation shown in Figure 2. Note that the image of a node under a morphism is implicitly given by its position, that is, the left-hand node is always mapped to a left-hand node, and analogously for the right-hand node.

4. Bisimilarity is a congruence

We now arrive at the main theorem of this paper, which says that the bisimilarity defined on labelled graph transition systems is a congruence. However, we need two more definitions first.

Definition 4.1 (Bisimulation and bisimilarity). Let $\mathscr{P}$ be a set of productions and $\mathscr{R}$ be a symmetric relation consisting of pairs of graphs with interfaces of the form $(J \to G, J \to G')$, also written $(J \to G)\,\mathscr{R}\,(J \to G')$.

The relation $\mathscr{R}$ is a *bisimulation* if whenever we have that $(J \to G)\mathscr{R}(J \to G')$ and a transition $(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H)$ (in words: $J \to G$ reduces to $K \to H$ with transition label $J \to F \leftarrow K$) can be derived from $\mathscr{P}$, then there exists a morphism $K \to H'$ and a transition $(J \to G') \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H')$ with the same transition label $J \to F \leftarrow K$ such that $(K \to H)\mathscr{R}(K \to H')$.

We write $(J \to G) \sim (J \to G')$ whenever there exists a bisimulation $\mathscr{R}$ that relates the two morphisms. The relation $\sim$ is called *bisimilarity*.

In order to state Theorem 4.3, we must be able to close a bisimulation or simply a relation under all possible contexts.

Definition 4.2 (Closure under contexts). Let $\mathscr{R}$ be a relation on graphs with interfaces as in Definition 4.1. We use $\hat{\mathscr{R}}$ to denote the closure of $\mathscr{R}$ under contexts, that is, $\hat{\mathscr{R}}$ is the smallest relation that contains, for every pair $(J \to G, J \to G') \in \mathscr{R}$ and for every context of the form $J \to E \leftarrow \overline{J}$, the pair of morphisms $(\overline{J} \to \overline{G}, \overline{J} \to \overline{G}')$ that results from the composition of $J \to G$ and $J \to E \leftarrow \overline{J}$, and $J \to G'$ and $J \to E \leftarrow \overline{J}$, respectively.

A relation $\mathscr{R}$ is a congruence, that is, closed under contexts, whenever $\hat{\mathscr{R}} = \mathscr{R}$. Since $\mathscr{R}$ is obviously contained in $\hat{\mathscr{R}}$, it suffices to show $\hat{\mathscr{R}} \subseteq \mathscr{R}$.

Theorem 4.3 (Bisimilarity is a congruence). Whenever $\mathscr{R}$ is a bisimulation, $\hat{\mathscr{R}}$ is a bisimulation as well. This implies that the bisimilarity relation $\sim$ is a congruence.

Proof. In this proof we will refer to the pushout-pullback properties of graph structure categories given in Appendix A.

With the following argument we can infer that $\hat{\sim} \subseteq \sim$, which implies that $\sim$ is a congruence. Whenever $(\overline{J} \to \overline{G}) \hat{\sim} (\overline{J} \to \overline{G}')$, there exists a bisimulation $\mathscr{R}$ such that $(\overline{J} \to \overline{G})\hat{\mathscr{R}}(\overline{J} \to \overline{G}')$. Since, as we will show, $\hat{\mathscr{R}}$ is a bisimulation as well, it follows that $(\overline{J} \to \overline{G}) \sim (\overline{J} \to \overline{G}')$

So, we let $\mathscr{R}$ be a bisimulation and let $(\overline{J} \to \overline{G})\hat{\mathscr{R}}(\overline{J} \to \overline{G}')$, where the pair $(\overline{J} \to \overline{G}, \overline{J} \to \overline{G}')$ has been obtained from the pair $(J \to G, J \to G')$ by composition with a context $J \to E \leftarrow \overline{J}$ as described in Definition 4.2. We assume that

$$(\overline{J} \to \overline{G}) \overset{\overline{J}\to\overline{F}\leftarrow\overline{K}}{\longrightarrow} (\overline{K} \to \overline{H}).$$

Our goal is to show that there exists a transition

$$(\overline{J} \to \overline{G}') \overset{\overline{J}\to\overline{F}\leftarrow\overline{K}}{\longrightarrow} (\overline{K} \to \overline{H}')$$

with $(\overline{K} \to \overline{H})\hat{\mathscr{R}}(\overline{K} \to \overline{H}')$, which implies that $\hat{\mathscr{R}}$ is a bisimulation. In Step 1 we will construct a transition $(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H)$, which implies a transition $(J \to G') \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H')$ with $(K \to H)\mathscr{R}(K \to H')$, since $\mathscr{R}$ is a bisimulation. In Step 2 we will extend the second transition to obtain the transition stated in our goal above.

Step 1: Our first assumption, $(\overline{J} \to \overline{G})\hat{\mathscr{R}}(\overline{J} \to \overline{G}')$, means that there is some pair $(J \to G)\mathscr{R}(J \to G')$ and a context $J \to E \leftarrow \overline{J}$ such that $\overline{J} \to \overline{G}$ and $\overline{J} \to \overline{G}'$ can be obtained by composing $J \to G$ and $J \to G'$ with this context. The second assumption is the transition $(\overline{J} \to \overline{G}) \overset{\overline{J}\to\overline{F}\leftarrow\overline{K}}{\longrightarrow} (\overline{K} \to \overline{H})$, which leads to the situation depicted in

Diagram (1), where the decomposition of $\overline{J} \to \overline{G}$ is shown explicitly and all morphisms are injective and all (basic) squares are pushouts, apart from the square consisting of the graphs $\overline{K}$, $\overline{C}$, $\overline{F}$, $\overline{G}^+$, which is a pullback.

$$\begin{tikzcd}
 & \overline{D} \arrow[r] \arrow[d] & L \arrow[d] & I \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\
G \arrow[r] & \overline{G} \arrow[r] & \overline{G}^+ & \overline{C} \arrow[l] \arrow[r] & \overline{H} \\
J \arrow[u] \arrow[r] & E \arrow[u] & & & \\
 & \overline{J} \arrow[u] \arrow[r] & \overline{F} \arrow[uu] & \overline{K} \arrow[l] \arrow[uu] \arrow[uur, dotted] &
\end{tikzcd} \tag{1}$$

$$\begin{tikzcd}
 & \overline{D} \arrow[r] \arrow[d] & L \arrow[d] & I \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\
G \arrow[r] & \overline{G} \arrow[r] & \overline{G}^+ & \overline{C} \arrow[l] \arrow[r] & \overline{H} \\
J \arrow[u] \arrow[r] & E \arrow[u] \arrow[r] & E_2 \arrow[u] & E_1 \arrow[l] \arrow[u] \arrow[ur, dotted] & \\
 & \overline{J} \arrow[u] \arrow[r] & \overline{F} \arrow[u] & \overline{K} \arrow[l] \arrow[u] \arrow[uur, dotted] &
\end{tikzcd} \tag{2}$$

We can now split the lower pushout and the lower pullback along E (see Diagram (2)). The left-hand square splits into two pushouts (by classical pushout splitting – see the remark after Property A.3) and the right-hand square splits into two pullbacks (by classical pullback splitting). Furthermore, all morphisms in Diagram (2) are injective.

We can now construct D as the pullback of $G \to \overline{G}$ and $\overline{D} \to \overline{G}$, followed by the construction of $\tilde{G}$ as the pushout (see Diagram (3)). In this way, we split the morphisms $G \to \overline{G}$ and $\overline{D} \to \overline{G}$ and obtain the morphism $\tilde{G} \to \overline{G}$ as an induced morphism. According to Property A.1, all morphisms in Diagram (3) are injective.

$$\begin{tikzcd}
 & \overline{D} \arrow[dr] \arrow[drr] & & \\
D \arrow[ur] \arrow[dr] & & \tilde{G} \arrow[r, dotted] & \overline{G} \\
 & G \arrow[ur] \arrow[urr] & &
\end{tikzcd} \tag{3}$$

In Diagram (2), we can now split the upper row of pushouts and the pushout at the far left by classical pushout splitting, and using Property A.3, respectively, which gives us Diagram (4). We obtain the graphs F_1, G^+, C and H. Note that the 'triangle' consisting of D, $\overline{D}$, G, $\tilde{G}$ in the upper left corner is also a pushout.

We can now complete the partial cubes in the middle of the diagram by adding F and K with corresponding morphisms (see Diagram (7)). Diagram (7) is also illustraed in Figure 10 in Appendix B, where graphs are represented schematically in a Venn-like fashion (see also Figure 9). We first focus on the left cube and construct F as the pullback of $G^+ \to \overline{G}^+$ and $E_2 \to \overline{G}^+$, and obtain $F_1 \to F$ as an induced morphism. From

Property A.1 we can infer that $F \to G^+$ and $F \to E_2$ are injective, and $F_1 \to F$ is injective since it is the first morphism of an injective composition.

$$\begin{tikzcd}
D \arrow[rr] \arrow[dd] & & \overline{D} \arrow[rr] \arrow[dl] \arrow[dd] & & L \arrow[dl] \arrow[dd] & & I \arrow[ll] \arrow[rr] \arrow[dl] \arrow[dd] & & R \arrow[dl] \arrow[dd] \\
& \tilde{G} \arrow[rr] \arrow[dr] & & G^+ \arrow[dr] & & C \arrow[ll] \arrow[rr] \arrow[dr] & & H \arrow[dr] & \\
G \arrow[rr] \arrow[ur] & & \overline{G} \arrow[rr] & & \overline{G}^+ & & \overline{C} \arrow[ll] \arrow[rr] & & \overline{H} \\
& F_1 \arrow[uu] \arrow[dr] & & & & & & & \\
J \arrow[uu] \arrow[rr] \arrow[ur] & & E \arrow[rr] \arrow[uu] & & E_2 \arrow[uu] & & E_1 \arrow[ll] \arrow[uu] \arrow[uurr, dotted] & & \\
& & \overline{J} \arrow[rr] \arrow[u] & & \overline{F} \arrow[u] & & \overline{K} \arrow[ll] \arrow[u] \arrow[uuurr, dotted] & &
\end{tikzcd} \tag{4}$$

