Exploring Conflict Reasons for Graph Transformation Systems

Extended Version

Leen Lambers[1] Jens Kosiol[2] and Daniel Strüber[3] Gabriele Taentzer[2]

[1] Hasso-Plattner-Institut, Universität Potsdam, Potsdam, Germany,
leen.lambers@hpi.de
[2] Philipps-Universität Marburg, Marburg, Germany,
{kosiolje, taentzer}@informatik.uni-marburg.de
[3] Chalmers University, University of Gothenburg, Gothenburg, Sweden,
danstru@chalmers.se

Abstract. Conflict and dependency analysis (CDA) is a static analysis for the detection of conflicting and dependent rule applications in a graph transformation system. Recently, granularity levels for conflicts and dependencies have been investigated focussing on delete-use conflicts and produce-use dependencies. A central notion for granularity considerations are (minimal) conflict and dependency reasons. For a rule pair, where the second rule is non-deleting, it is well-understood based on corresponding constructive characterizations how to efficiently compute (minimal) conflict and dependency reasons. We further explore the notion of (minimal) conflict reason for the general case where the second rule of a rule pair may be deleting as well. We present new constructive characterizations of (minimal) conflict reasons distinguishing delete-read from delete-delete reasons. Based on these constructive characterizations we propose a procedure for computing (minimal) conflict reasons and we show that it is sound and complete.

Keywords: Graph Transformation · Conflict analysis · Static analysis

1 Introduction

Graph transformation [1] is a formal paradigm with many applications. A graph transformation system is a collection of graph transformation rules that, in union, serve a common purpose. For many applications (see [2] for a survey involving 25 papers), it is beneficial to know all conflicts and dependencies that can occur for a given pair of rules. A conflict is a situation in which one rule application renders another rule application inapplicable. A dependency is a situation in which one rule application needs to be performed such that another rule application becomes possible. For a given rule set, a *conflict and dependency analysis* (CDA) technique is a means to compute a list of all pairwise conflicts and dependencies.

Inspired by the related concept from term rewriting, *critical pair analysis* (CPA, [3]) has been the established CDA technique for over two decades. CPA reports each conflict as a critical pair[1], that is, a minimal example graph together with a pair of rule applications from which a conflict arises. Recently it has been observed that applying CPA in practice is problematic: First, computing the critical pairs does not scale to large rules and rule sets. Second, the results are often hard to understand; they may contain many redundant conflicts that differ in subtle details only, typically not relevant for the use case at hand.

To address these drawbacks, in previous work, we presented the *multi-granular conflict and dependency analysis* (MultiCDA [5, 2]) for graph transformation systems. It supports the computation of conflicts on a given granularity level: On binary granularity, it reports if a given rule pair contains a conflict at all. On coarse granularity, it reports *minimal conflict reasons*, that is, problematic rule fragments shared by both rules that may give rise to a conflict. Fine granularity is, roughly speaking, the level of granularity provided by critical pairs. (In the terminology of our recent work [4], we here focus on different levels of overlap granularity with fixed coarse context granularity.) We showed that coarse-grained results are more usable than fine-grained ones in a diverse set of scenarios and can be used to compute the fine-grained results much faster.

In this work, we address a major *current limitation of MultiCDA* [2]. The computation of conflicts is only exact for cases where the second rule of the considered rule pair is non-deleting. In this case, it is well-understood how to compute conflicts efficiently, using constructive characterisations of minimal conflict reasons (for coarse granularity) and conflict reasons (for fine granularity). In the other case, MultiCDA can provide an overapproximation of the actual conflicts, by replacing the second rule with its non-deleting variant. On the one hand, this overapproximation may contain conflicts which can never actually arise. On the other hand, the overapproximation does not distinguish conflicts for rule pairs where the second rule is non-deleting from those for rule pairs where the second rule is deleting (i.e. no distinction between delete-delete and delete-read). The first issue leads to a MultiCDA that may report false positives, whereas the latter issue causes the MultiCDA to report true positives without the desired level of detail. Therefore the overapproximation presents an obstacle to the understandability of the results and the usability of the overall technique.

In this paper, we come up with the foundations for an *improved MultiCDA* avoiding this limitation, thus delivering in all cases exact as well as detailed results. To this end we present new constructive characterizations of (minimal) conflict reasons for rule pairs where the second rule may be deleting, distinguishing in particular delete-read (dr) from delete-delete (dd) reasons. Based on these constructive characterizations we propose a basic procedure for computing dr/dd (minimal) conflict reasons and we show that it is sound and complete. In particular, we learn that we can reduce the computation of dr/dd minimal reasons to the constructions presented for the overapproximation [2]. Moreover, the construction of dr/dd reasons can reuse the results computed for minimal reasons, representing the basis for an efficient computation of fine-grained results

[1] For brevity, since all conflict-specific considerations in this paper dually hold for dependencies (see our argumentation in [4]), we omit talking about dependencies.

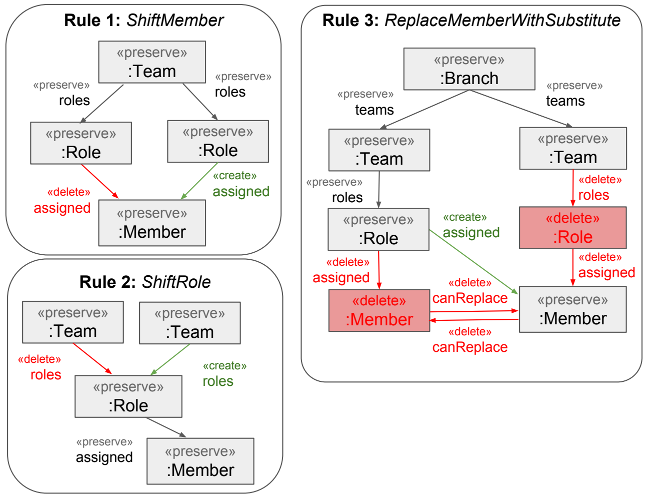


Fig. 1. Rules for running example in an integrated representation

based on coarse-grained ones. We illustrate our results using a running example modeling requirements for a project management software.

The rest of this paper is structured as follows: Section 2 introduces our running example. Section 3 revisits preliminaries. Section 4 introduces *delete-read* and *delete-delete* conflict reasons. Section 5 presents a new characterization of conflict reasons, accommodating both delete-read (dr) and delete-delete (dd) conflicts. Section 6 is devoted to the construction of conflict reasons based on the new characterizations. Section 7 discusses related work and concludes. We present some additional technicalities and proofs that are omitted from the main part of the paper in Appendices A and B.

2 Running example

In agile software development processes [6], enterprises quickly react to changes by flexibly adapting their team structures. Figure 1 introduces a set of rules describing requirements for a project management software. The rules are represented in the Henshin [7] syntax, using an integrated syntax with delete, create, and preserve elements. Delete and create elements are only contained in the LHS and RHS, respectively, whereas a preserved element represents an element that occurs both in the LHS and RHS.

The rules, focusing on restructuring and deletion cases for illustration purposes, stem from a larger rule overall set. The first two rules allow the project managers of a branch of the company to reassign roles and members between teams. Rule *ShiftMember* assigns a member to a different role in the same team. Rule *ShiftRole* moves a role and its assigned team member to a different team. The other rule deals with a team member leaving the company. Rule *ReplaceMemberWithSubstitute* (abbreviated to *ReplaceM* in what follows) removes a team member while filling the left role with a replacement team member, based on their shared expertise. The role of the replacement member in the existing project is deleted. Note that a variant of this rule, in which the existing role is not deleted, may exist as well.

Conflicts and dependencies between requirements such as those expressed with the rules from Fig. 1 can be automatically identified with a CDA technique. Doing so is useful for various purposes: To support project managers from different teams who may want to plan changes to the personnel structure as independently as possible. Or, for the software developers, to check whether conflicts and dependencies expected to arise actually do, thereby validating the correctness of the requirement specification. Therefore, we use this running example to illustrate the novel CDA concepts introduced in this paper.

3 Preliminaries

As a prerequisite for our exploration of conflict reasons, we recall the double-pushout approach to graph transformation as presented in [1]. Furthermore, we reconsider conflict notions on the transformation and rule level [8, 9, 2], including conflict reasons.

3.1 Graph Transformation and Conflicts

Throughout this paper we consider graphs and graph morphisms as presented in [1] for the category of graphs; all results can be easily extended to the category of typed graphs by assuming that each graph and morphism is typed over some fixed type graph TG.

Graph transformation is the rule-based modification of graphs. A *rule* mainly consists of two graphs: L is the left-hand side (LHS) of the rule representing a pattern that has to be found to apply the rule. After the rule application, a pattern equal to R, the right-hand side (RHS), has been created. The intersection K is the graph part that is not changed; it is equal to $L \cap R$ provided that the result is a graph again. The graph part that is to be deleted is defined by $L \setminus (L \cap R)$, while $R \setminus (L \cap R)$ defines the graph part to be created.

