Rate Equations for Graphs

Vincent Danos, Tobias Heindel, Ricardo Honorato-Zimmer, Sandro Stucki

Rate Equations for Graphs*

Vincent Danos[1], Tobias Heindel[2], Ricardo Honorato-Zimmer[3], and Sandro Stucki[4]

[1] CNRS, ENS-PSL, INRIA, Paris, France, `vincent.danos@ens.fr`
[2] Institute of Commercial Information Technology and Quantitative Methods, Technische Universität Berlin, Germany, `heindel@tu-berlin.de`
[3] Centro Interdisciplinario de Neurociencia de Valparaíso, Universidad de Valparaíso, Chile, `ricardo.honorato@cinv.cl`
[4] Dept. of Computer Science and Engineering, University of Gothenburg, Sweden, `sandro.stucki@gu.se`

Abstract. In this paper, we combine ideas from two different scientific traditions: 1) graph transformation systems (GTSs) stemming from the theory of formal languages and concurrency, and 2) mean field approximations (MFAs), a collection of approximation techniques ubiquitous in the study of complex dynamics. Using existing tools from algebraic graph rewriting, as well as new ones, we build a framework which generates rate equations for stochastic GTSs and from which one can derive MFAs of any order (no longer limited to the humanly computable). The procedure for deriving rate equations and their approximations can be automated. An implementation and example models are available online at https://rhz.github.io/fragger. We apply our techniques and tools to derive an expression for the mean velocity of a two-legged walker protein on DNA.

Keywords: Mean Field Approximations · Graph Transformation Systems · Algebraic Graph Rewriting · Rule-based Modelling

1 Introduction

Mean field approximations (MFAs) are used in the study of complex systems to obtain simplified and revealing descriptions of their dynamics. MFAs are used in many disparate contexts such as Chemical Reaction Networks (CRNs) and their derivatives [32,21,11], walkers on bio-polymers [22,42], models of epidemic spreading [25], and the evolution of social networks [18]. These examples witness both the power and universality of MFA techniques, and the fact that they are pursued in a seemingly ad hoc, case-by-case fashion.

The case of CRNs is particularly interesting because they provide a human-readable, declarative language for a common class of complex systems. The

* This is a preprint of a paper to be presented at the 18th International Conference on Computational Methods in Systems Biology (CMSB 2020). The conference version will be published in Springer's LNCS/LNBI series. RH-Z was supported by ANID FONDECYT/POSTDOCTORADO/No3200543.

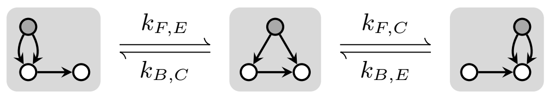


Fig. 1: Stukalin model of a walking DNA bimotor.

stochastic semantics of a CRN is given by a continuous-time Markov chain (CTMC) which gives rise to the so-called *master equation* (ME). The ME is a system of differential equations describing the time evolution of the probability of finding the CRN in any given state. Various tools have been developed to automate the generation and solution of the ME from a given CRN, liberating modellers from the daunting task of working with the ME directly (e.g. [23,43,34]).

Its high dimensionality often precludes exact solutions of the ME. This is where MFA techniques become effective. The generally countably infinite ME is replaced by a finite system of differential equations, called the *rate equations* (RE) [32,26], which describe the time evolution of the average occurrence count of individual species. Here, we extend this idea to the case of graphs and, in fact, the resulting framework subsumes all the examples mentioned above (including CRNs). The main finding is summarised in a single equation (15) which we call the *generalised rate equation for graphs* (GREG). In previous work, we have published a solution to this problem for the subclass of reversible graph rewriting systems [15,16]. The solution presented here is valid for *any* such system, reversible or not. The added mathematical difficulty is substantial and concentrates in the backward modularity Lemma 2. As in Ref. [16], the somewhat informal approach of Ref. [15] is replaced with precise category-theoretical language with which the backward modularity Lemma finds a concise and natural formulation.

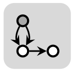
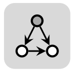
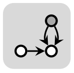
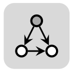
As the reader will notice, Equation (15) is entirely combinatorial and can be readily implemented. Our implementation can be played with at https://rhz.github.io/fragger. Its source can be found at https://github.com/rhz/fragger.

1.1 Two-legged DNA walker

Let us start with an example from biophysics [42]. The model describes a protein complex walking on DNA. The walker contains two special proteins – the *legs* – each binding a different DNA strand. The legs are able to move along the strands independently but can be at most m DNA segments apart.

Following Stukalin et al. [42], we are interested in computing the velocity at which a two-legged walker moves on DNA with $m = 1$. In this case, and assuming the two legs are symmetric, there are only two configurations a walker can be in: either extended (E) or compressed (C). Therefore all possible transitions can be compactly represented by the four rules shown in Fig. 1, where the grey node represents the walker and white nodes are DNA segments. The polarisation of the DNA double helix is represented by the direction of the edge that binds two consecutive DNA segments. Rules are labelled by two subscripts: the first tells us if the leg that changes position is moving *forward* (F) or *backward* (B), while the second states whether the rule extends or compresses the current configuration.

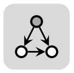
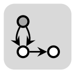
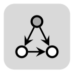
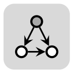
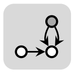
The *mean velocity* V of a single walker in the system can be computed from the rates at which they move forward and backward and their expected number of occurrences $\mathbb{E}[G_i]$, where G_i is in either of the three possible configurations depicted in Fig. 1, and $[G_i]$ is short for $[G_i]\,(X(t))$, the integer-valued random variable that tracks the number of occurrences of G_i in the (random) state of the system $X(t)$ at time t. We call any real- or integer-valued function on $X(t)$ an *observable*.

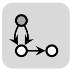
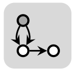
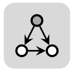
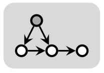
$V = \frac{1}{2}\Big(k_{F,E}\mathbb{E}\Big[$  $\Big] + k_{F,C}\mathbb{E}\Big[$  $\Big] - k_{B,E}\mathbb{E}\Big[$  $\Big] - k_{B,C}\mathbb{E}\Big[$  $\Big]\Big)$

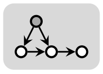
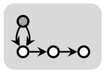
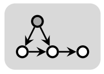
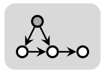
In the case there is only a single motor in the system, the observables $[G_i]$ are Bernoulli-distributed random variables, and the expectations $\mathbb{E}[G_i]$ correspond to the probabilities of finding the motor in the configuration G_i at any given time. Thus by constructing the ODEs for these observables, we can compute the mean velocity of a single motor in the system. That is, we must compute the rate equations for these graphs.

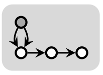
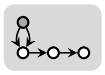
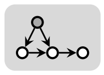
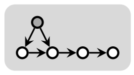
Intuitively, to compute rate equations we must find all ways in which the rules can create or destroy an occurrence of an observable of interest. When, and only when, a rule application and an occurrence of the given observable overlap, can this occurrence be created or destroyed. A systematic inventory of all such overlaps can be obtained by enumerating the so-called *minimal gluings* (MGs) of the graph underlying the given observable and the left- and right-hand sides of each rule in the system. MGs show how two graphs can overlap (full definition in the next section). Such an enumeration of MGs is shown in Fig. 2, where the two graphs used to compute the MGs are the extended walker motif – the middle graph in Fig. 1 – and the left-hand side of the forward-extension rule. The MGs are related and partially ordered by graph morphisms between them.

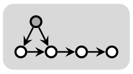
In theory, since we are gluing with the left-hand side of a rule each one of the MGs represents a configuration in which the application of the rule might destroy an occurrence of the observable. However, if we suppose that walkers initially have two legs, then 13 of the 21 MGs in Fig. 2 are impossible to produce by the rules, because no rule can create additional legs. Therefore those configurations will never be reached by the system and we can disregard them. If we further suppose the DNA backbone to be simple and non-branching, we eliminate three more gluings. Finally, if there is only one motor, the remaining four non-trivial gluings are eliminated. In this way, invariants can considerably reduce the number of gluings that have to be considered. Removing terms corresponding to observables which, under the assumptions above, are identically zero, we get the following series of ODEs. For readability, only a subset of the terms is shown, and we write G instead of the proper $\mathbb{E}[G]$ in ODEs.

$\frac{d}{dt}$  $= k_{F,E}$  $-k_{B,C}$  $-k_{F,C}$  $+k_{B,E}$ 

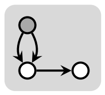
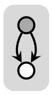
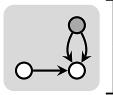
$\frac{d}{dt}$  $= -k_{F,E}$  $+k_{B,C}$  $+k_{F,C}$  $-\dots$

$\frac{d}{dt}$  $= \; k_{F,E}$  $-k_{B,C}$  $-k_{F,C}$  $+\dots$

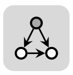
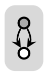
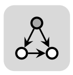
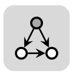
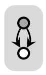
$\frac{d}{dt}$  $= -k_{F,E}$  $+k_{B,C}$  $+k_{F,C}$  $-\dots$

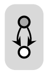
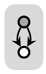
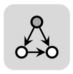
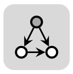
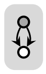
$\frac{d}{dt}$  $= \; \dots$

Notice how only graphs with extra white nodes to the right are obtained when computing the ODE for the left graph in Fig. 1. The opposite is true for the right graph in Fig. 1. This infinite expansion can be further simplified if we assume the DNA chain to be infinite or circular. In this case we can avoid boundary conditions and replace the left- and right-hand observables below by the simpler middle observable:

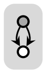
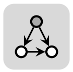
$\mathbb{E}\Big[$  $\Big] = \mathbb{E}\Big[$  $\Big] = \mathbb{E}\Big[$  $\Big]$

The infinite expansion above now boils down to a simple finite ODE system.

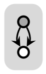
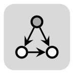
$\frac{d}{dt}$  $= \; k_{F,E}$  $-k_{B,C}$  $-k_{F,C}$  $+k_{B,E}$ 

$\frac{d}{dt}$  $= -k_{F,E}$  $+k_{B,C}$  $+k_{F,C}$  $-k_{B,E}$ 

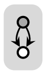
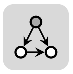
From the above ODEs and assumptions, we get the steady state equation.

