Transformation and Refinement of Rigid Structures

Citation for published version:
Danos, V, Heckel, R & Sobocinski, P 2014, Transformation and Refinement of Rigid Structures. in H Giese & B König (eds), *Graph Transformation: 7th International Conference, ICGT 2014, Held as Part of STAF 2014, York, UK, July 22-24, 2014. Proceedings.* vol. 8571, Lecture Notes in Computer Science, vol. 8571, Springer International Publishing, pp. 146-160. https://doi.org/10.1007/978-3-319-09108-2_10

Digital Object Identifier (DOI):
10.1007/978-3-319-09108-2_10

Transformation and Refinement of Rigid Structures

Vincent Danos[1], Reiko Heckel[2], and Pawel Sobocinski[3]

[1] School of Informatics, University of Edinburgh
[2] Department of Computer Science, University of Leicester
[3] Electronics and Computer Science, University of Southampton

Abstract. Stochastic rule-based models of networks and biological systems are hard to construct and analyse. Refinements help to produce systems at the right level of abstraction, enable analysis techniques and mappings to other formalisms. Rigidity is a property of graphs introduced in Kappa to support stochastic refinement, allowing to preserve the number of matches for rules in the refined system. In this paper: 1) we propose a notion of rigidity in an axiomatic setting based on adhesive categories; 2) we show how the rewriting of rigid structures can be defined systematically by requiring matches to be open maps reflecting structural features which ensure that rigidity is preserved; and 3) we obtain in our setting a notion of refinement which generalises that in Kappa, and allows a rule to be partitioned into a set of rules which are collectively equivalent to the original. We illustrate our approach with an example of a social network with dynamic topology.

1 Introduction

Graph transformations are a natural model for complex evolving networks including software architectures, social or technical networks, and chemical or biological systems. To address domain-specific requirements, modelling techniques have to tailor their notations, expressivity and analysis tools to a chosen class of problems. While benefiting from concepts and results of the general theory, domain-specific techniques can offer superior capabilities in the chosen domain. In order to avoid reinventing variants of the same concepts, an axiomatic approach to domain-specific graph transformation approaches is advisable.

Kappa [8], a stochastic rewriting approach for graphs representing molecular structures, is a case in point. For a particular class of (hyper)graphs and finely tuned constraints on rules and matches, its techniques for refinement, simulation and analysis [5,9] are significantly more powerful than those for standard (stochastic) graph transformations. An understanding of its relation with mainstream graph transformation is currently emerging (see also Sect. 6). The particular aim of the paper is to develop an axiomatic approach enabling the transfer of Kappa's refinement technique into transformation systems based on adhesive categories. This will allow a more general view of the domain-specific

constraints enabling Kappa's capabilities, which are at the heart of its success with biologists.[4]

One fundamental concept is stochastic rule refinement which, apart from a top-down method of developing models, enables techniques such as the thermodynamic approach [6] and the derivation of differential equations [14,7]. Refinements allow a rule to be replaced by a set of extended rules, jointly equivalent (in the sense of a stochastic bisimulation) to the original. We investigate the conditions under which Kappa-like refinement is possible in an adhesive setting. Alongside we present a model, based on typed attributed graphs, of a social network [11] as an interesting application for stochastic graph transformation.

The paper is organised as follows. We start with our general double-pushout setting: an adhesive ambient category of structures and its subcategories of patterns (e.g., left- and right-hand sides of rules) and states (objects to which the rewriting eventually applies). In Sec. 3 we turn to the fundamental notion of *rigidity*. This is a property of objects similar to the absence of V-structures in graphs [10], where no node is allowed to carry two or more edges unless they are distinguishable by their types or attributes, or those of their target nodes. We show how rigidity can be achieved canonically by placing negative constraints on structures. In Sec. 4 we show how to ensure that these negative constraints (and others) are invariant under rewriting, leading to the systematic extraction of match constraints based on a theory of matches as open maps [16]. With this material in place, we turn to rule refinements, generalising the notion of *growth policy* used in Kappa to specify them and illustrate this by a refinement of the social network model which is thermodynamically consistent in the sense of [6].

2 Structures, Patterns, and States

A type graph defines a structured vocabulary for instance graphs. However, depending on the interpretation of instances as states, patterns or arbitrary structures, they are subject to further constraints. *States* are the most constrained: negatively, by stating the absence of certain structures, or positively, requiring their presence. *Patterns* forming, e.g., the left- and right-hand sides of rules, are subject to negative constraints only, because they represent fragments of states, not deemed to be complete. *Structures* live in an adhesive ambient category for states and patterns. A category is *adhesive* [18] if it has pullbacks as well as pushouts for all pairs of morphisms where one is a mono, and where all such pushouts enjoy the van Kampen property. An example is the category of typed attributed graphs [12].

Assumption 1 *We assume an adhesive category* $\mathbf{C}$ *of* structures *equipped with*

[4] The language was featured in Nature both in July and November 2009, and hailed as one of the future "mainstream components of modern quantitative biology" and the "harbinger of an entirely new way of representing and studying cellular networks" in Nature Methods in 2011.

1. *a full subcategory* $\mathbf{PC}$ *of* $\mathbf{C}$*, called* pattern category*, closed under subobjects: for a monomorphism* $A \to B \in \mathbf{C}$*,* $B \in |\mathbf{PC}|$ *implies* $A \in |\mathbf{PC}|$*.*
2. *a full subcategory* $\mathbf{SC} \subseteq \mathbf{PC}$ *called* state category*.*

Due to 1, if a structure satisfies the constraints for patterns, all its substructures do. That means, such constraints are *negative*, demanding the absence of structure, not their presence. $\mathbf{SC}$ is defined by additional constraints on objects.

Remark 1. It follows that $\mathbf{PC}$ has pullbacks along pairs of morphisms where at least one is mono. They are constructed in $\mathbf{C}$ and, by closure of $\mathbf{PC}$ under subobjects, the pullback object is in $\mathbf{PC}$.

Rules are spans of monomorphisms. Transformations follow the double-pushout approach, with monomorphisms as matches [13]. We use a model of socially-driven evolution of opinions [11] to illustrate our concepts.

