Parallel and Sequential Independence for Borrowed Contexts*

Filippo Bonchi[1], Fabio Gadducci[1], and Tobias Heindel[2]

[1] Dipartimento di Informatica, Università di Pisa
[2] Institut für Informatik und Interaktive Systeme, Universität Duisburg-Essen

Abstract. Parallel and sequential independence are central concepts in the concurrency theory of the double pushout (DPO) approach to graph rewriting. However, so far those same notions were missing for DPO rewriting extended with borrowed contexts (DPOBC), a formalism used for equipping DPO derivations with labels and introduced for modeling open systems that interact with the environment.

In this work we propose the definition of parallel and sequential independence for DPOBC rewriting, and we prove that these novel notions allow generalizing the Church-Rosser and parallelism theorems holding for DPO rewriting. Most importantly, we show that the DPOBC version of these theorems still guarantees the local confluence and the parallel execution of pairs of independent DPOBC derivations.

1 Introduction

The dynamics of a computational device is often defined by a reduction system (RS): a set, representing the possible states of the device; and a relation among these states, representing the possible evolutions of the device. Most often, the states have a topological structure, and the relation is built from a finite set of reductions, in accordance to such a state structure. This construction could e.g. be performed by a matching mechanism, identifying which sub-component of a state is actually evolving, as well as denoting the successor state. This is the case for term and graph rewriting systems, where states are trees (graphs, respectively), and the reduction is built according to the DPO approach.

While RSs may convey the semantics with few rules, their drawback is poor compositionality: performing a computing step implicitly calls for a global choice of the subsystem where the reduction has to take place, making it difficult to model the dynamic behaviour of arbitrary standalone components. A solution is to express the behaviour of a computational device by a labelled transition system (LTS). Should the label associated to a component evolution faithfully express how that component might interact with the whole of the system, it would be possible to analyse on its own the behaviour of a single component, disregarding the larger systems it might occur in. Thus, a "well-behaved" LTS represents a fundamental step towards a compositional semantics of a computational device.

* Research partially supported by the EU FP6-IST IP 16004 SENSORIA.

H. Ehrig et al. (Eds.): ICGT 2008, LNCS 5214, pp. 226–241, 2008.

Classically, graph transformation fits in the "global view" of the matching / "local application" of rules dichotomy that we mentioned before. The solution proposed by Ehrig and König [1] for DPO rewriting takes into account *graph with interfaces* for system representation. Now, a state is a morphism $J \to G$, where J represents the part of G that may interact with the environment. A reduction $J \to G \Rightarrow K \to H$ is built according to the DPO approach, but it is labelled with a *borrowed context* $J \to F \leftarrow K$, intuitively representing the amount of information required from the environment, before a reduction may take place.

Considered merely as a mechanism for label synthesis, the extension of DPO with borrowed contexts (DPOBC) is an instance of the so-called *relative pushouts* approach [2] for constructing an LTS from an RS, guaranteeing that the associated bisimilarity is automatically a congruence. This shows that the formalism has a sound theoretical basis, and further venues of work include deriving suitable inference rules (e.g. in the so-called SOS style) for building synthesized LTSs, as attempted in [3]. However, graphs with interfaces actually represent on their own an interesting formalism for modeling open systems, already used e.g for the graphical encoding of process calculi [4] or of spatial logics [5]. So, the construction via DPOBC of an LTS that has graphs with interfaces as both states and labels should just represent the starting point for further investigations. First of all, a thorough analysis of the labelling should be carried out, in order to understand which systems are bisimilar. So far, we are only aware of the correspondence with standard interleaving semantics that has been proved for the LTS obtained by the graphical encoding of CCS in [6]. Moreover, since DPOBC may represent a foundational mechanism for rewriting (when considering graphs with interfaces as main actors in system representation), it seems natural to ask under which conditions we may talk about concurrency for DPOBC derivations. As a start, suitable notions of parallel and sequential independence should be provided, as well as proving these definitions adequate, by establishing Church-Rosser and parallelism properties. As a further sanity check, it should be pivotal to ensure that in a LTS the labels of the square of reductions induced by two parallel independent derivations are somehow well-behaved.

The paper has the following structure. Section 2 recalls the basic definitions of DPOBC rewriting and illustrates our running example modeling a simple protocol for message broadcasting. Section 3 discusses the nature of labels on those LTSs induced by DPOBC, arguing, also on the basis of our example, which relationships should hold between the labels of a square of concurrent derivations. Section 4 introduces the notion of parallel and sequential independence for DPOBC rewriting, and it states the main results of our paper, i.e., the theorem concerning the (local) Church-Rosser and parallelism properties. Moreover, it shows that the labels of a square of derivations, as induced by the Church-Rosser property, enjoy what we call the *strict diamond property*, as introduced in Section 3. Section 5 further discusses about labels of parallel derivations, pointing out why they cannot be decomposed. Finally, Section 6 draws some conclusions, and it outlines venues and hypotheses for further work.

2 Background

In this section we recall the basics of the double pushout (DPO) approach to rewriting [7,8]. In particular, we consider its variant where rules and matches are monomorphisms; and, most importantly, we introduce it in full generality, assuming an arbitrary adhesive category [9] for system representation (yet aiming at a presentation suitable for those familiar with standard graph transformation).

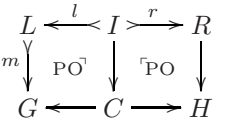
Moreover, we also present the main concepts underlying DPO rewriting with *borrowed contexts* (DPOBC), as proposed in [1]. In the following, we fix a chosen adhesive category $\mathbb{C}$, where rules and matches live. For all notions concerning adhesive categories we refer the reader to [9].

2.1 Double Pushout Rewriting with Monic Matches

The idea of DPO rewriting dates back to the early Seventies, and has been shown to be a viable rewriting mechanism in any adhesive category [9]. In the paper we consider its restriction to monic matches, as presented in [8], since the original definition of DPO rewriting with borrowed contexts is based on this variant.