Since the left and top squares of the cube are pushouts, this implies that the outer square in Diagram (5) is a pushout as well. Since the right square in Diagram (5) is a pullback, Property A.4 implies that both squares are pushouts. Similarly, we can infer that the outer square in Diagram (6) is a pushout, since the left and front squares of the cube are pushouts. Again by Property A.4, we get that both squares in Diagram (6) are pushouts. That is, the left cube commutes, all morphisms are injective and all squares are pushouts.

$$\begin{tikzcd}
F_1 \arrow[r] \arrow[d] & F \arrow[r] \arrow[d] & G^+ \arrow[d] \\
E \arrow[r] & E_2 \arrow[r] & \overline{G}^+
\end{tikzcd} \tag{5}$$

$$\begin{tikzcd}
F_1 \arrow[r] \arrow[d] & F \arrow[r] \arrow[d] & E_2 \arrow[d] \\
\tilde{G} \arrow[r] & G^+ \arrow[r] & \overline{G}^+
\end{tikzcd} \tag{6}$$

Focussing on the right-hand cube, we can infer from Property A.2 that the left and top squares of the cube are pullbacks, since they are pushouts. We construct K as the pullback of the morphisms $C \to \overline{C}$ and $E_1 \to \overline{C}$, and get $K \to F$ as an induced morphism by using the pullback property of the left square. The morphisms $K \to C$, $K \to E_1$ are injective by Property A.1, and $K \to F$ is injective as the first morphism of an injective composition. The left, top and right squares are pullbacks, so we can now infer by classical pullback decomposition that the bottom square is also a pullback. Furthermore, since the bottom, front and top squares are pullbacks, the back square is also a pullback. Finally, we can infer from Property A.6 that the right square is a pushout (since the left square is a pushout) and that the bottom square is a pushout (since the top square is a pushout). We thus obtain a commuting right cube where all morphisms are injective, all squares are pullbacks and the left, right, top and bottom squares are also pushouts. This implies, specifically, that the square consisting of K, E_1, H, $\overline{H}$ is a pushout since it is the composition of two pushouts.

$$\begin{tikzcd}
D \arrow[rr] \arrow[dd] & & \overline{D} \arrow[rr] \arrow[dl, dashed] \arrow[dd] & & L \arrow[dl] \arrow[dd] & & I \arrow[ll] \arrow[rr] \arrow[dl] \arrow[dd] & & R \arrow[dl] \arrow[dd] \\
& \tilde{G} \arrow[rr] \arrow[dr] & & G^+ \arrow[dr] & & C \arrow[ll] \arrow[rr] \arrow[dr] & & H \arrow[dr] & \\
G \arrow[rr] \arrow[ur] & & \overline{G} \arrow[rr] & & \overline{G}^+ & & \overline{C} \arrow[ll] \arrow[rr] & & \overline{H} \\
& F_1 \arrow[rr] \arrow[uu] \arrow[dr] & & F \arrow[uu] \arrow[dr] & & K \arrow[ll] \arrow[uu] \arrow[dr] \arrow[uurr, dotted] & & & \\
J \arrow[rr] \arrow[uu] \arrow[ur] & & E \arrow[rr] \arrow[uu] & & E_2 \arrow[uu] & & E_1 \arrow[ll] \arrow[uu] \arrow[uurr, dotted] & & \\
& & \overline{J} \arrow[rr] \arrow[u] & & \overline{F} \arrow[u] & & \overline{K} \arrow[ll] \arrow[u] \arrow[uuurr, dotted] & &
\end{tikzcd} \tag{7}$$

From Diagram (7) we can derive the following transition:

$$(J \to G) \xrightarrow{J \to F \leftarrow K} (K \to H),$$

using the notation of Definition 3.1. Since $\mathscr{R}$ is a bisimulation, this implies

$$(J \to G') \xrightarrow{J \to F \leftarrow K} (K \to H')$$

with $(K \to H)\mathscr{R}(K \to H')$. Furthermore, we can infer from Diagram (7) that $\overline{K} \to \overline{H}$ can be obtained by composing $K \to H$ with the context $K \to E_1 \leftarrow \overline{K}$.

Step 2: As mentioned above, we will now extend the transition from $J \to G'$ to $K \to H'$ with the $(K \to H)\mathscr{R}(K \to H')$ obtained above to construct a transition from $(\overline{J} \to \overline{G}')$ to $(\overline{K} \to \overline{H}')$ with $(\overline{K} \to \overline{H})\hat{\mathscr{R}}(\overline{K} \to \overline{H}')$. We will construct $\overline{K} \to \overline{H}'$ in such a way that it is the composition of $K \to H'$ with the context $K \to E_1 \leftarrow \overline{K}$. Recall also that $\overline{J} \to \overline{G}'$ is the composition of $J \to G'$ and the context $J \to E \leftarrow \overline{J}$.

We now cut away the upper layer of Diagram (7) to give Diagram (8), where all squares are pushouts, apart from the square consisting of $\overline{K}$, $\overline{F}$, E_1 and E_2, which is a pullback.

$$\begin{tikzcd}
& F_1 \arrow[rr] \arrow[dr] & & F \arrow[dr] & & K \arrow[ll] \arrow[dr] & \\
J \arrow[ur] \arrow[rr] & & E \arrow[rr] & & E_2 & & E_1 \arrow[ll] \\
& & \overline{J} \arrow[u] \arrow[rr] & & \overline{F} \arrow[u] & & \overline{K} \arrow[u] \arrow[ll]
\end{tikzcd} \tag{8}$$

From the derivation step of $J \to G'$ given earlier, one can derive Diagram (9) for some rule $L' \leftarrow I' \to R'$, where the lower right-hand square is a pullback and all other squares are pushouts. The morphism $J \to F$ is split by F_1, so we can split the two left-hand side pushouts by classical pushout splitting and by Property A.3, as shown in Diagram (10).

$$\begin{tikzcd}
D' \arrow[r] \arrow[d] & L' \arrow[d] & I' \arrow[l] \arrow[r] \arrow[d] & R' \arrow[d] \\
G' \arrow[r] & G'^{+} & C' \arrow[l] \arrow[r] & H' \\
J \arrow[r] \arrow[u] & F \arrow[u] & K \arrow[l] \arrow[u] \arrow[ur, dotted] &
\end{tikzcd} \tag{9}$$

$$\begin{tikzcd}
D' \arrow[r] \arrow[d] & \overline{D}' \arrow[r] \arrow[d] & L' \arrow[d] & I' \arrow[l] \arrow[r] \arrow[d] & R' \arrow[d] \\
G' \arrow[r] & \tilde{G}' \arrow[r] & G'^{+} & C' \arrow[l] \arrow[r] & H' \\
J \arrow[r] \arrow[u] & F_1 \arrow[r] \arrow[u] & F \arrow[u] & K \arrow[l] \arrow[u] \arrow[ur, dotted] &
\end{tikzcd} \tag{10}$$

Composing Diagrams (8) and (10) gives us Diagram (11).

$$\begin{tikzcd}
D' \arrow[rr] \arrow[dd] & & \overline{D}' \arrow[rr] \arrow[dl] & & L' \arrow[dl] & & I' \arrow[ll] \arrow[rr] \arrow[dl] & & R' \arrow[dl] \\
& \tilde{G}' \arrow[rr] & & G'^{+} & & C' \arrow[ll] \arrow[rr] & & H' & \\
G \arrow[ur] & & & & & & & & \\
& F_1 \arrow[rr] \arrow[uu] \arrow[dr] & & F \arrow[uu] \arrow[dr] & & K \arrow[ll] \arrow[uu] \arrow[uurr, dotted] \arrow[dr] & & & \\
J \arrow[uu] \arrow[ur] \arrow[rr] & & E \arrow[rr] & & E_2 & & E_1 \arrow[ll] & & \\
& & \overline{J} \arrow[u] \arrow[rr] & & \overline{F} \arrow[u] & & \overline{K} \arrow[u] \arrow[ll] & &
\end{tikzcd} \tag{11}$$

We still need to complete the two cubes in the middle of Diagram (11), which will lead to Diagram (13). We first construct $\overline{G}'$ as the pushout of $F_1 \to \tilde{G}'$ and $F_1 \to E$ and $\overline{G}'^{+}$ as the pushout of $F \to G'^{+}$ and $F \to E_2$ leading to an induced morphism $\overline{G}' \to \overline{G}'^{+}$. From Property A.1, we can infer that all morphisms apart from $\overline{G}' \to \overline{G}'^{+}$ are injective. By classical pushout decomposition, we have that the top square is a pushout (since the left, bottom and right squares are pushouts) and that the front square is a pushout (since the back, top and bottom squares are pushouts). From Property A.1, we can now infer that $\overline{G}' \to \overline{G}'^{+}$ is injective. Hence all squares in the left cube are pushouts, all morphisms are injective and all squares are also pullbacks.