$(k_{F,E} + k_{B,E})\mathbb{E}\Big[$  $\Big] = (k_{F,C} + k_{B,C})\mathbb{E}\Big[$  $\Big]$

Since we have only one motor,

$\mathbb{E}\Big[$  $\Big] + \mathbb{E}\Big[$  $\Big] = 1$

Using this, we can derive the steady state value for the mean velocity:

$V = \frac{1}{2}\Big((k_{F,E} - k_{B,E})\mathbb{E}\Big[$  $\Big] + (k_{F,C} - k_{B,C})\mathbb{E}\Big[$  $\Big]\Big)$

$$= \frac{(k_{F,C} + k_{B,C})(k_{F,E} - k_{B,E}) + (k_{F,E} + k_{B,E})(k_{F,C} - k_{B,C})}{2(k_{F,E} + k_{B,E} + k_{F,C} + k_{B,C})}$$

This exact equation is derived in Ref. [42]. We obtain it as a particular case of the general notion of rate equations for graph explained below. It is worth noting that, despite the simplicity of the equation, it is not easily derivable by hand. This and other examples are available to play with in our web app at https://rhz.github.io/fragger/. The example models include

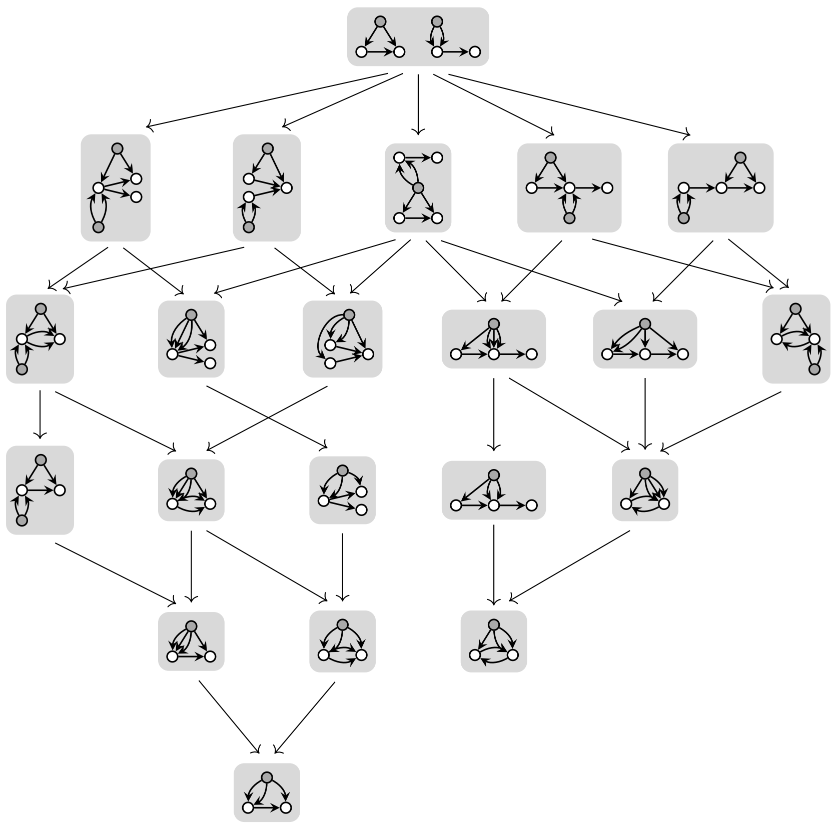


Fig. 2: The poset of minimal gluings of G_2 and G_1. The disjoint sum is at the top. Gluings are layered by the number of node and edge identifications or, equivalently, by the size of their intersection.

- the DNA walker model described above;
- a population model tracking parent-child and sibling relationships;
- the voter model from Ref. [15];
- the preferential attachment model from Ref. [16].

The DNA walker model presented in this introduction is small and reversible. It requires no approximation to obtain a finite expansion. By contrast, the population model and the preferential attachment model are irreversible; the population and the voter model require an approximation to obtain a finite expansion.

1.2 Discussion

The reasoning and derivation done above in the DNA walker can be made completely general. Given a *graph observable* $[F]$, meaning a function counting the number of embeddings of the graph F in the state X, one can build an ODE which describes the rate at which the mean occurrence count $\mathbb{E}([F](X(t)))$ changes over time.

Because the underlying Markov process $X(t)$ is generated by graph-rewriting rules, the one combinatorial ingredient to build that equation is the notion of *minimal gluings* (MGs) of a pair of graphs. Terms in the ODE for F are derived from the set of MGs of F with the left and right sides of the rules which generate $X(t)$. Besides, each term in F's ODE depends on the current state *only* via expressions of the form $\mathbb{E}([G])$ for G a graph defining a new observable. Thus each fresh observable $[G]$ can then be submitted to the same treatment, and one obtains in general a countable system of rate equations for graphs. In good cases (as in the walker example), the expansion is finite and there is no need for any approximation. In general, one needs to truncate the expansion. As the MFA expansion is a symbolic procedure one can pursue it in principle to any order.

The significance of the method hinges both on how many models can be captured in graph-like languages, and how accurate the obtained MFAs are. While these models do no exhaust all possibilities, GTSs seem very expressive. In addition, our approach to the derivation of rate equations for graphs uses a general categorical treatment which subsumes various graph-like structures such as: hyper-graphs, typed graphs, etc. [2,33]. This abstract view is mathematically convenient, and broadens the set of models to which the method applies.

What we know about the existence of solutions to the (in general) countable ODE systems generated by our method is limited. For general countable continuous-time Markov chains and observables, existence of a solution is not guaranteed [41]. Despite their great popularity, the current mathematical understanding of the quality of MFAs only addresses the case of CRNs and density-dependent Markov chains, with Kurtz' theory of scalings [21, Chap. 11], or the case of dynamics on static graphs [25]. Some progress on going beyond the formal point of view and obtaining existence theorems for solutions of REs for graphs were reported in Ref. [14]. Another limitation is the accuracy of MFAs once truncated (as they must be if one wants to plug them in an ODE solver). Even if an MFA can be built to any desired order, it might still fall short of giving a sensible picture of the dynamics of interest. Finally, it may also be that the cost of running a convincing approximation is about the same as that of simulating the system upfront.

1.3 Related work

This paper follows ideas on applying the methods of abstract interpretation to the differential semantics of *site graph* rewriting [28,13,24]. Another more remote influence is Lynch's finite-model theoretic approach to MFAs [37]. From the GTS side, the theory of site graph rewriting had long been thought to be a lucky anomaly until a recent series of work showed that most of its ingredients could be made sense of, and given a much larger basis of applications, through the use of algebraic graph-rewriting techniques [29,3,30]. These latter investigations motivated us to try to address MFA-related questions at a higher level of generality.

Relation to the rule-algebraic approach. Another approach to the same broad set of questions started with Behr et al. introducing ideas from representation theory commonly used in statistical physics [6]. They show that one can study the algebra of rules and their composition law in a way that is decoupled from what is actually being re-written. The "rule-algebraic" theory allows one to derive Kolmogorov equations for observables based on a systematic use of rule commutators (recently implemented in Ref. [10]). Interestingly, novel notions of graph rewriting appear [8]. Partial differential equations describing the generating function of such observables can be derived systematically [7]. As the theory can handle adhesive categories in general and sesqui-pushout rewriting [5], it offers an treatment of irreversible rewrites alternative to the one presented in this paper. (The rule-algebraic approach can also handle application conditions [9]). It will need further work to precisely pinpoint how these two threads of work articulate both at the theoretical and at the implementation levels.

Outline. The paper is organised as follows: §2 collects preliminaries on graph-rewriting and establishes the key *forward* and *backward modularity* lemmas; §3 derives our main result namely a concrete formula for the action of a generator associated to a set of graph-rewriting rules as specified in §2. From this formula, the rate equation for graphs follows easily. Basic category-theoretical definitions needed in the main text are given in App. A; axiomatic proofs in App. B.

2 Stochastic graph rewriting

We turn now to the graphical framework within which we will carry out the derivation of our generalised rate equation (GREG) in §3. We use a categorical approach know as algebraic graph rewriting, specifically the *single pushout (SPO)* approach [35,20]. The reasons for this choice are twofold: first, we benefit from a solid body of preexisting work; second, it allows for a succinct and 'axiomatic' presentation abstracting over the details of the graph-like structures that are being rewritten. Working at this high level of abstraction allows us to identify a set of generic properties necessary for the derivation of the GREG without getting bogged down in the details of the objects being rewritten. Indeed, while we only treat the case of directed multigraphs (graphs with an arbitrary number of directed edges between any two nodes) in this section, the proofs of all lemmas are set in the more general context of *adhesive categories* [33] in App. B. This extends the applicability of our technique to rewrite systems over typed graphs and hypergraphs, among others.

For the convenience of the reader, we reproduce from Ref. [16] our basic definitions for the category **Grph** of *directed multigraphs*. Next, we briefly summarise the SPO approach and its *stochastic semantics* [31]. We conclude with the *modularity lemmas*, which are key to the derivation of the GREG in the next section.

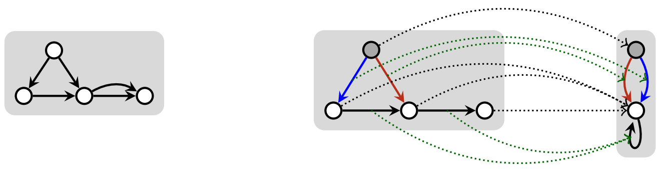


Fig. 3: Examples of a) a directed multigraph, b) a graph morphism.

2.1 The category of directed multigraphs

A *directed multigraph* G consists of a finite set of *nodes* V_G, a finite set of *edges* E_G, and *source* and *target* maps $s_G, t_G \colon E_G \to V_G$. A *graph morphism* $f \colon G \to H$ between graphs G and H is a pair of maps $f_E \colon E_G \to E_H$, $f_V \colon V_G \to V_H$ which preserve the graph structure, i.e. such that for all $e \in E_G$,

$$s_H(f_E(e)) = f_V(s_G(e)) \qquad \text{and} \qquad t_H(f_E(e)) = f_V(t_G(e)).$$

The graphs G and H are called the *domain* and *codomain* of f. A graph morphism $f \colon G \to H$ is a *monomorphism*, or simply a *mono*, if f_V and f_E are injective; it is a *graph inclusion* if both f_V and f_E are inclusion maps, in which case G is a *subgraph* of H and we write $G \subseteq H$. Every morphism $f \colon G \to H$ induces a subgraph $f(G) \subseteq H$ called the *direct image* (or just the *image*) of f in H, such that $V_{f(G)} = f_V(V_G)$ and $E_{f(G)} = f_E(E_G)$. Fig. 3 illustrates a graph and a graph morphism.