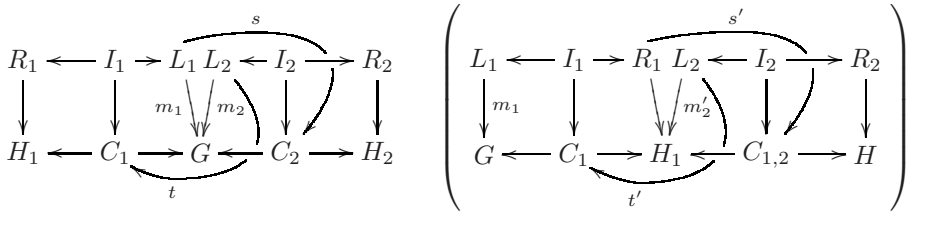
Definition 1 (Productions and rewriting). *A* DPO production p *is a span of monomorphisms* $p = L \leftarrow l \prec K \succ r \rightarrow R$*, and a* (monic) match of p *is a monomorphisms* $m\colon L \succ m \rightarrow G$*.*

A DPO direct derivation diagram *or* derivation *is a diagram consisting of two pushout squares as shown to the right. It witnesses that* p rewrites G into H at match m*, and we write* $G \vDash\langle p,m\rangle\Rightarrow H$*.*



Note that, by virtue of the properties holding in adhesive categories, all arrows in the above right diagram are monic. Thus, in the remainder of the paper all mentioned arrows are assumed to be monic, unless stated otherwise.

Definition 2 (Parallel and sequential independence). *Let us consider two productions* $p_i = L_i \leftarrow l_i - I_i - r_i \rightarrow R_i$ *for* $i \in \{1,2\}$*, and the two derivations* $H_1 \Leftarrow\langle m_1,p_1\rangle\dashv G \vDash\langle p_2,m_2\rangle\Rightarrow H_2$ *(*$G \vDash\langle p_1,m_1\rangle\Rightarrow H_1 \vDash\langle p_2,m_2'\rangle\Rightarrow H$*) shown below.*



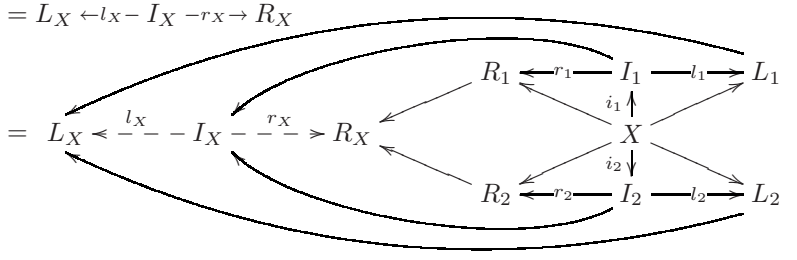
Then the two derivations are parallel independent (sequential independent) *if there exist morphisms* s *and* t *(*s' *and* t'*) such that they commute in the composed diagrams, as shown above.*

In the case of parallel independence, when thinking of the objects I_1, L_1, L_2, and I_2 as subobjects[1] of G, then the two derivations are independent if and only if both inclusions $L_1 \cap L_2 \subseteq I_1$ and $L_1 \cap L_2 \subseteq I_2$ hold, i.e. if $L_1 \cap L_2 = I_1 \cap I_2$; similarly for sequential independence: the illustrated derivations are independent if and only if $R_1 \cap L_2 = I_1 \cap I_2$.

As formulated precisely in Theorem 4, given two parallel independent derivations, it is possible to execute them in parallel, using a corresponding *parallel production*. As the latter concept is central to this paper, we define it in full detail. In its definition, $A_1 +_X A_2$ denotes pushout objects, e.g. $A_1 - a_1' \rightarrow (A_1 +_X A_2) \leftarrow a_2' - A_2$ would be the pushout of some span $A_1 \leftarrow a_1 - X - a_2 \rightarrow A_2$. Now, for any pair of arrows $g_1 \colon A_1 \rightarrow B$ and $g_2 \colon A_2 \rightarrow B$, $[g_1, g_2]_X \colon A_1 +_X A_2 \rightarrow B$ denotes the uniquely induced arrow. Further, for any span $B_1 \leftarrow f_1 - A_1 \leftarrow a_1 - X - a_2 \rightarrow A_2 - f_2 \rightarrow B_2$ with pushout $B_1 - b_1' \rightarrow (B_1 +_X B_2) \leftarrow b_2' - B_2$, $f_1 +_X f_2 \colon (A_1 +_X A_2) \rightarrow (B_1 +_X B_2)$ similarly denotes the uniquely induced arrow.

Definition 3 (Parallel productions). *Let* $p_1 = L_1 \leftarrow l_1 - I_1 - r_1 \rightarrow R_1$ *and* $p_2 = L_2 \leftarrow l_2 - I_2 - r_2 \rightarrow R_2$ *be productions, and let* $I_1 \leftarrow i_1 - X - i_2 \rightarrow I_2$ *be a span of morphisms. The* parallel production of p_1 and p_2 over $I_1 \leftarrow X \rightarrow I_2$ *is*

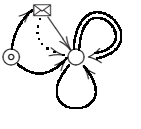
$$p_1 +_X p_2 = (L_1 +_X L_2) \leftarrow l_1 +_X l_2 - (I_1 +_X I_2) - r_1 +_X r_2 \rightarrow (R_1 +_X R_2)$$
$$= L_X \leftarrow l_X - I_X - r_X \rightarrow R_X$$



Theorem 4 (Church-Rosser vs. parallelism). *For any pair of productions* $p_1 = L_1 \leftarrow l_1 - I_1 - r_1 \rightarrow R_1$ *and* $p_2 = L_2 \leftarrow l_2 - I_2 - r_2 \rightarrow R_2$*, the following statements are equivalent for certain* $m_1' \colon L_1 \rightarrow H_2$ *and* $m_2' \colon L_2 \rightarrow H_1$*.*