In the second cube we construct $\overline{C}'$ as a pushout of $K \to C'$ and $K \to E_1$, and obtain $\overline{C}' \to \overline{G}'^{+}$ as an induced morphism. Since the left, bottom and right squares are pushouts, by classical pushout decomposition, the top square is a pushout as well. Hence, all new morphisms are injective. In order to show that the front square is a pullback, we consider Diagram (12), where the left square is the right square of the cube and the right square is the front square of the cube. Since the back and left squares of the cube are pullbacks, the outer square of Diagram (12) is also a pullback. Moreover, the left square below is a pushout. Hence Property A.5 implies that both squares in Diagram (12) are pullbacks.

$$\begin{tikzcd}
K \arrow[r] \arrow[d] & E_1 \arrow[r] \arrow[d] & E_2 \arrow[d] \\
C' \arrow[r] & \overline{C}' \arrow[r] & \overline{G}'^{+}
\end{tikzcd} \tag{12}$$

This implies that all morphisms in the right-hand cube are injective, all squares are pullbacks and the left, top, right and bottom squares are pushouts.

$$\begin{tikzcd}
D' \arrow[rr] \arrow[dd] & & \overline{D}' \arrow[rr] \arrow[dl] \arrow[dd] & & L' \arrow[dl] \arrow[dd] & & I' \arrow[ll] \arrow[rr] \arrow[dl] \arrow[dd] & & R' \arrow[dl] \arrow[dd] \\
& \tilde{G}' \arrow[rr] \arrow[dr] & & G'^{+} \arrow[dr] & & C' \arrow[ll] \arrow[rr] \arrow[dr] & & H' \arrow[dr] & \\
G' \arrow[ur] \arrow[rr] & & \overline{G}' \arrow[rr] & & \overline{G}'^{+} & & \overline{C}' \arrow[ll] \arrow[rr] & & \overline{H}' \\
& F_1 \arrow[uu] \arrow[rr] \arrow[dr] & & F \arrow[uu] \arrow[dr] & & K \arrow[ll] \arrow[uu] \arrow[dr] \arrow[uurr, dotted] & & & \\
J \arrow[uu] \arrow[ur] \arrow[rr] & & E \arrow[uu] \arrow[rr] & & E_2 \arrow[uu] & & E_1 \arrow[ll] \arrow[uu] \arrow[uurr, dotted] & & \\
& & \overline{J} \arrow[u] \arrow[rr] & & \overline{F} \arrow[u] & & \overline{K} \arrow[ll] \arrow[u] \arrow[uuurr, dotted] & &
\end{tikzcd} \tag{13}$$

In Diagram (13), the three right-hand squares in the upper row are all pushouts since each of them is the composition of two pushouts, the square $\overline{J}, \overline{F}, \overline{G}', \overline{G}'^{+}$ is a pushout and the square $\overline{K}, \overline{F}, \overline{C}', \overline{G}'^{+}$ is a pullback. Hence, by Definition 3.1, we infer that

$$(\overline{J} \to \overline{G}') \overset{\overline{J} \to \overline{F} \leftarrow \overline{K}}{\longrightarrow} (\overline{K} \to \overline{H}'),$$

and since the square consisting of K, H', E_1 and $\overline{H}'$ is also a pushout, as the composition of two pushouts, we can infer that $\overline{K} \to \overline{H}'$ can be obtained by composing $K \to H'$ and the context $K \to E_1 \leftarrow \overline{K}$. From our earlier considerations, we know that $\overline{K} \to \overline{H}$ is the composition of $K \to H$ with $K \to E_1 \leftarrow \overline{K}$, so $(\overline{K} \to \overline{H}) \hat{\mathscr{R}} (\overline{K} \to \overline{H}')$. This means that we have achieved the goal stated at the beginning of the proof, which implies that $\hat{\mathscr{R}}$ is a bisimulation and $\sim$ is a congruence. □

Remark: In the definition of rewriting with borrowed contexts (Definition 3.1) and in our main result above, Theorem 4.3 ('Bisimilarity is a congruence'), we have, respectively, assumed and shown that all graph morphisms are injective. In fact, it can be shown that our definitions and results can be extended to the case where the morphisms of the production $p = (L \leftarrow I \to R)$, the span $(G^+ \leftarrow C \to H)$ and the morphisms $J \to G$,

$F \to G^+$, $K \to C$, $K \to F$ and $K \to H$ are non-injective. This corresponds to the case of non-injective productions with injective matches $L \to G^+$ in the classical DPO approach, which has not been studied explicitly up to now. This extended version of rewriting with borrowed contexts has been proposed in a Dagstuhl seminar by Sassone and Sobociński and in their technical report Sassone and Sobociński (2004).

5. Proof techniques

In order to pursue the example further, we will first introduce a proof technique simplifying bisimilarity proofs. This technique is a straightforward instance of an up-to technique (Milner and Sangiorgi 1992b; Sangiorgi 1995). The underlying idea behind the technique is the observation that the relation $\mathscr{R}$ should be as small as possible, in order to obtain a compact proof. This goal can be reached by slightly extending the notion of bisimulation by demanding that if a transition is matched by another, the pair of resulting graphs can be found in $\mathscr{R}$ after removal of identical contexts. Hence, this extended notion of bisimulation is called 'bisimulation up to context'. We first need an auxiliary definition.

Definition 5.1 (Progression). Let $\mathscr{R}, \mathscr{S}$ be relations containing pairs of graphs with interfaces of the form $(J \to G, J \to G')$, where $\mathscr{R}$ is symmetric. We say that $\mathscr{R}$ progresses to $\mathscr{S}$, abbreviated by $\mathscr{R} \rightarrowtail \mathscr{S}$, if whenever $(J \to G)\,\mathscr{R}\,(J \to G')$ and $(J \to G) \xrightarrow{J \to F \leftarrow K} (K \to H)$, there exists a morphism $K \to H'$ such that $(J \to G') \xrightarrow{J \to F \leftarrow K} (K \to H')$ and $(K \to H)\,\mathscr{S}\,(K \to H')$.

For example, $\mathscr{R}$ is a bisimulation if and only if $\mathscr{R} \rightarrowtail \mathscr{R}$.

Definition 5.2 (Bisimulation up to context). Let $\mathscr{R}$ be a symmetric relation containing pairs of graphs with interfaces of the form $(J \to G, J \to G')$.

If $\mathscr{R} \rightarrowtail \hat{\mathscr{R}}$, then $\mathscr{R}$ is called a *bisimulation up to context*.

We will show in Proposition 5.3 that every bisimulation up to context is contained in the bisimilarity $\sim$. The attractiveness of bisimulations up to context stems from the fact that such a relation can be much smaller than the least bisimulation that contains it, and thus proofs can be compressed. This technique might even allow us to work with a finite relation instead of an infinite one.

Proposition 5.3 (Bisimulation up to context implies bisimilarity). Let $\mathscr{R}$ be a bisimulation up to context. Then $\mathscr{R} \subseteq \sim$.

Proof (Sketch). By examining the proof of Theorem 4.3 carefully, we can see that some simple modifications give us the following (stronger) theorem:

$$\text{If } \mathscr{R} \rightarrowtail \mathscr{S}, \text{ then } \hat{\mathscr{R}} \rightarrowtail \hat{\mathscr{S}} \text{ also.}$$

Since $\mathscr{R}$ is a bisimulation up to context, we have $\mathscr{R} \rightarrowtail \hat{\mathscr{R}}$. The stronger version of Theorem 4.3 now implies $\hat{\mathscr{R}} \rightarrowtail \widehat{(\hat{\mathscr{R}})}$. Attaching two contexts in sequence is equivalent to attaching only one context. Hence, we have $\widehat{(\hat{\mathscr{R}})} = \hat{\mathscr{R}}$, which implies $\hat{\mathscr{R}} \rightarrowtail \hat{\mathscr{R}}$, and hence that $\hat{\mathscr{R}}$ is a bisimulation, that is, $\hat{\mathscr{R}} \subseteq \sim$. This implies $\mathscr{R} \subseteq \hat{\mathscr{R}} \subseteq \sim$. □

Since composition with a context is defined only up to isomorphism, we can assume that $\hat{\mathscr{R}}$ is closed under isomorphism in the following sense: for every span $G \leftarrow J \rightarrow G'$ (also written $(J \rightarrow G, J \rightarrow G')$), all isomorphic spans $\tilde{G} \leftarrow \tilde{J} \rightarrow \tilde{G}'$ are also contained in $\hat{\mathscr{R}}$.

Similarly, we can restrict ourselves to abstract transitions when checking for bisimilarity. Assume that $(J \rightarrow G)\mathscr{R}(J \rightarrow G')$ and there is a transition $(J \rightarrow G) \overset{J \rightarrow F \leftarrow K}{\longrightarrow} (K \rightarrow H)$. A transition of the form $(J \rightarrow G) \overset{J \rightarrow \tilde{F} \leftarrow \tilde{K}}{\longrightarrow} (\tilde{K} \rightarrow \tilde{H})$ is considered to be equivalent to the first transition whenever we can find isomorphisms from $\tilde{F}, \tilde{K}, \tilde{H}$ to F, K, H, respectively, such that Diagram (14) commutes. Furthermore, each such commuting diagram corresponds to a transition.

In bisimulation proofs it is sufficient to inspect only one representative in such an equivalence class and to show the existence of a matching transition $(J \rightarrow G') \overset{J \rightarrow F \leftarrow K}{\longrightarrow} (K \rightarrow H')$ such that $(K \rightarrow H)\hat{\mathscr{R}}(K \rightarrow H')$.

The remaining transitions $(J \rightarrow G) \overset{J \rightarrow \tilde{F} \leftarrow \tilde{K}}{\longrightarrow} (\tilde{K} \rightarrow \tilde{H})$ in the equivalence class can then be dealt with as follows. From $(J \rightarrow G') \overset{J \rightarrow F \leftarrow K}{\longrightarrow} (K \rightarrow H')$ and Diagram (14), we can derive Diagram (15), where the arrows pointing upwards are again isomorphisms and the diagram commutes. This gives a transition $(J \rightarrow G') \overset{J \rightarrow \tilde{F} \leftarrow \tilde{K}}{\longrightarrow} (\tilde{K} \rightarrow \tilde{H}')$ such that $H \leftarrow K \rightarrow H'$ and $\tilde{H} \leftarrow \tilde{K} \rightarrow \tilde{H}'$ are isomorphic spans, from which it follows that $(\tilde{K} \rightarrow \tilde{H})\hat{\mathscr{R}}(\tilde{K} \rightarrow \tilde{H}')$.

$$\begin{tikzcd}
 & & F & K \arrow[l] \arrow[r] & H \\
G & J \arrow[l] \arrow[ur] \arrow[dr] & & & \\
 & & \tilde{F} \arrow[uu, "\wr"'] & \tilde{K} \arrow[l] \arrow[r] \arrow[uu, "\wr"'] & \tilde{H} \arrow[uu, "\wr"']
\end{tikzcd} \tag{14}$$

$$\begin{tikzcd}
 & & F & K \arrow[l] \arrow[r] & H' \\
G' & J \arrow[l] \arrow[ur] \arrow[dr] & & & \\
 & & \tilde{F} \arrow[uu, "\wr"'] & \tilde{K} \arrow[l] \arrow[r] \arrow[uu, "\wr"'] & \tilde{H}' \arrow[uu, "\wr"']
\end{tikzcd} \tag{15}$$

A different way of working 'up to isomorphism' would be to work directly on an abstract transition system, which can be obtained from the concrete transitions system used in this paper by quotienting by isomorphism (as is done in Sassone and Sobociński (2005)).