A *partial* graph morphism $p \colon G \rightharpoonup H$ is a pair of partial maps $p_V \colon V_G \rightharpoonup V_H$ and $p_E \colon E_G \rightharpoonup E_H$ that preserve the graph structure. Equivalently, p can be represented as a *span* of (total) graph morphisms, that is, a pair of morphisms $p_1 \colon K \to G$, $p_2 \colon K \to H$ with common domain K, where p_1 is mono and K is the *domain of definition* of p. We will use whichever representation is more appropriate for the task at hand. Graphs and graph morphisms form the category **Grph** (with the obvious notion of composition and identity) while graphs and partial graph morphisms form the category $\mathbf{Grph}_*$.

Graph morphisms provide us with a notion of pattern matching on graphs while partial graph morphisms provide the accompanying notion of rewrite rule. We restrict pattern matching to monos: a *match* of a pattern L in a graph G is a monomorphism $f \colon L \to G$. We write $[L, G]$ for the set of matches of L in G. We also restrict rules: a *rule* is a partial graph morphism $\alpha \colon L \rightharpoonup R = (\alpha_1 \colon K \to L, \alpha_2 \colon K \to R)$ where both α_1 and α_2 are monos. We say that L and R are α's left and right hand side (LHS and RHS). Rules are special cases of partial graph morphisms and compose as such. Given a rule $\alpha \colon L \rightharpoonup R = (\alpha_1, \alpha_2)$, we define the *reverse* rule $\alpha^\dagger \colon R \rightharpoonup L$ as the pair $\alpha^\dagger := (\alpha_2, \alpha_1)$, not to be confused with the inverse of α (which does not exist in general). Note that $-^\dagger$ is an involution, that is, $(\alpha^\dagger)^\dagger = \alpha$.

2.2 Graph rewriting

The basic rewrite steps of a GTS are called *derivations*. We first describe them informally. Fig. 4 shows a commutative square, with a match $f \colon L \to G$ on the

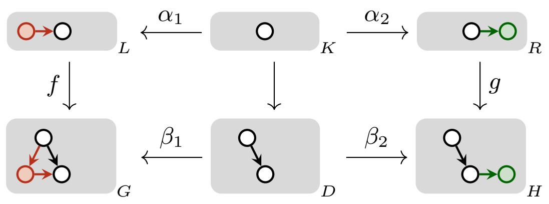


Fig. 4: A derivation.

left and a rule $\alpha\colon L \rightharpoonup R$, on top. The match f identifies the subgraph in G that is to be modified, while the rule α describes how to carry out the modification. In order to obtain the *comatch* $g\colon R \to H$ on the right, one starts by removing nodes and edges from $f(L)$ which do not have a preimage under $f \circ \alpha_1$, as well as any edges left dangling (coloured red in the figure). To complete the derivation, one extends the resulting match by adjoining to D the nodes and edges in R that do not have a preimage under α_2 (coloured green in the figure).

Derivations constructed in this way have the defining property of *pushout squares* (PO) in $\mathbf{Grph}_*$, hence the name SPO for the approach. Alternatively, one can describe a derivation through the properties of its inner squares: the left square is the *final pullback complement* (FPBC) of α_1 and f, while the right one is a PO in $\mathbf{Grph}$ [12]. (Definitions and basic properties of POs and FPBCs are given in App. A.)

Definition 1. *A* derivation *of a* comatch $g\colon R \to H$ *from a match* $f\colon L \to G$ *by a rule* $\alpha = (\alpha_1\colon K \to L, \alpha_2\colon K \to R)$ *is a diagram in* $\mathbf{Grph}$ *such as* (1)*, where the left square is an FPBC of* f *and* α_1 *and the right square is a PO,*

$$\begin{tikzcd} L \arrow[d, "f"'] & K \arrow[l, "\alpha_1"'] \arrow[r, "\alpha_2"] \arrow[d, "h"] & R \arrow[d, "g"] \\ G & D \arrow[l, "\beta_1"] \arrow[r, "\beta_2"'] & H \end{tikzcd} \quad (1) \qquad \begin{tikzcd} L \arrow[r, harpoon, "\alpha"] \arrow[d, "f"'] & R \arrow[d, "g"] \\ G \arrow[r, harpoon, "\beta"'] & H \end{tikzcd} \quad (2)$$

with h*,* g *matches and* $\beta = (\beta_1, \beta_2)$ *a rule, called the* corule *of the derivation.*

Equivalently, a derivation of g from f by α is a PO in $\mathbf{Grph}_*$ as in (2), with corule β. We will mostly use this second characterisation of derivations.

Write $f \Rightarrow_\alpha g$ if there is a derivation of g from f by α. Since derivations are POs of partial morphisms and $\mathbf{Grph}_*$ has all such POs [35], the relation $\Rightarrow_\alpha$ is *total*, that is, for any match f and rule α (with common domain), we can find a comatch g. However, the converse is not true: not every match g having the RHS of α as its domain is a comatch of a derivation by α. Which is to say, there might not exist f such that $f \Rightarrow_\alpha g$ (the relation $\Rightarrow_\alpha$ is not surjective). When there is such an f, we say g is *derivable* by α. Consider the example in Fig. 5. Here, g is α-derivable (as witnessed by f) but h is not: no match of the LHS could contain a "preimage" of the extra (red) edge e in the codomain of h because the target node of e has not yet been created.

We say a derivation $f \Rightarrow_\alpha g$ (with corule β) is *reversible* if $g \Rightarrow_{\alpha^\dagger} f$ (with corule $\beta^\dagger$), and *irreversible* otherwise. Clearly, derivations are not reversible in

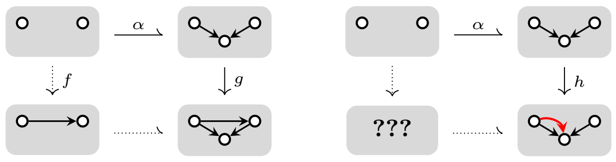


Fig. 5: The match g is α-derivable, while h is not.

general, otherwise $\Rightarrow_\alpha$ would be surjective. Consider the derivation shown in Fig. 4. The derivation removes two (red) edges from the codomain of f; the removal of the lower edge is specified in the LHS of α, whereas the removal of the upper edge is a *side effect* of removing the red node to which the edge is connected (graphs cannot contain dangling edges). Applying the reverse rule $\alpha^\dagger$ to the comatch g restores the red node and the lower red edge, but not the upper red edge. In other words, f is not $\alpha^\dagger$-derivable, hence the derivation in Fig. 4 is irreversible. In previous work, we have shown how to derive rate equations for graph transformation systems with only reversible derivations [13,15,16]. In §3, we overcome this limitation, giving a procedure that extends to the irreversible case.

Since POs are unique (up to unique isomorphism), $\Rightarrow_\alpha$ is also *functional* (up to isomorphism). The fact that derivations are only defined up to isomorphism is convenient as it allows us to manipulate them without paying attention to the concrete naming of nodes and edges. Without this flexibility, stating and proving properties such as Lemmas 2 and 3 below would be exceedingly cumbersome. On the other hand, when defining the stochastic semantics of our rewrite systems, it is more convenient to restrict $\Rightarrow_\alpha$ to a properly functional relation. To this end, we fix once and for all, for any given match $f\colon L \to G$ and rule $\alpha\colon L \rightharpoonup R$, a *representative* $f \Rightarrow_\alpha \alpha(f)$ from the corresponding isomorphism class of derivations, with (unique) comatch $\alpha(f)\colon R \to H$, and (unique) corule $f(\alpha)\colon G \rightharpoonup H$.

A set of rules $\mathcal{R}$ thus defines a *labelled transition system (LTS)* over graphs, with corules as transitions, labelled by the associated pair (f, α). Given a rule $\alpha\colon L \rightharpoonup R$, we define a stochastic rate matrix $Q_\alpha := (q^\alpha_{GH})$ over graphs as follows.

$$\begin{aligned} q^\alpha_{GH} &:= |\{f \in [L, G] \mid \alpha(f) \in [R, H]\}| && \text{for } G \neq H, \\ q^\alpha_{GG} &:= \textstyle\sum_{H \neq G} -q^\alpha_{GH} && \text{otherwise.} \end{aligned} \tag{3}$$

Given a *model*, that is to say a finite set of rules $\mathcal{R}$ and a *rate map* $k\colon \mathcal{R} \to \mathbb{R}^+$, we define the model rate matrix $Q(\mathcal{R}, k)$ as

$$Q(\mathcal{R}, k) := \textstyle\sum_{\alpha \in \mathcal{R}} k(\alpha) Q_\alpha \tag{4}$$

Thus a model defines a CTMC over $\mathbf{Grph}_*$. As $\mathcal{R}$ is finite, $Q(\mathcal{R}, k)$ is row-finite.