Example 1 (typed attributed graph transformation). Our ambient category $\mathbf{C}$ is that of attributed graphs over the type graph [12] in the top left of Fig. 1. The model features agents who vote for one of two parties and can be connected to other agents. We represent votes as node attributes $0, 1$. Connections are

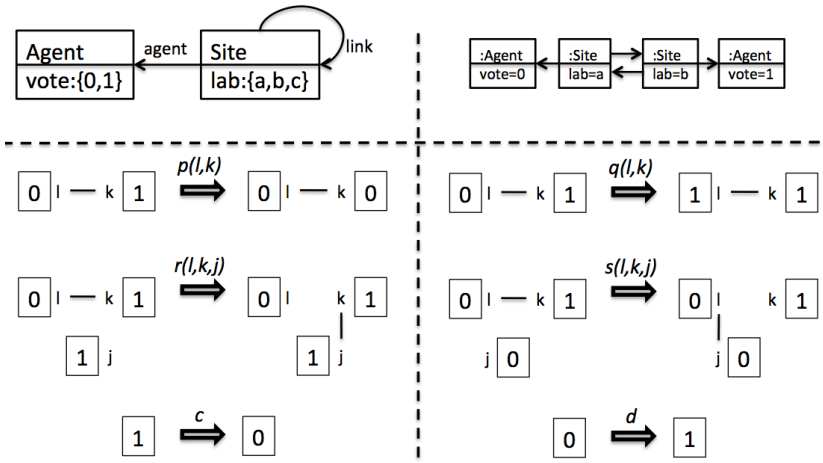


Fig. 1. type graph (top left), instance graph (top right) and rules in concrete syntax, where $l, k, j \in \{a, b, c\}$ (bottom)

identified by labels a, b, c on the sites they are attached to. Once restricted to rigid graphs, this will limit the number of an agent's connections to 3.

In the lower part of Fig. 1 rules are given in a condensed Kappa-like notation. Vote attributes are shown as labels inside Agents. Sites connected to Agents are shown by their labels only. Each Site label is attached to exactly one Agent, leaving agent edges implicit. Link edges are assumed to be symmetric, shown as undirected. To compare, the top right of Fig. 1 shows the left-hand side of rule p as full instance graph (omitting edge types, which can be inferred from sources and targets). Rules are given by rule schemata, e.g., $p(l, k)$ represents all

rules obtained by choosing for l, k any labels from a, b, c. That means, l, k are not variables in the sense of attributed graphs, to be instantiated by matches, but metavariables to express rule schemata.

Rules model the coevolution of votes and connections: if two connected agents hold different votes, either one is converted to the opinion of the other (rules p, q), or the link between them is broken and one makes a new connection to an agent of the same opinion (rules r, s). We also allow spontaneous change of opinion (rules c, d).

We define patterns and states by constraints on structures. A *positive constraint* is a mono $c : P \to Q$, satisfied by an object G if for every mono $P \to G$ there is a mono $Q \to G$ which makes the triangle commute. A *negative constraint* is an object P, satisfied by G if there is no mono $P \to G$. In Assumption 1, the axiomatic treatment abstracts from the way patterns and states are specified, but constraints will be used in our running example. Negative ones play a role in ensuring rigidity.

Example 2 (constraints). Pattern and state constraints are given in the top and

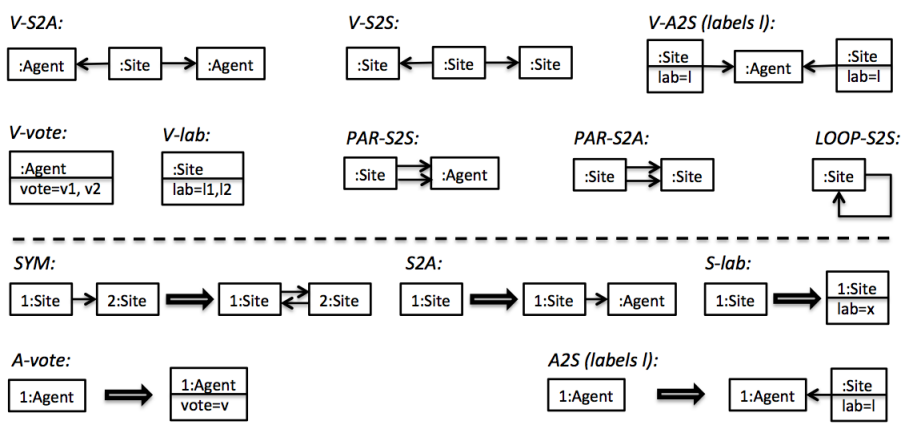


Fig. 2. Constraints on patterns (top) and states (bottom).

bottom of Fig. 2, resp. *Patterns* are subject to negative constraints, expressed by forbidden substructures. They include V-structures, parallel edges and loops, i.e., *V-S2A, V-S2S:* no Site is connected to two Agents nor Sites, *V-A2S:* no Agent is connected to two Sites with the same label, *V-vote:* no Agent has two vote attributes, *V-lab:* no Site has two labels and *PAR-S2A, PAR-S2S, LOOP-S2S:* there are no parallel edges or loops. The pattern category **PC** is the full subcategory of **C** satisfying the negative constraints.

States **SC** form the full subcategory of **PC** defined by constraints *SYM:* link edges are symmetric, *S2A:* every Site is connected to an Agent, *S-lab:* each Site has a label attribute, *A-vote:* each Agent has a vote, *A2S:* for all labels $l \in \{a, b, c\}$, each Agent has a Site labelled l. As before, l is a metavariable expanding *A2S* into three concrete constraints.

For the model presented we are interested in questions such as: What is the evolution over time of the number of agents holding certain votes, or of edges connecting agents of the same vs. those of different votes? What are their resulting long-term ratios? How do these correlate to initial conditions and rates assigned to rules? Kappa provides techniques to explore such questions by extracting differential equations (done manually in [11]) or deriving rates for rules from long-term probabilities of certain patterns. Our aim is to open up these techniques to a wider range of rewriting approaches. The notions in the following section provide the prerequisite.