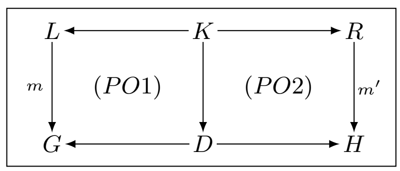
A *direct graph transformation* $G \overset{m,r}{\Longrightarrow} H$ between two graphs G and H is defined by first finding a graph morphism[2] m of the LHS L of rule r into G such that m is injective, and second by constructing H in two passes: (1) build $D := G \setminus m(L \setminus K)$, i.e., erase all graph elements that are to be deleted; (2) construct $H := D \cup m'(R \setminus K)$. The morphism m' has to be chosen such that a new copy of all graph elements that are to be created is added. It has been shown for graphs and graph transformations that r is applicable at m iff m fulfills the *dangling condition*. It is satisfied if all adjacent graph edges of a graph node to be

[2] A morphism between two graphs consists of two mappings between their nodes and edges being both structure-preserving w.r.t. source and target functions. Note that in the main text we denote inclusions by $\hookrightarrow$ and all other morphisms by $\rightarrow$.

deleted are deleted as well, such that D becomes a graph. Injective matches are usually sufficient in applications and w.r.t. our work here, they allow to explain constructions with more ease than for general matches. In categorical terms, a direct transformation step is defined using a so-called double pushout as in the following definition. Thereby step (1) in the previous informal explanation is represented by the first pushout and step (2) by the second one [1].

Definition 1 ((non-deleting) rule and transformation). *A* rule r *is defined by* $r = (L \overset{le}{\hookleftarrow} K \overset{ri}{\hookrightarrow} R)$ *with* $L, K,$ *and* R *being graphs connected by two graph inclusions. The non-deleting rule of* r *is defined by* $ND(r) = (L \overset{id_L}{\hookleftarrow} L \overset{ri'}{\hookrightarrow} R')$ *with* $(L \overset{ri'}{\hookrightarrow} R' \hookleftarrow R)$ *being the pushout of* (le, ri)*. Given rule* $r_1 = (L_1 \overset{le_1}{\hookleftarrow} K_1 \overset{ri_1}{\hookrightarrow} R_1)$*, square (1) in Figure 2 can be constructed as initial pushout over morphism* le_1*. It yields the* boundary graph B_1 *and the* deletion graph C_1*.*

A direct transformation $G \overset{m,r}{\Longrightarrow} H$ *which applies rule* r *to a graph* G *consists of two pushouts as depicted right. Rule* r *is* applicable *and the injective morphism* $m : L \to G$ *is called* match *if there exists a graph* D *such that* $(PO1)$ *is a pushout.*



Given a pair of transformations, a *delete-use conflict* [1] occurs if the match of the second transformation cannot be found anymore after applying the first transformation. Note that we do not consider delete-use conflicts of the second transformation on the first one explicitly. To get those ones as well, we simply consider the inverse pair of transformations. The following definition moreover distinguishes two cases [8]: (1) a *delete-read conflict* occurs if the match of the first transformation can still be found after applying the second one (2) a *delete-delete conflict* occurs if this is not the case, respectively.

Definition 2 (dr/dd conflict). *Given a pair of direct transformations* $(t_1, t_2) = (G \overset{m_1,r_1}{\Longrightarrow} H_1, G \overset{m_2,r_2}{\Longrightarrow} H_2)$ *applying rules* $r_1 : L_1 \overset{le_1}{\hookleftarrow} K_1 \overset{ri_1}{\hookrightarrow} R_1$ *and* $r_2 : L_2 \overset{le_2}{\hookleftarrow} K_2 \overset{ri_2}{\hookrightarrow} R_2$ *as depicted in Fig. 2.* Transformation pair (t_1, t_2) *is in* delete-use conflict *if there does not exist a morphism* $x_{21} : L_2 \to D_1$ *such that* $g_1 \circ x_{21} = m_2$*.* Transformation pair (t_1, t_2) *is in* dr conflict *if it is in delete-use conflict and if there exists a morphism* $x_{12} : L_1 \to D_2$ *such that* $g_2 \circ x_{12} = m_1$*.* Transformation pair (t_1, t_2) *is in* dd conflict *if it is in delete-use conflict and if there does not exist a morphism* $x_{12} : L_1 \to D_2$ *such that* $g_2 \circ x_{12} = m_1$*.*

3.2 Conflict Reasons

We consider delete-use conflicts between transformations where at least one deleted element of the first transformation is overlapped with some used element of the second transformation. This *overlap* is formally expressed by a span of graph morphisms between the deletion graph C_1 of the first rule, and the

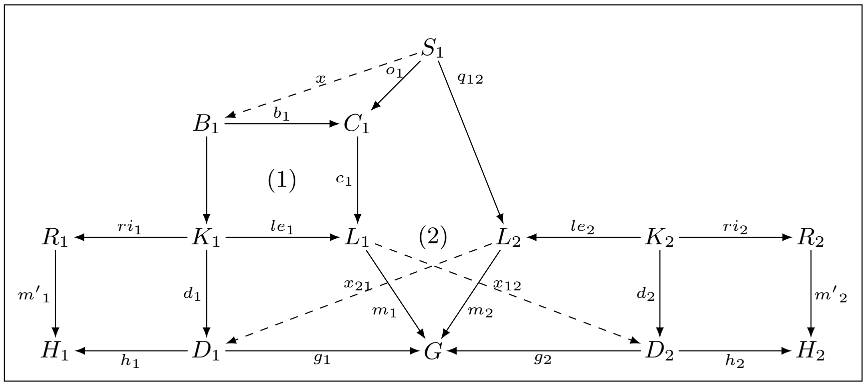


Fig. 2. Illustration of conflict and conflict reason

LHS of the second rule (Fig. 2). Remember that $C_1 := L_1 \setminus (K_1 \setminus B_1)$ contains the deletion part of a given rule and boundary graph B_1 consisting of all nodes needed to make $L_1 \setminus K_1$ a graph. $C_1 \setminus B_1$ may consist of several disjoint fragments, called *deletion fragments*. Completing a deletion fragment to a graph by adding all incident nodes (i.e. boundary nodes) it becomes a *deletion component* in C_1. Each two deletion components overlap in boundary nodes only; the union of all deletion components is C_1. If two transformations overlap such that there is at least one element of a deletion fragment included, they are in conflict.

The *overlap conditions* reintroduced in Def. 3 describe for an overlap of a given pair of rules under which conditions it may lead to a conflict (conflict condition), since there exist transformations (transformation condition) that overlap all elements as prescribed by the given overlap indeed (completeness condition). We call such an overlap *conflict reason* and it is minimal if no bigger one exists in which it can be embedded. Table 1 provides an overview over all conflict notions for rules (as reintroduced in Def. 4) and their overlap conditions.

General setting: For the rest of this paper, we assume the following basic setting: Given rules $r_1 : L_1 \overset{le_1}{\hookleftarrow} K_1 \overset{ri_1}{\hookrightarrow} R_1$ with the initial pushout (1) for $K_1 \overset{le_1}{\hookrightarrow} L_1$ and $r_2 : L_2 \overset{le_2}{\hookleftarrow} K_2 \overset{ri_2}{\hookrightarrow} R_2$, we consider a span $s_1 : C_1 \overset{o_1}{\hookleftarrow} S_1 \overset{q_{12}}{\rightarrow} L_2$ as depicted in Fig. 2.

Definition 3 (overlap conditions). *Given rules r_1 and r_2 as well as a span s_1,* overlap conditions *for the span s_1 of (r_1, r_2) are defined as follows:*

1. Weak conflict condition*: Span s_1 satisfies the* weak conflict condition *if there does not exist any injective morphism $x : S_1 \rightarrow B_1$ such that $b_1 \circ x = o_1$.*
2. Conflict condition*: Span s_1 satisfies the* conflict condition *if for each coproduct $\bigoplus_{i \in I} S_1^i$, where each S_1^i is non-empty and $S_1 = \bigoplus_{i \in I} S_1^i$, each of the induced spans $s_1^i : C_1 \overset{o_1^i}{\hookleftarrow} S_1^i \overset{q_{12}^i}{\rightarrow} L_2$ with $o_1^i = o_1|_{S_1^i}$ and $q_{12}^i = q_{12}|_{S_1^i}$ fulfills the weak conflict condition.*

3. Transformation condition*: Span s_1 satisfies the* transformation condition *if there is a pair of transformations* $(t_1, t_2) = (G \overset{m_1, r_1}{\Longrightarrow} H_1, G \overset{m_2, r_2}{\Longrightarrow} H_2)$ *via* (r_1, r_2) *with* $m_1(c_1(o_1(S_1))) = m_2(q_{12}(S_1))$ *(i.e. (2) is commuting in Fig. 2).*
4. Completeness condition*: Span s_1 satisfies the* completeness condition *if there is a pair of transformations* $(t_1, t_2) = (G \overset{m_1, r_1}{\Longrightarrow} H_1, G \overset{m_2, r_2}{\Longrightarrow} H_2)$ *via* (r_1, r_2) *such that (2) is the pullback of* $(m_1 \circ c_1, m_2)$ *in Fig. 2.*
5. Minimality condition*: A span* $s'_1 : C_1 \overset{o'_1}{\hookleftarrow} S'_1 \overset{q'_{12}}{\rightarrow} L_2$ *can be embedded into span s_1 if there is an injective morphism* $e : S'_1 \rightarrow S_1$*, called* embedding morphism*, such that* $o_1 \circ e = o'_1$ *and* $q_{12} \circ e = q'_{12}$*. If e is an isomorphism, then we say that the spans s_1 and s'_1 are* isomorphic*. (See (3) and (4) in Fig. 3.) Span s_1 satisfies the* minimality condition *w.r.t. a set SP of spans if any $s'_1 \in SP$ that can be embedded into s_1 is isomorphic to s_1.*

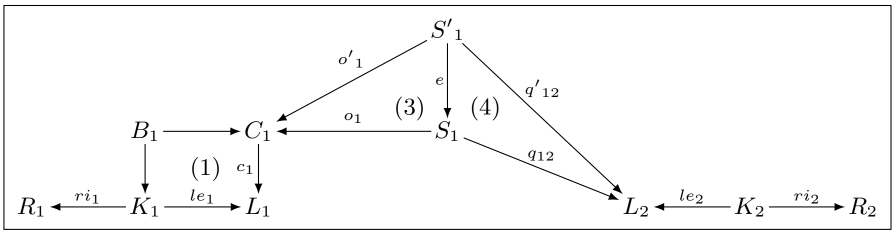


Fig. 3. Illustrating span embeddings

Note that span s_1 which fulfils the weak conflict condition, also fulfils the conflict condition iff S_1 does not contain any isolated boundary nodes [4].