A second proof technique limits the transitions of a graph G by avoiding partial matches that are included in the intersection of J (the interface of G) and I (the interface of the production). This situation is depicted schematically in Figure 11 in Appendix B. We first define a new transition relation $\rightarrow_d$ that mirrors this intuition.

Definition 5.4. Let $(J \rightarrow G) \overset{J \rightarrow F \leftarrow K}{\longrightarrow} (K \rightarrow H)$ be a transition of $J \rightarrow G$. We say that the transition is *independent* whenever we can add two injective morphisms $D \rightarrow I$ and $D \rightarrow J$ to the diagram in Definition 3.1 such that the diagram commutes, that is, $D \rightarrow I \rightarrow L = D \rightarrow L$ and $D \rightarrow J \rightarrow G = D \rightarrow G$ (see Diagram (16)). We write

$(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow_d} (K \to H)$ if the transition is not independent, in which case, the transition will be called *dependent*.

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[rr, bend left] \arrow[dd, bend right] & L \arrow[d] & I \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\
G \arrow[r] & G^+ & C \arrow[l] \arrow[r] & H \\
J \arrow[r] \arrow[u] & F \arrow[u] & K \arrow[l] \arrow[u] \arrow[ur, dotted] &
\end{tikzcd} \tag{16}$$

Let $\mathscr{R}$, $\mathscr{S}$ be two relations as in Definition 5.1. We say that $\mathscr{R}$ d-progresses to $\mathscr{S}$, abbreviated by $\mathscr{R} \rightarrowtail_d \mathscr{S}$, if whenever $(J \to G)\,\mathscr{R}\,(J \to G')$ and $(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow_d} (K \to H)$, there exists a morphism $K \to H'$ such that[†] $(J \to G') \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H')$ and $(K \to H)\,\mathscr{S}\,(K \to H')$.

We can now use the new relation $\rightarrowtail_d$ as the basis of a new proof technique.

Proposition 5.5.

(i) Let $\mathscr{R}$ be a relation with $(J \to G)\,\mathscr{R}\,(J \to G')$. Given an independent transition $(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H)$, there is an independent transition $(J \to G') \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H')$ via the same production and context such that $(K \to H)\,\hat{\mathscr{R}}\,(K \to H')$.

(ii) Let $\mathscr{R}$ be symmetric and $\mathscr{R} \rightarrowtail_d \hat{\mathscr{R}}$. This implies that $\mathscr{R}$ is contained in $\sim$.

Proof.

(i) We assume that $(J \to G)\,\mathscr{R}\,(J \to G')$ and $(J \to G) \overset{J\to F\leftarrow K}{\longrightarrow} (K \to H)$ is an independent transition (see Definition 5.4). That is, there are morphisms $D \to J$ and $D \to I$ such that Diagram (16) above commutes.

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[rr, bend left] \arrow[dd, bend right] & L \arrow[d] \arrow[dd, bend right] & I \arrow[l] \arrow[r] \arrow[d] \arrow[dd, bend right] & R \arrow[d] \\
G \arrow[r] \arrow[rr, bend left] & G^+ & C \arrow[l] \arrow[r] & H \\
J \arrow[r] \arrow[u] \arrow[rr, bend left] & F \arrow[u] & K \arrow[l] \arrow[u] &
\end{tikzcd} \tag{17}$$

We continue the proof by constructing the morphisms in Diagram (17). The morphisms $L \to F$ and $G \to C$ are constructed according to Property A.7.

We now show how to construct the two remaining morphisms $I \to K$ and $J \to K$. In Diagram (18) the right square is a pullback and the outer square is a pushout as well as a pullback. This gives us $I \to K$ as an induced morphism. We can then infer from Property A.4 that the left as well as the right square are pushouts. Analogously,

[†] Note that $J \to G'$ may answer with an independent transition.

we can construct a morphism $J \to K$ in Diagram (19), and it follows that the left and right squares are both pushouts.

$$\begin{tikzcd}
L \arrow[r] & F \arrow[r] & G^+ \arrow[d] \\
I \arrow[u] \arrow[r, dotted] \arrow[rr, bend right] & K \arrow[u] \arrow[r] & C
\end{tikzcd} \tag{18}$$

$$\begin{tikzcd}
G \arrow[r] & C \arrow[r] & G^+ \arrow[d] \\
J \arrow[u] \arrow[r, dotted] \arrow[rr, bend right] & K \arrow[u] \arrow[r] & F
\end{tikzcd} \tag{19}$$

We now construct F' in Diagram (20) as the pushout of $I \to K$ and $I \to R$, giving us $F' \to H$ as an induced morphism. From classical pushout decomposition, we can infer that the lower right square is also a pushout. Hence we can conclude that $K \to H$ can be obtained by composing $J \to G$ with the context $J \to F' \leftarrow K$ – (see Diagram (21)).

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[dd, bend right] \arrow[rr, bend left] & L \arrow[d] \arrow[dd, bend right] & I \arrow[l] \arrow[r] \arrow[d] \arrow[dd, bend right] & R \arrow[d] \arrow[dd, bend right] \\
G \arrow[r] \arrow[rr, bend left] & G^+ & C \arrow[l] \arrow[r] & H \\
J \arrow[u] \arrow[r] \arrow[rr, bend left] & F \arrow[u] & K \arrow[u] \arrow[l] \arrow[r] & F' \arrow[u]
\end{tikzcd} \tag{20}$$

$$\begin{tikzcd}
J \arrow[r] \arrow[d] & K \arrow[r] \arrow[d] & F' \arrow[d] & K \arrow[l] \\
G \arrow[r] & C \arrow[r] & H &
\end{tikzcd} \tag{21}$$

We now derive a corresponding transition of $J \to G'$ using the same rewriting rule. At this point, the morphisms shown in Diagram (22) are known to exist, where $D \to G'$ is the composition of $D \to J$ and $J \to G'$. An alternative representation of this diagram is given in Diagram (23).

$$\begin{tikzcd}
D \arrow[r] \arrow[d, dotted] \arrow[dd, bend right] \arrow[rr, bend left] & L \arrow[dd, bend right] & I \arrow[l] \arrow[r] \arrow[dd, bend right] & R \arrow[dd, bend right] \\
G' & & & \\
J \arrow[u] \arrow[r] \arrow[rr, bend left] & F & K \arrow[l] \arrow[r] & F'
\end{tikzcd} \tag{22}$$

$$\begin{tikzcd}
D \arrow[r] \arrow[d] & I \arrow[r] \arrow[d] \arrow[rr, bend left] & L \arrow[d] & R \arrow[d] \\
J \arrow[r] \arrow[d] & K \arrow[r] \arrow[rr, bend left] & F & F' \\
G' & & &
\end{tikzcd} \tag{23}$$

We complete Diagram (23) by constructing C' as the pushout of $J \to K$ and $J \to G'$, followed by the construction of G'^+ as the pushout of $K \to C'$ and $K \to F$. We then construct H' as the pushout of $K \to C'$ and $K \to F'$. From classical pushout composition properties, it follows that all (basic and non-basic squares in Diagram (24)) are pushouts. Rearranging this diagram gives us Diagram (25), where again all squares are pushouts. Hence the square consisting of K, F, C', G'^{+} is also a pullback (see Property A.2).

$$\begin{tikzcd}
D \arrow[r] \arrow[d] & I \arrow[r] \arrow[d] \arrow[rr, bend left] & L \arrow[d] & R \arrow[d] \\
J \arrow[r] \arrow[d] & K \arrow[r] \arrow[d] \arrow[rr, bend left] & F \arrow[d] & F' \arrow[d] \\
G' \arrow[r] & C' \arrow[r] \arrow[rr, bend left] & G'^{+} & H'
\end{tikzcd} \tag{24}$$

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[dd, bend right] \arrow[rr, bend left] & L \arrow[d] \arrow[dd, bend right] & I \arrow[l] \arrow[r] \arrow[d] \arrow[dd, bend right] & R \arrow[d] \arrow[dd, bend right] \\
G' \arrow[r] \arrow[rr, bend left] & G'^{+} & C' \arrow[l] \arrow[r] & H' \\
J \arrow[u] \arrow[r] \arrow[rr, bend left] & F \arrow[u] & K \arrow[u] \arrow[l] \arrow[r] & F' \arrow[u]
\end{tikzcd} \tag{25}$$

Hence we conclude from the definition of rewriting with borrowed contexts that $(J \to G') \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H')$, and, furthermore, $K \to H'$ can be obtained by composing $J \to G'$ with the context $J \to F' \leftarrow K$, as shown in Diagram (26) This implies $(K \to H)\, \hat{\mathscr{R}}\, (K \to H')$.

$$\begin{tikzcd}
J \arrow[r] \arrow[d] & K \arrow[r] \arrow[d] & F' \arrow[d] & K \arrow[l] \\
G' \arrow[r] & C' \arrow[r] & H' &
\end{tikzcd} \tag{26}$$

(ii) We first assume that $\mathscr{R} \rightarrowtail_d \mathscr{S}$ and show that this implies $\mathscr{R} \rightarrowtail \mathscr{S} \cup \hat{\mathscr{R}}$. Let $(J \to G)\, \mathscr{R}\, (J \to G')$ and $(J \to G) \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H)$. If this transition is dependent, then it follows from the definition of $\rightarrowtail_d$ that $(J \to G') \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H')$ and $(K \to H)\, \mathscr{S}\, (K \to H')$. If the transition is independent, we have shown in the first part of this proposition that $(J \to G') \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H')$ and $(K \to H)\, \hat{\mathscr{R}}\, (K \to H')$. Combining these, we obtain $\mathscr{R} \rightarrowtail \mathscr{S} \cup \hat{\mathscr{R}}$.
Hence $\mathscr{R} \rightarrowtail_d \hat{\mathscr{R}}$ implies that $\mathscr{R} \rightarrowtail \hat{\mathscr{R}} \cup \hat{\mathscr{R}} = \hat{\mathscr{R}}$, and thus $\mathscr{R}$ is a bisimulation up to context and is contained in $\sim$. □

6. Borrowed contexts at work: an example

We now show how to exploit this proof technique and prove that two graphs are bisimilar. We assume that the set of rules $\mathscr{P}$ shown in Figure 1 is given and consider the two graphs with the interfaces shown in Figure 3.