2.3 Composition and modularity of derivations

By the well-known Pushout Lemma, derivations can be composed horizontally (rule composition) and vertically (rule specialisation) in the sense that if inner squares below are derivations, so are the outer ones:

$$\begin{tikzcd}
L \arrow[r, "\alpha_1"] \arrow[d, "f"'] & R_1 \arrow[r, "\alpha_2"] \arrow[d, "g_1"] & R_2 \arrow[d, "g_2"] \\
G \arrow[r] & H_1 \arrow[r] & H_2
\end{tikzcd}
\qquad
\begin{tikzcd}
L \arrow[r, "\alpha_1"] \arrow[d, "f_1"'] & R \arrow[d, "g_1"] \\
G_1 \arrow[r, "\alpha_2"] \arrow[d, "f_2"'] & H_1 \arrow[d, "g_2"] \\
G_2 \arrow[r] & H_2
\end{tikzcd}$$

Derivations can also be decomposed vertically. First, one has a forward decomposition (which follows immediately from pasting of POs in $\mathbf{Grph}_*$):

Lemma 1 (Forward modularity). *Let α, β, γ be rules and f_1, f_2, g, g_1 matches such that diagrams (5) and (6) are derivations. Then there is a unique match g_2 such that diagram (7) commutes (in $\mathbf{Grph}_*$) and is a vertical composition of derivations.*

$$\begin{tikzcd}
L \arrow[r, "\alpha"] \arrow[d, "f_1"'] & R \arrow[dd, "g"] \\
S \arrow[d, "f_2"'] & \\
G \arrow[r, "\beta"] & H
\end{tikzcd} (5)
\qquad
\begin{tikzcd}
L \arrow[r, "\alpha"] \arrow[d, "f_1"'] & R \arrow[d, "g_1"] \\
S \arrow[r, "\gamma"] & T
\end{tikzcd} (6)
\qquad
\begin{tikzcd}
L \arrow[r, "\alpha"] \arrow[d, "f_1"'] & R \arrow[d, "g_1"'] \arrow[dd, bend left, "g"] \\
S \arrow[r, "\gamma"] \arrow[d, "f_2"'] & T \arrow[d, dashed, "g_2"'] \\
G \arrow[r, "\beta"] & H
\end{tikzcd} (7)$$

A novel observation, which will play a central role in the next section, is that one also has a backward decomposition:

Lemma 2 (Backward modularity). *Let α, β, γ be rules and f, f_1, g_1, g_2 matches such that diagrams (8) and (9) are derivations. Then there is a unique match f_2 such that diagram (10) commutes (in $\mathbf{Grph}_*$) and is a vertical composition of derivations.*

$$\begin{tikzcd}
L \arrow[r, "\alpha"] \arrow[dd, "f"'] & R \arrow[d, "g_1"] \\
& T \arrow[d, "g_2"] \\
G \arrow[r, "\beta"] & H
\end{tikzcd} (8)
\qquad
\begin{tikzcd}
L \arrow[d, "f_1"'] & R \arrow[l, "\alpha^\dagger"'] \arrow[d, "g_1"] \\
S & T \arrow[l, "\gamma^\dagger"']
\end{tikzcd} (9)
\qquad
\begin{tikzcd}
L \arrow[r, "\alpha"] \arrow[d, "f_1"] \arrow[dd, bend right, "f"'] & R \arrow[d, "g_1"] \\
S \arrow[r, "\gamma"] \arrow[d, dashed, "f_2"] & T \arrow[d, "g_2"] \\
G \arrow[r, "\beta"] & H
\end{tikzcd} (10)$$

Forward and backward modularity look deceptively similar, but while Lemma 1 is a standard property of POs, Lemma 2 is decidedly non-standard. Remember that derivations are generally irreversible. It is therefore not at all obvious that one should be able to transport factorisations of comatches backwards along a rule, let alone in a unique fashion. Nor is it obvious that the top half of the resulting decomposition should be reversible. The crucial ingredient that makes backward modularity possible is that both matches and rules are monos. Because rules are (partial) monos, we can reverse α and β in (8), and the resulting diagram still commutes (though it is no longer a derivation in general). The existence and uniqueness of f_2 is then a direct consequence of the universal property of (9), seen as a PO. The fact that (9) is reversible relies on matches also being monos, but in a more subtle way. Intuitively, the graph T cannot contain any superfluous edges of the sort that render the derivation in Fig. 4 irreversible because, g_2 being a mono, such edges would appear in H as subgraphs, contradicting the α-derivability of $g_2 \circ g_1$. Together, the factorisation of f and the reversibility of (9) then induce the decomposition in (10) by Lemma 1. A full, axiomatic proof of Lemma 2 is given in App. B.3.

Among other things, Lemma 2 allows one to relate *derivability* of matches to *reversibility* of derivations:

Lemma 3. *A match $g \colon R \to H$ is derivable by a rule $\alpha \colon L \rightharpoonup R$ if and only if the derivation $g \Rightarrow_{\alpha^\dagger} f$ is reversible.*

2.4 Gluings

Given $G_1 \subseteq H$ and $G_2 \subseteq H$, the *union* of G_1 and G_2 in H is the unique subgraph $G_1 \cup G_2$ of H, such that $V_{(G_1 \cup G_2)} = V_{G_1} \cup V_{G_2}$ and $E_{(G_1 \cup G_2)} = E_{G_1} \cup E_{G_2}$. The *intersection* $(G_1 \cap G_2) \subseteq H$ is defined analogously. The subgraphs of H form a complete distributive lattice with $\cup$ and $\cap$ as the join and meet operations. One can *glue* arbitrary graphs as follows:

Definition 2. *A* gluing *of graphs G_1, G_2 is a pair of matches $i_1 \colon G_1 \to U$, $i_2 \colon G_2 \to U$ with common codomain U; if in addition $U = i_1(G_1) \cup i_2(G_2)$, one says the gluing is* minimal.

Two gluings $i_1 \colon G_1 \to U$, $i_2 \colon G_2 \to U$ and $j_1 \colon G_1 \to V$, $j_2 \colon G_2 \to V$ are said to be *isomorphic* if there is an isomorphism $u \colon U \to V$, such that $j_1 = u \circ i_1$ and $j_2 = u \circ i_2$. We write $G_1 *_{\simeq} G_2$ for the set of isomorphism classes of minimal gluings (MG) of G_1 and G_2, and $G_1 * G_2$ for an arbitrary choice of representatives from each class in $G_1 *_{\simeq} G_2$. Given a gluing $\mu \colon G_1 \to H \leftarrow G_2$, denote by $\hat{\mu}$ its "tip", i.e. the common codomain $\hat{\mu} = H$ of μ.

It is easy to see the following (see App. B for an axiomatic proof):

Lemma 4. *Let G_1, G_2 be graphs, then $G_1 * G_2$ is finite, and for every gluing $f_1 \colon G_1 \to H$, $f_2 \colon G_2 \to H$, there is a unique MG $i_1 \colon G_1 \to U$, $i_2 \colon G_2 \to U$ in $G_1 * G_2$ and match $u \colon U \to H$ such that $f_1 = u \circ i_1$ and $f_2 = u \circ i_2$.*

See Fig. 2 in §1 for an example of a set of MGs.

3 Graph-based GREs

To derive the GRE for graphs (GREG) we follow the development in our previous work [15,16] with the important difference that we do not assume derivations to be reversible. The key technical innovation that allows us to avoid the assumption of reversibility is the backward modularity lemma (Lemma 2).

As sketched in §1.2, our GRE for graphs is defined in terms of graph observables, which we now define formally. Fix S to be the countable (up to iso) set of finite graphs, and let $F \in S$ be a graph. The *graph observable* $[F] : S \to \mathbb{N}$ is the integer-valued function $[F](G) := |[F, G]|$ counting the number of occurrences (i.e. matches) of F in a given graph G. Graph observables are elements of the vector space $\mathbb{R}^S$ of real-valued functions on S.

The stochastic rate matrix Q_α for a rule $\alpha\colon L \rightharpoonup R$ defined in (3) is a linear map on $\mathbb{R}^S$. Its action on an observable $[F]$ is given by

$$(Q_\alpha\,[F])(G) := \textstyle\sum_H q^\alpha_{GH}([F]\,(G) - [F]\,(H)) \qquad \text{for } G, H \in S. \tag{11}$$

Since the sum above is finite, $Q_\alpha\,[F]$ is indeed a well-defined element of $\mathbb{R}^S$. We call $Q_\alpha\,[F]$ the *jump* of $[F]$ relative to Q_α. Intuitively, $(Q_\alpha\,[F])(G)$ is the expected rate of change in $[F]$ given that the CTMC sits at G.

To obtain the GREG as sketched in §1, we want to express the jump as a finite linear combination of graph observables. We start by substituting the definition of Q_α in (11).

$$\begin{aligned}(Q_\alpha\,[F])(G) &= \textstyle\sum_H q^\alpha_{GH}([F]\,(H) - [F]\,(G))\\ &= \textstyle\sum_H \sum_{f\in[L,G] \text{ s.t. } \alpha(f)\in[R,H]} (|[F,H]| - |[F,G]|)\\ &= \textstyle\sum_{f\in[L,G]} (|[F,\mathrm{cod}(\alpha(f))]| - |[F,G]|)\,.\end{aligned}$$

where the simplification in the last step is justified by the fact that f and α uniquely determine $\alpha(f)$. The last line suggests a decomposition of $Q_\alpha\,[F]$ as $Q_\alpha\,[F] = Q^+_\alpha\,[F] - Q^-_\alpha\,[F]$, where Q^+_α produces new instances of F while Q^-_α consumes existing ones.

By Lemma 4, we can factor the action of the consumption term Q^-_α through the MGs $L * F$ of L and F to obtain

$$(Q^-_\alpha\,[F])(G) \;=\; \textstyle\sum_{f\in[L,G]} |[F,G]| \;=\; |[L,G]|\cdot|[F,G]| \;=\; \sum_{\mu\in L*F} |[\hat{\mu},G]|\,.$$

The resulting sum is a linear combination of a finite number of graph observables, which is exactly what we are looking for.

Simplifying the production term requires a bit more work. Applying the same factorisation Lemma 4, we arrive at

$$\begin{aligned}(Q^+_\alpha\,[F])(G) &= \textstyle\sum_{f\in[L,G]} |[F,\hat{\alpha}(f)]|\\ &= \textstyle\sum_{f\in[L,G]} \sum_{(\mu_1,\mu_2)\in R*F} |\{g\in[\hat{\mu},\hat{\alpha}(f)] \mid g\circ\mu_1 = \alpha(f)\}|\,.\end{aligned}$$

where $\hat{\alpha}(f) = \mathrm{cod}(\alpha(f))$ denotes the codomain of the comatch of f. To simplify this expression further, we use the properties of derivations introduced in §2.3. First, we observe that μ_1 must be derivable by α for the set of g's in the above expression to be nonempty.