1. *There are parallel independent derivations* $H_1 \Leftarrow \langle m_1, p_1 \rangle \Rightarrow G \Rightarrow \langle p_2, m_2 \rangle \Rightarrow H_2$*.*
2. *There are sequential independent derivations* $G \Rightarrow \langle p_1, m_1 \rangle \Rightarrow H_1 \Rightarrow \langle p_2, m_2' \rangle \Rightarrow H$*.*
3. *There are sequential independent derivations* $G \Rightarrow \langle p_2, m_2 \rangle \Rightarrow H_2 \Rightarrow \langle p_1, m_1' \rangle \Rightarrow H$*.*
4. *There is a derivation* $G \Rightarrow \langle (p_1 +_X p_2), [m_1, m_2]_X \rangle \Rightarrow H$ *with a parallel production* $p_1 +_X p_2$ *over the pullback* $I_1 \leftarrow X \rightarrow I_2$ *of the cospan* $I_1 - m_1 \circ l_1 \rightarrow G \leftarrow m_2 \circ l_2 - I_2$*.*

Example 5. Consider the category **Graph** $\downarrow T$ of typed graphs over the graph T shown to the right. Circled nodes ◎ represent users that want to send messages ✉ via a network where a network consists of nodes ○ and two different kinds of edges: a double arrow ⇒ represents a channel of unbounded capacity, while a single arrow → models a channel of capacity one.



[1] A suboject L of G is a monomorphism from L to G, and the intersection of two subobjets is achieved by taking pullbacks.

An edge from a message (user) to a node represent the fact that the message (user) is located at that node. The dotted arrow from a message to a node means that the message is addressed to that node, while an edge from a user to a message represents the fact that the user is the author of that message.

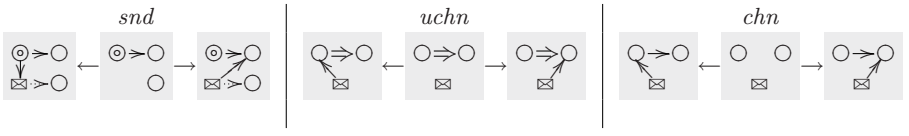
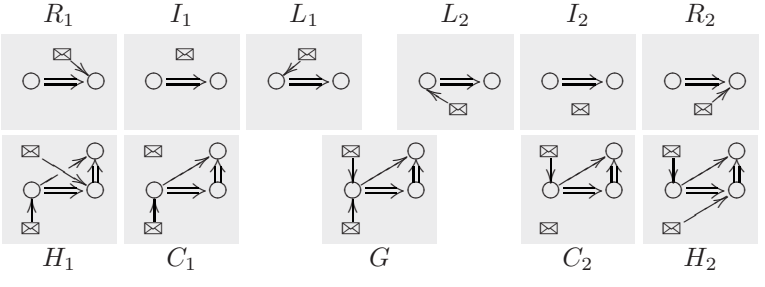


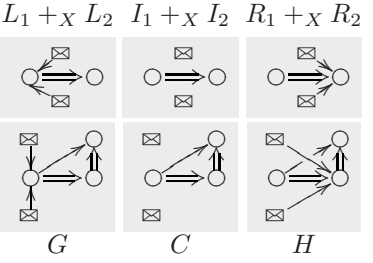
Fig. 1. Rewriting Rules

Consider the rule *snd* of the rewriting system that is shown in Fig. 1: if a user would like to deliver a message to a certain destination (singled out by the tip of the dotted arrow) while being located at another node, then the user can *send* the message by posting it via the network at the node he or she is located at. Then the message is *channeled* through the network, as modeled by the rules *chn* and *uchn*. When both edges of a message point to the same node, then the message has arrived at the destination.



As an example of parallel independent derivations, consider the graph G depicted above: it represents a small network with three nodes, two unbounded channels, one bounded channel and two messages on the same node. Both derivations use the production *uchn*, but in the first instance, on the left, the matching morphism $m_1 \colon L_1 \to G$ maps the message of L_1 into the upper one of G, while in the second instance, on the right, $m_2 \colon L_2 \to G$ maps the message into the lower one. Thus the first derivation moves the upper message, while the second moves the lower one. To see that the two derivations are parallel independent, consider the morphism $s \colon L_1 \to C_2$ that maps the message of L_1 into the upper message of C_2, and the morphism $t \colon L_2 \to C_1$ that maps the message of L_2 into the lower message of C_1.

We can also build a parallel rule by taking as common sub-object of I_1 and I_2 the graph $X = \bigcirc\!\!\Rightarrow\!\!\bigcirc$. The resulting production, namely $uchn +_X uchn$, is shown in the upper row to the right: in one single step, two messages can cross a channel of unbound capacity. The graph G can also be rewritten using this rule, which moves the two messages simultaneously.



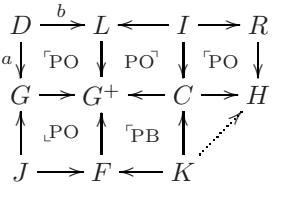
A channel of capacity one cannot transmit two messages in one step, though.

2.2 Borrowed Contexts

The DPO rewriting formalism models the dynamics of a (closed) system in isolation, i.e., without considering its interactions with the environment. As introduced in [1], DPOBC is an extension of DPO such that, instead of manipulating single objects G, arrows $J \to G$ are considered, where the domain J represents the interface of the (open) system G.

Whereas in DPO derivations the left-hand side L of a production must occur completely in G, in a DPOBC derivation it is possible that only part of the object L might occur in G; the missing part of the left hand side can be supplied by the environment at J, thus completing the partial match of L to a total one. That part of the left hand side of the rule, that was provided by the environment to allow a total match of L, is called the *borrowed context*.