We consider the symmetric relation

$$\mathscr{R} = \{(J \to G, J \to G'), (J \to G', J \to G)\}$$

and will show that it is contained in $\sim$ using Proposition 5.5. For each of the three rules there are several partial matches for both G and G'. Most of these matches lead to

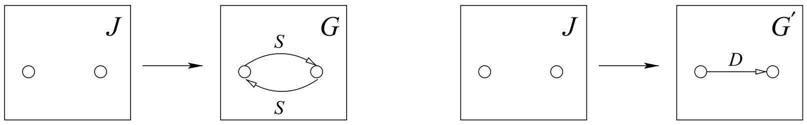

Fig. 3. Two graphs with interfaces that are bisimilar.

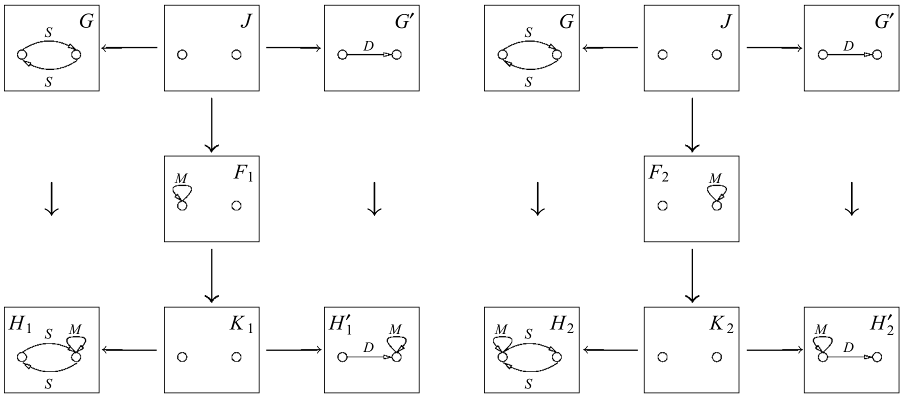

Fig. 4. Matching transitions of bisimilar graphs.

independent transitions in the sense of Definition 5.4, since the graph to be rewritten and the left-hand side overlap only in their interfaces.

We check the only two dependent transitions of $J \to G$, where both are instances of the [simplex] rule. These two transitions will be written

$$(J \to G) \xrightarrow{J \to F_i \leftarrow K_i} (K_i \to H_i),$$

where $i = 1, 2$. The graphs F_i, K_i, H_i and the corresponding morphisms are depicted in Figure 4. Note that we have already shown how to derive the transition of $J \to G$ for $i = 1$ in Figure 2 (where $F_1 = F$, $K_1 = K$, $H_1 = H$).

In order to show that $\mathscr{R}$ is a bisimulation up to context, we have to find matching transitions

$$(J \to G') \xrightarrow{J \to F_i \leftarrow K_i} (K_i \to H'_i)$$

for $i = 1, 2$, such that $(K_i \to H_i)\hat{\mathscr{R}}(K_i \to H'_i)$. Such transitions can be derived, and the graphs H'_i with their corresponding morphisms are also depicted in Figure 4. Note that the transition for $i = 1$ is an instance of the [duplex-1] rule (its derivation is depicted in Figure 5), while the transition for $i = 2$ is an instance of the [duplex-2] rule.

Furthermore, we have $(K_i \to H_i)\hat{\mathscr{R}}(K_i \to H'_i)$ for $i = 1, 2$, since these graphs can be obtained by composing $J \to G$ and $J \to G'$, respectively, with a context consisting of two nodes and a looping M-edge. After checking the two dependent transitions of $J \to G'$, we can conclude $(J \to G) \sim (J \to G')$ using Proposition 5.5.

This means that in every context we can replace a duplex connection by two simplex connections, and *vice versa*, without changing the behaviour of the overall system. Even this small example shows us that in order to obtain a bisimilarity result, proof techniques

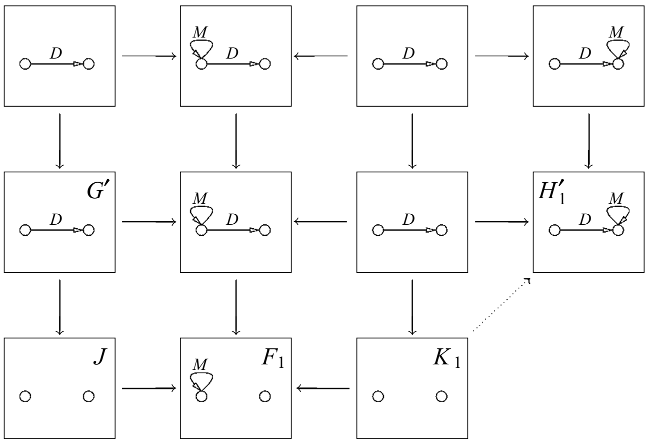

Fig. 5. Graphical representation of the derivation of the matching transition of $J \to G'$.

are needed in order to keep $\mathscr{R}$ finite. Otherwise we would have to deal with a relation containing infinitely many pairs.

7. Connection to relative pushouts

A question that arises naturally is whether our construction is connected to the relative pushouts (RPOs) of Leifer and Milner (Leifer and Milner 2000) or to the GRPOs of Sassone and Sobociński (Sassone and Sobociński 2003). In order to give a simplified answer to this question[†], we consider the category of cospans, the objects of which are graphs and the arrows of which are injective cospans (or contexts) over the category **Graph** (see Definition 2.3). Two cospans $J_1 \to G \leftarrow J_2$ and $J_2 \to G' \leftarrow J_3$ are composed using the following pushout construction, which results in a new cospan $J_1 \to \overline{G} \leftarrow J_3$.

$$\begin{tikzcd}
 & & J_2 \arrow[dl] \arrow[dr] & & \\
J_1 \arrow[r] & G \arrow[dr] & \mathrm{PO} & G' \arrow[dl] & J_3 \arrow[l] \\
 & & \overline{G} & &
\end{tikzcd}$$

Since the pushout is defined only up to isomorphism, we also consider the middle graph of a cospan up to isomorphism. The outer graphs are regarded as concrete graphs. This notion of ‘up to isomorphism’ creates some problems, which are explained below. The alternative, which is pursued in Sassone and Sobociński (2003) and Sassone and Sobociński (2004), would be to fix a choice of pushout. Then it turns out that associativity only holds up to isomorphism, which means that we are actually working inside a bicategory instead of a category.

[†] A more formal treatment can be found in Sassone and Sobociński (2004).

Before we continue, we will first define the notion of a relative pushout (Leifer 2001; Leifer and Milner 2000).

Definition 7.1 (Relative pushout). Consider, in any category **C**, a commuting square as shown in Diagram (27) such that $f_0;g_0 = f_1;g_1$. A *relative pushout (RPO)* for this commuting square is a triple h_0, h_1, h satisfying the following two properties:

(i) *Commutation:* $f_0;h_0 = f_1;h_1$ and $h_i;h = g_i$ for $i = 0, 1$ (see Diagram (28));
(ii) *Universality:* for any h'_0, h'_1, h' satisfying $f_0;h'_0 = f_1;h'_1$ and $h'_i;h' = g_i$ there exists a unique mediating arrow k such that $k;h' = h$ and $h_i;k = h'_i$ for $i = 0, 1$ (see Diagram (29)).

$$\begin{tikzcd}
\cdot \arrow[r, "f_0"] \arrow[d, "f_1"'] & \cdot \arrow[d, "g_0"] \\
\cdot \arrow[r, "g_1"'] & \cdot
\end{tikzcd} \tag{27}$$

$$\begin{tikzcd}
\cdot \arrow[rr, "f_0"] \arrow[dd, "f_1"'] & & \cdot \arrow[dd, "g_0"] \arrow[dl, "h_0"'] \\
& \cdot \arrow[dr, "h"] & \\
\cdot \arrow[ur, "h_1"] \arrow[rr, "g_1"'] & & \cdot
\end{tikzcd} \tag{28}$$

$$\begin{tikzcd}
\cdot \arrow[rrr, "f_0"] \arrow[dd, "f_1"'] & & & \cdot \arrow[dd, "g_0"] \arrow[dll, "h_0"'] \arrow[dl, "h'_0"] \\
& \cdot \arrow[r, dotted, "k"] \arrow[drr, "h"] & \cdot \arrow[dr, "h'"] & \\
\cdot \arrow[ur, "h_1"] \arrow[urr, "h'_1"'] \arrow[rrr, "g_1"'] & & & \cdot
\end{tikzcd} \tag{29}$$

A commuting square as in Diagram (27) is an *idem pushout (IPO)* if the triple g_0, g_1, *id* is a relative pushout for the same Diagram (27).

It is shown in Leifer (2001) that the square consisting of f_0, f_1, h_0, h_1 in Diagram (28) is an IPO whenever the triple h_0, h_1, h is an RPO.

