Deriving Bisimulation Congruences in the DPO Approach to Graph Rewriting (Long Version)*

Hartmut Ehrig[1] and Barbara König[2]

[1] Technische Universität Berlin, Germany
[2] Universität Stuttgart, Germany

ehrig@cs.tu-berlin.de koenigba@fmi.uni-stuttgart.de

Abstract. Motivated by recent work on the derivation of labelled transitions and bisimulation congruences from unlabelled reaction rules, we show how to solve this problem in the DPO (double-pushout) approach to graph rewriting. Unlike in previous approaches, we consider graphs as objects, instead of 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 smoothly integrates 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 sketch that the bisimilarity based on rewriting with borrowed contexts is a congruence relation.

1 Introduction

In the last few 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 [25]. 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 us to replace a subsystem with an equivalent one without changing the behaviour of the overall system and futhermore helps to 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 (see [20]) 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 technical report is the full version of [7]. Research partially supported by the BMBF project DACHIA, the TMR network SEGRAVIS and EPSRC grant R93346/01.

So the idea which was formulated in the papers of Leifer/Milner [15, 16], Sewell [28] and Sassone/Sobocinski [26] is to automatically derive a labelled transition system such that the resulting bisimilarity is a congruence. A central concept of this approach is to formalize the notion of minimal context which 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 one. This yields the labelled transition

$$a.P \stackrel{\_|\bar{a}.Q}{\rightarrow} P \mid Q,$$

saying that $a.P$ put into this contexts reacts and reduces to $P \mid Q$. Using all possible contexts as labels would also result in a bisimulation congruence, but we do not gain anything compared to quantification over all contexts.

In [15, 16] the notion of "minimal context" is formalized as the categorical concept of relative pushout respectively idem pushout. This notion has also been applied to bigraphs [12]. However, the theory is complicated by the fact that one can not 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 and to either work in a precategory or to construct a suitable category starting from such a precategory. Another approach, given by Sassone and Sobocinski [26], is to work with cells inside a 2-category.

It is our aim to achieve similar results in the context of graph rewriting [23], a framework which allows to model dynamic and concurrent systems consisting of interconnected components in a natural and intuitive way. Many process calculi such as the π-calculus [11, 22, 13] and the ambient calculus [10] can be translated into graph rewriting. We are specifically interested in the double-pushout (DPO) approach [3], one of the standard approaches to graph rewriting. So far, there is not yet a uniform theory of bisimulation for graph transformation systems. Previous work on this topic can be found in[1]. Using the concepts explained earlier would be possible in theory, but contradicts the philosophy behind graph rewriting where graphs are considered only up to isomorphism. Furthermore, deriving labels via relative pushouts is entirely non-trivial and can be rather complicated.

The approach which is presented in this paper is motivated by the work of Leifer/Milner and other contributions to this area, but does not directly rely on their theory. Instead we present a very simple way of deriving minimal contexts—we call them borrowed contexts—which smoothly integrates with the DPO approach and which has a very constructive nature. The only categorical concepts that are needed are standard 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 instead are graph morphisms which provide the necessary tracking information for nodes and edges which, in the case of graphs as arrows, can only be provided by adding support to a category. This work is based on ideas presented in [4], a paper which points out similarities and differences between Milner's bigraphs [12, 19] and the DPO approach to graph rewriting.

Our main result states that bisimilarity defined on graph rewriting with borrowed contexts is indeed a congruence relation (see Theorem 7).

The paper is structured as follows: In Section 2 we will give a short introduction to the DPO approach, followed by the definition of rewriting with borrowed contexts (Section 3). Section 4 provides the proof ideas that the resulting bisimilarity is a congruence. After having introduced a proof technique we continue with an example showing borrowed contexts at work in Section 5. We conclude with a comparison of our approach to the relative pushouts of Leifer and Milner in Section 6.

This paper requires only basic knowledge of category theory [17]. 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 at least in the category of sets and, as needed in the paper, in the category of graphs. The specific properties are presented in our technical report [8], which also contains the full proof of our main result.

2 The DPO Approach to Graph Rewriting

We will first define a family of categories of graphs and graph morphisms, being as general as possible by defining graph structures [6], which include different forms of graphs such as directed graphs and hypergraphs.

Definition 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 $l_s^A\colon A_s \to \Sigma_s$ and an OP-indexed family of mappings $(op^A)_{op\in OP}$ such that $op^A\colon A_s \to A_{s'}$ if $op \in OP_{s,s'}$.*

A graph structure morphism *$\varphi\colon A \to B$ is a sort-indexed family of mappings $\varphi = (\varphi_s\colon A_s \to B_s)_{s\in S}$ such that $l_s^A(x) = l_s^B(\varphi(x))$ and $op^B(\varphi(x)) = \varphi(op^A(x))$ for all $x \in A_s$. 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 $\varphi\colon A \to A$ is called* 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 respectively edge labels. In the following we will say "graph" instead of "graph structure" and "graph morphism" or just "morphism" instead of "graph structure morphism".

A category of graphs and graph morphisms 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. For additional properties of injective morphisms see Appendix A. So far we can obtain our main result (Theorem 7) only if we work with injective morphisms, which is, however, a natural requirement.

Definition 2 (Graph Transformation System). *A* rule *or* production *is a pair* $(\varphi_L\colon I \to L, \varphi_R\colon I \to R)$ *of injective graph morphisms. It can be applied to a graph* G*, resulting in a graph* H*, if there is an injective match morphism* $\varphi\colon L \to G$ *and we can find a graph* C *and morphisms such that the two squares in the following diagram are both pushouts.*

$$\begin{array}{ccccc} L & \xleftarrow{\varphi_L} & I & \xrightarrow{\varphi_R} & R \\ {\scriptstyle\varphi}\downarrow & & \downarrow & & \downarrow \\ G & \longleftarrow & C & \longrightarrow & H \end{array}$$

A graph transformation system *is a set* $\mathcal{P}$ *of productions.*

The diagram above consisting of two pushouts has led to the name double-pushout or DPO approach. The intuition behind this approach is to find a left-hand side L in a graph G, remove L apart from the interface I and to attach R to the interface in the remaining graph C, resulting in H.

Note: Instead of writing $(\varphi_L\colon I \to L, \varphi_R\colon I \to R)$ 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.

We use a running example throughout the paper which is deliberately kept very simple. Figure 1 shows three spans $L \leftarrow I \to R$ which form the rule set $\mathcal{P}$ 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—represented by a loop—are sent. A duplex connection can be used both ways, 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 contexualization.

Definition 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* $J \to E \leftarrow \overline{J}$*.*

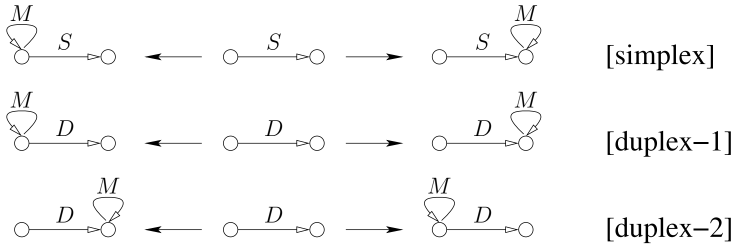


Fig. 1. Rules of a graph transformation system.

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

$$\begin{array}{ccccc} J & \longrightarrow & E & \longleftarrow & \overline{J} \\ \downarrow & & \downarrow & \swarrow & \\ G & \longrightarrow & \overline{G} & & \end{array}$$

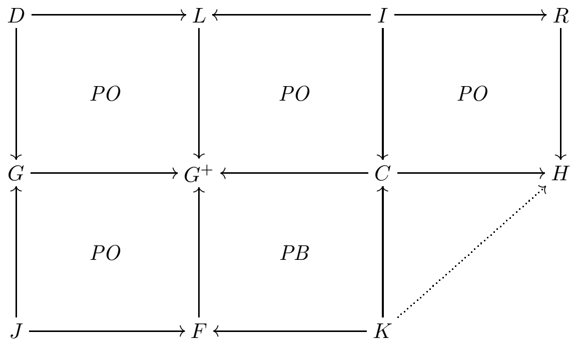
Note that composition is defined only up to isomorphism, since the pushout object is unique only up to isomorphism.