Definition 6 (DPOBC derivations, transformations and transitions) *Given a production $p = L \leftarrow l- I -r\to R$, a* DPO *direct derivation diagram with borrowed context, in short* BC *derivation, is a diagram of as shown to the right where the marked squares are either pushouts* (PO) *or pullbacks* (PB)*. If there exists a diagram of this form, then p* transforms $J \to G$ via the partial match $G \leftarrow a- D -b\to L$ to $K \to H$*, written*



$$J \to G \models_{\langle p,a\text{-}D\text{-}b\rangle} \Rightarrow K \to H.$$

If instead we want to focus on the interaction with the environment we say that $J \to G$ makes a transition *with borrowed context $J \to F \leftarrow K$ and becomes $K \to H$, written*

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

The roles of the squares in the diagram above can be thought of as follows: the upper left-hand pushout square ${}^{D}_{G}\!\downarrow\!{}^{\rightarrow}_{\rightarrow}\!\downarrow\!{}^{L}_{G+}$ glues the left-hand side L and the object G together using the partial match $G \leftarrow a- D -b\to L$, which maps (part of) the left-hand side L onto G. The resulting pushout object G^+ allows for a total match of L and can be rewritten as in the standard DPO approach, which produces the two pushout squares ${}^{L}_{G+}\!\downarrow\!{}^{\leftarrow}_{\leftarrow}\!\downarrow\!{}^{I}_{C}$ and ${}^{I}_{C}\!\downarrow\!{}^{\rightarrow}_{\rightarrow}\!\downarrow\!{}^{R}_{H}$ in the upper

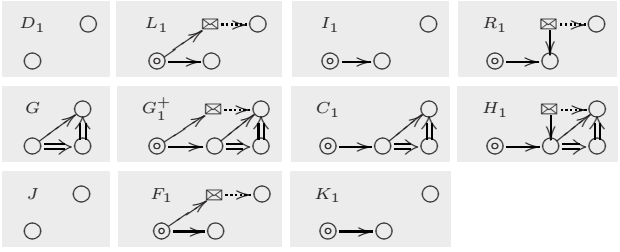


Fig. 2. A borrowed context derivation

row. The left-most bottom pushout ${}^{G}_{J}\!\uparrow\!{}^{\rightarrow}_{\rightarrow}\!\uparrow{}^{G+}_{F}$ glues the additional resources of the borrowed context $J \to F$, which are used to obtain the total match L into G^+, to the object G using the interface $J \to G$. Finally, the interface for the resulting object H and the second part of the label $J \to F \leftarrow K$ is obtained by intersecting the borrowed context F and the object C, i.e. by taking the pullback ${}^{G+}_{F}\!\uparrow\!{}^{\leftarrow}_{\leftarrow}\!\uparrow{}^{C}_{K}$. At the end, the new interface K includes what is preserved of the old interface and borrowed resources; finally, in the resulting $K \to H$ the object H additionaly contains everything produced by the "internal" DPO derivation.

Example 7. An example of a DPOBC derivation using the production *snd* can be found in Fig. 2. Here and in the remainder of the paper we do not explicitly describe the morphisms between graphs: we always assume that morphisms preserve the position of nodes and edges in the graphs. Moreover, we often denote an object in a specific derivation with the same (possibly indexed) identifier used in Def. 6. For example, the morphism $J \to G$ in Fig. 2 maps the leftmost and the topmost node of J into the leftmost and the topmost node of G, respectively.

The graph G represents a small network. Those nodes of this net occurring in the interface (the leftmost and the topmost one) are called *open*, since they are open to access from the environment, while each node that does not belong to the interface (e.g., the bottom right one in G) is called *closed*, as it has not been disclosed to the environment (yet).

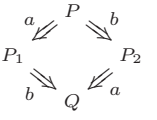
Observe that the graph G does not contain any messages ✉and hence cannot perform any DPO derivation on its own. However, if there are users in the environment that want to send messages between the open nodes, messages are posted at the open nodes. This is what happens in the derivation of Fig. 2: a user wants to deliver one message to the upper open node and hence posts it at the lower open node. Note that (ordinary) users can only post at open nodes, which corresponds to the fact that every (ordinary) Internet user can only access the Internet via a provider.

In Section 4 we show how sequential and parallel independence can be lifted to the realm of DPOBC rewriting. In the next section we first present some fundamental observations about the nature of labels obtained via borrowed contexts.

3 A Discussion on Concurrency and Labels

In the standard DPO approach, two derivations are parallel independent if each one does not consume the resources needed by the other one. In the case of DPOBC derivations, due to the possibility to borrow resources from the environment, the situation is more complex. The notion of independence that we are looking for should guarantee that a pair of branching transitions may be executed simultaneously and can be joined to complete a Church-Rosser square, and also that the labels of the derivations are "well-related", which means that no superfluous interaction with the environment takes place.

In *transition systems with independence* [10], a formalism based on LTSs equipped with an independence relation among transitions, the labels represent events: in parallel and sequential independent transitions, the labels are related as shown to the right.



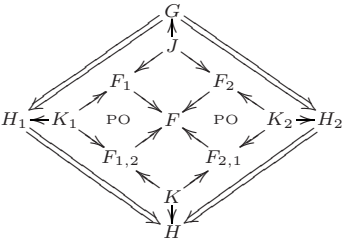
In such a Church-Rosser square, the two transitions $P \Rightarrow P_1$ and $P_2 \Rightarrow Q$ are the same event, namely a, and similarly $P \Rightarrow P_2$ and $P_1 \Rightarrow Q$ are the event b. To get an analogy to DPOBC, we must think of events as productions. However this analogy is very rough, as the BC labels really are contexts, and, in general, the productions themselves are not part of the labels any more. Hence we cannot expect that the labels on parallel sides of the Church-Rosser square are actually identical. Indeed, the source and target state of a DPOBC derivation usually have different interfaces, and hence all labels of transitions originating from the former are different from those departing from the latter.

However, recall that (borrowed) contexts can be composed via pushouts (since they are arrows of a bicategory of cospans [11]). Hence one would expect that the composition of the two composable contexts on the left and on the right yields the same result, namely the diagonal: in other terms, a DPOBC Church-Rosser square should satisfy what we call the *weak diamond* property.

Definition 8 (Weak diamond property) *Given four transitions as shown in the commutative diagram to the right, then they satisfy the* weak diamond property *if there is a derivation*

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

using a parallel rule and additionally the leftmost and rightmost square are pushouts.



The latter requirement means that the composition of the labels of the two transitions on the left and the two transitions on the right are equal to $J \to F \leftarrow K$, which is the label of the parallel derivation along the diagonal of the square.

However, the weak diamond property does not ensure that the labels of the joining transitions and of the parallel one do not borrow "extra"-resources. As an example consider the pair of branching derivations in Fig. 3, witnessing transitions $J \to G \xrightarrow{J \to F_1 \leftarrow K_1} K_1 \to H_1$ and $J \to G \xrightarrow{J \to F_2 \leftarrow K_2} K_2 \to H_2$, where the former relays the upper message and the latter the lower one.