In the work of Leifer and Milner, relative pushouts are used to formalise the notion of a 'minimal context', where morphisms in their framework are called arrows. Let (l, r) with $l, r\colon 0 \to m$ be a reaction rule consisting of two arrows, the source of which is a distinguished object 0, and let $a\colon 0 \to m'$ be another arrow. The task is to find arrows $D\colon m \to k$, $F\colon m' \to k$ such that $l;D = a;F$ and the commuting square is an IPO, that is, D and F are in some sense minimal arrows having this property. In this case there is a transition $a \xrightarrow{F} r;D$ (see Diagram (30)).

$$\begin{tikzcd}
\cdot \arrow[r, "l"] \arrow[d, "a"'] \arrow[dr, phantom, "\text{IPO}"] & \cdot \arrow[d, "D"] & \cdot \arrow[l, "r"'] \arrow[dl, "r;D"] \\
\cdot \arrow[r, "F"'] & \cdot &
\end{tikzcd} \tag{30}$$

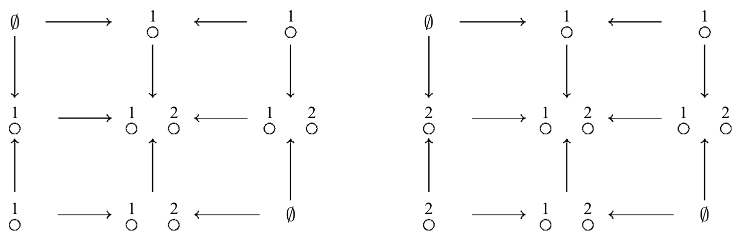

Fig. 6. Different diagrams for a graph with non-trivial automorphisms.

We will make a simplified attempt at constructing a relative pushout in the category of injective cospans over **Graph**, highlight connections to the proof of Theorem 4.3 and show the relationship to rewriting with borrowed contexts. The attempt is simplified in its handling of injectivity and isomorphisms. For the general case, see Sassone and Sobociński (2004).

Reaction rules (l, r) in our setting are pairs of injective cospans of the form $(\emptyset \to L \leftarrow I, \emptyset \to R \leftarrow I)$, thus a commuting square $l; D = a; F$ (which is not necessarily a relative pushout) corresponds to a square of injective cospans as shown in Diagram (31), where $l; D$ corresponds to $(\emptyset \to L \leftarrow I); (I \to \overline{C} \leftarrow \overline{K})$ and $a; F$ to $(\emptyset \to G \leftarrow J); (J \to \overline{F} \leftarrow \overline{K})$.

$$\begin{tikzcd}
\emptyset \arrow[r] \arrow[d] & L & I \arrow[l] \arrow[d] \\
G & & \overline{C} \\
J \arrow[u] \arrow[r] & \overline{F} & \overline{K} \arrow[l] \arrow[u]
\end{tikzcd} \tag{31}$$

$$\begin{tikzcd}
\emptyset \arrow[r] \arrow[d] & L \arrow[d] & I \arrow[l] \arrow[d] \\
G \arrow[r] & \overline{G}^{+} & \overline{C} \arrow[l] \\
J \arrow[u] \arrow[r] & \overline{F} \arrow[u] & \overline{K} \arrow[l] \arrow[u]
\end{tikzcd} \tag{32}$$

Since the two compositions of cospans coincide, there must be a graph $\overline{G}^{+}$ such that Diagram (32) commutes and the lower left and the upper right squares are pushouts. Although the graph $\overline{G}^{+}$ is unique up to isomorphism, Diagram (32) is not necessarily unique, not even up to isomorphism. This can be the result of non-trivial automorphisms of $\overline{G}^{+}$. The outcome of the following construction of a relative pushout depends on the specific choice of diagram.

An example for this situation is given in Figure 6, where the morphisms are indicated by numbered nodes. Note that in both diagrams the cospans and their compositions are isomorphic, but still, the two diagrams differ significantly and would lead to different outcomes of the following construction. In fact, the upper left square in the second diagram is a pushout, which is not the case in the first diagram. But in both diagrams all squares commute and the lower left and upper right squares are pushouts. A similarly problematic example is given in Leifer (2001, pages 80–81).

We therefore assume for simplicity in the following that the automorphism group of $\overline{G}^+$ consists of only one element (the identity), and thus that Diagram (32) is uniquely determined. Apart from this simplification, there are, in principle, two ways to force uniqueness: one can either add support to the category (Leifer 2001; Jensen and Milner 2003), that is, every graph item should have a unique unchangeable name; or one can work in a 2-categorical setting (Sassone and Sobociński 2004; Sassone and Sobociński 2003), where commuting cubes such as the one in Diagram (31) are additionally provided with an isomorphism giving a 'proof of structural congruence'. We will pursue the simplified case.

In the following, we construct Diagram (33) from Diagram (32) in several steps. First we construct D as the pullback of $L \to \overline{G}^+$ and $G \to \overline{G}^+$, and then construct G^+ as the pushout of $D \to G$ and $D \to L$. This gives us $G^+ \to \overline{G}^+$ as an induced morphism. Next we can split the two pushouts (the lower left and upper right squares) along G^+ by Property A.3, and obtain the graphs F and C. This is followed by the construction of K as the pullback of $F \to G^+$ and $C \to G^+$, and, finally, E as the pullback of $\overline{F} \to \overline{G}^+$ and $\overline{C} \to \overline{G}^+$. The morphisms $K \to E$ and $\overline{K} \to E$ are mediating morphisms for the second pullback.

$$\begin{tikzcd}
\emptyset \arrow[drrr] \arrow[dddr] & & & & & & \\
& D \arrow[rr] \arrow[dd] & & L \arrow[dl] \arrow[dd] & & I \arrow[ll] \arrow[dl] \arrow[dd] & \\
& & G^+ \arrow[dr] & & C \arrow[ll] \arrow[dr] & & \\
& G \arrow[ur] \arrow[rr] & & \overline{G}^+ & & \overline{C} \arrow[ll] & \\
& & F \arrow[uu] \arrow[dr] & & K \arrow[ll] \arrow[uu] \arrow[dr] & & \\
& J \arrow[uu] \arrow[ur] \arrow[rr] & & \overline{F} \arrow[uu] & & E \arrow[ll] \arrow[uu] & \\
& & & & & & \overline{K} \arrow[ulll] \arrow[ul] \arrow[uuul]
\end{tikzcd} \tag{33}$$

This construction resembles, on the one hand, some parts of the proof of Theorem 4.3 (see right cube in Diagram (7)). On the other hand, it corresponds exactly to the construction of groupoidal relative pushouts (GRPOs) in Sassone and Sobociński (2004) for the special case in which the isomorphism $\alpha\colon G+_J\overline{F} \xrightarrow{\sim} L+_I\overline{C}$ is the identity, as assumed above, where $\overline{G}^+ = G+_J\overline{F} = L+_I\overline{C}$. This means that the triple $I \to C \leftarrow K, J \to F \leftarrow K, K \to E \leftarrow \overline{K}$ of cospans constructed above and shown in Diagram (34) is a GRPO of the cospans in (31). In fact, it can be deduced from the construction in (33) that all the subdiagrams of (34) commute in the category of cospans, and it follows from the general proof in Sassone and Sobociński (2004) that (34) defines the relative pushout of (31).

$$\begin{tikzcd}
\emptyset \arrow[rr] \arrow[dd] & & L & & I \arrow[ll] \arrow[dl] \arrow[dd] \\
& & & C & \\
G & & K \arrow[ur] \arrow[dl] \arrow[dr] & & \overline{C} \\
& F & & E & \\
J \arrow[uu] \arrow[ur] \arrow[rr] & & \overline{F} & & \overline{K} \arrow[ll] \arrow[ul] \arrow[uu]
\end{tikzcd} \qquad (34)$$

Moreover, it is shown in Sassone and Sobociński (2004) that the construction of rewriting steps based on their GRPOs corresponds exactly to rewriting with borrowed contexts in our sense (see Definition 3.1). In fact, from the construction in Diagram (33), we obtain diagram (35), where squares (1), (2) and (3) have been constructed directly or by splitting as pushouts, and square (4) has been constructed as a pullback. Moreover, square (5) in Diagram (35) is a pushout according to the construction of the derived graph H in both approaches. Hence squares (1)–(5) define a rewrite step with borrowed contexts in our sense (see Definition 3.1).

$$\begin{tikzcd}
D \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & L \arrow[d] \arrow[dr, phantom, "(2)"] & I \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(5)"] & R \arrow[d] \\
G \arrow[r] \arrow[dr, phantom, "(3)"] & G^{+} \arrow[dr, phantom, "(4)"] & C \arrow[l] \arrow[r] & H \\
J \arrow[r] \arrow[u] & F \arrow[u] & K \arrow[l] \arrow[u] \arrow[ur, dotted] &
\end{tikzcd} \qquad (35)$$

8. Conclusion

We have presented a way to derive labelled transitions and bisimulation congruences for graph transformation systems. It is our hope that this work will be helpful for the transfer of concepts from the world of process algebras to the world of graph rewriting and *vice versa*. We believe that having graphs as objects (and graph morphisms as arrows) instead of having graphs as arrows is useful for tracking graph components, which enables us to state easily which components are associated with each other in different graphs.

The line of research this paper is concerned with has generated a lot of interest and contributions in the last few years, ranging from bigraphs over (G)RPOs to the work on borrowed contexts presented in this paper. We have tried to point out differences and similarities throughout the paper. Summarising, we would like to say that in some sense all these contributions are different views on the same subject matter. As has been shown in Sassone and Sobociński (2004), these approaches are not only related in spirit, but there is a clear and formal way to show their equivalence. One of our main contributions is to provide a simple and straightforward way to derive labels, using only some pushouts and one pullback (see Definition 3.1). This definition is definitely simpler and more accessible than the ones in other approaches, and lends itself better to the actual use in proofs or in mechanised verification.

Using only basic categorical concepts, our proof that bisimilarity is a congruence is slightly longer than other proofs. But taking into account that in the other approaches (Leifer and Milner 2000; Sassone and Sobociński 2004) an additional difficult proof obligation is the construction and universal property of relative pushouts and GRPOs, respectively, our proof is acutally shorter and easier to verify. In any case, our work can profit from the fact that our construction has been shown to have the universal property of a relative pushout. One can thus transfer other congruence results directly, for instance, the results for trace and failures equivalences (Leifer 2001) and for weak bisimilarity (Jensen to appear).