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 can be interpreted as composition of L with C respectively R with C along interface I. In the context of this paper however it is important to consider also 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 [9].

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 4 (Rewriting with borrowed contexts). *Let $\mathcal{P}$ 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 $(L \leftarrow I \rightarrow R) \in \mathcal{P}$ and there are graphs D, G^+, C and additional morphisms such that the following diagram commutes and the* *squares are either pushouts (PO) or pullbacks (PB) with injective morphisms.*



Symbolically this is denoted by the transition $(J \to G) \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H)$*, which is also called* 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 $G \leftarrow D \to L$ 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 $J \to F$ 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 two pushout complements that are constructed in Definition 4 may not exist. The middle square in the upper row can only be completed if the dangling edge condition is satisfied, i.e., 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. 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 context $J \to F$ to $J \to G$. In other words, the dangling edge condition is required also 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 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 $J \to F \to G^+$. This 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 Figure 7 and 8 in Appendix B.

In order to illustrate Definition 4, we regard rule [simplex] 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, i.e., the left-hand node is always mapped to a left-hand node, analogously for the right-hand node.

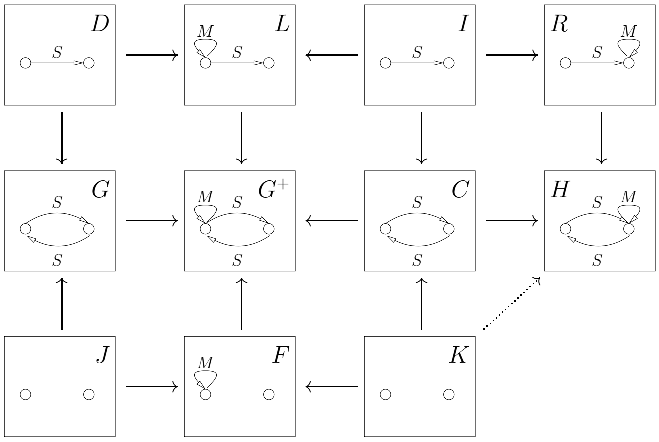


Fig. 2. Rewriting with borrowed contexts in the example graph transformation system.

4 Bisimilarity is a Congruence

We now arrive at the main theorem of this paper: We will show that the bisimilarity defined on labelled graph transition systems is a congruence. Before that we need two more definitions.

Definition 5 (Bisimulation and Bisimilarity). *Let $\mathcal{P}$ be a set of productions. Let $\mathcal{R}$ be a symmetric relation containing pairs of graphs with interfaces of the form $(J \to G, J \to G')$, also written $(J \to G)\,\mathcal{R}\,(J \to G')$.*

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

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

In order to state Theorem 7, we have to be able to close a bisimulation or simply a relation under all possible contexts.

Definition 6 (Closure under Contextualization). *Let $\mathcal{R}$ be a relation on graphs with interfaces as in Definition 5. By $\hat{\mathcal{R}}$ we denote the closure of $\mathcal{R}$ under contextualization, i.e., $\hat{\mathcal{R}}$ is the smallest relation that contains, for every pair $(J \to G, J \to G') \in \mathcal{R}$ and for every context of the form $J \to E \leftarrow \overline{J}$, the pair of morphisms $(\overline{J} \to \overline{G}, \overline{J} \to \overline{G}')$ which results from the composition of $J \to G$ and $J \to E \leftarrow \overline{J}$ respectively $J \to G'$ and $J \to E \leftarrow \overline{J}$.*

A relation $\mathcal{R}$ is a congruence, i.e., closed under contexts whenever $\hat{\mathcal{R}} = \mathcal{R}$. Since obviously $\mathcal{R}$ is contained in $\hat{\mathcal{R}}$, it suffices to show $\hat{\mathcal{R}} \subseteq \mathcal{R}$. We only give a proof sketch, the full proof can be found in [8].

Theorem 7 (Bisimilarity is a Congruence). *Whenever $\mathcal{R}$ is a bisimulation, then $\hat{\mathcal{R}}$ 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.

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

So let $\mathcal{R}$ be a bisimulation and let $(\overline{J} \to \overline{G}) \mathrel{\hat{\mathcal{R}}} (\overline{J} \to \overline{G}')$. We assume that

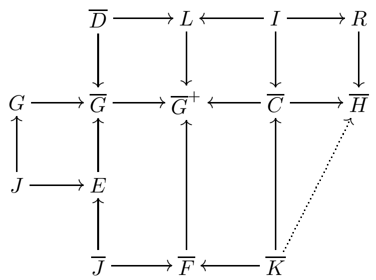
$$(\overline{J} \to \overline{G}) \xrightarrow{\overline{J} \to \overline{F} \leftarrow \overline{K}} (\overline{K} \to \overline{H}).$$

Our goal is to show that there exists a transition

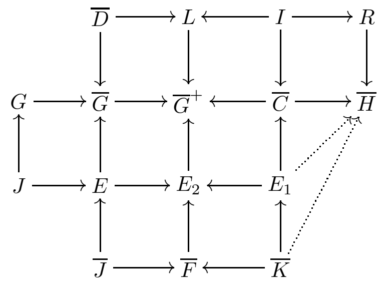
$$(\overline{J} \to \overline{G}') \xrightarrow{\overline{J} \to \overline{F} \leftarrow \overline{K}} (\overline{K} \to \overline{H}')$$

with $(\overline{K} \to \overline{H}) \mathrel{\hat{\mathcal{R}}} (\overline{K} \to \overline{H}')$, which implies that $\hat{\mathcal{R}}$ is a bisimulation. In Step 1 we will construct a transition $(J \to G) \xrightarrow{J \to F \leftarrow K} (K \to H)$ which implies a transition $(J \to G') \xrightarrow{J \to F \leftarrow K} (K \to H')$ with $(K \to H) \mathrel{\mathcal{R}} (K \to H')$, since $\mathcal{R}$ 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 $(\overline{J} \to \overline{G}) \mathrel{\hat{\mathcal{R}}} (\overline{J} \to \overline{G}')$ means that there is some pair $(J \to G) \mathrel{\mathcal{R}} (J \to G')$ 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 $(\overline{J} \to \overline{G}) \xrightarrow{\overline{J} \to \overline{F} \leftarrow \overline{K}} (\overline{K} \to \overline{H})$ 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.



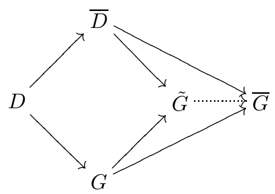
(1)



(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.

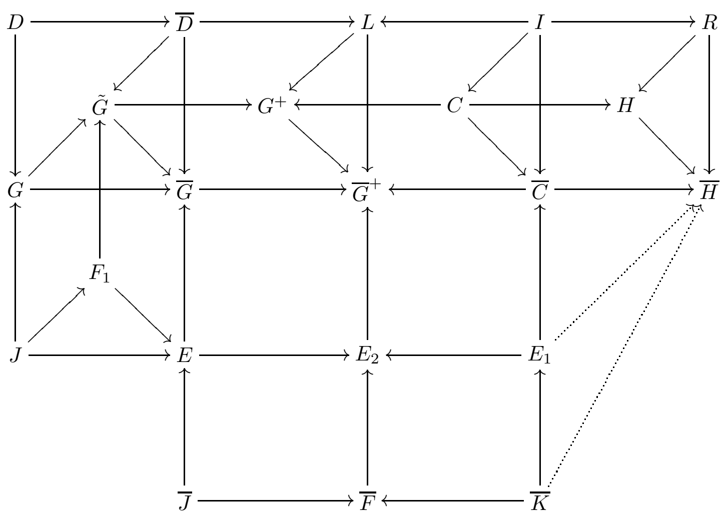
We can now construct D as the pullback of $G \to \overline{G}$ and $\overline{D} \to \overline{G}$, followed by the construction of $\tilde{G}$ as the pushout (see Diagram (3)). In this way we split the morphisms $G \to \overline{G}$ and $\overline{D} \to \overline{G}$ and we obtain the morphism $\tilde{G}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



(3)

In Diagram (2) we can now split the upper row of pushouts and the pushout to the very left by classical pushout splitting respectively using Property A.3 which results in 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)). 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 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.