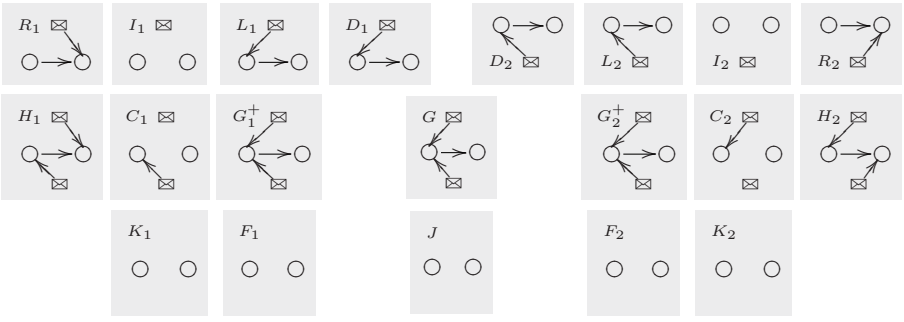
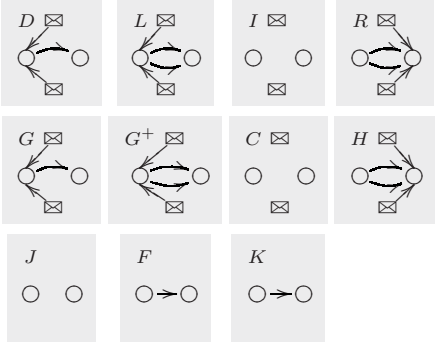


Fig. 3. Two non-parallel independent derivations

Note that both transitions use the same channel of capacity one, and this is also the reason why they turn out not to be parallel independent. In spite of this conflict we can construct the parallel derivation shown to the right, which borrows an extra channel from the environment. As shown in Fig. 4, the two derivations of Fig. 3 can be completed to a square satisfying the weak diamond property.



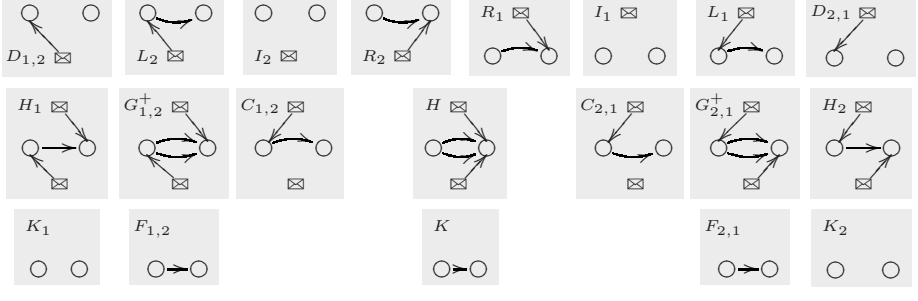


Fig. 4. The joining derivations of the weak diamond

Note that the label $K_2 \to F_{2,1} \leftarrow K$ in Fig. 4 is "bigger" than the corresponding $J \to F_1 \leftarrow K_1$ in Fig. 3 (and similarly for the other opposite labels of the square): the former consumes more environmental resources than the latter.

In order to guarantee that the joining derivations, as well as the parallel one, do not consume more resources than the two branching derivations, we have to require the following conditions.

Definition 9 (Strong diamond property). *The transitions in Def. 8 satisfy the* strong diamond property *if, additionally, the label $F_1 \to F \leftarrow F_2$ consists of a pair of jointly epic morphisms.*

Intuitively this means that all the resources that are borrowed by the parallel transition $J \to G \xrightarrow{J \to F \leftarrow K} K \to H$ have already been borrowed by one of the branching transitions. Since the composition of the contexts on the left (right) derivations must be equal to the parallel one, the joining derivations do not consume more than the branching ones.

However, also the strong diamond property is in some sense too weak. Indeed, it could be the case that the parallel derivation borrows less resources than the branching ones. In order to guarantee that the parallel transition consumes exactly the same resources of its two components, we have to require that the label F of the parallel derivation is the pushout of the two branching ones along the common interface, i.e., $F = F_1 +_J F_2$.

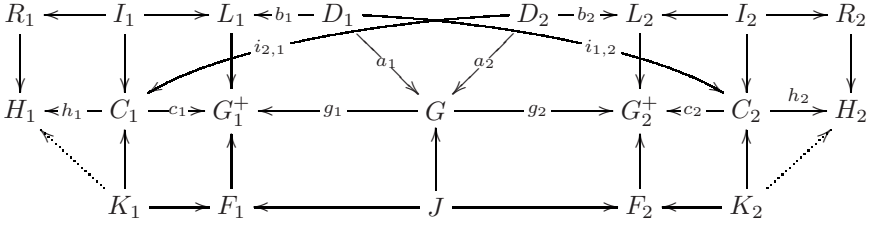
Definition 10 (Strict diamond property). *The transitions in Def. 8 satisfy the* strict diamond property *if, additionally, the upmost square of the diamond is a pushout.*

Note that the strict diamond property implies the strong one.

4 Parallel and Sequential Independence

In this section we introduce parallel and sequential independence for DPOBC rewriting, and we prove that these notions guarantee the satisfaction of the Church-Rosser property and the existence of a parallel transformation respecting the strict diamond property. In the next section we show that the vicevers is not always true, that is, a parallel transformation is not always decomposable into derivations satisfying the strict diamond property.

