On the Essence and Initiality of Conflicts

Guilherme Grochau Azzi[1] [0000−0002−3740−7002], Andrea Corradini[2] [0000−0001−6123−4175], and Leila Ribeiro[1]

[1] Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil
{ggazzi,leila}@inf.ufrgs.br
[2] Università di Pisa, Pisa, Italy
andrea@di.unipi.it

Abstract. Understanding conflicts between transformations and rules is an important topic in algebraic graph transformation. A conflict occurs when two transformations are not parallel independent, that is, when after applying one of them the other can no longer occur. We contribute to this research thread by proposing a new characterization of the root causes of conflicts, called "conflict essences". By exploiting a recently proposed characterization of parallel independence we easily show that the conflict essence of two transformations is empty iff they are parallel independent. Furthermore we show that conflict essences are smaller than the "conflict reasons" previously proposed, and that they uniquely determine the so-called "initial conflicts". All results hold in categories of Set-valued functors, which include the categories of graphs and typed graphs, and several of them hold in the more general adhesive categories.

Keywords: Graph Transformation · Double-Pushout · Parallel Independence · Conflict · Critical Pair · Initial Conflict

1 Introduction

Graph transformation is a formal model of computation with an intuitive graphical interpretation. Graphs are used to represent states of the system, while possible transitions are represented by transformation rules. The algebraic approach is not restricted to a particular notion of graph: using category theory, its definitions and results can be instantiated for different notions of graph (e.g. labelled, attributed) whose categories satisfy certain axioms.

An important topic in algebraic graph transformation is the study of parallel independence and conflicts. When two transformations are parallel independent they may be applied in any order; if a transformation can no longer happen after applying another, they are in conflict [5]. Understanding conflicts between transformations and rules provides great insight into the behaviour of a transformation system. Indeed, the potential conflicts between two rules, in a minimal context, can be enumerated by *Critical Pair Analysis* (CPA) and used to check local confluence of the system [9]. This has many applications, particularly in Model-Driven Development (e.g. [16,17,6]).

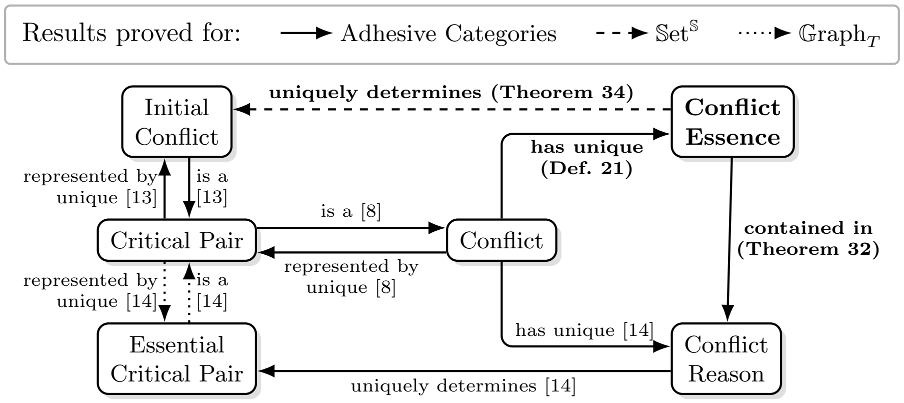


Fig. 1: Overview of conflicts and their root causes, where new concepts and results are in bold.

When two transformations are in conflict, understanding its root causes is often difficult. The formal characterization of *conflict reasons* [14] helps, but it may include elements unrelated to the conflict and lacks a direct connection to the definition of parallel independence. Moreover, critical pairs generated by any two rules are numerous and often redundant, hindering the application of CPA.

An overview of the results presented in this paper and of related concepts from the literature is depicted in Fig. 1. Based on conflict reasons, *essential critical pairs* were proposed as a subset of critical pairs. They were proven complete (for the categories of graphs and typed graphs [14]), in the sense that every critical pair is the embedding of some essential critical pair into a larger context. More recently, *initial conflicts* were also proposed and proven to be a complete subset of critical pairs (in any adhesive category [13]), and were proven to exist for the categories of graphs and typed graphs. Relations between initial conflicts and conflict reasons were not reported, to the best of our knowledge.

In this paper, we contribute to this research thread by proposing a new characterization for the root causes of conflicts, called *conflict essences*, based on a recently proposed characterization of parallel independence [3]. In any adhesive category with strict initial object we show that having an empty conflict essence is equivalent to parallel independence, and that conflict essences are smaller than conflict reasons. In categories of $\mathbb{S}$et-valued functors, we show that conflict essences uniquely determine initial conflicts. Furthermore, we identify sufficient conditions for this to hold in any adhesive category.

The reader should be familiar with basic concepts of Category Theory and with the DPO approach [5]. Some background notions and a motivating example are introduced in section 2. We define conflict essences in section 3, proving important properties and comparing them to conflict reasons. In section 4 we show that essences uniquely determine initial conflicts, and in section 5 we conclude.

2 Preliminaries

2.1 Algebraic Graph Transformation

In this section we briefly review the basic definitions of algebraic graph transformation, according to the Double-Pushout (DPO) approach [5]. We follow the generalization of DPO to work with objects of any adhesive category [12], which include variations of graphs (typed, labelled, attributed), and several other structures. A more recent and detailed introduction to algebraic graph transformation is available in [8], where the theory is generalized to $\mathcal{M}$-adhesive categories, which also encompass structures like Petri nets and algebraic specifications.

We begin by reviewing the notion of adhesive category, which underlies several other definitions, as well as some of its properties. For proofs and a detailed discussion we refer to [12].

Definition 1 (Adhesive Category). *A category $\mathbb{C}$ is called* **adhesive** *if (i) it has all pullbacks, (ii) it has all pushouts along monos, and (iii) such pushouts are* van Kampen (VK) *squares, which implies that they are preserved and reflected by pullbacks [12].*

Fact 2 (Properties of Adhesive Categories). *Let $\mathbb{C}$ be an adhesive category.*

1. *Pushouts along monos in $\mathbb{C}$ are pullbacks.*
2. *For every object C of $\mathbb{C}$, let $\mathbf{Sub}(C)$ be the partial order of its subobjects, i.e. equivalence classes of monic arrows with target C, where $f : A \rightarrowtail C$ and $g : B \rightarrowtail C$ are equivalent if there is an isomorphism $h : A \to B$ making the triangle commute. Then $\mathbf{Sub}(C)$ is a distributive lattice: intersection of subobjects is obtained as the pullback of the corresponding monos, union is obtained from a pushout over the intersection.*

We proceed by reviewing the basic concepts of DPO rewriting in an arbitrary category $\mathbb{C}$. Assumptions on $\mathbb{C}$ will be made explicit when needed.

Definition 3 (Rule, Match and Transformation). *A* **rule** *ρ is a span $\rho = L \overset{l}{\leftarrowtail} K \overset{r}{\rightarrowtail} R$, with monic l and r. We call L and R the* left- *and* right-hand sides*, respectively, while K is called the* interface*. A* **transformation system** *$\mathcal{G}$ is a finite set of rules.*

A **match** *for a rule ρ in an object G is a monic arrow $m : L \rightarrowtail G$. Given a match $m : L \to G$ for rule ρ, a* **transformation** *$G \overset{\rho,m}{\Longrightarrow} H$ corresponds to a diagram (1), where both squares are pushouts. In an adhesive category, the left pushout is guaranteed to be unique up to isomorphism, if it exists.*

$$\begin{tikzcd} L \arrow[d, tail, "m"'] \arrow[dr, phantom, "\urcorner"] & K \arrow[l, tail, "l"'] \arrow[r, tail, "r"] \arrow[d, tail, "k"] \arrow[dr, phantom, "\ulcorner"] & R \arrow[d, tail, "m'"] \\ G & D \arrow[l, tail, "l'"] \arrow[r, tail, "r'"'] & H \end{tikzcd} \quad (1)$$

In the technical development that follows we will use the notions of *strict initial objects* and of *initial pushouts* [9] that we recall here.