3 Rigidity

The analysis techniques mentioned above use a notion of rule refinement that preserves the dynamics of the system based on a property of structures called *rigidity*. This helps to ensure that the number of matches for a set of extended rules is the same as for the original.

Assumption 2 *We assume that* $\mathbf{C}$ *is extensive [1]—that is,* $\mathbf{C}$ *has binary coproducts, which are disjoint and stable under pullback*[5]*.*

$$\begin{tikzcd} A_1 \arrow[r] \arrow[d] & B_1 \arrow[d] \\ A \arrow[r, "h"] & B \arrow[r, "\varphi"] & B_1 + B_2 \end{tikzcd}$$

An arrow $h : A \to B \in \mathbf{C}$ *intersects all components of* B iff for all B_1, B_2 and isos $\varphi : B \to B_1 + B_2$ where B_1 is not initial, the pullback object A_1 is not initial. An object C is *rigid* iff all morphisms $h : A \to B$ that intersect all components of B behave like epis for morphisms between B and C, that is, for all $f, g : B \to C$ we have that $f \circ h = g \circ h$ implies $f = g$. A category $\mathbf{C}$ is *rigid* if all its objects are[6] (equivalently: if a morphism in $\mathbf{C}$ is epi iff it intersects all components of its target). The full subcategory of $\mathbf{C}$ of all rigid objects is denoted $Rg(\mathbf{C})$.

An example of a rigid category is **Set**, the category of sets and functions. A non-example is the category of directed multi graphs (V, E, src, tar): the inclusion of the one-vertex graph with no edges to the graph with one vertex and a self-loop is not epi. It is immediate from the definition to show, e.g. that if $\mathbf{C}$ is rigid then $\mathbf{C}/C$ is rigid, for any $C \in \mathbf{C}$.

Example 3 (rigidity). The forbidden pattern *V-A2S* in Fig. 2 provides an example of a non-rigid graph in our sample category $\mathbf{C}$. Assume A to be the graph with a single Agent and let B consist of an Agent and connected Site labelled l. While h clearly intersects the only component of B, there are two different ways

[5] A coproduct is *disjoint* if pulling back the injections yields the initial object, and *stable under pullback* if pulling back the injections along an arbitrary third morphisms always yields a coproduct diagram.

[6] This follows the tradition of regular, extensive and adhesive categories, in that finite colimits (here epis and initial object) are related with finite limits (here pullbacks).

to extend the only morphism from A to C to one from B to C, using the left or right Site node in C to map the single Site in B.

We are interested in rigidity because it gives us a means for refinement via epis. If pattern A is refined by B via epi h, each occurrence of a A in C extends to an occurrence of B in C in at most one way. In our example category **PC** this is because, starting from an agent there is at most one connected site of each label, and starting from a site, there is at most one connected site, attached to a unique agent.

Lemma 1. *In rigid categories, epis are stable under pullbacks along coproduct injections.*

Proof. Suppose that f is epi and the diagram below left is a pullback.

$$\begin{tikzcd} A_1 \arrow[r, "f_1"] \arrow[d] & B_1 \arrow[d] \\ A \arrow[r, "f"'] & B_1 + B_2 \end{tikzcd} \qquad \begin{tikzcd} X_1 \arrow[r, "g_1"] \arrow[d] & Y_1 \arrow[r] \arrow[d] & C_1 \arrow[d, "i_1"] \arrow[ddr, "i_1"] & \\ A_1 \arrow[r, "f_1"] \arrow[d] & B_1 \arrow[r, "\varphi"] \arrow[d] & C_1 + C_2 \arrow[d, "i_1"] & \\ A \arrow[r, "f"'] & B_1 + B_2 \arrow[r, "\varphi + B_2"'] & (C_1 + C_2) + B_2 \arrow[r, "\alpha"'] & C_1 + (C_2 + B_2) \end{tikzcd}$$

We need to show that f_1 is epi. Suppose that $\varphi : B_1 \to C_1 + C_2$ is an isomorphism. Let Y_1 and X_1 be obtained by pulling back. Then, since pullbacks paste together and α is an isomorphism, X_1 is the pullback of $(\alpha(\varphi + B_2))f$ along the injection $C_1 \to C_1 + (C_2 + B_2)$, thus $X_1 \neq 0$, as required.

In the following we limit ourselves to working with rigid patterns.

Assumption 3 **PC** *is rigid, and the inclusion to* **C** *creates coproducts.*

That means, for objects A, B in **PC** their coproduct in **C** is also in **PC**. In case of negative constraints defining patterns, it means that their satisfaction is closed under coproducts. This is ensured if the constraints are connected. An object $A \in \mathbf{C}$ is *connected* if for any coproduct $A_1 + A_2$ isomorphic to A, either $A_1 \cong A$ or $A_2 \cong A$.

Example 4 (rigid structures). The subcategory $\mathbf{PC} \subseteq \mathbf{C}$ defined by the pattern constraints in the top of Fig. 2 is indeed rigid, and since all constraints are connected, it has all coproducts, created by the inclusion to **C**.

These constraints arise in a canonical way, as minimal non-rigid objects in **C**: Disregarding *LOOP-S2S*, the pattern constraints in Fig. 2 represent all minimal non-rigid instance graphs (i.e., they are not rigid and have no proper subgraph that is). The only exception is *V-A2S* which has a non-rigid subgraph obtained by dropping site labels. However, in all our graphs (as patterns, rules, states) all sites carry a label to distinguish sites. *LOOP-S2S* is required because our model does not contain loops.

In general, constraints may arise from our knowledge of the domain, such as with the absence of cycles or non-labelled sites, but may also be derived systematically to ensure rigidity of the resulting category of patterns. One can think of this as a two-step process making explicit the distinction between constraints that are requirements-driven or derived canonically.