Definition 4 (conflict notions). *Let the rules r_1 and r_2 as well as a span s_1 be given.*

1. *Span s_1 is called* conflict part candidate *for the pair of rules* (r_1, r_2) *if it satisfies the conflict condition. Graph S_1 is called the* conflict graph *of s_1.*
2. *A conflict part candidate s_1 for* (r_1, r_2) *is a* conflict part *for* (r_1, r_2) *if s_1 fulfils the transformation condition.*
3. *A conflict part candidate s_1 for* (r_1, r_2) *is a* conflict atom candidate *for* (r_1, r_2) *if it fulfils the minimality condition w.r.t. the set of all conflict part candidates for* (r_1, r_2).
4. *A conflict part s_1 for* (r_1, r_2) *is a* conflict atom *if it fulfils the minimality condition w.r.t. the set of all conflict parts for* (r_1, r_2).
5. *A conflict part s_1 for* (r_1, r_2) *is a* conflict reason *for* (r_1, r_2) *if s_1 fulfils the completeness condition.*
6. *A conflict reason s_1 for* (r_1, r_2) *is* minimal *if it fulfils the minimality condition w.r.t. the set of all conflict reasons for* (r_1, r_2).

Conflict notions are in various interrelations as shown in [4]. Here, we recall those that are relevant for our further exploration of conflict reasons.

Table 1. Overview of conflict notions

Overlap condition / conflict notion	conflict condition	transf. condition	compl. condition	minimality condition
conflict part candidate	x			
conflict part	x	x		
conflict atom candidate	x			x
conflict atom	x	x		x
conflict reason	x	x	x	
min. conflict reason	x	x	x	x

Definition 5 (covering and composition of conflict parts).

1. *Given a conflict part s_1, the set A of all conflict atoms that can be embedded into s_1* covers *s_1 if for each conflict part $s'_1 : C_1 \stackrel{o'_1}{\hookleftarrow} S'_1 \stackrel{q'_{12}}{\rightarrow} L_2$ for (r_1, r_2) that can be embedded into s_1, it holds that s'_1 is isomorphic to s_1 if each atom in A can be embedded into s'_1.*
2. *Given a conflict part s_1, the set $M = \{s^m_i | \ i \in I\}$ of spans that can be embedded into s_1 via a corresponding set of embedding morphisms $E_M = \{e_i | \ i \in I\}$* composes *$s_1$ if the set E_M is jointly surjective.*

Fact 1 (Interrelations of conflict notions and characterization [4, 2]) *Let rules r_1 and r_2 as well as conflict part candidate s_1 for (r_1, r_2) be given.*

1. *If s_1 is a conflict part for (r_1, r_2), there is a conflict reason for (r_1, r_2) such that s_1 can be embedded into it.*
2. *If s_1 is a conflict atom candidate for rules (r_1, r_2), its conflict graph S_1 either consists of a node v s.t. $o_1(v) \in C_1 \setminus B_1$ or of an edge e with its incident nodes v_1 and v_2 s.t. $o_1(e) \in C_1 \setminus B_1$ and $o_1(v_1), o_1(v_2) \in B_1$.*
3. *If s_1 is a conflict part (esp. conflict reason) for rules (r_1, r_2), the set A of all conflict atoms that can be embedded into s_1 is non-empty and* covers *s_1.*
4. *If s_1 is a conflict reason for rule pair $(r_1, ND(r_2))$, it can be composed of all minimal conflict reasons for $(r_1, ND(r_2))$ that can be embedded into s_1.*
5. *If s_1 is a minimal conflict reason for rule pair $(r_1, ND(r_2))$, its conflict graph S_1 is a subgraph of a deletion component of C_1.*

4 DR/DD Conflict Reasons

Conflict reasons are constructed from conflict part candidates. We distinguish delete-read (dr) from delete-delete (dd) conflict part candidates (and consequently also reasons) by requiring that the dd candidate entails elements that are deleted by both rules, whereas the dr candidate does not.

Definition 6 (dr/dd conflict reason). *Let the rules r_1 and r_2 and a conflict part candidate s_1 for (r_1, r_2) be given.*

1. s_1 *is a* dr conflict part candidate *for* (r_1, r_2) *if there exists a morphism* $k_{12} : S_1 \rightarrow K_2$ *such that* $le_2 \circ k_{12} = q_{12}$.
2. s_1 *is a* dd conflict part candidate *for* (r_1, r_2) *otherwise.*

A conflict part, atom or (minimal) reason is a dr (dd) conflict part, atom or (minimal) reason, *respectively, if it is a dr (dd) conflict part candidate.*

A conflict atom consists of either a deleted node or deleted edge with incident preserved nodes (see Fact 1). DR atoms, where the conflict graph consists of a node, might possess incident edges that are deleted not only by the first, but also by the second rule. We say that a dr atom is pure if this is not the case.

Definition 7 (pure dr atom). *Given a conflict reason* $s_1 : C_1 \overset{o_1}{\hookleftarrow} S_1 \overset{q_{12}}{\hookrightarrow} L_2$ *and a dr atom* $s'_1 : C_1 \overset{o'_1}{\hookleftarrow} S'_1 \overset{q'_{12}}{\hookrightarrow} L_2$ *embedded into* s_1 *via* $e : S'_1 \rightarrow S_1$*, then* s'_1 *is* pure with respect to s_1 *if the conflict graph* S'_1 *consists of an edge, or if* S'_1 *consists of a node* x *and each edge* y *in* C_1 *with source or target node* $o'_1(x)$ *has a pre-image* y' *in* S_1 *with source or target node* $e(x)$ *s.t.* $q_{12}(y') \in le_2(K_2)$.

In this paper, we consider the general case of rule pairs where both rules may be deleting. This implies that conflicts may arise in both directions. The following definition therefore describes for a given pair of rules and a conflict part candidate how compatible counterparts look like for the reverse direction. It naturally leads to the notion of compatible conflict reasons that may occur in the same conflict in reverse directions. We distinguish compatible counterparts that overlap in at least one deletion item as special case, since they will be important for the dd conflict reason construction.

Definition 8 (compatibility, join, dd overlapping). *Given rules* r_1 *and* r_2 *with conflict part candidates* s_1 *for* (r_1, r_2) *and* $s_2 : C_2 \overset{o_2}{\hookleftarrow} S_2 \overset{q_{21}}{\hookrightarrow} L_1$ *for* (r_2, r_1) *as in Fig. 4.*

1. *Candidates* s_1 *and* s_2 *are* compatible *if the pullbacks* $S_1 \overset{a_1}{\longleftarrow} S' \overset{a_2}{\longrightarrow} S_2$ *of* $(c_1 \circ o_1, q_{21})$ *and* $S_1 \overset{a'_1}{\longleftarrow} S'' \overset{a'_2}{\longrightarrow} S_2$ *of* $(q_{12}, c_2 \circ o_2)$ *are isomorphic via an isomorphism* $i : S' \rightarrow S''$ *such that* $a'_1 \circ i = a_1$ *and* $a'_2 \circ i = a_2$*. We denote a representative of these pullbacks as* $s : S_1 \overset{a_1}{\hookleftarrow} S' \overset{a_2}{\hookrightarrow} S_2$.
2. *Let* $S_1 \overset{s'_1}{\hookrightarrow} S \overset{s'_2}{\hookleftarrow} S_2$ *be the pushout of* s*. Morphisms* ls_1 *and* ls_2 *are the universal morphisms arising from this pushout and the fact that* $c_1 \circ o_1 \circ a_1 = q_{21} \circ a_2$ *and* $c_2 \circ o_2 \circ a_2 = q_{12} \circ a_1$*. Then* $L_1 \overset{ls_1}{\hookleftarrow} S \overset{ls_2}{\hookrightarrow} L_2$ *as in Fig. 4 is called the* join *of* s_1 *and* s_2 *and* S *is called the* joint conflict graph.
3. *Compatible conflict part candidates* s_1 *for* (r_1, r_2) *and* s_2 *for* (r_2, r_1) *are* dd overlapping *if there do not exist morphisms* $k_1 : S' \rightarrow K_1$ *with* $le_1 \circ k_1 = c_1 \circ o_1 \circ a_1$ *and* $k_2 : S' \rightarrow K_2$ *with* $le_2 \circ k_2 = c_2 \circ o_2 \circ a_2$.

Conflict parts, atoms, or reasons are (dd overlapping) compatible if the corresponding conflict part candidates are, respectively.