Future work aims at a closer integration of the ideas presented in this paper with the rich theory of graph transformation systems. In this way we hope to establish an even closer correspondence between the world of process calculi and the world of graph rewriting.

For instance, we have obtained compositionality results for DPO transformations with borrowed context (Baldan *et al.* 2006). These results can be seen as a first step towards a structural operational semantics for adhesive rewriting systems, that is, towards a framework where the transition system associated with a graph transformation system can be defined inductively in SOS style.

Appendix A. Pushout-pullback properties

Definition 3 and the proof of Theorem 4.3 require several properties to hold in the category we are working in. All these conditions are stated below, and are satisfied by the category **Graph** of graph structures and graph structure morphisms for a given signature. Naturally, the result holds also for any other category satisfying these conditions, for instance, for categories where pushouts and pullbacks are constructed componentwise in the category **Set**.

More precisely, the results hold for all adhesive categories as introduced by Lack and Sobociński (Lack and Sobociński 2004; Lack and Sobociński 2005). In fact, all the following properties A.1–A.8 are valid in adhesive categories, provided ‘injective morphisms’ in A.1–A.8 is replaced by ‘monomorphisms’. The essential point of adhesive categories is the requirement that all pushouts along monomorphisms, where at least one of the given morphisms is a monomorphism, are van Kampen squares, or VK-squares for short. Property A.6 below corresponds roughly to one direction of the VK-square property. Intuitively, this part means that pushouts along monomorphisms are closed under pullbacks. More precisely, given the cube in diagram (40) where the front square is a pushout along a monomorphism, the VK-square property states that if the right and top squares in diagram (40) are pullbacks, then the back square is a pushout if and only if the left and bottom squares are pullbacks.

In the following, we state the properties A.1–A.8 without proofs – explicit proofs for most of A.1–A.8 can be found in Lack and Sobociński (2004; 2005); A.5 is shown in a separate note by Sobociński (Sobociński 2004); and the remaining properties are easy consequences of the others.

A.1. *Injectivity conditions*

The two morphisms generated by a pushout, pullback or pushout complement of injective morphisms are always injective. Furthermore, given two injective morphisms $G_2 \to G_4$, $G_3 \to G_4$, if we construct G_1 as the pullback, and then construct G_4' as the pushout of $G_1 \to G_2$, $G_1 \to G_3$ as shown in Diagram (36), we get that all morphisms, including the unique mediating morphism $G_4' \to G_4$, are injective.

$$\begin{tikzcd}
 & G_2 \arrow[dr] \arrow[drr, bend left=10] & & \\
G_1 \arrow[ur] \arrow[dr] & & G_4' \arrow[r, dotted] & G_4 \\
 & G_3 \arrow[ur] \arrow[urr, bend right=10] & &
\end{tikzcd} \tag{36}$$

A.2. *Injective pushouts are pullbacks*

A pushout that consists of four injective morphisms is also a pullback.

A.3. *Pushout complement splitting*

If the square in Diagram (37) below is a pushout and all morphisms are injective, then there exists a graph G_2 splitting the morphism $G_1 \to G_3$ and a morphism $G_2 \to G_5$ such that Diagram (38) consists of two pushouts and all morphisms are injective.

$$\begin{tikzcd}
G_1 \arrow[rr] \arrow[d] & & G_3 \arrow[d] \\
G_4 \arrow[r] & G_5 \arrow[r] & G_6
\end{tikzcd} \tag{37}$$

$$\begin{tikzcd}
G_1 \arrow[r] \arrow[d] & G_2 \arrow[r] \arrow[d] & G_3 \arrow[d] \\
G_4 \arrow[r] & G_5 \arrow[r] & G_6
\end{tikzcd} \tag{38}$$

Remark: Whenever the square in Diagram (37) is a pullback, the square splits into two pullbacks (as shown in Diagram (38)) in every category without the requirement of injectivity (classical pullback splitting). And if the square in Diagram (39) is a pushout, it splits into two pushouts in every category (classical pushout splitting).

$$\begin{tikzcd}
G_1 \arrow[r] \arrow[d] & G_2 \arrow[r] & G_3 \arrow[d] \\
G_4 \arrow[rr] & & G_6
\end{tikzcd} \tag{39}$$

A.4. *Special pushout-pullback decomposition*

In Diagram (38) above consisting of injective morphisms, whenever the outer square (consisting of G_1, G_3, G_4, G_6) is a pushout and the right square is a pullback, the left and right squares are both pushouts.

A.5. *Special pullback-pushout decomposition*

In Diagram (38) above consisting of injective morphisms, whenever the the outer square is a pullback and the left square is a pushout, the left and right squares are both pullbacks.

Remark: Compare this with the classical decomposition properties. Whenever the outer and left squares are pushouts, the right square is a pushout. Furthermore, whenever the outer and right squares are pullbacks, the left square is a pullback.

A.6. *Cube pushout-pullback property*

Given a commutative cube consisting of injective morphisms as shown in Diagram (40) below, where the bottom, left, right and top squares are pullbacks, whenever the front square is a pushout, the back square is a pushout as well.

$$\begin{tikzcd}
G_2 \arrow[dd] \arrow[dr] & & G_1 \arrow[ll] \arrow[dd] \arrow[dr] & \\
& G_4 \arrow[dd] & & G_3 \arrow[ll] \arrow[dd] \\
G_6 \arrow[dr] & & G_7 \arrow[ll] \arrow[dr] & \\
& G_8 & & G_9 \arrow[ll]
\end{tikzcd} \tag{40}$$

A.7. *Special pushout decomposition*

If all morphisms in Diagram (41) below are injective and the left and right squares are both pushouts and there exists a morphism $G_1 \to G_3$, then there exists a morphism $G_4 \to G_6$ such that the diagram commutes and the outer square is a pushout. In order to prove this property, we require Property A.8.

$$\begin{tikzcd}
G_1 \arrow[r] \arrow[d] \arrow[rr, bend left] & G_2 \arrow[d] & G_3 \arrow[l] \arrow[d] \\
G_4 \arrow[r] \arrow[rr, dotted, bend right] & G_5 & G_6 \arrow[l]
\end{tikzcd} \tag{41}$$

A.8. *Uniqueness of pushout complements*

Let $G_1 \to G_2$ and $G_2 \to G_4$ be two morphisms as shown in Diagram (42). Then there exists a unique graph G_3 and unique injective morphisms $G_1 \to G_3$, $G_3 \to G_4$ such that the square in Diagram (43) is a pushout.

$$\begin{tikzcd}
G_1 \arrow[r] & G_2 \arrow[d] \\
& G_4
\end{tikzcd} \tag{42}$$

$$\begin{tikzcd}
G_1 \arrow[r] \arrow[d] & G_2 \arrow[d] \\
G_3 \arrow[r] & G_4
\end{tikzcd} \tag{43}$$

Remark: In the remark at the end of Section 4 we pointed out that there is a non-injective extension of Definition 3.1 and Theorem 4.3. In fact, this requires corresponding non-injective extensions of Conditions A.1–A.8, which have been shown already in Sassone and

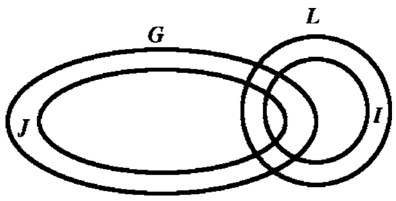

Fig. 7. Representation of a graph G overlapping with a left-hand side L.

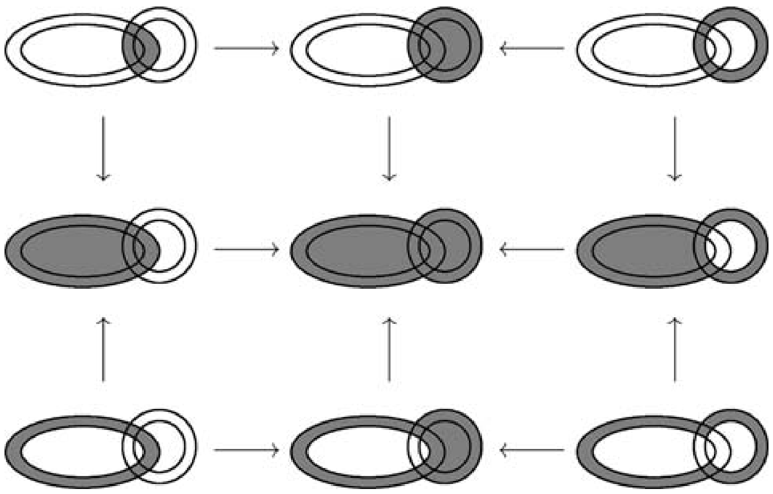

Fig. 8. Graphical represenation of rewriting with borrowed contexts with Venn-like diagrams.

Sobociński (2004) and Sobociński (2004), respectively. Moreover, it seems that generalising and proving the conditions for adhesive high-level replacement systems (Ehrig *et al.* 2004) accordingly is straightforward.

Appendix B. Graph representation by Venn diagrams

A graph G for which there exists an injective morphism $G \rightarrow G^+$ can be seen as a subobject of G^+. Since we are working with injective morphisms only, we can exploit this suboject relation and visualise diagrams by drawing graphs as Venn diagrams. Intuitively, this works since in an adhesive category the subobjects of an object form a distributive lattice.

We can illustrate Definition 3.1 schematically. The graph G that is to be rewritten and the left-hand side L including their interfaces J and I can both be represented as overlapping circles (see Figure 7 for a typical situation). Note that some areas might be empty. In Figure 8 we show a part of the Diagram of Definition 3.1, drawing graphs that are subojects of G^+ and ignoring the graphs at the far right, that is, the right-hand side R and the resulting graph H.