Definition 4 (Strict Initial Object). *An object* $\mathbf{0}$ *is* **initial** *in a category* $\mathbb{C}$ *if for each object* C *of* $\mathbb{C}$ *there is a unique arrow* $!_C : \mathbf{0} \to C$*. An initial object* $\mathbf{0}$ *is* **strict** *if for any arrow* $f : X \to \mathbf{0}$*, the source* X *is also initial. If any initial object is strict, all initial objects are, since they are isomorphic. Furthermore, if* $\mathbf{0}$ *is strict, then every arrow* $!_C : \mathbf{0} \to C$ *is mono.*

Many adhesive categories of interest, including the categories of functors presented in the next section, have a strict initial object, but not all of them: for example, the category of sets and partial functions is adhesive, but the initial object is not strict [12].

Definition 5 (Initial Pushout). *Given a morphism* $f : X \to Y$*, the outer rectangle of diagram (2) is an* **initial pushout (over** f**)** *when, for any pushout ① with monic* x *and* y*, there exist unique arrows* b^* *and* c^* *making the diagram commute. The subobject* $b : B \to X$ *is the* **boundary** *of* f*, while* $c : C \to Y$ *is the* **context**.

$$\begin{tikzcd}
B \arrow[r, dotted, tail, "b^*"] \arrow[rr, bend left, tail, "b"] \arrow[d, "f'"'] & U \arrow[r, tail, "x"] \arrow[d, "g"] \arrow[rd, phantom, "\textcircled{1}"] & X \arrow[d, "f"] \\
C \arrow[r, dotted, tail, "c^*"] \arrow[rr, bend right, tail, "c"'] & V \arrow[r, tail, "y"] & Y
\end{tikzcd} \tag{2}$$

The next two properties of initial pushouts will be useful later. For the proof of Lemma 7 we refer to [9].

Lemma 6. *In any category with a strict initial object* $\mathbf{0}$*, the square of diagram (3) is an initial pushout of* f *if and only if* f *is an isomorphism.*

$$\begin{tikzcd}
\mathbf{0} \arrow[r, tail, "!_X"] \arrow[d, "\mathrm{id}_{\mathbf{0}}"'] & X \arrow[d, "f"] \\
\mathbf{0} \arrow[r, tail, "!_Y"'] & Y
\end{tikzcd} \tag{3}$$

Lemma 7. *Initial pushouts are preserved and reflected by pushouts along monos. That is, assuming in diagram (4) that square ② is a pushout with monic* h_L *and* h_K*, and squares ① and ③ are initial pushouts, then there exist unique isomorphisms* b^* *and* c^* *making the diagram commute.*

$$\begin{tikzcd}
\overline{B} \arrow[r, tail, "\overline{b}"] \arrow[rrr, dotted, bend left, tail, "b^*"] \arrow[d, "d"] \arrow[rd, phantom, "\textcircled{1}"] & \overline{X} \arrow[r, tail, "h_K"] \arrow[d, "f"] \arrow[rd, phantom, "\textcircled{2}"] & X \arrow[d, "f"] \arrow[rd, phantom, "\textcircled{3}"] & B \arrow[l, tail, "b"'] \arrow[d, "d"] \\
\overline{C} \arrow[r, tail, "\overline{c}"] \arrow[rrr, dotted, bend right, tail, "c^*"'] & \overline{Y} \arrow[r, tail, "h_L"] & Y & C \arrow[l, tail, "c"']
\end{tikzcd} \tag{4}$$

The categories of graphs and of typed graphs, which are now introduced, are adhesive. Thus, the theory of DPO transformation applies to those categories.

Definition 8 (Categories of Graphs). *A* **graph** $G = (V, E, s, t)$ *has sets* V *of nodes and* E *of edges, along with source and target functions* $s, t : E \to V$*. A* **graph morphism** $f : G \to G'$ *is a pair of functions* $f = (f_V : V \to V', f_E : E \to E')$ *that preserve incidence, that is,* $f_V \circ s = s' \circ f_E$ *and* $f_V \circ t = t' \circ f_E$*. Graphs along with graph morphisms determine the* **category of graphs** $\mathbb{G}\mathrm{raph}$*.*

Given graph T*, called a* type graph*, we can view arrows* $g : G \to T$ *as graphs typed over* T*. The* **category of** T**-typed graphs** *is* $\mathbb{G}\mathrm{raph}_T = \mathbb{G}\mathrm{raph} \downarrow T$*.*

Example 9. As a motivating example, we use a model of an elevator system, which is based on the type graph of Figure 2. The system is composed of multiple floors (⬡) and elevators (🛗). Solid edges between floors indicate the next floor up, while dashed edges indicate that the source floor is below the target. Solid

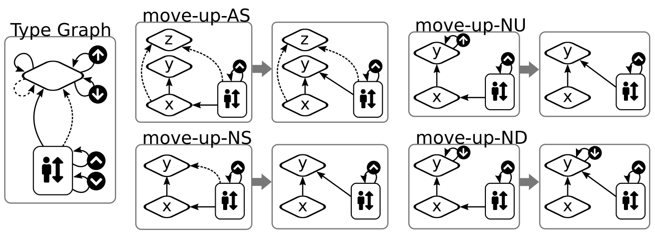


Fig. 2: Type graph and some transformation rules for an elevator system.

edges from an elevator to a floor indicate its position, while a self-loop edge of type ⏶ or ⏷ indicates its direction of motion. People on a floor may request an elevator, which is represented as a self-loop edge of type ⬆ or ⬇, depending on the direction the elevator is expected to go. People inside an elevator may request a stop at a specific floor, which is represented by a dashed arrow from the elevator to the floor.

Due to limited space, we present in Figure 2 only some rules, related to moving the elevator up. Only left- and right-hand sides are shown, their intersection is the interface. The elevator should only move up when given a reason to do so, i.e. some request involving a higher floor. The rule move-up-AS, for example, moves the elevator up one floor given a stop request for some floor above.

Since matches are required to be injective, this rule is not applicable when the requested floor is y. Thus, we must define another rule move-up-NS when the stop request involves the next floor up; then the request is fulfilled and the corresponding edge is deleted. We also depict the rules applicable when the next floor has requested an elevator to move up (move-up-NU) or down (move-up-ND).

2.2 Categories of Set-Valued Functors

Not all results of this paper are proven in terms of adhesive categories, some rely on categories of $\mathbb{S}$et-valued functors. In this section, we show how such categories generalize many important graph models and review some of their properties.

Definition 10 (Set-Valued Functor Category). *Given category $\mathbb{S}$, the category $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ has functors $\mathbb{S} \to \mathbb{S}\mathrm{et}$ as objects and natural transformations as arrows. If $t : F \dot{\to} G$ is a natural transformation between functors $F, G : \mathbb{S} \to \mathbb{S}\mathrm{et}$ and S is an object of $\mathbb{S}$, we denote by $t_S : F(S) \to G(S)$ the component of t on S.*

It is easy to see that the category of graphs is isomorphic to $\mathbb{S}\mathrm{et}^{\mathbb{G}}$, where $\mathbb{G}$ is depicted on diagram (5). A functor $G : \mathbb{G} \to \mathbb{S}\mathrm{et}$ selects two sets $G(V)$ and $G(E)$ as well as two functions $G(s), G(t) : G(E) \to G(V)$. A natural transformation $f : G \dot{\to} G'$ has two components $f_V : G(V) \to G'(V)$ and $f_E : G(E) \to G'(E)$, then naturality corresponds to preservation of incidence.

$$\begin{tikzcd} \mathbb{G} = V \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] & E \end{tikzcd} \quad (5)$$

$\mathbb{S}$et-valued functors generalize *graph structures*, which in turn generalize graphs and many variations (e.g. labelled graphs, hypergraphs, E-graphs) [10]. They are also closed w.r.t. the construction of slice categories.