Lemma 5. *Let $\alpha\colon L \rightharpoonup R$ be a rule and $f\colon L \to G$, $g\colon R \to H$, $g_1\colon R \to T$ matches such that $f \Rightarrow_\alpha g$, but g_1 is not derivable by α. Then there is no match $g_2\colon T \to H$ such that $g_2 \circ g_1 = g$.*

Proof. By the contrapositive of backward modularity. Any such g_2 would induce, by Lemma 2, a match $f_1\colon L \to S$ and a derivation $f_1 \Rightarrow_\alpha g_1$. □

We may therefore restrict the set $R * F$ of right-hand MGs under consideration to the subset $\alpha *_R F := \{(\mu_1, \mu_2) \in R * F \mid \exists h.\, h \Rightarrow_\alpha \mu_1\}$ of MGs with a first projection derivable by α. Next, we observe that the modularity Lemmas 1 and 2 establish a *one-to-one correspondence* between the set of factorisations of the comatches $\alpha(f)$ (through the MGs in $\alpha *_R F$) and a set of factorisations of the corresponding matches f.

Lemma 6 (correspondence of matches). *Let α, β, γ, f, f_1, g, g_1 such that diagrams (12) and (13) are derivations and g_1 is derivable by α. Then the set $M_L = \{f_2 \in [S, G] \mid f_2 \circ f_1 = f\}$ is in one-to-one correspondence with the set $M_R = \{g_2 \in [T, H] \mid g_2 \circ g_1 = g\}$.*

$$\begin{tikzcd} L \arrow[r, harpoon, "\alpha"] \arrow[d, "f"'] & R \arrow[d, "g"] \\ G \arrow[r, harpoon, "\beta"] & H \end{tikzcd} \quad (12) \qquad\qquad \begin{tikzcd} L \arrow[d, "f_1"'] & R \arrow[l, harpoon', "\alpha^\dagger"'] \arrow[d, "g_1"] \\ S & T \arrow[l, harpoon', "\gamma^\dagger"'] \end{tikzcd} \quad (13)$$

Proof. Since g_1 is α-derivable, the diagram (13) is reversible, that is, $f_1 \Rightarrow_\alpha g_1$, with corule γ (by Lemma 3). Hence, if we are given a match f_2 in M_L, we can forward-decompose (12) vertically along the factorisation $f_2 \circ f_1 = f$, resulting in the diagram below (by forward modularity, Lemma 1).
Furthermore, the comatch g_2 is unique with respect to this decomposition, thus defining a function $\phi\colon M_L \to M_R$ that maps any f_2 in M_L to the corresponding comatch $\phi(f_2) = g_2$ in M_R. We want to show that ϕ is a bijection. By backward modularity (Lemma 2), there is a match $f_2 \in M_L$ for any match $g_2 \in M_R$ such that $\phi(f_2) = g_2$ (surjectivity), and furthermore, f_2 is the unique match for which $\phi(f_2) = g_2$ (injectivity). $\square$

$$\begin{tikzcd} L \arrow[r, harpoon, "\alpha"] \arrow[d, "f_1"'] & R \arrow[d, "g_1"'] \arrow[dd, bend left, "g"] \\ S \arrow[r, harpoon, "\gamma"] \arrow[d, "f_2"'] & T \arrow[d, dashed, "g_2"'] \\ G \arrow[r, harpoon, "\beta"] & H \end{tikzcd}$$

Using Lemmas 5 and 6, we can simplify Q^+_α as follows:

$$\begin{aligned} (Q^+_\alpha[F])(G) &= \sum_{f \in [L,G]} \sum_{\mu \in \alpha *_R F} \left|\{g_2 \in [\hat{\mu}, \hat{\alpha}(f)] \mid g_2 \circ \mu_1 = \alpha(f)\}\right| \\ &= \sum_{\mu \in \alpha *_R F} \sum_{f \in [L,G]} \left|\left\{f_2 \in \left[\hat{\alpha}^\dagger(\mu_1), G\right] \mid f_2 \circ \alpha^\dagger(\mu_1) = f\right\}\right| \\ &= \sum_{\mu \in \alpha *_R F} \left|\left[\hat{\alpha}^\dagger(\mu_1), G\right]\right| \end{aligned}$$

If we set $\alpha *_L F := L * F$ to symmetrise notation, we obtain

$$Q_\alpha([F]) = \textstyle\sum_{\mu \in \alpha *_R F} \left[\hat{\alpha}^\dagger(\mu_1)\right] - \sum_{\mu \in \alpha *_L F} [\hat{\mu}] \tag{14}$$

Now, in general for a CTMC on a countable state space S, the Markov-generated and time-dependent probability p on S follows the master equation [38,1]: $\frac{d}{dt} p^T = p^T Q$. Given an abstract observable f in $\mathbb{R}^S$, and writing $\mathbb{E}_p(f) := p^T f$ for the expected value of f according to p, we can then derive the formal[5] Kolmogorov equation for f:

$$\tfrac{d}{dt}\,\mathbb{E}_p(f) \;=\; \tfrac{d}{dt}p^T f \;=\; p^T Q f \;=\; \mathbb{E}_p(Qf),$$

giving us an equation for the rate of change of the mean of $f(X(t))$. Following this general recipe gives us the GRE for graphs immediately from (14).

$$\frac{d}{dt}\,\mathbb{E}_p([F]) = -\sum_{\alpha\in\mathcal{R}} k(\alpha) \sum_{\mu\in\alpha *_L F} \mathbb{E}_p\left[\hat{\mu}\right] + \sum_{\alpha\in\mathcal{R}} k(\alpha) \sum_{\mu\in\alpha *_R F} \mathbb{E}_p\left[\hat{\alpha}^\dagger(\mu_1)\right]. \tag{15}$$

Remember that μ_1 denotes the left injection of the MG $\mu = (\mu_1, \mu_2)$ while $\hat{\mu}$ denotes its codomain, and that $\hat{\alpha}^\dagger(f) = \mathrm{cod}(\alpha^\dagger(f))$.

Unsurprisingly, the derivation of (15) was more technically challenging than that of the GRE for reversible graph rewrite systems (cf. [16, Theorem 2]). Yet the resulting GREs look almost identical (cf. [16, Eq. (7)]). The crucial difference is in the production term Q^+_α, where we no longer sum over the full set of right-hand MGs $R * F$ but only over the subset $\alpha *_R F$ of MGs that are α-derivable. This extra condition is the price we pay for dealing with irreversibility: irreversible rules can consume all MGs, but only produce some.

Note that the number of terms in (15) depends on the size of the relevant sets of left and right-hand MGs, which is worst-case exponential in the size of the graphs involved, due to the combinatorial nature of MGs. (See Fig. 2 in §1 for an example.) In practice, one often finds many pairs of *irrelevant* MGs, the terms of which cancel out exactly. This reduces the effective size of the equations but not the overall complexity of generating the GREG.

Finally, as said in §1.2, the repeated application of (15) will lead to an infinite expansion in general. In practice, the system of ODEs needs to be truncated. For certain models, one can identify invariants in the underlying rewrite system via static analysis, which result in a finite closure even though the set of reachable components is demonstrably infinite [17]. We have seen an example in §1.

4 Conclusion

We have developed a computer-supported method for mean field approximations (MFA) for stochastic systems with graph-like states that are described by rules of SPO rewriting. The underlying theory unifies a large and seemingly unstructured collection of MFA approaches which share a graphical "air de famille". Based on the categorical frameworks of graph transformation systems (GTS), we have developed MFA-specific techniques, in particular concerning the combinatorics of minimal gluings. The main technical hurdle consisted in showing that the set of subgraph observables is closed under the action of the rate matrix (a.k.a. the

[5] In the present paper, we elide the subtle issues of ensuring that the system of interest actually satisfies this equation. See the work of Spieksma [41] for the underlying mathematics or our previous work [14], which additionally considers computability of the solutions to arbitrary precision.

infinitesimal generator) of the continuous-time Markov chain generated by an irreversible GTS. The proof is constructive and gives us an explicit term for the derivative of the mean of any observable of interest.

Mean field approximation and moment-closure methods are of wide use in applications, as typical probabilistic systems tend to have state spaces which defy more direct approaches. To reach their full potential, MFAs need to be combined with reachability and invariant analysis (as illustrated in §1).

We have worked the construction at the general axiomatic level of SPO-rewriting with matches and rules restricted to monomorphisms. One interesting extension is to include nested application conditions (NACs) [27,39] where the application of a rule can be modulated locally by the context of the match. NACs are useful in practice, and bring aboard the expressive power of first order logic in the description of transformation rules. We plan to investigate the extension of our approach to NACs, and, in particular, whether it is possible to incorporate them axiomatically, and what additional complexity cost they might incur.

Another direction of future work is to improve on the method of truncation. In the literature, one often finds graphical MFAs used in combination with conditional independence assumptions to control the size of connected observables, as e.g. the so-called pair approximation [18,25]. As these methods are known to improve the accuracy of naive truncation, we wish to understand if and how they can be brought inside our formal approach.

References

1. Anderson, W.J.: Continuous-time Markov chains: An applications-oriented approach. Springer Science & Business Media (2012)
2. Baldan, P., Corradini, A., Heindel, T., König, B., Sobocinski, P.: Processes and unfoldings: concurrent computations in adhesive categories. Mathematical Structures in Computer Science **24** (2014). https://doi.org/10.1017/S096012951200031X
3. Bapodra, M., Heckel, R.: From graph transformations to differential equations. ECEASST **30** (2010). https://doi.org/10.14279/tuj.eceasst.30.431
4. Barr, M., Wells, C.: Category theory for computing science (2. ed.). Prentice Hall international series in computer science, Prentice Hall (1995)
5. Behr, N.: Sesqui-pushout rewriting: Concurrency, associativity and rule algebra framework. Electronic Proceedings in Theoretical Computer Science **309**, 23–52 (2019). https://doi.org/10.4204/eptcs.309.2
6. Behr, N., Danos, V., Garnier, I.: Stochastic mechanics of graph rewriting. In: Proc. 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2016), New York, NY, USA. p. 46–55. ACM (2016). https://doi.org/10.1145/2933575.2934537
7. Behr, N., Danos, V., Garnier, I.: Combinatorial conversion and moment bisimulation for stochastic rewriting systems. Logical Methods in Computer Science **16** (2020), https://lmcs.episciences.org/6628
8. Behr, N., Danos, V., Garnier, I., Heindel, T.: The algebras of graph rewriting. CoRR arXiv:1612.06240 (2016), http://arxiv.org/abs/1612.06240
9. Behr, N., Krivine, J.: Rewriting theory for the life sciences: A unifying framework for CTMC semantics. In: Gadducci, F., Kehrer, T. (eds.) Proc. Graph Transformation,