Proposition 1 (rigidity is closed under subobjects). *For monos $e : C \to D$, if D is rigid, so is C. That means, $Rg(\mathbf{C})$ is the full subcategory of $\mathbf{C}$ defined by the set of negative constraints consisting of all non-rigid objects in $\mathbf{C}$ that are without non-rigid proper subobjects.*

Proof. If C is not rigid, there exist A, B, h, f, g as above so that $f \circ h = g \circ h$, but $g \neq f$. This extends to a counterexample to D's rigidity because $e \circ f \circ h = e \circ g \circ h$ but, since e is mono, $g \neq f$ implies $e \circ g \neq e \circ f$. □

For example, in the category of multi graphs (i.e., allowing multiple parallel edges), minimal non-rigid graphs are graphs of the form $1 \to 2 \leftarrow 3$ and $1 \leftarrow 2 \to 3$, graphs of two nodes connected by parallel edges, and graphs with a single node and two loops. No graph containing any of these as subgraphs is rigid, restricting us to graphs made up of chains and circles.

The following will be useful later, when considering epis as refinements.

Proposition 2 (epis and monos in rigid categories). *In a rigid category with all coproducts, for every arrow $a : A \to B$ there exists a maximal epi-mono factorisation $A \xrightarrow{e} O \xrightarrow{f} B$, i.e., for every factorisation $A \xrightarrow{e'} O' \xrightarrow{f'} B$ with e' epi, there is a morphism $g : O' \to O$ commuting the resulting triangles. It follows that g is epi, and uniquely determined by e and e'. It is mono if f' is.*

Proof. Let $\mathcal{C}(B)$ be the set of coproduct injections into B that have a non-initial pullback along f. Clearly, $\mathcal{C}(B)$ is closed under binary joins (taken in the lattice of coproduct injections) and is downwards closed by Lemma 1.

Since $\mathbf{C}$ has arbitrary coproducts, it is closed under arbitrary joins, in particular, it has a maximum element X. Thus $B \cong X + Y$ for some Y and, by assumption, $Y \notin \mathcal{C}(B)$.

Hence we have pullback diagrams, from which, using extensivity, $A \cong A' + 0$ and thus φ is an iso.

$$\begin{tikzcd} A' \arrow[r, "a'"] \arrow[d, "\varphi"'] & X \arrow[d, "i"] \\ A \arrow[r, "a"'] & B \\ 0 \arrow[u] \arrow[r] & Y \arrow[u] \end{tikzcd}$$

Let $e = a'\varphi^{-1}$, and $f = i$. It follows from the construction of the maximal element X that the factorisation is in fact the maximal one, i.e., for every other epi-mono factorisation $A \to X' \to B$ there is a unique $X' \to X$ commuting the two triangles. □

In our example category $\mathbf{PC}$ the maximal factorisation is given by cutting off all components of B not intersecting with A. This is the largest O while e is epi, but not the only one. For example, we can reduce O to O' until e' is surjective.

Morphisms in $\mathbf{PC}$ that are mono as well as epi are used as pattern refinements, extending the source by adding connected structure. The category $\mathbf{ME}(A)$ *of mono-epis under* A has as objects all morphisms in $\mathbf{PC}$ starting in A

that are mono and epi. Morphisms in $\mathbf{ME}(A)$ between $b : A \to B$ and $c : A \to C$ are monos $B \to C$ in $\mathbf{PC}$ that commute the triangle.

Remark 2. It follows that arrows in $\mathbf{ME}(A)$ are epis and $\mathbf{ME}(A)$ is a preorder.

4 Matches as Open Maps

Given a rule $p : L \leftarrow K \to R$ in $\mathbf{PC}$ and an object $G \in \mathbf{SC}$, in order to apply p to G, we need a match $m : L \to G$. Apart from the standard gluing conditions, we restrict matches to a subclass of monos that satisfy suitable *reflection constraints*, i.e., matches have to reflect some of the structure of G. If this structure is not present in L, the match is invalid. Absence of structure in L acts like a negative application condition. In [16], such conditions were characterised categorically as *open maps* (used earlier to describe functional bisimulations [17]).

The idea is to use a subcategory $\mathbf{R}$ of $\mathbf{C}$ of "ordinary" morphisms to capture extensions of structure. An arrow $c : P \to Q$ in $\mathbf{R}$ can be seen as an implication saying that, for each occurrence of P in the source of an open morphism $m : X \to Y$, if a corresponding occurrence of Q can be found in Y, then this must give rise to a compatible occurrence of Q in X. If P represents a prefix of a path (e.g., in an LTS) and Q a possible extension, this amounts to a reflecting property for the paths specified. In our case, $c : P \to Q$ represents the forbidden embedding of a pattern into a state graph.

Given a subcategory $\mathbf{R}$ of $\mathbf{C}$, (historically called *path category*) a morphism $m : X \to Y$ is $\mathbf{R}$-*open* if commutative squares mapping a $c : P \to Q$ in $\mathbf{R}$ to m have a *fill-in*, i.e., a morphism f such that the resulting triangles PQX and QXY commute. If $\mathbf{R}$ is understood, we refer to m as *open*.

$$\begin{tikzcd} P \arrow[r] \arrow[d, "c"'] & X \arrow[d, "m"] \\ Q \arrow[r] \arrow[ur, dotted, "f"] & Y \end{tikzcd}$$

Embeddings are monic $\mathbf{R}$-open maps in $\mathbf{C}$ for a given path category $\mathbf{R}$. The (wide) subcategory of $\mathbf{PC}$ with $\mathbf{R}$-open monic morphisms only is called $\mathbf{PC}_{\mathbf{R}}$. We define the path category $\mathbf{R}$ by means of reflection constraints, i.e., morphisms generating $\mathbf{R}$ as the smallest subcategory of $\mathbf{C}$ containing the constraints.

Example 5. The reflection constraint in the top left of Fig. 3 states that there should not be links attached to an agent's sites in a graph if there are no such links in a pattern (i.e., a rule's left-hand side). As before, this is a family of constraints covering all possible labels. Consider the instance *R-AS2S(a,b)* (shown right of the constraint); the match of the *new* rule is not open (shown right), as there is no fill-in of the square. In this case, we see that using that match would lead to a state (bottom right) violating the pattern constraint *V-S2S* of Fig. 2, and non-rigid if $b = c$.