Definition 11. *A* **graph structure signature** *is an algebraic signature Σ containing only unary operator symbols. A* **graph structure** *is a Σ-algebra for a graph structure signature Σ. The category of algebras for a graph structure signature is then a category of graph structures.*

Lemma 12. *Every category of graph structures is isomorphic to $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ for some small, free category $\mathbb{S}$.*

Proof. A graph structure signature Σ defines a graph by taking sorts as nodes and operation symbols as edges. Let $\mathbb{S}$ be the free category generated by this graph. It is easy to see that the category of Σ-algebras is isomorphic to $\mathbb{S}\mathrm{et}^{\mathbb{S}}$. $\square$

Fact 13. *For any functor category $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ and any object $C : \mathbb{S} \to \mathbb{S}\mathrm{et}$ in it, the slice category $\mathbb{S}\mathrm{et}^{\mathbb{S}} \downarrow C$ is equivalent to a $\mathbb{S}\mathrm{et}$-valued functor category [15].*

Functor categories $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ are particularly well-behaved. They inherit a lot of structure from $\mathbb{S}\mathrm{et}$, and many categorical concepts can be considered pointwise for each object of $\mathbb{S}$. When dealing with such concepts, we will apply set-theoretical reasoning to $\mathbb{S}\mathrm{et}^{\mathbb{S}}$. Then, given $X, Y \in \mathbb{S}\mathrm{et}^{\mathbb{S}}$ and $f : X \to Y$, we will write X, Y and f instead of $X(S)$, $Y(S)$ and f_S for an implicit, universally quantified S.

Fact 14. *In any category of functors $\mathbb{S}\mathrm{et}^{\mathbb{S}}$, limits and colimits are constructed pointwise for each object of $\mathbb{S}$ [15]. In particular, the initial object $\mathbf{0}$ of $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ is strict, composed only of empty sets.*

Fact 15. *In any functor category $\mathbb{S}\mathrm{et}^{\mathbb{S}}$, a morphism $f : X \to Z$ is monic (epic) iff each component f_S is injective (surjective) [15]. A pair of morphisms $X \xrightarrow{f} Z \xleftarrow{g} Y$ is jointly epic iff each pair of components (f_S, g_S) is jointly surjective.*

Fact 16. *Given a small category $\mathbb{S}$, the category of functors $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ is a topos [11]. Then it is adhesive [12] and has unique epi-mono factorisations [11].*

In the context of graph transformation, we often have commutative squares that are both a pullback and a pushout. A set-theoretic characterization will be useful in the following. It underlies a construction of initial pushouts, which we omit due to limited space.

Lemma 17. *In any category of functors $\mathbb{S}\mathrm{et}^{\mathbb{S}}$, if square (6) is a pullback, then it is also a pushout iff both of the following hold for any element $z \in Z$.*

(i) If there is no $x \in X$ with $z = f(x)$, then there is a unique $y \in Y$ with $z = g(y)$.
(ii) If there is no $y \in Y$ with $z = g(y)$, then there is a unique $x \in X$ with $z = f(x)$.

$$\begin{tikzcd} W \arrow[r, "g'"] \arrow[d, "f'"'] & X \arrow[d, "f"] \\ Y \arrow[r, "g"'] & Z \end{tikzcd} \tag{6}$$

Lemma 18. *Any category of functors $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ has initial pushouts for all arrows.*

3 The Essence of Conflicting Transformations

An important tool for understanding the behaviour of a transformation system is the notion of *parallel independence*, which ensures that two transformations don't interfere with each other. Essentially, if two transformations $H_1 \overset{t_1}{\Longleftarrow} G \overset{t_2}{\Longrightarrow} H_2$ are parallel independent, then there exist transformations $H_1 \overset{t_2'}{\Longrightarrow} H$ and $H_2 \overset{t_1'}{\Longrightarrow} H$ reaching the same state. If they are not parallel independent, it is said they are *in conflict*.

Understanding the root causes of such conflicts is a subject of ongoing research [14,2]. In this section, we propose a formal characterization of these root causes and compare it to previous work. We show that our characterization has many useful properties, including a direct connection to the definition of parallel independence and being preserved by extension into larger contexts.

We start introducing parallel independence according to the so-called *essential definition* [3].

Definition 19 (Parallel Independence). *A pair of transformations (t_1, t_2) : $H_1 \overset{\rho_1, m_1}{\Longleftarrow} G \overset{\rho_2, m_2}{\Longrightarrow} H_2$ is* **parallel independent** *when, building the pullbacks ①, ② and ③ as in diagram (7), the arrows $K_1L_2 \to L_1L_2$ and $L_1K_2 \to L_1L_2$ are isomorphisms.*[3]

If t_1 and t_2 are not parallel independent, we say they are **in conflict***. When $K_1L_2 \to L_1L_2$ is not an isomorphism, we say t_1* **disables** *t_2; when $L_1K_2 \to L_1L_2$ is not an isomorphism, we say t_2 disables t_1.*

$$\begin{tikzcd}
 & K_1L_2 \arrow[rr, "\cong"] \arrow[d] \arrow[dr, phantom, "\lrcorner\ \textcircled{2}", very near start] & & L_1L_2 \arrow[dl, "p_1"'] \arrow[dr, "p_2"] \arrow[d, phantom, "\textcircled{1}"] & & L_1K_2 \arrow[ll, "\cong"'] \arrow[d] \arrow[dl, phantom, "\textcircled{3}\ \llcorner", very near start] & \\
R_1 \arrow[d, "n_1"'] & K_1 \arrow[l, "r_1"'] \arrow[r, "l_1"] \arrow[d, "k_1"'] & L_1 \arrow[dr, "m_1"] & & L_2 \arrow[dl, "m_2"'] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "k_2"] & R_2 \arrow[d, "n_2"] \\
H_1 & D_1 \arrow[l, "h_1"'] \arrow[rr, "g_1"] & & G & & D_2 \arrow[ll, "g_2"'] \arrow[r, "h_2"] & H_2
\end{tikzcd} \tag{7}$$

We refer the reader to [3] for a proof that this definition is equivalent to the traditional one (see e.g. [8]) if diagrams are taken in an adhesive category and rule morphisms are monic.

Example 20. Figure 3a shows a pair of parallel independent transformations obtained by applying the rules move-up-NU and move-up-ND of Example 9 to two different elevators. In Fig. 3b the transformations caused by the same rules are in conflict, since the edge between elevator and floor is deleted by both rules.

A disabling occurs when some element is matched by both rules and deleted by at least one of them, as illustrated in Example 20. Since matches are monic,

[3] Since l_1 and l_2 are monos, this is equivalent to requiring existence of arrows $L_1L_2 \to K_1$ and $L_1L_2 \to K_2$ making the resulting triangles commute. However, this simpler condition is not helpful for the characterization of conflicts.

they can be interpreted as subobjects of G and the pullback L_1L_2 as their intersection. Pulling it back along l_1 removes exactly the elements that would be deleted by the transformation using ρ_1 and that are matched by m_2. In order to determine such elements, we use an initial pushout over arrow $K_1L_2 \to L_1L_2$ (Def. 5). In fact in a functor category $\mathbb{S}\mathrm{et}^{\mathbb{S}}$ the context of a mono $f : X \rightarrowtail Y$ is the smallest subobject of Y containing all the elements which are not in the image of f. This brings us to the following definition.

$$\begin{tikzcd}
B_1 \arrow[r, "d_1"] \arrow[d, tail, "b_1"] \arrow[dr, phantom, "\textcircled{4}"] & C_1 \arrow[d, tail, "c_1"] & \\
K_1L_2 \arrow[r, dashed, "q_2"] \arrow[d, "q_1"] \arrow[dr, phantom, "\textcircled{2}"] & L_1L_2 \arrow[r, tail, "p_2"] \arrow[d, tail, "p_1"] \arrow[dr, phantom, "\textcircled{1}"] & L_2 \arrow[d, tail, "m_2"] \\
K_1 \arrow[r, "l_1"] & L_1 \arrow[r, tail, "m_1"] & G
\end{tikzcd} \tag{8}$$

Definition 21 (Conflict and Disabling Essence). *In any adhesive category, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a pair of transformations.*

The **disabling essence** $c_1 : C_1 \rightarrowtail L_1L_2$ *for* (t_1, t_2) *is obtained by taking pullbacks* $\textcircled{1}$ *and* $\textcircled{2}$ *as in diagram (8), then the initial pushout* $\textcircled{4}$ *over* q_2.