If we have a graph G in a context E and find a partial match of a left-hand side L, a schematical representation of such a situation can be drawn as in Figure 9. Diagram (7)

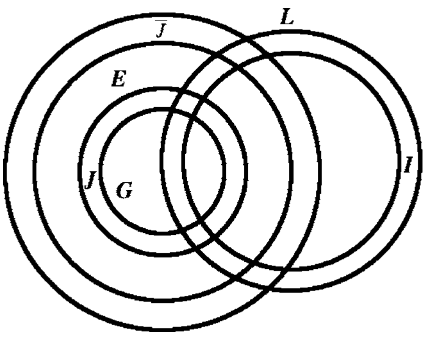

Fig. 9. Representation of a graph G with context E and overlapping with a left-hand side L.

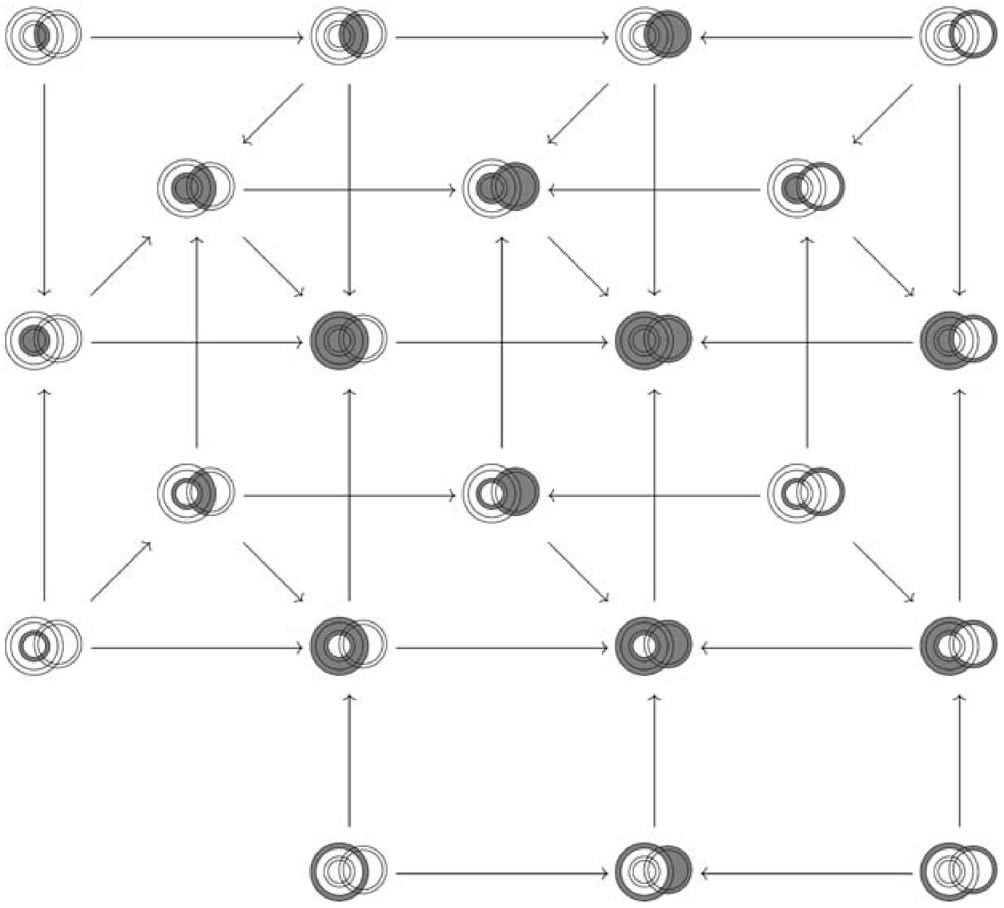

Fig. 10. Representing Diagram (7) in the proof of Theorem 4.3 with Venn diagrams.

in the proof of Theorem 4.3 can be represented with Venn diagrams as shown in Figure 10 using the schema of Figure 9. Again, we omit the graphs on the far right-hand side.

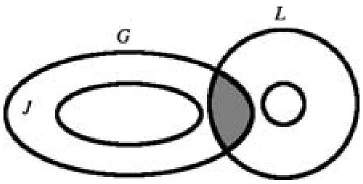

Fig. 11. The partial match is located in the overlap of the interfaces J and I.

The situation described in Definition 5.4 where the partial match occurs in the overlap of the interfaces J and I leading to an independent transition is shown in Figure 11.

Acknowledgements

We very grateful to Robin Milner, Vladimiro Sassone and Paweł Sobociński for sharing their ideas with us and for giving us valuable insights. We would also like to thank the anonymous referees for their helpful comments.

References

Baldan, P., Corradini, A. and Montanari, U. (2002) Bisimulation equivalences for graph grammars. In: Formal and Natural Computing – Essays Dedicated to Grzegorz Rozenberg. *Springer-Verlag Lecture Notes in Computer Science* **2300** 158–190.

Baldan, P., Ehrig, H. and König, B. (2006) Composition and decomposition of DPO transformations with borrowed context. In: *Proc. of ICGT '06* (to appear).

Bonchi, F., König, B. and Montanari, U. (2006) Saturated semantics for reactive systems. In: *Proc. of ICGT '06* (to appear).

Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Heckel, R. and Löwe, M. (1997) Algebraic approaches to graph transformation – part I: Basic concepts and double pushout approach. In: Rozenberg, G. (ed.) *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 1: Foundations*, Chapter 3, World Scientific.

Ehrig, H. (2002) Bigraphs meet double pushouts. *EATCS Bulletin* **78** 72–85.

Ehrig, H., Habel, A., Padberg, J. and Prange, U. (2004) Adhesive high-level replacement categories and systems. In: *Proc. of ICGT '04* (to appear).

Ehrig, H., Heckel, R., Korff, M., Löwe, M., Ribeiro, L., Wagner, A. and Corradini, A. (1997) Algebraic approaches to graph transformation – part II: Single pushout approach and comparison with double pushout approach. In: Rozenberg, G. (ed.) *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 1: Foundations*, Chapter 4, World Scientific.

Ehrig, H., Pfender, M. and Schneider, H. (1973) Graph grammars: An algebraic approach. In: *Proc. 14th IEEE Symp. on Switching and Automata Theory* 167–180.

Gadducci, F. and Heckel, R. (1997) An inductive view of graph transformation. In: Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT '97. *Springer-Verlag Lecture Notes in Computer Science* **1376** 223–237.

Gadducci, F. and Montanari, U. (2001) A concurrent graph semantics for mobile ambients. In: Brookes, S. and Mislove, M. (eds.) Proceedings of the 17th MFPS. *Electronic Notes in Computer Science* **45**.

Gadducci, F. and Montanari, U. (2002) Comparing logics for rewriting: Rewriting logic, action calculi and tile logic. *Theoretical Computer Science* **285** (2) 319–358.

Jensen, O. H. (to appear) *Bigraphs*, Ph.D. thesis, University of Aalborg.

Jensen, O. H. and Milner, R. (2003) Bigraphs and transitions. In: *Proc. of POPL 2003*, ACM 38–49.

König, B. (2000) A graph rewriting semantics for the polyadic π-calculus. In: *Workshop on Graph Transformation and Visual Modeling Techniques (Geneva, Switzerland), ICALP Workshops '00*, Carleton Scientific 451–458.

König, B. and Montanari, U. (2001) Observational equivalence for synchronized graph rewriting with mobility. In: Proc. of TACS '01. *Springer-Verlag Lecture Notes in Computer Science* **2215** 145–164.

Lack, S. and Sobociński, P. (2004) Adhesive categories. In: Proc. of FOSSACS '04. *Springer-Verlag Lecture Notes in Computer Science* (to appear).

Lack, S. and Sobociński, P. (2005) Adhesive and quasiadhesive categories. *RAIRO – Theoretical Informatics and Applications* **39** (3).

Leifer, J. J. (2001) *Operational congruences for reactive systems*, Ph.D. thesis, University of Cambridge Computer Laboratory.

Leifer, J. J. and Milner, R. (2000) Deriving bisimulation congruences for reactive systems. In: Proc. of CONCUR 2000. *Springer-Verlag Lecture Notes in Computer Science* **1877**.

Mac Lane, S. (1971) *Categories for the Working Mathematician*, Springer-Verlag.

Milner, R. (1999) *Communicating and Mobile Systems: the π-Calculus*, Cambridge University Press.

Milner, R. (2001) Bigraphical reactive systems. In: Proc. of CONCUR '01. *Springer-Verlag Lecture Notes in Computer Science* **2154** 16–35.

Milner, R. and Sangiorgi, D. (1992a) Barbed bisimulation. In: Proc. of ICALP '92. *Springer-Verlag Lecture Notes in Computer Science* **623**.

Milner, R. and Sangiorgi, D. (1992b) Techniques of weak bisimulation up-to. In: Proc. of CONCUR '92. *Springer-Verlag Lecture Notes in Computer Science* **630**.

Montanari, U. and Pistore, M. (1995) Concurrent semantics for the π-calculus. *Electronic Notes in Theoretical Computer Science* **1**.

Pierce, B. C. (1991) *Basic Category Theory for Computer Scientists (Foundations of Computing)*, MIT Press.

Rozenberg, G. (editor) (1997) *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 1: Foundations*, World Scientific.

Sangiorgi, D. (1995) On the proof method for bisimulation. In: Proc. of MFCS '95 (Mathematical Foundations of Computer Science). *Springer-Verlag Lecture Notes in Computer Science* **969** 479–488.

Sangiorgi, D. and Walker, D. (2001) *The π-calculus—A Theory of Mobile Processes*, Cambridge University Press.

Sassone, V. and Sobociński, P. (2003) Deriving bisimulation congruences: 2-categories vs precategories. In: Proc. of FoSSaCS 2003. *Springer-Verlag Lecture Notes in Computer Science* **2620** 409–424.

Sassone, V. and Sobociński, P. (2004) Congruences for contextual graph-rewriting. Technical Report RS-04-11, BRICS.

Sassone, V. and Sobociński, P. (2005) Reactive systems over cospans. In: *Proc. LICS '05*, IEEE 311–320.

Sewell, P. (2002) From rewrite rules to bisimulation congruences. *Theoretical Computer Science* **274** (1–2) 183–230.

Sobociński, P. (2004) Special PB-PO property in adhesive cats (note).