For a rule and negative constraints, it is possible to construct a set of reflection constraints such that, given a match into a graph satisfying the negative constraints, the match satisfies the reflection constraints (is open) iff the transformation does not lead to a graph violating the negative constraints. This

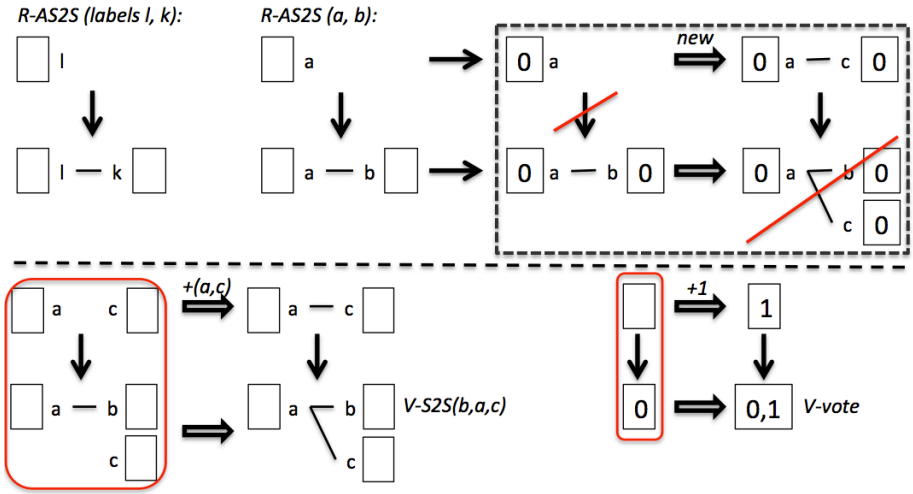


Fig. 3. Reflection constraints and preserving constraints (top) and derivation of reflection constraints from negative constraints and minimal rules (bottom)

provides us with a systematic way of defining the reflection constraints needed to guarantee preservation of rigidity.

We derive reflection constraints for a selection of minimal rules serving as generators for all rules allowed in a model. Reflection constraints that guarantee preservation of rigidity for this selection will be sufficient for all derived rules.

A span $L \xleftarrow{l} K \xrightarrow{r} R$ is a *minimal creation rule* if R is atomic (i.e., cannot be obtained as a union of two proper subobjects), l iso and r mono but not iso; it is a *minimal deletion rule* if L is atomic, r iso and l mono but not iso. An object is *finite* if it has only finitely many subobjects.

Proposition 3. *In an adhesive category* $\mathbf{C}$ *with initial pushouts [12], if* G *and* H *are finite, every transformation* $G \overset{p}{\Longrightarrow} H$ *can be decomposed as a finite sequence of transformations* $G = G_0 \overset{p_1}{\Longrightarrow} \ldots \overset{p_n}{\Longrightarrow} G_n = H$ *via minimal rules* p_i.

Proof. Split the given transformation into a deletion and a creation phase, represented by monos $G \xleftarrow{g} D \xrightarrow{h} H$. By symmetry, it is enough to consider h. Since H is finite, there exists a (non-unique) finite chain of monos:

$$D = D_0 \xrightarrow{h_1} \ldots \xrightarrow{h_n} D_n = H$$

which compose to h such that none of the h_i is iso nor can it be decomposed into $h_i = h_{i2} \circ h_{i1}$ with h_{i2}, h_{i1} not iso. Initial pushouts over $h_{i+1} : D_i \longrightarrow D_{i+1}$ yield transformations $D_i \overset{p_{i+1}}{\Longrightarrow} D_{i+1}$ with $p_{i+1} : L_{i+1} \to R_{i+1}$. These are minimal rules because p_{i+1} is mono and R_{i+1} is atomic: assuming another mono $S \to R_{i+1}$ such that $S \cup L_{i+1} = R_{i+1}$, this results in a pushout pre-composable to the initial one. By initiality therefore, $S \cong R_{i+1}$. $\square$

Hence, in order to guarantee that state properties are invariant, it is sufficient to ensure that they are preserved by a set of minimal rules that can implement all effects of the rules in a model. If we are concerned with negative constraints, it is even enough to consider minimal creation rules.

For a given set of negative constraints $\mathcal{N}$, let the *set of reflection constraints* $\mathbf{R}_{\mathcal{N}}$ be the set of morphisms $L \to O$ as in the diagram with:

$$\begin{tikzcd}
 & L \cap P \arrow[r] \arrow[dl] \arrow[dr] & R \cap P \arrow[dl] \arrow[d] \\
L \arrow[r, "r"] \arrow[d] & R \arrow[d] & P \arrow[dl] \\
O \arrow[r] & U &
\end{tikzcd}$$

- $r : L \to R$ a minimal creation rule
- P in $\mathcal{N}$
- $R \to U \leftarrow P$ jointly epi (their pullback is a pushout)
- the pullback $L \cap P$ of $L \to R \to U \leftarrow P$ a proper subobject of $R \cap P$
- the pushout complement $LROU$ exists
- O satisfies all constraints in $\mathcal{N}$

Theorem 1. *Assume an adhesive category* $\mathbf{C}$ *with initial pushouts and epi-mono factorisation, and let* $\mathcal{N}$ *be a set of negative constraints with* $\mathbf{R}_{\mathcal{N}}$ *constructed for rule* r *as above. Then, for all transformations over finite objects* $G \overset{r,m}{\Longrightarrow} H$ *such that* G *satisfies* $\mathcal{N}$*,* H *satisfies* $\mathcal{N}$ *if* m *is* $\mathbf{R}_{\mathcal{N}}$*-open.*

Proof. Every counterexample, where the result of a transformation using r violates constraint P, can be reduced to U by an epi-mono factorisation, leading to the diagram above since open maps are closed under composition and prefix. □

The reverse is not true because the construction of $\mathbf{R}_{\mathcal{N}}$ does not take into account the individual rules of the system, but works for arbitrary rules. For example, any identical rule would preserve negative constraints even on matches that are not $\mathbf{R}_{\mathcal{N}}$-open.