The **conflict essence** $c : C \to L_1L_2$ *is the union in* $\mathbf{Sub}(L_1L_2)$ *of the disabling essences* c_1 *for* (t_1, t_2) *and* c_2 *for* (t_2, t_1). *Recall from Fact 2 that this is obtained as a pushout over the pullback of* (c_1, c_2).

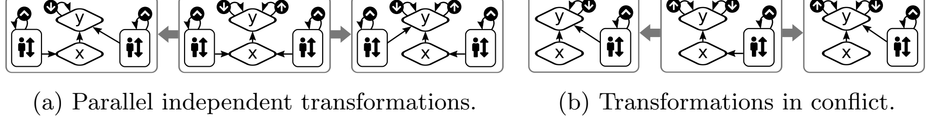
Remark 22. The disabling and conflict essences are also subobjects of G, since the composite $m_1 \circ p_1 \circ c = m_2 \circ p_2 \circ c$ is a monomorphism $C \rightarrowtail G$.

Example 23. The disabling essences for Ex. 20 are constructed in Fig. 4. In Fig. 4a, where transformations are independent, the essence is empty. In Fig. 4b, where a disabling exists, the essence contains an edge from elevator to floor.

From Example 23 we may expect that an empty disabling essence is equivalent to having no disabling. More generally, we can show that this holds whenever the essence is the strict initial object, establishing a direct correspondence between conflict essences and the definition of parallel independence.

Theorem 24. *In any adhesive category with a strict initial object, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a pair of transformations. Then the disabling essence is initial iff* t_1 *doesn't disable* t_2, *and the conflict essence is initial iff* t_1 *and* t_2 *are parallel independent.*

Proof. The case for disabling essences follows directly from Lemma 6: the disabling essence is initial if and only if q_2 in diagram (8) is an isomorphism, which means that t_1 doesn't disable t_2, according to Def. 19. For conflict essences, recall that the strict initial object is the bottom element of the lattice $\mathbf{Sub}(L_1L_2)$.



(a) Parallel independent transformations. (b) Transformations in conflict.

Fig. 3: Examples of parallel independence.

Since $c = c_1 \cup c_2$, C is initial iff C_1 and C_2 are initial, which is equivalent to having no disablings and thus parallel independence. □

Another important property of the disabling essence is that it factors uniquely through the context of the (initial pushout over) arrow $K_1 \overset{l_1}{\to} L_1$. This means that the essence only contains deleted elements, or elements incident to them. The next result is exploited in subsection 3.2 to relate our notion to *disabling reasons* [14]. It would also be the basis for a precise comparison of conflict essences with the *basic conflict conditions* introduced in [2], which is left for future work.

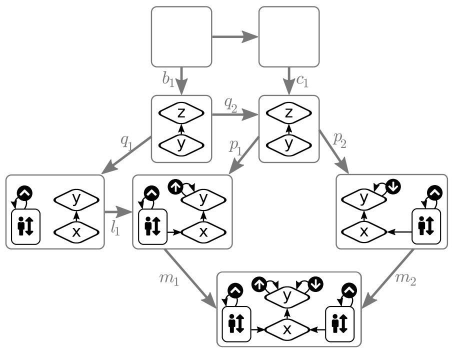
$$\begin{tikzcd}
B_{l1} \arrow[r] \arrow[d, tail] \arrow[dr, phantom, "①"] & C_{l1} \arrow[d, tail, "c_{l1}"'] & C_1 \arrow[l, tail, dotted, "h"'] \arrow[d, tail, "c_1"] \\
K_1 \arrow[r, "l_1"'] & L_1 & L_1L_2 \arrow[l, tail, "p_1"]
\end{tikzcd} \quad (9)$$

Lemma 25. *In any adhesive category, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a transformation pair, let* $c_1 : C_1 \rightarrowtail L_1L_2$ *be its disabling essence and let square* ① *be the initial pushout over* l_1 *in diagram (9). Then the disabling essence of* (t_1, t_2) *factors uniquely through the context* C_{l1} *over* l_1*, i.e., there is a unique mono* $h : C_1 \rightarrowtail C_{l1}$ *with* $p_1 \circ c_1 = c_{l1} \circ h$.

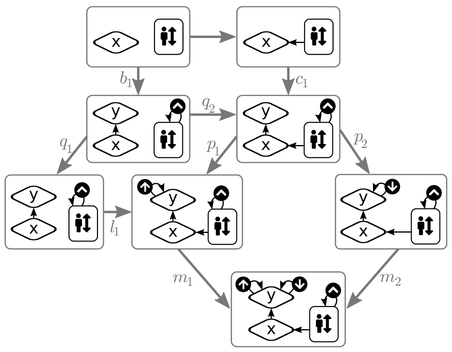
3.1 Conflict Essence and Extension

The *extension* of a transformation into a larger context underlies the concept of *completeness* of critical pairs and initial conflicts: any pair of conflicting transformations is the extension of a critical pair [9,8] and of an initial conflict [13]. It ensures that checking each critical pair (or each initial conflict) for strict confluence guarantees local confluence for the entire transformation system [9,8].

Definition 26 (Extension Diagram). *An* **extension diagram** *over transformation* $t : G \overset{\rho,m}{\Longrightarrow} H$ *and* extension morphism $e : G \to \overline{G}$ *is a diagram (10)*



(a) Disabling essence for Figure 3a.



(b) Disabling essence for Figure 3b.

Fig. 4: Examples of disabling essence.

where $\overline{m} = e \circ m$ is monic and there is a transformation $\overline{t} : \overline{G} \overset{\rho,\overline{m}}{\Longrightarrow} \overline{H}$ defined by the four pushout squares of diagram (11).

$$\begin{tikzcd}
G \arrow[r, Rightarrow, "t"] \arrow[d, "e"] & H \arrow[d, "f"] \\
\overline{G} \arrow[r, Rightarrow, "\overline{t}"] & \overline{H}
\end{tikzcd} \qquad (10)$$

$$\begin{tikzcd}
L \arrow[d, tail, "m"'] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d, tail, "k"] & R \arrow[d, tail, "n"] \\
G \arrow[d, "e"'] & D \arrow[l, "g"'] \arrow[r, "h"] \arrow[d, "d"] & H \arrow[d, "f"] \\
\overline{G} & \overline{D} \arrow[l, "g'"'] \arrow[r, "\overline{h}"] & \overline{H}
\end{tikzcd} \qquad (11)$$

It was already shown that conflicts are *reflected* by extension [13], i.e. when the extension of a transformation pair is in conflict, the original transformation pair is in conflict as well. It turns out they are also *preserved* by extension in categories of set-valued functors, and in particular of graphs and typed graphs. Furthermore, conflict *essences* are also preserved, which means the root causes of a conflict don't change with extension.