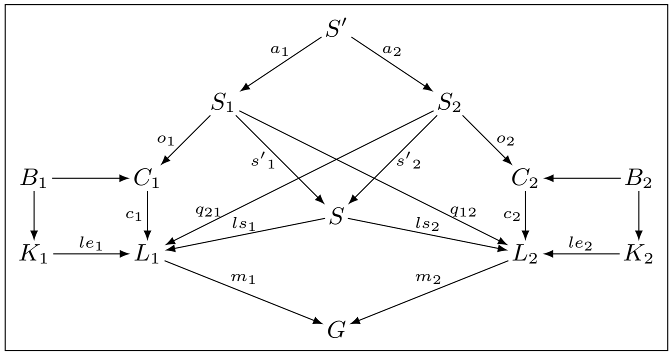


Fig. 4. Compatible conflict part candidates

As clarified in the following proposition a dr conflict reason can be responsible on its own for a conflict: if only a dr conflict reason is overlapped by corresponding matches, then we obtain a dr initial conflict. Contrarily, for a dd conflict reason, there exists at least one compatible conflict reason for the reverse rule pair that it can be overlapped with leading to a dd initial conflict. The idea of initial conflicts [9] is that they describe all possible conflicts in a minimal way by overlapping as less elements as possible from both rules.

Proposition 1 (dr/dd conflict reasons and initial conflicts).

- *Given a* dr *conflict reason* $s_1 : C_1 \overset{o_1}{\hookleftarrow} S_1 \overset{q_{12}}{\hookrightarrow} L_2$ *for rule pair* (r_1, r_2)*, then the pushout* $(m_1 : L_1 \to K, m_2 : L_2 \to K)$ *of* $L_1 \overset{c_1 \circ o_1}{\hookleftarrow} S_1 \overset{q_{12}}{\hookrightarrow} L_2$ *determines the matches of an dr initial conflict* $(t_1, t_2) = (K \overset{m_1, r_1}{\Longrightarrow} P_1, K \overset{m_2, r_2}{\Longrightarrow} P_2)$ *with the pullback of* $(m_1 \circ c_1, m_2)$ *being isomorphic to* s_1*.*
- *Given a* dd *conflict reason* s_1 *for rule pair* (r_1, r_2)*, then there exists a non-empty set* $DD(s_1)$ *of dd overlapping compatible dd conflict reasons for rule pair* (r_2, r_1) *s.t. for each* s_2 *in* $DD(s_1)$ *the pushout* $(m_1 : L_1 \to K, m_2 : L_2 \to K)$ *of the join of* (s_1, s_2) *determines the matches of an dd initial conflict* $(t_1, t_2) = (K \overset{m_1, r_1}{\Longrightarrow} P_1, K \overset{m_2, r_2}{\Longrightarrow} P_2)$*.*

Finally, we can conclude from the overapproximation already considered in [2] the following relationship between conflict reasons and overapproximated ones.

Proposition 2 (overapproximating conflict reasons). *If a span* s_1 *is a conflict reason for rule pair* (r_1, r_2)*, it is a dr conflict reason for* $(r_1, ND(r_2))$*.*

5 Characterizing DR/DD Conflict Reasons

Table 2 gives a preview of characterization results for dr/dd conflict reasons described in this section and used for coming up with basic procedures for constructing them in Sect. 6. We start with characterizing dr/dd conflict reasons via atoms (Prop. 3). We proceed to characterize dr/dd minimal conflict reasons, showing that we can reuse the constructions for a pair of rules, where the second one is non-deleting (Prop. 4 and Prop. 5). We conclude with characterizing dr/dd conflict reasons via minimal ones (Corr. 1). We distinguish dr from dd conflict reasons and learn that the dd case is more involved than the dr case.

Table 2. Characterizing dr/dd conflict reasons for rule pair (r_1, r_2)

conflict notion	**characterization result**
dr conflict reason	covered by pure dr atoms only (Prop. 3)
dd conflict reason	covered by at least one dd or non-pure dr atom and arbitrary number of pure dr atoms (Prop. 3)
dr min. conflict reason	equals min. reason for $(r_1, ND(r_2))$ (Prop. 4)
dd min. conflict reason	composed of min. reasons for $(r_1, ND(r_2))$ being dd conflict part candidates for (r_1, r_2) (Prop. 5)
dr conflict reason	composed of dr min. conflict reasons only (Corr. 1)
dd conflict reason	composed of min. conflict reasons where at least one of which is dd (Corr. 1)

Characterizing DR/DD Reasons via Atoms. From the characterization of conflict reasons via atoms (see Fact 1), we can conclude that dr reasons are covered (see Def. 5) by pure dr atoms (see Def. 7). Moreover, each dd reason entails at least one dd atom or non-pure dr atom.

Proposition 3 (dr/dd conflict reason characterization). *A* dr conflict reason *is covered by pure dr atoms only. On the contrary, a* dd conflict reason *is covered by at least one dd atom or non-pure dr atom and an arbitrary number of pure dr atoms.*

From the above characterization it follows that it makes sense to distinguish as special case dd conflict reasons that are covered by dd atoms only.

Definition 9 (pure dd conflict reason). *A dd conflict reason is* pure *if it is covered by dd atoms only.*

Characterizing DR/DD Minimal Reasons. A dr minimal conflict reason for a given rule pair equals the minimal conflict reason for the rule pair, where the second rule of the given rule pair has been made non-deleting.

Proposition 4 (dr minimal conflict reason characterization). *Each dr minimal conflict reason s_1 for rule pair (r_1, r_2) is a dr minimal conflict reason for rule pair $(r_1, ND(r_2))$.*

We can therefore conclude that the conflict graph of a dr minimal conflict reason is again a subgraph of one deletion component (see Fact 1).

Example 1 (dr minimal conflict reason). Figure 5 shows two dr minimal conflict reasons as examples, one for (r_1, r_2) and one for (r_2, r_1). Note that they do not overlap in elements to be deleted. They are the same minimal reasons as for the cases where the second rule is made non-deleting. For comparison, AGG [10] computes 4 critical pairs for (r_1, r_2) one of which is an initial conflict. Figure 5 shows a critical pair but not the initial conflict. For obtaining the initial conflict, it is enough to overlap merely the dr minimal conflict reason for (r_1, r_2) (see Prop. 1).

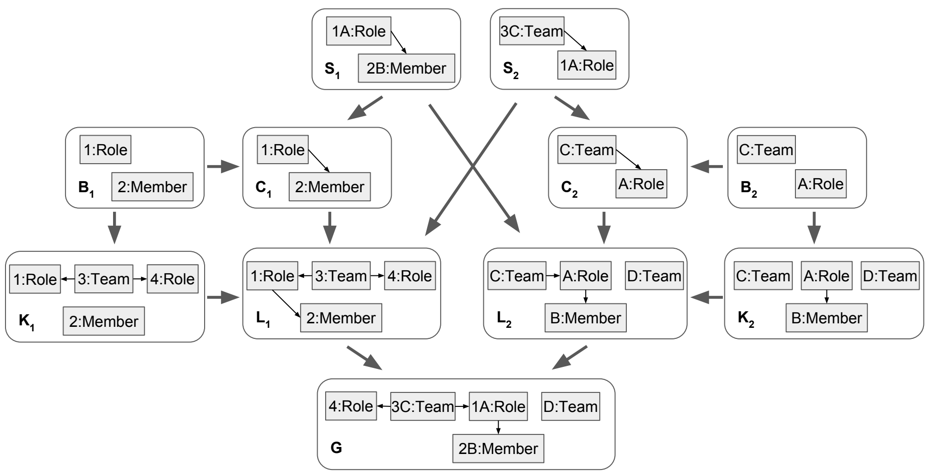


Fig. 5. Two dr minimal conflict reasons for rules *ShiftMember* and *ShiftRole*

A dd minimal conflict reason for a rule pair is composed (Def. 5) of minimal reasons for the rule pair, where the second rule has been made non-deleting.

Proposition 5 (dd minimal conflict reason characterization). *Given a dd minimal conflict reason s_1 for (r_1, r_2), s_1 is composed of a set $M = \{s_i^m | \ i \in I\}$ of minimal conflict reasons for $(r_1, ND(r_2))$. Moreover, each reason in M is a dd conflict part candidate for (r_1, r_2).*

Remember that the conflict graph of each minimal conflict reason for a rule pair, where the second rule is non-deleting, consists of a subgraph of one deletion component. We can therefore conclude that the conflict graph of a dd minimal conflict reason is a subgraph of one or more deletion components.

Example 2 (dd minimal conflict reason). Figure 6 shows an example of a dd minimal conflict reason s_1 for rule pair (*ReplaceM*, *ReplaceM*). s_1 is a dd conflict reason since S_1 cannot be mapped to K_2 in a suitable way. Furthermore, we see that graph G can be constructed such that the completeness condition is fulfilled and m_1 and m_2 are matches. It remains to show that s_1 is indeed minimal. Conflict part candidate ${s'}_1$ would also be a promising candidate. The resulting graph G, however, would not merge nodes 4:Role with D:Role. Morphism m_1 would not satisfy the dangling condition then. For a conflict part candidate comprising nodes 2B: Member, 4D: Role, and 5E:Team we can argue similarly. Due to Prop. 5, s_1 has to be composed of minimal conflict reasons for (*ReplaceM*, ND(*ReplaceM*)) which have to be deletion components as shown in [2]. Hence, there are no further possibilities to choose a smaller span than s_1. Note that a dd conflict reason is not always pure. For example, the atom 1A:Member is a (non-pure) dr atom. In addition to this dd minimal conflict reason there exist two more. Their conflict graphs contain the following sets of nodes: {1B:Member, 2A:Member, 3D:Role} and {2A:Member, 4C:Role, 5F:Team}.