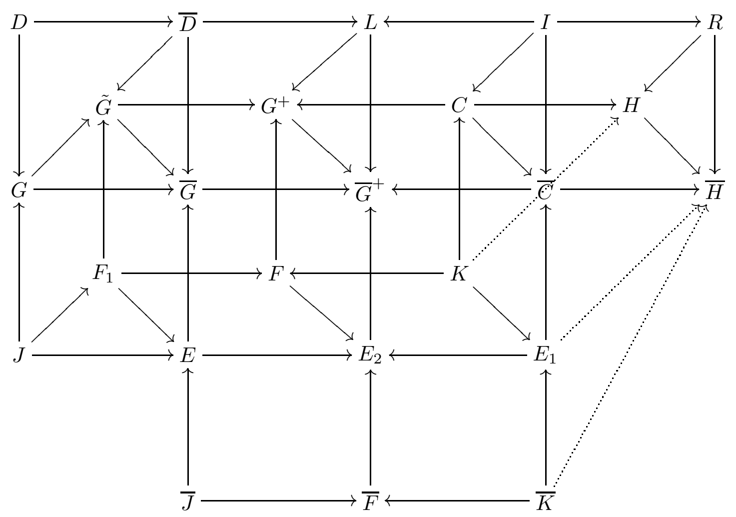


(4)

Since the left and top square 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 obtain that both squares in Diagram (6) are pushouts. That is, the left cube commutes, all morphisms are injective and all squares are pushouts.

$$\begin{array}{ccccc} F_1 & \longrightarrow & F & \longrightarrow & G^+ \\ \downarrow & & \downarrow & & \downarrow \\ E & \longrightarrow & E_2 & \longrightarrow & \overline{G}^+ \end{array} \qquad (5) \qquad\qquad \begin{array}{ccccc} F_1 & \longrightarrow & F & \longrightarrow & E_2 \\ \downarrow & & \downarrow & & \downarrow \\ \tilde{G} & \longrightarrow & G^+ & \longrightarrow & \overline{G}^+ \end{array} \qquad (6)$$

Focussing on the right cube, we can infer from Property A.2 that the left and top square of the cube are pullbacks, since they are pushouts. We construct K as pullback of the morphisms $C \to \overline{C}$ and $E_1 \to \overline{C}$ and obtain $K \to F$ as induced morphism by using the pullback property of the left square. The morphisms $K \to C$, $K \to E_1$ are injective by Property 1 and $K \to F$ is injective as the first morphism of an injective composition. We can now infer by classical pullback decomposition: Since the left, top and right squares are pullbacks, 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. Analogously we can conclude that the three right-hand squares in the upper row are pushouts.



(7)

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

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

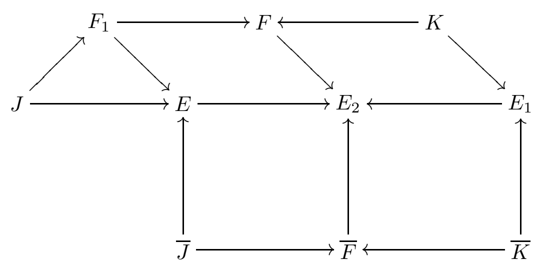
using the notation of Definition 4. Since $\mathcal{R}$ is a bisimulation, this implies

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

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

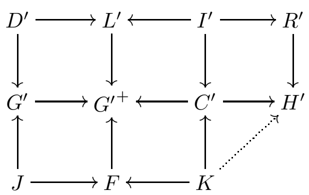
Step 2: As mentioned above we will now extend the transition from $J \to G'$ to $K \to H'$ with $(K \to H)\,\mathcal{R}\,(K \to H')$ obtained above to construct a transition from $(\overline{J} \to \overline{G}')$ to $(\overline{K} \to \overline{H}')$ with $(\overline{K} \to \overline{H})\,\hat{\mathcal{R}}\,(\overline{K} \to \overline{H}')$. 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) and we obtain 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.

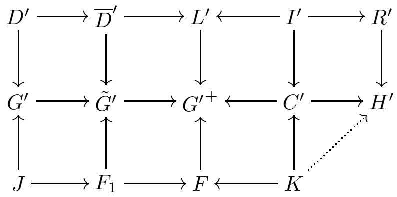


(8)

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 and therefore we can split the two left-hand side pushouts by classical pushout splitting and by Property A.3 as shown in Diagram (10).

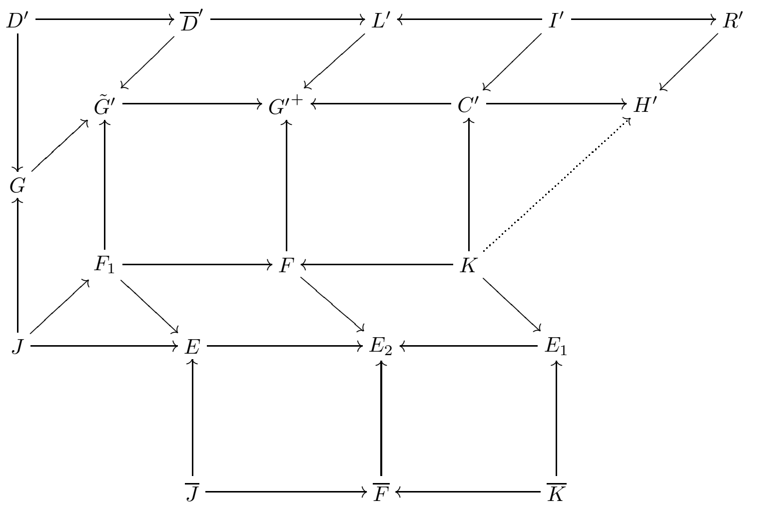


(9)



(10)

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



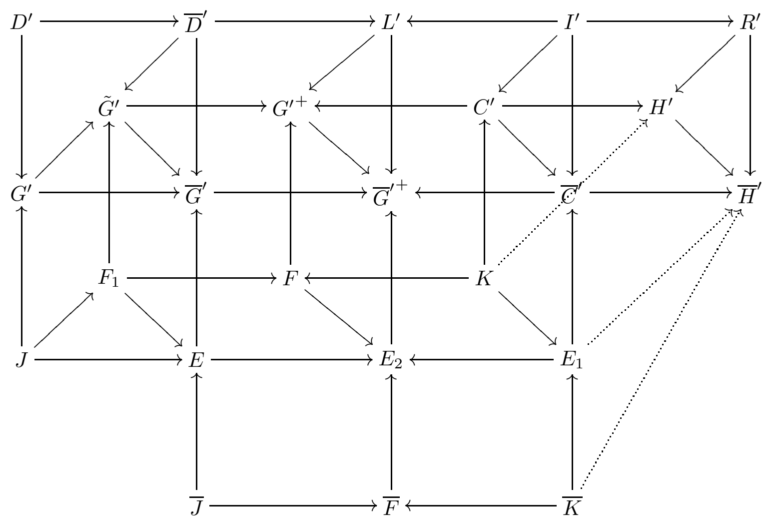
(11)