Example 6 (minimal rules and reflection constraints). At the bottom of Fig 3 we show the two minimal creation rules *+(a,c), +1* that all creation effects in our model are derived from. We only allow to add links between existing nodes (e.g., as part of rules rule changing links) and vote attributes (e.g., in rules changing the attribute). Therefore, deriving reflection constraints for these two we cover all possible cases where negative constraints could be violated.

We obtain two families of constraints, comprising respectively the left-hand side of the minimal rule and the graph it is applied to. The one derived from *+(a,c)* is similar to *R-AS2S(a,b)* above except for the additional agent with site labelled c. We adopt the somewhat stronger constraint which, as seen in the previous example, also addresses node creation, but observe that they are equivalent for all graphs that contain one more agent than *+(a,c)*'s left hand-side, with a c-labelled site attached. The second constraint prevents agents with undefined vote to be mapped to agents with defined vote attribute.

We define the notion of *rigid adhesive structure transformation (RAST)* in **PC** and **SC**. We say that a rule $p = L \leftarrow K \to R$ is *derivable* from a set of rules M if there is a derivation: $L = G_0 \overset{p_1,m_1}{\Longrightarrow} \ldots \overset{p_n,m_n}{\Longrightarrow} G_n = R$ with $p_i \in M$ such that p's span is the composition of the bottom spans of these steps via pullback.

Definition 1 (rigid adhesive structure transformation (RAST)). *Given* $\mathbf{PC}$ *and* $\mathbf{SC}$ *as before and a set of minimal rules* M *with creation rules* $M^+ \subseteq M$*. Let* $\mathcal{N}$ *be the subset of all objects in* $|\mathbf{C}| \setminus |\mathbf{PC}|$ *without proper subobjects in* $\mathbf{PC}$*, and* $\mathbf{R}_{\mathcal{N}}$ *the path category over reflection constraints derived for rules in* M^+ *and constraints* $\mathcal{N}$*.*

The set of permissible rules *in* $\mathbf{PC}/\mathbf{SC}$ *is given by the set of all rules derivable from* M *whose transformations via* $\mathbf{R}_{\mathcal{N}}$*-open maps preserve states, i.e., for all transformations* $G \overset{p,m}{\Longrightarrow} H$ *with* p *derivable from* M *and* m $\mathbf{R}_{\mathcal{N}}$*-open,* $G \in \mathbf{SC}$ *implies* $H \in \mathbf{SC}$*. A* RAST model *is a set of permissible rules, applied using double pushouts in* $\mathbf{C}$ *using open maps as matches.*

Example 7 (RAST). In our model, all rules are derived from the minimal creation rules *+(a,b), +1, +0* and their inverse deletion rules. Note that, for example, deletion rule *-1* by itself does not preserve state constraints because it removes the required vote attribute of an agent, violating constraints *A-vote*. However, the derived rule c in Fig. 1 *is* permissible because it replaces one attribute value by another. Using similar arguments it is easy to see that all our rules in Fig. 1 satisfy this requirement.

As demonstrated by the example and underlying theory, a Kappa-like RAST approach can be constructed systematically as follows.

- Define a type graph able to represent intended structures in rigid graphs.
- Define negative pattern constraints as minimal non-rigid objects (or stronger), as well as additional state constraints.
- Define permissible actions as subset of minimal rules.
- Derive reflection constraints from pattern constraints and minimal rules.
- Derive permissible rules based on minimal rules, state and reflection constraints.

Next we demonstrate how RAST models can be refined thanks to rigidity.

5 Growth Policies and Refinements

A pattern A can be seen as a predicate over objects in $\mathbf{SC}$, validated by a match $a : A \to G$. Refining the pattern, we want to replace A by a set of extensions $e : A \to E$, each representing a stronger predicate, such that their exclusive disjunction is equivalent to A. Intuitively, the set of solutions for A should split into disjoint subsets of solutions for the extended patterns. In other words, for every occurrence $a : A \to G$ there should be a single extension $e : A \to E$ with a unique decomposition of a as $b \circ e$, with $b : E \to G$.

If left-hand sides are taken to be patterns, this ensures that matches of the original rules are in one-to-one correspondence with matches of their refinements. In a stochastic transformation system, one can therefore replace a rule by its refinement without affecting the behaviour as expressed by its Markov chain.

Example 8 (rule refinement). A refinement for rule p of Fig. 1 is shown in Fig. 4. It is easy to check on a couple of examples that matches for p factor uniquely through exactly one of the extensions. Since matches are open maps, reflecting links, if an agent in a rule shows a site that is not connected, this site cannot be connected in the state graph. For example, the rule in the top right cannot be applied to a graph where the agent voting 1 has any further connections, because all three sites are present. Instead, the agent voting 0 in the same rule can have two more connections on the sites not mentioned in the rule. Each of the rules describes exactly one embedding of the original p into an immediate context.

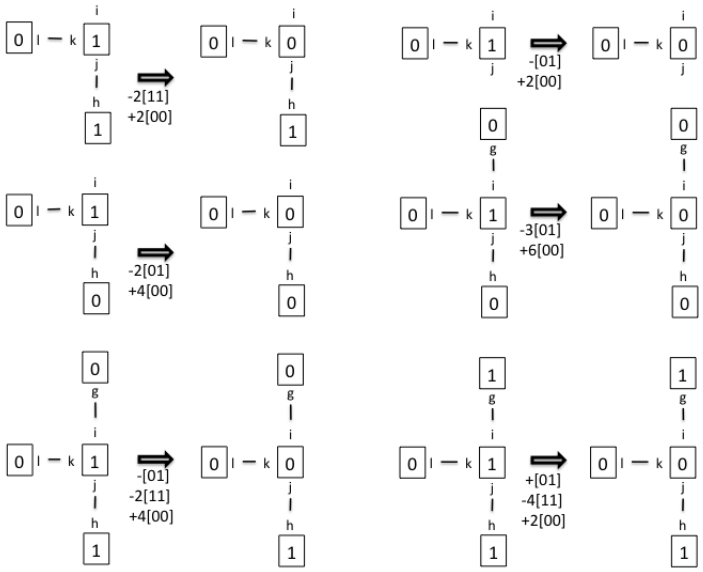


Fig. 4. A refinement for rule p. Labels $l, k, j, i, h, g \in \{a, b, c\}$; i, j, k pairwise distinct. Below each rule, the vector of change in the number of occurrences of the 3 different types of edges is indicated.