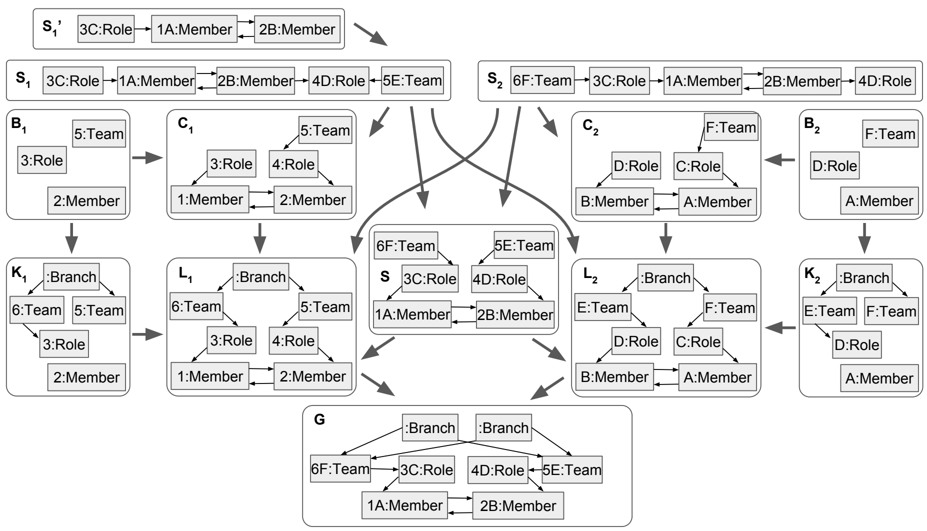


Fig. 6. A dd (minimal) conflict reason for the rule pair (*ReplaceM*,*ReplaceM*)

For the rule pair (*ReplaceM*, *ND(ReplaceM)*), four minimal conflict reasons exist instead. Their conflict graphs contain the following sets of nodes: {1B:Member, 2A:Member, 3D:Role}, {2A:Member, 4C:Role, 5F:Team}, {1A:Member, 2B:Member, 3C:Role}, and {2B:Member, 4D:Role, 5E:Team}. While the first two are also conflict graphs of dd minimal conflict reasons for rule pair (*ReplaceM*, *ReplaceM*), this is not the case for the last two as the dangling condition is not satisfied in those cases. Moreover, the dd minimal conflict reason in Figure 6 is a conflict reason for (*ReplaceM*, *ND(ReplaceM)*) but not a minimal one. For comparison, for the rule pair (*ReplaceM*, *ReplaceM*) AGG [10] computes 71 critical pairs. Figure 6 shows one initial conflict (according to Prop. 1).

Characterizing DR/DD Reasons via Minimal Reasons. The following proposition was proven for a rule pair with the second rule non-deleting, but it can be generalized to the case where the second rule is not necessarily non-deleting. It allows us to construct also for this general case conflict reasons from minimal ones by composing them appropriately.

Proposition 6 (composition of conflict reasons by minimal reasons). *Given a conflict reason s_1 for (r_1, r_2), there is a set of minimal conflict reasons for (r_1, r_2) s_1 is composed of (Def. 5).*

This allows to establish the following relationship between dr/dd conflict reasons and minimal ones.

Corollary 1 (composition of dr/dd conflict reasons by minimal ones).

- *A* dr *conflict reason is composed of dr minimal conflict reasons only.*
- *A* dd *conflict reason is composed of minimal conflict reasons where at least one of which is dd.*

6 Constructing DR/DD Conflict Reasons

It is known how to construct (minimal) conflict reasons for a rule pair, where the second rule is non-deleting [2]. Proposition 4 tells us that each *dr minimal conflict reason* for a rule pair (r_1, r_2) equals a minimal conflict reason for the rule pair $(r_1, ND(r_2))$. Each such minimal conflict reason for the rule pair $(r_1, ND(r_2))$ that is in addition dr for (r_1, r_2) delivers us a dr minimal conflict reason. From Corollary 1 we know that each *dr conflict reason* is a composition of dr minimal conflict reasons again such that their construction is analogous to the one already presented in [2]. In the following, we construct *dd minimal conflict reasons* from rule pairs, which is much more involved than the dr case.

Definition 10 (composability, composition of conflict part candidates). *Given rules r_1 and r_2 with conflict part candidates s_1 and $s'_1 : C_1 \overset{o'_1}{\hookleftarrow} S'_1 \overset{q'_{21}}{\hookrightarrow} L_2$ for (r_1, r_2).*

1. *Candidates s_1 and s'_1 are* composable *if the pullbacks $s : S_1 \overset{a_1}{\longleftarrow} S' \overset{a_2}{\longrightarrow} S'_1$ of (o_1, o'_1) and $S_1 \overset{a'_1}{\longleftarrow} S'' \overset{a'_2}{\longrightarrow} S'_1$ of (q_{21}, q'_{21}) are isomorphic via an isomorphism $i : S' \to S''$ such that $a'_1 \circ i = a_1$ and $a'_2 \circ i = a_2$. We denote a representative of these pullbacks as s.*
2. *Let $S_1 \overset{s'_1}{\hookrightarrow} S \overset{s'_2}{\hookleftarrow} S'_1$ be the pushout of s. Morphisms ls_1 and ls_2 are the universal morphisms arising from this pushout and the fact that $o_1 \circ a_1 = o'_1 \circ a_2$ and $q_{21} \circ a_2 = q'_{21} \circ a_1$. Then $C_1 \overset{ls_1}{\hookleftarrow} S \overset{ls_2}{\hookrightarrow} L_2$ is called the* composition *of s_1 and s'_1.*
3. *Given a set C of candidates, they are composable for $|C| < 2$. If C is larger, each two of its candidates have to be composable. The composition of all candidates in C is the candidate itself if $|C| = 1$ and the successive composition of its candidates otherwise.*

Conflict parts, atoms, or reasons are composable if the corresponding conflict part candidates are, respectively.

Construction (dd minimal conflict reasons).
Let the rules r_1 and r_2 be given.

- Let CPC_1 be the set of all minimal conflict reasons for $(r_1, ND(r_2))$ which are dd conflict part candidates for (r_1, r_2).

– Given a conflict part candidate s_1, let $CPC_2(s_1)$ be the set of all conflict reasons $s_2 : C_2 \overset{o_2}{\hookleftarrow} S_2 \overset{q_{21}}{\hookrightarrow} L_1$ for $(r_2, ND(r_1))$ such that s_1 is dd overlapping compatible with s_2.

The set *DDMCR* of all dd minimal conflict reasons for (r_1, r_2) can be constructed as follows (compare Fig. 4):

1. Let *DDMCR* be the empty set and $n := 1$.
2. For each subset M of n composable candidates in CPC_1 for which the composition of a subset $M' \subset M$ of $n-1$ candidates is not in *DDMCR* yet:
   (a) Compose all candidates in M to a candidate s_1 and construct $CPC_2(s_1)$.
   (b) For each s_2 in $CPC_2(s_1)$:
      – Construct the pushout $L_1 \overset{m_1}{\hookrightarrow} G \overset{m_2}{\hookleftarrow} L_2$ of the join of s_1 and s_2.
      – If m_1 is a match for rule r_1 and m_2 a match for r_2 and if the pullback of $(m_1 \circ o_1, m_2)$ is isomorphic to s_1, then add s_1 to *DDMCR* and break.
   (c) $n := n + 1$

Remark: Note that a composition of 0 candidates is trivially not in an empty *DDMCR*. This construction terminates since CPC_1 is finite and it has finitely many subsets. n is increased maximally to the size of CPC_1.

Example 3 (Construction of dd min. conflict reason). We construct a dd minimal conflict reason for rule pair (*ReplaceM*, *ReplaceM*). We start with $n = 1$ and choose s'_1 including conflict graph S'_1 in Fig. 6. It is a min. reason for (*ReplaceM*, *ND(ReplaceM)*) not belonging to *DDCMR* yet. As discussed in Example 2, it is not a conflict reason for (*ReplaceM*, *ReplaceM*). Hence, we cannot find a suitable $s_2 \in CPC_2(s'_1)$. The argumentation for the other minimal conflict reason for (*ReplaceM*, *ND(ReplaceM)*) is analogous. Hence, we have to set $n = 2$. As s_1 is a composition of two minimal conflict reasons for (*ReplaceM*, *ND(ReplaceM)*), we choose this candidate next. Figure 6 shows that there is an $s_2 \in CPC_2(s_1)$ such that two matches m_1 and m_2 with the pullback of $(m_1 \circ o_1, m_2)$ being isomorphic to s_1 can be constructed. Hence, s_1 is in *DDMCR*.

Theorem 2 (Correctness dd min. conflict reason construction). *Given two rules r_1 and r_2, the construction above yields dd minimal conflict reasons for (r_1, r_2) only (soundness) and all those (completeness).*

Proof. Soundness: Because of Prop. 5 we know that a dd minimal conflict reason for s_1 for (r_1, r_2) is composed of a set of minimal conflict reasons for $(r_1, ND(r_2))$, where each of them is a dd conflict part candidate for (r_1, r_2). In Step 2 of the construction we select exactly these building bricks for minimal conflict reasons and compose them if composable. We then perform a breadth-first search (w.r.t. size of composition) over all possible compositions of minimal conflict reasons for $(r_1, ND(r_2))$. The search returns all compositions for which we can find a compatible conflict part candidate (see Corollary 2) that leads to a dd reason for (r_1, r_2) (see Prop. 1). We only continue searching for new minimal reasons if we did not find a successful smaller composition already.