It remains 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 pushout of $K \to C'$ and $K \to E_1$ and obtain $\overline{C}' \to \overline{G}'^{+}$ as induced morphism. By classical pushout decomposition we have: Since the left, bottom and right squares are pushouts, it follows that 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.

$$\begin{array}{ccccc} K & \longrightarrow & E_1 & \longrightarrow & E_2 \\ \downarrow & & \downarrow & & \downarrow \\ C' & \longrightarrow & \overline{C}' & \longrightarrow & \overline{G}'^{+} \end{array} \tag{12}$$

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



(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 4 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 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 earlier considerations we know that $\overline{K} \to \overline{H}$ is the composition of $K \to H$ with $K \to E_1 \leftarrow \overline{K}$ and hence $(\overline{K} \to \overline{H})\, \hat{\mathcal{R}}\, (\overline{K} \to \overline{H}')$. This means that we have achieved our goal stated at the beginning of the proof, which implies that $\hat{\mathcal{R}}$ is a bisimulation and $\sim$ is a congruence. □

5 Borrowed Contexts at Work: An Example

In order to further pursue the example we will first introduce a proof technique simplifying bisimilarity proofs. This technique is a straightforward instance of an up-to technique [18, 24]. The underlying idea behind the technique is the observation that the relation $\mathcal{R}$ 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: We now demand that if a transition is matched by another, the pair of resulting graphs can be found in $\mathcal{R}$ after removal of identical contexts. Hence, this extended notion of bisimulation is called "bisimulation up to context". We first need an auxiliary definition.

Definition 8 (Progression). *Let $\mathcal{R}, \mathcal{S}$ be relations containing pairs of graphs with interfaces of the form $(J \to G, J \to G')$, where $\mathcal{R}$ is symmetric. We say that $\mathcal{R}$ progresses to $\mathcal{S}$, abbreviated by $\mathcal{R} \rightarrowtail \mathcal{S}$, if whenever $(J \to G)\, \mathcal{R}\, (J \to G')$ 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 $(K \to H)\, \mathcal{S}\, (K \to H')$.*

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

Definition 9 (Bisimulation up to Context). *Let $\mathcal{R}$ be a symmetric relation containing pairs of graphs with interfaces of the form $(J \to G, J \to G')$.*
If $\mathcal{R} \rightarrowtail \hat{\mathcal{R}}$, then $\mathcal{R}$ is called bisimulation up to context*.*

We will show in Proposition 10 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 10 (Bisimulation up to Context implies Bisimilarity). *Let $\mathcal{R}$ be a bisimulation up to context. Then it holds that $\mathcal{R} \subseteq \sim$.*

Proof (Proof Sketch). By carefully examining the proof of Theorem 7 again we can see that some simple modifications give us the following (stronger) theorem:

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

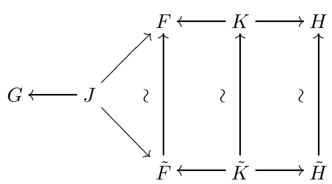
Since $\mathcal{R}$ is a bisimulation up to context we have $\mathcal{R} \rightarrowtail \hat{\mathcal{R}}$. The stronger version of Theorem 7 now implies $\hat{\mathcal{R}} \rightarrowtail \widehat{(\hat{\mathcal{R}})}$. Since the composition of contexts is associative we have $\widehat{(\hat{\mathcal{R}})} = \hat{\mathcal{R}}$, which implies $\hat{\mathcal{R}} \rightarrowtail \hat{\mathcal{R}}$ and hence that $\hat{\mathcal{R}}$ is a bisimulation, i.e., $\hat{\mathcal{R}} \subseteq \sim$. This implies $\mathcal{R} \subseteq \hat{\mathcal{R}} \subseteq \sim$. □

Since contextualization is defined only up to isomorphism, we can assume that $\hat{\mathcal{R}}$ is closed under isomorphism in the following sense: For every span $G \leftarrow J \rightarrow G'$, all isomorphic spans $\tilde{G} \leftarrow \tilde{J} \rightarrow \tilde{G}'$ are also contained in $\hat{\mathcal{R}}$.

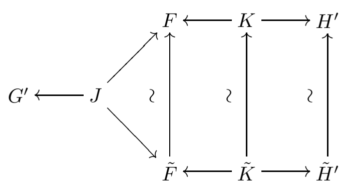
Similarly, we can restrict ourselves to abstract transitions when checking for bisimilarity: Assume that $(J \to G)\,\mathcal{R}\,(J \to G')$ and there are two transitions

$$(J \to G) \xrightarrow{J \to F \leftarrow K} (K \to H) \quad \text{and} \quad (J \to G) \xrightarrow{J \to \tilde{F} \leftarrow \tilde{K}} (\tilde{K} \to \tilde{H})$$

with isomorphisms from $\tilde{F}, \tilde{K}, \tilde{H}$ to F, K, H respectively such that the entire diagram commutes (see Diagram (14)). It is sufficient to show the existence of a transition $(J \to G') \xrightarrow{J \to F \leftarrow K} (K \to H')$ such that $(K \to H)\,\hat{\mathcal{R}}\,(K \to H')$. From this transition and Diagram (14) we can derive Diagram (15), where the arrows pointing upwards are isomorphisms and the diagram commutes. In such a situation we can infer the existence of a transition $(J \to G') \xrightarrow{J \to \tilde{F} \leftarrow \tilde{K}} (\tilde{K} \to \tilde{H}')$ such that $H \leftarrow K \rightarrow H'$ and $\tilde{H} \leftarrow \tilde{K} \rightarrow \tilde{H}'$ are isomorphic spans, from which it follows that $(\tilde{K} \to \tilde{H})\,\hat{\mathcal{R}}\,(\tilde{K} \to \tilde{H}')$.



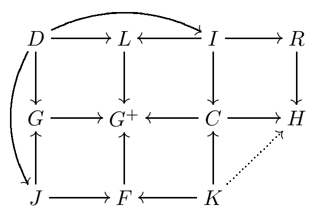
(14)



(15)

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

Definition 11. *Let* $(J \to G) \xrightarrow{J \to F \leftarrow K} (K \to H)$ *be a transition of* $J \to G$*. We say that the transition is* independent *whenever we can add two morphisms* $D \to J$ *and* $D \to I$ *to the diagram in Definition 4 such that the diagram commutes, i.e.,* $D \to I \to L = D \to L$ *and* $D \to J \to G = D \to G$ *(see Diagram (16)). We write* $(J \to G) \xrightarrow{J \to F \leftarrow K}_d (K \to H)$ *if the transition is not independent. In this case, the transition will be called* dependent.



(16)

Let $\mathcal{R}$*,* $\mathcal{S}$ *be two relations as in Definition 8. We say that* $\mathcal{R}$ *d-progresses to* $\mathcal{S}$*, abbreviated by* $\mathcal{R} \rightarrowtail_d \mathcal{S}$*, if whenever* $(J \to G)\,\mathcal{R}\,(J \to G')$ *and* $(J \to G) \xrightarrow{J \to F \leftarrow K}_d$

$(K \to H)$, *there exists a morphism* $K \to H'$ *such that*[1] $(J \to G') \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H')$ *and* $(K \to H)\, \mathcal{S}\, (K \to H')$.

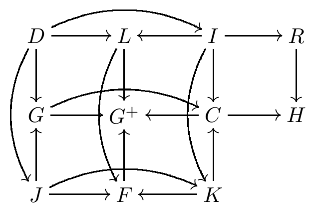
We can now use the new relation $\rightarrowtail_d$ as the basis of a new proof technique.

Proposition 12.

1. *Let* $\mathcal{R}$ *be a relation with* $(J \to G)\, \mathcal{R}\, (J \to G')$. *Given an independent transition* $(J \to G) \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H)$, *then 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* $(K \to H)\, \hat{\mathcal{R}}\, (K \to H')$.
2. *Let* $\mathcal{R}$ *be symmetric and let* $\mathcal{R} \rightarrowtail_d \hat{\mathcal{R}}$. *This implies that* $\mathcal{R}$ *is contained in* $\sim$.