Growth policies specify refinements by stating how patterns can be extended. A *growth policy* Γ_A for an object A is a subset of $|\mathbf{ME}(A)|$ such that:
(1) Γ_A is consistent with states: For all $a : A \to G \in \mathbf{PC}$, $G \in |\mathbf{SC}|$, if $a = i \circ b$ is the maximal epi-mono factorisation of a, then $b \in \Gamma_A$.
(2) Γ_A is closed under pullbacks in $\mathbf{ME}(A)$: Given the outer diagram in $\mathbf{PC}$ and let $e : A \to E$ be formed using the pullback of f_1 and f_2 in $\mathbf{PC}$, $e_1, e_2 \in \Gamma_A$ implies $e \in \Gamma_A$.

$$\begin{tikzcd}
 & A \arrow[dl, bend right, "e_1"'] \arrow[d, "e"] \arrow[dr, bend left, "e_2"] & \\
 & E \arrow[dl] \arrow[dr] & \\
E_1 \arrow[dr, "f_1"'] & & E_2 \arrow[dl, "f_2"] \\
 & G &
\end{tikzcd}$$

Both assumptions are met by growth policies in Kappa [6]. (1) means that cutting off disconnected context of a state we obtain an epi satisfying the policy; (2) is a statement of the local nature of policies.

A *refinement of a pattern* A as defined by Γ_A is given by the set of representatives $a \in \Gamma_A$ of all isomorphism classes $[a : A \to B]$ such that for all $a' : A \to B' \in \Gamma_A$, $a' \to a$ implies $a \to a'$. In other words, up to iso a is a

minimal object in Γ_A with respect to the preorder $\mathbf{ME}(A)$. We write $\Gamma(A)$ for the refinement, although it is only defined up to iso.

Theorem 2 (refinement). *Every mono $a : A \to G \in \mathbf{PC}$ with $G \in |\mathbf{SC}|$ factors uniquely as $a = f \circ e$ for a unique $e \in \Gamma(A)$ and monic f.*

$$\begin{tikzcd} A \arrow[r] \arrow[d, "o"] & A \arrow[d, "a"] & 0 \arrow[l] \arrow[d] \\ O \arrow[r, "in"] & O + G' & G' \arrow[l] \end{tikzcd}$$

Proof. We show the existence of a decomposition first. Given $a : A \to G$, let $O + G'$ be a coproduct object in $\mathbf{C}$ isomorphic to G, and such that O is the coproduct of all connected components of G intersecting with a, and G' is the coproduct of the remaining components. In the first diagram, o is obtained via pullback of a and in. Since $\mathbf{C}$ is extensive, pullbacks with coproduct injections yield the top row as a coproduct where, by neutrality of the initial object, the left injection is the identity on A. By construction o is epi, hence in $\mathbf{ME}(A)$, and therefore (by condition 1 above) o is in Γ_A.

$$\begin{tikzcd} E \arrow[d, dashed, "i"] & A \arrow[l, "e"'] \arrow[dl, "o"] \arrow[dd, "a"] \\ O \arrow[dr, "in"'] & \\ & G \end{tikzcd}$$

Let e be the smallest element of Γ_A such that $e \to o$ (existing by condition 2, and unique by construction), and i be the corresponding (unique) morphism in $\mathbf{ME}(A)$. Then, $e \in \Gamma(A)$, and $in \circ i$ is mono since in is a coproduct injection and i is mono (by definition of $\mathbf{ME}(A)$).

$$\begin{tikzcd} & A \arrow[d, "e_0"] \arrow[ddl, bend right, "e"'] \arrow[ddr, bend left, "e'"] & \\ & E_0 \arrow[dl] \arrow[dr] & \\ E \arrow[r, "i"] \arrow[dr, "f"'] & O \arrow[d, "in"] & E' \arrow[l, "j"'] \arrow[dl, "f'"] \\ & G & \end{tikzcd}$$

Suppose now we are given another decomposition $f' \circ e' = a$ with e' in $\Gamma(A)$ and f' mono. The maximal epi-mono factorisation of a via O yields $j : E' \to O$. Its pullback with i in $\mathbf{PC}$ yields a decomposition $A \xrightarrow{e_0} E_0 \to G$ which is in Γ_A by closure under pullbacks. Since f' is mono, so is j and therefore $E_0 \to E$. Since, by assumption, e is minimal, e_0 and e are equal.

Instead of limiting morphisms a, f to be monic, we can choose any restriction to a class of morphisms which "behave like monos", i.e., which is closed under composition, identities, pullbacks and prefixes and includes coproduct injections. The case in point are of course open maps, as shown below.

Proposition 4. $\mathbf{PC_R}$ *has all pullbacks.*

Proof. Pullbacks preserve monos, pullbacks with monos preserve open maps [16]. Monos and open maps decompose so universal arrows are monic and open.

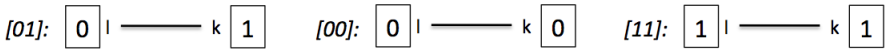
Assumption 4 *We assume that coproduct injections in* $\mathbf{PC}$ *are* $\mathbf{R}$*-open.*

This holds if the target objects of all reflection constraints are connected. With this, the refinements can be applied to open monic matches as they are used in Kappa. Putting everything together we get our notion of refinement:

Corollary 1. *Every monic* $\mathbf{R}$*-open* $a : A \to G \in \mathbf{PC}$ *with* $G \in |\mathbf{SC}|$ *factors uniquely as* $a = f \circ e$ *for a unique* $e \in \Gamma(A)$ *and monic* $\mathbf{R}$*-open* f.