Lemma 27 (Essence Inheritance). *In a category of functors $\mathbb{Set}^{\mathbb{S}}$, let $(\overline{t_1}, \overline{t_2}) : \overline{H_1} \overset{\rho_1,\overline{m_1}}{\Longleftarrow} \overline{G} \overset{\rho_2,\overline{m_2}}{\Longrightarrow} \overline{H_2}$ and $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ be two pairs of transformations such that both extension diagrams of (12) exist for some $f : \overline{G} \to G$. Then the transformation pairs have isomorphic conflict and disabling essences. That is, given the mediating morphism $h_L : \overline{L_1L_2} \rightarrowtail L_1L_2$ between the pullback objects and the conflict (disabling) essences $\overline{c}$ of $(\overline{t_1}, \overline{t_2})$ and c of (t_1, t_2), then $h_L \circ \overline{c} \cong c$ in $\mathbf{Sub}(L_1L_2)$.*

$$\begin{tikzcd}
\overline{H_1} \arrow[d] & \overline{G} \arrow[l, Rightarrow, "\overline{t_1}"'] \arrow[r, Rightarrow, "\overline{t_2}"] \arrow[d, "f"] & \overline{H_2} \arrow[d] \\
H_1 & G \arrow[l, Rightarrow, "t_1"'] \arrow[r, Rightarrow, "t_2"] & H_2
\end{tikzcd} \qquad (12)$$

Proof. **[Disabling]** Consider diagram (13) where $\overline{L_1L_2}$ and L_1L_2 are pullback objects for $(\overline{m_1}, \overline{m_2})$ and (m_1, m_2), respectively, and h_L their unique mediating morphism. Constructing the squares ② and ① + ② as pullbacks, by decomposition there is a unique monomorphism h_K making ① a pullback. We will show that ① is also a pushout. Then, since initial pushouts are reflected by pushouts along monomorphisms (Lemma 7), there is an isomorphism $h_C : \overline{C_1} \to C_1$ between the contexts of $\overline{q_2}$ and q_2, with $c_1 \circ h_C = h_L \circ \overline{c_1}$. This makes c_1 and $h_L \circ \overline{c_1}$ isomorphic in $\mathbf{Sub}(L_1L_2)$.

$$\begin{tikzcd}
\overline{K_1L_2} \arrow[r, tail, "h_K"] \arrow[d, tail, "\overline{q_2}"'] \arrow[rr, tail, bend left, "\overline{q_1}"] \arrow[dr, phantom, "\textcircled{1}"] & K_1L_2 \arrow[r, tail, "q_1"] \arrow[d, tail, "q_2"] \arrow[dr, phantom, "\textcircled{2}"] & K_1 \arrow[r, "\overline{k}"] \arrow[d, tail, "l_1"] \arrow[rr, tail, bend left, "k"] \arrow[dr, phantom, "\textcircled{3}"] & \overline{D_1} \arrow[r, "f'"] \arrow[d, tail, "\overline{g}"] \arrow[dr, phantom, "\textcircled{4}"] & D_1 \arrow[d, tail, "g"] \\
\overline{L_1L_2} \arrow[r, tail, "h_L"] \arrow[rr, tail, bend right, "\overline{p_1}"'] & L_1L_2 \arrow[r, tail, "p_1"] & L_1 \arrow[r, tail, "\overline{m_1}"] \arrow[rr, tail, bend right, "m_1"'] & \overline{G} \arrow[r, "f"] & G
\end{tikzcd} \qquad (13)$$

Note that, because the left square of (12) is an extension diagram, there exist pushouts ③ and ④ in diagram (13). Without loss of generality, assume all vertical morphisms of (13) are inclusions, as well as h_L and h_K.

In order to show that pullback $\textcircled{1}$ is also a pushout, by Lemma 17 it suffices to show that (h_L, q_2) is jointly epic. We will show that every element $x \in L_1L_2$ that is not in $\overline{L_1L_2}$ must be in K_1L_2. That is, it must be preserved by step t_1.

So assume such an $x \in L_1L_2 \setminus \overline{L_1L_2}$ and consider the two elements $y_1 = \overline{m_1}(p_1(x)) \in \overline{G}$ and $y_2 = \overline{m_2}(p_2(x)) \in \overline{G}$. These elements of $\overline{G}$ are distinct, but identified by f. In fact, $x \notin \overline{L_1L_2}$ implies that $\overline{m_1}(p_1(x)) \neq \overline{m_2}(p_2(x))$, that is, $y_1 \neq y_2$; instead, since (p_1, p_2) is a pullback of (m_1, m_2), we have $m_1(p_1(x)) = m_2(p_2(x))$ which is equivalent to $f(y_1) = f(y_2)$.

Since $\textcircled{4}$ is a pushout and pullback, and $f(y_1) \in G$ has two distinct preimages by f, it follows from Lemma 17 that $f(y_1)$ has a unique preimage by g. That is, $f(y_1) \in D_1 \subseteq G$. Then, since $\textcircled{2} + \textcircled{3} + \textcircled{4}$ is a pullback and $f(\overline{m_1}(p_1(x))) = f(y_1) = g(f(y_1))$, x must have a preimage by q_2. That is, $x \in K_1L_2 \subseteq L_1L_2$.

In conclusion, every $x \in L_1L_2 \setminus \overline{L_1L_2}$ is such that $x \in K_1L_2$ and therefore $\textcircled{1}$ is a pushout. This implies that q_2 and $\overline{q_2}$ have isomorphic initial pushouts, making $h_L \circ \overline{c_1}$ isomorphic to c_1 in $\mathbf{Sub}(L_1L_2)$.

[Conflict] Follows directly from the previous point. Given the unions c and $\overline{c}$ of $c_1, c_2 \in \mathbf{Sub}(L_1L_2)$ and $\overline{c_1}, \overline{c_2} \in \mathbf{Sub}(\overline{L_1L_2})$, it is trivial to show that $\overline{c} \circ h_L$ is isomorphic to c when $\overline{c_j} \circ h_L$ is isomorphic to c_j for $j \in \{1, 2\}$.

Corollary 28. *In a category of functors* $\mathbb{S}\mathrm{et}^{\mathbb{S}}$*, assume the extension diagrams of (12) exist. Then* t_1 *disables* t_2 *if and only if* $\overline{t_1}$ *disables* $\overline{t_2}$*. Furthermore,* t_1 *and* t_2 *are in conflict if and only if* $\overline{t_1}$ *and* $\overline{t_2}$ *are in conflict.*

3.2 Comparing Reasons and Essences

Conflict essences provide some advantages over the *conflict reason spans* proposed in [14]. In order to simplify the comparison, we introduce conflict and disabling reasons in a slightly different but equivalent way, characterizing them as subobjects of L_1L_2, the pullback object of the matches.

Definition 29 (Disabling Reason). *In an adhesive category, let* $(t_1, t_2) : H_1 \overset{\rho_1, m_1}{\Longleftarrow} G \overset{\rho_2, m_2}{\Longrightarrow} H_2$ *be a pair of transformations.*

The **disabling reason** $s_1 : S_1 \rightarrowtail L_1L_2$ *for* (t_1, t_2) *is the mediating morphism obtained by constructing the initial pushout over* l_1 *as in diagram (14), then the pullbacks* (o_1, s_{12}) *of* $(m_1 \circ c_{l1}, m_2)$ *and* (p_1, p_2) *of the matches.*

A disabling reason satisfies the **conflict condition** *if there is no morphism* $b^* : S_1 \to B_{l1}$ *making diagram (14) commute.*

If $s_1 : S_1 \rightarrowtail L_1L_2$ *and* $s_2 : S_2 \rightarrowtail L_1L_2$ *are the disabling reasons of* (t_1, t_2) *and* (t_2, t_1)*, the* **conflict reason subobject** $s : S \rightarrowtail L_1L_2$ *is constructed as follows. If both* s_1 *and* s_2 *satisfy the conflict condition, then* $s = s_1 \cup s_2$ *in* $\mathbf{Sub}(L_1L_2)$*. If only* s_1 *satisfies the conflict condition, then* $s = s_1$*; analogously for* s_2*.*

$$\begin{tikzcd}
 & & S_1 \arrow[dll, dotted, "b^*"'] \arrow[dl, "o_1"'] \arrow[d, dotted, "s_1"] \arrow[ddr, bend left, "s_{12}"] & \\
B_{l1} \arrow[r, "l_1'"] \arrow[d, "b_{l1}"'] & C_{l1} \arrow[d, "c_{l1}"'] & L_1L_2 \arrow[dl, "p_1"'] \arrow[dr, "p_2"] & \\
K_1 \arrow[r, "l_1"'] & L_1 \arrow[dr, "m_1"'] & & L_2 \arrow[dl, "m_2"] \\
 & & G &
\end{tikzcd}$$

(14)

A conflict reason *span* is defined in [14] as the span $L_1 \overset{c_{l_1}\circ o_1}{\leftarrow} S_1 \overset{s_{12}}{\to} L_2$ if only s_1 satisfies the conflict condition, symmetrically if only s_2 satisfies it, and as a span obtained by building a pushout over a pullback if both satisfy the condition. The equivalence with Def. 29 follows by the bijection between spans of monos commuting with $L_1 \overset{m_1}{\to} G \overset{m_2}{\leftarrow} L_2$ and monos to L_1L_2, and observing that the construction in the third case is identical to that of unions of subobjects.