13th International Conference, (ICGT 2020), Bergen, Norway. LNCS, vol. 12150. Springer (2020). https://doi.org/10.1007/978-3-030-51372-6_11

10. Behr, N., Saadat, M.G., Heckel, R.: Commutators for stochastic rewriting systems: Theory and implementation in Z3. CoRR arXiv:2003.11010 (2020), http://arxiv.org/abs/2003.11010
11. Bortolussi, L., Hillston, J., Latella, D., Massink, M.: Continuous approximation of collective system behaviour: A tutorial. Performance Evaluation **70**(5), 317–349 (2013). https://doi.org/10.1016/j.peva.2013.01.001
12. Corradini, A., Heindel, T., Hermann, F., König, B.: Sesqui-pushout rewriting. In: Corradini, A., Ehrig, H., Montanari, U., Ribeiro, L., Rozenberg, G. (eds.) Proc. Graph Transformations, Third International Conference (ICGT 2006), Natal, Rio Grande do Norte, Brazil. LNCS, vol. 4178, pp. 30–45. Springer (2006). https://doi.org/10.1007/11841883_4
13. Danos, V., Harmer, R., Honorato-Zimmer, R., Stucki, S.: Deriving rate equations for site graph rewriting systems. In: Proc. 4th International Workshop on Static Analysis and Systems Biology (SASB 2013), Seattle, WA, USA (to appear)
14. Danos, V., Heindel, T., Garnier, I., Simonsen, J.G.: Computing continuous-time markov chains as transformers of unbounded observables. In: Esparza, J., Murawski, A.S. (eds.) Proc. Foundations of Software Science and Computation Structures, 20th International Conference (FOSSACS 2017), Uppsala, Sweden. LNCS, vol. 10203, pp. 338–354 (2017). https://doi.org/10.1007/978-3-662-54458-7_20
15. Danos, V., Heindel, T., Honorato-Zimmer, R., Stucki, S.: Approximations for stochastic graph rewriting. In: Merz, S., Pang, J. (eds.) Proc. Formal Methods and Software Engineering – 16th International Conference on Formal Engineering Methods (ICFEM 2014), Luxembourg, Luxembourg. LNCS, vol. 8829, pp. 1–10. Springer (2014). https://doi.org/10.1007/978-3-319-11737-9_1
16. Danos, V., Heindel, T., Honorato-Zimmer, R., Stucki, S.: Moment semantics for reversible rule-based systems. In: Krivine, J., Stefani, J. (eds.) Proc. Reversible Computation, 7th International Conference (RC 2015), Grenoble, France. LNCS, vol. 9138, pp. 3–26. Springer (2015). https://doi.org/10.1007/978-3-319-20860-2_1
17. Danos, V., Honorato-Zimmer, R., Jaramillo-Riveri, S., Stucki, S.: Coarse-graining the dynamics of ideal branched polymers. In: Proc. 3rd International Workshop on Static Analysis and Systems Biology (SASB 2012), Deauville, France. ENTCS, vol. 313, pp. 47–64 (2015). https://doi.org/10.1016/j.entcs.2015.04.018
18. Durrett, R., Gleeson, J.P., Lloyd, A.L., Mucha, P.J., Shi, F., Sivakoff, D., Socolar, J.E.S., Varghese, C.: Graph fission in an evolving voter model. Proceedings of the National Academy of Sciences **109**(10), 3682–3687 (2012). https://doi.org/10.1073/pnas.1200709109
19. Dyckhoff, R., Tholen, W.: Exponentiable morphisms, partial products and pullback complements. Journal of Pure and Applied Algebra **49**(1–2), 103–116 (1987). https://doi.org/10.1016/0022-4049(87)90124-1
20. 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: Rozenberg, G. (ed.) Handbook of Graph Grammars and Computing by Graph Transformation, pp. 247–312. World Scientific, River Edge, NJ, USA (1997)
21. Ethier, S.N., Kurtz, T.G.: Markov Processes: Characterization and Convergence. Wiley (1986)
22. Evans, M.R., Ferrari, P.A., Mallick, K.: Matrix representation of the stationary measure for the multispecies TASEP. Journal of Statistical Physics **135**(2), 217–239 (2009). https://doi.org/10.1007/s10955-009-9696-2

23. Fages, F., Soliman, S.: Formal cell biology in Biocham. In: Bernardo, M., Degano, P., Zavattaro, G. (eds.) Formal Methods for Computational Systems Biology, 8th International School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFM 2008), Bertinoro, Italy, Advanced Lectures. LNCS, vol. 5016, pp. 54–80. Springer (2008). https://doi.org/10.1007/978-3-540-68894-5_3
24. Feret, J., Danos, V., Krivine, J., Harmer, R., Fontana, W.: Internal coarse-graining of molecular systems. Proceedings of the National Academy of Sciences **106**(16), 6453–6458 (2009). https://doi.org/10.1073/pnas.0809908106
25. Gleeson, J.P.: High-accuracy approximation of binary-state dynamics on networks. Physical Review Letters **107**(6), 068701 (2011). https://doi.org/10.1103/PhysRevLett.107.068701
26. Grima, R., Thomas, P., Straube, A.V.: How accurate are the nonlinear chemical Fokker-Planck and chemical Langevin equations? The Journal of Chemical Physics **135**(8), 084103 (2011). https://doi.org/10.1063/1.3625958
27. Habel, A., Pennemann, K.H.: Correctness of high-level transformation systems relative to nested conditions. Mathematical Structures in Computer Science **19**(2), 245–296 (2009). https://doi.org/10.1017/S0960129508007202
28. Harmer, R., Danos, V., Feret, J., Krivine, J., Fontana, W.: Intrinsic information carriers in combinatorial dynamical systems. Chaos **20**(3) (2010). https://doi.org/10.1063/1.3491100
29. Hayman, J., Heindel, T.: Pattern graphs and rule-based models: The semantics of Kappa. In: Pfenning, F. (ed.) Proc. Foundations of Software Science and Computation Structures, 16th International Conference (FOSSACS 2013), Rome, Italy. LNCS, vol. 7794, pp. 1–16. Springer (2013). https://doi.org/10.1007/978-3-642-37075-5_1
30. Heckel, R.: DPO transformation with open maps. In: Ehrig, H., Engels, G., Kreowski, H.J., Rozenberg, G. (eds.) Proc. Graph Transformations, 6th International Conference (ICGT 2012), Bremen, Germany. LNCS, vol. 7562, pp. 203–217. Springer (2012). https://doi.org/10.1007/978-3-642-33654-6_14
31. Heckel, R., Lajios, G., Menge, S.: Stochastic graph transformation systems. Fundamenta Informaticae **74**(1), 63–84 (2006)
32. van Kampen, N.: Stochastic processes in physics and chemistry. North-Holland (2007), 3rd edition
33. Lack, S., Sobociński, P.: Adhesive and quasiadhesive categories. ITA **39**(3), 511–545 (2005). https://doi.org/10.1051/ita:2005028
34. Lopez, C.F., Muhlich, J.L., Bachman, J.A., Sorger, P.K.: Programming biological models in Python using PySB. Molecular Systems Biology **9**(1) (2013). https://doi.org/10.1038/msb.2013.1
35. Löwe, M.: Algebraic approach to single-pushout graph transformation. Theoretical Computer Science **109**(1&2), 181–224 (1993). https://doi.org/10.1016/0304-3975(93)90068-5
36. Löwe, M.: Graph rewriting in span-categories. In: Ehrig, H., Rensink, A., Rozenberg, G., Schürr, A. (eds.) Proc. Graph Transformations, 5th International Conference (ICGT 2010), Enschede, The Netherlands. LNCS, vol. 6372, pp. 218–233. Springer (2010). https://doi.org/10.1007/978-3-642-15928-2_15
37. Lynch, J.F.: A logical characterization of individual-based models. In: Proc. Twenty-Third Annual IEEE Symposium on Logic in Computer Science (LICS 2008), Pittsburgh, PA, USA. pp. 379–390. IEEE Computer Society (2008). https://doi.org/10.1109/LICS.2008.27
38. Norris, J.R.: Markov chains. Cambridge series in statistical and probabilistic mathematics, Cambridge University Press (1998)

39. Rensink, A.: Representing first-order logic using graphs. In: Ehrig, H., Engels, G., Parisi-Presicce, F., Rozenberg, G. (eds.) Proc. Graph Transformations, Second International Conference (ICGT 2004), Rome, Italy. LNCS, vol. 3256, pp. 319–335. Springer (2004). https://doi.org/10.1007/978-3-540-30203-2_23
40. Robinson, E., Rosolini, G.: Categories of partial maps. Inf. Comput. **79**(2), 95–130 (1988). https://doi.org/10.1016/0890-5401(88)90034-X
41. Spieksma, F.M.: Kolmogorov forward equation and explosiveness in countable state Markov processes. Annals of Operations Research **241**(1), 3–22 (2016). https://doi.org/10.1007/s10479-012-1262-7
42. Stukalin, E.B., Phillips III, H., Kolomeisky, A.B.: Coupling of two motor proteins: a new motor can move faster. Physical Review Letters **94**(23), 238101 (2005). https://doi.org/10.1103/PhysRevLett.94.238101
43. Thomas, P., Matuschek, H., Grima, R.: Intrinsic noise analyzer: a software package for the exploration of stochastic biochemical kinetics using the system size expansion. PloS ONE **7**(6), e38518 (2012). https://doi.org/10.1371/journal.pone.0038518

A Pushout and pull-back complements

Algebraic graph rewriting relies on certain category-theoretical *limits* and *colimits* [4]. We give definitions of the relevant (co-)limits here along with some of their basic properties. Among these, pullback complements are the least known. We refer the interested reader to Ref. [19,12] for a thorough treatment.

$$\begin{tikzcd}
Q \arrow[drr, bend left, "g_1"] \arrow[dr, dashed, "u"] \arrow[ddr, bend right, "g_2"'] & & \\
& P \arrow[r, "p_1"] \arrow[d, "p_2"'] & Y \arrow[d, "f_2"] \\
& X \arrow[r, "f_1"'] & Z
\end{tikzcd} \quad (16)
\qquad
\begin{tikzcd}
Z \arrow[r, "f_1"] \arrow[d, "f_2"'] & X \arrow[d, "i_2"] \arrow[ddr, bend left, "g_2"] & \\
Y \arrow[r, "i_1"'] \arrow[drr, bend right, "g_1"'] & P \arrow[dr, dashed, "u"] & \\
& & Q
\end{tikzcd} \quad (17)$$

Let $\mathcal{C}$ be a category.