Completeness: By checking in the construction for *all* possible combinations of minimal conflict reasons for $(r_1, ND(r_2))$ if we can find a compatible conflict part candidate leading to an initial conflict (see Prop. 1), we find *all* minimal conflict reasons for $(r_1, ND(r_2))$. □

Having constructions for dr/dd minimal conflict reasons at hand, we can compute *dd conflict reasons*. Each dd reason is composed from minimal reasons, where at least one of them is dd (see Cor. 1). Their construction is thus analogous to the one for dd minimal conflict reasons with the following two differences: (1) Instead of CPC_1 we have the set MCR_1 of minimal dr/dd reasons for (r_1, r_2). (2) Step 2 considers each set of n composable minimal reasons in MCR_1 with at least one of them dd, no matter if the composition of a subset is already present in the result set or not (since we do not need minimality). Soundness and completeness follows analogous to the proof of Theorem 2 based on Cor. 1 instead of Prop. 5 and omitting the argument for minimality.

7 Related Work and Conclusion

Our paper continues a recent line of research on conflict and dependency analysis (CDA) for graph transformations aiming to improve on the previous CDA technique of critical pair analysis (CPA). Originally inspired by the CPA in term and term graph rewriting [3], the CPA theory has been extended to graph transformation and generally, to $\mathcal{M}$-adhesive transformation systems [11, 1].

Azzi et al. [12] conducted similar research to identify root causes of conflicting transformations as initiated in [8] and continued in [5]. Their work is based upon an alternative characterization of parallel independence [13] that led to a new categorical construction of initial transformation pairs. The most important difference is that we define our conflict notions (including the dr/dd characterization) for rule pairs instead of transformation pairs [12] with the aim of coming up with efficient CDA. Moreover, we consider conflict atoms and (minimal) reasons, whereas Azzi et al. [12] focus conflict reasons (in our terminology).

In this paper, we extend the foundations for computing conflicts and dependencies for graph transformations in a multi-granular way. In particular, our earlier work relied on an over-approximation of (minimal) conflict reasons; we assumed a non-deleting version of the second rule of the considered rule pair as input. In contrast, our present work introduces a new constructive characterization of (minimal) conflict reasons distinguishing dr from dd reasons and we present a basic computation procedure that is sound and complete. Building on our recent work [2], we now support precise conflict computation for any given granularity level, from binary (conflict atom) over medium (minimal conflict reason) to fine (conflict reason). Future work is needed to implement the presented constructions, to evaluate effiency and to investigate usability.

Acknowledgements. This work was partially funded by the German Research Foundation, project "Triple Graph Grammars (TGG) 2.0: Reliable and Scalable Model Integration".

References

1. H. Ehrig, K. Ehrig, U. Prange, and G. Taentzer, *Fundamentals of Algebraic Graph Transformation*, ser. Monographs in Theoretical Computer Science. Springer, 2006.
2. L. Lambers, D. Strüber, G. Taentzer, K. Born, and J. Huebert, "Multi-granular conflict and dependency analysis in software engineering based on graph transformation," in *Int. Conf. on Software Engineering (ICSE)*. ACM, 2018, pp. 716–727, Extended version at: www.uni-marburg.de/fb12/swt/forschung/publikationen/2018/LSTBH18-TR.pdf.
3. D. Plump, "Critical Pairs in Term Graph Rewriting," in *Mathematical Foundations of Computer Science*, 1994, pp. 556–566.
4. L. Lambers, K. Born, J. Kosiol, D. Strüber, and G. Taentzer, "Granularity of conflicts and dependencies in graph transformation systems: A two-dimensional approach," *Journal of Logical and Algebraic Methods in Programming*, vol. 103, pp. 105–129, 2019.
5. K. Born, L. Lambers, D. Strüber, and G. Taentzer, "Granularity of conflicts and dependencies in graph transformation systems," in *International Conference on Graph Transformation (ICGT)*, 2017, pp. 125–141.
6. K. Beck, M. Beedle, A. Van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries *et al.*, "Manifesto for agile software development," 2001.
7. T. Arendt, E. Biermann, S. Jurack, C. Krause, and G. Taentzer, "Henshin: Advanced Concepts and Tools for In-Place EMF Model Transformations," in *Int. Conf. on Model-Driven Engineering Languages and Systems (MoDELS)*, 2010, pp. 121–135.
8. L. Lambers, H. Ehrig, and F. Orejas, "Efficient conflict detection in graph transformation systems by essential critical pairs," *Electr. Notes Theor. Comput. Sci.*, vol. 211, pp. 17–26, 2008.
9. L. Lambers, K. Born, F. Orejas, D. Strüber, and G. Taentzer, *Initial Conflicts and Dependencies: Critical Pairs Revisited*. Cham: Springer International Publishing, 2018, pp. 105–123.
10. AGG, "Attributed Graph Grammar system," http://user.cs.tu-berlin.de/ gragra/agg/.
11. H. Ehrig, J. Padberg, U. Prange, and A. Habel, "Adhesive high-level replacement systems: A new categorical framework for graph transformation," *Fundam. Inform.*, vol. 74, no. 1, pp. 1–29, 2006.
12. G. G. Azzi, A. Corradini, and L. Ribeiro, "On the essence and initiality of conflicts," in *Graph Transformation - 11th International Conference, ICGT 2018, Held as Part of STAF 2018, Toulouse, France, June 25-26, 2018, Proceedings*. Springer, 2018, pp. 99–117.
13. A. Corradini, D. Duval, M. Löwe, L. Ribeiro, R. Machado, A. Costa, G. G. Azzi, J. S. Bezerra, and L. M. Rodrigues, "On the essence of parallel independence for the double-pushout and sesqui-pushout approaches," in *Graph Transformation, Specifications, and Nets: In Memory of Hartmut Ehrig*, R. Heckel and G. Taentzer, Eds. Cham: Springer International Publishing, 2018, pp. 1–18.

A Additional Technicalities

This section contains technicalities that are necessary to proof the main results of our paper.

Definition 11 (initial conflict [9]). *A pair of transformations* $ic : (G \overset{r_1,m_1}{\Longrightarrow} H_1, G \overset{r_2,m_2}{\Longrightarrow} H_2)$ *for a pair of rules* (r_1, r_2) *with deletion and boundary graphs* C_i *and* B_i *over the morphisms* $le_i : K_i \to L_i$ *for* $i = 1, 2$ *(compare Fig. 2 for the asymmetric case) is an* initial conflict *if* ic *has the following properties:*

1. *Minimal context:* m_1 *and* m_2 *are jointly surjective.*
2. *At least one conflicting element being deleted by* r_1 *and used by* r_2*:* $m_1(L_1) \cap m_2(L_2) \not\subseteq m_1(le_1(K_1))$.
3. *Overlap in deletion graphs only:* $m_1(L_1) \cap m_2(L_2) \subseteq (m_1(c_1(C_1)) \cap m_2(L_2)) \cup (m_1(L_1) \cap m_2(c_2(C_2)))$.
4. *No isolated boundary node in overlap graph:*
   $\forall x \in m_1(c_1(b_1(B_1))) \cap m_2(L_2) :$
   $\exists e \in m_1(c_1(C_1)) \cap m_2(L_2) : x = src(e) \vee x = tgt(e)$ *and*
   $\forall x \in m_2(c_2(b_2(B_2))) \cap m_1(L_1) :$
   $\exists e \in m_2(c_2(C_2)) \cap m_1(L_1) : x = src(e) \vee x = tgt(e)$.

Lemma 1 (Composition of pullback with monomorphism). *Let* $b_1 : B_1 \to C$ *and* $b_2 : B_2 \to C$ *be arbitrary morphisms in a category and* $m : C \to D$ *be a monomorphism. Then a span* $B_1 \overset{a_1}{\leftarrow} A \overset{a_2}{\rightarrow} B_2$ *is a pullback of* (b_1, b_2) *if and only if it is a pullback of* $(m \circ b_1, m \circ b_2)$.

Proof. For the whole argument, compare Fig. 7. First, let the span $B_1 \overset{a_1}{\leftarrow} A \overset{a_2}{\rightarrow} B_2$ be a pullback of $B_1 \overset{b_1}{\rightarrow} C \overset{b_2}{\leftarrow} B_2$. Let $q_1 : Q \to B_1$ and $q_2 : Q \to B_2$ be morphisms such that $m \circ b_1 \circ q_1 = m \circ b_2 \circ q_2$. Since m is monic, this implies $b_1 \circ q_1 = b_2 \circ q_2$. By the universal property of A, there exists a unique morphism $h : Q \to A$ such that $a_1 \circ h = q_1$ and $a_2 \circ h = q_2$. Hence, the span $B_1 \overset{a_1}{\leftarrow} A \overset{a_2}{\rightarrow} B_2$ is a pullback of $B_1 \overset{m \circ b_1}{\longrightarrow} D \overset{m \circ b_2}{\longleftarrow} B_2$ as well.