Fact 30. *Given morphisms $L_1 \overset{m_1}{\to} G \overset{m_1}{\leftarrow} L_2$ with pullback object L_1L_2, the set of spans $L_1 \overset{f}{\leftarrowtail} S \overset{g}{\rightarrowtail} L_2$ commuting with (m_1, m_2) is isomorphic to the set of monos $h : S \rightarrowtail L_1L_2$.*

Proof. Given a span $L_1 \overset{f}{\leftarrowtail} S \overset{g}{\rightarrowtail} L_2$ commuting with (m_1, m_2), there is a unique h making (15) commute. Since $p_1 \circ h$ is monic, h is also monic, thus we constructed a unique mono corresponding to (f, g). Given monic $h : S \rightarrowtail L_1L_2$, we can construct the span $(p_1 \circ h, p_2 \circ h)$. These constructions establish a bijection. □

$$\begin{tikzcd}
 & S \arrow[d, dashed, "h"] \arrow[ddl, bend right, "f"'] \arrow[ddr, bend left, "g"] & \\
 & L_1L_2 \arrow[dl, "p_1"'] \arrow[dr, "p_2"] & \\
L_1 \arrow[dr, "m_1"'] & & L_2 \arrow[dl, "m_2"] \\
 & G &
\end{tikzcd} \tag{15}$$

Note that the relation between disabling reasons and any condition of parallel independence is not very direct. It has been shown that a disabling exists (in the sense of Def. 19) if and only if the reason satisfies the conflict condition [14], but the proof is much more involved than that of Theorem 24.

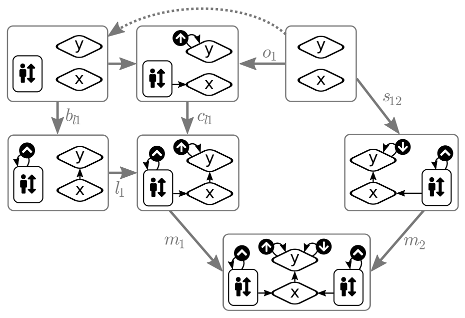
Interestingly, both Def. 29 and Def. 21 use the same operations, but in reversed orders. More explicitly, the disabling reason is obtained by first taking the context of (the initial pushout over) l_1, containing all elements deleted by ρ_1 (and the boundary), and then the intersection with the image of m_2. Instead, the disabling essence first restricts on the elements which are matched by both transformations, and then takes the context, thus filtering out boundary elements of l_1 that are not relevant for the conflict. This suggests that disabling essences are in general smaller than disabling reasons, as illustrated by the following example.

Example 31. The disabling reasons for Example 20 are constructed in Figure 5. In Figure 5a, even though the transformations were independent, the reason contains both floors. In Figure 5b, the floor y is part of the reason despite not being involved in the conflict.

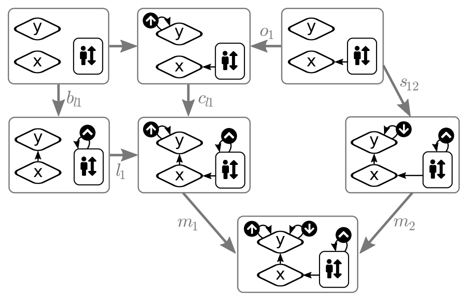
As Example 31 shows, the disabling and conflict reasons may contain elements that aren't directly related to the conflict. In the case of graphs, these are *isolated boundary nodes* [13], i.e. nodes adjacent to a deleted edge where this deletion does not cause a disabling. A comparison with Example 23 indicates that this is not the case for the essences.

The presence of isolated boundary nodes provides another disadvantage: extending a transformation pair may modify the disabling reason by introducing new isolated boundary nodes, as shown in Figure 6. This cannot happen for conflict essences, as proved in Lemma 27.

We conclude this section with a formal proof that every conflict essence is more precise than the corresponding reason, since the former factors uniquely

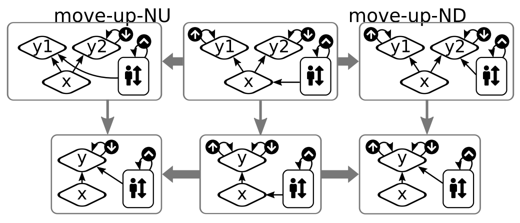


(a) Disabling reason for Figure 3a.

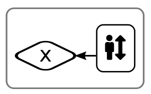


(b) Disabling reason for Figure 3b.

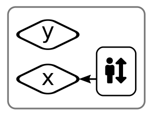
Fig. 5: Examples of disabling reason.



(a) Extension diagrams.



(b) Reason for upper pair.



(c) Reason for lower pair.

Fig. 6: Extended transformations with distinct disabling reasons.

through the latter. Indeed, essences are subobjects of reasons, since the unique factoring is a monomorphism.

Theorem 32 (Precision of Essences). *In any adhesive category, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a pair of transformations with conflict (disabling) essence* $c : C \rightarrowtail L_1L_2$ *and reason* $s : S \rightarrowtail L_1L_2$*. Then* $c \subseteq s$ *in* $\mathbf{Sub}(L_1L_2)$*, that is, there is a unique monomorphism* $h : C \rightarrowtail S$ *such that* $c = s \circ h$*.*

Proof. **[Disabling]** By Lemma 25, there is a unique monomorphism $f : C_1 \rightarrowtail C_{l1}$ with $p_1 \circ c_1 = c_{l1} \circ f$, where C_{l1} is the context of l_1. By Def. 29, the rectangle of diagram (16) is a pullback. Then there is a unique h with $p_2 \circ s_1 \circ h = p_2 \circ c_1$. Since p_2 is monic, we have $s_1 \circ h = c_1$.

[Conflict] If only s_1 satisfies the conflict condition, then $s = s_1$ by Def. 29. In this case t_1 does not disable t_2, thus by Theorem 24 c_2 is the bottom of $\mathbf{Sub}(L_1L_2)$, which implies $c = c_1 \cup \bot = c_1$, and thus $c = c_1 \subseteq s_1 = s$.

$$\begin{tikzcd}
C_1 \arrow[r, dotted, tail, "h"] \arrow[rr, bend left, "c_1"] \arrow[dr, tail, "f"'] & S_1 \arrow[r, tail, "s_1"] \arrow[d, "o_1"] & L_1L_2 \arrow[r, tail, "p_2"] \arrow[d, "p_1"] & L_2 \arrow[d, "m_2"] \\
& C_{l1} \arrow[r, tail, "c_{l1}"] & L_1 \arrow[r, tail, "m_1"] & G
\end{tikzcd} \tag{16}$$

The case when only s_2 satisfies the conflict condition is symmetrical. If both s_1 and s_2 satisfy it, then $s = s_1 \cup s_2$. Since $c_1 \subseteq s_1$ and $c_2 \subseteq s_2$, then it follows from distributivity of $\mathbf{Sub}(L_1L_2)$ that $c = c_1 \cup c_2 \subseteq s_1 \cup s_2 = s$. □

4 Conflict Essences and Initial Conflicts

In this section we show how conflict essences can be used to characterize initial conflicts. Although this is only proven for categories of $\mathbb{S}$et-valued functors, the characterization is stated completely in categorical terms, unlike the previous element-based characterization in the category of typed graphs [13]. Thus, this formulation may be applicable to other categories.

In order to curb the redundancy of critical pairs, a notion of initiality with respect to embedding was introduced in [13].