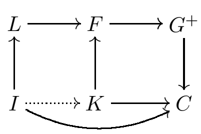
Proof.

1. We assume that $(J \to G)\, \mathcal{R}\, (J \to G')$ and $(J \to G) \overset{J \to F \leftarrow K}{\longrightarrow} (K \to H)$ is an independent transition (see Definition 11). That is, there are morphisms $D \to J$ and $D \to I$ such that Diagram (16) above commutes.

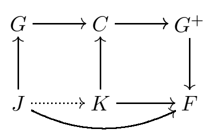


(17)

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. 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 induced morphism. We can then infer from Property A.4 that the left as well as the right square are pushouts. Analogously we can construct a morphism $J \to K$ in Diagram (19) and it follows that the left and the right square are both pushouts.



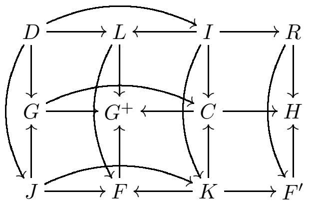
(18)



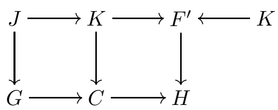
(19)

We now construct F' in Diagram (20) as the pushout of $I \to K$ and $I \to R$ giving us $F' \to H$ as 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).

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

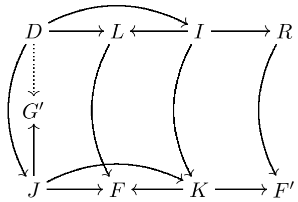


(20)

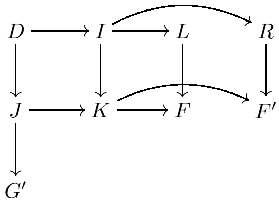


(21)

We now derive a corresponding transition of $J \to G'$ with the same rewriting rule. So far the morphisms shown in Diagram (22) are known, 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).

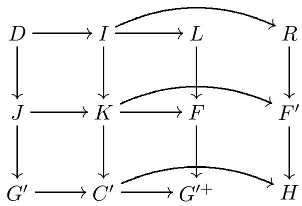


(22)

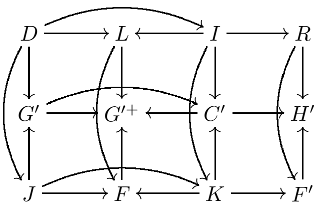


(23)

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).

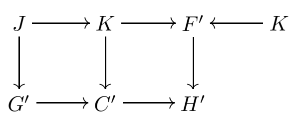


(24)



(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 $(K \to H)\, \hat{\mathcal{R}}\, (K \to H')$.



(26)

2. We first assume that $\mathcal{R} \rightarrowtail_d \mathcal{S}$ and we show that this implies $\mathcal{R} \rightarrowtail \mathcal{S} \cup \hat{\mathcal{R}}$. Let $(J \to G)\, \mathcal{R}\, (J \to G')$ 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 $(K \to H)\, \mathcal{S}\, (K \to H')$. 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 $(K \to H)\, \hat{\mathcal{R}}\, (K \to H')$. Combined we obtain $\mathcal{R} \rightarrowtail \mathcal{S} \cup \hat{\mathcal{R}}$.
Hence $\mathcal{R} \rightarrowtail_d \hat{\mathcal{R}}$ implies that $\mathcal{R} \rightarrowtail \hat{\mathcal{R}} \cup \hat{\mathcal{R}} = \hat{\mathcal{R}}$, hence $\mathcal{R}$ is a bisimulation up to context and is contained in $\sim$.

□

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

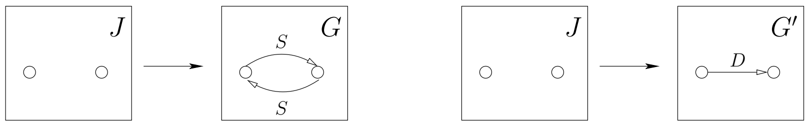


Fig. 3. Two graphs with interfaces which are bisimilar.

We consider the symmetric relation

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

and we will show that it is contained in $\sim$ using Proposition 12. For each of the three rules there are several partial matches for both G and G'. Most of these matches lead to independent transitions in the sense of Definition 11, since the graph to be rewritten and the left-hand side overlap only in their interfaces.

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

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

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 first transition of $J \to G$ in Figure 2 (where $F_1 = F$, $K_1 = K$, $H_1 = H$).

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

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

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

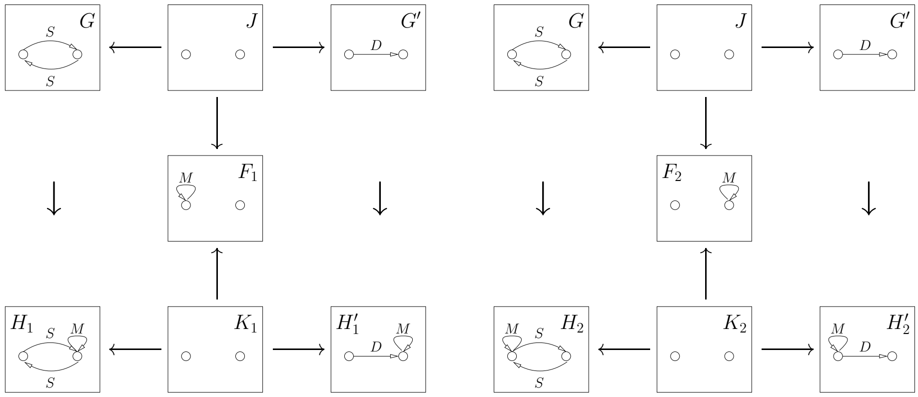


Fig. 4. Matching transitions of bisimilar graphs.

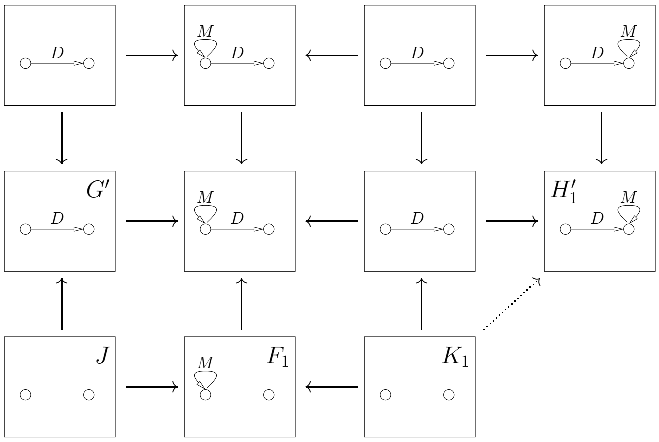


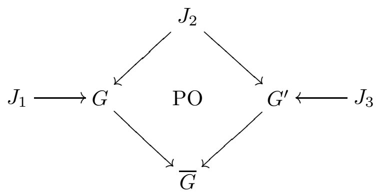
Fig. 5. Graphical representation of the derivation of the matching transition of $J \rightarrow G'$.

Furthermore, it holds that $(K_i \to H_i)\,\hat{\mathcal{R}}\,(K_i \to H_i')$ for $i = 1, 2$, since these graphs can be obtained by composing $J \to G$ respectively $J \to G'$ 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 12.

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

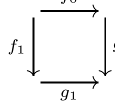
6 Connection to Relative Pushouts

A question which naturally arises is whether our construction is in some way connected to the relative pushouts of Leifer and Milner [16] or to the GRPOs of Sassone and Sobocinski [26]. In order to partially answer this question we regard the category of cospans, the objects of which are graphs and the arrows of which are injective cospans (or contexts) as in Definition 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 $J_1 \to \overline{G} \leftarrow J_3$. 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.