Definition 11 (Parallel independent derivations). *Let $J \to G$ be an object with interface, and moreover let $J \to G \vDash\langle p_1,a_1\text{-}D_1\text{-}b_1\rangle\Rightarrow K_1 \to H_1$ and $J \to G \vDash\langle p_2,a_2\text{-}D_2\text{-}b_2\rangle\Rightarrow K_2 \to H_2$ be two* DPOBC *derivations. These are* parallel independent *if for every pair of witnessing derivation diagrams as shown below there exist morphisms $i_{1,2}\colon D_1 \to C_2$ and $i_{2,1}\colon D_2 \to C_1$ which satisfy the equations $c_1 \circ i_{2,1} = g_1 \circ a_2$ and $c_2 \circ i_{1,2} = g_2 \circ a_1$.*

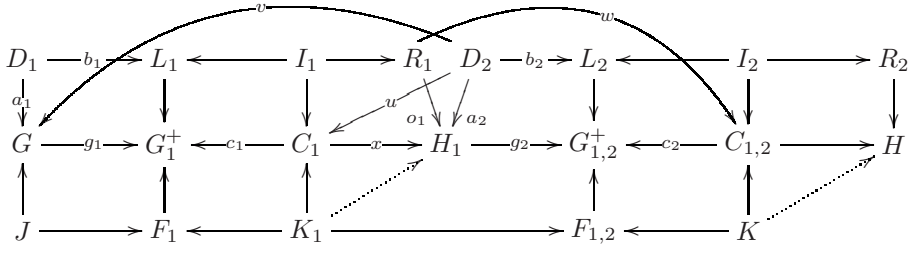


Intuitively, the existence of $i_{1,2}\colon D_1 \to C_2$ ensures that D_1 (the part of G that is needed to perform the first derivation) is left untouched by the second derivation. Thm. 16 shows that this condition is sufficient to guarantee the strict diamond property, but here we want to point out that it is also necessary for the strong diamond property. Indeed, if the first derivation would consume part of what is consumed by the second, then the parallel production could not be applied without borrowing more resources from the environment.

The above definition specializes to the standard DPO notion of parallel independence, since then b_1 and b_2 (and thus g_1 and g_2) would be identities.

Example 12. Consider the derivation in Fig.2. It is parallel independent with itself. Given another one copy of the derivation where all the graphs are labeled with D_2, L_2, $I_2 \ldots$ one can easily construct the morphism $i_{2,1}\colon D_2 \to C_1$ mapping the leftmost and the topmost node of D_2 (recall that D_2 is the same of D_1) onto the leftmost and the topmost node of C_1, respectively. Analogously for $i_{1,2}\colon D_1 \to C_2$. Intuitively, these derivations are parallel independent because they just add some structure to the nets, i.e., two users from the environment put two messages in the same node, and clearly this could happen concurrently. The derivations in Fig. 3 instead are not parallel independent, since the channel in D_1 does not occur in C_2.

Definition 13 (Sequential independent derivations). *Let $J \to G$ be an object with interface, and moreover let $J \to G \models\langle p_1,a_1\text{-}D_1\text{-}b_1\rangle\Rightarrow K_1 \to H_1$ and $K_1 \to H_1 \models\langle p_2,a_2\text{-}D_2\text{-}b_2\rangle\Rightarrow K \to H$ be two* DPOBC *derivations. These are* sequential independent *if for any witnessing pair of derivation diagrams as shown below, there exist morphisms $u\colon D_2 \to C_1$, $v\colon D_2 \to G$ and $w\colon R_1 \to C_{1,2}$ which satisfy the equations $c_1 \circ u = g_1 \circ v$, $c_2 \circ w = g_2 \circ o_1$ and $a_2 = x \circ u$.*



Intuitively, the existence of v requires that what is needed by the second derivation (i.e., D_2) occurs in G and thus it is not added by the borrowed context. The existence of u requires that D_2 is not produced by the first derivation, while the existence of w guarantees that the second rule does not consume anything that the first one has produced.

Note that the definition above subsumes the notion of sequential independence in DPO rewriting, since for DPO derivations the object G_1^+ coincides with G, and thus the the existence of u implies the existence of v.

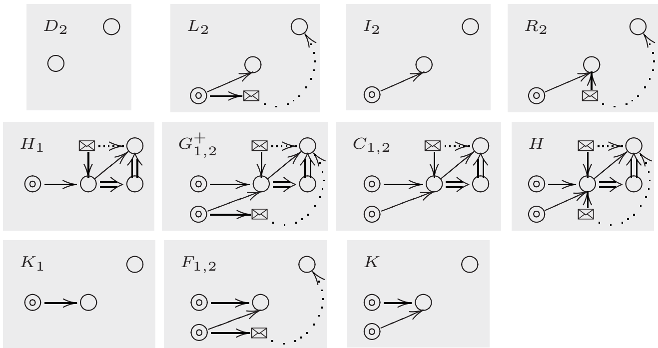


Fig. 5. A derivation that is sequential independent with that of Fig. 2

Example 14. The DPOBC derivations shown in Fig. 2 and Fig. 5 are sequential independent. One can safely take u, v, w as those morphisms preserving the position of nodes and edges into the respective graphs. Intuitively, the first derivation takes from the environment a user sending a message, and the second derivation takes another user sending another message. Clearly both derivations are sequential independent, and we can switch the order in which users are added (this is formally stated in Thm. 15).

Now, we can state our main result.

Theorem 15 (Church-Rosser and parallelism for Borrowed Contexts) *For any pair of productions* $p_1 = L_1 \xleftarrow{l_1} I_1 \xrightarrow{r_1} R_1$ *and* $p_2 = L_2 \xleftarrow{l_2} I_2 \xrightarrow{r_2} R_2$, *the following are equivalent for certain* $a'_1 \colon D_1 \to H_2$ *and* $a'_2 \colon D_2 \to H_1$.

1. *There are parallel independent derivations*

$$K_1 \to H_1 \Leftarrow\langle b_1\text{-}D_1\text{-}a_1, p_1\rangle\dashv J \to G \text{ and } J \to G \models\langle p_2, a_2\text{-}D_2\text{-}b_2\rangle\Rightarrow K_2 \to H_2.$$

2. *There are sequential independent derivations*

$$J \to G \models\langle p_1, a_1\text{-}D_1\text{-}b_1\rangle\Rightarrow K_1 \to H_1 \text{ and } K_1 \to H_1 \models\langle p_2, a'_2\text{-}D_2\text{-}b_2\rangle\Rightarrow K \to H.$$

3. *There are sequential independent derivations*

$$J \to G \models\langle p_2, a_2\text{-}D_2\text{-}b_2\rangle\Rightarrow K_2 \to H_2 \text{ and } K_2 \to H_2 \models\langle p_1, a'_1\text{-}D_1\text{-}b_1\rangle\Rightarrow K \to H.$$