Definition 33 (Initial Transformation Pair). *Given the pair of transformation steps* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$*, a pair* $(s_1, s_2) : J_1 \overset{\rho_1,n_1}{\Longleftarrow} I \overset{\rho_2,n_2}{\Longrightarrow} J_2$ *is an* **initial transformation pair** *for* (t_1, t_2) *if it satisfies both of the following.*

(i) The pair (s_1, s_2) *can be embedded into* (t_1, t_2) *as in diagram (17).*

(ii) For every pair $(\overline{t_1}, \overline{t_2}) : \overline{H_1} \overset{\rho_1,\overline{m_1}}{\Longleftarrow} \overline{G} \overset{\rho_2,\overline{m_2}}{\Longrightarrow} \overline{H_2}$ *that can be embedded into* (t_1, t_2) *as in diagram (18), then* (s_1, s_2) *can be embedded into* $(\overline{t_1}, \overline{t_2})$*.*

$$\begin{tikzcd}
J_1 \arrow[d] & I \arrow[l, Rightarrow, "s_1"'] \arrow[r, Rightarrow, "s_2"] \arrow[d, dashed, "f"] & J_2 \arrow[d] \\
H_1 & G \arrow[l, Rightarrow, "t_1"] \arrow[r, Rightarrow, "t_2"'] & H_2
\end{tikzcd} \qquad (17)$$

$$\begin{tikzcd}
J_1 \arrow[d] & I \arrow[l, Rightarrow, "s_1"'] \arrow[r, Rightarrow, "s_2"] \arrow[d, dashed, "h"] & J_2 \arrow[d] \\
\overline{H_1} \arrow[d] & \overline{G} \arrow[l, Rightarrow, "\overline{t_1}"'] \arrow[r, Rightarrow, "\overline{t_2}"] \arrow[d, dashed, "g"] & \overline{H_2} \arrow[d] \\
H_1 & G \arrow[l, Rightarrow, "t_1"'] \arrow[r, Rightarrow, "t_2"] & H_2
\end{tikzcd} \qquad (18)$$

Initial transformation pairs are not guaranteed to exist in any category, but they exist in the category of typed graphs [13]. It turns out this is also true for any category of $\mathbb{S}$et-valued functors, where initial conflicts can be constructed as pushouts of the conflict essence.

Theorem 34 (Construction of Initial Transformation Pairs). *In any category of functors* $\mathbb{S}\mathrm{et}^{\mathbb{S}}$*, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a pair of transformations with conflict essence* $c : C \rightarrowtail L_1L_2$*. Then the pushout* $L_1 \overset{n_1}{\to} I \overset{n_2}{\leftarrow} L_2$ *of* $L_1 \overset{p_1 \circ c}{\leftarrow} C \overset{p_2 \circ c}{\to} L_2$ *determines an initial transformation pair* $(s_1, s_2) : J_1 \overset{\rho_1,n_1}{\Longleftarrow} I \overset{\rho_2,n_2}{\Longrightarrow} J_2$ *for* (t_1, t_2)*.*

Proof. We have to show that $L_1 \overset{n_1}{\to} I \overset{n_2}{\leftarrow} L_2$ (i) determines a transformation pair (s_1, s_2) which (ii) can be embedded into (t_1, t_2) via some morphism f and (iii) can be embedded into any other pair of transformation steps $(\overline{t_1}, \overline{t_2})$ that is embedded into (t_1, t_2) via some morphism g.

Note that (iii) follows from (ii). In fact, consider the mediating morphism $p : \overline{L_1L_2} \to L_1L_2$. By essence inheritance, the conflict essence of $(\overline{t_1}, \overline{t_2})$ is $\overline{c} : C \rightarrowtail \overline{L_1L_2}$ with $c = p \circ \overline{c}$. Then (n_1, n_2) is also the pushout of $(\overline{p_1} \circ \overline{c}, \overline{p_2} \circ \overline{c})$, since $\overline{p_1} \circ \overline{c} = p_1 \circ p \circ \overline{c} = p_1 \circ c$ and analogously $\overline{p_2} \circ \overline{c} = p_2 \circ c$. Then by (ii) the transformation determined by (n_1, n_2) can be embedded into $(\overline{t_1}, \overline{t_2})$.

To prove (i) and (ii), note that (n_1, n_2) is jointly epic, since it is a pushout. There is also a unique morphism $f : I \to G$ making diagram (19) commute.

Now consider the diagram (20), where (p_1, p_2) is the pullback of (m_1, m_2) and ① is the initial pushout of q_2. From the transformation step t_1, there is also a pushout ③ + ④, and we can construct a pullback ④. We can also show $n_1 \circ p_1 \circ c_1 = n_1 \circ p_2 \circ c_1$. In fact, since the conflict essence is the union of the disablings, there is $h : C_1 \rightarrowtail C$ with $c_1 = c \circ h$. Then $n_j \circ p_j \circ c_1 = n_j \circ p_j \circ c \circ h$ for $j \in \{1, 2\}$, and $n_1 \circ p_1 \circ c = n_2 \circ p_2 \circ c$ by construction of (n_1, n_2).

$$\begin{tikzcd}
 & C \arrow[dl, tail, "p_1 \circ c"'] \arrow[dr, tail, "p_2 \circ c"] & \\
L_1 \arrow[dr, tail, "n_1"] \arrow[ddr, tail, bend right, "m_1"'] & & L_2 \arrow[dl, tail, "n_2"'] \arrow[ddl, tail, bend left, "m_2"] \\
 & I \arrow[d, dotted, "f"] & \\
 & G &
\end{tikzcd} \quad (19)$$

$$\begin{tikzcd}
 & & L_2 \arrow[dr, tail, "n_2"] & & \\
C_1 \arrow[r, tail, "c_1"] \arrow[dr, phantom, "①"] & L_1L_2 \arrow[ur, tail, "p_2"] \arrow[dr, tail, "p_1"] & & I \arrow[r, "f"] \arrow[dr, phantom, "④"] & G \\
 & & L_1 \arrow[ur, tail, "n_1"] \arrow[ur, phantom, "③"] & & \\
B_1 \arrow[u, tail, "d_1"] \arrow[r, tail, "b_1"'] & K_1L_2 \arrow[u, tail, "q_2"] \arrow[dr, tail, "q_1"'] \arrow[ur, phantom, "②"] & & D'_1 \arrow[uu, tail, "\overline{g}"] \arrow[dr, "f'"] & \\
 & & K_1 \arrow[uu, tail, "l_1"] \arrow[ur, tail, dotted, "k'_1"] \arrow[rr, tail, "k_1"] & & D_1 \arrow[uuu, tail, "g"']
\end{tikzcd} \quad (20)$$

Then, by the following lemma, ③ and ④ are pushouts in diagram (20). Pushout ③ ensures the existence of the transformation step s_1, as required by (i). Pushout ④ ensures that s_1 can be embedded into t_1, as required by (ii). The proof that a transformation step s_2 exists and can be embedded into t_2 is analogous, using the disabling reason of (t_2, t_1). □

Lemma 35 (PO-PB Decomposition by Disabling Essence). *In any category of functors* $\mathbb{S}\mathrm{et}^{\mathbb{S}}$*, assume that in diagram (20) the triangle and squares ①–④ commute and all morphisms except f and f' are monic. Let $m_1 = f \circ n_1$ and $m_2 = f \circ n_2$. Assume also: (n_1, n_2) is jointly epic; (p_1, p_2) is the pullback of (m_1, m_2); $n_1 \circ p_1 \circ c_1 = n_2 \circ p_2 \circ c_2$; ③ + ④ is a pushout; ② and ④ are pullbacks; ① is the initial pushout of q_2. Then both squares ③ and ④ are pushouts.*

Proof. We will show that square ③ is a pushout, which by decomposition implies that ④ is also a pushout. Without loss of generality, assume that all vertical morphisms of diagram (20) as well as c_1 and b_1 are inclusions.

By Lemma 17, it suffices to show that every element $x \in I \setminus n_1(L_1)$ has a unique preimage by $\overline{G}$, that is $x \in D'_1 \subseteq I$. Since (n_1, n_2) is jointly epic and $x \notin n_1(L_1)$, we must have $x \in n_2(L_2)$. Thus, there is $y_2 \in L_2$ with $x = n_1(y_2)$.