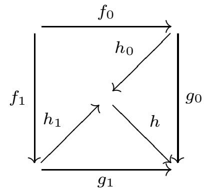


Before we pursue this issue further we will first define the notion of relative pushout (see [15, 16]).

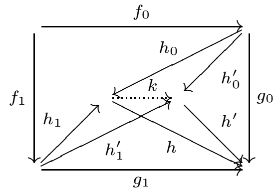
Definition 13 (Relative Pushout). *Consider—in any category* **C**—*a commuting square as shown in Diagram (27) such that* $g_0; f_0 = g_1; f_1$. *A relative pushout for this commuting square is a triple* h_0, h_1, h *satisfying the following two properties: (i) commutation:* $h_0; f_0 = h_1; f_1$ *and* $h; h_i = g_i$ *for* $i = 0, 1$ *(see Diagram (28); (ii) universality: for any* h_0', h_1', h' *satisfying* $h_0'; f_0 = h_1'; f_1$ *and* $h'; h_i' = g_i$ *for* $i = 0, 1$, *there exists a unique mediating arrow* k *such that* $h'; k = h$ *and* $k; h_i = h_i'$ *(see Diagram (29)).*



(27)



(28)



(29)

In the work of Leifer and Milner relative pushouts are used to formalize the notion of "minimal context". Let (l, r) with $l, r\colon 0 \to m$ 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 D, F are in some sense minimal arrows with this property. In this case there is a transition $a \stackrel{F}{\to} r; D$. The notion of "being minimal" is formalized by the notion of relative pushouts where h_1, h_0 are minimal with respect to the original commuting square.

We make an attempt at constructing a relative pushout in the category of cospans, highlight connections to the proof of Theorem 7 and point out arising problems. The attempt will only be partly successful due to inherent problems already pointed out in earlier work (see, e.g., the counterexample in [15] on pages 80/81), but we feel that it will nevertheless help to illustrate existing connections.

Reaction rules in our setting are pairs of 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 cospans as shown in Diagram (30).

$$\begin{array}{ccccc} \emptyset & \longrightarrow & L & \longleftarrow & I \\ \downarrow & & & & \downarrow \\ G & & & & \overline{C} \\ \uparrow & & & & \uparrow \\ J & \longrightarrow & \overline{F} & \longleftarrow & \overline{K} \end{array} \qquad (30)$$

$$\begin{array}{ccccc} \emptyset & \longrightarrow & L & \longleftarrow & I \\ \downarrow & & \downarrow & & \downarrow \\ G & \longrightarrow & \overline{G}^+ & \longleftarrow & \overline{C} \\ \uparrow & & \uparrow & & \uparrow \\ J & \longrightarrow & \overline{F} & \longleftarrow & \overline{K} \end{array} \qquad (31)$$

Since the two compositions of cospans coincide, there must be a graph $\overline{G}^+$ such that Diagram (31) commutes and the lower left and the upper right squares are pushouts. Although the graph $\overline{G}^+$ is unique up to isomorphism, Diagram (31) is not necessarily unique, not even up to isomorphism. This can be caused by non-trivial automorphisms of $\overline{G}^+$. The outcome of the following construction depends on the specific choice of diagram.

An example for this situation is given in Figure 6, 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 and would lead to different outcomes of the following construction.

We therefore assume in the following that the automorphism group of $\overline{G}^+$ consists of only one element (the identity) and that thus Diagram (31) is uniquely determined. A method to force this uniqueness is to work in a category with support, where every node and edge has a unique name that is not lost under composition. In order not to complicate matters we choose not to pursue this direction at the moment.

In the following we construct Diagram (32) in several steps. First we construct D as the pullback of $L \to \overline{G}^+$ and $G \to \overline{G}^+$, followed by the construction of G^+ as the pushout of $D \to G$ and $D \to L$. This gives us $G^+ \to \overline{G}^+$ as induced morphism. Next we can split the two pushouts (the lower left square and the

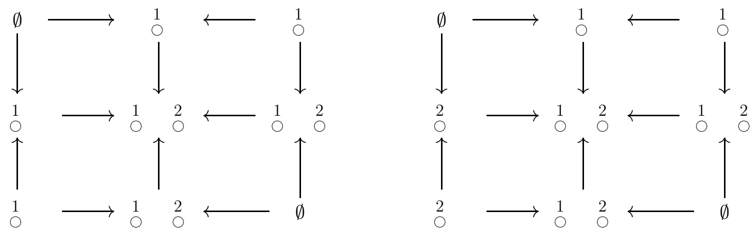
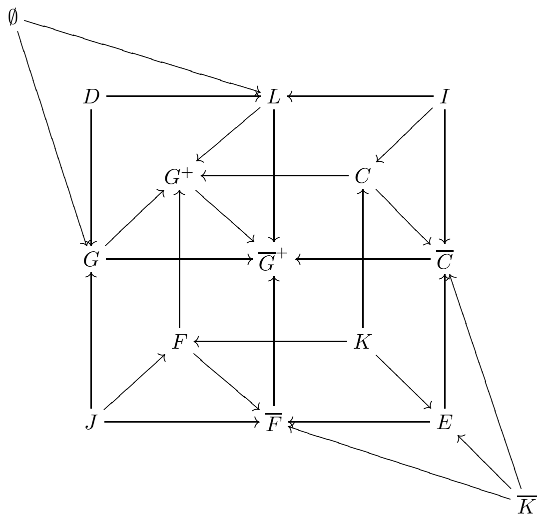


Fig. 6. Different diagrams for a graph with non-trivial automorphisms.

upper right square) 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.



(32)

This construction resembles the proof of Theorem 7 in many places. We conjecture that the triple $I \to C \leftarrow K$, $J \to F \leftarrow K$, $K \to E \leftarrow \overline{K}$ of cospans can be seen as a relative pushout. However in order to establish this result, we have to choose carefully in which category to take relative pushouts. It seems to be necessary to add support to the category or to work with GRPOs as it is done in [26].

A more complete answer to this question can be found in [27].

7 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 and thus enables us to easily state which components are associated with each other in different graphs. Hence we need not consider explicit names for graph components.

We have made some investigations concerning the adaptation of the concept of relative pushouts for cospans of graphs. However, there are fundamental problems, mainly caused by graphs having non-trivial automorphisms (see, e.g., the counterexample in [15] on pages 80/81, which can be directly transferred into our framework). We believe, however, that our construction is very close in spirit to the notion of relative pushouts introduced by Leifer and Milner and that it should be possible to show the equivalence of these two notions in a suitable graph category with support.

Our results do not only hold in graph structure categories, but also in other categories which satisfy certain properties typical for the categories of sets and high-level replacement systems [5]. In this context it is also interesting to point out that most of the categorical properties we need hold already in adhesive categories [14].

In the future we also plan to address the following two questions: How should weak bisimilarity be defined and is it a congruence? Do our results still hold if we allow for non-injective morphisms? In connection with the latter question it is interesting to note that several properties stated in Appendix A cease to hold if we give up the requirement of injectivity. Furthermore we plan to introduce more proof techniques in order to simplify bisimulation proofs. Whenever a graph and a left-hand side overlap only in their interfaces, another graph with the same interface will certainly be able to match the corresponding rewriting step with borrowed context, since this step only changes the interface itself. Hence it should be possible to remove some superfluous transitions without changing the underlying bisimilarity.

Another interesting question would be to find out which bisimulation congruences are produced by the various encodings of π-calculus into graph rewriting and to see in what way they are related to existing congruences for this calculus. It also remains to determine in what way our bisimilarity is related to dynamic bisimulation as presented in [2, 21].

Acknowledgements: We would like to thank the anonymous referees for their helpful comments. We are especially grateful to Robin Milner for suggesting this research topic and for sharing his ideas with us.