Moreover, 1 implies that for $X = D_1 \cap D_2$,

4. *there is a parallel derivation*

$$J \to G \models\langle p_1 +_X p_2, [a_1, a_2]_X\text{-}(D_1 +_X D_2)\text{-}b_1 +_X b_2\rangle\Rightarrow K \to H.$$

The main difference with respect to parallelism for DPO is that the fourth statement does not imply the others, as we discuss in the next section. Now, we want

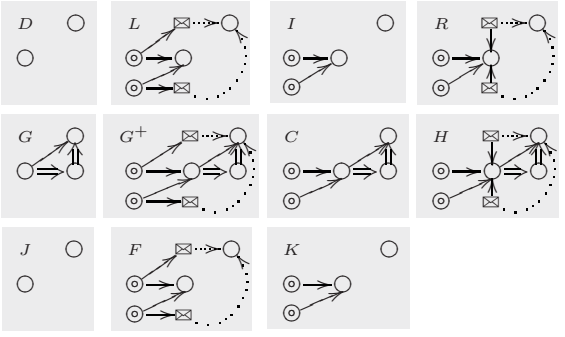


Fig. 6. Parallel derivation: two users put two messages in the same node

to point out that the opposite derivations in the Church-Rosser square employ exactly the same $b_1 \colon D_1 \to L_1$ and $b_2 \colon D_2 \to L_2$. Intuitively this means that the same resources of the system are used and thus the same are borrowed. This is analogous to LTSs with independence (discussed in Section 3), where opposite transitions in a concurrent diamond are labeled with the same event.

Theorem 16 (Church-Rosser and parallelism on labels). *The labels on the derivations described in the first three items of Thm. 15 form a diagram as the one shown in Def. 8 where the leftmost, the rightmost and the topmost squares are pushouts and the lowest is a pullback.*

Therefore our construction respects the strict diamond property. Moreover, since the lowest square of the diamond turns out to be a pullback, the diamond is a GRPO in the bicategory of cospans. Indeed, in [12] it is shown that every strong diamond where the lowest square is a pullback is a GRPO.

Example 17. In Ex. 12 we have shown that the derivation in Fig. 2 is parallel independent with itself. The joining derivations closing the Church-Rosser square

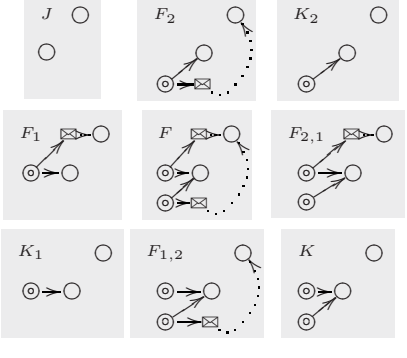


Fig. 7. A strict diamond

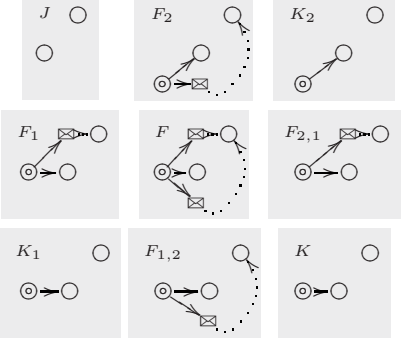


Fig. 8. A strong diamond

are two copies of the derivation shown in Fig. 5 (that is sequential independent with the one in Fig. 2), while the corresponding parallel derivation is shown in Fig. 6. This employs the parallel rule $snd +_X snd$ where X is the intersection of D_1 and D_2 over G, i.e., $X = \bigcirc\,\bigcirc$ (recall that D_2 is equal to D_1). This rule essentially states that two users can leave two messages at the same node in one single step. Thus the derivation borrows from the environment two users with two messages. These are exactly the same resources that are borrowed by performing sequentially first the derivation in Fig. 2 and then the one in Fig. 5, i.e., labels respect the strict diamond property as shown in Fig. 7.

5 From Parallel to Parallel Independent Derivations

The item 4 of Thm. 15 states that for any pair of parallel independent derivations there exists a parallel derivation whose labels enjoy the strict diamond property. Unfortunately, the converse does not hold: there exist derivations using a parallel rule that cannot be decomposed into derivations forming a strict diamond.

Consider the derivation in Fig. 9, for parallel production $snd +_Y snd$ over $Y = \circledcirc\,\bigcirc\,\bigcirc$. This production allows a user to leave two messages in one step. Thus the derivation borrows one user with two messages as context. However, there exists no strict diamond with label $J \to F \leftarrow K$ of Fig. 9 as parallel label.

Indeed, any DPOBC derivation (by the rules in Fig. 1) contains at most one message. In order to have a user owning two messages as parallel label, we need to identify the users. Since the label F of the derivation (according to the strict diamond property) must be the pushout of F_1 and F_2 along J; and since the interface J does not contain any user, then users cannot be identified in F.

Intuitively, strict diamonds can not merge borrowed resources, while parallel derivations can. Indeed, given a pair of parallel independent transformations, we can construct a parallel transformation by gluing the productions over a graph containing resources common to both F_1 and F_2 that do not occur in G. Since these resources are merged in the left hand side of the parallel production, they are merged also in F. And since the resources are not in J, F is not the pushout of F_1 and F_2 along J. Thus, the resulting labels do not form a strict diamond.

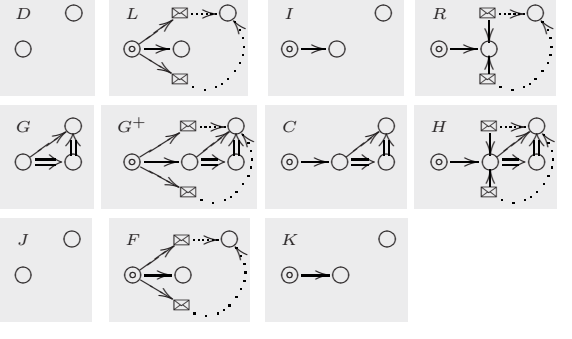


Fig. 9. Parallel derivation: one user sends two messages at the same node

More concretely, the parallel derivation in Fig. 9 is constructed by taking the two copies of the derivations in Fig. 2 (since it is parallel independent with itself) and by gluing the productions over J: this graph contains a user, that is an item occurring in F_1 and F_2, but not in G.