Now, consider $f(x) \in G$. Recall that (m_1, g) is a pushout and thus jointly epic, then $f(x)$ must be in the image of m_1 or g. We will show that it is always in the image of g, that is, $f(x) \in D_1$. Then, since ④ is a pullback, $x \in D'_1$.

When $f(x)$ is in the image of m_1 there is $y_1 \in L_1$ with $m_1(y_1) = f(x) = m_2(y_2)$. Then since (p_1, p_2) is a pullback there is a unique $z \in L_1L_2$ with $p_1(z) = y_1$ and $p_2(z) = y_2$. But z cannot be in C_1, otherwise we would have $x = n_2(p_2(c_1(z))) = n_1(p_1(c_1(z)))$, contradicting the assumption that $x \notin n_1(L_1)$. Thus, we must have $z \in K_1L_2$. Then, since squares ②–④ commute in diagram (20), we have $f(x) = k_1(q_1(z)) \in D_1$. □

Note that the proof of Theorem 34 doesn't directly depend on details of the category of functors. Thus, it holds in any category with essence inheritance and PO-PB decomposition by disabling essence.

Conflicting transformation pairs that are themselves initial are called initial conflicts. They provide a suitable subset of critical pairs, being complete in the sense that any conflicting transformation pair is a unique extension of some initial conflict [13]. We provide a simple characterization for initial conflicts.

Definition 36 (Initial Conflict). *A pair of conflicting transformation steps* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *is an* initial conflict *when it is isomorphic to its initial transformation pair.*

Corollary 37. *In a category of functors* $\mathbb{S}\text{et}^{\mathbb{S}}$*, let* $(t_1, t_2) : H_1 \overset{\rho_1,m_1}{\Longleftarrow} G \overset{\rho_2,m_2}{\Longrightarrow} H_2$ *be a pair of transformations with non-empty conflict essence. Then the following are equivalent:*

(i) (t_1, t_2) *is an initial conflict*
(ii) the conflict essence of (t_1, t_2) *is isomorphic to* L_1L_2
(iii) the pullback square to the right is also a pushout.

$$\begin{tikzcd} L_1L_2 \arrow[r, "p_1"] \arrow[d, "p_2"'] & L_1 \arrow[d, "m_1"] \\ L_2 \arrow[r, "m_2"'] & G \end{tikzcd}$$

5 Conclusions

In this paper we have introduced conflict essences as a formal characterization of the root causes of conflicts. We have shown that, in adhesive categories, they are empty if and only if the transformations are parallel independent, and they are not larger than the previously proposed conflict reasons. In categories of set-valued functors, which include the categories of graphs and typed graphs, we have shown that essences are preserved by extension, and that they uniquely determine initial conflicts. We have also identified two sufficient conditions for the existence of initial conflicts in an adhesive category: essence inheritance (Lemma 27) and PO-PB decomposition by disabling essence (Lemma 35).

As future work, we intend to adapt the definitions of this paper to the Sesqui-Pushout approach [4]. In the context of the DPO approach, we intend to apply the conflict conditions of [2] to the essences. From a more practical perspective, it should be possible to improve the efficiency of Critical-Pair Analysis, as implemented by AGG [18] and Verigraph [1] by enumerating conflict essences and initial conflicts instead of critical pairs. Furthermore, integrating this work with constraints and application conditions [7] is important for practical applications.

Acknowledgements

The authors would like to acknowledge the brazilian agency CAPES for their support in the form of scholarships.

References

1. Azzi, G.G., Bezerra, J.S., Ribeiro, L., Costa, A., Rodrigues, L.M., Machado, R.: The verigraph system for graph transformation. In: Graph Transformation, Specifications, and Nets. LNCS, vol. 10800, pp. 160–178. Springer (2018), https://doi.org/10.1007/978-3-319-75396-6_9
2. Born, K., Lambers, L., Strüber, D., Taentzer, G.: Granularity of conflicts and dependencies in graph transformation systems. In: ICGT. LNCS, vol. 10373, pp. 125–141. Springer (2017), https://doi.org/10.1007/978-3-319-61470-0_8
3. Corradini, A., Duval, D., Löwe, M., Ribeiro, L., Machado, R., Costa, A., Azzi, G.G., Bezerra, J.S., Rodrigues, L.M.: On the essence of parallel independence for the double-pushout and sesqui-pushout approaches. In: Graph Transformation, Specifications, and Nets. LNCS, vol. 10800, pp. 1–18. Springer (2018), https://doi.org/10.1007/978-3-319-75396-6_1
4. Corradini, A., Heindel, T., Hermann, F., König, B.: Sesqui-pushout rewriting. In: ICGT. LNCS, vol. 4178, pp. 30–45. Springer (2006), https://doi.org/10.1007/11841883_4
5. Corradini, A., Montanari, U., Rossi, F., Ehrig, H., Heckel, R., Löwe, M.: Algebraic approaches to graph transformation - part I: basic concepts and double pushout approach. In: Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations, pp. 163–246. World Scientific (1997)
6. Cota, É.F., Ribeiro, L., Bezerra, J.S., Costa, A., da Silva, R.E., Cota, G.: Using formal methods for content validation of medical procedure documents. I. J. Medical Informatics 104, 10–25 (2017), https://doi.org/10.1016/j.ijmedinf.2017.04.012
7. Ehrig, H., Ehrig, K., Habel, A., Pennemann, K.: Constraints and application conditions: From graphs to high-level structures. In: ICGT. LNCS, vol. 3256, pp. 287–303. Springer (2004)
8. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. Monographs in Theoretical Computer Science. An EATCS Series, Springer (2006), https://doi.org/10.1007/3-540-31188-2
9. Ehrig, H., Habel, A., Padberg, J., Prange, U.: Adhesive high-level replacement categories and systems. In: ICGT. LNCS, vol. 3256, pp. 144–160. Springer (2004), https://doi.org/10.1007/978-3-540-30203-2_12
10. Ehrig, H., Heckel, R., Korff, M., Löwe, M., Ribeiro, L., Wagner, A., Corradini, A.: Algebraic approaches to graph transformation - part II: single pushout approach and comparison with double pushout approach. In: Handbook of Graph Grammars. pp. 247–312. World Scientific (1997)
11. Johnstone, P.T.: Sketches of an elephant: A topos theory compendium, vol. 2. Oxford University Press (2002)
12. Lack, S., Sobocinski, P.: Adhesive and quasiadhesive categories. ITA 39(3), 511–545 (2005), https://doi.org/10.1051/ita:2005028
13. Lambers, L., Born, K., Orejas, F., Strüber, D., Taentzer, G.: Initial conflicts and dependencies: Critical pairs revisited. In: Graph Transformation, Specifications, and Nets. LNCS, vol. 10800, pp. 105–123. Springer (2018), https://doi.org/10.1007/978-3-319-75396-6_6
14. Lambers, L., Ehrig, H., Orejas, F.: Efficient conflict detection in graph transformation systems by essential critical pairs. ENTCS 211, 17–26 (2008), https://doi.org/10.1016/j.entcs.2008.04.026

15. MacLane, S., Moerdijk, I.: Sheaves in geometry and logic: A first introduction to topos theory. Springer (2012)
16. Mens, T., Taentzer, G., Runge, O.: Analysing refactoring dependencies using graph transformation. Software and System Modeling 6(3), 269–285 (2007), `https://doi.org/10.1007/s10270-006-0044-6`
17. Oliveira Jr., M., Ribeiro, L., Cota, É.F., Duarte, L.M., Nunes, I., Reis, F.: Use case analysis based on formal methods: An empirical study. In: WADT. LNCS, vol. 9463, pp. 110–130. Springer (2014), `https://doi.org/10.1007/978-3-319-28114-8_7`
18. Taentzer, G.: AGG: A graph transformation environment for modeling and validation of software. In: AGTIVE. LNCS, vol. 3062, pp. 446–453. Springer (2003), `https://doi.org/10.1007/978-3-540-25959-6_35`