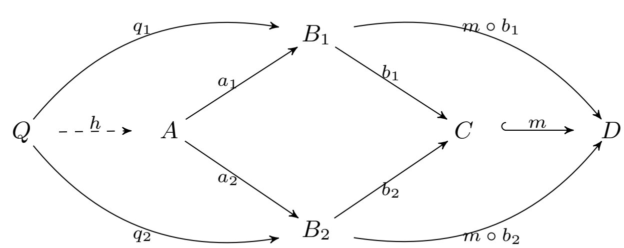


Fig. 7. Composing a pullback square with a monomorphism

Secondly, let the span $B_1 \overset{a_1}{\leftarrow} A \overset{a_2}{\rightarrow} B_2$ be a pullback of $B_1 \overset{m \circ b_1}{\longrightarrow} D \overset{m \circ b_2}{\longleftarrow} B_2$. Let $q_1 : Q \to B_1$ and $q_2 : Q \to B_2$ be morphisms such that $b_1 \circ q_1 = b_2 \circ q_2$. By composition with m also $m \circ b_1 \circ q_1 = m \circ b_2 \circ q_2$ holds. Again by the universal property of A, there exists a unique morphism $h : Q \to A$ such that $a_1 \circ h = q_1$ and $a_2 \circ h = q_2$. Hence, the span $B_1 \overset{a_1}{\leftarrow} A \overset{a_2}{\rightarrow} B_2$ is a pullback of $B_1 \overset{b_1}{\rightarrow} C \overset{b_2}{\leftarrow} B_2$ as well. □

Lemma 2 (Compatibility of conflict parts). *Let rules r_1 and r_2 with conflict part candidates s_1 for (r_1, r_2) and $s_2 : C_2 \overset{o_2}{\hookleftarrow} S_2 \overset{q_{21}}{\hookrightarrow} L_1$ for (r_2, r_1) as in Fig. 4 be given. If there exists a common graph G such that both rule r_1 and r_2 are applicable to G and s_1 and s_2 are conflict parts with respect to the according matches m_1 and m_2, then the conflict parts s_1 and s_2 are compatible.*

Proof. By assumption, there exists a graph G and matches $m_1 : L_1 \to G$ and $m_2 : L_2 \to G$ such that $m_1 \circ c_1 \circ o_1 = m_2 \circ q_{12}$ and $m_1 \circ q_{21} = m_2 \circ c_2 \circ o_2$. By Lemma 1, computing the pullback of $(c_1 \circ o_2, q_{21})$ is the same as computing the pullback of $(m_1 \circ c_1 \circ o_2, m_1 \circ q_{21})$ since m_1 is injective. But $m_1 \circ c_1 \circ o_1 = m_2 \circ q_{12}$ and $m_1 \circ q_{21} = m_2 \circ c_2 \circ o_2$ by assumption. Again, by Lemma 1, computing the pullback of $(m_2 \circ q_{12}, m_2 \circ c_2 \circ o_2)$ is the same as computing the pullback of $(q_{12}, c_2 \circ o_2)$. Thus, s_1 and s_2 are compatible. □

Fact 3 (Overapproximating conflicts [2]) *Given a pair of rules (r_1, r_2), the following holds: If transformation pair $(t_1, t_2) = (G \overset{r_1,m_1}{\Longrightarrow} H_1, G \overset{r_2,m_2}{\Longrightarrow} H_2)$ is in delete-use conflict, then transformation pair $(t_1, t_2') = (G \overset{r_1,m_1}{\Longrightarrow} H_1, G \overset{ND(r_2),m_2}{\Longrightarrow} H_2')$ is in dr conflict according to Def. 2.*

Lemma 3 (Covering of incident edges). *Let a conflict reason s_1 for (r_1, r_2) and a node $x \in S_1$ be given such that there exists an edge $e \in S_1$ that is incident to x. If $o_1(x) \in C_1 \setminus B_1$ or $q_{12}(x) \in L_2 \setminus K_2$ (interpreted as differences of sets), then there is no conflict reason s_1' that embeds into s_1, contains x but does not contain e.*

Proof (of Lemma 3). Assume such a conflict reason s_1' to exist and G' be a graph to which both rules are applicable at matches m_1' and m_2' such that s_1' is the corresponding conflict reason. Then, $m_1(c_1(o_1(e(x))))$ is a node to be deleted in G', either by application of r_1 at m_1' or of r_2 at m_2'. But both $m_2'(q_{12}(e))$ and $m_1'(c_1(o_1(e)))$ are necessarily incident edges to $m_1(c_1(o_1(e(x))))$ in G' and m_1' does not match $m_2'(q_{12}(e))$ and m_2' does not match $m_1'(c_1(o_1(e)))$ (otherwise, S_1' could not arise as pullback of $(m_1' \circ c_1, m_2')$). Hence, neither r_1 is applicable at match m_1' nor r_2 at match m_2' which is a contradiction to the existence of s_1'. □

B Additional Proofs

Proof (of Proposition 1).

- It follows from Theorem 6 in [4] that an initial conflict $(t_1', t_2') = (K' \overset{m_1',r_1}{\Longrightarrow} P_1, K' \overset{m_2',r_2}{\Longrightarrow} P_2)$ exists with the pullback of $(m_1' \circ c_1, m_2')$ being isomorphic to s_1. Consider the pushout $(m_1 : L_1 \to K, m_2 : L_2 \to K)$ of $L_1 \overset{c_1 \circ o_1}{\hookleftarrow} S_1 \overset{q_{12}}{\hookrightarrow} L_2$ with the induced morphism $k' : K \to K'$. We can deduce that m_1 and m_2 are the matches of an essential critical pair (t_1, t_2) that can be embedded into (t_1', t_2') ([8], Theorem 4.1, part 1). In particular, (t_1, t_2) is an initial conflict, since it is an essential critical pair that does not overlap any isolated

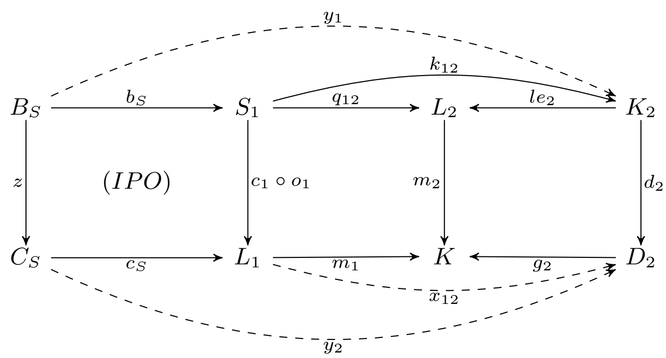


Fig. 8. Showing the morphism $x_{12} : L_1 \to D_2$ to exist

boundary nodes [9]. Because of initiality it follows that (t'_1, t'_2) and (t_1, t_2) are isomorphic.
We now show that (t_1, t_2) (being isomorphic to (t'_1, t'_2)) is dr. By definition there exists a morphism $k_{12} : S_1 \to K_2$ such that $le_2 \circ k_{12} = q_{12}$.
Compute the intial pushout over the morphism $c_1 \circ o_1$ as depicted in Fig. 8. Since initial pushouts are closed by composition with pushouts (compare Lemma 6.5 in [1]), the two left squares form an initial pushout over m_2. Since the right square is a pushout over m_2 as well, there exist morphisms $y_1 : B_S \to K_2$ and $y_2 : C_S \to D_2$ such that $m_1 \circ c_S = g_2 \circ y_2$. With that property we compute

$$\begin{aligned} g_2 \circ d_2 \circ k_{12} \circ b_S &= m_2 \circ le_2 \circ k_{12} \circ b_S \\ &= m_2 \circ q_{12} \circ b_S \\ &= m_1 \circ c_1 \circ o_1 \circ b_S \\ &= m_1 \circ c_S \circ z \\ &= g_2 \circ y_2 \circ z \ . \end{aligned}$$

Since g_2 is injective and $d_2 \circ k_{12} \circ b_S = y_2 \circ z$, the the universal property the left pushout square implies the existence of a morphism $x_{12} : L_1 \to D_2$ such that $x_{12} \circ c_S = y_2$ and $x_{12} \circ c_1 \circ o_1 = d_2 \circ k_{12}$. Now,

$$g_2 \circ x_{12} \circ c_S = g_2 \circ y_2 = m_1 \circ c_S$$

and

$$\begin{aligned} g_2 \circ x_{12} \circ (c_1 \circ o_1) &= g_2 \circ d_2 \circ k_{12} \\ &= m_2 \circ le_2 \circ k_{12} \\ &= m_2 \circ q_{12} \\ &= m_1 \circ (c_1 \circ o_1) \end{aligned}$$

and since c_S and $c_1 \circ o_1$ are jointly surjective (as co-projections of a pushout), this implies $g_2 \circ x_{12} = m_1$.