References

1. Paolo Baldan, Andrea Corradini, and Ugo Montanari. Bisimulation equivalences for graph grammars. In *Formal and Natural Computing - Essays Dedicated to Grzegorz Rozenberg*, pages 158–190. Springer, 2002. LNCS 2300.

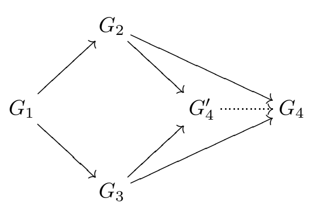
2. Roberto Bruni, Ugo Montanari, and Vladimiro Sassone. Open ended systems, dynamic bisimulation and tile logic. In *Proc. of IFIP TCS 2000*. Springer, 2000. LNCS 1872.
3. A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. Algebraic approaches to graph transformation—part I: Basic concepts and double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.1: Foundations*, chapter 3. World Scientific, 1997.
4. H. Ehrig. Bigraphs meet double pushouts. *EATCS Bulletin*, 78:72–85, October 2002.
5. H. Ehrig, M. Gajewsky, and F. Parisi-Presicce. High-level replacement systems with applications to algebraic specifications and Petri nets. In H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors, *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.3: Concurrency, Parallellism, and Distribution*, chapter 6, pages 341–400. World Scientific, 1999.
6. H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. Algebraic approaches to graph transformation—part II: Single pushout approach and comparison with double pushout approach. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.1: Foundations*, chapter 4. World Scientific, 1997.
7. Hartmut Ehrig and Barbara König. Deriving bisimulation congruences in the DPO approach to graph rewriting. In *Proc. of FOSSACS '04*, LNCS. Springer, 2004. to appear.
8. Hartmut Ehrig and Barbara König. Deriving bisimulation congruences in the DPO approach to graph rewriting. Technical report, Universität Stuttgart, 2004. to appear.
9. F. Gadducci and R. Heckel. An inductive view of graph transformation. In *Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT '97*, pages 223–237. Springer-Verlag, 1997. LNCS 1376.
10. F. Gadducci and U. Montanari. A concurrent graph semantics for mobile ambients. In S. Brookes and M. Mislove, editors, *Proceedings of the 17th MFPS*, volume 45 of *Electronic Notes in Computer Science*. Elsevier Science, 2001.
11. F. Gadducci and U. Montanari. Comparing logics for rewriting: Rewriting logic, action calculi and tile logic. *Theoretical Computer Science*, 285(2):319–358, 2002.
12. Ole Høgh Jensen and Robin Milner. Bigraphs and transitions. In *Proc. of POPL 2003*, pages 38–49. ACM, 2003.
13. Barbara König. A graph rewriting semantics for the polyadic π-calculus. In *Workshop on Graph Transformation and Visual Modeling Techniques (Geneva, Switzerland), ICALP Workshops '00*, pages 451–458. Carleton Scientific, 2000.
14. Stephen Lack and Pawel Sobocinski. Adhesive categories. In *Proc. of FOSSACS '04*, LNCS. Springer, 2004. to appear.
15. James J. Leifer. *Operational congruences for reactive systems*. PhD thesis, University of Cambridge Computer Laboratory, September 2001.
16. James J. Leifer and Robin Milner. Deriving bisimulation congruences for reactive systems. In *Proc. of CONCUR 2000*, 2000. LNCS 1877.
17. Saunders Mac Lane. *Categories for the Working Mathematician*. Springer-Verlag, 1971.
18. R. Milner and D. Sangiorgi. Techniques of weak bisimulation up-to. In *Proc. of CONCUR '92*. Springer-Verlag, 1992. LNCS 630.
19. Robin Milner. Bigraphical reactive systems. In *Proc. of CONCUR '01*, pages 16–35. Springer-Verlag, 2001. LNCS 2154.

20. Robin Milner and Davide Sangiorgi. Barbed bisimulation. In *Proc. of ICALP '92*. Springer-Verlag, 1992. LNCS 623.
21. U. Montanari and V. Sassone. Dynamic congruence vs. progressing bisimulation for CCS. *Fundamenta Informaticae*, 16:171–196, 1992.
22. Ugo Montanari and Marco Pistore. Concurrent semantics for the π-calculus. *Electronic Notes in Theoretical Computer Science*, 1, 1995.
23. Grzegorz Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.1: Foundations*, volume 1. World Scientific, 1997.
24. Davide Sangiorgi. On the proof method for bisimulation. In *Proc. of MFCS '95 (Mathematical Foundations of Computer Science)*, pages 479–488. Springer, 1995. LNCS 969.
25. Davide Sangiorgi and David Walker. *The π-calculus—A Theory of Mobile Processes*. Cambridge University Press, 2001.
26. Vladimiro Sassone and Pawel Sobocinski. Deriving bisimulation congruences: 2-categories vs precategories. In *Proc. of FoSSaCS 2003*, pages 409–424, 2003.
27. Vladimiro Sassone and Pawel Sobocinski. Coinductive reasoning for contextual graph-rewriting. Unpublished, 2004.
28. Peter Sewell. From rewrite rules to bisimulation congruences. *Theoretical Computer Science*, 274(1–2):183–230, 2002.

A Pushout-Pullback Properties

Definition 3 and the proof of Theorem 7 require the following properties to hold in the category we are working in. All these conditions are satisfied by a category 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 component-wise in the category **Set**. We will state the properties but give no proofs.

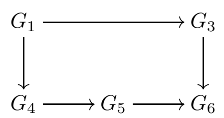
A.1 Injectivity Conditions: The two morphisms generated by a pushout, pullback or pushout complement of injective morphisms are always injective. Furthermore, given two injective morphisms $G_2 \to G_4$, $G_3 \to G_4$, if we construct G_1 as the pullback, followed by a construction of G'_4 as the pushout of $G_1 \to G_2$, $G_1 \to G_3$ as shown in Diagram (33), all morphisms, including the unique mediating morphism $G'_4 \to G_4$ are injective.



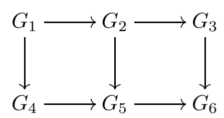
(33)

A.2 Injective Pushouts are Pullbacks: A pushout which consists of four injective morphisms is also a pullback.

A.3 Pushout Complement Splitting: If the square in Diagram (34) 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 (35) consists of two pushouts and all morphisms are injective.

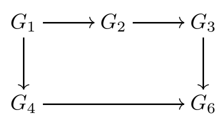


(34)



(35)

Remark: Whenever the square in Diagram (34) is a pullback, then the square splits into two pullbacks (as shown in Diagram (35)) in every category without the requirement of injectivity (classical pullback splitting). And if the square in Diagram (36) is a pushout, it splits into two pushouts in every category (classical pushout splitting).



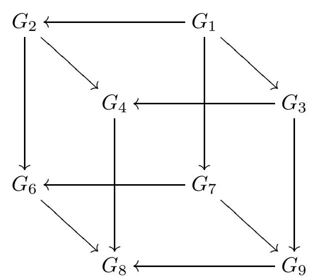
(36)

A.4 Special Pushout-Pullback Decomposition: In Diagram (35) 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, then the left square and the right squares are both pushouts.

A.5 Special Pullback-Pushout Decomposition: In Diagram (35) above consisting of injective morphisms, whenever the the outer square is a pullback and the left square is a pushout, then the left square and the right squares are both pullbacks.

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

A.6 Cube Pushout-Pullback Property: Given a commutative cube consisting of injective morphisms as shown in Diagram (37) below, where the bottom, left, front and top squares are pullbacks. Then whenever the front square is a pushout, the back square is a pushout as well.