Definition 3 (Pullback). *A* pullback *of a cospan of morphisms $X \xrightarrow{f_1} Z \xleftarrow{f_2} X$ in $\mathcal{C}$ is a span $X \xleftarrow{p_1} P \xrightarrow{p_2} Y$ making the bottom-right square in (16) commute, and such that for any other span $X \xleftarrow{g_1} Q \xrightarrow{g_2} Y$ for which the outer square commutes, there is a unique morphism $u\colon Q \to P$ making the diagram commute.*

Definition 4 (Pushout). *A* pushout *of a span of morphisms $X \xrightarrow{f_1} Z \xleftarrow{f_2} Y$ in $\mathcal{C}$ is a cospan $X \xrightarrow{i_1} P \xleftarrow{i_2} Y$ making the top-left square in (17) commute, and such that for any other cospan $X \xrightarrow{g_1} Q \xleftarrow{g_2} Y$ for which the outer square commutes, there is a unique morphism $u\colon P \to Q$ making the diagram commute.*

$$\begin{tikzcd}
P \arrow[dr, "p"'] \arrow[drr, bend left, "f_1'"] \arrow[ddd, "g_1'"'] & & \\
& X \arrow[r, "f_1"'] \arrow[d, "g_1"'] & Y \arrow[d, "f_2"] \\
& W \arrow[r, "g_2"] & Z \\
Q \arrow[ur, dashed, "u"] \arrow[urr, bend right, "g_2'"'] & &
\end{tikzcd} \quad (18)$$

Definition 5 (Final pullback complement). *A* final pullback complement (FPBC) *(or simply* pullback complement*) of a pair of composable morphisms $X \xrightarrow{f_1} Y \xrightarrow{f_2} Z$ in some category $\mathcal{C}$ is a pair of composable morphisms $X \xrightarrow{g_1} W \xrightarrow{g_2} Z$ making the right inner square in* (18) *a pullback, such that for any other pullback $P \xrightarrow{f'_1} Y \xrightarrow{f_2} Z \xleftarrow{g'_2} Q \xleftarrow{g'_1} P$ and morphism $p\colon P \to X$ for which the diagram commutes, there is a unique morphism $u\colon Q \to W$ that makes the diagram commute.*

The following lemmas, pertaining to the composition of pullbacks, pushouts and FPBCs, respectively, are used throughout the proofs in App. B. The first two are dual versions of the well-known "pasting" lemma for pullbacks and pushouts, and we leave their proofs as an exercise to the reader. A proof of the third lemma can be found in [36, Proposition 5].

$$\begin{tikzcd}
A \arrow[r, "f_1"] \arrow[d, "g_1"'] & B \arrow[r, "f_2"] \arrow[d, "g_2"] & C \arrow[d, "g_3"] \\
D \arrow[r, "h_1"'] & E \arrow[r, "h_2"'] & F
\end{tikzcd} \tag{19}$$

Lemma 7 (Pasting of pullbacks). *Suppose the right inner square in* (19) *is a pullback in some category $\mathcal{C}$. Then the left inner square is a pullback if and only if the outer square is.*

Lemma 8 (Pasting of pushouts). *Suppose the left inner square in* (19) *is a pushout in some category $\mathcal{C}$. Then the right inner square is a pushout if and only if the outer square is.*

Lemma 9 (Composition of FPBCs). *Consider again diagram* (19) *in some category $\mathcal{C}$,*

- *(horizontal composition) if $A \xrightarrow{g_1} D \xrightarrow{h_1} E$ and $B \xrightarrow{g_2} E \xrightarrow{h_2} F$ are the FPBCs of $A \xrightarrow{f_1} B \xrightarrow{g_2} E$ and $B \xrightarrow{f_2} C \xrightarrow{g_3} F$, respectively, then $A \xrightarrow{g_1} D \xrightarrow{h_2 \circ h_1} F$ is the FPBC of $A \xrightarrow{f_2 \circ f_1} C \xrightarrow{g_3} F$;*
- *(vertical composition) if $A \xrightarrow{f_1} B \xrightarrow{g_2} E$ and $B \xrightarrow{f_2} C \xrightarrow{g_3} F$ are the FPBCs of $A \xrightarrow{g_1} D \xrightarrow{h_1} E$ and $B \xrightarrow{g_2} E \xrightarrow{h_2} F$, respectively, then $A \xrightarrow{f_2 \circ f_1} C \xrightarrow{g_3} F$ is the FPBC of $A \xrightarrow{g_1} D \xrightarrow{h_2 \circ h_1} F$.*

B Generalised proofs of lemmas

This section contains detailed proofs of the various lemmas introduced in previous sections. We will present the proofs in a slightly more general setting, namely that of *sesqui-pushout (SqPO) rewriting* [12] in arbitrary *adhesive categories* [33]. To be precise, we assume an ambient category $\mathcal{G}$, such that

– $\mathcal{G}$ is adhesive (among other things, this implies that $\mathcal{G}$ has all pullbacks as well as all pushouts along monomorphisms, that monomorphism are stable under pushout, and that all such pushouts are also pullbacks, cf. [33]),
– $\mathcal{G}$ has all final pullback complements (FPBCs) above monomorphisms.

Both these assumptions hold in **Grph**. Within $\mathcal{G}$, we define derivations as in Def. 1, taking matches and rules to be monomorphisms and spans thereof, respectively.

Alternatively, rules can be seen as partial maps [40] in the category $\mathcal{G}_*$, generalising the interpretation of rules as partial graph morphisms in $\mathbf{Grph}_*$. Derivations can then be shown to correspond exactly to pushouts of rules along monomorphisms in $\mathcal{G}_*$ [2, Proposition 2.10], and composition of derivations corresponds to pushout composition in $\mathcal{G}_*$.

B.1 Proof of Lemma 4 (minimal gluings)

Let G_1 and G_2 be graphs, then

1. the set $G_1 * G_2$ of MGs of G_1 and G_1 is finite, and
2. for every cospan $G_1 \xrightarrow{f_1} H \xleftarrow{f_2} G_2$ of matches, there is a unique MG $(G_1 \xrightarrow{i_1} U \xleftarrow{i_2} G_2) \in G_1 * G_2$ and match $u\colon U \to H$ such that $f_1 = u \circ i_1$ and $f_2 = u \circ i_2$.

Proof. For this proof we will make two additional assumptions on $\mathcal{G}$, namely that $\mathcal{G}$ has all binary products, and that the objects of $\mathcal{G}$ are finitely powered, that is, any object A in $\mathcal{G}$ has a finite number of subobjects. Both these assumptions hold in **Grph**.

Recall that the subobjects of any object A in $\mathcal{G}$ form a poset category $\mathbf{Sub}(A)$ with *subobject intersections* as products and *subobject unions* as coproducts. By stability of monomorphisms under pullback, products (intersections) in $\mathbf{Sub}(A)$ are given by pullbacks in $\mathcal{G}$, and since $\mathcal{G}$ is adhesive, coproducts (unions) in $\mathbf{Sub}(A)$ are given by pushouts of pullbacks in $\mathcal{G}$. See [33, Theorem 5.1] for more details.

$$\begin{tikzcd}
 & G_1 \cap G_2 \arrow[dl, "p_1"'] \arrow[dr, "p_2"] & \\
G_1 \arrow[dr, "i_1"'] \arrow[ddr, bend right, "f_1"'] & & G_2 \arrow[dl, "i_2"] \arrow[ddl, bend left, "f_2"] \\
 & G_1 \cup G_2 \arrow[d, dashed, "u"] & \\
 & H &
\end{tikzcd} \tag{20}$$

We will start by showing that any cospan $G_1 \xrightarrow{f_1} H \xleftarrow{f_2} G_2$ of matches in $\mathcal{G}$ factorises uniquely through an element of $G_1 * G_2$. Given such a cospan, let $u\colon G_1 \cup G_2 \to H$ be a representative in $\mathcal{G}$ of the subject union of f_1 and f_2 in $\mathbf{Sub}(H)$, with coproduct injections $i_1\colon G_1 \to G_1 \cup G_2$ and $i_2\colon G_2 \to G_1 \cup G_2$ as in (20). Since u is the mediating morphism of a pullback, it is unique up to isomorphism of $G_1 \cup G_2$. It remains to show that $G_1 \xrightarrow{i_1} G_1 \cup G_2 \xleftarrow{i_2} G_2$ is a MG. By adhesiveness of $\mathcal{G}$, the pushout square at the top of (20) is also a pullback, and hence an intersection of i_1 and i_2 in $\mathbf{Sub}(G_1 \cup G_2)$. It follows that $\mathrm{id}_{G_1 \cup G_2}$ represents the subobject union of i_2 and i_2 in $\mathbf{Sub}(G_1 \cup G_2)$ and hence $G_1 \xrightarrow{i_1} G_1 \cup G_2 \xleftarrow{i_2} G_2$ is indeed a MG.