A refinement a of the left-hand side L extends to a rule p if p is applicable to a, i.e., the relevant pushout complement exists. While this is not guaranteed in general, reflection constraints can be used to restrict refinements of L. In our example rules do not delete nodes, so the pushout complement always exists.

Example 9 (growth policy for balanced refinement). The refinement in Fig. 4 is motived by the desire to track the number of links between agents with different votes vs. those of the same vote, counting occurrences of the following patterns.



Depending on the context, p may destroy or create different numbers of occurrences of these patterns. This is what motivates the refinement in Fig. 4: it replaces p with a set of jointly equivalent rules, each of which are *balanced* with respect to [01], [00], [11], i.e., they create or destroy a fixed number of occurrences of each pattern. The *balance* is given below the rule arrow, e.g., the rule in the top right destroys one occurrence of [01] and creates two of [00]. (Occurrences of [11] or [00] always come in pairs because of their symmetry.) The growth policy leading to this refinement is based on analysing partial overlaps of rules and patterns, requiring to extend a rule whenever a pattern overlaps a subrule in a way such that an application would create or destroy the pattern [6].

6 Related Work

Hayman and Heindel [15] developed a categorical generalisation of Kappa in the more general single-pushout setting, but did not consider growth policies and rule refinements. Rather than restricting matches to ensure preservation of constraints such as rigidity, constraints were incorporated into the construction of a transformation as a pushout of partial maps. Instead, we follow [16] in using a double-pushout approach which defines constraints on matches to preserve rigidity. Solutions for (causal) trace compressions introduced in [4] have been re-understood categorically in [3] using fibrations. A concrete fibrational approach to rule restriction was proposed in [2] in a DPO setting. It would be interesting to rethink and generalise this approach following the ideas in this paper.

7 Conclusion

Based on an analysis of the concept of rigid graphs at a categorical level we have defined a generic approach to rigid adhesive structure transformation and a Kappa-like notion of rule refinement based on growth policies. We have also developed a methodology for obtaining concrete instances of the generic approach in a systematic fashion, choosing an adhesive ambient category and defining appropriate constraints on graphs and rules. In addition, we have used a recent example from the literature on complex social networks to serve as an illustration of this methodology and show the concrete interest of our development.

We believe that this approach provides a platform to transfer analysis techniques and tools for stochastic systems from Kappa to adhesive structure transformations. For instance the (infinite) system of differential equations, which is central to the analysis of our example [11], can be now seen as a special case of the fragmentation technique developed in [7] and be produced by mechanical means (up to an arbitrary precision).

References

1. Carboni, A., Lack, S., Walters, R.: Introduction to extensive and distributive categories. Journal of Pure and Applied Algebra 84, 145–158 (1993)
2. Danos, V., Harmer, R., Winskel, G.: Constraining rule-based dynamics with types. Mathematical Structures in Computer Science 23(2), 272–289 (2013)
3. Danos, V., Feret, J., Fontana, W., Harmer, R., Hayman, J., Krivine, J., Thompson-Walsh, C.D., Winskel, G.: Graphs, rewriting and pathway reconstruction for rule-based models. In FSTTCS. LIPIcs, vol. 18, pp. 276–288. (2012)
4. Danos, V., Feret, J., Fontana, W., Harmer, R., Krivine, J.: Rule-based modelling of cellular signalling. In: Caires, L., Vasconcelos, V. (eds.) 18th International Conference on Concurrency Theory. LNCS, vol. 4703, pp. 17–41 (Sep 2007)
5. Danos, V., Feret, J., Fontana, W., Krivine, J.: Scalable simulation of cellular signaling networks. In APLAS. LNCS, vol. 4807, pp. 139–157 (2007)
6. Danos, V., Harmer, R., Honorato-Zimmer, R.: Thermodynamic Graph Rewriting. In CONCUR'13. LNCS, vol. 8052, pp. 380–394. Springer-Verlag (Aug 2013)
7. Danos, V., Honorato-Zimmer, R., Jaramillo-Riveri, S., Stucki, S.: Deriving rate equations for site graph rewriting systems. In: Workshop on Static Analysis and Systems Biology, SASB, Seattle (2013)
8. Danos, V., Laneve, C.: Formal molecular biology. Theor. Comput. Sci. 325(1), 69–110 (2004)
9. Deeds, E., Krivine, J., Feret, J., Danos, V., Fontana, W.: Combinatorial complexity and compositional drift in protein interaction networks. PloS one e32032 (2012)
10. Dorr, H.: Efficient Graph Rewriting and Its Implementation (Lecture Notes in Computer Science). Springer-Verlag (1995)
11. Durrett, R., Gleeson, J., Lloyd, A., Mucha, P., Shi, F., Sivakoff, D., Socoloar, J., Varghese, C.: Graph fission in an evolving voter model. Proceedings of the National Academy of Science 109, 3682–3687 (2012)
12. Ehrig, H., Ehrig, K., Prange, U., Taentzer, G.: Fundamentals of Algebraic Graph Transformation. EATCS Monographs Theor. Comput. Sci., Springer (2006)
13. Ehrig, H., Pfender, M., Schneider, H.: Graph grammars: an algebraic approach. In: IEEE Symposium on Switching and Automata Theory. pp. 167–180. IEEE (1973)
14. Feret, J., Danos, V., Harmer, R., Krivine, J., Fontana, W.: Internal coarse-graining of molecular systems. PNAS 106(16), 6453–8 (Apr 2009)
15. Hayman, J., Heindel, T.: Pattern graphs and rule-based models: The semantics of Kappa. In: Proc. FOSSACS 2013. LNCS, vol. 7794, pp. 1–16. Springer (2013)
16. Heckel, R.: DPO Transformation with Open Maps. In ICGT. LNCS, vol. 7562, pp. 203–217. Springer (2012)
17. Joyal, A., Nielsen, M., Winskel, G.: Bisimulation from open maps. Inf. Comput. 127(2), 164–185 (1996)
18. Lack, S., Sobociński, P.: Adhesive Categories. In: Proc. FOSSACS 2004. LNCS, vol. 2987, pp. 273–288. Springer (2004)