It seems thus clear that, in order to guarantee the converse of the parallelism theorem, we should move from strict to strong diamonds, since requiring that the morphisms $F_1 \to F$ and $F_2 \to F$ in the diamond are jointly epic allows to merge resources in F_1 and F_2 that are not in J. Indeed, it is worth noting that the labels of the derivation of the above example form a strong diamond (shown in Fig. 8) where the branching derivations are those of Fig. 2. So far, however, we were not able to prove such a correspondence, and we leave it for future work.

6 Conclusions and Future Work

Our work introduces the notions of parallel and sequential independence for DPOBC derivations, generalizing the corresponding concepts for DPO derivations. It then establishes a Church-Rosser theorem for independent derivations.

The crucial problem of lifting these properties and results from DPO to DPOBC rewriting is taking into account the borrowing of resources from the environment, and providing a precise description of the complex interactions between labels of independent transitions. Indeed, given two branching derivations, it is often the case that there exists a pair of joining derivations, even if these latter might differ with respect to resource usage when compared with the branching ones. With the strict diamond property we establish conditions to require in order to have that the joining derivations use neither more nor less environmental resources.

Our results provide the basis for further studies on the concurrent behaviour of open systems modeled via DPOBC rewriting. Since the standard DPOBC bisimilarity is interleaving, we plan to refine it to a true concurrent bisimilarity and compare it with similar behavioural equivalences proposed in the literature, e.g. history preserving bisimilarity [13,14] and related work on Petri nets [15,16].

For this, we plan to study the behavioural equivalence that arises by applying the standard DPOBC bisimilarity not to a given set of productions but to its *parallel closure*, which contains for every two rules also all their parallel compositions: using the standard results, this yields a bisimulation congruence.

As far as true concurrent semantics is concerned, we would like to explore the possibility to provide a notion of graph process for DPOBC rewriting, along the line of the theory developed for DPO rewriting [17]: in this respect, establishing the Church-Rosser property represents the cornerstone of such a development.

References

1. Ehrig, H., König, B.: Deriving bisimulation congruences in the DPO approach to graph rewriting with borrowed contexts. Mathematical Structures in Computer Science 16(6), 1133–1163 (2006)
2. Leifer, J., Milner, R.: Deriving bisimulation congruences for reactive systems. In: Palamidessi, C. (ed.) CONCUR 2000. LNCS, vol. 1877, pp. 243–258. Springer, Heidelberg (2000)

3. Baldan, P., König, B., Ehrig, H.: Composition and decomposition of DPO transformations with borrowed context. In: Corradini, A., Ehrig, H., Montanari, U., Ribeiro, L., Rozemberg, G. (eds.) ICGT 2006. LNCS, vol. 4178, pp. 153–167. Springer, Heidelberg (2006)
4. Gadducci, F.: Term graph rewriting and the π-calculus. In: Ohori, A. (ed.) APLAS 2003. LNCS, vol. 2895, pp. 37–54. Springer, Heidelberg (2003)
5. Gadducci, F., Lluch-Lafuente, A.: Graphical encoding of a spatial logic for the π-calculus. In: Mossakowski, T., Montanari, U., Haveraaen, M. (eds.) CALCO 2007. LNCS, vol. 4624, pp. 209–225. Springer, Heidelberg (2007)
6. Bonchi, F., Gadducci, F., König, B.: Process bisimulation via a graphical encoding. In: Corradini, A., Ehrig, H., Montanari, U., Ribeiro, L., Rozemberg, G. (eds.) ICGT 2006. LNCS, vol. 4178, pp. 168–183. Springer, Heidelberg (2006)
7. Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Heckel, R., Löwe, M.: Algebraic approaches to graph transformation I: Basic concepts and double pushout approach. In: Rozenberg, G. (ed.) Handbook of Graph Grammars and Computing by Graph Transformation, vol. 1, pp. 163–245. World Scientific, Singapore (1997)
8. Habel, A., Müller, J., Plump, D.: Double-pushout graph transformation revisited. Mathematical Structures in Computer Science 11, 637–688 (2001)
9. Lack, S., Sobociński, P.: Adhesive and quasiadhesive categories. Theoretical Informatics and Applications 39, 511–545 (2005)
10. Winskel, G., Nielsen, M.: Models for concurrency. In: Abramsky, S., Gabbay, D.M., Maibaum, T.S. (eds.) Handbook of Logic in Computer Science, vol. 4, pp. 1–148. Oxford University Press, Oxford (1992)
11. Sassone, V., Sobociński, P.: Reactive systems over cospans. In: Logic in Computer Science, pp. 311–320. IEEE Computer Society Press, Los Alamitos (2005)
12. Sobociński, P.: Deriving bisimulation congruences from reduction systems. PhD thesis, BRICS, Department of Computer Science, University of Aaurhus (2004)
13. Rabinovich, A.M., Trakhtenbrot, B.A.: Behaviour structures and nets. Fundamenta Informatica 11(4), 357–404 (1988)
14. van Glabbeek, R.J., Goltz, U.: Equivalence notions for concurrent systems and refinement of actions. In: Kreczmar, A., Mirkowska, G. (eds.) MFCS 1989. LNCS, vol. 379, pp. 237–248. Springer, Heidelberg (1989)
15. Baldan, P., Corradini, A., Ehrig, H., Heckel, R., König, B.: Bisimilarity and behaviour-preserving reconfiguration of open petri nets. In: Mossakowski, T., Montanari, U., Haveraaen, M. (eds.) CALCO 2007. LNCS, vol. 4624, pp. 126–142. Springer, Heidelberg (2007)
16. Nielsen, M., Priese, L., Sassone, V.: Characterizing behavioural congruences for Petri nets. In: Lee, I., Smolka, S.A. (eds.) CONCUR 1995. LNCS, vol. 962, pp. 175–189. Springer, Heidelberg (1995)
17. Corradini, A., Montanari, U., Rossi, F.: Graph processes. Fundamenta Informaticae 26, 241–265 (1996)