The finiteness of $G_1 * G_2$ follows from a similar argument. First, note that $|G_1 * G_2| = |G_1 *_{\simeq} G_2|$, so it is sufficient to show that $G_1 *_{\simeq} G_2$ is finite. Being a subobject union, every MG is the pushout of a span $G_1 \xleftarrow{p_1} G_1 \cap G_2 \xrightarrow{p_2} G_2$ of matches as in (20). Since isomorphic spans have isomorphic pushouts, there can be at most as many isomorphism classes of MGs of G_1 and G_2 as there are isomorphism classes of spans over G_1 and G_2. Furthermore, the spans $G_1 \xleftarrow{p_1} X \xrightarrow{p_2} G_2$ are in one-to-one correspondence with the pairings $\langle p_1, p_2 \rangle \colon X \to G_1 \times G_2$ in $\mathcal{G}$, which represent subobjects in $\mathbf{Sub}(G_1 \times G_2)$ (with isomorphic spans corresponding to identical subobjects). Since $G_1 \times G_2$ is finitely powered, there are only a finite number of such subobjects, and hence there can only be a finite number of isomorphism classes of spans over G_1 and G_2, which concludes the proof. □

B.2 Proof of Lemma 1 (forward modularity)

Let f_1, f_2, g, g_1 be matches, and α, β, γ rules, such that the diagrams (5) and (6) are derivations. Then there is a unique match g_2, such that diagram (7) commutes and is a vertical composition of derivations.

$$\begin{tikzcd}
L \arrow[r, harpoon, "\alpha"] \arrow[d, "f_1"'] & R \arrow[dd, "g"] \\
S \arrow[d, "f_2"'] & \\
G \arrow[r, harpoon, "\beta"] & H
\end{tikzcd} \quad (5) \qquad
\begin{tikzcd}
L \arrow[r, harpoon, "\alpha"] \arrow[d, "f_1"'] & L \arrow[d, "g_1"] \\
S \arrow[r, harpoon, "\gamma"] & T
\end{tikzcd} \quad (6) \qquad
\begin{tikzcd}
L \arrow[r, harpoon, "\alpha"] \arrow[d, "f_1"'] & R \arrow[d, "g_1"'] \arrow[dd, bend left, "g"] \\
S \arrow[r, harpoon, "\gamma"] \arrow[d, "f_2"'] & T \arrow[d, dashed, "g_2"'] \\
G \arrow[r, harpoon, "\beta"] & H
\end{tikzcd} \quad (7)$$

Proof. Using the universal property of the pushout (6), we obtain the mediating morphism g_2 and apply the Pasting Lemma for pushouts to conclude that the lower square in (7) is a pushout. □

B.3 Proof of Lemma 2 (backward modularity)

Let f, f_1, g_1, g_2 be matches, and α, β, γ rules, such that the diagrams (8) and (9) are derivations. Then there is a unique match f_2, such that diagram (10) commutes and is a vertical composition of derivations.

$$\begin{tikzcd}
L \arrow[r, harpoon, "\alpha"] \arrow[dd, "f"'] & R \arrow[d, "g_1"] \\
& T \arrow[d, "g_2"] \\
G \arrow[r, harpoon, "\beta"] & H
\end{tikzcd} \quad (8) \qquad
\begin{tikzcd}
L \arrow[d, "f_1"'] & R \arrow[l, harpoon', "\alpha^\dagger"'] \arrow[d, "g_1"] \\
S & T \arrow[l, harpoon', "\gamma^\dagger"']
\end{tikzcd} \quad (9) \qquad
\begin{tikzcd}
L \arrow[r, harpoon, "\alpha"] \arrow[d, "f_1"] \arrow[dd, bend right, "f"'] & R \arrow[d, "g_1"] \\
S \arrow[r, harpoon, "\gamma"] \arrow[d, dashed, "f_2"] & T \arrow[d, "g_2"] \\
G \arrow[r, harpoon, "\beta"] & H
\end{tikzcd} \quad (10)$$

Proof. The proof is in three steps: we first construct f_1 and f_2 in $\mathcal{G}$, then we show that diagram (10) is indeed a composition of derivations, and finally we verify the uniqueness of f_2 for this property.

Consider diagram (21) below, which is the underlying diagram in $\mathcal{G}$ of derivation (8) from the lemma:

$$\begin{tikzcd} L \arrow[d, "f"'] & K \arrow[l, "\alpha_1"'] \arrow[r, "\alpha_2"] \arrow[d, "h"'] & R \arrow[d, "g_1"] \\ & & T \arrow[d, "g_2"] \\ G & D \arrow[l, "\beta_1"'] \arrow[r, "\beta_2"] & H \end{tikzcd} \quad (21) \qquad \begin{tikzcd} L \arrow[d, "f_1"] \arrow[dd, bend right, "f"'] & K \arrow[l, "\alpha_1"'] \arrow[r, "\alpha_2"] \arrow[d, "h_1"] & R \arrow[d, "g_1"] \\ S \arrow[d, "f_2"] & E \arrow[l, "\gamma_1"'] \arrow[r, "\gamma_2"] \arrow[d, "h_2"] & T \arrow[d, "g_2"] \\ G & D \arrow[l, "\beta_1"'] \arrow[r, "\beta_2"] & H \end{tikzcd} \quad (22)$$

The right-hand square is a pushout along monomorphisms, and hence it is also a pullback in $\mathcal{G}$, and we can decompose it along g_1 and g_2 to obtain the upper and lower right squares of diagram (22). By stability of pushouts in $\mathcal{G}$ (see [33, Lemma 4.7]), both these squares are also pushouts. To complete diagram (22), let its upper-left square be a pushout, and f_2 the unique mediating morphism such that the right-hand side of the diagram commutes.

Note that all morphisms in (22), except possibly f_2, are monic. The composites $\gamma_1 \circ h_1$ and $\gamma_2 \circ h_1$ are pushout complements of $f_1 \circ \alpha_1$ and $g_1 \circ \alpha_2$, respectively, and hence by [12, proposition 12], they are also FPBCs. It follow that the upper half of (22) is indeed the underlying diagram in $\mathcal{G}$ of both the derivations in (9) and the upper half of (10). For the lower half of (22) to also be a derivation, f_2 must be a match, so we need to show that it is monic. To show this, let $S \xrightarrow{i_1} G' \xleftarrow{i_2} D$ be a pushout of $S \xleftarrow{\gamma_1} E \xrightarrow{h_2} D$, and let $u \colon G' \to G$ be its mediating morphism with respect to the lower-left square of (22). Since h_2 is monic, so is i_1 (by adhesiveness of $\mathcal{G}$). By pasting of pushouts, u is also the mediating morphism of the pushout $L \xrightarrow{i_1 \circ f_1} G' \xleftarrow{i_2} D$ with respect to the left-hand square in (21), which in turn, is also a pullback square. In fact, the composite pushout is the union of the subobjects represented by f and β_1, and hence by [33, Theorem 5.1], u is a monomorphism. It then follows that the composite $f_2 = u \circ i_1$ is also a monomorphism.

$$\begin{tikzcd} L \arrow[d, "f_1"] \arrow[dd, bend right, "f"'] & K \arrow[l, "\alpha_1"'] \arrow[r, "\alpha_2"] \arrow[d, "h_1"] & R \arrow[d, "g_1"'] \arrow[dd, bend left, "g"] \\ S \arrow[d, "f_2"] & E \arrow[l, "\gamma_1"'] \arrow[r, "\gamma_2"] \arrow[d, "h_2'"] & T \arrow[d, "g_2'"'] \\ G & D \arrow[l, "\beta_1"'] \arrow[r, "\beta_2"] & H \end{tikzcd} \quad (23) \qquad \begin{tikzcd} L \arrow[d, "f_1"] \arrow[dd, bend right, "f"'] & K \arrow[l, "\alpha_1"'] \arrow[r, "\alpha_2"] \arrow[d, "h_1"] & R \arrow[d, "g_1"'] \arrow[dd, bend left, "g"] \\ S \arrow[d, "f_2'"] & E \arrow[l, "\gamma_1"'] \arrow[r, "\gamma_2"] \arrow[d, "h_2''"] & T \arrow[d, "g_2"'] \\ G & D \arrow[l, "\beta_1"'] \arrow[r, "\beta_2"] & H \end{tikzcd} \quad (24)$$

Now let h_2' and g_2' be matches such that diagram (23) commutes and is a composition of tiles as per Lemma 1. Then we have $\beta_1 \circ h_2 = f_2 \circ \gamma_1 = \beta_1 \circ h_2'$, and hence $h_2 = h_2'$ because β_1 is monic. Furthermore, the top-right square of (23) is a pushout, and hence g_2' is the unique mediating morphism such that $\beta_2 \circ h_2 = g_2' \circ \gamma_2$ and $g = g_2' \circ g_1$. But from diagram (22) we know that $\beta_2 \circ h_2 = g_2 \circ \gamma_2$ and $g = g_2 \circ g_1$, and hence $g_2' = g_2$. It follows that the bottom half of (22) is indeed a derivation.

Finally, let f'_2 and h''_2 be any matches such that diagram (24) commutes and is a composition of tiles. Then $h''_2 = h_2$ (because $\beta_2 \circ h''_2 = g_2 \circ \gamma_2 = \beta_2 \circ h_2$ and β_2 is monic) and $f'_2 = f_2$ (because it is the unique mediating morphism of the top-left pushout-square such that $\beta_1 \circ h_2 = f'_2 \circ \gamma_1$ and $f = f'_2 \circ f_1$), which concludes the proof. □

B.4 Proof of Lemma 3 (derivability)

A match $g\colon R \to H$ is derivable by a rule $\alpha\colon L \rightharpoonup R$ if and only if $g \Rightarrow_{\alpha \circ \alpha^\dagger} g$. Equivalently, g is derivable from f by α if and only if the derivation $g \Rightarrow_{\alpha^\dagger} f$ is reversible.

Proof. This is a direct consequence of Lemma 2. First, assume that $g\colon R \to H$ is derivable by $\alpha\colon L \rightharpoonup R$ from some match $f\colon L \to G$, and let $h\colon L \to E$ be the comatch of some derivation $g \Rightarrow_{\alpha^\dagger} h$. By Lemma 2 (setting $g_1 = g$ and $f_1 = h$), the derivation $h \Rightarrow_\alpha g$ exists, and so does $g \Rightarrow_{\alpha \circ \alpha^\dagger} g$ (by horizontal composition of derivations).

Now assume that we are given the derivation $g \Rightarrow_{\alpha \circ \alpha^\dagger} g$ instead, and let $f'\colon L \to G'$ and $h'\colon R \to E'$ be the comatches of some derivations $g \Rightarrow_{\alpha^\dagger} f'$ and $f' \Rightarrow_\alpha h'$. By horizontal composition and uniqueness of derivations up to isomorphism, we have $g \Rightarrow_{\alpha \circ \alpha^\dagger} h'$ and $g = u \circ h'$ for some (unique) isomorphism $u\colon E' \xrightarrow{\simeq} H$. Hence there is a derivation $f' \Rightarrow_\alpha g$. □