(37)

A.7 Special Pushout Decomposition: If all morphisms in Diagram (38) below are injective, the left and right square 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 (see next property).

$$\begin{array}{ccccc} G_1 & \longrightarrow & G_2 & \longleftarrow & G_3 \\ \downarrow & & \downarrow & & \downarrow \\ G_4 & \longrightarrow & G_5 & \longleftarrow & G_6 \end{array} \qquad (38)$$

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 (39). Then there exists a unique graph G_3 and unique morphisms $G_1 \to G_3$, $G_3 \to G_4$ such that in square in Diagram (40) is a pushout.

$$\begin{array}{ccc} G_1 & \longrightarrow & G_2 \\ & & \downarrow \\ & & G_4 \end{array} \qquad (39) \qquad\qquad \begin{array}{ccc} G_1 & \longrightarrow & G_2 \\ \downarrow & & \downarrow \\ G_3 & \longrightarrow & G_4 \end{array} \qquad (40)$$

B Graph Representation by Venn Diagrams

We can illustrate Definition 4 by drawing graphs as Venn diagrams. The graph G which 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 depict a part of the Diagram of Definition 4, drawing graphs that can be mapped into G^+ and ignoring the graphs to the very right.

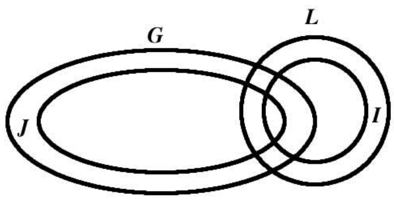


Fig. 7. Representation of a graph G overlapping with a left-hand side L.

If have a graph G in a context E and find a partial match of a left-hand side L, a typical situation is depicted in Figure 9. Diagram (7) in the proof of Theorem 7 can be represented with Venn diagrams as shown in Figure 10. Again we omit the graphs to the very right.

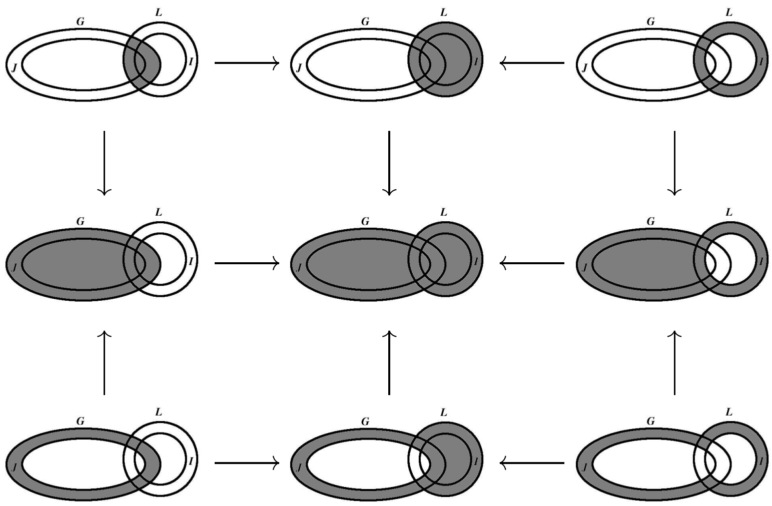


Fig. 8. Graphical represenation of rewriting with borrowed contexts with Venn-like diagrams.

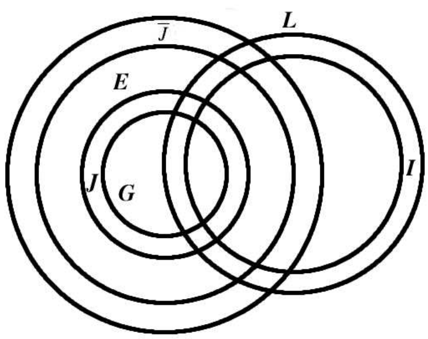


Fig. 9. Representation of a graph G with context E and overlapping with a left-hand side L.

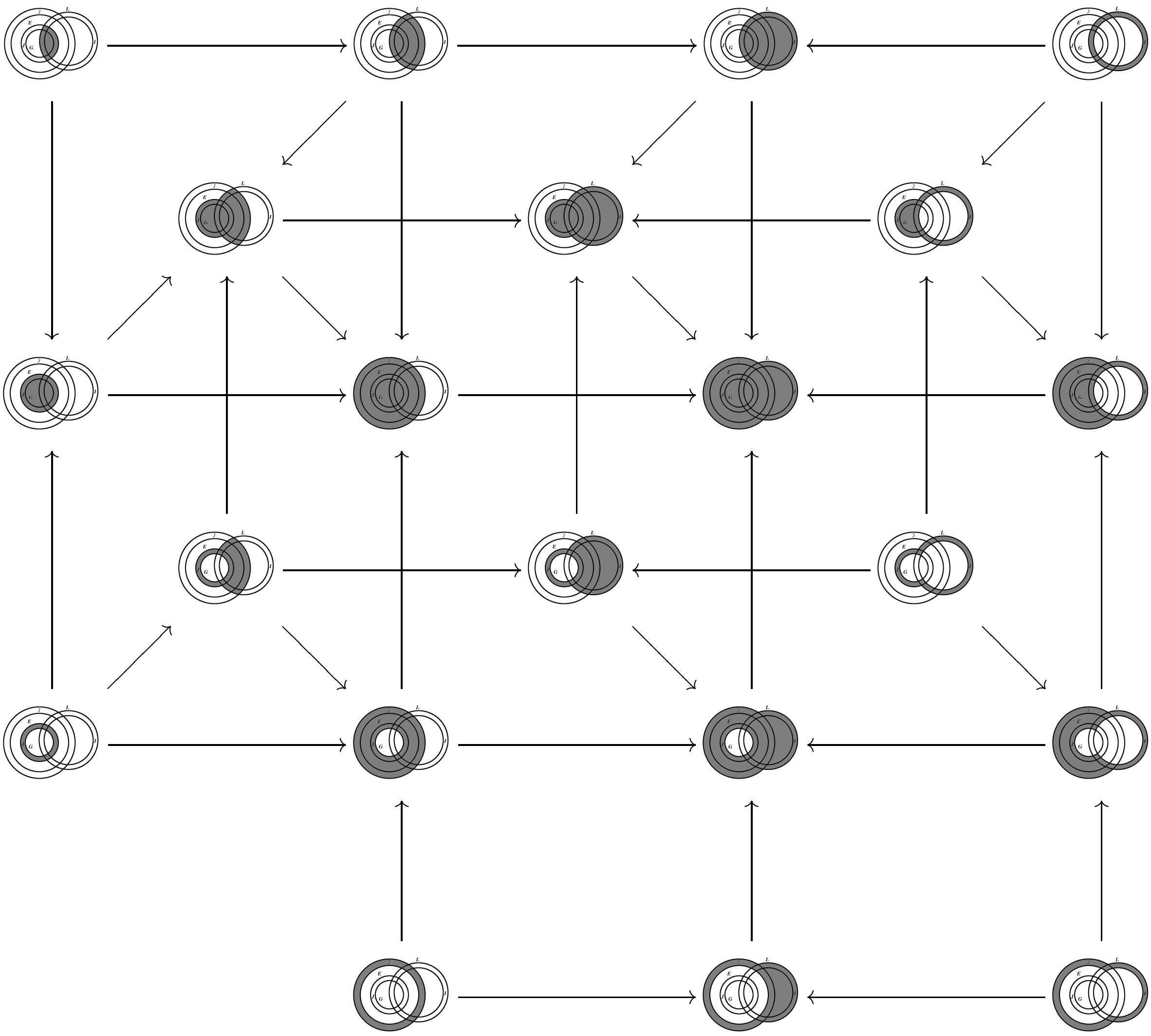


Fig. 10. Representing Diagram (7) in the proof of Theorem 7 with Venn diagrams.

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

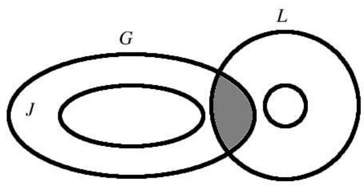


Fig. 11. The partial match is located in the overlap of the interfaces J and I.