– It follows from Theorem 6 in [4] that an initial conflict $(t'_1, t'_2) = (K' \stackrel{m'_1,r_1}{\Longrightarrow} P_1, K' \stackrel{m'_2,r_2}{\Longrightarrow} P_2)$ exists with the pullback of $(m'_1 \circ c_1, m'_2)$ being isomorphic to s_1. Consider also the pullback s_2 of $(m'_1, m'_2 \circ c_1)$.
  By construction, the preconditions of Lemma 2 are met and s_1 and s_2 are compatible, hence. The matches induced by the pushout of the join of both reasons $L_1 \stackrel{ls_1}{\leftarrow} S \stackrel{ls_2}{\rightarrow} L_2$ lead to an essential critical pair (t_1, t_2) that can be embedded into (t'_1, t'_2) ([8], Theorem 4.1, part 2). In particular, (t_1, t_2) is an initial conflict, since it is an essential critical pair that does not overlap any isolated boundary nodes [9]. Therefore by initiality both pairs are isomorphic. Moreover (t_1, t_2) is a dd transformation: Assume a morphism $x_{12} : L_1 \rightarrow D_2$ to exist such that $g_2 \circ x_{12} = m_1$, i.e., assume the transformation to be dr. Then $m_2 \circ q_{12} = m_1 \circ c_1 \circ o_1 = g_2 \circ x_{12} \circ c_1 \circ o_1$ and the universal property of K_2 as pullback of (m_2, g_2) implies the existence of a morphism $k_{12} : S_1 \rightarrow K_2$ such that $le_2 \circ k_{12} = q_{12}$. But this contradicts the assumption of s_1 being dd.
  We show that s_1 and s_2 are moreover dd-overlapping. Since s_1 is a dd conflict reason, we know that there exists an element y in S_1 s.t. $c_1 \circ o_1(y) \in L_1 \setminus K_1$ and $q_{12}(y) \in L_2 \setminus K_2$. Assume that only boundary nodes would be deleted by r_2, then since S_1 does not contain any isolated boundary nodes, we have a contradiction, since the incident edge would need to be deleted as well. Now it follows that $q_{12}(y)$ has a preimage y' in $C_2 \setminus B_2$ such that $c_2(y') = q_{12}(y)$. Since s_1 was built as pullback of $m'_1 \circ c_1$ and m'_2 we know that $m'_1(c_1(o_1(y))) = m'_2(q_{12}(y))$. We thus have that $m'_1(c_1(o_1(y))) = m'_2(c_2(y'))$. Because of compatibility, we know that there is a preimage of y' in S_2. Therefore, it follows that also in S' there is a preimage such that no morphism $k_1 : S' \rightarrow K_1$ exists with $le_1 \circ k_1 = c_1 \circ o_1 \circ a_1$ and no morphism $k_2 : S' \rightarrow K_2$ exists with $le_2 \circ k_2 = c_2 \circ o_2 \circ a_2$.
  Finally, s_2 is indeed a conflict reason. By construction s_2 fulfills the completeness and transformation condition. From the fact that s_1 and s_2 are dd overlapping, it follows that s_2 fulfills the weak conflict condition. Since s_2 was constructed from an initial conflict, it also fulfills the conflict condition.

Proof (of Proposition 2). Since span s_1 is a conflict reason for rule pair (r_1, r_2), there is an initial conflict (Prop. 1) $(t_1, t_2) = (G \stackrel{r_1,m_1}{\Longrightarrow} H_1, G \stackrel{r_2,m_2}{\Longrightarrow} H_2)$. Due to Fact 3, the transformation pair $(t_1, t'_2) = (G \stackrel{r_1,m_1}{\Longrightarrow} H_1, G \stackrel{ND(r_2),m_2}{\Longrightarrow} H'_2)$ is a dr conflict. Span s_1 is a conflict reason for $(r_1, ND(r_2))$, since it fulfills the transformation condition because of the existence of (t_1, t'_2). Moreover, the conflict condition and completeness condition are still fulfilled.

Proof (of Proposition 3). Let s_1 be a dr conflict reason for rules r_1 and r_2. By Definition 6 there exists a morphism $k_{12} : S_1 \rightarrow K_2$ such that $le_2 \circ k_{12} = q_{12}$. From Fact 1 we know that a dr conflict reason is covered by conflict atoms. Assume that there exists a dd conflict atom or non-pure dr conflict atom $s'_1 : C_1 \stackrel{o'_1}{\hookleftarrow} S'_1 \stackrel{q'_{12}}{\rightarrow} L_2$ that embeds into s_1 via morphism $e : S'_1 \rightarrow S_1$. Define a morphism $k'_{12} : S'_1 \rightarrow K_2$ via $k'_{12} := k_{12} \circ e$. Since e is an embedding it holds that $q'_{12} = q_{12} \circ e = le_2 \circ k_{12} \circ e = le_2 \circ k'_{12}$. This is a contradiction to s'_1 being a dd conflict atom or non-pure dr conflict atom.

Let s_1 be a dd reason for rules r_1 and r_2. By Definition 6 there does not exist a morphism $k_{12} : S_1 \to K_2$ such that $le_2 \circ k_{12} = q_{12}$. Assume that s_1 is covered by pure dr atoms only. Then this leads to a contradiction, since it would be possible to construct a morphism $k_{12} : S_1 \to K_2$ such that $le_2 \circ k_{12} = q_{12}$, since for each of the atoms covering s_1 such a morphism exists and all incident edges of pure dr atoms are preserved by definition by the second rule as well. □

Proof (of Proposition 4). By Corollary 2, s_1 is a dr conflict reason for the rule pair $(r_1, ND(r_2))$. We show that it is also minimal.

Assume s_1 to not be minimal as conflict reason for the rule pair $(r_1, ND(r_2))$, i.e., there exists a conflict reason $s'_1 : C_1 \overset{o'_1}{\hookleftarrow} S'_1 \overset{q'_{12}}{\hookrightarrow} L_2$ which embeds into s_1 via morphism $e : S'_1 \to S_1$. By the transformation condition, there exists a graph G' and matches m'_1 and m'_2 for r_1 and $ND(r_2)$, respectively, such that r_1 and $ND(r_2)$ are applicable at these matches and S'_1 is the pullback of $(m'_1 \circ c_1, m'_2)$. We show that also r_2 is applicable at match m'_2. Thus, s'_1 would be a conflict reason for (r_1, r_2) as well which contradicts the minimality of s_1. By Proposition 1, we can assume without loss of generality that $L_1 \overset{m'_1}{\hookrightarrow} G' \overset{m'_2}{\hookleftarrow} L_2$ is the pushout of $(c_1 \circ o'_1, q'_{12})$. By its universal property, there is a morphism $f : G' \to G$ such that $f \circ m'_2 = m_2$. The only way for r_2 to not be applicable at m'_2 is that it deletes a node without deleting an incident edge. Assume $x \in G'$ to be such a node, i.e., it is an element of $m'_2(L_2 \setminus K_2)$. Then $f(x)$ is an element of $m_2(L_2 \setminus K_2)$ in G and since s_1 is dr, $f(x) \notin m_2(le_2(k_{12}(S_1)))$. By commutativity of the involved morphisms, $x \notin m'_2(le_2(k_{12}(e(S'_1))))$. Hence, every edge that is incident to x also stems from $L_2 \setminus K_2$ and is deleted by application of r_2 at match m'_2 as well. □

Proof (of Proposition 5). Consider s_1 as minimal conflict reason for (r_1, r_2), then we know from Corollary 2 that s_1 is also a conflict reason for $(r_1, ND(r_2))$. From Fact 1 we know that each such conflict reason is composed of minimal conflict reasons for $(r_1, ND(r_2))$. Let $M = \{s_i^m | \, i \in I\}$ be this set of minimal conflict reasons for $(r_1, ND(r_2))$ that s_1 is composed of. Assume that s_i^m is a dr conflict part candidate for (r_1, r_2). Since s_i^m is moreover a minimal reason for $(r_1, ND(r_2))$ it is a dr minimal reason for (r_1, r_2) due to Proposition 4. Because of Proposition 1 we know that s_i^m gives rise via pushout construction (overlapping merely s_i^m) to a dr initial conflict. This is a contradiction with s_1 being minimal. □

Proof (of Proposition 6). Let x be an arbitrary element of S_1. First, if x is an edge such that both the source and target nodes of $o_1(x)$ and $q_{12}(x)$ already are elements of B_1 and of K_2, respectively, the subgraph S'_1 of S_1 only consisting of x and its incident nodes, is a minimal conflict reason: Gluing L_1 and L_2 along S'_1 results in a graph to which both r_1 and r_2 are applicable and S'_1 is the according conflict reason. Moreover, S'_1 is obviously minimal.

Secondly, let $x \in S_1$ be any other element. Let $s'_1 : C_1 \overset{o'_1}{\hookleftarrow} S'_1 \overset{q'_{12}}{\rightarrow} L_2$ be a conflict reason that contains x, embeds into s_1, and into which no other conflict reason that contains x embeds. We show that s'_1 is a minimal conflict reason. It is not difficult to see that S'_1 consists of only one connected component and does not contain an edge e such that both the source and target nodes of $o'_1(e)$ and $q'_{12}(e)$ already are elements of B_1 and of K_2. Assume $s_2 : C_1 \overset{o_2}{\hookleftarrow} S_2 \overset{q_{22}}{\rightarrow} L_2$ to be a conflict reason that can be embedded into s'_1 and whose conflict graph S_2 does not contain x. Then $S'_1 \setminus S_2$, defined via initial pushout, contains x and, moreover, by Lemma 3 none of its boundary nodes is to be deleted by either r_1 or r_2 (otherwise, the incident edges would have been elements of S_2). We show $S'_1 \setminus S_2$ to be a conflict reason. This is a contradiction to the minimality of S'_1.

Glue L_1 and L_2 along $S_2 \setminus S'_1$. We show that r_1 and r_2 are applicable to the resulting graph G_2 at the induced matches. The only possibility for r_1 or r_2 to not be applicable is a dangling edge. Let n be a node in G_2 that is to be deleted by r_1. If n does not stem from $S'_1 \setminus S_2$, none of its incident edges does. Thus, every incident edge is to be deleted by r_1 as well. If n stems from $S'_1 \setminus S_2$ it can be a boundary node, i.e., it may also be contained in S_2. Assume n to be a boundary node. As boundary node, an incident edge needs to exist that does not belong to S_2. But since s_2 is a conflict reason, Lemma 3 implies that every edge incident to x also belongs to S_2. Hence, this situation cannot occur. Thus, $x \notin S_2$. But this implies that every edge incident to x is to be deleted as well by r_1—otherwise s'_1 would not be a conflict reason. A completely analogous argument shows that r_2 is applicable to G_2. $\square$